Data Book

32bit Micro controller
TLCS-900/H1 series

TMP92CZ26AXBG

TENTATIVE

It's first version technical data sheet.

Since this revision 0.2 is still under working, there may be some mistakes in it.

When you will start to design, please order the latest one.

Rev0.2 09/Dec./2005

Table of Contents

TLCS-900/H1 Devices
TMP92CZ26A

CMOS 32-Bit Micro controllers

TMP92CZ26AXBG

1. Outline and Features

TMP92CZ26A is high-speed advanced 32-bit micro-controller developed for controlling equipment which processes mass data.

TMP92CZ26AXBG is housed in a 228-pin BGA package.

(1) CPU : 32-bit CPU(High-speed 900/H1 CPU)

- Compatible with TLCS-900/L1 instruction code
- 16Mbytes of linear address space
- General-purpose register and register banks
- Micro DMA : 8channels (62.5ns/4 bytes at f_{SYS} = 80MHz, best case)

(2) Minimum instruction execution time : 12.5ns (at f_{SYS} = 80MHz)

(3) Internal RAM: 288K-byte (can be used for program, data and display memory)
Internal ROM: 8 K-byte(memory for Boot only)

It enables that load user program from USB, UART to Internal RAM.

(4) External memory expansion
- Expandable up to 3.1G bytes (shared program/data area)
- Can simultaneously support 8/16-bit width external data bus
  ····· Dynamic data bus sizing
- Separate bus system

(5) Memory controller
- Chip select output : 4 channel
- One channel in 4 channels is enabled detailed AC enable setting

(6) 8-bit timers : 8 channels

(7) 16-bit timer/event counter : 2 channel

(8) General-purpose serial interface : 1 channels
- UART/synchronous mode
- IrDA ver1.0 (115.2 kbps) selectable :

  (There is the restriction in the setting baud rate when use this function together other functions)

(9) Serial bus interface: 1 channel
- I^2C bus mode only

(10) USB (universal serial bus) controller: 1 channel
- Support to USB (REV1.1)
- Full-speed (12 Mbps) (Low-speed is not supported.)
- Endpoint 0: Control 64 bytes × 1-FIFO

  Endpoint 1: BULK (output) 64 bytes × 2-FIFO

  Endpoint 2: BULK (input) 64 bytes × 2-FIFO

  Endpoint 3: Interrupt (input) 8 bytes × 1-FIFO
- Descriptor RAM: 384 bytes

(11) I^2S (Inter-IC Sound)interface: 2 channel
- I^2S bus mode selectable (Master, transmission only)
- Data Format is supported Left/Right Justify
- Built in FIFO buffer of 128 bytes (64 bytes × 2) every each channels.

(12) LCD controller
- Supported up to monochrome, 4, 16 and 64 gray levels and 256/4096 color for STN
- Supported up to 4096/65536/262144/16777216 color for TFT
- Supported up to PIP (Picture In Picture Display)
- Supported up to H/W Rotation function for support to various LCDM

(13) SDRAM controller :1 channel
- Supported 16M, 64M, 128M, 256M and 512Mbit SDR (Single-data-rate) SDRAM
- Can use not only as Data RAM for LCD display but also operate program direct from SDRAM

(14) Timer for real-time clock (RTC)
- Based on TC8521A

(15) Key-on wakeup (Interrupt key input)

(16) 10-bit A/D converter (Built in Sample Hold circuit) : 6 channels

(17) Touch screen interface

- Built-in Switch of Low-resistor, and available to delete external components for shift change row/column

(18) Watch dog timer

(19) Melody/alarm generator

- Melody: Output of clock 4 to 5461Hz
- Alarm: Output of the 8 kinds of alarm pattern
- 5 kinds of interval interrupt

(20) MMU

- Expandable up to 3.1G bytes (3 local area/8 bank method)
- Independent bank for each Program, Read-data, Write-data, Source and Destination of DMAC (Odd channel/Even channel) and LCD-display-data

(21) Interrupts: 56 interrupts

• 9 CPU interrupts	·····	Software interrupt instruction and illegal instruction
• 38 internal interrupts	·····	Seven selectable priority levels
• 9 external interrupts	·····	Seven selectable priority levels (8 interrupt selectable negative/positive of edge)

(22) DMAC function: 6 channels

- High-speed data transfer enable by controlling which convert micro DMA function and this function

(23) Input/Output ports : 136 pins (Except Data bus (16bit), Address bus (24bit) and RD pin)

(24) Nand_Flash interface: 2 channel

- Available to connect directly with NAND flash
- Supported up to SLC type and MLC type
- Data Bus 8/16 Bit, Page Size 512/2048 Bytes
- Built-in Rees Solomon calculation circuits which enabled correct 4-address, and detect error more than 5-address

(25) SPI controller : 1 channel

- Supported up to SPI mode of SD card and MMC card
- Built-in FIFO buffer of 32 bytes to each Input/Output

(26) Product/Sum calculation: 1 channel

- calculation 32×32+64 =64Bit , 64-32×32 = 64Bit , 32×32-64 =64Bit
- I/O method

(27) Signed calculation is supported.

(28) Stand-by function

- Three Halt modes : IDLE2 (programmable), IDLE1, STOP
- Each pin status programmable for stand-by mode
- Built-in power supply management circuits (PMC) for leak current provision

(29) Clock controller

- Built-in two blocks of clock doubler (PLL). PLL supplies 48 MHz for USB and 80 MHz for CPU from 10MHz
- Clock gear function: Selectable high-frequency clock fc to fc/16
- Clock for Timer (fs = 32.768 kHz)

(30) Operating voltage:

- Internal V_{CC}= 1.5V, External I/O Vcc = 3.0 to 3.6 V
- 2 power supplies (Internal power supply (1.4 to 1.6), External power supply (3.0 to 3.6)

(31) Package

- 228 pin FBGA :P-FBGA228-1515-0.80A5

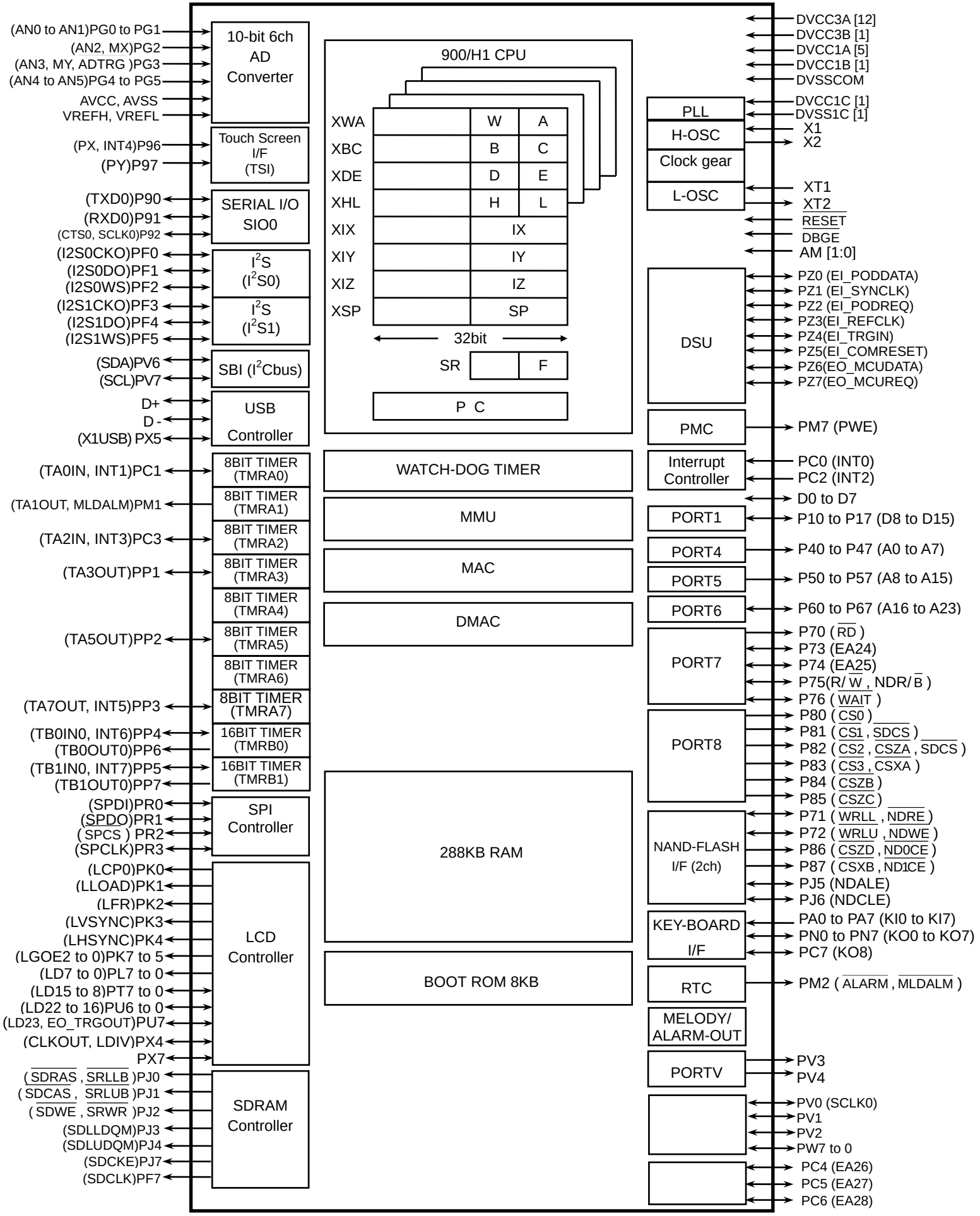


Figure 1.1 Block Diagram of TMP92CZ26A

2. Pin Assignment and Pin Functions

The assignment of input/output pins for TMP92CZ26A, their names and functions are as follows;

2.1 Pin Assignment Diagram (Top View)

Figure 2.1.1 shows the pin assignment of the TMP92CZ26A.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17
D1	D2	D3		D5	D6	D7	D8	D9	D10	D11	D12	D13		D15	D16	D17
E1	E2	E3	E4										E14	E15	E16	E17
F1	F2	F3	F4		F6	F7	F8	F9	F10	F11			F14	F15	F16	F17
G1	G2	G3	G4		G6	G7					G12		G14	G15	G16	G17
H1	H2	H3	H4		H6						H12		H14	H15	H16	H17
J1	J2	J3	J4		J6		**TMP92CZ26A**				J12		J14	J15	J16	J17
K1	K2	K3	K4		K6		P-FBGA228				K12		K14	K15	K16	K17
L1	L2	L3	L4		L6		TOP VIEW				L12		L14	L15	L16	L17
M1	M2	M3	M4		M6	M7	M8	M9	M10	M11	M12		M14	M15	M16	M17
N1	N2	N3	N4										N14	N15	N16	N17
P1	P2	P3		P5	P6	P7	P8	P9	P10	P11	P12	P13		P15	P16	P17
R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17
T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17
U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17

Figure 2.1.1 Pin assignment diagram (P-FBGA228)

4 balls of A1, A17, U1 and U17 (most outside 4 corner of BGA package) are Dummy Balls. These balls are not connected with internal LSI chip, electrical characteristics.

A1 and U1, A17 and U17 are shorted in internal package. It is recommended that using to OPEN check of mounting if mounting this LSI to Target board.

Example: If checking signal (or voltage) via A1-U1-U17-A17, short U17 and U1 on Target board beforehand, and input signal (or voltage) from A1, and check voltage of A17.

Table 2.1.1 Pin number and the name

Ball No.	Pin name	Ball No.	Pin name	Ball No.	Pin name	Ball No.	Pin name
A1	Dummy1	D9	P73,EA24	J15	PT5,LD13	P15	PK4,LHSYNC
A2	PG2,AN2, MX	D10	PF4,I2S1DO	J16	P47,A7	P16	P13,D11
A3	PA6,KI6	D11	PF7,SDCLK	J17	P46,A6	P17	P14,D12
A4	PA5,KI5	D12	PJ4,SDLUDQM	K1	PN3,KO3	R1	X2
A5	PA3,KI3	D13	P85, $\overline{\text{CSZC}}$	K2	PN4,KO4	R2	PC7,KO8
A6	PA1,KI1	D15	PU6,LD22	K3	PN5,KO5	R3	PC3,INT3,TA2IN
A7	DVCC1A5	D16	P61,A17	K4	PN6,KO6	R4	PX5,X1USB
A8	PF1,I2S0DO	D17	P60,A16	K6	DVCC3A2	R5	PP7,TB1OUT0
A9	PJ6,NDCLE	E1	P96,PX,INT4	K12	DVCC3A7	R6	PP1,TA3OUT
A10	PJ1, $\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$	E2	PW1	K14	PT4,LD12	R7	PP3,INT5,TA7OUT
A11	P87, $\overline{\text{CSXB}}$, $\overline{\text{ND1CE}}$	E3	PW2	K15	PT3,LD11	R8	PP5,INT7,TB1IN0
A12	P83, $\overline{\text{CS3}}$, $\overline{\text{CSXA}}$	E4	PW3	K16	P45,A5	R9	PR2, $\overline{\text{SPCS}}$
A13	P81, $\overline{\text{CS1}}$, $\overline{\text{SDCS}}$	E14	PU7,LD23,EO_TRGOUT	K17	P44,A4	R10	PX7
A14	P72, $\overline{\text{WRLU}}$, $\overline{\text{NDWE}}$	E15	PU4,LD20	L1	PK2,LFR	R11	PZ0,EI_PODDATA
A15	P70, $\overline{\text{RD}}$	E16	P57,A15	L2	PN7,KO7	R12	PZ2,EI_PODREQ
A16	P65,A21	E17	P56,A14	L3	PM1,MLDALM,TA1OUT	R13	PZ4,EI_TRGIN
A17	Dummy3	F1	DVCC1B1	L4	PM7,PWE	R14	PZ6,EO_MCUDATA
B1	VREFH	F2	PW6	L6	DVSS3	R15	PZ7,EO_MCUREQ
B2	PG5,AN5	F3	PW5	L12	DVSS7	R16	P15,D13
B3	PG3,AN3,MY, $\overline{\text{ADTRG}}$	F4	PW4	L14	PT2,LD10	R17	DVCC1A3
B4	PA7,KI7	F6	DVCC3A12	L15	PT1,LD9	T1	X1
B5	PA2,KI2	F7	DVCC3A11	L16	P43,A3	T2	AM0
B6	PA0,KI0	F8	DVSS11	L17	P42,A2	T3	AM1
B7	PF2,I2S0WS	F9	DVCC3A10	M1	PK3,LVSYNC	T4	PP6,TB0OUT0
B8	PF0,I2S0CKO	F10	DVSS10	M2	PC0,INT0	T5	PL0,LD0
B9	PJ5,NDALE	F11	DVCC3A9	M3	PM2, $\overline{\text{ALARM}}$, $\overline{\text{MLDALM}}$	T6	PL2,LD2
B10	PJ2, $\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$	F14	PU5,LD21	M4	P90,TXD0	T7	PL4,LD4
B11	PJ0, $\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$	F15	PU2,LD18	M6	DVCC3A3	T8	PL5,LD5
B12	P86. $\overline{\text{CSZD}}$, $\overline{\text{ND0CE}}$	F16	P55,A13	M7	DVSS4	T9	PR1,SPDO
B13	P82, $\overline{\text{CS2}}$, $\overline{\text{CSZA}}$, $\overline{\text{SDCS}}$	F17	P54,A12	M8	DVCC3A4	T10	PL6,LD6
B14	P75,R/ $\overline{\text{W}}$,NDR/ $\overline{\text{B}}$	G1	DVCC3B1	M9	DVSS5	T11	PK1,LLOAD
B15	P71, $\overline{\text{WRLL}}$, $\overline{\text{NDRE}}$	G2	PW7	M10	DVCC3A5	T12	P00,D0
B16	P64,A20	G3	PV0,SCLK0	M11	DVSS6	T13	P02,D2
B17	DVCC1A4	G4	PV1	M12	DVCC3A6	T14	P04,D4
C1	AVCC	G6	DVSS1	M14	PK7,LGOE2	T15	P06,D6
C2	VREFL	G7	DVSS12	M15	PT0,LD8	T16	P11,D9
C3	PG4,AN4	G12	DVSS9	M16	P41,A1	T17	P12,D10
C4	PG1,AN1	G14	PU3,LD19	M17	P40,A0	U1	Dummy2
C5	PA4,KI4	G15	PU0,LD16	N1	DVCC1A1	U2	$\overline{\text{RESET}}$
C6	PC5,EA27	G16	P53,A11	N2	PC1,INT1,TA0IN	U3	D+
C7	P76, $\overline{\text{WAIT}}$	G17	P52,A10	N3	P91,RXD0	U4	D-
C8	PF5,I2S1WS	H1	PV7,SCL	N4	DVSS1C	U5	DVCC1A2
C9	PF3,I2S1CKO	H2	PV6,SDA	N14	PK6,LGOE1	U6	PL1,LD1
C10	PJ7,SDCKE	H3	PV3	N15	PK5,LGOE0	U7	PL3,LD3
C11	PJ3,SDLLDQM	H4	PV2	N16	P17,D15	U8	XT1
C12	P84, $\overline{\text{CSZB}}$	H6	DVCC3A1	N17	P16,D14	U9	XT2
C13	P80, $\overline{\text{CS0}}$	H12	DVCC3A8	P1	DVCC1C	U10	PL7.LD7
C14	P67,A23	H14	PU1,LD17	P2	PC2,INT2	U11	PK0,LCP0
C15	P66,A22	H15	PT7,LD15	P3	P92,SCLK0, $\overline{\text{CTS0}}$	U12	P01,D1
C16	P63,A19	H16	P51,A9	P5	PX4,CLKOUT, LDIV	U13	P03,D3
C17	P62,A18	H17	P50,A8	P6	PP2,TA5OUT	U14	P05,D5
D1	P97,PY	J1	PN2,KO2	P7	PP4,INT6,TB0IN0	U15	P07,D7
D2	AVSS	J2	PN1,KO1	P8	PR0,SPDI	U16	P10,D8
D3	PW0	J3	PN0,KO0	P9	PR3,SPCLK	U17	Dummy4
D5	PG0,AN0	J4	PV4	P10	$\overline{\text{DBGE}}$		
D6	PC6,EA28	J6	DVSS2	P11	PZ1,EI_SYNCLK		
D7	PC4,EA26	J12	DVSS8	P12	PZ3,EI_REFCLK		
D8	P74,EA25	J14	PT6,LD14	P13	PZ5,EI_COMRESET		

Note1: The P96, P97 and PG0~PG5 operate with the AVCC power supply.
Note2: The PW0~PW7 and PV0~PV7 operate with the DVCC3B power supply.
Note3: The X1 and X2 operate with the DVCC1C power supply.

2.2 Pin names and Functions

The names of the input/output pins and their functions are described below.

Table 2.2.1 Pin names and functions (1/6)

Pin name	Number of Pins	I/O	Functions
D0 to D7	8	I/O	Data: Data bus D0 to D7.
P10 to P17 D8 to D15	8	I/O I/O	Port 1: I/O port. Input or output is specifiable in units of bit. Data : Data bus D8 to D15.
P40 to P47 A0 to A7	8	Output Output	Port 4: Output port. Address : Address bus A0 to A7.
P50 to P57 A8 to A15	8	Output Output	Port 5: Output port. Address : Address bus A8 to A15.
P60 to P67 A16 to A23	8	I/O Output	Port 6 : I/O port. Input or output is specifiable in units of bit. Address : Address bus A16 to A23.
P70 $\overline{RD}$	1	Output Output	Port 70 : Output port. Read : Outputs strobe signal to read external memory.
P71 $\overline{WRLL}$ $\overline{NDRE}$	1	I/O Output Output	Port 71 : Output port. Write : Outputs strobe signal to write data on pins D0 to D7. NAND Flash read : Outputs strobe signal to read external NAND-Flash.
P72 $\overline{WRLU}$ $\overline{NDWE}$	1	I/O Output Output	Port 72 : I/O port. Write : Outputs strobe signal to write data on pins D8 to D15. NAND Flash write : Write enable for NAND Flash.
P73 EA24	1	I/O Output	Port 73 : I/O port. Expanded address 24.
P74 EA25	1	I/O Output	Port 74 : I/O port. Expanded address 25.
P75 R/$\overline{W}$ NDR/$\overline{B}$	1	I/O Output Input	Port 75 : I/O port. Read/Write : “High” represents read or dummy cycle and “Low” write cycle. NAND Flash Ready(1) / Busy(0) input.
P76 $\overline{WAIT}$	1	I/O Input	Port 76: I/O port. Wait: Signal used to request CPU bus wait.
P80 $\overline{CS0}$	1	Output Output	Port 80: Output port. Chip select 0: Outputs “Low” when address is within specified address area.
P81 $\overline{CS1}$ $\overline{SDCS}$	1	Output Output Output	Port 81 : Output port Chip select 1: Outputs “Low” when address is within specified address area. Chip select for SDRAM : Outputs “Low” when the address is within SDRAM address area.
P82 $\overline{CS2}$ $\overline{CSZA}$ $\overline{SDCS}$	1	Output Output Output Output	Port 82 : Output port. Chip select 2: Outputs “Low” when address is within specified address area. Expanded address ZA : Outputs “Low” when address is within specified address area. Chip select for SDRAM : Outputs “0” when the address is within SDRAM address area.
P83 $\overline{CS3}$ $\overline{CSXA}$	1	Output Output Output	Port 83 : Output port. Chip select 3: Outputs “Low” when address is within specified address area. Expanded address XA : Outputs “Low” when address is within specified address area.
P84 $\overline{CSZB}$	1	Output Output	Port 84 : Output port. Expanded address ZB : Outputs “Low” when address is within specified address area.
P85 $\overline{CSZC}$	1	Output Output	Port 85 : Output port. Expanded address ZC : Outputs “Low” when address is within specified address area.

Table 2.2.1 Pin names and functions (2/6)

Pin name	Number of Pins	I/O	Functions
P86 $\overline{\text{CSZD}}$ $\overline{\text{ND0CE}}$	1	Output Output Output	Port 86 : Output port. Expanded address ZD : Outputs “Low” when address is within specified address area. Chip select of NAND Flash 0: Outputs “Low” when NAND Flash 0 is enable.
P87 $\overline{\text{CSXB}}$ $\overline{\text{ND1CE}}$	1	Output Output Output	Port 87 : Output port. Expanded address XB : Outputs “Low” when address is within specified address area. Chip select of NAND Flash 1: Outputs “Low” when NAND Flash 1 is enable.
P90 TXD0	1	I/O Output	Port 90: I/O port. Transmit data of serial 0: programmable open drain output.
P91 RXD0	1	I/O Input	Port 91: I/O port. (Schmitt input) Receive data of serial 0.
P92 SCLK0 $\overline{\text{CTS0}}$	1	I/O I/O Input	Port 92: I/O port. (Schmitt input) Clock I/O of serial 0 Enable to send data of serial 0 (Clear to send).
P96 INT4 PX	1	Input Input Output	Port 96: Input port. (schmitt input, with pull-up resistor) Interrupt request pin 4 : Interrupt request pin with programmable rising/falling edge. X-Plus : Pin connected to X+ pin for Touch Screen I/F.
P97 PY	1	Input Output	Port 97: Input port. (schmitt input) Y-Plus : Pin connected to Y+ pin for Touch Screen I/F.
PA0 to PA7 KI0 to KI7	8	Input Input	Port A0 to A7: Input port. Key input 0 to 7: For key on wake-up 0 to 7. (Schmitt input, with pull-up resistor)
PC0 INT0	1	I/O Input	Port C0: I/O port. (Schmitt input) Interrupt request pin 0 : Interrupt request pin with programmable rising/falling edge.
PC1 INT1 TA0IN	1	I/O Input Input	Port C1: I/O port. (Schmitt input) Interrupt request pin 1 : Interrupt request pin with programmable rising/falling edge. Timer A0 input: Input pin of 8 bit timer 0.
PC2 INT2	1	I/O Input	Port C2: I/O port. (Schmitt input) Interrupt request pin 2 : Interrupt request pin with programmable rising/falling edge.
PC3 INT3 TA2IN	1	I/O Input Input	Port C3: I/O port. (Schmitt input) Interrupt request pin 3 : Interrupt request pin with programmable rising/falling edge. Timer A2 input: Input pin of 8 bit timer 2.
PC4 EA26	1	I/O Output	Port C4: I/O port. Expanded address 26.
PC5 EA27	1	I/O Output	Port C5: I/O port. Expanded address 27.
PC6 EA28	1	I/O Output	Port C6: I/O port. Expanded address 28.
PC7 KO8	1	I/O Output	Port C7: I/O port. Key output 8: Key scan strobe pin (programmable open drain output).

Table 2.2.1 Pin names and functions (3/6)

Pin name	Number of Pins	I/O	Functions
PF0 I2S0CKO	1	I/O Output	Port F0: I/O port. Outputs clock of I2S0.
PF1 I2S0DO	1	I/O Output	Port F1: I/O port. Outputs data of I2S0.
PF2 I2S0WS	1	I/O Output	Port F2: I/O port. Outputs word select signal of I2S0.
PF3 I2S0WS	1	I/O Output	Port F3: I/O port. Outputs clock of I2S1.
PF4 I2S1CKO	1	I/O Output	Port F4: I/O port. Outputs data of I2S1.
PF5 I2S1WS	1	I/O Output	Port F5: I/O port. Outputs word select signal of I2S1.
PF7 SDCLK	1	Output Output	Port F7: Output port. Clock for SDRAM.
PG0 to PG1 AN0 to AN1	2	Input Input	Port G0 to G1: Input port. Analog input pin 0 to 1 : Input pin of A/D converter.
PG2 AN2 MX	1	Input Input Output	Port G2: Input port. Analog input pin 2 : Input pin of A/D converter. X-Minus : Pin connected to X- pin for Touch Screen I/F.
PG3 AN3 MY $\overline{\text{ADTRG}}$	1	Input Input Output Input	Port G3: Input port. Analog input pin 3 : Input pin of A/D converter. Y-Minus : Pin connected to Y- pin for Touch Screen I/F. A/D Trigger : Request signal of A/D start.
PG4 to PG5 AN4 to AN5	2	Input Input	Port G4 to G5: Input port. Analog input pin 4 to 5 : Input pin of A/D converter.
PJ0 $\overline{\text{SDRAS}}$ $\overline{\text{SRLLB}}$	1	Output Output Output	Port J0: Output port. Outputs strobe signal of SDRAM row address. Data enable signal for D0 to D7 of SRAM.
PJ1 $\overline{\text{SDCAS}}$ $\overline{\text{SRLUB}}$	1	Output Output Output	Port J1: Output port. Outputs strobe signal of SDRAM column address. Data enable signal for D8 to D15 of SRAM.
PJ2 $\overline{\text{SDWE}}$ $\overline{\text{SRWR}}$	1	Output Output Output	Port J2: Output port. Outputs write enable signal of SDRAM. Write enable of SRAM: Outputs strobe signal to write data.
PJ3 SDLLDQM	1	Output Output	Port J3: Output port. Data enable signal for D0 to D7 of SDRAM.
PJ4 SDLUDQM	1	Output Output	Port J4: Output port. Data enable signal for D8 to D15 of SDRAM.
PJ5 NDALE	1	I/O Output	Port J5: I/O port. Address latch enable signal of NAND Flash.
PJ6 NDCLE	1	I/O Output	Port J6: I/O port. Command latch enable signal of NAND Flash.
PJ7 SDCKE	1	Output Output	Port J7: Output port. Clock enable signal of SDRAM.

Table 2.2.1 Pin names and functions (4/6)

Pin name	Number of Pins	I/O	Functions
PK0 LCP0	1	Output Output	Port K0: Output port. Signal for LCD driver.
PK1 LLOAD	1	Output Output	Port K1: Output port. Signal for LCD driver.: Data load signal
PK2 LFR	1	Output Output	Port K2: Output port. Signal for LCD driver.
PK3 LVSYNC	1	Output Output	Port K3: Output port. Signal for LCD driver. : Vertical sync signal
PK4 LHSYNC	1	Output Input	Port K4: Output port. Signal for LCD driver. : Horizontal sync signal.
PK5 LGOE0	1	Output Output	Port K5: Output port. Signal for LCD driver.
PK6 LGOE1	1	Output Output	Port K6: Output port. Signal for LCD driver.
PK7 LGOE2	1	Output Output	Port K7: Output port. Signal for LCD driver.
PL0 to PL7 LD0 to LD7	8	Output Output	Port L0 to L7: Output port. Data bus for LCD driver: LD0 to LD7.
PM1 TA1OUT MLDALM	1	Output Output Output	Port M1: Output port. Timer A1 output: Output pin of 8 bit timer 1. Melody / Alarm output pin.
PM2 $\overline{\text{ALARM}}$ $\overline{\text{MLDALM}}$	1	Output Output Output	Port M2: Output port. Alarm output from RTC. Melody / Alarm output pin (inverted).
PM7 PWE	1	Output Output	Port M7 : Output port External power supply control output: Pin to control ON/OFF of external power supply. In stand-by mode, outputs “L” level. In other than stand-by mode, outputs “H” level.
PN0 to PN7 KO0 to KO7	8	I/O Output	Port N: I/O port. Key output 0 to 7 : Key scan strobe pin (programmable open drain output).
PP1 TA3OUT	1	I/O Output	Port P1: I/O port. Timer A3 output: Output pin of 8 bit timer 3.
PP2 TA5OUT	1	I/O Output	Port P2: I/O port. Timer A5 output: Output pin of 8 bit timer 5.
PP3 INT5 TA7OUT	1	I/O Input Output	Port P3: I/O port. (Schmitt input) Interrupt request pin 5 : Interrupt request pin with programmable rising/falling edge. Timer A7 output: Output pin of 8 bit timer 7.
PP4 INT6 TB0IN0	1	I/O Input Input	Port P4: I/O port. (Schmitt input) Interrupt request pin 6 : Interrupt request pin with programmable rising/falling edge. Timer B0 input: Input pin of 16 bit timer 0.
PP5 INT7 TB1IN0	1	I/O Input Input	Port P5: I/O port. (Schmitt input) Interrupt request pin 7 : Interrupt request pin with programmable rising/falling edge. Timer B1 input: Input pin of 16 bit timer 1.
PP6 TB0OUT0	1	Output Output	Port P6: I/O port. Timer B0 output: Output pin of 16 bit timer 0.
PP7 TB1OUT0	1	Output Output	Port P7: I/O port. Timer B1 output: Output pin of 16 bit timer 1.
PR0 SPDI	1	I/O Input	Port R0: I/O port. Data input pin of SD card.
PR1 SPDO	1	I/O Output	Port R1: I/O port. Data output pin of SD card.
PR2 $\overline{\text{SPCS}}$	1	I/O Output	Port R2: I/O port. Chip select signal of SD card.

Table 2.2.1 Pin names and functions (5/6)

Pin name	Number of Pins	I/O	Functions
PR3 SPCLK	1	I/O Output	Port R3: I/O port. Clock output pin of SD card.
PT0 to PT7 LD8 to LD15	8	I/O Output	Port T0 to T7: I/O port. Data bus for LCD driver: LD8 to LD15.
PU0 to PU4,PU6 LD16 to LD20,LD22	6	I/O Output	Port U0 to U4 , U6: I/O port Data bus for LCD driver: LD16 to LD20, LD22.
PU5 LD21	1	I/O Output	Port U5: I/O port Data bus for LCD driver: LD21
PU7 LD23 EO_TRGOUT	1	I/O Output Output	Port U7: I/O port Data bus for LCD driver: LD23 Debug mode output pin
PV0 SCLK0	1	I/O Output	Port V0 : I/O port Clock I/O of serial 0.
PV1	1	I/O	Port V1: I/O port.
PV2	1	I/O	Port V2: I/O port.
PV3 to PV4	2	Output	Port V3 to V4: Output port.
PV6 SDA	1	I/O I/O	Port V6: I/O port Send/receive data in I^2C mode.
PV7 SCL	1	I/O I/O	Port V7: I/O port Input/output clock in I^2C mode.
PW0 to PW7	8	I/O	Port W0 to W7: I/O port.
PX4 CLKOUT LDIV	1	Output Output Output	Port X4 : Output port Internal clock output pin Output pin for LCD driver
PX5 X1USB	1	I/O Input	Port X5: I/O port. Clock input pin of USB.
PX7	1	I/O	Port X7: I/O port.
PZ0 EI_PODDATA	1	I/O Input	Port Z0: I/O port. (Schmitt input) Debug mode input pin
PZ1 EI_SYNCLK	1	I/O Input	Port Z1: I/O port. (Schmitt input) Debug mode input pin
PZ2 EI_PODREQ	1	I/O Input	Port Z2: I/O port. (Schmitt input) Debug mode input pin
PZ3 EI_REFCLK	1	I/O Input	Port Z3: I/O port. (Schmitt input) Debug mode input pin
PZ4 EI_TRGIN	1	I/O Input	Port Z4: I/O port. (Schmitt input) Debug mode input pin
PZ5 EI_COMRESET	1	I/O Input	Port Z5: I/O port. (Schmitt input) Debug mode input pin
PZ6 EO_MCUDATA	1	I/O Output	Port Z6: I/O port. (Schmitt input) Debug mode output pin
PZ7 EO_MCUREQ	1	I/O Output	Port Z7: I/O port. (Schmitt input) Debug mode output pin

Table 2.2.1 Pin names and functions (6/6)

Pin name	Number of Pins	I/O	Functions
D+, D-	2	I/O	Data pin connected to USB. In case USB is not used, connect both pins to pull-up(DVCC3A) or pull-down resistor for protect current flows it.
CLKOUT	1	Output	Internal clock output pin.
AM1,AM0	2	Input	Operation mode; Fix to AM1="0",AM0="1" for 16 bit external bus starting. Fix to AM1="1",AM0="0" is prohibit to set. Fix to AM1="1",AM0="1" for BOOT (32 bit internal Mask ROM) starting. Fix to AM1="0",AM0="0" is prohibited to set.
$\overline{\text{DBGE}}$	1	Input	Input pin in debug mode. (This pin is set to "Debug mode" by input "0".)
X1/X2	2	I/O	High-frequency oscillator circuit connection pin.
XT1/XT2	2	I/O	Low-frequency oscillator circuit connection pin.
$\overline{\text{RESET}}$	1	Input	Reset : Initialize TMP92CZ26A (schmitt input , with pull-up resistor)
VREFH	1	Input	Pin for reference voltage input to A/D converter(H).
VREFL	1	Input	Pin for reference voltage input to A/D converter(L).
AVCC	1	–	Power supply pin for A/D converter.
AVSS	1	–	GND pin for AD converter (0V).
DVCC3A	12	–	Power supply pin for peripheral I/O-A (Connect all DVCC3A pins to power supply pin.)
DVCC3B	1	–	Power supply pin for peripheral I/O-B (Connect all DVCC3B pins to power supply pin.)
DVCC1A	5	–	Power supply pin for internal logic-A. (Connect all DVCC1A pins to power supply pin.)
DVCC1B	1	–	Power supply pin for internal logic-B. (Keep the voltage DVCC1A level.)
DVSSCOM	12	–	GND pin (0V). (Connect all DVSS pins to GND(0V).)
DVCC1C	1	–	Power supply pin for High speed oscillator. (Keep the voltage DVCC1A level.)
DVSS1C	1	–	GND pin (0V). (Connect to GND(0V).)
Dummy4-1	4	–	Dummy1 and Dummy2, Dummy3 and Dummy4 are shorted in package. (These pins are not connected with internal LSI chip.)

Table 2.2.2 shows the range of operational voltage for power supply pins.

Table 2.2.2 the range of operational voltage for power supply pins

Power supply pin	Range of operational voltage
DVCC1A	1.4V~1.6V
DVCC1B	
DVCC1C	
DVCC3A	3.0V~3.6V
DVCC3B	
AVCC	

3. Operation

This section describes the basic components, functions and operation of the TMP92CZ26A.

3.1 CPU

The TMP92CZ26A contains an advanced high-speed 32-bit CPU (900/H1 CPU)

3.1.1 CPU Outline

900/H1 CPU is high-speed and high-performance CPU based on 900/L1 CPU. 900/H1 CPU has expanded 32-bit internal data bus to process Instructions more quickly.

Outline is as follows:

Table 3.1.1Outline of TMP92CZ26A

Parameter	TMP92CZ26A	
Width of CPU Address Bus	24-bit	
Width of CPU Data Bus	32-bit	
Internal Operating Frequency	Max 80MHz	
Minimum Bus Cycle	1-clock access (12.5ns at 80MHz)	
Internal RAM	32-bit 2-1-1-1 clock access	
Internal Boot ROM	32 bit 2-clock access	
Internal I/O	8-bit, 2-clock access	INTC,SDRAMC, MEMC,LCDC, TSI,PORT, PMC
	16-bit, 2-clock access	MMU,USB, NDFC,SPIC,DMAC
	32-bit, 2-clock access	I2S MAC
	32-bit, 1-clock access	MAC
	8-bit, 5 to 6-clock access	TMRA,TMRB, SIO,RTC, MLD/ALM, SBI CGEAR,ADC,WDT
External memory (SRAM, MASKROM etc.)	8/16-bit 2-clock access (can insert some waits)	
External memory (SDRAM)	16-bit 1-clock access	
External memory (NAND FLASH)	8/16-bit 2-clock access (can inset some waits)	
Minimum Instruction Execution Cycle	1-clock(12.5ns at 80MHz)	
Conditional Jump	2-clock(25.0ns at 80MHz)	
Instruction Queue Buffer	12-byte	
Instruction Set	Compatible with TLCS-900/L1 (LDX instruction is deleted)	
CPU mode	Only maximum mode	
Micro DMA	8-channel	
Hardware DMA	6-channel	

3.1.2 Reset Operation

When resetting the TMP92CZ26A microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input Low for at least 20 system clocks (32μs at X1=10MHz).

At reset, since the clock doublers (PLL0) is bypassed and clock-gear is set to 1/16, system clock operates at 625 kHz(X1=10MHz).

When the Reset has been accepted, the CPU performs the following. CPU internal registers do not change when the Reset is released.

- Sets the Stack Pointer (XSP) to 00000000H.
- Sets bits <IFF2:0> of the Status Register (SR) to "111" (thereby setting the Interrupt Level Mask Register to level 7).
- Clears bits <RFP1:0> of the Status Register to 00 (thereby selecting Register Bank 0).

When the Reset is released, the CPU starts executing instructions according to the Program Counter settings.

- Sets the Program Counter (PC) as follows in accordance with the Reset Vector stored at address FFFF00H~FFFF02H:

  PC<7:0> ← data in location FFFF00H
  PC<15:8> ← data in location FFFF01H
  PC<23:16> ← data in location FFFF02H

When the Reset is accepted, the CPU sets internal I/O, ports and other pins as follows.

- Initializes the internal I/O registers as table of "Special Function Register" in Section 5.

Note1: This LSI builds in RAM internally. However, the data in internal RAM may not be held by Reset operation. After reset, initialize the data in internal RAM.

Note2: This LSI builds in PMC function (for reducing stand-by current by blocking the power supply of DVCC1A and DVCC1C). However, if executing reset operation without supplying DVCC1A and DVCC1C, the current may flow to internal. When reset this LSI, supply the power of DVCC1A and DVCC1C first and wait until the power supply stabilizes.

Figure 3.1.2 shows reset timing chart. Figure 3.1.2 shows the example of order of supplying power and the timing of releasing reset.

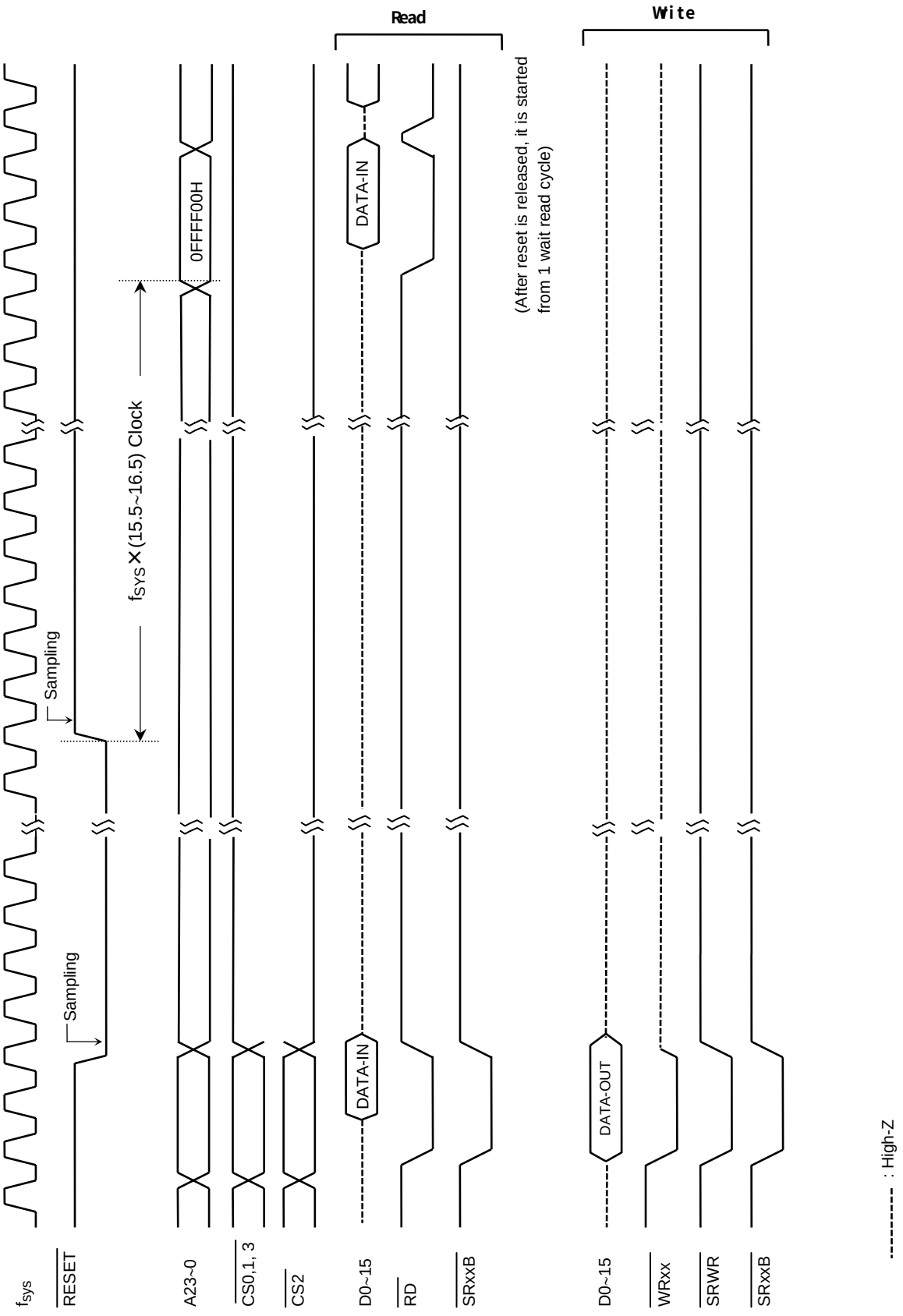
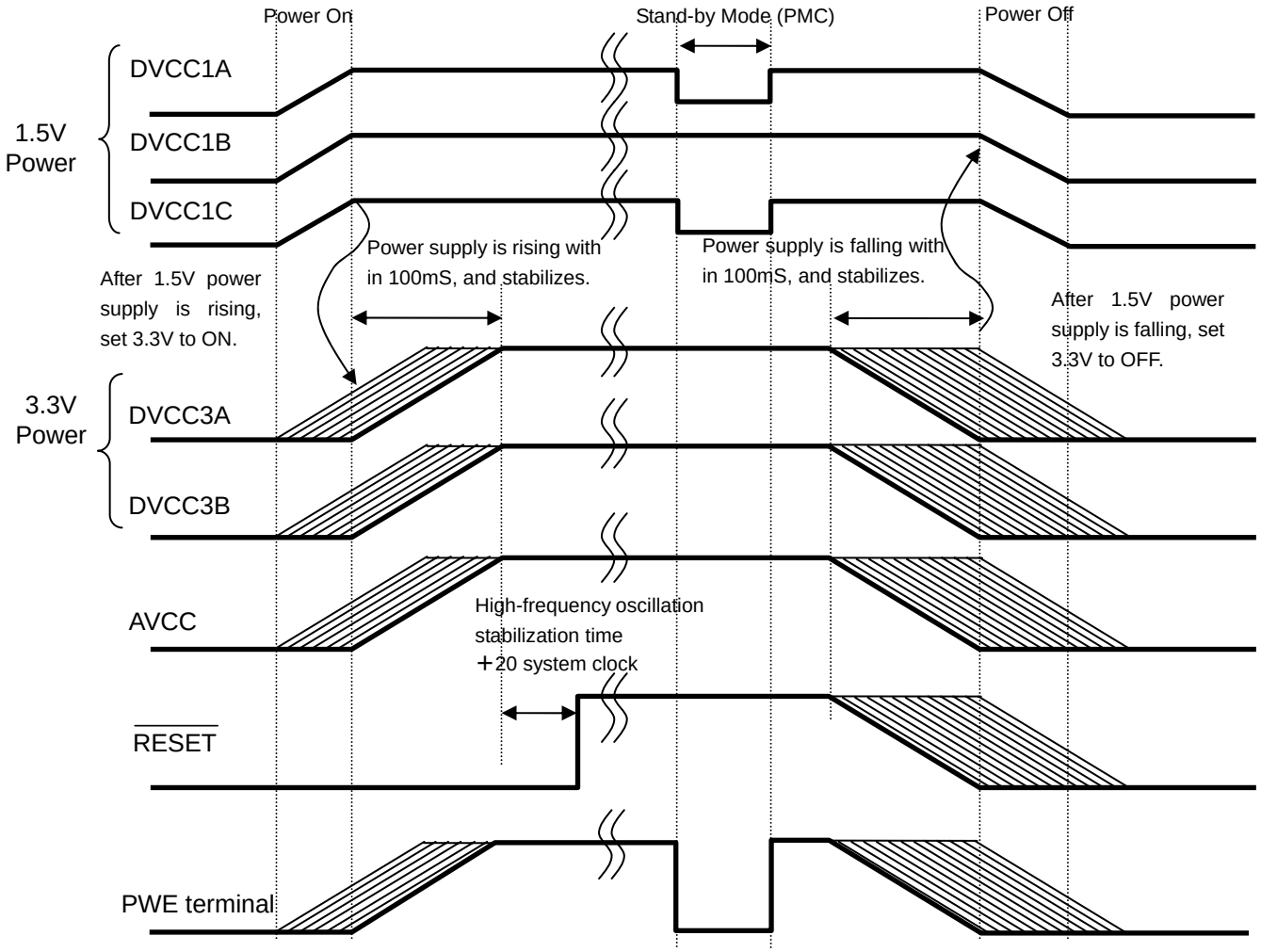


Figure 3.1.1 TMP92CZ26A Reset timing chart

This LSI has the restriction for the order of supplying power. Be sure to supply external 3.3V power with 1.5V power is supplied.



Note1: Inernal 1.5 V and External 3.3V power supply can be set to ON/OFF at the same time. However, external pin may become unstable condition momentary. Therefore, set external power supply to ON/OFF during internal power supply is stabile like above figure if there is possibility to affect machinery connected with micro controller.

Note2: When setting to ON, don't set 3.3V power supply earlier than 1.5V power supply. When setting to OFF, don't set to 3.3V power supply later than 1.5 V power supply.

Figure 3.1.2 Power on Reset Timing Example

3.1.3 Setting of AM0 and AM1

Set AM1 and AM0 pins as Table 3.1.2 shows according to system usage.

Table 3.1.2 Operation Mode Setup Table

Mode Setup input pin				Operation Mode
$\overline{\text{RESET}}$	AM1	AM0	$\overline{\text{DBGE}}$	
	0	1	0	Debug mode
			1	16-bit external bus starting
	1	0	0	Test mode (Prohibit to set)
			1	
	1	1	0	Test mode (Prohibit to set)
			1	BOOT(32-bit internal-MROM) starting (BOOT mode)
	0	0	0	Test mode (Prohibit to set)
			1	

3.2 Memory Map

Figure 3.2.1 is a memory map of the TMP92CZ26A.

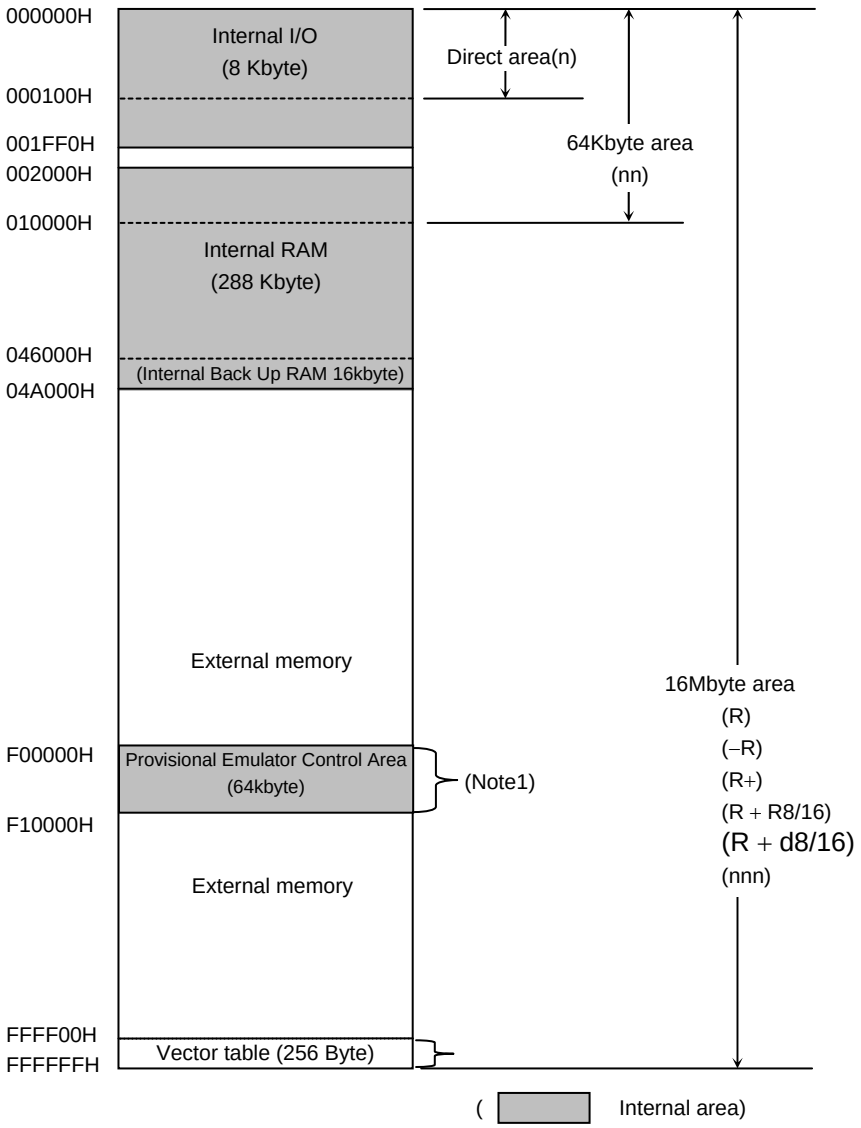


Figure 3.2.1 Memory Map

Note1: Don't use specified 64kbyte area of above 16M byte when using debug mode. This is because the area is reserved for control in the debug mode.

Note2: Don't use the last 16-byte area (FFFFF0H to FFFFFFH). This area is reserved as internal area.

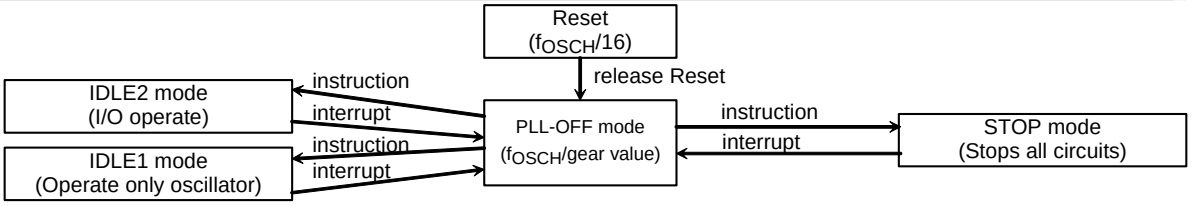
3.3 Clock Function and Standby Function

TMP92CZ26A contains (1) clock gear, (2) clock doubler (PLL), (3) standby controller and (4) noise-reducing circuit. They are used for low-power, low-noise systems.
This chapter is organized as follows:

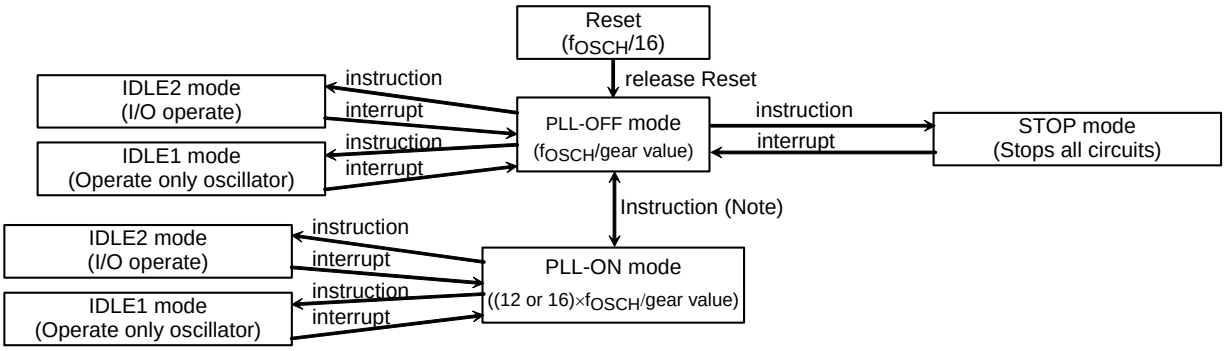
The clock operating modes are as follows: (a) PLL-OFF Mode (X1, X2 pins only), (b) PLL-ON Mode (X1, X2, and PLL).

Figure 3.3.1 shows a transition figure.

The clock frequency input from the X1 and X2 pins is called fOSCH and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<GEAR2:0> is called the system clock fSYS. And one cycle of fSYS is defined to as one state.



(a) PLL-OFF mode transition figure



(b) PLL-OFF , PLL-ON mode transition figure

Note 1: If you shift from PLL-ON mode to PLL-OFF mode, execute following setting in the same order.
(1) Change CPU clock (Set “0” to PLLCR0<FCSEL>)
(2) Stop PLL circuit (Set “0” to PLLCR1<PLLON>)

Note 2: It’s prohibited to shift from PLL-ON mode to STOP mode directly.
You should set PLL-OFF mode once, and then shift to STOP mode.

Figure 3.3.1 System clock block diagram

The clock frequency input from the X1 and X2 pins is called f_{OSCH} and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<GEAR2:0> is called the system clock f_{SYS}. And one cycle of f_{SYS} is defined to as one state.

3.3.1 Block diagram of system clock

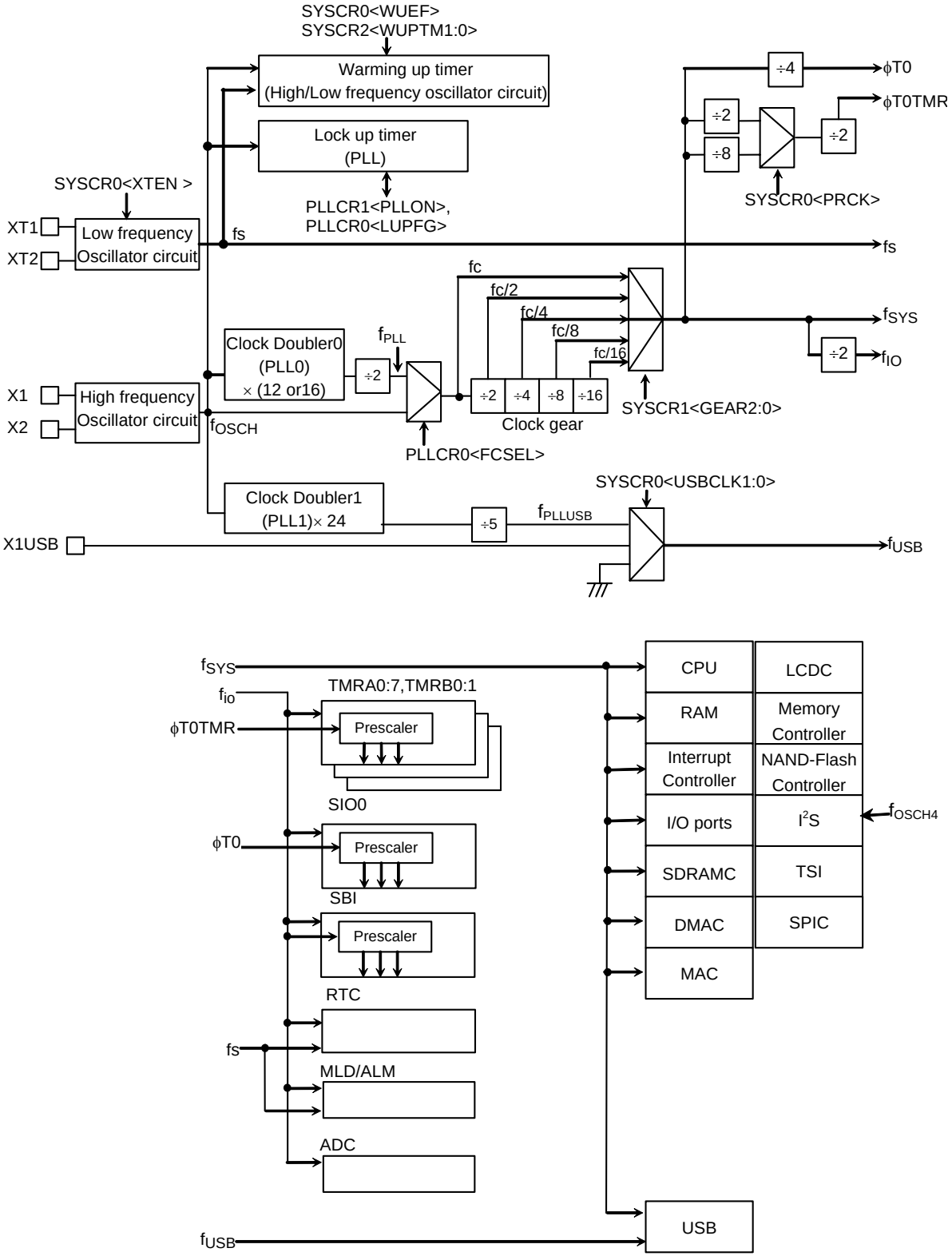


Figure 3.3.2 Block Diagram of System clock

TMP92CZ26A has two PLL circuits: one is for CPU (PLL0) and the other for USB (PLL1). Each PLL can be controlled independently. Frequency of external oscillator is 6 to 10MHz. Don't connect oscillator more than10MHz. When clock is input by using external oscillator, range of input frequency is 6 to10MHz. Don't input the clock over 10MHz.

Table 3.3.1 Setting example for f_{OSCH}

	High frequency: f_{OSCH}	System clock: f_{SYS}	System clock: f_{SYS}	USB clock: f_{USB}
(a) PLL, USB (PLL0 ON/PLL1ON)	10.0 MHz	Max 80 MHz	Max 60 MHz	48 MHz
(b) PLL, No USB (PLL0 ON/PLL1OFF)	Max 10.0 MHz	Max 80 MHz	Max 60 MHz	–
(c) No PLL, No USB (PLL0 OFF/PLL1OFF)	Max 10.0 MHz	Max 10 MHz	Max 10 MHz	–

Note: When using USB, set high-frequency oscillator to 10.0 MHz.

3.3.2 SFR

SYSCR0 (10E0H)

	7	6	5	4	3	2	1	0
bit Symbol		XTEN	USBCLK1	USBCLK0		WUEF		PRCK
Read/write		R/W	R/W	R/W		R/W		R/W
After Reset		1	0	0		0		0
Function		Low -frequency oscillator circuit (fs) 0: Stop 1: Oscillation	Select the clock of USB(f_{USB}) 00:Disable 01: Reserved 10:X1USB 11:f_{PLLUSB}			Warm-up Timer 0: Write Don't care Note3 1: Write start timer 0: Read end warm-up 1: Read do not end warm-up		Select Prescaler clock 0: f_{SYS}/2 1: f_{SYS}/8

SYSCR1 (10E1H)

	7	6	5	4	3	2	1	0
bit Symbol						GEAR2	GEAR1	GEAR0
Read/write						R/W		
After Reset						1	0	0
Function						Select gear value of high frequency (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 101: Reserved 110: Reserved 111: Reserved		

SYSCR2 (10E2H)

	7	6	5	4	3	2	1	0
bit Symbol	–	CKOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0		
Read/write	R/W	R/W	R/W	R/W	R/W	R/W		
After Reset	0	0	1	0	1	1		
Function	Always write "0"	Select CLKOUT 0: f_{SYS} 1: f_S	Warm-Up Timer 00: reserved 01: 2^8/inputted frequency 10:2^{14}/inputted frequency 11:2^{16}/inputted frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			

Note1: SYSCR0<bit7><bit3><bit1>,SYSCR1<bit7:3> and SYSCR2<bit1:0> are read as undefined value.

Note2: By reset, low frequency oscillator circuit is enabled.

Note3: Don't write SYSCR0 resiter during warming up. Because the warm-up end flag doesn't become enable if write "0" to SYSCR0<WUEF> bit during warming up.
(Read-modify-write is prohibited for SYSCR0 register during warming up.)

Figure 3.3.3 SFR for system clock

		7	6	5	4	3	2	1	0
EMCCR0 (10E3H)	Bit symbol	PROTECT				–	EXTIN	DRVOSCH	DRVOSCL
	Read/Write	R				R/W	R/W	R/W	R/W
	After reset	0				0	0	1	1
	Function	Protect flag 0: OFF 1: ON				Always write “0”.	1: External clock	fc oscillator drive ability 1: NORMAL 0: WEAK	fs oscillator drive ability 1: NORMAL 0: WEAK
EMCCR1 (10E4H)	Bit symbol	Switching the protect ON/OFF by write to following 1st-KEY,2nd-KEY 1st-KEY: EMCCR1=5AH,EMCCR2=A5H in succession write 2nd-KEY: EMCCR1=A5H,EMCCR2=5AH in succession write							
	Read/Write								
	After reset								
	Function								
EMCCR2 (10E5H)	Bit symbol								
	Read/Write								
	After reset								
	Function								

Note: In case restarting the oscillator in the stop oscillation state (e.g. Restart the oscillator in STOP mode), set EMCCR0<DRVOSCH>, <DRVOSCL>=”1”.

Figure 3.3.4 SFR for system clock

PLLCR0 (10E8H)

	7	6	5	4	3	2	1	0
bit symbol		FCSEL	LUPFG					
Read/Write		R/W	R					
After reset		0	0					
Function		Select fc-clock 0 : f_{OSCH} 1 : f_{PLL}	Lock-up timer Status flag 0 : not end 1 : end					

Note: Be carefull that logic of PLLCR0<LUPFG> is different from 900/L1's DFM.

PLLCR1 (10E9H)

	7	6	5	4	3	2	1	0
bit symbol	PLL0	PLL1	LUPSEL					PLLTIMES
Read/Write	R/W	R/W	R/W					R/W
After reset	0	0	0					0
Function	PLL0 for CPU 0: Off 1: On	PLL1 for USB 0: Off 1: On	Select stage of Lock up counter 0: 12 stage (for PLL0) 1:13 stage (for PLL1)					Select the number of PLL 0: ×12 1: ×16

Figure 3.3.5 SFR for PLL

PxDR (xxxxH)

	7	6	5	4	3	2	1	0
bit symbol	Px7D	Px6D	Px5D	Px4D	Px3D	Px2D	Px1D	Px0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Output/Input buffer drive-register for standby-mode							

(Purpose and method of using)

- This register is used to set each pin-status at stand-by mode.
- All ports have this format's register. ("x" means port-name.)
- For each register, refer to 3.5 Function of Ports.
- Before "HALT" instruction is executed, set each register pin-status. They will be effective after CPU executes "HALT" instruction.
- This register is effective in all stand-by modes (IDLE2, IDLE1 or STOP).
- This register is effective when using PMC function. For details, refer to PMC section.

The truth table to control Output/Input-buffer is below.

OE	PxnD	Output buffer	Input buffer
0	0	OFF	OFF
0	1	OFF	ON
1	0	OFF	OFF
1	1	ON	OFF

Note1: OE means an output enable signal before stand-by mode. Basically, PxCR is used as OE.

Note2: "n" in PxnD means bit-number of PORTx.

Figure 3.3.6 SFR for drive register

3.3.3 System clock controller

The system clock controller generates the system clock signal (f_{SYS}) for the CPU core and internal I/O.

SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling of each oscillator. SYSCR1<GEAR2:0> sets the high frequency clock gear to either 1, 2, 4, 8 or 16 (fc, fc/2, fc/4, fc/8, fc/16). These functions can reduce the power consumption of the equipment in which the device is installed.

The combination of settings <XEN> = "1", <SYSCK> = "0" and <GEAR2 to 0> = "100" will be PLL-OFF mode and cause the system clock (f_{SYS}) to be set to fc/16 after reset.

For example, f_{SYS} is set to 625 kHz when the 10MHz oscillator is connected to the X1 and X2 pins.

(1) Clock gear controller

f_{SYS} is set according to the contents of the Clock Gear Select Register SYSCR1<GEAR2: 0> to either fc, fc/2, fc/4, fc/8 or fc/16. Using the clock gear to select a lower value of f_{SYS} reduces power consumption.

(Example)

Changing clock gear

```
SYSCR1   EQU   10E1H

         LD    (SYSCR1),XXXXX001B   ;  Changes system clock fSYS to fc/2
         LD    (DUMMY),00H          ;  Dummy instruction
```

X: don't care

(High-speed clock gear changing)

To change the clock gear, write the register value to the SYSCR1<GEAR2 to 0> register. It is necessary the warming up time until changing after writing the register value.

There is the possibility that the instruction next to the clock gear changing instruction is executed by the clock gear before changing. To execute the instruction next to the clock gear switching instruction by the clock gear after changing, input the dummy instruction as follows (instruction to execute the write cycle).

(Example)

```
SYSCR1   EQU   10E1H
         LD    (SYSCR1),XXXXX010B   ;  Changes fSYS to fc/4
         LD    (DUMMY),00H          ;  Dummy instruction
         Instruction to be executed after clock gear changed
```

3.3.4 Clock doubler (PLL)

PLL0 outputs the f_{PLL} clock signal, which is 12 or 16 times as fast as f_{OSCH}. That is, the low-speed frequency oscillator can be used as external oscillator, even though the internal clock is high-frequency.

Since Reset initializes PLL0 to stop status, setting to PLLCR0 and PLLCR1-register is needed before use.

Like an oscillator, this circuit requires time to stabilize. This is called the lock-up time and it is measured by 12-stage binary counter. Lock-up time is about 0.41ms at f_{OSCH} = 10MHz.

PLL (PLL1) which is special for USB is build in. Lock-up time is about 0.82ms at f_{OSCH} = 10MHz measured by 13-stage binary counter.

Note1: Input frequency limitation for PLL
The limitation of input frequency (High frequency oscillation) for PLL is following.
f_{OSCH} = X to X MHz (Vcc = 1.4 to 1.6V)

Note2: PLLCR0<LUPFG>
The logic of PLLCR0<LUPFG> is different from 900/L1's DFM.
Be careful to judge an end of lock-up time.

Note3: PLLCR1<PLL0>, PLLCR1<PLL1>
It's prohibited to turn ON both PLL0 and PLL1 simultaneously.
If turning ON simultaneously, one PLL should be turn ON after finishing the lock up of the other PLL.

Figure 3.3.7 shows the frequency of f_{SYS} when using PLL and clock gear at f_{OSCH} =10MHz.

f_{OSH}	f_{PLL}	Frequency of f_{SYS}				
		fc	fc/2	fc/4	fc/8	fc/16
10MHz	f_{OSH} 10MHz	10MHz	5MHz	2.5MHz	1.25MHz	625KHz
	×12 120MHz	60MHz	30MHz	15MHz	7.5MHz	3.75MHz
	×16 160MHz	80MHz	40MHz	20MHz	10MHz	5MHz

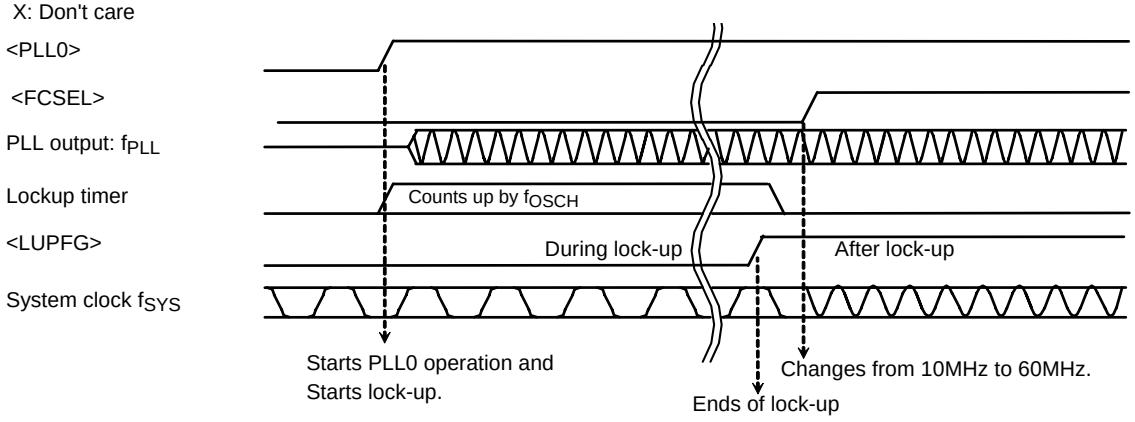
Figure 3.3.7 The frequency of f_{SYS} at f_{OSH} =10MHz

The following is a setting example for PLL0-starting and PLL0-stopping.

(Example-1) PLL0-starting

```
PLLCR0    EQU    10E8H
PLLCR1    EQU    10E9H
          LD     (PLLCR1),1XXXXXXXXB     ;  Enables PLL0 operation and starts lock-up.
LUP:      BIT    5,(PLLCR0)              ;
          JR     Z,LUP                   ;  Detects end of lock-up
          LD     (PLLCR0), X1XXXXXXB     ;  Changes fc from 10 MHz to 60 MHz.
```

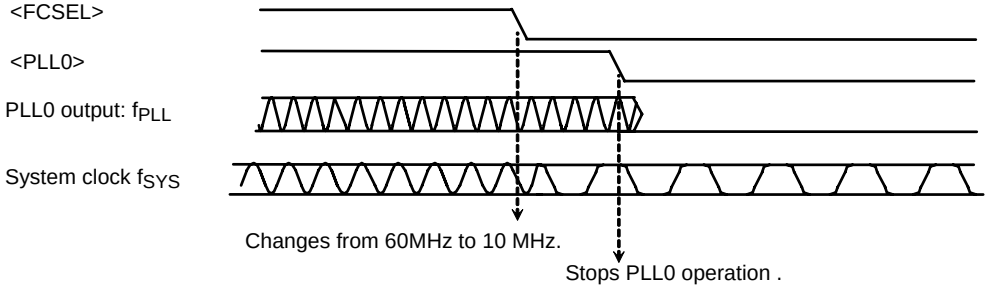
X: Don't care



(Example-2) PLL0-stopping

```
PLLCR0    EQU    10E8H
PLLCR1    EQU    10E9H
          LD     (PLLCR0),X0XXXXXXB      ;  Changes fc from 60 MHz to10 MHz.
          LD     (PLLCR1),0XXXXXXXB      ;  Stop PLL
```

X: Don't care



Note) PLL1 operates as well.

Limitation point on the use of PLL0

1. If you stop PLL operation during using PLL0, you should execute following setting in the same order.

```
        LD      (PLLCR0),X0XXXXXXB    ;   Change the clock fPLL to fOSCH
        LD      (PLLCR1),0XXXXXXXB    ;   Stop PLL0
```

X: Don't care

2. If you shift to STOP mode during using PLL, you should execute following setting in the same order.

```
        LD      (SYSCR2),XXXX01XXB    ;   Set the STOP mode
        LD      (PLLCR0), X0XXXXXXB   ;   Change the system clock fPLL to fOSCH
        LD      (PLLCR1), 0XXXXXXXB   ;   Stop PLL0
        HALT                          ;   Shift to STOP mode
```

X: Don't care

Examples of settings are below;

(1) Start Up / Change Control

(OK) High frequency oscillator operation mode(f_{OSCH})→PLL0 start up

→ PLL0 use mode (f_{PLL})

```
        LD      (PLLCR1), 1XXXXXXXB   ;   PLL0 start up / lock up start
LUP:    BIT     5,(PLLCR0)            ;
        JR      Z,LUP                 ;   Check for the flag of lock up end
        LD      (PLLCR0), X1XXXXXXB   ;   Change the system clock fOSCH to fPLL
```

X: Don't care

(2) Change / Stop Control

(OK) PLL0 use mode (f_{PLL})→ High frequency oscillator operation mode(f_{OSCH})

→ PLL0 Stop

```
        LD      (PLLCR0),X0XXXXXXB    ;   Change the system clock fPLL to fOSCH
        LD      (PLLCR1),0XXXXXXXB    ;   Stop PLL0
```

X: Don't care

(OK) PLL0 use mode (f_{PLL}) → Set the STOP mode

→High frequency oscillator operation mode (f_{OSCH}) → PLL stop

→ HALT(High frequency oscillator stop)

```
        LD      (SYSCR2),XXXX01XXB    ;   Set the STOP mode
                                      ;   (This command can be executed before use of PLL0)
        LD      (PLLCR0),X0XXXXXXB    ;   Change the system clock fPLL to fOSCH
        LD      (PLLCR1),0XXXXXXXB    ;   Stop PLL0
        HALT                          ;   Shift to STOP mode
```

X: Don't care

(NG) PLL0 use mode (f_{PLL}) → Set the STOP mode

→ HALT(High frequency oscillator stop)

```
        LD      (SYSCR2),XXXX01XXB    ;   Set the STOP mode
                                      ;   (This command can be executed before use of PLL0)
        HALT                          ;   Shift to STOP mode
```

X: Don't care

3.3.5 Noise reduction circuits

Noise reduction circuits are built in, allowing implementation of the following features.

(1) Reduced drivability for high-frequency oscillator circuit

(2) Reduced drivability for low-frequency oscillator circuit

(3) Single drive for high-frequency oscillator circuit

(4) SFR protection of register contents

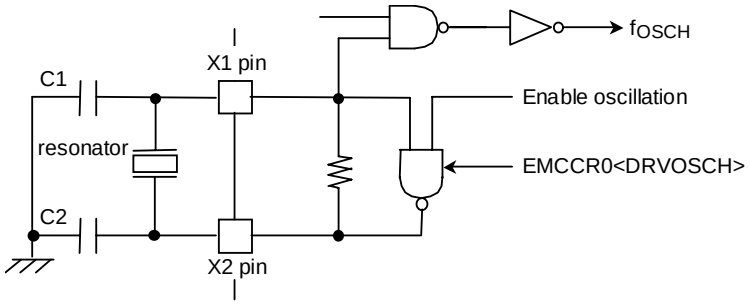
These are set in EMCCR0 to EMCCR2 registers.

(1) Reduced drivability for high-frequency oscillator circuit

(Purpose)

Reduces noise and power for oscillator when a resonator is used.

(Clock diagram)



(Setting method)

The drivability of the oscillator is reduced by writing"0" to EMCCR0<DRVOSCH> register. By reset, <DRVOSCH> is initialized to "1" and the oscillator starts oscillation by normal-drivability when the power-supply is on.

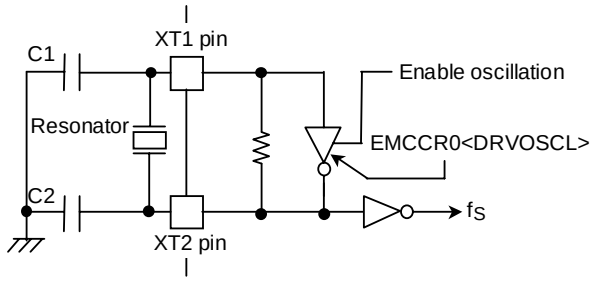
Note: This function (EMCCR0<DRVODCH>= "0") is available to use in case f_{OSCH} = 6 to 10MHz condition.

(2) Reduced drivability for low-frequency oscillator circuit

(Purpose)

Reduces noise and power for oscillator when a resonator is used.

(Block diagram)



(Setting method)

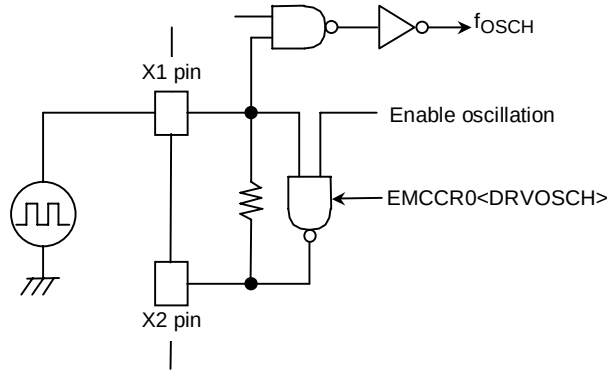
The drivability of the oscillator is reduced by writing 0 to the EMCCR0<DRVOSCL> register. By Reset, <DRVOSCL> is initialized to “1”.

(3) Single drive for high-frequency oscillator circuit

(Purpose)

Not need twin-drive and protect mistake-operation by inputted noise to X2 pin when the external-oscillator is used.

(Block diagram)



(Setting method)

The oscillator is disabled and starts operation as buffer by writing “1” to EMCCR0<EXTIN> register. X2-pin is always outputted”1”.

By reset,<EXTIN> is initialized to “0”.

Note: Do not write EMCCR0<EXTIN> = “1” when using external resonator.

(4) Runaway provision with SFR protection register

(Purpose)

Provision in runaway of program by noise mixing.

Write operation to specified SFR is prohibited so that provision program in runaway prevents that it is in the state which is fetch impossibility by stopping of clock, memory control register (Memory controller, MMU) is changed.

And error handling in runaway becomes easy by INTP0 interruption.

Specified SFR list

1. Memory controller

   B0CSL/H, B1CSL/H, B2CSL/H, B3CSL/H, BECSL/H
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3, PMEMCR,
   MEMCR0, CSTMGCR, WRTMGCR, RDTMGCR0
   RDTMGCR1, BROMCR

2. MMU

   LOCALPX/PY/PZ, LOCALLX/LY/LZ,
   LOCALRX/RY/RZ, LOCALWX/WY/WZ,
   LOCALESX/ESY/ESZ, LOCALEDX/EDY/EDZ,
   LOCALOSX/OSY/OSZ, LOCALODX/ODY/ODZ

3. Clock gear

   SYSCR0, SYSCR1, SYSCR2, EMCCR0

4. PLL

   PLLCR0,PLLCR1

5. PMC

   PMCCTL

(Operation explanation)

Execute and release of protection (write operation to specified SFR) becomes possible by setting up a double key to EMCCR1 and EMCCR2 register.

(Double key)

1st-KEY: Succession writes in 5AH at EMCCR1 and A5H at EMCCR2

2nd-KEY: Succession writes in A5H at EMCCR1 and 5AH at EMCCR2

A state of protection can be confirmed by reading EMCCR0<PROTECT>.

By reset, protection becomes OFF.

And INTP0 interruption occurs when write operation to specified SFR was executed with protection on state.

3.3.6 Standby controller

(1) Halt Modes and Port Drive-register

When the HALT instruction is executed, the operating mode switches to IDLE2, IDLE1 or STOP Mode, depending on the contents of the SYSCR2<HALTM1 to 0> register and each pin-status is set according to PxDR-register.

PxDR (xxxxH)

	7	6	5	4	3	2	1	0
bit symbol	Px7D	Px6D	Px5D	Px4D	Px3D	Px2D	Px1D	Px0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Output/Input buffer drive-register for standby-mode							

(Purpose and method of using)

- This register is used to set each pin-status at stand-by mode.
- All ports have this format's register. ("x" means port-name.)
- For each register, refer to 3.5 Function of Ports.
- Before "HALT" instruction is executed, set each register pin-status. They will be effective after CPU executes "HALT" instruction.
- This register is effective in all stand-by modes (IDLE2, IDLE1 or STOP).
- This register is effective when using PMC function. For details, refer to PMC section.

The truth table to control Output/Input-buffer is below.

OE	PxnD	Output buffer	Input buffer
0	0	OFF	OFF
0	1	OFF	ON
1	0	OFF	OFF
1	1	ON	OFF

Note1: OE means an output enable signal before stand-by mode.Basically, PxCR is used as OE.

Note2: "n" in PxnD means bit-number of PORTx.

The subsequent actions performed in each mode are as follows:

a. IDLE2: Only the CPU halts.

The internal I/O is available to select operation during IDLE2 mode by setting the following register.

Table 3.3.2 shows the registers of setting operation during IDLE2 mode.

Table 3.3.2 SFR setting operation during IDLE2 mode

Internal I/O	SFR
TMRA01	TA01RUN<I2TA01>
TMRA23	TA23RUN<I2TA23>
TMRA45	TA45RUN<I2TA45>
TMRA67	TA67RUN<I2TA67>
TMRB0	TB0RUN<I2TB0>
TMRB1	TB1RUN<I2TB1>
SIO0	SC0MOD1<I2S0>
SBI	SBIBR0<I2SBI>
A/D converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

b. IDLE1: Only the oscillator, RTC (real-time clock), and MLD continue to operate.

c. STOP: All internal circuits stop operating.

The operation of each of the different Halt Modes is described in Table 3.3.3.

Table 3.3.3 I/O operation during Halt Modes

Halt Mode		IDLE2	IDLE1	STOP
SYSCR2 <HALTM1:0>		11	10	01
Block	CPU, MAC	Stop		
	I/O ports	Depends on PxDR register setting		
	TMRA, TMRB	Available to select Operation block	Stop	
	SIO,SBI			
	A/D converter			
	WDT			
	I2S, LCDC, SDRAMC, Interrupt controller, SPIC, DMAC, NDFC, USB	Operate		
	RTC, MLD		Operate	

(2) How to release the Halt mode

These HALT states can be released by resetting or requesting an interrupt. The halt release sources are determined by the combination between the states of interrupt mask register <IFF2:0> and the halt modes. The details for releasing the HALT status are shown in Table 3.3.4.

- Released by requesting an interrupt

The operating released from the halt mode depends on the interrupt enabled status. When the interrupt request level set before executing the HALT instruction exceeds the value of interrupt mask register, the interrupt due to the source is processed after releasing the halt mode, and CPU status executing an instruction that follows the HALT instruction. When the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, releasing the halt mode is not executed.(in non-maskable interrupts, interrupt processing is processed after releasing the halt mode regardless of the value of the mask register.) However only for INT0 to INT5, INT6, INT7(unsynchronous interrupt), INTKEY,INTRTC, INTALM interrupts, even if the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, releasing the halt mode is executed. In this case, interrupt processing, and CPU starts executing the instruction next to the HALT instruction, but the interrupt request flag is held at “1”.

- Releasing by resetting

Releasing all halt status is executed by resetting.

When the STOP mode is released by RESET, it is necessary enough resetting time to set the operation of the oscillator to be stable.

When releasing the halt mode by resetting, the internal RAM data keeps the state before the “HALT” instruction is executed. However the other settings contents are initialized. (Releasing due to interrupts keeps the state before the “HALT” instruction is executed.)

Table 3.3.4 Source of Halt state clearance and Halt clearance operation

Status of Received Interrupt			Interrupt Enabled (interrupt level) ≥ (interrupt mask)			Interrupt Disabled (interrupt level) < (interrupt mask)		
Halt mode			IDLE2	IDLE1	STOP	IDLE2	IDLE1	STOP
Source of Halt state clearance	Interrupt	INTWDT	◎	×	×	–	–	–
		INT0 to 5 (Note1) INTKEY	◎	◎	◎*1	○	○	○*1
		INTUSB	◎	◎*2	×	○	○*2	×
		INT6 to 7(PORT) (Note1)	◎	◎	◎*1	○	○	○*1
		INT6 to 7(TMRB)	◎	×	×	×	×	×
		INTALM, INTRTC	◎	◎	×	○	○	×
		INTTA0 to 7, INTTP0 INTTB00 to 01, INTTB10 to 11 INTRX,INTTX, INTSBI INTI2S0 to 1, INTLCD, INTAD, INTADHP INTSPIRX,INTSPITX INTRSC, INTRDY INTDMA0 to 5	◎	×	×	×	×	×
	RESET		Reset initializes the LSI					

◎: After clearing the Halt mode, CPU starts interrupt processing.

○: After clearing the Halt mode, CPU resumes executing starting from instruction following the HALT instruction.

×: It can not be used to release the halt mode.

–: The priority level (interrupt request level) of non-maskable interrupts is fixed to 7, the highest priority level. There is not this combination type.

*1: Releasing the halt mode is executed after passing the warmming-up time.

*2: 6 interrupts of all 24 INTUSB sources can release Halt state from IDLE1 mode. Therefore, the system of low power dissipation can be built. However, the way of use is limited as below.

- Shift to IDLE1 mode :
  Execute Halt instruction when the flag of INT_SUS or INT_CLKSTOP is “1” (SUSPEND state)

- Release from IDLE1 mode :
  Release Halt state by the request of INT_RESUME or INT_CLKON (request of release SUSPEND)
  Release Halt state by the request of INT_URST_STR or INT_URST_END (request of RESET)

Note1: When the Halt mode is cleared by an INT0 interrupt of the level mode in the interrupt enabled status, hold level H until starting interrupt processing. If level L is set before holding level L, interrupt processing is correctly started.

(Example - releasing IDLE1 Mode)

An INT0 interrupt clears the Halt state when the device is in IDLE1 Mode.

```
Address
8200H    LD      (PCFC), 02H          ; Sets PC1 to INT0 interrupt.
8203H    LD      (IIMC0), 00H         ; Select INT0 interrupt rising edge.
8206H    LD      (INTE0), 06H         ; Sets INT0 interrupt level to 6.
8209H    EI      5                    ; Sets CPU interrupt level to 5.
820BH    LD      (SYSCR2), 28H        ; Sets Halt mode to IDLE1 mode.
820EH    HALT                         ; Halts CPU.
```


```
820FH    LD      XX, XX
```

(3) Operation

a. IDLE2 Mode

In IDLE2 Mode, only specific internal I/O operations, as designated by the IDLE2 Setting Register, can take place. Instruction execution by the CPU stops.

Figure 3.3.8 illustrates an example of the timing for clearance of the IDLE2 Mode Halt state by an interrupt.

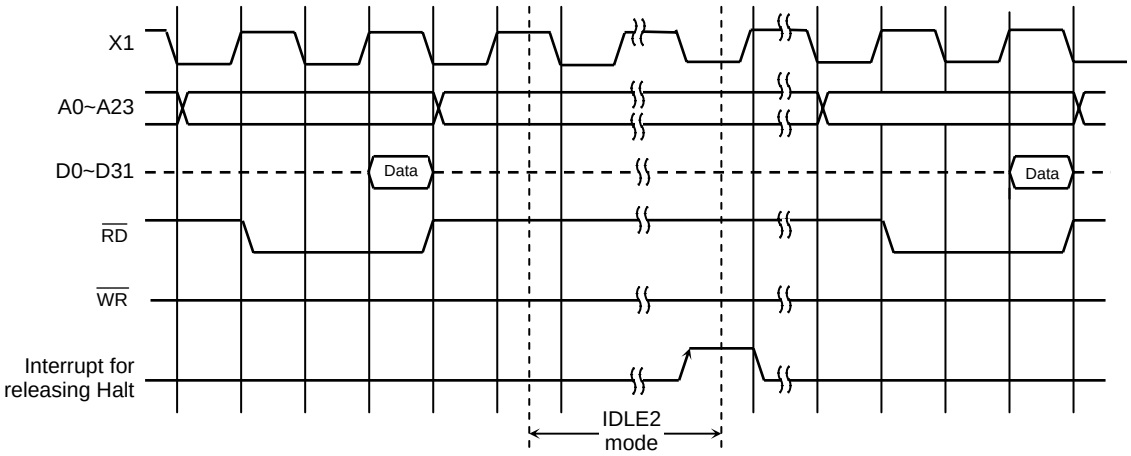


Figure 3.3.8 Timing chart for IDLE2 Mode Halt state cleared by interrupt

b. IDLE1 Mode

In IDLE1 Mode, only the internal oscillator and the RTC and MLD continue to operate. The system clock stops.

In the Halt state, the interrupt request is sampled asynchronously with the system clock; however, clearance of the Halt state (i.e. restart of operation) is synchronous with it.

Figure 3.3.9 illustrates the timing for clearance of the IDLE1 Mode Halt state by an interrupt.

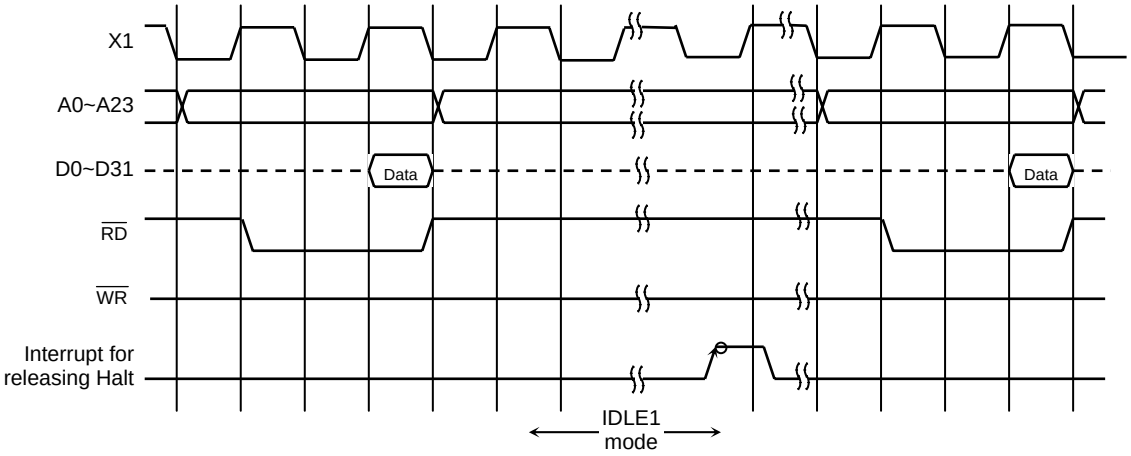


Figure 3.3.9 Timing chart for IDLE1 Mode Halt state cleared by interrupt

c. STOP Mode

When STOP Mode is selected, all internal circuits stop, including the internal oscillator.

After STOP Mode has been cleared system clock output starts when the warm-up time has elapsed, in order to allow oscillation to stabilize.

Figure 3.3.10 illustrates the timing for clearance of the STOP Mode Halt state by an interrupt.

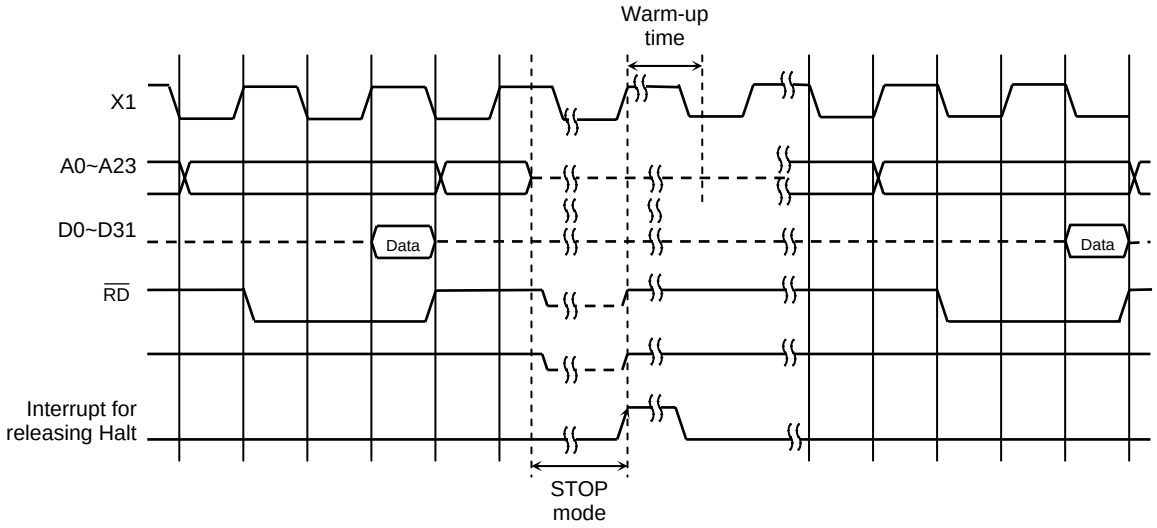


Figure 3.3.10 Timing chart for STOP Mode Halt state cleared by interrupt

Table 3.3.5 Example of warming-up time after releasing STOP-mode

@f_{OSCH} =10 MHz

SYSCR2<WUPTM1:0>		
01 (2^8)	10 (2^{14})	11 (2^{16})
25.6 us	1.6384 ms	6.5536 ms

Table 3.3.6 Input Buffer State Table

Port Name	Input Function Name	Input Buffer State						
		During Reset	When the CPU is operating		In HALT mode (IDLE2/1/STOP)			
					<PxDR>=1		<PxDR>=0	
			When Used as function Pin	When Used as Input port	When Used as function Pin	When Used as Input port	When Used as function Pin	When Used as Input port
D0-D7	D0-D7	OFF	ON upon external read	–	OFF	–	OFF	–
P10-P17	D8-D15	16bit Start OFF Boot Start ON		ON		ON		OFF
P60-P67	–	16bit Start OFF Boot Start ON	–		–		–	
P71-P74	–	ON	–		–		–	
P75	NDR/ $\overline{W}$		ON		ON		OFF	
P76	$\overline{WAIT}$							
P90	–		–		–		–	
P91	RXD0		ON		ON		OFF	
P92	$\overline{CTS0}$,SCLK0							
P96 *1	INT4							
P97	–		–		–	ON	–	
PA0-PA7 *1	KI0-7		ON		ON		OFF	
PC0	INT0							
PC1	INT1,TA0IN							
PC2	INT2							
PC3	INT3,TA2IN							
PC4-PC7	–		–		–		–	
PF0-PF5	–							
PG0-PG2 PG4,PG5 *2	–	OFF	–	ON upon port read	–	OFF	–	
PG3 *2	$\overline{ADTRG}$		ON		ON		ON	
PJ5-PJ6	–	ON	–	ON	–	ON	–	OFF
PN0-PN7	–							
PP1-PP2	–							
PP3	INT5		ON		ON		OFF	
PP4	INT6,TB0IN0							
PP5	INT7,TB1IN0							
PR0	SPDI							
PR1-PR3	–		–		–		–	
PT0-PT7	–							
PU0-PU4, PU6,PU7	–							
PU5	–		ON		ON		OFF	
PV0-PV2	–							
PV6-PV7	SDA, SCL							
PW0-PW7	–							
PX5	X1USB							
PX7	–							
PZ0-PZ5	EI_PODDATA, EI_SYNCLK, EI_PODREQ, EI_REFCLK, EI_TRGIN, EI_COMRESET						ON	
PZ6-PZ7	–		–		–		–	
$\overline{DBGE}$	–	Always ON						
D+, D-	–							
$\overline{RESET}$	–							
AM0,AM1	–							
X1,XT1	–				IDLE2/DLE1: ON			

ON: The buffer is always turned on. A current flows the input buffer if the input pin is not driven.

OFF: The buffer is always turned off.

- : No applicable

*1: Port having a pull-up/pull-down resistor.

*2: AIN input does not cause a current to flow through the buffer.

Table 3.3.7 Output buffer State Table (1/2)

Port Name	Output Function Name	Output Buffer State							
		During Reset	When the CPU is operating		In HALT mode (IDLE2/1/STOP)			
					<PxDR>=1		<PxDR>=0	
			When Used as function Pin	When Used as Output port	When Used as function Pin	When Used as Output port	When Used as function Pin	When Used as Output port
D0-7	D0-7	OFF	ON upon external write	–	OFF	–	OFF	–
P10-17	D8-15	16bit Start ON Boot Start OFF		ON		ON		OFF
P40-P47	A0-A7	ON	ON	ON	ON	ON	OFF	
P50-P57	A8-A15							
P60-67	A16-A23	16bit Start ON Boot Start OFF						
P70	$\overline{RD}$	ON						
P71	$\overline{WRLL}$, $\overline{NDRE}$	OFF						
P72	$\overline{WRLU}$, $\overline{NDWE}$							
P73	EA24							
P74	EA25							
P75	R/ $\overline{W}$							
P76	–		–		–		–	OFF
P80	$\overline{CS0}$	ON	ON	ON	ON	ON	OFF	
P81	$\overline{CS1}$, $\overline{SDCS}$							
P82	$\overline{CS2}$, $\overline{CSZA}$, $\overline{SDCS}$							
P83	$\overline{CS3}$, $\overline{CSXA}$							
P84	$\overline{CSZB}$							
P85	$\overline{CSZC}$							
P86	$\overline{CSZD}$, $\overline{ND0CE}$							
P87	$\overline{CSXB}$, $\overline{ND1CE}$							
P90	TXD0	OFF	ON		ON		OFF	
P91	–		–		–		–	
P92	SCLK0							
P96	PX	OFF		–		–		–
P97	PY							
PC0-PC3	–		–	ON	–	ON	–	OFF
PC4	EA26		ON		ON		OFF	
PC5	EA27							
PC6	EA28							
PC7	KO8							
PF0	I2S0CKO							
PF1	I2S0DO							
PF2	I2S0WS							
PF3	I2S1CKO							
PF4	I2S1DO							
PF5	I2S1WS							
PF7	SDCLK	ON						
PG2	MX	OFF		–		–		–
PG3	MY							
PJ0	$\overline{SDRAS}$, $\overline{SRLLB}$	ON						
PJ1	$\overline{SDCAS}$, $\overline{SRLUB}$							
PJ2	$\overline{SDWE}$, $\overline{SRWR}$							
PJ3	SDLLDQM							
PJ4	SDLUDQM							
PJ5	NDALE	OFF						
PJ6	NDCLE							
PJ7	SDCKE	ON		ON		ON		OFF
PK0	LCP0							
PK1	LLOAD							
PK2	LFR							
PK3	LVSYNC							
PK4	LHSYNC							
PK5	LGOE0							
PK6	LGOE1							
PK7	LGOE2							
PL0-PL7	LD0-LD7							

Table 3.3.8 Output buffer state table (2/2)

Port Name	Output Function Name	Output Buffer State						
		During Reset	When the CPU is operating		In HALT mode (IDLE2/1/STOP)			
					<PxDR>=1		<PxDR>=0	
			When Used as function Pin	When Used as Output port	When Used as function Pin	When Used as Output port	When Used as function Pin	When Used as Output port
PM1	MLDALM,TA1OUT	ON	ON	ON	ON	ON	OFF	OFF
PM2	MLDALM , ALARM							
PM7	PWE							
PN0-PN7	KO0-KO7	OFF						
PP1	TA3OUT							
PP2	TA5OUT							
PP3	TA7OUT							
PP4-PP5	–		–		–		–	
PP6	TB0OUT0	ON	ON		ON		OFF	
PP7	TB1OUT0							
PR0	–	OFF	–		–		–	
PR1	SPDO		ON		ON		OFF	
PR2	SPCS							
PR3	SPCLK							
PT0-PT7	LD8-LD15							
PU0-PU6	LD16-LD22							
PU7	LD23							
	EO_TRGOUT						ON	
PV0	SCLK0						OFF	
PV1	–		–		–		–	
PV2	–							
PV3-PV4	–	ON						
PV6	SDA	OFF	ON		ON		OFF	
PV7	SCL							
PW0-PW7	–		–		–		–	
PX4	CLKOUT, LDIV	ON	ON		ON		OFF	
PX5	–	OFF	–		–		–	
PX7	–							
PZ0-PZ5	–							
PZ6-PZ7	EO_MCUDATA, EO_MCUREQ		ON		ON		ON	
D+, D-	–	OFF	ON/OF depend on USBC operation					
X2	–	Always ON					IDLE2/1:ON, STOP: output "H"	
XT2	–						IDLE2/1:ON, STOP: output "HZ"	

ON: The buffer is always turned on. When the bus is released, however, output buffers for some pins are turned off.

OFF: The buffer is always turned off.

- : No applicable

*1: Port having a pull-up/pull-down resistor.

3.4 Boot ROM

The TMP92CZ26A contains boot ROM for downloading a user program, and supports two kinds of downloading methods.

3.4.1 Operation Modes

The TMP92CZ26A has two operation modes: MULTI mode and BOOT mode. The operation mode is selected according to the AM1 and AM0 pin levels when $\overline{\text{RESET}}$ is asserted.

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

(2) BOOT mode: After reset, the CPU fetches instructions from internal boot ROM and executes them. The boot ROM loads a user program into internal RAM from USB, or via UART, and then branches to the internal RAM. In this way the user program starts boot operation. Table 3.4.2 shows an outline of boot operation.

Table 3.4.1 Operation Modes

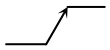
Mode Setting Pins			Operation Mode	
$\overline{\text{RESET}}$	AM1	AM0		
	0	1	MULTI	Start from external 16-bit bus memory
	1	0	TEST (Setting prohibited)	
	1	1	BOOT (Start from internal boot ROM)	
	0	0	TEST (Setting prohibited)	

Table 3.4.2 Outline of Boot Operation

Name	Priority	Loading			Operation after Loading
		Source	I/F	Destination	
(a)	1	PC (UART)	UART	Internal RAM	Branch to internal RAM
(b)	2	PC (USB_HOST)	USB		

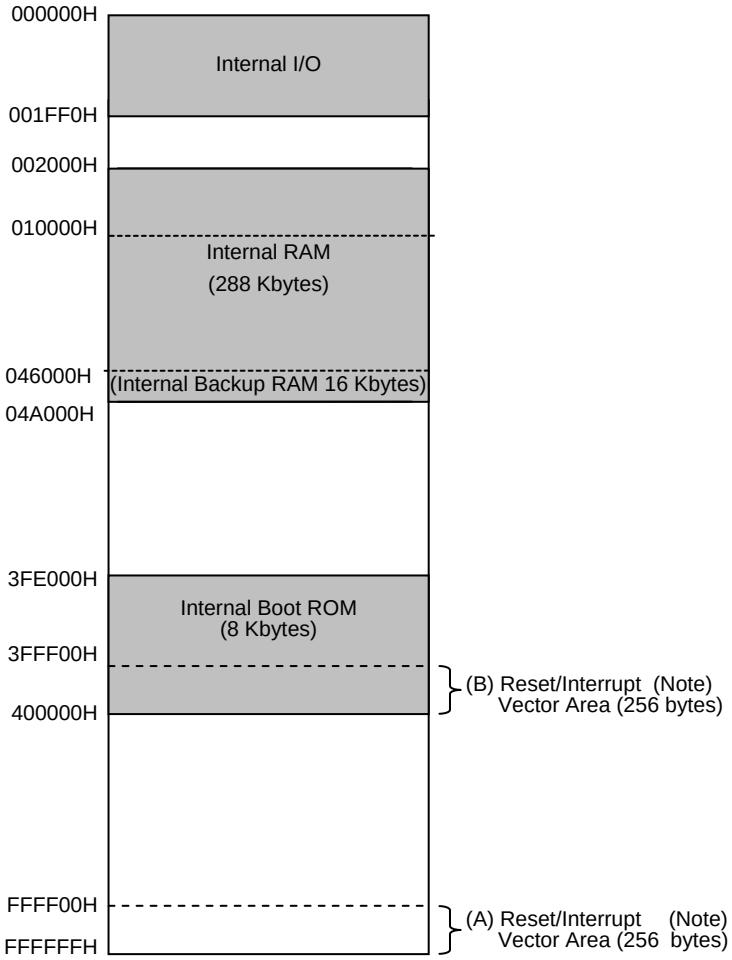
3.4.2 Hardware Specifications of Internal Boot ROM

(1) Memory map

Figure 3.4.1 shows a memory map of BOOT mode.

The boot ROM incorporated in the TMP92CZ26A is an 8-Kbyte ROM area mapped to addresses 3FE000H to 3FFFFFH.

In MULTI mode, the boot ROM is not mapped and the above area is mapped as an external area.



Note: BROMCR<VACE> = "1" : (B) when booting
BROMCR<VACE> = "0" : (A) when multi mode

Figure 3.4.1 Memory Map of BOOT Mode

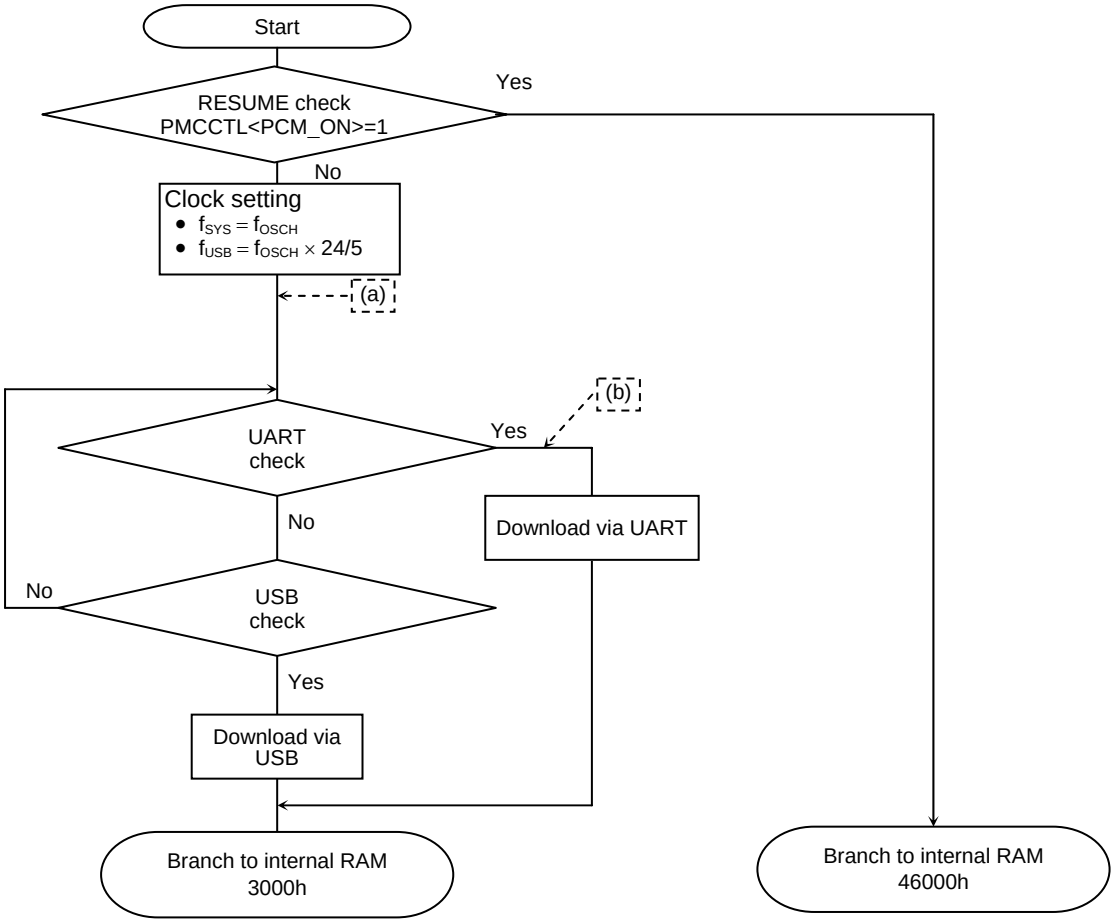
(2) Switching the boot ROM area to an external area

After the boot sequence is executed in BOOT mode, an application system program may start running without a reset being asserted. In this case, it is possible to switch the boot ROM area to an external area.

3.4.3 Outline of Boot Operation

The method for downloading a user program can be selected from two types: from UART, or via USB.

After reset, the boot program on the internal boot ROM executes as shown in Figure 3.4.2. Regardless of the downloading method used, the boot program downloads a user program into the internal RAM and then branches to the internal RAM. Figure 3.4.3 shows how the boot program uses the internal RAM (common to all the downloading methods).



Note 1: To download a user program via USB, a USB device driver and special application software are needed on the PC.

Note 2: To download a user program via UART, special application software is needed on the PC.

Note 3: The (a), (b) in the above flowchart indicate points where the settings of external port pins are changed. For details, see Table 3.4.3.

Figure 3.4.2 Flowchart for Internal Boot ROM Operation

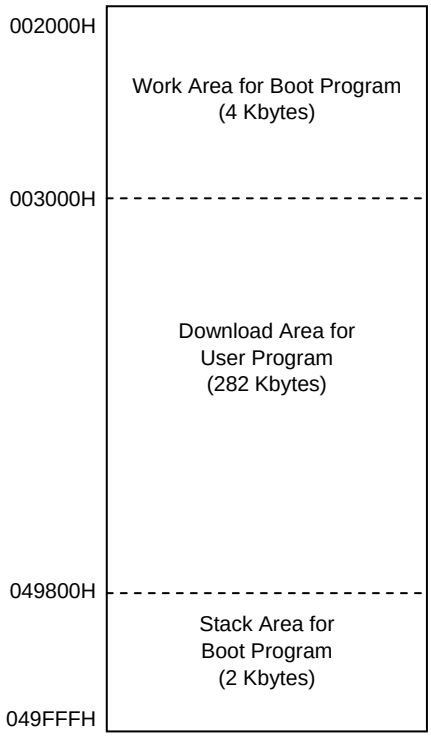


Figure 3.4.3 How the Boot Program Uses Internal RAM

(1) Port settings

Table 3.4.3 shows the port settings by the boot program. When designing your application system, please also refer to Table 3.4.4 for recommended pin connections for using the boot program.

The boot program only sets the ports shown in the table below; other ports are left as they are after reset or at startup of the boot program.

Table 3.4.3 Port Settings by the Boot Program

Port Name		Function Name	I/O	Description		
				(a)	(b)	(c)
UART	P90	TXD0	Output	No change from after reset state (input port)	No change from (a)	Set as TXD0 output pin
	P91	RXD0	Input	Set as RXD0 input pin		No change from (b)
USB	–––	D+	I/O	No change		
	–––	D–	I/O			
	PU6	PUCTL	Output	No change from after reset state (input port)	Set as output port	No change from (b)

Table 3.4.4 Recommended Pin Connections

Port Name		Function Name	I/O	Recommended Pin Connections for Each Download Method	
				UART	USB
UART	P90	TXD0	Output	Connect to the level shifter.	No special setting is needed for booting via USB.
	P91	RXD0	Input		Add a pull-up resistor (100 kΩ recommended) to prevent transition to UART processing.
USB	–––	D+	I/O	No special setting is needed for booting via UART.	Connect to the USB connector by adding a dumping resistor (27Ω recommended) and a programmable pull-up resistor (1.5 kΩ recommended). When USB is not accessed, the pin level should be fixed with a resistor to prevent flow-through current.
	–––	D–	I/O	If USB is not used, add a pull-up or pull-down resistor to prevent flow-through current on the D+/D- pins.	Connect to the USB connector by adding a dumping resistor (27Ω recommended). When USB is not accessed, the pin level should be fixed with a resistor to prevent flow-through current.
	PU6	PUCTL	Output	–	This pin is used to control ON/OFF of the D+ pin's pull-up resistor. Add a switch externally so that the pull-up is turned on when "1". Reset sets this pin as an input port, so add a pull-down resistor (100 kΩ recommended).

Note 1: When a user program is downloaded from UART and USB is used in the system, the pull-up resistor for USB's D+ pin should not be turned on in BOOT mode.

Note 2: When a user program is downloaded via USB, do not start the UART application software on the PC.

Note 3: When a user program is downloaded via UART, do not connect a USB connector.

Note 4: When USB is not used, the D+ and D- pins must be pulled up or down to prevent flow-through current.

(2) I/O register settings

Table 3.4.5 shows the I/O registers that are set by the boot program.

After the boot sequence, if execution moves to an application system program without a reset being asserted, the settings of these I/O registers must be taken into account. Also note that the registers in the CPU and the internal RAM remain in the state after execution of the boot program.

Table 3.4.5 I/O Register Settings by Boot Program

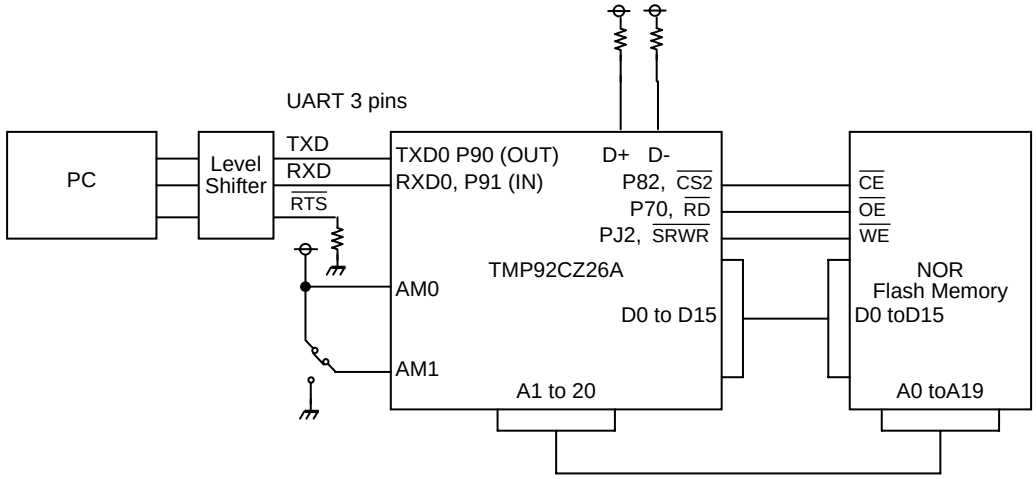
Register Name	Set Value	Description
WDMOD	00H	Watchdog timer not active
WDCR	B1H	Watchdog timer disabled
SYSCR0	70H	High-frequency and low-frequency oscillators operating
SYSCR1	00H	Clock gear = 1/1
SYSCR2	2CH	Initial value
PLLCR0	00H	PLL clock not used
PLLCR1	00H or 60H	Normally PLL is disabled. However, only in the case of booting via USB, PLL is activated for USB.
INTEUSB	04H	USB interrupt level setting
INTETC01	44H	INTTC interrupt level setting

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

3.4.4 Downloading a User Program via UART

(1) Connection example

Figure 3.4.4 shows an example of connections for downloading a user program via UART (using a 16-bit NOR Flash memory device as program memory).



Note: When USB is not used, add a pull-up or pull-down resistor to the D+ and D- pins to prevent flow-through current.

Figure 3.4.4 UART Connection Example

(2) UART interface specifications

SIO channel 0 is used for downloading a user program.

The UART communication format in BOOT mode is shown below. Before booting, the PC must also be set up with the same conditions.

Although the default baud rate is 9600 bps, this can be changed as shown in Table 3.4.8.

Serial transfer mode:	:	UART (asynchronous) mode, full-duplex
Data length	:	8 bits
Parity bit	:	None
STOP bit	:	1 bit
Handshake	:	None
Baud rate (default)	:	9600 bps

(3) UART data transfer format

Table 3.4.6 to Table 3.4.11 show the supported frequencies, data transfer format, baud rate modification command, operation command, and version management information, respectively.

Please also refer to the description of boot program operation later in this section.

Table 3.4.6 Supported Frequencies (X1)

6.00 MHz	8.00 MHz	9.00 MHz	10.00 MHz

Note: The built-in PLL (clock multiplier) is not used regardless of the oscillation frequency.

Table 3.4.7 Transfer Format

	Byte Number to Transfer	Transfer data from PC to TMP92CZ26A	Baud Rate	Transfer data from TMP92CZ26A to PC
Boot ROM	1st byte 2nd byte	Matching data (5AH) –	9600 bps	– (Frequency measurement and baud rate auto setting) OK: Echo back data (5AH) Error: No transfer
	3rd byte to 6th byte	–		Version management information (See Table 3.4.10)
	7th byte	–		Frequency information
	8th byte 9th byte	Baud rate modification command (See Table 3.4.8.) –		– OK: Echo back data Error: Error code x 3
	10th byte to (n – 4)th byte	User program Intel Hex format (binary)	New baud rate	NG: Operation stop by checksum error
	(n – 3)th byte	–		OK: SUM (High) (See (4)-c).)
	(n – 2)th byte	–		OK: SUM (Low)
	(n – 1)th byte n'th byte	User program start command (C0H) (See Table 3.4.9.) –		– OK: Echo back data (C0H) Error: Error code x 3
RAM	–	Branch to user program start address		

"Error code x 3" means that the error code is transmitted three times. For example, if the error code is 62H, the TMP92CZ26A transmits 62H three times. For error codes, see (4)-b).

Table 3.4.8 Baud Rate Modification Command

Baud Rate (bps)	9600	19200	38400	57600	115200
Modification Command	28H	18H	07H	06H	03H

Note 1: If f_{OSCH} (oscillation frequency) is 10.0 MHz, 57600 and 115200 bps are not supported.

Note 2: If f_{OSCH} (oscillation frequency) is 6.00, 8.00, or 9.00 MHz, 38400, 57600, and 115200 bps are not supported.

Table 3.4.9 Operation Command

Operation Command	Operation
C0H	User program start

Table 3.4.10 Version Management Information

Version Information	ASCII Code
FRM1	46H, 52H, 4DH, 31H

Table 3.4.11 data of measuring frequency

X1-X2 oscillator frequency (MHz)	6.000	8.000	9.000	10.000
	09H	0AH	08H	0BH

(4) Description of the UART boot program operation

The boot program receives a user program sent from the PC via UART and transfers it to the internal RAM. If the transfer ends normally, the boot program calculates SUM and sends the result to the PC before executing the user program. The execution start address is the first address received. The boot program enables users to perform customized on-board programming.

When UART is used to download a user program, the maximum allowed program size is 282 Kbytes (3000H – 49800H). (The extended Intel Hex format is supported.)

a) Operation procedure

1. Connect the serial cable. This must be done before the microcontroller is reset.
2. Set the AM1 and AM0 pins to “1” and reset the microcontroller.
3. The receive data in the 1st byte is matching data (5AH). Upon starting in BOOT mode, the boot program goes to a state in which it waits for matching data. When matching data is received, the initial baud rate of the serial channel is automatically set to 9600 bps.
4. The 2nd byte is used to echo back 5AH to the PC upon completion of the automatic baud rate setting in the 1st byte. If automatic baud rate setting fails, the boot program stops operation.
5. The 3rd through 6th bytes are used to send the version management information of the boot program in ASCII code. The PC should check that the correct version of the boot program is used.
6. The 7th byte is used to send information on the measured frequency. The PC should check that the frequency of the resonator is measured correctly.
7. The receive data in the 8th byte is baud rate modification data. The five kinds of baud rate modification data shown in Table 3.4.8 are available. Even when

the baud rate is not changed, the initial baud rate data (28H: 9600 bps) must be sent. Baud rate modification becomes effective after the echo back transmission is completed.

8. The 9th byte is used to echo back the received data to the PC when the data received in the 8th byte is one of the baud rate modification data corresponding to the operating frequency of the microcontroller. Then, the baud rate is changed. If the received baud rate data does not correspond to the operating frequency, the boot program stops operation after sending the baud rate modification error code (62H).
9. The receive data in the 10th to (n-4)th bytes is received as binary data in Intel Hex format. No echo back data is returned to the PC.
   The boot program ignores received data and does not send error code to the PC until it receives the start mark (3AH for ":") of Intel Hex format. After receiving the start mark, the boot program receives a range of data from record length to checksum and writes the received data to the specified RAM addresses successively.
   If a receive error or checksum error occurs, the boot program stops operation without sending error code to the PC.
   The boot program executes the SUM calculation routine upon detecting the end record. Thus, after sending the end record, the PC should be placed in a state in which it waits for SUM data.
10. The (n-3)th and (n-2)th bytes are used to send the SUM value to the PC in the order of upper byte and lower byte. For details on how to calculate SUM, see "SUM calculation" to be described later. SUM calculation is performed after detecting the end record only when no receives error or checksum error has occurred. Immediately after SUM calculation is completed, the boot program sends the SUM value to the PC. After sending the end record, the PC should determine whether or not writing to RAM has completed successfully based on whether or not the SUM value is received from the boot program.
11. After sending the SUM value, the boot program waits for the user program start command (C0H). If the SUM value is correct, the PC should send the user program start command in the (n-1)th byte.
12. The n'th byte is used to echo back the user program start command to the PC. After sending the echo back data, the boot program sets the stack pointer to 4A000H and jumps to the address that is received first as Intel Hex format data.
13. If the user program start command is not correct or a receive error has occurred, the boot program stops operation after sending the error code to the PC three times.

b) Error codes

The boot program uses the error codes shown in Table 3.4.12 to notify the PC of its processing status.

Table 3.4.12 Error Codes

Error Code	Meaning
62H	Unsupported baud rate
64H	Invalid operation command
A1H	Framing error in received data
A3H	Overrun error in received data

Note 1: If a receive error occurs while a user program is being received, no error code will be sent to the PC.

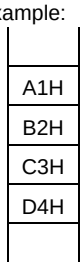
Note 2: After sending an error code, the boot program stops operation.

c) SUM calculation

1. Calculation method

SUM is calculated by adding data in bytes and is returned in words, as explained below.

Example:



If the data to be calculated consists of the 4 bytes shown to the left, SUM is calculated as follows:

A1H + B2H + C3H + D4H = 02EAH
SUM (HIGH) = 02H
SUM (LOW) = EAH

2. Data to be calculated

SUM is calculated from the data at the first received address through the last received address.

Even if received addresses are not continuous, unwritten addresses are also included in SUM calculation. The user program should not contain unwritten gaps.

d) Notes on Intel Hex format (binary)

1. After receiving the checksum of a record, the boot program waits for the start mark (3AH for ":") of the next record. If data other than 3AH is received between records, it is ignored.
2. Once the PC program has finished sending the checksum of an end record, it must wait for 2 bytes of data (upper and lower bytes of SUM) before sending any other data. This is because after receiving the checksum of an end record, the boot program calculates SUM and returns the result to the PC in 2 bytes.
3. Writing to areas other than internal RAM may cause incorrect operation. To transfer a record, set the paragraph address to 0000H.
4. Since the address pointer is initially set to 00H, the record type to be transferred first does not have to be an address record.
5. Addresses 3000H to 49800H are allocated as the user program download area.
6. A user program in Intel Hex format (ASCII codes) must be converted into binary data in advance, as explained in the example below.

Example: How to convert an Intel Hex file into binary format

The following shows how an Intel Hex format file is displayed on a text editor.

```
: 103000000607F100030000F201030000B1F16010B7
: 00000001FF
```

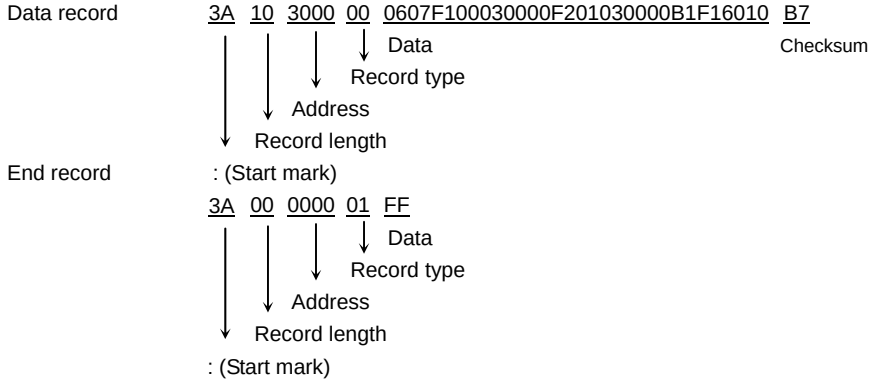
However, the actual data consists of ASCII codes, as shown below.

```
3A313033303030303030363037463130303033303030304632303130333030303030
42314631363031304237 0D0A3A303030303030303146460D0A
```

Thus, the ASCII codes must be converted into binary data based on the conversion rules shown in the table below.

ASCII Code	Binary Data
3A	3A (Only 3A remains the same.)
30 to 39	0 to 9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete

Intel Hex format



e) User program receive error

If either of the following error conditions occurs while a user program is being received, the boot program stops operation.

If the record type is other than 00H, 01H, or 02H

If a checksum error occurs

f) Measured frequency/baud rate error

When the boot program receives matching data, it measures the oscillation frequency. If an error is within plus or minus 3%, the boot program decides on that frequency.

Each baud rate includes a setting error as shown in Table 3.4.13. For example, in the case of 10.00 MHz /9600 bps, the baud rate is actually set at 9615.38 bps. To establish communication, the sum of the baud rate setting error and the measured frequency error must be within plus or minus 3 %.

Table 3.4.13 Baud Rate Setting Errors (%)

	9600 bps	19200 bps	38400 bps	57600 bps	115200 bps
6.000 MHz	0.2	0.2	–	–	–
8.000 MHz	0.2	0.2	–	–	–
9.000 MHz	0.2	–0.7	–	–	–
10.000 MHz	0.2	0.2	–1.4	–	–

–: Not supported

(5) Others

a) Handshake function

Although the $\overline{CTS}$ pin is available in the TMP92CZ26A, the boot program does not use it for transfer control.

b) RS-232C connector

The RS-232C connector must not be connected or disconnected while the boot program is running.

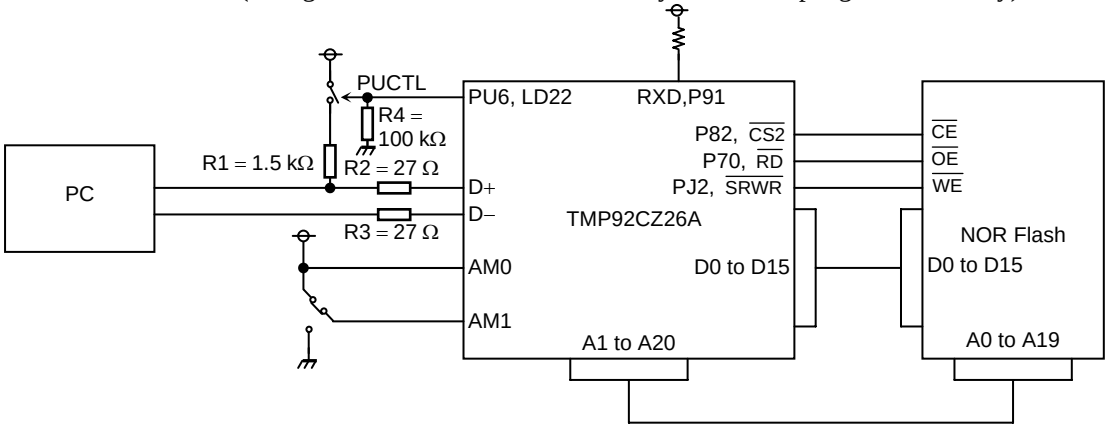
c) Software on the PC

When downloading a user program via UART, special application software is needed on the PC.

3.4.5 Downloading a User Program via USB

(1) Connection example

Figure 3.4.5 shows an example of connections for downloading a user program via USB (using a 16-bit NOR Flash memory device as program memory).



Note 1: The value of pull-up and pull-down resistors are recommended values.

Note 2: The PU6 and LD22 pins are assigned as PUCTL (pull-up control) output for USB. Be careful about this if the system uses the 24-bit TFT display function.

Note 3: Since the input gates of the D+ and D- pins are always open even at unused (unaccessed) times, these pins must be set to a fixed level to prevent flow-through current. Although the level setting is not specified in the above diagram, be sure to fix the level of the D+ and D- pins by referring to the chapter on USB.

Figure 3.4.5 USB Connection Example

(2) USB interface specifications

When a user program is downloaded via USB, the oscillation frequency should be set to 10.00 MHz. The transfer speed should be fixed to full speed (12 Mbps).

The boot program uses the following two transfer types.

Table 3.4.14 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.

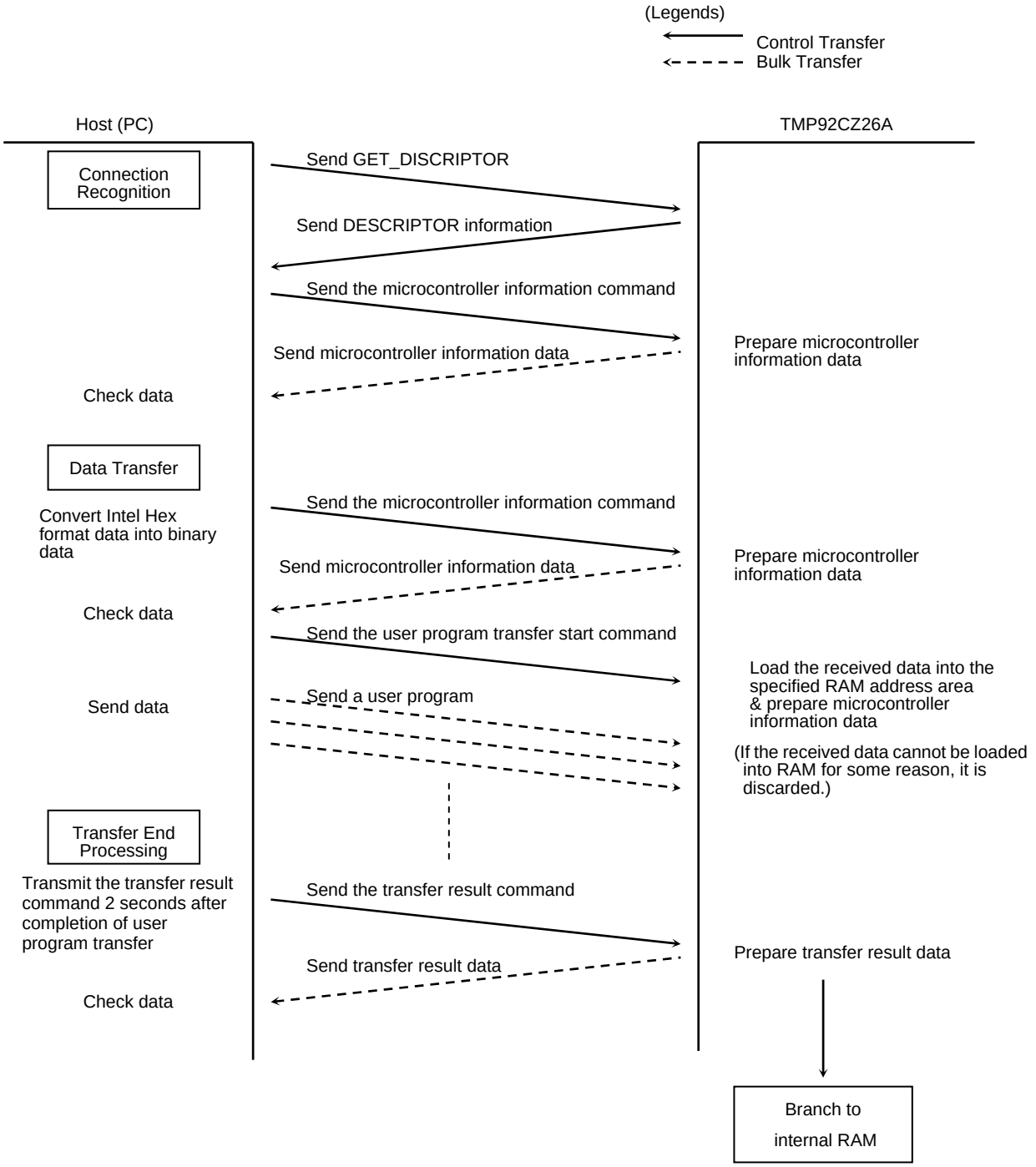


Figure 3.4.6 Overall Flowchart

Table 3.4.15 Vendor Request Commands

Command Name	Value of bRequest	Operation	Notes
Microcontroller information command	00H	Send microcontroller information	Microcontroller information data is sent by bulk IN transfer after the setup stage is completed.
User program transfer start command	02H	Receive a user program	Set the 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	04H	Send the transfer result	Transfer result data is sent by bulk IN transfer after the setup stage is completed.

Table 3.4.16 Setup Command Data Structure

Field Name	Value	Meaning
bmRequestType	40H	D7 0: Host to Device D6-D5 2: Vendor D4-D0 0: Device
bRequest	00H, 02H, 04H	00H: Microcontroller information 02H: User program transfer start 04H: User program transfer result
wValue	00H~FFFFH	Own data number (Not used by boot program)
wIndex	00H~FFFFH	User program size (Used when starting a user program transfer)
wLength	0000H	Fixed

Table 3.4.17 Standard Request Commands

Standard Request	Response Method
GET_STATUS	Automatic response by hardware
CLEAR_FEATURE	Automatic response by hardware
SET_FEATURE	Automatic response by hardware
SET_ADDRESS	Automatic response by hardware
GET_DISCRIPTOR	Automatic response by hardware
SET_DISCRIPTOR	Not supported
GET_CONFIGRATION	Automatic response by hardware
SET_CONFIGRATION	Automatic response by hardware
GET_INTERFACE	Automatic response by hardware
SET_INTERFACE	Automatic response by hardware
SYNCH_FRAME	Ignored

Table 3.4.18 Information Returned by GET_DISCRIPTOR

DeviceDescriptor

Field Name	Value	Meaning
Blength	12H	18 bytes
BdescriptorType	01H	Device descriptor
BcdUSB	0110H	USB Version 1.1
BdeviceClass	00H	Device class (Not in use)
BdeviceSubClass	00H	Sub command (Not in use)
BdeviceProtocol	00H	Protocol (Not in use)
BmaxPacketSize0	40H	EP0 maximum packet size (64 bytes)
IdVendor	0930H	Vendor ID
IdProduct	6504H	Product ID (0)
BcdDevice	0001H	Device version (v0.1)
Imanufacturer	00H	Index value of string descriptor indicating manufacturer name
Iproduct	00H	Index value of string descriptor indicating product name
IserialNumber	00H	Index value of string descriptor indicating product serial number
BnumConfigurations	01H	There is one configuration.

ConfigrationDescriptor

Field Name	Value	Meaning
bLength	09H	9 bytes
bDescriptorType	02H	Configuration descriptor
wTotalLength	0020H	Total length (32 bytes) which each descriptor of both configuration descriptor, interface and endpoint is added.
bNumInterfaces	01H	There is one interface.
bConfigurationValue	01H	Configuration number 1
iConfiguration	00H	Index value of string descriptor indicating configuration name (Not in use)
bmAttributes	80H	Bus power
MaxPower	31H	Maximum power consumption (49 mA)

InterfaceDescriptor

Field Name	Value	Meaning
bLength	09H	9 bytes
bDescriptorType	04H	Interface descriptor
bInterfaceNumber	00H	Interface number 0
bAlternateSetting	00H	Alternate setting number 0
bNumEndpoints	02H	There are two endpoints.
bInterfaceClass	FFH	Specified device
bInterfaceSubClass	00H	
bInterfaceProtocol	50H	Bulk only protocol
iIinterface	00H	Index value of string descriptor indicating interface name (Not in use)

EndpointDescriptor

Field Name	Value	Meaning
<Endpoint1>		
blength	07H	7 bytes
bDescriptorType	05H	Endpoint descriptor
bEndpointAddress	01H	EP1= OUT
bmAttributes	02H	Bulk transfer
wMaxPacketSize	0040H	Payload 64 bytes
bInterval	00H	(Ignored for bulk transfer)
<Endpoint2>		
bLength	07H	7 bytes
bDescriptor	05H	Endpoint descriptor
bEndpointAddress	82H	EP2 = IN
bmAttributes	02H	Bulk transfer
wMaxPacketSize	0040H	Payload 64 bytes
bInterval	00H	(Ignored for bulk transfer)

Table 3.4.19 Information Returned for the Microcontroller Information Command

Microcontroller Information	ASCII Code
TMP92CZ26A	54H, 4DH, 50H, 39H, 32H, 43H, 5AH, 32H, 36H,20H, 20H, 20H, 20H, 20H, 20H

Table 3.4.20 Information Returned for the User Program Transfer Result Command

Transfer Result	Value	Error Conditions
No error	00H	
User program not received	02H	The user program transfer result is received without the user program transfer start command being received first.
Received file not in Intel Hex format	04H	The first data of a user program is not ":" (3AH).
User program size error	06H	The size of a received user program is larger than the value set in wIndex of the user program transfer start command.
Download address error	08H	The specified user program download address is not in the designated area. The user program size is over 10 Kbytes.
Protocol error or other error	0AH	The user program transfer start or user program transfer result command is received first. A checksum error is detected in the Intel Hex file. A record type error is detected in the Intel Hex file. The length of an address record in the Intel Hex file is 3 or longer. The length of an end record in the Intel Hex file is other than 0.

(3) Description of the USB boot program operation

The boot program loads a user program in Intel Hex format sent from the PC into the internal RAM. When the user program has been loaded successfully, the user program starts executing from the first address received.

The boot program thus enables users to perform customized on-board programming.

a. Operation procedure

1. Connect the USB cable.
2. Set 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_DISCRIPTOR command.
4. The PC sends the microcontroller information command by control transfer (vendor request). After the setup stage is completed, the PC checks microcontroller information data by bulk IN transfer.
5. Upon receiving the microcontroller information command, the boot program prepares microcontroller information in ASCII code.
6. The PC prepares the user program to be loaded by converting an Intel Hex file into binary format.
7. The PC sends the user program transfer start command by control 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 about two seconds and then sends the user program transfer result command by control transfer (vendor request). After the setup stage is completed, the PC checks the transfer result by bulk IN transfer.
9. Upon receiving the user program transfer result command, the boot program prepares the transfer result value to be returned.
10. If the transfer result is other than OK, the boot program enters 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 (3AH for ":") of the next record. If data other than 3AH is received between records, it is ignored.
2. Since the address pointer is initially set to 00H, the record type to be transferred first does not have to be an address record.
3. Addresses 3000H to 497FFH (282 Kbytes) are allocated as the user program download area. The user program should be contained within this area.
4. A user program in Intel Hex format (normally written in ASCII code) must be converted into binary data before it can be transferred. See the example below for how to convert an Intel Hex file into binary format.

   When a user program is downloaded via USB, the maximum allowed record length is 250 bytes.

Example: Transfer data when writing 16-byte data in Intel Hex format from address 3000H

The following shows how an Intel Hex format file is displayed on a text editor.

: 103000000607F100030000F201030000B1F16010B7
: 00000001FF

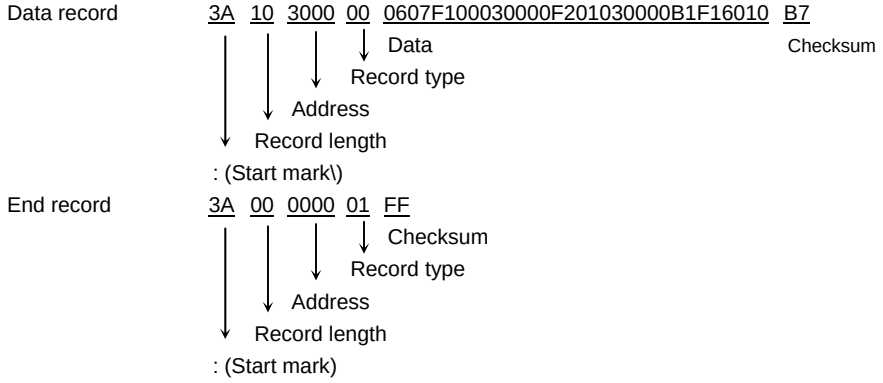
However, the actual data consists of ASCII codes, as shown below.

3A313033303030303030363037463130303033303030304632303130333030303030
42314631363031304237 0D0A3A303030303030303146460D0A

Thus, the ASCII codes must be converted into binary data based on the conversion rules shown in the table below.

ASCII Code	Binary Data
3A	3A (Only 3A remains the same.)
30~39	0~9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete

The above Intel Hex file is converted into binary data as follows:



(4) Others

a) USB connector

The USB connector must not be connected or disconnected while the boot program is running.

b) Software on the PC

To download a user program via USB, a USB device driver and special application software are needed on the PC.

3.5 Interrupts

Interrupts are controlled by the CPU Interrupt Mask Register <IFF2 to 0> (bits 12 to 14 of the Status Register) and by the built-in interrupt controller.

TMP92CZ26A has a total of 56 interrupts divided into the following five types:

Interrupts generated by CPU: 9 sources
- Software interrupts: 8 sources
- Illegal Instruction interrupt: 1 source

Internal interrupts: 38 sources
- Internal I/O interrupts: 30 sources
- Micro DMA Transfer End interrupts /HDMA Transfer End interrupts: 6 sources
- Micro DMA Transfer End interrupts: 2 source

External interrupts: 9 sources
- Interrupts on external pins (INT0 to INT7, INTKEY)

A fixed individual interrupt vector number is assigned to each interrupt source. Any one of seven levels of priority can also be assigned to each maskable interrupt. Non-maskable interrupts have a fixed priority level of 7, the highest level.

When an interrupt is generated, the interrupt controller sends the priority of that interrupt to the CPU. When more than one interrupt are generated simultaneously, the interrupt controller sends the priority value of the interrupt with the highest priority to the CPU. (The highest priority level is 7, the level used for non-maskable interrupts.)

The CPU compares the interrupt priority level which it receives with the value held in the CPU interrupt mask register <IFF2:0>. If the priority level of the interrupt is greater than or equal to the value in the interrupt mask register, the CPU accepts the interrupt.

However, software interrupts and illegal instruction interrupts generated by the CPU, and are processed irrespective of the value in <IFF2:0>.

The value in the interrupt mask register <IFF2:0> can be changed using the EI instruction (EI num sets <IFF2:0> to num). For example, the command EI3 enables the acceptance of all non-maskable interrupts and of maskable interrupts whose priority level, as set in the interrupt controller, is 3 or higher. The commands EI and EI0 enable the acceptance of all non-maskable interrupts and of maskable interrupts with a priority level of 1 or above (hence both are equivalent to the command EI1).

The DI instruction (Sets <IFF2:0> to 7) is exactly equivalent to the EI7 instruction. The DI instruction is used to disable all maskable interrupts (since the priority level for maskable interrupts ranges from 0 to 6). The EI instruction takes effect as soon as it is executed.

In addition to the general-purpose interrupt processing mode described above, there is also a micro DMA processing mode that can transfer data to internal/external memory and built-in I/O, and HDMA processing mode. In micro DMA mode the CPU, and in HDMA mode the DMA controller automatically transfers data in 1byte, 2byte or 4byte blocks. HDMA mode allows transfer faster than Micro DMA mode.

In addition, the TMP92CZ26A also has a software start function in which micro DMA and HDMA processing is requested in software rather than by an interrupt. Figure 3.5.1 is a flowchart showing overall interrupts processing.

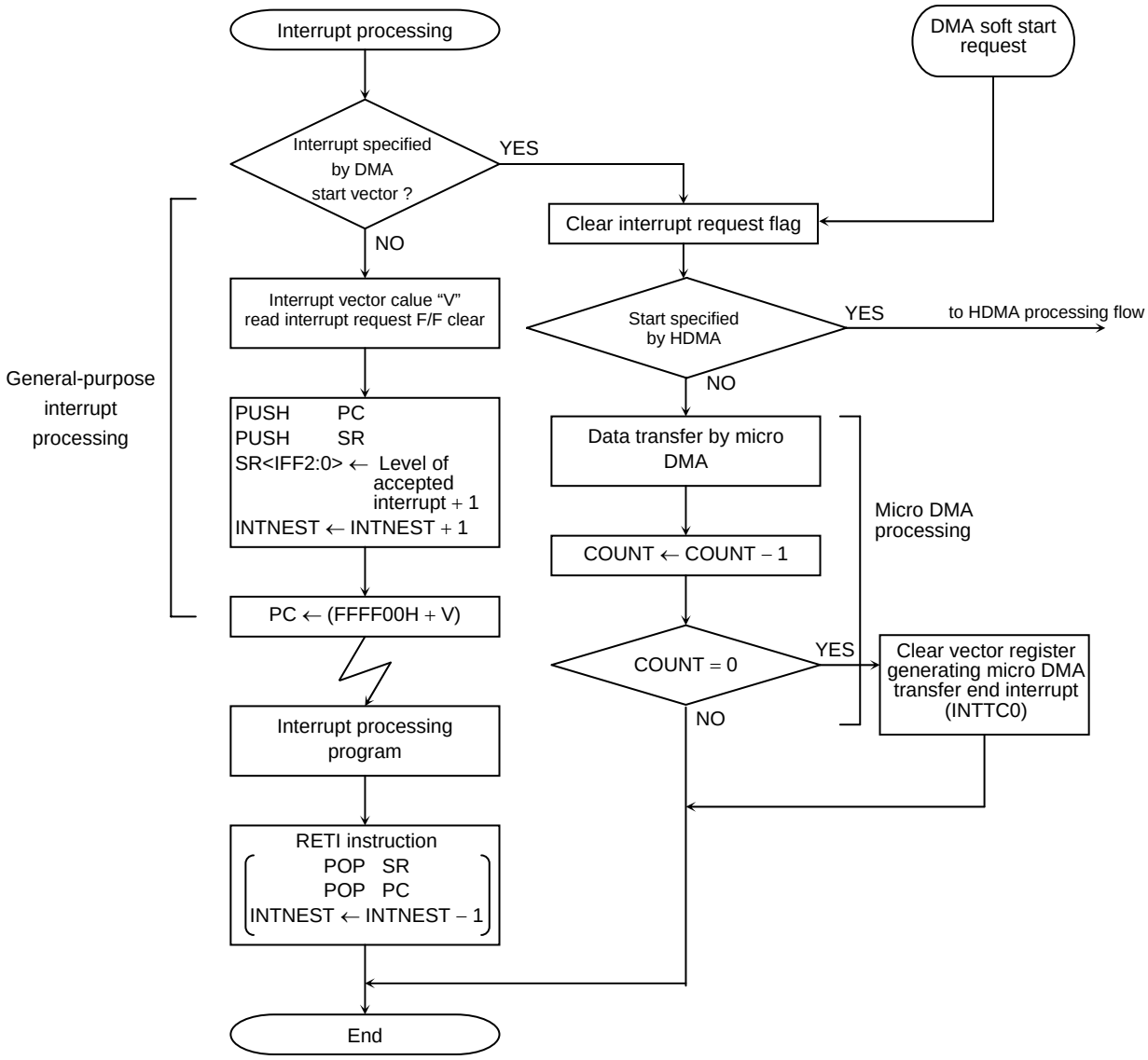


Figure 3.5.1 Interrupt processing Sequence

3.5.1 General-purpose Interrupt Processing

When the CPU accepts an interrupt, it usually performs the following sequence of operations. However, in the case of software interrupts and illegal instruction interrupts generated by the CPU, the CPU skips steps (1) and (3), and executes only steps (2), (4), and (5).

(1) The CPU reads the interrupt vector from the interrupt controller. When more than one interrupt with the same priority level have been generated simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt requests. (The default priority is determined as follows: The smaller the vector value, the higher the priority.)

(2) The CPU pushes the program counter (PC) and status register (SR) onto the top of the stack (Pointed to by XSP).

(3) The CPU sets the value of the CPU's interrupt mask register <IFF2:0> to the priority level for the accepted interrupt plus 1. However, if the priority level for the accepted interrupt is 7, the register's value is set to 7.

(4) The CPU increments the interrupt nesting counter INTNEST by 1.

(5) The CPU jumps to the address given by adding the contents of address FFFF00H + the interrupt vector, then starts the interrupt processing routine.

On completion of interrupt processing, the RETI instruction is used to return control to the main routine. RETI restores the contents of the program counter and the status register from the stack and decrements the interrupt nesting counter INTNEST by 1.

Non-maskable interrupts cannot be disabled by a user program. Maskable interrupts, however, can be enabled or disabled by a user program. A program can set the priority level for each interrupt source. (A priority level setting of 0 or 7 will disable an interrupt request.) If an interrupt request is received for an interrupt with a priority level equal to or greater than the value set in the CPU interrupt mask register <IFF2:0>, the CPU will accept the interrupt. The CPU interrupt mask register <IFF2:0> is then set to the value of the priority level for the accepted interrupt plus 1.

If during interrupt processing, an interrupt is generated with a higher priority than the interrupt currently being processed, or if, during the processing of a non-maskable interrupt processing, a non-maskable interrupt request is generated from another source, the CPU will suspend the routine which it is currently executing and accept the new interrupt. When processing of the new interrupt has been completed, the CPU will resume processing of the suspended interrupt.

If the CPU receives another interrupt request while performing processing steps (1) to (5), the new interrupt will be sampled immediately after execution of the first instruction of its interrupt processing routine. Specifying DI as the start instruction disables nesting of maskable interrupts.

After a reset, initializes the interrupt mask register <IFF2:0> to 111, disabling all maskable interrupts.

Table 3.5.1 shows the TMP92CZ26A interrupt vectors and micro DMA start vectors. FFFF00H to FFFFFFH (256 bytes) is designated as the interrupt vector area.

Table 3.5.1 TMP92CZ26A Interrupt Vectors and Micro DMA/HDMA Start Vectors

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value	Address Refer to Vector	Micro DMA /HDMA Start Vector
1	Non maskable	Reset or [SWI0] instruction	0000H	FFFF00H	
2		[SWI1] instruction	0004H	FFFF04H	
3		Illegal instruction or [SWI2] instruction	0008H	FFFF08H	
4		[SWI3] instruction	000CH	FFFF0CH	
5		[SWI4] instruction	0010H	FFFF10H	
6		[SWI5] instruction	0014H	FFFF14H	
7		[SWI6] instruction	0018H	FFFF18H	
8		[SWI7] instruction	001CH	FFFF1CH	
9		(Reserved)	0020H	FFFF20H	
10		INTWD: Watchdog timer	0024H	FFFF24H	
–	Maskable	Micro DMA (Note 2)	–	–	–
11		INT0: INT0 pin input	0028H	FFFF28H	0AH(Note 1)
12		INT1: INT1 pin input	002CH	FFFF2CH	0BH
13		INT2: INT2 pin input	0030H	FFFF30H	0CH
14		INT3: INT3 pin input	0034H	FFFF34H	0DH
15		INT4: INT4 pin input (TSI)	0038H	FFFF38H	0EH
16		INTALM: ALM(8KHz, 512Hz, 64Hz, 2Hz, 1Hz)	003CH	FFFF3CH	0FH
17		INTTA4: 8-bit timer 4	0040H	FFFF40H	10H
18		INTTA5: 8-bit timer 5	0044H	FFFF44H	11H
19		INTTA6: 8-bit timer 6	0048H	FFFF48H	12H
20		INTTA7: 8-bit timer 7	004CH	FFFF4CH	13H
21		INTP0: Protect 0 (Write to SFR)	0050H	FFFF50H	14H
22		(Reserved)	0054H	FFFF54H	15H
23		INTTA0: 0	0058H	FFFF58H	16H
24		INTTA1: 8-bit timer 1	005CH	FFFF5CH	17H
25		INTTA2: 8-bit timer 2	0060H	FFFF60H	18H
26		INTTA3: 8-bit timer 3	0064H	FFFF64H	19H
27		INTTB0: 16-bit timer 0	0068H	FFFF68H	1AH
28		INTTB1: 16-bit timer 0	006CH	FFFF6CH	1BH
29		INTKEY: Key wakeup	0070H	FFFF70H	1CH
30		INTRTC: RTC (Alarm interrupt)	0074H	FFFF74H	1DH
31		(Reserved)	0078H	FFFF78H	1EH
32		INTLCD: LCDC	007CH	FFFF7CH	1FH
33		INTRX: Serial receive end	0080H	FFFF80H	20H (Note 1)
34		INTTX: Serial transmission end	0084H	FFFF84H	21H
35		INTTB10: 16-bit timer 1	0088H	FFFF88H	22H
36		INTTB11: 16-bit timer 1	008CH	FFFF8CH	23H
37		INT5: INT5 pin input	0090H	FFFF90H	24H
38		INT6: INT6 pin input	0094H	FFFF94H	25H
39		INT7: INT7 pin input	0098H	FFFF98H	26H
40		INTI2S0: I2S (Channel 0)	009CH	FFFF9CH	27H
41		INTI2S1: I2S (Channel 1)	00A0H	FFFFA0H	28H
42		INTADM: AD Monitor function	00A4H	FFFFA4H	29H
43		INTSBI: SBI	00A8H	FFFFA8H	2AH
44		INTSPIRX: SPIC receive	00ACH	FFFFACH	2BH
45		INTSPITX: SPIC transmission	00B0H	FFFFB0H	2CH
46		INTRSC: NAND Flash controller	00B4H	FFFFB4H	2DH
47		INTRDY: NAND Flash controller	00B8H	FFFFB8H	2EH
48		INTUSB: USB	00BCH	FFFFBCH	2FH
49		(Reserved)	00C0H	FFFFC0H	30H
50		(Reserved)	00C4H	FFFFC4H	31H

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value	Address Refer to Vector	Micro DMA /HDMA Start Vector
51	Maskable	INTADHP: AD most priority conversion end	00C8H	FFFFC8H	32H
52		INTAD: AD conversion end	00CCH	FFFFCCH	33H
53		INTTC0/INTDMA0: Micro DMA0 /HDMA0 end	00D0H	FFFFD0H	34H
54		INTTC1/INTDMA1: Micro DMA1 /HDMA1 end	00D4H	FFFFD4H	35H
55		INTTC2/INTDMA2: Micro DMA2 /HDMA2 end	00D8H	FFFFD8H	36H
56		INTTC3/INTDMA3: Micro DMA3 /HDMA3 end	00DCH	FFFFDCH	37H
57		INTTC4/INTDMA4: Micro DMA4 /HDMA4 end	00E0H	FFFFE0H	38H
58		INTTC5/INTDMA5: Micro DMA5 /HDMA5 end	00E4H	FFFFE4H	39H
59		INTTC6 : Micro DMA6 end	00E8H	FFFFE8H	3AH
60		INTTC7 : Micro DMA7 end	00ECH	FFFFECH	3BH
– to –		(Reserved)	00F0H : 00FCH	FFFFF0H : FFFFFCH	– to –

Note 1: When standing-up micro DMA/HDMA , set at edge detect mode.

Note 2 : Micro DMA default priority.

Micro DMA stands up prior to other maskable interrupt.

3.5.2 Micro DMA processing

In addition to general-purpose interrupt processing, the TMP92CZ26A also includes a micro DMA function and HDMA function. This section explains about Micro DMA function. For the HDMA function, please refer 3.23 DMA controller.

Micro DMA processing for interrupt requests set by micro DMA is performed at the highest priority level for maskable interrupts (Level 6), regardless of the priority level of the interrupt source.

Because the micro DMA function has been implemented with the cooperative operation of CPU, when CPU is a state of standby (IDLE2,IDLE1,STOP) by HALT instruction, the requirement of micro DMA will be ignored (Pending).

Micro DMA is supported 8 channels and can be transferred continuously by specifying the micro DMA burst function in the following.

Note: When using the micro DMA transfer end interrupt, always write “1” to bit 7 of SIMC register.

(1) Micro DMA operation

When an interrupt request is generated by an interrupt source that specified by the micro DMA /HDMA start vector register, and Micro DMA start is specified by DMA selection register, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request. When IFF = 7, Micro DMA request cannot be accepted.

The 8 micro DMA channels allow micro DMA processing to be set for up to 8 types of interrupt at once.

When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared. Data in 1byte or 2byte or4byte blocks is automatically transferred at once from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decremented by “1”. If the value of the counter after it has been decremented is not “0”, DMA processing ends with no change in the value of the micro DMA start vector register. If the value of the decremented counter is “0”, a micro DMA transfer end interrupt (INTTC0 to INTTC7) is sent from the CPU to the interrupt controller.

In addition, the micro DMA /HDMA start vector register is cleared to “0”, the next micro DMA operation is disabled and micro DMA processing terminates.

If an interrupt request is triggered for the interrupt source in use during the interval between the time at which the micro DMA /HDMA start vector is cleared and the next setting, general-purpose interrupt processing is performed at the interrupt level set. Therefore, if the interrupt is only being used to initiate micro DMA /HDMA (and not as a general-purpose interrupt), the interrupt level should first be set to 0 (e.g., interrupt requests should be disabled).

If micro DMA and general-purpose interrupts are being used together as described above, the level of the interrupt which is being used to initiate micro DMA processing should first be set to a lower value than all the other interrupt levels. In this case, edge-triggered interrupts are the only kinds of general interrupts which can be accepted.

If micro DMA requests are set simultaneously for more than one channel, priority is not based on the interrupt priority level but on the channel number: The lower the channel number, the higher the priority (Channel 0 thus has the highest priority and channel 7 the lowest).

Note: Don't start any micro DMAs by one interrupt. If any micro DMA are set by it, micro DMA that channel number is biggest (priority is lowest) is not started.(Because interrupt flag is cleared by micro DMA that priority is highest)

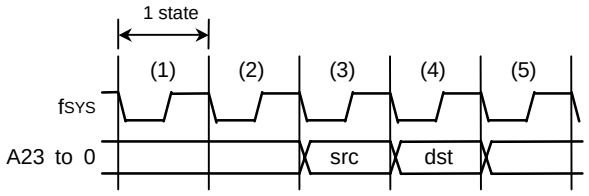
Although the control registers used for setting the transfer source and transfer destination addresses are 32 bits wide, this type of register can only output 24-bit addresses. Accordingly, micro DMA can only access 16 Mbytes (The upper 8 bits of a 32-bit address are not valid).

Three micro DMA transfer modes are supported: 1byte transfer, 2byte (One word) transfers and 4byte transfers. After a transfer in any mode, the transfer source and transfer destination addresses will either be incremented or decremented, or will remain unchanged. This simplifies the transfer of data from memory to memory, from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the various transfer modes, see section 3.5.2 (4) "Detailed description of the transfer mode register".

Since a transfer counter is a 16-bit counter, up to 65536 micro DMA processing operations can be performed per interrupt source (Provided that the transfer counter for the source is initially set to 0000H).

Micro DMA processing can be initiated by any one of 48 different interrupts – the 47 interrupts shown in the micro DMA start vectors in Table 3.5.1 and a micro DMA soft start.

Figure 3.5.2 shows a 2-byte transfer carried out using a micro DMA cycle in Transfer Destination Address INC Mode (micro DMA transfers are the same in every mode except Counter Mode). (The conditions for this cycle are as follows: both source and destination memory are internal-RAM and multipled by 4 numbered source and destination addresses).



(Note) Actually, src and dst address are not outputted to A23-0 pins because they are address of internal-RAM.

Figure 3.5.2 Timing for micro DMA cycle

States (1) and (2): Instruction fetch cycle (Prefetches the next instruction code)
State (3): Micro DMA read cycle.
State (4): Micro DMA write cycle.
State (5): (The same as in state (1), (2).)

(2) Soft start function

The TMP92CZ26A can initiate micro DMA/HDMA either with an interrupt or by using the micro DMA /HDMA soft start function, in which micro DMA or HDMA is initiated by a Write cycle which writes to the register DMAR.

Writing "1" to each bit of DMAR register causes micro DMA or HDMA to be performed once. On completion of the transfer, the bits of DMAR for the completed channel are automatically cleared to "0".

When writing again "1" to it, soft start can execute continuously until the DMA transfer counter (DMACn) or HDMA transfer counter B (HDMACBn) become "0".

When a burst is specified by the register DMAB, data is transferred continuously from the initiation of micro DMA until the value in the micro DMA transfer counter is "0".

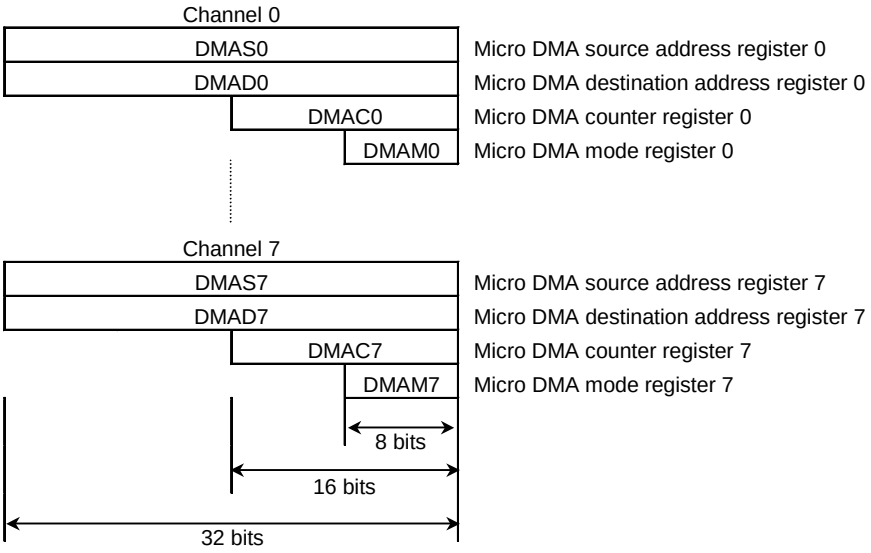
Note1: If it is started by software, don't set any channels to start in same time.

Note2: If be started sequentially, restart it after confirming micro DMA of all channels is completed (all micro DMA are set to "0").

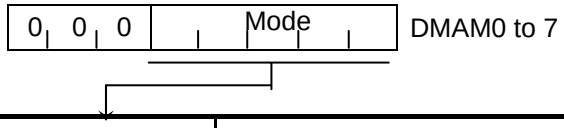
Symbol	NAME	Address	7	6	5	4	3	2	1	0
DMAR	DMA Request	109H (Prohibit RMW)	DREQ7	DREQ6	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1	DREQ0
			R/W							
			0	0	0	0	0	0	0	0
			1: Start DMA							

(3) Transfer control registers

The transfer source address and the transfer destination address are set in the following registers. An instruction of the form LDC cr,r can be used to set these registers.



(4) Detailed description of the transfer mode register



DMAMn[4:0]	Mode Description	Execution Time
0 0 0 z z	Destination INC mode (DMADn +) ← (DMASn) DMACn ← DMACn - 1 if DMACn = 0 then INTTCn	5 states
0 0 1 z z	Destination DEC mode (DMADn -) ← (DMASn) DMACn ← DMACn - 1 if DMACn = 0 then INTTCn	5 states
0 1 0 z z	Source INC mode (DMADn) ← (DMASn +) DMACn ← DMACn - 1 if DMACn = 0 then INTTCn	5 states
0 1 1 z z	Source DEC mode (DMADn) ← (DMASn -) DMACn ← DMACn – 1 if DMACn = 0 then INTTCn	5 states
1 0 0 z z	Source and destination INC mode (DMADn +) ← (DMASn +) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	6 states
1 0 1 z z	Source and destination DEC mode (DMADn -) ← (DMASn -) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	6 states
1 1 0 z z	Destination and fixed mode (DMADn) ← (DMASn) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	5 states
1 1 1 00	Counter mode DMASn ← DMASn + 1 DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	5 states

ZZ: 00 = 1-byte transfer
01 = 2-byte transfer
10 = 4-byte transfer
11 = Reserved

Note 1: n stands for the micro DMA channel number (0 to 7).

DMADn+/DMASn+: Post increment (Register value is incremented after transfer).

DMADn–/DMASn–: Post decrement (Register value is decremented after transfer).

"I/O" signifies fixed memory addresses; "memory" signifies incremented or decremented memory addresses.

Note 2: The transfer mode register should not be set to any value other than those listed above.

Note 3: The execution state number shows number of best case (1-state memory access).

3.5.3 Interrupt Controller Operation

The block diagram in Figure 3.5.3 shows the interrupt circuits. The left-hand side of the diagram shows the interrupt controller circuit. The right-hand side shows the CPU interrupt request signal circuit and the halt release circuit.

For each of the 59 interrupts channels there is an interrupt request flag (consisting of a flip-flop), an interrupt priority setting register and a micro DMA /HDMA start vector register. The interrupt request flag latches interrupt requests from the peripherals.

The flag is cleared to "0" in the following cases: when a reset occurs, when the CPU reads the channel vector of an interrupt it has received, when the CPU receives a micro DMA request (when micro DMA is set), when the CPU receives a HDMA request (when HDMA is set), when a micro DMA burst transfer is terminated, and when an instruction that clears the interrupt for that channel is executed (by writing a micro DMA start vector to the INTCLR register).

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g., INTE0 or INTE12). Six interrupt priorities levels (1 to 6) are provided. Setting an interrupt source's priority level to 0 (or 7) disables interrupt requests from that source.

If more than one interrupt request with a given priority level are generated simultaneously, the default priority (The interrupt with the lowest priority or, in other words, the interrupt with the lowest vector value) is used to determine which interrupt request is accepted first. The 3rd and 7th bits of the interrupt priority setting register indicate the state of the interrupt request flag and thus whether an interrupt request for a given channel has occurred.

If several interrupts are generated simultaneously, the interrupt controller sends the interrupt request for the interrupt with the highest priority and the interrupt's vector address to the CPU. The CPU compares the mask value set in <IFF2:0> of the status register (SR) with the priority level of the requested interrupt; if the latter is higher, the interrupt is accepted. Then the CPU sets SR<IFF2:0> to the priority level of the accepted interrupt + 1. Hence, during processing of the accepted interrupt, new interrupt requests with a priority value equal to or higher than the value set in SR<IFF2:0> (e.g., interrupts with a priority higher than the interrupt being processed) will be accepted.
When interrupt processing has been completed (e.g., after execution of a RETI instruction), the CPU restores to SR<IFF2:0> the priority value which was saved on the stack before the interrupt was generated.

The interrupt controller also includes eight registers which are used to store the micro DMA /HDMA start vector. Writing the start vector of the interrupt source for the micro DMA or /HDMA processing (See Table), enables the corresponding interrupt to be processed by micro DMA or HDMA processing. The values must be set in the micro DMA parameter registers (e.g., DMAS and DMAD) or HDMA parameter registers (e.g., HDMAS, and HDMAD) prior to micro DMA or HDMA processing.

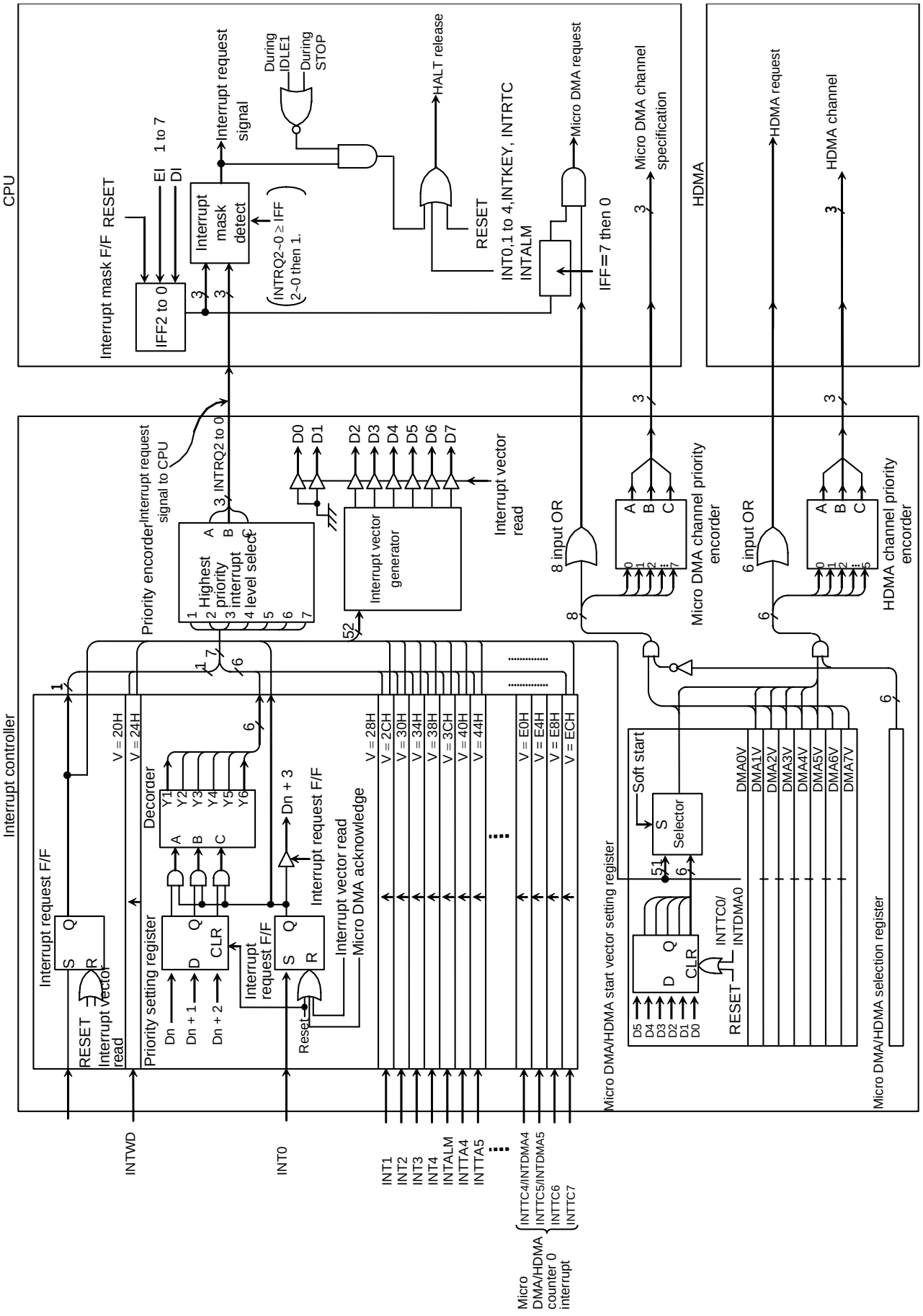
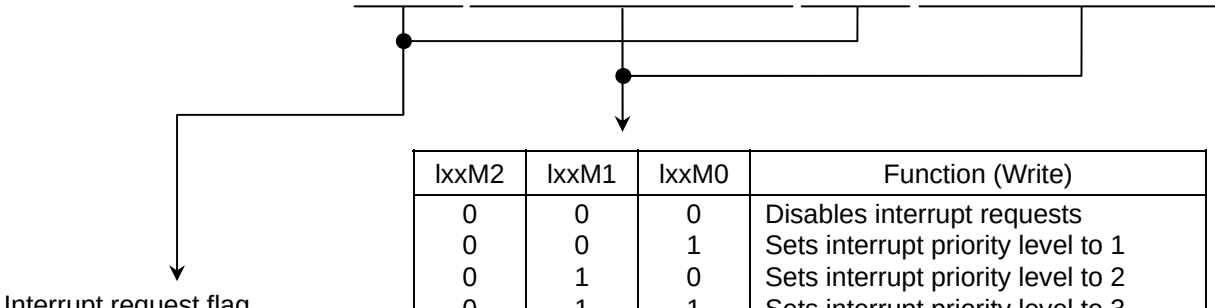


Figure 3.5.3 Block Diagram of Interrupt Controller

(1) Interrupt priority setting registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0	INT0 enable	F0H	–				INT0			
			–	–	–	–	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			Always write “0”.				0	0	0	0
INTE12	INT1 & INT2 enable	D0H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	D1H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	D2H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE7	INT7 enable	D3H	–				INT7			
			–	–	–	–	I7C	I7M2	I7M1	I7M0
			R	R/W			R	R/W		
			Always write “0”.				0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	D4H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	D5H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA45	INTTA4 & INTTA5 enable	D6H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA67	INTTA6 & INTTA7 enable	D7H	INTTA7 (TMRA7)				INTTA6 (TMRA6)			
			ITA7C	ITA7M2	ITA7M1	ITA7M0	ITA6C	ITA6M2	ITA6M1	ITA6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

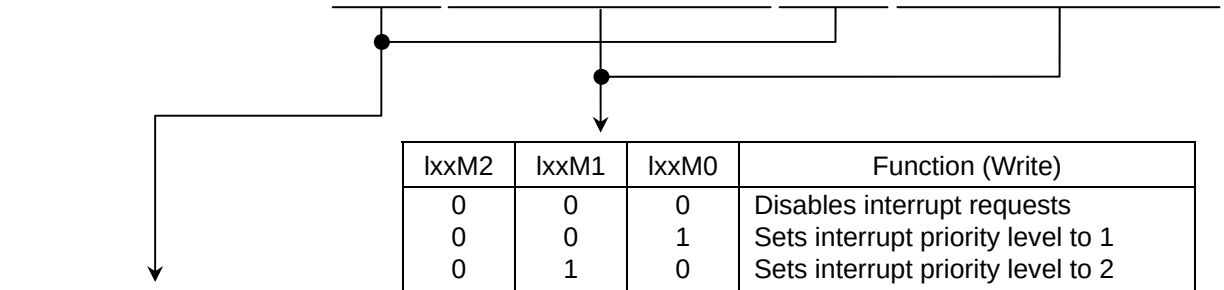


IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB0	INTTB00 & INTTB01 enable	D8H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB1	INTTB10 & INTTB11 enable	D9H	INTTB11 (TMRB1)				INTTB10 (TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES0	INTRX0 & INTTX0 enable	DBH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESBIADM	INTSBI & INTADM enable	E0H	INTADM				INTSBI			
			IADM0C	IADMM2	IADMM1	IADMM0	ISBI0C	ISBIM2	ISBIM1	ISBIM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESPI	INTSPI enable	E1H	INTSPITX				INTSPIRX			
			ISPITC	ISPITM2	ISPITM1	ISPITM0	ISPIRC	ISPIRM2	ISPIRM1	ISPIRM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEUSB	INTUSB enable	E3H	–				INTUSB			
			–	–	–	–	IUSBC	IUSBM2	IUSBM1	IUSBM0
							R	R/W		
			Always write “0”.				0	0	0	0
INTEALM	INTALM enable	E5H	–				INTALM			
			–	–	–	–	IALMC	IALMM2	IALMM1	IALMM0
							R	R/W		
			Always write “0”.				0	0	0	0
INTERTC	INTRTC enable	E8H	–				INTRTC			
			–	–	–	–	IRC	IRM2	IRM1	IRM0
							R	R/W		
			Always write “0”.				0	0	0	0
INTEKEY	INTKEY enable	E9H	–				INTKEY			
			–	–	–	–	IKC	IKM2	IKM1	IKM0
							R	R/W		
			Always write “0”.				0	0	0	0

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTELCD	INTLCD enable	EAH	–				INTLCD			
			–	–	–	–	ILCD1C	ILCDM2	ILCDM1	ILCDM0
			R				R	R/W		
			Always write “0”.				0	0	0	0
INTEI2S01	INTI2S0 & INTI2S1 enable	EBH	INTI2S1				INTI2S0			
			II2S1C	II2S1M2	II2S1M1	II2S1M0	I I2S0C	II2S0M2	II2S0M1	II2S0M0
			R	R/W			R/W	R/W		
			0	0	0	0	0	0	0	0
INTENDFC	INTRSC & INTRDY enable	ECH	INTRSC				INTRDY			
			IRSCC	IRSCM2	IRSCM1	IRSCM0	IRDYC	IRDYM2	IRDYM1	IRDYM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 enable	EEH	–				INTP0			
			–	–	–	–	IP0C	IP0M2	IP0M1	IP0M0
			R	R/W			R	R/W		
			Always write “0”.				0	0	0	
0INTEAD	INTAD & INTADHP enable	EFH	INTADHP				INTAD			
			IADHPC	IADHPM2	IADHPM1	IADHPM0	IADC	IADM2	IADM1	IADM0
			R	R/W			R/W	R/W		
			0	0	0	0	0	0	0	0



Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETC01 /INTEDMA01	INTTC0/INTDMA0 & INTTC1/INTDMA1 enable	F1H	INTTC1/INTDMA1				INTTC0/INTDMA0			
			ITC1C /IDMA1C	ITC1M2 /IDMA1M2	ITC1M1 /IDMA1M1	ITC1M0 /IDMA1M0	ITC0C /IDMA0C	ITC0M2 /IDMA0M2	ITC0M1 /IDMA0M1	ITC0M0 /IDMA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23 /INTEDMA23	INTTC2/INTDMA2 & INTTC3/INTDMA3 enable	F2H	INTTC3/INTDMA3				INTTC2/INTDMA2			
			ITC3C /IDMA3C	ITC3M2 /IDMA3M2	ITC3M1 /IDMA3M1	ITC3M0 /IDMA3M0	ITC2C /IDMA2C	ITC2M2 /IDMA2M2	ITC2M1 /IDMA2M1	ITC2M0 /IDMA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC45 /INTEDMA45	INTTC4/INTDMA4 & INTTC5/INTDMA5 enable	F3H	INTTC5/INTDMA5				INTTC4/INTDMA4			
			ITC5C /IDMA5C	ITC5M2 /IDMA5M2	ITC5M1 /IDMA5M1	ITC5M0 /IDMA5M0	ITC4C /IDMA4C	ITC4M2 /IDMA4M2	ITC4M1 /IDMA4M1	ITC4M0 /IDMA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC67	INTTC6 & INTTC7 enable	F4H	INTTC7 (DMA7)				INTTC6 (DMA6)			
			ITC7C	ITC7M2	ITC7M1	ITC7M0	ITC6C	ITC6M2	ITC6M1	ITC6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTWDT	INTWD enable	F7H	–				INTWD			
			–	–	–	–	ITCWD	–	–	–
			R	R/W			R			
			Always write “0”.				0	–	–	–

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Interrupt request flag

(2) External interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC0	Interrupt input mode control 0	F6H (Prohibit RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	–
			W	W	W	W	W	W	R/W	R/W
			0	0	0	0	0	0	0	0
			INT5EDGE 0: Rising 1: Falling	INT4EDGE 0: Rising 1: Falling	INT3EDGE 0: Rising 1: Falling	INT2EDGE 0: Rising 1: Falling	INT1EDGE 0: Rising 1: Falling	INT0EDGE 0: Rising 1: Falling	INT0 0: Edge mode 1: Level mode	Always write “0”.
IIMC1	Interrupt input mode control 0	FAH (Prohibit RMW)							I7EDGE	I6EDGE
									W	W
									0	0
									INT7EDGE 0: Rising 1: Falling	INT6EDGE 0: Rising 1: Falling

Note 1: Disable INT0 request before changing INT0 pin mode from level sense to edge sense. (change <I0LE>from “1” to “0”)

```
DI
LD      (IIMC0), XXXXXX0-B      ; Switches from level to edge.
LD      (INTCLR), 0AH           ; Clears interrupt request flag.

NOP                             ; Wait EI execution
NOP
NOP
EI
```

Note 2: X: Don’t care, –: No change

Note 3: See electrical characteristics in section 4 for external interrupt input pulse width.

Note 4: In port setting, if 16 bit timer input is selected and capture control is executed, INT6 and INT7 don’t depend on IIMC1 register setting. INT6 and INT7 operate by setting TBnMOD<TBnCPM1:0>.

Settings of External Interrupt Pin Function

Interrupt	Pin Name	Mode	Setting Method
INT0	PC0	Rising edge	<I0LE> = 0,<I0EDGE> = 0
		Falling edge	<I0LE> = 0, <I0EDGE> = 1
		High level	<I0LE> = 1
INT1	PC1	Rising edge	<I1EDGE> = 0
		Falling edge	<I1EDGE> = 0
INT2	PC2	Rising edge	<I2EDGE> = 0
		Falling edge	<I2EDGE> = 1
INT3	PC3	Rising edge	<I3EDGE> = 0
		Falling edge	<I3EDGE> = 1
INT4	P96	Rising edge	<I4EDGE> = 0
		Falling edge	<I4EDGE> = 1
INT5	PP3	Rising edge	<I5EDGE> = 0
		Falling edge	<I5EDGE> = 1
INT6	PP4	Rising edge	<I6EDGE> = 0
		Falling edge	<I6EDGE> = 1
INT7	PP5	Rising edge	<I7EDGE> = 0
		Falling edge	<I7EDGE> = 1

(3) SIO receive interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
SIMC	SIO interrupt mode control	F5H (Prohibit RMW)	–	–						IR0LE
			W	W						W
			0	0						1
			Always write “0” (Note)	Always write “0”						0:INTRX0 edge mode 1:INTRX0 level mode

Note: When using the micro DMA transfer end interrupt, always write “1”.

INTRX0 edge enable

0	Edge detect INTRX0
1	“H” level INTRX0

(4) Interrupt request flag clear register

The interrupt request flag is cleared by writing the appropriate micro DMA /HDMA start vector, as given in Table 3.5.1 to the register INTCLR.

For example, to clear the interrupt flag INT0, perform the following register operation after execution of the DI instruction.

```
INTCLR ← 0AH        ; Clears interrupt request
                      flag INT0.
```

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTCLR	Interrupt clear control	F8H (Prohibit RMW)	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt vector							

(5) Micro DMA start vector registers

These registers assign micro DMA /HDMA processing to sets which source corresponds to DMA. The interrupt source whose micro DMA /HDMA start vector value matches the vector set in one of these registers is designated as the micro DMA /HDMA start source.

When the micro DMA transfer counter (DMACn) or HDMA transfer counter B (HDMACBn) value reaches “0”, the micro DMA /HDMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA /HDMA start vector register is cleared, and the micro DMA /HDMA start source for the channel is cleared. Therefore, in order for micro DMA /HDMA processing to continue, the micro DMA /HDMA start vector register must be set again during processing of the micro DMA /HDMA transfer end interrupt.

If the same vector is set in the micro DMA /HDMA start vector registers of more than one channel, the lowest numbered channel takes priority.

Accordingly, if the same vector is set in the micro DMA /HDMA start vector registers for two different channels, the interrupt generated on the lower-numbered channel is executed until micro DMA /HDMA transfer is complete. If the micro DMA /HDMA start vector for this channel has not been set in the channel’s micro DMA /HDMA start vector register again, micro DMA /HDMA transfer for the higher-numbered channel will be commenced. (This process is known as micro DMA /HDMA chaining.)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	100H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 start vector	101H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 start vector	102H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 start vector	103H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
DMA4V	DMA4 start vector	104H			DMA4V5	DMA4V4	DMA4V3	DMA4V2	DMA4V1	DMA4V0
					R/W					
					0	0	0	0	0	0
					DMA4 start vector					
DMA5V	DMA5 start vector	105H			DMA5V5	DMA5V4	DMA5V3	DMA5V2	DMA5V1	DMA5V0
					R/W					
					0	0	0	0	0	0
					DMA5 start vector					
DMA6V	DMA6 start vector	106H			DMA6V5	DMA6V4	DMA6V3	DMA6V2	DMA6V1	DMA6V0
					R/W					
					0	0	0	0	0	0
					DMA6 start vector					
DMA7V	DMA7 start vector	107H			DMA7V5	DMA7V4	DMA7V3	DMA7V2	DMA7V1	DMA7V0
					R/W					
					0	0	0	0	0	0
					DMA7 start vector					

(6) Micro DMA/HDMA select register

This register selectable that is started either Micro DMA or HDMA processing.

Micro DMA /HDMA start vector register (DMAnV) shared with both functions. When interrupt which match with vector value that is set to DMA/HDMA start vector register generated, use this register.

Symbol	NAME	Address	7	6	5	4	3	2	1	0
DMASEL	Micro DMA/HDMA select	10AH			DMASEL5	DMASEL4	DMASEL3	DMASEL2	DMASEL1	DMASEL0
					R/W					
					0	0	0	0	0	0
					0:Micro DMA5 1:HDMA5	0:Micro DMA4 1:HDMA4	0:Micro DMA3 1:HDMA3	0:Micro DMA2 1:HDMA2	0:Micro DMA1 1:HDMA1	0:Micro DMA0 1:HDMA0

(7) Specification of a micro DMA burst

Specifying the micro DMA burst function causes micro DMA transfer, once started, to continue until the value in the transfer counter register reaches “0”. Setting any of the bits in the register DMAB which correspond to a micro DMA channel (as shown below) to “1” specifies that any micro DMA transfer on that channel will be a burst transfer.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAB	DMA burst	108H	DBST7	DBST6	DBST5	DBST4	DBST3	DBST2	DBST1	DBST0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request on Burst mode							

(8) Notes

The instruction execution unit and the bus interface unit in this CPU operate independently. Therefore, if immediately before an interrupt is generated, the CPU fetches an instruction which clears the corresponding interrupt request flag, the CPU may execute this instruction in between accepting the interrupt and reading the interrupt vector. In this case, the CPU will read the default vector 0004H and jump to interrupt vector address FFFF04H.

To avoid this, an instruction which clears an interrupt request flag should always be preceded by a DI instruction. And in the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing and more than 3-instructions (e.g., "NOP" × 3 times). If placed EI instruction without waiting NOP instruction after execution of clearing instruction, interrupt will be enable before request flag is cleared.

In the case of changing the value of the interrupt mask register <IFF2:0> by execution of POP SR instruction, disable an interrupt by DI instruction before execution of POP SR instruction.

In addition, please note that the following two circuits are exceptional and demand special attention.

INT0 level mode	In level mode INT0 is not an edge-triggered interrupt. Hence, in level mode the interrupt request flip-flop for INT0 does not function. The peripheral interrupt request passes through the S input of the flip-flop and becomes the Q output. If the interrupt input mode is changed from edge mode to level mode, the interrupt request flag is cleared automatically.
	If the CPU enters the interrupt response sequence as a result of INT0 going from 0 to 1, INT0 must then be held at 1 until the interrupt response sequence has been completed. If INT0 is set to level mode so as to release a halt state, INT0 must be held at 1 from the time INT0 changes from 0 to 1 until the halt state is released. (Hence, it is necessary to ensure that input noise is not interpreted as a 0, causing INT0 to revert to 0 before the halt state has been released.)
	When the mode changes from level mode to edge mode, interrupt request flags which were set in level mode will not be cleared. Interrupt request flags must be cleared using the following sequence.
	
	DI
	LD (IIMC0), 00H ; Switches from level to edge.
	LD (INTCLR), 0AH ; Clears interrupt request flag.
	NOP ; Wait EI execution
	NOP
	NOP
	EI

INTRX	In level mode (The register SIMC<IRxLE> set to "1"), the interrupt request flip-flop can only be cleared by a reset or by reading the serial channel receive buffer. It cannot be cleared by an instruction.

Note: The following instructions or pin input state changes are equivalent to instructions which clear the interrupt request flag.

INT0: Instructions which switch to level mode after an interrupt request has been generated in edge mode.
The pin input changes from high to low after an interrupt request has been generated in level mode. ("H" → "L")

INTRX: Instructions which read the receive buffer.

3.6 DMAC (DMA Controller)

The TMP92CZ26A incorporates a DMA controller (DMAC) having six channels. This DMAC can realize data transfer faster than the micro DMA function by the 900/H1 CPU.

The DMAC has the following features:

1) Six independent channels of DMA
2) Two types of transfer start requests

   Hardware request (using an interrupt source connected with the INTC) or software request can be selected for each channel.
3) Various source/destination combinations

   The combination of transfer source and destination can be selected for each channel from the following four types: memory to memory, memory to I/O, I/O to memory, I/O to I/O.
4) Transfer address mode

   Only the dual address mode is supported.
5) Dual-count mechanism and DMA end interrupt

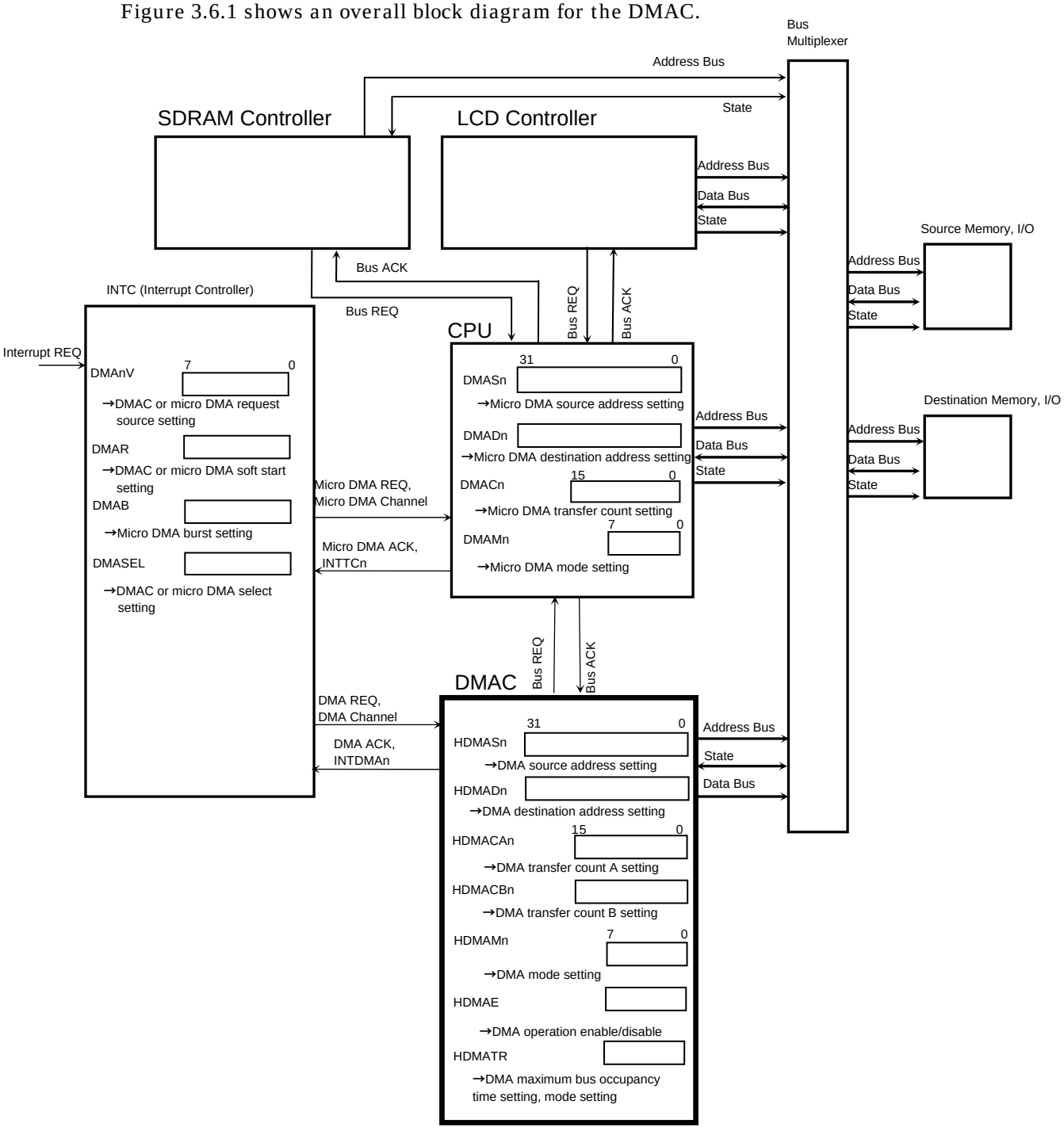
   Two count registers are provided to execute multiple DMA transfers by one DMA request and to generate multiple DMA requests at a time. The DMA end interrupt (INTDMA0 to INTDMA5) is also provided so that a general-purpose interrupt routine can be used to prepare for the next processing.
6) Priorities among DMA channels (the same as the micro DMA acceptance specifications of the INTC)

   DMA requests are basically accepted in the order in which they are asserted. If more than one request is asserted simultaneously or it looks as if two requests were asserted simultaneously because one of the requests has been put on hold while other processing was being performed, the smaller-numbered channel is given a higher priority.
7) DMAC bus occupancy limiting function

   The DMAC incorporates a special timer for limiting its bus occupancy time to avoid excessive interference with the CPU or LCDC operation.
8) The DMAC can be used in HALT (IDLE2) mode.

3.6.1 Block Diagram

Figure 3.6.1 shows an overall block diagram for the DMAC.



Note: “n” denotes a channel number. Micro DMA has eight channels (0 to 7) and DMA has six channels (0 to 5).

Figure 3.6.1 Overall Block Diagram

3.6.2 SFRs

The DMAC has the following SFRs. These registers are connected to the CPU via a 16-bit data bus.

(1) HDMASn (DMA Transfer Source Address Setting Register)

The HDMASn register is used to set the DMA transfer source address. When the source address is updated by DMA execution, HDMASn is also updated.

HDMAS0 to HDMAS5 have the same configuration.

Although the bus sizing function is supported, the address alignment function is not supported. Therefore, specify an even-numbered address for transferring 2 bytes and an address that is an integral multiple of 4 for transferring 4 bytes.

HDMASn Register

HDMASn

	7	6	5	4	3	2	1	0
bit Symbol	DnSA7	DnSA6	DnSA5	DnSA4	DnSA3	DnSA2	DnSA1	DnSA0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Source address [7:0] for DMAn							
	15	14	13	12	11	10	9	8
bit Symbol	DnSA15	DnSA14	DnSA13	DnSA12	DnSA11	DnSA10	DnSA9	DnSA8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Source address [15:8] for DMAn							
	23	22	21	20	19	18	17	16
bit Symbol	DnSA23	DnSA22	DnSA21	DnSA20	DnSA19	DnSA18	DnSA17	DnSA16
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Source address [23:16] for DMAn							

	Source address [23:16]	Source address [15:8]	Source address [7:0]
Channel 0	(0902H)	(0901H)	HDMAS0 (0900H)
Channel 1	(0912H)	(0911H)	HDMAS1 (0910H)
Channel 2	(0922H)	(0921H)	HDMAS2 (0920H)
Channel 3	(0932H)	(0931H)	HDMAS3 (0930H)
Channel 4	(0942H)	(0941H)	HDMAS4 (0940H)
Channel 5	(0952H)	(0951H)	HDMAS5 (0950H)

Note: Read-modify-write instructions can be used on all these registers.

Figure 3.6.2 HDMASn Register

(2) HDMADn (DMA Transfer Destination Address Setting Register)

The HDMADn register is used to set the DMA transfer destination address. When the destination address is updated by DMA execution, HDMADn is also updated.

HDMAD0 to HDMAD5 have the same configuration.

Although the bus sizing function is supported, the address alignment function is not supported. Therefore, specify an even-numbered address for transferring 2 bytes and an address that is an integral multiple of 4 for transferring 4 bytes.

HDMADn Register

HDMADn

	7	6	5	4	3	2	1	0
bit Symbol	DnDA7	DnDA6	DnDA5	DnDA4	DnDA3	DnDA2	DnDA1	DnDA0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Destination address [7:0] for DMAn							
	15	14	13	12	11	10	9	8
bit Symbol	DnDA15	DnDA14	DnDA13	DnDA12	DnDA11	DnDA10	DnDA9	DnDA8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Destination address [15:8] for DMAn							
	23	22	21	20	19	18	17	16
bit Symbol	DnDA23	DnDA22	DnDA21	DnDA20	DnDA19	DnDA18	DnDA17	DnDA16
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Destination address [23:16] for DMAn							

	Destination address [23: 16]	Destination address [15: 8]	Destination address [7: 0]
Channel 0	(0906H)	(0905H)	HDMAD0 (0904H)
Channel 1	(0916H)	(0915H)	HDMAD1 (0914H)
Channel 2	(0926H)	(0925H)	HDMAD2 (0924H)
Channel 3	(0936H)	(0935H)	HDMAD3 (0934H)
Channel 4	(0946H)	(0945H)	HDMAD4 (0944H)
Channel 5	(0956H)	(0955H)	HDMAD5 (0954H)

Note: Read-modify-write instructions can be used on all these registers.

Figure 3.6.3 HDMADn Register

(3) HDMACAn (DMA Transfer Count A Setting Register)

The HDMACAn register is used to set the number of times a DMA transfer is to be performed by one DMA request. HDMACAn contains 16 bits and can specify up to 65536 transfers (0001H = one transfer, FFFFH = 65535 transfers, 0000H = 65536 transfers). Even when the transfer count A is updated by DMA execution, HDMACAn is not updated.

HDMACA0 to HDMACA5 have the same configuration.

HDMACAn Register

HDMACAn	7	6	5	4	3	2	1	0
bit Symbol	DnCA7	DnCA6	DnCA5	DnCA4	DnCA3	DnCA2	DnCA1	DnCA0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transfer count A [7:0] for DMAn							
	15	14	13	12	11	10	9	8
bit Symbol	DnCA15	DnCA14	DnCA13	DnCA12	DnCA11	DnCA10	DnCA9	DnCA8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transfer count A [15:8] for DMAn							

	Transfer count A [15: 8]	Transfer count A [7: 0]
Channel 0	(0909H)	HDMACA0 (0908H)
Channel 1	(0919H)	HDMACA1 (0918H)
Channel 2	(0929H)	HDMACA2 (0928H)
Channel 3	(0939H)	HDMACA3 (0938H)
Channel 4	(0949H)	HDMACA4 (0948H)
Channel 5	(0959H)	HDMACA5 (0958H)

Note: Read-modify-write instructions can be used on all these registers.

Figure 3.6.4 HDMACAn Register

(4) HDMACBn (DMA Transfer Count B Setting Register)

The HDMACBn register is used to set the number of times a DMA request is to be made. HDMACBn contains 16 bits and can specify up to 65536 requests (0001H = one request, FFFFH = 65535 requests, 0000H = 65536 requests). When the transfer count B is updated by DMA execution, HDMACBn is also updated.

HDMACB0 to HDMACB5 have the same configuration.

HDMACBn Register

HDMACBn	7	6	5	4	3	2	1	0
bit Symbol	DnCB7	DnCB6	DnCB5	DnCB4	DnCB3	DnCB2	DnCB1	DnCB0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transfer count B [7:0] for DMAn							
	15	14	13	12	11	10	9	8
bit Symbol	DnCB15	DnCB14	DnCB13	DnCB12	DnCB11	DnCB10	DnCB9	DnCB8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transfer count B [15:8] for DMAn							

	Transfer count B [15: 8]	Transfer count B [7: 0]
Channel 0	(090BH)	HDMACB0 (090AH)
Channel 1	(091BH)	HDMACB1 (091AH)
Channel 2	(092BH)	HDMACB2 (092AH)
Channel 3	(093BH)	HDMACB3 (093AH)
Channel 4	(094BH)	HDMACB4 (094AH)
Channel 5	(095BH)	HDMACB5 (095AH)

Note: Read-modify-write instructions can be used on all these registers.

Figure 3.6.5 HDMACBn Register

(5) HDMAMn (DMA Transfer Mode Setting Register)

The HDMAMn register is used to set the DMA transfer mode.

HDMAM0 to HDMAM5 have the same configuration.

HDMAMn Register

HDMAMn	7	6	5	4	3	2	1	0
bit Symbol				DnM4	DnM3	DnM2	DnM1	DnM0
Read/Write				R/W				
After reset				0	0	0	0	0
Function				DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved (Note 2)			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

	Transfer mode [7: 0]
Channel 0	HDMAM0 (090CH)
Channel 1	HDMAM1 (091CH)
Channel 2	HDMAM2 (092CH)
Channel 3	HDMAM3 (093CH)
Channel 4	HDMAM4 (094CH)
Channel 5	HDMAM5 (095CH)

Note 1: Read-modify-write instructions can be used on all these registers.

Note 2: INC: Post-increment

Dec: Post-decrement

I/O: Fixed memory address

MEM: Memory address to be incremented or decremented

Figure 3.6.6 HDMAMn Register

(6) HDMAE (DMA Operation Enable Register)

The HDMAE register is used to enable or disable the DMAC operation.

Bits 0 to 5 correspond to channels 0 to 5. Unused channels should be set to “0”.

HDMAE Register

HDMAE (097EH)

	7	6	5	4	3	2	1	0
bit Symbol			DMAE5	DMAE4	DMAE3	DMAE2	DMAE1	DMAE0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			DMA channel operation 0: Disable 1: Enable					

Note: Read-modify-write instructions can be used on this register.

Figure 3.6.7 HDMAE Register

(7) HDMATR (DMA Maximum Bus Occupancy Time Setting Register)

The HDMATR register is used to set the maximum duration of time the DMAC can occupy the bus. The TMP92CZ26A does not have priority levels for bus arbitration. Therefore, once the DMAC owns the bus, other masters (such as the LCDC) must wait until the DMAC completes its transfer operation and releases the bus. This could lead to problems in the system. For example, if the LCDC cannot own the bus as required, the LCD display function may not work properly. To avoid such a situation, the DMAC limits the duration of its bus occupancy by using this timer register. When the DMAC occupies the bus for the duration of time set in this register, it releases the bus even if the specified DMA operation has not been completed yet. After waiting for 16 states, the DMAC asserts a bus request again to execute the rest of the DMA operation.

The DMAC counts the bus occupancy time regardless of which channel is occupying the bus. To set the maximum bus occupancy time, ensure that the HDMAE register is set to “00H” and set HDMATR<DMATE> to “1” and <DMATR6:0> to the desired value.

Note: In case of using S/W start with HDMA, transmission start is to set to "1" DMAR register. However DMAR register can't be used to confirm flag of transmission end. DMAR register reset to "0" when HDMA release bus occupation once with HDMATR function.

HDMATR Register

HDMATR (097FH)

	7	6	5	4	3	2	1	0
bit Symbol	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Timer operation 0: Disable 1: Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by “maximum bus occupancy time / (256/f_{SYS})”. “00H” cannot be set.						

Note: Read-modify-write instructions can be used on this register.

Figure 3.6.8 HDMATR Register

3.6.3 DMAC Operation Description

(1) Overall flowchart

Figure 3.6.9 shows a flowchart for DMAC operation when an interrupt (DMA) is requested.

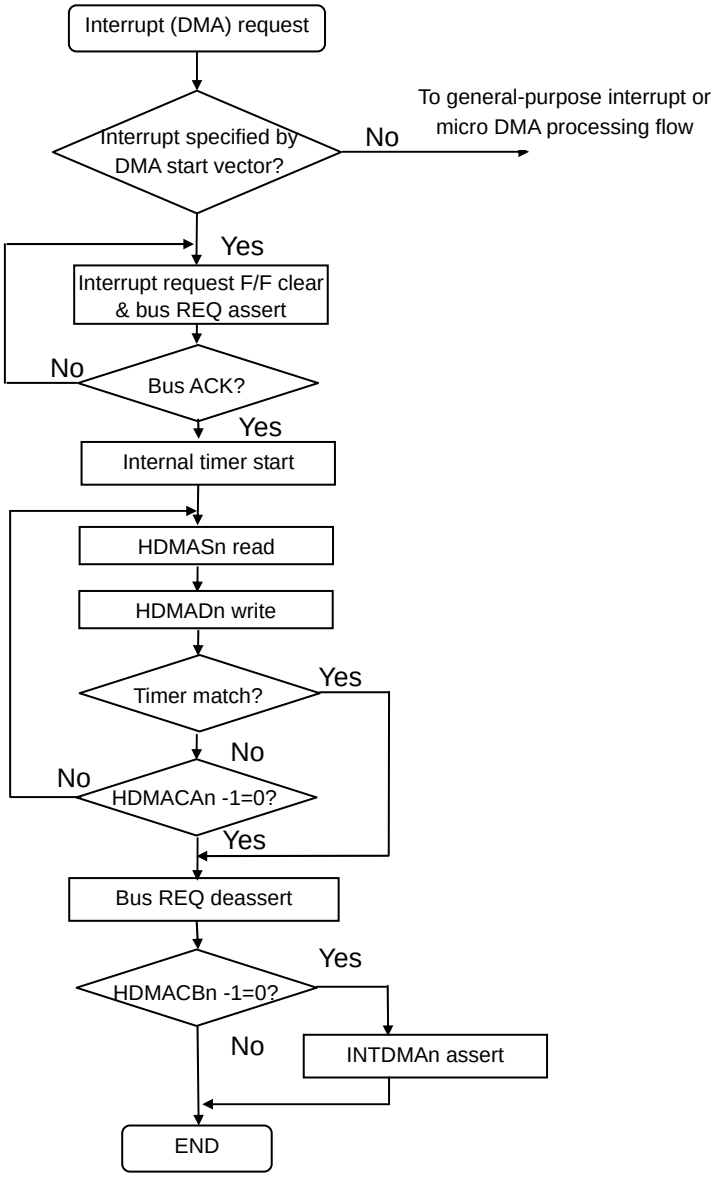


Figure 3.6.9 Overall Flowchart

(2) Bus arbitration

The TMP92CZ26A includes three controllers (DMA controller, LCD controller, SDRAM controller) that function as bus masters apart from the CPU. These controllers operate independently and assert a bus request as required. The controller that receives a bus acknowledgement acts as the bus master. No priorities are assigned to these three controllers, and bus requests are processed in the order in which they are asserted. Once one of the controllers owns the bus, bus requests from other controllers are put on hold until the bus is released again. While one of the controllers is occupying the bus, CPU processing including non-maskable interrupt requests is also put on hold.

(3) Transfer source and destination memory setting

Either internal or external memory can be set as the source and destination memory or I/O to be accessed by the DMAC. Even when the MMU is used in external memory, the addresses to be accessed by the DMAC should be specified using logical addresses. The DMAC accesses the specified source and destination addresses according to the bus width and number of waits set in the memory controller and the bank settings made in the MMU.

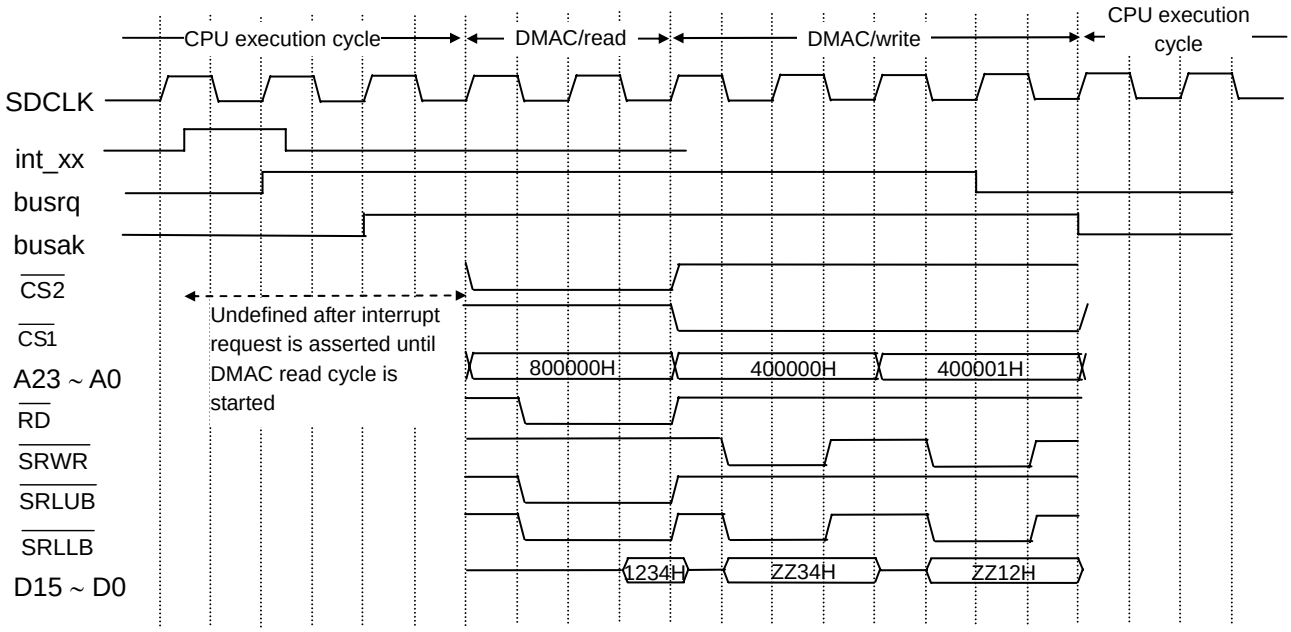
Although the bus sizing function is supported, the address alignment function is not supported. Therefore, specify an even-numbered address for transferring 2 bytes and an address that is an integral multiple of 4 for transferring 4 bytes.

Table 3.6.1 Difference point of address setting between HDMA and micro DMA

	Data Length	HDMA	Micro DMA
Source address	1byte	No restriction	No restriction
	2byte	Even address	
	4byte	Address in multiples of 4	
Destination address	1byte	No restriction	
	2byte	Even address	
	4byte	Address in multiples of 4	

(4) Operation timing

The following diagram shows an example of operation timing for transferring 2 bytes from 16-bit memory connected with the $\overline{CS2}$ area to 8-bit memory connected with the $\overline{CS1}$ area.



3.6.4 Setting Example

This section explains how to set the DMAC using an example.

(1) Transferring music data from internal RAM to I2S by DMA transfer

The 32 Kbytes of data stored in the internal RAM at addresses 2000H to 9FFFH shall be transferred to FIFO-RAM via I2S. Each time an INTI2S request is asserted, 64 bytes (4 bytes x 16 times) shall be transferred to FIFO-RAM using DMAC channel 0. Since INTI2S is an FIFO empty interrupt, the first data must be set in advance. Therefore, only the first 64 bytes shall be transferred by DMA soft start. After 32 Kbytes have been transferred, the INTDMA0 interrupt routine shall be activated to prepare for the next processing.

(a) Main routine

No	Instruction	Comments
1	ldl (hdmas0),2000H	; Source address = 2000H
2	ldl (hdmad0),i2sbuf	; Destination address = i2sbuf
3	ldw (hdmaca0),16	; Counter A = 16
4	ldw (hdmacb0),512	; Counter B = 512 (32768/64)
5	ldb (hdmam0),0AH	; Transfer mode = source INC, 4 bytes
6	set 0,(hdmae)	; Enable DMA channel 0.
7	ld (dmar),01H	; Transfer the first 64 bytes by DMA soft start.
8	nop	;
9	ld (dma0v),i2s_vector	; INTI2S = DMA0
10	ld (intedma01),xxH	; INTDMA level = x
11	ldw (i2sctl0),xxxxH	; Set operation mode for I2S.
12	ldw (i2sctl1),xxxxH	; Start I2S transmission.
13	ei xx	; Enable CPU interrupts.

(b) INTDMA0 interrupt routine

No	Instruction	Comments
1	res 0,(hdmae)	; Disable DMA channel 0.
2	:	
3	:	
4	:	
5	:	
6		
7		
8		
9		
10		
11	reti	;

3.6.5 Note

1. In case of using S/W start with HDMA, transmission start is to set to "1" DMAR register. However DMAR register can't be used to confirm flag of transmission end. DMAR register reset to "0" when HDMA release bus occupation once with HDMATR function. We recommend to use HDMACBn register (counter value) to confirm flag of transmission end.

3.6.6 Considerations for Using More Than One Bus Master

In the TMP92CZ26A, the LCD controller, SDRAM controller, and DMA controller may act as the bus master apart from the CPU. Therefore, care must be exercised to enable each of these functions to operate smoothly.

To facilitate explanation of DMA operation performed by each bus master, the DMA transfer operation performed by the DMA controller is defined as “HDMA”, the display RAM read operation performed by the LCD controller as “LDMA”, and the SDRAM auto refresh operation performed by the SDRAM controller as “ARDMA”.

The following explains various cases where two or more bus masters may operate at the same time.

(1) CPU + HDMA

The DMA controller performs DMA transfer (HDMA) after issuing a bus request to the CPU and getting a bus acknowledgement. The DMA controller may be active while the CPU is in HALT mode (IDLE2 mode only), in which case HDMA does not interfere with the CPU operation. However, if HDMA is started while the CPU is active, the CPU cannot execute instructions while HDMA is being performed.

Before activating the DMA controller, therefore, it is necessary to estimate the CPU stop time (defined as “t_{STOP} (HDMA)”) based on the transfer time, transfer start interval, and number of channels to be used.

$$\text{CPU bus stop rate} = t_{STOP}\ (\text{HDMA})[s] / \text{HDMA start interval}[s]$$

$$\text{HDMA start interval}[s] = \text{HDMA start interrupt period}[s]$$

Note: The HDMA start interval depends on the period of the HDMA start interrupt source. However, it is also possible to start HDMA by software.

$$t_{STOP}\ (\text{HDMA})[s] = (\text{Source read time} + \text{Destination write time}) \times \text{Transfer count} + \alpha$$

state/byte

Memory Type / Read / Write	Internal RAM	External SDRAM 16-bit bus	External SRAM 16-bit bus	External SRAM 8-bit bus
Read	1 / 4 (Note 1)	Burst 1 / 2 (Note 2) 1 word 6 / 2 (Note 2)	2 / 2 (Note 3)	2 / 1 (Note 3)
Write	1 / 4	Burst 1 / 2 (Note 2) 1 word 3 / 2 (Note 2)	2 / 2 (Note 3)	2 / 1 (Note 3)

Note 1: 2-1-1-1 access. Each consecutive address can be accessed in 1 state.

Note 2: The transfer speed varies depending on the combination of source and destination.

a) When the source or destination is internal RAM or internal I/O (SFR), burst access (6-1-1-1 access) is possible. Only consecutive addresses on the same page can be accessed in 1 state. Additional 4 states are needed at the end of each burst access.

b) When the source or destination is other than internal RAM or internal I/O, 1-word access is used.

Note 3: In the case of 0 waits

state/byte

I/O Type / Read / Write	I2S	NANDF	USB	SPI
Read	–	2 / 2	2 / 2	2 / 4
Write	2 / 4	2 / 2	2 / 2	2 / 4

Sample 1) Calculation example for CPU + HDMA

Conditions:

CPU operation speed (f_{SYS}) : 60 MHz

I2S sampling frequency : 48 KHz (60 MHz/25/50 = 48 KHz)

I2S data transfer bit length : 16 bits

DMAC channel 0 used to transfer 5 Kbytes from internal RAM to I2S

Calculation example:

DMAC source data read time:

Internal RAM data read time = 1 state/4 bytes (However, the first 1 byte requires 2 states.)

DMAC destination write time:

I2S register write time = 2 states/4 bytes

Transfer count

To transfer 5 Kbytes of data in 4-byte units, the transfer count is calculated as follows:

5 Kbytes/4 bytes = 1280 [times]

Since I2S generates an interrupt for every 64 bytes, the DMAC's counter A is set to 16 (64 bytes/4 bytes = 16 times) and counter B is set to 80.

* Since an interrupt is generated 80 times, the first read to internal RAM (which requires 1 additional state) occurs 80 times, requiring additional 80 states in total. In addition, from bus REQ to bus ACK, an overhead time of 2 states is also needed for each interrupt request, requiring additional 160 states in total.

t_{STOP} (HDMA) = (((1 + 2) × 16) × 80) + 80 + 160) / fSYS [S] = 68 [μS]

HDMA start interval [s] = 1 / I2S sampling frequency [Hz] × (64 / 16)

= 83.33 [mS]

CPU bus stop rate = t_{STOP} (HDMA) [s] / HDMA start interval [s]

= 68 [μS] / 83.33 [mS] = 0.08 [%]

(2) CPU + LDMA

The LCD controller performs DMA transfer (LDMA) after issuing a bus request to the CPU and getting a bus acknowledgement.

If LDMA is not performed properly, the LCD display function cannot work properly. Therefore, LDMA must have higher priority than the CPU. While LDMA is being performed, the CPU cannot execute instructions.

To display data on the LCD using the LCD controller, it is necessary to estimate to what degree LDMA would interfere with the CPU operation based on the display RAM type, display RAM bus width, LCDD type, display pixel count, and display quality.

The time the CPU stops operation while the LCD controller transfers data for one line is defined as "t_{STOP} (LDMA)", which is calculated as shown below for each display mode.

$$t_{STOP}\text{ (LDMA)} = (\text{SegNum} \times K / 8) \times t_{LRD}$$

16-bit external SRAM	: t_{LRD} = (2 + wait count) / f_{SYS} [Hz] / 2
Internal RAM	: t_{LRD} = 1 / f_{SYS} [Hz] / 4
16-bit external SDRAM	: t_{LRD} = 1 / f_{SYS} [Hz] / 2
SegNum	: Number of segments to be displayed
K	: Number of bits needed for displaying 1 pixel

Monochrome	K = 1
4 gray scales	K = 2
16 gray scales	K = 4
256 colors	K = 8
4096 colors	K = 12
65536 colors	K = 16
262144/16777216 colors	K = 24

Note 1: When SDRAM is used, the overhead time is added as shown below.

$$t_{STOP}\text{ [s]} = (\text{SegNum} \times K/8) \times t_{LRD} + ((1/f_{SYS}) \times 8)$$

Note 2: When internal RAM is used, the overhead time is added as shown below.

$$t_{STOP}\text{ [s]} = (\text{SegNum} \times K/8) \times t_{LRD} + (1/f_{SYS})$$

The CPU bus stop rate indicates what proportion of the 1-line data update time t_{LP} is taken up by t_{STOP}(LDMA) and is calculated as follows:

$$\text{CPU bus stop rate} = t_{STOP}\text{ (LDMA) [s] / LHSYNC [period: s]}$$

Sample2) Calculation examples for CPU + LDMA

Conditions 1:

CPU operation speed (f_{SYS})	: 60 MHz
Display RAM	: Internal RAM
Display size	: QVGA (320seg × 240com)
Display quality	: 65536 colors (TFT)
Refresh rate	: 70 Hz (including 20 clocks of dummy cycles)

Calculation example 1:

t_{STOP} (LDMA) = ((SegNum × K / 8) × t_{LRD}) + (1 / f_{SYS} [Hz])

= ((320 × 16 / 8) × 1 / f_{SYS} [Hz] / 4) + (1 / f_{SYS} [Hz])

= ((640) × 16.67 [ns] / 4) + 16.67 [ns]

= 2.68 [μs]

LHSYNC [period: s] = 1/70 [Hz] /(COM+20=260) = 54.95 [μs]

CPU bus stop rate = t_{STOP} (LCD)[s] / LHSYNC [period: s]

= 2.68 [μs] / 54.95 [μs] = 4.88 [%]

Conditions 2:

CPU operation speed (f_{SYS})	: 10 MHz
Display RAM	: 16-bit external SRAM (0 waits)
Display size	: QVGA (240seg × 320com)
Display quality	: 4096 colors (STN)
Refresh rate	: 100 Hz (0 dummy cycles)

Calculation example 2:

t_{STOP} (LDMA) = (SegNum × K / 8) × tLRD

= (240 × 12 / 8) × (2 + wait count) / f_{SYS} [Hz] / 2

= (360) × 200 [ns] / 2

= 36 [μs]

LHSYNC [period: s] = 1/100 [Hz] / (COM = 240) = 41.67 [μs]

CPU bus stop rate = t_{STOP} (LCD)[s] / LHSYNC [period: s]

= 36 [μs] / 41.67 [μs] = 86.40 [%]

(3) CPU + LDMA + ARDMA

The SDRAM controller owns the bus not only when SDRAM is used as the LCD display RAM but also when SDRAM is used as work, data, or stack area. The SDRAM controller occupies the bus (ARDMA) while it refreshes SDRAM data by the auto refresh function.

No special consideration is needed for the ARDMA time normally as it ends within several clocks per specified number of states. However, if the LCD controller occupies the bus continuously, ARDMA cannot be executed at normal intervals and refresh data is stored in a counter specifically provided in the SDRAM controller. In this case, ARDMA is executed successively after the LCD controller releases the bus.

The priorities among the three bus masters should be set in the order of LCDC > SDRAMC > CPU. The time the CPU stops operation while the LCD controller and SDRAM controller are transferring data for one line is defined as "t_{STOP} (LDMA・ARDMA)", which is calculated as follows:

$$t_{STOP}\text{ (LDMA・ARDMA)} = t_{STOP}\text{ (LDMA)[s]} - (t_{STOP}\text{ (LDMA)[s]} / \text{AR interval [s]} \times 2 / f_{SYS}\text{ [Hz]})$$

$$\text{CPU bus stop rate} = t_{STOP}\text{ (LDMA・ARDMA)[s]} / \text{LHSYNC [period: s]}$$

Auto Refresh Intervals

SDRCR<SRS2: 0>			Auto Refresh Interval (states)	Frequency (System Clock)					
SRS2	SRS1	SRS0		6 MHz	10MHz	20MHz	40MHz	60MHz	80MHz
0	0	0	47	7.8	4.7	2.4	1.18	0.78	0.59
0	0	1	78	13.0	7.8	3.9	1.95	1.30	0.98
0	1	0	156	26.0	15.6	7.8	3.90	2.60	1.95
0	1	1	312	52.0	31.2	15.6	7.80	5.20	3.90
1	0	0	468	78.0	46.8	23.4	11.70	7.80	5.85
1	0	1	624	104.0	62.4	31.2	15.60	10.40	7.80
1	1	0	936	156.0	93.6	46.8	23.40	15.60	11.70
1	1	1	1248	208.0	124.8	62.4	31.20	20.80	15.60

Unit: [μs]

Sample3) Calculation example for CPU + LDMA + ARDMA

Conditions:

CPU operating speed(f_{SYS})	: 60 MHz
Display RAM	: 16-bit external SDRAM
Display size	: QVGA (320seg × 240com)
Display quality	: 65536 colors (TFT)
Refresh rate	: 70 Hz (including 20 clocks of dummy cycles)
SDRAM auto refresh	: Every 936 states (15.6 μs)

Calculation example:

t_{STOP} (LDMA) $=((SegNum \times K / 8) \times t_{LRD}) + (8 / f_{SYS}\ [Hz])$

$= ((320 \times 16 / 8) \times 1 / f_{SYS}\ [Hz] / 2) + (8 / f_{SYS}\ [Hz])$

$= ((640) \times 16.67\ [ns] / 2) + 133.33\ [ns]$

$= 5.47\ [\mu s]$

LHSYNC [period:s] $= 1/70\ [Hz] / (COM + 20 = 260) = 54.95\ [\mu s]$

Since SDRAM is auto-refreshed once or less in 5.47 [μs]:

t_{STOP} (ARDMA) $= 2 / f_{SYS}\ [Hz] = 33.33\ [ns]$

CPU bus stop rate $= t_{STOP}$(LDMA・ARDMA) [s] / LHSYNC [period:s]

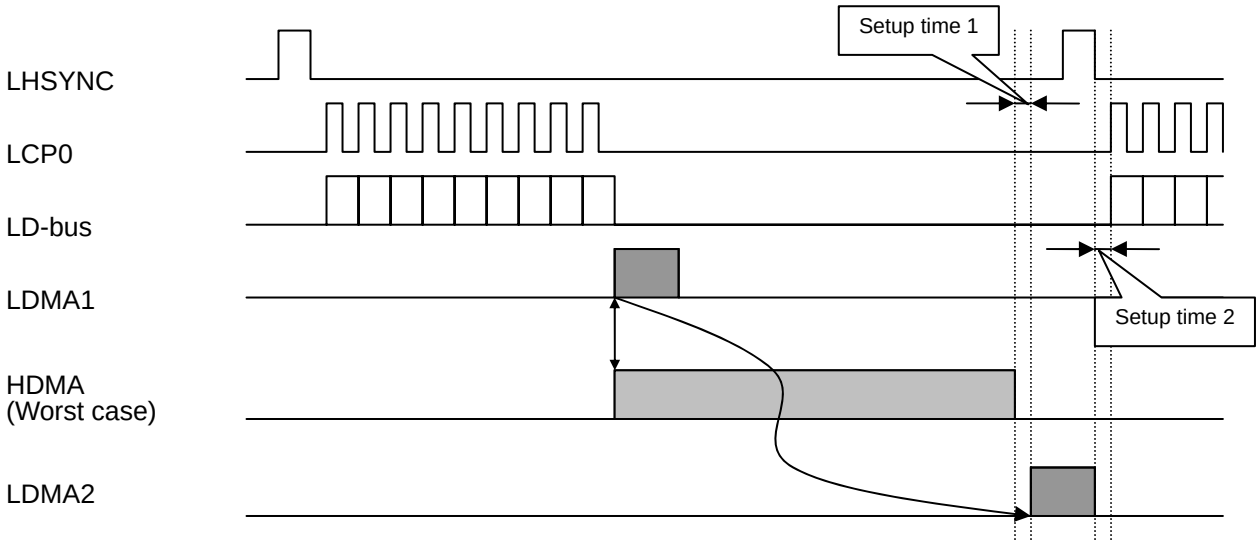
$= (5.47\ [\mu s] + 33.33\ [ns]) / 54.95\ [\mu s] = 10.01\ [\%]$

(4) CPU + LDMA+ ARDMA + HDMA

This is a case in which all the bus masters are active at the same time.

Since the LCD display function cannot work properly if the LCD controller cannot perform LDMA properly, the priorities among the four bus masters should be set in the order of LDMA > ARDMA > HDMA > CPU.

Before calculating the CPU bus stop rate, the conditions for proper LCD display shall be considered first.



The above diagram shows the LHSYNC signal, LCP0 signal, and LD-bus signal for transferring data from the LCD controller to the LCD driver, and the transfer operation (LDMA1) for reading data from the display RAM into the FIFO buffer in the LCD controller.

LDMA is started immediately after data has been transferred to the LCD driver. If HDMA is started immediately before LDMA1 is started, LDMA must wait until HDMA has finished before it can be started (LDMA2). LDMA2 must finish operation before the LCD driver output for the next stage is started.

LHSYNC [period: s] – LCD driver data transfer time [s] – t_{STOP}(LCD) [s]

= HDMA continuous time [s] + CPU operation time [s]

In the case of STN display

LCD driver data transfer time [s] = SegNum/8 × (1/f_{SYS}) × (LD bus transfer speed)

In the case of TFT display

LCD driver data transfer time [s] = SegNum × (1/ f_{SYS}) × (LD bus transfer speed)

Sample 4) Calculation example for CPU + LDMA+ ARDMA + HDMA

Conditions:

CPU operation speed (f_{SYS})	: 60 MHz
Display RAM	: QVGA (320seg × 240com)
Display quality	: 65536 colors (TFT)
Refresh rate	: 70 Hz (including 20 clocks of dummy cycles)
SDRAM Auto Refresh	: Every 936 states (15.6 μs)
SDRAM	: 16-bit width
HDMA	: Transfers 5 Kbytes from internal RAM to I2S

Calculation example:

t_{STOP} (LDMA) $= ((SegNum \times K / 8) \times tLRD) + (1 / f_{SYS}\ [Hz])$

$= ((320 \times 16 / 8) \times 1 / f_{SYS}\ [Hz] / 4) + (1 / f_{SYS}\ [Hz])$

$= ((640) \times 16.67\ [ns] / 4) + 16.67\ [ns]$

$= 2.68\ [\mu s]$

LHSYNC [period: s] $= 1/70\ [Hz] / (COM+20 = 260) = 54.95\ [\mu s]$

t_{STOP} (HDMA) $= (((1 + 2) \times 16) \times 80) + 80 + 160) / f_{SYS}\ [s] = 68\ [\mu s]$

LCD driver data transfer time [s]

$= SegNum \times (1/f_{SYS}) \times$ (LD bus transfer speed)

$= 320 \times (1/60\ MHz) \times 16 = 85\ [\mu s]$

Since LHSYNC [period: s] < LCD driver data transfer time [s], this setting is not possible.

When the transfer speed is changed to x4, the LCD driver data transfer time is calculated as follows:

(The transfer speed should be adjusted according to the required specifications.)

LCD driver data transfer time [s]

$= SegNum \times (1/f_{SYS}) \times$ (LD bus transfer speed)

$= 320 \times (1 / 60MHz) \times 4 = 21.3\ [\mu s]$

LHSYNC [period: s] − LCD driver data transfer time [s] − t_{STOP}(LDMA)

$= 54.95\ [\mu s] - 21.3\ [\mu s] - 2.68\ [\mu s] = 30.94\ [\mu s]$

To realize proper LCD display, the maximum time HDMA can occupy the bus at a time (maximum HDMA time) must be set to 30.92 [μS] or less. Although transferring all 5 Kbytes from the internal RAM to I2S requires t_{STOP} (HDMA) = 68 [μs], the maximum HDMA time should be limited by using the HDMATR register.

HDMATR Register

HDMATR (097FH)

	7	6	5	4	3	2	1	0
bit Symbol	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Timer operation 0: Disable 1: Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by "maximum bus occupancy time / (256/fSYS)". "00H" cannot be set.						

Note: Read-modify-write instructions can be used on this register.

By writing "87H" to the HDMATR register, the maximum HDMA time is set to 29.9 [μs] (256 × 7 × (1 / f_{SYS})). Since HDMA start interval [period:s] = 83.33 [ms] is longer than LHSYNC [period:s] = 54.95 [μs], it is assumed that HDMA transfer occurs once during LHSYNC [period:s].

Since SDRAM is auto-refreshed once or less in 5.47 [μs]:

t_{STOP} (ARDMA) = 2 / f_{SYS} [Hz] = 33.33 [ns]

The time LDMA, ARDMA, and HDMA all occupy the bus is defined as:

t_{STOP}(LDMA・ARDMA・HDMA)

Based on the above, the CPU bus stop rate is calculated as follows:

CPU bus stop rate = t_{STOP}(LDMA・ARDMA・HDMA) [s] / LHSYNC [period:s]

= (5.47 [μs] + 33.33 [ns]+29.9 [μs]) / 54.95 [μs] = 64.42 [%]

Note: To be precise, the bus assert time and RAM access time are added each time the HDMA transfer time is forcefully terminated at 29.9 [μs].

Sample 5) Calculation example when using CPU + LCDC + SDRAMC + HDMA at same time (Worst case)

Conditions:

CPU operation speed (f_{SYS}) : 80MHz

Display RAM : Internal RAM

Display size : QVGA (320seg × 240com)

Display quality : 16777216 color (TFT)

Refresh rate : 70Hz

HDMA : Transfers 225 Kbytes from internal RAM to SDRAM

Calculation example:

t_{STOP} (LCD) = ((SegNum × K/8) × t_{LRD}) + (1/f_{SYS} [Hz])

= ((320 × 24/8) × 1/f_{SYS} [Hz]/4) + (1/f_{SYS} [Hz])

= ((960) × 12.5 [nS]/4) + 12.5 [nS]

= 3.0125 [μS]

LHSYNC [period: S] = 1/70 [Hz]/ (COM+20) = 54.9 [μS]

t_{STOP} (HDMA) = (((2 + 1) × 4) × 57600) + 28800 + 14400)/f_{SYS} [S] = 9180 [μS]

LCD driver data transfer time [S]

= SegNum × (1/fsys) × (LD bus transfer speed)

= 320 × (1/80MHz) × 8 = 32 [μS]

LHSYNC [cycle S] - LCD driver data transfer time [S] – t_{STOP} (LCD)

= 54.9 [μS] – 32 [μS] – 3.0125 [μS] = 19.8875 [μS]

To realize proper LCD display, the maximum time HDMA can occupy the bus at a time (maximum HDMA time) must be set to 19.8875 [μS] or less. Although transferring all 225 Kbytes from the internal RAM to SDRAM requires t_{STOP} (HDMA) = 9180 [μs], the maximum HDMA time should be limited by using the HDMATR register.

HDMATR register

HDMATR (097FH)

	7	6	5	4	3	2	1	0
Bit Symbol	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Timer operation 0: Disable 1:Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by "Maximum bus occupancy time / (256/f_{SYS})". "00H" cannot be set.						

Note: Read-modify-write instructions can be used on this register.

By writing "86H" to the HDMATR register, the maximum HDMA time is set to 19.2[μs] (256 × 6 × (1 / f_{SYS})).

Note: To be precise, the bus assert time and RAM access time are added each time the HDMA transfer time is forcefully terminated at 19.2 [μs].

3.7 Function of ports

TMP92CZ26A has I/O port pins that are shown in Table 3.7.1 in addition to functioning as general-purpose I/O ports, these pins are also used by internal CPU and I/O functions. Table 3.7.2 lists I/O registers and their specifications.

Table 3.7.1 Port Functions (1/3) (R: PD= with programmable pull-down resistor, U= with pull-up resistor)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for built-in function
Port 1	P10 to P17	8	I/O	–	bit	D8 to D15
Port 4	P40 to P47	8	Output	–	bit	A0 to A7
Port 5	P50 to P57	8	Output	–	bit	A8 to A15
Port 6	P60 to P67	8	I/O	–	bit	A16 to A23
Port 7	P70	1	Output	–	(Fixed)	$\overline{RD}$
	P71	1	I/O	–	bit	$\overline{WRLL}$, $\overline{NDRE}$
	P72	1	I/O	–	bit	$\overline{WRLU}$, $\overline{NDWE}$
	P73	1	I/O	–	bit	EA24
	P74	1	I/O	–	bit	EA25
	P75	1	I/O	–	bit	R/$\overline{W}$, NDR/$\overline{B}$
	P76	1	I/O	–	bit	$\overline{WAIT}$
Port 8	P80	1	Output	–	(Fixed)	$\overline{CS0}$
	P81	1	Output	–	(Fixed)	$\overline{CS1}$, $\overline{SDCS}$
	P82	1	Output	–	(Fixed)	$\overline{CS2}$, $\overline{CSZA}$
	P83	1	Output	–	(Fixed)	$\overline{CS3}$, $\overline{CSXA}$
	P84	1	Output	–	(Fixed)	$\overline{CSZB}$
	P85	1	Output	–	(Fixed)	$\overline{CSZC}$
	P86	1	Output	–	(Fixed)	$\overline{CSZD}$, $\overline{ND0CE}$
	P87	1	Output	–	(Fixed)	$\overline{CSXB}$, $\overline{ND1CE}$
Port 9	P90	1	I/O	–	bit	TXD0
	P91	1	I/O	–	bit	RXD0
	P92	1	I/O	–	bit	SCLK0, $\overline{CTS0}$
	P96	1	Input	PD	(Fixed)	INT4, PX
	P97	1	Input	–	(Fixed)	PY
Port A	PA0 to PA7	8	Input	U	(Fixed)	KI0 to KI7
Port C	PC0	1	I/O	–	bit	INT0
	PC1	1	I/O	–	bit	INT1, TA0IN
	PC2	1	I/O	–	bit	INT2
	PC3	1	I/O	–	bit	INT3, TA2IN
	PC4	1	I/O	–	bit	EA26
	PC5	1	I/O	–	bit	EA27
	PC6	1	I/O	–	bit	EA28
	PC7	1	I/O	–	bit	KO8
Port F	PF0	1	I/O	–	bit	I2S0CKO
	PF1	1	I/O	–	bit	I2S0DO
	PF2	1	I/O	–	bit	I2S0WS
	PF3	1	I/O	–	bit	I2S1CKO
	PF4	1	I/O	–	bit	I2S1DO
	PF5	1	I/O	–	bit	I2S1WS
	PF7	1	Output	–	(Fixed)	SDCLK
Port G	PG0 to PG1	2	Input	–	(Fixed)	AN0 to AN1
	PG2	1	Input	–	(Fixed)	AN2, MX
	PG3	1	Input	–	(Fixed)	AN3, $\overline{ADTRG}$, MY
	PG4 to PG5	2	Input	–	(Fixed)	AN4 to AN5

Table 3.7.1 Port Functions (2/3)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for built-in function
Port J	PJ0	1	Output	–	(Fixed)	$\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$
	PJ1	1	Output	–	(Fixed)	$\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$
	PJ2	1	Output	–	(Fixed)	$\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$
	PJ3	1	Output	–	(Fixed)	SDLLDQM
	PJ4	1	Output	–	(Fixed)	SDLUDQM
	PJ5	1	I/O	–	bit	NDALE
	PJ6	1	I/O	–	bit	NDCLE
	PJ7	1	Output	–	(Fixed)	SDCKE
Port K	PK0	1	Output	–	(Fixed)	LCP0
	PK1	1	Output	–	(Fixed)	LLOAD
	PK2	1	Output	–	(Fixed)	LFR
	PK3	1	Output	–	(Fixed)	LVSYNC
	PK4	1	Output	–	(Fixed)	LHSYNC
	PK5	1	Output	–	(Fixed)	LGOE0
	PK6	1	Output	–	(Fixed)	LGOE1
	PK7	1	Output	–	(Fixed)	LGOE2
Port L	PL0 to PL7	8	Output	–	(Fixed)	LD0 to LD7
Port M	PM1	1	Output	–	(Fixed)	MLDALM, TA1OUT
	PM2	1	Output	–	(Fixed)	$\overline{\text{ALARM}}$, MLDALM
	PM7	1	Output	–	(Fixed)	PWE
Port N	PN0 to PN7	8	I/O	–	bit	KO0 to KO7
Port P	PP1	1	I/O	–	bit	TA3OUT
	PP2	1	I/O	–	bit	TA5OUT
	PP3	1	I/O	–	bit	INT5, TA7OUT
	PP4	1	I/O	–	bit	INT6, TB0IN0
	PP5	1	I/O	–	bit	INT7, TB1IN0
	PP6	1	Output	–	(Fixed)	TB0OUT0
	PP7	1	Output	–	(Fixed)	TB1OUT0
Port R	PR0	1	I/O	–	bit	SPDI
	PR1	1	I/O	–	bit	SPDO
	PR2	1	I/O	–	bit	$\overline{\text{SPCS}}$
	PR3	1	I/O	–	bit	SPCLK
Port T	PT0 to PT7	8	I/O	–	bit	LD8 to LD15
Port U	PU0 to PU4 ,PU6	6	I/O	–	bit	LD16 to LD20 , LD22
	PU5	1	I/O	–	bit	LD21
	PU7	1	I/O	–	bit	LD23, EO_TRGOUT
Port V	PV0	1	I/O	–	bit	SCLK0
	PV1	1	I/O	–	bit	–
	PV2	1	I/O	–	bit	–
	PV3	1	Output	–	(Fixed)	–
	PV4	1	Output	–	(Fixed)	–
	PV6	1	I/O	–	bit	SDA
	PV7	1	I/O	–	bit	SCL
Port W	PW0 to PW7	8	I/O	–	bit	–
Port X	PX4	1	Output	–	bit	CLKOUT, LDIV
	PX5	1	I/O	–	bit	X1USB
	PX7	1	I/O	–	bit	–

Table 3.7.1 Port Functions (3/3)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for built-in function
Port Z	PZ0	1	I/O	–	bit	EI_PODDATA
	PZ1	1	I/O	–	bit	EI_SYNCLK
	PZ2	1	I/O	–	bit	EI_PODREQ
	PZ3	1	I/O	–	bit	EI_REFCLK
	PZ4	1	I/O	–	bit	EI_TRGIN
	PZ5	1	I/O	–	bit	EI_COMRESET
	PZ6	1	I/O	–	bit	EO_MCUDATA
	PZ7	1	I/O	–	bit	EO_MCUREQ

Table 3.7.2 I/O Port and Specifications (1/4) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port 1	P10 toP17	Input port	X	0	0	None
		Output port	X	1		
		D8 to D15 bus	X	X	1	
Port 4	P40 to P47	Output port	X	None	0	None
		A0 to A7 Output	X	None	1	
Port 5	P50 to P57	Output port	X	None	0	None
		A8 to A15 Output	X	None	1	
Port 6	P60 to P67	Input port	X	0	0	None
		Output port	X	1		
		A16 to A23 Output	X	X	1	
Port 7	P70 to P76	Output port	X	1	0	None
	P71 to P76	Input port	X	0	0	
	P70	$\overline{RD}$ Output	X	None	1	
	P71	$\overline{WRLL}$ Output	1	1	1	
		$\overline{NDRE}$ Output	0			
	P72	$\overline{WRLU}$ Output	1	1	1	
		$\overline{NDWE}$ Output	0			
	P73	EA24 Output	X	1	1	
	P74	EA25 Output	X	1	1	
	P75	R/$\overline{W}$ Output	X	1	1	
		NDR/B Input	X	0	1	
	P76	$\overline{WAIT}$ Input	X	0	1	
Port 8	P80 to P87	Output port	X	None	0	0
	P80	$\overline{CS0}$ Output	X		1	None
	P81	$\overline{CS1}$ Output	X		1	0
		$\overline{SDCS}$ Output	X		X	1
	P82	$\overline{CS2}$ Output	X		1	0
		$\overline{CSZA}$ Output	X		0	1
		$\overline{SDCS}$ Output	X		1	1
	P83	$\overline{CS3}$ Output	X		1	0
		$\overline{CSXA}$ Output	X		X	1
	P84	$\overline{CSZB}$ Output	X		1	None
	P85	$\overline{CSZC}$ Output	X		1	
	P86	$\overline{CSZD}$ Output	X		1	0
		$\overline{ND0CE}$ Output	X		1	1
	P87	$\overline{CSXB}$ Output	X		1	0
		$\overline{ND1CE}$ Output	X		1	1

Table3.7.2 I I/O Port and Specifications (2/4) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port 9	P90, P92	Input port	X	0	0	None
	P91	Input port, RXD0 Input	X	0	None	None
	P96	Input port	X	None	0	None
	P97	Input port	X	None	None	None
	P90 to P92	Output port	X	1	0	0
	P90	TXD0 Output	X	1	1	0
		TXD0 Output (Open-drain)	X	1	1	1
	P92	SCLK0 Output	X	1	1	0
		SCLK0, $\overline{CTS0}$ Input	X	0	0	0
	P96	INT4 Input	X	None	1	None
Port A	PA0 to PA7	Input port	X	None	0	None
		KI0 to KI7 Input	X		1	
Port C	PC0 to PC7	Input port	X	0	0	None
		Output port	X	1	0	
	PC0	INT0 Input	X	0	1	
	PC1	INT1 Input	X	0	1	
		TA0IN Input	X	1	1	
	PC2	INT2 Input	X	0	1	
	PC3	INT3 Input	X	0	1	
		TA2IN Input	X	1	1	
	PC4	EA26 Output	X	0	1	
	PC5	EA27 Output	X	0	1	
	PC6	EA28 Output	X	0	1	
	PC7	KO8 Output (Open-drain)	X	1	1	
Port F	PF0 to PF5	Input port	X	0	0	None
	PF0 to PF5	Output port	X	1	0	
	PF7	Output port	X	None	0	
	PF0	I2S0CKO Output	X	X	1	
	PF1	I2S0DO Output	X	X	1	
	PF2	I2S0WS Output	X	X	1	
	PF3	I2S1CKO Output	1	X	1	
	PF4	I2S1DO Output	X	X	1	
	PF5	I2S1WS Output	X	X	1	
	PF7	SDCLK Output	X	None	1	

Table3.7.2 I/O Port and Specifications (3/4) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port G	PG0 to PG5	Input port	X	None	0	None
		AN0 to AN5 Input				
	PG3	$\overline{\text{ADTRG}}$ Input			1	
	PG2	MX Output Note:			0	
	PG3	MY Output Note:				
Port J	PJ5 to PJ6	Input port	X	0	0	None
	PJ5 to PJ6	Output port	X	1	0	
	PJ0 to PJ4, PJ7	Output port	X	None	0	
	PJ0	$\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$ Output	X	None	1	
	PJ1	$\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$ Output	X		1	
	PJ2	$\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$ Output	X		1	
	PJ3	SDLLDQM Output	X		1	
	PJ4	SDLUDQM Output	X		1	
	PJ5	NDALE Output	X	1	1	
	PJ6	NDCLE Output	X			
	PJ7	SDCKE Output	X	None	1	
Port K	PK0 to PK7	Output port	X	None	0	None
	PK0	LCP0 output	X		1	
	PK1	LLOAD output	X		1	
	PK2	LFR output	X		1	
	PK3	LVSYNC output	X		1	
	PK4	LHSYNC output	X		1	
	PK5	LGOE0 output	X		1	
	PK6	LGOE1 output	X		1	
	PK7	LGOE2 output	X		1	
Port L	PL0 to PL7	Output port	X	None	0	None
	PL0 to PL7	LD0 to LD7 Output	X		1	
Port M	PM1 to PM2	Output port	X	None	0	None
	PM1	TA1OUTOutput	0		1	
		MLDALM Output	1		1	
	PM2	$\overline{\text{MLDALM}}$ Output	0		1	
		$\overline{\text{ALARM}}$ Output	1		1	
	PM7	PWE Output	X		1	
Port N	PN0 to PN7	Input port	X	0	0	None
		Output port (CMOS Output)	X	1	0	
		KO Output (Open-drain Output)	X		1	
Port P	PP1 to PP5	Input port	X	0	0	None
	PP1 to PP5	Output port	X	1	0	
	PP6 to PP7	Output port	X	None	0	
	PP1	TA3OUT output	X	1	1	
	PP2	TA5OUT output	X	1	1	
	PP3	INT5 input	X	0	1	
		TA7OUT output	X	1		
	PP4	INT6 input	X	0	1	
		TB0IN0 input	X	1		
	PP5	INT7 input	X	0	1	
		TB1IN0 input	X	1		
	PP6	TB0OUT0 output	X	None	1	
	PP7	TB1OUT1 output	X		1	

Note: Case of using touch screen

Table 3.7.2 I/O Port and Specifications (4/4) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port R	PR0 to PR3	Input port	X	0	0	None
	PR0 to PR3	Output port	X	1	0	
	PR0	SPDI Input	X	0	1	
	PR1	SPDO Output	X	1	1	
	PR2	$\overline{\text{SPCS}}$ Output	X	1	1	
	PR3	SPCLK Output	X	1	1	
Port T	PT0 to PT7	Input port	X	0	0	None
	PT0 toPT7	Output port	X	1	0	
	PT0 to PT7	LD8 to LD15 Output	X	1	1	
Port U	PU0 to PU7	Input port	X	0	0	None
	PU0 to PU7	Output port	X	1	0	
	PU0 to PU7	LD16 to LD23 Output	X	1	1	
	PU7	EO_TRGOUT ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
Port V	PV0 to PV2	Input port	X	0	0	None
	PV0 to PV4	Output port	X	1	0	
	PV6 to PV7	Input port	X	0	0	0
	PV6 to PV7	Output port	X	1	0	
	PV6 to PV7	Output port (Open-drain)	X	1	0	1
	PV0	SCLK0 Output	X	1	1	None
	PV6	SDA I/O	X	1	1	0
		SDA I/O (Open-drain)	X	1	1	1
	PV7	SCL I/O	X	1	1	0
		SCL I/O (Open-drain)	X	1	1	1
Port W	PW0 to PW7	Input port	X	0	0	None
	PW0 to PW7	Output port	X	1	0	
Port X	PX5, PX7	Input port	X	0	0	None
	PX4	Output port	X	None	0	
	PX5, PX7	Output port	X	1	0	
	PX4	CLKOUT Output	0	None	1	
		LDIV Output	1		1	
	PX5	X1USB Input	X	0	1	
Port Z	PZ0 to PZ7	Input port	X	0	0	None
		Output port	X	1	0	
	PZ0	EI_PODDATA ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	None
	PZ1	EI_SYNCLK ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
	PZ2	EI_PODREQ ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
	PZ3	EI_REFCLK ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
	PZ4	EI_TRGIN ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
	PZ5	EI_COMRESET ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
	PZ6	EO_MCUDATA ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	
	PZ7	EO_MCUREQ ($\overline{\text{DBGE}}$ = "0") Note:	X	X	X	

Note: When Debug mode, it is set to the Debug pin regardless of port setting.

3.7.1 Port 1 (P10 to P17)

Port1 is an 8-bit general-purpose I/O port. Bits can be individually set as either inputs or outputs by control register P1CR and function register P1FC.

In addition to functioning as a general-purpose I/O port, port1 can also function as a data bus (D8 to D15).

Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 1 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Data bus (D8 to D15)
1	0	Don't use this setting
1	1	Input port (P10 to P17)

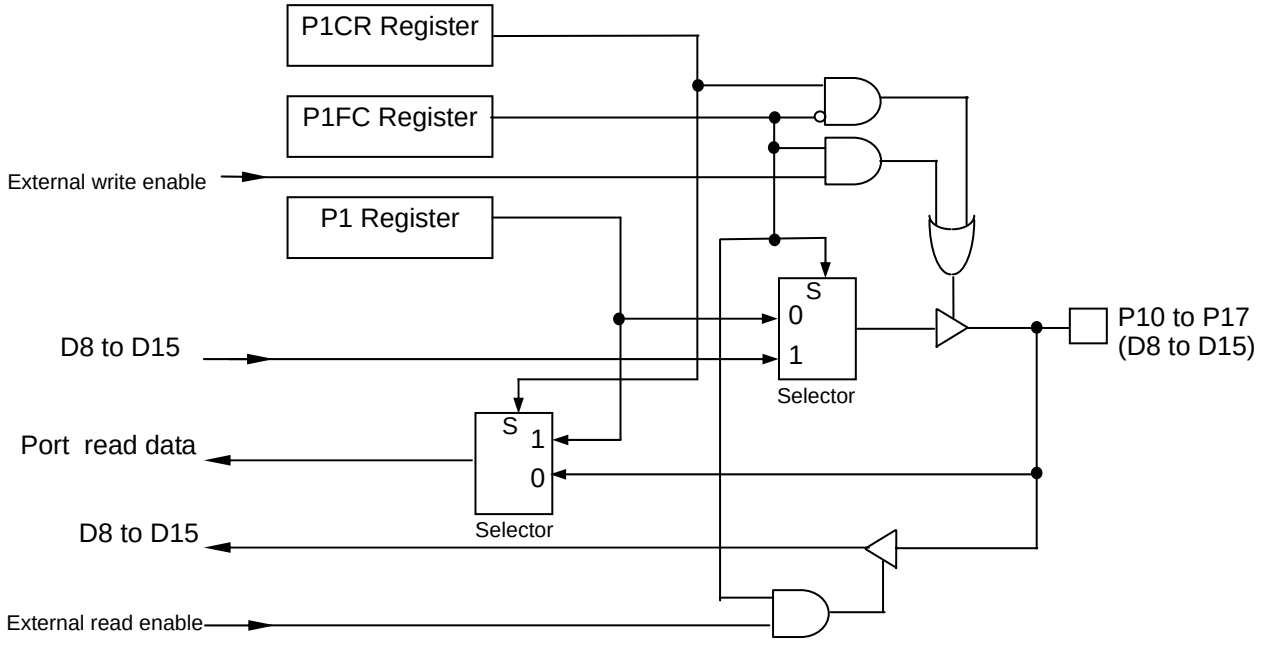


Figure 3.7.1 Port1

Port 1 register

P1 (0004H)

	7	6	5	4	3	2	1	0
bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”)							

Port 1 Control register

P1CR (0006H)

	7	6	5	4	3	2	1	0
bit Symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 1 Function register

P1FC (0007H)

	7	6	5	4	3	2	1	0
bit Symbol								P1F
Read/Write								W
After reset Note2:								0/1
Function								0: Port 1:Data bus (D8 to D15)

Port 1 Drive register

P1DR (0081H)

	7	6	5	4	3	2	1	0
bit Symbol	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note1: Read-modify-write is prohibited for P1CR, P1FC.

Note2: It is set to “Port” or “Data bus” by AM pins state.

Figure 3.7.2 Register for Port1

3.7.2 Port 4 (P40 to P47)

Port4 is an 8-bit general-purpose Output ports. In addition to functioning as a general-purpose Output port, port4 can also function as an address bus (A0 to A7). Each bit can be set individually for function. Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 4 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address bus (A0 to A7)
1	0	Don't use this setting
1	1	Output port (P40 to 47)

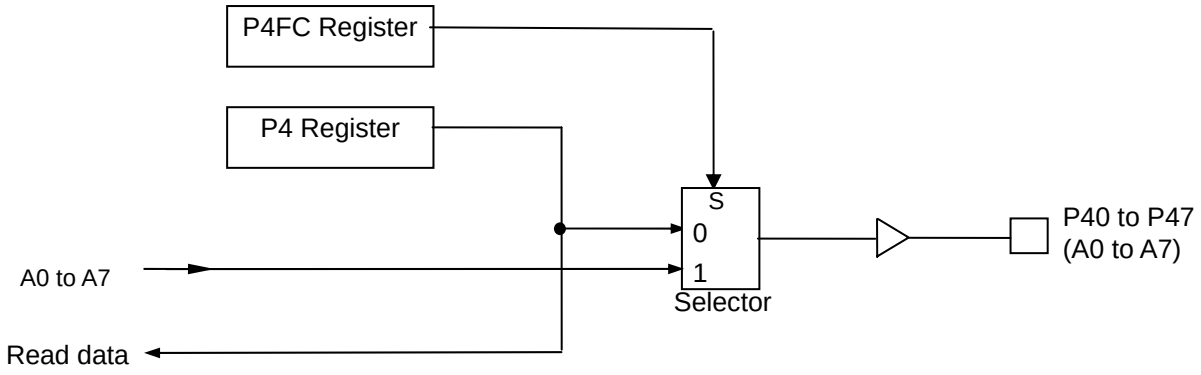


Figure 3.7.3 Port4

Port 4 register

P4 (0010H)

	7	6	5	4	3	2	1	0
bit Symbol	P47	P46	P45	P44	P43	P42	P41	P40
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Port 4 Function register

P4FC (0013H)

	7	6	5	4	3	2	1	0
bit Symbol	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
Read/Write	W							
After reset Note2:	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Function	0:Port 1:Address bus (A0 to A7)							

Port 4 Drive register

P4DR (0084H)

	7	6	5	4	3	2	1	0
bit Symbol	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note1: Read-modify-write is prohibited for P4FC.

Note2: It is set to “Port” or “Data bus” by AM pins state.

Figure 3.7.4 Register for Port1r

3.7.3 Port 5 (P50 to P57)

Port5 is an 8-bit general-purpose Output ports. In addition to functioning as a general-purpose I/O port, port5 can also function as an address bus (A8 to A15). Each bit can be set individually for function. Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 5 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address bus (A8 ~ A15)
1	0	Don't use this setting
1	1	Output port (P50 ~ P57)

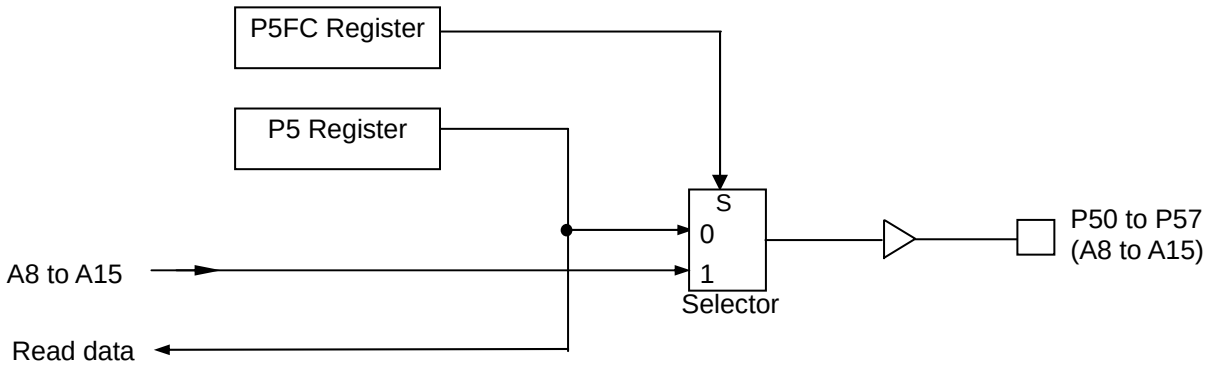


Figure 3.7.5 Port5

Port 5 register

P5 (0014H)

	7	6	5	4	3	2	1	0
bit Symbol	P57	P56	P55	P54	P53	P52	P51	P50
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Port 5 Function register

P5FC (0017H)

	7	6	5	4	3	2	1	0
bit Symbol	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
Read/Write	W							
After reset Note2:	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Function	0:Port 1:Address bus (A8 to A15)							

Port 5 Drive register

P5DR (0085H)

	7	6	5	4	3	2	1	0
bit Symbol	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note1: Read-modify-write is prohibited for P5FC.

Note2: It is set to “Port” or “Data bus” by AM pins state.

Figure 3.7.6 Register for Port5

3.7.4 Port 6 (P60 to P67)

Port6 is an 8-bit general-purpose I/O ports. Bits can be individually set as either inputs or outputs and function by control register P6CR and function register P6FC.

In addition to functioning as a general-purpose I/O port, port6 can also function as an address bus (A16 to A23). Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 6 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address bus(A16 ~ A23)
1	0	Don't use this setting
1	1	Input port(P60 ~ P67)

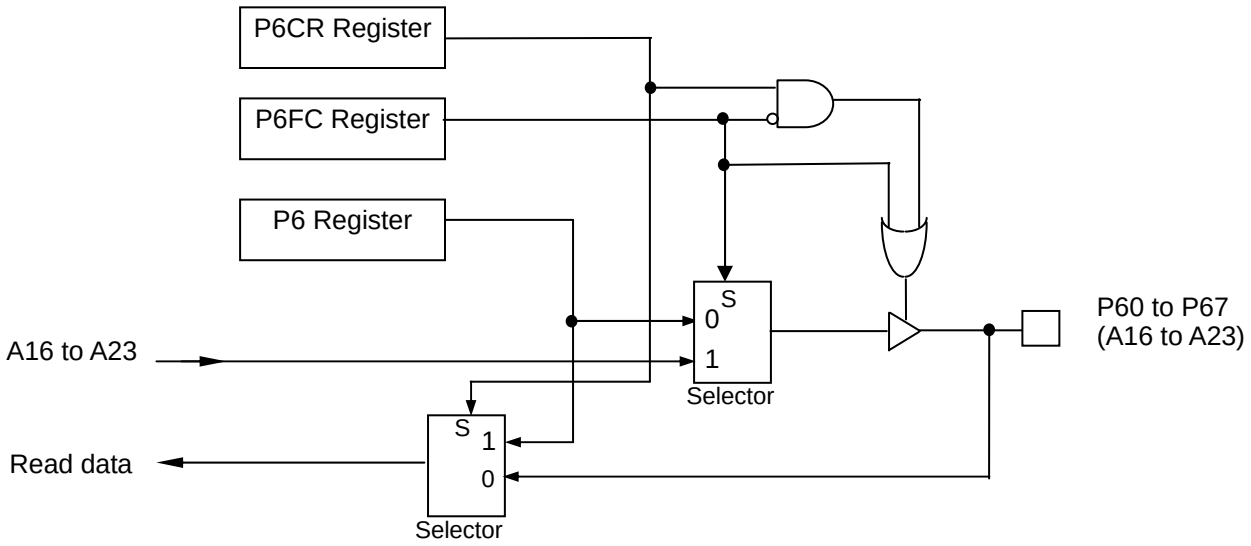


Figure 3.7.7 Port6

Port 6 register

P6 (0018H)

	7	6	5	4	3	2	1	0
bit Symbol	P67	P66	P65	P64	P63	P62	P61	P60
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”)							

Port 6 Control register

P6CR (001AH)

	7	6	5	4	3	2	1	0
bit Symbol	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0:Input 1:Output							

Port 6 Function register

P6FC (001BH)

	7	6	5	4	3	2	1	0
bit Symbol	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
Read/Write	W							
After reset Note2:	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Function	0: Port 1:Address bus (A16 to A23)							

Port 6 Drive buffer register

P6DR (0086H)

	7	6	5	4	3	2	1	0
bit Symbol	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note: Read-modify-write is prohibited for P6CR, P6FC.

Note2: It is set to “Port” or “Data bus” by AM pins state.

Figure 3.7.8 Register for Port6

3.7.5 Port 7 (P70 to P76)

Port7 is a 7-bit general-purpose I/O port (P70 is used for output only). Bits can be individually set as either inputs or outputs by control register P7CR and function register P7FC. In addition to functioning as a general-purpose I/O port, P70 to P76 pins can also function interface-pin for external memory.

A reset initializes P70 pin to output port mode, and P71 to P76 pins to input port mode.

Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 7 to the following function pins:

Initial setting of P70 pin

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	$\overline{RD}$ pin
1	0	Don't use this setting
1	1	Output port (P70)

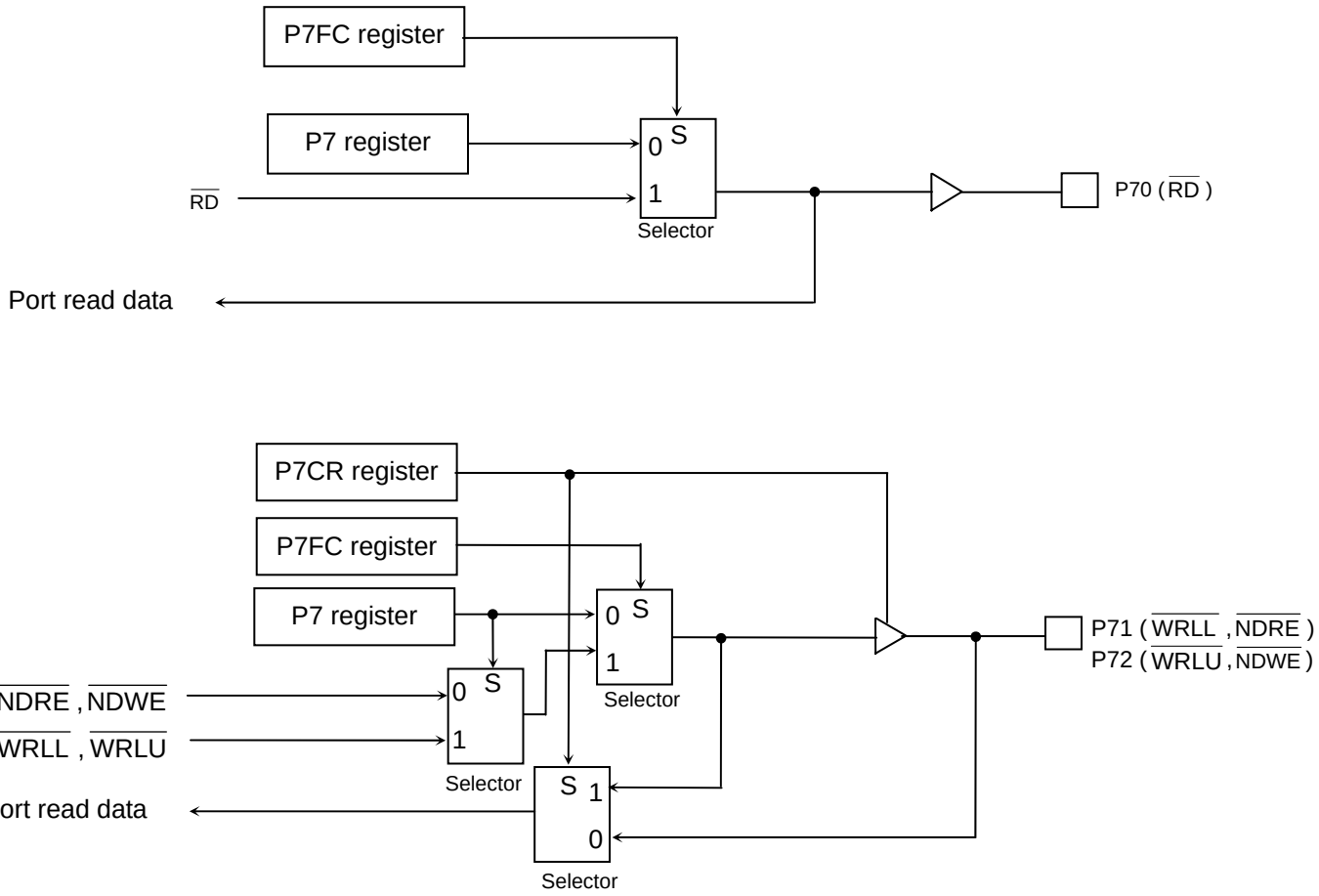


Figure 3.7.9 Port7

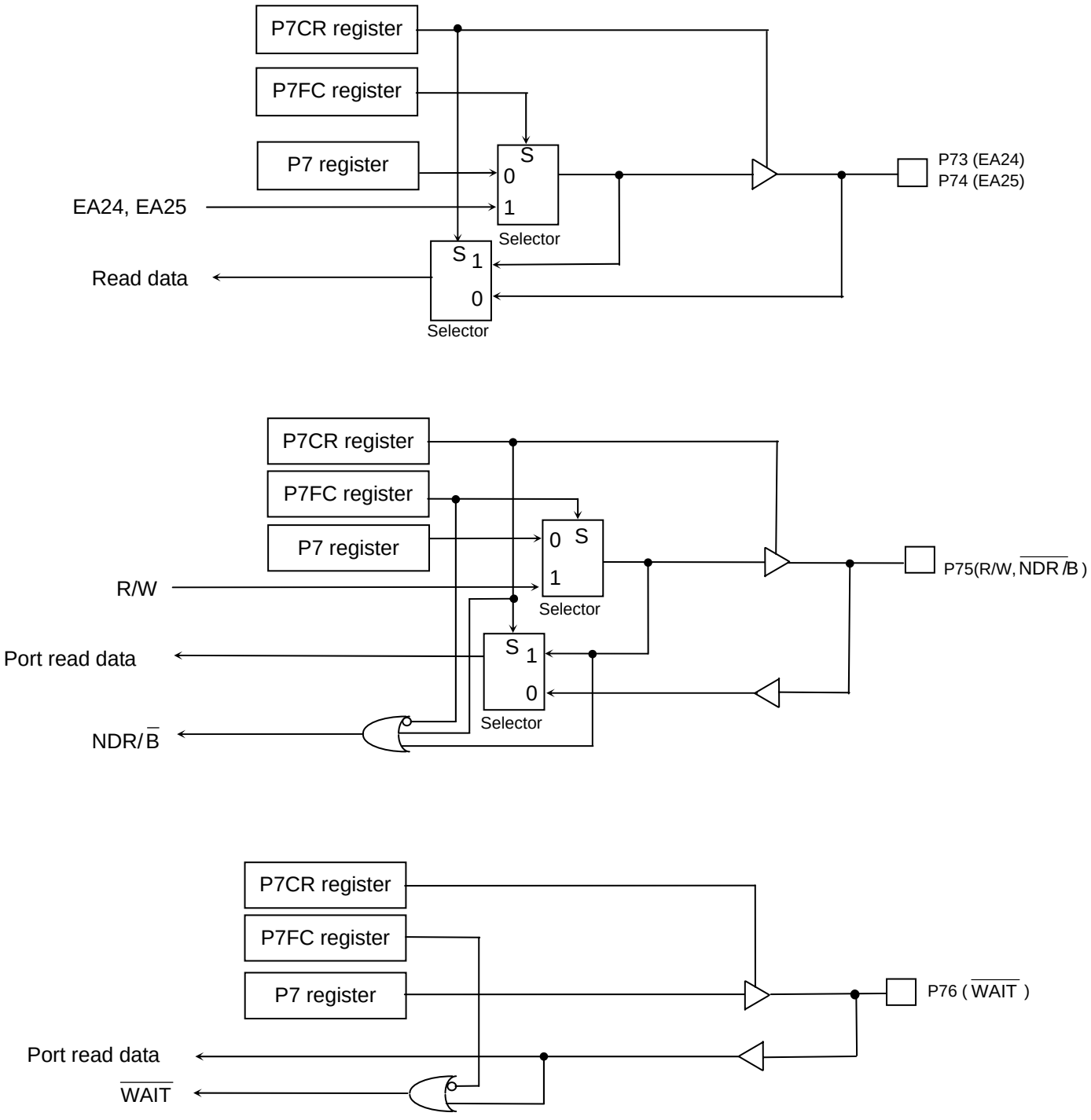


Figure 3.7.10 Port7

Port 7 register

P7 (001CH)

	7	6	5	4	3	2	1	0
bit Symbol		P76	P75	P74	P73	P72	P71	P70
Read/Write		R/W						
After reset		Data from external port (Output latch register is set to “1”)		Data from external port (Output latch register is cleared to “0”)		Data from external port (Output latch register is set to “1”)		1

Port 7 Control register

P7CR (001EH)

	7	6	5	4	3	2	1	0
bit Symbol		P76C	P75C	P74C	P73C	P72C	P71C	
Read/Write		W						
After reset		0	0	0	0	0	0	
Function		0: Input 1: Output						

Port 7 Function register

P7FC (001FH)

	7	6	5	4	3	2	1	0
bit Symbol		P76F	P75F	P74F	P73F	P72F	P71F	P70F
Read/Write		W						
After reset		0	0	0	0	0	0	0/1 Note3:
Function		0:Port 1: $\overline{WAIT}$	Refer to following table			0:Port 1: $\overline{NDWE}$ at <P72>=0 $\overline{WRLU}$ at <P72>=1	0:Port 1: $\overline{NDRE}$ at <P71>=0 $\overline{WRLL}$ at <P71>=1	0:Port 1: $\overline{RD}$

Port 7 Drive register

P7DR (0087H)

	7	6	5	4	3	2	1	0
bit Symbol		P76D	P75D	P74D	P73D	P72D	P71D	P70D
Read/Write		R/W						
After reset		1	1	1	1	1	1	1
Function		Input/Output buffer drive register for standby mode						

P73 setting

<P73C> / <P73F>	0	1
0	Input Port	Output Port
1	Reserved	EA24Output

P72 setting

<P72C> / <P72F>	0	1
0	Input Port	Output Port
1	Reserved	$\overline{NDWE}$ Output (at <P72>=0) $\overline{WRLU}$ Output (at <P72>=1)

P71 setting

<P71C> / <P71F>	0	1
0	Input Port	Output Port
1	Reserved	$\overline{NDRE}$ Output (at <P71>=0) $\overline{WRLL}$ Output (at <P71>=1)

P76 setting

<P76C> / <P76F>	0	1
0	Input Port	Output Port
1	$\overline{WAIT}$ Input	Reserved

P75 setting

<P75C> / <P75F>	0	1
0	Input Port	Output Port
1	NDR/$\overline{B}$ Input	R/W Output

P74 setting

<P74C> / <P74F>	0	1
0	Input Port	Output Port
1	Reserved	EA25Output

Note1: Read-modify-write is prohibited for P7CR, P7FC.

Note2: When $\overline{NDRE}$ and $\overline{NDWE}$ are used, set registers by following order to avoid outputting negative glitch.

Order	Registser	bit2	bit1
(1)	P7	0	0
(2)	P7FC	1	1
(3)	P7CR	1	1

Note3: Note2: It is set to “Port” or “Data bus” by AM pins state.

Figure 3.7.11 Register for Port7

3.7.6 Port 8 (P80 to P87)

Port 80 to 87 are 8-bit output ports. Resetting sets output latch of P82 to “0” and output latches of P80 to P81, P83 to P87 to “1”. But if it is started at boot mode (AM [1:0]= “11”), output latch of P82 is set to “1”.

Port 8 also function as interface-pin for external memory.

Writing “1” in the corresponding bit of P8FC, P8FC2 enables the respective functions.

Resetting resets P8FC to “0” and P8FC2 to “0”, sets all bits to output ports.

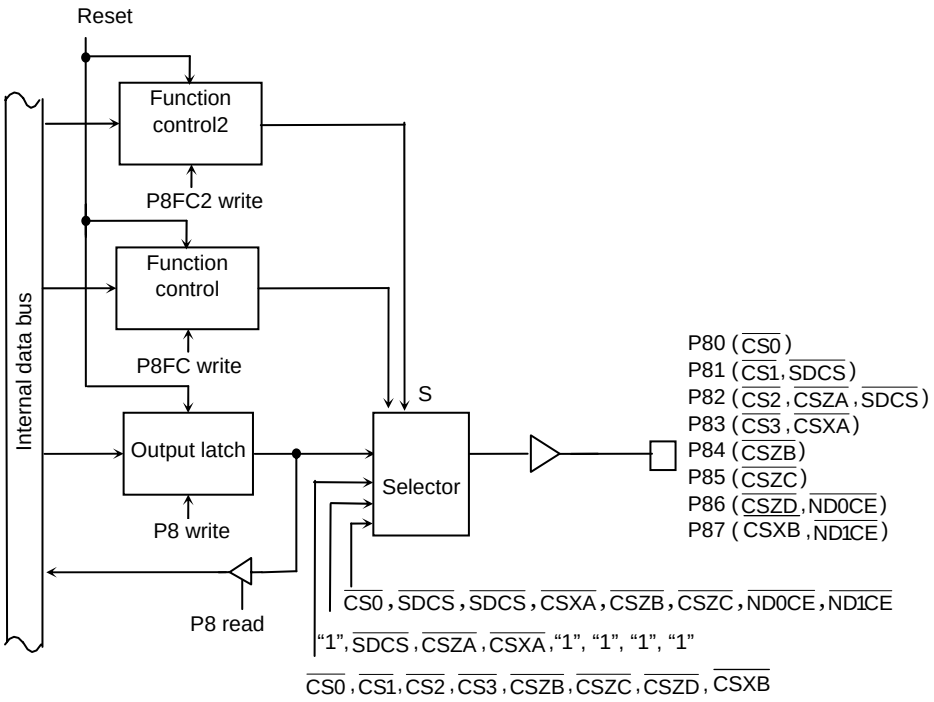


Figure 3.7.12 Port 8

Port 8 register

P8 (0020H)

	7	6	5	4	3	2	1	0
bit Symbol	P87	P86	P85	P84	P83	P82	P81	P80
Read/Write	R/W							
After reset	1	1	1	1	1	0 (Note3)	1	1

Port 8 Function register

P8FC (0023H)

	7	6	5	4	3	2	1	0
bit Symbol	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: <P87F2>	0: Port 1: <P86F2>	0: Port 1: $\overline{CSZC}$	0: Port 1: $\overline{CSZB}$	Refer to following table		0: Port 1: $\overline{CS1}$	0: Port 1: $\overline{CS0}$

Port 8 Function registers 2

P8FC2 (0021H)

	7	6	5	4	3	2	1	0
bit Symbol	P87F2	P86F2			P83F2	P82F2	P81F2	
Read/Write	W				W			
After reset	0	0			0	0	0	
Function	0: $\overline{CSXB}$ 1: $\overline{ND1CE}$	0: $\overline{CSZD}$ 1: $\overline{ND0CE}$			Refer to following table		0: <P81F> 1: $\overline{SDCS}$	

Port 8 Drive register

P8DR (0088H)

	7	6	5	4	3	2	1	0
bit Symbol	P87D	P86D	P85D	P84D	P83D	P82D	P81D	P80D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

P86 setting

<P86F> / <P86F2>	0	1
0	Output port	$\overline{CSZD}$ Output
1	Don't setting	$\overline{ND0CE}$ Output

P83 setting

<P83F> / <P83F2>	0	1
0	Output port	$\overline{CS3}$ Output
1	$\overline{CSXA}$ Output	

P82 setting

<P82F> / <P82F2>	0	1
0	Output port	$\overline{CS2}$ Output
1	$\overline{CSZA}$ Output	$\overline{SDCS}$ Output

P87 setting

<P87F> / <P87F2>	0	1
0	Output port	$\overline{CSXB}$ Output
1	Don't setting	$\overline{ND1CE}$ Output

Note1: Read-modify-write is prohibited for P8FC and P8FC2.

Note2: Don't write "1" to P8<P82>- register before setting P82-pin to /CS2 or /CSZA because of P82-pin output "0" as /CE for program memory by reset.

Note3: If it is started at boot mode (AM [1:0]= "11"), output latch of P82 is set to "1".

Note4: When $\overline{ND0CE}$ and $\overline{ND1CE}$ are used, set registers by following order.

Order	Registser	bit2	bit1
(1)	P8	1	1
(2)	P8FC2	1	1
(3)	P8FC	1	1

Figure 3.7.13 Register for Port 8

3.7.7 Port 9 (P90 to P92, P96, P97)

P90 to P92 are 3-bit general-purpose I/O port. I/O can be set on bit basis using the control register. Resetting sets P90 to P92 to input port and all bits of output latch to"1".

P96 to P97 are 2-bit general-purpose input port.

Writing "1" in the corresponding bit of P9FC enables the respective functions.

Resetting resets the P9FC to "0", and sets all bits to input ports.

(1) Port 90 (TXD0), Port 91 (RXD0), Port 92 (SCLK0, $\overline{\text{CTS0}}$)

Port 90 to 92 are general-purpose I/O port. They are also used either SIO0. Each pin is below.

	SIO mode (SIO0 module)	UART, IrDA mode (SIO0 module)
P90	TXD0 (Data output)	TXD0 (Data output)
P91	RXD0 (Data input)	RXD0 (Data input)
P92	SCLK0 (Clock input or output)	$\overline{\text{CTS0}}$ (Clear to send)

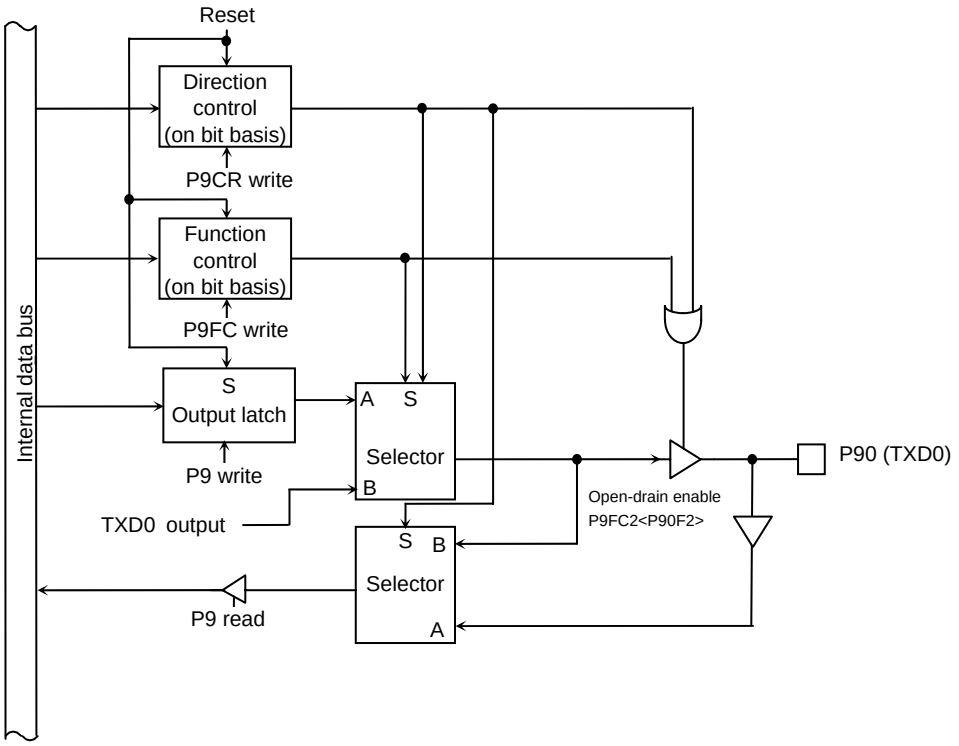


Figure 3.7.14 P90

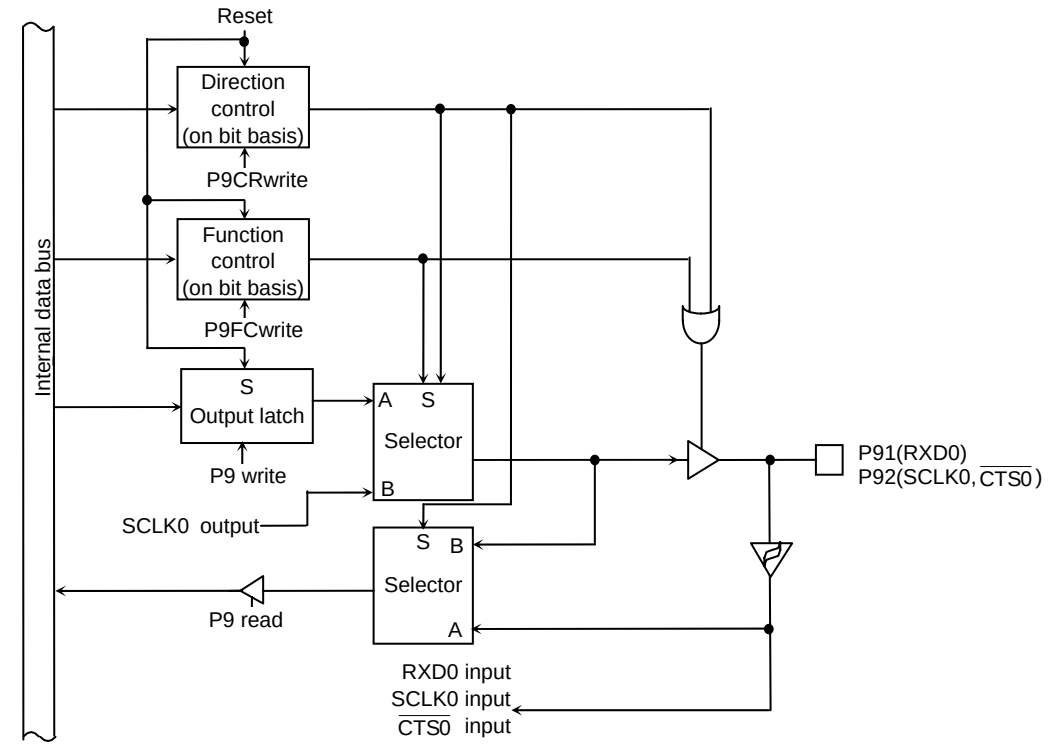


Figure 3.7.15 P91, 92

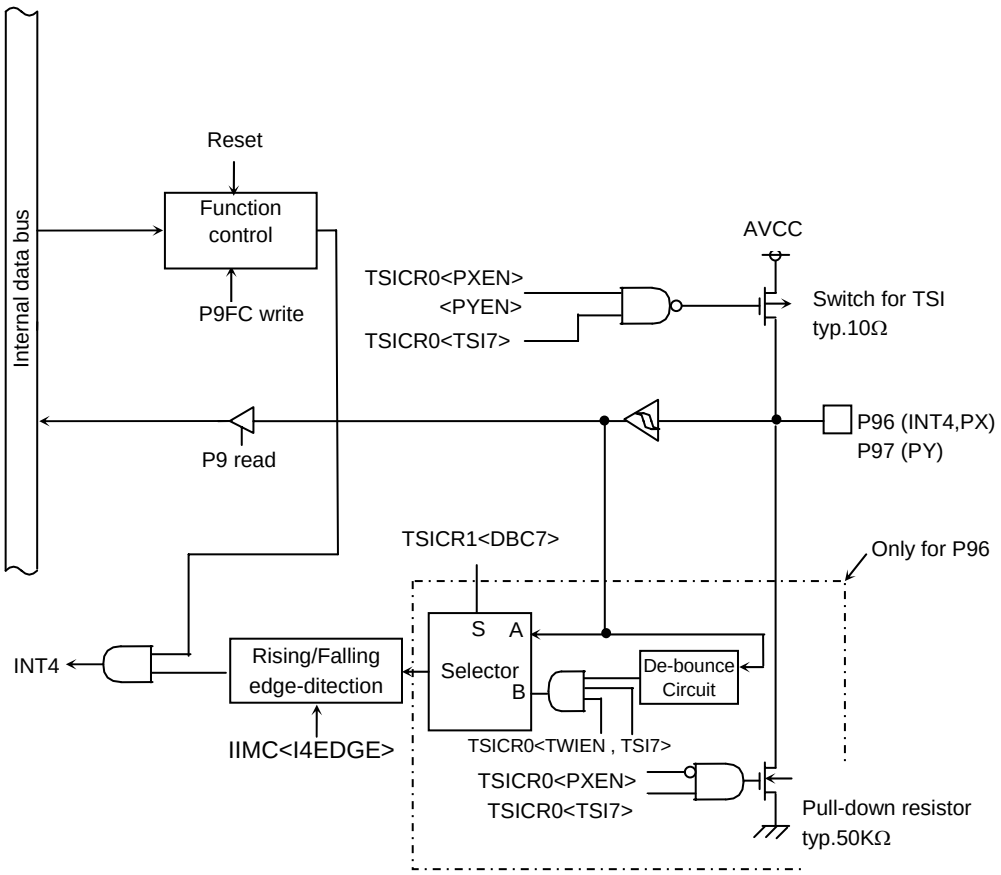


Figure 3.7.16 Port 96,97

Port 9 register

P9 (0024H)

	7	6	5	4	3	2	1	0
bit Symbol	P97	P96				P92	P91	P90
Read/Write	R					R/W		
After reset	Data from external port					Data from external port (Output latch register is set to “1”)		

Port 9 control register

P9CR (0026H)

	7	6	5	4	3	2	1	0
bit Symbol						P92C	P91C	P90C
Read/Write						W		
After reset						0	0	0
Function						Refer to following table		

Port 9 function register

P9FC (0027H)

	7	6	5	4	3	2	1	0
bit Symbol		P96F				P92F		P90F
Read/Write		W				W		W
After reset		0				0		0
Function		0: Input port 1: INT4				Refer to following table		Refer to following table

Port 9 Function registers 2

P9FC2 (0025H)

	7	6	5	4	3	2	1	0
bit Symbol	–					–		P90F2
Read/Write	W					W		W
After reset	0					0		0
Function	Always write “0”					Always write “0”		0:CMOS 1: open-drain

Port 9 drive register

P9DR (0089H)

	7	6	5	4	3	2	1	0
bit Symbol	P97D	P96D				P92D	P91D	P90D
Read/Write	R/W					R/W		
After reset	1	1				1	1	1
Function	Input/Output buffer drive register for standby mode							

P92 setting

<P92C> / <P92F>	0	1
0	Input port, $\overline{CTS0}$ Input	Input port, $\overline{CTS0}$ Input
1	Don’t setting	Don’t setting

P91 setting

<P91C>	
0	1
Input port RXD0 Input	Output port

P90 setting

<P90C> / <P90F>	0	1
0	Input port	Output port
1	Don’t setting	TXD0 Output

Note 1: Read-modify-write is prohibited for P9CR, P9FC and P9FC2.

Note 2: When setting P96 pin to INT4 input, set P9DR<P96D> to “0” (prohibit input), and when driving P96 pin to “0”, execute HALT instruction. This setting generates INT4 inside. If don’t using external interrupt in HALT condition, set like an interrupt don’t generated. (e.g. change port setting)

Figure 3.7.17 Register for Port 9

3.7.8 Port A (PA0 to PA7)

Port A0 to A7 are 8-bit general-purpose input ports with pull-up resistor. In addition to functioning as general-purpose I/O ports, port A0 to A7 can also Key-on wake-up function as Keyboard interface. The various functions can each be enabled by writing a “1” to the corresponding bit of the Port A Function Register (PAFC).

Resetting resets all bits of the register PAFC to “0” and sets all pins to be input port.

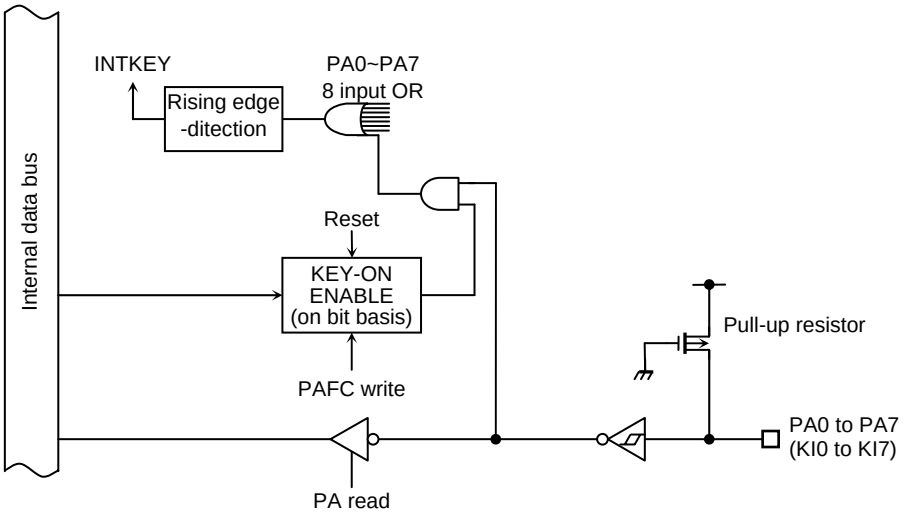


Figure 3.7.18 Port A

When PAFC = “1”, if either of input of KI0-KI7 pins falls down, INTKEY interrupt is generated. INTKEY interrupt can release all HALT mode.

Port A register

PA (0028H)

	7	6	5	4	3	2	1	0
bit Symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	R							
After reset	Data from external port							

Port A Function register

PAFC (002BH)

	7	6	5	4	3	2	1	0
bit Symbol	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: KEY IN disable 1: KEY IN enable							

Port A Drive register

PADR (008AH)

	7	6	5	4	3	2	1	0
bit Symbol	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note 1: Read-modify-write is prohibited for PAFC.

Figure 3.7.19 Register for Port A

3.7.9 Port C (PC0 to PC7)

PC0 to PC7 are 8-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets Port C to an input port. It also sets all bits of the output latch register to “1”.

In addition to functioning as a general-purpose I/O port, Port C can also function as input pin for timers (TA0IN, TA2IN), input pin for external interruption (INT0 to INT3), Extension address function (EA26, EA27, EA28) and output pin for Key (KO8). Above setting is used the function register PCFC. Edge select of external interruption establishes it with IIMC register, which there is in interruption controller.

(1) PC0 (INT0), PC2 (INT2)

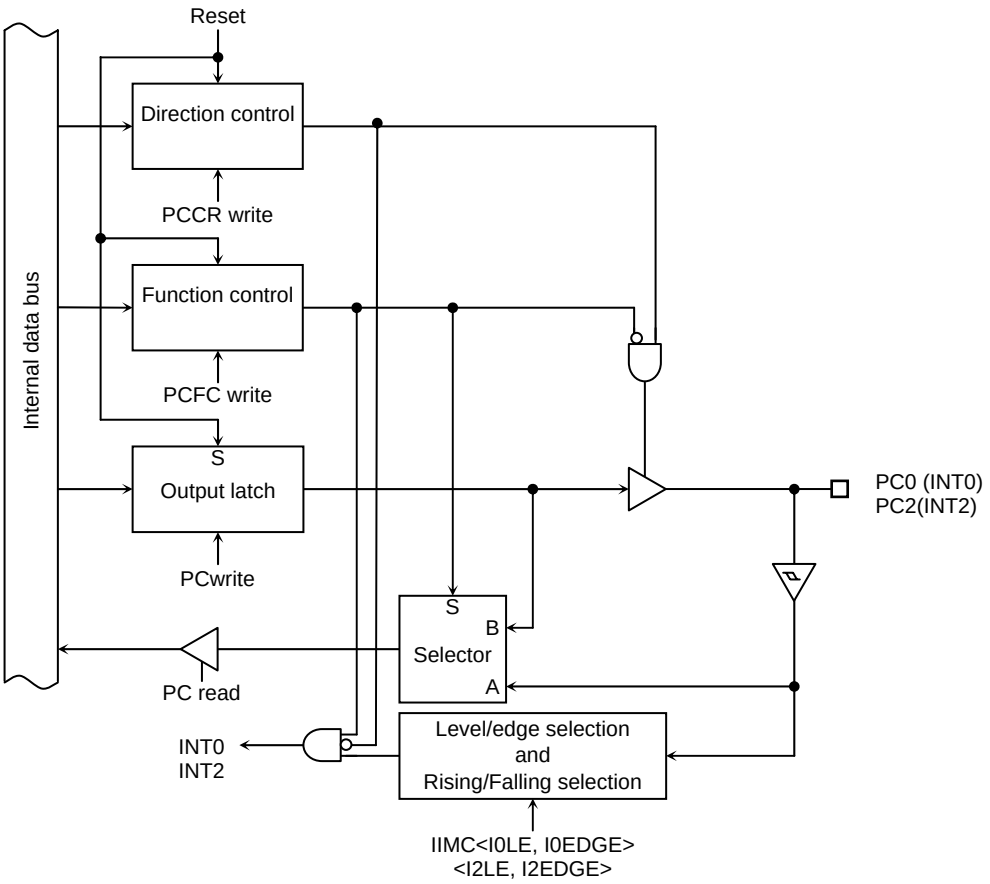


Figure 3.7.20 Port C0, C2

(2) PC1 (INT1, TA0IN), PC3 (INT3, TA2IN)

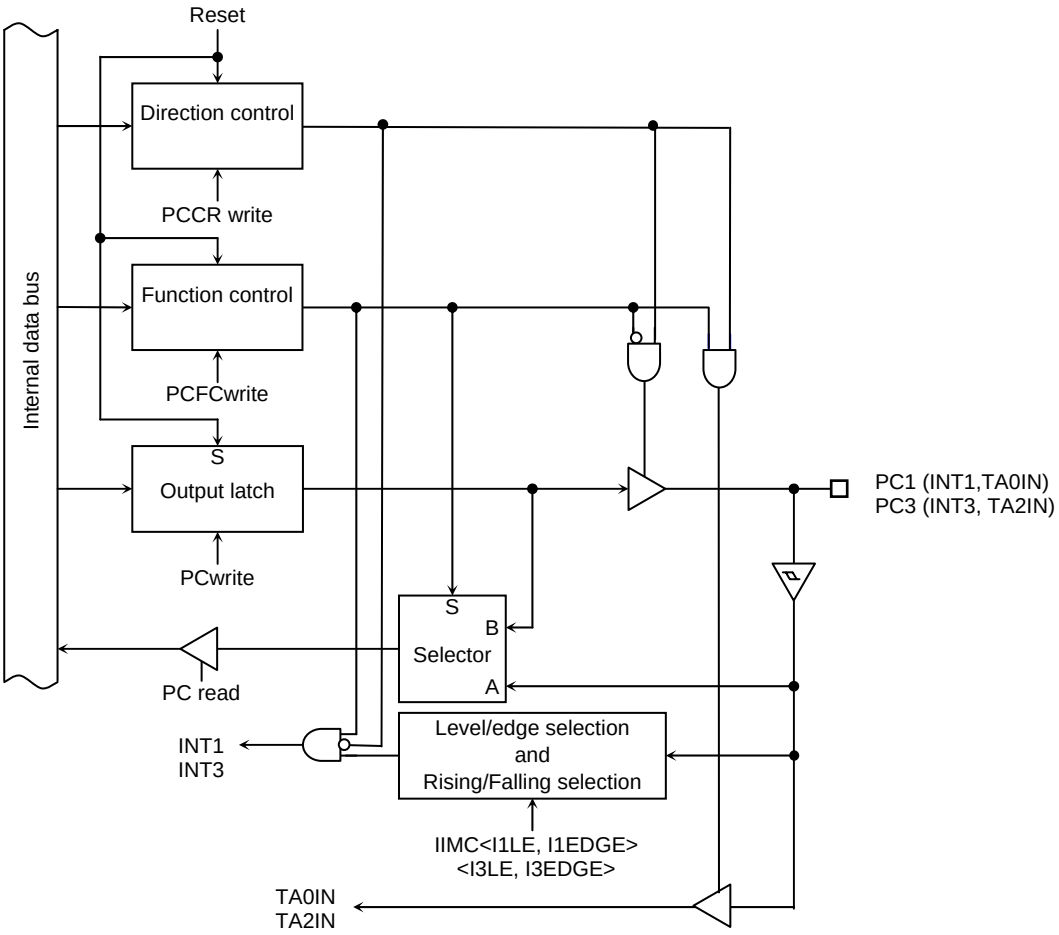


Figure 3.7.21 Port C1,C3

(3) PC4 (EA26), PC5 (EA27), PC6 (EA28)

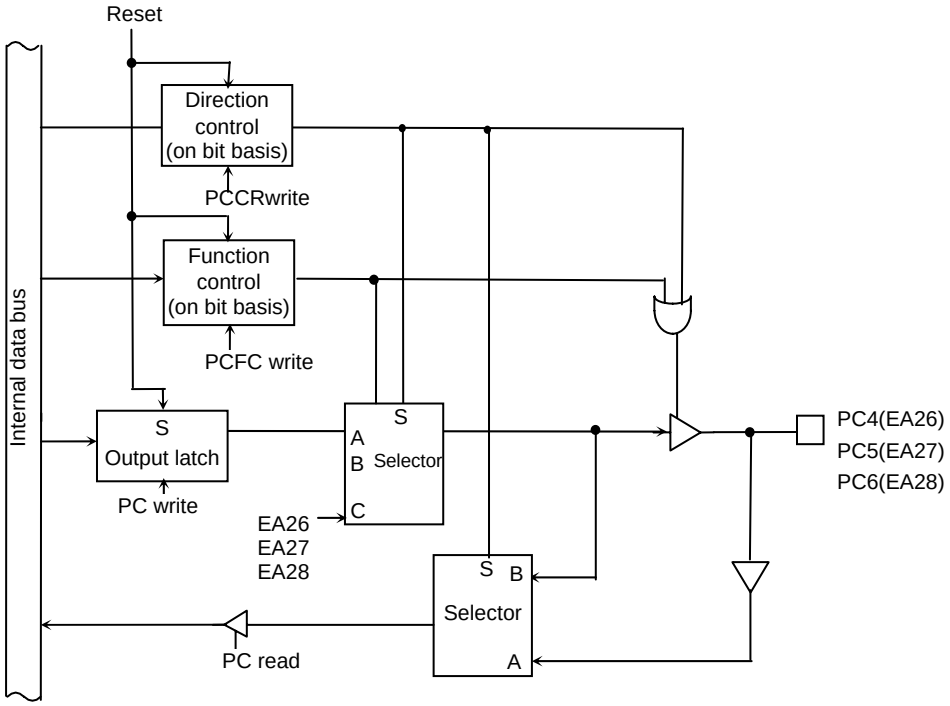


Figure 3.7.22 Port C4, C5, C6

(4) PC7 (KO8)

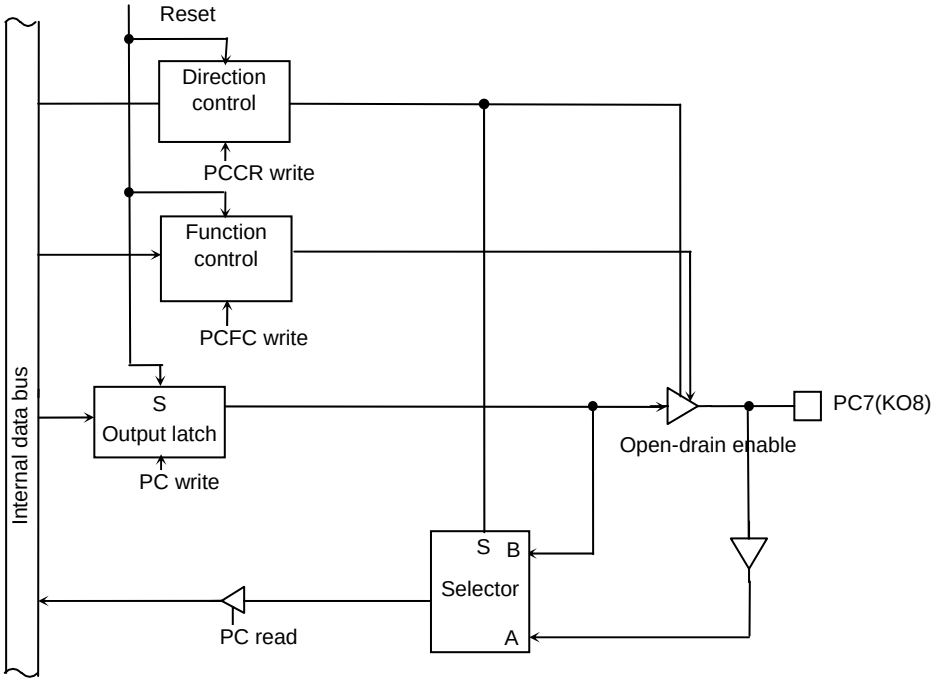


Figure 3.7.23 Port C7

Port C register

PC (0030H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Read/Write	R/W							
After reset	Data from external port (Output latch register is set to “1”)							

Port C control register

PCCR (0032H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7C	PC6C	PC5C	PC4C	PC3C	PC2C	PC1C	PC0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port C function register

PCFC (0033H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7F	PC6F	PC5F	PC4F	PC3F	PC2F	PC1F	PC0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Refer to following table							

Port C drive register

PCDR (008CH)

	7	6	5	4	3	2	1	0
bit Symbol	PC7D	PC6D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

PC2 setting

<PC2C> / <PC2F>	0	1
0	Input port	Output port
1	INT2	Don’t setting

PC1 setting

<PC1C> / <PC1F>	0	1
0	Input port	Output port
1	INT1	TA0IN input

PC0 setting

<PC0C> / <PC0F>	0	1
0	Input port	Output port
1	INT0	Don’t setting

PC5 setting

<PC5C> / <PC5F>	0	1
0	Input port	Output port
1	EA27output	Reserved

PC4 setting

<PC4C> / <PC4F>	0	1
0	Input port	Output port
1	EA26 output	Reserved

PC3 setting

<PC3C> / <PC3F>	0	1
0	Input port	Output port
1	INT3	TA2IN input

PC7 setting

<PC7C> / <PC7F>	0	1
0	Input port	Output port
1	Don’t setting	KO8output (Open-drain)

PC6 setting

<PC6C> / <PC6F>	0	1
0	Input port	Output port
1	EA28output	Reserved

Note 1: Read-Modify-Write is prohibited for the registers PCCR, PCFC.

Note 2: When setting PC3-PC0 pins to INT3-INT0 input, set PCDR<PC3D: PC0D> to “0000”(prohibit input), and when driving PC3-PC0 pins to “0”, execute HALT instruction. This setting generates INT3-INT0 inside. If don’t use external interrupt in HALT condition, set like an interrupt don’t generated. (e.g. change port setting)

Figure 3.7.24 Register for Port C

3.7.10 Port F (PF0 to PF5, PF7)

Port F0 to F5 are 6-bit general-purpose I/O ports. Resetting sets PF0 to PF5 to be input ports. It also sets all bits of the output latch register to “1”. In addition to functioning as general-purpose I/O port pins, PF0 to PF5 can also function as the output for I²S0, I²S1. A pin can be enabled for I/O by writing a “1” to the corresponding bit of the Port F Function Register (PFFC).

Port F7 is 1-bit general-purpose output port. In addition to functioning as general-purpose output port, PF7 can also function as the SDCLK output. Resetting sets PF7 to be a SDCLK output port.

(1) Port F0 (I2S0CKO), Port F1 (I2S0DO), Port F2 (I2S0WS), Port F3 (I2S1CKO), Port F4 (I2S1DO), Port F5 (I2S1WS), Port F0 to F5 are general-purpose I/O port. They are also used either I²S. Each pin is below.

	I2Smode (I2S0Module)
PF0	I2S0CKO (Clock output)
PF1	I2S0DO (Data output)
PF2	I2S0WS (Word-select output)

	I2Smode (I2S1Module)
PF4	I2S1CKO (Clock output)
PF5	I2S1DO (Data output)
PF6	I2S1WS (Word-select output)

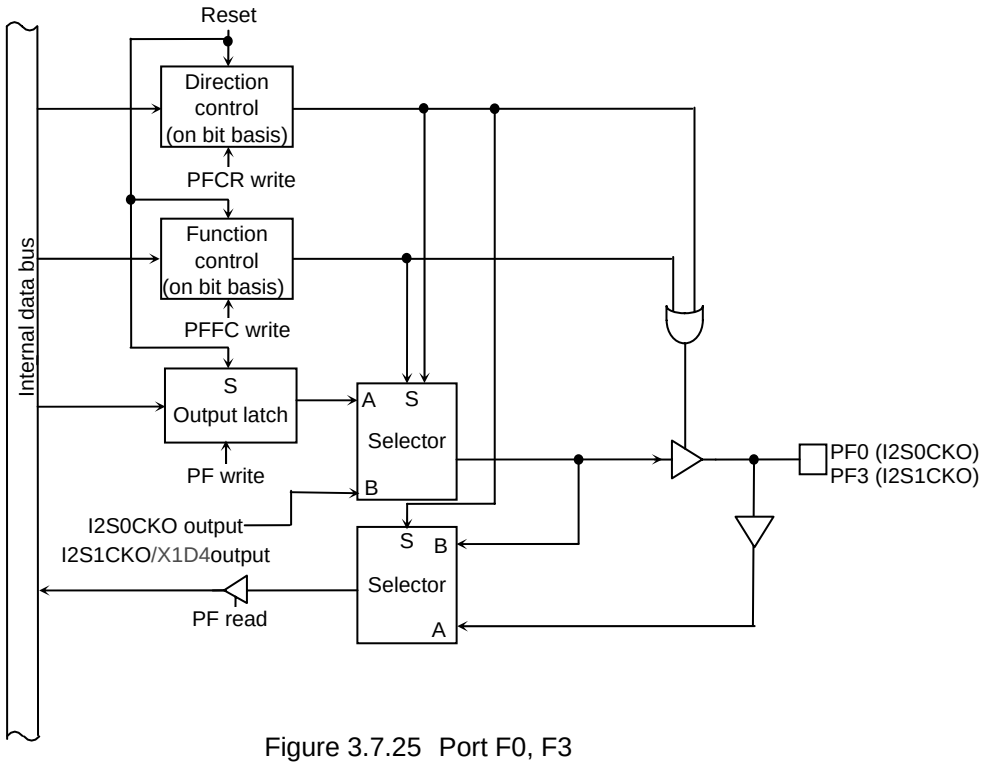


Figure 3.7.25 Port F0, F3

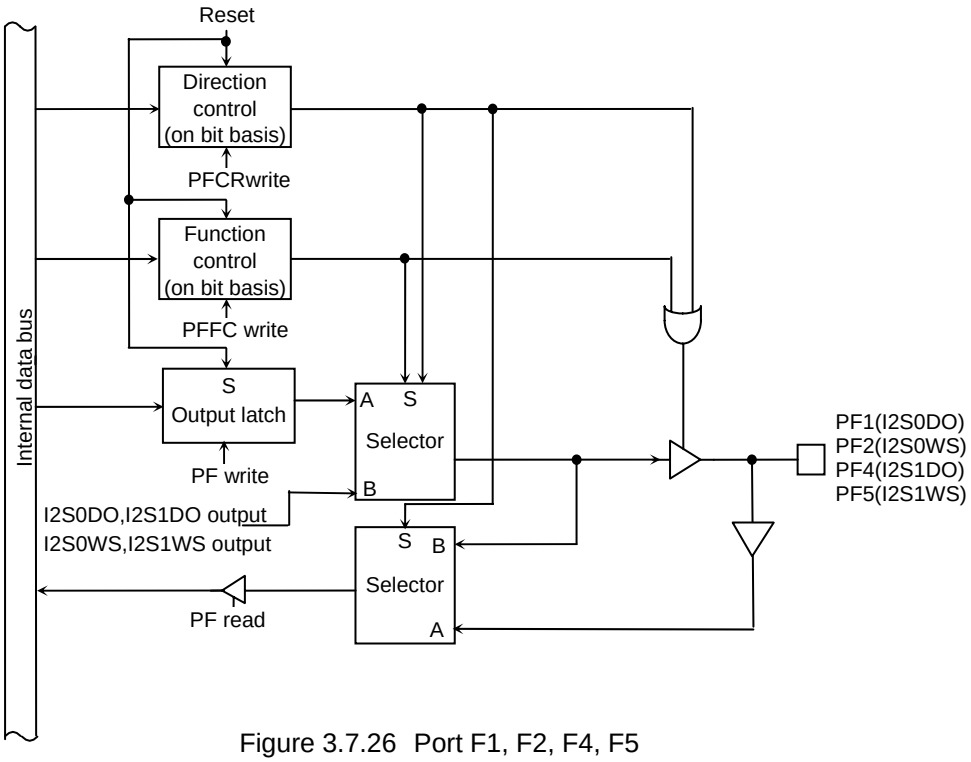


Figure 3.7.26 Port F1, F2, F4, F5

(2) Port F7 (SDCLK),

Port F7 is general-purpose output port. In addition to functioning as general-purpose output port, PF7 can also function as the SDCLK output.

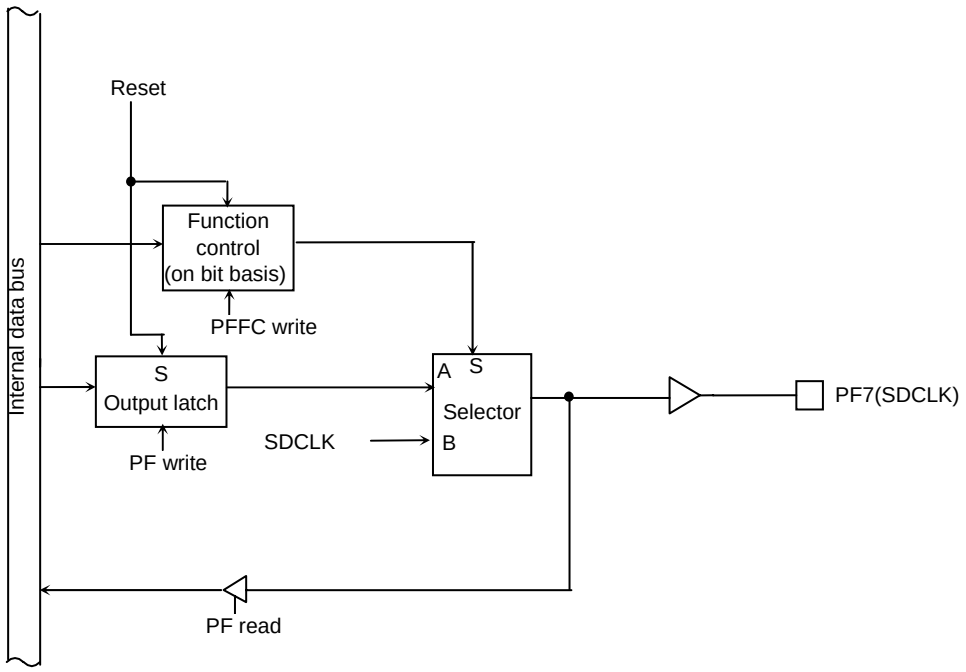


Figure 3.7.27 Port F7

Port F register

PF (003CH)

	7	6	5	4	3	2	1	0
bit Symbol	PF7		PF5	PF4	PF3	PF2	PF1	PF0
Read/Write	R/W		R/W					
After reset	1		Data from external port (Output latch register is set to “1”)					

Port F control register

PFCR (003EH)

	7	6	5	4	3	2	1	0
bit Symbol			PF5C	PF4C	PF3C	PF2C	PF1C	PF0C
Read/Write			W					
After reset			0	0	0	0	0	0
Function			Refer to following table					

Port F function register

PFFC (003FH)

	7	6	5	4	3	2	1	0
bit Symbol	PF7F		PF5F	PF4F	PF3F	PF2F	PF1F	PF0F
Read/Write	W		W					
After reset	1		0	0	0	0	0	0
Function	0: Port 1: SDCLK		Refer to following table					

Port F drive register

PFDR (008FH)

	7	6	5	4	3	2	1	0
bit Symbol	PF7D	PF6D	PF5D	PF4D	PF3D	PF2D	PF1D	PF0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

PF2 setting

<PF2C> / <PF2F>	0	1
0	Input port	Output port
1	I2S0WS output	

PF1 setting

<PF1C> / <PF1F>	0	1
0	Input port	Output port
1	I2S0DO output	

PF0 setting

<PF0C> / <PF0F>	0	1
0	Input port	Output port
1	I2S0CKOoutput	

PF5 setting

<PF5C> / <PF5F>	0	1
0	Input port	Output port
1	I2S1WS output	

PF4 setting

<PF4C> / <PF4F>	0	1
0	Input port	Output port
1	I2S1DO output	

PF3 setting

<PF3C> / <PF3F>	0	1
0	Input port	Output port
1	I2S1CKOoutput	

Note 1: Read-Modify-Write is prohibited for the registers PFCR, PFFC and PFFC2.

Figure 3.7.28 Register for Port F

3.7.11 Port G (PG0 to PG5)

PG0 to PG5 are 6-bit input port and can also be used as the analog input pins for the internal AD converter. PG3 can also be used as ADTRG pin for the AD converter.

PG2, PG3 can also be used as MX, MY pin for Touch screen interface.

(PG) register is prohibited to access by byte. All the instruction (Arithmetic/Logical/ Bit operation and rotate/shift instruction) access by byte are prohibited. Word access is always needed.

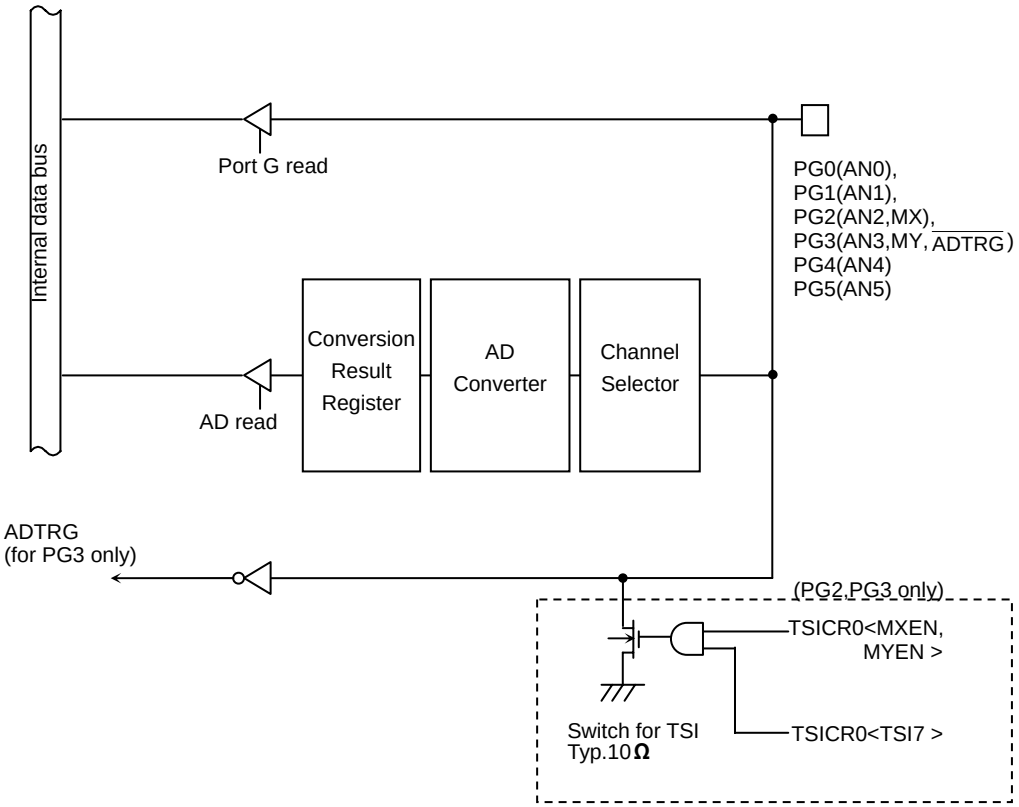


Figure 3.7.29 Port G

Port G register

PG (0040H)

	7	6	5	4	3	2	1	0
Bit Symbol			PG5	PG4	PG3	PG2	PG1	PG0
Read/Write			R					
After reset			Data from external port					

Note: Selection of the input channel of AD converter and ADTRG input mode register is enabled by setting AD converter.

Port G Function register

PGFC (0043H)

	7	6	5	4	3	2	1	0
Bit Symbol					PG3F			
Read/Write					W			
After reset					0			
Function					0: Input port or AN3 1: $\overline{\text{ADTRG}}$			

Port G driver register

PGDR (0090H)

	7	6	5	4	3	2	1	0
Bit Symbol					PG3D	PG2D		
Read/Write					R/W			
After reset					1	1		
Function					Input/Output buffer drive register for standby mode			

Figure 3.7.30 Register for Port G

Note 1: Read-Modify-Write is prohibited for the registers PGFC.

Note 2: (PG) register is prohibited to access by byte. All the instruction (Arithmetic/ Logical/ Bit operation and rotate/ shift instruction) access by byte are prohibited. Word access is always needed.

Example: LD wa, (PG) : Using only “a” register data, and cancel “w” register data.

Note 3: Don’t use PG register at the state that mingles Analog input and Digital input.

3.7.12 Port J (PJ0 to PJ7)

PJ0 to PJ4 and PJ7 are 6-bit output port. Resetting sets the output latch PJ to "1", and they output "1". PJ5 to PJ6 are 2-bit input/output port. In addition to functioning as port, Port J also functions as output pins for SDRAM ($\overline{SDRAS}$, $\overline{SDCAS}$, $\overline{SDWE}$, SDLLDQM, SDLUDQM, and SDCKE), SRAM ($\overline{SRWR}$, $\overline{SRLLB}$ and $\overline{SRLUB}$) and NAND-Flash(NDALE and NDCLE). Above setting is used the function register PJFC.

But Output signal either SDRAM or SRAM for PJ0 to PJ2 are selected automatically according to the setting of memory controller.

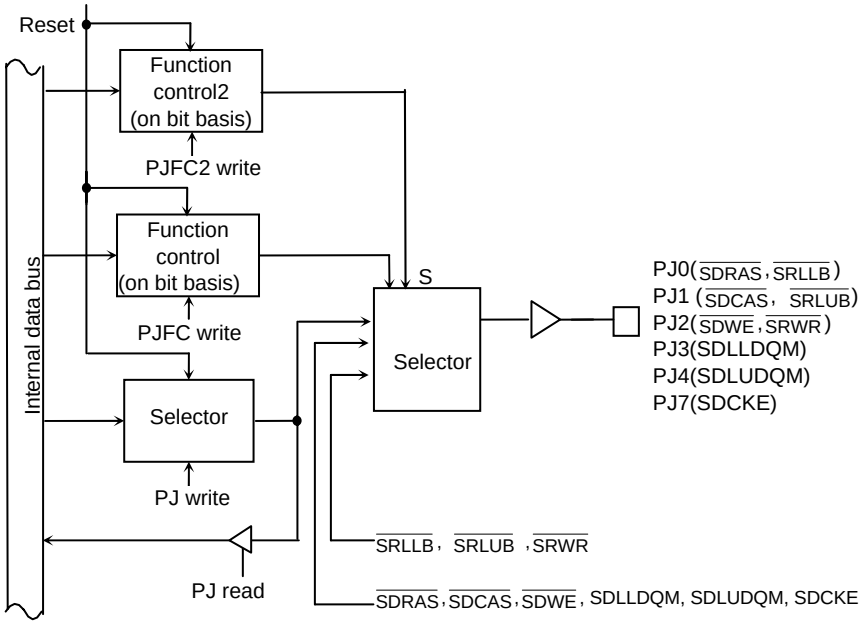


Figure 3.7.31 Port J0 to J4 and J7

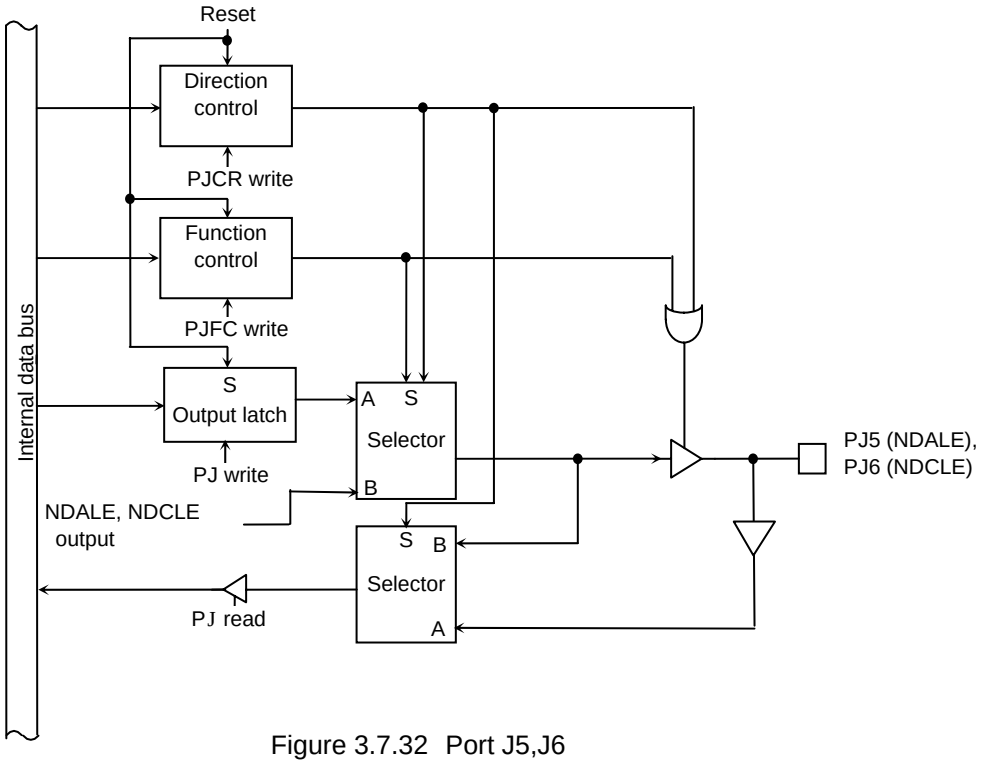


Figure 3.7.32 Port J5,J6

Port J register

PJ (004CH)	7	6	5	4	3	2	1	0
bit Symbol	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
Read/Write	R/W							
After reset	1	Data from external port (Output latch register is set to “1”)		1	1	1	1	1

Port J control register

PJCR (004EH)	7	6	5	4	3	2	1	0
bit Symbol		PJ6C	PJ5C					
Read/Write		W						
After reset		0	0					
Function		0: Input, 1: Output						

Port J function register

PJFC (004FH)	7	6	5	4	3	2	1	0
bit Symbol	PJ7F	PJ6F	PJ5F	PJ4F	PJ3F	PJ2F	PJ1F	PJ0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SDCKE	0: Port 1: NDCLE	0: Port 1: NDALE	0: Port 1: SDLUDQM	0: Port 1: SDLLDQM	0: Port 1: $\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$	0: Port 1: $\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$	0: Port 1: $\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$

Port J drive register

PJDR (0093H)	7	6	5	4	3	2	1	0
bit Symbol	PJ7D	PJ6D	PJ5D	PJ4D	PJ3D	PJ2D	PJ1D	PJ0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note 1: Read-Modify-Write is prohibited for the registers PJCR and PJFC.

Figure 3.7.33 Register for Port J

3.7.13 Port K (PK0 to PK7)

PK0 to PK7 are 8-bit output ports. Resetting sets the output latch PK to “0”, and PK0 to PK7 pins output “0”. In addition to functioning as output port function, Port K also function as output pins for LCD controller (LCP0, LHSYNC, LLOAD, LFR, LVSYNC, and LGOE0 to LGOE2).

Above setting is used the function register PKFC.

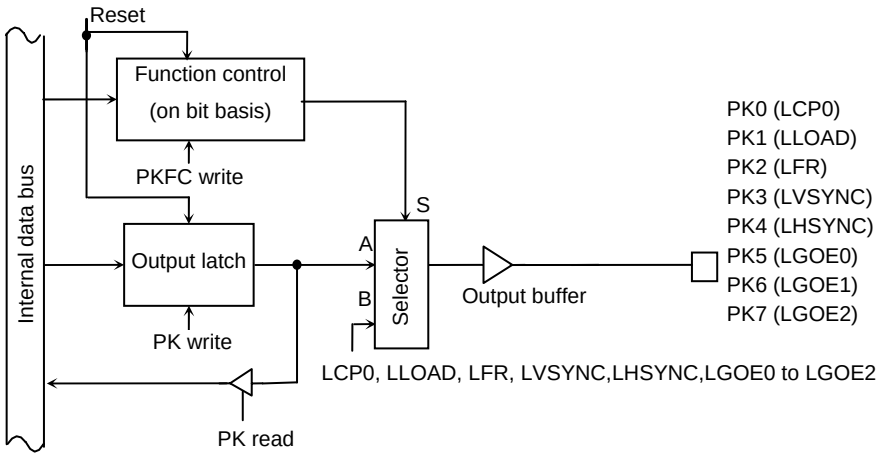


Figure 3.7.34 Port K0 to K7

Port K register

PK (0050H)

	7	6	5	4	3	2	1	0
bit Symbol	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Port K function register

PKFC (0053H)

	7	6	5	4	3	2	1	0
bit Symbol	PK7F	PK6F	PK5F	PK4F	PK3F	PK2F	PK1F	PK0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0:Port 1:LGOE2	0:Port 1:LGOE1	0:Port 1:LGOE0	0:Port 1: LHSYNC	0: Port 1: LVSYNC	0: Port 1: LFR	0: Port 1: LLOAD	0: Port 1: LCP0

Port K drive register

PKDR (0094H)

	7	6	5	4	3	2	1	0
bit Symbol	PK7D	PK6D	PK5D	PK4D	PK3D	PK2D	PK1D	PK0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note 1: Read-Modify-Write is prohibited for the registers PKFC.

Figure 3.7.35 Register for Port K

3.7.14 Port L (PL0 to PL7)

PL0 to PL7 are 8-bit output ports. Resetting sets the output latch PL to “0”, and PL0 to PL7 pins output “0”. In addition to functioning as a general-purpose output port, Port L can also function as a data bus for LCD controller (LD0 to LD7). Above setting is used the function register PLFC.

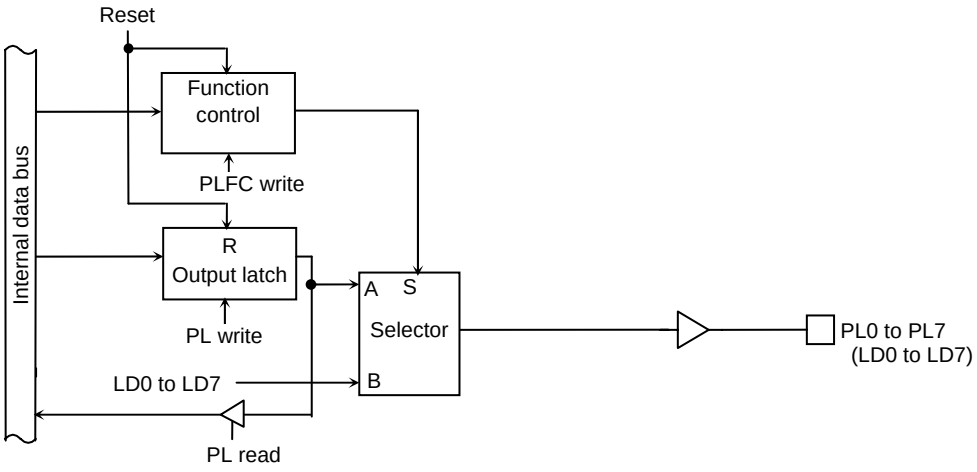


Figure 3.7.36 Port L0 to L7

Port L register

PL (0054H)

	7	6	5	4	3	2	1	0
bit Symbol	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Port L function register

PLFC (0057H)

	7	6	5	4	3	2	1	0
bit Symbol	PL7F	PL6F	PL5F	PL4F	PL3F	PL2F	PL1F	PL0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: Data bus for LCDC (LD7 toLD0)							

Port L drive register

PLDR (0095H)

	7	6	5	4	3	2	1	0
bit Symbol	PL7D	PL6D	PL5D	PL4D	PL3D	PL2D	PL1D	PL0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note 1: Read-Modify-Write is prohibited for the registers PLFC.

Figure 3.7.37 Register for Port L

3.7.15 Port M (PM1, PM2, PM7)

PM1, PM2 and PM7 are 3-bit output ports. Resetting sets the output latch PM to "1", and PM1, PM2 and PM7 pins output "1". In addition to functioning as output ports, Port M also function as output pin for timers (TA1OUT), output pins for RTC alarm ($\overline{\text{ALARM}}$), output pin for melody/alarm generator (MLDALM, $\overline{\text{MLDALM}}$) and Power control pin (PWE). Above setting is used the function register PMFC.

PM1 has two output function which MLDALM and TA1OUT, and PM2 has two output function which $\overline{\text{ALARM}}$ and $\overline{\text{MLDALM}}$. This selection is used PM<PM1>, PM<PM2>.

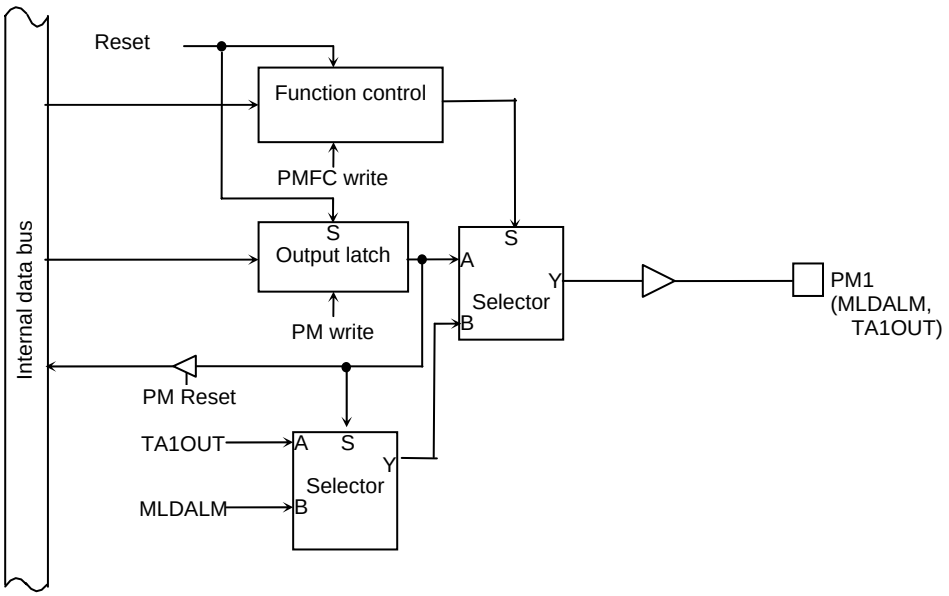


Figure 3.7.38 Port M1

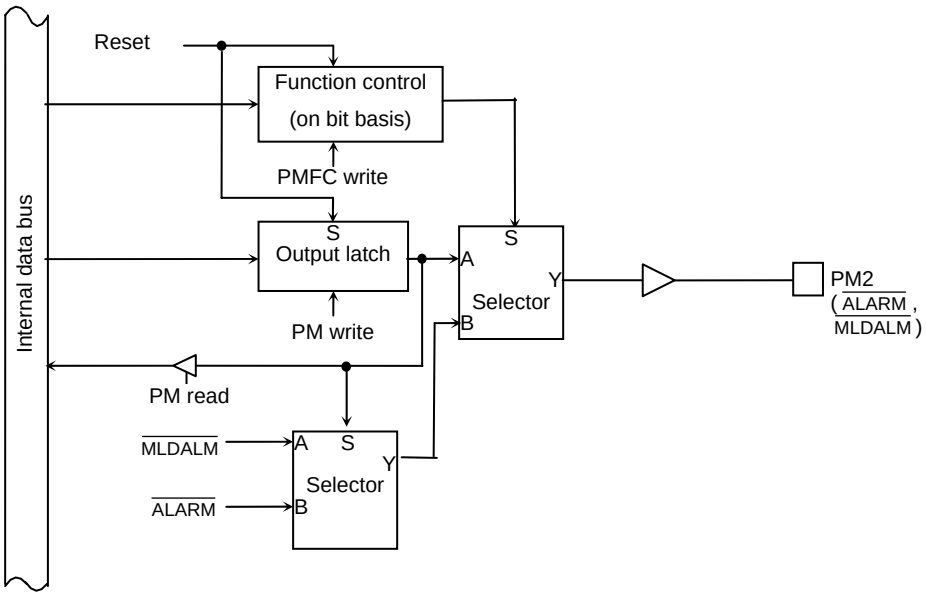


Figure 3.7.39 Port M2

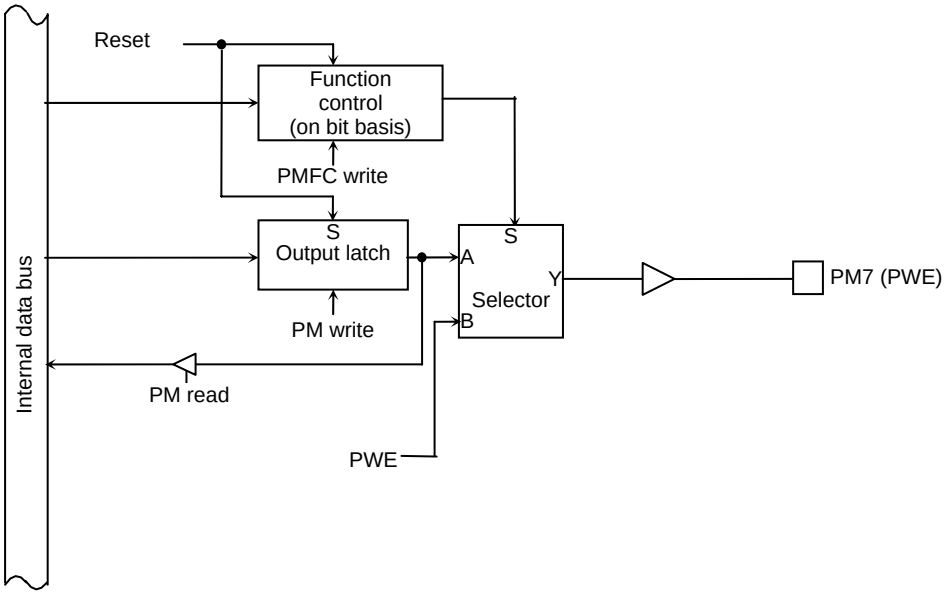


Figure 3.7.40 Port M7

Port M register

PM (0058H)

	7	6	5	4	3	2	1	0
bit Symbol	PM7					PM2	PM1	
Read/Write	R/W					R/W		
After reset	1					1	1	

Port M function register

PMFC (005BH)

	7	6	5	4	3	2	1	0
bit Symbol	PM7F					PM2F	PM1F	
Read/Write	W					W		
After reset	0					0	0	
Function	0: Port 1: PWE					0: Port 1: $\overline{\text{ALARM}}$ at <PM2>=1, $\overline{\text{MLDALM}}$ at <PM2>=0	0: Port 1: MLDALM at <PM1>=1, TA1OUT at <PM1>=0	

Port M drive register

PMDR (0096H)

	7	6	5	4	3	2	1	0
bit Symbol	PM7D					PM2D	PM1D	
Read/Write	R/W					R/W		
After reset	1					1	1	
Function	Input/Output buffer drive register for standby mode					Input/Output buffer drive register for standby mode		

Note 1: Read-Modify-Write is prohibited for the registers PMFC.

Figure 3.7.41 Register for Port M

3.7.16 Port N (PN0 to PN7)

PN0 to PN7 are 8-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets Port N to an input port. In addition to functioning as a general-purpose I/O port, Port N can also function as interface pin for key-board (KO0 to KO7). This function can set to open-drain type output buffer.

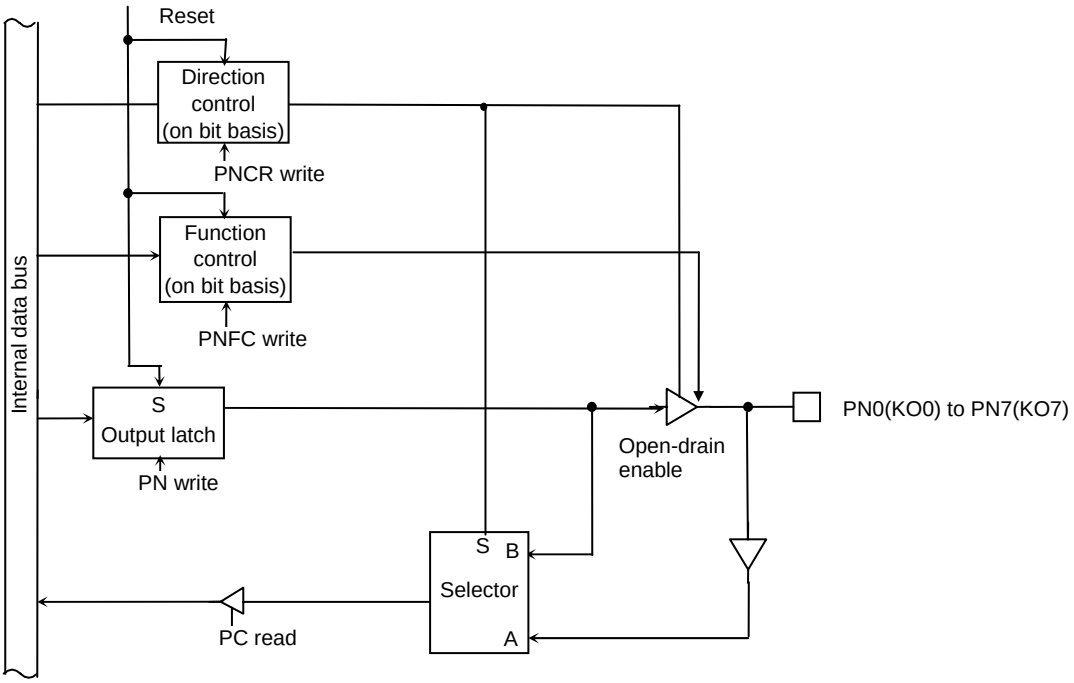


Figure 3.7.42 Port N

Port N register

PN (005CH)

	7	6	5	4	3	2	1	0
bit Symbol	PN7	PN6	PN5	PN4	PN3	PN2	PN1	PN0
Read/Write	R/W							
After reset	Data from external port (Output latch register is set to “1”)							

Port N control register

PNCR (005EH)

	7	6	5	4	3	2	1	0
bit Symbol	PN7C	PN6C	PN5C	PN4C	PN3C	PN2C	PN1C	PN0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port N function register

PNFC (005FH)

	7	6	5	4	3	2	1	0
bit Symbol	PN7F	PN6F	PN5F	PN4F	PN3F	PN2F	PN1F	PN0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: CMOS output 1: Open-drain output							

Port N drive register

PNDR (0097H)

	7	6	5	4	3	2	1	0
bit Symbol	PN7D	PN6D	PN5D	PN4D	PN3D	PN2D	PN1D	PN0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note 1: Read-Modify-Write is prohibited for the registers PNCR and PNFC.

Figure 3.7.43 Register for Port N

3.7.17 Port P (PP1 to PP7)

Port P1 to P5 are 6-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port P1 to P5 to input port and output latch to “0”. In addition to functioning as general-purpose I/O port pins, P0 to P5 can also function as output pin for timers (TA3OUT, TA5OUT, TA7OUT), input pin for timers (TB0IN0, TB1IN0), input pin for external interruption (INT5 to INT7).

Port P6 and P7 are 2-bit output port. Resetting sets output latch to “0”.In addition to functioning as output port, PP6 and PP7 can also function as output pin for timers (TB0OUT0, TB1OUT1).

Above setting is used the control register PPCR and function register PPFC.

Edge select of external interruption establishes it with IIMC register, which there is in interruption controller.

In port setting, if 16 bit timer input is selected and capture control is executed, INT6 and INT7 don't depend on IIMC1 register setting. INT6 and INT7 operate by setting TBnMOD<TBnCPM1:0>.

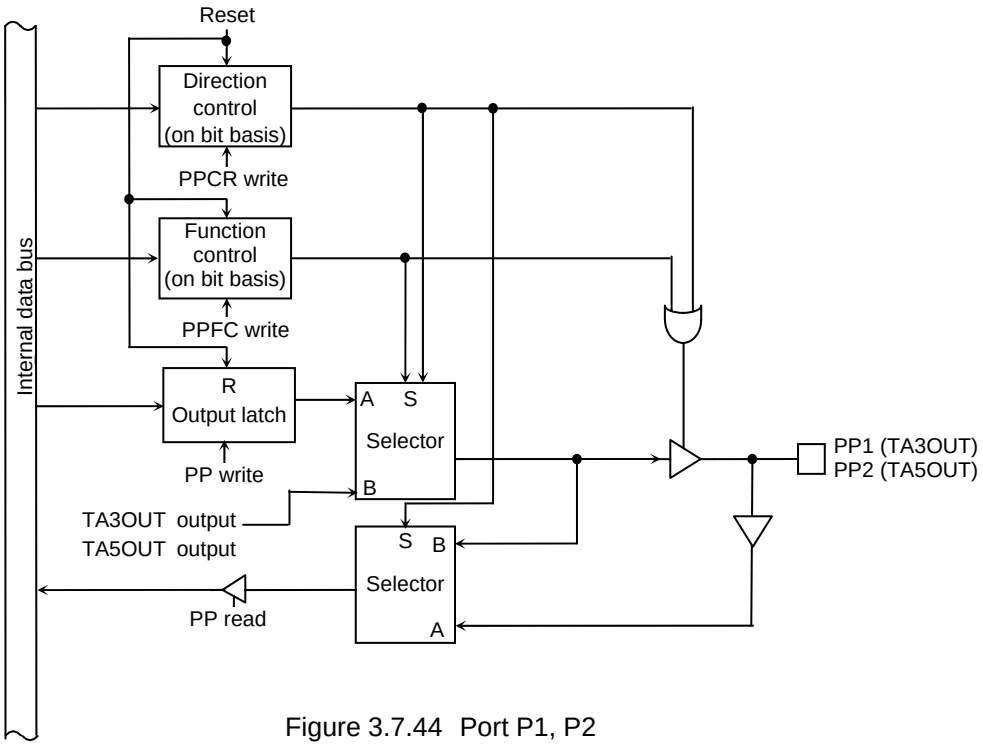


Figure 3.7.44 Port P1, P2

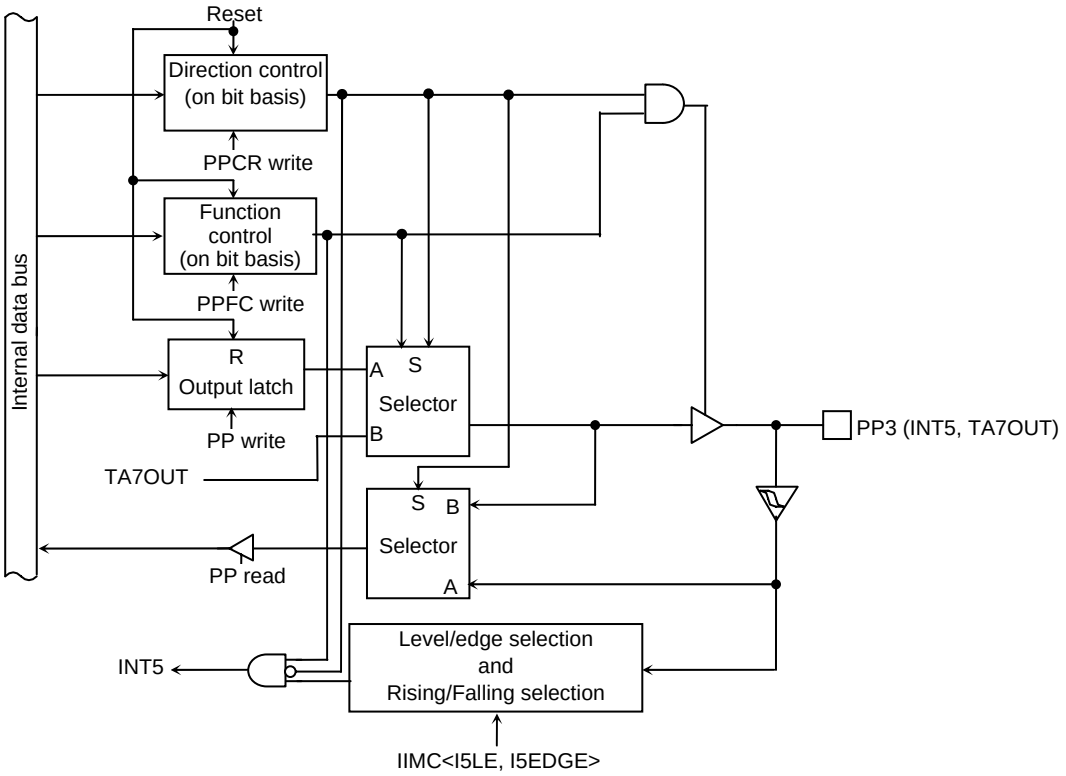


Figure 3.7.45 Port P3

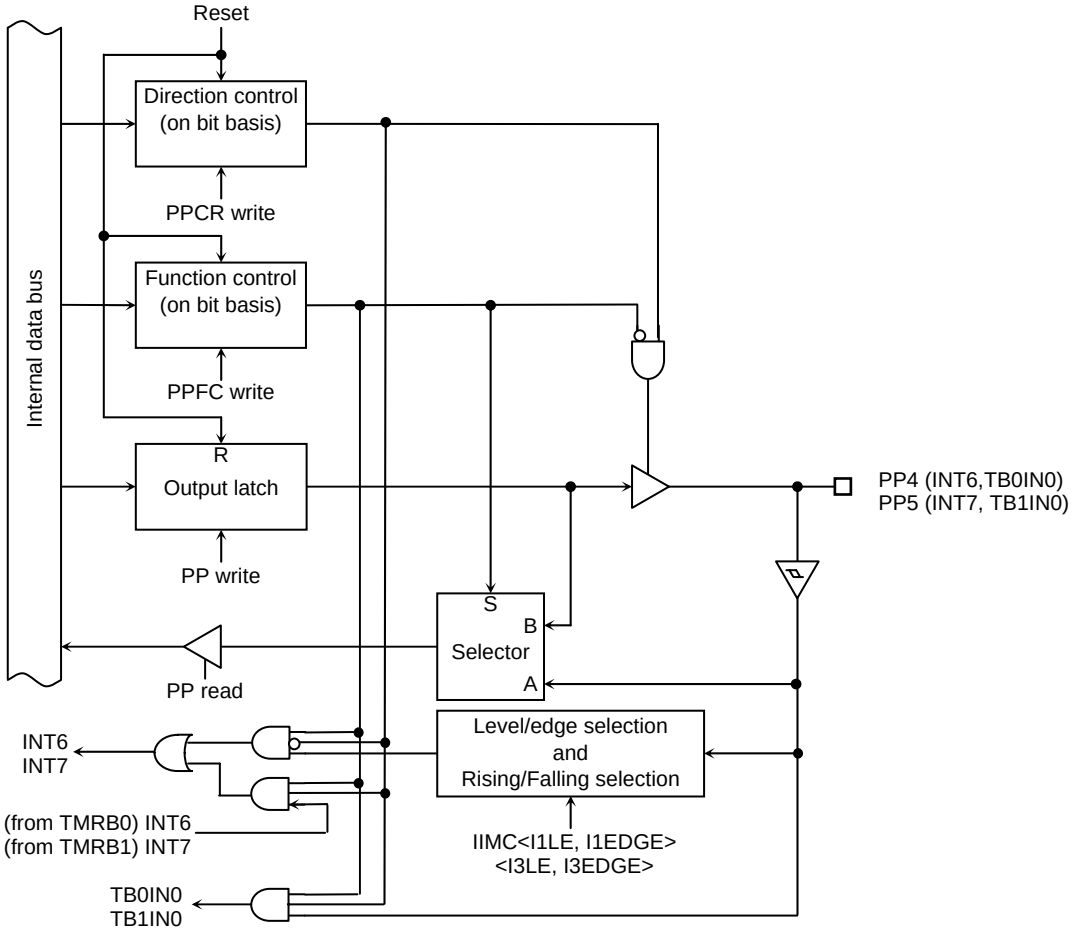


Figure 3.7.46 Port P4,P5

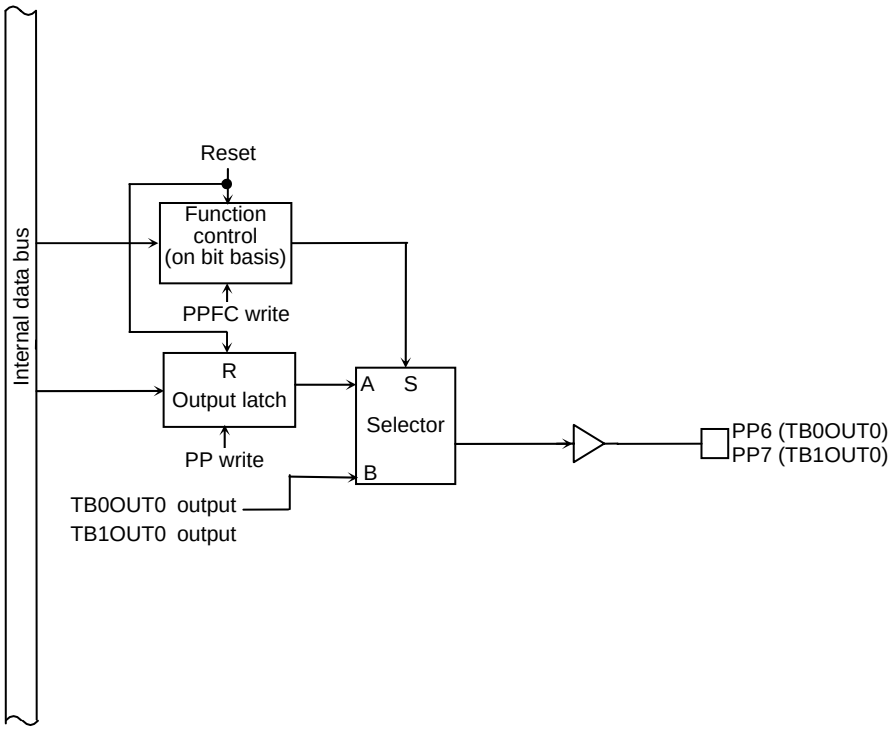


Figure 3.7.47 Port P6, P7

Port P register

PP (0060H)

	7	6	5	4	3	2	1	0
bit Symbol	PP7	PP6	PP5	PP4	PP3	PP2	PP1	
Read/Write	R/W							
After reset	0	0	Data from external port (Output latch register is cleared to “0”)					

Port P control register

PPCR (0062H)

	7	6	5	4	3	2	1	0
bit Symbol			PP5C	PP4C	PP3C	PP2C	PP1C	
Read/Write			W					
After reset			0	0	0	0	0	
Function			0: Input 1: Output					

Port P function register

PPFC (0063H)

	7	6	5	4	3	2	1	0
bit Symbol	PP7F	PP6F	PP5F	PP4F	PP3F	PP2F	PP1F	
Read/Write	W							
After reset	0	0	0	0	0	0	0	
Function	0:Port 1:TB1OUT0	0:Port 1:TB0OUT0	Refer to following table					

Port P drive register

PPDR (0098H)

	7	6	5	4	3	2	1	0
bit Symbol	PP7D	PP6D	PP5D	PP4D	PP3D	PP2D	PP1D	
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	
Function	Input/Output buffer drive register for standby mode							

PP3 setting

<PP3C> / <PP3F>	0	1
0	Input port	Output port
1	INT5 input	TA7OUT output

PP2 setting

<PP2C> / <PP2F>	0	1
0	Input port	Output port
1	Don’t setting	TA5OUT output

PP1 setting

<PP1C> / <PP1F>	0	1
0	Input port	Output port
1	Don’t setting	TA3OUT output

PP5 setting

<PP5C> / <PP5F>	0	1
0	Input port	Output port
1	INT7 input	TB1IN0 input

PP4 setting

<PP4C> / <PP4F>	0	1
0	Input port	Output port
1	INT6 input	TB0IN0 input

Note1: Read-Modify-Write is prohibited for the registers PPCR, PPFC.

Note2: When setting PP5, PP4, PP3 pins to INT7,INT6,INT5 input, set PPDR<PP5D:3D> to “0000” (prohibit input), and when driving PP5,PP4,PP3 pins to “0”, execute HALT instruction. This setting generates INT7, INT6, and INT5 inside. If don’t using external interrupt in HALT condition, set like an interrupt don’t generated.

Figure 3.7.48 Register for Port P

3.7.18 Port R (R0 to R3)

Port R0 to R3 are 4-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port R0 to R3 to input port and output latch to “0”. In addition to functioning as general-purpose I/O port pins, PR0 to PR3 can also function as SPI controller pin (SPCLK, $\overline{\text{SPCS}}$, SPDO and SPDI).

Above setting is used the control register PRCR and function register PRFC.

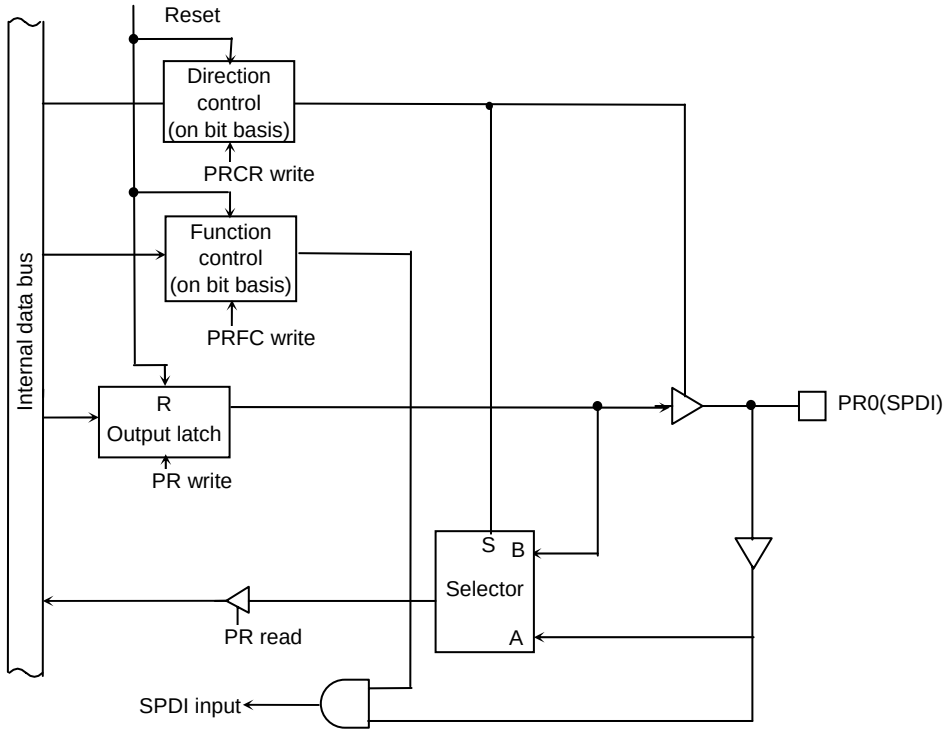


Figure 3.7.49 Port R0

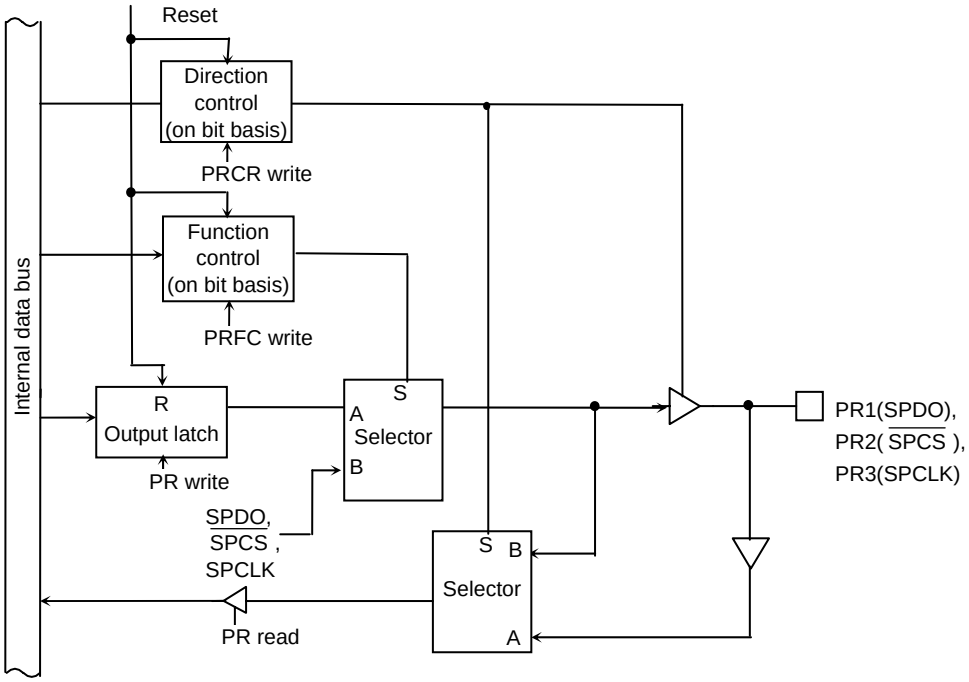


Figure 3.7.50 Port R1 to R3

Port R register

PR (0064H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3	PR2	PR1	PR0
Read/Write					R/W			
After reset					Data from external port (Output latch register is cleared to “0”)			

Port R control register

PRCR (0066H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3C	PR2C	PR1C	PR0C
Read/Write					W			
After reset					0	0	0	0
Function					0: Input, 1: Output			

Port R function register

PRFC (0067H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3F	PR2F	PR1F	PR0F
Read/Write					W			
After reset					0	0	0	0
Function					0: Port 1: SPCLK	0: Port 1: $\overline{\text{SPCS}}$	0: Port 1: SPDO	0: Port 1: SPDI

PR1 setting

<PR1C> / <PR1F>	0	1
0	Input port	Output port
1	(Reserved)	SPDO output

PR0 setting

<PR0C> / <PR0F>	0	1
0	Input port	Output port
1	SPDI input	(Reserved)

PR3setting

<PR3C> / <PR3F>	0	1
0	Input port	Output port
1	(Reserved)	SPCLK output

PR2 setting

<PR2C> / <PR2F>	0	1
0	Input port	Output port
1	(Reserved)	$\overline{\text{SPCS}}$ Output

Port R drive register

PRDR (0099H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3D	PR2D	PR1D	PR0D
Read/Write					R/W			
After reset					1	1	1	1
Function					Input/Output buffer drive register for standby mode			

Note: Read-Modify-Write is prohibited for the registers PRCR, PRFC.

Figure 3.7.51 Register for Port R

3.7.19 Port T (PT0 to PT7)

Port T0 to T7 are 8-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port T0 to T7 to input port and output latch to “0”. In addition to functioning as general-purpose I/O port pins, PT0 to PT7 can also function as data bus pin for LCD controller (LD8 to LD15).

Above setting is used the control register PTCR and function register PTFC.

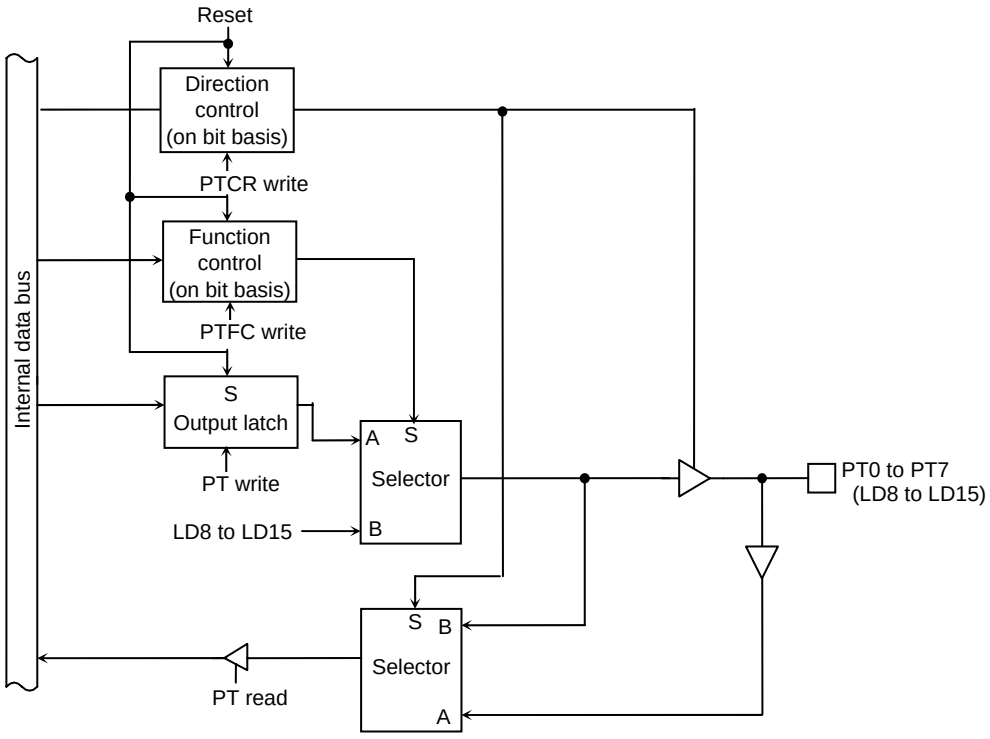


Figure 3.7.52 Port T0 to T7

Port T register

PT (00A0H)

	7	6	5	4	3	2	1	0
bit Symbol	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”)							

Port T control register

PTCR (00A2H)

	7	6	5	4	3	2	1	0
bit Symbol	PT7C	PT6C	PT5C	PT4C	PT3C	PT2C	PT1C	PT0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port T function register

PTFC (00A3H)

	7	6	5	4	3	2	1	0
bit Symbol	PT7F	PT6F	PT5F	PT4F	PT3F	PT2F	PT1F	PT0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: Data bus for LCDC (LD15 to LD8)							

Port T drive register

PTDR (009BH)

	7	6	5	4	3	2	1	0
bit Symbol	PT7D	PT6D	PT5D	PT4D	PT3D	PT2D	PT1D	PT0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note1: Read-Modify-Write is prohibited for the registers PTCR, PTFC.

Note2: When PT is used as LD15 to LD8, set applicable PTnC to”1”.

Figure 3.7.53 Register for Port T

3.7.20 Port U (PU0 to PU7)

Port U0 to U7 are 8-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port U0 to U7 to input port and output latch to “0”. In addition to functioning as general-purpose I/O port pins, PU0 to PU7 can also function as data bus pin for LCD controller (LD16 to LD23) and SDCLK input function.

Above setting is used the control register PUCR and function register PUFC.

In addition to functioning as above function, PU7 can also function as communication for debug mode (EO_TRGOUT). These functions are operated when it is started in debug mode. In this case, PU7 can not be used as LD23 function.

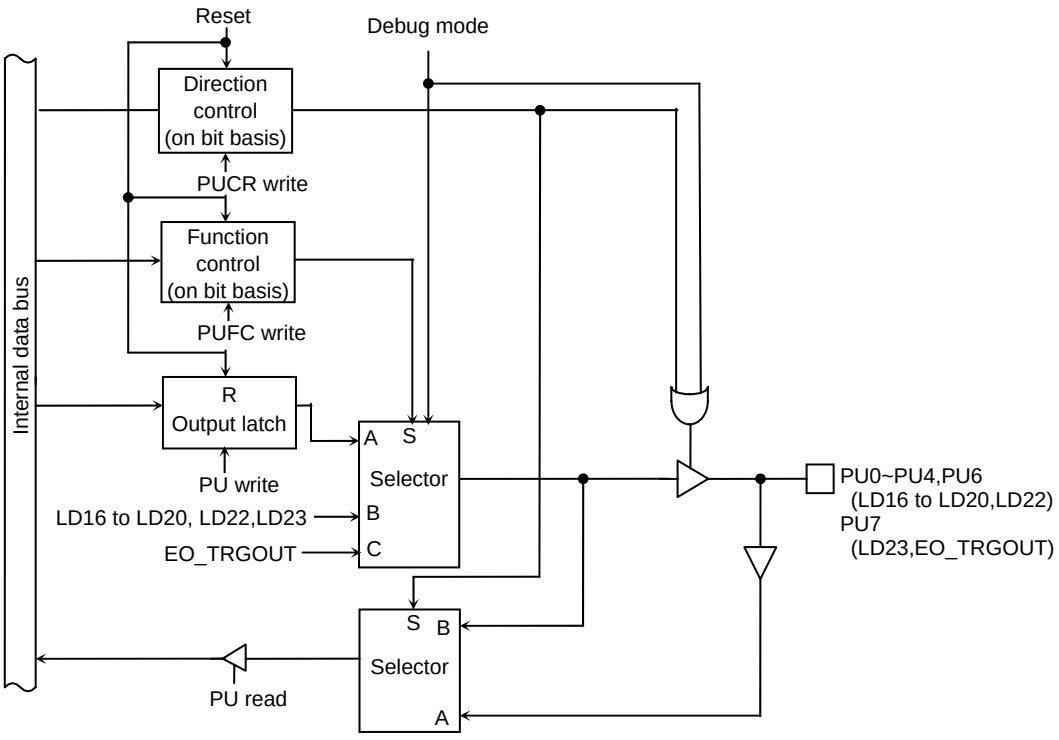


Figure 3.7.54 Port U0 to U4 , U6 , U7

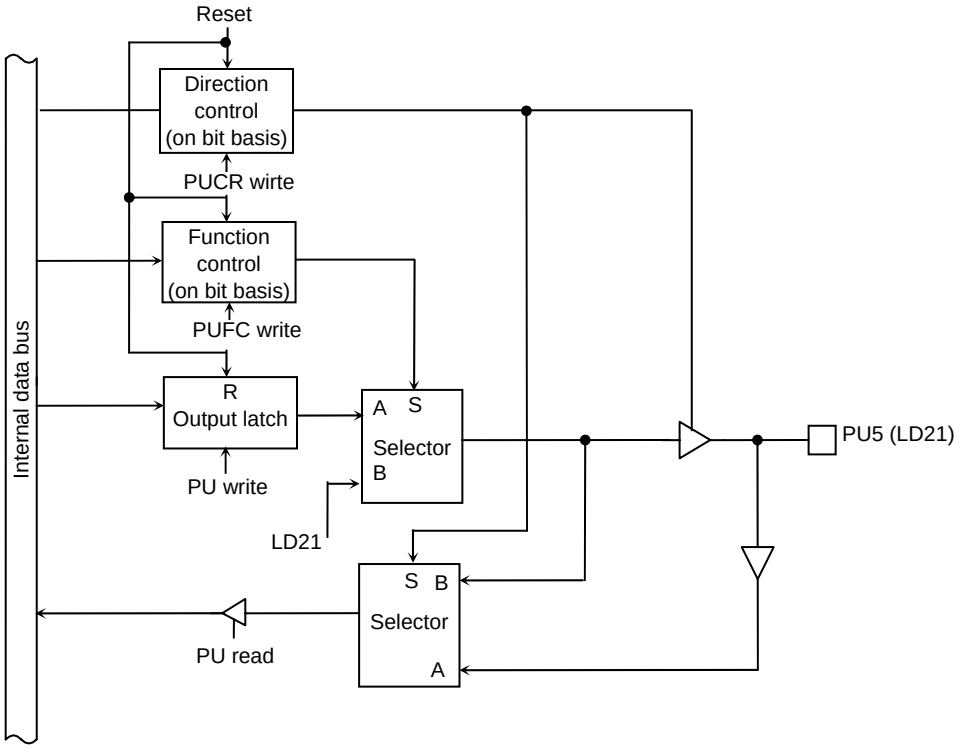


Figure 3.7.55 Port U5

Port U register

PU (00A4H)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”)							

Port U control register

PUCR (00A6H)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7C	PU6C	PU5C	PU4C	PU3C	PU2C	PU1C	PU0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port U function register

PUFC (00A7H)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7F	PU6F	PU5F	PU4F	PU3F	PU2F	PU1F	PU0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: LD23	0: Port 1: LD22	0: Port 1: LD21@ <PU5C>=1	0: Port 1: LD20	0: Port 1: LD19	0: Port 1: LD18	0: Port 1: LD17	0: Port 1: LD16

Note: When PU is used as LD23 to LD16, set applicable PUnC to “1”.

Port U drive register

PUDR (009CH)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7D	PU6D	PU5D	PU4D	PU3D	PU2D	PU1D	PU0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note1: Read-Modify-Write is prohibited for the registers PUCR, PUFC.

Note2: When use PU as LD23 to LD16, set PUnC to “1”. When use PU5 as LD21, set PU5C to “1”.

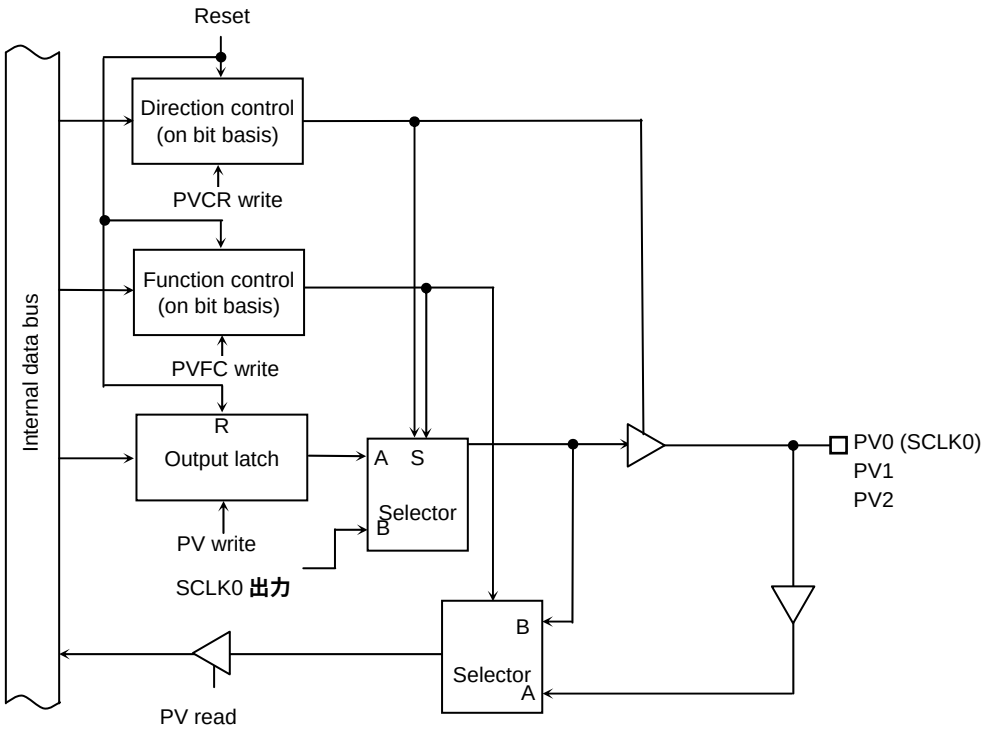
Figure 3.7.56 Register for Port U

3.7.21 Port V (PV0 to PV4, PV6, PV7)

Port V0 to V2, V6 and V7 are 5-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port V0 to V2, V6 and V7 to input port and output latch to “0”. In addition to functioning as general-purpose I/O port pins, PV can also function as input or output pin for SBI (SDA, SCL) and output for SIO(SCLK0) (Note).

Above setting is used the control register PVCR and function register PVFC.

Port V3 and V4 are 2-bit general-purpose output ports. Resetting clear port V3 and V4 to output latch to “0”.



Note: SIO function support function that input clock from SCLK0, basically. However, if setting to PV0 pin, this function supports only the output function.

Figure 3.7.57 Port V0 to V2

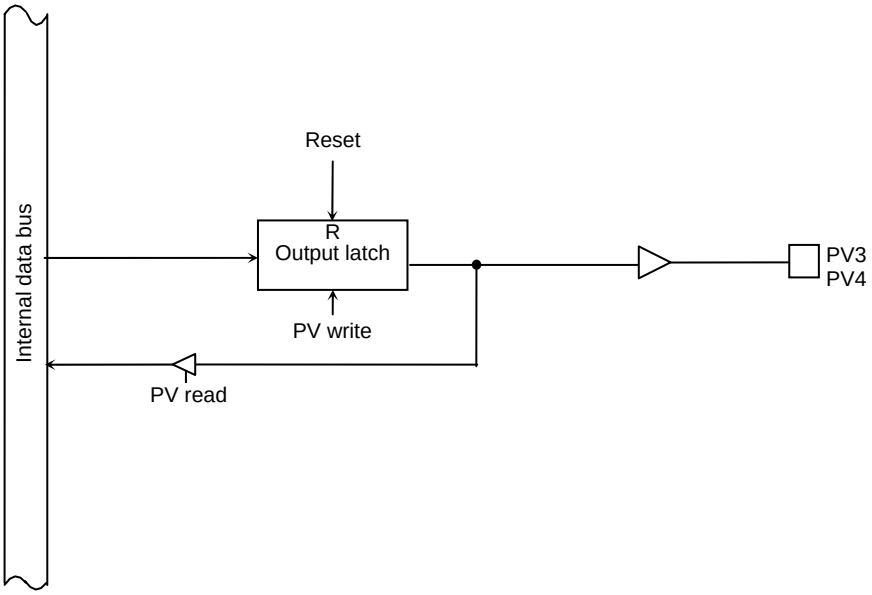


Figure 3.7.58 Port V3, V4

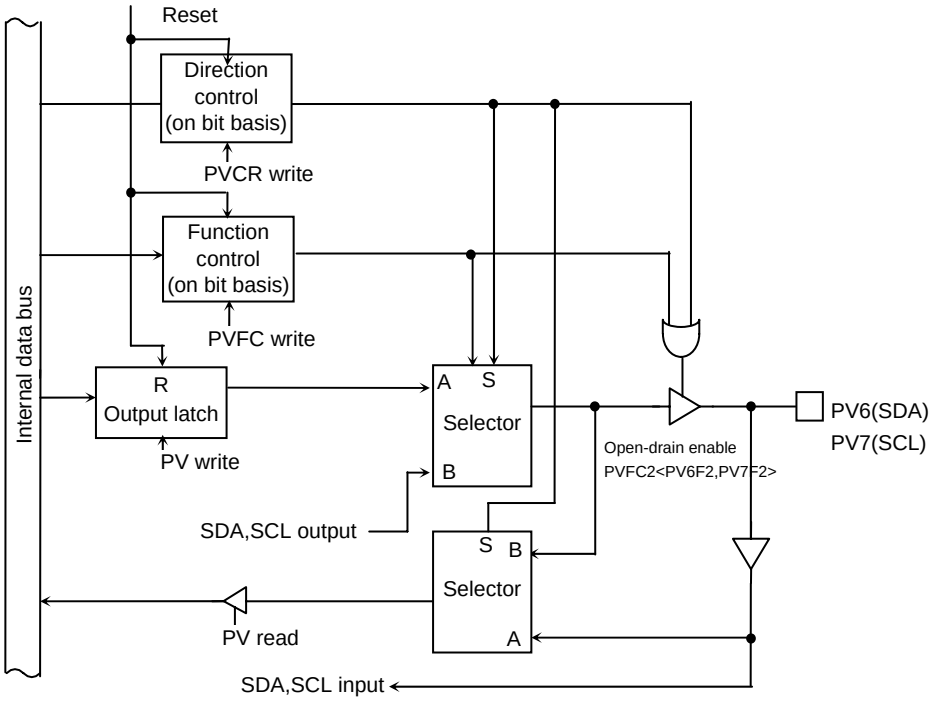


Figure 3.7.59 Port V6, V7

Port V register

PV (00A8H)

	7	6	5	4	3	2	1	0
bit Symbol	PV7	PV6		PV4	PV3	PV2	PV1	PV0
Read/Write	R/W			R/W				
After reset	Data from external port (Output latch register is cleared to “0”)			Data from external port (Output latch register is cleared to “0”)				

Port V control register

PVCR (00AAH)

	7	6	5	4	3	2	1	0
bit Symbol	PV7C	PV6C				PV2C	PV1C	PV0C
Read/Write						W		
After reset	0	0				0	0	0
Function	0: Input 1: Output					0: Input 1: Output		

Port V function register

PVFC (00ABH)

	7	6	5	4	3	2	1	0
bit Symbol	PV7F	PV6F				PV2F	PV1F	PV0F
Read/Write	W					W		
After reset	0	0				0	0	0
Function	Refer to following table					Refer to following table		

PV2 setting

<PV2C> / <PV2F>	0	1
0	Input port	Output port
1	Reserved	Reserved

PV1 setting

<PV1C> / <PV1F>	0	1
0	Input port	Output port
1	Reserved	Reserved

PV0 setting

<PV0C> / <PV0F>	0	1
0	Input port	Output port
1	Reserved	SCLK0 output

Note: SCLK0 is only output.

PV7 setting

<PV7C> / <PV7F>	0	1
0	Input port	Output port
1	Reserved	SCL I/O

PV6 setting

<PV6C> / <PV6F>	0	1
0	Input port	Output port
1	Reserved	SDA I/O

Port V function register 2

PVFC2 (00A9H)

	7	6	5	4	3	2	1	0
bit Symbol	PV7F2	PV6F2						
Read/Write	W	W						
After reset	0	0						
Function	0:CMOS 1:Open -drain	0:CMOS 1:Open -drain						

Port V drive register

PVDR (009DH)

	7	6	5	4	3	2	1	0
bit Symbol	PV7D	PV6D		PV4D	PV3D	PV2D	PV1D	PV0D
Read/Write	R/W			R/W				
After reset	1	1		1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note: Read-Modify-Write is prohibited for the registers PVCR, PVFC and PVFC2.

Figure 3.7.60 Register for Port V

3.7.22 Port W (PW0 to PW7)

Port W0 to W7 are 8-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port W0 to W7 to input port and output latch to “0”.

Above setting is used the control register PWCR and function register PWFC.

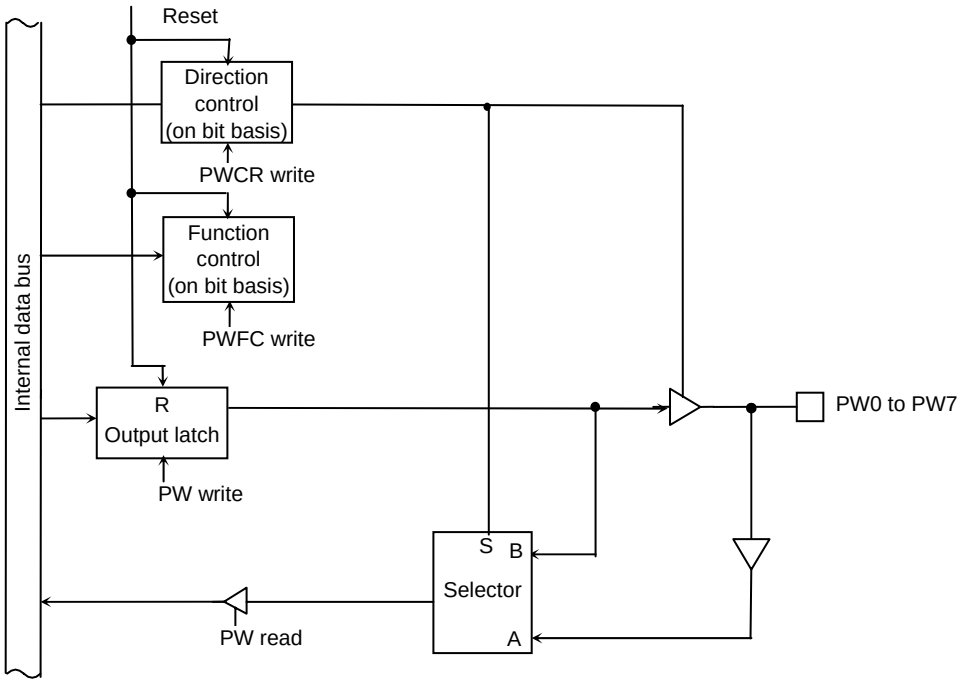


Figure 3.7.61 Port W0 to W7

Port W register

PW (00ACH)

	7	6	5	4	3	2	1	0
bit Symbol	PW7	PW6	PW5	PW4	PW3	PW2	PW1	PW0
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”)							

Port W control register

PWCR (00AEH)

	7	6	5	4	3	2	1	0
bit Symbol	PW7C	PW6C	PW5C	PW4C	PW3C	PW2C	PW1C	PW0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port W function register

PWFC (00AFH)

	7	6	5	4	3	2	1	0
bit Symbol	PW7F	PW6F	PW5F	PW4F	PW3F	PW2F	PW1F	PW0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: Reserved							

Port W drive register

PWDR (009EH)

	7	6	5	4	3	2	1	0
bit Symbol	PW7D	PW6D	PW5D	PW4D	PW3D	PW2D	PW1D	PW0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note1: Read-Modify-Write is prohibited for the registers PWCR, PWFC.

Figure 3.7.62 Register for Port W

3.7.23 Port X (PX4, PX5 and PX7)

Port X5 and X7 are 2-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port X5 and X7 to input port and output latch to “0”. In addition to functioning as general-purpose I/O port pins, PX5 and PX7 can also function as USB clock input pin (X1USB).

Above setting is used the control register PXCR and function register PXFC.

Port X4 is 1-bit general-purpose output port. Resetting sets output latch to “0”. In addition to functioning as general-purpose output port, PX4 can also function as system clock output pin (CLKOUT) and output pin (LDIV). This setting is used the PX register and function register PXFC.

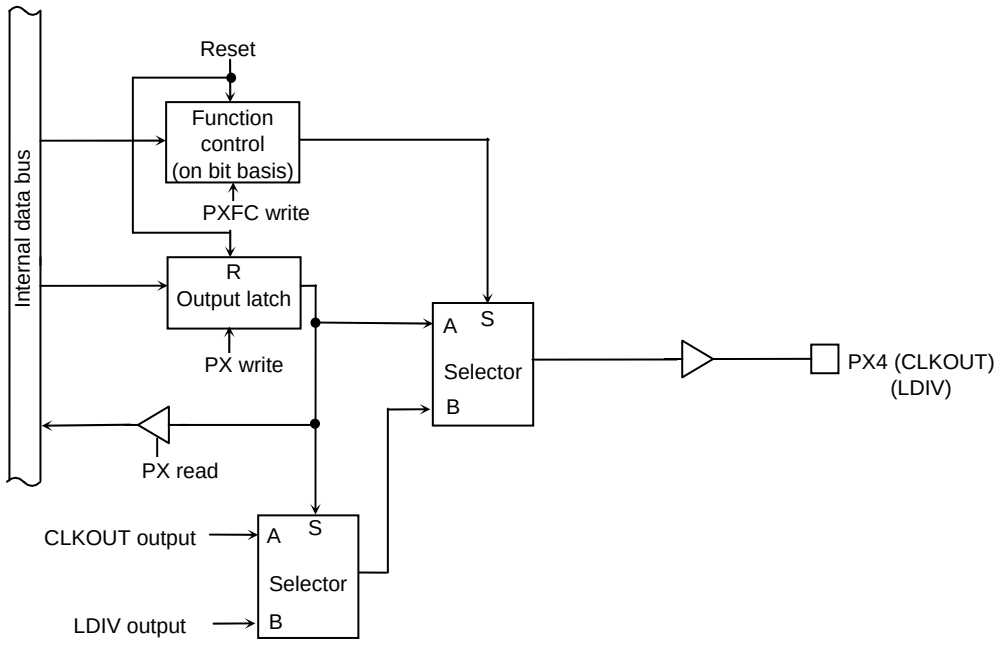


Figure 3.7.63 Port X4

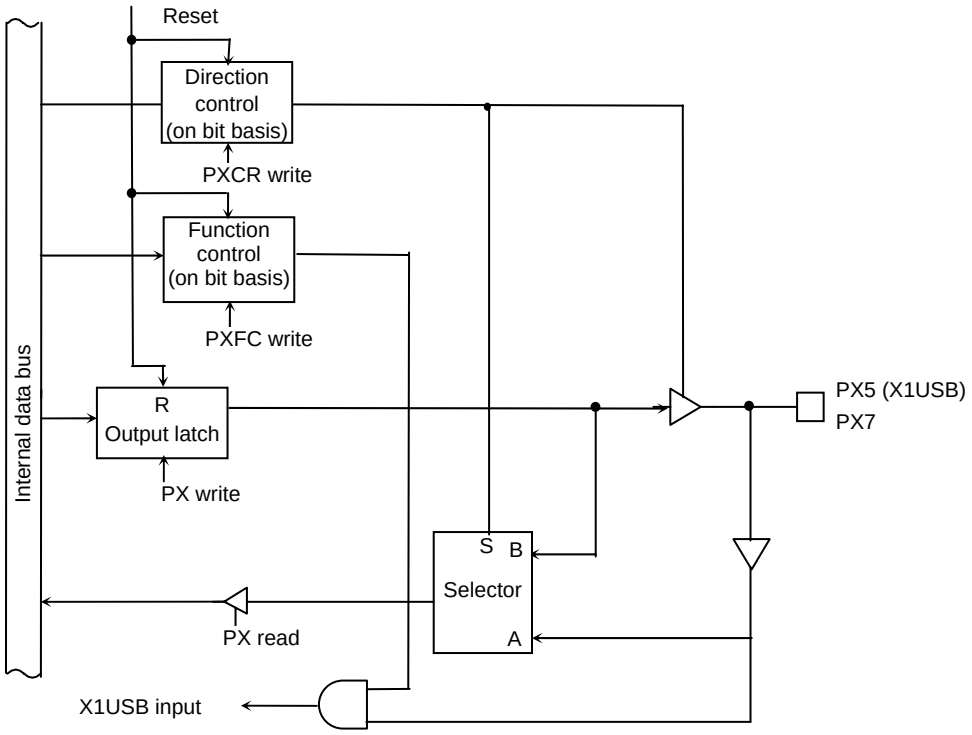


Figure 3.7.64 Port X5, X7

Port X register

PX (00B0H)	7	6	5	4	3	2	1	0
bit Symbol	PX7		PX5	PX4				
Read/Write	R/W		R/W					
After reset	Data from external port (Output latch register is cleared to “0”)							

Port W control register

PXCR (00B2H)	7	6	5	4	3	2	1	0
bit Symbol	PX7C		PX5C					
Read/Write	W		W					
After reset	0		0					
Function	0: Input 1: Output		0: Input 1: Output					

Port W function register

PXFC (00B3H)	7	6	5	4	3	2	1	0
bit Symbol	PX7F		PX5F	PX4F				
Read/Write	W		W					
After reset	0		0	0				
Function	0:Port 1: Reserved		0:Port 1:X1USB input	0:Port 1:CLKOUT at <PX4> = 0 LDIV at <PX4> = 1				

Port W drive register

PXDR (009FH)	7	6	5	4	3	2	1	0
bit Symbol	PXD7		PXD5	PXD4				
Read/Write	R/W		R/W					
After reset	1		1	1				
Function	Input/Output buffer drive register for standby mode							

Note: Read-Modify-Write is prohibited for the registers PWCR, PWFC.

Figure 3.7.65 Register for Port X

3.7.24 Port Z (PZ0 to PZ7)

Port Z0 to Z7 are 8-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port Z0 to Z7 to input port and output latch to “0”.

In addition to functioning as general-purpose I/O port function, Port Z can also function as communication for debug mode (EI_PODDATA, EI_SYNCLK, EI_PODREQ, EI_REFCLK, EI_TRGIN, EI_COMRESET, EO_MCUDATA and EO_MCUREQ). These functions are operated when it is started in debug mode. (There is not Function register in this port. When $\overline{\text{DBGE}}$ is set to “0”, this port set to debug communication function.)

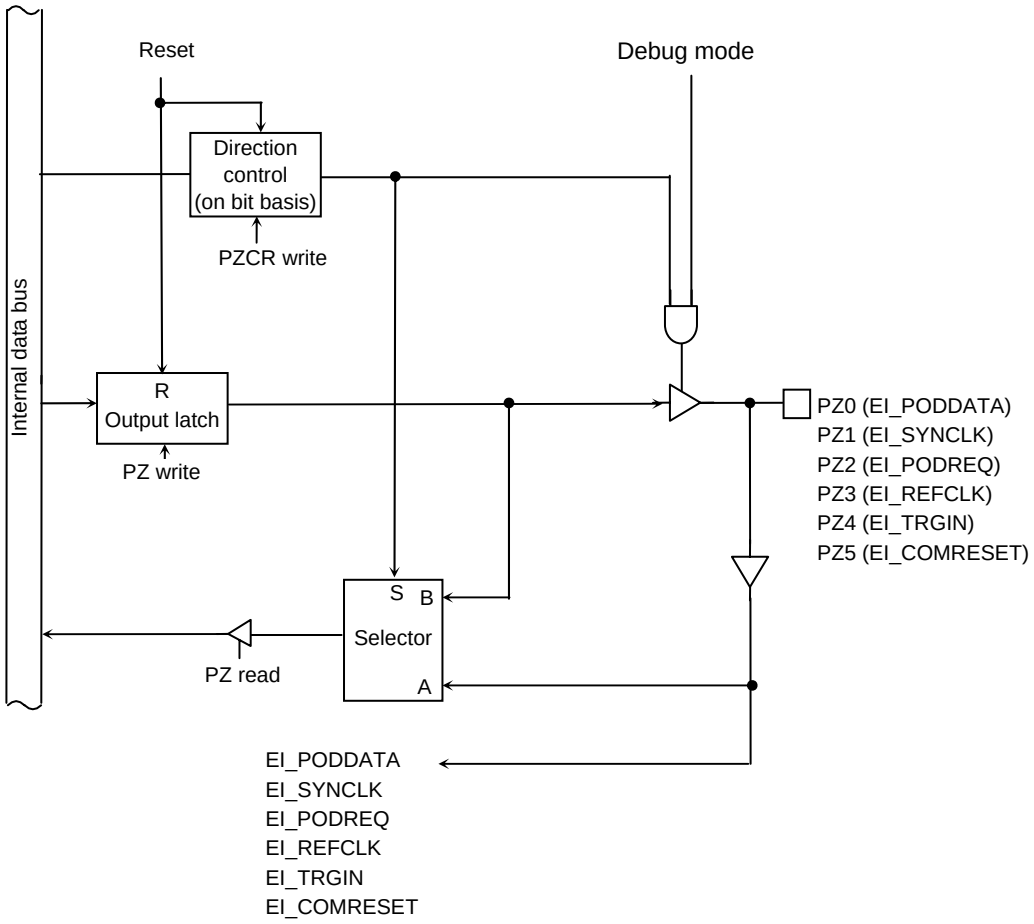


Figure 3.7.66 Port Z0 to Z5

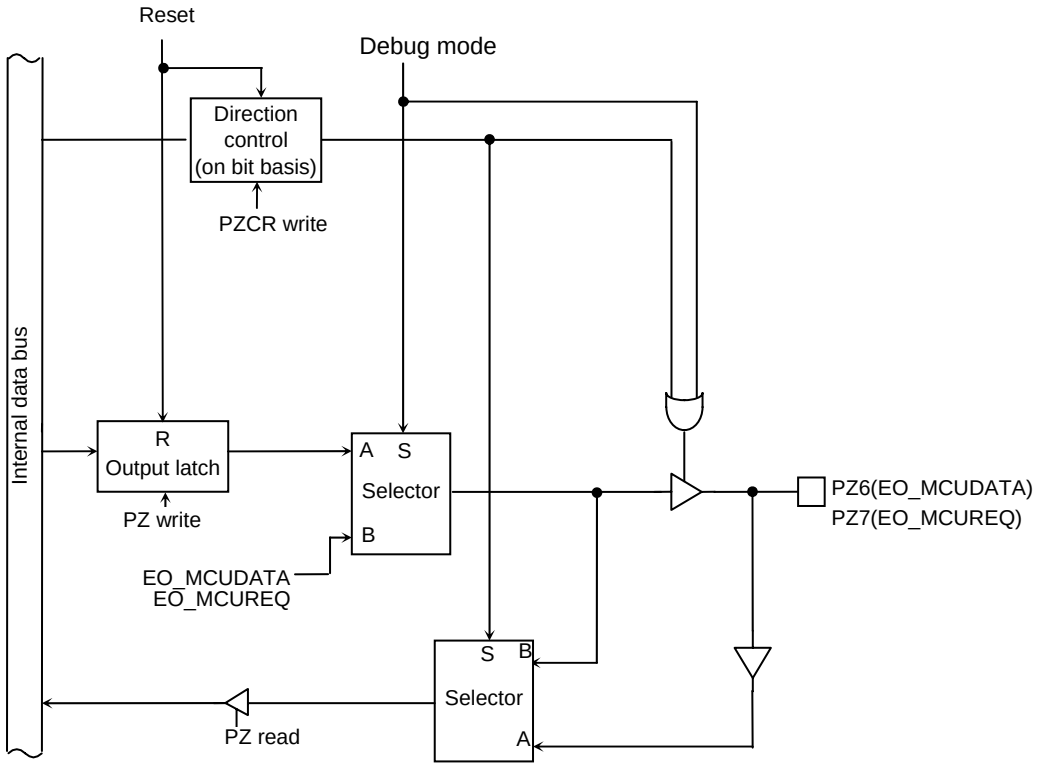


Figure 3.7.67 Port Z6 to Z7

Port Z register

PZ (0068H)	7	6	5	4	3	2	1	0
bit Symbol	PZ7	PZ6	PZ5	PZ4	PZ3	PZ2	PZ1	PZ0
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”)							

Port Z control register

PZCR (006AH)	7	6	5	4	3	2	1	0
bit Symbol	PZ7C	PZ6C	PZ5C	PZ4C	PZ3C	PZ2C	PZ1C	PZ0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port Z drive register

PZDR (009AH)	7	6	5	4	3	2	1	0
bit Symbol	PZ7D	PZ6D	PZ5D	PZ4D	PZ3D	PZ2D	PZ1D	PZ0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note: Read-Modify-Write is prohibited for the registers PZCR.

Figure 3.7.68 Register for Port Z

3.8 Memory Controller (MEMC)

3.8.1 Functions

TMP92CZ26A has a memory controller with a variable 4-block address area that controls as follows.

(1) 4-block address area support

Specifies a start address and a block size for 4-block address area (block0 to 3).

* SRAM or ROM : All CS-blocks (CS0 to CS3) are supported.
* SDRAM : Either CS1 or CS2-blocks is supported.
* Page-ROM : Only CS2-blocks is supported.
* NAND-Flash : CS setting is not needed. If using NAND-Flash, set BROMCR<CSDIS> to "1" as external area for avoiding conflicting with other CS memory.

(2) Connecting memory specifications

Specifies SRAM, ROM, SDRAM as memories to connect with the selected address areas.

(3) Data bus width selection

Whether 8-bit or 16bit is selected as the data bus width of the respective block address areas.

(4) Wait control

Wait specification bit in the control register and WAIT input pin control the number of waits in the external bus cycle. The number of waits of read cycle and write cycle can be specified individually. The number of waits is controlled in 15 mode mentioned below.

0 to 10 wait, 12wait, 16 wait, 20 wait 4+N wait (controls with $\overline{\text{WAIT}}$ pin)

3.8.2 Control register and Operation after reset release

This section describes the registers to control the memory controller, the state after reset release and necessary settings.

(1) Control Register

The control registers of the memory controller are as follows and Table 3.8.1 to Table 3.8.2.

- Control register: BnCSH/BnCSL(n=0 to 3, EX)
  Sets the basic functions of the memory controller that is the connecting memory type, the number of waits to be read and written.
- Memory start address register: MSARn(n=0 to 3)
  Sets a start address in the selected address areas.
- Memory address mask register: MAMR (n=0 to 3)
  Sets a block size in the selected address areas.
- Page ROM control register: PMEMCR
  Sets to control Page-ROM.
- Adjust the timing of control signal register: CSTMGCR, WRTMGCR, RDTMGCRn
  Adjust the timing of rising/falling edge of control signals.
- Internal-Boot ROM control register: BROMCR
  Sets to access Boot-ROM.

Table 3.8.1 Control register

		7	6	5	4	3	2	1	0
B0CSL (0140H)	Bit symbol	B0WW3	B0WW2	B0WW1	B0WW0	B0WR3	B0WR2	B0WR1	B0WR0
	Read/Write	R/W							
	After Reset	0	0	1	0	0	0	1	0
B0CSH (0141H)	Bit Symbol	B0E			B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
	Read/Write	R/W			R/W				
	After Reset	0			0	0	0	0	0
MAMR0 (0142H)	Bit Symbol	M0V20	M0V19	M0V18	M0V17	M0V16	M0V15	M0V14-V9	M0V8
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
MSAR0 (0143H)	Bit Symbol	M0S23	M0S22	M0S21	M0S20	M0S19	M0S18	M0S17	M0S16
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
B1CSL (0144H)	Bit symbol	B1WW3	B1WW2	B1WW1	B1WW0	B1WR3	B1WR2	B1WR1	B1WR0
	Read/Write	R/W							
	After Reset	0	0	1	0	0	0	1	0
B1CSH (0145H)	Bit Symbol	B1E			B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
	Read/Write	R/W			R/W				
	After Reset	0			0	0	0	0	0
MAMR1 (0146H)	Bit Symbol	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	M1V15-V9	M1V8
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
MSAR1 (0147H)	Bit Symbol	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
B2CSL (0148H)	Bit symbol	B2WW3	B2WW2	B2WW1	B2WW0	B2WR3	B2WR2	B2WR1	B2WR0
	Read/Write	R/W							
	After Reset	0	0	1	0	0	0	1	0
B2CSH (0149H)	Bit Symbol	B2E	B2M		B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
	Read/Write	R/W			R/W				
	After Reset	1	0		0	0	0	0	0
MAMR2 (014AH)	Bit Symbol	M2V22	M2V21	M2V20	M2V19	M2V18	M2V17	M2V16	M2V15
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
MSAR2 (014BH)	Bit Symbol	M2S23	M2S22	M2S21	M2S20	M2S19	M2S18	M2S17	M2S16
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
B3CSL (014CH)	Bit symbol	B3WW3	B3WW2	B3WW1	B3WW0	B3WR3	B3WR2	B3WR1	B3WR0
	Read/Write	R/W							
	After Reset	0	0	1	0	0	0	1	0
B3CSH (014DH)	Bit Symbol	B3E			B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
	Read/Write	R/W			R/W				
	After Reset	0			0	0	0	0	0
MAMR3 (014EH)	Bit Symbol	M3V22	M3V21	M3V20	M3V19	M3V18	M3V17	M3V16	M3V15
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
MSAR3 (014FH)	Bit Symbol	M3S23	M3S22	M3S21	M3S20	M3S19	M3S18	M3S17	M3S16
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1

Table 3.8.2 Control register

		7	6	5	4	3	2	1	0
BEXCSL (0159H)	Bit Symbol	BEXWW3	BEXWW2	BEXWW1	BEXWW0	BEXWR3	BEXWR2	BEXWR1	BEXWR0
	Read/Write	R/W							
	After Reset	0	0	1	0	0	0	1	0
BEXCSH (0158H)	Bit Symbol				BEXREC	BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
	Read/Write				R/W				
	After Reset				0	0	0	0	0
PMEMCR (0166H)	Bit Symbol				OPGE	OPWR1	OPWR0	PR1	PR0
	Read/Write				R/W	R/W		R/W	
	After Reset				0	0	0	1	0
CSTMGCR (0168H)	Bit Symbol			TACSEL1	TACSEL0			TAC1	TAC0
	Read/Write			R/W				R/W	
	After Reset			0	0			0	0
WRTMGCR (0169H)	Bit Symbol			TCWSEL1	TCWSEL0	TCWS1	TCWS0	TCWH1	TCWH0
	Read/Write			R/W		R/W		R/W	
	After Reset			0	0	0	0	0	0
RDTMGCR0 (016AH)	Bit Symbol	B1TCRS1	B1TCRS0	B1TCRH1	B1TCRH0	B0TCRS1	B0TCRS0	B0TCRH1	B0TCRH0
	Read/Write	R/W		R/W		R/W		R/W	
	After Reset	0	0	0	0	0	0	0	0
RDTMGCR1 (016BH)	Bit Symbol	B3TCRS1	B3TCRS0	B3TCRH1	B3TCRH0	B2TCRS1	B2TCRS0	B2TCRH1	B2TCRH0
	Read/Write	R/W		R/W		R/W		R/W	
	After Reset	0	0	0	0	0	0	0	0
BROMCR (016CH)	Bit Symbol						CSDIS	ROMLESS	VACE
	Read/Write						R/W		
	After Reset						1	0/1	1/0
RAMCR (016DH)	Bit Symbol								-
	Read/Write								R/W
	After Reset								Always write “1”

(2) Operation after releasing reset

The data bus width at starting is determined depending on state of AM1/AM0 pins after releasing reset. Then, the external memory access as follows;

AM1	AM0	Start Mode
0	0	Don't use this setting
0	1	Start with 16-bit data bus (note)
1	0	Don't use this setting
1	1	Start with BOOT(32-bit internal-MROM)

Note: A memory to be used to start after releasing reset is either NOR-Flash or Masked-ROM.NAND-Flash, SDRAM can't be used.

AM1/AM0 pins are valid only just after releasing reset. In other cases, the data bus width is the value set in the control register <BnBUS 1:0>.

After reset, only control register (B2CSH/B2CSL) of the block address area 2 is effective automatically. (B2CSH<B2E> is set to "1" by reset).

The data bus width which is specified by AM1/AM0 pin is loaded to the bit to specify the bus width of the control register in the block address area 2.

The block address area 2 is set to address 000000H to FFFFFFH by reset (B2CSH<B2M> is reset to "0") .

After releasing reset, the block address areas are specified by MSARn and MAMRn. Then, set BnCS.

Set BnCSH<BnE> to "1" in order to enable the setting.

3.8.3 Basic functions and register setting

In this section, setting of the block address area, the connecting memory and the number of waits out of the memory controller's functions are described.

(1) Block address area specification

The block address areas of CS0 to CS3 are specified by MSAR0 to MSAR3 and MAMR0 to MAMR3.

(a) Memory start address register

Figure 3.8.1 shows the memory start address registers. MSAR0 to MSAR3 set the start addresses for the CS0 to CS3 areas. Set the upper eight bits (A23 to A16) of the start address in <S23:16>. The lower 16 bits of the start address (A15 to A0) are permanently set to 0. Accordingly, the start address can only be set in 64-Kbyte increments, starting from 000000H. Figure 3.8.2 shows the relationship between the start address and the start address register value.

Memory Start Address Registers (for areas CS0 to CS3)

MSAR0 (0143H) / MSAR1 (0147H)
MSAR2 (014BH) / MSAR3 (014FH)

	7	6	5	4	3	2	1	0
Bit symbol	S23	S22	S21	S20	S19	S18	S17	S16
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Determines A23 to A16 of start address							

→ Sets start addresses for areas CS0 to CS3

Figure 3.8.1 Memory Start Address Register

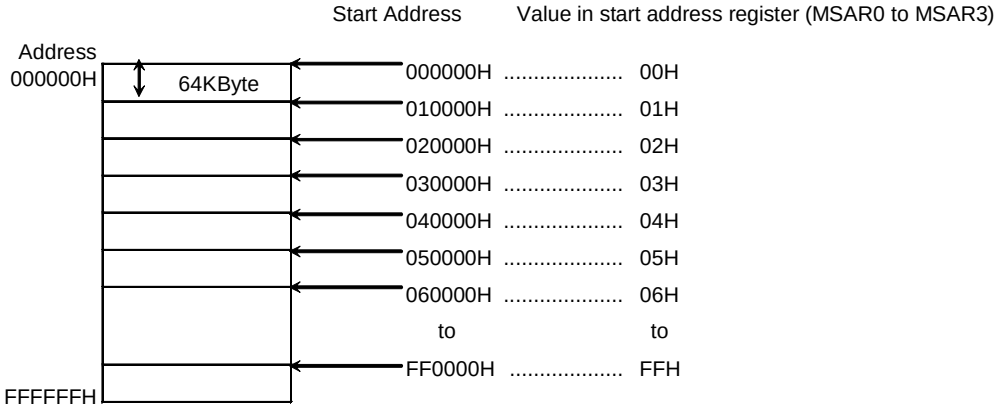


Figure 3.8.2 Relationship between Start Address and Start Address Register Value

(b) Memory address mask registers

Figure 3.8.3 shows the memory address mask registers. MAMR0 to MAMR3 are used to set the size of the CS0 to CS3 areas by specifying a mask for each bit of the start address set in MAMR0 to MAMR3. The compare operation used to determine if an address is in the CS0 to CS3 areas is only performed for bus address bits corresponding to bits set to 0 in these registers.

Also, the address bits that can be masked by MAMR0 to MAMR3 differ between CS0 to CS3 areas.

Block address area CS0 : A20 to A8

Block address area CS1 : A21 to A8

Block address area CS2 to CS3 : A22 to A15

Accordingly, the size that can be each area is different.

Note: After releasing reset, only the control register of the block address area 2 is valid. The control register of the block address area 2 has <B2M> bit. Setting <B2M> bit to "0" sets the block address area 2 to addresses 000000H to FFFFFFH. Setting <B2M> bit to "1" specifies the start address and the address area size as it is in the other block address area.

Memory Address Mask Register (for CS0 area)

MAMR0 (0142H)

	7	6	5	4	3	2	1	0
Bit symbol	V20	V19	V18	V17	V16	V15	V14~9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets size of CS0 area 0: Used for address compare							

Range of possible settings for CS0 area size: 256Bytes to 2MBytes

Memory Address Mask Register (for CS1 area)

MAMR1 (0146H)

	7	6	5	4	3	2	1	0
Bit symbol	V21	V20	V19	V18	V17	V16	V15~9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets size of CS1 area 0: Used for address compare							

Range of possible settings for CS1 area size: 256Bytes to 4MBytes

Memory Address Mask Register (for CS2,CS3 area)

MAMR2 (014AH) / MSAR3 (014FH)

	7	6	5	4	3	2	1	0
Bit symbol	V22	V21	V20	V19	V18	V17	V16	V15
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets size of CS2 or CS3 area 0: Used for address compare							

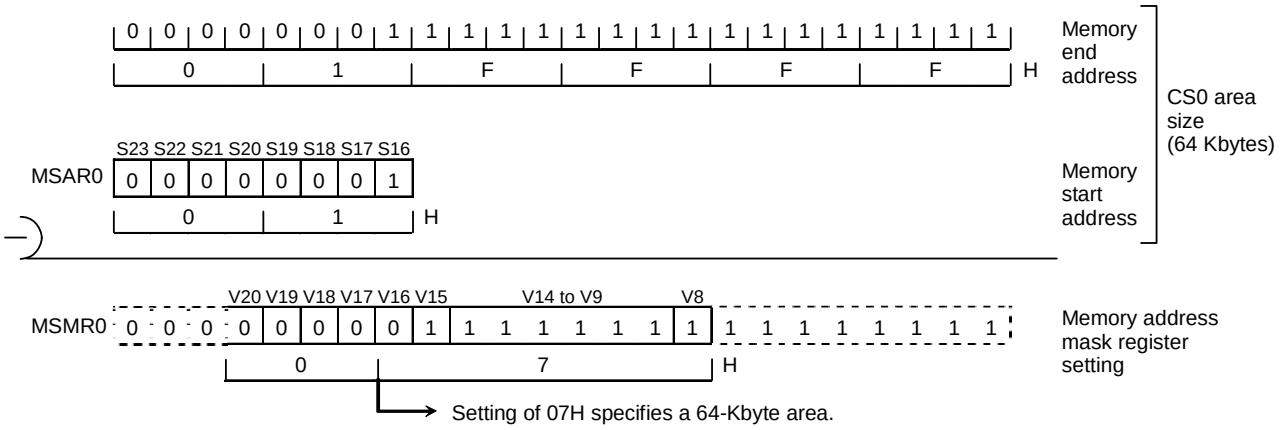
Range of possible settings for CS2 or CS3 area size: 32KBytes to 8MBytes

Figure 3.8.3 Memory Address Mask Registers

(c) Setting memory start addresses and address areas

An example of specifying a 64-Kbyte address area starting from 010000H using the CS0 areas i describes.

Set 01H in MSAR0<S23:16> (Corresponding to the upper 8 bits of the start address). Next, calculate the difference between the start address and the anticipated end address (01FFFFH) based on the size of the CS0 area. Bits 20 to 8 of the result correspond to the mask value to be set for the CS0 area. Setting this value in MAMR0<V20:8> sets the area size. This example sets 07H in MAMR0 to specify a 64K-byte area.



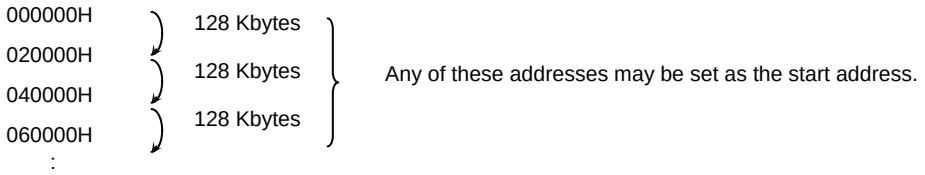
(d) Address area size specification

Table 3.8.3 shows the relationship between CS area and area size. "Δ" indicates areas that cannot be set by memory start address register and address mask register combinations. When setting an area size using a combination indicated by "Δ", set the start address mask register in the desired steps starting from 000000H.

If the CS2 area is set to 16 Mbytes or if two or more areas overlap, the smaller CS area number has the higher priority.

Example: To set the area size for CS0 to 128 Kbytes:

a. Valid start addresses



b. Invalid start addresses

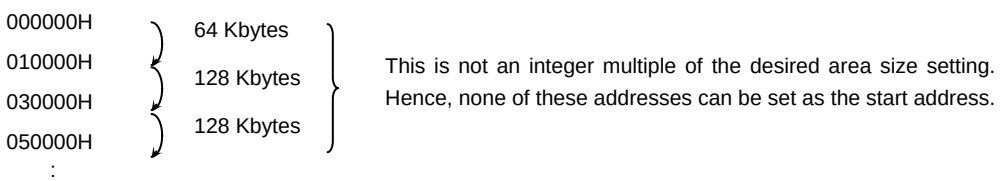


Table 3.8.3 Valid Area Sizes for Each CS Area

Size (Byte) / CS area	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	Δ	Δ	Δ	Δ	Δ		
CS1	○	○		○	Δ	Δ	Δ	Δ	Δ	Δ	
CS2			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ
CS3			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ

Note:"Δ" indicates areas that cannot be set by memory start address register and address mask register combinations.

(e) Block address area Priority

When the set block address area overlaps with the built-in memory area, or both two address areas overlap, the block address area is processed according to priority as follows.

Built-in I/O > Built-in memory > Block address area 0 > 1 > 2 > 3

(f) Wait control for outside the block address area of CS0 to CS3

Also, that any accessed areas outside the address spaces set by CS0 to CS3 are processed as the CSEX space. Therefore, settings of CSEX (BEXCSH, L-register) apply for the control of wait cycles, data bus width, etc,.

(2) Connection Memory Specification

Setting BnCSH<BnOM1:0> specifies the memory type to be connected with the block address areas. The interface signal is output according to the set memory as follows;

BnCSH<BnOM1:0>

BnOM1	BnOM0	Function
0	0	SRAM/ROM (Default)
0	1	(Reserved)
1	0	(Reserved)
1	1	SDRAM

Note1: SDRAM should be set only with CS1 or CS2 .

(3) Data Bus Width Specification

The data bus width is set for every block address area. The bus size is set by BnCSH<BnBUS1:0> as follows;

BnCSH<BnBUS1:0>

<BnBUS1>	<BnBUS0>	Function
0	0	8-bit bus mode (Default)
0	1	16-bit bus mode
1	0	Reserved
1	1	Don't use this setting

Note1: SDRAM should be set to "01"(16-bit bus).

This way of changing the data bus width depending on the address being accessed is called "dynamic bus sizing". The part where the data is output to is depended on the data width, the bus width and the start address.

The number of external data bus pin in TMP92CZ26A are 16 pin. Therefore, please ignore the bus width of memory = 32 bit in the table.

Note: Since there is a possibility of abnormal writing/reading of the data if two memories with different bus width are put in consecutive address, do not execute a access to both memories with one command.

Operand Data Size (bit)	Operand Start Address	Bus width of Memory (bit)	CPU Address	CPU Data			
				D31 to D24	D23 to D16	D15 to D8	D7 to D0
8	4n + 0	8/16/32	4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
	4n + 1	8	4n + 1	xxxxx	xxxxx	xxxxx	b7 to b0
		16/32	4n + 1	xxxxx	xxxxx	b7 to b0	xxxxx
	4n + 2	8/16	4n + 2	xxxxx	xxxxx	xxxxx	b7 to b0
		32	4n + 2	xxxxx	b7 to b0	xxxxx	xxxxx
	4n + 3	8	4n + 3	xxxxx	xxxxx	xxxxx	b7 to b0
		16	4n + 3	xxxxx	xxxxx	b7 to b0	xxxxx
		32	4n + 3	b7 to b0	xxxxx	xxxxx	xxxxx
16	4n + 0	8	(1) 4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
		16/32	4n + 0	xxxxx	xxxxx	b15 to b8	b7 to b0
	4n + 1	8	(1) 4n + 1	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 2	xxxxx	xxxxx	xxxxx	b15 to b8
		16	(1) 4n + 1	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 2	xxxxx	xxxxx	xxxxx	b15 to b8
		32	4n + 1	xxxxx	b15 to b8	b7 to b0	xxxxx
	4n + 2	8	(1) 4n + 2	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
		16	4n + 2	xxxxx	xxxxx	b15 to b8	b7 to b0
		32	4n + 2	b15 to b8	b7 to b0	xxxxx	xxxxx
	4n + 3	8	(1) 4n + 3	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
		16	(1) 4n + 3	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
		32	(1) 4n + 3	b7 to b0	xxxxx	xxxxx	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
32	4n + 0	8	(1) 4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 2	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 3	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 0	xxxxx	xxxxx	b15 to b8	b7 to b0
			(2) 4n + 2	xxxxx	xxxxx	b31 to b24	b23 to b16
		32	4n + 0	b31 to b24	b23 to b16	b15 to b8	b7 to b0
	4n + 1	8	(1) 4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 2	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 3	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 1	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 2	xxxxx	xxxxx	b23 to b16	b15 to b8
			(3) 4n + 4	xxxxx	xxxxx	xxxxx	b31 to b24
		32	(1) 4n + 1	b23 to b16	b15 to b8	b7 to b0	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b31 to b24
	4n + 2	8	(1) 4n + 2	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 3	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 4	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 5	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 2	xxxxx	xxxxx	b15 to b8	b7 to b0
			(2) 4n + 4	xxxxx	xxxxx	b31 to b24	b23 to b16
		32	(1) 4n + 2	b15 to b8	b7 to b0	xxxxx	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	b31 to b24	b23 to b16
	4n + 3	8	(1) 4n + 3	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 5	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 6	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 3	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	b23 to b16	b15 to b8
			(3) 4n + 6	xxxxx	xxxxx	xxxxx	b31 to b24
		32	(1) 4n + 3	b7 to b0	xxxxx	xxxxx	xxxxx
			(2) 4n + 4	xxxxx	b31 to b24	b23 to b16	b15 to b8

xxxxx: During read, input data to the bus is ignored. At write, the bus is high impedance and the write strobe signal remains no-active.

(4) Wait control

The external bus cycle completes for two states minimum(25 ns at f_{SYS} = 80 MHz).

Setting the BnCSL<BnWW3:0> specifies the number of waits in the write cycle, and BnCSL<BnWR3:0> specifies the number of waits in the read cycle. <BnWW3:0> is set with the same method as <BnWR3:0> as follows;

BnCSL<BnWW>/<BnWR>

<BnWW3> <BnWR3>	<BnWW2> <BnWR2>	<BnWW1> <BnWR1>	<BnWW0> <BnWR0>	Function
0	0	0	1	2 states (0 waits) access fixed mode
0	0	1	0	3 states (1 wait) access fixed mode (Default)
0	1	0	1	4 states (2 waits) access fixed mode
0	1	1	0	5 states (3 waits) access fixed mode
0	1	1	1	6 states (4 waits) access fixed mode
1	0	0	0	7 states (5 waits) access fixed mode
1	0	0	1	8 states (6 waits) access fixed mode
1	0	1	0	9 states (7 waits) access fixed mode
1	0	1	1	10 states (8 waits) access fixed mode
1	1	0	0	11 states (9 waits) access fixed mode
1	1	0	1	12 states (10 waits) access fixed mode
1	1	1	0	14 states (12 waits) access fixed mode
1	1	1	1	18 states (16 waits) access fixed mode
0	1	0	0	22 states (20 waits) access fixed mode
0	0	1	1	6 states + $\overline{\text{WAIT}}$ pin input mode
others				(Reserved)

Note 1:For SDRAM, above setting is ineffective. Refer to the section 3.18 SDRAM controller.
Note 2:For NAND flash, this setting is ineffective.

(i) Waits number fixed mode

The bus cycle is completed with the set states. The number of states is selected from 2 states (0 waits) to 12 states (10 waits), 14 states(12 waits), 18 states(16 waits) and 22 states(20 waits).

(ii) $\overline{\text{WAIT}}$ pin input mode

This mode samples the $\overline{\text{WAIT}}$ input pins. It continuously samples the $\overline{\text{WAIT}}$ pin state and inserts a wait if the pin is active. The bus cycle is minimum 6 states. The bus cycle is completed when the wait signal is non active (“High” level) at 6 states. The bus cycle is extended as long as the wait signal is active in case more than 6 states.

(5) Recovery (Data hold) cycle control

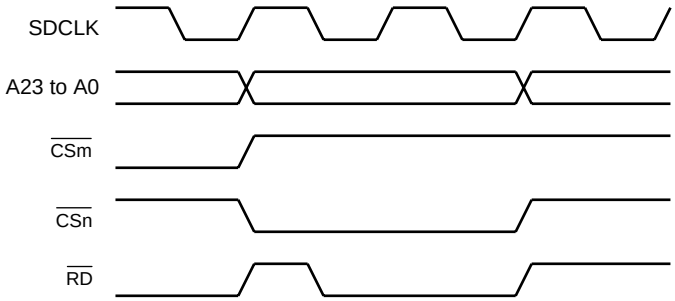
Some memory have an AC specification about data hold time from $\overline{CE}$ or $\overline{OE}$ for read cycle and a data confliction problem may occur. To avoid this problem, 1-dummy cycle can be inserted after CSm-block access cycle by setting "1" to BmCSH<BmREC> register.

This 1-dummy cycle is inserted when the next cycle is for another CS-block.

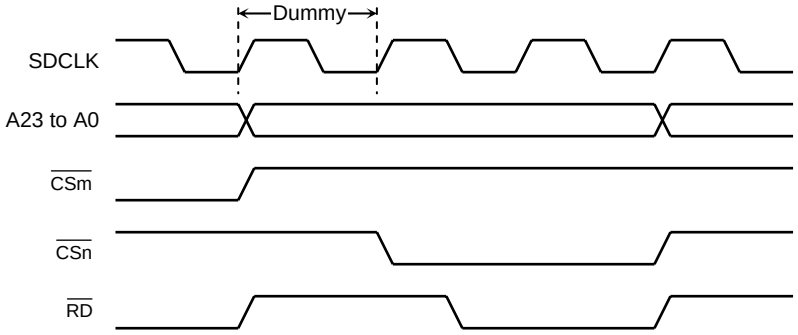
BnCSH<BnREC>

0	No dummy cycle is inserted (Default).
1	Dummy cycle is inserted.

- When not inserting a dummy cycle (0 waits)



- When inserting a dummy cycle (0 waits)



(6) Adjust Function for the timing of control signal

This function can change the timing of $\overline{CSn}$, $\overline{CSZx}$, $\overline{CSXx}$, R/$\overline{W}$, $\overline{RD}$, $\overline{WRxx}$, $\overline{SRWR}$ and $\overline{SRxxB}$ signals and adjust the timing according to the set-up/hold time of the memories.

As for the $\overline{CSn}$, $\overline{CSZx}$, $\overline{CSXx}$, R/$\overline{W}$ and $\overline{WRxx}$, $\overline{SRWR}$, $\overline{SRxxB}$ (at write cycle), it can be changed for only 1 CS area. While for $\overline{RD}$ and $\overline{SRxxB}$ (at read cycle), it can be changed for all CS areas. As for CS area and EX area which is not set this function, it operates with base bus timing (Refer to (7)).

This can not be used together with BnCSH<BnREC> function.

For control signal of SDRAM, it can be adjusted in SDRAM controller.

CSTMGCR<TxxSEL1:0>, WRTMGCR<TxxSEL1:0>

00	Change the timing of CS0 area
01	Change the timing of CS1 area
10	Change the timing of CS2 area
11	Change the timing of CS3 area

CSTMGCR<TAC1:0>

00	TAC = 0 × f_{SYS} (Default)
01	TAC = 1 × f_{SYS}
10	TAC = 2 × f_{SYS}
11	(Reserved)

TAC:The delay from (A23-0) to (CSn, CSZx, CSXx, R/W).

WRTMGCR<TCWS/H1:0>

00	TCWS/H = 0.5 × f_{SYS} (Default)
01	TCWS/H = 1.5 × f_{SYS}
10	TCWS/H = 2.5 × f_{SYS}
11	TCWS/H = 3.5 × f_{SYS}

TCWS:The delay from (CSn) to (WRxx,SRWR,SRxxB).

TCWH:The delay from (WRxx,SRWR,SRxxB) to (CSn).

RDTMGCR0/1<BnTCRH1:0>

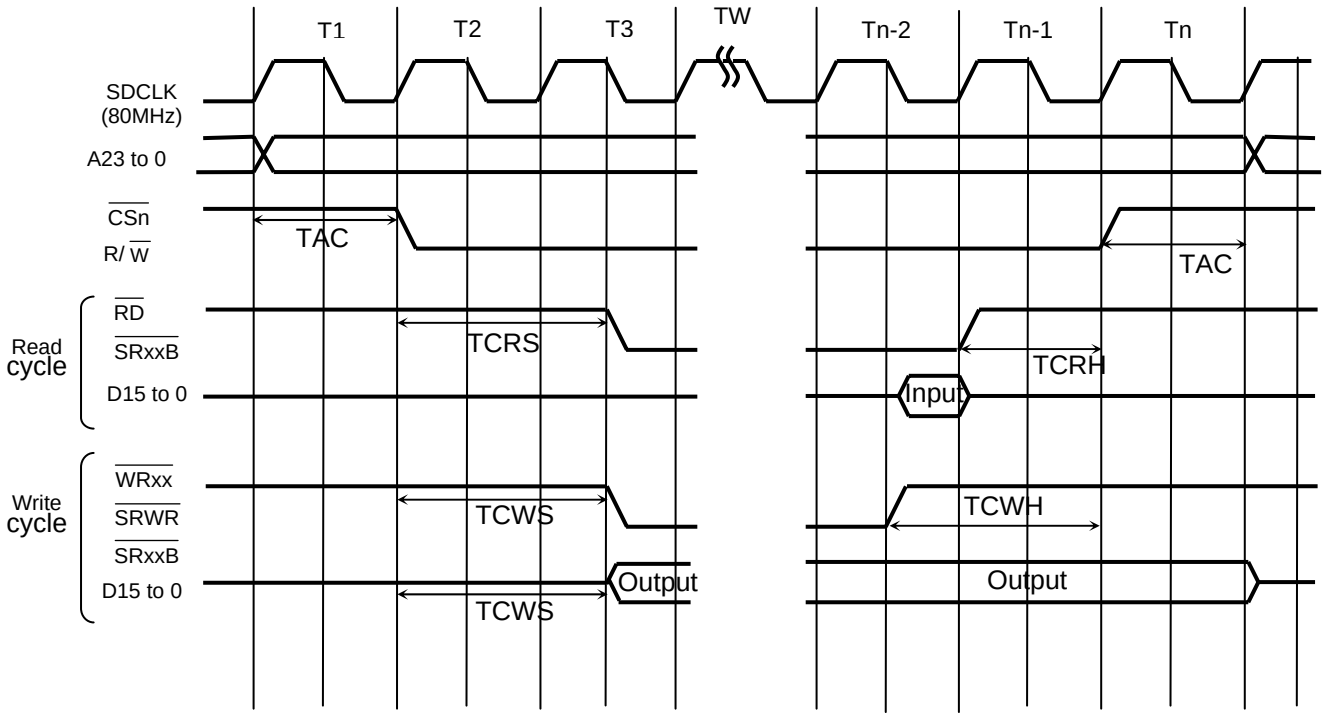
00	TCRH = 0 × f_{SYS} (Default)
01	TCRH = 1 × f_{SYS}
10	TCRH = 2 × f_{SYS}
11	TCRH = 3 × f_{SYS}

TCRH:The delay from (RD,SRxxB) to (CSn).

RDTMGCR0/1<BnTCRS1:0>

00	TCRS = 0.5 × f_{SYS} (Default)
01	TCRS = 1.5 × f_{SYS}
10	TCRS = 2.5 × f_{SYS}
11	TCRS = 3.5 × f_{SYS}

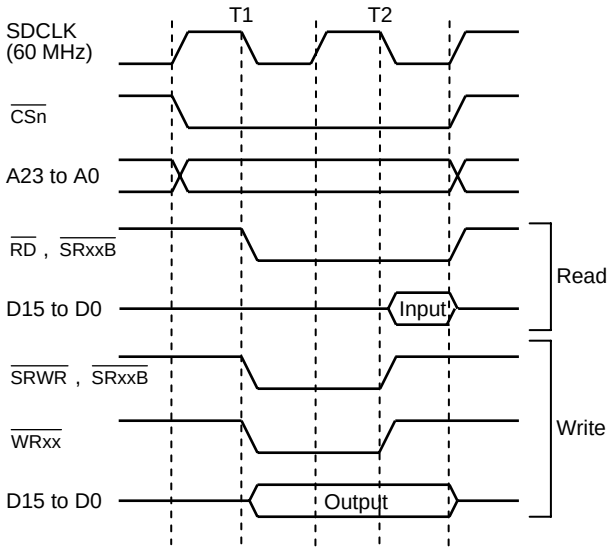
TCRS:The delay from (CSn) to (RD,SRxxB).



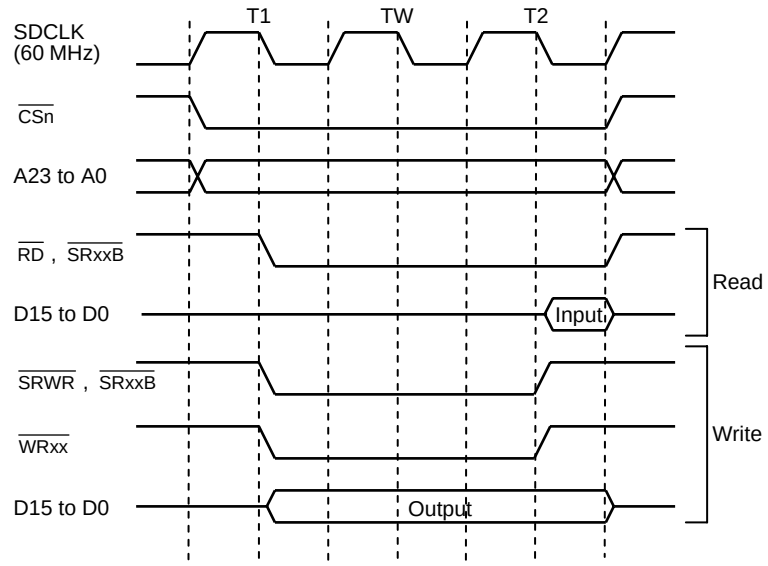
Note: TW cycle is inserted by setting BnCSL register. If it is set to 0-Wait, TW cycle is not inserted.

(7) Basic bus timing

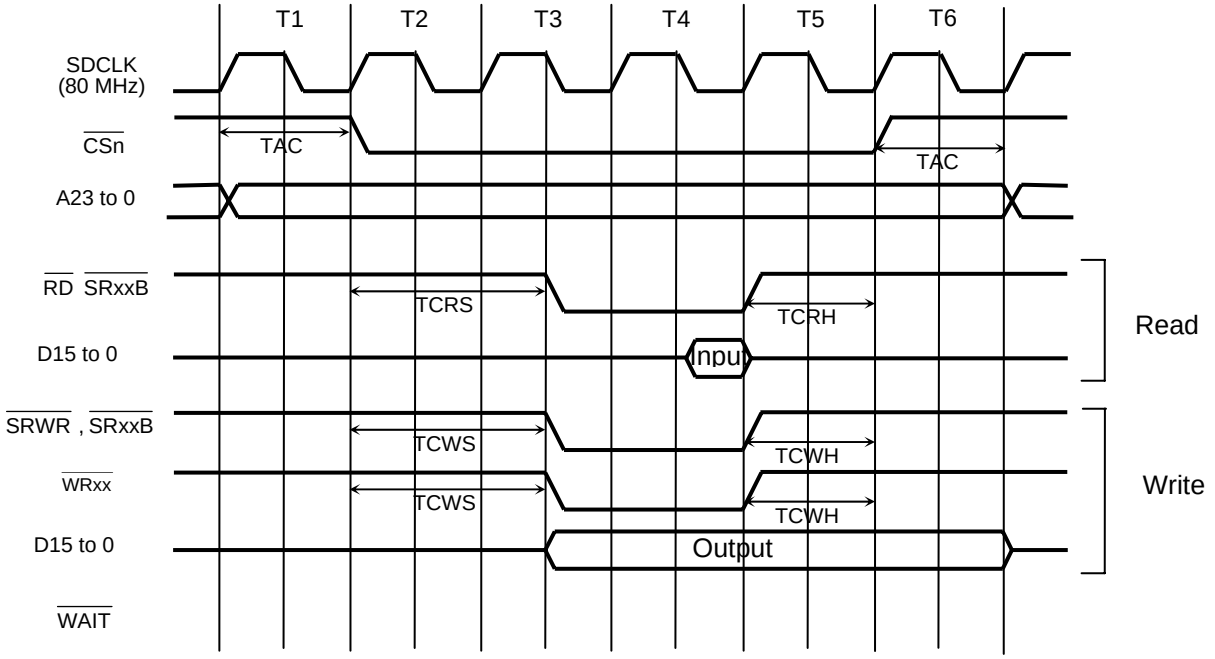
(a) External read/write cycle (0 waits)



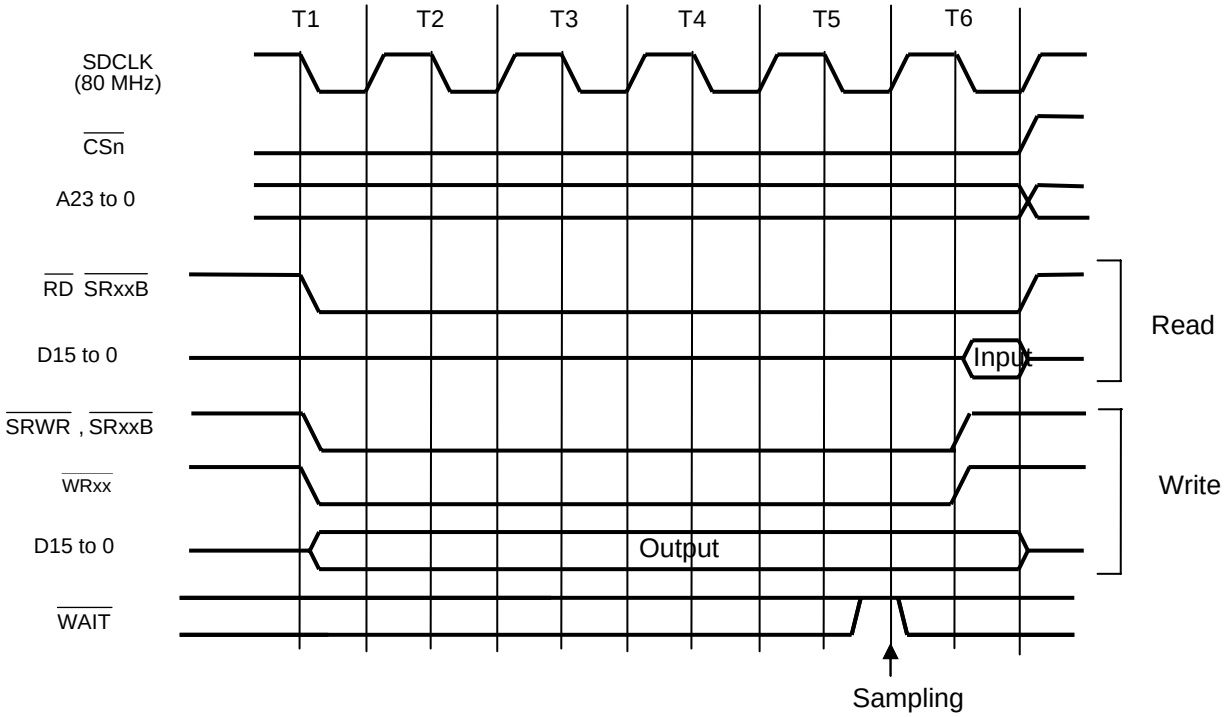
(b) External read/write cycle (1 wait)



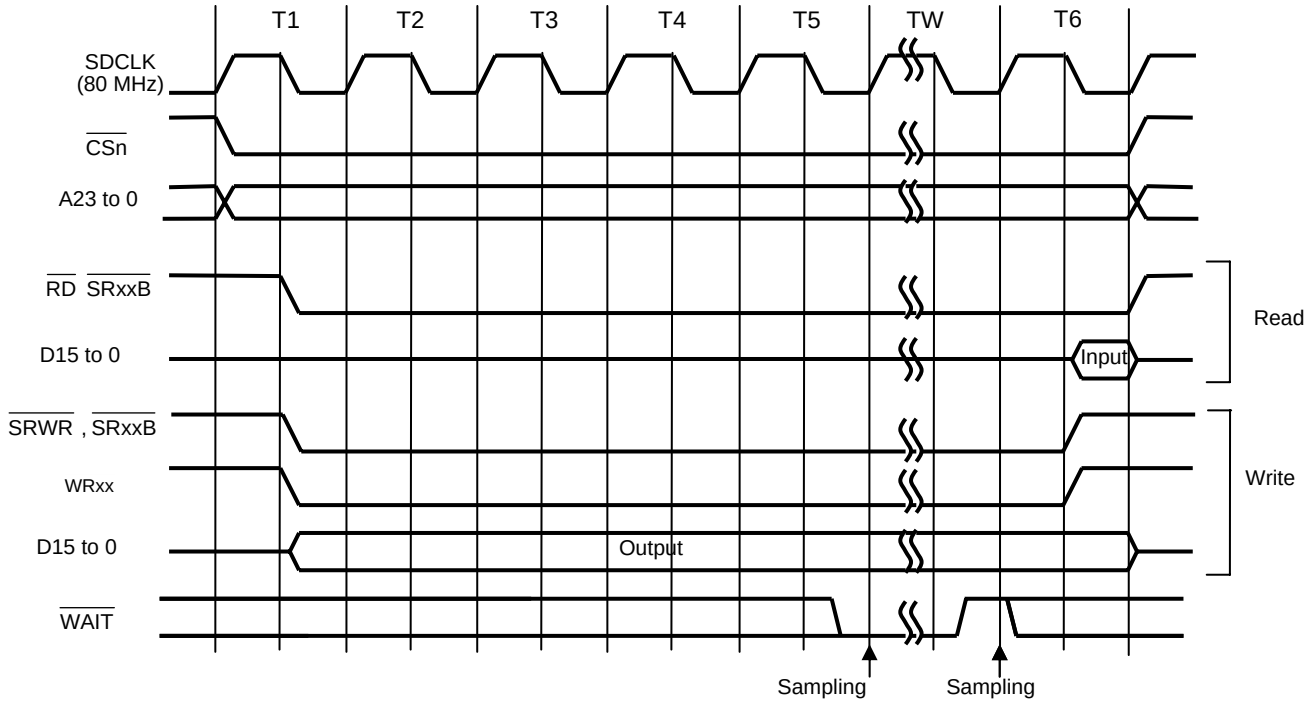
(c) External read bus cycle (1 wait + TAC: $1f_{SYS}$ + TCRS: $1.5f_{SYS}$ + TCRH: $1f_{SYS}$)
External write bus cycle (1 wait + TAC: $1f_{SYS}$ + TCWS/H: $1.5f_{SYS}$)



(d) External read/write cycle (4 waits + $\overline{WAIT}$ pin input mode)

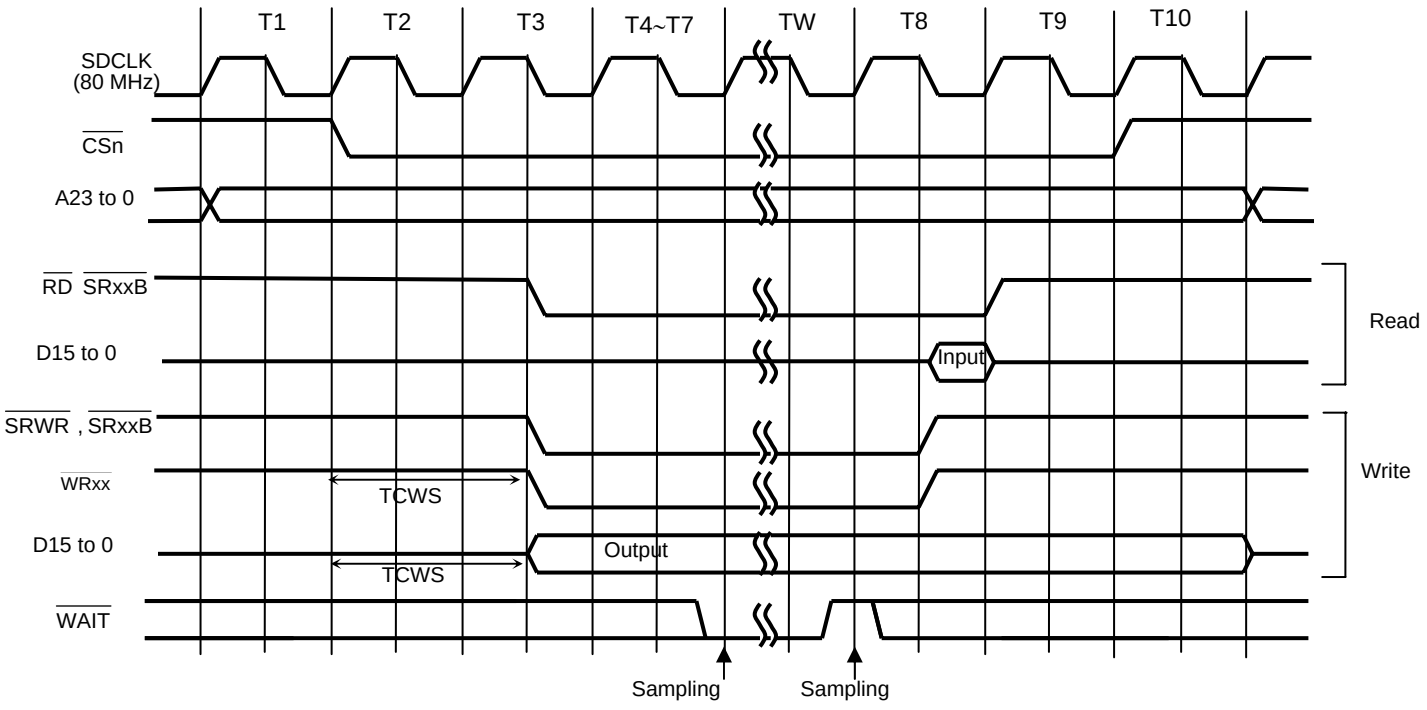


(e) External read/write cycle (4 waits + $\overline{\text{WAIT}}$ pin input mode)



(f) External read bus cycle (4 waits + $\overline{\text{WAIT}}$ pin input mode +TAC: $1f_{SYS}$ + TCRS: $1.5f_{SYS}$ + TCRH: $1f_{SYS}$)

External write bus cycle (4 waits + $\overline{\text{WAIT}}$ pin input mode + TAC: $1f_{SYS}$ + TCWS/H: $1.5f_{SYS}$)



(8) Connecting to external memory

Figure 3.8.4 shows an example of how to connect external 16-bit SRAM and 16-bit NOR flash to the TMP92CZ26A.

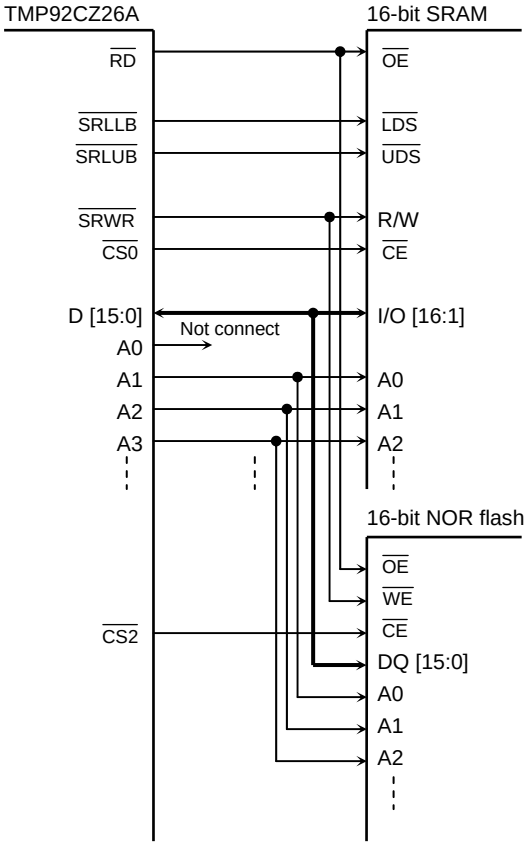


Figure 3.8.4 Example of External 16-Bit SRAM and NOR Flash Connection

3.8.4 ROM Page mode Access Control

This section describes ROM page mode accessing and how to set registers. ROM page mode is set by PMEMCR.

(1) Operation and how to set the registers

TMP92CZ26A supports ROM access with the page mode. The ROM access with the page mode is specified only in CS2.

Setting PMEMCR<OPGE> to "1" sets the memory access of CS2 to ROM page mode access.

The number of read cycles is set by the PMEMCR<OPWR1:0>.

PMEMCR<OPWR1:0>

<OPWR1>	<OPWR0>	Number of Cycle in a Page
0	0	1 state (n-1-1-1 mode) (n ≥ 2)
0	1	2 state (n-2-2-2 mode) (n ≥ 3)
1	0	3 state (n-3-3-3 mode) (n ≥ 4)
1	1	4 state (n-4-4-4 mode) (n ≥ 5)

Note: Set the number of waits "n" to the control register (BnCSL) in each block address area.

The page size (the number of bytes) of ROM in the CPU size is set to PMEMCR<PR1:0>. When data is read out until a border of the set page, the controller completes the page reading operation. The start data of the next page is read in the normal cycle. The following data is set to page read again.

PMEMCR<PR1:0>

<PR1>	<PR0>	ROM Page Size
0	0	64 bytes
0	1	32 bytes
1	0	16 bytes (Default)
1	1	8 bytes

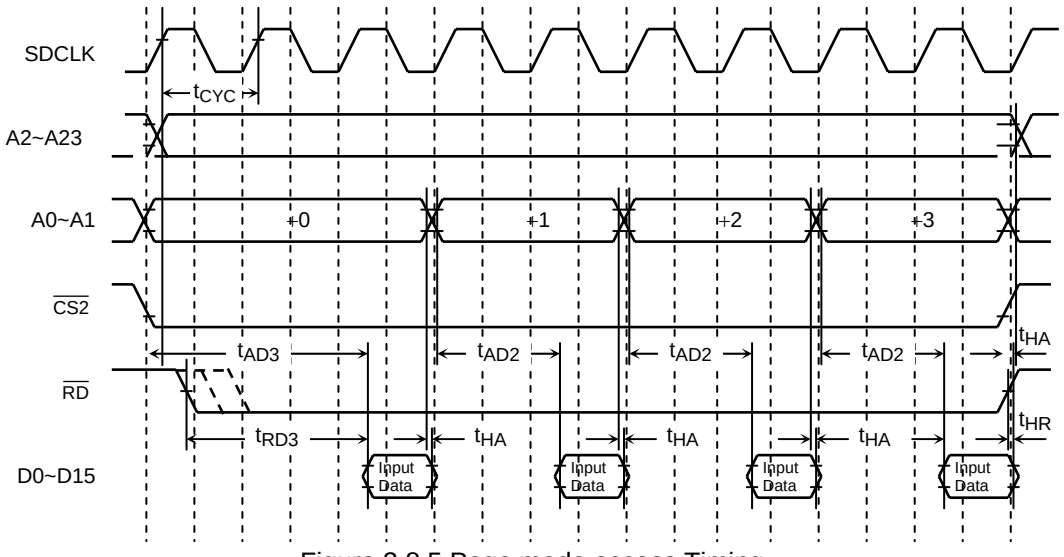


Figure 3.8.5 Page mode access Timing

3.8.5 Internal Boot ROM Control

This section describes about built-in boot ROM.

For the specification of S/W in boot ROM, refer to the section 3.4 boot ROM.

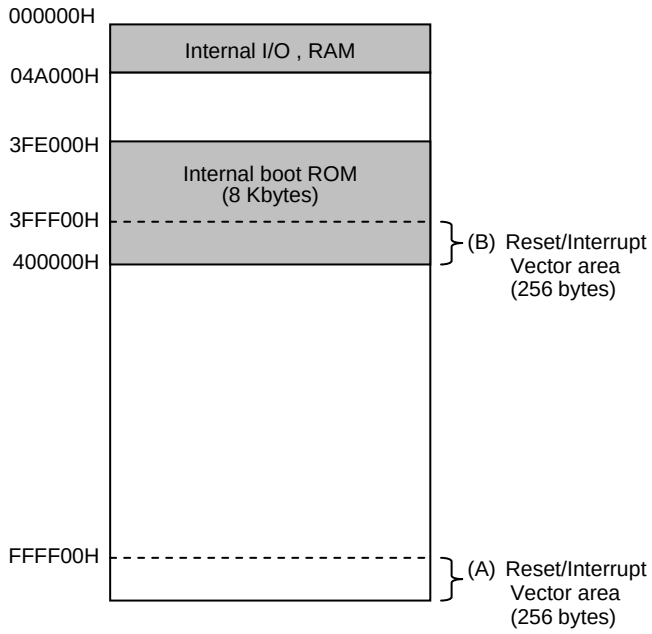
(1) BOOT mode

BOOT mode is started by following AM1 and AM0 pins condition with reset.

AM1	AM0	Start mode
0	0	Don't use this setting
0	1	Start with 16-bit data bus
1	0	Don't use this setting
1	1	Start with boot (32-bit internal MROM)

(2) Boot ROM memory map

Boot ROM is consist of 8-Kbyte masked ROM and assigned 3FE000H to 3FFFFFH address.



(3) Reset/interrupt address conversion circuit

Originally, reset/interrupt vector area is assigned FFFF00H to FFFFEFH ((A) area) in TLCS-900/H1.

But because boot ROM is assigned to another area, reset/interrupt vector address conversion circuit is prepared.

In BOOT mode, reset/interrupt vector area is assigned 3FFF00H to 3FFFEFH ((B) area) area by it. And after boot sequence, its area can be changed to (A) area by setting BROMCR<VACE> to "0". So, (A) area can be used only for application system program.

This BROMCR<VACE> is initialized to "1" in BOOT mode. At another starting mode, this register has no meaning.

Note: The last 16-byte area (FFFFF0H to FFFFFFH) is reserved for an emulator. So, this area is not changed by <VACE> register.

(4) Disappearing boot ROM

After boot sequence in BOOT mode, an application system program may continue to run without reset asserting. In this case, an external memory which is mapped 3FE000H to 3FFFFFH address can not be accessed because of boot ROM is assigned.

To solve it, internal boot ROM can be disappered by setting BROMCR<ROMLESS> to "1".

This <ROMLESS> is initialized to "0" in BOOT mode. At another starting mode, this bit is initialized to "1".

If this bit has been set to "1", writing "0" is disabled.

BROMCR (016CH)

	7	6	5	4	3	2	1	0
Bit symbol						CSDIS	ROMLESS	VACE
Read/Write						R/W		
After Reset						1	0/1 (note)	1/0 (note)
Function						Nand_Flash area CS output 0: Enable 1: Disable	Boot ROM 0: use 1: not use	Vector address conversion 0: Disable 1: Enable

Note: The value after reset release is depending on start mode.

3.8.6 Cautions

(1) Note the timing between $\overline{CS}$ and $\overline{RD}$

If the load capacitance of the $\overline{RD}$ (Read signal) is greater than that of the $\overline{CS}$ (Chip select signal), it is possible that an unintended read cycle occurs due to a delay in the read signal. Such an unintended read cycle may cause a trouble as in the case of (a) in Figure 3.8.6.

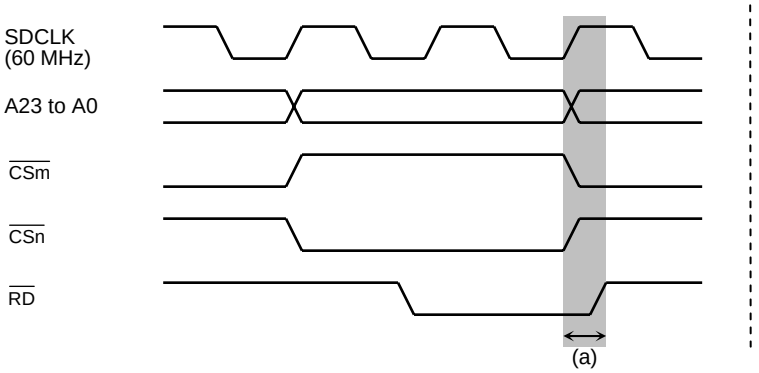


Figure 3.8.6 Read Signal Delay Read Cycle

Example: When using an externally connected NOR flash which users JEDEC standard commands, note that the toggle bit may not be read out correctly. If the read signal in the cycle immediately preceding the access to the NOR flash does not go high in time, as shown in Figure 3.8.7, an unintended read cycle like the one shown in (b) may occur.

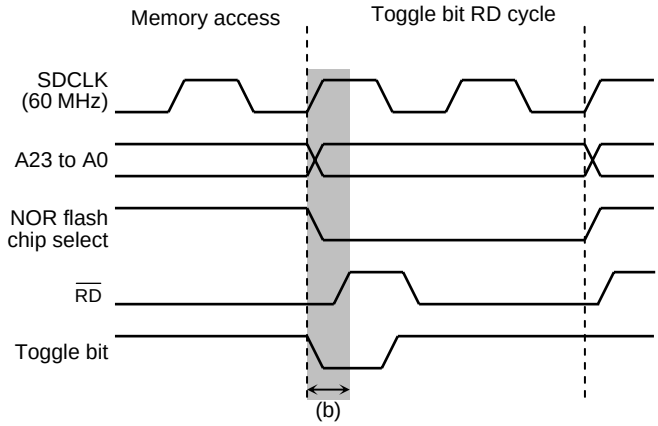


Figure 3.8.7 NOR Flash Toggle Bit Read Cycle

When the toggle bit reverse with this unexpected read cycle, CPU always reads same value of the toggle bit, and cannot read the toggle bit correctly.

To avoid this phenomenon, the data polling function control is recommended. Or use the adjust timing function for rising edge of $\overline{RD}$ (RDTMGCRn<BnTCRH1:0>) in order to avoid generating this phenomenon.

(2) Note the NAND flash area setting

Figure 3.8.8 shows a memory map for NAND flash.

And since CS3 area is recommended to assign address from 000000H to 3FFFFFH, this case is explained.

In this case, "NAND flash" and CS3 area are overlapped. But $\overline{CS3}$ pin don't become active by setting BROMCR<CSDIS> to "1". And also $\overline{CS0}$ to $\overline{CS3}$, $\overline{SDCS}$, $\overline{CSXA}$ to $\overline{CSXB}$, $\overline{CSZA}$ to $\overline{CSZD}$ pins don't become to active.

Note1: In this case, the address from 000000H to 049FFFH of 296 Kbytes in CS3's memory can't be used.

Note2: 16 byte area (001FF0H to 001FFFH) for NAND Flash are fixed like a following without relationship to setting CS bock. Therefore, NAND flash area don't according to CS3 area setting.

(NAND-Flash area specification)

1. bus width : Depend on NDFMCR1<BUSW> in NAND Flash controller.

2.WAIT control : Depend on NDFMCR<SPLW1:0>,<SPHW1:0> in NAND Flash controller

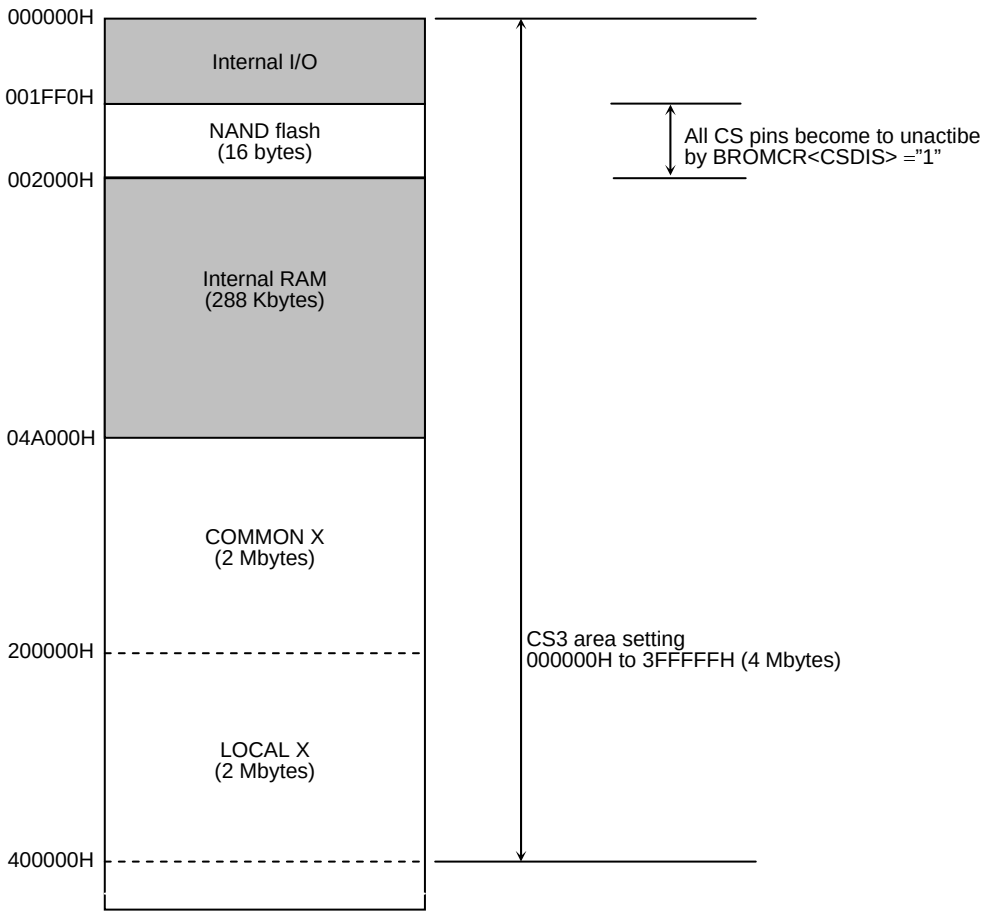


Figure 3.8.8 Recommended CS3 setting

3.9 External Memory Extension Function (MMU)

This is MMU function which can expand program/data area to 3.1G bytes by having 4-type local area.

The recommendation address memory map is shown in Figure 3.9.1.

However, when total capacity of used memory is less than 16M bytes, please refer to section of Memory controller. Setting of register in MMU is not necessary.

An area which can be set as BANK is called LOCAL-area. Since the address for LOCAL area is fixed, it cannot be changed.

And, area that cannot be set as BANK is called COMMON-area.

Basically one series of program should be closed within one bank. Please don't jump to the same LOCAL-area in the different bank directly by JP instruction and so on. Refer to the examples as follows.

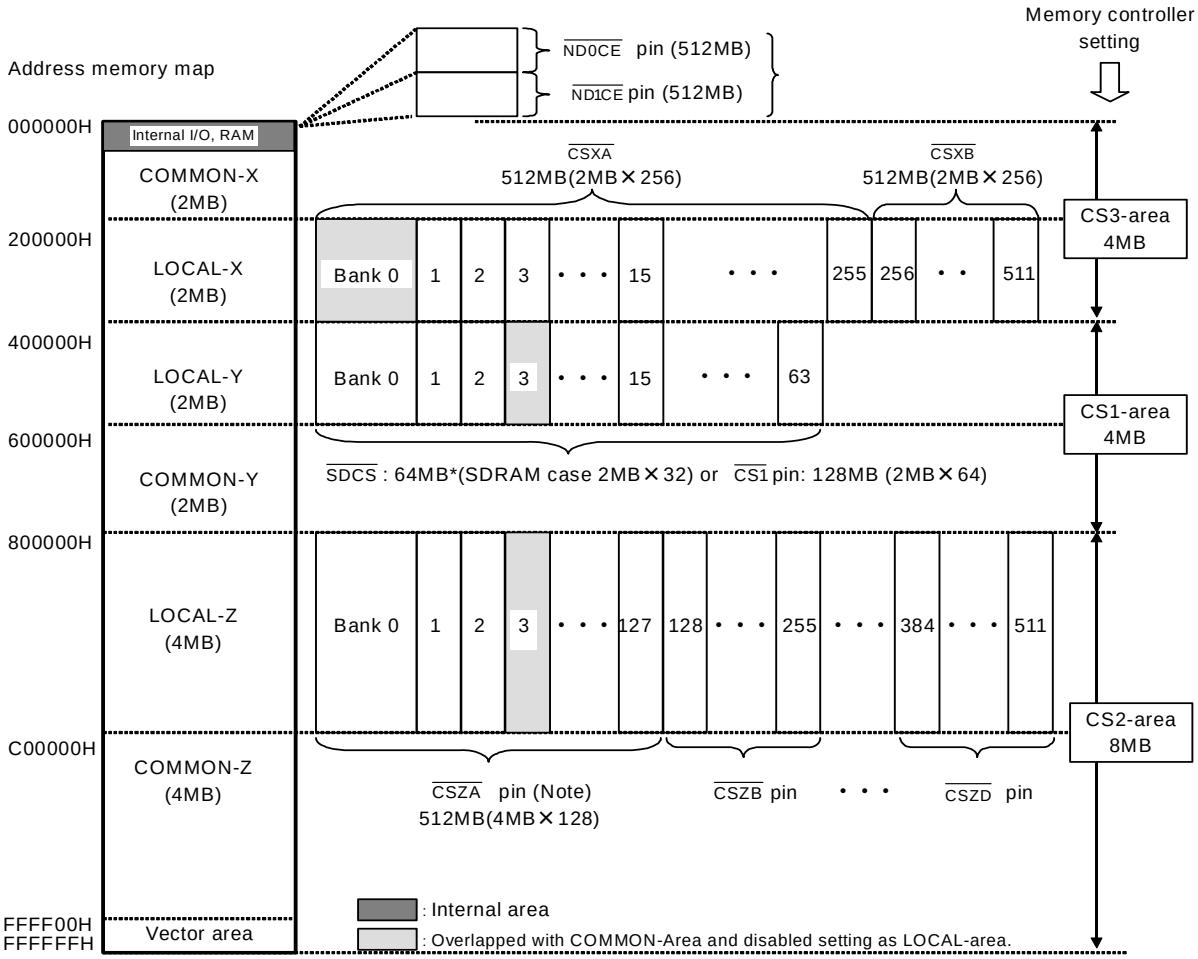
TMP92CZ26A has following external pins to connect external memory-LSI.

Address bus	: EA28, EA27, EA26, EA25, EA24 and A23 to A0
Chip Select	: $\overline{CS0}$ to $\overline{CS3}$, $\overline{CSXA}$ to $\overline{CSXB}$, $\overline{CSZA}$ to $\overline{CSXD}$, $\overline{SDCS}$, $\overline{ND0CE}$ and $\overline{ND1CE}$
Data bus	: D15 to D0

3.9.1 Recommended memory map

Figure 3.9.1 shows one of recommendation address memory map. It can be expanded to maximum memory size.

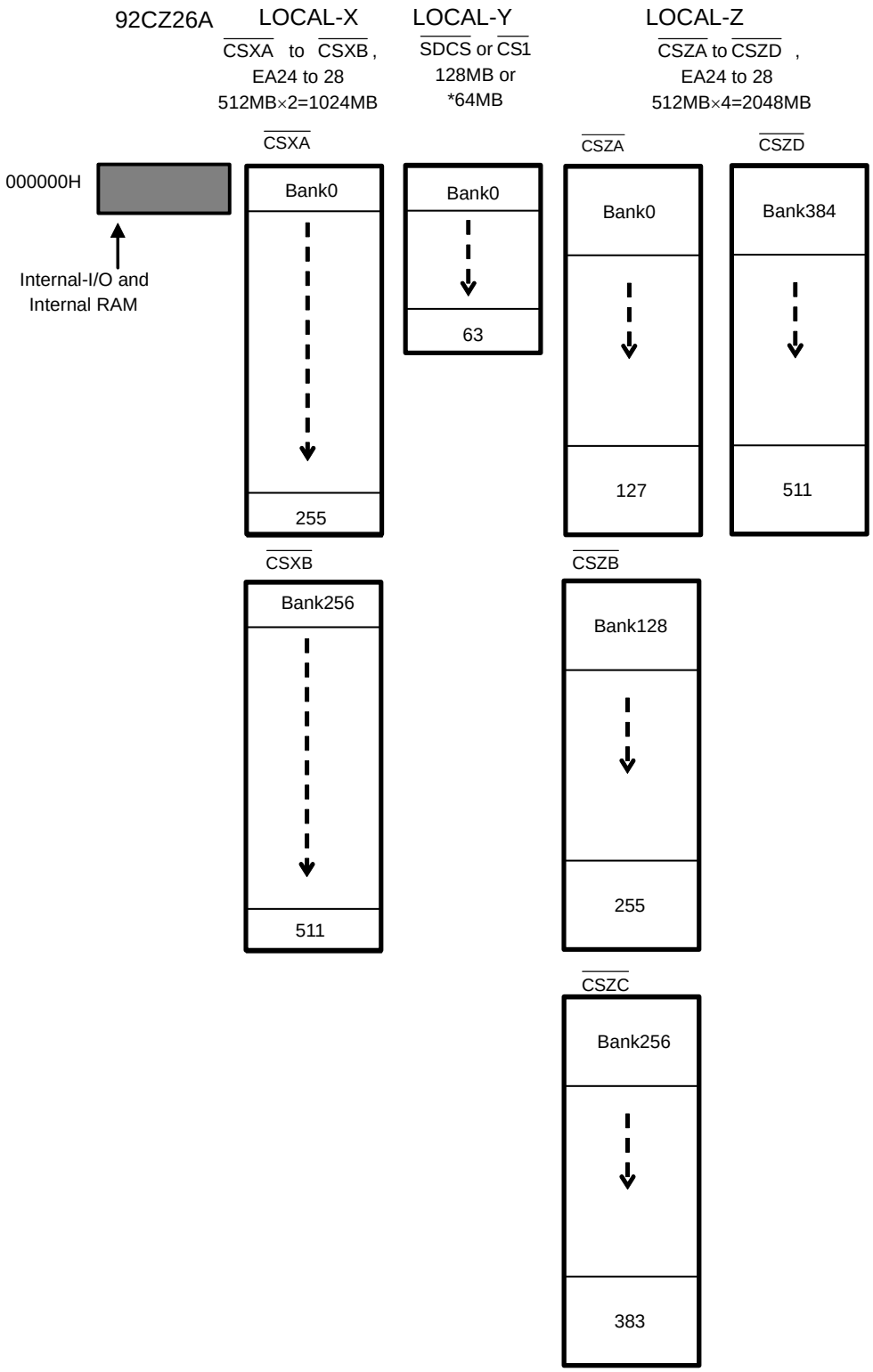
Figure 3.9.3 also shows one of recommendation address memory map. It's for a simple memory system like internal Boot-ROM with NAND-Flash and SDRAM.



Note1: CSZA is a chip-select for not only bank0 to 127 of LOCAL-Z but also COMMON-Z.

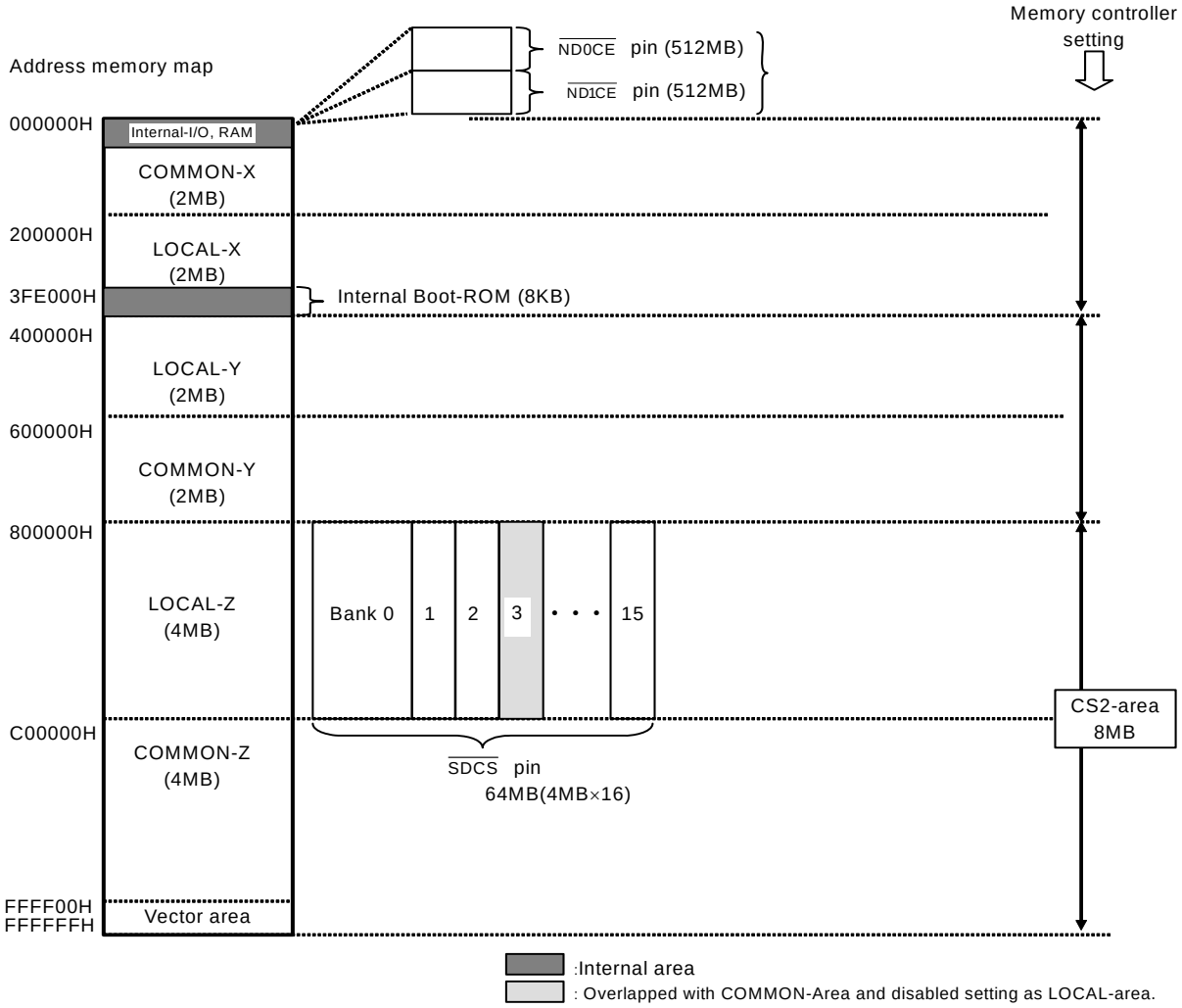
Note2: In case of connect SDRAM to Y-area, 64MB(2MB×32) is maximum

Figure 3.9.1 Recommendation memory map for maximum specification (Logical address)



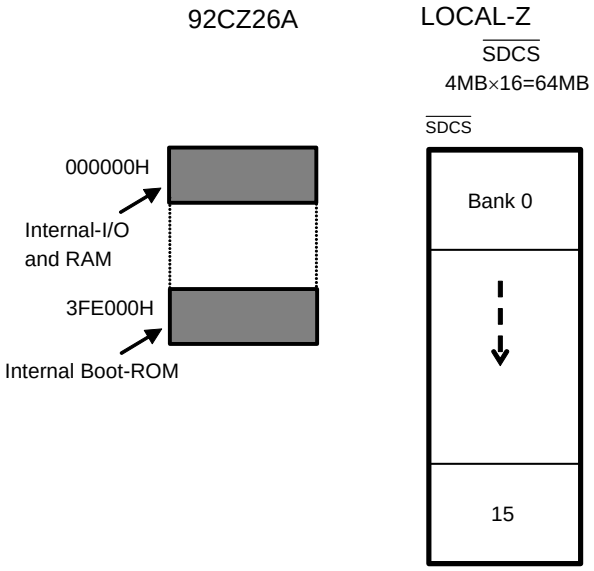
Note: In case of connect SDRAM to Y-area, 64MB(2MB×32) is maximum

Figure 3.9.2 Recommendation memory map for maximum specification (Physical address)



Note: In case of connect SDRAM to Z-area, 64MB (4MB×16) is maximum

Figure 3.9.3Recommendation memory map for simple system (Logical address)



Note: In case of connect SDRAM to Z-area, 64MB(4MB×16) is maximum

Figure 3.9.4 Recommendation memory map for simple system (Physical address)

3.9.2 Control register

There are 24-registers for MMU. They are prepared for 8-purpose using (as Program, read-data, write-data and LCDC-display-data, source-data for odd/even number channel DMA, destination-data for odd/even number channel DMA), and 3-local area (LOCAL-X, Y and Z). These 8-purpose registers can access a data accessed easily.

(How to use)

At first, set enable register and using bank-number of each LOCAL register. In that case, set a combination pin and memory setting to the Ports and Memory controller. After that, if CPU or LCDC access to logical address of the local area, MMU converts logical address to physical address according to the bank number, and output it. The physical address is output to the external address bus pin. By this operation, accessing to external memory becomes possible. And, if accessed same logical address, physical address is changed by bank that be set to register in program, and enable accessing that memory of other bank.

Note:

1) When set the bank page, it inhibit to set overlapped area with common area (because Local area and common area shows same physical address)
2) In the LOCAL-area, changing Program bank number (LOCALPX, Y or Z) is disabled. Program bank setting of each local area must change in common area. (But bank setting of data-Read, data-Write and LCDC-display data can change also in local area.)
3) After data bank number (LOCALRn, LOCALWn or LOCALLn, LOCALEDn, LOCALSn, LOCALODn; "n" means X, Y or Z) register is set by an instruction, don't access its memory by next instruction because of some clocks are needed to be effective MMU setting. In this case, insert dummy instruction which accesses SFR or another memory between them like following example.

```
(Example)   ld      xix, 200000h      ;
            ldw     (localrx), 8001h  ;  read-data bank number is set
            ldw     wa, (localrx)     ;  <---- Inserted Dummy instruction which accesses SFR
            ldw     wa, (xix)         ;  instruction which reads bank1 of local-X area.
```

4) When LOCAL-Z area is used, Chip select signal $\overline{\text{CSZA}}$ should be assigned to P82-pin.
In this case, $\overline{\text{CSZA}}$ works as chip select signal for not only bank0 to 15 but also COMMON-Z.
But for it, following setting after reset is needed before P82 setting.

```
ldw     (localpz), 8000h  ; LOCAL-Z Bank enable for program
ldw     (localrz), 8000h  ; LOCAL-Z Bank enable for data read
ldw     (localwz), 8000h  ; LOCAL-Z Bank enable for data write          (*1)
ldw     (locallz), 8000h  ; LOCAL-Z Bank enable for LCD display memory  (*2)
ld      (p8fc), -----0--B ; Assign P82 to CSZA
ld      (p8fc2), -----1--B ;
```

(*1) If COMMON-Z area is not used as data write memory, this setting is not needed.
(*2) If COMMON-Z area is not used as LCD display memory, this setting is not needed.

3.9.2.1 Program bank register

The bank number used as program memory is set to these registers. In certain bank, cannot diverge directly to different bank of same local area. To change program bank number in the same local area is disable.

LOCAL-X register for Program

		7	6	5	4	3	2	1	0
LOCALPX (880H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(881H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for Program

		7	6	5	4	3	2	1	0
LOCALPY (882H)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(883H)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for Program

		7	6	5	4	3	2	1	0
LOCALPZ (884H)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(885H)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 010000000 to 011111111 CSZB 100000000 to 101111111 CSZC 110000000 to 111111111 CSZD						

3.9.2.2 LCD display bank register

The bank page used as LCD display memory is set to these registers. Since the bank register for CPU and LCDC are prepared independently, the bank page for CPU (Program, Read-data, write-data) can change during LCD display on.

LOCAL-X register for LCD

		7	6	5	4	3	2	1	0
LOCALLX (888H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(889H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000～011111111 CSXA 100000000～111111111 CSXB						

LOCAL-Y register for LCD

		7	6	5	4	3	2	1	0
LOCALLY (88AH)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(88BH)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for LCD

		7	6	5	4	3	2	1	0
LOCALLZ (88CH)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(88DH)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 100000000 to 101111111 CSZC 010000000 to 011111111 CSZB 110000000 to 111111111 CSZD						

3.9.2.3 Read-data bank register

The bank number used as read-data memory is set to these registers. The following is an example which read data bank register of LOCAL-X is set to "1". When "ldw wa, (xix)" instruction is executed, the bank becomes effective at only read data (operand) for xix address.

(Example)

```
ld      xix, 200000h      ;
ld      (localrx), 8001h  ; Set Read data bank.
ldw     wa,(localrx)      ; <----Insert dummy instruction that access to SFR
ldw     wa, (xix)         ; Read bank1 of LOCAL-X area
```

LOCAL-X register for read

		7	6	5	4	3	2	1	0
LOCALRX (890H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(891H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for read

		7	6	5	4	3	2	1	0
LOCALRY (892H)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(893H)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for read

		7	6	5	4	3	2	1	0
LOCALRZ (894H)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(895H)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 010000000 to 011111111 CSZB 100000000 to 101111111 CSZC 110000000 to 111111111 CSZD						

3.9.2.4 Write-data bank register

The bank number used as write data memory is set to these registers. The following is an example which data bank register of LOCAL-X is set to "1". When "ldw (xix), wa" instruction is extended, the bank becomes effective at only cycle for xix address.

(Example)

```
ld      xix, 200000h      ;
ld      (localwx), 8001h  ; Set Write data bank.
ldw     wa, (localwx)     ; <----Insert dummy instruction that access to SFR
ldw     (xix), wa         ;  Write to bank1 of LOCAL-X area
```

LOCAL-X register for write

LOCALWX (898H)

	7	6	5	4	3	2	1	0
bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)							

(899H)

	15	14	13	12	11	10	9	8
bit Symbol	LXE							X8
Read/Write	R/W							R/W
After reset	0							0
Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for write

LOCALWY (89AH)

	7	6	5	4	3	2	1	0
bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Set BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)					

(89BH)

	15	14	13	12	11	10	9	8
bit Symbol	LYE							
Read/Write	R/W							
After reset	0							
Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for write

LOCALWZ (89CH)

	7	6	5	4	3	2	1	0
bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)							

(89DH)

	15	14	13	12	11	10	9	8
bit Symbol	LZE							Z8
Read/Write	R/W							R/W
After reset	0							0
Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 100000000 to 101111111 CSZC 010000000 to 011111111 CSZB 110000000 to 111111111 CSZD						

3.9.2.5 DMA-function bank register

In addition to functioning as read/write function of CPU, this LSI can also function which transfer data at high-speed by internal DMAC becoming bus master. (Please refer to DMAC section)

In Bank for only DMA that different from Bank for CPU or LCDC display data, although condition of program bank, read-bank and write-bank for CPU, bank of Source address and Destination address are enable during operate DMA.

DMAC which assignment is possible in this LSI is 5-channel. But bank controller is 2-type. Even-channel of DMA-channel 0, 2 and 4 become E-group (ES and ED group), odd-channel of DMA-channel 1and 3 become O-group (OS and OD group). Assignment every channel is disable in same group.

Following shows examples of setting bank for DMA_Source address to 1 in LOCALX area and setting bank for DMA_Destination address to 2 in LOCALY area. If Source address which set to XXX by using DMA function was set to LOCALX-area and Destination address was set to LOCALY-area, when DMA of channel 0 is start, LOCALX bank1 is set to source and LOCALY bank2 is set to destination.

(Example)

```
ldw     (localesx), 8001h  ; Set DMA source bank for channel 0
ldw     (localedy), 8002h  ; Set DMA destination bank for channel 0
DMA channel 0 start
```

LOCAL-X register for even-group DMA source

		7	6	5	4	3	2	1	0
LOCALESX (8A0H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8A1H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for even-group DMA source

		7	6	5	4	3	2	1	0
LOCALESY (8A2H)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(8A3H)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for even-group DMA source

		7	6	5	4	3	2	1	0
LOCALESZ (8A4H)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8A5H)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 100000000 to 101111111 CSZC 010000000 to 011111111 CSZB 110000000 to 111111111 CSZD						

LOCAL-X register for even-group DMA destination

		7	6	5	4	3	2	1	0
LOCALEDX (8A8H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8A9H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for even-group DMA destination

		7	6	5	4	3	2	1	0
LOCALEDY (8AAH)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(8ABH)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for even-group DMA destination

		7	6	5	4	3	2	1	0
LOCALEDZ (8ACH)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8ADH)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 100000000 to 101111111 CSZC 010000000 to 011111111 CSZB 110000000 to 111111111 CSZD						

LOCAL-X register for odd-group DMA source

		7	6	5	4	3	2	1	0
LOCALOSX (8B0H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8B1H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for odd-group DMA source

		7	6	5	4	3	2	1	0
LOCALOSY (8B2H)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(8B3H)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for odd-group DMA source

		7	6	5	4	3	2	1	0
LOCALOSZ (8B4H)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8B5H)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 100000000 to 101111111 CSZC 010000000 to 011111111 CSZB 110000000 to 111111111 CSZD						

LOCAL-X register for odd-group DMA destination

		7	6	5	4	3	2	1	0
LOCALODX (8B8H)	bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8B9H)	bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALX 0: Disable 1: Enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y register for odd-group DMA destination

		7	6	5	4	3	2	1	0
LOCALODY (8BAH)	bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
		15	14	13	12	11	10	9	8
(8BBH)	bit Symbol	LYE							
	Read/Write	R/W							
	After reset	0							
	Function	BANK for LOCALY 0: Disable 1: Enable							

LOCAL-Z register for odd-group DMA destination

		7	6	5	4	3	2	1	0
LOCALODZ (8BCH)	bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
		15	14	13	12	11	10	9	8
(8BDH)	bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	After reset	0							0
	Function	BANK for LOCALZ 0: Disable 1: Enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000 to 001111111 CSZA 100000000 to 101111111 CSZC 010000000 to 011111111 CSZB 110000000 to 111111111 CSZD						

3.9.3 Setting example

This is in case of using like following condition.

No.	Used as	Memory	Setting	MMU-area	Logical address	Physical address
(a)	Main Routine	NOR-Flash (16MB, 1pcs)	$\overline{CSZA}$, 32bit, 1wait	COMMON-Z	C00000H to FFFFFFH	
(b)	Character-ROM			Bank0 in LOCAL-Z	800000H to BFFFFFH	000000H to 3FFFFFH
(c)	Sub Routine	SRAM (16MB, 1pcs)	$\overline{CS1}$, 16bit, 0wait	Bank0 in LOCAL-Y	400000H to 5FFFFFH	000000H to 1FFFFFH
(d)	LCD Display-RAM			Bank1 in LOCAL-Y		200000H to 3FFFFFH
(e)	Stack-RAM	Internal-RAM (288KB)	---- (32bit, 2-1-1-1clk)	Bank2 in LOCAL-Y	002000H to 049FFFH	

(a) Main routine (COMMON-Z)

Logical Address	Physical Address	No	Instruction	Comment
		1	org C00000H	;
C00000H	<-(Same)	2	ldw (mamr2),80FFH	; CS2 800000-ffffff/8MB
C000xxH	<-	3	ldw (b2csl), C222H	; CS2 32bit ROM, 1wait
		4	ldw (mamr1),40FFH	; CS1 400000-7fffff/4MB
		5	ldw (b1csl), 8111H	; CS1 16bit RAM, 0wait
		5.1	ldw (localpz),8000H	; Enable LOCAL-Z Bank for program
		5.2	ldw (localrz),8000H	; Enable LOCAL-Z Bank for read-data
		6	ld (p8fc), 02H	;
		7	ld (p8fc2), 04H	;
		9	ld xsp,48000H	; Stack Pointer = 48000H
		10	ldw (localpy),8000H	; Bank0 in LOCAL-Y is set as Program bank for sub routine
		11	:	;
C000yyH	<-	12	call 400000H	; Call Sub routine
		13	:	;
		14	:	;
		15	:	;

- From No.2 to No.8 instructions are setting of Ports and Memory controller.
- No.9 is a setting for stack pointer. It is assigned to internal-RAM.
- No.10 is a setting to execute for No.12's instruction.
- No.12 is an instruction to call sub routine. When CPU outputs 400000H address, MMU will convert and output 000000H physical address to external address bus: A23 to A0. And $\overline{CS1}$ for SRAM will be asserted because of logical address is in an area for CS1 at the same time. By these instructions, CPU cans brunch to sub-routine.

  (Note: This example is based on sub routine program is already written on SRAM.)

(b) Sub routine (Bank-0 in LOCAL-Y)

Logical address	Physical address	No	Instruction	Comment
		16	org 400000H	;
400000H	000000H	17	ldw (localwy),8001H	; Bank1 in LOCAL-Y is set to write-data for LCD Display RAM
4000xxH	0000xxH	18	ldw (locally), 8001H	; Bank1 in LOCAL-Y is set as LCD display RAM
		19	ldw (localrz), 8001H	; Bank0 in LOCAL-Z is set as read-data for Character-RAM
		20	ld xiy,800000H	; Index address register for read Character-ROM
		21	ld wa,(xiy)	; Read Character-ROM
		22	:	; Convert it to display-data
		23	ld (localpy), 82H	;
		24	ld xix, 400000H	; Index address register for write LCD Display data
		25	ld (xix), bc	; Write LCD Display data
		26	:	; Set LCD Controller
		27	:	;
		28	ld xiz, 400000H	; Set LCD Start address to LCDC
		29	ld (lsarcl), xiz	;
		30	ld (lcdctl0),01H	; Start LCD Display operation
		31	:	;
5000yyH	1000yyH	32	ret	;

- No.17 and No.18 are setting for Bank-1 of LOCAL-Y. In this case, LCD Display data is written to SRAM by CPU. So, (LOCALWY) and (LOCALLY) should be set to same bank-1.
- No.19 is a setting for Bank-0 of LOCAL-Z to read data from character-ROM.
- No.20 and No.21 are instructions to read data from character-ROM. When CPU outputs 800000H address, this MMU will convert and output 000000H address to external address bus: A23 to A0. And /CSZA for NOR-Flash will be asserted because of logical address is in an area for CS2 at the same time.

  By these instructions, CPU can read data from character ROM.
- No.23 is an instruction which changes Program bank number in the LOCAL-area. This setting is disabled.
- No.24 and No.25 are instructions to write data to SRAM. When CPU outputs 400000H address, this MMU will convert and output 200000H address to external address bus: A23 to A0. And /CS1 for SRAM will be asserted because of logical address is in an area for CS1 at the same time.

  By these instructions, CPU can write data to SRAM.
- No.28 and No.29 are setting to set LCD starting address to LCD Controller. When LCDC outputs 400000H address in DMA-cycle, this MMU will convert and output 200000H address to external address bus: A23 to A0. And /CS1 for SRAM will be asserted because of logical address is in an area for CS1 at the same time.

  By these instructions, LCDC can read data from SRAM.
- No.30 is an instruction to start LCD display operation.

3.10 SDRAM Controller (SDRAMC)

The TMP92CZ26A incorporates an SDRAM controller (SDRAMC) for accessing SDRAM that can be used as data memory, program memory, or display memory.

The SDRAMC has the following features:

(1) Supported SDRAM

Data rate type	: SDR (single data rate) type only
Memory capacity	: 16 / 64 / 128 / 256 / 512 Mbits
Number of banks	: 2 banks / 4 banks
Data bus width	: 16 bits
Read burst length	: 1 word / full page
Write mode	: Single mode / Burst mode

(2) Supported initialization sequence commands

Precharge All command

Eight Auto Refresh commands

Mode Register Set command

(3) Access mode

	CPU Cycle	HDMA Cycle	LCDC Cycle
Burst length	1 word	1 word or full page selectable	Full page
Addressing mode	Sequential	Sequential	Sequential
CAS latency (clock)	2	2	2
Write mode	Single	Single or burst selectable	

(4) Access cycles

CPU access cycles

Read cycle	: 1 word, 4-3-3-3 states (minimum)
Write cycle	: Single, 3-2-2-2 states (minimum)
Data size	: 1 byte / 1 word / 1 long-word

HDMA access cycles

Read cycle	: 1 word, 4-3-3-3 states / full page, 4-1-1-1 states (minimum)
Write cycle	: Single, 3-2-2-2 states (minimum) / burst, 2-1-1-1 states (minimum)
Data size	: 1 byte / 1 word / 1 long-word

LCDC access cycles

Read cycle	: Full page, 4-1-1-1 states (minimum)
Data size	: 1 word

(5) Auto generation of refresh cycles

- Auto Refresh is performed while the SDRAM is not being accessed.
- The Auto Refresh interval is programmable.
- The Self Refresh function is also supported.

Note: The SDRAM address area is determined by the CS1 or CS2 setting of the memory controller. However, the number of bus cycle states is controlled by the SDRAMC.

3.10.1 Control Registers

The SDRAMC has the following control registers.

SDRAM Access Control Register

SDACR (0250H)

	7	6	5	4	3	2	1	0
Bit symbol	SRDS	–	SMUXW1	SMUXW0	SPRE			SMAC
Read/Write	R/W							R/W
After reset	1	0	0	0	0			0
Function	Read data shift function 0: Disable 1: Enable	Always write “0”	Address multiplex type 00: Type A (A9-) 01: Type B (A10-) 10: Type C (A11-) 11: Reserved		Read/Write commands 0: Without auto pre-charge 1: With auto precharge			SDRAM controller 0: Disable 1: Enable

SDRAM Command Interval Setting Register

SDCISR (0251H)

	7	6	5	4	3	2	1	0
Bit symbol		STMRD	STWR	STRP	STRCD	STRC2	STRC1	STRC0
Read/Write		R/W						
After reset		1	1	1	1	1	0	0
Function		TMRD 0: 1 CLK 1: 2 CLK	TWR 0: 1 CLK 1: 2 CLK	TRP 0: 1 CLK 1: 2 CLK	TRCD 0: 1 CLK 1: 2 CLK	TRC 000: 1 CLK 100: 5 CLK 001: 2 CLK 101: 6 CLK 010: 3 CLK 110: 7 CLK 011: 4 CLK 111: 8 CLK		

SDRAM Refresh Control Register

SDRCR (0252H)

	7	6	5	4	3	2	1	0
Bit symbol	–			SSAE	SRS2	SRS1	SRS0	SRC
Read/Write	R/W			R/W				
After reset	0			1	0	0	0	0
Function	Always write “0”			Self Refresh auto exit function 0:Disable 1:Enable	Refresh interval 000: 47 states 100: 468 states 001: 78 states 101: 624 states 010: 156 states 110: 936 states 011: 312 states 111: 1248 states			Auto Refresh 0:Disable 1:Enable

SDRAM Command Register

SDCMM (0253H)

	7	6	5	4	3	2	1	0
Bit symbol						SCMM2	SCMM1	SCMM0
Read/Write						R/W		
After reset						0	0	0
Function						Command issue (Note 1) (Note 2) 000: Don't care 001: Initialization sequence a. Precharge All command b. Eight Auto Refresh commands c. Mode Register Set command 010: Precharge All command 100: Reserved 101: Self Refresh Entry command 110: Self Refresh Exit command Others: Reserved		

Note 1: <SCMM2:0> is automatically cleared to "000" after the specified command is issued. Before writing the next command, make sure that <SCMM2:0> is "000". In the case of the Self Refresh Entry command, however, <SCMM2:0> is not cleared to "000" by execution of this command. Thus, this register can be used as a flag for checking whether or not Self Refresh is being performed.

Note 2: The Self Refresh Exit command can only be specified while Self Refresh is being performed.

SDRAM HDMA Burst Length Select Register

SDBLS (0254H)

	7	6	5	4	3	2	1	0
Bit symbol			SDBL5	SDBL4	SDBLS	SDBL2	SDBL1	SDBL0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			For HDMA5	For HDMA4	For HDMA3	For HDMA2	For HDMA1	For HDMA0
			HDMA burst length 0: 1 Word read / Single write 1: Full page read / Burst write					

Figure 3.10.1 Control Registers

3.10.2 Operation Description

(1) Memory access control

The SDRAMC is enabled by setting SDACR<SMAC> to "1".

When one of the bus masters (CPU, LCDC, DMAC) generates a cycle to access the SDRAM address area, the SDRAMC outputs SDRAM control signals.

Figure3.10.2 to Figure3.10.5 shows the timing for accessing the SDRAM. The number of SDRAM access cycles is controlled by the SDRAMC and does not depend on the number of waits controlled by the memory controller.

(a) Command issue function

The SDRAMC issues commands as specified by the SDCMM register. The SDRAMC also issues commands automatically for each SDRAM access cycle generated by each bus master.

Table 3.10.1 shows the commands that are issued by the SDRAMC.

Table 3.10.1 Commands Issued by the SDRAMC

Command	CKE_{n-1}	CKE_n	SDxxDQM	A10	A15-11 A9-0	$\overline{SDCS}$	$\overline{SDRAS}$	$\overline{SDCAS}$	$\overline{SDWE}$
Bank Activate	H	H	H	RA	RA	L	L	H	H
Precharge All	H	H	H	H	X	L	L	H	L
Read	H	H	L	L	CA	L	H	L	H
Read with Auto Precharge	H	H	L	H	CA	L	H	L	H
Write	H	H	L	L	CA	L	H	L	L
Write with Auto Precharge	H	H	L	H	CA	L	H	L	L
Mode Register Set	H	H	H	L	M	L	L	L	L
Burst Stop	H	H	H	X	X	L	H	H	L
Auto Refresh	H	H	H	X	X	L	L	L	H
Self Refresh Entry	H	L	H	X	X	L	L	L	H
Self Refresh Exit	L	H	H	X	X	H	H	H	H

Note 1: H = High level, L = Low level, RA = Row address, CA = Column address, M = Mode data, X = Don't care

Note 2: CKE_n = CKE level in the command input cycle
CKE_{n-1} = CKE level in a cycle immediately before the command input cycle

(b) Address multiplex function

In access cycles, the A0 to A15 pins output low/column multiplexed addresses. The multiplex width is set by SDACR<SMUXW1:0>. Table3.10.2 shows the relationship between the multiplex width and low/column addresses.

Table3.10.2 Address Multiplex

92CZ26A Pin Name	SDRAM Access Cycle Address			
	Row Address			Column Address
	Type A <SMUXW> = 00	Type B <SMUXW> = 01	Type C <SMUXW> = 10	
A0	A9	A10	A11	A1
A1	A10	A11	A12	A2
A2	A11	A12	A13	A3
A3	A12	A13	A14	A4
A4	A13	A14	A15	A5
A5	A14	A15	A16	A6
A6	A15	A16	A17	A7
A7	A16	A17	A18	A8
A8	A17	A18	A19	A9
A9	A18	A19	A20	A10
A10	A19	A20	A21	AP *
A11	A20	A21	A22	Row Address
A12	A21	A22	A23	
A13	A22	A23	EA24	
A14	A23	EA24	EA25	
A15	EA24	EA25	EA26	

*AP: Auto Precharge

(c) Burst length

When the CPU accesses the SDRAM, the burst length is fixed to 1-word read/single write. When the LCDC accesses the SDRAM, the burst length is fixed to full page.

The burst length can be selected for SDRAM read and write accesses by HDMA if the following conditions are satisfied:

- The HDMA transfer mode is an increment mode.
- Transfers are made between the SDRAM and internal RAM or internal I/O.

In other cases, HDMA operation can only be performed in 1-word read/single write mode. Use SDBLS<SDBL5:0> to set the burst length for each HDMA channel.

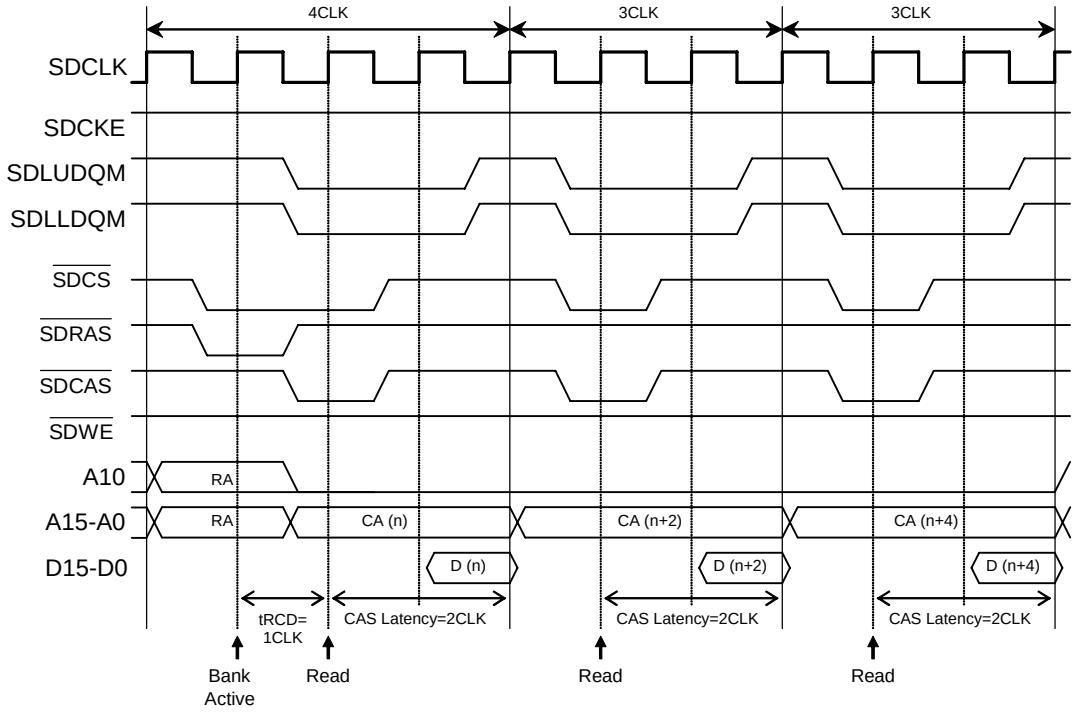


Figure3.10.2 1-Word Read Cycle Timing

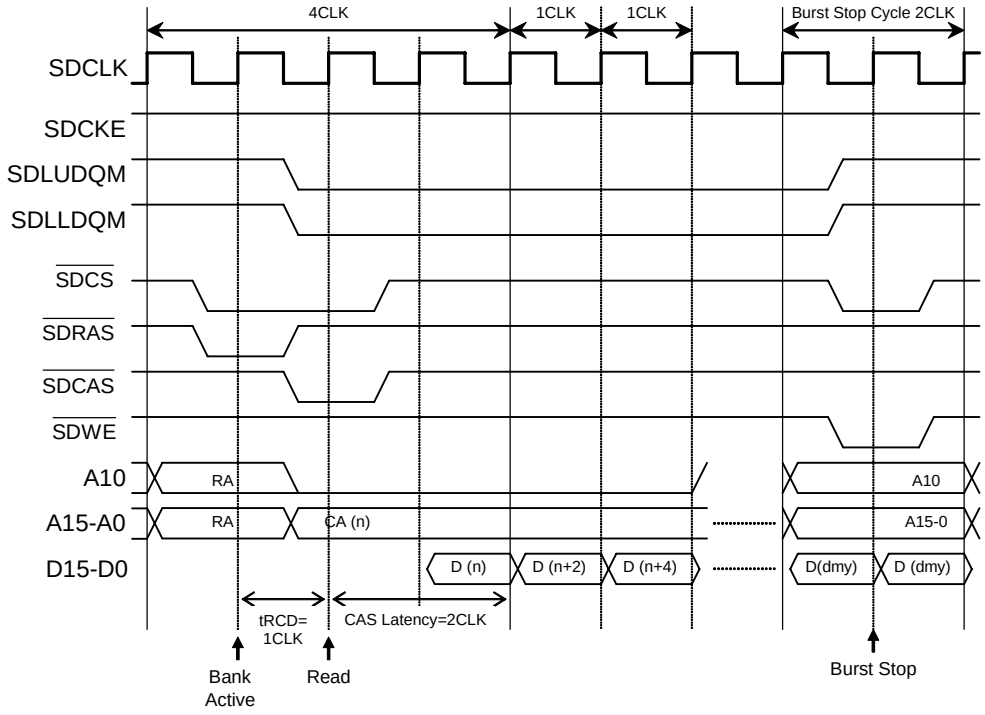


Figure3.10.3 Full-Page Read Cycle Timing

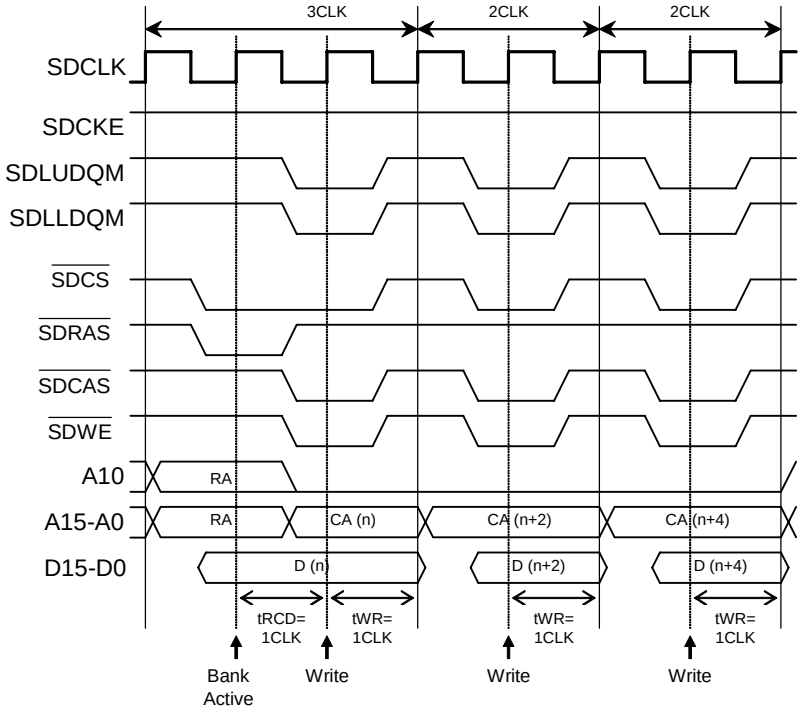


Figure3.10.4 Single Write Cycle Timing

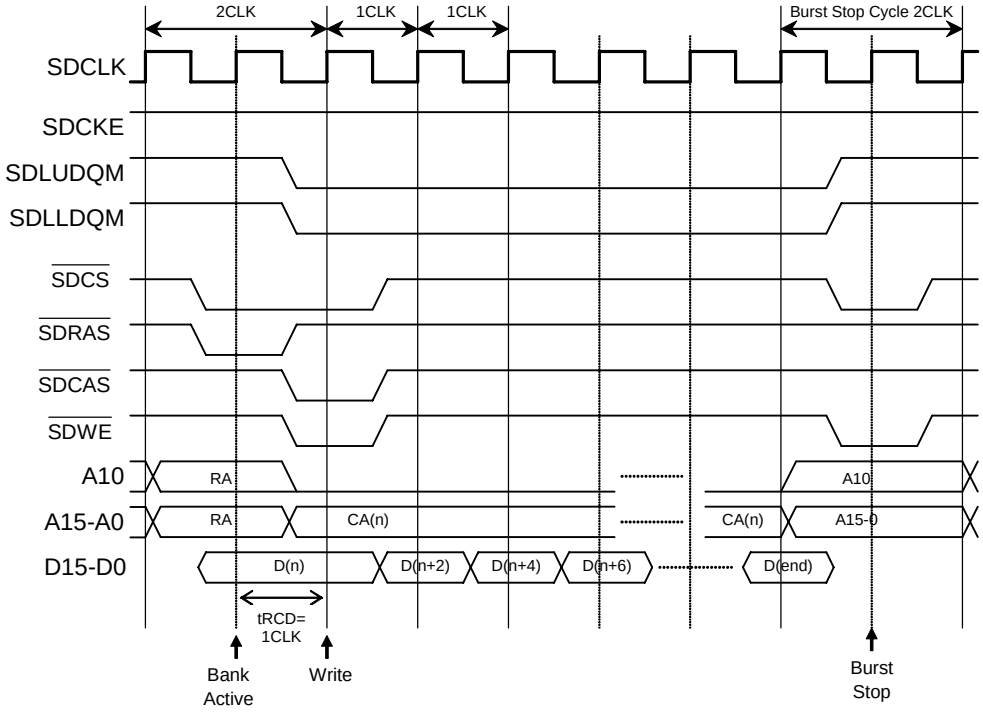


Figure3.10.5 Burst Write Cycle Timing

(2) Execution of instructions on SDRAM

The CPU can execute instructions that are stored in the SDRAM. However, the following operations cannot be performed.

a) Executing the HALT instruction
b) Changing the clock gear setting
c) Changing the settings in the SDACR, SDCMM, and SDCISR registers

These operations, if needed, must be executed by branching to other memory such as internal RAM.

(3) Command interval adjustment function

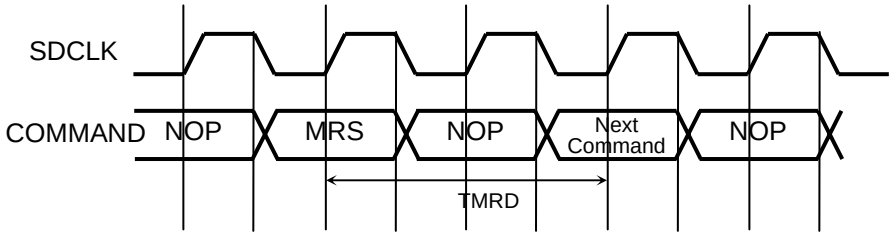
Command execution intervals can be adjusted for each command. This function enables the SDRAM to be accessed at optimum cycles even if the operationg frequency is changed by clock gear.

Command intervals should be set in the SDCISR register according to the operating frequency of the TMP92CZ26A and the AC specifications of the SDRAM.

The SDCICR register must not be changed while the SDRAM is being accessed.

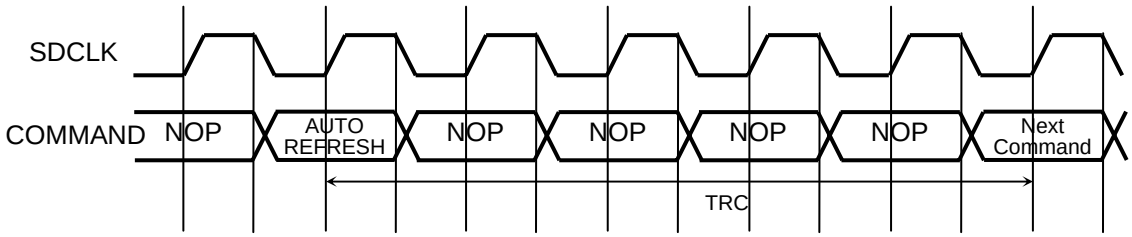
The timing waveforms for various cases are shown below.

(a) Mode Register Set command



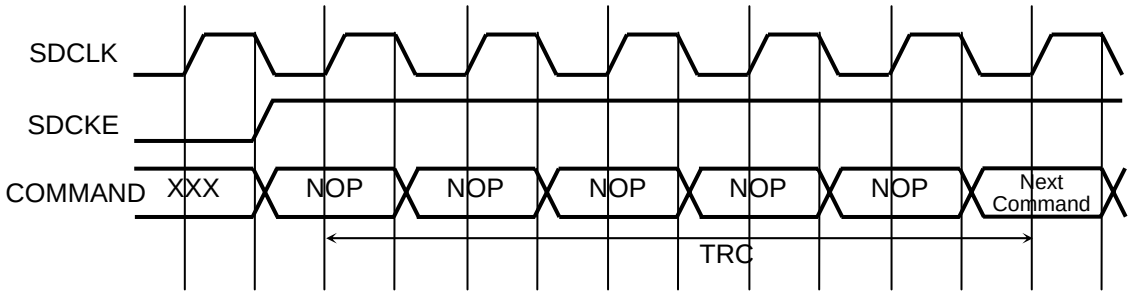
*TMRD=2CLK (SDCISR<STMRD>=”1”)

(b) Auto Refresh command



*TRC=5CLK (SDCISR<STRC2:0>=”100”)

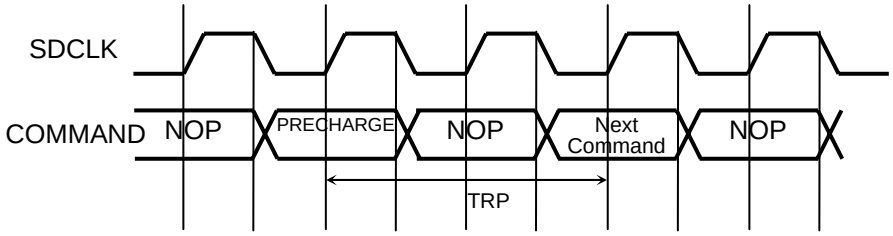
(c) Self Refresh Exit



*TRC=5CLK (SDCISR<STRC2:0>= “100”)

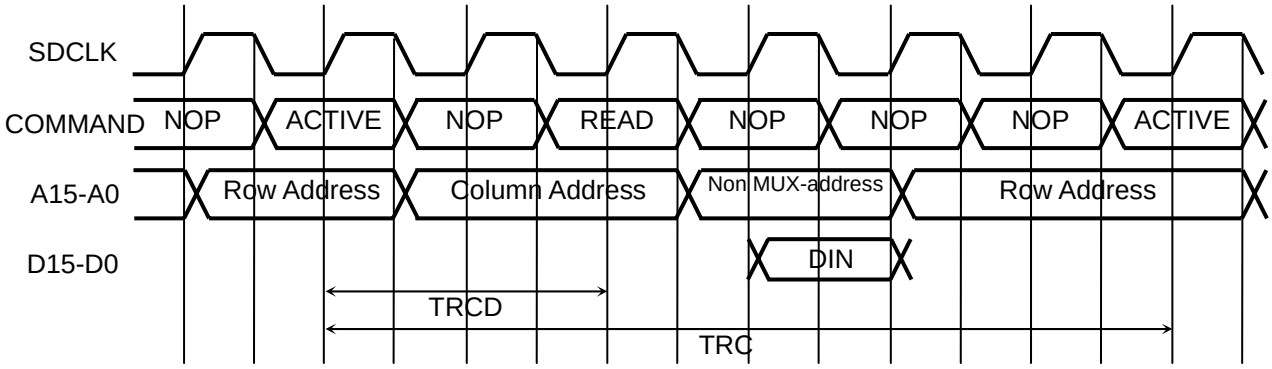
Exit Self Refresh

(d) Precharge command



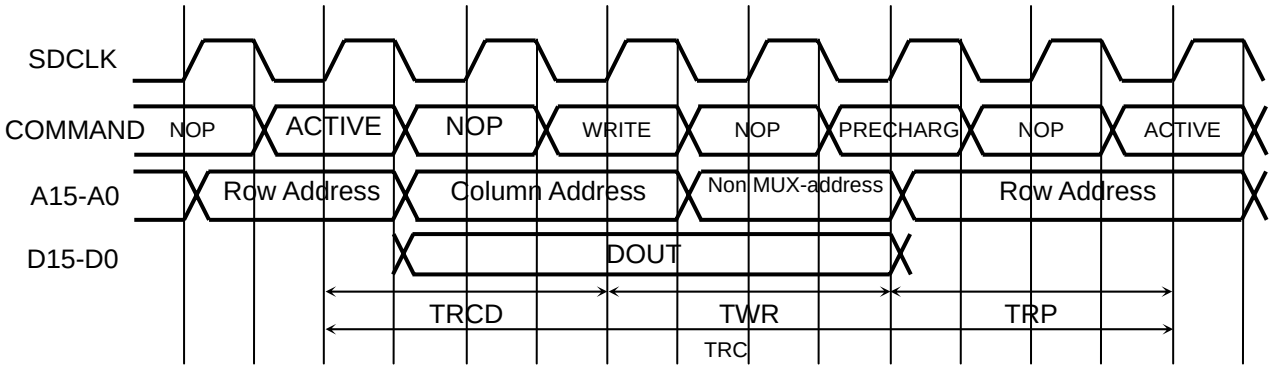
*TRP=2CLK (SDCISR<STRP>= “1”)

(e) Read cycle



*TRCD=2CLK (SDCISR<STRCD>= “1”)
*TRC=6CLK (SDCISR<STRC2:0>= “101”)

(f) Write cycle

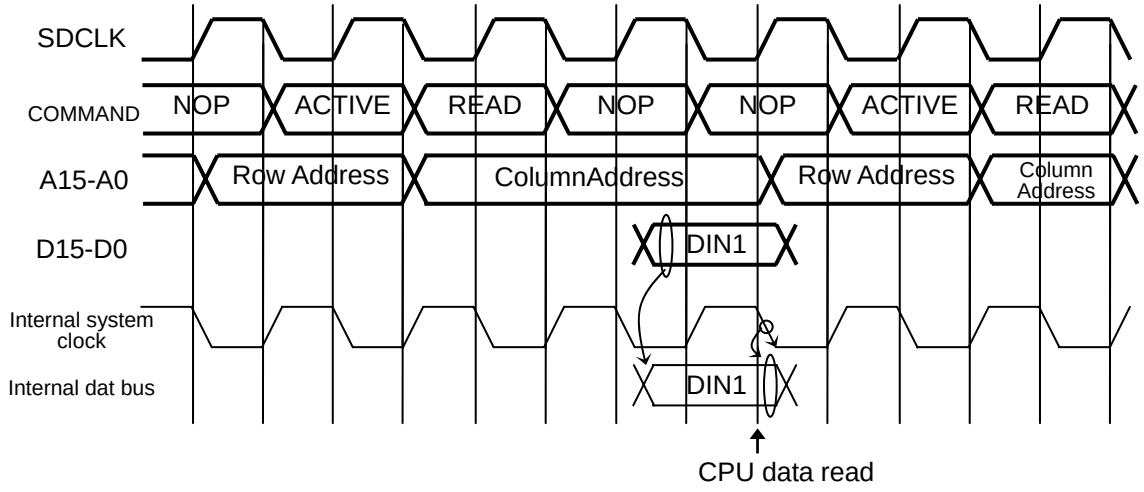


*TRCD=2CLK (SDCISR<STRCD>= “1”)
*TWR=2CLK (SDCISR<STWR>= “1”)
*TRP=2CLK (SDCISR<STRP>= “1”)
*TRC=6CLK (SDCISR<STRC2:0>= “101”)

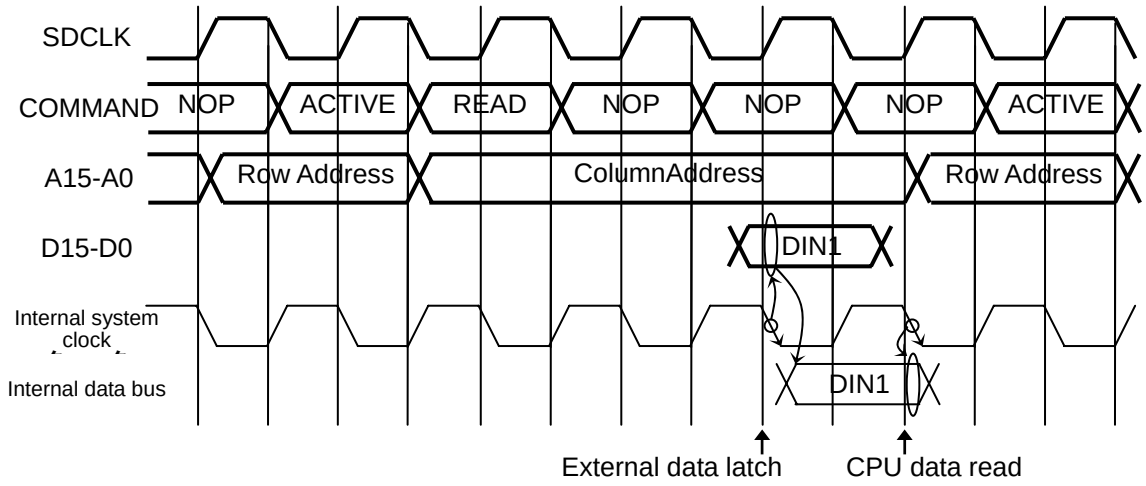
(4) Read data shift function

If the AC specifications of the SDRAM cannot be satisfied when data is read from the SDRAM, the read data can be latched in a port circuit so that the CPU can read the data in the next state. When this read data shift function is used, the read cycle requires additional one state. The write cycle is not affected. The timing waveforms for various cases are shown below.

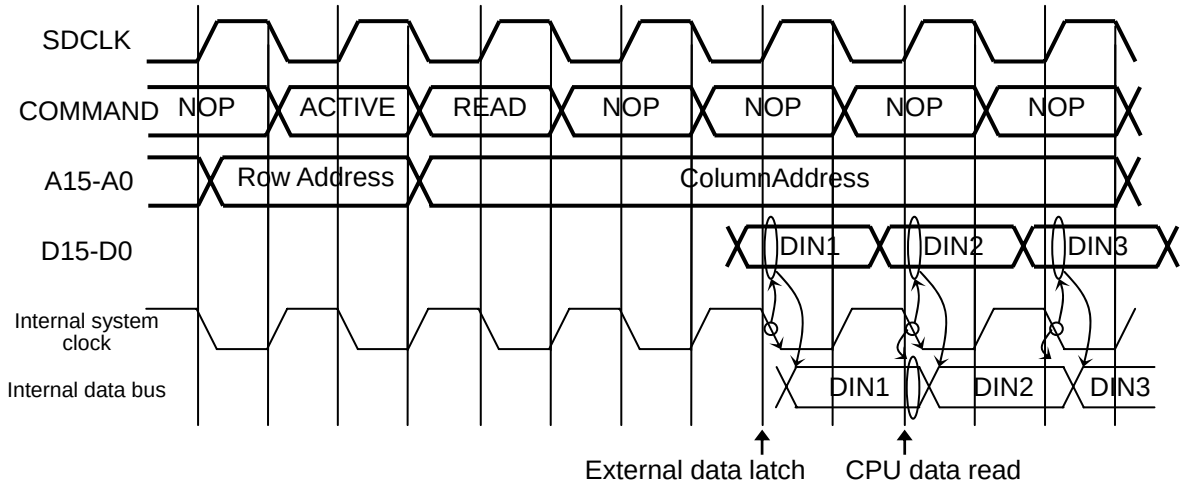
(a) 1-word read, the read data shift function disabled (SDACR<SRCS> = "0")



(b) 1-word read, the read data shift function enabled (SDACR<SRDS> = "1", <SRDSCK>="0")



(c) Full-page read, the read data shift function enabled (SDACR<SRDS> = “1”, <SRDSCK> = “0”)



(5) Read/Write commands

The Read/Write commands to be used in 1-word read/single write mode can be specified by using SDACR<SPRE>.

When SDACR<SPRE> is set to “1”, the Read/Write commands are executed with Auto Precharge. When Auto Precharge is enabled, the SDRAM is automatically precharged internally at every access cycle. Thus, the SDRAM is always in a “bank idle” state while it is not being accessed. This helps reduce the power consumption of the SDRAM but at the cost of degradation in performance as the Bank Active command is needed at every access cycle.

When SDACR<SPRE> is set to “0”, the Read/Write commands are executed without Auto Precharge. In this case, the SDRAM is not precharged at every access cycle and is always in a “bank active” state. This increases the power consumption of the SDRAM, but improves performance as there is no need to issue the Bank Active command at every access cycle. If an access is made to outside the SDRAM page boundaries or if the Auto Refresh command is issued, the SDRAMC automatically issues the Precharge All command.

(6) Refresh control

The TMP92CZ26A supports two kinds of refresh commands: Auto Refresh and Self Refresh.

(a) Auto Refresh

When SDRCR<SRC> is set to "1", the Auto Refresh command is automatically issued at intervals specified by SDRCR<SRS2:0>. The Auto Refresh interval can be specified in a range of 47 states to 1248 states (0.78 μs to 20.8 μs at f_{SYS} = 60 MHz).

The CPU operation (instruction fetch and execution) is halted while the Auto Refresh command is being executed. Figure3.10.6 shows the Auto Refresh cycle timing, and Table3.10.3 shows the Auto Refresh interval settings. The Auto Refresh function cannot be used in IDLE1 and STOP modes. In these modes, use the Self Refresh function to be explained next.

Note: A system reset disables the Auto Refresh function.

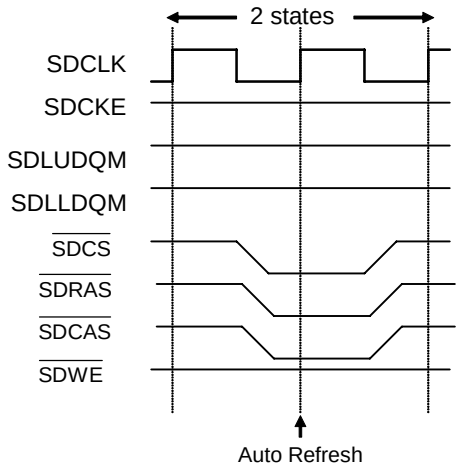


Figure3.10.6 Auto Refresh Cycle Timing

Table3.10.3 Auto Refresh Intervals

Unit [μs]

SDRCR<SRS2:0>			Auto Refresh Interval (states)	Frequency (System Clock)					
SRS2	SRS1	SRS0		6 MHz	10 MHz	20 MHz	40 MHz	60 MHz	80 MHz
0	0	0	47	7.8	4.7	2.4	1.18	0.78	0.59
0	0	1	78	13.0	7.8	3.9	1.95	1.30	0.98
0	1	0	156	26.0	15.6	7.8	3.90	2.60	1.95
0	1	1	312	52.0	31.2	15.6	7.80	5.20	3.90
1	0	0	468	78.0	46.8	23.4	11.70	7.80	5.85
1	0	1	624	104.0	62.4	31.2	15.60	10.40	7.80
1	1	0	936	156.0	93.6	46.8	23.40	15.60	11.70
1	1	1	1248	208.0	124.8	62.4	31.20	20.80	15.60

(b) Self Refresh

The Self Refresh Entry command is issued by setting SDCMM<SCMM2:0> to “101”. Figure3.10.7 shows the Self Refresh cycle timing. Once Self Refresh is started, the SDRAM is refreshed internally without the need to issue the Auto Refresh command.

Note 1: When standby mode is released by a system reset, the I/O registers are initialized and the Self Refresh state is exited. Note that the Auto Refresh function is also disabled at this time.

Note 2: The SDRAM cannot be accessed while it is in the Self Refresh state.

Note 3: To execute the HALT instruction after the Self Refresh Entry command, insert at least 10 bytes of NOP or other instructions between the instruction to set SDCMM<SCMM2:0> to “101” and the HALT instruction.

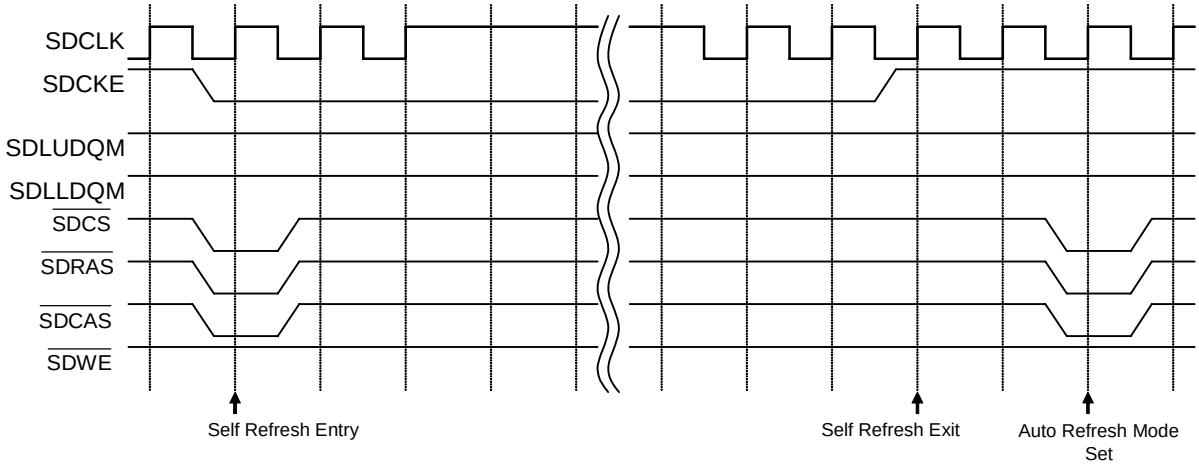


Figure3.10.7 Self Refresh Cycle Timing

The Self Refresh state can be exited by the Self Refresh Exit command. The Self Refresh Exit command is executed when SDCMM<SCMM2:0> is set to “110”. It is also executed automatically in synchronization with HALT mode release. In either of these two cases, Auto Refresh is performed immediately after the Self Refresh state is exited. Then, Auto Refresh is executed at specified intervals. Exiting the Self Refresh state clears SDCMM<SCMM2:0> to “000”.

Setting SDRCR<SSAE> to “0” disables automatic execution of the Self Refresh Exit command in synchronization with HALT release. The auto exit function should also be disabled in cases where the SDRAM operation requirements cannot be met as the operation clock frequency is reduced by clock gear down, as shown in Figure3.10.8.

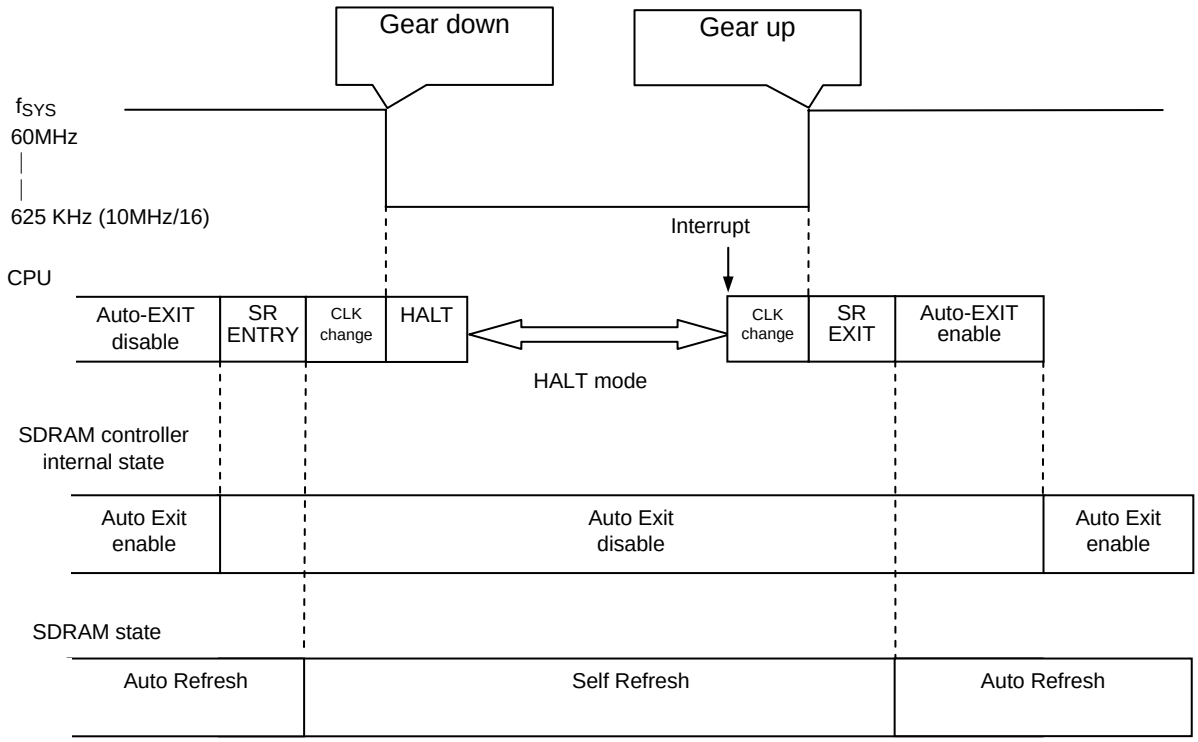


Figure3.10.8 Execution Flow for Executing HALT Instruction after Clock Gear Down

(7) SDRAM initialization sequence

After reset release, the following sequence of commands can be executed to initialize the SDRAM.

1. Precharge All command
2. Eight Auto Refresh commands
3. Mode Register Set command

The above commands are issued by setting SDCMM<SCMM2:0> to “001”. While these commands are issued, the CPU operation (instruction fetch, execution) is halted. Before executing the initialization sequence, appropriate port settings must be made to enable the SDRAM control signals and address signals (A0 to A15).

After the initialization sequence is completed, SDCMM<SCMM2:0> is automatically cleared to “000”.

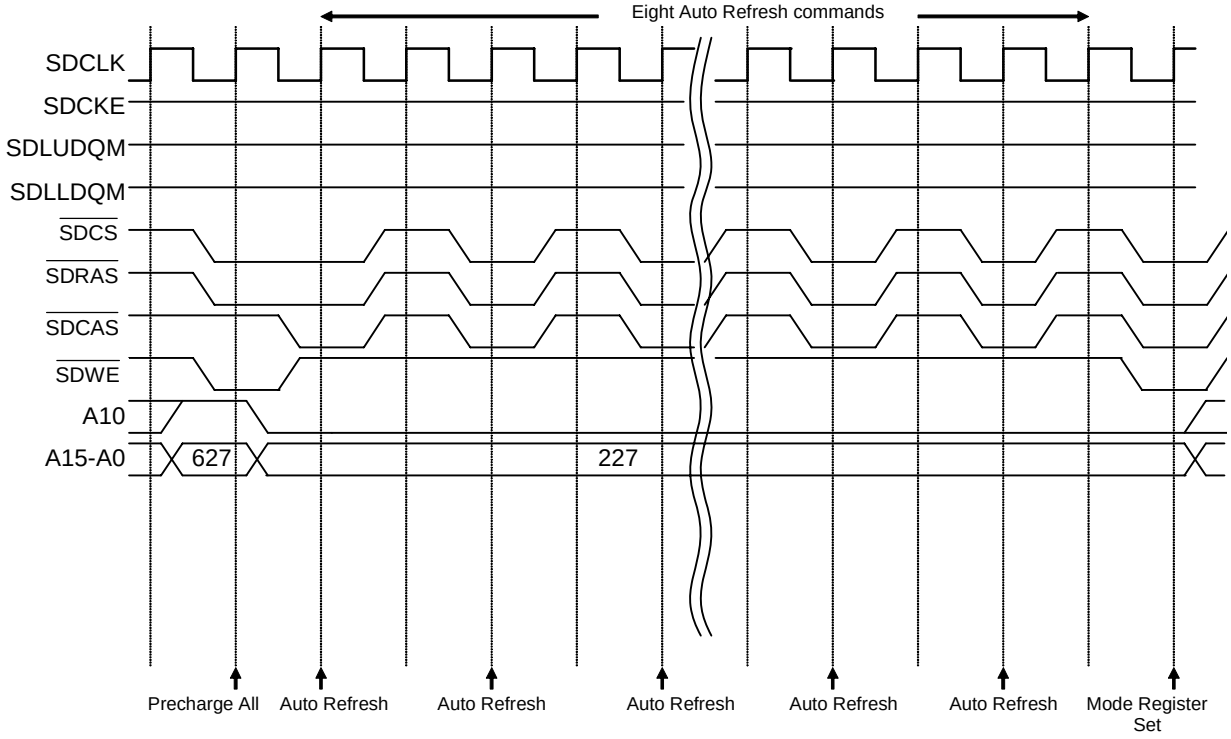


Figure3.10.9 Initialization Sequence Timing

(8) Connection example

Figure3.10.10 shows an example of connections between the TMP92CZ26A and SDRAM.

Table3.10.4 Pin Connections

92CZ26A Pin Name	SDRAM Pin Name				
	Data Bus Width 16 bits				
	16M	64M	128M	256M	512M
A0	A0	A0	A0	A0	A0
A1	A1	A1	A1	A1	A1
A2	A2	A2	A2	A2	A2
A3	A3	A3	A3	A3	A3
A4	A4	A4	A4	A4	A4
A5	A5	A5	A5	A5	A5
A6	A6	A6	A6	A6	A6
A7	A7	A7	A7	A7	A7
A8	A8	A8	A8	A8	A8
A9	A9	A9	A9	A9	A9
A10	A10	A10	A10	A10	A10
A11	BS	A11	A11	A11	A11
A12	–	BS0	BS0	A12	A12
A13	–	BS1	BS1	BS0	BS0
A14	–	–	–	BS1	BS1
A15	–	–	–	–	–
$\overline{\text{SDCS}}$	CS	CS	CS	CS	CS
SDLUDQM	UDQM	UDQM	UDQM	UDQM	UDQM
SDLLDQM	LDQM	LDQM	LDQM	LDQM	LDQM
$\overline{\text{SDRAS}}$	RAS	RAS	RAS	RAS	RAS
$\overline{\text{SDCAS}}$	CAS	CAS	CAS	CAS	CAS
$\overline{\text{SDWE}}$	WE	WE	WE	WE	WE
SDCKE	CKE	CKE	CKE	CKE	CKE
SDCLK	CLK	CLK	CLK	CLK	CLK
SDACR <SMUXW>	00: TypeA	00: TypeA	01: TypeB	01: TypeB	10: TypeC

: Command address pin of SDRAM

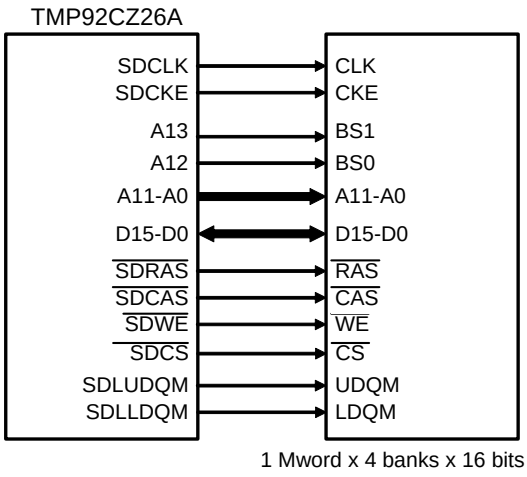


Figure3.10.10 An Example of Connections between TMP92CZ26A and SDRAM

3.10.3 An Example of Calculating HDMA Transfer Time

The following shows an example of calculating the HDMA transfer time when SDRAM is used as the transfer source.

1) Transfer from SDRAM to internal SRAM

Conditions:

System clock (f_{SYS})	: 60 MHz
SDRAM read cycle	: Full page (5-1-1-1), 16-bit data bus 16-bit data bus
SDRAM Auto Refresh interval	: 936 states (15.6 μs)
Internal RAM write cycle	: 1 state, 32-bit data bus
Number of bytes to transfer	: 512 bytes

Calculation example:

Transfer time = (SDRAM read time + SRAM write time) × transfer count
+ (SDRAM burst start + stop time)
+ (Precharge time + Auto Refresh time) × Auto Refresh count

(a) Read/write time
(SDRAM read 1 state × 2 + Internal RAM write 1 state) × 512 bytes/4 bytes
= 384 states × 1/60 MHz
= 6.4 μs

(b) Burst start/stop time
Start (TRCD: 2CLK) 5 states + Stop 2 states
= 7states/60 MHz
= 0.117 μs

(c) Auto Refresh time
Based on the above (a), Auto Refresh occurs once or zero times in 384 states. It is assumed that Auto Refresh occurs once here.

(Precharge (TRP: 2CLK) 2 states + AREF (TRC: 5CLK) 5 states) ×AREF once
= 7 states × 1/60 MHz
= 0.117 μs

Total transfer time = (a) + (b) + (c)
= 6.4 μs + 0.117 μs + 0.117 μs
= 6.634 μs

3.10.4 Considerations for Using the SDRAMC

This section describes the points that must be taken into account when using the SDRAMC. Please carefully read the following to ensure proper use of the SDRAMC.

1) WAIT access

When SDRAM is used, the following restriction applies to memory access to other than the SDRAM.

In the external WAIT pin input setting of the memory controller, the maximum external WAIT period that can be set is limited to “Auto Refresh interval × 8190”.

2) Execution of the Self Refresh Entry, Initialization Sequence, or Precharge All command before the HALT instruction

Execution of the commands issued by the SDRAMC (Self Refresh Entry, Initialization Sequence, Precharge All) requires several states after the SDCMM register is set.

Therefore, to execute the HALT instruction after one of these commands, be sure to insert at least 10 bytes of NOP or other instructions.

3) Auto Refresh interval setting

When SDRAM is used, the system clock frequency must be set to satisfy the minimum operation frequency and minimum Auto Refresh interval of the SDRAM to be used.

In a system in which SDRAM is used and the clock is geared up and down, the Auto Refresh interval must be set carefully.

Before changing the Auto Refresh interval, ensure that SDRCR<SRC> is set to “0” to disable the Auto Refresh function.

4) Changing SFR settings

Before changing the settings of the SDACR<SPRE> and SDCISR registers, ensure that the SDRAMC is disabled (SDACR<SMAC> =“0”).

5) Disabling the SDRAMC

Set the following procedure, when disable the SDRAMC.

```
        LD   (SDCMM),0x02   ;  Issue to All Bank Precharge
LOOP:   LD   A,(SDCMM)      ;  Read SDCMM
        CP   A,0x00         ;  Palling it until the All Bank Precharge command is
                               finished
        JP   NZ,LOOP        ;
        LD   (SDACR),0x00   ;  Stop the SDRAM controller
```

3.11 NAND Flash Controller (NDFC)

3.11.1 Features

The NAND Flash Controller (NDFC) is provided with dedicated pins for connecting with NAND Flash memory.

The NDFC also has an ECC calculation function for error correction and supports two types of ECC calculation methods. The ECC calculation method using Hamming codes can be used for NAND Flash memory of SLC (Single Level Cell) type and is capable of detecting a single-bit error for every 256 bytes. The ECC calculation method using Reed-Solomon codes can be used for NAND Flash memory of MLC (Multi Level Cell) type and is capable of detecting four error addresses for every 518 bytes.

Although the NDFC has two channels (channel 0, channel 1), all pins except for Chip Enable are shared between the two channels. Only the operation of channel 0 is explained here.

The NDFC has the following features:

1) Controls the NAND Flash memory interface through registers.
2) Supports 8-bit and 16-bit NAND Flash memory devices.
3) Supports page sizes of 512 bytes and 2048 bytes.
4) Supports large-capacity block sizes over 256 Kbytes.
5) Includes an ECC generation circuit using Hamming codes (for SLC type).
6) Includes a 4-address (4-byte) error detection circuit using Reed-Solomon coding/ encoding techniques (for MLC type).

Note 1: The $\overline{WP}$ (Write Protect) pin of NAND Flash is not supported. If this function is needed, prepare it on an external circuit.

Note 2: The two channels cannot be accessed simultaneously. It is necessary to switch between the two channels.

3.11.1 Block Diagram

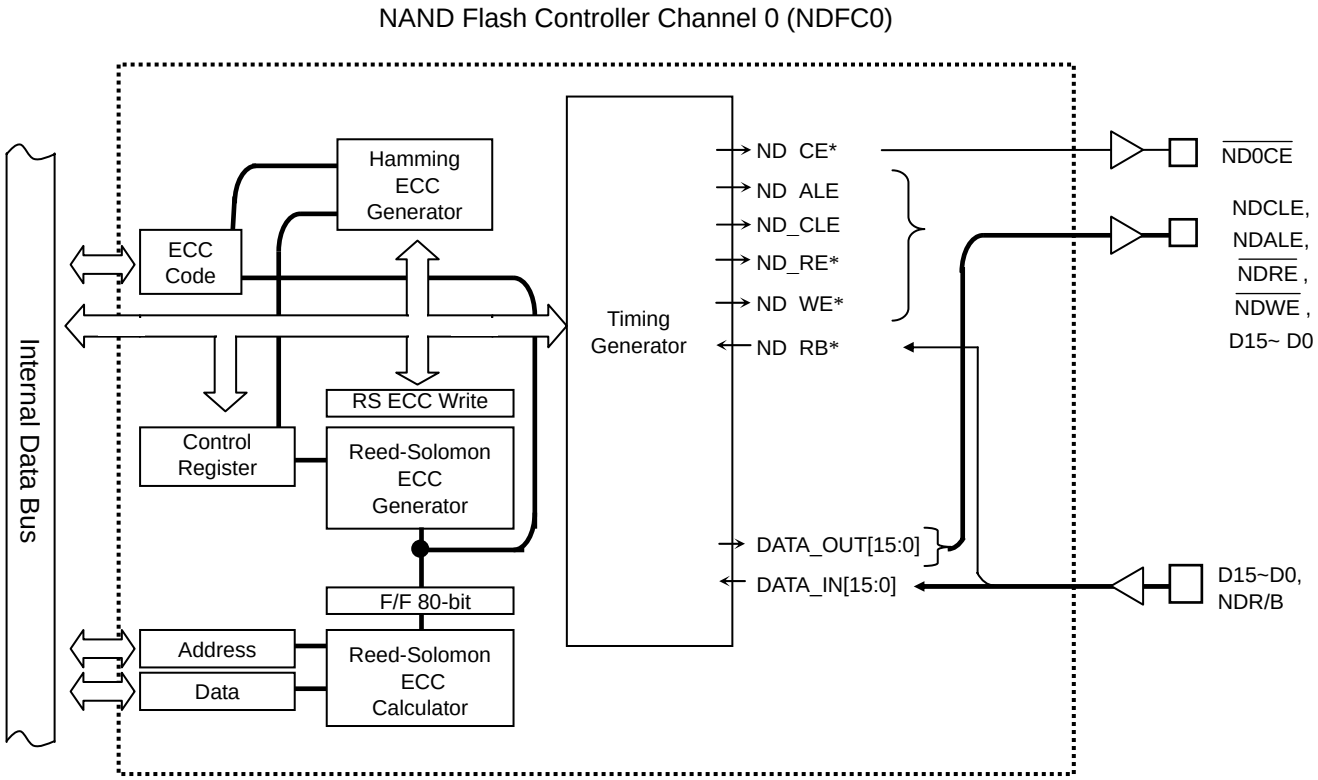


Figure 3.11.1 Block Diagram for NAND Flash Controller

3.11.2 Operation Description

3.11.2.1 Accessing NAND Flash Memory

The NDFC accesses data on NAND Flash memory indirectly through its internal registers. This section explains the operations for accessing the NAND Flash.

Since no dedicated sequencer is provided for generating commands to the NAND Flash, the levels of the NDCLE, NDALE, and $\overline{\text{NDCE}}$ pins must be controlled by software.

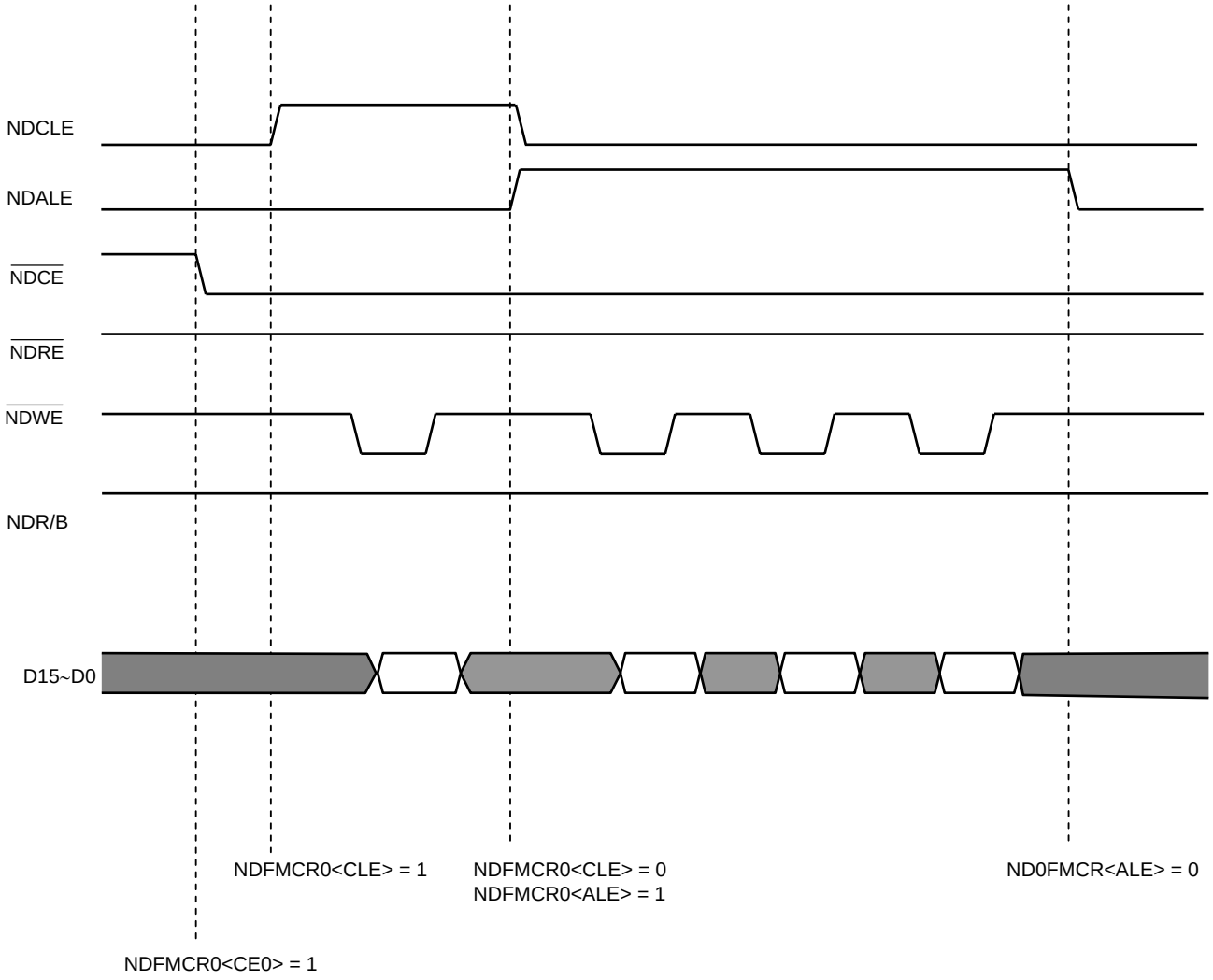


Figure 3.11.2 Basic Timing for Accessing NAND Flash

The $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ signals are explained next. Write and read operations to and from the NAND Flash are performed through the ND0FDTR register. The actual write operation completes not when the ND0FDTR register is written to but when the data is written to the external NAND Flash. Likewise, the actual read operation completes not when the ND0FDTR register is read but when the data is read from the external NAND Flash.

At this time, the Low and High widths of $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ can be adjusted according to the CPU operating speed (f_{SYS}) and the access time of the NAND Flash. (For details, refer to the electrical characteristics.)

The following shows an example of accessing the NAND Flash in 6 clocks by setting NDFMCR0<SPLW1:0>=2 and NDFMCR0<SPHW1:0>=2. (In write cycles, the data drive time also becomes longer.)

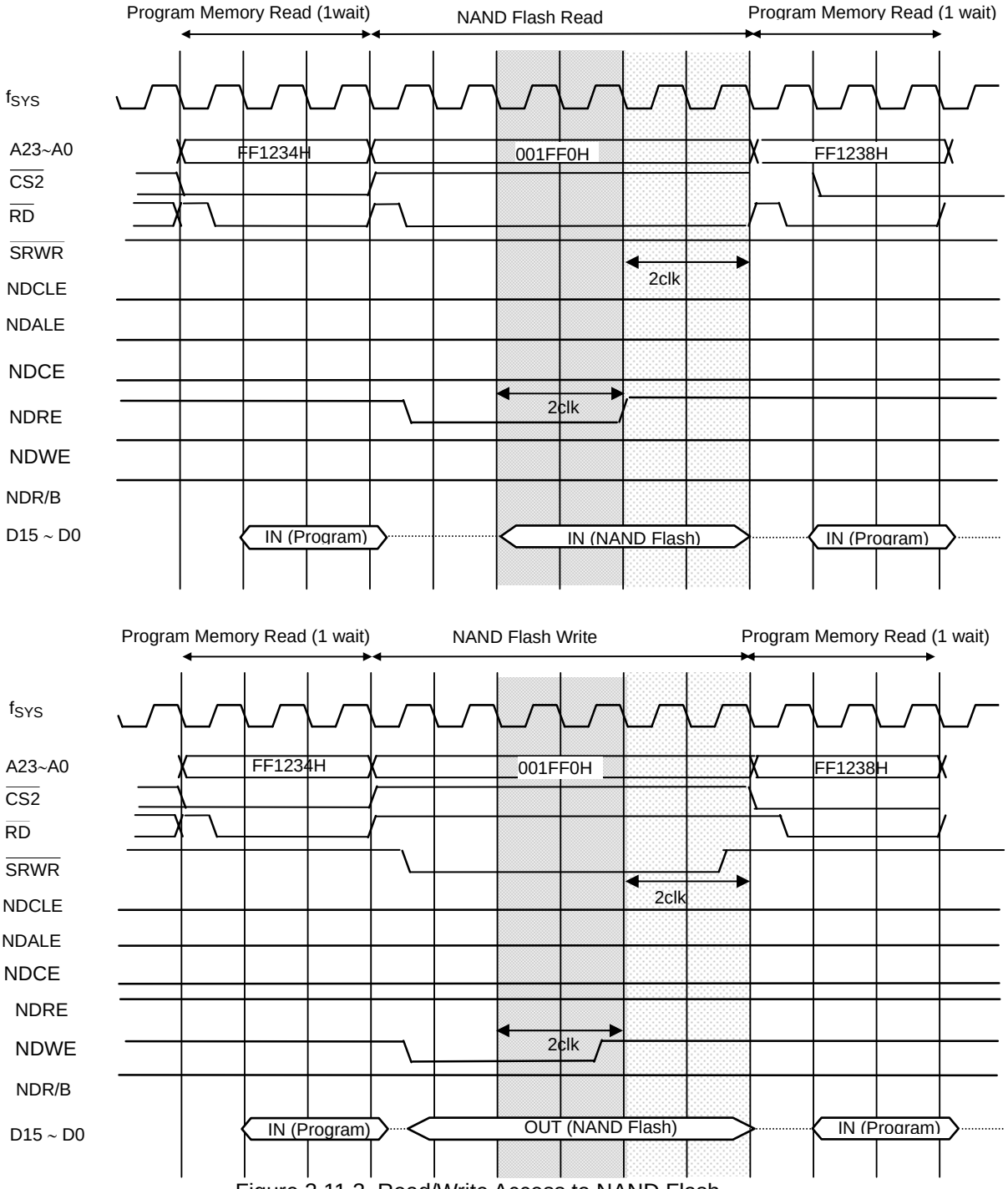


Figure 3.11.3 Read/Write Access to NAND Flash

3.11.3 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 3.11.4 shows a basic flowchart for ECC control.

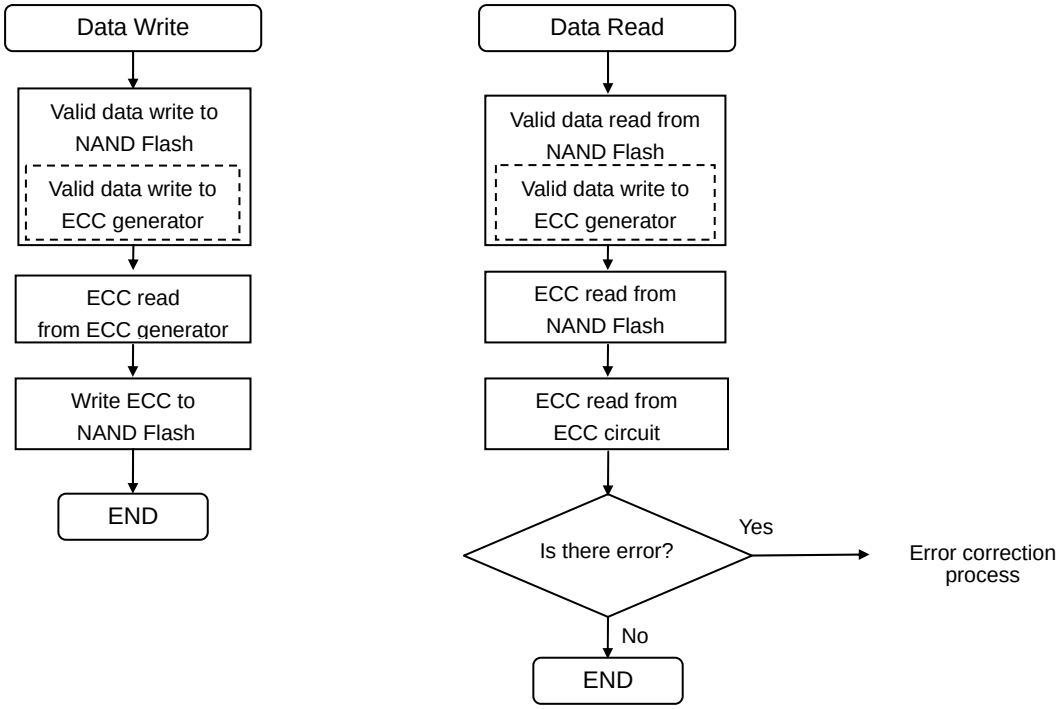


Figure 3.11.4 Basic Flow of ECC Control

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 actual NAND Flash memory, the ECC generator in the NDFC simultaneously generates ECC for the read data.
2. The ECC for the written data and the ECC for the read data are compared to detect and correct error bits.

3.11.3.1 Differences between Hamming Codes and Reed-Solomon Codes

The NDFC includes an ECC generator supporting NAND Flash memory devices of SLC (or 2LC: two states) type and MLC (or 4LC: four states) type.

The ECC calculation using Hamming codes (supporting SLC) 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 SmartMedia™, Hamming codes should be used.

The ECC calculation using Reed-Solomon codes (supporting MLC) generates 80 bits of ECC for every 1 byte to 518 bytes of valid data and is capable of detecting and correcting error bits at four addresses for every 518 bytes. When using Reed-Solomon codes, error bit detection calculation is supported by hardware and only error bit correction needs to be implemented by software.

The differences between Hamming codes and Reed-Solomon codes are summarized in Table 3.11.1.

Table 3.11.1 Differences between Hamming Codes and Reed-Solomon Codes

	Hamming	Reed-Solomon
Maximum number of correctable errors	1 bit	4 addresses (All the 8 bits at one address are correctable.)
Number of ECC bits	22 bits/256 bytes	80 bits/up to 518 bytes
Error bit detection method	Software	Hardware
Error bit correction method	Software	Software
Error bit detection time	Depends on the software to be used.	See the table below.
Others	Supports SmartMedia™.	-

Number of Error Bits	Reed-Solomon Error Bit Detection Time (Unit: Clocks)	Notes
4	813 (max)	These values indicate the total number of clocks for detecting error bit(s) not including the register read/write time by the CPU.
3	648 (max)	
2	358 (max)	
1	219 (max)	
0	1	

3.11.3.2 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 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 are rearranged, respectively, so that the lower 2 bytes 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 each pair of bits 0 to 21 of the XOR result is either 01B or 10B, it is determined that the error data is correctable and error correction is performed accordingly. If the XOR result contains either 00B or 11B, it is determined that the error data is not correctable and the error correction process terminates here.

	An Example of Correctable XOR Result	An Example of Uncorrectable XOR Result
Hexadecimal	26a65a	2ea65a
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) The line and bit positions of the error are detected using the line parity and column parity of the XOR result, respectively. The error bit thus detected is then inverted. This completes the error correction process.

Example: When the XOR result is 26a65aH

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 = 212 *Error at address 212

Column parity: 10 01 10
⇩ ⇩ ⇩
1 0 1 = 5 *Error in bit 5

Based on the above, error correction is performed by inverting the data in bit 5 at address 212.

Reed-Solomon ECC

- The ECC generator generates 80 bits of ECC for up to 518 bytes of valid data. If the NAND Flash 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 000H to 007H, 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. However, if the error exists in the ECC area, only 4symbol (include the error in the ECC area) can correct the error to this LSI. Please be careful.)

2) If the error address indicated by the NDRSCAn register is in the range of 008H to 20DH, the actual error address is obtained by subtracting this address from 20 DH.

   (If the valid data is processed as 512 byte, the actual error address is obtained by subtracting this address from 207H when the error address in the range of 008H to 207H.)

   Example 1:

   NDRSCAn = 005H, NDRSCDn = 04H = 00000100B

   As the error address (005H) is in the range of 000H to 007H, no correction is needed.
   (Although an error exists in bit 2, no correction is needed.)

   Example 2:

   NDRSCAn = 083H, NDRSCDn = 81H = 10000001B

   The actual error address is obtained by subtracting 083H from 20DH. Thus, the error correction process inverts the data in bits 7 and 0 at address 18AH.
   (If the valid data is 512 byte, the actual error address is obtained by subtracting 083H from 207H. Thus, the error correction process inverts the data in bits 7 and 0 at address 184H.)

Note: If the error address (after converted) is in the range of 000H to 007H, 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, Reed-Solomon codes calculate each error bit precisely even if it is the redundant area (ECC).

3.11.4 Description of Registers

NAND Flash Control 0 Register

NDFMCR0 (08C0H)

Read-modify-write instructions cannot be used.

	7	6	5	4	3	2	1	0
bit Symbol	WE	ALE	CLE	CE0	CE1	ECCE	BUSY	ECCRST
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	W
After reset	0	0	0	0	0	0	0	0
Function	WE enable 0: Disable 1: Enable	ALE control 0: "L" out 1: "H" out	CLE control 0: "L" out 1: "H" out	CE0 control 0: "H" out 1: "L" out	CE1 control 0: "H" out 1: "L" out	ECC circuit control 0: Disable 1: Enable	NAND Flash state 1: Busy 0: Ready	ECC reset control 0: – 1: Reset *Always read as "0".

(08C1H)

Read-modify-write instructions cannot be used.

	15	14	13	12	11	10	9	8
bit Symbol	SPLW1	SPLW0	SPHW1	SPHW0	RSECCL	RSEDN	RSESTA	RSECGW
Read/Write	R/W						W	R/W
After reset	0	0	0	0	0	0	0	0
Function	Strobe pulse width ($\overline{\text{Low}}$ width of $\overline{\text{NDRE}}$, $\overline{\text{NDWE}}$) Inserted width = (f_{SYS}) × (set value)		Strobe pulse width ($\overline{\text{High}}$ width of $\overline{\text{NDRE}}$, $\overline{\text{NDWE}}$) Inserted width = (f_{SYS}) × (set value)		Reed-Solomon ECC latch 0: Disable 1: Enable	Reed-Solomon operation 0: Encode (Write) 1: Decode (Read)	Reed-Solomon error calculation start 0: – 1: Start *Always read as "0".	Reed-Solomon ECC generator write control 0: Disable 1: Enable

Figure 3.11.5 NAND Flash Mode Control 0 Register

(a) <ECCRST >

The <ECCRST> bit is used for both Hamming and Reed-Solomon codes.

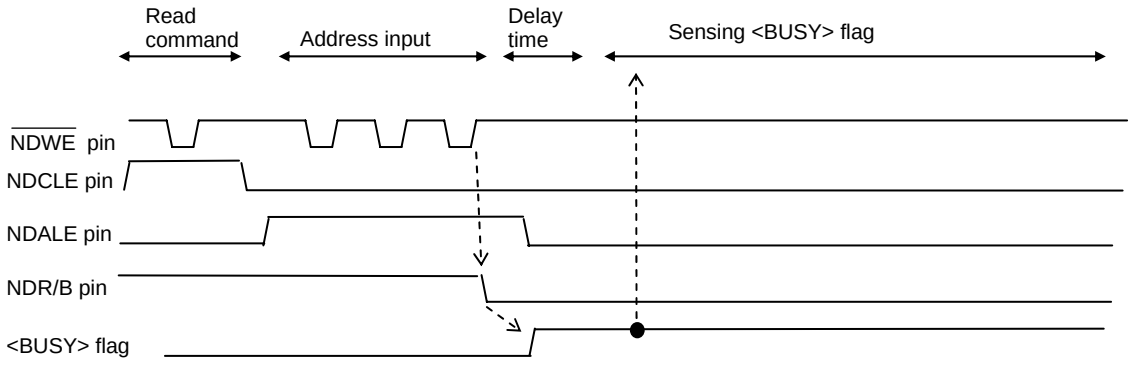
When NDFMCR1<ECCS>="0", setting this bit to "1" clears the Hamming ECC in the ECC generator. When NDFMCR1<ECCS>="1", setting this bit to "1" clears the Reed-Solomon ECC. Note that this bit is ineffective when NDFMCR0<ECCE>="0". Before writing to this bit, ensure that NDFMCR0<ECCE>="1".

(b) <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 (NDR/B 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 states, a change in the NDR/B pin state is reflected on the <BUSY> flag after some delay. It is therefore necessary to inert a delay time by software (e.g. ten NOP instructions) before checking this flag.



(c) <ECCE>

The <ECCE> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to enable or disable the ECC generator. To reset the ECC in the ECC generator (to set <ECCRST> to "1"), the ECC generator must be enabled (<ECCE> = "1").

(d) <CE1:0>, <CLE>, <ALE>

The <CE1:0>, <CLE>, and <ALE> bits are used for both Hamming and Reed-Solomon codes to control the pins of the NAND Flash memory.

(e) <WE>

The <WE> bit is used for both Hamming and Reed-Solomon codes to enable or disable write operations.

(f) <RSECGW>

The <RSECGW> bit is used only for Reed-Solomon codes. When Hamming codes are used, this bit should be set to "0".

Since valid data and ECC are processed differently, the NDFC needs to know whether valid data or ECC is to be read. This control is implemented by software using this bit.

To read valid data from the NAND Flash, set <RSECGW> to "0". To read ECC written in the redundant area in the NAND Flash, set <RSECGW> to "1".

Note 1: Valid data and ECC cannot be read continuously by DMA transfer. After valid data has been read, DMA transfer should be stopped once to change the <RSECGW> bit from "0" to "1" before ECC can be read.

Note 2: Immediately after ECC is read from the NAND Flash, the NAND Flash access operation or error bit calculation cannot be performed for a duration of 20 system clocks (f_{SYS}). It is necessary to insert 20 NOP instructions or the like.

(g) <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. Setting <RSESTA> to "1" starts this calculation.

(h) <RSEDN>

The <RSEDN> bit is used only for Reed-Solomon codes. When using Hamming codes, this bit should be set to "0".

For a write operation, this bit should be set to "0" (encode) to generate ECC. The ECC read from the NDECCRDn register is written to the redundant area in the NAND Flash. For a read operation, this bit should be set to "1" (decode). In this case, valid data is read from the NAND Flash and the ECC written in the redundant area is also read to generate an intermediate code for calculating the error address and error bit position.

(i) <RSECCL>

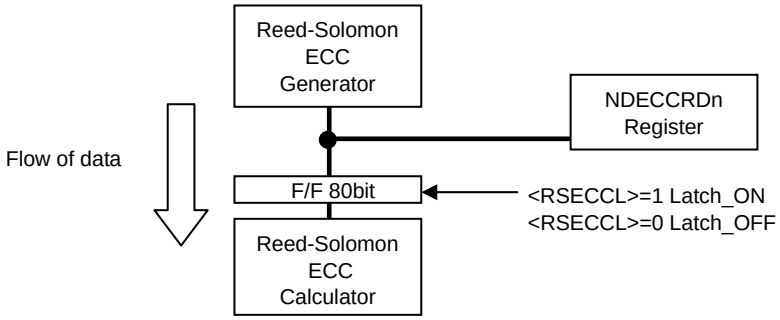
The <RSECCL> bit is used only for Reed-Solomon codes. When using Hamming codes, this bit should be set to “0”.

The Reed-Solomon processing unit is comprised of two elements: an ECC generator and an ECC calculator. The latter is used to calculate the error address and error bit position.

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. At this time, no special care is needed if ECC generation and error calculation are performed serially. If these operations need to be performed parallely, the intermediate code used for error calculation must be latched while the calculation is being performed. The <RSECCL> bit is provided to enable this latch operation.

When <RSECCL> is set to “1”, the intermediate code is latched so that the ECC generator can generate the ECC for another page without problem while the ECC calculator is calculating the error address and error bit position. At this time, the ECC generator can perform both encode (write) and decode (read) operations.

When <RSECCL> is set to “0”, the latch is released and the contents of the ECC calculator are updated as the data in the ECC generator is updated.



(j) <SPHW1:0>

The <SPHW1:0> bits are used for both Hamming and Reed-Solomon codes.

These bits are used to specify the High width of the $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ signals. The High width to be inserted is obtained by multiplying the value set in these bits by f_{SYS}.

(k) <SPLW1:0>

The <SPLW1:0> bits are used for both Hamming and Reed-Solomon codes.

These bits are used to specify the Low width of the $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ signals. The Low width to be inserted is obtained by multiplying the value set in these bits by f_{SYS}.

NAND Flash Control 1 Register

		7	6	5	4	3	2	1	0
NDFMCR1 (08C2H)	bit Symbol	INTERDY	INTRSC				BUSW	ECCS	SYSCKE
	Read/Write	R/W	R/W				R/W	R/W	R/W
	After reset	0	0				0	0	0
	Function	Ready interrupt 0: Disable 1: Enable	Reed-Solomon calculation end interrupt 0: Disable 1: Enable				Data bus width 0: 8-bit 1: 16-bit	ECC calculation 0:Hamming 1: Reed-Solomon	Clock control 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(08C3H)	bit Symbol	STATE3	STATE2	STATE1	STATE0	SEER1	SEER0		
	Read/Write	R							
	After reset	0	0	0	0	Undefined	Undefined		
	Function	Status read (See the table below.)							

Table 3.11.2 Reed-Solomon Calculation Result Status Table

STATE<3:0>	Meaning
0000	Calculation ended 0 (No error)
0001	Calculation ended 1(5 or more symbols in error; not correctable)
0010	Calculation ended 2 (Error found)
0011	
0100~1111	Calculation in progress

Note: The <STATE3:0> value becomes effective after the calculation has started.

SEER<1:0>	Meaning
00	1-address error
01	2-address error
10	3-address error
11	4-address error

Note: The <SEER1:0> value becomes effective after the calculation has ended.

(a) <SYSCKE>

The <SYSCKE> bit is used for both Hamming and Reed-Solomon codes.

When using the NDFC, this bit must be set to "1" to enable the system clock. When not using the NDFC, power consumption can be reduced by setting this bit to "0".

(b) <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 clearing ECC.

(c) <BUSW>

The <BUSW> bit is used for both Hamming and Reed-Solomon codes.

This bit specifies the bus width of the NAND Flash to be accessed ("0" = 8 bits, "1" = 16 bits). No other setting is required in the memory controller.

(d) <INTRSC>

The <INTRSC> bit is used only for Reed-Solomon codes. When using Hamming codes, this bit should be set to "0".

This bit is used to enable or disable the interrupt to be generated when the calculation of error address and error bit position has ended.

The interrupt is enabled when this bit is set to “1” and disabled when “0”.

(e) <INTRDY>

The <INTRDY> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to enable or disable the interrupt to be generated when the status of the NDR/B pin of the NAND Flash changes from “busy” (0) to “ready” (1). The interrupt is enabled when this bit is set to “1” and disabled when “0”.

(f) <STATE3:0>、<SEER1:0>

The <STATE3:0> and <SEER1:0> bits are used only for Reed-Solomon codes. When using Hamming codes, they have no meaning.

These bits are used as flags to indicate the result of error address and error bit calculation. For details, see Table 3.11.2.

NAND Flash Data Register 0

		7	6	5	4	3	2	1	0
NDFDTR0 (1FF0H)	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
	Read/Write	R/W							
	After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (7-0)							
		15	14	13	12	11	10	9	8
(1FF1H)	bit Symbol	D15	D14	D13	D12	D11	D10	D9	D8
	Read/Write	R/W							
	After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (15-8)							

NAND Flash Data Register 1

		7	6	5	4	3	2	1	0
NDFDTR1 (1FF2H)	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
	Read/Write	R/W							
	After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (7-0)							
		15	14	13	12	11	10	9	8
(1FF3H)	bit Symbol	D15	D14	D13	D12	D11	D10	D9	D8
	Read/Write	R/W							
	After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (15-8)							

Note: Although these registers allow both read and write operations, no flip-flop is incorporated. Since write and read operations are performed in different manners, it is not possible to read out the data that has been just written.

Figure 3.11.6 NAND Flash Data Registers (NDFDTR0, NDFDTR1)

Write and read operations to and from the NAND Flash memory are performed by accessing the NDFDTR0 register. When you write to this register, the data is written to the NAND Flash. When you read from this register, the data is read from the NAND Flash. The NDFDTR0 register is used for both channel 0 and channel 1.

A total of 4 bytes are provided as data registers to enable 4-byte DMA transfer. For example, 4 bytes of data can be transferred from 32-bit internal RAM to 8-bit NAND Flash memory by DMA operation by setting the destination address as NDFDTR0. (NDFDTR1 cannot be set as the destination address.) The actual DMA operation is performed by first reading 4 bytes from the internal RAM and then writing 1 byte to the NAND Flash four times from the lowest address.

To access data in the NAND Flash, be sure to access NDFDTR0 (at address 1FF0). For details, see Table 3.11.3.

Table 3.11.3 How to Access the NAND Flash Data Register

Write

Access Data Size	Example of instruction	8-bit NAND Flash	16-bit NAND Flash
1-byte access	ld (0x1FF0),a	Supported	Not supported
2-byte access	ld (0x1FF0),wa	Supported	Supported
4-byte access	ld (0x1FF0),xwa	Supported	Supported

Read

Access Data Size	Example of instruction	8-bit NAND Flash	16-bit NAND Flash
1-byte access	ld a,(0x1FF0)	Supported	Not supported
2-byte access	ld wa,(0x1FF0)	Supported	Supported
4-byte access	ld xwa,(0x1FF0)	Supported	Supported

NAND Flash ECC Register 0

		7	6	5	4	3	2	1	0
NDECCRD0 (08C4H)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08C5H)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 1

		7	6	5	4	3	2	1	0
NDECCRD1 (08C6H)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08C7H)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 2

		7	6	5	4	3	2	1	0
NDECCRD2 (08C8H)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08C9H)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 3

		7	6	5	4	3	2	1	0
NDECCRD3 (08CAH)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08CBH)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 4

		7	6	5	4	3	2	1	0
NDECCRD4 (08CCH)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08CDH)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

Figure 3.11.7 NAND Flash ECC Registers

The NAND Flash ECC register is used to read ECC generated by the ECC generator.

After valid data has been written to or read from the NAND Flash, setting NDFMCR0<ECCE> to "0" causes the corresponding ECC to be set in this register. (The ECC in this register is updated when NDFMCR0<ECCE> changes from "1" to "0".)

When Hamming codes are used, 22 bits of ECC are generated for up to 256 bytes of valid data. In the case of Reed-Solomon codes, 80 bits of ECC are generated for up to 518 bytes of valid data. A total of 80 bits of registers are provided, arranged as five 16-bit registers. These registers must be read in 16-bit units and cannot be accessed in 32-bit units.

After ECC calculation has completed, in the case of Hamming codes, the 16-bit line parity for the first 256 bytes is stored in the NDECCRD0 register, the 6-bit column parity for the first 256 bytes in the NDECCRD1 register (<ECCE7:2>), the 16-bit line parity for the second 256 bytes in the NDECCRD2 register, and the 6-bit column parity for the second 256 bytes in the NDECCRD3 register (<ECCD7:2>). In this case, the NDECCRD4 register is not used.

In the case of Reed-Solomon codes, 80 bits of ECC are stored in the NDECCRD0, NDECCRD1, NDECCRD2, NDECCRD3 and NDECCRD4 registers.

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] Line parity (for the first 256 bytes)	[15:0] Reed-Solomon ECC code 79:64
NDECCRD1	[7:2] Column parity (for the first 256 bytes)	[15:0] Reed-Solomon ECC code 63:48
NDECCRD2	[15:0] Line parity (for the second 256 bytes)	[15:0] Reed-Solomon ECC code 47:32
NDECCRD3	[7:2] Column parity (for the second 256 bytes)	[15:0] Reed-Solomon ECC code 31:16
NDECCRD4	Not in use	[15:0] Reed-Solomon ECC code 15:0

The table below shows an example of how ECC is written to the redundant area in the NAND Flash memory when using Reed-Solomon codes.

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.

Register Name	Reed-Solomon	NAND Flash Address
NDECCRD0	[15:0] Reed-Solomon ECC code 79:64	Upper 8 bits [79:72]→ address 518 Lower 8 bits [71:64] → address 519
NDECCRD1	[15:0] Reed-Solomon ECC code 63:48	Upper 8 bits [63:56] → address 520 Upper 8 bits [55:48] → address 521
NDECCRD2	[15:0] Reed-Solomon ECC code 47:32	Upper 8 bits [47:40] → address 522 Lower 8 bits [39:32] → address 523
NDECCRD3	[15:0] Reed-Solomon ECC code 31:16	Upper 8 bits [31:24] → address 524 Lower 8 bits [23:16] → address 525
NDECCRD4	[15:0] Reed-Solomon ECC code 15:0	Upper 8 bits [15:8] → address 526 Lower 8 bits [7:0] → address 527

NAND Flash Reed-Solomon Calculation Result Address Register

		7	6	5	4	3	2	1	0
NDRSCA0 (08D0H)	bit Symbol	RS0A7	RS0A6	RS0A5	RS0A4	RS0A3	RS0A2	RS0A1	RS0A0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08D1H)	bit Symbol							RS0A9	RS0A8
	Read/Write							R	
	After reset							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
		7	6	5	4	3	2	1	0
NDRSCA1 (08D4H)	bit Symbol	RS1A7	RS1A6	RS1A5	RS1A4	RS1A3	RS1A2	RS1A1	RS1A0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08D5H)	bit Symbol							RS1A9	RS1A8
	Read/Write							R	
	After reset							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
		7	6	5	4	3	2	1	0
NDRSCA2 (08D8H)	bit Symbol	RS2A7	RS2A6	RS2A5	RS2A4	RS2A3	RS2A2	RS2A1	RS2A0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08D9H)	bit Symbol							RS2A9	RS2A8
	Read/Write							R	
	After reset							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
		7	6	5	4	3	2	1	0
NDRSCA3 (08DCH)	bit Symbol	RS3A7	RS3A6	RS3A5	RS3A4	RS3A3	RS3A2	RS3A1	RS3A0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08DDH)	bit Symbol							RS3A9	RS3A8
	Read/Write							R	
	After reset							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	

Figure 3.11.8 NAND Flash Reed-Solomon Calculation Result Address Register

If error is found at only one address, the error address is stored in the NDRSCA0 register. If error is found at two addresses, the NDRSCA0 and NDRSCA1 registers are used to store the error addresses. In this manner, up to four error addresses can be stored in the NDRSCA0 to NDRSCA3 registers.

The number of error addresses can be checked by NDFMCR1<SEER1:0>.

NAND Flash Reed-Solomon Calculation Result Data Register

		7	6	5	4	3	2	1	0
NDRSCD0 (08D2H)	bit Symbol	RS0D7	RS0D6	RS0D5	RS0D4	RS0D3	RS0D2	RS0D1	RS0D0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
		7	6	5	4	3	2	1	0
NDRSCD1 (08D6H)	bit Symbol	RS1D7	RS1D6	RS1D5	RS1D4	RS1D3	RS1D2	RS1D1	RS1D0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
		7	6	5	4	3	2	1	0
NDRSCD2 (08DAH)	bit Symbol	RS2D7	RS2D6	RS2D5	RS2D4	RS2D3	RS2D2	RS2D1	RS2D0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
		7	6	5	4	3	2	1	0
NDRSCD3 (08DEH)	bit Symbol	RS3D7	RS3D6	RS3D5	RS3D4	RS3D3	RS3D2	RS3D1	RS3D0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							

Figure 3.11.9 NAND Flash Reed-Solomon Calculation Result Data Register

If error is found at only one address, the error data is stored in the NDRSCD0 register. If error is found at two addresses, the NDRSCD0 and NDRSCD1 registers are used to store the error data. In this manner, the error data at up to four addresses can be stored in the NDRSCD0 to NDRSCD3 registers.

The number of error addresses can be checked by NDFMCR1<SEER1:0>.

3.11.5 An Example of Accessing NAND Flash of SLC Type

1. Initialization

```
;
; ***** Initialize NDFC *****
;       Conditions: 8-bit bus, CE0, SLC, 512 (528) bytes/page, Hamming codes
;
        ld      (ndfmcr1),0001h  ; 8-bit bus, Hamming ECC, SYSCK-ON
        ld      (ndfmcr0),2000h  ; SPLW1:0=0, SPHW1:0=2
```

2. Write

Writing valid data

```
; ***** Write valid data*****
;
        ldw     (ndfmcr0),2010h  ; CE0 enable
        ldw     (ndfmcr0),20B0h  ; WE enable, CLE enable
        ld      (ndfdtr0),80h    ; Serial input command
        ldw     (ndfmcr0),20D0h  ; ALE enable
        ld      (ndfdtr0),xxh    ; Address write (3 or 4 times)
        ldw     (ndfmcr0),2095h  ; Reset ECC, ECCE enable, CE0 enable
        ld      (ndfdtr0),xxh    ; Data write (512 times)
```

Generating ECC → Reading ECC

```
; ***** Read ECC *****
;
        ldw     (ndfmcr0),2010h  ; ECC circuit disable

        ldw     xxxx,(ndeccrd0)  ; Read ECC from internal circuit
;               1'st Read:       D15-0 > LPR15:0          For first 256 bytes
        ldw     xxxx,(ndeccrd1)  ; Read ECC from internal circuit
;               2'nd Read:       D15-0 > FFh+CPR5:0+11b   For first 256 bytes
        ldw     xxxx,(ndeccrd0)  ; Read ECC from internal circuit
;               3'rd Read:       D15-0 > LPR15:0          For second 256 bytes
        ldw     xxxx,(ndeccrd1)  ; Read ECC from internal circuit
;               4'th Read:       D15-0 > FFh+CPR5:0+11b   For second 256 bytes
```

Writing ECC to NAND Flash

```
; ***** Write dummy data & ECC*****
;
        ldw     (ndfmcr0),2090h  ; ECC circuit disable, data write mode
        ld      (ndfdtr0),xxh    ; Redundancy area data write (16 times)
;               Write to D520:   LPR7:0     > D7-0   For second 256 bytes
;               Write to D521:   LPR15:8    > D7-0   For second 256 bytes
;               Write to D522:   CPR5:0+11b > D7-0   For second 256 bytes
;               Write to D525:   LPR7:0     > D7-0   For first 256 bytes
;               Write to D526:   LPR15:8    > D7-0   For first 256 bytes
;               Write to D527:   CPR5:0+11b > D7-0   For first 256 bytes
```

```
Executing page program
; ***** Set auto page program*****
;
        ldw     (ndfmcr0),20B0h  ; WE enable, CLE enable
        ld      (ndfdtr0),10h    ; Auto page program command
        ldw     (ndfmcr0),2010h  ; WE disable, CLE disable
;
;       Wait setup time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
;

Reading status
; ***** Read Status*****
;
        ldw     (ndfmcr0),20B0h  ; WE enable, CLE enable
        ld      (ndfdtr0),70h    ; Status read command
        ldw     (ndfmcr0),2010h  ; WE disable, CLE disable
        ld      xx,(ndfdtr0)     ; Status read
```

3. Read

Reading valid data

```
; ***** Read valid data*****
;
        ldw     (ndfmcr0),2010h   ; CE0 enable
        ldw     (ndfmcr0),20B0h   ; WE enable, CLE enable
        ld      (ndfdtr0),00h     ; Read command
        ldw     (ndfmcr0),20D0h   ; ALE enable
        ld      (ndfdtr0),xxh     ; Address write (3 or 4 times)
;
;       Wait setup time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
;
        ldw     (ndfmcr0),2015h   ; Reset ECC, ECCE enable, CE0 enable
        ld      xx,(ndfdtr0)      ; Data read (512 times)
        ldw     (ndfmcr0),2010h   ; ECC circuit disable
        ld      xx,(ndfdtr0)      ; Redundancy data read (8 times)
        ld      xx,(ndfdtr0)      ; ECC data read (3 times)
        ld      xx,(ndfdtr0)      ; Redundancy data read (2 times)
        ld      xx,(ndfdtr0)      ; ECC data read (3 times)
```

Generating ECC → Reading ECC

```
; ***** Read ECC *****
;
        ldw     (ndfmcr0),2010h   ; ECC circuit disable
        ldw     xxxx,(ndeccrd0)   ; Read ECC from internal circuit
;               1'st Read:        D15-0 > LPR15:0          For first 256 bytes
        ldw     xxxx,(ndeccrd1)   ; Read ECC from internal circuit
;               2'nd Read:        D15-0 > FFh+CPR5:0+11b   For first 256 bytes
        ldw     xxxx,(ndeccrd0)   ; Read ECC from internal circuit
;               3'rd Read:        D15-0 > LPR15:0          For second 256 bytes
        ldw     xxxx,(ndeccrd1)   ; Read ECC from internal circuit
;               4'th Read:        D15-0 > FFh+CPR5:0+11b   For second 256 bytes
```

Software processing

The ECC data generated for the read operation and the ECC in the redundant area in the NAND Flash are compared. If any error is found, the error processing routine is performed to correct the error data. For details, see 3.11.3.2 “Error Correction Methods”.

4. ID Read

The ID read routine is as follows:

```
ldw     (ndfmcr0),20B0h  ; WE Enable, CLE enable
ld      (ndfdtr0),90h    ; Write ID read command
ldw     (ndfmcr0),20D0h  ; ALE enable, CLE disable
ld      (ndfdtr0),00h    ; Write 00
ldw     (ndfmcr0),2010h  ; WE disable, CLE disable
ld      xx,(ndfdtr0)     ; Read 1'st ID maker code
ld      xx,(ndfdtr0)     ; Read 2'nd ID device code
```

3.11.6 An Example of Accessing NAND Flash of MLC Type (When the valid data is processed as 518byte)

1. Initialization

```
;
; ***** Initialize NDFC *****
;       Conditions: 16-bit bus, CE1, MLC, 2048 (2112) bytes/page, Reed-Solomon codes
;
        ld      (ndfmcr1),0007h  ; 16-bit bus, Reed-Solomon ECC, SYSCK-ON
        ld      (ndfmcr0),5000h  ; SPLW1:0=1, SPHW1:0=1
```

2. Write

Writing valid data

```
; ***** Write valid data*****
;
        ldw     (ndfmcr0),5008h  ; CE1 enable
        ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
        ldw     (ndfdtr0),0080h  ; serial input command
        ldw     (ndfmcr0),50C8h  ; ALE enable
        ldw     (ndfdtr0),00xxh  ; Address write ( 4 or 5 times)
        ldw     (ndfmcr0),508Dh  ; Reset ECC code, ECCE enable
        ldw     (ndfdtr0),xxxxh  ; Data write (259-times/:518byte)
                                                (256-times/512byte)
```

Generating ECC → Reading ECC

```
; ***** Read ECC *****
;
        ldw     (ndfmcr0),5008h  ; ECC circuit disable
        ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
        ldw     (ndfdtr0),0080h  ; serial input command
        ldw     (ndfmcr0),50C8h  ; ALE enable
        ldw     (ndfdtr0),00xxh  ; Address write ( 4 or 5 times)

        ldw     xxxx,(ndeccrd0)  ; Read ECC from internal circuit
;               Read:   D79-64
        ldw     xxxx,(ndeccrd1)  ; Read ECC from internal circuit
;               Read:   D63-48
        ldw     xxxx,(ndeccrd2)  ; Read ECC from internal circuit
;               Read:   D47-32
        ldw     xxxx,(ndeccrd3)  ; Read ECC from internal circuit
;               Read:   D31-16
        ldw     xxxx,(ndeccrd4)  ; Read ECC from internal circuit
;               Read:   D15-0
```

Writing ECC to NAND Flash

```
; ***** Write dummy data & ECC *****
;
        ldw     (ndfmcr0),5088h  ; ECC circuit disable, data write mode
        ldw     (ndfdtr0),xxxxh  ; Redundancy area data write
;               Write to 207-206hex address:        > D79-64
        ldw     (ndfdtr1),xxxxh  ; Redundancy area data write
;               Write to 209-208hex address:        > D63-48
        ldw     (ndfdtr0),xxxxh  ; Redundancy area data write
;               Write to 20B-20Ahex address:        > D47-32
        ldw     (ndfdtr1),xxxxh  ; Redundancy area data write
;               Write to 20D-20Chex address:        > D31-16
        ldw     (ndfdtr0),xxxxh  ; Redundancy area data write
;               Write to 20F-20Ehex address:        > D15-0
;
;       The write operation is repeated four times to write 2112 bytes.
```

Executing page program

```
; ***** Set auto page program*****
;
        ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
        ldw     (ndfdtr0),0010h  ; Auto page program command
        ldw     (ndfmcr0),5008h  ; WE disable, CLE disable
;
;       Wait set up time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
```

※ In case of LB type NANDF, programming page size is normally each 2112 bytes and ECC calculation is processed each 518 (512) bytes. Please take care of programming flow. In details, refer the NANDF memory specifications.

Reading status

```
; ***** Read status*****
;
        ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
        ldw     (ndfdtr0),0070h  ; Status read command
        ldw     (ndfmcr0),5008h  ; WE disable, CLE disable
        ldw     xxxx,(ndfdtr0)   ; Status read
```

3. Read (including ECC data read)

Reading valid data

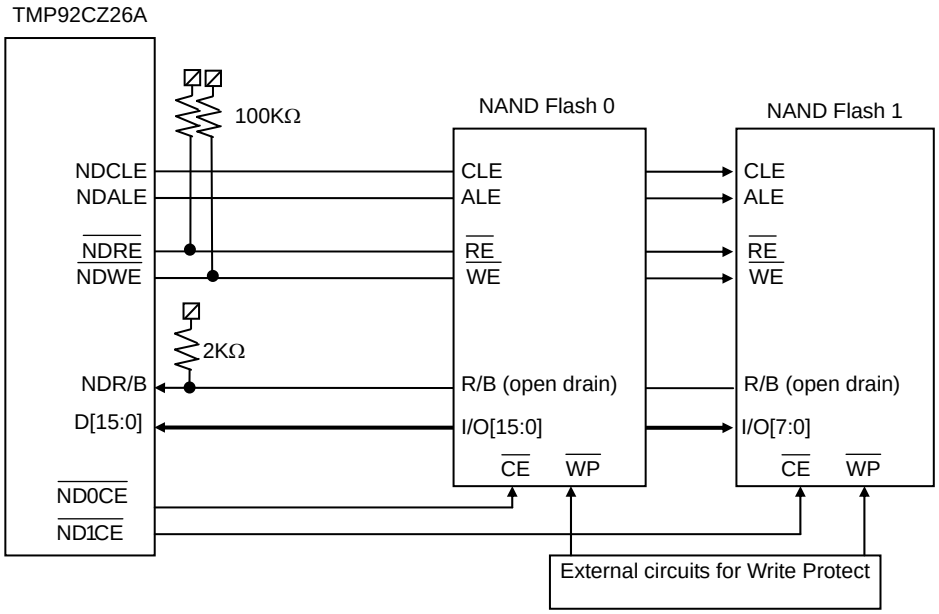
```
; ***** Read valid data*****
;
        ldw     (ndfmcr0),5008h   ; CE1 enable
        ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
        ldw     (ndfdtr0),0000h   ; Read command 1
        ldw     (ndfmcr0),50C8h   ; ALE enable
        ldw     (ndfdtr0),00xxh   ; Address write (4 or 5 times)
        ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
        ldw     (ndfdtr0),0030h   ; Read command 2
;
;       Wait set up time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
;
        ldw     (ndfmcr0),540Dh   ; ECC reset, ECC circuit enable, decode mode
        ldw     xxxx,(ndfdtr0)    ; Data read (259 times: 518 bytes)
                                            (256-times:512byte)
        ldw     (ndfmcr0),550Ch   ; RSECGW enable
        ldw     xxxx,(ndfdtr0)    ; Read ECC (5 times: 80 bits)
;
;       Wait set up time (20 system clocks)
;
(1) Error bit calculation
        ldw     (ndfmcr1),0047h   ; Error bit calculation interrupt enable
        ldw     (ndfmcr0),560Ch   ; Error bit calculation circuit start
;
;       Wait set up time
;       Interrupt routine (End of calculation for Reed-Solomon Error bit)
;
INT:    ldw     xxxx,(ndfmcr1)    ; Check error status "STATE3:0, SEER1:0"
;
;       If error is found, the error processing routine is performed to
;        correct the error data. For details see 3.11.3.2 "Error Correction
;       Methods".
;
;       The read operation is repeated four times to read 2112 bytes.
;
```

4. ID Read

The ID read routine is as follows:

```
ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
ldw     (ndfdtr0),0090h  ; Write ID read command
ldw     (ndfmcr0),50C8h  ; ALE enable, CLE disable
ldw     (ndfdtr0),0000h  ; Write 00
ldw     (ndfmcr0),5008h  ; WE disable, CLE disable
ldw     xxxx,(ndfdtr0)   ; Read 1'st ID maker code
ldw     xxxx,(ndfdtr1)   ; Read 2'ndID device code
```

3.11.7 An Example of Connections with NAND Flash



Note 1: A reset sets the $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ pins as input ports, so pull-up resistors are needed.

Note 2: The pull-up resistor value for the NDR/B pin must be set appropriately according to the NAND Flash memory to be used and the capacity of the board (typical: 2 KΩ).

Note 3: The $\overline{\text{WP}}$ (Write Protect) pin of NAND Flash is not supported. When this function is needed, prepare it on an external circuit.

Figure 3.11.10 An Example of Connections with NAND Flash

3.12 8 Bit Timer (TMRA)

The TMP92CZ26A features 8 channel (TMRA0 to TMRA7) built-in 8-bit timers.

These timers are paired into 4 modules: TMRA01, TMRA23, TMRA45 and TMRA67. Each module consists of 2 channels and can operate in any of the following 4 operating modes.

- 8-bit interval timer mode
- 16-bit interval timer mode
- 8-bit programmable square wave pulse generation output mode (PPG: Variable duty cycle with variable period)
- 8-bit pulse width modulation output mode (PWM – Variable duty cycle with constant period)

Figure 3.12.1 to Figure 3.12.4 show block diagrams for TMRA01 to TMRA67.

Each channel consists of an 8-bit up counter, an 8-bit comparator and an 8-bit timer register. In addition, a timer flip-flop and a prescaler are provided for each pair of channels.

The operation mode and timer flip-flops are controlled by 5bytes registers SFRs (Special-function registers).

Each of the 4 modules (TMRA01 to TMRA67) can be operated independently. All modules operate in the same manner; hence only the operation of TMRA01 is explained here.

The contents of this chapter are as follows.

Table 3.12.1 Registers and Pins for Each Module

Specification \ Module		TMRA01	TMRA23	TMRA45	TMRA67
External pin	Input pin for external clock	TA0IN (Shared with PC1)	TA2IN (Shared with PC3)	Low-frequency clock fs	Low-frequency clock fs
	Output pin for timer flip-flop	TA1OUT (Shared with PM1)	TA3OUT (Shared with PP1)	TA5OUT (Shared with PP2)	TA7OUT (Shared with PP3)
SFR (Address)	Timer run register	TA01RUN (1100H)	TA23RUN (1108H)	TA45RUN (1110H)	TA67RUN (1118H)
	Timer register	TA0REG (1102H) TA1REG (1103H)	TA2REG (110AH) TA3REG (110BH)	TA4REG (1112H) TA5REG (1113H)	TA6REG (111AH) TA7REG (111BH)
	Timer mode register	TA01MOD (1104H)	TA23MOD (110CH)	TA45MOD (1114H)	TA67MOD (111CH)
	Timer flip-flop control register	TA1FFCR (1105H)	TA3FFCR (110DH)	TA5FFCR (1115H)	TA7FFCR (111DH)

3.12.1 Block Diagram

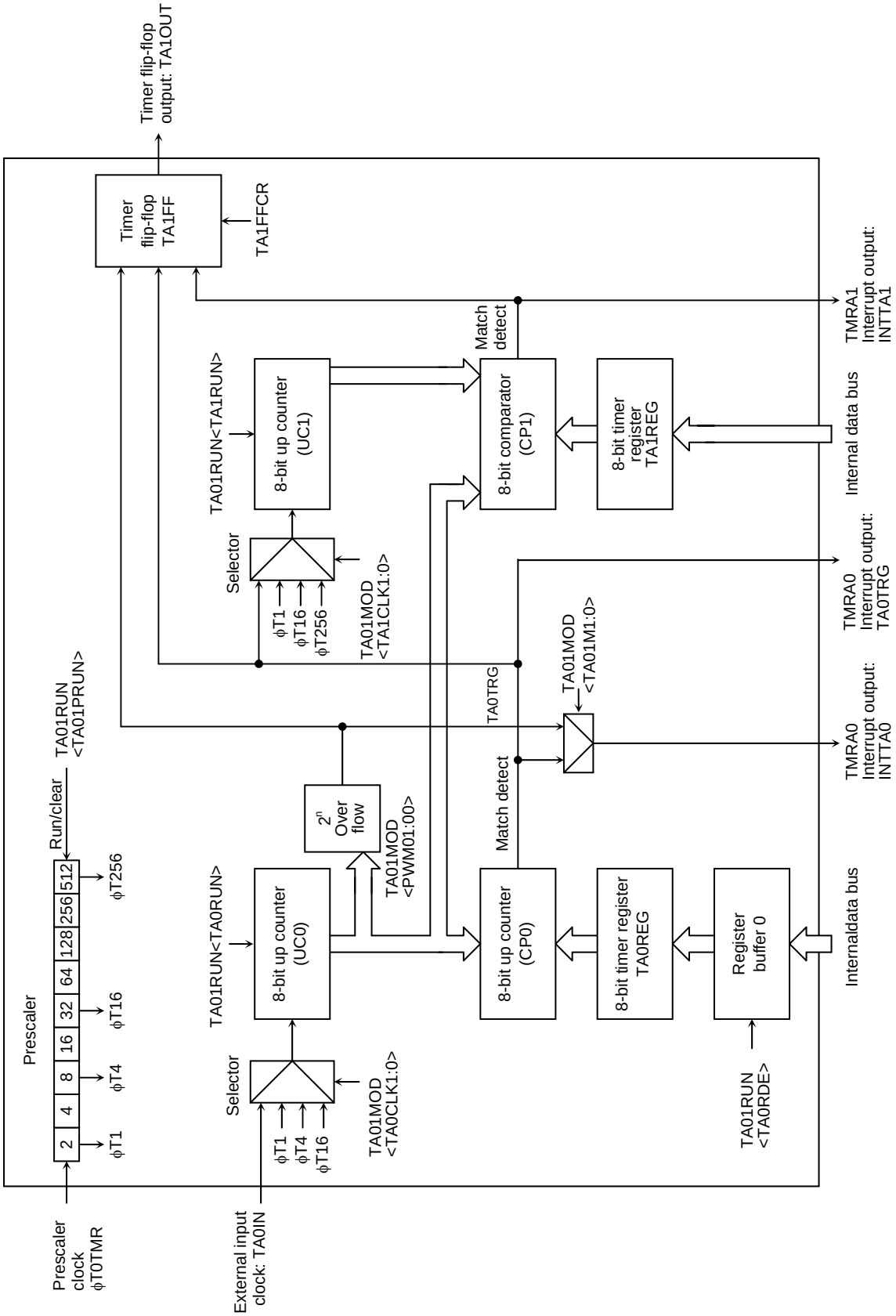


Figure 3.12.1 TMRA01 Block Diagram

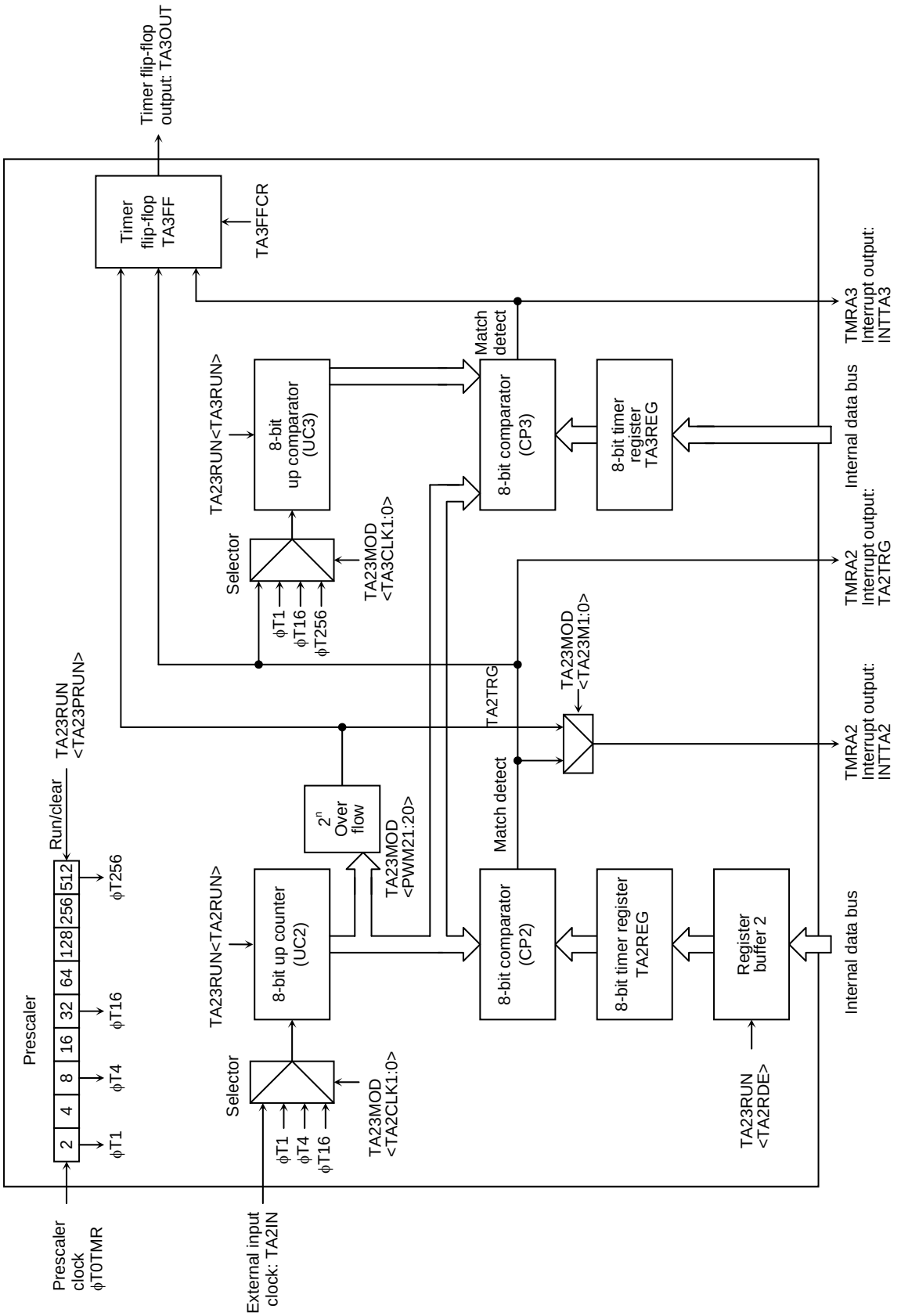


Figure 3.12.2 TMRA23 Block Diagram

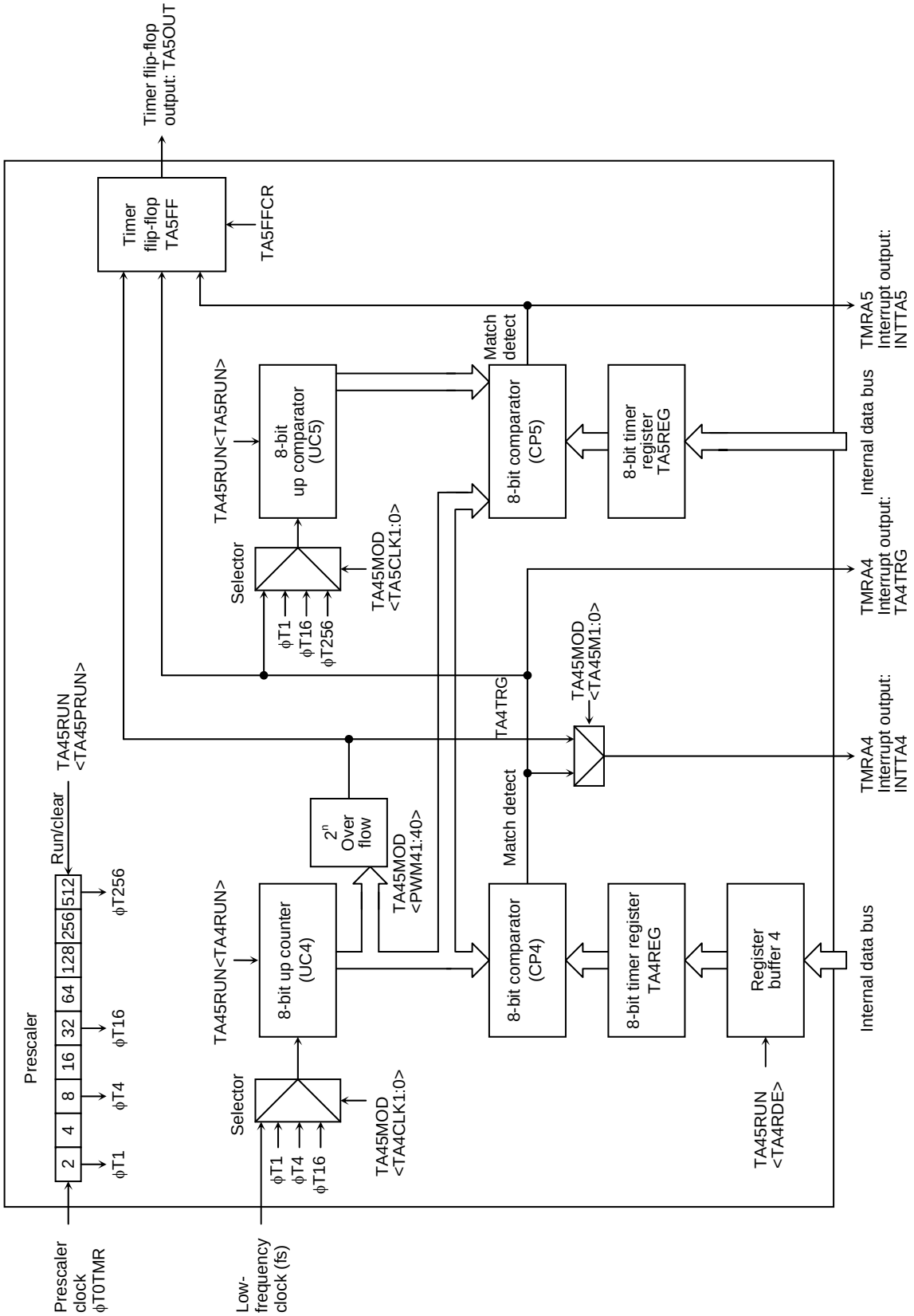


Figure 3.12.3 TMRA45 Block Diagram

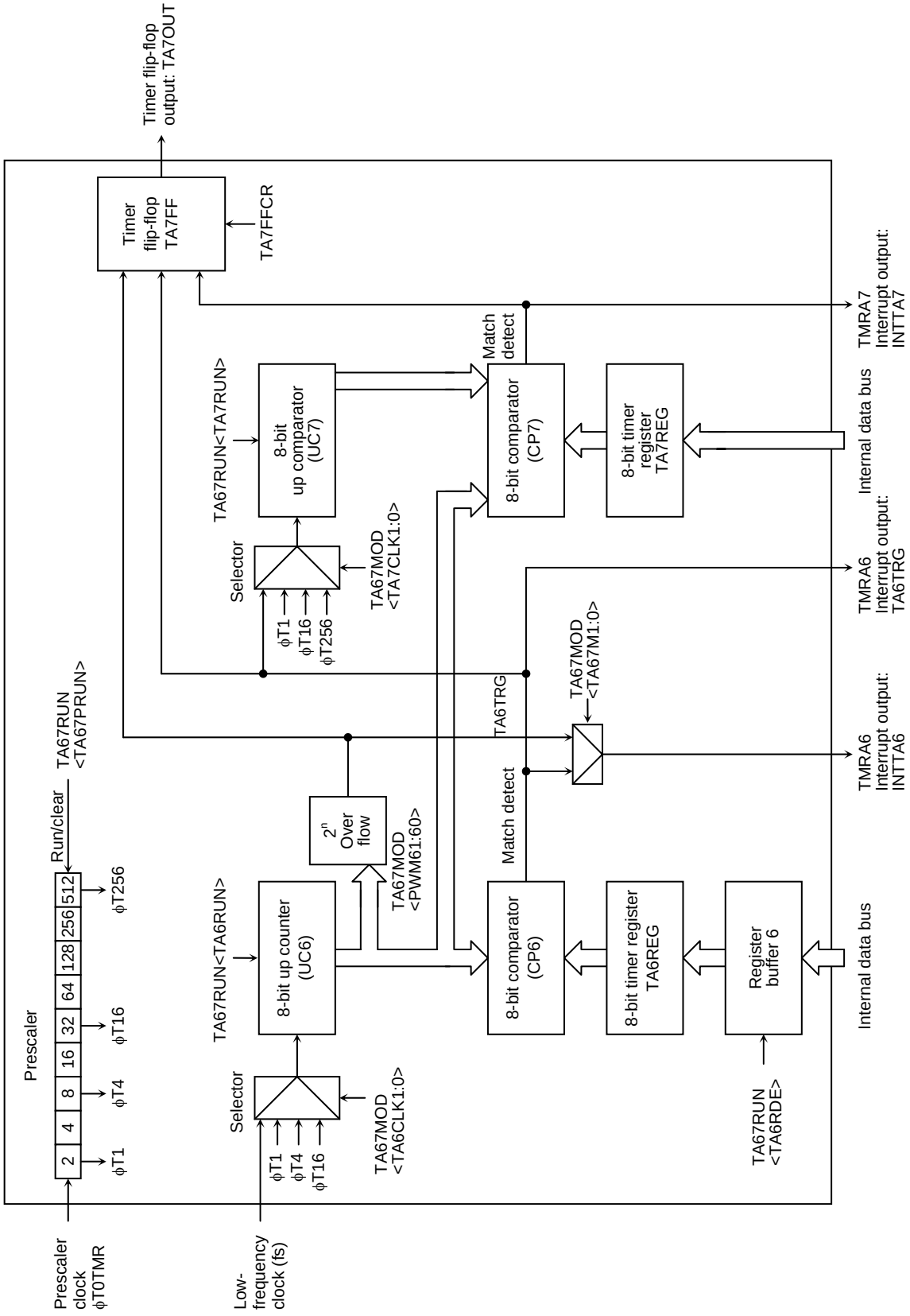


Figure 3.12.4 TMRA67 Block Diagram

3.12.2 Operation of Each Circuit

(1) Prescaler

A 9-bit prescaler generates the input clock to TMRA01.The clock ϕT0 is selected using the prescaler clock selection register SYSCR0<PRCK>.

The prescaler operation can be controlled using TA01RUN<TA0PRUN> in the timer control register. Setting <TA0PRUN> to 1 starts the count; setting <TA0PRUN> to 0 clears the prescaler to 0 and stops operation. Table shows the various prescaler output clock resolutions.

(Although the prescaler and the timer counter can be started separately, the timer counter's operation depends on the prescaler's input timing.)

Table 3.12.2 Prescaler Output Clock Resolution

	Clock gear selection SYSCR1 <GEAR2:0>	Prescaler of clock gear SYSCR0 <PRCK>	_	Timer counter input clock Prescaler of TMRA TAxxMOD<TAxCLK1:0>			
				ϕT1(1/2)	ϕT4(1/8)	ϕT16(1/32)	ϕT256(1/512)
fc	000(1/1)	0(1/2)	1/2	fc/8	fc/32	fc/128	fc/2048
	001(1/2)			fc/16	fc/64	fc/256	fc/4096
	010(1/4)			fc/32	fc/128	fc/512	fc/8192
	011(1/8)			fc/64	fc/256	fc/1024	fc/16384
	100(1/16)			fc/128	fc/512	fc/2048	fc/32768
	000(1/1)	1(1/8)		fc/32	fc/128	fc/512	fc/8192
	001(1/2)			fc/64	fc/256	fc/1024	fc/16384
	010(1/4)			fc/128	fc/512	fc/2048	fc/32768
	011(1/8)			fc/256	fc/1024	fc/4096	fc/65536
	100(1/16)			fc/512	fc/2048	fc/8192	fc/131072

(2) Up counters (UC0 and UC1)

These are 8-bit binary counters which count up the input clock pulses for the clock specified by TA01MOD.

The input clock for UC0 is selectable and can be either the external clock input via the TA0IN pin or one of the three internal clocks ϕT1, ϕT4 or ϕT16. The clock setting is specified by the value set in TA01MOD<TA01CLK1:0>.

The input clock for UC1 depends on the operation mode. In 16-bit timer mode, the overflow output from UC0 is used as the input clock. In any mode other than 16-bit timer mode, the input clock is selectable and can either be one of the internal clocks ϕT1, ϕT16 or ϕT256, or the comparator output (The match detection signal) from TMRA0.

For each interval timer the timer operation control register bits TA01RUN <TA0RUN> and TA01RUN<TA1RUN> can be used to stop and clear the up counters and to control their count. A reset clears both up counters, stopping the timers.

Note: TMR45 and TMR67 can select low-frequency clock(fs) instead of external clock input.

(3) Timer registers (TA0REG and TA1REG)

These are 8-bit registers, which can be used to set a time interval. When the value set in the timer register TA0REG or TA1REG matches the value in the corresponding up counter, the comparator match detect signal goes active. If the value set in the timer register is 00H, the signal goes active when the up counter overflows.

The TA0REG are double buffer structure, each of which makes a pair with register buffer.

The setting of the bit TA01RUN<TA0RDE> determines whether TA0REG's double buffer structure is enabled or disabled. It is disabled if <TA0RDE> = 0 and enabled if <TA0RDE> = 1.

When the double buffer is enabled, data is transferred from the register buffer to the timer register when a 2^n overflow occurs in PWM mode, or at the start of the PPG cycle in PPG mode. Hence the double buffer cannot be used in timer mode.

(When using the double buffer, method of renewing timer register is only overflow in PWM mode or frequency agreement in PPG mode.)

A reset initializes <TA0RDE> to 0, disabling the double buffer. To use the double buffer, write data to the timer register, set <TA0RDE> to 1, and write the following data to the register buffer. Figure 3.12.5 shows the configuration of TA0REG.

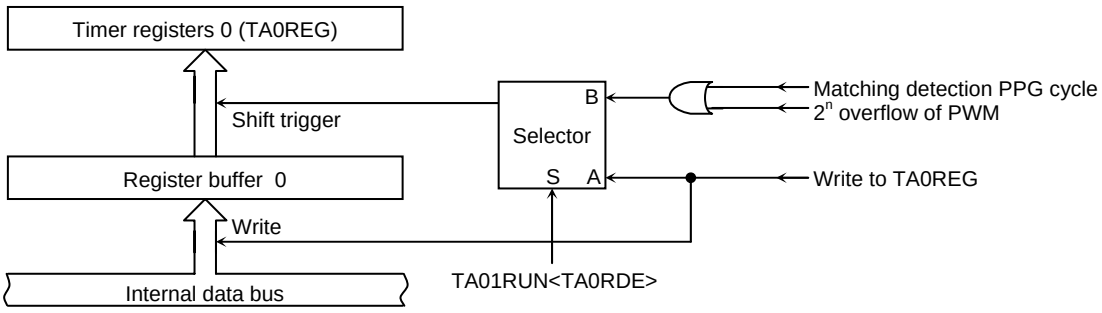


Figure 3.12.5 Configuration of timer register (TA0REG)

Note: The same memory address is allocated to the timer register and the register buffer 0. When <TA0RDE> = 0, the same value is written to the register buffer 0 and the timer register; when <TA0RDE> = 1, only the register buffer 0 is written to.

(4) Comparator (CP0, CP1)

The comparator compares the value in an up counter with the value set in a timer register. If they match, the up counter is cleared to 0 and an interrupt signal (INTTA0 or INTTA1) is generated. If timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

Note: If a value smaller than the up-counter value is written to the timer register while the timer is counting up, this will cause the timer to overflow and an interrupt cannot be generated at the expected time. (The value in the timer register canbe changed without any problem if the new value is larger than the up-counter value.) In 16-bit interval timer mode, be sure to write to both TA0REG and TA1REG in this order (16 bits in total), The compare circuit will not function if only the lower 8 bits are set.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detects signal (8-bit comparator output) of each interval timer.

Whether inversion is enabled or disabled is determined by the setting of the bit TA1FFCR<TA1FFIE> in the timer flip-flops control register. A reset clears the value of TA1FF to 0. Writing 01 or 10 to TA1FFCR<TA1FFC1:0> sets TA1FF to 0 or 1. Writing 00 to these bits inverts the value of TA1FF. (This is known as software inversion.)

The TA1FF signal is output via the TA1OUT pin. When this pin is used as the timer output, the timer flip-flop should be set beforehand using the port function registers.

The condition for TA1FF inversion varies with mode as shown below

8-bit interval timer mode	: UC0 matches TA0REG or UC1 matches TA1REG (Select either one of the two)
16-bit interval timer mode	: UC0 matches TA0REG or UC1 matches TA1REG
80bit PWM mode	: UC0 matches TA0REG or a 2^n overflow occurs
8-bit PPG mode	: UC0 matches TA0REG or UC0 matches TA1REG

Note: If an inversion by the match-detect signal and a setting change via the TMRA1 flip-flopcontrol register occur simultaneously, the resultant operation varies depending on the situation, as shown below.

- If an inversion by the match-detect signal and an inversion via the register occur simultaneously, the flip-flop will be inverted only once.
- If an inversion by the match-detect signal and an attempt to set the flip-flop to 1 via the register occur simultaneously, the timer flip-flop will be set to 1.
- If an inversion by the match-detect signal and an attempt to clear the flip-flop to 0 via the register occur simultaneously the flip-flop will be cleared to 1.

Be sure to stop the timer before changing the flip-flop incersion setting.

If the setting is chaged while the timer is counting, proper operation cannot be obtained.

3.12.3 SFR

TMRA01 RUN Register

TA01RUN (1100H)

	7	6	5	4	3	2	1	0
Bit symbol	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
Read/Write	R/W				R/W			
After Reset	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA01 prescaler	Up counter (UC1)	Up counter (UC0)
						0: Stop and clear 1: Run (Count up)		

TA0REG double buffer control

0	Disable
1	Enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA01RUN are “1” when read.

TMRA23 RUN Register

TA23RUN (1108H)

	7	6	5	4	3	2	1	0
Bit symbol	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
Read/Write	R/W				R/W			
After Reset	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA23 prescaler	Up counter (UC3)	Up counter (UC2)
						0: Stop and clear 1: Run (Count up)		

TA3REG double buffer control

0	Disable
1	Enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA23RUN are “1” when read.

Figure 3.12.6 Register for TMRA (1)

TMRA45 RUN Register

TA45RUN (1110H)

	7	6	5	4	3	2	1	0
Bit symbol	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
Read/Write	R/W				R/W			
After Reset	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA45 prescaler	Up counter (UC5)	Up counter (UC4)
						0: Stop and clear 1: Run (Count up)		

TA4REG double buffer control

0	Disable
1	Enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA45RUN are “1” when read.

TMRA67RUN Register

TA67RUN (1118H)

	7	6	5	4	3	2	1	0
Bit symbol	TA6RDE				I2TA67	TA67PRUN	TA7RUN	TA6RUN
Read/Write	R/W				R/W			
After Reset	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA67 prescaler	Up counter (UC7)	Up counter (UC6)
						0: Stop and clear 1: Run (Count up)		

TA6REG double buffer control

0	Disable
1	enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA67RUN are “1” when read.

Figure 3.12.7 Register for TMRA (2)

TMRA01 Mode Register

TA01MOD (1104H)

	7	6	5	4	3	2	1	0
Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA1 00: TA0TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA0 00: TA0IN pin 01: φT1 10: φT4 11: φT16	

TMRA0 input clock

<TA0CLK1:0>	00	TA0IN (External input)
	01	φT1
	10	φT4
	11	φT16

TMRA1 input clock

		TA01MOD<TA01M1:0>≠01	TA01MOD<TA01M1:0>=01
<TA1CLK1:0>	00	Comparator output from TMRA0	Overflow output from TMRA0 (16-bit timer mode)
	01	φT1	
	10	φT16	
	11	φT256	

PWM cycle selection

<PWM01:00>	00	Reserved
	01	2^6 × Clock source
	10	2^7 × Clock source
	11	2^8 × Clock source

TMRA01 operation mode selection

<TA01MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA0), 8-bit timer (TMRA1)

Figure 3.12.8 Register for TMRA (4)

TMRA23 Mode Register

TA23MOD (110CH)

	7	6	5	4	3	2	1	0
Bit symbol	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA3 clock for TMRA3 00: TA2TRG 01: φT1 10: φT16 11: φT256		TMRA2 clock for TMRA2 00: TA2IN pin 01: φT1 10: φT4 11: φT16	

TMRA2 input clock

<TA2CLK1:0>	00	TA2IN (External input)
	01	φT1
	10	φT4
	11	φT16

TMRA3 input clock

		TA23MOD<TA23M1:0>≠01	TA23MOD<TA23M1:0>=01
<TA3CLK1:0>	00	Comparator output from TMRA2	Overflow output from TMRA2 (16-bit timer mode)
	01	φT1	
	10	φT16	
	11	φT256	

PWM cycle selection

<PWM21:20>	00	Reserved
	01	2^6 × Clock source
	10	2^7 × Clock source
	11	2^8 × Clock source

TMRA23 operation mode selection

<TA23MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA2), 8-bit timer (TMRA3)

Figure 3.12.9 Register for TMRA (5)

TMRA45 Mode Register

TA45MOD (1114H)	7	6	5	4	3	2	1	0
Bit symbol	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA5 clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA4 clock for TMRA4 00: low-frequency clock 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA4 input clock

<TA4CLK1:0>	00	low-frequency clock(fs)
	01	ϕT1
	10	ϕT4
	11	ϕT16

TMRA5 input clock

		TA45MOD<TA45M1:0>≠01	TA45MOD<TA45M1:0>=01
<TA5CLK1:0>	00	Comparator output from TMRA4	Overflow output from TMRA4 (16-bit timer mode)
	01	ϕT1	
	10	ϕT16	
	11	ϕT256	

PWM cycle selection

<PWM41:40>	00	Reserved
	01	$2^6 \times$ Clock source
	10	$2^7 \times$ Clock source
	11	$2^8 \times$ Clock source

TMRA45 operation mode selection

<TA45MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA4), 8-bit timer (TMRA5)

Figure 3.12.10 Register for TMRA (6)

TMRA67 Mode Register

TA67MOD (111CH)	7	6	5	4	3	2	1	0
Bit symbol	TA67M1	TA67M0	PWM61	PWM60	TA7CLK1	TA7CLK0	TA6CLK1	TA6CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA7 clock for TMRA7 00: TA6TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA6 clock for TMRA6 00: low-frequency clock 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA6 input clock

<TA6CLK1:0>	00	low-frequency clock(fs)
	01	ϕT1
	10	ϕT4
	11	ϕT16

TMRA1 input clock

		TA67MOD<TA67M1:0>≠01	TA67MOD<TA67M1:0>=01
<TA7CLK1:0>	00	Comparator output from TMRA6	Overflow output from TMRA6 (16-bit timer mode)
	01	ϕT1	
	10	ϕT16	
	11	ϕT256	

PWM cycle selection

<PWM61:60>	00	Reserved
	01	$2^6 \times$ Clock source
	10	$2^7 \times$ Clock source
	11	$2^8 \times$ Clock source

TMRA67 operation mode selection

<TA67MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA6), 8-bit timer (TMRA7)

Figure 3.12.11 Register for TMRA (7)

TMRA1 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA1FFCR (1105H)	Bit symbol					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
	Read/Write					R/W		R/W	
	After reset					1	1	0	0
Read-modify-write instructions are prohibited.	Function					00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don't care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1: TMRA1

Inversion signal for timer flip-flop 1 (TA1FF)
(Don't care except in 8-bit timer mode)

TA1FFIS	0	Inversion by TMRA0
	1	Inversion by TMRA1

Inversion of TA1FF

TA1FFIE	0	Disabled
	1	Enabled

Control of TA1FF

<TA1FFC1:0>	00	Inverts the value of TA1FF (Software inversion)
	01	Sets TA1FF to "1"
	10	Clears TA1FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA1FFCR are "1" when read.

Figure 3.12.12 Register for TMRA (8)

TMRA3 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA3FFCR (110DH)	Bit symbol					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
	Read/Write					R/W		R/W	
Read-modify-write instructions are prohibited.	After reset					1	1	0	0
	Function					00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don't care		TA3FF control for inversion 0: Disable 1: Enable	TA3FF inversion select 0: TMRA2 1: TMRA3

Inversion signal for timer flip-flop 3 (TA3FF)
(Don't care except in 8-bit timer mode)

TA3FFIS	0	Inversion by TMRA2
	1	Inversion by TMRA3

Inversion of TA3FF

TA3FFIE	0	Disabled
	1	Enabled

Control of TA3FF

<TA3FFC1:0>	00	Inverts the value of TA3FF (Software inversion)
	01	Sets TA3FF to "1"
	10	Clears TA3FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA3FFCR are "1" when read.

Figure 3.12.13 Register for TMRA (9)

TMRA5 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA5FFCR (1115H)	Bit symbol					TA5FFC1	TA5FFC0	TA5FFIE	TA5FFIS
	Read/Write					R/W		R/W	
	After reset					1	1	0	0
Read-modify-write instructions are prohibited.	Function					00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF control for inversion 0: Disable 1: Enable	TA5FF inversion select 0: TMRA4 1: TMRA5

Inversion signal for timer flip-flop 5 (TA5FF)
(Don't care except in 8-bit timer mode)

TA5FFIS	0	Inversion by TMRA4
	1	Inversion by TMRA5

Inversion of TA5FF

TA5FFIE	0	Disabled
	1	Enabled

Control of TA5FF

<TA5FFC1:0>	00	Inverts the value of TA5FF (Software inversion)
	01	Sets TA5FF to "1"
	10	Clears TA5FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA5FFCR are "1" when read.

Figure 3.12.14 Register for TMRA (10)

TMRA7 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA7FFCR (111DH)	Bit symbol					TA7FFC1	TA7FFC0	TA7FFIE	TA7FFIS
	Read/Write					R/W		R/W	
	After reset					1	1	0	0
Read-modify-write instructions are prohibited.	Function					00: Invert TA7FF 01: Set TA7FF 10: Clear TA7FF 11: Don't care		TA7FF control for inversion 0: Disable 1: Enable	TA7FF inversion select 0: TMRA6 1: TMRA7

Inversion signal for timer flip-flop 7 (TA7FF)
(Don't care except in 8-bit timer mode)

TA7FFIS	0	Inversion by TMRA6
	1	Inversion by TMRA7

Inversion of TA7FF

TA7FFIE	0	Disabled
	1	Enabled

Control of TA7FF

<TA7FFC1:0>	00	Inverts the value of TA7FF (Software inversion)
	01	Sets TA7FF to "1"
	10	Clears TA7FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA7FFCR are "1" when read.

Figure 3.12.15 Register for TMRA (11)

Timer Registers

		7	6	5	4	3	2	1	0
TA0REG (1102H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA1REG (1103H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA2REG (110AH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA3REG (110BH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA4REG (1112H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA5REG (1113H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA6REG (111AH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TA7REG (111BH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0

Note: All registers are prohibited to execute read-modify-write instruction.

Figure 3.12.16 TMRA Registers

3.12.4 Operation in Each Mode

(1) 8-bit timer mode

Both TMRA0 and TMRA1 can be used independently as 8-bit interval timers.

a. Generating interrupts at a fixed interval (Using TMRA1)

To generate interrupts at constant intervals using TMRA1 (INTTA1), first stop TMRA1 then set the operation mode, input clock and a cycle to TA01MOD and TA1REG register respectively. Then, enable the interrupt INTTA1 and start TMRA1 counting.

Example: To generate an INTTA1 interrupt every 20 us at f_C = 50 MHz, set each register as follows;

* Clock state: Clcok gear : 1/1; Prescaler of clock gear : 1/2

	MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	← –	X	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	← 0	0	X	X	0	1	X	X	Select 8-bit timer mode and select φT1 (0.16 μs at f_C = 50 MHz) as the input clock.
TA1REG	← 0	1	1	1	1	1	0	1	Set TA1REG to 20 μs ÷ φT1 = 125(7DH)
INTETA1	← X	1	0	1	X	–	–	–	Enable INTTA1 and set it to level 5.
TA01RUN	← –	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don't Care、–: No change

Select the input clock using Table 3.12.2.

Note: The input clocks for TMRA0 and TMRA1 are different from as follows.
TMRA0: TA0IN input, φT1, φT4 or φT16.
TMRA1: Match output of TMRA0, φT1, φT16, and φT256.

b. Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 3.2μs square wave pulse from the TA1OUT pin at f_C= 50 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 may be used.

* Clock state: Clcok gear : 1/1; Prescaler of clock gear : 1/2

		7	6	5	4	3	2	1	0	
TA01RUN	←	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to “0”.
TA01MOD	←	0	0	X	X	0	1	X	X	Select 8-bit timer mode and select φT1 (0.16 μs at f_C = 50 MHz) as the input clock.
TA1REG	←	0	0	0	0	1	0	1	0	Set the timer register to 3.2 μs ÷ φT1 ÷ 2 = 0AH
TA1FFCR	←	X	X	X	X	1	0	1	1	Clear TA1FF to “0” and set it to invert on the match detect signal from TMRA1.
PM	←	–	X	X	X	X	–	0	X	Set PM1 to function as the TA1OUT pin.
PMFC	←	–	X	X	X	X	–	1	X	
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don’t care, –: No change

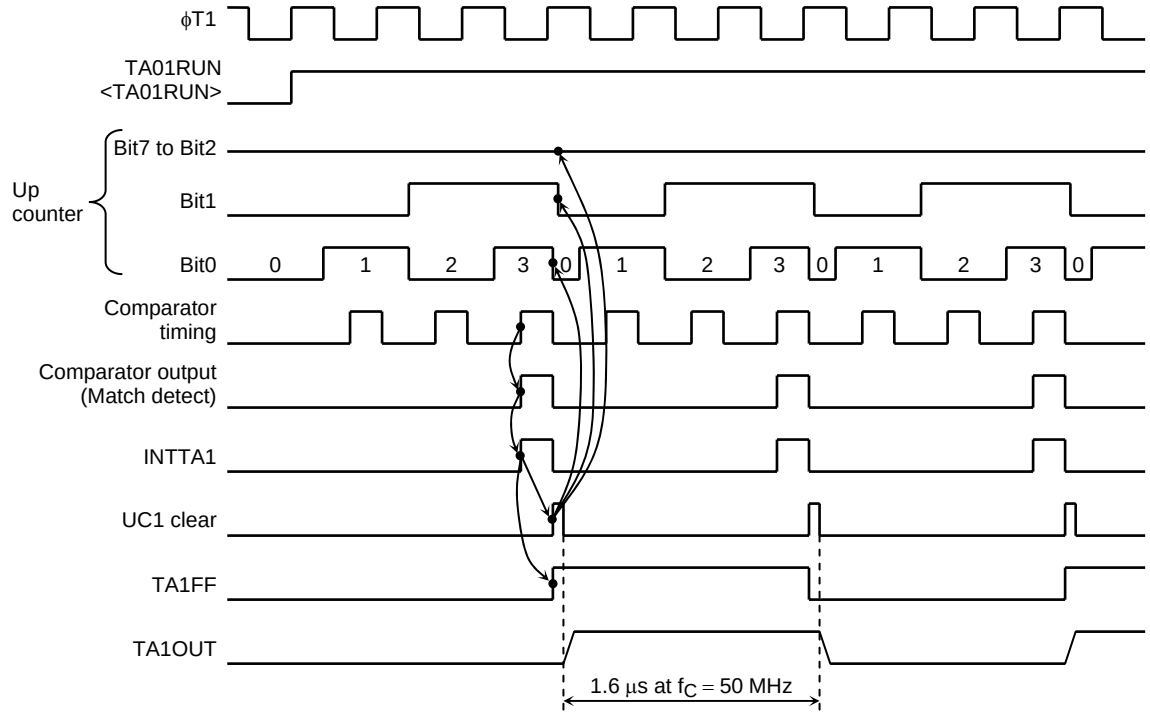


Figure 3.12.17 Square Wave Output Timing Chart (50% duty)

c. Making TMRA1 count up on the match signal from the TMRA0 comparator

Select 8-bit timer mode and set the comparator output from TMRA0 to be the input clock to TMRA1.

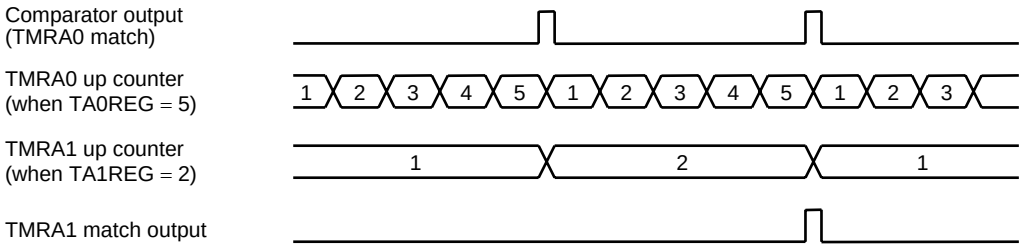


Figure 3.12.18 TMRA1 Count Up on Signal from TMRA0

(2) 16 bit timer mode

Pairing the two 8-bit timers TMRA0 and TMRA1 configures a 16-bit interval timer. To make a 16-bit interval timer in which TMRA0 and TMRA1 are cascaded together, set TA01MOD<TA01M1:0> to 01.

In 16-bit timer mode, the overflow output from TMRA0 is used as the input clock for TMRA1, regardless of the value set in TA01MOD<TA01CLK1:0>. Table 3.12.2shows the relationship between the timer (Interrupt) cycle and the input clock selection.

Example: To generate an INTTA1 interrupt every 0.13 s at f_{SYS} = 50 MHz, set the timer registers TA0REG and TA1REG as follows:

* Clock state
- Clcok gear : 1/1
- Prescaler of clock gear : 1/2

If ϕT16 (2.6 μs at f_{SYS} = 50 MHz) is used as the input clock for counting, set the following value in the registers: 0.13 s ÷ 2.6 μs = 50000 = C350H; e.g. set TA1REG to C3H and TA0REG to 50H.

The comparator match signal is output from TMRA0 each time the up counter UC0 matches TA0REG, though the up counter UC0 is not be cleared.

In the case of the TMRA1 comparator, the match detect signal is output on each comparator pulse on which the values in the up counter UC1 and TA1REG match. When the match detect signal is output simultaneously from both the comparator TMRA0 and TMRA1, the up counters UC0 and UC1 are cleared to 0 and the interrupt INTTA1 is generated. Also, if inversion is enabled, the value of the timer flip-flop TA1FF is inverted.

Example: When TA1REG = 04H and TA0REG = 80H

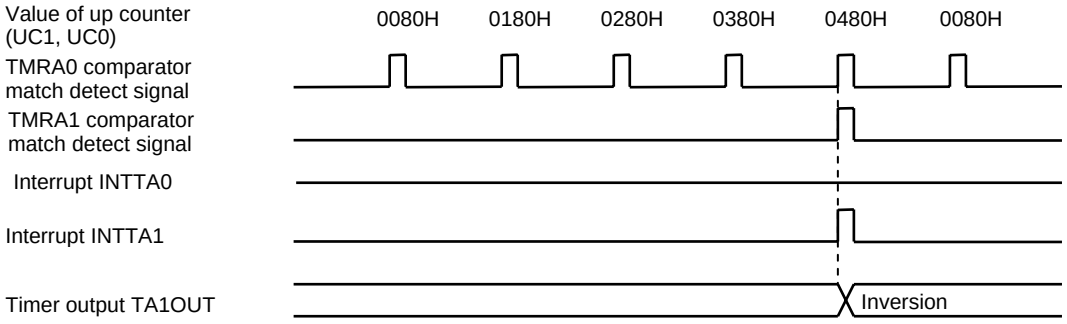


Figure 3.12.19 Timer Output by 16-Bit Timer Mode

(3) 8-bit PPG (Programmable pulse generation) output mode

Square wave pulses can be generated at any frequency and duty ratio by TMRA0. The output pulses may be active-low or active-high. In this mode TMRA1 cannot be used.

TMRA0 outputs pulses on the TA1OUT pin.

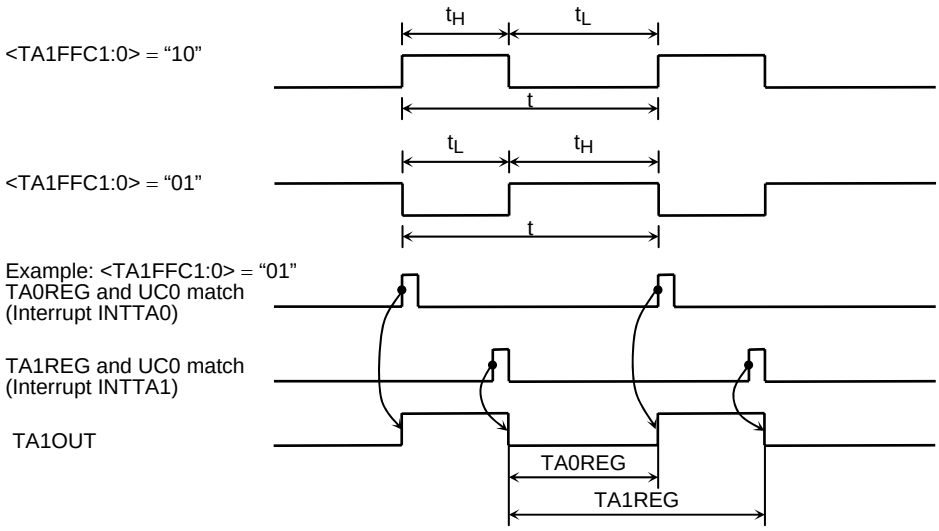


Figure 3.12.20 8-Bit PPG Output Waveforms

In this mode a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC0) matches the value in one of the timer registers TA0REG or TA1REG.

The value set in TA0REG must be smaller than the value set in TA1REG.

Although the up counter for TMRA1 (UC1) is not used in this mode, TA01RUN<TA1RUN> should be set to 1 so that UC1 is set for counting.

Figure 3.12.21 shows a block diagram representing this mode.

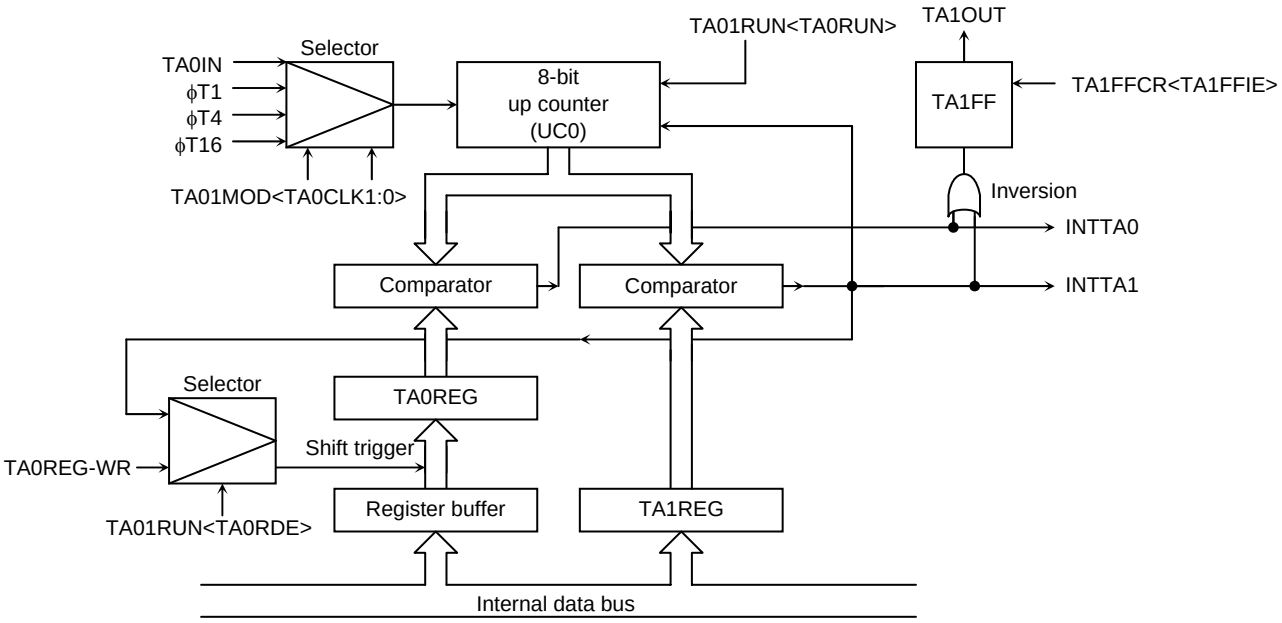


Figure 3.12.21 Block Diagram of 8-Bit PPG Output Mode

If the TA0REG double buffer is enabled in this mode, the value of the register buffer will be shifted into TA0REG each time TA1REG matches UC0.

Use of the double buffer facilitates the handling of low-duty waves (when duty is varied).

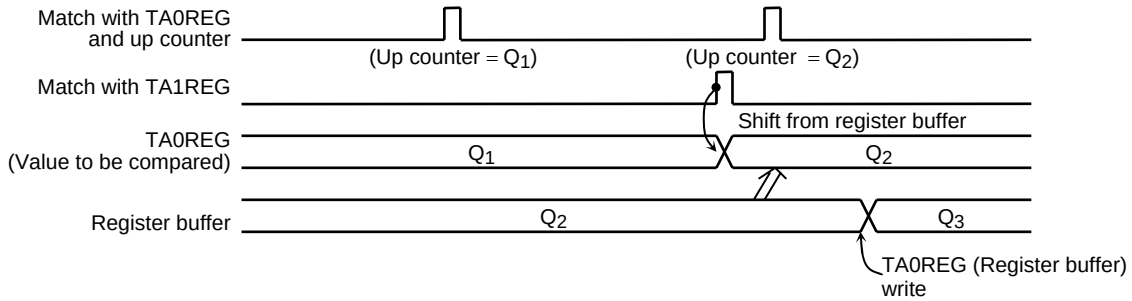
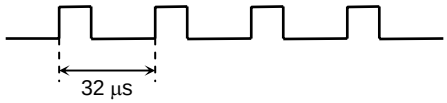


Figure 3.12.22 Operation of Register Buffer

Note: The values that can be set in TAxREG renge from 01h to 00h (equivalent to 100h). If the maximum value 00h is set , the match-detect signal goes active when the up-counter overfolws.

Example: To generate 1/4 duty 31.25 kHz pulses (at f_C= 50 MHz)



* Clock state
- Clcok gear : 1/1
- Prescaler of clock gear : 1/2

Calculate the value which should be set in the timer register.
To obtain a frequency of 31.25 kHz, the pulse cycle t should be: t = 1/31.25kHz = 32 μs
φT1 = 0.16 μs (at 50 MHz);
32 μs ÷ 0.16 μs = 200
Therefore set TA1REG to 200 (C8H)
The duty is to be set to 1/4: t × 1/4 = 32 μs × 1/4 = 8 μs
8 μs ÷ 0.16 μs = 50
Therefore, set TA0REG = 50 = 32H.

		7	6	5	4	3	2	1	0	
TA01RUN	←	–	X	X	X	–	–	0	0	Stop TMRA0 and TMRA1 and clear it to “0”.
TA01MOD	←	1	0	X	X	X	X	0	1	Set the 8-bit PPG mode, and select φT1 as input clock.
TA0REG	←	0	0	0	0	1	0	1	0	Write 32H.
TA1REG	←	1	1	0	0	1	0	0	0	Write C8H.
TA1FFCR	←	X	X	X	X	0	1	1	X	Set TA1FF, enabling both inversion and the double buffer. Writing 10 provides negative logic pulse.
PM	←	–	X	X	X	X	–	0	X	Set PM1 as the TA1OUT pin.
PMFC	←	–	X	X	X	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	1	1	Start TMRA0 and TMRA1 counting.

X: Don't care, –: No change

(4) 8-bit PWM (Pulse width modulation) output mode

This mode is only valid for TMRA0. In this mode, a PWM pulse with the maximum resolution of 8 bits can be output.

When TMRA0 is used the PWM pulse is output on the TA1OUT pin (Shared with PM1). TMRA1 can also be used as an 8-bit timer.

The timer output is inverted when the up counter (UC0) matches the value set in the timer register TA0REG or when 2^n counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01:00>). The up counter UC0 is cleared when 2^n counter overflow occurs.

The following conditions must be satisfied before this PWM mode can be used.

Value set in TA0REG < Value set for 2^n counter overflow

Value set in TA0REG ≠ 0

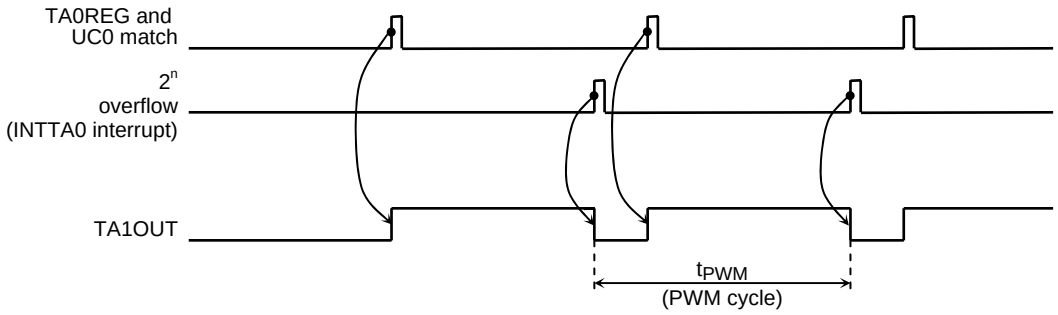


Figure 3.12.23 8-Bit PWM Waveforms

Figure 3.12.24 shows a block diagram representing this mode.

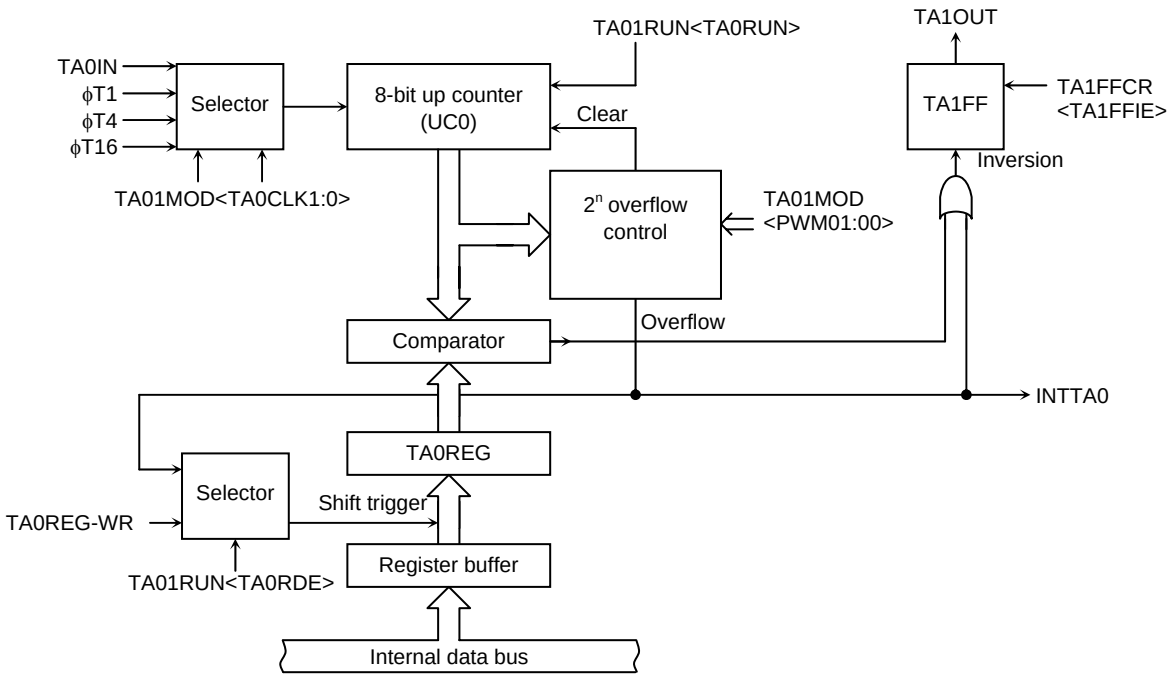


Figure 3.12.24 Block Diagram of 8-Bit PWM Mode

In this mode the value of the register buffer will be shifted into TA0REG if 2^n overflow is detected when the TA0REG double buffer is enabled.

Use of the double buffer facilitates the handling of low duty ratio waves.

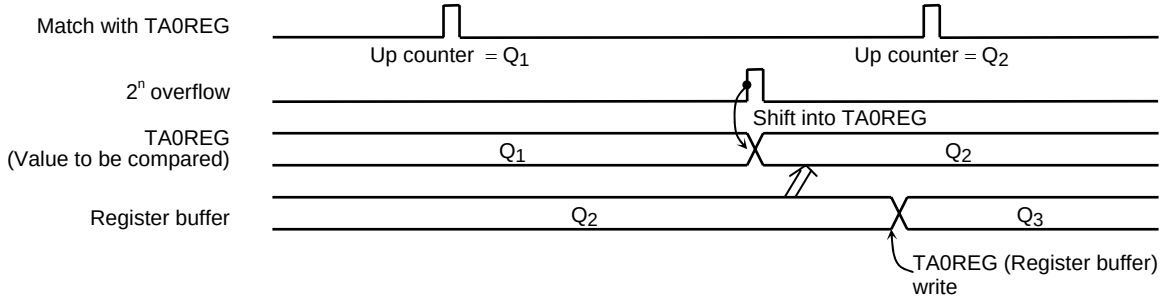
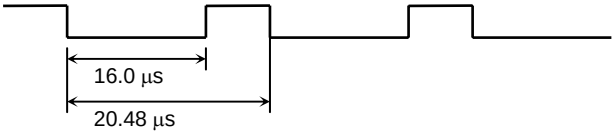


Figure 3.12.25 Register Buffer Operation

Example: To output the following PWM waves on the TA1OUT pin (at f_C = 50 MHz).



* Clock state — Clcok gear : 1/1; Prescaler of clock gear : 1/2

To achieve a 20.48μs PWM cycle by setting φT1 to 0.16 μs (at f_C = 50 MHz):
20.48 μs ÷ 0.16 μs = 128
2^n = 128
Therefore n should be set to 7.
Since the low level period is 16.0 μs when φT1 = 0.16 μs,
set the following value for TAREG:
16.0 μs ÷ 0.16 μs = 100 = 64H

	MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	← –	X	X	X	–	–	–	0	Stop TMRA0 and clear it to 0
TA01MOD	← 1	1	1	0	X	X	0	1	Select 8-bit PWM mode (cycle: 2^7) and select φT1 as the input clock.
TA0REG	← 0	1	1	0	0	1	0	0	Write 64H.
TA1FFCR	← X	X	X	X	1	0	1	X	Clear TA1FF to 0, enable the inversion and double buffer.
PM	← –	X	X	X	X	–	0	0	Set PM1 as the TA1OUT pin.
PMFC	← –	X	X	X	X	–	1	X	
TA01RUN	← 1	X	X	X	–	1	–	1	Start TMRA0 counting.

X: Don't care, –: No change

Table 3.12.3 PWM Cycle

	Clock gear selection SYSCR1 <GEAR2:0>	Prescaler of clock gear SYSCR0 <PRCK>		PWM cycle TAxxMOD<PWMx1:0>								
				2^6 (x64)			2^7(x128)			2^8(x256)		
				TAxxMOD<TAxCLK1:0>			TAxxMOD<TAxCLK1:0>			TAxxMOD<TAxCLK1:0>		
				φT1(x2)	φT4(x8)	φT16(x32)	φT1(x2)	φT4(x8)	φT16(x32)	φT1(x2)	φT4(x8)	φT16(x32)
1/fc	000(x1)	0(x2)	x2	512/fc	2048/fc	8192/fc	1024/fc	4096/fc	16384/fc	2048/fc	8192/fc	32768/fc
	001(x2)			1024/fc	4096/fc	16384/fc	2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc
	010(x4)			2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc
	011(x8)			4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc
	100(x16)			8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc
	000(x1)	1(x8)		2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc
	001(x2)			4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc
	010(x4)			8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc
	011(x8)			16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc	65536/fc	262144/fc	1048576/fc
	100(x16)			32768/fc	131072/fc	524288/fc	65536/fc	262144/fc	1048576/fc	131072/fc	524288/fc	2097152/fc

(5) Settings for each mode

Table 3.12.4 shows the SFR settings for each mode.

Table 3.12.4 Timer Mode Setting Registers

Register Name	TA01MOD				TA1FFCR
<Bit Symbol>	<TA01M1:0>	<PWM01:00>	<TA1CLK1:0>	<TA0CLK1:0>	TA1FFIS
Function	Timer Mode	PWM Cycle	Upper Timer Input Clock	Lower Timer Input Clock	Timer F/F Invert Signal Select
8-bit timer × 2 channels	00	–	Lower timer match φT1, φT16, φT256 (00, 01, 10, 11)	External clock φT1, φT4, φT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
16-bit timer mode	01	–	–	External clock φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit PPG × 1 channel	10	–	–	External clock φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit PWM × 1 channel	11	2^6, 2^7, 2^8 (01, 10, 11)	–	External clock φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit timer × 1 channel	11	–	φT1, φT16, φT256 (01, 10, 11)	–	Output disabled

–: Don't care

3.13 16 bit timer / Event counter (TMRB)

The TMP92CZ26A incorporates two multifunctional 16-bit timer/event counter (TMRB0, TMRB1) which have the following operation modes:

- 16 bit interval timer mode
- 16 bit event counter mode
- 16 bit programmable pulse generation mode (PPG)

Can be used following operation modes by capture function.

- Frequency measurement mode
- Pulse width measurement mode

Timer/event counter consists of a 16-bit up counter, two 16-bit timer registers (One of them with a double-buffer structure), a 16-bit capture registers, two comparators, a capture input controller, a timer flip-flop and a control circuit.

Timer/event counter is controlled by an 11-byte control SFR.Each channel(TMRB0,TMRB1) operate independently.In this section, the explanation describes only for TMRB0 because each channel is identical operation except for the difference as follows;

Table 3.13.1 Difference between TMRB0 and TMRB1

Specification \ Channel		TMRB0	TMRB1
External pins	External clock/ capture trigger input pins	TB0IN0 (Shared with PP4)	TB1IN0 (Shared with PP5)
	Timer flip-flop output pins	TB0OUT0 (Shared with PP6)	TB1OUT0 (Shared with PP7)
SFR (Address)	Timer run register	TB0RUN (1180H)	TB1RUN (1190H)
	Timer mode register	TB0MOD (1182H)	TB1MOD (1192H)
	Timer flip-flop control register	TB0FFCR (1183H)	TB1FFCR (1193H)
	Timer register	TB0RG0L (1188H) TB0RG0H (1189H) TB0RG1L (118AH) TB0RG1H (118BH)	TB1RG0L (1198H) TB1RG0H (1199H) TB1RG1L (119AH) TB1RG1H (119BH)
	Capture register	TB0CP0L (118CH) TB0CP0H (118DH) TB0CP1L (118EH) TB0CP1H (118FH)	TB1CP0L (119CH) TB1CP0H (119DH) TB1CP1L (119EH) TB1CP1H (119FH)

3.13.1 Block diagram

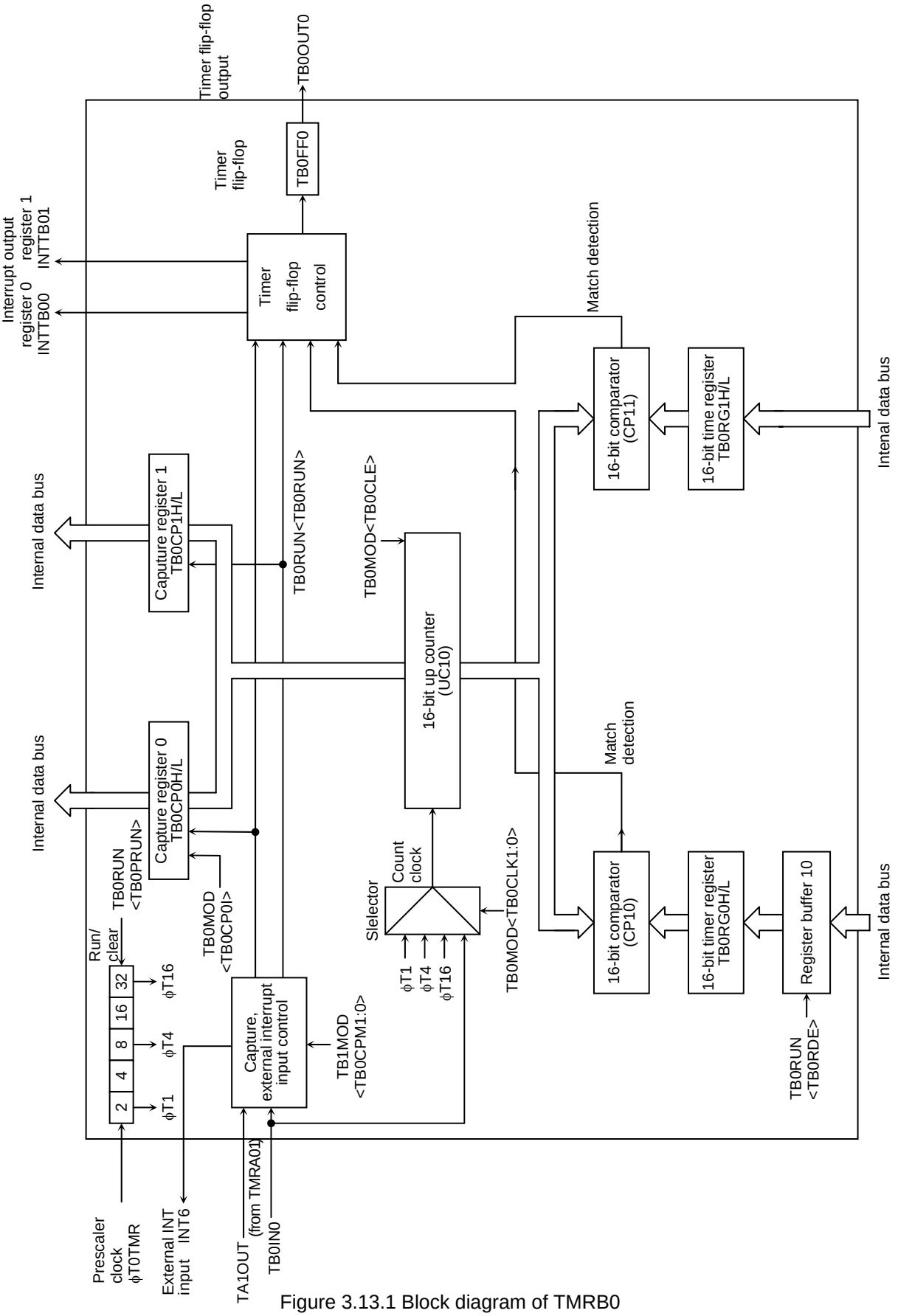


Figure 3.13.1 Block diagram of TMRB0

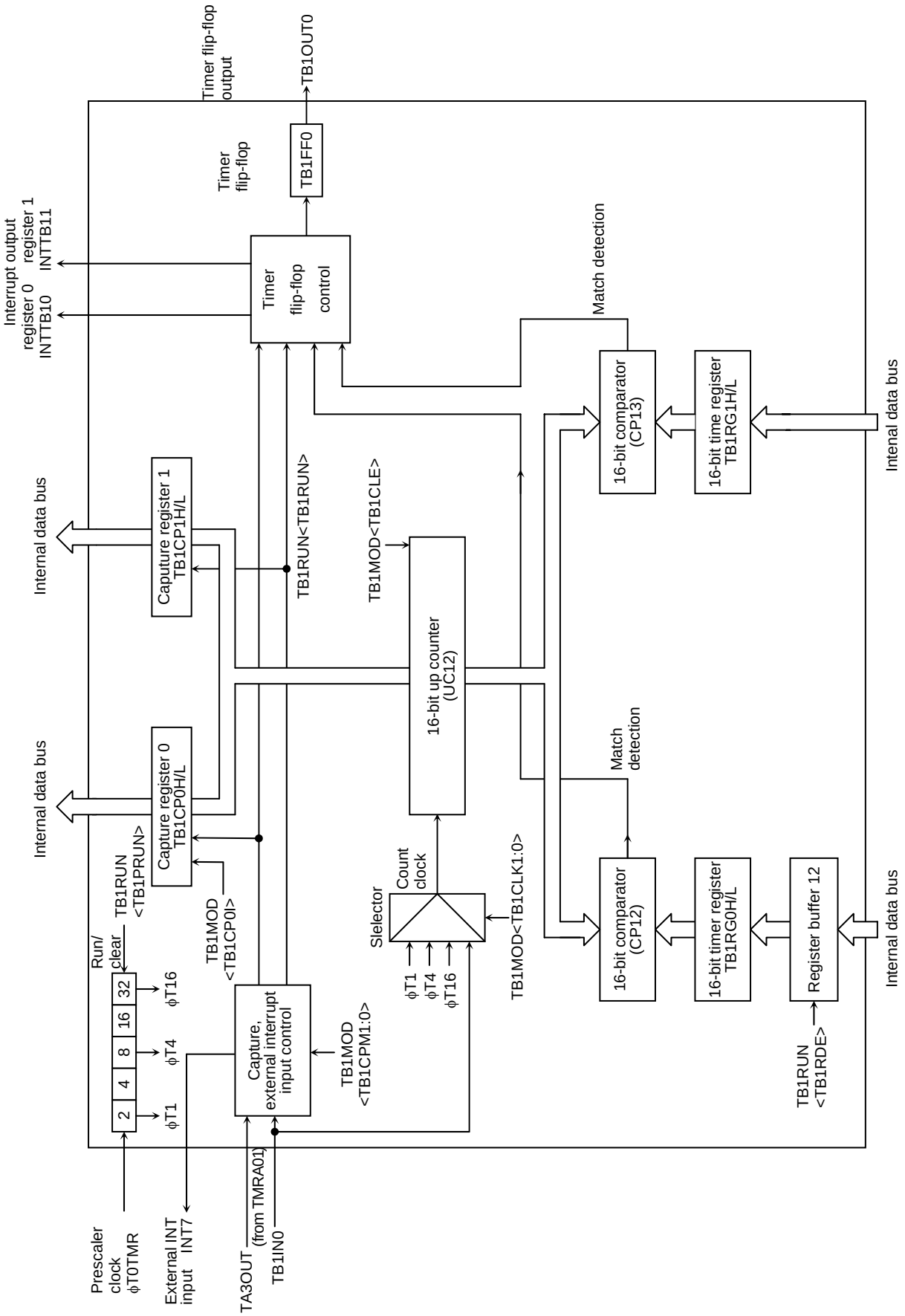


Figure 3.13.2 Block diagram of TMRB1

3.13.2 Operation

(1) Prescaler

The 5-bit prescaler generates the source clock for TMRB0. The prescaler clock (ϕT0) is selected by the register SYSCR0<PRCK> of clock gear. This prescaler can be started or stopped using TB0RUN<TB0RUN>. Counting starts when <TB0RUN> is set to "1"; the prescaler is cleared to "0" and stops operation when <TB0RUN> is cleared to "0".

The resolution of prescaler is showed in the Table 3.13.2.

Table 3.13.2 Prescaler Clock Resolution

	Clock gear selection SYSCR1 <GEAR2:0>	Prescaler of clock gear SYSCR0 <PRCK>	–	Timer counter input clock Prescaler of TMRB TBxMOD<TBxCLK1:0>		
				ϕT1(1/2)	ϕT4(1/8)	ϕT16(1/32)
fc	1/1	1/2	1/2	fc/8	fc/32	fc/128
	1/2			fc/16	fc/64	fc/256
	1/4			fc/32	fc/128	fc/512
	1/8			fc/64	fc/256	fc/1024
	1/16			fc/128	fc/512	fc/2048
	1/1	1/8		fc/32	fc/128	fc/512
	1/2			fc/64	fc/256	fc/1024
	1/4			fc/128	fc/512	fc/2048
	1/8			fc/256	fc/1024	fc/4096
	1/16			fc/512	fc/2048	fc/8192

(2) Up counter (UC10)

UC10 is a 16-bit binary counter which counts up pulses input from the clock specified by TB0MOD<TB0CLK1:0>.

Any one of the prescaler internal clocks ϕT1, ϕTB0 and ϕT16 or an external clock input via the TB0IN0 pin can be selected as the input clock. Counting or stopping and clearing of the counter is controlled by TB0RUN<TB0RUN>.

When clearing is enabled, the up counter UC10 will be cleared to zero each time its value matches the value in the timer register TB0RG1H/L. Clearing can be enabled or disabled using TB0MOD<TB0CLE>.

If clearing is disabled, the counter operates as a free running counter.

(3) Timer registers (TB0RG0H/L, TB0RG1H/L)

These two 16-bit registers are used to set the interval time. When the value in the up counter UC10 matches the value set in this timer register, the comparator match detect signal will go active.

Setting data for both upper and lower timer registers is needed. For example, using 2-byte data transfer instruction or using 1-byte data transfer instruction twice for lower 8 bits and upper 8 bits in order.

(The compare circuit will not operate if only the lower 8 bits are written. Be sure to write to both timer registers (16 bits) from the lower 8 bits followed by the upper 8 bits.)

The TB0RG0H/L timer register has a double-buffer structure, which is paired with register buffer 10. The value set in TB0RUN<TB0RDE> determines whether the double-buffer structure is enabled or disabled: it is disabled when <TB0RDE> = "0", and enabled when <TB0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer 10 to the timer register when the values in the up counter (UC10) and the timer register TB0RG1H/L match.

The double buffer circuit incorporates two flags to indicate whether or not data is written to the lower 8 bits and the upper 8 bits of the register buffer, respectively. Only when both flags are set can data be transferred from the register buffer to the timer register by a match between the up-counter UC10 and the timer register TB0RG1. This data transfer is performed so long as 16-bit data is written in the register buffer regardless of the register buffer to the timer register unexpectedly as explained below.

For example, let us assume that an interrupt occurs when only the lower 8 bits (L1) of the register buffer data (H1L1) have been written and the interrupt routine includes writes to all 16 bits in the register buffer and a transfer of the data to the timer register. In this case, if the higher 8 bits (H1) are written after the interrupt routine is completed, only the flag for the higher 8 bits will be set, the flag for the lower 8 bits having been cleared in the interrupt routine. Therefore, even if a match occurs between UC10 and TB0RG1, no data transfer will be performed.

Then, in an attempt to set the next set of data (H2L2) in the register buffer, when the lower 8 bits (L2) are written, this will cause the flag for the lower 8 bits to be set as well as the flag for the higher 8 bits which has been set by writing the previous data (H1). If a match between UC10 and TB0RG1 occurs before the higher 8 bits (H2) are written, this will cause unexpected data (H1L2) to be sent to the timer register instead of the intended data (H2L2).

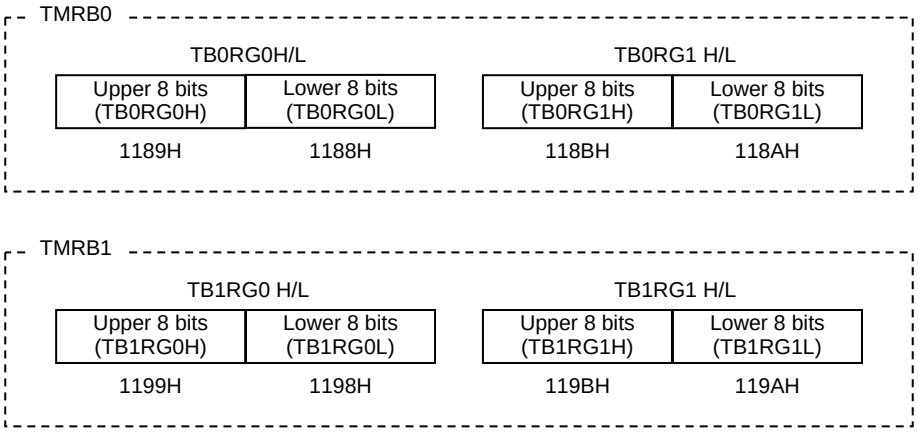
To avoid such transfer timing problems due to interrupts, the DI instruction (disable interrupts) and the EI (enable interrupts) can be executed before and after setting data in the register buffer, respectively.

After a reset, TB0RG0H/L and TB0RG1H/L are undefined. If the 16-bit timer is to be used after a reset, data should be written to it beforehand.

On a reset <TB0RDE> is initialized to "0", disabling the double buffer. To use the double buffer, write data to the timer register, set <TB0RDE> to "1", then write data to the register buffer 10 as shown below.

TB0RG0H/L and the register buffer 10 both have the same memory addresses (1188H and 1189H) allocated to them. If <TB0RDE> = “0”, the value is written to both the timer register and the register buffer 10. If <TB0RDE> = “1”, the value is written to the register buffer 10 only.

The addresses of the timer registers are as follows:



The timer registers are write-only registers and thus cannot be read.

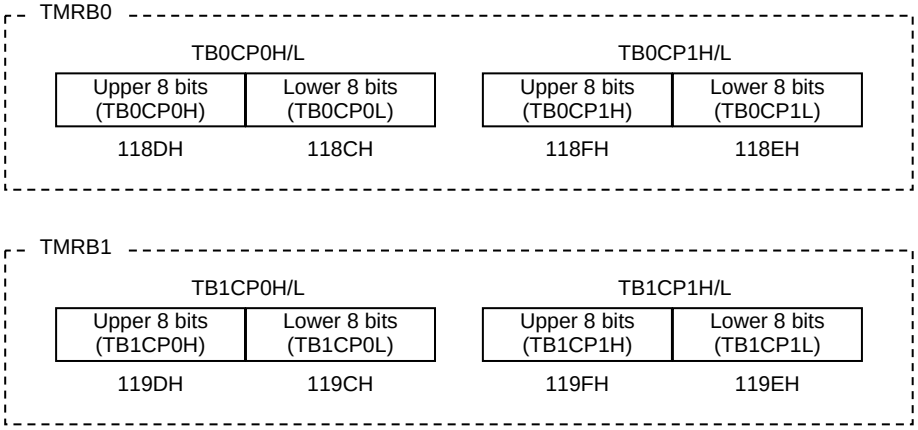
(4) Capture registers (TB0CP0H/L, TB0CP1H/L)

These 16-bit registers are used to latch the values in the up counter (UC10).

Data in the capture registers should be read all 16 bits. For example, using a 2-byte data load instruction or two 1-byte data load instructions. The least significant byte is read first, followed by the most significant byte.

(during capture is read, capture operation is prohibited. In that case, the lower 8 bits should be read first, followed by the 8 bits.)

The addresses of the capture registers are as follows;



The capture registers are read-only registers and thus cannot be written to.

(5) Capture input and external interrupt control

This circuit controls the timing to latch the value of up-counter UC10 into TB0CP0H/L and TB0CP1H/L, and generates external interrupt.The latch timing of capture register and selection of edge for external interrupt is controlled by TB0MOD<TB0CPM1:0>.

The value in the up-counter (UC10) can be loaded into a capture register by software. Whenever 0 is written to TB0MOD<TB0CP0I>, the current value in the up counter (UC10) is loaded into capture register TB0CP0H/L. It is necessary to keep the prescaler in RUN mode (e.g., TB0RUN<TB0PRUN> must be held at a value of 1).

(6) Comparators (CP10, CP11)

CP10 and CP11 are 16-bit comparators which compare the value in the up counter UC10 with the value set in TB0RG0H/L or TB0RG1H/L respectively, in order to detect a match. If a match is detected, the comparator generates an interrupt (INTTB00 or INTTB01 respectively).

(7) Timer flip-flops (TB0FF0, TB0FF1)

These flip-flops are inverted by the match detect signals from the comparators and the latch signals to the capture registers. Inversion can be enabled and disabled for each element using TB0FFCR<TB0C0T1, TB0E1T1, TB0E0T1>.

After a reset the value of TB0FF0 is undefined. If "00" is written to TB0FFCR <TB0FF0C1:0> or <TB0FF1C1:0>, TB0FF0 will be inverted. If "01" is written to the capture registers, the value of TB0FF0 will be set to "1". If "10" is written to the capture registers, the value of TB0FF0 will be set to "0".

Note: If an inversion by the match-detect signal and a setting change via the TB0FFCR register occurs simultaneously, the resultant operation varies depending on the situation, as shown below.

- If an inversion by the match-detect signal and an inversion via the register occur simultaneously, the flip-flop will be inverted only once.
- If an inversion by the match-detect siganl and an attempt to set the flip-flop to 1 via the register occur simultaneously, the flip-flop will be set to 1.
- If an inversion by the match-detect signal and an attmept to cleare the flip-flop to 0 via the register occur simultanerously, the flip-flop will be cleared to 0.

If an inversion by match-detect signal and inversion disable setting occur simultaneously, two case (it is inverted and it is not inverted) are occurred. Therefore, if changing inversion control (inversion enable/disable), stop timer operation beforehand.

The values of TB0FF0 and TB0FF1 can be output via the timer output pins TB0OUT0 (which is shared with PP6) and TB0OUT1 (which is shared with PP7). Timer output should be specified using the port P function register.

3.13.3 SFR

TMRB0 RUN Register

TB0RUN (1180H)

	7	6	5	4	3	2	1	0
Bit symbol	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
Read/Write	R/W	R/W			R/W	R/W		R/W
After Reset	0	0			0	0		0
Function	Double buffer 0: disable 1: enable	Always write “0”			In IDLE2 mode 0: Stop 1: Operate	TMRB0 prescaler		Up counter (UC10)
						0: Stop and clear 1: Run (Count up)		

Count operation

<TB0PRUN>, <TB0RUN>	0	Stop and clear
	1	Count up

Note: The 1, 4 and 5 of TB0RUN are read as “1” value.

TMRB1 RUN Register

TB1RUN (1190H)

	7	6	5	4	3	2	1	0
Bit symbol	TB1RDE	–			I2TB1	TB1PRUN		TB1RUN
Read/Write	R/W	R/W			R/W	R/W		R/W
After Reset	0	0			0	0		0
Function	Double buffer 0: disable 1: enable	Always write “0”			In IDLE2 mode 0: Stop 1: Operate	TMRB1 prescaler		Up counter (UC12)
						0: Stop and clear 1: Run (Count up)		

Count operation

<TB1PRUN>, <TB1RUN>	0	Stop and clear
	1	Count up

Note: The 1, 4 and 5 of TB1RUN are read as “1” value.

Figure 3.13.3 Register for TMRB (1)

TMRB0 Mode Register

	7	6	5	4	3	2	1	0
TB0MOD (1182H) Bit symbol	–	–	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
Read/Write	R/W		W*	R/W				
After Reset	0	0	1	0	0	0	0	0
Function	Always write "0".		Software capture control 0: Execute 1: Undefined	Capture timing 00: Disable INT6 occurs at rising edge 01: TB0IN0 ↑ INT6 occurs at rising edge 10: TB0IN0 ↑ TB0IN0 ↓ INT6 occurs at falling edge 11: TA1OUT ↑ TA1OUT ↓ INT6 occurs at rising edge			Control Up counter 0: Disable 1: Enable	TMRB0 source clock 00: TB0IN0 input 01: φT1 10: φT4 11: φT16	

Prohibit read-modify-write

TMRB0 source clock

<TB0CLK1:0>	00	TB0IN0 pin input
	01	φT1
	10	φT4
	11	φT16

Control clearing for up counter (UC10)

<TB0CLE>	0	Disable
	1	Enable clearing by match with TB0RG1

Capture/interrupt timing

		Capture control	INT6 control
<TB0CPM1:0>	00	Disable	INT6 occurs at the rising edge of TB0IN0
	01	Capture to TB0CP0H/L at rising edge of TB0IN0	
	10	Capture to TB0CP0H/L at rising edge of TB0IN0 Capture to TB0CP1H/L at falling edge of TB0IN0	INT6 occurs at the rising edge of TB0IN0
	11	Capture to TB0CP0H/L at rising edge of TA1OUT Capture to TB0CP1H/L at falling edge of TA1OUT	INT6 occurs at the rising edge of TB0IN0

Software capture

<TB0CP0I>	0	The value of up counter is captured to TB0CP0H/L
	1	Undefined

Figure 3.13.4 Register for TMRB (2)

TMRB1 Mode Register

		7	6	5	4	3	2	1	0
TB1MOD (1192H)	Bit symbol	–	–	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
	Read/Write	R/W		W*	R/W				
	After Reset	0	0	1	0	0	0	0	0
Prohibit read-modify-write	Function	Always write "0".		Software capture control 0: Execute 1: Undefined	Capture timing 00: Disable INT7 occurs at rising edge 01: TB1IN0 ↑ INT7 occurs at rising edge 10: TB1IN0 ↑ TB1IN0 ↓ INT7 occurs at falling edge 11: TA3OUT ↑ TA3OUT ↓ INT7 occurs at rising edge		Control Up counter 0: Disable 1: Enable	TMRB1 source clock 00: TB1IN0 input 01: φT1 10: φT4 11: φT16	

TMRB1 source clock

<TB1CLK1:0>	00	TB1IN0 pin input
	01	φT1
	10	φT4
	11	φT16

Control clearing for up counter (UC12)

<TB1CLE>	0	Disable
	1	Enable clearing by match with TB1RG1H/L

Capture/interrupt timing

		Capture control	INT7 control
<TB1CPM1:0>	00	Disable	INT7 occurs at the rising edge of TB1IN0
	01	Capture to TB1CP0H/L at rising edge of TB1IN0	
	10	Capture to TB1CP0H/L at rising edge of TB1IN0 Capture to TB1CP1H/L at falling edge of TB1IN0	INT7 occurs at the rising edge of TB1IN0
	11	Capture to TB1CP0H/L at rising edge of TA3OUT Capture to TB1CP1H/L at falling edge of TA3OUT	INT7 occurs at the rising edge of TB1IN0

Software capture

<TB1CP0I>	0	The value of up counter is captured to TB1CP0H/L
	1	Undefined (Note)

Figure 3.13.5 Register for TMRB (3)

TMRB0 Flip-Flop Control Register

TB0FFCR (1183H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
Read/Write	W*		R/W				W*	
After Reset	1	1	0	0	0	0	1	1
Function	Always write “11”		TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB0FF0 00: Invert 01: Set 10: Clear 11: Undefined	
Prohibit read-modify-write	*Always read as “11”.		When capture UC10 to TB0CP1H/L	When capture UC10 to TB0CP0H/L	When UC10 matches with TB0RG1H/L	When UC10 matches with TB0RG0H/L	*Always read as “11”.	

Timer flip-flop control(TB0FF0)

<TB0FF0C1:0>	00	Invert
	01	Set to “11”
	10	Clear to “00”
	11	Undefined (Always read as “11”)

TB0FF0 control
Inverted when UC10 value matches the valued in TB0RG0H/L

<TB0E0T1>	0	Disable trigger
	1	Enable trigger

TB0FF0 control
Inverted when UC10 value matches the valued in TB0RG1H/L

<TB0E1T1>	0	Disable trigger
	1	Enable trigger

TB0FF0 control
Inverted when UC10 value is captured into TB0CP0H/L

<TB0C0T1>	0	Disable trigger
	1	Enable trigger

TB0FF0 control
Inverted when UC10 value is captured into TB0CP1H/L

<TB0C1T1>	0	Disable trigger
	1	Enable trigger

Figure 3.13.6 Register for TMRB (4)

TMRB1 Flip-Flop Control Register

	7	6	5	4	3	2	1	0
TB1FFCR (1193H) Bit symbol	–	–	TB1C1T1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
Read/Write	W*		R/W				W*	
After Reset	1	1	0	0	0	0	1	1
Function	Always write “11”		TB1FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB1FF0 00: Invert 01: Set 10: Clear 11: Don’t care	
Prohibit read-modify-write	*Always read as “11”.		When capture UC12 to TB1CP1H/L	When capture UC12 to TB1CP0H/L	When UC12 matches with TB1RG1H/L	When UC12 matches with TB1RG0H/L	*Always read as “11”.	

Timer flip-flop control(TB1FF0)

<TB1FF0C1:0>	00	Invert
	01	Set to “11”
	10	Clear to “00”
	11	Don’t care

TB1FF0 control
Inverted when UC12 value matches the valued in TB1RG0H/L

<TB1E0T1>	0	Disable trigger
	1	Enable trigger

TB1FF0 control
Inverted when UC12 value matches the valued in TB1RG1H/L

<TB1E1T1>	0	Disable trigger
	1	Enable trigger

TB1FF0 control
Inverted when UC12 value is captured into TB1CP0H/L

<TB1C0T1>	0	Disable trigger
	1	Enable trigger

TB1FF0 control
Inverted when UC12 value is captured into TB1CP1H/L

<TB1C1T1>	0	Disable trigger
	1	Enable trigger

Figure 3.13.7 Register for TMRB (5)

		7	6	5	4	3	2	1	0
TB0RG0L (1188H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB0RG0H (1189H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB0RG1L (118AH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB0RG1H (118BH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB1RG0L (1198H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB1RG0H (1199H)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB1RG1L (119AH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
TB1RG1H (119BH)	bit Symbol	–	–	–	–	–	–	–	–
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0

Note: All registers are prohibited to execute read-modify-write instruction.

Figure 3.13.8 Register for TMRB (6)

3.13.4 Operation in Each Mode

(1) 16 bit timer mode

Generating interrupts at fixed intervals

In this example, the interrupt INTTB01 is set to be generated at fixed intervals. The interval time is set in the timer register TB0RG1H/L.

		7	6	5	4	3	2	1	0	
TB0RUN	←	–	0	X	X	–	–	X	0	Stop TMRB0
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01and set interrupt level 4.
										Disable INTTB00
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable the trigger
TB0MOD	←	0	0	1	0	0	1	*	*	Select internal clock for input and
						(** = 01, 10, 11)				disable the capture function.
TB0RG1	←	*	*	*	*	*	*	*	*	Set the interval time
		*	*	*	*	*	*	*	*	(16 bits).
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

(2) 16 bit event counter mode

In 16 bit timer mode as described in above, the timer can be used as an event counter by selecting the external clock (TB0IN0 pin input) as the input clock. Up counter (UC10) counts up at the rising edge of TB0IN0 input. To read the value of the counter, first perform “software capture” once and read the captured value.

		7	6	5	4	3	2	1	0	
TB0RUN	←	–	0	X	X	–	–	X	0	Stop TMRB0
PPCR	←	X	X	–	1	–	–	–	X	Set PP4 to input mode for TB0IN0
PPFC	←	–	–	–	1	–	–	–	X	
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01 and sets interrupt level 4
										Disable INTTB00
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable trigger
TB0MOD	←	0	0	1	0	0	1	0	0	Select TB0IN0 as the input clock
TB0RG1	←	*	*	*	*	*	*	*	*	Set the number of counts
		*	*	*	*	*	*	*	*	(16 bit)
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0

X: Don't care, –: No change

When used as an event counter, set the prescaler in RUN mode.
(TB0RUN <TB0PRUN> = “1”)

(3) 16-bit programmable pulse generation (PPG) output mode

Square wave pulses can be generated at any frequency and duty ratio. The output pulse may be either low active or high active.

The PPG mode is obtained by inversion of the timer flip-flop TB0FF0 that is to be enabled by the match of the up counter UC10 with timer register TB0RG0H/L or TB0RG1H/L and to be output to TB0OUT0. In this mode the following conditions must be satisfied.

(Value set in TB0RG0) < (Value set in TB0RG1)

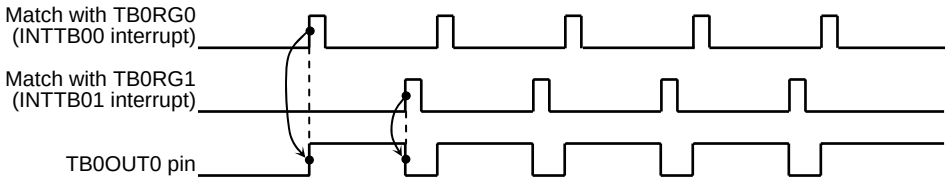


Figure 3.13.9 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0H/L double buffer is enabled in this mode, the value of register buffer 10 will be shifted into TB0RG0H/L at match with TB0RG1H/L. This feature facilitates the handling of low-duty waves.

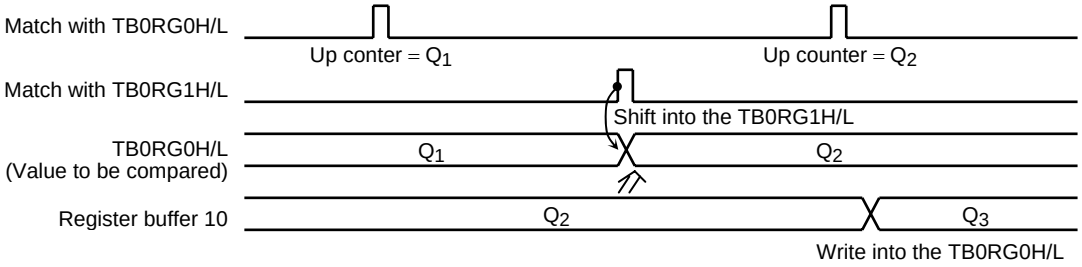


Figure 3.13.10 Operation of double buffer

Note: The values that can be set in TBxRGx range from 0001h to 0000h (equivalent to 10000h). If the maximum value 000h is set, the match-detect signal goes active when the up-counter overflows.

The following block diagram illustrates this mode.

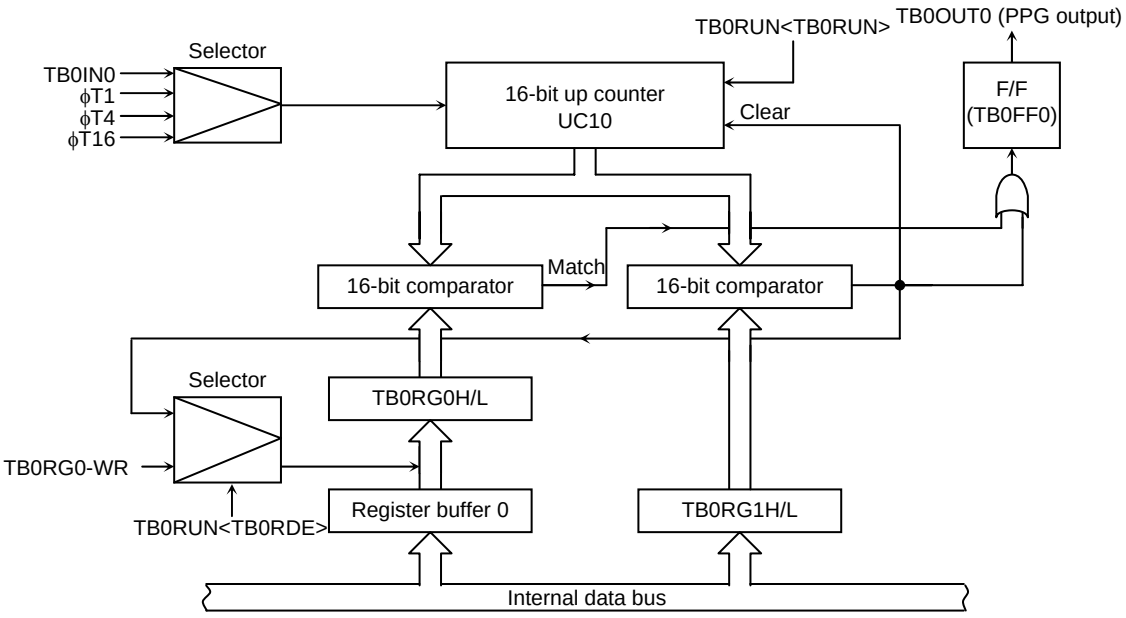


Figure 3.13.11 Block Diagram of 16-Bit Mode

The following example shows how to set 16-bit PPG output mode:

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	–	X	0	Disable the TB0RG0 double buffer and stop TMRB0.
TB0RG0	←	*	*	*	*	*	*	*	*	Set the duty ratio
		*	*	*	*	*	*	*	*	(16 bit)
TB0RG1	←	*	*	*	*	*	*	*	*	Set the frequency
		*	*	*	*	*	*	*	*	(16 bit)
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0H/L double buffer. (The duty and frequency are changed on an INTTB01 interrupt.)
TB0FFCR	←	X	X	0	0	1	1	1	0	Set the mode to invert TB0FF0 at the match with TB0RG0H/L/TB0RG1H/L. Set TB0FF0 to 0.
TB0MOD	←	0	0	1	0	0	1	*	*	Select the internal clock as the input clock and disable the capture function. (** = 01, 10, 11)
PPFC	←	–	1	–	–	–	–	–	X	Set PP6 to function as TB0OUT0
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

(4) Application examples of capture function

Used capture function, they can be applied in many ways, for example;

1. One-shot pulse output from external trigger pulse
2. Frequency measurement
3. Pulse width measurement

1. One-shot pulse output from external trigger pulse

Set the up counter UC10 in free-running mode with the internal input clock, input the external trigger pulse from TB0IN0 pin, and load the value of up counter into capture register TB0CP0H/L at the rising edge of the TB0IN0 pin.

When the interrupt INT6 is generated at the rising edge of TB0IN0 input, set the TB0CP0H/L value (c) plus a delay time (d) to TB0RG0H/L (=c+d), and set the above set value (c+d) plus a one-shot pulse width (p) to TB0RG1H/L (=c+d+p).

The TB0FFCR<TB0E1T1, TB0E0T1> register should be set “11” and that the TB0FF0 inversion is enabled only when the up counter value matches TB0RG0H/L or TB0RG1H/L. When interrupt INTTB01 occurs, this inversion will be disabled after one-shot pulse is output.

The (c), (d) and (p) correspond to c, d, and p in the Figure 3.13.12.

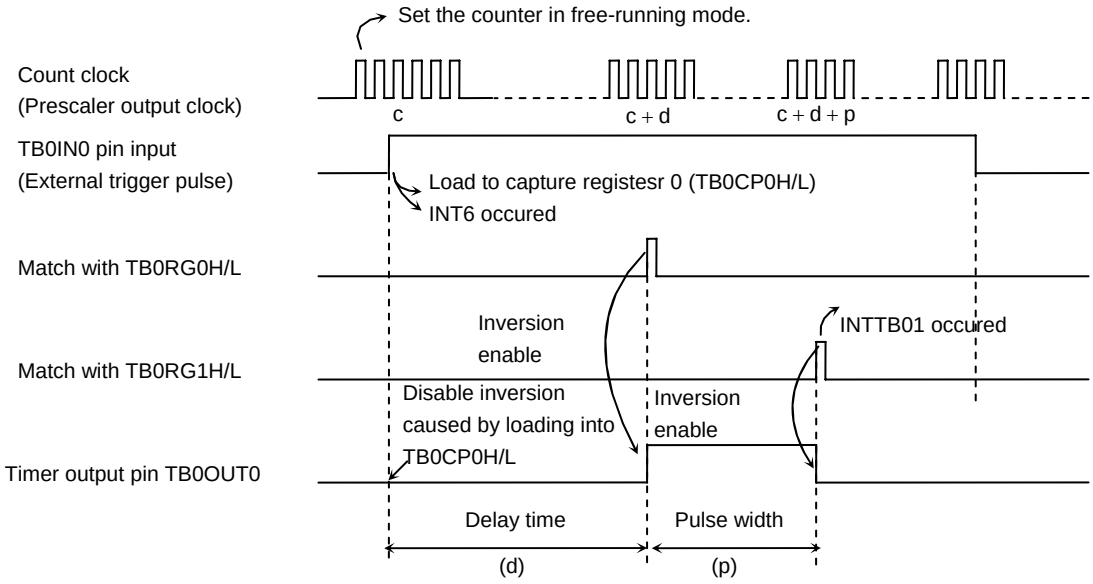


Figure 3.13.12 One-shot Pulse Output (with delay)

Example: To output 2ms one-shot pulse with 3ms delay to the external trigger pulse to TB0IN0pin

*Clock state

System clock : f_{SYS}
Prescaler clock : $f_{SYS}/4$

Main setting

```
TB0MOD   ← X X 1 0 1 0 0 1   Free-running
                             Count with φT1
                             Load to TB0CP0H/L at the rising edge of TB0IN0
TB0FFCR  ← X X 0 0 0 0 1 0   Clear TB0FF0 to "0"
                             Disable TB0FF0 inversion
PPFC     ← – 1 – – – – – X   Select PP6 as TB0OUT0 pin (port setting)

INTE56   ← X 1 0 0 X – – –   Enable INT6
INTETB0  ← X 0 0 0 X 0 0 0   Disable INTTB00, INTTB01
TB0RUN   ← – 0 X X – 1 X 1   Start TMRB0
```

Setting in INT6 routine

```
TB0RG0   ← TB0CP0 + 3ms/φT1
TB0RG1   ← TB0RG0 + 2ms/φT1
TB0FFCR  ← X X – – 1 1 – –   Enable TB0FF0 inversion when the up counter value
                             matches TB0RG0H/L or TB0RG1H/L
INTETB0  ← X 1 0 0 X 0 0 0   Enable INTTB01
```

Setting in INTTB01 routine

```
TB0FFCR  ← X X – – 0 0 – –   Disable TB0FF0 inversion when the up counter value
                             matches TB0RG0H/L or TB0RG1H/L
INTETB0  ← X 0 0 0 X 0 0 0   Disable INTTB01
```

X: Don't care, –: No change

When delay time is unnecessary, invert timer flip-flop TB0FF0 when the up counter value is loaded into capture register (TB0CP0H/L), and set the TB0CP0H/L value (c) plus the one –shot pulse width (p) to TB0RG1H/L when the interrupt INT6 occurs. The TB0FF0 inversion should be enabled when the up counter (UC10) value matched TB0RG1H/L, and disabled when generating the interrupt INTTB01.

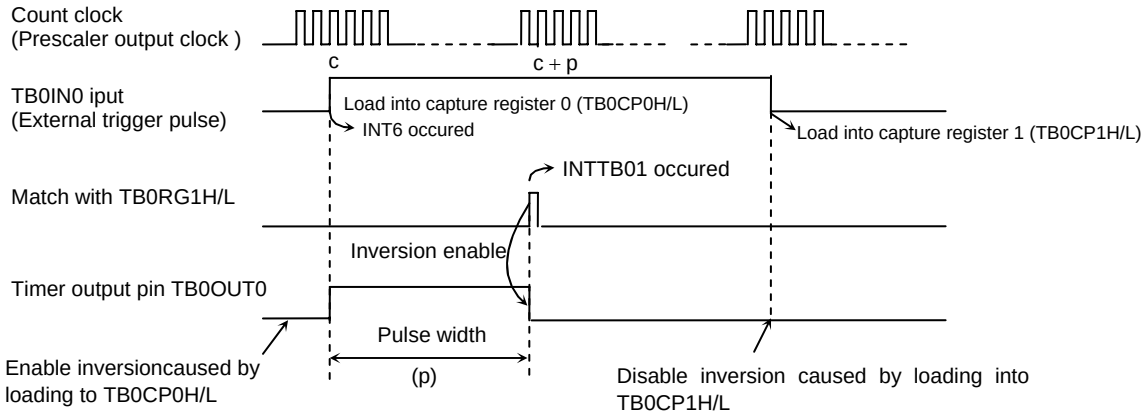


Figure 3.13.13 One-shot Pulse Output (without delay)

2. Frequency measurement

The frequency of the external clock can be measured in this mode. The clock is input through the TB0IN0 pin, and its frequency is measured by the 8 bit timers TMRA01 and the 16 bit timer/event counter (TMRB0).

The TB0IN0 pin input should be selected for the input clock of TMRB0. Set to TB0MOD<TB0CPM1:0>="11". The value of the up counter is loaded into the capture register TB0CP0H/L at the rising edge of the timer flip-flop TA1FF of 8bit timers (TMRA01), and TB0CP1H/L at its falling edge.

The frequency is calculated by the difference between the loaded values in TB0CP0H/L and TB0CP1H/L when the interrupt (INTTA0 or INTTA1) is generated by either 8 bit timer.

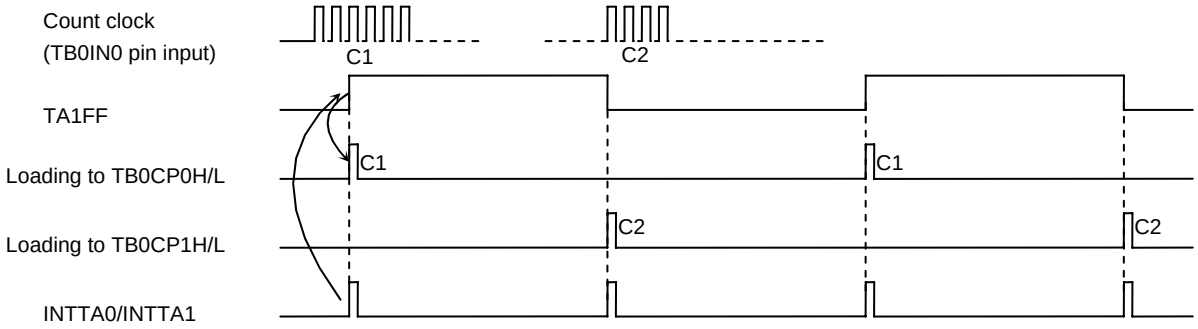


Figure 3.13.14 Frequency Measurement

For example, if the value for the level 1 width of TA1FF of the 8 bit timer is set to 0.5[s] and the difference between TB0CP0H/L and TB0CP1H/L is 100, the frequency will be 100/0.5[s] =200[Hz].

Note: The frequency in this examole is calculated with 50% duty.

3. Pulse width measurement

This mode allows measuring the H level width of an external pulse. While keeping the 16 bit timer/event counter counting (free-running) with the internal clock input, the external pulse is input through the TB0IN0 pin. Then the capture function is used to load the UC10 values into TB0CP0H/L and TB0CP1H/L at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT6 occurs at the falling edge of TB0IN0.

The pulse width is obtained from the difference between the values of TB0CP0H/L and TB0CP1H/L and the internal clock cycle.

For example, if the internal clock is 0.8[us] and the difference between TB0CP0H/L and TB0CP1H/L is 100, the pulse width will be $100 \times 0.8[us] = 80us$

Additionally, the pulse width which is over the UC10 maximum count time specified by the clock source can be measured by changing software.

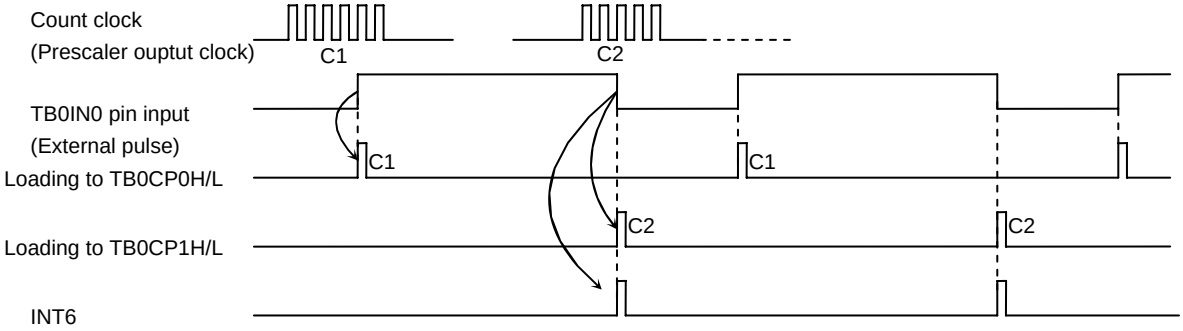


Figure 3.13.15 Pulse Width Measurement

Note: Only in this pulse width measuring mode(TB0MOD<TB0CPM1:0> "10"), external interrupt INT6 occurs at the falling edge of TB0IN0 pin input. In other modes, it occurs at the rising edge.

The width of L level can be measured by multiplying the difference between the first C1 and the second C0 at the second INT6 interrupt and the internal clock cycle together.

3.14 Serial Channels (SIO)

TMP92CZ26A includes 1 serial I/O channel (SIO0). For both channels either UART mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission) can be selected. And, SIO0 includes data modulator that supports the IrDA 1.0 infrared data communication specification.

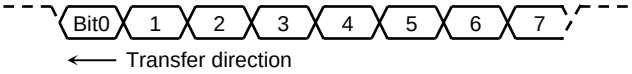
- I/O interface mode
  - Mode 0: For transmitting and receiving I/O data using the synchronizing signal SCLK for extending
- UART mode
  - Mode 1: 7-bit data
  - Mode 2: 8-bit data
  - Mode 3: 9-bit data

In mode 1 and mode 2, a parity bit can be added. Mode 3 has a wakeup function for making the master controller start slave controllers via a serial link (A multi-controller system).

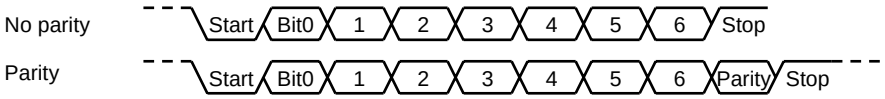
Figure 3.14.1 is block diagrams for each channel.

SIO0 is compounded mainly prescaler, serial clock generation circuit, receiving buffer and control circuit, transmission buffer and control circuit.

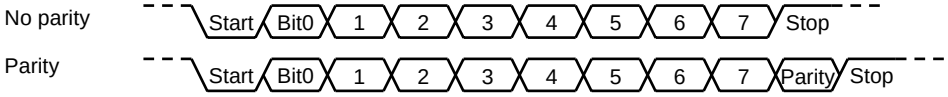
- Mode 0 (I/O interface mode)



- Mode 1 (7-bit UART mode)



- Mode 2 (8-bit UART mode)



- Mode 3 (9-bit UART mode)

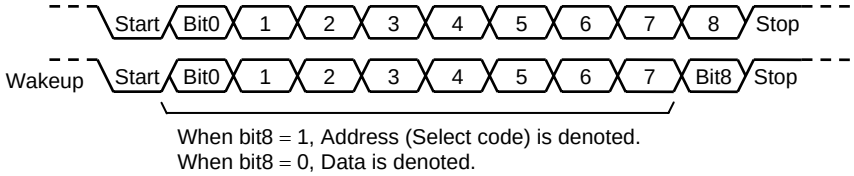


Figure 3.14.1 Data Formats

3.14.1 Block Diagram

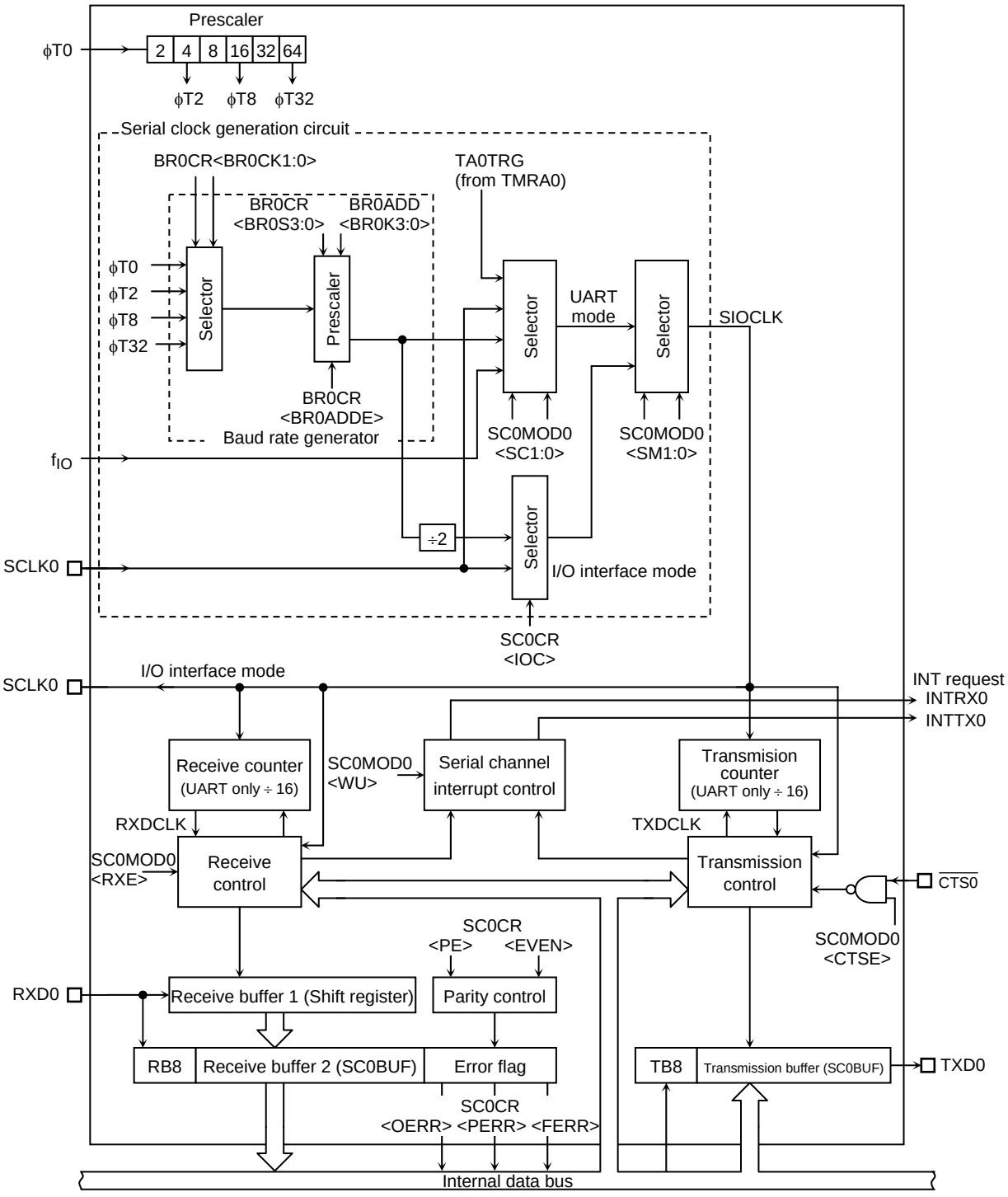


Figure 3.14.2 Block Diagram

3.14.2 Operation of Each Circuit

(1) Prescaler

There is a 6-bit prescaler for generating a clock to SIO0. The prescaler can be run by selecting the baud rate generator as the serial transfer clock.

Table 3.14.1 shows prescaler clock resolution into the baud rate generator.

Table 3.14.1 Prescaler Clock Resolution to Baud Rate Generator

-	Clock gear SYSCR1 <GEAR2:0>	-	Clock Resolution			
			ϕT0	ϕT2	ϕT8	ϕT32
fc	000(1/1)	1/4	f_{SYS}/4	f_{SYS}/16	f_{SYS}/64	f_{SYS}/256
	001(1/2)		f_{SYS}/8	f_{SYS}/32	f_{SYS}/128	f_{SYS}/512
	010(1/4)		f_{SYS}/16	f_{SYS}/64	f_{SYS}/256	f_{SYS}/1024
	011(1/8)		f_{SYS}/32	f_{SYS}/128	f_{SYS}/512	f_{SYS}/2048
	100(1/16)		f_{SYS}/64	f_{SYS}/256	f_{SYS}/1024	f_{SYS}/4096

XXX:Don't care

The baud rate generator selects between 4-clock inputs: ϕT0, ϕT2, ϕT8, and ϕT32 among the prescaler outputs.

(2) Baud rate generator

The baud rate generator is the circuit which generates transmission/receiving clock and determines the transfer rate of the serial channels.

The input clock to the baud rate generator, ϕT0, ϕT2, ϕT8 or ϕT32, is generated by the 6-bit prescaler which is shared by the timers. One of these input clocks is selected using the BR0CR<BR0CK1:0> field in the baud rate generator control register.

The baud rate generator includes a frequency divider, which divides the frequency by 1 or N + (16 − K)/16 to 16 values, determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3:0> and BR0ADD<BR0K3:0>.

- In UART mode

When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3:0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CK<BR0S3:0>. (N = 1, 2, 3 ... 16)

When BR0CR<BR0ADDE> = 1

The N + (16 − K)/16 division function is enabled. The baud rate generator divides the selected prescaler clock by N + (16 − K)/16 using the value of N set in BR0CR<BR0S3:0> (N = 2, 3 ... 15) and the value of K set in BR0ADD<BR0K3:0> (K = 1, 2, 3 ... 15)

Note: If N = 1 or N = 16, the N + (16 − K)/16 division function is disabled. Clear BR0CR<BR0ADDE> to 0.

- In I/O interface mode

The N + (16 − K)/16 division function is not available in I/O interface mode. Clear BR0CR<BR0ADDE> to 0 before dividing by N.

The method for calculating the transfer rate when the baud rate generator is used is explained below.

- In UART mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

- In I/O interface mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 2$$

- Integer divider (N divider)

  For example, when the source clock frequency (f_c) is 19.6608 MHz, the input clock is ϕT2, the frequency divider N (BR0CR<BR0S3:0>) = 8, and BR0CR<BR0ADDE> = 0, the baud rate in UART Mode is as follows:

*Clock state { System clock : 1/1; Prescaler clock : 1/2

$$\text{Baud Rate} = \frac{f_C/16}{8} \div 16 = 19.6608106 \times 10^6 \div 16 \div 8 \div 16 = 9600 \text{ (bps)}$$

Note: The N + (16 – K) / 16 division function is disabled and setting BR0ADD <BR0K3:0> is invalid.

- N+(16-K)/16 divider (UART Mode only)

  Accordingly, when the source clock frequency (fc) = 15.9744 MHz, the input clock is ϕT2, the frequency divider N (BR0CR<BR0S3:0>) = 6, K (BR0ADD<BR0K3:0>) = 8, and BR0CR <BR0ADDE> = 1, the baud rate in UART Mode is as follows:

*Clock state { System clock : 1/1; Prescaler clock : 1/2

$$\text{Baud Rate} = \left(\frac{f_C/16}{6 + \frac{(16-8)}{16}} \right) \div 16 = 15.9744 \times 10^6 \div 16 \div \left(6 + \frac{8}{16}\right) \div 16 = 9600 \text{ (bps)}$$

Table 3.14.2 show examples of UART Mode transfer rates.

Additionally, the external clock input is available in the serial clock. (Serial Channel 0). The method for calculating the baud rate is explained below:

- In UART Mode

  Baud rate = external clock input frequency ÷ 16

  It is necessary to satisfy (external clock input cycle) ≥ $4/f_{SYS}$

- In I/O Interface Mode

  Baud rate = external clock input frequency

  It is necessary to satisfy (external clock input cycle) ≥ $16/f_{SYS}$

Table 3.14.2 Transfer Rate Selection

Unit (kbps)

(When baud rate generator is used and BR0CR<BR0ADDE> = 0)

f_{SYS} [MHz]	Input Clock / Frequency Divider N	φT0 (f_{SYS}/4)	φT2 (f_{SYS}/16)	φT8 (f_{SYS}/64)	φT32 (f_{SYS}/256)
7.3728	1	115.200	28.800	7.200	1.800
↑	3	38.400	9.600	2.400	0.600
↑	6	19.200	4.800	1.200	0.300
↑	A	11.520	2.880	0.720	0.180
↑	C	9.600	2.400	0.600	0.150
↑	F	7.680	1.920	0.480	0.120
9.8304	1	153.600	38.400	9.600	2.400
↑	2	76.800	19.200	4.800	1.200
↑	4	38.400	9.600	2.400	0.600
↑	5	30.720	7.680	1.920	0.480
↑	8	19.200	4.800	1.200	0.300
↑	0	9.600	2.400	0.600	0.150
44.2368	6	115.20	28.800	7.200	1.800
↑	9	76.800	19.200	4.800	1.200
58.9824	2	460.800	115.200	28.800	7.200
↑	3	307.200	76.800	19.200	4.800
↑	5	184.320	46.080	11.520	2.880
↑	6	153.600	38.400	9.600	2.400
↑	8	115.200	28.800	7.200	1.800
↑	C	76.800	19.200	4.800	1.200
↑	F	61.440	15.360	3.840	0.960
73.728	1	1152.000	288.000	72.000	18.000
↑	3	384.000	96.000	24.000	6.000
↑	6	192.000	48.000	12.000	3.000
↑	A	115.200	28.800	7.200	1.800
↑	C	96.000	24.000	6.000	1.500
↑	F	76.800	19.200	4.800	1.200

Note: Transfer rates in I/O interface mode are eight times faster than the values given above.

Timer out clock (TA0TRG) can be used for source clock of UART mode only.

Calculation method the frequency of TA0TRG

Frequency of TA0TRG = Baud rate × 16

Note: In case of I/O interface mode, prohibit to use TA0TRG for source clock.

(3) Serial clock generation circuit

This circuit generates the basic clock for transmitting and receiving data.

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = 0, the basic clock is generated by dividing the output of the baud rate generator by 2, as described previously.
In SCLK Input Mode with the setting SC0CR<IOC> = 1, the rising edge or falling edge will be detected according to the setting of the SC0CR<SCLKS> register to generate the basic clock.

- In UART Mode

The SC0MOD0 <SC1:0> setting determines whether the baud rate generator clock, the internal clock f_{IO}, the match detect signal from timer TMRA0 or the external clock (SCLK0) is used to generate the basic clock SIOCLK.

(4) Receiving counter

The receiving counter is a 4-bit binary counter used in UART Mode, which counts up the pulses of the SIOCLK clock. It takes 16 SIOCLK pulses to receive 1 bit of data; each data bit is sampled three times - on the 7th, 8th and 9th clock cycles.
The value of the data bit is determined from these three samples using the majority rule.
For example, if the data bit is sampled respectively as 1, 0 and 1 on 7th, 8th and 9th clock cycles, the received data bit is taken to be 1. A data bit sampled as 0, 0 and 1 is taken to be 0.

(5) Receiving control

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = 0, the RXD0 signal is sampled on the rising or falling edge of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.
In SCLK Input Mode with the setting SC0CR<IOC> = 1, the RXD0 signal is sampled on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting

- In UART Mode

The receiving control block has a circuit, which detects a start bit using the majority rule. Received bits are sampled three times; when two or more out of three samples are 0, the bit is recognized as the start bit and the receiving operation commences.
The values of the data bits that are received are also determined using the majority rule.

(6) The Receiving Buffers

To prevent Overrun errors, the Receiving Buffers are arranged in a double-buffer structure.

Received data is stored one bit at a time in Receiving Buffer 1 (which is a shift register). When 7 or 8 bits of data have been stored in Receiving Buffer 1, the stored data is transferred to Receiving Buffer 2 (SC0BUF); these causes an INTRX0 interrupt to be generated. The CPU only reads Receiving Buffer 2 (SC0BUF). Even before the CPU reads receiving Buffer 2 (SC0BUF), the received data can be stored in Receiving Buffer 1. However, unless Receiving Buffer 2 (SC0BUF) is read before all bits of the next data are received by Receiving Buffer 1, an overrun error occurs. If an Overrun error occurs, the contents of Receiving Buffer 1 will be lost, although the contents of Receiving Buffer 2 and SC0CR<RB8> will be preserved.

SC0CR<RB8> is used to store either the parity bit - added in 8-Bit UART Mode - or the most significant bit (MSB) - in 9-Bit UART Mode.

In 9-Bit UART Mode the wake-up function for the slave controller is enabled by setting SC0MOD0<WU> to 1; in this mode INTRX0 interrupts occur only when the value of SC0CR<RB8> is 1.

SIO interrupt mode is selectable by the register SIMC.

Note1: The double buffer structure does not support SC0CR<RV08>.

Note2: If the CPU reads receive buffer 2 while data is being transferred from receive buffer 1 to receive buffer 2, the data may not be read properly. To avoid this situation, a read of receive buffer 2 should be triggered by a receive interrupt.

(7) Notes for Using Receive Interrupts

- Receive interrupts can be detected either in level or edge mode. For details, see the description of the SIO/SEI receive interrupt mode select register SIMC in the section on interrupts.
- When receive interrupts are set to level mode, once an interrupt occurs, the same interrupt will occur repeatedly even after control has jumped to the interrupt routine unless interrupts are disabled.

(8) Transmission counters

The transmission counter is a 4-bit binary counter which is used in UART Mode and which, like the receiving counter, counts the SIOCLK clock pulses; a TXDCLK pulse is generated every 16 SIOCLK clock pulses.

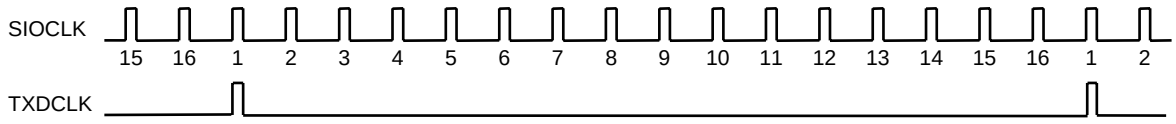


Figure 3.14.3 Generation of the transmission clock

(8) Transmission controller

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = 0, the data in the Transmission Buffer is output one bit at a time to the TXD0 pin on the rising edge or falling of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.
In SCLK Input Mode with the setting SC0CR<IOC> = 1, the data in the Transmission Buffer is output one bit at a time on the TXD0 pin on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

- In UART Mode

When transmission data sent from the CPU is written to the Transmission Buffer, transmission starts on the rising edge of the next TXDCLK.

Handshake function

Serial Channels 0 has a $\overline{CTS0}$ pin. Use of this pin allows data can be sent in units of one frame; thus, Overrun errors can be avoided. The handshake functions is enabled or disabled by the SC0MOD <CTSE> setting.

When the $\overline{CTS0}$ pin goes High on completion of the current data send, data transmission is halted until the $\overline{CTS0}$ pin goes Low again. However, the INTTX0 Interrupt is generated, it requests the next data send to the CPU. The next data is written in the Transmission Buffer and data sending is halted.

Though there is no $\overline{RTS}$ pin, a handshake function can be easily configured by setting any port assigned to be the $\overline{RTS}$ function. The $\overline{RTS}$ should be output "High" to request send data halt after data receive is completed by software in the RXD interrupt routine.

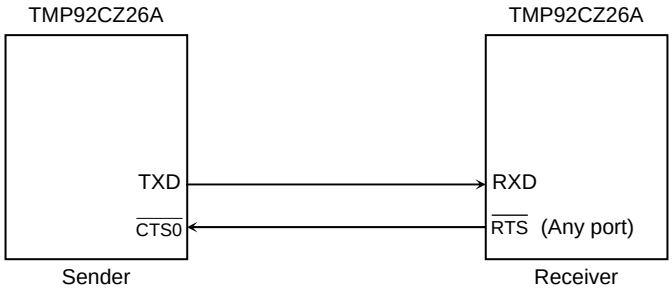
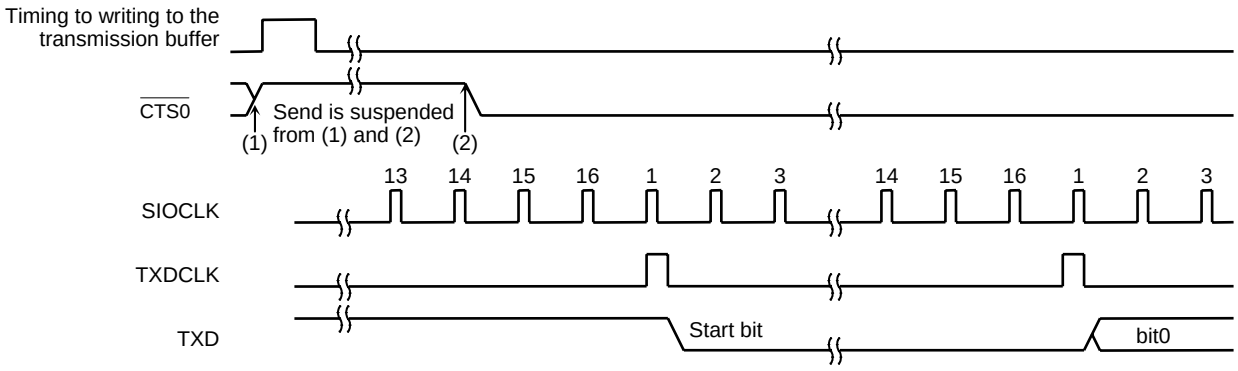


Figure 3.14.4 Handshake function



Note 1: (1) If the $\overline{CTS0}$ signal goes High during transmission, no more data will be sent after completion of the current transmission.

Note 2: (2) Transmission starts on the first falling edge of the TXDCLK clock after the $\overline{CTS0}$ signal has fallen.

Figure 3.14.5 $\overline{CTS0}$ (Clear to send) Timing

(9) Transmission buffer

The transmission buffer (SC0BUF) shifts out and sends the transmission data written from the CPU form the least significant bit (LSB) in order. When all the bits are shifted out, the transmission buffer becomes empty and generates an INTTX0 interrupt.

(10) Parity control circuit

When SC0CR<PE> in the serial channel control register is set to 1, it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART mode or 8-bit UART mode. The SC0CR<EVEN> field in the serial channel control register allows either even or odd parity to be selected.

In the case of transmission, parity is automatically generated when data is written to the transmission buffer SC0BUF. The data is transmitted after the parity bit has been stored in SC0BUF<TB7> in 7-bit UART mode or in SC0MOD0<TB8> in 8-bit UART mode. SC0CR<PE> and SC0CR<EVEN> must be set before the transmission data is written to the transmission buffer.

In the case of receiving, data is shifted into receiving buffer 1, and the parity is added after the data has been transferred to receiving buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> in 7-bit UART mode or with SC0CR<RB8> in 8-bit UART mode. If they are not equal, a parity error is generated and the SC0CR<PERR> flag is set.

(11) Error flags

Three error flags are provided to increase the reliability of data reception.

1. Overrun error <OERR>

If all the bits of the next data item have been received in receiving buffer 1 while valid data still remains stored in receiving buffer 2 (SC0BUF), an overrun error is generated.

The below is a recommended flow when the overrun error is generated.

(INTRX interrupt routine)

1) Read receiving buffer

2) Read error flag

3) If <OERR> = 1

then

a) Set to disable receiving (Write 0 to SC0MOD0<RXE>)

b) Wait to terminate current frame

c) Read receiving buffer

d) Read error flag

e) Set to enable receiving (Write 1 to SC0MOD0<RXE>)

f) Request to transmit again

4) Others

Note: Overrun errors are generated only with regard to receive buffer 2 (SC0BUF). Thus, if SC0CR<RB8> is not read, no overrun error will occur.

2. Parity error <PERR>

The parity generated for the data shifted into receiving buffer 2 (SC0BUF) is compared with the parity bit received via the RXD pin. If they are not equal, a parity error is generated.

Note: The parity error flag is cleared every time it is read. However, if a parity error is detected w¥twice in succession and the parity error flag is read between the two parity errors, it may seem as if the flag had not been cleared. To avoid this situation, a read of the parity error flag should be riggered by a receive interrupt.

3. Framing error <FERR>

The stop bit for the received data is sampled three times around the center. If the majority of the samples are 0, a Framing error is generated.

(12) Timing generation

a. In UART Mode

Receiving

Mode	9-Bit (Note)	8-Bit + Parity (Note)	8-Bit, 7-Bit + Parity, 7-Bit
Interrupt timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit
Framing error timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error timing	—	Center of last bit (parity bit)	Center of stop bit
Overrun error timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit

Note1: In 9-Bit and 8-Bit + Parity Modes, interrupts coincide with the ninth bit pulse.
Thus, when servicing the interrupt, it is necessary to wait for a 1-bit period (to allow the stop bit to be transferred) to allow checking for a framing error.

Note2: The higher the transfer rate, the later than the middle receive interrupts and errors occur.

Transmitting

Mode	9-Bit	8-Bit + Parity	8-Bit, 7-Bit + Parity, 7-Bit
Interrupt timing	Just before stop bit is transmitted	Just before stop bit is transmitted	Just before stop bit is transmitted

b. I/O interface

Transmission Interrupt timing	SCLK Output Mode	Immediately after last bit. (See Figure 3.14.13.)
	SCLK Input Mode	Immediately after rise of last SCLK signal Rising Mode, or immediately after fall in Falling Mode. (See Figure 3.14.14.)
Receiving Interrupt timing	SCLK Output Mode	Timing used to transfer received to data Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See Figure 3.14.15.)
	SCLK Input Mode	Timing used to transfer received data to Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See Figure 3.14.16.)

3.14.3 SFR

SC0MOD0 (1202H)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Transfer data bit 8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wake up function 0: disable 1: enable	Serial Transmission Mode 00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{IO} 11: External clcok (SCLK0 input)	

Serial transmission clock source (UART)

00	TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{IO}
11	External clock (SCLK0 input)

Note: The clock selection for the I/O interface mode is controlled by the serial bontrol register (SC0CR).

Serial Transmission Mode

00	I/O Interface Mode	
01	UART mode	7-bit mode
10		8-bit mode
11		9-bit mode

Wake-up function

	9-Bit UART	Other Modes
0	Interrupt generated whenever data is received	Don't care
1	Interrupt generated only when RB8 = 1	

Receiving Function

0	Receive disabled
1	Receive enabled

Handshake function (/CTS pin)

0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Figure 3.14.6 Serial Mode Control Register (channel 0, SC0MOD0)

	7	6	5	4	3	2	1	0
bit Symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when read)			R/W	
After Reset	Undefined	0	0	0	0	0	0	0
Function	Received data bit 8	Parity 0: odd 1: even	Parity addition 0: disable 1: enable	1: error			0: SCLK0 ↑ 1: SCLK0 ↓	0: baud rate generator 1: SCLK0 pin input
				Overrun	Parity	Framing		

SC0CR (1201H)

Prohibit to Read modify Write

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Edge selection for SCLK pin (Input / Output Mode)

0	Transmits and receivers data on rising edge of SCLK0.
1	Transmits and receivers data on falling edge SCLK0.

Framing Error flag
Parity Error flag
Overrun Error flag
Cleared to 0 when read

Parity addition enables

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data 8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.14.7 Serial Control Register (channel 0, SC0CR)

BR0CR (1203H)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Always write “0”	+(16−K)/16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency setting			

+(16−K)/16 division enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

00	Internal clock ϕT0
01	Internal clock ϕT2
10	Internal clock ϕT8
11	Internal clock ϕT32

BR0ADD (1204H)

	7	6	5	4	3	2	1	0
bit Symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write					R/W			
After reset					0	0	0	0
Function					Sets frequency divisor “K” (divided by N + (16-K) / 16)			

Sets baud rate generator frequency divisor

BR0ADD <BR0K3:0> \ BR0CR <BR0S3:0>	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
	0000(N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (UART only) to 1111(N = 15) 0000(N = 16)
0000	Disable	Disable	Divided by N
0001(K = 1) to 1111(K = 15)	Disable	Divided by N + (16-K) /16	

Note1:Availability of +(16-K)/16 division function

N	UART mode	I/O mode
2 to 15	○	×
1 , 16	×	×

The baud rate generator can be set “1” in UART mode and disable +(16-K)/16 division function.Don’t use in I/O interface mode.

Note2:Set BR0CR <BR0ADDE> to 1 after setting K (K = 1 to 15) to BR0ADD<BR0K3:0> when +(16-K)/16 division function is used. Writes to unused bits in the BR0ADD register do not affext operation, and undefined data is read from these unused bits.

Figure 3.14.8 Baud rate generator control (channel 0, BR0CR, BR0ADD)

SC0BUF (1200H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

Note: Prohibit read modify write for SC0BUF.

Figure 3.14.9 Serial Transmission/Receiving Buffer Registers (channel 0, SC0BUF)

SC0MOD1 (1205H)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0						
Read/Write	R/W	R/W						
After Reset	0	0						
Function	IDLE2 0: Stop 1: Run	duplex 0: half 1: full						

Figure 3.14.10 Serial Mode Control Register 1 (channel 0, SC0MOD1)

3.14.4 Operation in each mode

(1) Mode 0 (I/O Interface Mode)

This mode allows an increase in the number of I/O pins available for transmitting data to or receiving data from an external shift register.

This mode includes the SCLK output mode to output synchronous clock SCLK and SCLK input mode to input external synchronous clock SCLK.

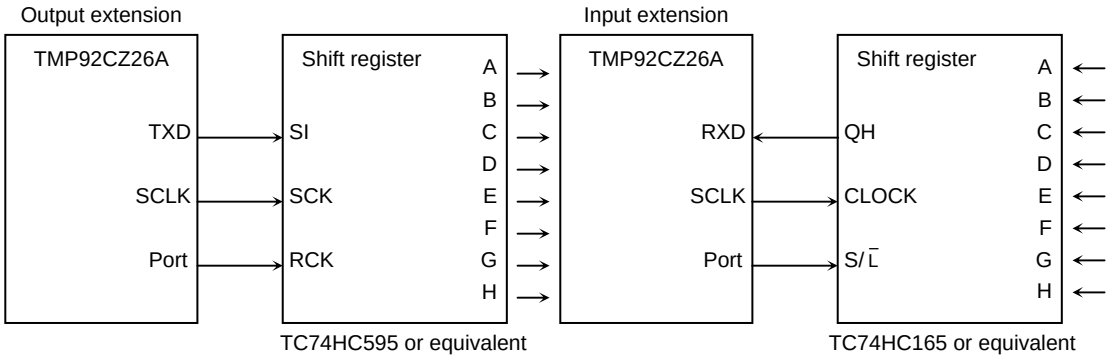


Figure 3.14.11 SCLK Output Mode connection example

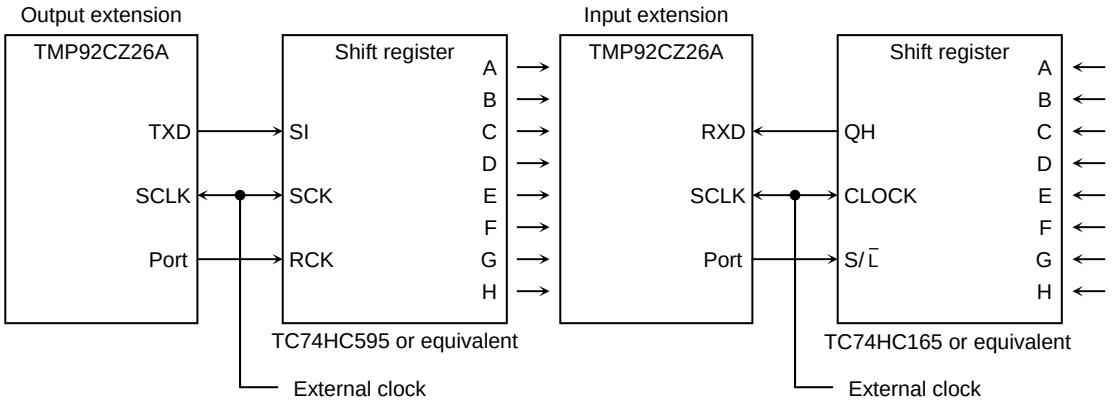


Figure 3.14.12 Example of SCLK Input Mode Connection

a. Transmission

In SCLK output mode 8-bit data and a synchronous clock are output on the TXD0 and SCLK0 pins respectively each time the CPU writes the data to the Transmission Buffer. When all data is output, INTES0 <ITX0C> will be set to generate the INTTX0 interrupt.

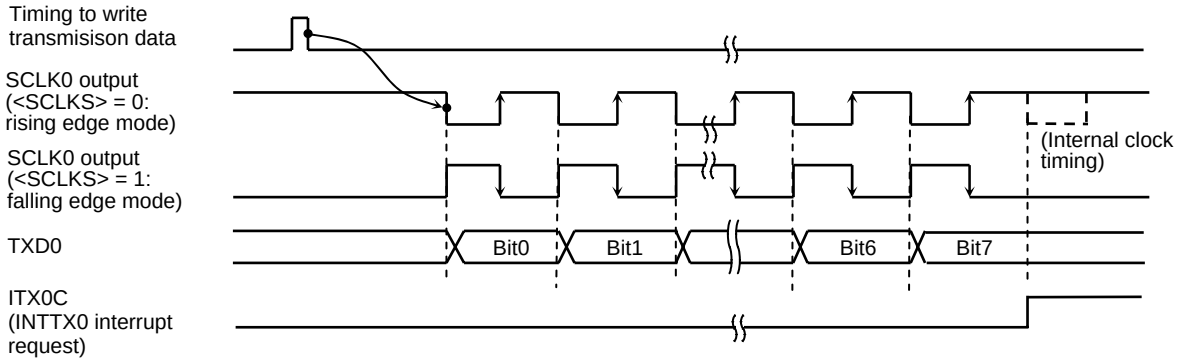


Figure 3.14.13 Transmitting Operation in I/O Interface Mode (SCLK0 Output Mode)

In SCLK Input Mode, 8-bit data is output on the TXD0 pin when the SCLK0 input becomes active after the data has been written to the Transmission Buffer by the CPU.

When all data is output, INTES0 <ITX0C> will be set to generate INTTX0 interrupt.

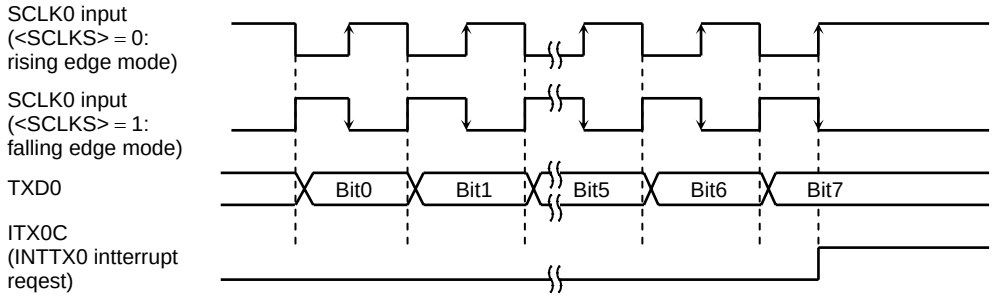


Figure 3.14.14 Transmitting Operation in I/O Interface Mode (SCLK0 Input Mode)

b. Receiving

In SCLK Output Mode the synchronous clock is output on the SCLK0 pin and the data is shifted to Receiving Buffer 1. This is initiated when the Receive Interrupt flag INTES0<IRX0C> is cleared as the received data is read. When 8-bit data is received, the data is transferred to Receiving Buffer 2 (SC0BUF) following the timing shown below and INTES0<IRX0C> is set to 1 again, causing an INTRX0 interrupt to be generated.

Setting SC0MOD0<RXE> to 1 initiates SCLK0 output.

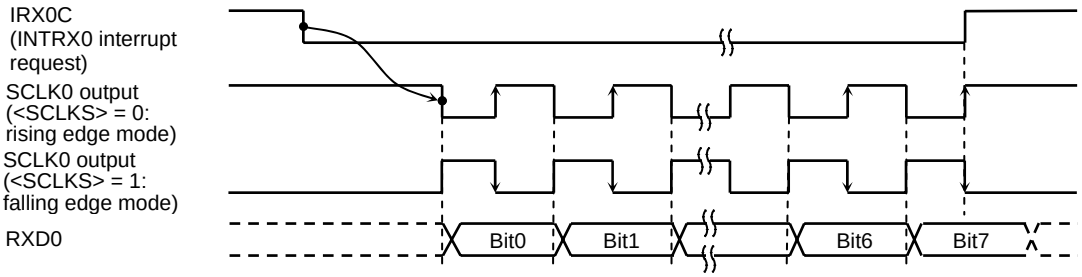


Figure 3.14.15 Receiving operation in I/O Interface Mode (SCLK0 Output Mode)

In SCLK Input Mode the data is shifted to Receiving Buffer 1 when the SCLK input goes active. The SCLK input goes active when the Receive Interrupt flag INTES0 <IRX0C> is cleared as the received data is read. When 8-bit data is received, the data is shifted to Receiving Buffer 2 (SC0BUF) following the timing shown below and INTES0 <IRX0C> is set to 1 again, causing an INTRX0 interrupt to be generated.

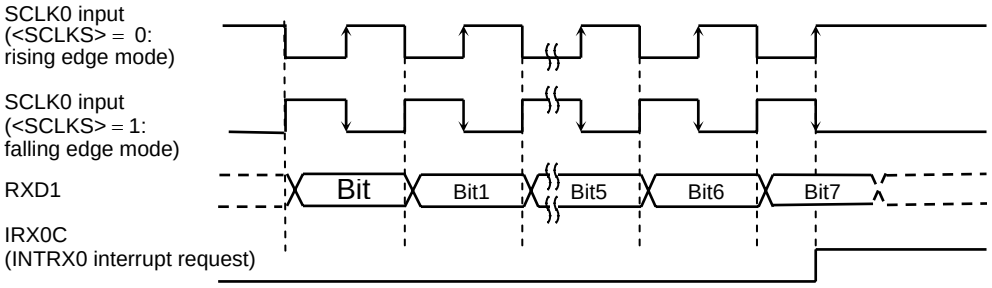


Figure 3.14.16 Receiving Operation in I/O interface Mode (SCLK0 Input Mode)

Note: The system must be put in the Receive Enable state (SC0MOD0<RXE> = 1) before data can be received.

c. Transmission and Receiving (Full Duplex Mode)

When Full Duplex Mode is used, set the Receive Interrupt Level to 0 and set enable the level of transmit interrupt(1 to 6). Ensure that the program which transmits the interrupt reads the receiving buffer before setting the next transmit data.
The following is an example of this:

Example: Channel 0, SCLK output
Baud rate = 9600 bps
fsys = 2.4576 MHz

Main routine

	7	6	5	4	3	2	1	0	
INTES0	X	0	0	1	X	0	0	0	Set the INTTX0 level to 1. Set the INTRX0 level to 0.
P9CR	X	X	X	X	X	1	0	1	Set P90, P91 and P92 to function as the TXD0,
P9FC	–	–	X	X	X	1	X	1	RXD0 and SCLK0 pins respectively.
SC0MOD0	–	–	–	–	0	0	–	–	Select I/O interface mode.
SC0MOD1	–	1	X	X	X	X	X	X	Select full duplex mode.
SC0CR	–	–	–	–	–	–	0	0	SCLK0 output mode, select rising edge
BR0CR	0	0	0	1	1	0	0	0	Baud rate = 9600 bps.
SC0MOD0	–	–	1	–	–	–	–	–	Enable receiving.
SC0BUF	*	*	*	*	*	*	*	*	Set the transmit data and start.

INTTX0 interrupt routine

A_{CC}	←	SC0BUF							Read the receiving buffer.
SC0BUF		*	*	*	*	*	*	*	*	Set the next transmit data.

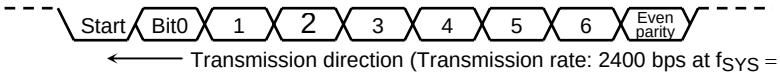
X: Don't care, –: No change

(2) Mode 1 (7-bit UART Mode)

7-Bit UART Mode is selected by setting the Serial Channel Mode Register SC0MOD0<SM1:0> field to 01.

In this mode a parity bit can be added. Use of a parity bit is enabled or disabled by the setting of the Serial Channel Control Register SC0CR<PE> bit; whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (enabled).

Setting example: When transmitting data of the following format, the control registers should be set as described below.



← Transmission direction (Transmission rate: 2400 bps at f_{SYS} = 19.6608 MHz)

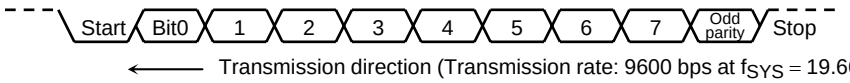
		7	6	5	4	3	2	1	0	
P9CR	←	X	X	X	X	X	–	–	1	} Set P90 to function as the TXD0 pin.
P9FC	←	–	–	X	X	X	–	X	1	
SC0MOD0	←	X	0	–	X	0	1	0	1	Select 7-bit UART mode.
SC0CR	←	–	1	1	–	–	–	–	–	Add even parity.
BR0CR	←	0	0	1	0	1	0	0	0	Set the transfer rate to 2400 bps.
INTES0	←	X	1	0	0	X	0	0	0	Enable the INTTX0 interrupt and set it to interrupt level 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

X: Don't care, –: No change

(3) Mode 2 (8-Bit UART Mode)

8-Bit UART Mode is selected by setting SC0MOD0<SM1,SM0> to 10. In this mode a parity bit can be added (use of a parity bit is enabled or disabled by the setting of SC0CR<PE>); whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (enabled).

Setting example: When receiving data of the following format, the control registers should be set as described below.



← Transmission direction (Transmission rate: 9600 bps at f_{SYS} = 19.6608 MHz)

Main routine

		7	6	5	4	3	2	1	0	
P9CR	←	X	X	X	X	X	–	0	–	Set P91 to function as the RXD0 pin.
P9FC	←	–	–	X	X	X	–	X	–	
SC0MOD0	←	–	–	1	–	1	0	0	1	Enable receiving in 8-bit UART mode.
SC0CR	←	–	0	1	–	–	–	–	–	Add odd parity.
BR0CR	←	0	0	0	1	1	0	0	0	Set the transfer rate to 9600 bps.
INTES0	←	X	1	0	0	X	0	0	0	Enable the INTTX0 interrupt and set it to interrupt level 4.

Interrupt routine

A_{CC} ← SC0CR AND 00011100 } Check for errors

if $A_{CC} \neq 0$ then ERROR

A_{CC} ← SC0BUF Read the received data

X: Don't care, –: No change

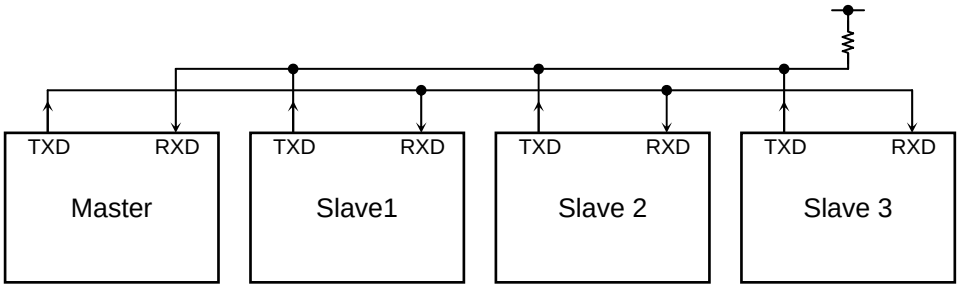
(4) Mode 3 (9-Bit UART Mode)

9-Bit UART Mode is selected by setting SC0MOD0<SM1:0> to 11. In this mode parity bit cannot be added.

In the case of transmission the MSB (9th bit) is written to SC0MOD0<TB8>. In the case of receiving it is stored in SC0CR<RB8>. When the buffer is written and read, the MSB is read or written first, before the rest of the SC0BUF data.

Wake-up function

In 9-Bit UART Mode, the wake-up function for slave controllers is enabled by setting SC0MOD0<WU> to 1. The interrupt INTRX0 can only be generated when<RB8> = 1.

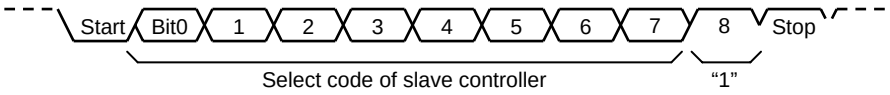


Note: The TXD pin of each slave controller must be in Open-Drain Output Mode.

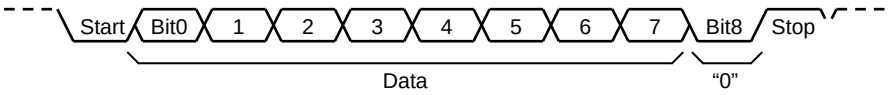
Figure 3.14.17 Serial Link using Wake-up function

Protocol

1. Select 9-Bit UART Mode on the master and slave controllers.
2. Set the SC0MOD0<WU> bit on each slave controller to 1 to enable data receiving.
3. The master controller transmits data one frame at a time. Each frame includes an 8-bit select code which identifies a slave controller. The MSB (bit 8) of the data (<TB8>) is set to 1.

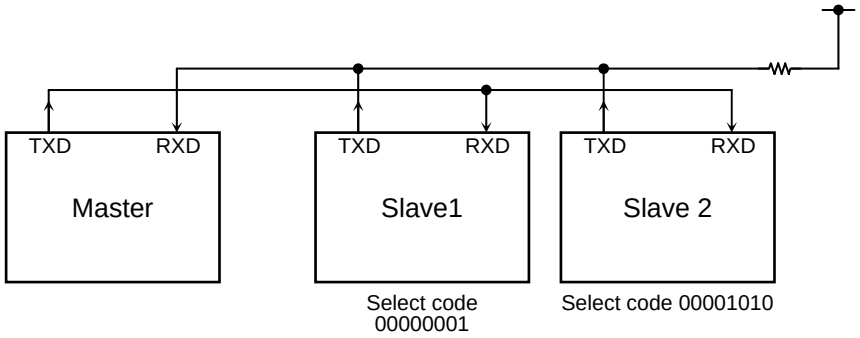


4. Each slave controller receives the above frame. Each controller checks the above select code against its own select code. The controller whose code matches clears its <WU> bit to 0.
5. The master controller transmits data to the specified slave controller (the controller whose SC0MOD0<WU> bit has been cleared to 0). The MSB (bit 8) of the data (<TB8>) is cleared to 0.



6. The other slave controllers (whose <WU> bits remain at 1) ignore the received data because their MSBs (bit 8 or <RB8>) are set to 0, disabling INTRX0 interrupts.
   The slave controller whose <WU> bit = 0 can also transmit to the master controller. In this way it can signal the master controller that the data transmission from the master controller has been completed.

Setting example: To link two slave controllers serially with the master controller using the internal clock f_{IO} as the transfer clock.



- Setting the master controller

Main routine

P9CR	← X X X X X – 0 1	Set P90 and P91 to function as the TXD0 and RXD0 pins respectively.
P9FC	← – – X X X – X 1	
INTES0	← X 1 0 0 X 1 0 1	Enable the INTTX0 interrupt and set it to Interrupt Level 4. Enable the INTRX0 interrupt and set it to Interrupt Level 5.
SC0MOD0	← 1 0 1 0 1 1 1 0	Set f_{IO} as the transmission clock for 9-Bit UART Mode.
SC0BUF	← 0 0 0 0 0 0 0 1	Set the select code for slave controller 1.

Interrupt routine (INTTX0)

SC0MOD0	← 0 – – – – – – –	Set TB8 to 0.
SC0BUF	← * * * * * * * *	Set data for transmission.

- Setting the slave controller

Main routine

P9CR	← X X X X X – 0 1	Select P91 and P90 to function as the RXD0 and TXD0 pins respectively (open-drain output).
P9FC	← – – X X X – X 1	
P9FC2	← X X X X X X X 1	
INTES0	← X 1 0 0 X 1 0 0	Enable INTRX0 and INTTX0.
SC0MOD0	← 0 0 1 1 1 1 1 0	Set <WU> to 1 in 9-Bit UART Transmission Mode using f_{SYS} as the transfer clock.

Interrupt routine (INTRX0)

Acc ← SC0BUF
if Acc =Select code
Then SC0MOD0 ← – – – – 0 – – – – Clear <WU> to 0.

3.14.5 Support for IrDA

SIO0 includes support for the IrDA 1.0 infrared data communication specification. Figure 3.14.8 shows the block diagram.

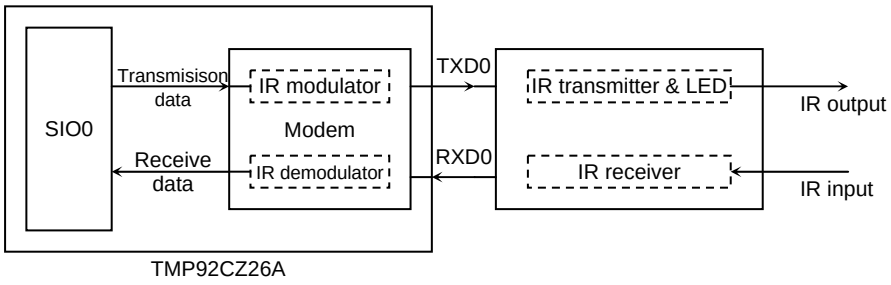


Figure 3.14.18 Block Diagram

(1) Modulation of the transmission data

When the transmit data is 0, the modem outputs 1 to TXD0 pin with either 3/16 or 1/16 times for width of baud-rate. The pulse width is selected by the SIRCR<PLSEL>. When the transmit data is 1, the modem outputs 0.

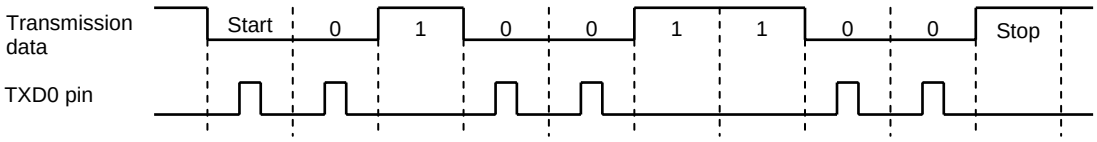


Figure 3.14.19 Transmission example

(2) Modulation of the receive data

When the receive data is the effective width of pulse “1”, the modem outputs “0” to SIO0. Otherwise the modem outputs “1” to SIO0. The effective pulse width is selected by SIRCR<SIRWD3 to SIRWD0>.

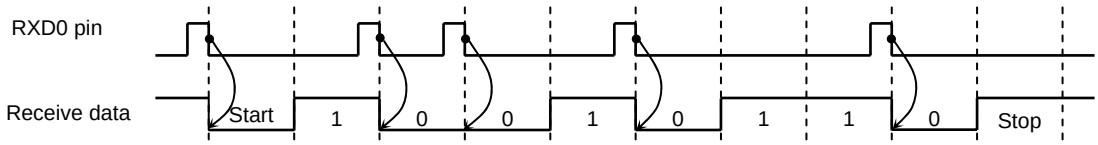


Figure 3.14.20 Receiving example

(3) Data format

The data format is fixed as follows:

- Data length: 8-bit
- Parity bits: none
- Stop bits: 1bit

(4) SFR

Figure 3.14.21 shows the control register SIRCR. Set the data SIRCR during SIO0 is stopping. The following example describes how to set this register:

```
1) SIO setting                  ; Set the SIO to UART Mode.
        ↓
2) LD (SIRCR), 07H              ; Set the receive data pulse width to 16×.
3) LD (SIRCR), 37H              ; TXEN, RXEN Enable the Transmission and receiving.
        ↓
4) Start transmission           ; The modem operates as follows:
   and receiving for SIO0       • SIO0 starts transmitting.
                                • IR receiver starts receiving.
```

(5) Notes

1. Baud rate for IrDA

When IrDA is operated, set 01 to SC0MOD0<SC1:0> to generate baud-rate.

The setting except above (TA0TRG, f_{IO} and SCLK0-input) cannot be used.

2. The pulse width for transmission

The IrDA 1.0 specification is defined in Table 3.14.3.

Table 3.14.3 Baud rate and pulse width specifications

Baud Rate	Modulation	Rate Tolerance (% of rate)	Pulse Width (minimum)	Pulse Width (typical)	Pulse width (maximum)
2.4 kbps	RZI	±0.87	1.41 μs	78.13 μs	88.55 μs
9.6 kbps	RZI	±0.87	1.41 μs	19.53 μs	22.13 μs
19.2 kbps	RZI	±0.87	1.41 μs	9.77 μs	11.07 μs
38.4 kbps	RZI	±0.87	1.41 μs	4.88 μs	5.96 μs
57.6 kbps	RZI	±0.87	1.41 μs	3.26 μs	4.34 μs
115.2 kbps	RZI	±0.87	1.41 μs	1.63 μs	2.23 μs

The infra-red pulse width is specified either baud rate T× 3/16 or 1.6 μs (1.6 μs is equal to 3/16 pulse width when baud rate is 115.2 kbps).

The TMP92CZ26A has the function selects the pulse width of Transmission either 3/16 or 1/16. But 1/16 pulse width can be selected when the baud rate is equal or less than 38.4 kbps.

As the same reason, + (16 – k)/16 division function in the baud rate generator of SIO0 can not be used to generate 115.2 kbps baud rate.

Also when the 38.4 kbps and 1/16 pulse width, + (16-K)/16 division function can not be used.

Table 3.14.4 Baud rate and pulse width for (16 – K) / 16 division function

Pulse Width	Baud Rate					
	115.2 Kbps	57.6 Kbps	38.4 Kbps	19.2 Kbps	9.6 Kbps	2.4 Kbps
T × 3/16	× (Note)	○	○	○	○	○
T × 1/16	–	–	×	○	○	○

○: Can be used (16 – K)/16 division function

×: Cannot be used (16 – K)/16 division function

–: Cannot be set to 1/16 pulse width

Note: Can be used (16 – K)/16 division function at a special condition.

SIRCR (1207H)

	7	6	5	4	3	2	1	0
Bit symbol	PLSEL	RXSEL	TXEN	RXEN	SIRWD3	SIRWD2	SIRWD1	SIRWD0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Select transmit pulse width 0: 3/16 1: 1/16	Receive data 0: "H" pulse 1: "L" pulse	Transmit 0: disable 1: enable	Receive 0: disable 1: enable	Select receive pulse width Set effective pulse width for equal or more than 2x × (value + 1) + 100ns Can be set : 1 to 14 Can not be set : 0, 15			

Select receive pulse width

Formula: Effective pulse width ≥ 2x × (value + 1) + 100ns

$x = 1/f_{FPH}$

0000	Cannot be set
0001	Equal or more than 4x + 100ns
to	
1110	Equal or more than 30x + 100ns
1111	Can not be set

Receive (recovery) operation

0	Disable receiving operation (Received data is ignored)
1	Enabled receiving operation

Transmit (modulation) operation

0	Disabled transmission operation (Input from SIO is ignored)
1	Enabled transmission operation

Select transmit pulse width

0	3/16 pulse width
1	1/16 pulse width

Figure 3.14.21 IrDA Control Register

3.15 Serial Bus Interface (SBI)

The TMP92CZ26A has a 1-channel serial bus interface which an I²C bus mode. This circuit supports only I²C bus mode (Multi master).

The serial bus interface is connected to an external device through PV6 (SDA) and PV7 (SCL) in the I²C bus mode.

Each pin is specified as follows.

	PVFC2<PV7F2, PV6F2>	PVCR<PV7C, PV6C>	PVFC<PV7F, PV6F>
I²C bus mode	11	11	11

3.15.1 Configuration

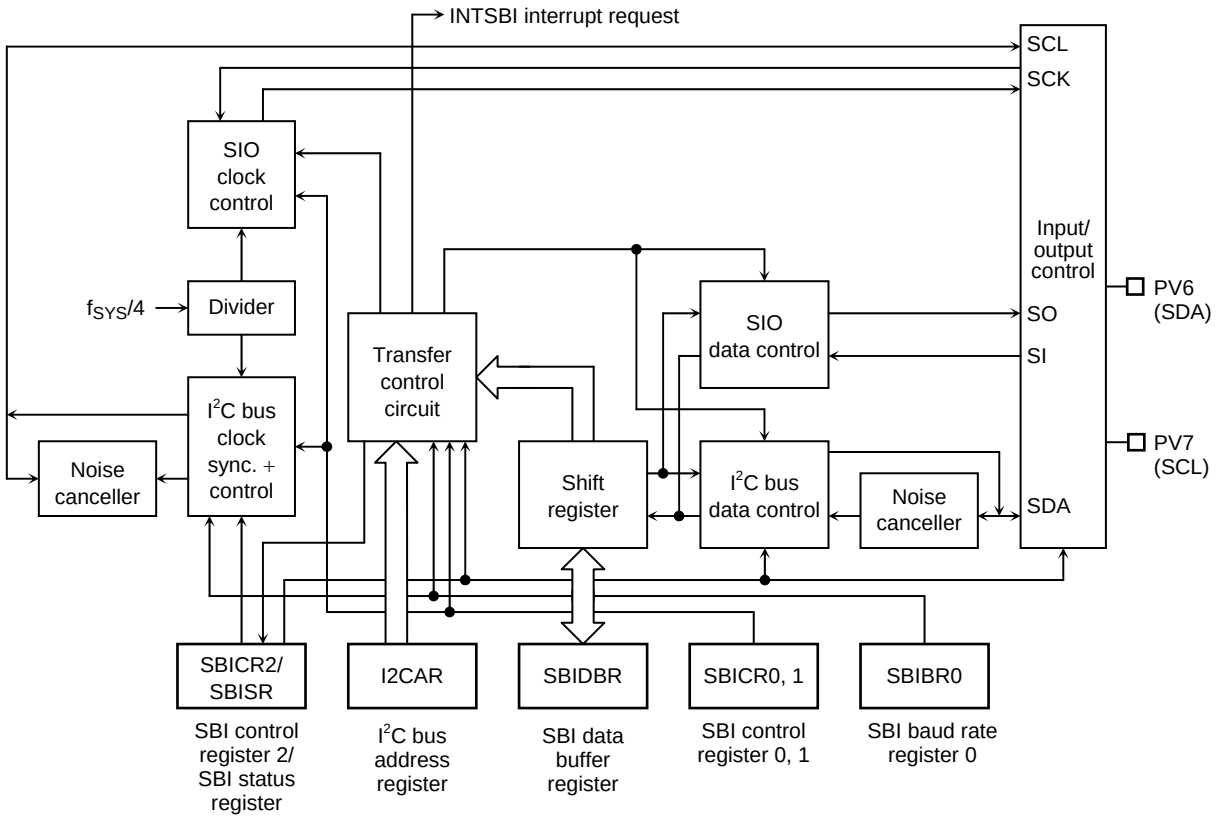


Figure 3.15.1 Serial bus interface (SBI)

3.15.2 Serial Bus Interface (SBI) Control

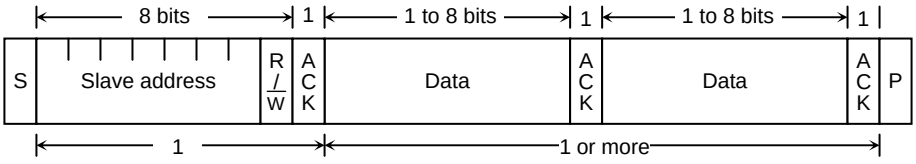
The following registers are used to control the serial bus interface and monitor the operation status.

- Serial bus interface control register 0 (SBICR0)
- Serial bus interface control register 1 (SBICR1)
- Serial bus interface control register 2 (SBICR2)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register (SBISR)
- Serial bus interface baud rate register 0 (SBIBR0)

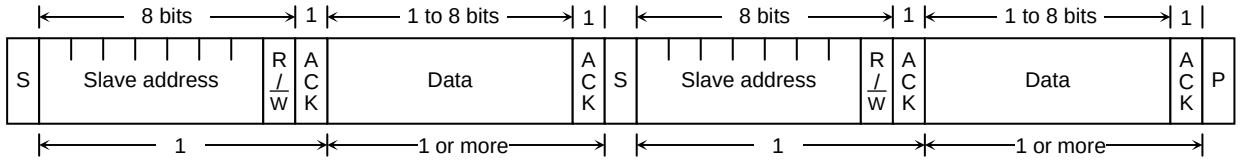
3.15.3 The Data Formats in the I²C Bus Mode

The data formats in the I²C bus mode is shown below.

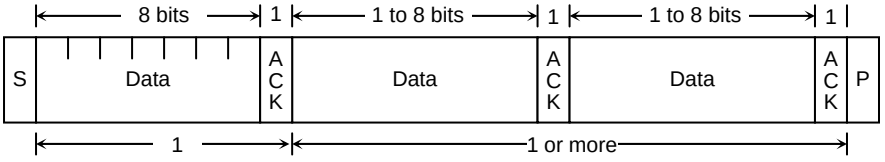
(a) Addressing format



(b) Addressing format (with restart)



(c) Free data format (data transferred from master device to slave device)



S: Start condition

R/$\overline{W}$: Direction bit

ACK: Acknowledge bit

P: Stop condition

Figure 3.15.2 Data format in the I²C bus mode

3.15.4 I²C Bus Mode Control Register

The following registers are used to control and monitor the operation status when using the serial bus interface (SBI) in the I²C bus mode.

Serial Bus Interface Control Register 0

		7	6	5	4	3	2	1	0
SBICR0 (1247H)	Bit symbol	SBIEN	−	−	−	−	−	−	−
	Read/Write	R/W	R						
	After Reset	0	0	0	0	0	0	0	0
Prohibit Read-modify-Write	Function	SBI operation 0 : disable 1 : enable	Always read “0”.						

<SBIEN> : When using SBI, <SBIEN> should be set “1” (SBI operation enable) before setting each register of SBI module.

Figure 3.15.3 Registers for the I²C bus mode

Serial Bus Interface Control Register 1

		7	6	5	4	3	2	1	0
SBICR1 (1240H)	Bit symbol	BC2	BC1	BC0	ACK	–	SCK2	SCK1	SCK0/ SWRMON
	Read/Write	R/W			R/W	R	R/W		R/W
	After Reset	0	0	0	0	1	0	0	0/1(Note2)
Prohibit Read-modify-write	Function	Number of transferred bits (Note 1)			Acknowledge mode specification 0: Not generate 1: Generate	Always read as "1".	Internal serial clock selection and software reset monitor		

Internal serial clock selection <SCK2:0> at write

f_{SYS}=80MHz (Output to SCL pin), Clock gear = fc/1

000	n = 4	–
001	n = 5	–
010	n = 6	–
011	n = 7	–
100	n = 8	68 kHz
101	n = 9	36 kHz
110	n = 10	18 kHz
111	(Reserved)	(Reserved)

System Clock: f_{SYS} (=80MHz)
Clock Gear : fc/1
$fscl = \frac{f_{SYS}/4}{2^n + 35}$ [Hz]

Software reset state monitor <SWRMON> at read

0	During software reset
1	(Initial Data)

Acknowledge mode specification

0	Not generate clock pulse for acknowledge signal
1	Generate clock pulse for acknowledge signal

Number of bits transferred

<BC2:0>	<ACK> = 0		<ACK> = 1	
	Number of clock pulses	Bits	Number of clock pulses	Bits
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

Note1: For the frequency of the SCL line clock, see 3.15.5 (3) Serial clock.

Note2: The initial data of SCK0 is "0", the initialdata of SWRMON is "1" if SBI operation is enable (SBICR0<SBIEN>="1"). If SBI operation is disable (SBICR0<SBIEN>="0"), the initialdata of SWRMON is "0".

Note3: This I^2C bus circuit does not support Fast-mode, it supports the Standard mode only. Although the I^2C bus circuit itself allows the setting of a baud rate over 100kbps, the compliance with the I^2C specification is not guaranteed in that case.

Figure 3.15.4 Registers for the I^2C bus mode

Serial Bus Interface Control Register 1

		7	6	5	4	3	2	1	0
SBICR2 (1243H)	Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
	Read/Write	W				W (Note 1)		W (Note 1)	
	After reset	0	0	0	1	0	0	0	0
Prohibit Read-modify-write	Function	Master/Slave selection 0:Slave 1:Master	Transmitter /Receiver selection 0:Receiver 1:Transmitter	Start/Stop condition Generation 0:Generate stop condition 1:Generate start condition	Cancel INTSBI interrupt request 0:Don't care 1:Cancel interrupt request	Serial bus interface operating mode selection (Note 2) 00: Port mode 01: (Reserved) 10: I²C Bus mode 11: (Reserved)		Software reset generate write "10" and "01", then an internal reset signal is generated.	

Serial bus interface operating mode selection (Note2)

00	Port Mode (Serial Bus Interface output disabled)
01	Reserved
10	I²C Bus Mode
11	Reserved

Note 1: Reading this register functions as SBISR register.

Note 2: Switch a mode to port mode after confirming that the bus is free.

Switch a mode between I²C bus mode and port mode after confirming that input signals via port are high-level.

Figure 3.15.5 Registers for the I²C bus mode

Table 3.15.1Resolution of base clock

@f_{SYS} = 80MHz

Clock Gear <GEAR1:0>	Base Clock Resolution
000(fc)	$f_{SYS}/2^2$ (50ns)
001(fc/2)	$f_{SYS}/2^3$ (0.1us)
010(fc/4)	$f_{SYS}/2^4$ (0.2us)
011(fc/8)	$f_{SYS}/2^5$ (0.4us)
100(fc/16)	$f_{SYS}/2^6$ (0.8us)

Serial Bus Interface Status Register

		7	6	5	4	3	2	1	0
SBISR (1243H)	Bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
	Read/Write	R							
	After reset	0	0	0	1	0	0	0	0
Prohibit Read-modify-write	Function	Master/ Slave status monitor 0:Slave 1:Master	Transmitter/ Receiver status monitor 0:Receiver 1:Tranmitter	I^2C bus status monitor 0:Free 1:Busy	INTSBI interrupt request monitor 0: Interrupt requested 1: Interrupt canceled	Arbitration lost detection monitor 0: – 1: Detected	Slave address match detection monitor 0: Undetected 1: Detected	GENERAL CALL detection monitor 0: Undetected 1: Detected	Last received bit monitor 0: 0 1: 1

Last received bit monitor

0	Last received bit was 0
1	Last received bit was 1

GENERAL CALL detection monitor

0	Undetected
1	GENERAL CALL detected

Slave address match detection monitor

0	Slave address don't match or Undetected
1	Slave address match or GENERAL CALL detected

Arbitration lost detection monitor

0	–
1	Arbitration lost

Note1: Writing in this register functions as SBICR2.

Note2: The initialdata SBISR<PIN> is "1" if SBI operation is enable (SBICR0<SBIEN>="1"). If SBI operation is disable (SBICR0<SBIEN>="0"), the initialdata of SBISR<PIN> is "0".

Figure 3.15.6 Registers for the I^2C bus mode

Serial Bus Interface Baud Rate Register 0

		7	6	5	4	3	2	1	0
SBIBR0 (1244H) Prohibit Read-modify -write	Bit symbol	–	I2SBI	–	–	–	–	–	–
	Read/Write	W	R/W	R					R/W
	After reset	0	0	1	1	1	1	1	0
	Function	Always read “0”	IDLE2 0: Stop 1: Run	Always read as “1”					Always write “0”.

Operation during IDLE 2 mode

0	Stop
1	Operation

Serial Bus Interface Data Buffer Register

		7	6	5	4	3	2	1	0
SBIDBR (1241H) Prohibit Read-modify -write	Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	Read/Write	R (received)/W (transfer)							
	After reset	Undefined							

Note1: When writing transmitted data, start from the MSB (bit 7).Receiving data is placed from LSB(bit0).

Note2: SBIDBR can’t be read the written data because of it has buffer for writing and buffer for reading individually.Therefore Read modify write instruction (e.g.“BIT” instruction) is prohibitted.

Note3:Written data to SBIDBR is cleared by INTSBI signal.

I^2C Bus Address Register

		7	6	5	4	3	2	1	0
I2CAR (1242H) Prohibit Read-modify -write	Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Slave address selection for when device is operating as slave device							Address recognition mode specification

Address recognition mode specification

0	Slave address recognition
1	Non slave address recognition

Figure 3.15.7 Registers for the I^2C bus mode

3.15.5 Control in I^2C Bus Mode

(1) Acknowledge Mode Specification

When slave address is matched or detecting GENERAL CALL, and set the SBICR1<ACK> to "1", TMP92CZ26A operates in the acknowledge mode. The TMP92CZ26A generates an additional clock pulse for an Acknowledge signal when operating in Master Mode. In the transmitter mode during the clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the Low in order to generate the acknowledge signal.

Clear the <ACK> to "0" for operation in the Non-Acknowledge Mode; The TMP92CZ26A does not generate a clock pulse for the Acknowledge signal when operating in the Master Mode.

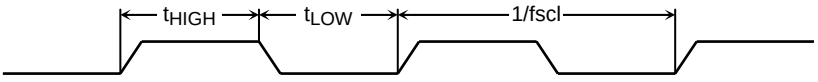
(2) Number of transfer bits

The SBICR1<BC2:0> is used to select a number of bits for next transmitting and receiving data.
Since the <BC2:0> is cleared to 000 as a start condition, a slave address and direction bit transmission are executed in 8 bits. Other than these, the <BC2:0> retains a specified value.

(3) Serial clock

a. Clock source

The SBICR1 <SCK2:0> is used to select a maximum transfer frequency outputted on the SCL pin in Master Mode. Set the baud rates, which have been calculated according to the formula below, to meet the specifications of the I2C bus, such as the smallest pulse width of t_{LOW}.



$t_{LOW} = (2^{n-1} + 29)/(f_{SYS}/4)$
$t_{HIGH} = (2^{n-1} + 6)/(f_{SYS}/4)$
$fscl = 1/(t_{LOW} + t_{HIGH})$

$$= \frac{f_{SYS}/4}{2^n + 35}$$

SBICR1<SCK2:0>	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10

Figure 3.15.8 Clock source

b. Clock synchronization

In the I²C bus mode, in order to wired-AND a bus, a master device which pulls down a clock line to low-level, in the first place, invalidate a clock pulse of another master device which generates a high-level clock pulse. The master device with a high-level clock pulse needs to detect the situation and implement the following procedure.
The TMP92CZ26A has a clock synchronization function for normal data transfer even when more than one master exists on the bus.
The example explains the clock synchronization procedures when two masters simultaneously exist on a bus.

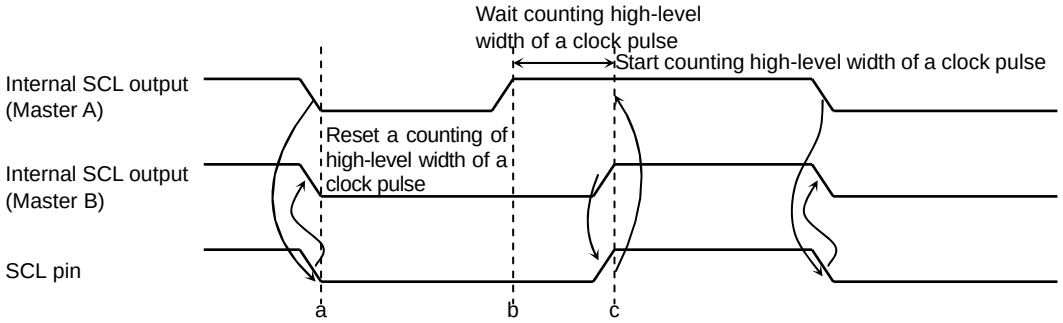


Figure 3.15.9 Clock synchronization

As Master A pulls down the internal SCL output to the Low level at point “a”, the SCL line of the bus becomes the Low-level. After detecting this situation, Master B resets a counter of High-level width of an own clock pulse and sets the internal SCL output to the Low-level.
Master A finishes counting Low-level width of an own clock pulse at point “b” and sets the internal SCL output to the High-level. Since Master B holds the SCL line of the bus at the Low-level, Master A wait for counting high-level width of an own clock pulse. After Master B finishes counting low-level width of an own clock pulse at point “c” and Master A detects the SCL line of the bus at the High-level, and starts counting High-level of an own clock pulse. The clock pulse on the bus is determined by the master device with the shortest High-level width and the master device with the longest Low-level width from among those master devices connected to the bus.

(4) Slave address and address recognition mode specification

When the TMP92CZ26A is used as a slave device, set the slave address <SA6:0> and <ALS> to the I2CAR. Clear the <ALS> to “0” for the address recognition mode.

(5) Master/Slave selection

Set the SBICR2<MST> to “1” for operating the TMP92CZ26A as a master device. Clear the SBICR2<MST> to “0” for operation as a slave device. The <MST> is cleared to “0” by the hardware after a stop condition on the bus is detected or arbitration is lost.

(6) Transmitter/Receiver selection

Set the SBICR2<TRX> to "1" for operating the TMP92CZ26A as a transmitter. Clear the <TRX> to "0" for operation as a receiver.

In Slave Mode,

- Data with an addressing format is transferred
- A slave address with the same value that an I2CAR
- A GENERAL CALL is received (all 8-bit data are "0" after a start condition)

The <TRX> is set to "1" by the hardware if the direction bit ($R/\overline{W}$) sent from the master device is "1", and is cleared to "0" by the hardware if the bit is "0".

In the Master Mode, after an Acknowledge signal is returned from the slave device, the <TRX> is cleared to "0" by the hardware if a transmitted direction bit is "1", and is set to "1" by the hardware if it is "0". When an Acknowledge signal is not returned, the current condition is maintained.
The <TRX> is cleared to "0" by the hardware after a stop condition on the I^2C bus is detected or arbitration is lost.

(7) Start/Stop condition generation

When the SBISR<BB> is "0", slave address and direction bit which are set to SBIDBR are output on a bus after generating a start condition by writing "1" to the SBICR2 <MST, TRX, BB, PIN>. It is necessary to set transmitted data to the data buffer register (SBIDBR) and set "1" to <ACK> beforehand.

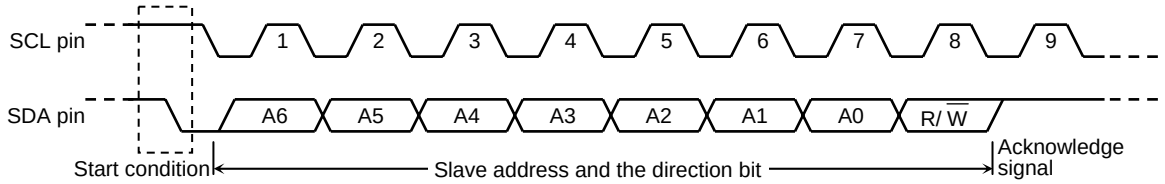


Figure 3.15.10 Start condition generation and slave address generation

When the <BB> is "1", a sequence of generating a stop condition is started by writing "1" to the <MST, TRX, PIN>, and "0" to the <BB>. Do not modify the contents of <MST, TRX, BB, PIN> until a stop condition is generated on a bus.

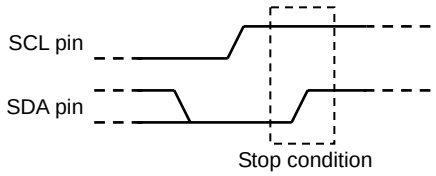


Figure 3.15.11 Stop condition generation

The state of the bus can be ascertained by reading the contents of SBISR<BB>. SBISR<BB> will be set to 1 if a start condition has been detected on the bus, and will be cleared to 0 if a stop condition has been detected.

(8) Interrupt service requests and interrupt cancellation

When a serial bus interface interrupt request (INTSBI) occurs, the SBICR2 <PIN> is cleared to "0". During the time that the SBICR2<PIN> is "0", the SCL line is pulled down to the Low level.

The <PIN> is cleared to "0" when a 1-word of data is transmitted or received. Either writing/reading data to/from SBIDBR sets the <PIN> to "1".

The time from the <PIN> being set to "1" until the SCL line is released takes t_{LOW}.

In the address recognition mode (<ALS> = "0"), <PIN> is cleared to "0" when the received slave address is the same as the value set at the I2CAR or when a GENERAL CALL is received (all 8-bit data are "0" after a start condition). Although SBICR2<PIN> can be set to "1" by the program, the <PIN> is not clear it to "0" when it is written "0".

(9) Serial bus interface operation mode selection

SBICR2<SBIM1:0> is used to specify the serial bus interface operation mode. Set SBICR2< SBIM1:0> to "10" when the device is to be used in I²C Bus Mode after confirming pin condition of serial bus interface to "H".

Switch a mode to port after confirming a bus is free.

(10) Arbitration lost detection monitor

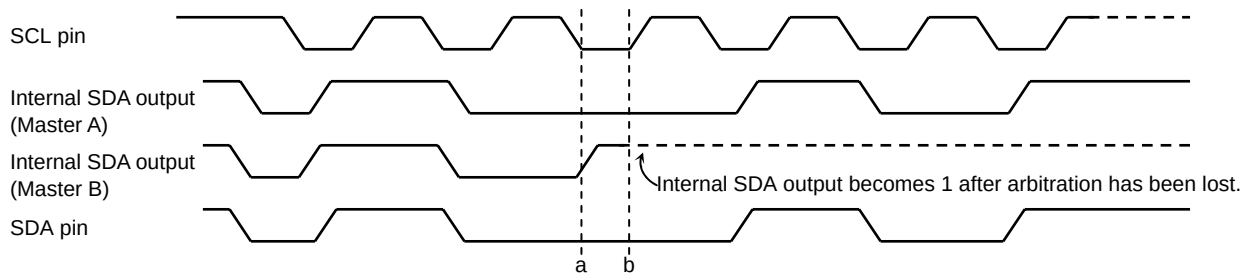
Since more than one master device can exist simultaneously on the bus in I²C Bus Mode, a bus arbitration procedure has been implemented in order to guarantee the integrity of transferred data.

In case set start condition bit with bus is busy, start condition is not output on SCL and SDA pin, but arbitration lost is generated.

Data on the SDA line is used for I²C bus arbitration.

The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on the bus. Master A and Master B output the same data until point "a". After Master A outputs "L" and Master B, "H", the SDA line of the bus is wire-AND and the SDA line is pulled down to the Low-level by Master A. When the SCL line of the bus is pulled up at point b, the slave device reads the data on the SDA line, that is, data in Master A. A data transmitted from Master B becomes invalid. The state in Master B is called "ARBITRATION LOST". Master B device which loses arbitration releases the internal SDA output in order not to affect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.

Figure 3.15.12 Arbitration lost



The TMP92CZ26A compares the levels on the bus's SDA line with those of the internal SDA output on the rising edge of the SCL line. If the levels do not match, arbitration is

lost and SBISR<AL> is set to “1”.
When SBISR<AL> is set to “1”, SBISR<MST, TRX> are cleared to “00” and the mode is switched to Slave Receiver Mode. Thus, clock output is stopped in data transfer after setting <AL>=“1”.

SBISR<AL> is cleared to “0” when data is written to or read from SBIDBR or when data is written to SBICR2.

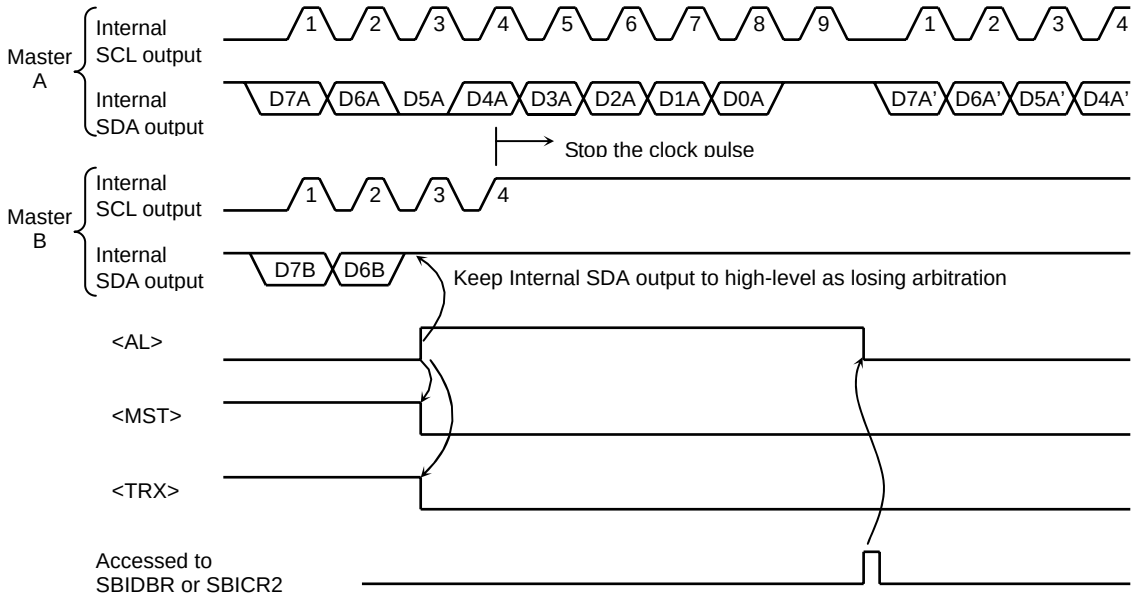


Figure 3.15.13 Example of when TMP92CZ26A is a master device B

(D7A = D7B, D6A = D6B)

(11) Slave address match detection monitor

SBISR<AAS> is set to “1” in Slave Mode, in Address Recognition Mode (i.e. when I2CAR<ALS> = “0”), when a GENERAL CALL is received, or when a slave address matches the value set in I2CAR. When I2CAR<ALS> = “1”, SBISR<AAS> is set to “1” after the first word of data has been received. SBISR<AAS> is cleared to “0” when data is written to or read from the data buffer register SBIDBR.

(12) GENERAL CALL detection monitor

SBISR<AD0> is set to “1” in Slave Mode, when a GENERAL CALL is received (all 8-bit received data is “0”, after a start condition). SBISR<AD0> is cleared to “0” when a start condition or stop condition is detected on the bus.

(13) Last received bit monitor

The SDA line value stored at the rising edge of the SCL line is set to the SBISR<LRB>. In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the SBISR<LRB>.

(14) Software Reset function

The software Reset function is used to initialize the SBI circuit, when SBI is rocked by external noises, etc.

An internal Reset signal pulse can be generated by setting SBICR2<SWRST1:0> to "10" and "01". This initializes the SBI circuit internally. All command registers and status registers are initialized as well.

SBICR1<SWRMON>is automatically set to "1" after the SBI circuit has been initialized.

Note: If the software reset is executied , operation selection is reset, and its mode is set to port mode from I2C mode.

(15) Serial Bus Interface Data Buffer Register (SBIDBR)

The received data can be read and transferred data can be written by reading or writing the SBIDBR.

In the master mode, after the start condition is generated the slave address and the direction bit are set in this register.

(16) I²CBUS Address Register (I2CAR)

I2CAR<SA6:0> is used to set the slave address when the TMP92CZ26A functions as a slave device.

The slave address output from the master device is recognized by setting the I2CAR<ALS> to "0". The data format is the addressing format. When the slave address is not recognized at the <ALS> = "1", the data format is the free data format.

(17) Setting register for IDLE2 mode operation (SBIBR0)

SBIBR0<I2SBI> is the register setting operation/stop during IDLE2-mode. Therefore, setting <I2SBI> is necessary before the HALT instruction is executed.

3.15.6 Data Transfer in I^2C Bus Mode

(1) Device initialization

Set the SBICR1<ACK, SCK2:0>, Set SBIBR1 to "1" and clear bits 7 to 5 and 3 in the SBICR1 to "0".
Set a slave address <SA6:0> and the <ALS> (<ALS> = "0" when an addressing format) to the I2CAR.
For specifying the default setting to a slave receiver mode, clear "0" to the <MST, TRX, BB> and set "1" to the <PIN>, "10" to the <SBIM1:0>.

```
                  7 6 5 4 3 2 1 0
SBICR1    ← 0 0 0 X 0 X X X      Set acknowledge and SCL clock.
I2CAR     ← X X X X X X X X      Set slave address and address recognition mode.
SBICR2    ← 0 0 0 1 1 0 0 0      Set to slave receiver mode.
Note: X: Don't care
```

(2) Start condition and slave address generation

a. Master Mode

In the Master Mode, the start condition and the slave address are generated as follows.
Check a bus free status (when <BB> = "0").
Set the SBICR1<ACK> to "1" (Acknowledge Mode) and specify a slave address and a direction bit to be transmitted to the SBIDBR.
When SBICR2<BB> = "0", the start condition are generated by writing "1111" to SBICR2<MST, TRX, BB, PIN>. Subsequently to the start condition, nine clocks are output from the SCL pin. While eight clocks are output, the slave address and the direction bit which are set to the SBIDBR. At the 9th clock, the SDA line is released and the acknowledge signal is received from the slave device.
An INTSBI interrupt request occurs at the falling edge of the 9th clock. The <PIN> is cleared to "0". In the Master Mode, the SCL pin is pulled down to the Low-level while <PIN> is "0". When an interrupt request occurs, the <TRX> is changed according to the direction bit only when an acknowledge signal is returned from the slave device.

Setting in main routine

```
                  7 6 5 4 3 2 1 0
┌→ Reg.       ← SBISR
│  Reg.       ← Reg. e 0x20
│  if Reg.    ≠ 0x00                  Wait until bus is free.
└─ Then
   SBICR1     ← X X X 1 X X X X      Set to acknowledgement mode.
   SBIDBR1    ← X X X X X X X X      Set slave address and direction bit.
   SBICR2     ← 1 1 1 1 1 0 0 0      Generate start condition.
```

In INTSBI interrupt routine

```
INTCLR ← 0X2a        Clear the interrupt request
Process
End of interrupt
```

b. Slave Mode

- In the Slave Mode, the start condition and the slave address are received.

  After the start condition is received from the master device, while eight clocks are output from the SCL pin, the slave address and the direction bit that are output from the master device are received.

  When a GENERAL CALL or the same address as the slave address set in I2CAR is received, the SDA line is pulled down to the Low-level at the 9th clock, and the acknowledge signal is output.

  An INTSBI interrupt request occurs on the falling edge of the 9th clock. The <PIN> is cleared to “0”. In Slave Mode the SCL line is pulled down to the Low-level while the <PIN> = “0”.

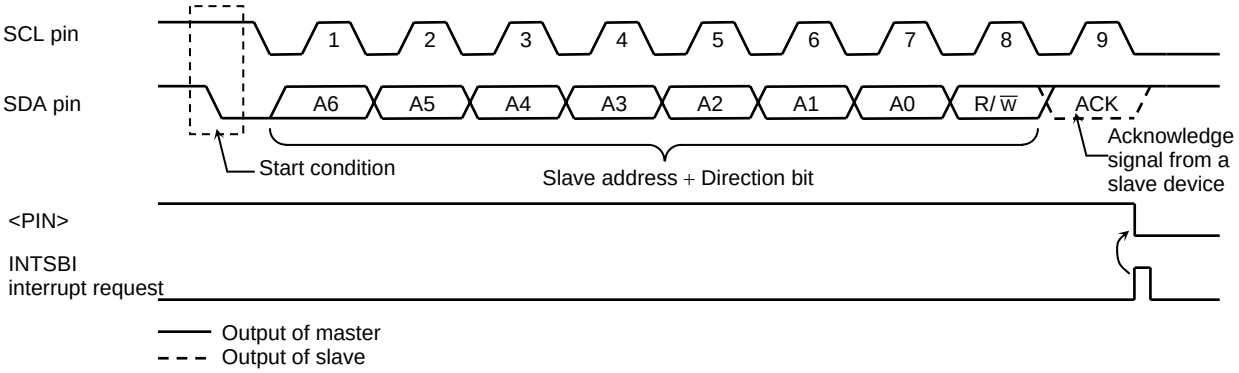


Figure 3.15.14 Start condition generation and slave address transfer

(3) 1-word Data Transfer

Check the <MST> by the INTSBI interrupt process after the 1-word data transfer is completed, and determine whether the mode is a master or slave.

a. If <MST> = "1" (Master Mode)

Check the <TRX> and determine whether the mode is a transmitter or receiver.

When the <TRX> = "1" (Transmitter mode)

Check the <LRB>. When <LRB> is "1", a receiver does not request data. Implement the process to generate a stop condition (Refer to 3.15.6 (4)) and terminate data transfer.
When the <LRB> is "0", the receiver is requests new data. When the next transmitted data is 8 bits, write the transmitted data to SBIDBR. When the next transmitted data is other than 8 bits, set the <BC2:0> <ACK> and write the transmitted data to SBIDBR. After written the data, <PIN> becomes "1", a serial clock pulse is generated for transferring a new 1-word of data from the SCL pin, and then the 1-word data is transmitted. After the data is transmitted, an INTSBI interrupt request occurs. The <PIN> becomes "0" and the SCL line is pulled down to the Low-level. If the data to be transferred is more than one word in length, repeat the procedure from the <LRB> checking above.

```
INTSBI interrupt
if MST = 0
Then shift to the process when slave mode
if TRX = 0
Then shift to the process when receiver mode.
if LRB = 0
Then shift to the process that generates stop condition.
                    7 6 5 4 3 2 1 0
  SBICR1    ← X X X X X X X X      Set the bit number of transmit and ACK.
  SBIDBR    ← X X X X X X X X      Write the transmit data.
  End of interrupt
  Note: X: Don't care
```

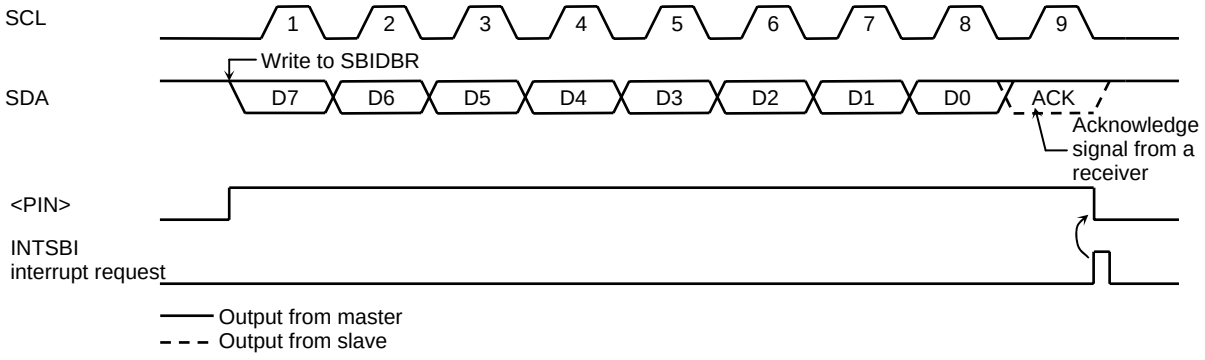


Figure 3.15.15 Example in which <BC2:0> = "000" and <ACK> = "1" in transmitter mode

When the <TRX> is "0" (Receiver mode)

When the next transmitted data is other than 8 bits, set <BC2:0> <ACK> and read the received data from SBIDBR to release the SCL line (data which is read immediately after a slave address is sent is undefined). After the data is read, <PIN> becomes "1".

Serial clock pulse for transferring new 1 word of data is defined SCL and outputs "L" level from SDA pin with acknowledge timing.

An INTSBI interrupt request then occurs and the <PIN> becomes "0", Then the TMP92CZ26A pulls down the SCL pin to the Low-level. The TMP92CZ26A outputs a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

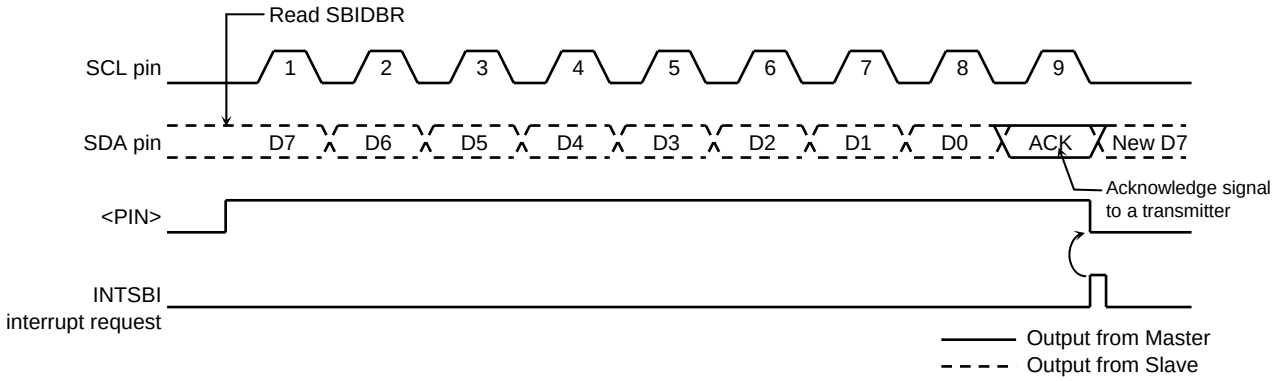


Figure 3.15.16 Example of when <BC2:0> = "000", <ACK> = "1" in receiver mode

In order to terminate the transmission of data to a transmitter, clear <ACK> to "0" before reading data which is 1-word before the last data to be received. The last data word does not generate a clock pulse as the Acknowledge signal. After the data has been transmitted and an interrupt request has been generated, set <BC2:0> to "001" and read the data. The TMP92CZ26A generates a clock pulse for a 1-bit data transfer. Since the master device is a receiver, the SDA line on the bus remains High. The transmitter interprets the High signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete.
After the one data bit has been received and an interrupt request been generated, the TMP92CZ26A generates a stop condition (see Section 3.15.6 (4) Stop condition generation) and terminates data transfer.

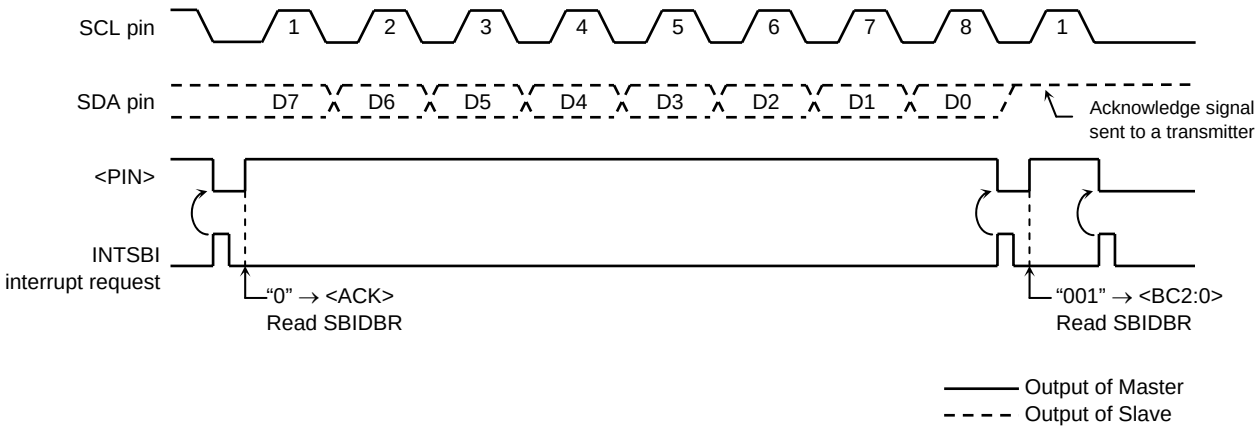


Figure 3.15.17 Termination of data transfer in master receiver mode

Example: In case receive data N times

INTSBI interrupt (After transmitting data)

		7	6	5	4	3	2	1	0	
SBICR1	←	X	X	X	X	X	X	X	X	Set the bit number of receive data and ACK.
Reg.	← SBIDBR									Load the dummy data.

End of interrupt

INTSBI interrupt (Receive data of 1st to (N−2) th)

		7	6	5	4	3	2	1	0	
Reg.	← SBIDBR									Load the data of 1st to (N−2)th.

End of interrupt

INTSBI interrupt ((N−1) th Receive data)

		7	6	5	4	3	2	1	0	
SBICR1	←	X	X	X	0	0	X	X	X	Not generate acknowledge signal
Reg.	← SBIDBR									Load the data of (N−1)th

End of interrupt

INTSBI interrupt (Nth Receive data)

		7	6	5	4	3	2	1	0	
SBICR1	←	0	0	1	0	0	X	X	X	Generate the clock for 1bit transmit
Reg.	← SBIDBR									Receive the data of Nth.

End of interrupt

INTSBI interrupt (After receiving data)

The process of generating stop condition — Finish the transmit of data

End of interrupt

Note: X: Don't care

b. If <MST> = 0 (Slave Mode)

In the slave mode the TMP92CZ26A operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, an INTSBI interrupt request occurs when the TMP92CZ26A receives a slave address or a GENERAL CALL from the master device, or when a GENERAL CALL is received and data transfer is complete, or after matching received address. In the master mode, the TMP92CZ26A operates in a slave mode if it losing arbitration. An INTSBI interrupt request occurs when a word data transfer terminates after losing arbitration. When an INTSBI interrupt request occurs the <PIN> is cleared to "0" and the SCL pin is pulled down to the Low-level. Either reading/writing from/to the SBIDBR or setting the <PIN> to "1" will release the SCL pin after taking t_{LOW} time.
Check the SBISR<AL>, <TRX>, <AAS>, and <AD0> and implements processes according to conditions listed in the next table.

Example: In case matching slave address in slave receive mode, direction bit is "1".
INTSBI interrupt
if TRX = 0
Then shift to other process
if AL = 1
Then shift to other process
if AAS = 0
Then shift to other process

		7	6	5	4	3	2	1	0	
SBICR1	←	X	X	X	1	X	X	X	X	Set the bit number of transmit.
SBIDBR	←	X	X	X	X	X	X	X	X	Set the data of transmit.

Note: X: Don't care

Table 3.15.2 Operation in the slave mode

<TRX>	<AL>	<AAS>	<AD0>	Conditions	Process
1	1	1	0	The TMP92CZ26A loses arbitration when transmitting a slave address and receives a slave address for which the value of the direction bit sent from another master is “1”.	Set the number of bits a word in <BC2:0> and write the transmitted data to SBIDBR
	0	1	0	In Salve Receiver Mode, the TMP92CZ26A receives a slave address for which the value of the direction bit sent from the master is “1”.	
		0	0	In Salve Transmitter Mode, a single word of is transmitted.	Check the <LRB> setting. If <LRB> is set to “1”, set <PIN> to “1” since the receiver win no request the data which follows. Then, clear <TRX> to “0” to release the bus. If <LRB> is cleared to “0”, set <BC2:0> to the number of bits in a word and write the transmitted data to SBIDBR since the receiver requests next data.
0	1	1	1/0	The TMP92CZ26A loses arbitration when transmitting a slave address and receives a slave address or GENERAL CALL for which the value of the direction bit sent from another master is “0”.	Read the SBIDBR for setting the <PIN> to “1” (reading dummy data) or set the <PIN> to “1”.
		0	0	The TMP92CZ26A loses arbitration when transmitting a slave address or data and terminates word data transfer.	
	0	1	1/0	In Slave Receiver Mode, the TMP92CZ26A receives a slave address or GENERAL CALL for which the value of the direction bit sent from the master is “0”.	
		0	1/0	In Slave Receiver Mode, the TMP92CZ26A terminates receiving word data.	Set <BC2:0> to the number of bits in a word and read the received data from SBIDBR.

(4) Stop condition generation

When SBISR<BB> = "1", the sequence for generating a stop condition start by writing "1" to SBICR2<MST, TRX, PIN> and "0" to SBICR2<BB>. Do not modify the contents of SBICR2<MST, TRX, PIN, BB> until a stop condition has been generated on the bus. When the bus's SCL line has been pulled Low by another device, the TMP92CZ26A generates a stop condition when the other device has released the SCL line and SDA pin rising.

```
                7 6 5 4 3 2 1 0
SBICR2    ← 1 1 0 1 1 0 0 0      Generate stop condition.
```

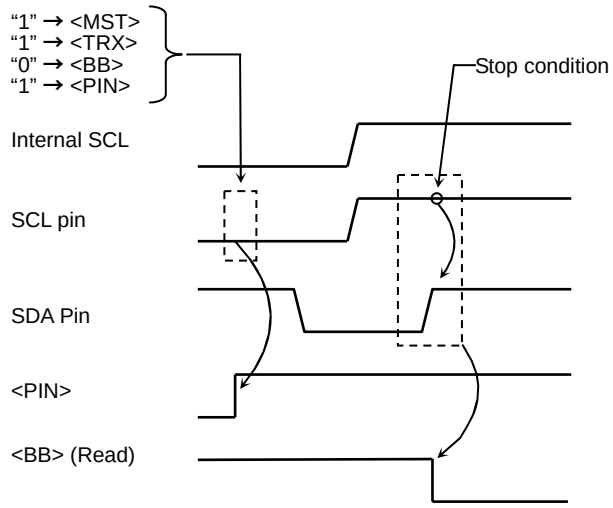


Figure 3.15.18 Stop condition generation (Single master)

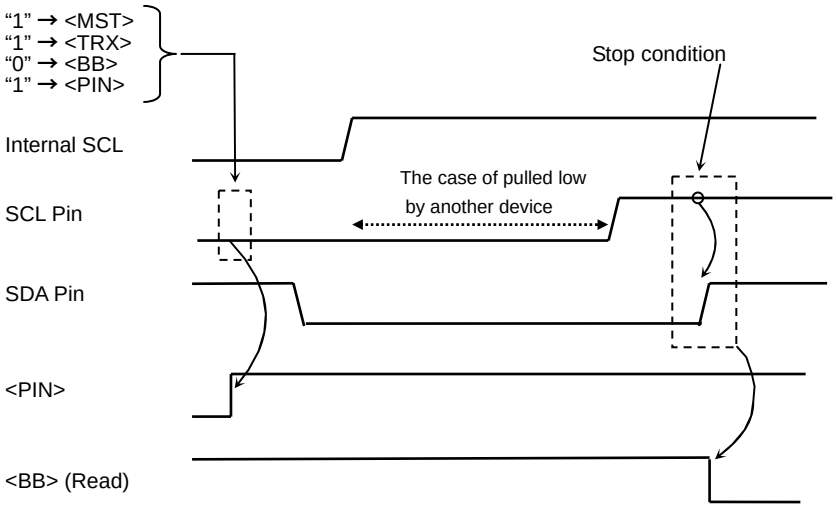


Figure 3.15.19 Stop condition generation (Multi master)

(5) Restart

Restart is used during data transfer between a master device and a slave device to change the data transfer direction.

The following description explains how to restart when the TMP92CZ26A is in Master Mode.

Clear SBICR2<MST, TRX, and BB> to 0 and set SBICR2<PIN> to 1 to release the bus. The SDA line remains High and the SCL pin is released. Since a stop condition has not been generated on the bus, other devices assume the bus to be in busy state.

And confirm SCL pin, that SCL pin is released and become bus-free state by SBISR<BB> = "0" or signal level "1" of SCL pin in port mode. Check the <LRB> until it becomes 1 to check that the SCL line on a bus is not pulled down to the low-level by other devices. After confirming that the bus remains in a free state, generate a start condition using the procedure described in (2).
In order to satisfy the set-up time requirements when restarting, take at least 4.7 μs of waiting time by software from the time of restarting to confirm that the bus is free until the time to generate the start condition.

```
                    7 6 5 4 3 2 1 0
SBICR2    ← 0 0 0 1 1 0 0 0     Release the bus
if SBISR<BB> ≠ 0                Check if SCL pin is released.
Then
if SBISR<LRB> ≠ 1               Check if SCL pin of other device is "L" level.
Then
4.7 μs Wait
SBICR1    ← X X X 1 X X X X     Set acknowledgement mode.
SBIDBR    ← X X X X X X X X     Set the slave address and direction bit.
SBICR2    ← 1 1 1 1 1 0 0 0     Generate start condition.
Note: X: Don't care
```

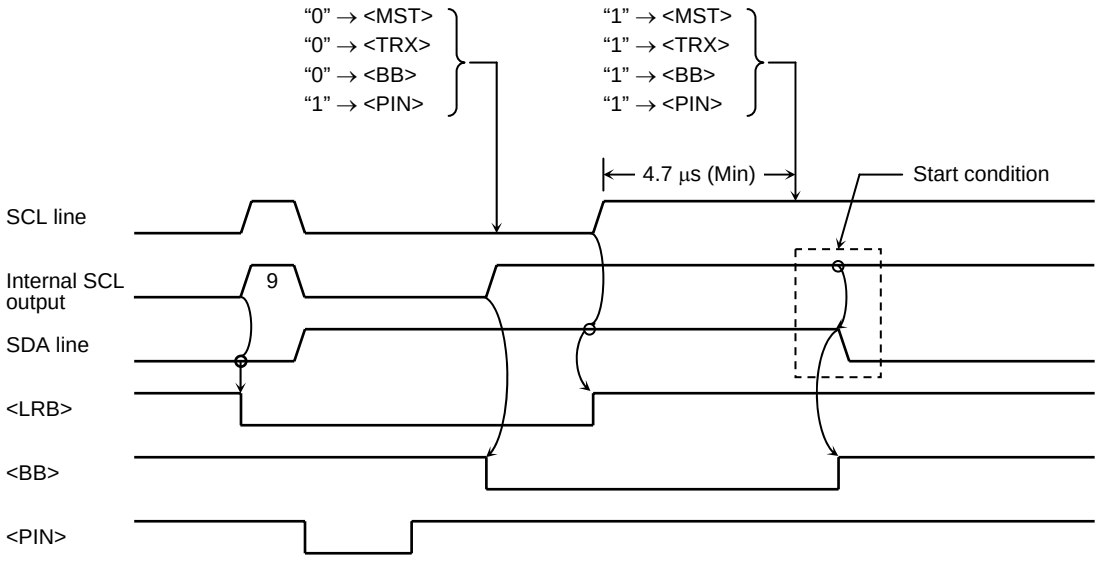


Figure 3.15.20 Timing chart for generate restart

Note: Don't write <MST> = "0", when <MST> = "0" condition. (Cannot be restarted)

3.16 USB Controller

3.16.1 Outline

This USB controller (UDC) is designed for various serial links to construct USB system. The outline is as follows:

(1) Compliant with USB rev1.1

(2) Full-speed: 12 Mbps (Not supported low-speed (1.5 Mbps))

(3) Auto bus enumeration with 384-byte descriptor RAM

(4) Supported 3 kinds of transfer type: Control, interrupt and bulk

Endpoint 0:	Control	64 bytes × 1-FIFO
Endpoint 1:	BULK (out)	64 bytes × 2-FIFO
Endpoint 2:	BULK (in)	64 bytes × 2-FIFO
Endpoint 3:	Interrupt (in)	8 bytes × 1-FIFO

(5) Built-in DPLL which generates sampling clock for receive data

(6) Detecting and generating SOP, EOP, RESUME, RESET and TIMEOUT

(7) Encoding and decoding NRZI data

(8) Inserting and discarding stuffed bit

(9) Detecting and checking CRC

(10) Generating and decoding packet ID

(11) Built-in power management function

(12) Supported dual packet mode

Note1: TMP92CZ26A don't have special terminal that control pull-up resister for D+pin. So, need to add external switch and control it.

Note2: There are some difference between our specification and USB 1.1. Refer and check "3.16.11 Notice and restrictions at first".

3.16.1.1 System Configuration

The USB controller (UDC) is consisted of following 3 blocks.

1. 900/H1 CPU I/F
2. UDC core block (DPLL, SIE, IFM and PWM), request controller, descriptor RAM and 4 endpoint FIFO
3. USB transceiver

About above “1.” is explained at 3.16.2, and “2.” is 3.16.3.

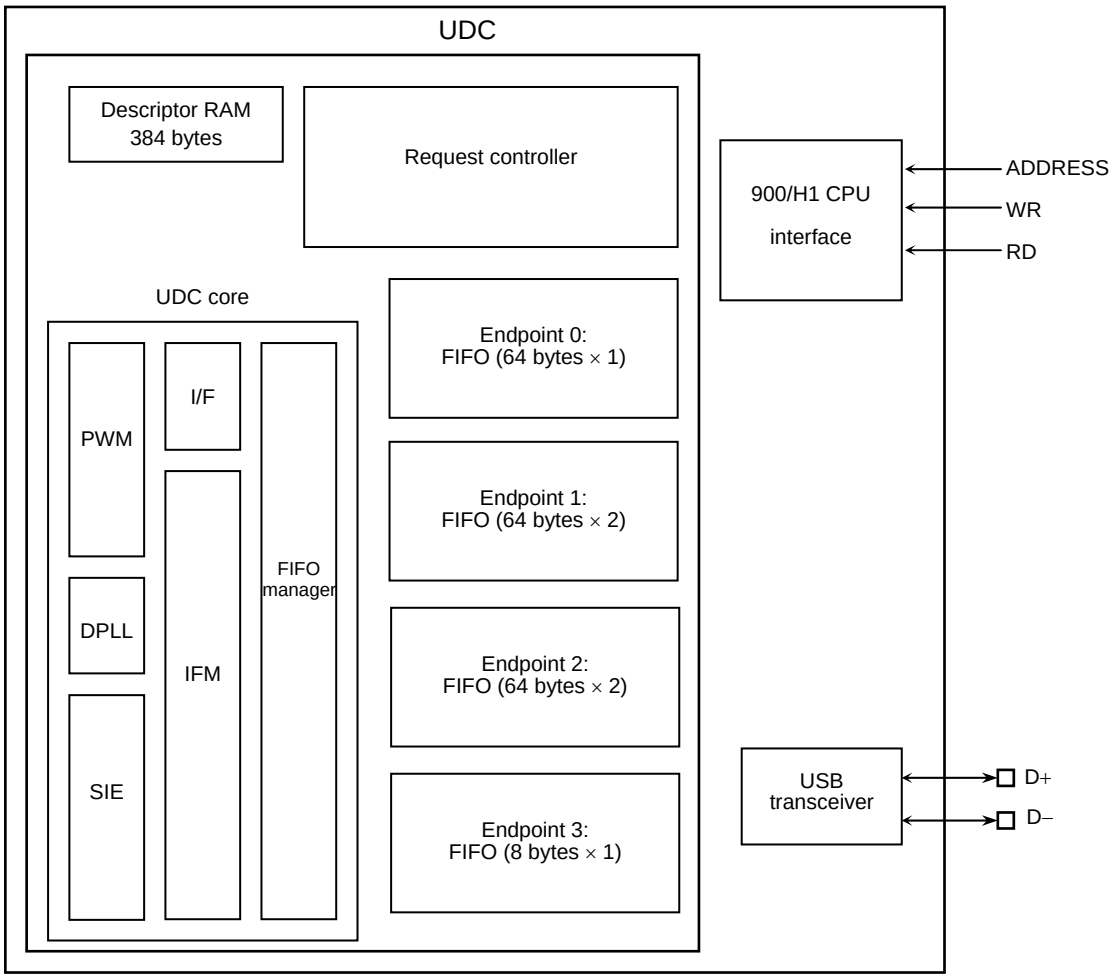
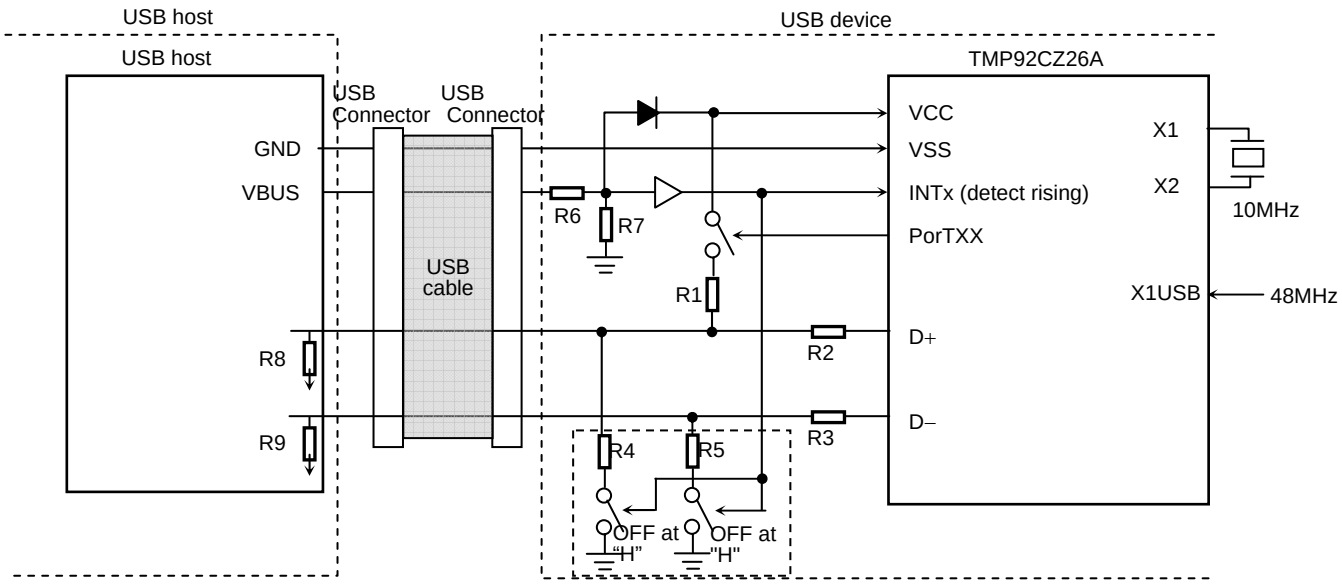


Figure 3.16.1 UDC Block Diagram

3.16.1.2 Example



If using USB controller in TMP92CZ26A, above setting is needed.

1) Pull-up of D^+ pin
   - In the USB standard, in Full Speed connection, D^+ pin must be set to pull-up. And this pull-up is needed ON/OFF control by S/W.
     Recommendation value: R1=1.5kΩ
2) Add cascade resistor of D^+, D^-signal
   - In the USB standard, in D+, D^- signal, cascade resistor must be added to each signal. Recommendation value : R2=27Ω, R3=27Ω
3) Flow current provision of the Connector connection and D^+ pin, D^- pin
   - In D^+, D^- pin of TMP92CZ26A, level must be fixed for flow current provision when not using (it is not connect to host). In this case, it is showed that method of controlling the pull-down resistor for fixed level by using detection signal of connector connection.
     Recommendation value: R4=10kΩ, R5=10kΩ
   - It is showed as example of the connector connection detection that method of detecting by using VBUS (5V voltage).

   Note: If rising of waveform is solw, recommned that likely baffering for waveform.
     Recommendation value: R6=60kΩ, R7=100kΩ
     (VBUS reducing current when suspend<500μA)
4) Connect oscillator of 10MHz to X1,X2, or input 48MHz clock to X1USB
   - If using USB by using the combination external 10MHz oscillator and internal, Stage of external hub which can be used is restricted by the precision of internal (Max 3 stages).
   - If 5 stages connection is needed for external hub, it is needed that input 48MHz clock from X1USB pin (Restriction ≤±2500ppm.)
5) HOST side pull-down resistor
   - In the USB regulation, set pull-down D^+ pin and D^- signal at USB_HOST side.
     Recommendation value: R8=15kΩ, R9=15kΩ

Note: Above connection and resistor etc, is example. Operation is not guranteed. Please confirm the newest USB standar and the operation on your setting.

3.16.2 900/H1 CPU I/F

The 900/H1 CPU I/F is a bridge between 900/H1 CPU and UDC and it mainly works following operations.

- INTUSB (interrupt from UDC) generation
- A bridge for SFR
- USB clock control (48 MHz)

3.16.2.1 SFRs

The 900/H1 CPU I/F have following SFRs to control UDC and USB transceiver.

- USB control

  USBCR1 (USB control register 1)

- USB interrupt control

  USBINTFR1 (USB interrupt flag register 1)
  USBINTFR2 (USB interrupt flag register 2)
  USBINTFR3 (USB interrupt flag register 3)
  USBINTFR4 (USB interrupt flag register 4)
  USBINTMR1 (USB interrupt mask register 1)
  USBINTMR2 (USB interrupt mask register 2)
  USBINTMR3 (USB interrupt mask register 3)
  USBINTMR4 (USB interrupt mask register 4)

Figure 3.16.2 900/H1 CPU I/F SFR

Address	Read/Write	SFR Symbol
07F0H	R/W	USBINTFR1
07F1H	R/W	USBINTFR2
07F2H	R/W	USBINTFR3
07F3H	R/W	USBINTFR4
07F4H	R/W	USBINTMR1
07F5H	R/W	USBINTMR2
07F6H	R/W	USBINTMR3
07F7H	R/W	USBINTMR4
07F8H	R/W	USBCR1

3.16.2.2 USBCR1 Register

This register is used to set USB clock enables, transceiver enable etc.

USBCR1 (07F8H)

	7	6	5	4	3	2	1	0
bit Symbol	TRNS_USE	WAKEUP					SPEED	USBCLKE
Read/Write	R/W	R/W					R/W	R/W
After reset	0	0					1	0
Function								

- TRNS_USE (Bit7)

  0: Disable USB transceiver

  1: Enable USB transceiver

  Set to “1” for TMP92CZ26A.

- WAKEUP (Bit6)

  0: –

  1: Start remote-wakeup-function

  When the remote-wakeup-function is needed, at first check the Current_Config<REMOTE WAKEUP>.

  If the <REMOTE WAKEUP> = “1” (means SUSPEND-status), write “1”, and “0” to <WAKEUP> after checking by this, remote-wakeup-function will be started.

  If the <REMOTE WAKEUP> = “0” or EP0, 1, 2, 3_STATUS<SUSPEND> = “0”, don’t write “1” to <WAKEUP>.

- SPEED (Bit1)

  1: Full speed (12 MHz)

  0: Reserved

  This bit selects USB speed.

  Set to “1” for TMP92CZ26A.

- USBCLKE (Bit0)

  0: Disable USB clock

  1: Enable USB clock

  This bit controls to supply USB clock.

  The USB clock (named “f_{USB}”: 48MHz) is generated by an internal PLL. When the USB is started to use, write “1” to <USBCLKE> after confirmed the lock up of PLL is terminated.

  And when the PLL is stopped, stop PLL after writing “0” to <USBCLKE>.

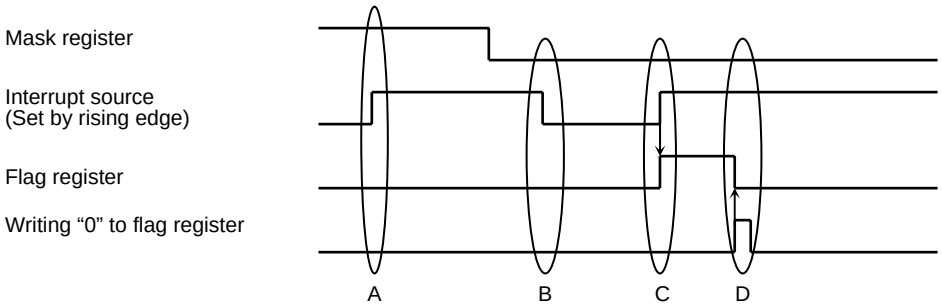
3.16.2.3 USBINTFRn, MRn Register

These SFRs control to generate INTUSB (only one interrupt to CPU) because the UDC outputs 23 interrupt source.

The USBINTMRn are mask registers and the USBINTFRn are flag registers. In the INTUSB routine, execute operations according to generated interrupt source after checking USBINTFRn.

The below is the common specification for all MASK and FLAG registers.

(Common spec for all mask and flag registers.)



A: The flag register is not set because mask register = “1”.

B: The flag register is not set because interrupt souce changes “1” → “0”.

C: The flag register is set because mask register = “0” and interrupt souce changes “0” → “1”.

D: The flag register is reset to “0” by writing “0” to flag register.

Note 1: Both “INTUSB generated number” and “bit number which is set to flag register” are not always equal. In the INTUSB interrupt routine, clear FLAG register (USBINTFRn) after checking it. The interrupt request flag, which is occurred between jump to the INTUSB interrupt routine and read flag register (USBINTFRn), is kept in interrupt controller.

Therefore, after returning from the interrupt routine, CPU jumps to INTUSB interrupt routine again. And when read the flag register (USBINTFRn), none of the bits are set to “1”. For this case, special software is needed in order not to finish as error routine.

Note 2: When USBINTMRn or USBINTFRn is written, disable INTUSB (write 00H to INTEUSB register) before it.

USBINTFR1 (07F0H)

Prohibit to read modify write

	7	6	5	4	3	2	1	0
bit Symbol	INT_URST_STR	INT_URST_END	INT_SUS	INT_RESUME	INT_CLKSTOP	INT_CLKON		
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		
After reset	0	0	0	0	0	0		
Function	When read 0: Not generate interrupt 1: Generate interrupt			When write 0: Clear flag 1: –				

Note: Above interrupts can release Halt state from IDLE2 and IDLE1 mode. (STOP mode can not be released)

*Those 6 interrupts of all 24 INTUSB sources can release Halt state from IDLE1 mode. Therefore, the system of low power dissipation can be built. However, the way of use is limited as below.

Shift to IDLE1 mode :

Execute Halt instruction when the flag of INT_SUS or INT_CLKSTOP is “1” (SUSPEND state)

Release from IDLE1 mode :

Release Halt state by the request of INT_RESUME or INT_CLKON (request of release SUSPEND)

Release Halt state by the request of INT_URST_STR or INT_URST_END (request of RESET)

- INT_URST_STR (Bit7)

  This is a flag for INT_URST_STR (“USB reset” start - interrupt).

  This is set to “1” when the UDC started to receive “USB reset” signal from USB-host.

  An application program has to initialize whole UDC by this interrupt.

- INT_URST_END (Bit6)

  This is a flag for INT_URST_END (“USB reset” end - interrupt).

  This is set to “1” when the UDC receive “USB reset end” signal from USB-host.

- INT_SUS (Bit5)

  This is a flag for INT_SUS (suspend - interrupt).

  This is set to “1” when USB change to “suspend status”.

- INT_RESUME (Bit4)

  This is a flag for INT_RESUME (resume - interrupt).

  This is set to “1” when USB change to “resume status”.

- INT_CLKSTOP (Bit3)

  This is a flag for INT_CLKSTOP (enable stopping clock supply - interrupt).

  This is set to “1” when USB enable stopping clock supply after changing to “suspend status”.

- INT_CLKON (Bit2)

  This is a flag for INT_CLKON (enabled starting clock supply - interrupt).

  This is set to “1” when USB enable starting clock supply after change to “resume status”.

USBINTFR2 (07F1H)

Prohibit to read modify write

	7	6	5	4	3	2	1	0
bit Symbol	EP1_FULL_A	EP1_Empty_A	EP1_FULL_B	EP1_Empty_B	EP2_FULL_A	EP2_Empty_A	EP2_FULL_B	EP2_Empty_B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	When read 0: Not generate interrupt 1: Generate interrupt				When write 0: Clear flag 1: –			

Note: Above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode can not be released.)

USBINTFR3 (07F2H)

Prohibit to read modify write

	7	6	5	4	3	2	1	0
bit Symbol	EP3_FULL_A	EP3_Empty_A	EP3_FULL_B	EP3_Empty_B				
Read/Write	R/W	R/W	R/W	R/W				
After reset	0	0	0	0				
Function	When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

Note: Above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode can not be released.)

- EPx_FULL_A/B:

  (When transmitting)
  This is set to “1” when CPU full write data to FIFO_A/B.

  (When receiving)
  This is set to “1” when UDC full receive data to FIFO_A/B.

- EPx_Empty_A/B:

  (When transmitting)
  This is set to “1” when FIFO become empty after transmission.

  (When receiving)
  This is set to “1” when FIFO become empty after CPU read all data from FIFO.

Note: The flag of EPx_FULL_A/B and EPx_Empty_A/B are not status flag. Therefore, check DATASET register if the FIFO-status is needed.

USBINTFR4 (07F3H)

Prohibit to read modify write

	7	6	5	4	3	2	1	0
bit Symbol	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
After reset	0	0	0	0	0	0	0	
Function	When read 0: Not generate interrupt 1: Generate interrupt				When write 0: Clear flag 1: –			

Note: Above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode can not be released.)

- INT_SETUP (Bit7)

  This is a flag for INT_SETUP (setup - interrupt).

  This is set to "1" when the UDC receive request that S/W (software) control is needed from USB host.

  By S/W (INT_SETUP routine), at first, read device request of 8-bytes from UDC and execute operation according to each request.

- INT_EP0 (Bit6)

  This is a flag for INT_EP0 (received data of the data phase for Control transfer type - interrupt).

  This is set to "1" when the UDC receive data of the data phase for Control transfer type. At the Control write transfer, data reading from FIFO is needed if this interrupt occur. At the Control read transfer, transmission data writing to FIFO is needed if this interrupts occurred.

  By host may don't assert "ACK" of last packet in the data stage. In that case, this interrupt cannot be generated. So, ignore this interrupt of after last packet data was written in the data stage because the transmission data number is specified by the host, or it depends on the capacity of the device.

- INT_STAS (Bit5)

  This is a flag for INT_STAS (status stage end - interrupt).

  This is set to "1" when the status stage end.

  If this interrupt is generated, it means that request ended normally.

  If this interrupt is not generated and INT_SETUP is generated, EP0_STATUS <STAGE_ERR> is set to "1" and it means that request didn't end normally.

- INT_STASN (Bit4)

  This is a flag for INT_STASN (change host status stage - interrupt).

  This is set to “1” when the USB host change to status stage at the Control read transfer type. This interrupt is needed if data length is less than wLength (specified by the host).

  But if the USB host change to status stage, this interrupt is always generated because of this signal is designed by using NAK of first packet. So, to avoid that this interrupt always generate, use mask register USBINTMRn. Disable this interrupt before data of last payload is written.

- INT_EPxN (Bit3, 2, 1)

  This is a flag for INT_EPxN (NAK acknowledge to the USB host - interrupt).

  This is set to “1” when the Endpoint1, 2 and 3 transmit NAK.

USBINTMR1 (07F4H)

	7	6	5	4	3	2	1	0
bit Symbol	MSK_URST_STR	MSK_URST_END	MSK_SUS	MSK_RESUME	MSK_CLKSTOP	MSK_CLKON		
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		
After reset	1	1	1	1	1	1		
Function	0: Be not masked 1: Be masked							

- MSK_URST_STR (Bit7)

  This is a mask register for USBINTFR1<INT_URST_STR>.

- MSK_URST_END (Bit6)

  This is a mask register for USBINTFR1<INT_URST_END>.

- MSK_SUS (Bit5)

  This is a mask register for USBINTFR1<INT_SUS>.

- MSK_RESUME (Bit4)

  This is a mask register for USBINTFR1<INT_RESUME>.

- MSK_CLKSTOP (Bit3)

  This is a mask register for USBINTFR1<INT_CLKSTOP>.

- MSK_CLKON (Bit2)

  This is a mask register for USBINTFR1<INT_CLKON>.

USBINTMR2 (07F5H)

	7	6	5	4	3	2	1	0
bit Symbol	EP1_MSK_FA	EP1_MSK_EA	EP1_MSK_FB	EP1_MSK_EB	EP2_MSK_FA	EP2_MSK_EA	EP2_MSK_FB	EP2_MSK_EB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	1	1	1	1	1	1	1	1
Function	0: Be not masked 1: Be masked							

- EP1/2_MSK_FA/FB/EA/EB

  This is a mask register for USBINTFR2<EPx_FULL_A/B> or <EPx_Empty_A/B>.

USBINTMR3 (07F6H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_MSK_FA	EP3_MSK_EA						
Read/Write	R/W	R/W						
After reset	1	1						
Function	0: Be not masked 1: Be masked							

- EP3_MSK_FA/FB/EA/EB:

  This is a mask register for USBINTFR3<EP3_FULL_A> or <EP3_Empty_A>.

USBINTMR4 (07F7H)

	7	6	5	4	3	2	1	0
bit Symbol	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
After reset	1	1	1	1	1	1	1	
Function	0: Be not masked 1: Be masked							

- MSK_SETUP (Bit7)

  This is a mask register for USBINTFR4<INT_SETUP>.
- MSK_EP0 (Bit6)

  This is a mask register for USBINTFR4<INT_EP0>.
- MSK_STAS (Bit5)

  This is a mask register for USBINTFR4<INT_STAS>.
- MSK_STASN (Bit4)

  This is a mask register for USBINTFR4<INT_STASN>.
- MSK_EP1N (Bit3)

  This is a mask register for USBINTFR4<INT_EP1N>.
- MSK_EP2N (Bit2)

  This is a mask register for USBINTFR4<INT_EP2N>.
- MSK_EP3N (Bit1)

  This is a mask register for USBINTFR4<INT_EP3N>.

3.16.3 UDC CORE

3.16.3.1 SFRs

The UDC CORE has following SFRs to control UDC and USB transceiver.

a) FIFO

Endpoint 0 to 3 FIFO register

b) Device request

bmRequestType	register	bRequest	register
wValue_L	register	wValue_H	register
wIndex_L	register	wIndex_H	register
wLength_L	register	wLength_H	register

c) Status

Current_Config	register	USB_STATE	register
StandardRequest	register	Request	register
EPx_STATUS	register		

d) Setup

EPx_BCS	register	EPx_SINGLE	register
Standard Request Mode	register	Request Mode	register
Descriptor RAM	register	PortStatus	register

e) Control

EPx_MODE	register	EOP	register
COMMAND	register	INT_ Control	register
Setup Received	register	USBREADY	register

f) Others

ADDRESS	register	DATASET	register
EPx_SIZE_L_A	register	EPx_SIZE_H_A	register
EPx_SIZE_L_B	register	EPx_SIZE_H_B	register
FRAME_L	register	FRAME_H	register
USBBUFF TEST	register		

Figure 3.16.3 UDC CORE SFRs (1/3)

Address	Read/Write	SFR Symbol
0500H	R/W	Descriptor RAM0
0501H	R/W	Descriptor RAM1
0502H	R/W	Descriptor RAM2
0503H	R/W	Descriptor RAM3
⋮	⋮	⋮
067DH	R/W	Descriptor RAM381
067EH	R/W	Descriptor RAM382
067FH	R/W	Descriptor RAM383
0780H	R/W	ENDPOINT0
0781H	R/W	ENDPOINT1
0782H	R/W	ENDPOINT2
0783H	R/W	ENDPOINT3
*0784H	R/W	ENDPOINT4
*0785H	R/W	ENDPOINT5
*0786H	R/W	ENDPOINT6
*0787H	R/W	ENDPOINT7
*0788H	–	Reserved
0789H	R/W	EP1_MODE
078AH	R/W	EP2_MODE
078BH	R/W	EP3_MODE
*078CH	R/W	EP4_MODE
*078DH	R/W	EP5_MODE
*078EH	R/W	EP6_MODE
*078FH	R/W	EP7_MODE
0790H	R	EP0_STATUS
0791H	R	EP1_STATUS
0792H	R	EP2_STATUS
0793H	R	EP3_STATUS
*0794H	R	EP4_STATUS
*0795H	R	EP5_STATUS
*0796H	R	EP6_STATUS
*0797H	R	EP7_STATUS
0798H	R	EP0_SIZE_L_A
0799H	R	EP1_SIZE_L_A
079AH	R	EP2_SIZE_L_A
079BH	R	EP3_SIZE_L_A
*079CH	R	EP4_SIZE_L_A
*079DH	R	EP5_SIZE_L_A
*079EH	R	EP6_SIZE_L_A
*079FH	R	EP7_SIZE_L_A
07A1H	R	EP1_SIZE_L_B
07A2H	R	EP2_SIZE_L_B
07A3H	R	EP3_SIZE_L_B
*07A4H	R	EP4_SIZE_L_B
*07A5H	R	EP5_SIZE_L_B
*07A6H	R	EP6_SIZE_L_B
*07A7H	R	EP7_SIZE_L_B
*07A8H	–	Reserved

Note: “*” is not used at TMP92CZ26A.

Figure 3.16.4 UDC CORE SFRs (2/3)

Address	Read/Write	SFR Symbol
07A9H	R	EP1_SIZE_H_A
07AAH	R	EP2_SIZE_H_A
07ABH	R	EP3_SIZE_H_A
*07ACH	R	EP4_SIZE_H_A
*07ADH	R	EP5_SIZE_H_A
*07AEH	R	EP6_SIZE_H_A
*07AFH	R	EP7_SIZE_H_A
07B1H	R	EP1_SIZE_H_B
07B2H	R	EP2_SIZE_H_B
07B3H	R	EP3_SIZE_H_B
*07B4H	R	EP4_SIZE_H_B
*07B5H	R	EP5_SIZE_H_B
*07B6H	R	EP6_SIZE_H_B
*07B7H	R	EP7_SIZE_H_B
07C0H	R	bmRequestType
07C1H	R	bRequest
07C2H	R	wValue_L
07C3H	R	wValue_H
07C4H	R	wIndex_L
07C5H	R	wIndex_H
07C6H	R	wLength_L
07C7H	R	wLength_H
07C8H	W	Setup Received
07C9H	R	Current_Config
07CAH	R	Standard Request
07CBH	R	Request
07CCH	R	DATASET1
07CDH	R	DATASET2
07CEH	R	USB_STATE
07CFH	W	EOP
07D0H	W	COMMAND
07D1H	R/W	EPx_SINGLE1
*07D1H	R/W	EPx_SINGLE2
07D3H	R/W	EPx_BCS1
*07D4H	R/W	EPx_BCS2
*07D5H	R/W	Reserved
07D6H	R/W	INT_Control
*07D7H	R/W	Reserved
07D8H	R/W	Standard Request Mode
07D9H	R/W	Request Mode
*07DAH	R/W	Reserved
*07DBH	R/W	Reserved
*07DCH	R/W	Reserved
*07DDH	R/W	Reserved
07DEH	W	ID_CONTROL
07DFH	R	ID_STATE

Note: “*” is not used at TMP92CZ26A.

Figure 3.16.5 UDC CORE SFRs (3/3)

Address	Read/Write	SFR Symbol
07E0H	R/W	Port_Status
07E1H	R	FRAME_L
07E2H	R	FRAME_H
07E3H	R	ADDRESS
*07E4H	–	Reserved
*07E5H	–	Reserved
07E6H	R/W	USBREADY
*07E7H	–	Reserved
07E8H	W	Set Descriptor STALL

Note: "*" is not used at TMP92CZ26A.

3.16.3.2 EPx_FIFO Register (x: 0 to 3)

This register is prepared for each endpoint independently.

This is the window register from or to FIFO RAM.

In the auto bus enumeration, the request controller in UDC set mode, which is defined at endpoint descriptor for each endpoint automatically. By this, each endpoint is set to voluntary direction.

Endpoint0 (0780H)

	7	6	5	4	3	2	1	0
bit Symbol	EP0_DATA7	EP0_DATA6	EP0_DATA5	EP0_DATA4	EP0_DATA3	EP0_DATA2	EP0_DATA1	EP0_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Endpoint1 (0781H)

	7	6	5	4	3	2	1	0
bit Symbol	EP1_DATA7	EP1_DATA6	EP1_DATA5	EP1_DATA4	EP1_DATA3	EP1_DATA2	EP1_DATA1	EP1_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Endpoint2 (0782H)

	7	6	5	4	3	2	1	0
bit Symbol	EP2_DATA7	EP2_DATA6	EP2_DATA5	EP2_DATA4	EP2_DATA3	EP2_DATA2	EP2_DATA1	EP2_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Endpoint3 (0783H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_DATA7	EP3_DATA6	EP3_DATA5	EP3_DATA4	EP3_DATA3	EP3_DATA2	EP3_DATA1	EP3_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Note: Read or write these window registers by using load instruction of 1 byte because of each register have only 1 byte address. Don't use load instruction of 2 bytes or 4 bytes.

The device request that received from the USB host is stored to following 8-byte registers.

The 8-byte registers are bmRequestType, bRequest, wValue_L, wValue_H, wIndex_L, wIndex_H, wLength_L and wLength_H. They are updated whenever new SETUP token is received from host...

When the UDC receive without error, INT_SETUP interrupt is asserted and it means the new device request has been received.

And there is a request which is operated automatically by UDC. It depends on received request.

In that case, the UDC don't assert INT_SETUP interrupt. A request which the UDC is operating now can be checked by reading STANDARD_REQUEST_FLAG and REQUEST_FLAG.

3.16.3.3 bmRequestType Register

This register shows the bmRequestType field of device request.

bmRequestType (07C0H)

	7	6	5	4	3	2	1	0
bit Symbol	DIRECTION	REQ_TYPE1	REQ_TYPE0	RECIPIENT4	RECIPIENT3	RECIPIENT2	RECIPIENT1	RECIPIENT0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

DIRECTION (Bit7)
0: from host to device
1: from device to host

REQ_TYPE [1:0] (Bit6 to bit5)
00: Standard
01: Class
10: Vendor
11: (Reserved)

RECEIPIENT [4:0] (Bit4 to bit0)
00000: Device
00001: Interface
00010: Endpoint
00011: etc.
Others: (Reserved)

3.16.3.4 bRequest Register

This register shows the bRequest field of device request.

bRequest (07C1H)

	7	6	5	4	3	2	1	0
bit Symbol	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

(Standard)
00000000: GET_STATUS
00000001: CLEAR_FEATURE
00000010: Reserved
00000011: SET_FEATURE
00000100: Reserved
00000101: SET_ADDRESS
00000110: GET_DESCRIPTOR
00000111: SET_DESCRIPTOR
00001000: GET_CONFIGURATION
00001001: SET_CONFIGURATION
00001010: GET_INTERFACE
00001011: SET_INTERFACE
00001100: SYNCH_FRAME

(Printer class)
00000000: GET_DEVICE_ID
00000001: GET_PORT_STATUS
00000010: SOFT_RESET

3.16.3.5 wValue Register

There are 2 registers; the wValue_L register and wValue_H register. wValue_L shows the lower-byte of wValue field of device request and wValue_H register shows upper byte.

wValue_L (07C2H)

	7	6	5	4	3	2	1	0
bit Symbol	VALUE_L7	VALUE_L6	VALUE_L5	VALUE_L4	VALUE_L3	VALUE_L2	VALUE_L1	VALUE_L0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

wValue_H (07C3H)

	7	6	5	4	3	2	1	0
bit Symbol	VALUE_H7	VALUE_H6	VALUE_H5	VALUE_H4	VALUE_H3	VALUE_H2	VALUE_H1	VALUE_H0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

3.16.3.6 wIndex Register

There are 2 registers, the wIndex_L register and wIndex_H register. the wIndex_L register shows the lower byte of wIndex field of device request and wIndex_H register shows upper byte.

These are usually used to transfer index or offset.

wIndex_L (07C4H)

	7	6	5	4	3	2	1	0
bit Symbol	INDEX_L7	INDEX_L6	INDEX_L5	INDEX_L4	INDEX_L3	INDEX_L2	INDEX_L1	INDEX_L0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

wIndex_H (07C5H)

	7	6	5	4	3	2	1	0
bit Symbol	INDEX_H7	INDEX_H6	INDEX_H5	INDEX_H4	INDEX_H3	INDEX_H2	INDEX_H1	INDEX_H0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

3.16.3.7 wLength Register

There are 2 registers, the wLength_L register and wLength_H register. the wLength_L register shows the lower-byte of wLength field of device request and wLength_H register shows upper byte.

In case of data phase, these registers show byte number to transfer.

wLength_L (07C6H)

	7	6	5	4	3	2	1	0
bit Symbol	LENGTH_L7	LENGTH_L6	LENGTH_L5	LENGTH_L4	LENGTH_L3	LENGTH_L2	LENGTH_L1	LENGTH_L0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

wLength_H (07C7H)

	7	6	5	4	3	2	1	0
bit Symbol	LENGTH_H7	LENGTH_H6	LENGTH_H5	LENGTH_H4	LENGTH_H3	LENGTH_H2	LENGTH_H1	LENGTH_H0
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

3.16.3.8 Setup Received Register

This register informs for the UDC that an application program recognized INT_SETUP interrupt.

SetupReceived (07C8H)

	7	6	5	4	3	2	1	0
bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
Read/Write	W	W	W	W	W	W	W	W
After reset	0	0	0	0	0	0	0	0

If this register is accessed by an application program, the UDC release to disabling access to EP0's FIFO RAM because the UDC recognized the device request is received.

This is to protect data stored in EP0 in the time from continuous request has been asserted to an application program recognized INT_SETUP interrupt.

Therefore, write "00H" to this register when the device request in INT_SETUP routine is recognized.

Note : When EP0_FIFO is accessed register after wrote to this register, the recovery time of 2clock at 12MHz is needed.

3.16.3.9 Current_Config Register

This register shows the present value that is set by SET_CONFIGURATION and SET_INTERFACE.

Current_Config (07C9H)

	7	6	5	4	3	2	1	0
bit Symbol	REMOTEWAKEUP		ALTERNATE[1]	ALTERNATE[0]	INTERFACE[1]	INTERFACE[0]	CONFIG[1]	CONFIG[0]
Read/Write	R		R	R	R	R	R	R
After reset	0		0	0	0	0	0	0

CONFIG[1:0] (Bit1 to bit0)

00: UNCONFIGURED	Set to UNCONFIGURED by the host.
01: CONFIGURED1	Set to CONFIGURED 1 by the host.
10: CONFIGURED2	Set to CONFIGURED 2 by the host.

INTERFACE[1:0] (Bit3 to bit2)

00: INTERFACE0	Set to INTERFACE 0 by the host.
01: INTERFACE1	Set to INTERFACE 1 by the host.
10: INTERFACE2	Set to INTERFACE 2 by the host.

ALTERNATE[1:0] (Bit5 to bit4)

00: ALTERNATE0	Set to ALTERNATE 0 by the host.
01: ALTERNATE1	Set to ALTERNATE 1 by the host.
10: ALTERNATE2	Set to ALTERNATE 2 by the host.

REMOTE WAKEUP (Bit7)

0: Disable	Disabled remote wakeup by the host.
1: Enable	Enabled remote wakeup by the host.

Note1: Config, INTERFACE and ALTERNATE each support 3 kinds (0,1 and 2).

Note2: If each request is controlled by S/W, this register is not set.

3.16.3.10 Standard Request Register

This register shows the standard request that is executing now.
A bit which is set to "1" shows present executing request.

Standard Recuest (07CAH)

	7	6	5	4	3	2	1	0
bit Symbol	S_INTERFACE	G_INTERFACE	S_CONFIG	G_CONFIG	G_DESCRIPT	S_FEATURE	C_FEATURE	G_STATUS
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

S_INTERFACE (Bit 7) : SET_INTERFACE
G_INTERFACE (Bit 6) : GET_INTERFACE
S_CONFIG (Bit 5) : SET_CONFIGRATION
G_CONFIG (Bit 4) : GET_CONFIGRATION
G_DESCRIPT (Bit 3) : GET_DESCRIPTOR
S_FEATURE (Bit 2) : SET_FEATURE
C_FEATURE (Bit 1) : CLEAR_FEATURE
G_STATUS (Bit 0) : GET_STATUS

3.16.3.11 Request Register

This register shows the device request that is executing now.
A bit which is set to "1" shows present executing request.

Request (07CBH)

	7	6	5	4	3	2	1	0
bit Symbol		SOFT_RESET	G_PORT_STS	G_DEVICE_ID	VENDOR	CLASS	ExSTANDARD	STANDARD
Read/Write		R	R	R	R	R	R	R
After reset		0	0	0	0	0	0	0

SOFT_RESET (Bit 6) : SOFT_RESET
G_PORT_STS (Bit 5) : GET_PORT_STATUS
G_DEVICE_ID (Bit 4) : GET_DEVICE_ID
VENDOR (Bit 3) : Vender class request
CLASS (Bit 2) : Class request
ExSTANDARD (Bit 1) : Not support auto Bus Enumeration (SET_DESCRIPTOR, SYNCH_FRAME)
STANDARD (Bit 0) : Standard request

3.16.3.12 DATASET Register

This register shows whether FIFO has data or not.

The application program can be checked it by accessing this register that whether FIFO has data or not.

In the receiving status, when valid data transfer from USB host finished, bit which correspond to applicable endpoint is set to “1” and generate interrupt. And, when application read data of 1-packet, this bit is cleared to “0”. In the transmitting status, when it terminated that 1-packet data transfer to FIFO, this bit is set to “1”. And when valid data is transferred to USB host, this bit is cleared to “0” and generates interrupt.

DATASET1 (07CCH)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_DSET_B	EP3_DSET_A	EP2_DSET_B	EP2_DSET_A	EP1_DSET_B	EP1_DSET_A		EP0_DSET_A
Read/Write	R	R	R	R	R	R		R
After reset	0	0	0	0	0	0		0

DATASET2 (07CDH)

	7	6	5	4	3	2	1	0
bit Symbol	EP7_DSET_B	EP7_DSET_A	EP6_DSET_B	EP6_DSET_A	EP5_DSET_B	EP5_DSET_A	EP4_DSET_B	EP4_DSET_A
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

Note: DATASET1<EP3_DSET_B>, DATASET2 registers are not used at TMP92CZ26A.

- Single packet mode
  (DATASET1: Bit0, bit2, bit4 and bit6 DATASET2: Bit0, bit2, bit4 and bit6)

  These bits show whether FIFO of applicable endpoint has data or not.

  In endpoint of receiving mode, if bit 1 of applicable endpoint is “1”, data that should be read exist to FIFO. Access EPx_SIZE register, and grasp size of data that should be read, and read data of its size. When this bit is “0”, data that should be read does not exist.

  In endpoint of transmitting mode, if bit of applicable endpoint is “0”, CPU can be transferred data under the payload. If its bit is “1”, because of FIFO have transfer waiting data, transfer data to FIFO in UDC after applicable bit was cleared to “0”. When short-packet is transferred, access EOP register after writing transmission data to applicable endpoint.

- Dual packet mode
  (DATASET1: Bit3, bit5 and bit7 DATASET2: Bit1, bit3 bit5 and bit7)

  These bits become effective in the dual packet mode. This mode has FIFO of 2-packets.

  Each packet (called packet-A, packet-B) has DATASET-bit.

  In isochroous transfer, it shows data transfer that can access in present frame the packet. This is different from above one. In this case, the bit that whether A or B is set to “1”, it is renewed according as shifting flame.

Note1: In the receiving mode, if bits that A-packet and B-packet of applicable endpoint are “1”, read data that packet-number should be received, after checking DATASIZE<PACKET_ACTIVE>.

Note2: In the transmitting mode, if the both A and B bits are not “1”, it means that there are space in FIFO. So, write data for payload or less to FIFO. If transmission become short-packet, write “0” to EOP<EPn_EOPB> after writing data to the FIFO. The maximum size that can be written to A or B packet is same with maximum payload size. If the both A and B bits are “0”, continuous writing of double maximum payload size are available.

Note3: In the dual packet transmitting mode, if both A and B packet are empty and EOP<EPn_EOPB> is written “0”, the NULL-data is set to FIFO. In the single mode, the NULL-data is also set to FIFO if the above operation is executed by A packet don’t have data state.

3.16.3.13 EPx_STATUS Register (x: 0 to 7)

These registers are status registers for each endpoint. The <SUSPEND> is common for all endpoint.

		7	6	5	4	3	2	1	0
EP0_STATUS (0790H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP1_STATUS (0791H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP2_STATUS (0792H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP3_STATUS (0793H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP4_STATUS (0794H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP5_STATUS (0795H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP6_STATUS (0796H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0
		7	6	5	4	3	2	1	0
EP7_STATUS (0797H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	After reset		0	0	1	1	1	0	0

Note: EP4, 5, 6 and 7_STATUS registers are not used TMP92CZ26A.

TOGGLE Bit (Bit6)

This bit shows status of toggle sequence bit.

0: TOGGLE Bit0
1: TOGGLE Bit1

SUSPEND (Bit5)

This bit shows status of power management of UDC.

0: RESUME
1: SUSPEND

In the SUSPEND status, some limitation about accessing to UDC is needed.

For the detail, refer 3.10.9.

STATUS [2:0] (Bit4 to bit2)

These bits show status of endpoint of UDC.

The status show whether transfer it or not, or show result of transfer. . These are depending on transfer type.

(For the Isochronous transfer type, refer 3.10.6.)

000: READY	Receiving:	Device can be received. In the endpoint 1 to 7, this register is initialized to “READY” by setting transfer type at SET_CONFIGURATION. In the endpoint 0, this register is initialized to “READY” by detecting USB reset from the host. This is initialized to “READY” by terminating the status stage without error.
	Transmitting:	Basically, this is same with “Receiving”. But in transmitting, when data for transmission is set to FIFO and answer to token from host and transfer data to host collect and received ACK, status register is not change, and it keeps “READY”. In this case, EPx_Empty_A or EPx_Empty_B interrupt show terminates transfer correctly.
001: DATAIN		UDC set to DATAIN and generates EPx_FULL_A or EPx_FULL_B interrupt when data is received from the host without error.
010: FULL		Refer 3.10.8 (2) Details for the STATUS register.
011: TX_ERR		After transfer data to IN token from host, UDC set TX-ER to status register when it is not received “ACK” from host. In this case, an interrupt is not generated. The hosts re-try and transfer IN token to this.
100: RX_ERR		UDC set RX_ERR to status register without transmitting “ACK” to host when an error (like a CRC-error) is detected in data of received token. In this case, an interrupt is not generated. The hosts re-try and transfer IN token to this.
101: BUSY		This status is used only for the control transfer type and it is set when a token of status-stage is received from the host after terminated data-stage. When status-stage can be finished, terminate correctly and returns to READY. This is not used in the Bulk and interrupts transfer type.
110: STALL		This status shows that applicable endpoint is STALL status. This status, return STALL-handshake except SETUP-token. In the control endpoint, returns to READY from stall condition when SETUP-token is received. In the other endpoint, returns to READY when initialization command of FIFO is received. (Note) In Automatically answer of Set_Interface request, request to interface 4 to 6 may not become to request error. If this is problem, in Set_Interface request answer, set Standard Request Mode <S_INTERFACE> to “1” and use software.
111: INVALID		This status shows that applicable endpoint is UNCONFIGURED status. In this status, the UDC has no reaction when token is received from the host. By reset, all endpoint set to INVALID status. Only endpoint 0 returns to READY by receiving USB-reset. Applicable endpoint returns to READY by configured.

FIFO_DISABLE (Bit1)

0: FIFO enabled
1: FIFO disabled

This bit symbol shows FIFO status except EP0.

If the FIFO is set to disabled, the UDC transmits NAK handshake forcibly for the all transfer. Disabled or enabled is set by COMMAND register. This bit is cleared to "0" when transfer type is changed.

STAGE_ERROR (Bit0)

0: SUCCESS
1: ERROR

This bit symbol shows that status stage is not terminated correctly. ERROR is set when a status stage is not terminated correctly and new SETUP token is received.

When this bit is "1", this bit is cleared to "0" by read EP0_STATUS register. This bit is not cleared even if normal control transfer or other transfer is executed after. To clear, read this bit. When software transaction is finished and UDC writes EOP register, UDC shifts to status register and waits termination of status stage. In this case, if software is needed to confirm that status stage is terminated correctly, when a new request flag is received, it can be confirmed that whether last request terminate correctly or not. And during request routine in software, when new request flag is asserted, it can be confirmed that whether last request is canceled or not halfway.

3.16.3.14 EPx_SIZE Register (x: 0 to 7)

These registers have following function.

a) In the receiving, showing data number for 1 packet which was received correctly.

b) In the transmitting, it shows payload size. But it shows length value when short packet is transferred.
This register is not needed to read when it is transmitting.

c) Showing dual packet mode and effective packet.

Each endpoint has H (High)-register that shows upper bit 9 to bit7 of data size and L (Low) register which shows lower bit 6 to bit0 and control bit of FIFO.
And each H/L register has 2-set for dual-packet mode.
By reset, these are initialized to maximum payload size.

		7	6	5	4	3	2	1	0
EP0_SIZE_L_A (0798H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP1_SIZE_L_A (0799H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_L_A (079AH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_L_A (079BH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_L_A (079CH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_L_A (079DH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_L_A (079EH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_L_A (079FH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	After reset	1	0	0	0	1	0	0	0

Note EP4,5,6,7_SIZE_L_A registers are not used at TMP92CZ26A.

		7	6	5	4	3	2	1	0
EP1_SIZE_L_B (07A1H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_L_B (07A2H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_L_B (07A3H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_L_B (07A4H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_L_B (07A5H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_L_B (07A6H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_L_B (07A7H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0

Note EP3,4,5,6,7_SIZE_L_B registers are not used at TMP92CZ26A.

		7	6	5	4	3	2	1	0
EP1_SIZE_H_A (07A9H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_H_A (07AAH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_H_A (07ABH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_H_A (07ACH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_H_A (07ADH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_H_A (07AEH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_H_A (07AFH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0

Note EP4,5,6,7_SIZE_H_A registers are not used at TMP92CZ26A.

		7	6	5	4	3	2	1	0
EP1_SIZE_H_B (07B1H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
EP2_SIZE_H_B (07B2H)		7	6	5	4	3	2	1	0
	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
EP3_SIZE_H_B (07B3H)		7	6	5	4	3	2	1	0
	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
EP4_SIZE_H_B (07B4H)		7	6	5	4	3	2	1	0
	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
EP5_SIZE_H_B (07B5H)		7	6	5	4	3	2	1	0
	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
EP6_SIZE_H_B (07B6H)		7	6	5	4	3	2	1	0
	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0
EP7_SIZE_H_B (07B7H)		7	6	5	4	3	2	1	0
	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	After reset						0	0	0

Note EP3,4,5,6,7_SIZE_H_B registers are not used at TMP92CZ26A.

DATASIZE[9:7] (H register: Bit2 to bit0)

DATASIZE[6:0] (L register: Bit6 to bit0)

In receiving, data number that 1 packet received from the host is shown. This is renewed when a data from the host is received with no error.

PKT_ACTIVE (L register: Bit7)

1: OUT_ENABLE

0: OUT_DISABLE

When dual-packet mode is selected, this bit show packet that can be accessed. In this case, the UDC accesses packets that divide FIFO (Packet A and Packet B) mutually. When FIFO in UDC is accessed by CPU, refer to this bit. If receiving endpoint, start reading from packet that this bit is “1”. In single-packet mode, this bit is no meaning because of the packet-A is always used.

3.16.3.15 FRAME Register

This register shows frame number which is issued with SOF token from the host and is used for Isochronous transfer type.

Each HIGH and LOW registers show upper and lower bits.

FRAME_L (07E1H)

	7	6	5	4	3	2	1	0
bit Symbol	–	T[6]	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
Read/Write	R	R	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0

FRAME_H (07E2H)

	7	6	5	4	3	2	1	0
bit Symbol	T[10]	T[9]	T[8]	T[7]		CREATE	FRAME_STS1	FRAME_STS0
Read/Write	R	R	R	R		R	R	R
After reset	0	0	0	0		0	1	0

T[10:7] (H register: Bit7 to bit4)

T[6:0] (L register: Bit6 to bit0)

These bits are renewed when SOF-token is received. And it shows frame-number.

CREATE (H register: Bit2)

0: DISABLE
1: ENABLE

These bits show enable function that generate SOF SOF automatically from UDC. This is used for the case of receiving error of SOF token.

This function is set by accessing COMMAND register.

By reset, this bit is initialized to "0".

FRAME STS[1:0]
(H register: Bit1 and bit0)

0: BEFORE
1: VALID
2: LOST

These bits show the status whether a frame number that is shown FRAME register is correct or not. At the LOST status, a correct frame number is undefined.

If this register is "VALID", number that is shown to FRAME register is correct.

If this register is "BEFORE", when SOF auto generation, BEFORE condition shows it from USB host controller inside that from SOF generation time to receive SOF token. Correct value as frame-number is value that is selected from FRAME register value.

3.16.3.16 ADDRESS Register

This register shows device address which is specified by the host in bus enumeration.

By reading this register, a present address can be confirmed.

ADDRESS (07E3H)

	7	6	5	4	3	2	1	0
bit Symbol		A6	A5	A4	A3	A2	A1	A0
Read/Write		R	R	R	R	R	R	R
After reset		0	0	0	0	0	0	0

ADDRESS [6:0] (Bit6 to bit0)

The UDC compares this register and address in all packet ID, and UDC judges whether it is an effective transaction or not.

This is initialized to "00H" by USB reset.

3.16.3.17 EOP Register

This register is used when a dataphase of control transfer type terminate or when a short packet is transmitting of bulk-IN, interrupt-IN.

EOP (07CFH)

	7	6	5	4	3	2	1	0
bit Symbol	EP7_EOPB	EP6_EOPB	EP5_EOPB	EP4_EOPB	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
Read/Write	W	W	W	W	W	W	W	W
After reset	1	1	1	1	1	1	1	1

Note: EOP<EP7_EOPB, EP6_EOPB, EP5_EOPB, EP4_EOPB> registers are not used at TMP92CZ26A.

In a dataphase of control transfer type, write “0” to <EP0_EOPB> when all transmission data is written to the FIFO, or read all receiving data from the FIFO. UDC is terminated status stage by this signal.

When short packet is transmitted by using bulk-IN or interrupt-IN endpoint, use this for terminate writing transmission data. In this case, write “0” to <EP0_EOPB> of writing endpoint. Write “1” to another bit.

3.16.3.18 Port Status Register

This register is used when a request of printer class is received.

In case of request of GET_PORT_STATUS, the UDC operates automatically by using this data.

Port Status (07E0H)

	7	6	5	4	3	2	1	0
bit Symbol	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
Read/Write	W	W	W	W	W	W	W	W
After reset	0	0	0	1	1	0	0	0

Note: TMP92CZ26A don't use this register because of not support to printer-class.

The data should be written before receiving request.

Write "0" to <Reserved> bit of this register. This register is initialized to "18H" by reset.

3.16.3.19 Standard Request Mode Register

This register set answer for Standard Request either answer automatically in Hardware or control in software. Each bit mean kind of request.

When this register is set applicable bit to "0", answer is executed automatically by hardware. When this register is set applicable bit to "1", answer is controlled by software. If request is received during hardware control, interrupt signal (INT_SETUP, INT_EP0, INT_STAS, INT_STAN) is set to disable. If request is received during software control, interrupt signal is asserted, and it is controlled by software.

Standard Request Mode (07D8H)

	7	6	5	4	3	2	1	0
bit Symbol	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0

S_Intetface (Bit 7) : SET_INTERFACE
G_Interface (Bit 6) : GET_INTERFACE
S_Config (Bit 5) : SET_CONFIGRATION
G_Config (Bit 4) : GET_CONFIGRATION
G_Descript (Bit 3) : GET_DESCRIPTOR
S_Feature (Bit 2) : SET_FEATURE
C_Feature (Bit 1) : CLEAR_FEATURE
G_Status (Bit 0) : GET_STATUS

3.16.3.20 Request Mode Register

This register set answer for Class Request either answer automatically in Hardware or control in software. Each bit mean kind of request.

When this register is set applicable bit to “0”, answer is executed automatically by hardware. When this register is set applicable bit to “1”, answer is controlled by software. If request is received during hardware control, interrupt signal (INT_SETUP, INT_EP0, INT_STAS, INT_STATUSN) is set to disable. If request is received during software control, interrupt signal is asserted, and it is controlled by software.

Request Mode (07D9H)

	7	6	5	4	3	2	1	0
bit Symbol		Soft_Reset	G_Port_Sts	G_DeviceId				
Read/Write		R/W	R/W	R/W				
After reset		0	0	0				

Note: TMP92CZ26A don’t use this register because of printer-class is not support automatic answer.

-	(Bit 7)	: Reserved
Soft_Reset	(Bit 6)	: SOFT_RESET
G_Port_Sts	(Bit 5)	: GET_PORT_STATUS
G_Config	(Bit 4)	: GET_DEVICE_ID
G_Descript	(Bit 3 to 0)	: Reserved

Note1: SET_ADDRESS request is supported by only auto-answer .

Note2: SET_DESCRIPTOR and SYNCH_FRAME are controlled by only software .

Note3: Vendor Request and Class Request (Printer Class and so on) are controlled by only software.

Note4: INT_SETUP, EP0, STAS and STASN interrupts assert only when it is software-control.

3.16.3.21 COMMAND Register

This register sets COMMAND at each endpoint. This register can be set selection of endpoint in bit6 to bit4 and kind of COMMAND in bit3 to bit0.

COMMAND for endpoint that is supported is ignored.

COMMAND (07D0H)

	7	6	5	4	3	2	1	0
bit Symbol		EP[2]	EP[1]	EP[0]	Command[3]	Command[2]	Command[1]	Command[0]
Read/Write		W	W	W	W	W	W	W
After reset		0	0	0	0	0	0	0

Note: When writing to this register, the recovery time of 2clock at 12MHz is needed. If writing continuously, insert dummy instruction more than 250 ns.

EP [2:0] (Bit6 to bit4)

000: Select endpoint 0
001: Select endpoint 1
010: Select endpoint 2
011: Select endpoint 3

COMMAND [3:0] (Bit3 to bit0)

0000: Reserved
0001: Reserved

0010: SET_DATA0 — This COMMAND clear toggle sequence bit of applicable endpoint (EP0 to EP3).
If this COMMAND is inputted, it set toggle sequence bit of applicable endpoint to “0” compulsively. Data toggle for transfer is renewed automatically by UDC. However, if setting toggle sequence bit of endpoint to “0” compulsively, this COMMAND execution need. If control transfer type and Isochronous transfer type, execution this COMMND don’t need because of controlling in hardware.

0011: RESET — This COMMAND reset applicable endpoint (EP0 to EP3).
If this COMMAND is inputted, applicable endpoint is initialized. CLEAR_FEATURE request stall endpoint. When this stall is cleared, execute this COMMAND. (This command doesn’t affect to transfer mode.)
This command Initialize following item.
- Clear toggle sequence bit of applicable endpoint.
- Clear STALL of applicable endpoint.
- Set to FIFO_ENABLE condition.

0100: STALL — This COMMAND set applicable endpoint to STALL (EP0 to EP3).
If STALL handshake must be return as answer for device request, execute this command.

0101: INVALID — This COMMAND set condition to prohibition using applicable endpoint (EP1 to EP3).
If UDC detect USB_RESET signal from USB host, it set all endpoint (except endpoint 0) to prohibition using it automatically. If Config and Interface are changed by device request, set endpoint that is not used to prohibit using.

0110: CREATE_SOF — This COMMAND set quasi-SOF generation function to enable (EP0).
Default is set to disable, it need using for Isochronous transfer.

0111: FIFO_DISABLE — This COMMAND set FIFO of applicable endpoint to disable (EP1 to EP3).
If this command is set from external, all of transfer for applicable endpoint returns NAK. When it is set from external if during receiving packet, this becomes valid from next token. This command doesn’t affect packet that is transferring.

1000: FIFO_ENABLE	This COMMAND set FIFO of applicable endpoint to enable (EP1 to EP3). If FIFO is set to disable by FIFO_DISABLE COMMAND, this command is used for release disable condition. If during receiving packet, this becomes valid from next token. If USB_RESET is detected from host and RESET COMMAND execute and transfer mode is set by using SET_CONFIG and SET_INTERFACE request, applicable endpoint become FIFO_ENABLE condition.
1001: INIT_DESCRIPTOR	This COMMAND is used if descriptor RAM is rewritten during operates system (EP0). If UDC detect USB_RESET from host controller, it read content of descriptor RAM automatically, and it set various setting. If descriptor RAM is changed during operates system, it must read setting again. Therefore, execute this command. Case of connects to USB host, this function start reading automatically. Therefore, don't have to execute this command.
1010: FIFO_CLEA	This COMMAND initializes FIFO of applicable endpoint (EP1 to EP3). However, EPx_STATUS<TOGGLE> is not initialized. If resetting by software, execute this COMMAND. This command Initialize following item. • Clear STALL of applicable endpoint. • Set to FIFO_ENABLE condition.
1011: STAL_CLEAR	This COMMAND clear STALL of applicable endpoint (EP1 to EP3). If clearing only STALL of endpoint, execute this COMMAND.

3.16.3.22 INT_Control Register

INT_STASN interrupt is disabled and enabled by value that is written to this register.

This is initialized to disable by external reset. When setup packet is received, it becomes to disable.

INT_Control ($\overline{07}$D6H)

	7	6	5	4	3	2	1	0
bit Symbol								Status_nak
Read/Write								R/W
After reset								0

In control read transfer, if host terminate dataphase in small data length (smaller than data length that is specified to wLength by host), device side and stage management cannot be synchronized. Therefore, INT_STASN interrupt inform that shift to status stage.

If this interrupt don't need, it can set to disable because of this interrupt is asserted every status stage.

STATUS_NAK (Bit0)

0: INT_STATSN interrupt disable
1: INT_STATSN interrupt enable

3.16.3.23 USB STATE Register

This register shows device state of present for connection with USB host.

USB STATE (07CEH)

	7	6	5	4	3	2	1	0
bit Symbol						Configured	Addressed	Default
Read/Write						R/W	R	R
After reset						0	0	1

Inside UDC, answer for each Device Request is managed by referring this bits (Configured, Addressed and Default). If transaction for SET_CONFIG request is executed by using software, write present state to this register. If host appointconfig0, this becomes Unconfigured. And returning to Addressed state is needed. Therefore, if host appoint config0, write bit2 to "0".

When Configured bit (Bit2) is written "0", Addressed bit (bit 1) is set automatically by hardware. When host appoint config value that supported by device, device must execute mode setting of each endpoint by using value that is appointed by endpoint-descriptor in the config-descriptor. After finish mode setting, set Configured bit (Bit2) to "1" before access EOP register. When this bit is set to "1", Addressed bit (Bit1) is set to "0" automatically.

Bit2 to bit0

000: Default
010: Addressed
100: Configured

3.16.3.24 EPx_MODE Register (x: 1 to 3)

This register sets transfer mode of endpoint (EP1 to EP3).

If transaction of SET_CONFIG and SET_INTERFACE are set to software control, this control must use appointed config or interface. When it is setting mode, access this register.

		7	6	5	4	3	2	1	0
EP1_MODE (0789H)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	After reset			0	0	0	0	0	0
		7	6	5	4	3	2	1	0
EP2_MODE (078AH)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	After reset			0	0	0	0	0	0
		7	6	5	4	3	2	1	0
EP3_MODE (078BH)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	After reset			0	0	0	0	0	0

There is limitation to timing that can be written.

If transaction for SET_CONFIG and SET_INTERFACE are set to software control, after received INT_SETUP interrupt, finish writing before access EOP register. This register prohibits writing when it is other timing, and it is ignored.

DIRECTION (Bit0)

0: OUT Direction of from host to device
1: IN Direction of from device to host

MODE [1:0] (Bit2 and bit1)

00: Control transfer type
01: Isochronous transfer type
10: Bulk transfer type or interrupt transfer type
11: Interrupt (No toggle)

Note: If setting endpoint that is set to Isochronous transfer mode to “no use”, after changed to Isochronous mode, set to “no use” by COMMAND register.

PAYLOAD [2:0] (Bit3, bit4 and bit5)

000: 8 bytes
001: 16 bytes
010: 32 bytes
011: 64 bytes
0100:128 bytes
0101:256 bytes
0110:512 bytes
0111:1023 bytes (Note1, 2)

Note1: Max packet size of Isochronous transfer type is 1023 bytes.

Note2: If except 8, 16, ..., 1023 was set to wMaxPacketSize of descriptor, Payload more than descriptor value is set by auto-answer of Set_Configration and Set_Interface.

Others (Bit6 and bit7) Reserved

3.16.3.25 EPx_SINGLE Register

This register sets mode of FIFO in each endpoint (SINGLE/DUAL).

EPx_SINGLE1 (07D1H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	
Read/Write	R/W	R/W	R/W		R/W	R/W	R/W	
After reset	0	0	0		0	0	0	

Note: Endpoint 3 support only SINGLE mode at TMP92CZ26A.

Bit number

0: No use
1: EP1_SINGLE
2: EP2_SINGLE
3: EP3_SINGLE
4: No use
5: EP1_SELECT
6: EP2_SELECT
7: EP3_SELECT

When EPx_SELECT bit is “1”, EPx_SINGLE bit become valid in following content.

0: DUAL mode 1: SINGLE mode

If set ting content of EPx_SINGLE bit to valid, set EPx_SELECT bit to “1”.

0: Invalid 1: Valid

3.16.3.26 EPx_BCS Register

This register set mode that access to FIFO in each endpoint.

EPx_BCS1 (07D3H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_BCS	EP2_BCS	EP1_BCS	
Read/Write	R/W	R/W	R/W		R/W	R/W	R/W	
After reset	0	0	0		0	0	0	

Bit number

0: No use
1: EP1_BCS
2: EP2_BCS
3: EP3_BCS
4: No use
5: EP1_SELECT
6: EP2_SELECT
7: EP3_SELECT

Always write “1” to EPx_BCS bit.

0: Reserved 1: CPU access

If setting content of EPx_BCS bit to valid, set EPx_SELECT bit to “1”.

0: Invalid 1: Valid

3.16.3.27 USBREADY Register

This register informs finishing writing data to descriptor RAM on UDC.
After assigned data to descriptor RAM, write “0” to bit0.

USBREADY (07E6H)

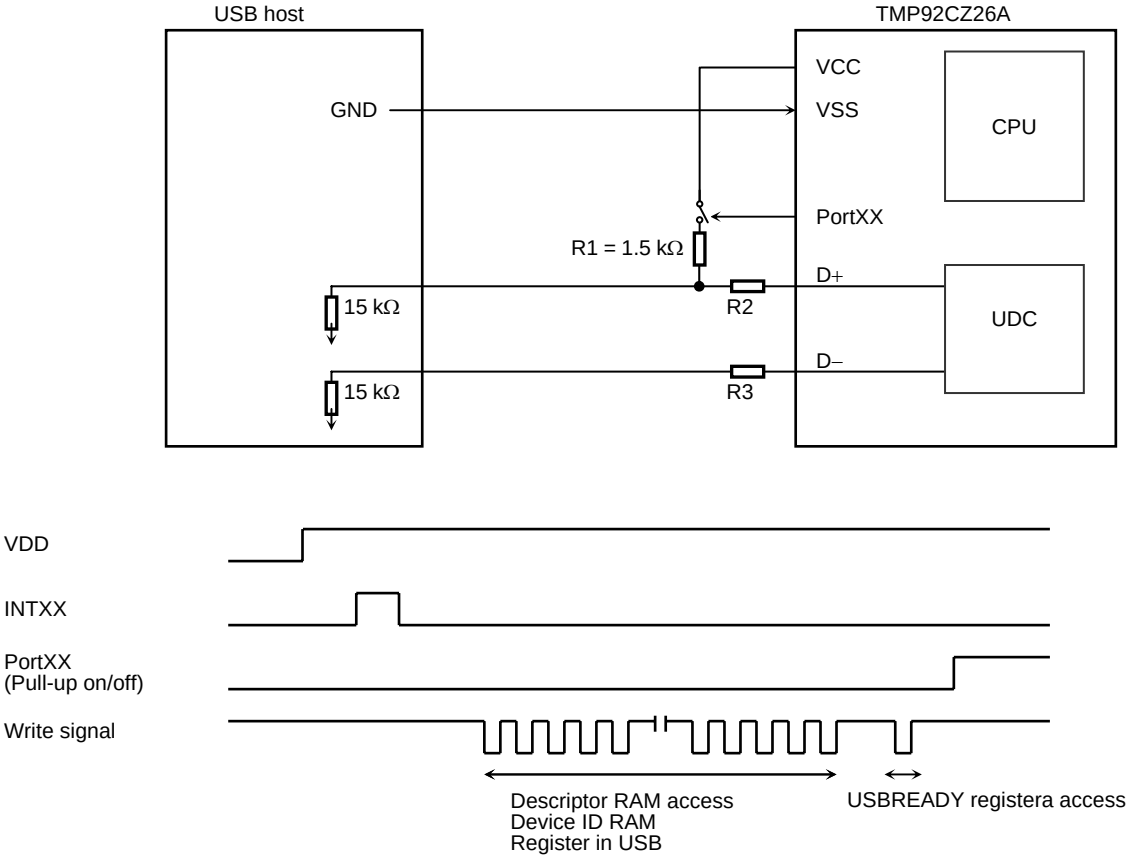
	7	6	5	4	3	2	1	0
bit Symbol								USBREADY
Read/Write								R/W
After reset								0

USBREADY (Bit0)

0: Writing to descriptor RAM was finished.

1: Writing to descriptor RAM is enable.

(However, when during connecting to host, writing to descriptor RAM is prohibited.)



Detect level of VDD signal from USB cable, and execute initialize sequence. In this case, UDC disable detecting USB_RESET signal until USBREADY register is written “0” after released USB_RESET.

If pull-up resister on D+ signal is controlled by using control signal, when pull-up resister is connected to host in OFF condition, this condition is equivalent condition with USB_RESET signal by pull-down resister in host side. Therefore UDC isn’t detected in USB_RESET until write “0” to USBREADY register

Note1: Pull-up resister and control switch are needed at external of TMP92CZ26A.

Note2: Above setting is example when communication. It is needed special circuit for prevent flow current at connector connect detection , no-use, no connection.

3.16.3.28 Set Descriptor STALL Register

This register sets whether returns STALL automatically in data stage or status stage for Set Descriptor Request.

Set Descriptor STALL (07E8H)

	7	6	5	4	3	2	1	0
bit Symbol								S_D_STALL
Read/Write								W
After reset								0

Bit0: S_D_STALL

0: Software control (Default)

1: Automatically STALL

3.16.3.29 Descriptor RAM Register

This register is used for store descriptor to RAM. Size of descriptor is 384 bytes. However, when storing descriptor, write according to descriptor RAM structure sample.

Descriptor RAM (0500H) ~ (067FH)

	7	6	5	4	3	2	1	0
bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

This register can Read/Write only following timing; before detect USB_RESET, during processing SET_DESCRIPTOR request.

SET_DESCRIPTOR request processes from INT_SETUP assert until access EOP register.

If there is rewriting request of descriptor in SET_DESCRIPTOR, process request following sequence.

1) Read descriptor that is transferred by SET_DESCRIPTOR requests every packet.
2) When reading descriptor number of last packet finished, write all descriptors to RAM for descriptor.
3) When writing finished, execute INIT_DESCRIPTOR of COMMAND register.
4) When all process finished, access EOP register, and finish status stage.
5) When INT_STAS is received, it shows normal finish of status stage.

If USB_RESET is detected, it starts reading automatically. Therefore, when it connect to host, executing of INIT_DESCRIPTOR command is not needed.

3.16.4 Descriptor RAM

This area stores descriptor that is defined in USB. Device, Config, Interface, Endpoint and String descriptor must set to RAM by using following format.

Device descriptor	18 bytes
Config 1 descriptor (Interfaces, endpoints)	Under 255 bytes
Config 2 descriptor (Interfaces, ENDPOINT)	Under 255 bytes
String0 length	1 byte
String1 length	1 byte
String2 length	1 byte
String3 length	1 byte
String0 descriptor	Under 63 bytes
String1 descriptor	Under 63 bytes
String2 descriptor	Under 63 bytes
String3 descriptor	Under 63 bytes

Note 1: If String Descriptor is supported, set StringxLength area to size0. No support String Dedcriptor is returned STALL.

Note 2: Config Descriptior refers to descriptor sample.

Note 3: Sequencer in UDC decides Config number, Interface number and Endpoint number. Therefore, if supporting Endpoint number is small, assign address according as priority.

Note 4: This function become effective only case of store descriptor as RAM.

Note 5: RAM size is total 384 bytes.

Note 6: Possible timing in RD/WR of descriptor RAM is only before detect USB_RESET and processing SET_DESCRIPTOR request. (Prohibit access except this timing.)
Writing must finish before connect to USB host and processing SET_DESCRIPTOR request.
SET_DESCRIPTOR request processes from INT_SETUP assert until access EOP register.

Descriptor RAM setting example:

Address	Data	Description	Description
Device Descriptor			
500H	12H	bLength	
501H	01H	bDescriptorType	Device Descriptor
502H	00H	bcdUSB (L)	USB Spec 1.00
503H	01H	bcdUSB (H)	IFC's specify own
504H	00H	bDeviceClass	
505H	00H	bDeviceSubClass	
506H	00H	bDeviceProtocol	
507H	08H	bMaxPacketSize0	
508H	6CH	bVendor (L)	Toshiba
509H	04H	bVendor (H)	
50AH	01H	IdProduct (L)	
50BH	10H	IdProduct (H)	
50CH	00H	bcdDevice (L)	Release 1.00
50DH	01H	bcdDevice (H)	
50EH	00H	bManufacture	
50FH	00H	IProduct	
510H	00H	bSerialNumber	
511H	01H	bNumConfiguration	
Config1 Descriptor			
512H	09H	BLength	
513H	02H	bDescriptorType	Config Descriptor
514H	4EH	wtotalLength (L)	78 bytes
515H	00H	wtotalLength (H)	
516H	01H	bNumInterfaces	
517H	01H	bConfigurationValue	
518H	00H	iConfiguration	
519H	A0H	bmAttributes	Bus powered-remote wakeup
51AH	31H	MaxPower	98 mA
Interface0 Descriptor AlternateSetting0			
51BH	09H	bLength	
51CH	04H	bDescriptorType	Interface Descriptor
51DH	00H	bInterfaceNumber	
51EH	00H	bAlternateSetting	AlternateSetting0
51FH	01H	bNumEndpoint	
520H	07H	bInterfaceClass	
521H	01H	bInterfaceSubClass	
522H	01H	bInterfaceProtocol	
523H	00H	iInterface	
Endpoint1 Descriptor			
524H	07H	bLength	
525H	05H	bDescriptorType	Endpoint Descriptor
526H	01H	bEndpointAddress	OUT
527H	02H	bmAttributes	BULK
528H	40H	wMaxPacketSize (L)	64 bytes
529H	00H	wMaxPacketSize (H)	
52AH	00H	bInterval	

Address	Data	Description	Description
Interface0 Descriptor AlternateSetting1			
52BH	09H	bLength	
52CH	04H	bDescriptorType	Interface Descriptor
52DH	00H	bInterfaceNumber	
52EH	01H	bAlternateSetting	AlternateSetting1
52FH	02H	bNumEndpoints	
530H	07H	bInterfaceClass	
531H	01H	bInterfaceSubClass	
532H	02H	bInterfaceProtocol	
533H	00H	iInterface	
Endoint1 Descriptor			
534H	07H	bLength	
535H	05H	bDescriptorType	Endpoint Descriptor
536H	01H	bEndpointAddress	OUT
537H	02H	bmAttributes	BULK
538H	40H	wMaxPacketSize (L)	64 bytes
539H	00H	wMaxPacketSize (H)	
53AH	00H	bInterval	
Endpoint2 Descriptor			
53BH	07H	bLength	
53CH	05H	bDescriptorType	Endpoint Descriptor
53DH	82H	bEndpointAddress	IN
53EH	02H	bmAttributes	BULK
53FH	40H	wMaxPacketSize (L)	64 bytes
540H	00H	wMaxPacketSize (H)	
541H	00H	bInterval	
Interface0 Descriptor AlternateSetting2			
542H	09H	bLength	
543H	04H	bDescriptorType	Interface Descriptor
544H	00H	bInterfaceNumber	
545H	02H	bAlternateSetting	AlternateSetting2
546H	03H	bNumEndpoints	
547H	FFH	bInterfaceClass	
548H	00H	bInterfaceSubClass	
549H	FFH	bInterfaceProtocol	
54AH	00H	iInterface	
Endpoint1 Descriptor			
54BH	07H	bLength	
54CH	05H	bDescriptorType	Endpoint Descriptor
54DH	01H	bEndpointAddress	OUT
54EH	02H	bmAttributes	BULK
54FH	40H	wMaxPacketSize (L)	64 bytes
550H	00H	wMaxPacketSize (H)	
551H	00H	bInterval	
Endpoint2 Descriptor			
552H	07H	bLength	
553H	05H	bDescriptorType	Endpoint Descriptor
554H	82H	bEndpointAddress	IN
555H	02H	bmAttributes	BULK
556H	40H	wMaxPacketSize (L)	64 bytes
557H	00H	wMaxPacketSize (H)	
558H	00H	bInterval	

Address	DATA	Description	Description
Endpoint3 Descriptor			
559H	07H	bLength	
55AH	05H	bDescriptorType	Endpoint Descriptor
55BH	83H	bEndpointAddress	IN
55CH	03H	bmAttributes	Interrupt
55DH	08H	wMaxPacketSize (L)	8 bytes
55EH	00H	wMaxPacketSize (H)	
55FH	01H	bInterval	1 ms
String Descriptor Length Setup Area			
560H	04H	bLength	Length of String Descriptor0
561H	10H	bLength	Length of String Descriptor1
562H	00H	bLength	Length of String Descriptor2
563H	00H	bLength	Length of String Descriptor3
String Descriptor0			
564H	04H	bLength	
565H	03H	bDescriptorType	String Descriptor
566H	09H	bString	Language ID 0x0409
567H	04H	bString	
String Descriptor1			
568H	10H	bLength	
569H	03H	bDescriptorType	String Descriptor
56AH	00H	bString	(Toshiba)
56BH	54H	bString	T
56CH	00H	bString	
56DH	6FH	bString	o
56EH	00H	bString	
56FH	73H	bString	s
570H	00H	bString	
571H	68H	bString	h
572H	00H	bString	
573H	69H	bString	i
574H	00H	bString	
575H	62H	bString	b
576H	00H	bString	
577H	61H	bString	a
String Descriptor2			
String Descriptor3			

3.16.5 Device Request

3.16.5.1 Standard request

UDC support automatically answer in standard request.

(1) GET_STATUS Request

This request returns status that is appointed of receive side, automatically.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B 10000001B 10000010B	GET_STATUS	0	0 Interface endpoint	2	Device, interface or endpoint status

Request to device returns following information according to priority of little endian.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote wakeup	Self power
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- Remote wakeup — It returns present remote wakeup setting. This bit is set or reset by SET_FEATURE or CLEAR_FEATURE request. Default is value that is set to bmAttributes field in Config descriptor.
- Self power — It returns present power supply setting. This bit return Self or Bus Power according to value that is set to bmAttributes field in Config descriptor.

Request to interface returns 00H of number of 2 bytes.

Request to endpoint returns in according to priority of little endian following information.

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

- HALT — It return halts status of endpoint that is selected.

(2) CLEAR_FEATURE request

This request clears or disables particular function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	CLEAR_ FEATURE	Feature selector	0 Interface endpoint	0	None

- Reception side device
  - Feature selector: 1 Present remote wakeup setting is disabled.
  - Feature selector: except 1 STALL state
- Reception side interface
  - STALL state
- Reception side end point
  - Feature selector: 0 Halt of applicable endpoint is cleared.
  - Feature selector: except 0 STALL state

Note: If it request to endpoint that is not exist, it stall.

(3) SET_FEATURE request

This request set or enables particular function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	SET_ FEATURE	Feature selector	0 Interface endpoint	0	None

- Reception side device
  - Feature selector: 1 Present remote wakeup setting is disabled.
  - Feature selector: except 1 STALL state
- Reception side interface
  - STALL state
- Reception side end point
  - Feature selector: 0 Halt of applicable endpoint
  - Feature selector: except 0 STALL state

Note: If it request to endpoint that is not exist, it stall.

(4) SET_ADDRESS request

This request set device address. Following request answer by using this device address.

Answer of request is used present device address until status stage of this request finish normally.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ADDRESS	Device Address	0	0	None

(5) GET_DESCRIPTOR request

This request returns appointed descriptor.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ DESCRIPTOR	Descriptor type and Descriptor index	0 or Language ID	Descriptor length	Descriptor

- Device Device transmits device descriptor that is stored to descriptor RAM.

- Config Config transmits config descriptor that is stored to descriptor RAM.

  At this point, it transmits not only config descriptor but also interface and endpoint descriptor.

- String String transmits string descriptor of index that is appointed lower byte of wValue field.

Note: Decriptor of short data length in wLength and descriptor length is transmitted by automatically answer of Get_Descriptor.

(6) SET_DESCRIPTOR request

This request sets or enables particular function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ Descriptor	Descriptor type and Descriptor index	0 or Language ID	Descriptor length	Descriptor

Automatically answer of this request does not support.

According to INT_SETUP interrupt, if receiving request was discerned as SET_DESCRIPTOR request, take back data after it confirmed EP0_DSET_A bit of DATASET register is “1”. When finishing, access EOP register, and write “0” to EP0_EOPB bit. Therefore, status stage finish. Transaction is same with vendor request.

Pleas refer to vendor request section.

(7) GET_CONFIGURATION request

This request returns configuration value of present device.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ CONFIG	0	0	1	Configuration value

If it is not configured, it returns “0”. If configuration, it returns configuration value.

(8) SET_CONFIGURATION request

This request sets device configuration.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ CONFIG	Configuration value	0	0	None

It configured in value that is appointed by using lower byte of wValue field.

When this value is “0”, it is not configured.

(9) GET_INTERFACE request

This request returns AlternateSetting value that is set by appointed interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000001B	GET_ INTERFACE	0	Interface	1	Alternate setting

If there is not appointed interface, it become to STALL state.

(10) SET_INTERFACE request

This request selects AlternateSetting in appointed interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000001B	SET_ INTERFACE	Alternate setting	Interface	0	None

If there is not appointed interface, it become STALL state.

(11) SYNCH_FRAME request

This request transmits synchronous frame of endpoint.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000010B	SYNCH_FRAME	0	Endpoint	2	Frame No.

Automatically answer of this request does not support.

According to INT_SETUP interrupt, if receiving request was discerned as SYNCH_FRAME request, write data of 2byte in Frame No after it confirmed EP0_DSET_A bit of DATASET register is “0”. When finishing, access EOP register, and write “0” to EP0_EOPB bit. Therefore, status stage finish. It can be used only in case of endpoint support isochronous transfer type and support this request. Transaction is same with vendor request.

Pleas refer to vendor request section.

3.16.5.2 Printer Class Request

UDC does not support “Automatic answer” of printer class request.

Transaction for Class request is the same as vendor request; answering to INT_SETUP interrupt.

3.16.5.3 Vendor request (Class request)

UDC doesn't support “Automatic answer” of Vendor request.

According to INT_SETUP interrupt, access register that device request is stored, and discern receiving request. If this request is vendor request, control UDC from external, and execute transaction for Vendor request.

Below is explanation for case of data phase is transmitting (Control read), and case of data phase is receiving (Control write).

(a) Control Read request

bmRequestType	bRequest	wValue	wIndex	wLength	Data
110000xxB	Vender peculiar	Vender peculiar	Vender peculiar	Vender peculiar (Expire 0)	Vendor data

When INT_SETUP is received, judge contents of receiving request by bmRequestType, bRequest, wValue, wIndex and wLength registers. And execute transaction for each request. As application, access Setup_Received register after request was judged. And it must inform that INT_SETUP interrupt was recognized to UDC.

After transmitting data prepared in application, access DATASET register, and confirm EP0_DSET_A bit is “0”. After confirming, write data FIFO of endpoint 0. If transmitting data more than payload, write data after it confirmed whether a bit of EP0_DSET_A in DATASET register is “0”. (INT_ENDPOINT0 interrupt is can be used.) If writing all data finished, write “0” to EP0 bit of EOP register. When UDC receive it, status stage finish automatically.

And when UDC finish status stage normally, INT_STATUS interrupt is asserted. If finishing status stage normally is recognized to external application, manage this stage by using this interrupt signal. If status stage cannot be finished normally and during status stage, maybe new SETUP token is received. In this case, when INT_SETUP interrupt signal is asserted, “1” is set to STAGE_ERROR bit of EP0_STATUS register. And it informs it to external that status stage cannot be finished normally.

And maybe dataphase finish in data number that is short than value showed to wLength by protocol of control read transfer type in USB. If application program is configured by using only wLength value, transaction for it cannot be when host shift to status stage without arriving at expecting data number. At this point, shifting to status stage can be confirmed by using INT_STATUSNAK interrupt signal. (However, releasing mask of STATUS_NAK bit by using interrupt control register is needed.) In Vendor Request, this problem will not generate because of receiving buffer size is set to host controller by driver, actually.

Note: In every host, data (data that is transmitted from device by payload of 8 bytes) may be recognized to short packet until confirming payload size of device side. And it may become to above case on the exterior. Therefore, if controlling standard request by using software, be careful.)

(b) Control write/request

There is no dataphase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor peculiar	Vendor peculiar	Vendor peculiar	0	None

When INT_SETUP is received, judge contents of receiving request by bmRequestType, bRequest, wValue, wIndex, wLength registers. And execute transaction for each request. As application, access Setup_Received register after request was judged. And it must inform that INT_SETUP interrupt was recognized to UDC. If transaction of application finished, write "0" to EP0 bit of EOP register. When UDC receive it, status stage finish automatically.

There is dataphase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor peculiar	Vendor peculiar	Vendor peculiar	Vendor peculiar (Except for 0)	Vendor data

When INT_SETUP is received, judge contents of receiving device request by bmRequestType, bRequest, wValue, wIndex, wLength registers. And execute transaction for each request. As application, access Setup_Received register after request was judged. And it must inform that INT_SETUP interrupt was recognized to UDC.

After receiving data prepared in application, access DATASET register, and confirm EP0_DSET is "1". After confirming, read data FIFO of endpoint 0. If receiving data more than payload, write data after it confirmed whether a bit of EP0_DSET_A in DATASET register is "1". (INT_ENDPOINT0 interrupt is can be used.) If reading all data finish, write "0" to EP0 bit of EOP register. When UDC receive it, status stage finished automatically.

And when UDC finish status stage normally, INT_STATUS interrupt is asserted. If finishing status stage normally is recognized to external application, manage this stage by using this interrupt signal. If status stage cannot be finished normally and during status stage, maybe new SETUP token is received. In this case, when INT_SETUP interrupt signal is asserted, "1" is set to STAGE_ERROR bit of EP0_STATUS register. And it informs it to external that status stage cannot be finished normally.

Below is control flow in UDC watch from application.

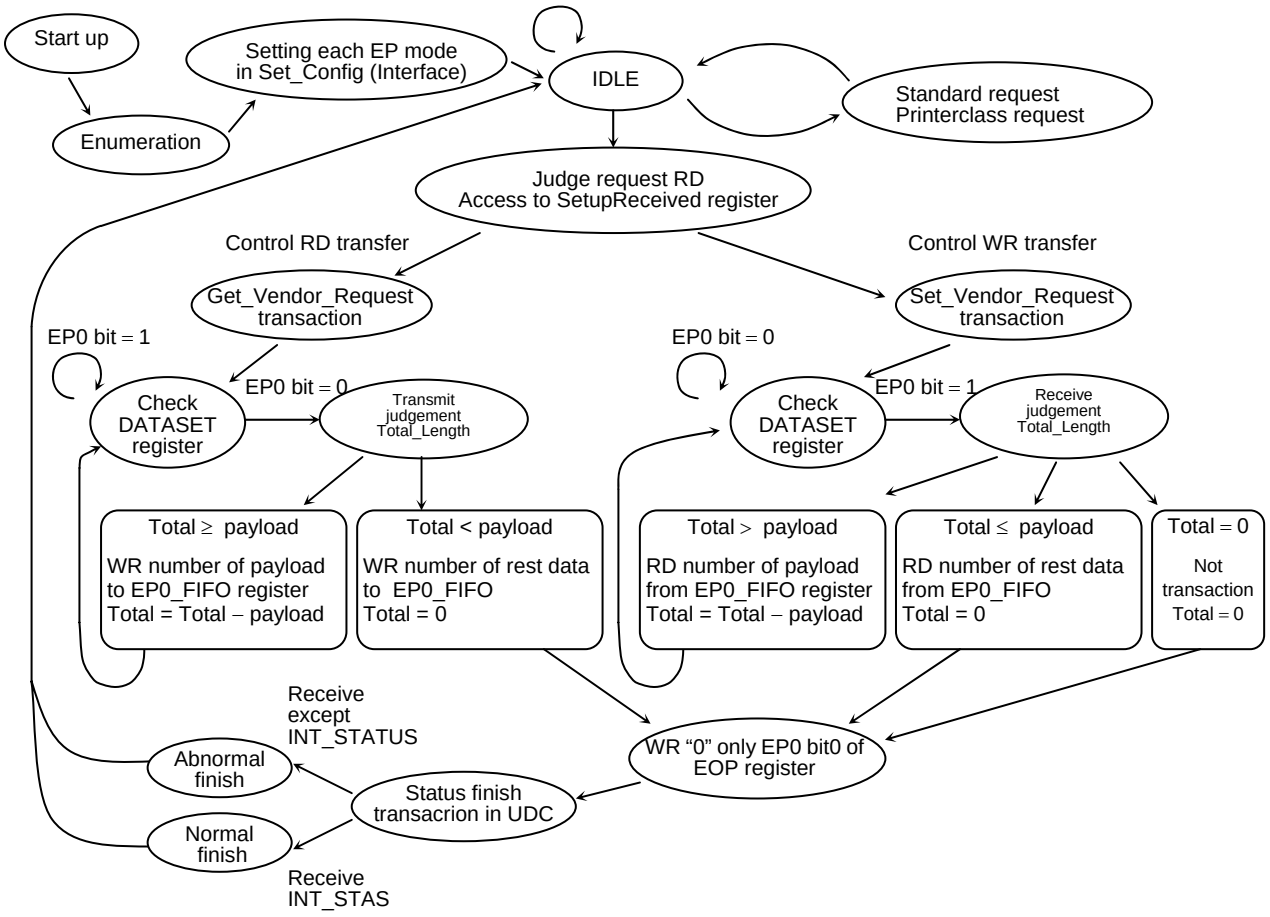


Figure 3.16.6 Control Flow in UDC Watch from Application

Note 1: There is not special case in this flow such as overlap receive SETUP packet.
Please refer to chaptor 4.5.2.3.

Note 2: This flow shows various request. However, transaction can be divided every each interrupt.

3.16.6 Transfer mode and Protocol Transaction

UDC perform automatically in hardware as follows;

- Receive packet
- Judge address endpoint transfer mode
- Error process
- Confirm toggle bit CRC of data receiving packet
- Generate including toggle bit CRC of data transmitting packet
- Handshake answer

(1) Protocol outline

Format of USB packet is showed to below. This is processed during transmission and receiving by hardware into UDC.

- SYNC field

  This field always exists first of each packet, and input data and internal CLK is synchronized in UDC.

- Packet identification field (PID)

  This field follows on SYNC field at every USB packet. UDC judge PID type and judge transfer type by decoding this cord.

- Address field

  UDC confirms whether this function was appointed or not from host by using this field. UDC compares with address that was set to ADDRESS register. If an address accords with it, UDC continues process. If an address doesn't accord, UDC ignores this token.

- Endpoint field

  If sub-channels more than two is needed in field of 4 bits, it decides it function. UDC can be supported endpoint except for control endpoint (max 7 endpoint). Token for endpoint that is permitted is ignored.

- Frame number field

  Field of 11 bits is added +1 at every frame by host. This field follows to SOF token that is transmitted in first of each frame, and frame number is appointed. UDC reads content of this field when SOF token is received, and it sets frame number to FRAME register.

- Data field

  This field is data of unit byte in 0 to 1023 bytes. When receiving it, UDC transfers only part of this data to FIFO, after CRC was confirmed, interrupt signal is asserted. And UDC informs finishing transferring data to FIFO. When transmitting, following IN token, data of FIFO is transferred. Finally, data CRC field is attached.

- CRC function

  Token is attached 5 bits, data is attached CRC of 15 bits. UDC compares CRC of received data with attached CRC automatically. When transmission, CRC is generated automatically and it is transmitted. This function may be compared by various transfer modes.

(2) Transfer mode

UDC support transfer mode in FULL speed.

- FULL speed device

  Control transfer type

  Interrupt transfer type

  Bulk transfer type

  Isochronous transfer type

Following is explanation of UDC operation in each transfer mode.

Explanation of data flow is explanation until FIFO.

(a) Bulk transfer type

Bulk transfer type warrants transferring no error between host and function by using detect error and retry. Basically, 3 phases (token, data and handshake are used) are used. However, if flow control and STALL condition, data phase is changed to hand shake phase, and it become to 2 phases. UDC holds status of every each endpoint, and it control flow control in hardware. Each endpoint condition can be confirmed by using EPx_STATUS register.

(a-1) Transmission bulk mode

Below is transaction format of bulk transfer during transmitting.

- Token: IN
- Data: DATA0/DATA1, NAK, STALL
- Handshake: ACK

Control flow

Below is control-flow when UDC receive IN token.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether conform applicable endpoint transfer mode with IN token. If it doesn't conform, state return to IDLE.
2. Condition of EPx_STATUS register is confirmed.
   - INVALI condition: State return to IDLE.
   - STAL condition: Stall handshake is returned and state return to IDLE.

   FIFO condition is confirmed, if data number of 1 packet is not prepared, NAK handshake is returned, and state return to IDLE.

   If data number of 1 packet is prepared to FIFO, it shifts to 3.
3. Data packet is generated.

   Data packet generated by using toggle bit register in UDC.

   Next, it transfers data from FIFO of internal UDC to SIE, and data packet is generated. At this point, it confirms transferred data number. And if there is more than max payload size of each endpoint, bit stuff error is generated, and finish transfer. And STATUS becomes to STALL.
4. CRC bit (counted transfer data of FIFO from first to last) is attached to last.
5. When ACK handshake from host is received,
   - Clear FIFO.
   - Clear DATASET register.
   - Renew toggle bit, and prepare for next.
   - Set STATUS to READY.

   UDC finishes normally. FIFO can be received next data.

   If it is time out without receiving ACK from host,
   - Set STATUS to TX_ERR.
   - Put back addles pointer of FIFO.

   Execute above setting. And wait next retry keeping FIFO data.

   This flow is Figure 3.16.7.

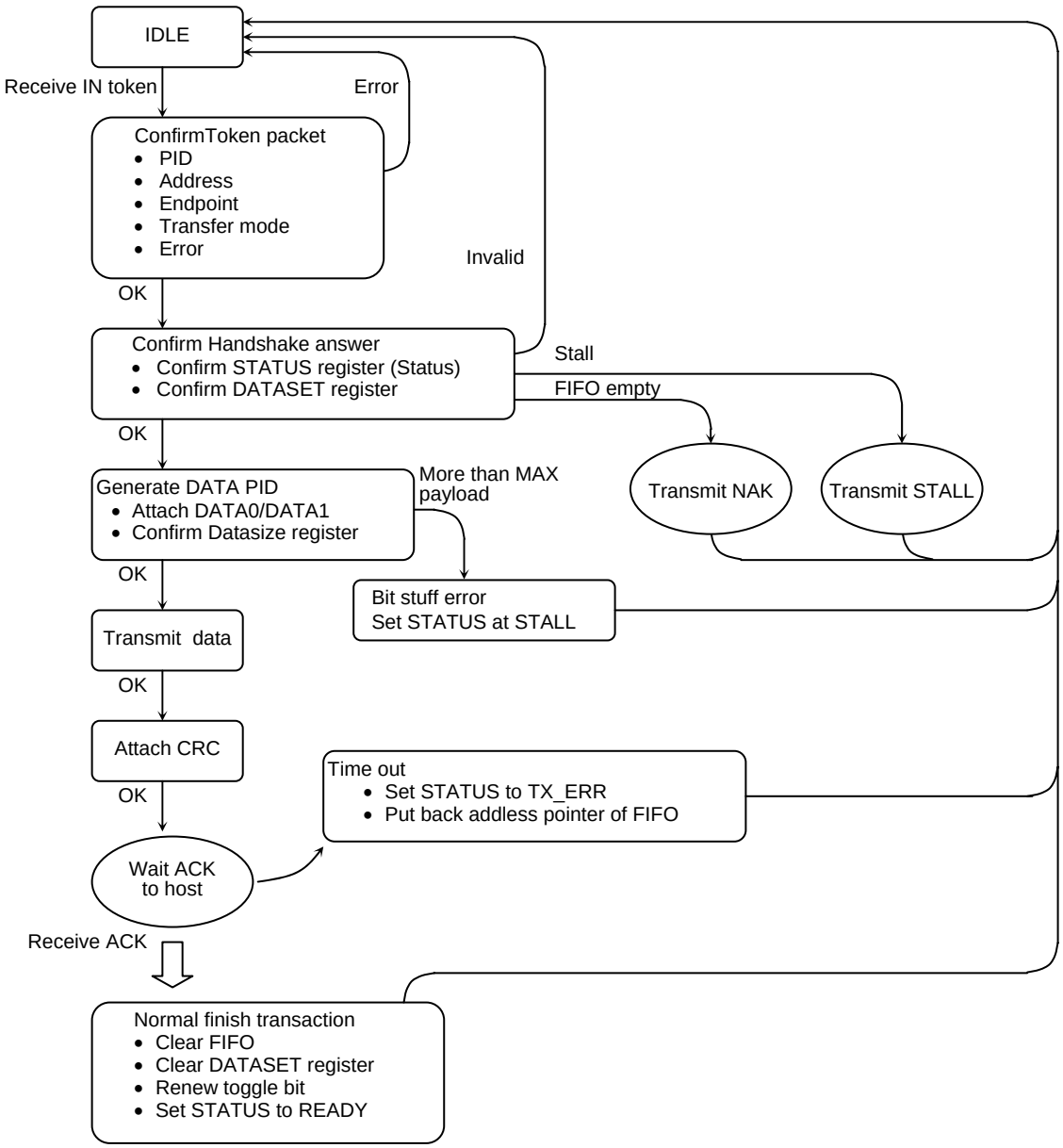


Figure 3.16.7 Control Flow in UDC (Bulk transfer type (transmission)/Interrupt transfer type (transmission))

(a-2) Receiving bulk mode

Below is transaction format receiving bulk transfer type. It has to follow below.

- Token: OUT
- Data: DATA0/DATA1
- Handshake: ACK, NAK, STALL

Control flow

Below is control-flow when UDC receive IN token.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether conform applicable endpoint transfer mode with OUT token. If it doesn't conform, state return to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State return to IDLE.
   - STALL condition: When dataphase finish, stall handshake is returned and state return to IDLE, and data is canceled.

   FIFO condition is confirmed, if data number of 1 packet is not prepared, present transferred data is canceled, NAK handshake is returned after dataphase, and state return to IDLE.
3. Data packet is received.
   Data is transferred from SIE of internal UDC to FIFO. At this point, it confirms transferred data number. And if there is more than max payload size of each endpoint, STATUS become to STALL and state return to IDLE. ACK handshake doesn't return.
4. After last data was transferred, and compare counted CRC with transferred CRC. If it doesn't conform, it sets STATUS to RX_ERR and state return to IDLE. At this point it doesn't return ACK.
   After retry, when next data is received normally, STATUS changes to DATIN. If it doesn't accord data toggle, it was judged don't take ACK in last loading. And now loading is regarded retry of last loading and data cancel. Set STATUS as RX_ERR, return to host and return IDLE. FIFO address pointer returns. And it can be received next data.
5. If CRC compare with toggle and it finished normally, ACK handshake is returned. Bellow is process in UDC.
   - Set transfer data number to DATASIZE register.
   - Set DATASET register.
   - Renew toggle bit, and prepare for next.
   - Set STATUS to READY.

   UDC finishes normally.

   This flow is Figure 3.16.8.

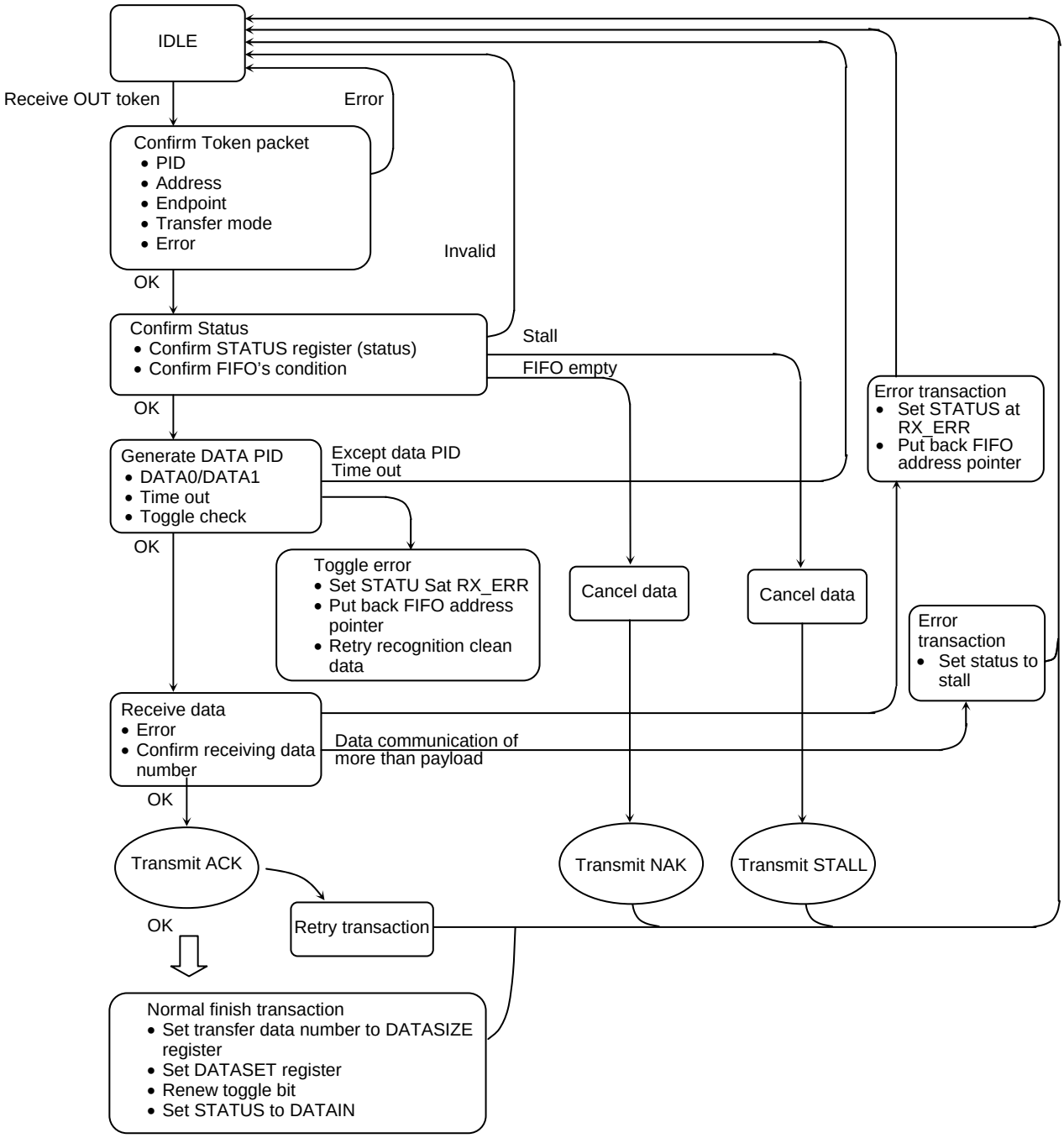


Figure 3.16.8 Control Flow in UDC (Bulk transfer type (Receiving))

(b) Interrupt transfer type

Interrupt transfer type use transaction format same with transmission bulk transfer.

When transmission by using toggle bit, hardware setting and answer in UDC are same with transmission bulk transfer. Interrupt transfer can be transferred without using toggle bit. In this case, if ACK handshake from host is not received, toggle bit is renewed, and finish normally. UDC clears FIFO for next transfer.

(b-1) Interrupt transmitting mode (Toggle mode)

UDC operation is same with bulk transmission mode. Please refer to section (a).

(b-2) Interrupt transmission mode (Not toggle mode)

This is same bulk transmission mode basically. However, if ACK handshake from host is not received, transaction is different.

After transmit data packet,

When ACK handshake from host is received,

- Clear FIFO.
- Clear DATASET register.
- Renew toggle bit and prepare for next.
- Set STATUS to READY.

UDC finishes normally by above transaction. FIFO can be received next data.

If it is time out without receiving ACK from host,

- Clear FIFO.
- Clear DATASET register.
- Renew toggle bit and prepare for next.
- Set STATUS to TX_ERR.

Execute above setting. This setting is same with except STATUS.

(c) Control transfer type

Control transfer type is configured in below three stages.

- Setup stage
- Data stage
- Status stage

Data stage is skipped sometimes. Each stage is configured in one or plural transaction. UDC executes each transaction while managing of three stages in hardware. Control transfer type has below 3 type by whether there is data stage or not, or direction.

- Control read transfer type
- Control write transfer type
- Control write transfer type (Not data stage)

3-transfer sequences are shown in Figure 3.16.10, Figure 3.16.11 and Figure 3.16.12.

UDC answers automatically about standard request in hardware. Class request, vendor request have to intervening CPU on controlling UDC.

Below is control flow in UDC and control flow in intervening CPU.

(c-1) Setup stage

Setup stage is same with transmission bulk transaction except case of token ID become to SETUP.

However, control flow in UDC differ it.

- Token: SETUP
- Data: DATA 0
- Handshake: ACK

Control flow

Below is control flow in UDC when SETUP token is received.

1. SETUP token packet is received and address, endpoint number and error are confirmed. And it checks whether applicable endpoint is the control transfer mode.
2. STATUS register state is confirmed.

State return to IDLE only it is INVALID state.

In bulk transfer mode, receiving data is enabled by STATUS registers value and FIFO condition. However, in SETUP stage, STATUS is returned to READY and accessing from CPU to FIFO is prohibited always, and internal FIFO of endpoint 0 is cleared. And it prepares for following dataphase.

If CPU accesses Setup Received registers in UDC, it recognizes as Device request is received, and accessing from CPU to EP0 is enabled.

There is this function for receiving it if new request is received in during present device request is not finishing normally.

3. Data packet is received.

   Device request of 8 bytes from SIE in UDC is transferred to below request register.

   - bmRequestType register
   - bmRequest register
   - wValue register
   - wIndex register
   - wLength register

4. After last data was transferred, and compare counted CRC with transferred CRC. If it doesn't conform, it sets STATUS to RX_ERR and state return to IDLE. At this point it doesn't return ACK, and host retry.
5. If CRC compare with toggle and it finish normally, ACK handshake is returned to host. Bellow is process in UDC.
   - Receiving device request is judged whether software control or hardware control, if request need control in software, request is informed receiving to external by asserting INT_SETUP interrupt. If using hardware, INT_SETUP interrupt is not asserted.
   - According to stage control flow, prepare for next stage.
   - Set STATUS to DATAIN.
   - Set toggle bit to "1".

Setup stage finishes by above.

This flow is Figure 3.16.6.

8-byte data that is transferred by this SETUP stage is device request.

CPU must process correspond it device request.

UDC detects following contents only from data of 8 bytes, and it manages stage in hardware.

- There is data stage or not
- Data stage direction

It judges control read transfer type, control write transfer type, control write transfer type (not data phase) by them.

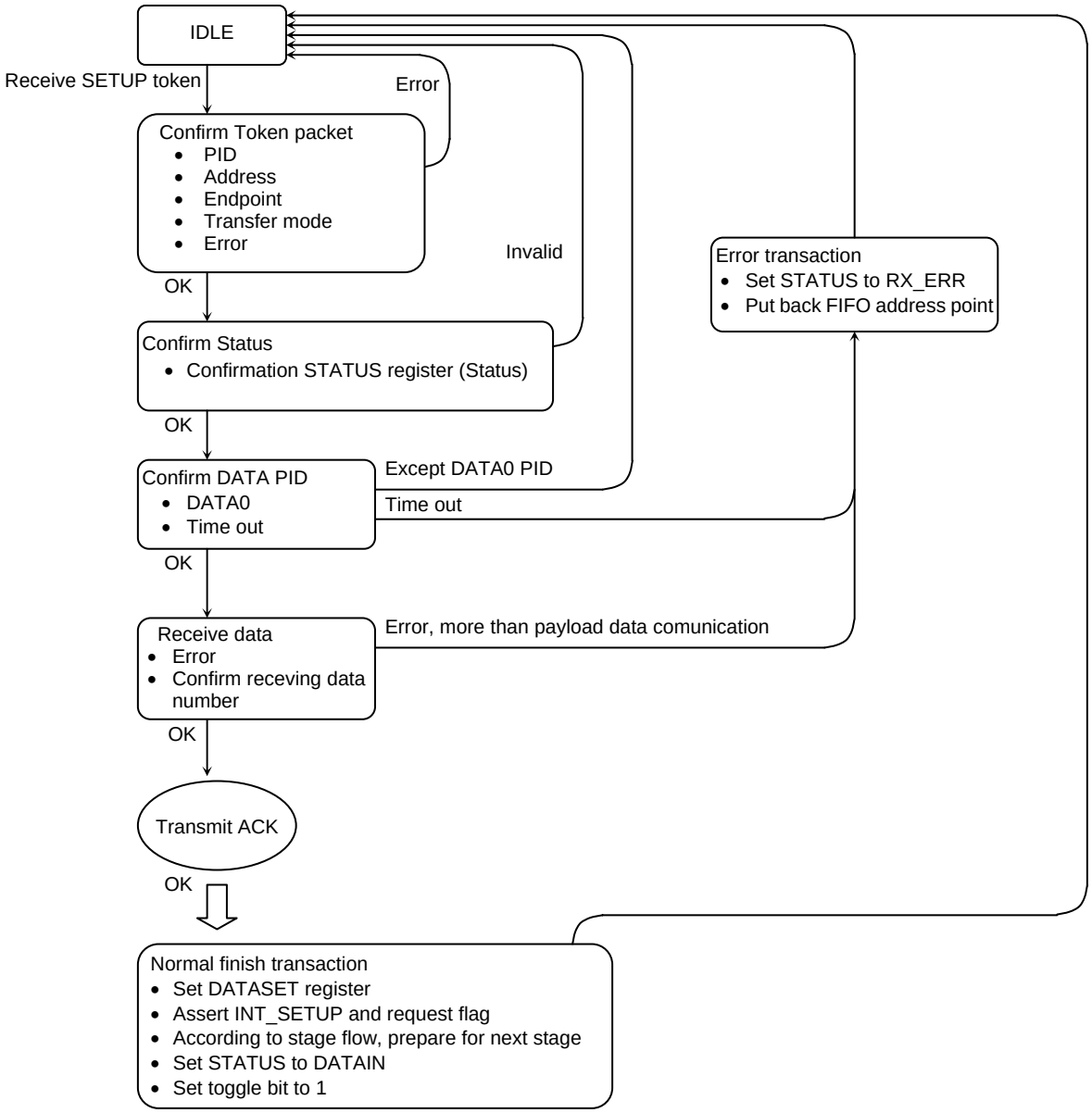


Figure 3.16.9 Control Flow in UDC (Setup stage)

(c-2) Data stage

Data stage is configured by one or plural transaction base on toggle sequence.

Transaction is same with format transmission or receiving bulk transaction.

However, below is difference.

- Toggle bit start from "1" by SETUP stage.
- It judges whether right or not by comparing IN and OUT token with direction bit of device request. If token that direction is reverse was received, it is recognized as status stage.
- INT_ENDPOINT0 interrupt is asserted.

(c-3) Status stage

Status stage is configured 0-data-length packet with DATA1's PID and handshake behinds IN or OUT token. It uses transaction that direction different with preceding stage.

Combination is below.

- Control read transfer type: OUT
- Control write transfer type: IN
- Control write transfer type (not dataphase): IN

UDC processes status stage base of control flow in control transfer type. At this point, CPU must write "0" to EP0 bit of EOP register in last transaction for status stage finish normally.

Below is detail of status stage.

(c-3-1) IN status stage

Below is IN status stage transaction format.

- Token: IN
- Data: DATA1 (0 data length), NAK, STALL
- Handshake: ACK

Control flow

Below is transaction flow of IN status stage in UDC.

1. Token packet is received and address, endpoint number and error are confirmed. If it doesn't conform, state return to IDLE. If status stage is enabled base on stage control flow in UDC, advance next stage.
2. STATUS register state is confirmed.
   - INVALID condition: State return to IDLE.
   - STALL condition: Stall handshake is returned and state return to IDLE.

   It confirm whether EOP register is accessed or not by external. If it is not accessing, NAK handshake is returned for continue control transfer. And state return to IDLE.
3. If EOP register is accessed was confirmed, 0-data-length data packet and CRC are transmitted.

4. If ACK handshake from host is received,
   - Set STATU to READY.
   - Assert INT_STATUS interrupt.

It finishes normally by above transaction.

If it is time out without receiving ACK from host,

- Set STATUS register to TX_ERR and state return IDLE. And wait restring status stage.

At this point, if new SETUP stage is started without status stage finish normally, UDC sets error to STATUS register.

(c-3-2) OUT status stage

Below is transaction format of OUT status stage.

- Token: OUT
- Data: DATA1 (0 data length)
- Handshake: ACK, NAK, STALL

Control flow

Below is transaction flow of OUT status stage in UDC.

1. Token packet is received and address, endpoint number and error are confirmed. If it doesn't conform, state return to IDLE. If status stage is enabled base on stage control flow in UDC, advance next stage.
2. STATUS register state is confirmed.
   - INVALID condition: State return to IDLE.
   - STALL condition: Data is cleared, stall handshake is returned, and state return to IDLE.

   It confirm whether EOP register is accessed or not by external. If it is not accessing, NAK handshake is returned for continue control transfer. And state return to IDLE.
3. If EOP register is accessed was confirmed, 0-data-length data packet and CRC are received.
4. If there is not error in data, ACK handshake is transmitted to host.
   - Set STATUS to READY.
   - Assert INT_STATUS interrupt.

It finishes normally by above transaction.

If there is error in data, ACK handshake is not returned.

- Set RX_ERR to STATUS register and return to IDLE. It waits retrying status stage.

At this point, if new SETUP stage is started without status stage finish normally, UDC sets error to STATUS register. Sequence of this protocol refers to section supplement.

(c-4) Stage management

UDC manages each stage of control transfer by hardware.

Each stage is changed by receiving token from USB host, or CPU accesses register. Each stage in control transfer type has to process combination software. UDC detect following contents from 8-byte data in SETUP stage. (It contents is showed to following.) And, stage is managed by judging control transfer type.

- There is data stage or not
- Data stage direction

Control read transfer type is jugged control write transfer type, control write transfer type (No data stage) by them.

Below are various conditions for changing stage in control transfer.

If receiving token for next stage from host before switching next stage from state of internal UDC, NAK handshake is returned and BUSY is informed to USB host. In all control transfer type, if SETUP token is received from host always, present transaction is stopped, and it switches SETUP stage in UDC. CPU receive new INT_SETUP even if it is processing previous control transfer.

Stage change condition of control read transfer type

1. Receive SETUP token from host
   - Start setup stage in UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt to external.
   - Change data stage into the UDC.
2. Receive IN token from host
   - CPU receive request from request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register for inform that recognized INT_SETUP interrupt to UDC.
   - According to Device request, monitor EP0 bit of DATASET register, and write data to FIFO.
   - If UDC is set data of payload to FIFO or CPU set short packet transfer in EOP register, EP0 bit of DATASET register is set.
   - UDC transfers data that is set to FIFO to host by IN token interrupts.
   - When CPU finish transaction, it writes “0” to EP0 bit of EOP register.
   - Change status stage in UDC.
3. Receive OUT token from host.
   - Return ACK to OUT token, and state change to IDLE in UDC.
   - Assert INT_STATUS interrupt to external.

These changing conditions are shown in Figure 3.16.10.

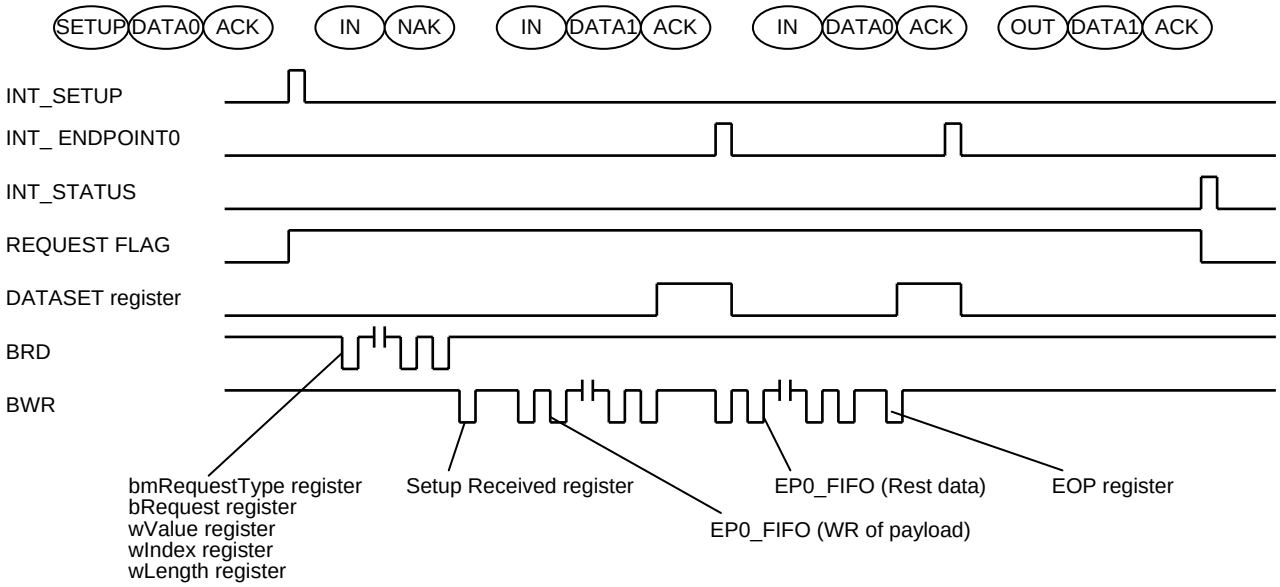


Figure 3.16.10 The Control Flow in UDC (Control Read Transfer Type)

Stage change condition of control write transfer type

1. Receive SETUP token from host.
   - Start setup stage in UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt to external.
   - Change data stage in UDC.
2. Receive OUT token from host.
   - CPU receive request from request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register for inform that recognized INT_SETUP interrupt to UDC.
   - Receive dataphase data normally, and set EP0 bit of DATASET register.
   - CPU receives data in FIFO by setting DATASET.
   - CPU process receiving data by device request.
   - When CPU finish transaction, it writes "0" to EP0 bit of EOP register.
   - Change status stage in UDC.
3. Receive IN token from host.
   - Return data packet of 0 data to IN token, and state change to IDLE in UDC.
   - Assert INT_STATUS interrupt to external when receive ACK for 0 data packet.

These changing conditions are shown in Figure 3.16.11.

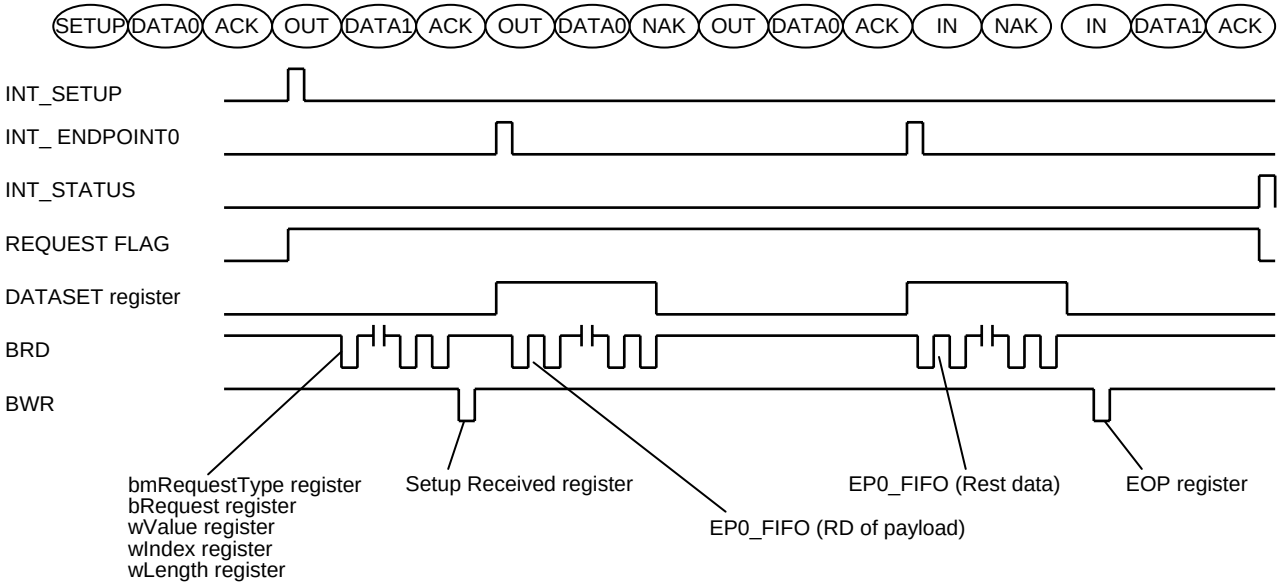


Figure 3.16.11 The Control Flow in UDC (Control Write Transfer Type)

In control read transfer type, transaction number of data stage do not always accord with data number that is apppointed by device request. Therefore, CPU can be processd by using INT_STATUSNAK interrupt. However, when class and vendor request is used, be accord wLength value with data transfer number in data phase. By this setting, using this interrupt is not need. Data stage data can be confirmed by accessing DATASIZE register.

Stage change condition of control write (no data stage) transfer type

1. Receive SETUP token from host
   - Start setup stage in UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt to external.
   - Change data stage in UDC.
2. Receive IN token from host
   - CPU receive request from request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register for inform that recognized INT_SETUP interrupt to UDC.
   - CPU process receiving data by device request.
   - When CPU finish transaction, it writes “0” to EP0 bit of EOP register.
   - Change status stage in UDC.
   - Return data packet of 0 data to IN token, and state change to IDLE in UDC.
   - Assert INT_STATUS interrupt to external when receive ACK for 0 data packet.

These change condition is Figure 3.16.12.

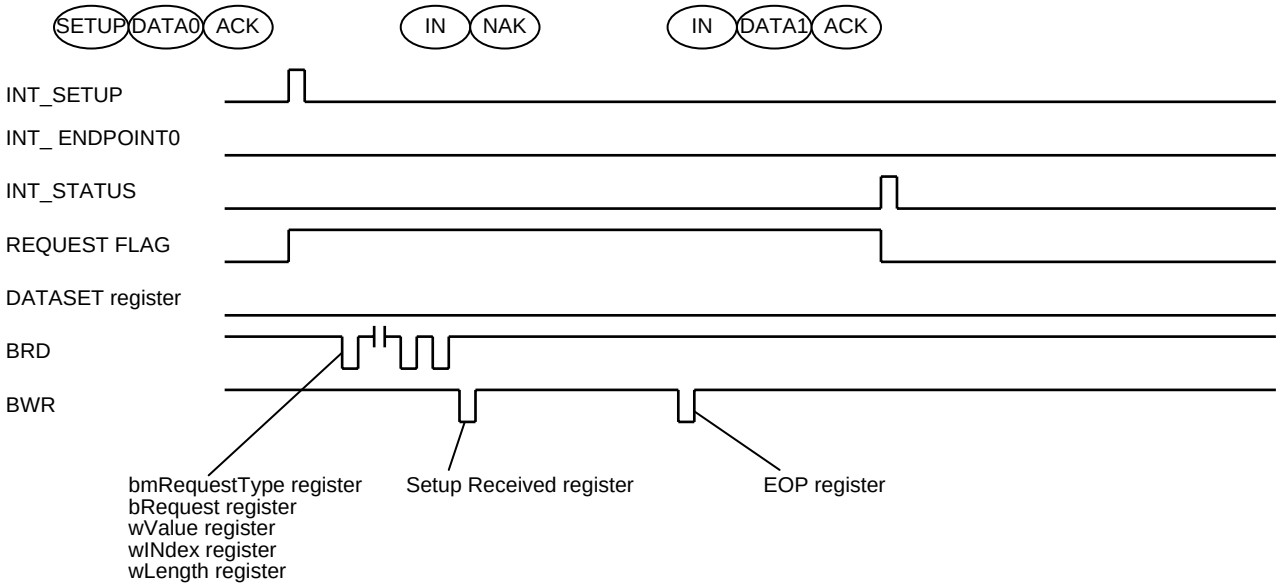


Figure 3.16.12 The Control Flow in UDC (Control Write Transfer Type not Dataphase)

(d) Isochronous transfer type

Isochronous transfer type is guaranteed transfer by data number that is limited every each frame.

However, this transfer don't retry when error occurs. Therefore, Isochronous transfer type transfer only 2 phases (token, data) and it doesn't use handshake phase. And data PID for data phase is DATA0 always because of this transaction doesn't support toggle sequence. Therefore, UDC doesn't confirm when data PID is receiving mode.

Isochronous transfer type process data every frame. Therefore, all transaction for finish transfer use receiving SOF token. UDC use FIFO that is divided into two in Isochronous transfer type.

(d-1) Isochronous transmission mode

Isochronous transfer type format in transmitting is below transaction format.

- Token: IN
- Data: DATA0

Control flow

Isochronous transfer type is frame management. And data that write to FIFO in endpoint is transmitted by IN token in next frame.

Below are two conditions in FIFO of Isochronous transmission mode transferring.

X. FIFO for storing data that transmits to host in present frame (DATASET register bit = 1)

Y. FIFO for storing data for transmitting host in next frame (DATASET register bit = 0)

FIFO that is divided into two (packet A and packet B) conditions is whether X condition or Y condition. Below flow is explained as X Condition (packet A), Y Condition (packet B) in present frame.

X and Y conditions change one after the other by receiving SOF.

Below is control flow in UDC when receiving IN token.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether conform applicable endpoint transfer mode with IN token. If it doesn't conform, state return to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State return to IDLE.
3. Data packet is generated.

   Data packet is generated. At this point, data PID attach DATA0 always. Next, data is transferred from FIFO (X condition) of packet A in UDC to SIE. And it generate DATA packet.
4. CRC bit (counted transfer data of FIFO from first to last) is attached to last.

5. Below is transaction when SOF token from host is received.
   - Change the packet A's FIFO from X Condition to Y Condition. And clear data.
   - Change the packet B from Y Condition to X Condition.
   - Set frame number to frame register.
   - Assert SOF and inform that frame is incremented to external.
   - DATASET register clears packet A bit and it sets packet B bit arrangement loading in present frame.
   - Set STATUS to READY.

   UDC finishes normally by above transaction.

   Packet A's FIFO can be received next data.

In renewed frame, Packet A's FIFO interchange packet B's FIFO, and transaction is used same flow.

If SOF token is not received by error and so on, this data is lost because of frame is not renewed. Nothing problem in receiving PID and if frame data is received with CRC error, USB sets LOST to STATUS on FRAME register, and frame number is not renewed. However, in this case, SOF is asserted and FIFO condition is renewed. If SOF token is received without transmit and transfer Isochronous in frame, UDC clears FIFO (X Condition) and sets STATUS to FULL.

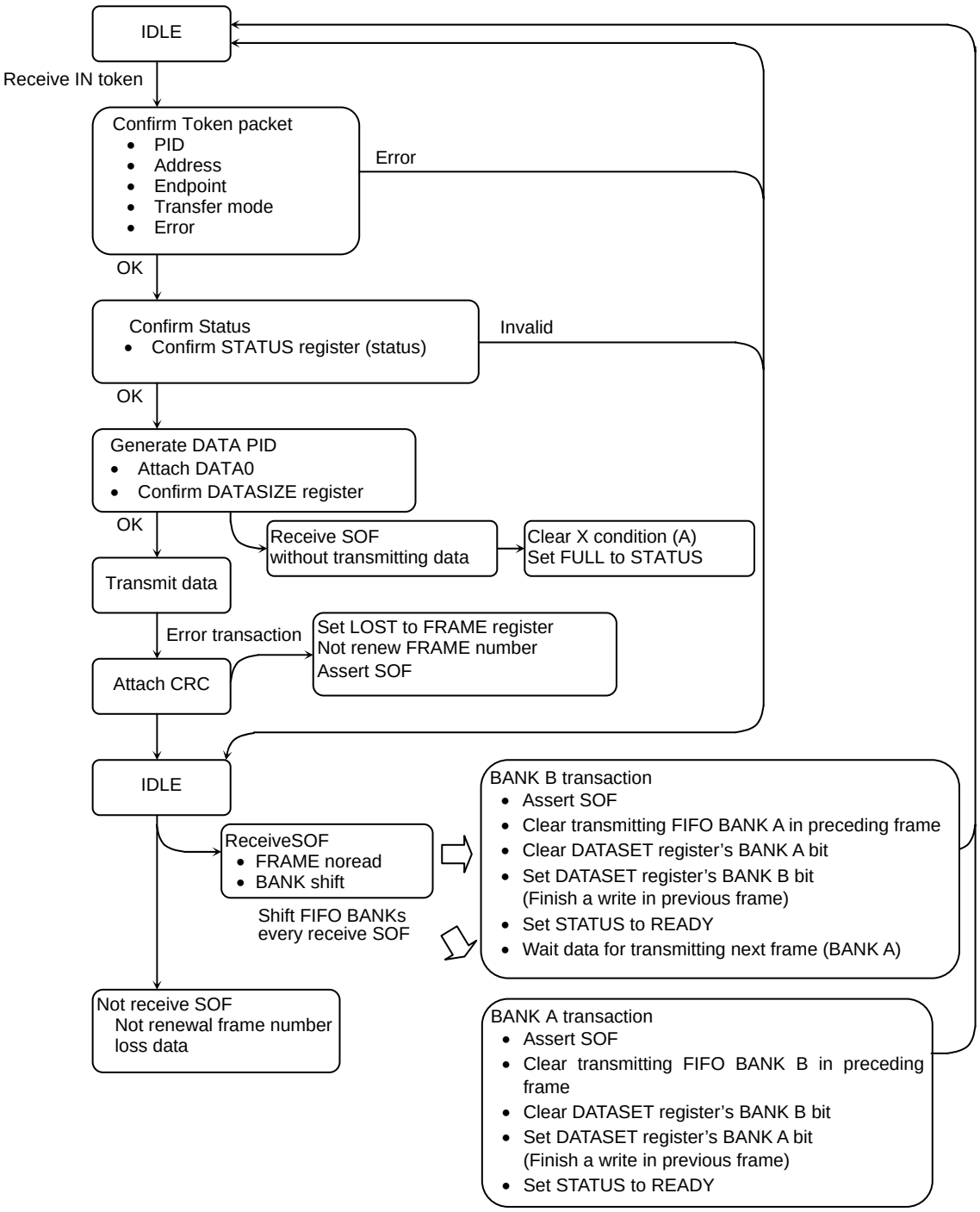


Figure 3.16.13 Control Flow in UDC (Isochronous transfer type (Transmission))

(d-2) Isochronous receiving mode

Isochronous transfer type format in receiving is below transaction format.

- Token: OUT
- Data: DATA0

Control flow

Isochronous transfer type is frame management. And data that is written to FIFO by OUT token is received to CPU in next frame.

Below are two conditions in FIFO of Isochronous receiving mode transferring

X. FIFO for storing data that received from host in present frame (DATASET register bit = 0)

Y. FIFO for storing data for transmitting host in previous frame (DATASET register bit = 1)

FIFO that is divided into two (packet A and packet B) conditions is whether X condition or Y condition. Below flow is explained as X Condition (packet A), Y Condition (packet B) in present frame.

X and Y conditions change one after the other by receiving SOF.

Below is control flow in UDC when receiving OUT token.

All transaction is processed by hardware.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether conform applicable endpoint transfer mode with OUT token. If it doesn't conform, state return to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State return to IDLE.
3. Data packet is received.
   Data is transferred from SIE into the UDC to packet A's FIFO (X Condition).
4. After last data was transferred, and compare counted CRC with transferred CRC. When transfer finish, result is reflected to STATUS. However, data is stored FIFO, data number that packet A is received is set to DATASIZE register of packet A.
5. Below is transaction when SOF token from host is received.
   - Change the packet A's FIFO from X Condition to Y Condition.
   - Change the packet B from Y Condition to X Condition, and clear data. Prepare for next transfer.
   - Set frame number to frame register.
   - Assert SOF and inform that frame is incremented to external.
   - DATASET register set packet A bit and it clear packet B bit arrangement loading in present frame.
   - If CRC comparison result agree it, DATAIN is set to STATUS. If result doesn't agree, RX_ERR is set to STATUS.

UDC finishes normally by above transaction.

CPU takes back packet A's data.

In renewed frame, Packet A's FIFO interchange packet B's FIFO, and transaction is used same flow.

If SOF token is not received by error and so on, this data is lost because of frame is not renewed. Nothing problem in receiving PID and if frame data is received with CRC error, USB sets LOST to STATUS on FRAME register, and frame number is not renewed. However, in this case, SOF is asserted and FIFO condition is renewed. If SOF token is received without transmit and transfer Isochronous in frame, UDC clears FIFO (X Condition) and sets STATUS to FULL.

These are shown in Figure 3.16.14.

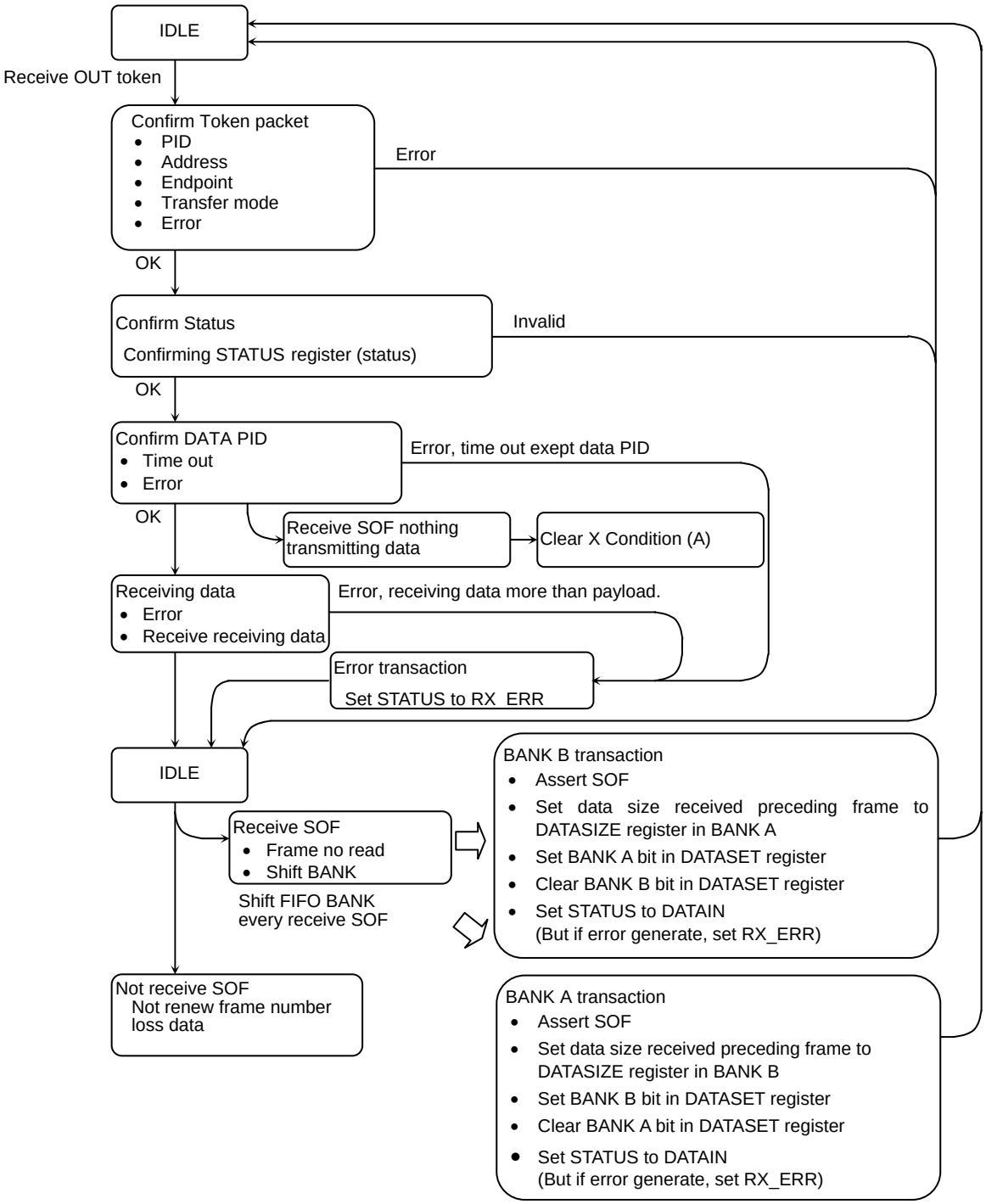


Figure 3.16.14 Control Flow in UDC (Isochronous transfer type (Receiving))

3.16.7 Bus Interface and Access to FIFO

(1) CPU bus interface

UDC prepares two types of FIFO access, single packet and dual packet. In single packet mode, FIFO capacity that is implemented by hardware is used as big FIFO. In dual packet mode, FIFO capacity that is divided into two is used as two FIFOs. And it uses as independent FIFO. Even if UDC is transmitting and receiving to USB host, it can be used bus efficient by to possible load to FIFO.

But control transfer type receives only single packet mode.

Epx_SINGLE signal in dual packet mode must be fixed to "0". If this signal is fixed to "0", FIFO register runs in single mode.

Sample: If you use endpoint 1 to dual packet of payload 64 bytes.

EP1_FIFO size	:	Prepare 128 bytes
EP1_SINGLE signal	:	Hold 0
EP1 Descriptor setting		
Direction	:	Optional
Max payload size	:	64 bytes
Transfer mode	:	Optional

(a) Single packet mode

This is data sequence of single packet mode when CPU bus interface is used. Figure 3.16.15 is receiving sequence. Figure 3.16.16 is transmitting sequence. Main of this chapter is access to FIFO. Data sequence with USB host refer to chapter 5.

Endpoint 0 can't be changed mode for exclusive single packet mode. Single packet and dual packet of endpoint 1 to 3 can change by setting Epx_SINGLE register. When transferring, don't change packet.

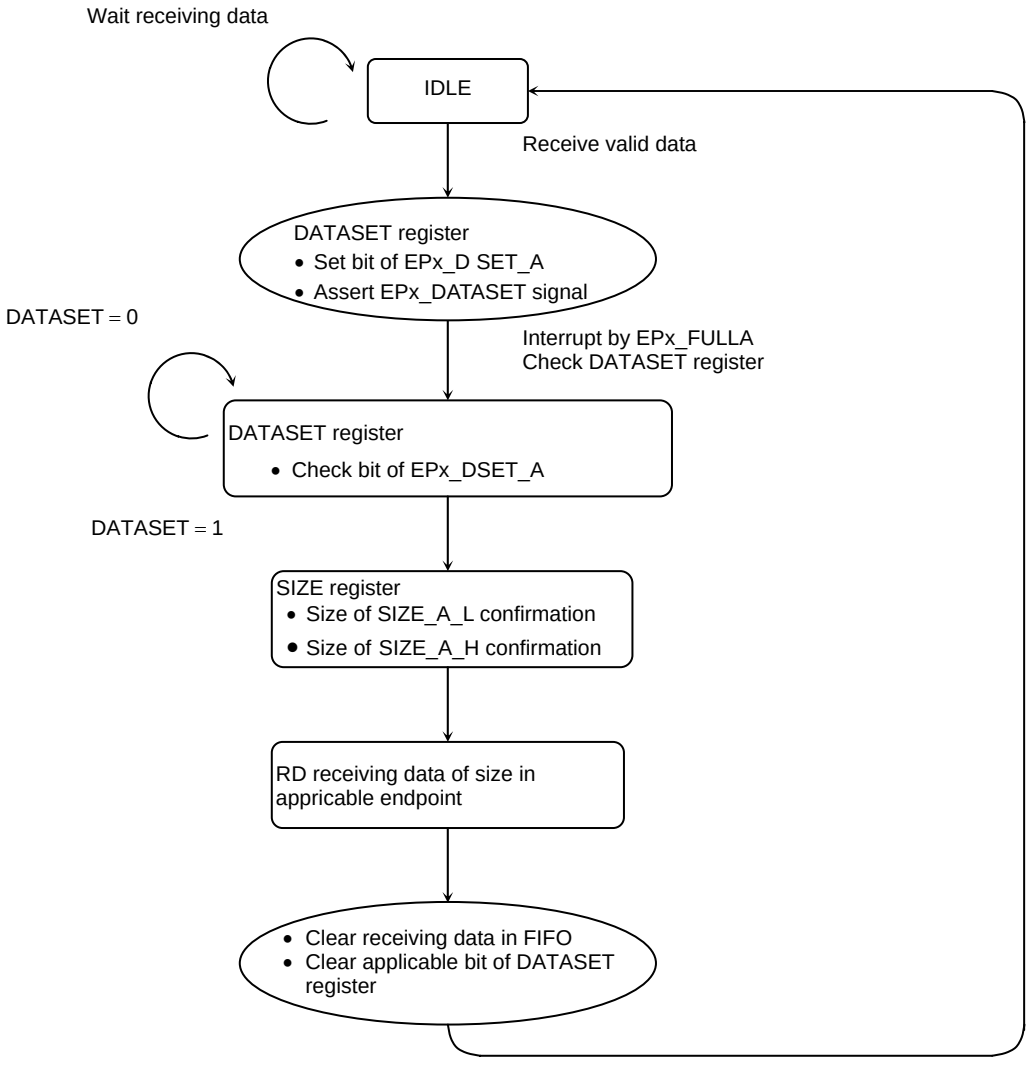


Figure 3.16.15 Receiving Sequence in Single Packet Mode

Below is transmitting sequence in single packet mode.

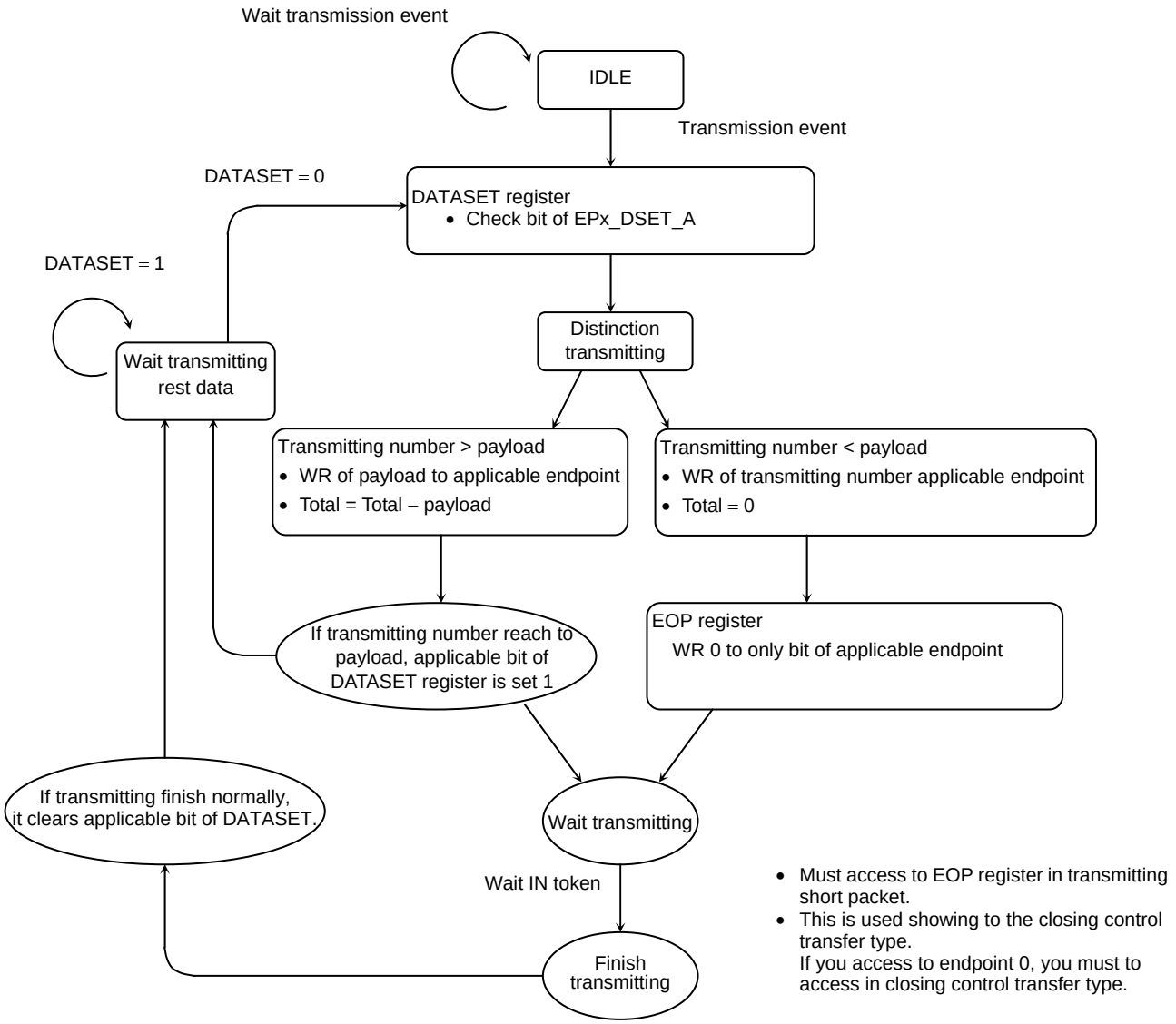


Figure 3.16.16 Transmitting Sequence in Single Packet Mode

(b) Dual packet mode

In dual packet mode, FIFO is divided into A and B packet, it is controlled according to priority in hardware. It can be performed at once, transmitting and receiving data to USB host and exchanges to external of UDC. When it reads out data from FIFO for receiving, confirm condition of two packets, and consider the order of priority. If it has received data to two packets, UDC outputs from first receiving data by FIFO that can be accessed are common in two packets. DATASIZE register is prepared every packet A and packet B. First, CPU must recognize data number of first receiving packet by PACKET_ACTIVE bit. If PACKET_ACTIVE bit was set to 1, that packet is received, first. Packet A and packet B set data turn about always.

Below is this sequence.

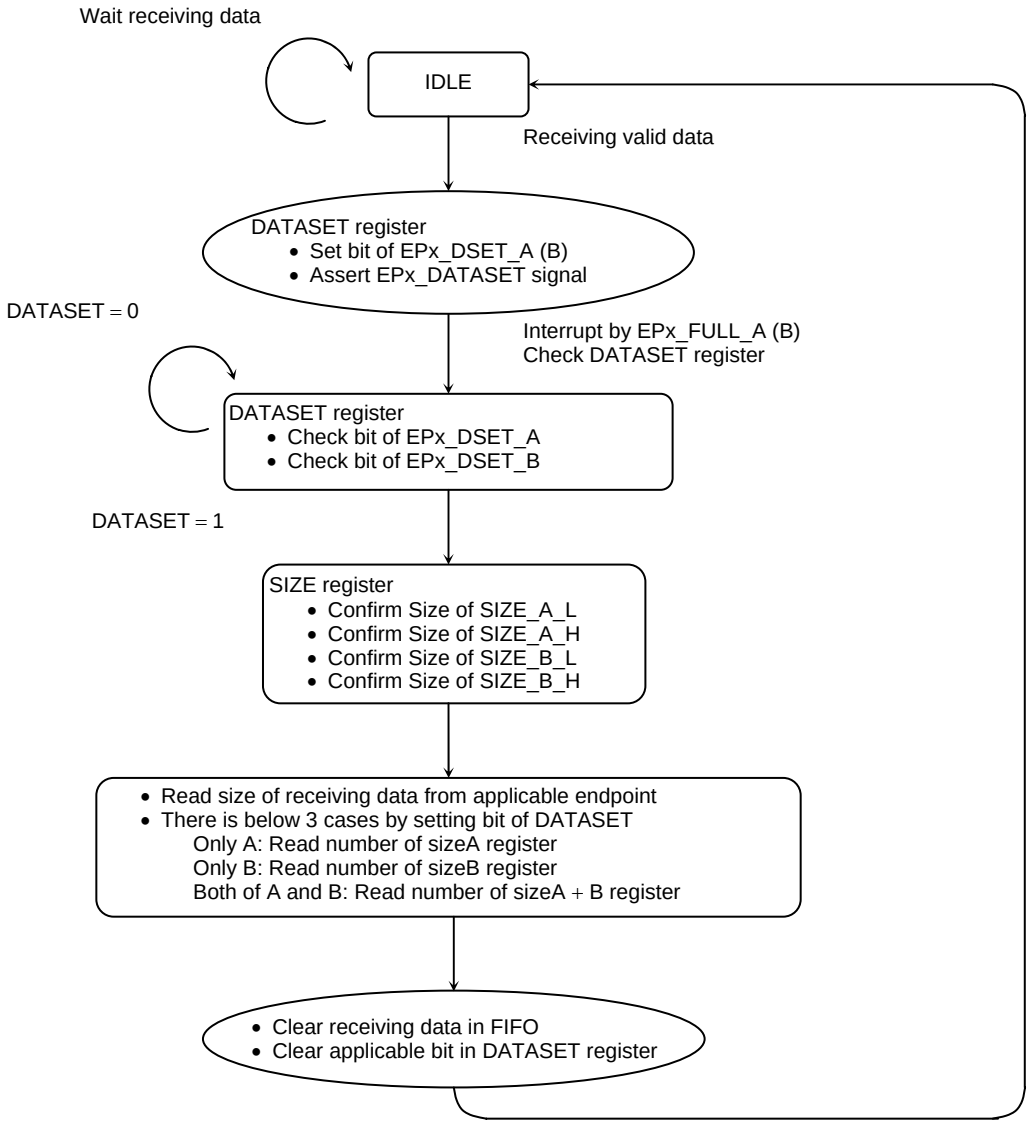


Figure 3.16.17 Receiving Sequence in Dual Packet Mode

When it writes data to FIFO in transmitting, confirm condition of two packets, and consider the order of priority. When transfer data number is set, set to which packet A and packet B, judge by PACKET_ACTIVE bit. Packet that bit is set to 0 is bit that transfer now.

In transmitting and receiving, logic of PACKET_ACTIVE bit is reversed. Therefore, please caution in transmitting.

Below is this sequence.

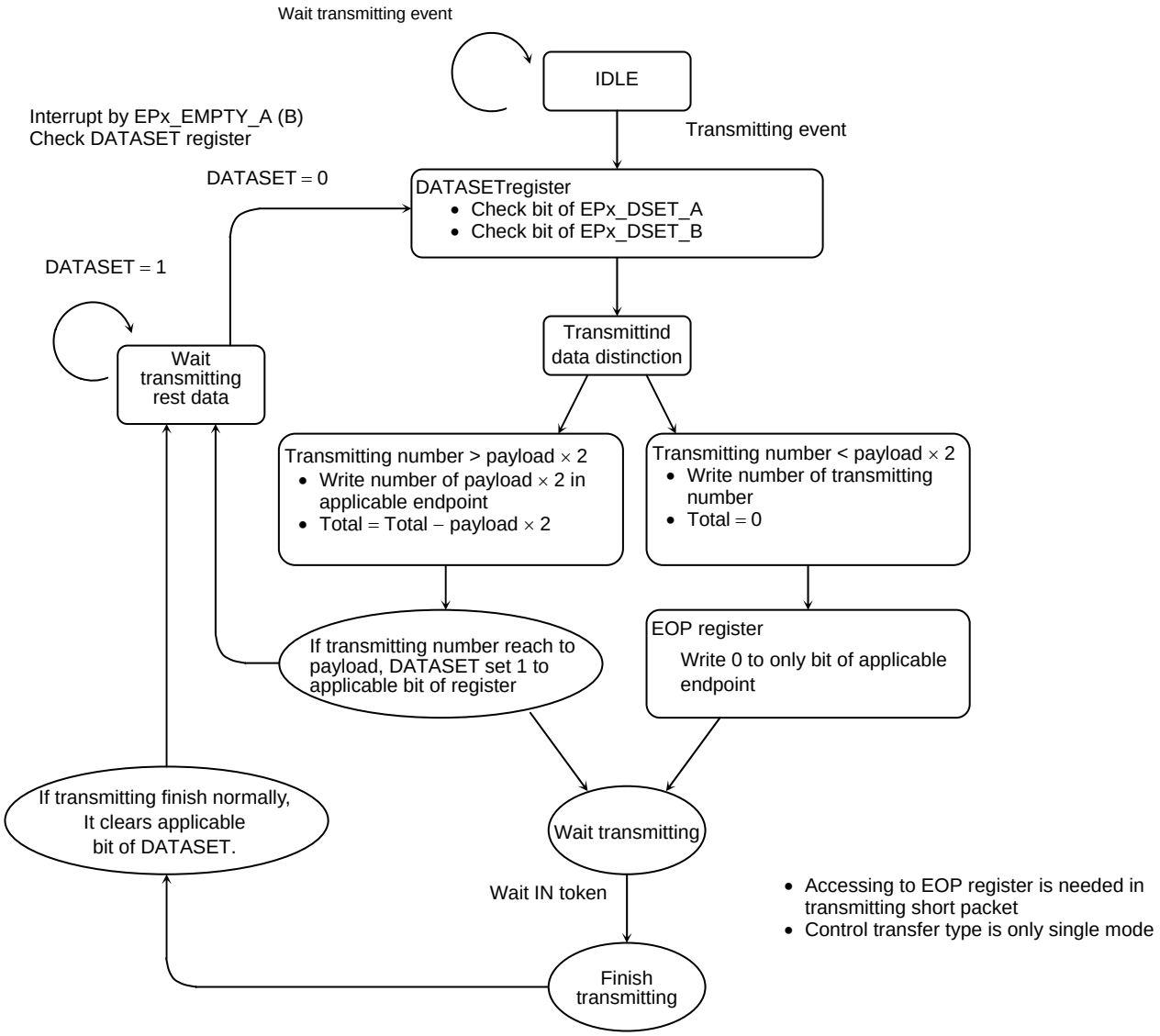


Figure 3.16.18 Transmitting Sequence in Dual Packet Mode

(c) Issuance of NULL packet

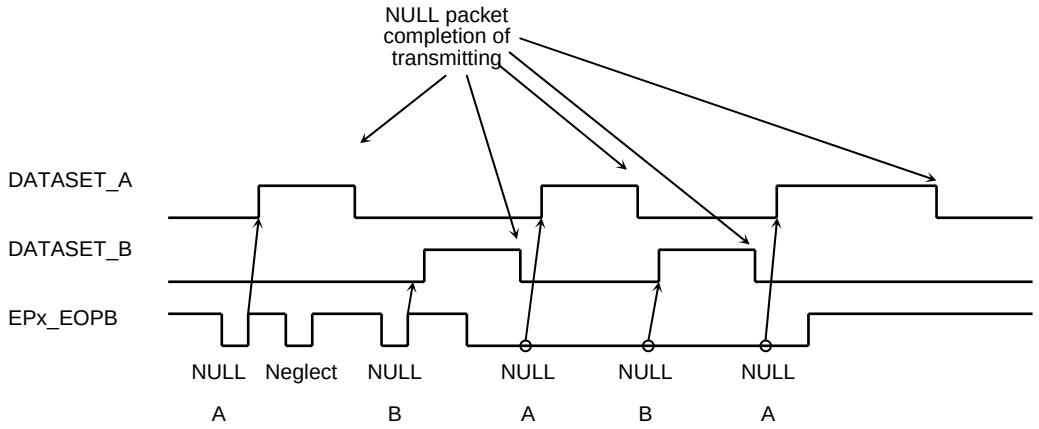
If transmitting NULL packet, by input L pulse from EPx_EOPB signal, data of 0 length is set to FIFO, and it can be transferred NULL packet to IN token.

But if it set NULL data to FIFO, it is valid only case of SET signal is L level condition (case of FIFO is empty). If it answer to receiving IN token by using NULL packet in a certain period, it is answered by keeping EPx_EOPB signal to L level.

However, if mode is dual packet mode, EPx_DATASET signal assert L level for showing space of data. Therefore, data condition (both data have not data) cannot be confirmed from external.

Note: NULL packet can be set also accessing EOP register.

Example:



(2) Interrupt control

Interrupt signal is prepared. This function use adept system.

Detail refers to 3.10.2 900/H1 CPU I/F.

3.16.8 USB Device answer

USB controller (UDC) sets various register and initialization in UDC in detecting of hardware reset, detecting of USB bus reset, and enumeration answer.

Below is explaining about each condition.

(1) Condition in detect in bus reset.

When UDC detects bus reset on USB signal line, it initializes internal register, and it prepares enumeration operation from USB host. After detect in USB reset, UDC sets ENDPOINT0 to control transfer type 8-byte payload and default address for using default pipe. And endpoint except for it is prohibited.

Register name		Initial value
ENDPOINT STATUS	EP0	40H
	Except for EP0	5CH

(2) Detail of STATUS register

Status register that was prepared every endpoint shows condition of every endpoint in UDC.

Each condition affects transfer various USB. Condition changing in each transfer type refers to chapter 5.

EPx_STATUS register value is 0 to 3, and it shows conditions of below. 0 to 4 are result of various transfers. It can be confirmed previous result that is transferred to endpoint by confirming from external of UDC.

0 READY
1 DATAIN
2 FULL
3 TX_ERR
4 RX_ERR

These conditions mean that endpoint operate normally. Meaning that is showed is different every transfer mode. Therefore, please refer to below each transfer mode column.

ISO transfer mode

Below is transfer condition of frame before one. Receiving SOF renews this.

	OUT (RX)	IN (TX)
Initial	READY	READY
Not transfer	READY	FULL
Finish normally	DATAIN	READY
Detect in error	RXERR	TXERR

Transfer mode of except ISO transfer

This is result previous transfer. When transfer finish, this is renewed.

	OUT, SETUP	IN
Initial	READY	READY
Transfer finish normally	DATAIN	READY
Status stage finish	READY	READY
Transfer error	RXERR	TXERR

"Initial" is that renew RESET, USB reset, Current_Config register. In detect error, it doesn't generate EPx_DATASET except toggle transfer mode and Isochronous transfer mode of interrupt.

5 to 7 in showing of status register mean that endpoint is special condition.

5 BUSY — BUSY generate only endpoint of control transfer. If UDC transfer in control writes transfer, when CPU isn't finishing enumeration transaction, and if it receive ID of status stage from USB host, BUZY is set. STATUS is BUZY until CPU finishes enumeration transaction and EP0 bit of EOP register is written 0 in UDC. If CPU enumeration transaction finishes and EP0 bit of EOP register is written 0 and status stage from USB host finish normally, it displays READY.

Please refer to 5.2.3 in chapter 5.

6 STALL — STALL show that endpoint is STALL condition.

This condition generate if it violates protocol or error in bus enumeration. If return endpoint to condition that transfer can normally, device request of USB is needed. This request returns condition normally. But control endpoint returns to condition normally by receiving SETUP token. And it become to SETUP stage.

7 INVALID — This condition shows condition that endpoint can't be used. UDC sets condition that isn't appointed in ENDPOINT to INVALID condition, and it ignores all of token for this endpoint. In initializing, this condition generate always. When UDC detects hardware reset, it sets all endpoint to INVALID condition. Next, if USB reset is received, endpoint 0 only is renewed to READY. Other endpoint that is defined on disruptor, is renewed if SET_CONFIG request finish normally.

3.16.9 Power Management

USB controller (UDC) can be switched from optional resume condition (turn on the power supply condition) to suspend (Suspension) condition, and it can be returned from suspends condition to turn on the power supply condition.

This function can be set to low electricity consumption by operating CLK supplying for UDC.

(1) Switch to suspend condition

USB host can be set USB device to suspend condition by keeping on IDLE state. UDC switches to suspend condition by below process.

- UDC switches to suspend condition if it detect IDLE state of more than 3 ms on USB signal. At this point, set SUSPEND bit of STATUS register to "1".
- After switch to suspend condition, if besides pass away 2 ms, UDC renews USBINTFR1<INT_SUS> from "0" to "1". After USBINTFR1<INT_CLKSTOP> was renewed from "0" to "1", set USBCR1<USBCLKE> to "0", and be stopped supply of CLK (USB_CLK).

  In this condition, all register value into the UDC is kept. However, accessing from external can't be accessed except reading of STATUS register, Current_Config register, and USBINTFR1, USBINTFR2, USBINTMR1, USBINTMR2 and USBCR1

(2) Return from suspend condition by host resume

Way to UDC change from suspend condition to resume condition have two type; resume condition output from USB host and remote wakeup.

When activity of bus on USB signal restore by resume condition output from USB, UDC reset SUSPEND output from "1" to "0", and it resets SUSPEND bit of STATUS register from "0". And it resumed system. Resume condition output from this host keep on no less than during 10 ms. Therefore effective protocol occurring on USB signal line is after pass away this time.

(3) Return from suspend condition by remote wakeup

Remote wakeup is system for prompt resume from suspending USB device to USB host. Remote wakeup isn't supported by condition. And remote wakeup is limited using from USB host by bus enumeration.

Function of remote wakeup in UDC can be used when it is permitted.

Setting remote wakeup by bus can be confirmed bit7 of Current_Config register. When this bit is "1", remote wakeup can be used. Remote wakeup doesn't disable in this bit. Therefore, if this bit show disable, must not set remote wakeup. If it fill the conditions, output resume condition output to USB host by writing USBCR1<WAKEUP> from "1" to "0" of UDC in suspend condition. And it prompts resume from UDC to host. After UDC changes to suspend condition, during 2 ms ignore WAKEUP input. Therefore, remote wakeup become effective by USBINTFR1<INT_SUS> was set to "1".

(4) Low power consumption by control of CLK input signal

When UDC switches to suspend condition, it stops CLK and switches to low power consumption condition. But as system, this function enables besides low power consumption by stopping source of CLK that is supplied from external. CLK that supply to UDC can be controlled clock supply to USB by using USBINTFR1<INT_SUS> and <INT_CLKSTOP>.

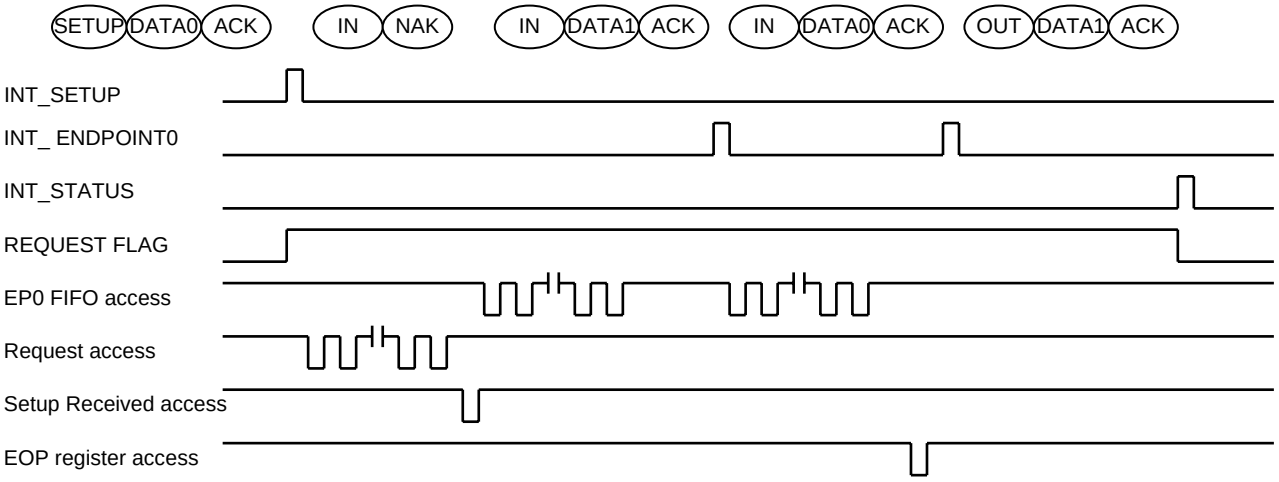
If UDC switches to suspend condition, USBINTFR1<INT_SUS> is set to "1", and <INT_CLKSTOP> is set to "1". After confirmation, stop supply CLK (USBCLK) by setting "0" to USBCR1<USBCLKE>. If SUSPEND signal is set to "0" by resuming from host, supply normal CLK to UDC within 3 ms.

When it uses remote wakeup, supplying stable CLK to UDC before using is needed. When it uses doublers circuit as generation source, above control is needed.

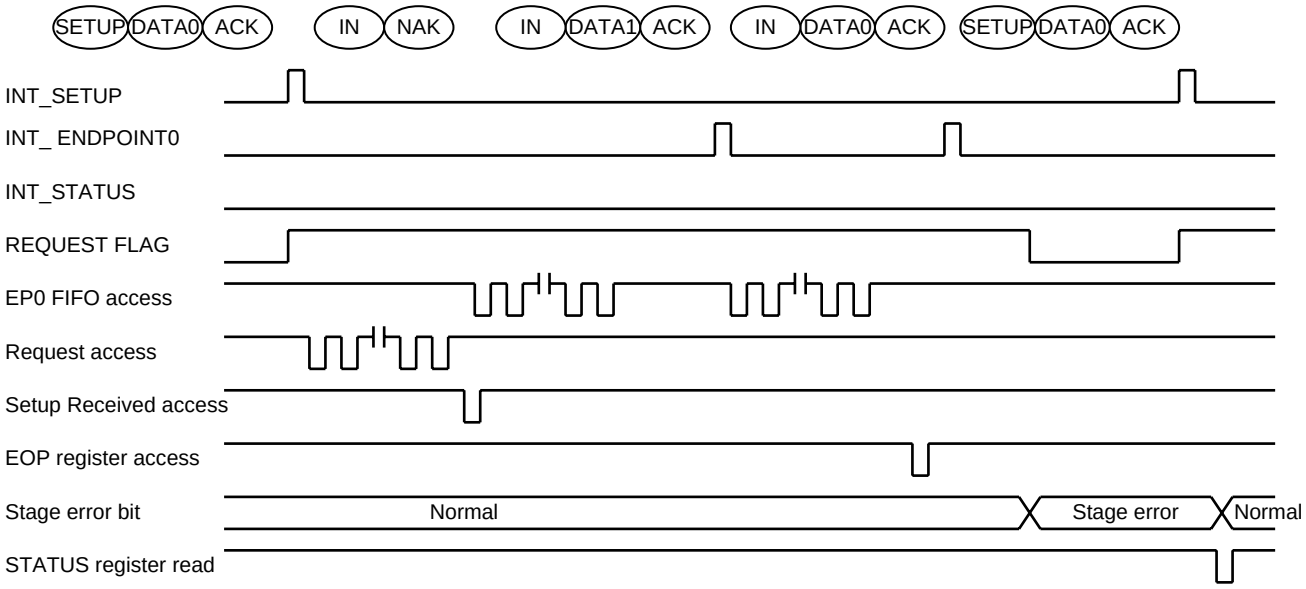
3.16.10 Supplement

(1) External access flow to USB communication

a) Normally movement



b) Stage error



(2) Register beginning value

Register Name	Beginning Value OUTSIDE Reset	Beginning Value USB_RESET
bmRequestType	0x00	0x00
bRequest	0x00	0x00
wValue_L	0x00	0x00
wValue_H	0x00	0x00
wIndex_L	0x00	0x00
wIndex_H	0x00	0x00
wLength_L	0x00	0x00
wLength_H	0x00	0x00
Current_Config	0x00	0x00
Standard request	0x00	0x00
Request	0x00	0x00
DATASET	0x00	0x00
Port Status	0x18	Hold
Standard request mode	0x00	Hold
Request mode	0x00	Hold

Register Name	Beginning Value OUTSIDE Reset	Beginning Value USB_RESET
INT control	0x00	0x00
USBBUFF_TEST	0x00	Hold
USB state	0x01	0x01
EPx_MODE	0x00	0x00
EPx_STATUS	0x1C	0x1C
EPx_SIZE_L_A	0x88	0x88
EPx_SIZE _L_B	0x08	0x08
EPx_SIZE_H_A	0x00	0x00
EPx_SIZE_H_B	0x00	0x00
FRAME_L	0x00	0x00
FRAME_H	0x02	0x02
ADRESS	0x00	0x00
EPx_SINGLE	0x00	Hold
EPx_BCS	0x00	Hold
ID_STATE	0x01	0x00

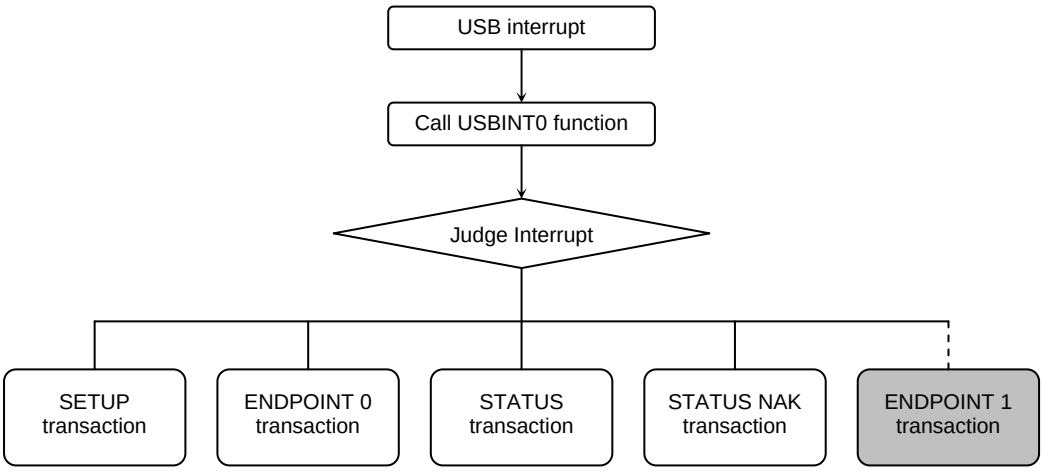
Note 1: Above initial value is value that is initialized by external reset, USB_RESET. This value may differs display value by various condition.
Please refer to register configure of chapter 2.

Note 2: Initial value of EPx_SIZE_L_A, EPx_SIZE_L_B, EPx_SIZE_H_A, EPx_SIZE_H_B registers differ by size of FIFO.
EP0_STATUS register is initialized to 0x00 after received USB_RESET.

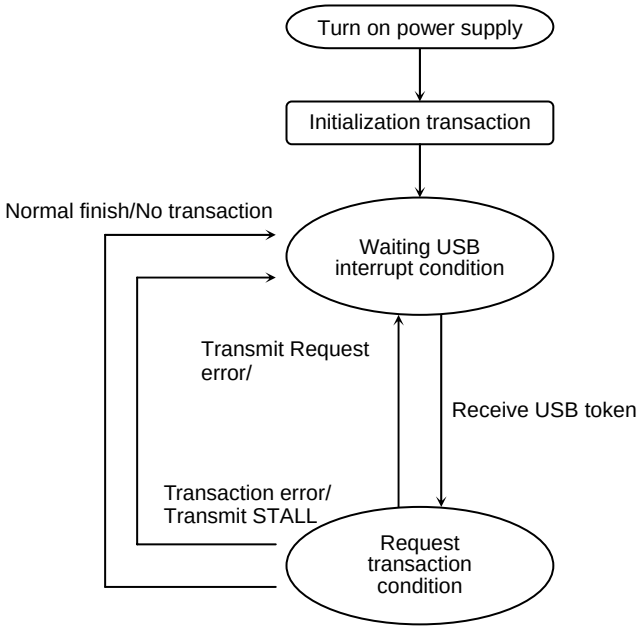
Note 3: Initial value of ID_STATE register is initialized by external reset, BRESET. When USB_RESET signal is received from host, it isinitialized to 0x00.

(3) USB control flow chart

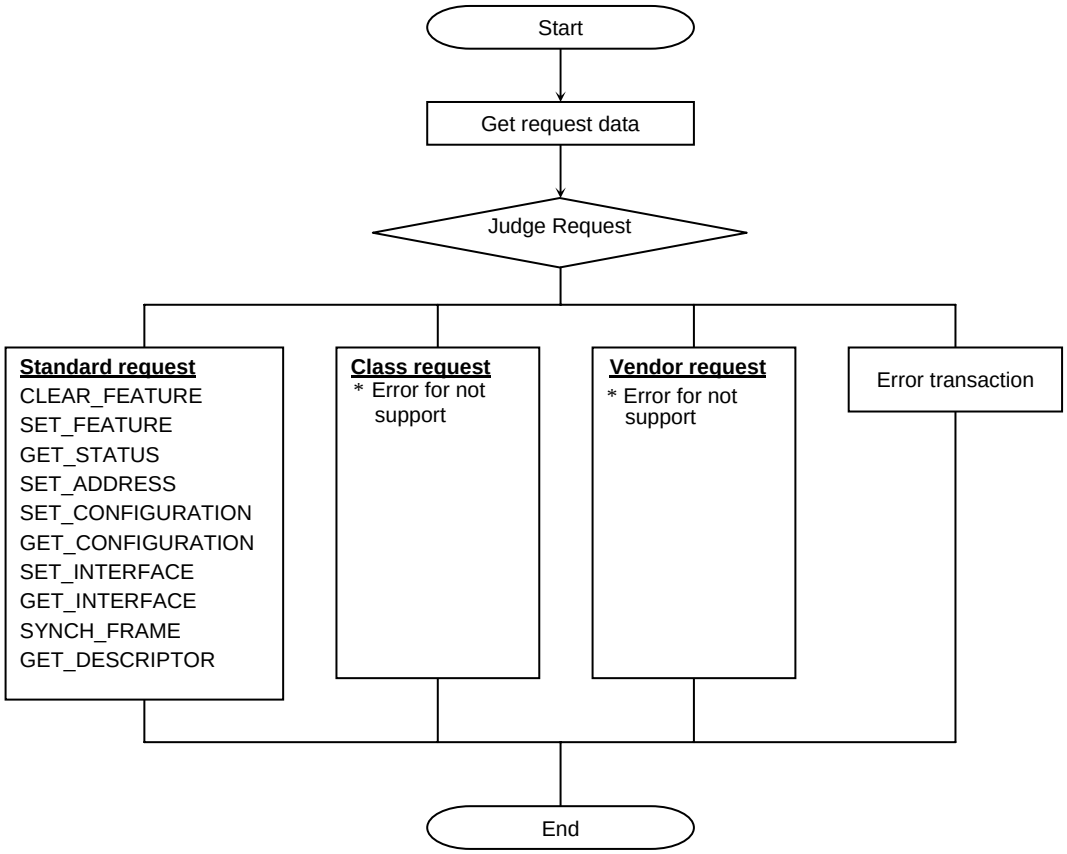
(a) Transaction for standard request (Outline flowchart (Example))



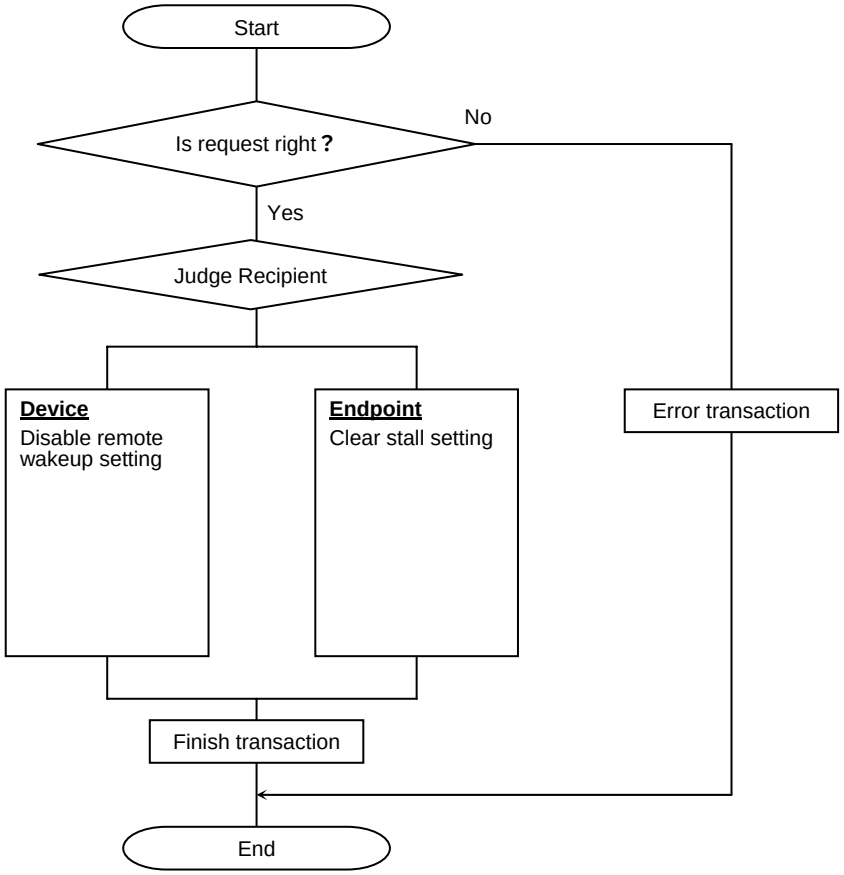
(b) Condition change



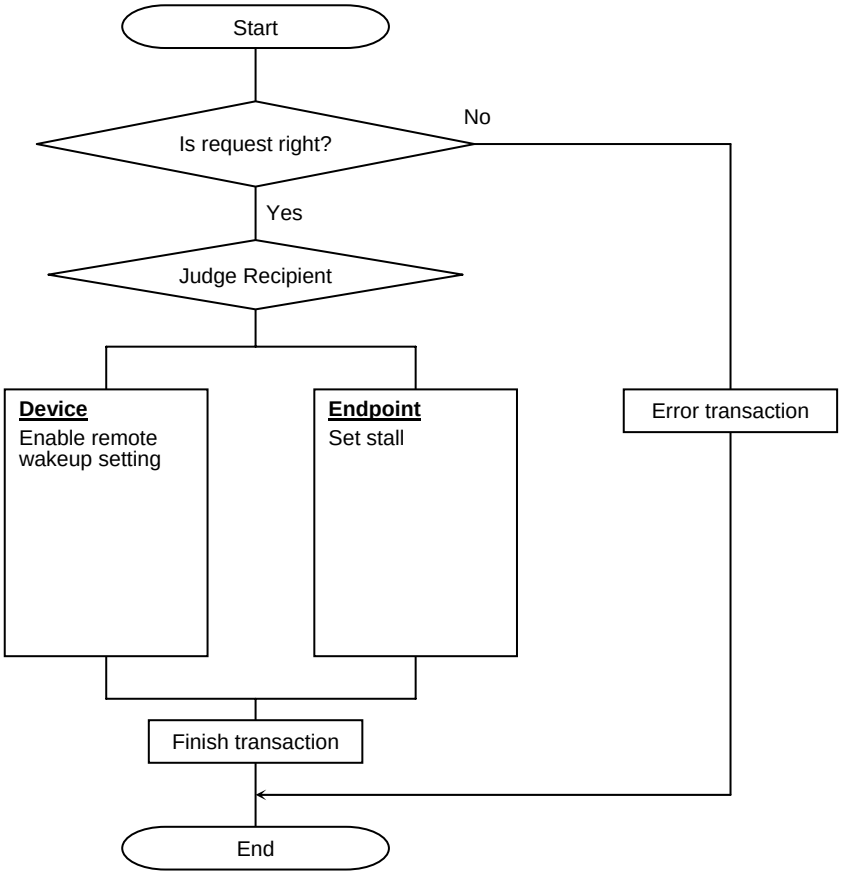
(c) Device request and various request judgment



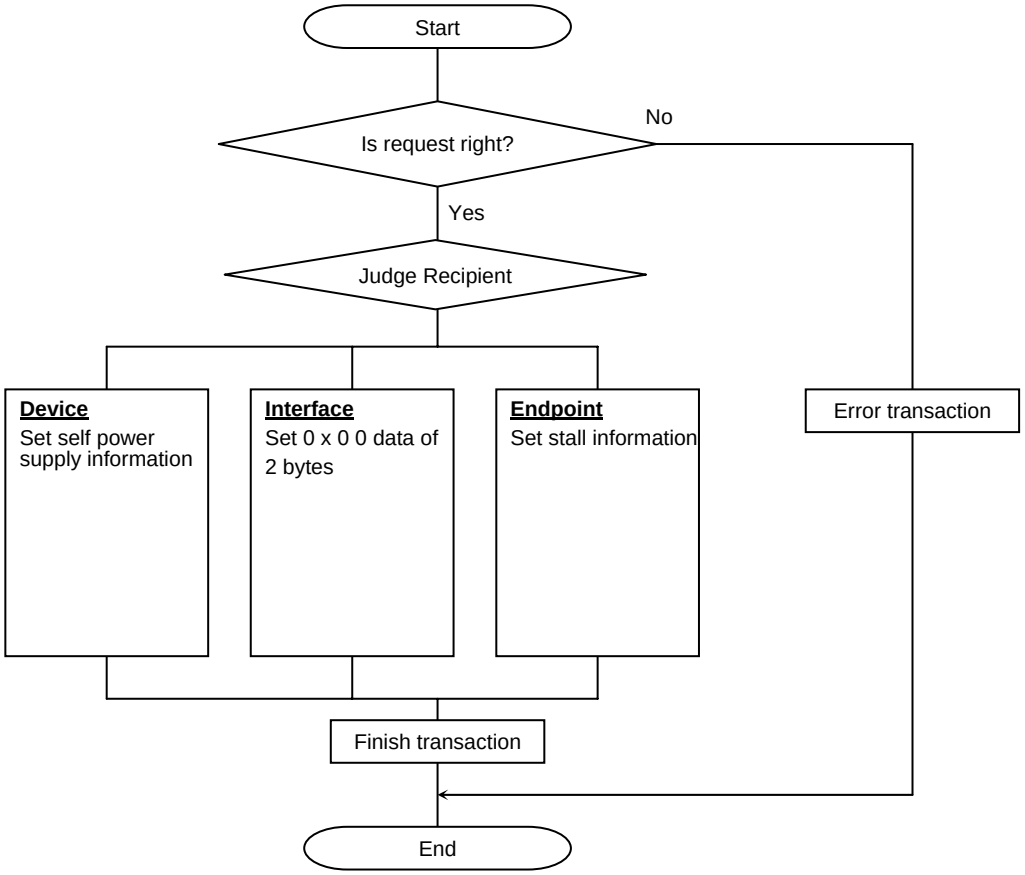
(c-1) CLEAR_FEATURE request transaction



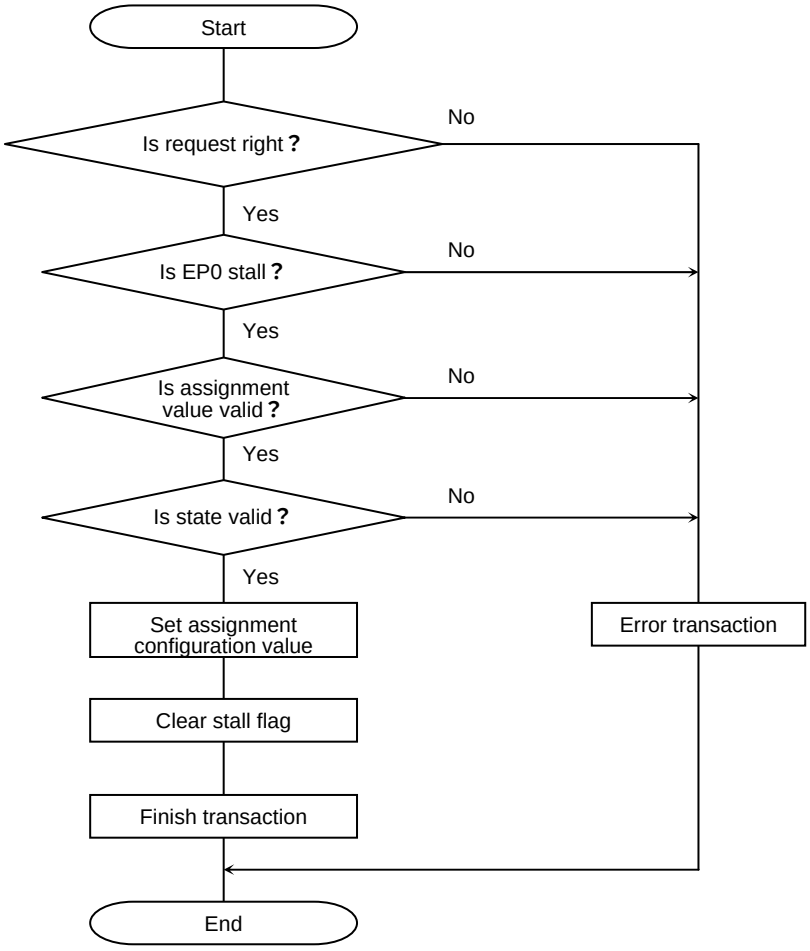
(c-2) SET_FEATURE request transaction



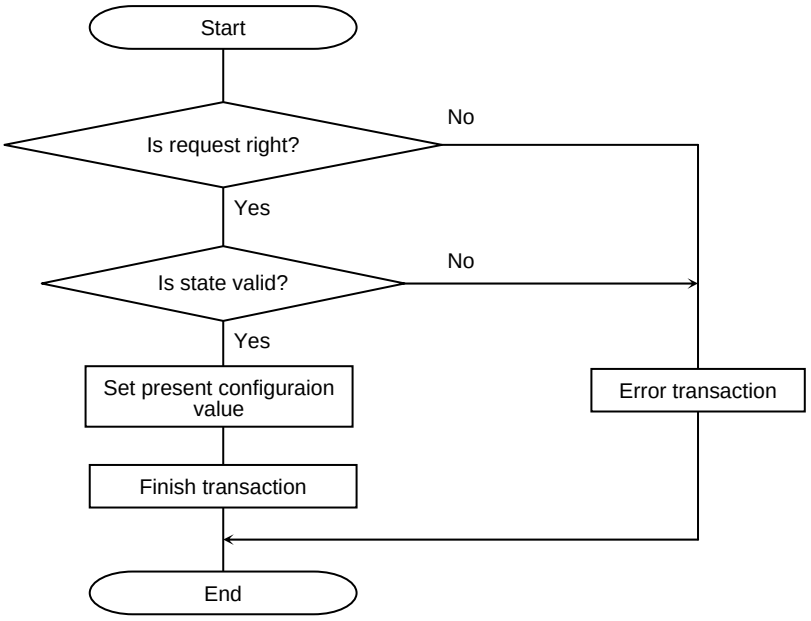
(c-3) GET_STATUS request transaction



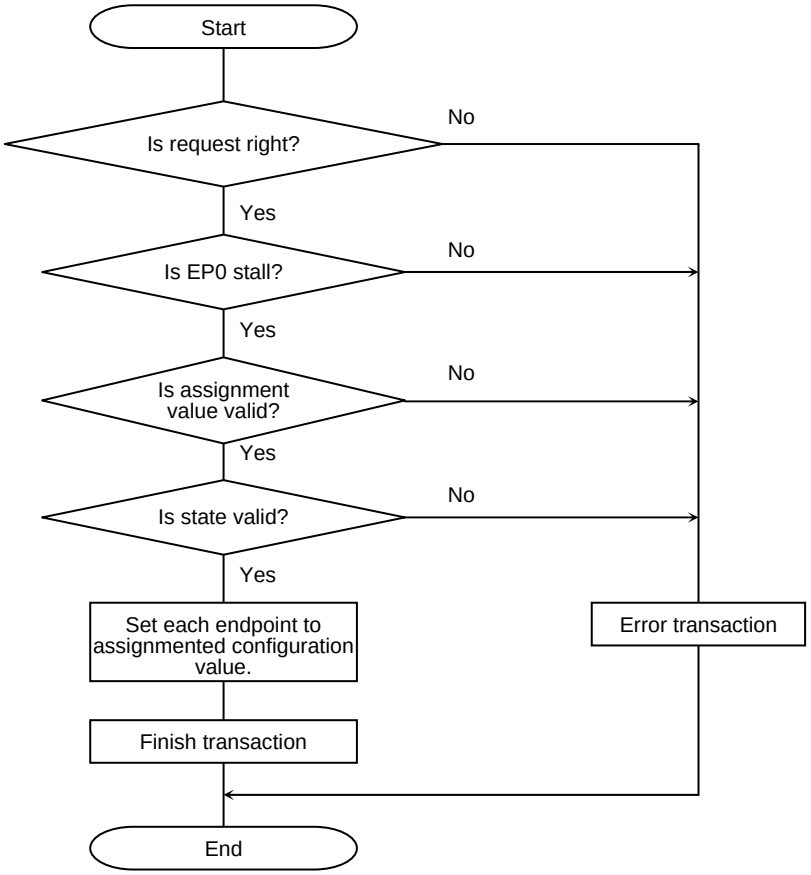
(c-4) SET_CONFIGRATION request transaction



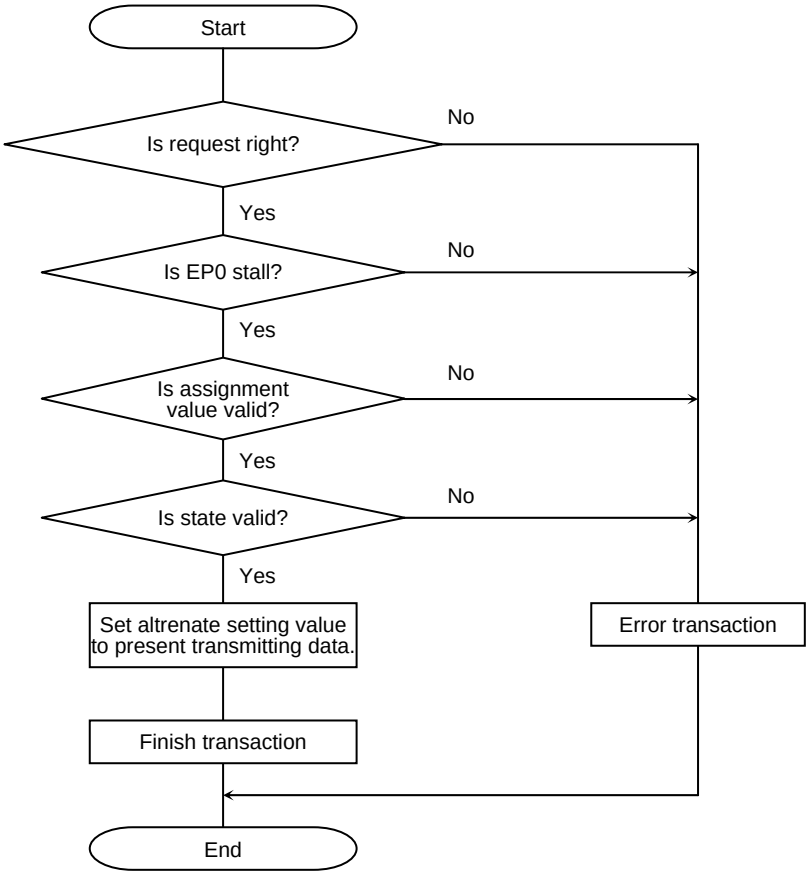
(c-5) GET_CONFIGRATION request transaction



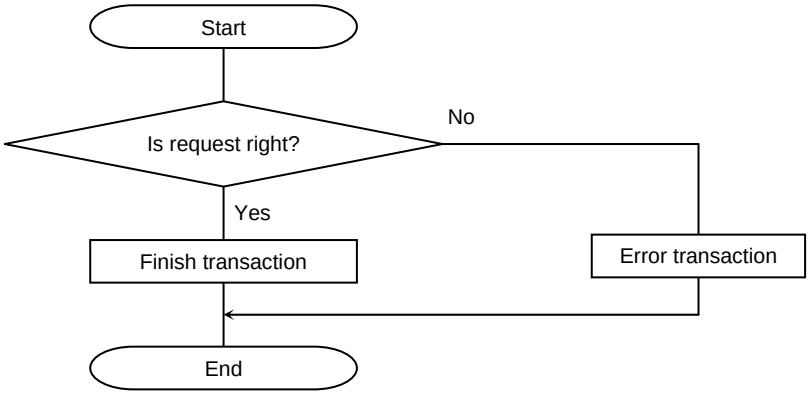
(c-6) SET_INTERFACE request transaction



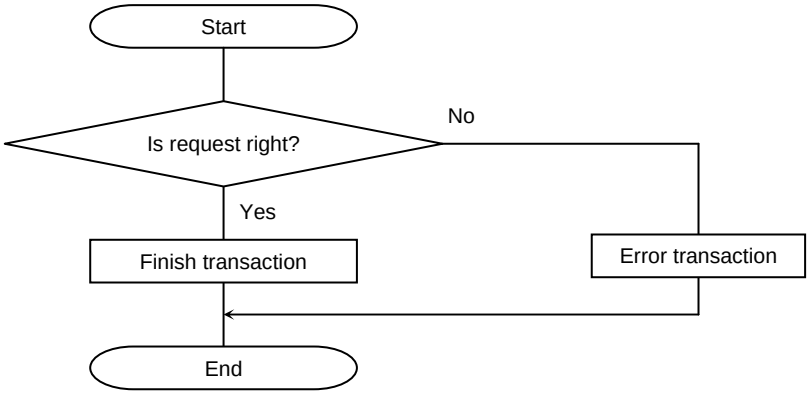
(c-7) SYNCH_FRAME request transaction



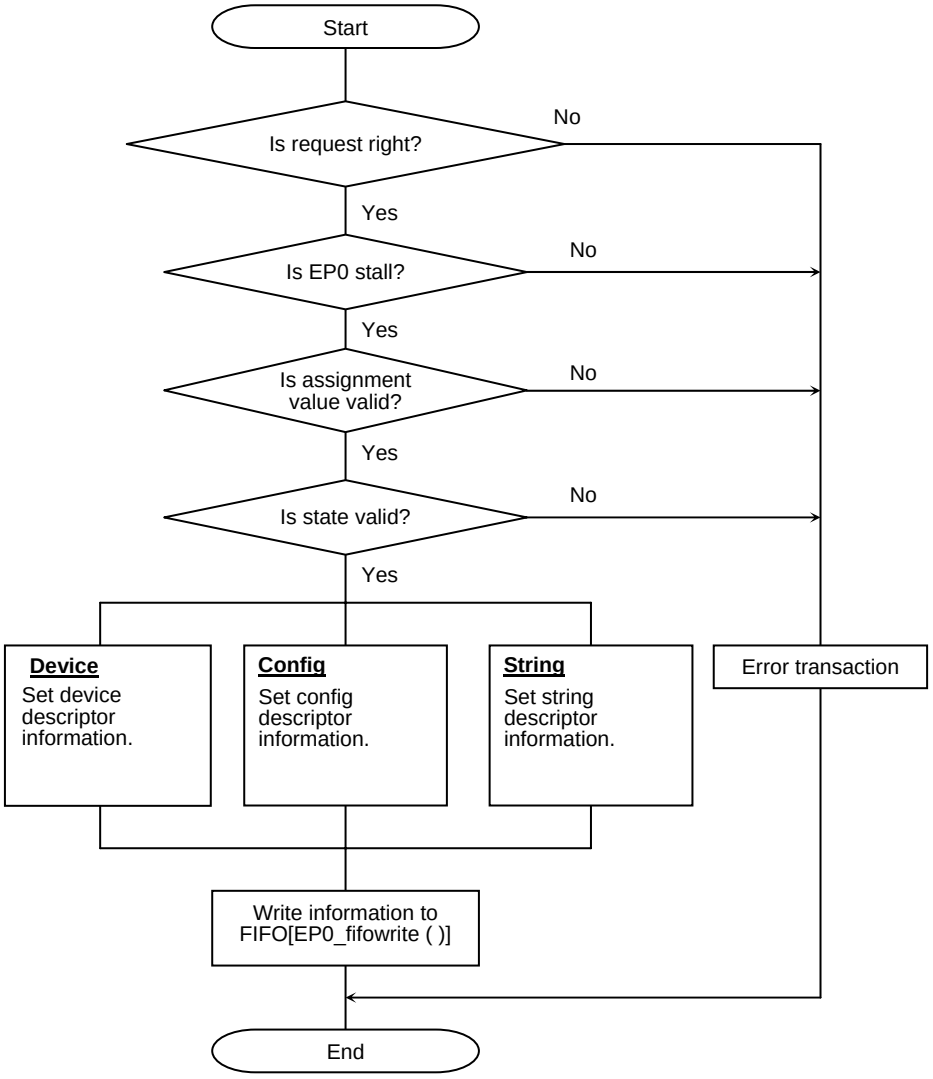
(c-8) SYNCH_FRAME request transaction



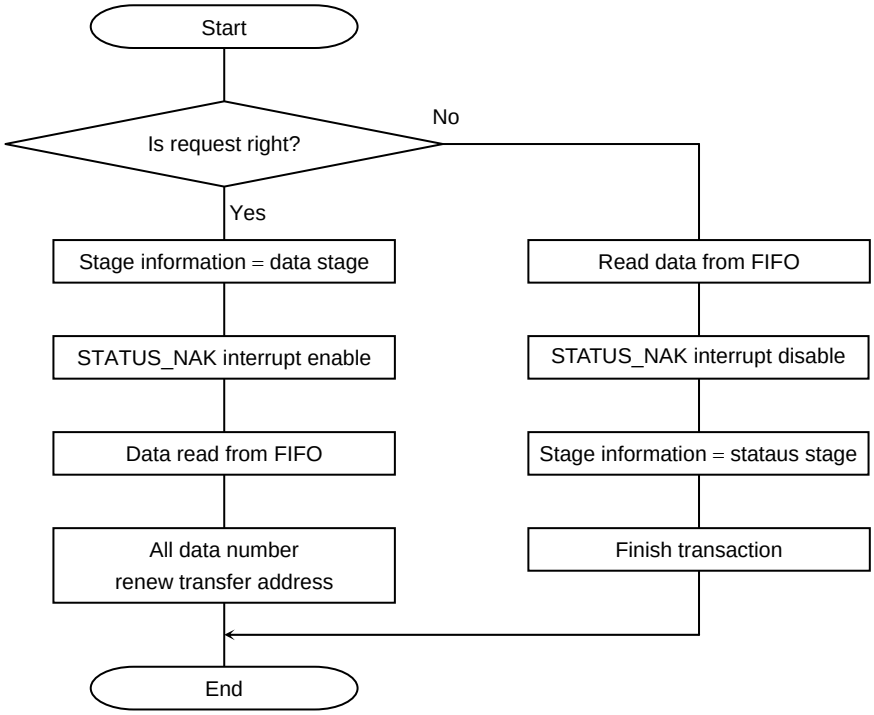
(c-9) SET_DESCRIPTOR request transaction



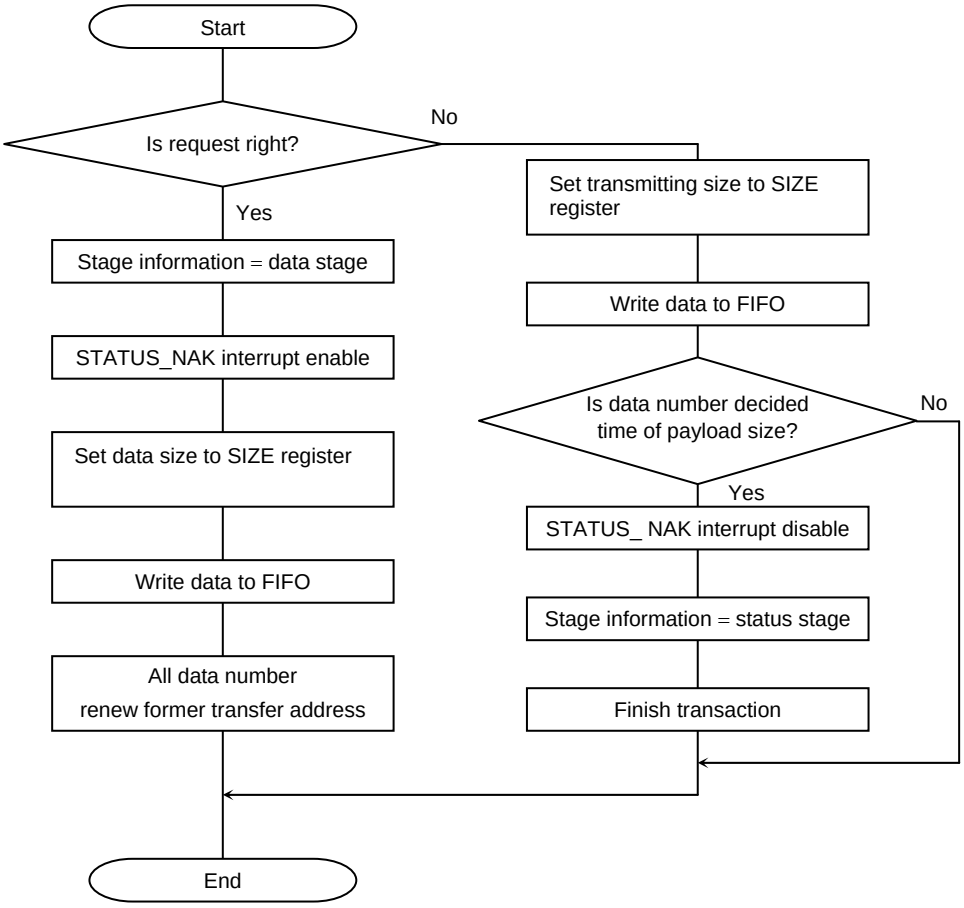
(c-10) GET_DESCRIPTOR request transaction



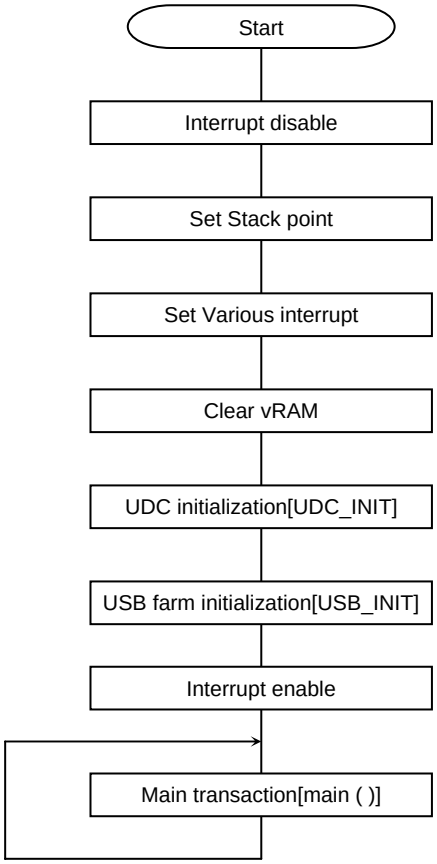
(c-11) Data read transaction to FIFO by EP0



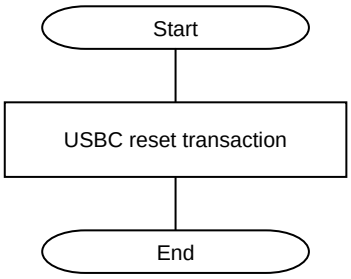
(c-12) Data write transaction to FIFO by EP0



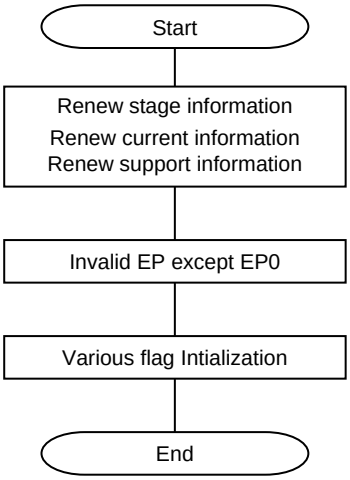
(c-13) Beginning setting transaction of microcontroller



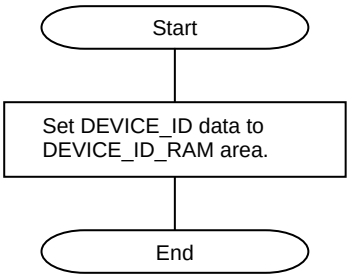
(c-14) Begining setting transaction of UDC



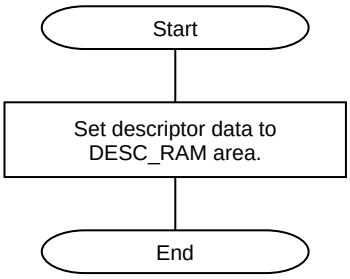
(c-15) Beginning transaction of USB farm changing number



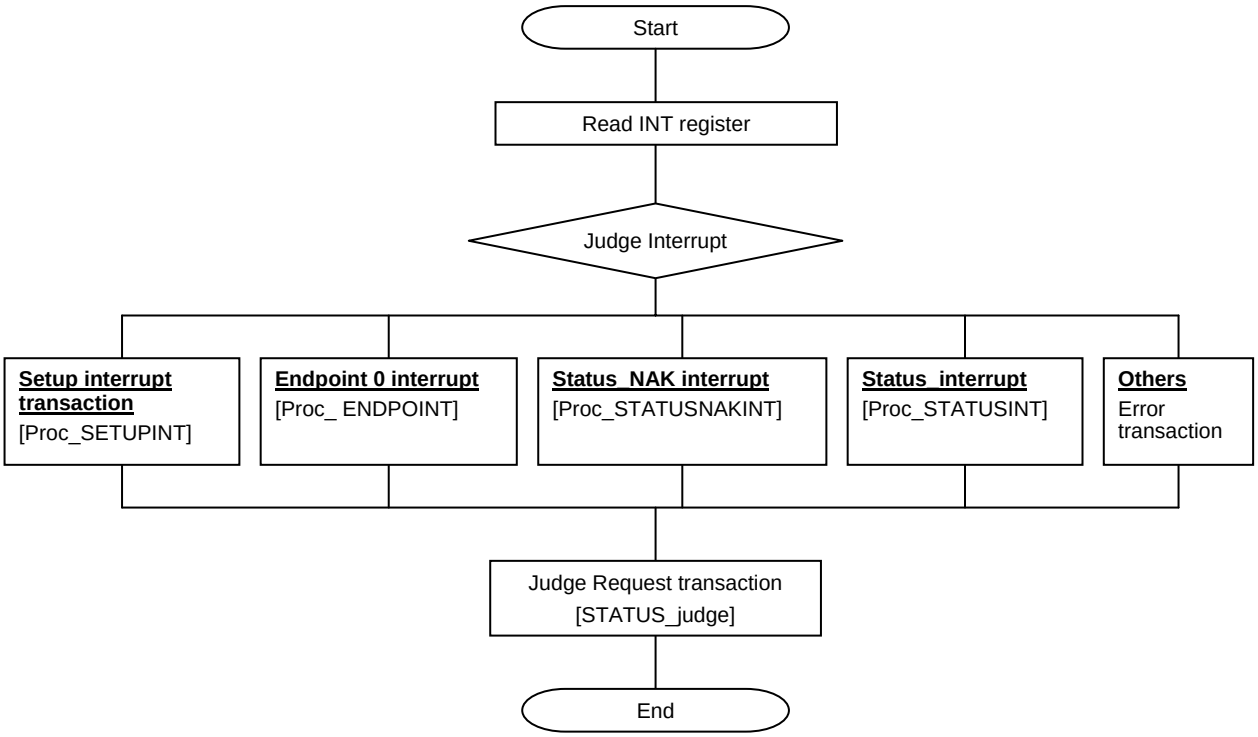
(c-16) Set DEVICE_ID data to DEVICE_ID of UDC



(c-17) Descriptor data set transaction



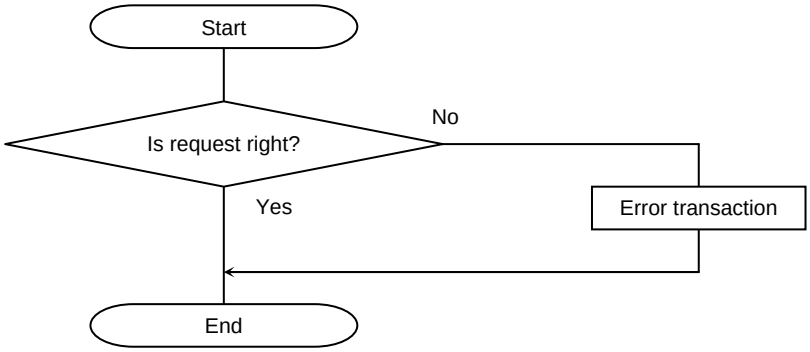
(c-18) USB interrupt transaction



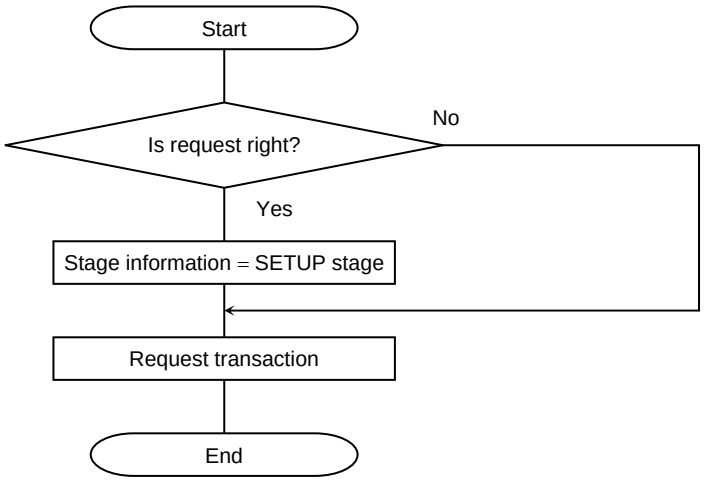
(c-19) Dummy function for not using maskable interrupts.

- Transaction performs nothing, therefore outline flow is skipping.

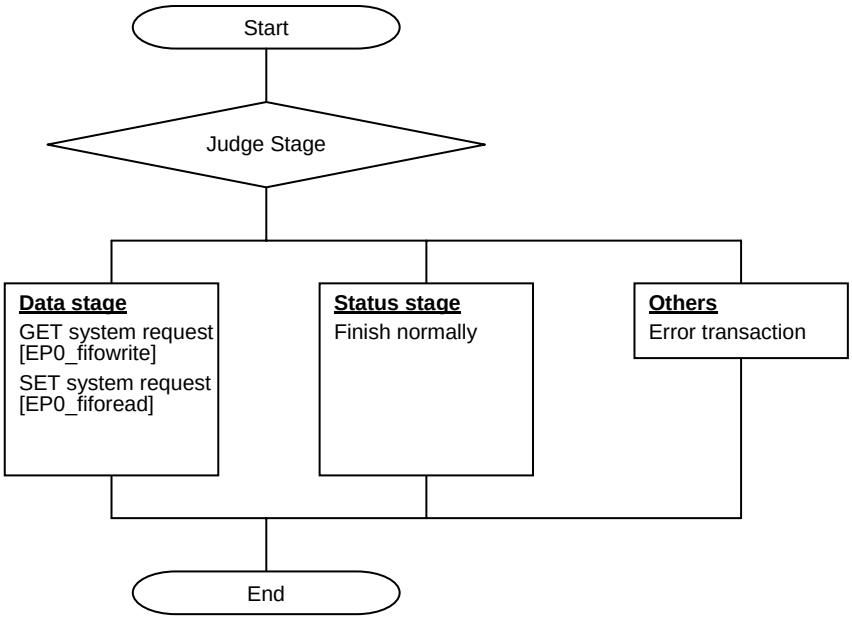
(c-20) Request judgment transaction. If transaction result is error, it puts STALL command.



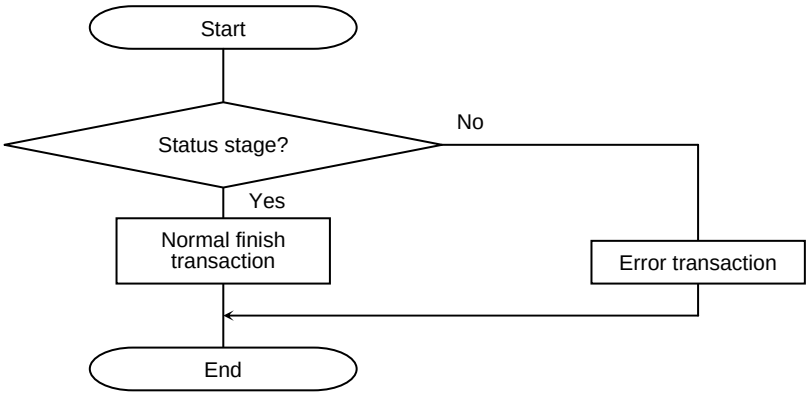
(c-21) SETUP stage transaction



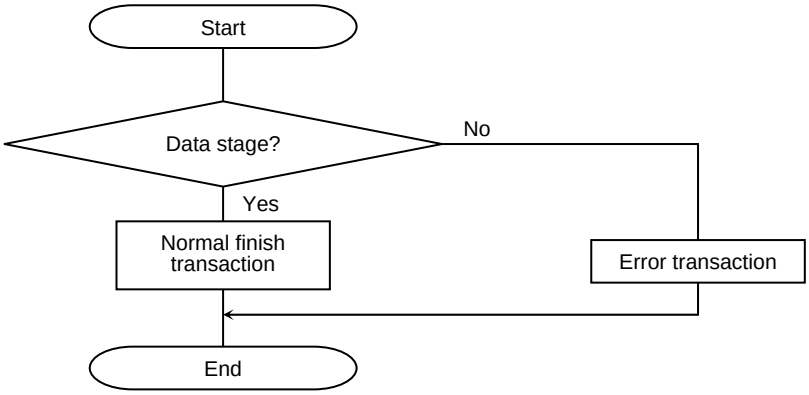
(c-22) Perform endpoint 0 transaction in except for SETUP stage.



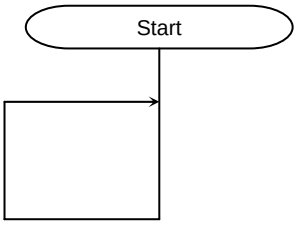
(c-23) Status stage interrupt transaction



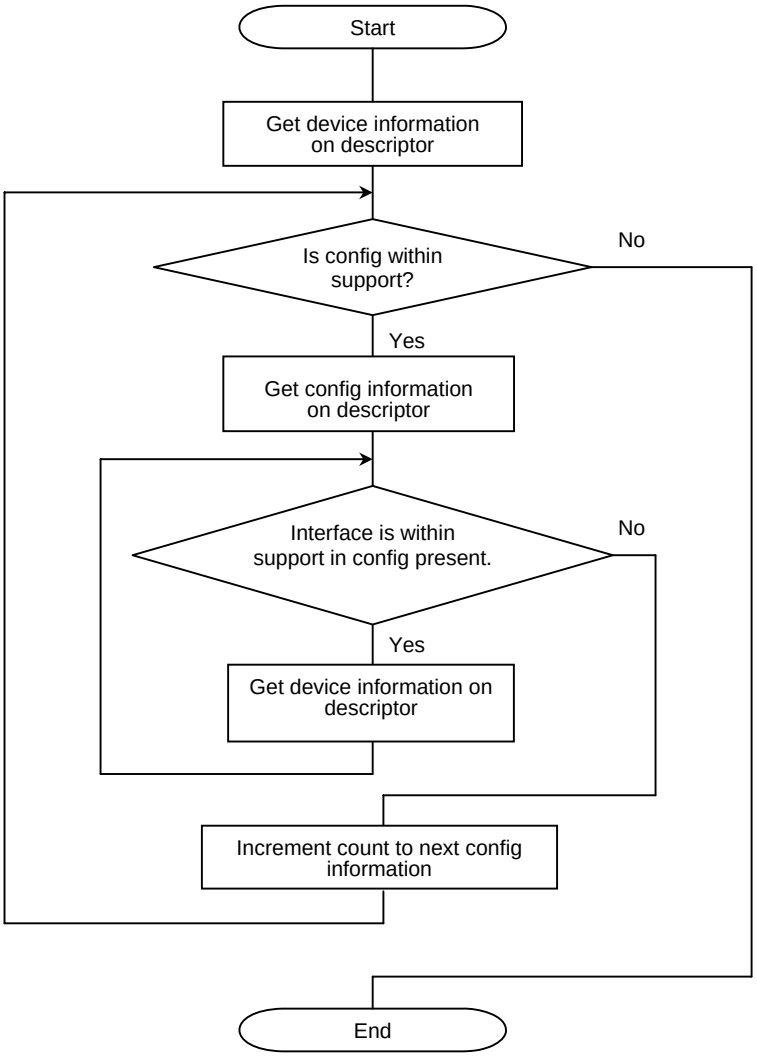
(c-24) STATUS_NAK interrupt transaction



(c-25) This transaction is no transaction by USB transaction perform in interrupts.



(c-26) Getting descriptor information (reration of standard request)



3.16.11 Points to Note and Restrictions

1. Limitation of writing to COMMAND register in special timing

   When “STALL” command is issued, ENDPOINT status might be shift to “INVALID”. To avoid this problem, keep the below routine.

   a. BULK (IN/OUT)

      In case issue STALL command to endpoint in BULK transfer, be sure to issue STALL command after stop RD/WR accessing to endpoint; that is UDC returns NAK in the response of token from host. INT_EPxNAK should be used to detect NAK transmit.

   b. CONTROL OUT with data stage (software response)

      If STALL needs to be set for endpoint 0 judging from request after receiving INT_SETUP interrupt, access to SetupReceived register. After that, issue STALL command after detecting INT_ENDPOINT0 interrupt.

   c. CONTROL OUT without data stage (software response)

      If STALL needs to be set for endpoint 0 judging from request after receiving INT_SETUP interrupt, issue STALL command before access to eop register.

   d. CONTROL IN(software response)

      If STALL needs to be set for endpoint 0 judging from request after receiving INT_SETUP interrupt, issue STALL command before set the first transmit data to host.

2. Limitation of EPx_STATUS<STATUS2:0> when execute USB_RESET command

   EPx_STATUS<STATUS2:0> may indicates different condition, if execute USB_RESET command to the endpoint in the process of token. To avoid this phenomenon, do not RESET the endpoint in transferring. (It is available in the process of request that needs USB_RESET to that endpoint.)

3. When generating toggle error of device controller

   a. UDC operation

      If USB host fail to receive ACK transmitted from UDC in OUT transfer, USB host transmits the same data to UDC again. When the FIFO is available to receive, UDC detects toggle error because of detecting the same data(having the same toggle as the data which is received just before) and returns ACK. UDC rejects it because the data have already received normally. While, if FIFO is not available, UDC returns NAK and informs USB host that is unable to receive.

4. If using USB device controller in TMP92CZ26A, the crystal oscillator (USB standard ≤ 10 MHz±2500ppm) is recommended. And in this case, the stage of external hub can be used until max 3 stages by the precision of this USB device controller and the internal clock. If USB compliance (USB logo) is needed, the 5 stages connection is needed for external hub. And it is needed that input 48MHz clock from X1USB pin (USB standard ≤ ±2500ppm.)

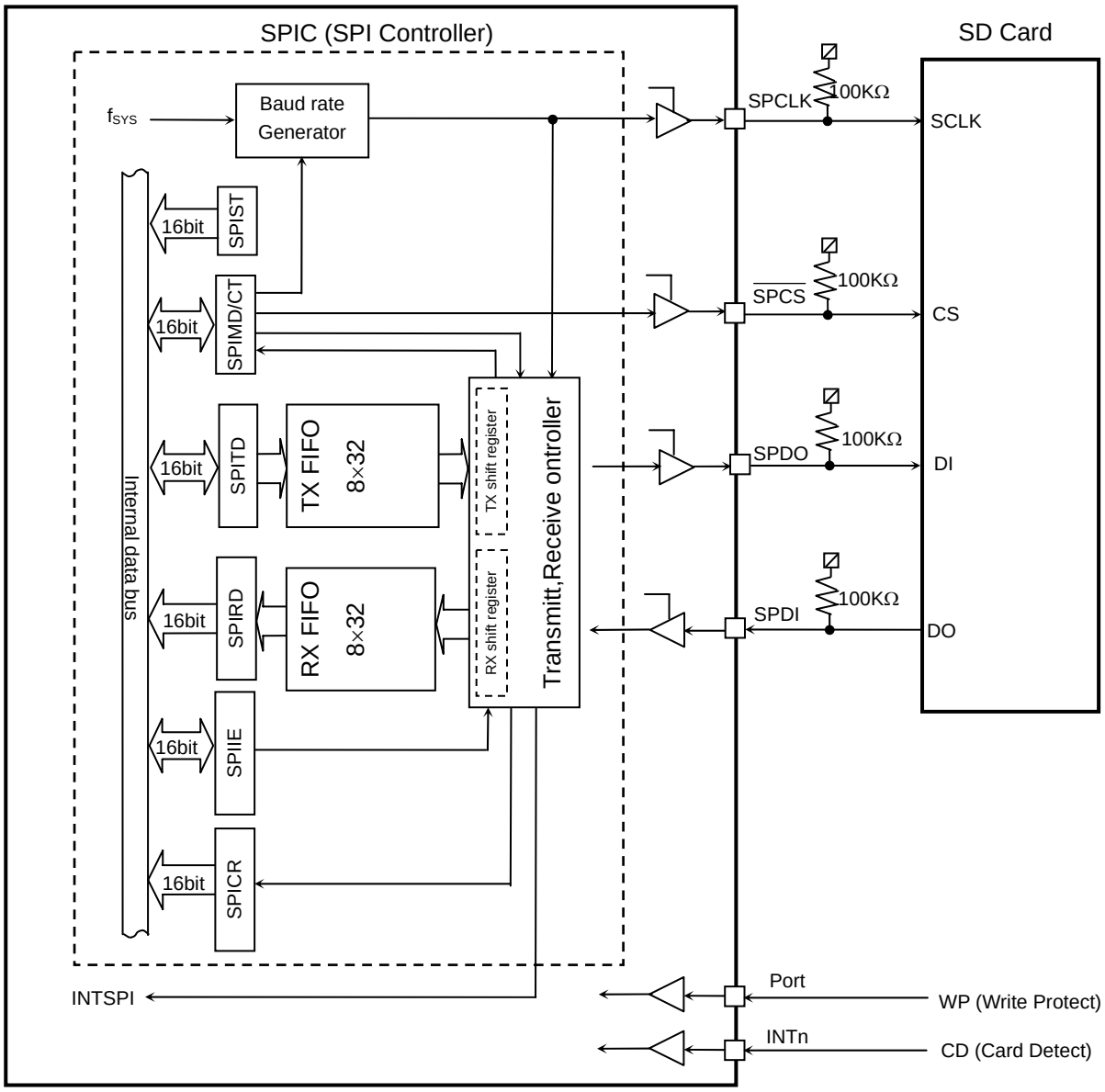
3.17 SPIC (SPI Controller)

SPIC is a Serial Peripheral Interface Controller that supports only master mode.
It can be connected to SD card, MMC (Multi Media Card) etc. in SPI mode.
The features are as follows;

1) 32 byte –FIFO (Transmit / Receive)
2) Generate CRC7 and CRC16 (Transmit / Receive data)
3) Baud Rate: 20Mbps max
4) Connect several SD cards and MMC. (Use other output port for /SPCS pin as /CS)
5) Use as general clock synchronous SIO.
   MSB/LSB-first, 8/16bit data length, rising/falling edge
6) 2 Interrupts: INTSPITX (Trans interruption), INTSPIRX (receive interruption)
   Select Read/Mask for interrupts: RFUL, TEMP, REND and TEND

3.17.1 Block diagram

It shows block diagram and connection to SD card in Figure 3.17.1.



Note1: SPCLK, $\overline{\text{SPCS}}$, SPDO and SPDI pins are set to input port (Port PR3, PR2, PR1, PR0) by reset.

These signals are needed pull-up resister to fix voltage level, could you adjust resistance value for your final set.

Note2: Please use general input port or interrupt signal for WP (Write Protect) and CD (Card Detect).

Figure 3.17.1 Block diagram and Connection example

3.17.2 SFR

SFR of SPIC are as follows.These area connected to CPU with 16 bit data bus.

(1) SPIMD(SPI Mode setting register)

SPIMD register is for operation mode or clock etc.

SPIMD Register

		7	6	5	4	3	2	1	0
SPIMD (820H)	bit Symbol	SWRST	XEN				CLKSEL2	CLKSEL1	CLKSEL0
	Read/Write	W	R/W				R/W		
	After Reset	0	0				1	0	0
Prohibit to Read Modify Write	Function	Software Reset 0: don't care 1: Reset	SYSCK 0: disable 1: enable				Select Baud Rate(Note1) 000:Reserved 001: f_{SYS}/2 010: f_{SYS}/3 011: f_{SYS}/4	100:f_{SYS}/8 101: f_{SYS}/16 110: f_{SYS}/64 111: f_{SYS}/256	
		15	14	13	12	11	10	9	8
(821H)	bit Symbol	LOOPBACK	MSB1ST	DOSTAT		TCPOL	RCPOL	TDINV	RDINV
	Read/Write	R/W				R/W			
	After Reset	0	1	1		0	0	0	0
	Function	LOOPBACK Test mode 0:disbale 1:enable	Start bit for Transmit / Receive 0:LSB 1:MSB	SPDO pin state (no transmit) 0:fixed to "0" 1:fixed to "1"		Synchronous clock edge during transmitting 0: fall 1: rise	Synchronous clock edge during receiving 0: fall 1: rise	Invert data During transmitting 0: disable 1: enable	Invert data During receiving 0: disable 1: enable

Note: Maximum speed of this SD card is 20Mbps in SD card SPI mode.
When setting the baud rates, select less than 20Mbps according to the operation speed of CPU (f_{SYS}).

Figure 3.17.2 SPIMD register

(a) <LOOPBACK>

Because Internal SPDO can be input to internal SPDI, it can be used as test.

Set <XEN>=1 and <LOOPBACK>=1, outputs clock from SPCLK pin regardless of operation of transmit/receive.

Please change the setting when transmitting/receiving is not in operation.

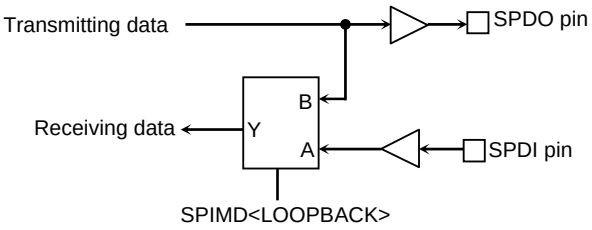


Figure 3.17.3 <LOOPBACK> Function

(b) <MSB1ST>

Select the start bit of transmit/receive data

Please don't change the setting of this register when transmitting/receiving is in operation.

(c) <DOSTAT>

Set the status of SPDO pin when data communication is not operating (after transmitting or during receiving).

Please don't change the setting of this register when transmitting/receiving is in operation.

(d) <TCPOL>

Select the edge of synchronous clock.

Please change the setting when <XEN>bit is “0”. And set the same value as <RCPOL>.

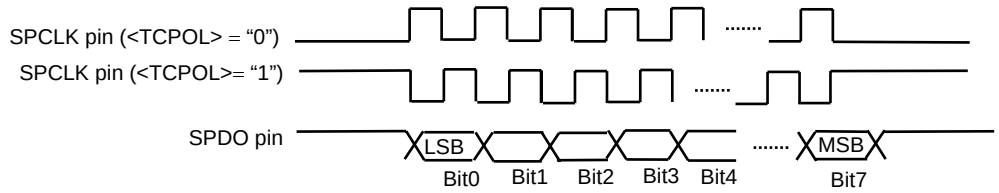


Figure 3.17.4 <TCPOL> Register Function

(e) <RCPOL>

Select the edge of synchronous clock during receiving.

Please change the setting during SPIMD<XEN>= “0”. And set the same value as <TCPOL>.

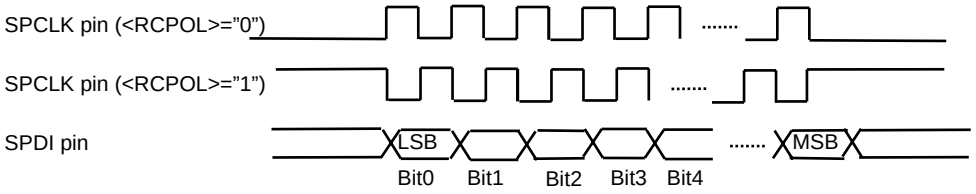


Figure 3.17.5 <TCPOL>Register Function

(f) <TDINV>

Select logical invert/no invert when outputs transmitted data from SPDO pin.

Please don't change the setting of this register when transmitting/receiving is in operation.

(g) <RDINV>

Select logical invert/no invert for received data from SPDI pin.

Please don't change the setting of this register when transmitting/receiving is in operation.

(h) <SWRST>

This bit is for Software reset of transmit/receive FIFO pointer. Write SPICT<TXE> to "0" at <XEN>="1", and stop transmitting. After that, by writing <SWRST> to "1", the read/write pointer of transmit/receive FIFO are initialized.

When writing SPICT<TXE> to "0", stops transmission after the UNIT data in transmitting is transmitted. Write <SWRST> to"1", the data in the transmit FIFO becomes to invalid.

The data in the transmit shift register is cleared simultaneously. Therefore, the data is not output if transmit is restarted after executed software reset.

Please do not write <SWRST> to "1" during transmission. In case of receiving, the received data in the receive FIFO buffer becomes invalid. However, the UNIT data in receiving is loaded to receive FIFO as valid data.

In case of sequential receive, receiving operates sequentially even if the data of receive buffer becomes invalid. Therefore, stops receive operation by writing SPICT<RXE>="0"after finishing to receive all the data in receiving. And all the receive operation is stopped by writing <SWRST>="1"after checking no UNIT data in receiving (namely after REND interrupt or the time to receive 1UNIT).

During receiving, do not write <SWRST>="1".

Software reset can be executed by 1 shot operation; writing <SWRST>="1" (needless to write <SWRST>="0"). Writing <XEN>="1"and <SWRST>="1"simultaneously is permitted.

(i) <XEN>

Enable/disable control of root clock this SPI controller.

(j) <CLKSEL2:0>

Select baud rate. Baud rate is created from f_{SYS} and settings are in under table.
Please change the setting when transmitting/receiving are not in operation.

Note: When setting the baud rates, select less than 20Mbps according to the operation speed of CPU (f_{SYS}).

Table 3.17.1 Example of Baud Rate

<CLKSEL2:0>	Baud Rate [Mbps]	
	f_{SYS} =60MHz	f_{SYS} =80MHz
f_{SYS}/2	–	–
f_{SYS} /3	20	–
f_{SYS} /4	15	20
f_{SYS} /8	7.5	10
f_{SYS} /16	3.75	5
f_{SYS} /64	0.9375	1.25
f_{SYS} /256	0.234375	0.3125

(2) SPICT(SPI Control Register)

SPICT register is for data length or CRC etc.

SPICT Register

		7	6	5	4	3	2	1	0
SPICT (822H)	bit Symbol	CEN	SPCS_B	UNIT16	TXMOD	TXE	FDPXE	RXMOD	RXE
	Read/Write	R/W			R/W	R/W	R/W	R/W	R/W
	After Reset	0	1	0	0	0	0	0	0
	Function	communication control 0: disable 1: enable	/SPCS pin 0: output "0" 1: output "1"	Data length 0: 8bit 1: 16bit	Transmit mode 0: UNIT 1:Sequential	Transmit control 0: disable 1: enable	Alignment in Full duplex 0: disable 1: enable	Receive Mode 0: UNIT 1:Sequential	Receive control 0: disable 1: enable
		15	14	13	12	11	10	9	8
(823H)	bit Symbol	CRC16_7_B	CRCRX_TX_B	CRCRESET_B					
	Read/Write	R/W							
	After Reset	0	0	0					
	Function	CRC select 0: CRC7 1: CRC16	CRC data 0: Transmit 1: receive	CRC calculate register 0:Reset 1:Release Reset					

Figure 3.17.6 SPICT Register

(a) <CRC16_7_B>

Select CRC7 or CRC16 to calculate.

(b) <CRCRX_TX_B>

Select input data to CRC calculation circuit.

(c) <CRCRESET_B>

Initialize CRC calculate register.

The process that calculating CRC16 of transmits data and sending CRC next to transmit data is explained as follows.

(1) Set SPICT <CRC16_7_B> to select CRC7 or CRC16 and <CRCRX_TX_B> to select calculating data.
(2) To reset SPICR register, write “1” after write<CRCRESET_B> to "0".
(3) Write transmit data to SPITD register, and wait for finish transmission all data.
(4) Read SPICR register, and obtain the result of CRC calculation.
(5) Transmit CRC which is obtained in (4) by the same way as (3).

CRC calculation of receive data is the same process.

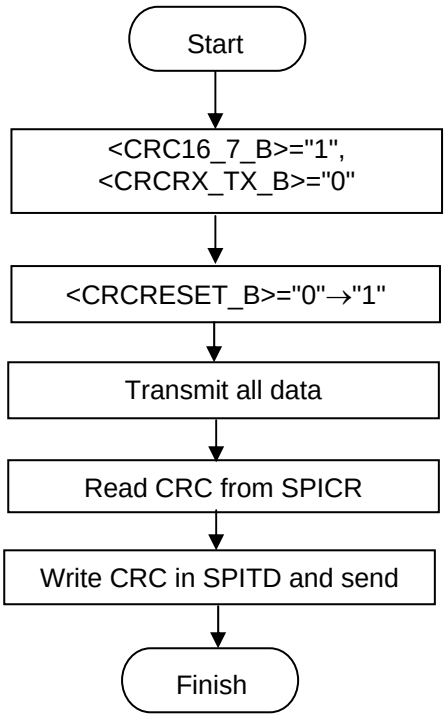


Figure 3.17.7 Flow chart of CRC calculation process

(d) <CEN>

Select enable/disable of the pin for SD card or MMC.

When the card isn't inserted or no-power supply to DVcc, penetrated current is flowed because SPDI pin becomes floating. In addition, current is flowed to the card because $\overline{\text{SPCS}}$, SPCLK and SPDO pin output "1". This register can avoid these matters.

If write <CEN> to "0" with PRCR and PRFC selecting $\overline{\text{SPCS}}$, SPCLK, SPDO and SPDI signal, SPDI pin is prohibited to input (avoiding penetrated current) and $\overline{\text{SPCS}}$, SPCLK, SPDO pin become high impedance.

Please write <CEN>="1" after card is inserted, supply power to Vcc of card and supply clock to this circuit (SPIMD<XEN>="1").

(e) <SPCS_B>

Set the value that outputs to $\overline{\text{SPCS}}$ pin.

(f) <UNIT16>

Select the length of transmit/receive data. Data length is described as UNIT downward. Please don't change the setting of this register when transmitting/receiving is in operation.

(g) <FDPXE>

Select whether using alignment function for transmit/receive per UNIT during full duplex.

Please don't change the setting of this register when transmitting/receiving is in operation.

(h)<TXMOD>

Select UNIT/Sequential transmission. During transmission, it is prohibited to change the transmission mode; Sequential→UNIT, UNIT → Sequential.

For UNIT transmit, the data in transmit FIFO is invalid. TEMP interrupt generates when the data is shifted from transmit data register (SPITD) to transmit buffer.

For sequential transmit, 32 bytes of the data in FIFO is valid. TEMP interrupt generates when the space of the FIFO becomes 16 bytes size and 32 bytes.

(i)<TXE>

Set enable/disable of transmit. Transmission starts when set to "1"after writing transmit data to transmit FIFO or set to "1" before writing transmit data to transmit FIFO. During transmission, it is possible to change enable/disable. If cleared to "0" during transmission, transmission is stopped after finishing transmitting the UNIT data in transmitting.

(j)<RXMOD>

Select UNIT/Sequential receives. During receiving, it is prohibited to change receiving mode; Sequential→UNIT, UNIT → Sequential

In UNIT receive mode, receive FIFO is invalid and RFUL interrupt generates when the received data is shifted from receive buffer to receive data register (SPIRD).

In sequential receive mode, receive FIFO is valid and RFUL interrupt generates when 16 and 32 bytes of the data is loaded to the FIFO.

(k) <RXE>

In UNIT receive mode, receives only 1 UNIT data by writing “1”.

When reading receive data register (SPIRD) with the condition “1”, receives one time additionally.

In sequential mode, receiving is kept sequentially until FIFO becomes full by writing “1”. During receiving, it is possible to change enable/disable. If writing “0”during receiving, receiving is stopped after finishing receiving the UNIT data in receiving.

[Transmit/Receive operation mode]

This SPI Controller supports 6 operations as below.

These are selected in <FDPXE>, <RXMOD>, <RXE>, <TXMOD>, <TXE> registers.

Table 3.17.2 Transmit/Receive operation mode

Operation mode	Register setting					Description
	<FDPXE>	<TXMOD>	<TXE>	<RXMOD>	<RXE>	
(1) UNIT transmit	0	0	1	x	x	Transmit written data per UNIT
(2) Sequential transmit	0	1	1	x	x	Transmit written data in FIFO sequentially
(3) UNIT receive	0	x	x	0	1	Receive only 1 UNIT of data
(4) Sequential receive	0	x	x	1	1	Receive automatically if buffer has space
(5) UNIT transmit/receive	1	0	1	0	1	Transmit/receive 1 UNIT of data with aligning transmit/receive data per each UNIT
(6)Sequential transmit/receive	1	1	1	1	1	Transmit/receive sequentially with aligning transmit/receive data per each UNIT

x: don’t care

Difference points between UNIT transmission and Sequential transmission

UNIT transmit mode can be selected by writing SPICT<TXMOD>= “0”.

The transmit FIFO is invalid in UNIT transmit mode. The UNIT transmit starts when writing UNIT data with the condition SPICT<TXE>= “1” or writing SPICT<TXE>= “1” after writing 1UNIT data in the transmit buffer. During transmission, it is prohibited to change the transmission mode;

For UNIT transmit, TEMP interrupt generates when the data is shifted from transmit data register (SPITD) to transmit buffer. TEND interrupt generates when the UNIT transmit is finished.

Sequential transmit mode can be selected by writing SPICT<TXMOD>= “1”. 32bytes of the FIFO becomes valid in sequential transmit mode. Writing data in transmit FIFO every 16 bytes is always needed. If writing other than 16 bytes, TEMP interrupt does not generate normally.

The written transmit data is shifted by turn with the condition SPICT<TXE>= “1”. Or shifted by turn when writing SPICT<TXE>= “1” after writing data in transmit FIFO.

The transmission is kept executing as long as data exists. Therefore the transmission can be kept sequentially while the transmit FIFO (32 bytes size) has no space. During transmission, it is prohibited to change;

Sequential transmit→UNIT transmit

UNIT transmit → Sequential transmit

During transmission, it is possible to change enable/disable. If writing SPICT<TXE>= “0” during transmission, transmission is stopped after finishing to transmit the UNIT data in transmitting.

TEMP interrupt generates when the space of FIFO becomes 16 and 32 bytes size. TEND interrupt generates when the UNIT transmit is finished.

Difference points between UNIT receive and Sequential receive

UNIT receive is the mode that receiving only 1 UNIT data. UNIT receive mode can be selected by writing SPICT<RXMOD>= “0”.

The receive FIFO is invalid in the UNIT receive mode. By writing SPICT<RXE>= “1”, receives 1UNIT data, loads received data in receive data register (SPIRD) and then stop receiving. Reading (SPIRD) register should be executed after writing SPICT<RXE>= “0”. If reading (SPIRD) register with the condition SPICT<RXE>= “1” 1 UNIT data is received again. During receiving, it is prohibited to change;

Sequential receive→UNIT receive

UNIT receive → Sequential receive

RFUL and REND interrupts generate when UNIT receiving is finished.

Sequential receive is the mode that receiving the data sequentially and automatically when receive FIFO has space. Sequential receive is selected by writing SPICT<RXMOD>= “1”.

The 32 bytes size of receive FIFO becomes valid in sequential receive mode. Reading the data in receive FIFO every 16 bytes is always needed. If reading other than 16 bytes, RFUL interrupt does not generate normally.

Received data is loaded to receive FIFO by writing SPICT<RXE>= “1”.

Receiving next data is kept automatically unless data receive FIFO becomes full (32bytes). Therefore receiving is not stopped every UNIT but kept sequentially. During receiving, it is prohibited to change receiving mode;

If writing SPICT<RXE>= “0” during receiving, receiving is stopped after finishing to receive the UNIT data in receiving.

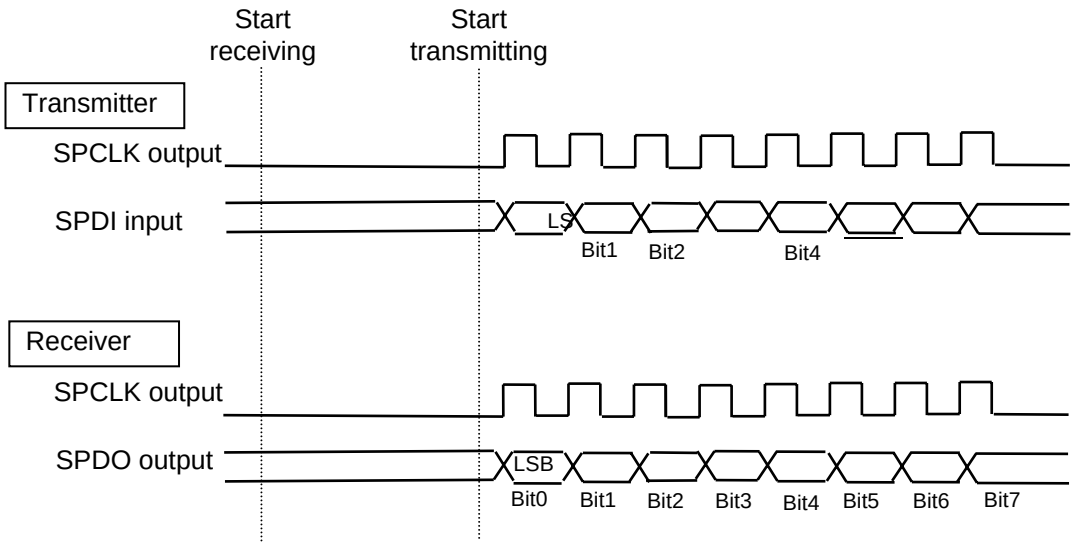
RFUL interrupt generates when 16 and 32 bytes of the data is loaded to the FIFO. REND interrupt generates when receiving 32 bytes size of the data is finished.

Transmit/Receive

When transmitting or receiving, write <FDPXE>= “1”

Writing <FDPXE>= “1” first, and SPICT<RXE>= “1” and keep waiting state for starting UNIT receiving. When writing SPICT<RXE>= “1”after <ALGNEN>= “1”, receiving does not start right away. This is because the data to transmit at the same time has not been prepared. Transmit/receive start when writing the data to (SPITD) register with the condition <TXE>= “1”.

The waveform of each transmit/receive operation is as follows;



Note: If transmit/receive are not operated simultaneously, please communicate with the condition <FDPXE>=”0”.

Figure 3.17.8 Transmit/Receive

(3) Interrupt

In INTC (interrupt controller), interrupt is divided roughly into 2 kinds; transmit interrupt (INTSPITX) and receive interrupt (INTSPIRX). Besides in this SPI circuit, there are 4 kinds of interrupts; 2 transmit interrupts 2 receive interrupts.

- **Transmit interrupt**

TEMP (Empty interrupt of transmit FIFO) and TEND (End interrupt of transmit).

As for TEMP interrupt, the timing of generation differs according to transmit mode; UNIT/sequential.

If transmit is sequencial, writing the data to transmit FIFO every 16 bytes is always needed. If writing other than 16 bytes, TEMP interrupt does not generate normally.

UNIT transmit mode

TEMP interrupt generates when the data is shift from transmit data register (SPITD) to transmit buffer since transmit FIFO is invalid.

TEND interrupt generates when the last UNIT transmit is finished (the falling edge of the last bit clock) with the FIFO empty.

Sequential transmit mode

TEMP interrupt generates from 2 phenomenon. One is when the space of FIFO becomes 16 bytes size and the other 32 bytes size.

TEND interrupt generates when the last UNIT transmit is finished (the falling edge of the last bit clock) with the FIFO empty.

- **Receive interrupt**

RFUL (Receive FIFO interrupt) and REND (Receive finish interrupt).

As for RFUL interrupt, the timing of generation differs according to receive mode; UNIT/sequential.

If transmit is sequencial, reading the data from receive FIFO every 16 bytes is always needed. If reading other than 16 bytes, RFUL interrupt does not generate normally.

UNIT receive

RFUL interrupt generates the same timing as REND since the receive FIFO becomes invalid.

RFUL and REND interrupt generate when the data is shifted from receive buffer to receive data register (SPIRD).

Sequential receive

RFUL interrupt generates from 2 phenomenon. One is when 16 bytes size of data is loaded to receive FIFO and the other 32 bytes size of data.

REND interrupt generates when the receive FIFO becomes full (32bytes).

(3-1) SPIST (SPI Status Register)

SPIST shows 4 statuses.

SPIST Register

		7	6	5	4	3	2	1	0
SPIST (824H)	bit Symbol					TEMP		TEND	REND
	Read/Write					R		R	
	After reset					1		1	0
	Function					Transmit FIFO Status 0: no space 1: having space		Transmit Status 0: during transmission or having transmission data 1: finish	Receive Status 0: during receiving or not having receiving data 1: finish or not having space
		15	14	13	12	11	10	9	8
(825H)	bit Symbol								
	Read/Write								
	After reset								
	Function								

Figure 3.17.9 SPIST Register

(a) <TEMP>

For UNIT transmission, it is cleared to “0” when valid data exists in transmit register (SPITD). It is set to “1” when no valid data exists.

For Sequential transmission, it is set to “1” when no valid data exists in transmit buffer.

(b) <TEND>

This bit is cleared to “0” when valid data to transmit exists in the shift register/FIFO buffer or when transmission. It is set to “1” when no valid data exists in the transmit data register/FIFO buffer and finish transmitting all the data.

(c) <REND>

For UNIT receiving, it is set to “1” when finish receiving and valid data was loaded to receive data register (when valid data exists). It is cleared to “0” when no valid data exists in receive register (SPIRD). It is set to “1” when no valid data exists or during receiving.

For Sequential receiving, it is set to “1” when valid data of 32 bytes exist in receive FIFO after finish receiving last data. It is cleared to “0” even if having space of 1byte.

RFUL flag does not exist because meaning is the same with REND flag.

(3-2) SPIIE(SPI Interrupt Enable Register)

SPIIE register is for enable 4 interrupts.

SPIIE Register

		7	6	5	4	3	2	1	0
SPIIE (82CH)	bit Symbol					TEMPIE	RFULIE	TENDIE	RENDIE
	Read/Write					R/W			
	After Reset					0	0	0	0
	Function					TEMP interrupt 0:enable 1:disable	RFUL interrupt 0:enable 1:disable	TEND interrupt 0:enable 1:disable	REND interrupt 0:enable 1:disable
		15	14	13	12	11	10	9	8
(82DH)	bit Symbol								
	Read/Write								
	After Reset								
	Function								

Figure 3.17.10 SPIIE Register

(a) <TEMPIE>

Set enable/disable of TEMP interrupt.

(b)<RFULIE>

Set enable/disable of RFUL interrupt.

(c)<TENDIE>

Set enable/disable of TEND interrupt.

(d)<RENDIE>

Set enable/disable of REND interrupt.

Note: As for 4 interrupts; 2 transmit interrupts (INTSPITX; TEMP, TEND) and 2 receive interrupts (INTSPIRX; RFUL, REND), it should be selected one from TEMP and TEND, one from RFUL and REND when using simultaneously. (Please do not select TEMP and TEND simultaneously. Or RFUL and REND simultaneously.)

(4) SPICR (SPI CRC Register)

CRC result of Transmit/Receive data is set to SPICR register.

SPICR Register

		7	6	5	4	3	2	1	0
SPICR (826H)	bit Symbol	CRCD7	CRCD6	CRCD5	CRCD4	CRCD3	CRCD2	CRCD1	CRCD0
	Read/Write	R							
	After Reset	0	0	0	0	0	0	0	0
	Function	CRC result register [7:0]							
		15	14	13	12	11	10	9	8
(827H)	bit Symbol	CRCD15	CRCD14	CRCD13	CRCD12	CRCD11	CRCD10	CRCD9	CRCD8
	Read/Write	R							
	After Reset	0	0	0	0	0	0	0	0
	Function	CRC result register [15:8]							

Figure 3.17.11 SPICR Register

(a) <CRCD15:0>

The result which is calculated according to the setting; SPICT<CRC16_7_b>, <CRCRX_TX_B> and <CRCRESET_B>, are loaded to this register.

In case CRC16, all bits are valid.

In case CRC7, lower 7 bits are valid.

The flow will be showed to calculate CRC16 of received data for instance by flowchart.

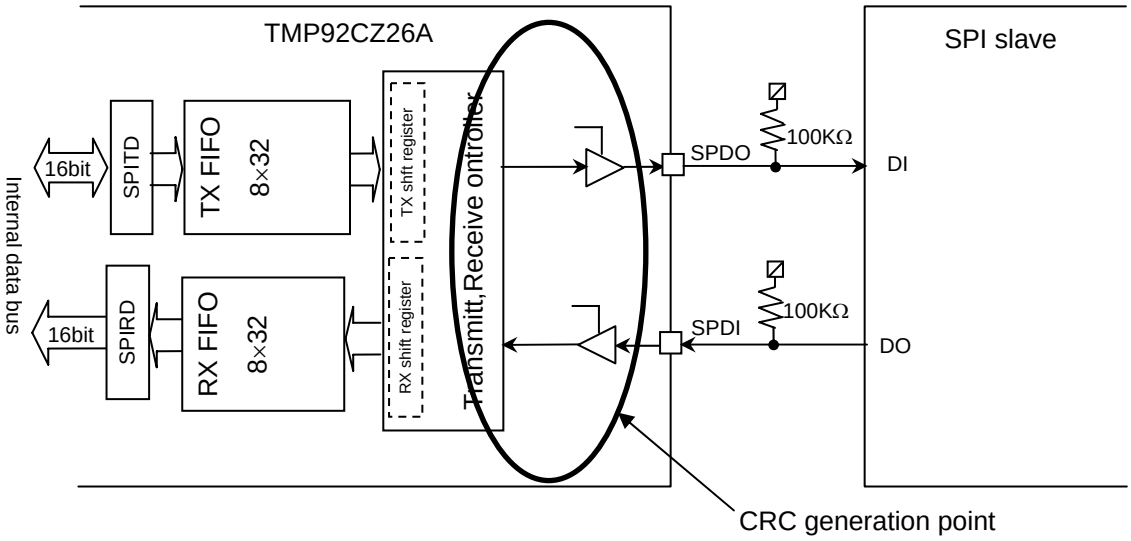
Firstly, initialize CRC calculation register by writing <CRCRESET_B>= “1” after setting <CRC16_7_b>= “1”, <CRCRX_TX_B>=”0”, <CRCRESET_B>= “0”.

Next, finish transmitting all bits to calculate CRC by writing data in SPITD register.

Please sense SPIST<TEND> to confirm whether receiving is finished.

If read SPICR register after finishing, CRC16 of received data can be read.

Note: CRC is generated in I/O point. Please take care soft ware process to compare the CRC when used FIFO.



(5) SPITD (SPI Transmit Data Register)

SPITD0, SPITD1 registers are for writing transmitted data.

SPITD0 Register

		7	6	5	4	3	2	1	0
SPITD0 (830H)	bit Symbol	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Transmit data register [7:0]							
		15	14	13	12	11	10	9	8
(831H)	bit Symbol	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Transmit data register [15:8]							

SPITD1 Register

		7	6	5	4	3	2	1	0
SPITD1 (832H)	bit Symbol	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Transmit data register [7:0]							
		15	14	13	12	11	10	9	8
(833H)	bit Symbol	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Transmit data register [15:8]							

Figure 3.17.12 SPITD Register

This bit is for writing transmitted data. When read, the last written data is read. The data is overwritten if write next data with transmit FIFO is not empty.

Transmit register exist 4bytes. Therefore, it is possible writing by using 4byte instruction (use DMA together it etc.)

However, when write data (Destination address), writing the data from 830 addresses is always needed.

Method of writing data (instruction) is restricted. Please refer to following table.

Transmit data write size	Instruction example	UNIT transmission (No using FIFO)		Sequential transmission (Using FIFO)	
		1byte transmission <unit16>=0	2 byte transmission <unit16>=1	1 byte transmission <unit16>=0	2 byte transmission <unit16>=1
1byte write	ld (0x830),a	○	×	Prohibit	×
2byte write	ld (0x830),wa	×	○	○	○
4byte write	ld (0x830),xwa	×	×	○	○

○: All data that written by CPU is transmitted

×: Invalid data that except for written by CPU is transmitted

(6) SPIRD (SPI Receive Data Register)

SPIRD0, SPIRD1 registers are for reading received data.

SPIRD0 Register

		7	6	5	4	3	2	1	0
SPIRD0 (834H)	bit Symbol	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	Receive data register [7:0]							
		15	14	13	12	11	10	9	8
(835H)	bit Symbol	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	Receive data register [15:8]							

SPIRD1 Register

		7	6	5	4	3	2	1	0
SPIRD1 (836H)	bit Symbol	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	Receive data register [7:0]							
		15	14	13	12	11	10	9	8
(837H)	bit Symbol	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0
	Function	Receive data register [15:8]							

Figure 3.17.13 SPIRD register

This bit is for reading received data. When read, read it after confirming status of RFUL or REND. The data is overwritten if write next data with transmit FIFO is not empty.

Receive register exist 4bytes. Therefore, it is possible reading by using 4byte instruction (use DMA together it etc.)

However, when read data basically, read the data from 834 addresses. (There is exception)

Method of reading data (instruction) is restricted. Please refer to following table.

Receive data read size	Instruction example	UNIT receiving (No using FIFO)		Sequential receiving (Using FIFO)	
		1byte receiving <unit16>=0	2 byte receiving <unit16>=1	1 byte receiving <unit16>=0	2 byte receiving <unit16>=1
1byte read	ld a,(0x834)	○	○	Prohibit	Prohibit
	ld a,(0x835)	×	○	Prohibit	Prohibit
2 byte read	ld wa,(0x834)	△*1	○	○	○
4 byte read	ld xwa,(0x834)	△*2	△*3	○	○

○: Read only valid data when CPU is reading.

△: Read valid data + invalid data when CPU is reading. Invalid data must be deleted after read.

×: Read only invalid data when CPU is reading.

*1: 834 address = valid data, 835 address = Invalid data,

*2: 834 address = valid data, 835 address = Invalid data, 836 address = Invalid data, 837 address = Invalid data

*3: 834 address = valid data, 835 address = valid data, 836 address = Invalid data, 837 address = Invalid data

- Note of FIFO buffer

There are following notes in this SPIC.

1) Transmit

- Data is overwritten if write data with condition transmit FIFO buffer is FULL. Interrupt and transmission are not executed normally because write-pointer in FIFO becomes abnormal condition. Therefore, manage number of writing by using software.
- If transmit is sequential, writing the data to transmit FIFO every 16 bytes is always needed. If writing other than 16 bytes, TEMP interrupt does not generate normally.

Note: If transmitting it by except 16 byte, use UNIT transmitting.

2) Receive

- If read data with condition receive FIFO is empty, undefined data is read. Interrupt and receiving are not executed normally because read-pointer in FIFO becomes abnormal condition. Therefore, manage number of reading by using software.
- If receive is sequential, reading the data from receive FIFO every 16 bytes is always needed. If reading other than 16 bytes, RFUL interrupt does not generate normally.

Note: If transmitting it by except 16 byte, use UNIT receiving.

3) CRC

CRC is generated in I/O point. Please take care soft ware process to compare the CRC when used FIFO.

Ex. Sequential receive

1. Start sequential receive
2. finish valid data receive (FIFO_Full)
3. disable receive
4. valid data read from FIFO to temporary buffer(internal RAM)
5. CRC1 read from CRC generator in SPI circuit
6. CRC2 receive (enable UNIT receive from SD-CARD)
7. compare CRC1 and CRC2

Note: Above 2 to 4 process can be used DMAC, however it must stop sequential receive (process 3) before to get CRC2 form SD-CARD.

3.18 I²S (Inter-IC Sound)

The TMP92CZ26A incorporates serial output circuitry that is compliant with the I²S format. This function enables the TMP92CZ26A to be used for digital audio systems by connecting an LSI for audio output such as a DA converter.

The I²S unit has the following features:

Table 3.18.1 I²S Operation Features

Item	Description
Number of Channels	2 channels
Format	I²S-format compliant Right-justified and left-justified formats supported Stereo / monaural Master transmission only
Pins used	1. I2SnCKO (clock output) 2. I2SnDO (output) 3. I2SnWS (Word Select output)
WS frequency	Refer to “Setting the transfer clock generator and Word Select signal”.
Data transfer rate	
Transmission buffer	64 bytes x 2
Direction of data	MSB-first or LSB-first selectable
Data length	8 bits or 16 bits
Clock edge	Rising edge or falling edge
Interrupt	INTI2Sn (64-byte FIFO empty interrupt)

3.18.1 Block Diagram

The I²S unit contains two channels: channel 0 and channel 1. Each channel can be controlled and made to output independently.

Figure 3.18.1 shows a block diagram for I²S channel 0.

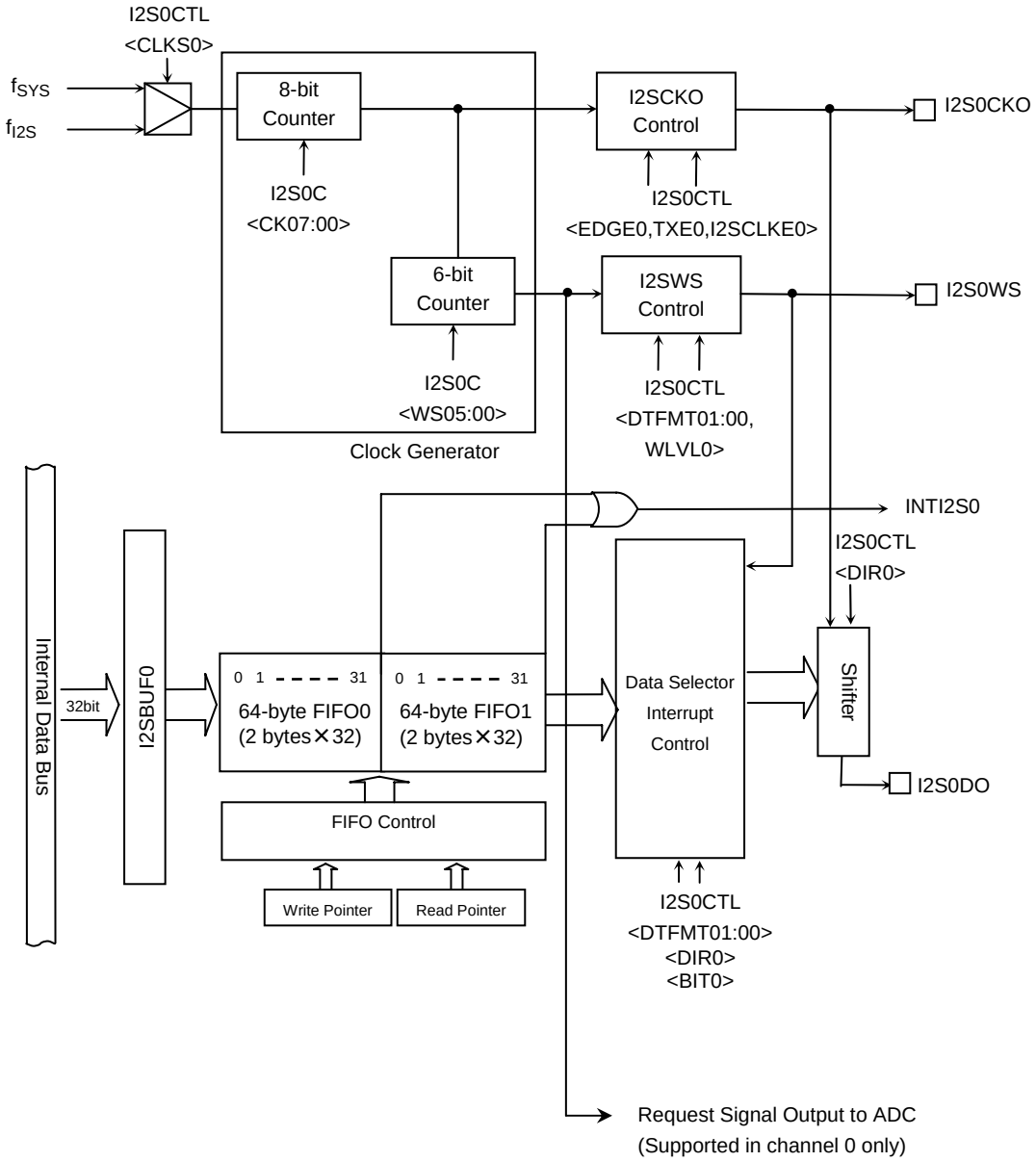


Figure 3.18.1 I²S Block Diagram

3.18.2 SFRs

The I²S unit is provided with the following registers. These registers are connected to the CPU via a 32-bit data bus. The transmission buffers I2S0BUF and I2S1BUF must be accessed using 4-byte load instructions.

I2S0 Control Register

		7	6	5	4	3	2	1	0
I2S0CTL (1808H)	bit Symbol	TXE0	*CNTE0		DIR0	BIT0	DTFMT01	DTFMT00	SYSCKE0
	Read/Write	R/W	R/W		R/W	R/W	R/W	R/W	R/W
	After reset	0	0		0	0	0	0	0
	Function	Transmission 0: Stop 1: Start	Counter control 0: Clear 1: Start		Transmission start bit 0:MSB 1:LSB	Bit length 0: 8 bits 1: 16 bits	Output format 00: I²S 10: Right 01: Left 11: Reserved		System clock 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(1809H)	bit Symbol	CLKS0			FSEL0	TEMP0	WLVL0	EDGE0	CLKE0
	Read/Write	R/W			R/W	R	R/W	R/W	R/W
	After reset	0			0	1	0	0	0
	Function	Source clock 0: f_{SYS} 1: f_{PLL}			Stereo /monaural 0: Stereo 1: Monaural	Transmission FIFO state 0: Data 1: No data	WS level 0: Low left 1: High left	Data output clock edge 0: Falling 1: Rising	Clock operation (after transmis-sion) 0: Enable 1: Disable

I2S0 Divider Value Setting Register

		7	6	5	4	3	2	1	0
I2S0C (180AH)	bit Symbol	CK07	CK06	CK05	CK04	CK03	CK02	CK01	CK00
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	Divider value for CK signal (8-bit counter)							
		15	14	13	12	11	10	9	8
(180BH)	Bit symbol			WS05	WS04	WS03	WS02	WS01	WS00
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	After reset			0	0	0	0	0	0
	Function			Divider value for WS signal (6-bit counter)					

I2S0 Buffer Register

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2S0BUF (1800H)	bit Symbol	B015	B014	B013	B012	B011	B010	B009	B008	B007	B006	B005	B004	B003	B002	B001	B000
	Read/Write	W															
Read-modify-write instructions cannot be used.	After reset	Undefined															
	Function	Transmission buffer register (FIFO)															
		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	bit Symbol	B031	B030	B09	B028	B027	B026	B025	B024	B023	B022	B021	B020	B019	B018	B017	B016
	Read/Write	W															
	After reset	Undefined															
	Function	Transmission buffer register (FIFO)															

Figure 3.18.2 I²S Channel 0 Control Registers

I2S1 Control Register

		7	6	5	4	3	2	1	0
I2S1CTL (1818H)	bit Symbol	TXE1	*CNTE1		DIR1	BIT1	DTFMT11	DTFMT10	SYSCKE1
	Read/Write	R/W	R/W		R/W	R/W	R/W	R/W	R/W
	After reset	0	0		0	0	0	0	0
	Function	Transmission 0: Stop 1: Start	Counter control 0: Clear 1: Start		Transmission start bit 0: MSB 1: LSB	Bit length 0: 8 bits 1:16 bits	Output format 00: I²S 10: Right 01: Left 11: Reserved		System clock 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(1819H)	bit Symbol	CLKS1			FSEL1	TEMP1	WLVL1	EDGE1	CLKE1
	Read/Write	R/W			R/W	R	R/W	R/W	R/W
	After reset	0			0	1	0	0	0
	Function	Source clock 0: f_{SYS} 1: f_{PLL}			Stereo /monaural 0: Stereo 1: Monaural	Transmission FIFO state 0: Data 1: No data	WS level 0: Low left 1: High left	Data output clock edge 0: Falling 1: Rising	Clock operation (after transmis-sion) 0: Enable 1: Disable

I2S1 Divider Value Setting Register

		7	6	5	4	3	2	1	0
I2S1C (181AH)	bit Symbol	CK17	CK16	CK15	CK14	CK13	CK12	CK11	CK10
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	Divider value for CK signal (8-bit counter)							
		15	14	13	12	11	10	9	8
(181BH)	Bit symbol			WS15	WS14	WS13	WS12	WS11	WS10
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	After reset			0	0	0	0	0	0
	Function			Divider value for WS signal (6-bit counter)					

I2S1 Buffer Register

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2S1BUF (1810H)	bit Symbol	B115	B114	B113	B112	B111	B110	B109	B108	B107	B106	B105	B104	B103	B102	B101	B100
	Read/Write	W															
	After reset	Undefined															
Read-modify-write instructions cannot be used.	Function	Transmission buffer register (FIFO)															
		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	bit Symbol	B131	B130	B129	B128	B127	B126	B125	B124	B123	B122	B121	B120	B119	B118	B117	B116
	Read/Write	W															
	After reset	Undefined															
	Function	Transmission buffer register (FIFO)															

Figure 3.18.3 I^2S Channel 1 Control Registers

3.18.3 Description of Operation

(1) Settings the transfer clock generator and Word Select signal

In the I²S unit, the clock frequencies for the I2SnCKO and I2SnWS signals are generated using the system clock (f_{SYS}) as a source clock. The system clock is divided by a prescaler and a dedicated clock generator to set the transfer clock and sampling frequency.

The counters are started by setting I2SnCTL<CNTEn> to "1" and are stopped and cleared by setting <CNTEn> to "0".

A) Clock generator

- 8-bit counter

  This is an 8-bit counter that generates the I2SnCKO signal by dividing the clock selected by I2SnCTL<CLKSn>.

- 6-bit counter

  This is a 6-bit counter that generates the I2SnWS signal by dividing the I2SnCKO signal.

B) Word Select

- Word Select signal (I2SnWS)

  The I2SnWS signal is used to distinguish the position of valid data and whether left data or right data is being transmitted in the I²S format. This signal is clocked out in synchronization with the data transfer clock. In only channel 0, this signal can be used as an AD conversion trigger signal for the ADC. How valid data is to be output in relation to the WS signal can be specified as I²S format, left-justified, or right-justified. In only channel 0, an interrupt request can be output to the ADC on the rising edge of the WS signal. (This is controlled by the ADC's control register.)

(2) Data format

This circuit support I2S format, left justify and right justify format by setting I2SnCTL<DTFMTn1:n0> register. And support stereo and monaural both, controlled by I2SnCTL<FSELn> register.

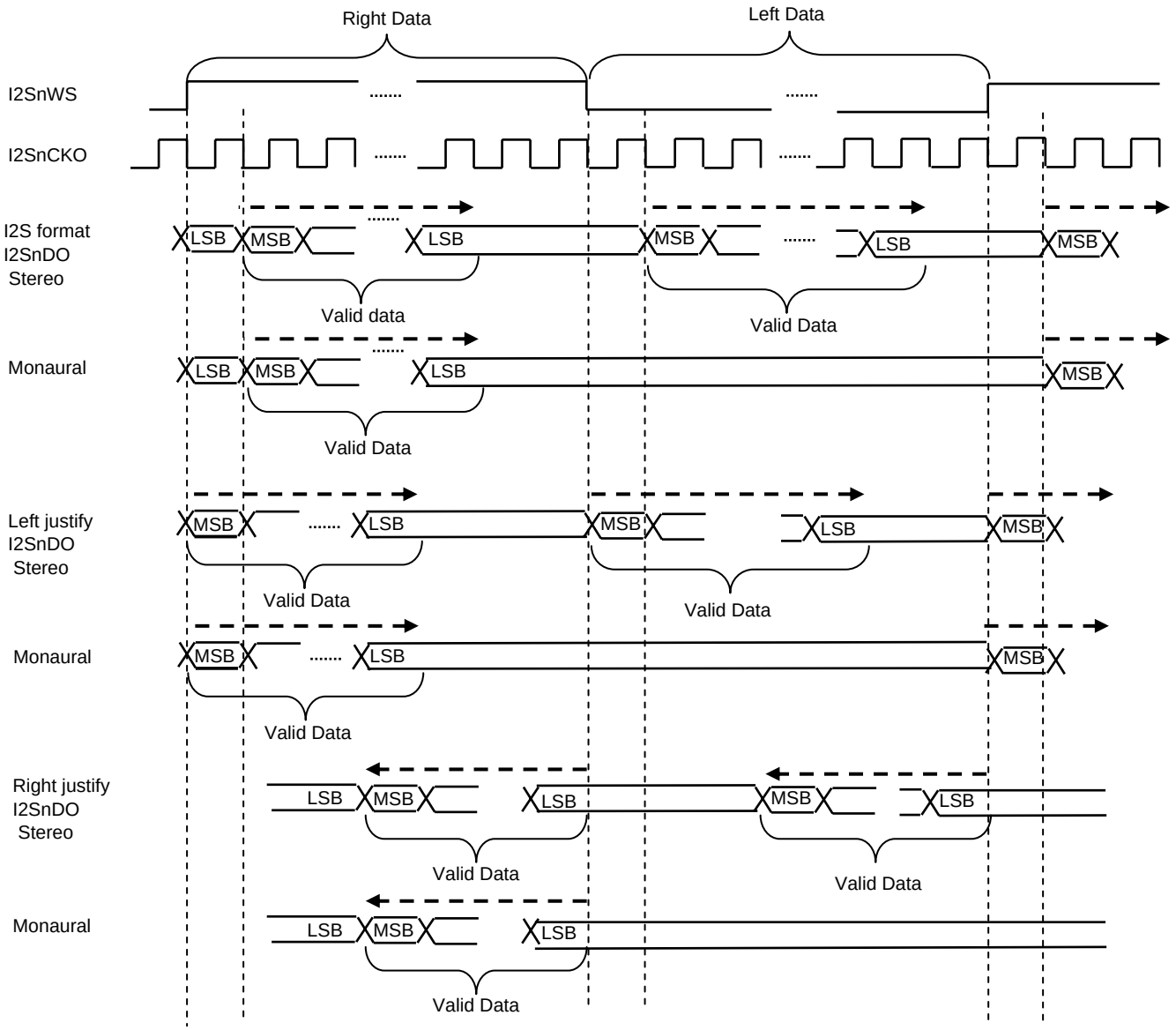


Figure 3.18.4 Output Format

(2) Setting example for the clock generator (8-bit counter/6-bit counter)

The clock generator generates the reference clock for setting the data transfer speed and sampling frequency.

		7	6	5	4	3	2	1	0
I2S0C (180AH)	bit Symbol	CK07	CK06	CK05	CK04	CK03	CK02	CK01	CK00
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	Divider value for CK signal (8-bit counter)							
		15	14	13	12	11	10	9	8
(180BH)	Bit symbol			WS05	WS04	WS03	WS02	WS01	WS00
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	After reset			0	0	0	0	0	0
	Function			Divider value for WS signal (6-bit counter)					

- Setting the transfer clock I2SnCKO

The transfer clock is generated by dividing the clock selected by I2SnCTL <CLKSn>. An 8-bit counter is provided to divide the source clock by 3 to 256. (The divider value cannot be set to 1 or 2.)

Note: The transfer clock must not exceed 10 MHz. Make sure that the transfer clock is set to within 10 MHz by an appropriate combination of source clock frequency and divider value.

8-bit counter set value	Divider value
00000000	256
00000001	1
11111111	255

When f_{SYS} = 60 MHz and I2SnC<CKn7:0> = 150, the data transfer speed is set as follows:

I2SnCKO = f_{SYS}/150

= 60 [MHz]/150 = 400 [kbps]

Note: It is recommended that the value to be set in I2SnC<CKn7:0> be an even number. Although it is possible to set an odd number, the clock duty of the CK signal does not become 50%. Setting an odd number causes the High width of the I2SnCK0 signal to become longer by one f_{sys} or f_{PLL} pulse than the Low width. (When <EDGE> = 0, the Low width becomes longer than the High width.)

- Setting the sampling frequency WS

The sampling frequency is set by dividing the transfer clock (CK) described above. A 6-bit counter is provided to divide the transfer clock by 16 to 64. (The divider value cannot be set to 1 to 15.)

6-bit counter set value	Divider value
000000	64
000001	1
111111	63

When f_{SYS} = 60 MHz, I2SnC<CKn7:0> = 150, and I2SnC<WSn5:0> = 50, the sampling frequency is set as follows:

I2SnCKO = f_{SYS} / 150 / 50

= 60 [MHz] / 150 / 50 = 8 [kHz]

Based on the above, the transfer clock is set to 400 kbps, and the sampling frequency is set to 8 kHz in this example.

Note 1: The value to be set in I2SnC<WSn5:0> must be 16 or larger (18 or larger for I2S transfer) when the data length is 8 bits and 32 or larger (34 or larger for I2S transfer) when the data length is 16 bits.

Note 2: It is recommended that the value to be set in I2SnC<WSn5:0> be an even number. Although it is possible to set an odd number, the clock duty of the WS signal does not become 50%. Setting an odd number causes the High width of the WS signal to become longer by one I2SnCK0 pulse than the Low width.

- Special function

  As a special function available only in channel 0, the rising edge of the WS signal can be used as an AD conversion start trigger for the AD converter in this LSI. Setting I2S0CTL<SYSKE0>=1 and I2S0CTL<CNTE0>=1 enables the WS signal to be sent to the AD converter. This can be done regardless of the setting of I2S0CTL<TXE0>.

  For details about AD conversion using the WS signal, refer to the chapter on the AD converter.

(3) FIFO buffer and data format

The I²S unit is provided with a 128-byte FIFO buffer (32-bit wide x 32-entry). The data written to the 4 bytes (32 bits) of the I2SnBUF register is written to this FIFO buffer. This FIFO must be written in units of 4 bytes. It is also necessary to consider the output order and to distinguish between right data and left data.

To write data to the I2SnBUF register, be sure to use a 4-byte load instruction. If a 1-byte load instruction is used, invalid data will be transmitted. In case of using 1-byte or 2-byte transmission instruction, FIFO buffer isn't renewed and transmission isn't started.

And window addresses are 1800H (channel 0) and 1810H (channel1).

Write Data Size	Example instruction	8-bit width	16-bit width
1-byte access	ld (0x1800),a	Not allowed	Not allowed
2-byte access	ld (0x1800),wa	Not allowed	Not allowed
4-byte access	ld (0x1800),xwa	OK	OK

Also note that data must be written in units of 64 bytes using the following sequence:

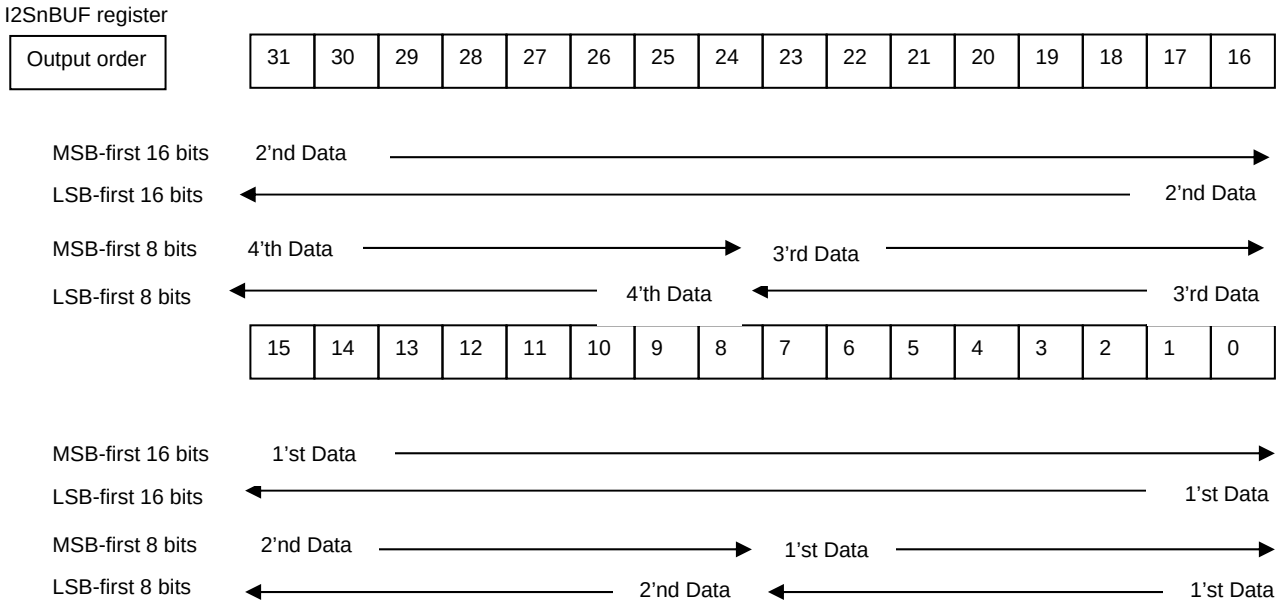
4-byte load instruction × 16 times = 64-byte data write

If data is not written in units of 64 bytes, interrupts cannot be generated at the normal timing.

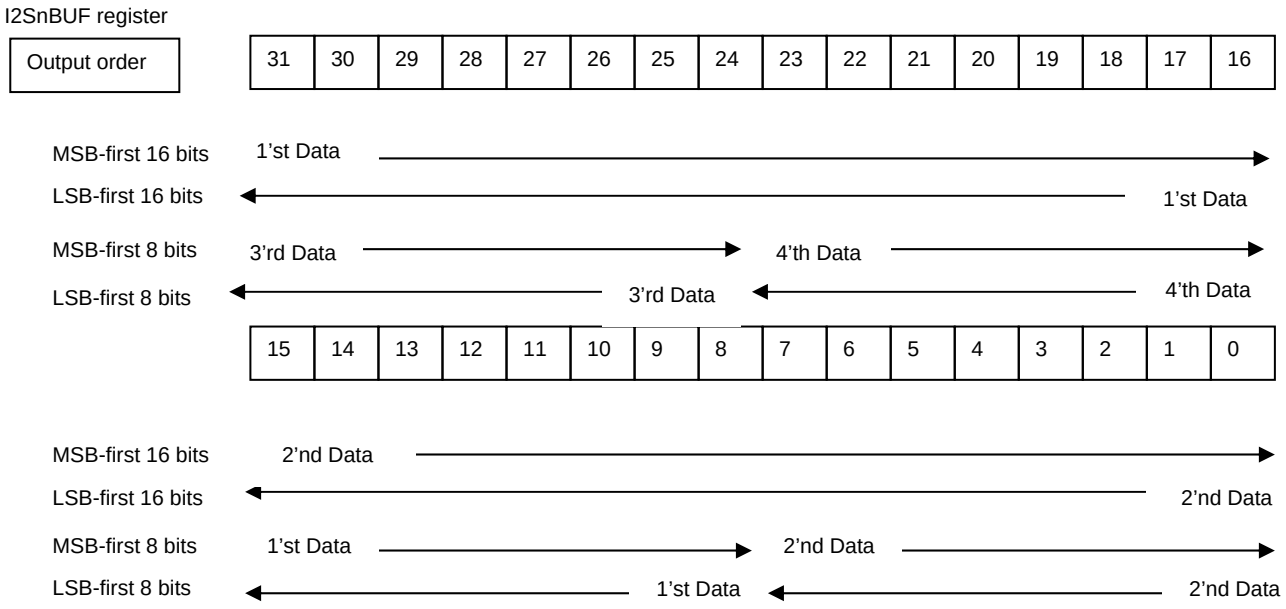
The I2SnCTL<TEMPn> flag is set to “1” when the FIFO buffer for each channel contains no valid data. If there is even one byte of valid data in the FIFO, the flag is cleared to “0”. (The <TEMPn> flag is set to “1” as soon as the last valid data in the FIFO is sent to the transmission shift register.)

The following shows how written data is output under various conditions.

When I2SnCTL<WLVLn> = 0



When I2SnCTL<WLVLn> = 1

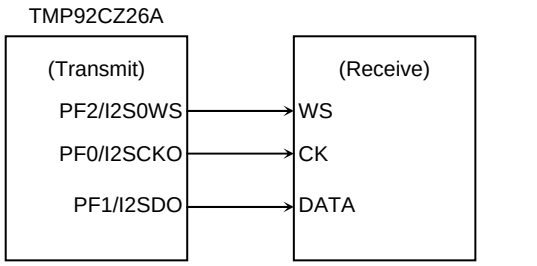


Note: In case of using monaural setting, and change right / left: I2SnCTL<WLVLn>, data output order change off 1'st data and 2'nd data.

3.18.4 Detailed Description of Operation

(1) Connection example

Figure 3.18.5 shows an example of connections between the TMP92CZ26A and an external LSI (DA converter) using channel 0.



Example: DA converter

Note: After reset, PF0 to PF2 are placed in a high-impedance state. Connect each pin with a pull-up or pull-down resistor as necessary.

Figure 3.18.5 Connection Example between the TMP92CZ26A and an External LSI

(2) Operation procedure

The I²S unit incorporates a 128-byte FIFO buffer that is divided into two 64-byte units. Whenever each 64-byte buffer space becomes empty, an INTI2Sn interrupt is generated. The next data to be transmitted should be written to the FIFO in the interrupt routine.

Example settings and timing diagram are shown below.

(Example settings) I2S0WS = 8 KHz, I2SnCKO = 400 kHz, data transmission on the rising edge (at f_{SYS} = 50 MHz)

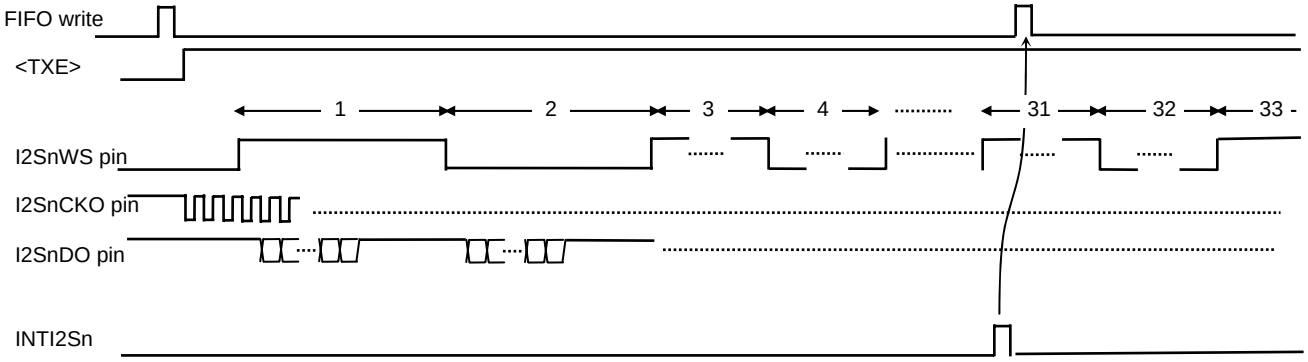
(Main routine)

	7	6	5	4	3	2	1	0	
INTEI2S01	X	–	–	–	X	0	0	1	Set interrupt level.
PFCR	X	X	–	–	–	–	–	–	Set pins: PF0 (I2S0CKO), PF1 (I2S0DO), PF2 (I2S0WS)
PFFC	–	X	–	–	–	1	1	1	
I2S0SC	1	0	0	1	0	1	1	0	Divider value N=150
	X	X	1	1	0	0	1	0	Divider value K=50
I2S0CTL	0	0	X	0	1	0	0	1	Set transmit mode (I^2S mode, MSB-first, 16-bit).
	0	X	X	X	X	0	0	0	Falling edge, WS=0 Left, clock stop.
I2S0BUF	*	*	*	*	*	*	*	*	Write left and right data to FIFO (4 bytes x 32 = 128 bytes).
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	
I2S0CTL	1	0	X	0	1	0	0	1	Start transmission.
	0	X	X	X	X	0	0	0	

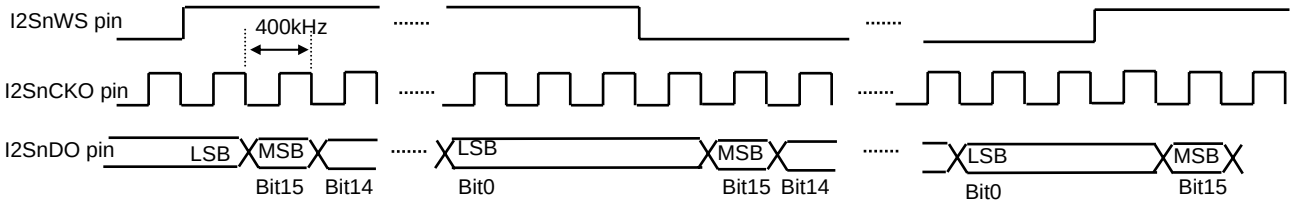
(INTI2S Interrupt Routine)

I2S0BUF	*	*	*	*	*	*	*	*	Write left and right data to FIFO (4 bytes x 16 = 64 bytes).
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	

X: Don't care, –: No change



Overall Timing Diagram



Detailed Timing Diagram

Figure 3.18.6 Timing Diagrams (I2S FMT/Stereo/16bit/MSB first)

(3) Considerations for using the I²S unit

1) INTI2Sn generation timing

Every 4bytes data trance from FIFO buffer to shift register per one time.

An INTI2Sn interrupt is generated under two conditions. One is when there are 64 bytes of empty space in the FIFO (after 61- 64th byte has been transferred to the shift register). The other is when the FIFO becomes completely empty (after 125 - 128th byte has been transferred to the shift register). Therefore, INTI2Sn indicates that there are 64 bytes or 128 bytes of empty space in the FIFO, enabling the next data to be written.

The FIFO must be written in units of 64 bytes. Since the FIFO can contain 128 bytes of data, I²S output can be performed continuously as long as there are 64 bytes of data in the FIFO. It is also possible to check the FIFO state by using the I2SnCTL<TEMPn> flag.

2) I2SnCTL<TXEn>

Transmission is started by setting I2SnCTL <TXEn> to “1”. Once <TXEn> is set to “1”, transmission is continued automatically as long as the FIFO contains the data to be transmitted. While <TXE> is set to “1” (transmission in progress), the other bits in the I2SnCTL register must not be changed.

To stop transmission, make sure that the FIFO is empty by checking the I2SnCTL<TEMPn> flag. Then, after waiting for two periods of the I2SWS signal (after all the data has been transmitted), set <TXEn> to “0”. In case monaural setting, make sure that the FIFO is empty by checking the I2SnCTL<TEMPn> flag. Then, after waiting for four periods of the I2SWS signal (after all the data has been transmitted), set <TXEn> to “0”.

If <TXEn> is set to “0” while data is being transmitted, the transmission is stopped

immediately. At the same time, the read and write pointers of the FIFO, the data in the output shift register and the clock generator are all cleared. (However, when I2SnCTL<CNTEn>=1, the clock generator is not cleared. To clear the clock generator, I2SnCTL<CNTEn> must be set to "0"). Therefore, if transmission is stopped and then resumed, no data will be output.

The WS signal stops at Low level and the CK signal stops at Low level when the rising edge is selected and at High level when the falling edge is selected.

3) I2SnCTL<CNTEn>

I2SnCTL<CNTEn> is used to control the clock generator (8-bit counter, 6-bit counter) for generating the I2SnCKO and I2SnWSOsignals.

Setting I2SnCTL<CNTEn> to "1" starts the counters, and setting this bit to "0" stops the counters. Normally, I²S data transmission is executed by setting both I2SnCTL<TXEn> and <CNTEn> to "1". When transmission is stopped by setting I2SnCTL<TXEn> to "0" with I2SnCTL<CNTEn>=1, the clock generator is not cleared. To clear the clock generator, I2SnCTL<CNTEn> must be set to "0".

4) FIFO buffer

The I²S unit is provided with a 128-byte FIFO. Although it is not necessary to use all 128 bytes in the FIFO, data should basically be written in units of 64 bytes using an INTI2Sn interrupt as a trigger. If data is written to the FIFO without waiting for an INTI2Sn interrupt or in units other than 64 bytes, interrupts cannot be generated properly.

If the last set of data, for which an interrupt is not needed, contains less than 64 bytes, set I2SnCTL<TXEn> to "0" to stop the transmission after writing the data, then checking that the <TEMPn> flag is set to "1", and waiting for two I2SWS periods (i.e., after all the data has been transmitted). In case monaural setting, make sure that the FIFO is empty by checking the I2SnCTL<TEMPn> flag. Then, after waiting for four periods of the I2SWS signal (after all the data has been transmitted), set <TXEn> to "0".

5) I2SnBUF

When writing data to the I2SnBUF register, be sure to use long-word data load instructions. Word data load or byte data load instructions cannot be used.

Examples)

```
ld      (I2SnBUF), xwa;     OK
ld      (I2SnBUF), wa;      NG
ld      (I2SnBUF), a;       NG
```

3.19 LCD Controller (LCDC)

The TMP92CZ26A incorporates an LCD controller (LCDC) for controlling an LCD driver LSI (LCD module). This LCDC supports display sizes from 64 × 64 to 640 × 480 dots for monochrome, grayscale, and 4096-color display and from 64 × 64 to 320 × 320 dots for color display using 65536 or more colors. The supported LCD driver (LCD module) types are STN (Super Twisted Nematic) and digital RGB input TFT (Thin Film Transistor).

- STN support

With LCD drivers supporting STN, an 8-bit data interface is used to realize monochrome, 4-graysale, 16-grayscale, 64-grayscale, 256-color, 4096-color display.

After required settings such as the operation mode, display RAM start address, and LCD size (common, segment) are made in the I/O registers, the start register is set to enable the LCDC. The LCDC outputs a bus request to the CPU, reads data from the display RAM, converts the data as necessary, and writes it to a dedicated FIFO buffer.

- TFT support

With LCD drivers supporting digital RGB input TFT, an 8- to 24-bit data interface is used to realize 4096-color, 65536-color, 262144-color, and 16777216-color display. The data transfer method is the same as in the case of STN.

The LCDC controls LCD display operations using 8-bit RGB (R3:G3:B2), 12-bit RGB (R4:G4:B4), 16-bit RGB (R5:G6:B5), 18-bit RGB (R6:G6:B6), or 24-bit RGB (R8:G8:B8) display data, the shift clock LCP0 for capturing data, the frame signal LFR, the data load signal LLOAD, and the LDIV signal for indicating the inversion of data output. The LDIV signal can be used effectively in reducing noise and power consumption.

The LCDC also has horizontal synchronization signal LHSYNC and vertical synchronization signal LVSYNC for controlling gate drivers, and three programmable OE pins for supporting various signals of the TFT driver to be used.

3.19.1 LCDC Features according to LCD Driver Type

Table 3.19.1 LCDC Features according to LCD Driver Type

(This table assumes the connection with a TOSHIBA-made LCD driver.)

LCD Driver		Shift Register Type	
		TFT	STN
Display colors		256/4096/65536/262144/16777216 colors	Monochrome, 4/16/64 grayscale levels 256/4096 colors
Number of pixels that can be displayed		**For 65536 colors or less** Rows (Commons): 64, 96, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 128, 160, 240, 320, 640	Rows (Commons): 64, 96, 120, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 120, 128, 160, 240, 320, 480, 640
		For 65536 colors or more Rows (Commons): 64, 96, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 128, 160, 240, 320	–
Data rotation function		Horizontal flip, vertical flip, horizontal and vertical flip, 90-degree rotation (supported for QVGA size, 65536 colors only)	
PIP function support		A sub window can be inserted.	
Source data bus width (SRAM, SDRAM)		16 bits (32 bits: internal RAM)	16 bits (32 bits: internal RAM)
Destination data bus width (LCD driver)		8 to 24 bits	8 bits
Maximum transfer rate (VRAM read) (at f_{SYS} = 80 MHz)		4.17 ns/byte at internal RAM	
External Pins	LCD driver data bus: LD23 to LD0 pins	To be connected to LCD driver data bus. ・8-bit mode: LD7 to LD0 ・TFT mode: LD23 to LD0	
	LCP0 pin	Data shift clock for TFT source driver	Shift clock pulse output pin 0. To be connected to column driver's CP pin. The LCD driver latches the data bus value on the falling edge of this pin.
	LHSYNC pin	Vertical shift clock for TFT gate driver	Latch pulse output pin. To be connected to the LCD driver's LP pin. The display data in the LCD driver's output line register is updated on the rising edge of this pin.
	LLOAD pin	Enable signal for TFT source driver to load data to TFT panel	N/A
	LGOE0 to LGOE2 pins	Adjustment signal for TFT gate driver's gate control signal	N/A
	LFR pin	LCD alternate signal output pin. To be connected to column/row driver's FR pin.	LCD alternate signal output pin. To be connected to column/row driver's FR pin.
	LVSYNC pin	This signal indicates the start of shift clock capture by TFT gate driver.	Frequency that sets LCD refresh rate
	LDIV pin	This signal indicates the inversion of data. To be connected to TFT source driver having the data inversion function.	N/A

3.19.2 SFRs

LCDMODE0 Register

LCDMODE0 (0280H)

	7	6	5	4	3	2	1	0
bit Symbol	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	1	1	0	0	0	0
Function	Display RAM 00: Internal RAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transfer speed SCPW2= 0 00: 2-clk 01: 4-clk 10: 8-clk 11: 16-clk SCPW2= 1 00: 6-clk 01: 12-clk 10: 24-clk 11: 48-clk		Mode selection 0000: Reserved 1000: Reserved 0001: SR (mono) 1001: Reserved 0010: SR (4-gray) 1010: TFT (256-color) 0011: Reserved 1011: TFT (4096-color) 0100: SR (16-gray) 1100: TFT (64K-color) 0101: SR (64-gray) 1101:TFT(256K-,16M-color) 0110: STN (256-color) 1110 : Reserved 0111:STN (4096-color)1111: Reserved		

Note: When SDRAM is used as the LCDC's display RAM, it can only be accessed by "burst 1-clock access".

LCDMODE1 Register

LCDMODE1 (0281H)

	7	6	5	4	3	2	1	0
bit Symbol	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	W	W
After reset	0	0	0	0	0	0	0	0
Function	Data rotation function (Supported for 64K-color: 16bps only) 000: Normal 100: 90-degree 001: Horizontal flip 101: Reserved 010: Vertical flip 110: Reserved 011: Horizontal & vertical flip 111: Reserved			LD bus inversion 0: Normal 1: Invert	Auto bus inversion 0: Disable 1: Enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	LFR edge 0: LHSYNC Front Edge 1:LHSYNC Rear Edge	LD bus Trance Speed 0: normal 1: 1/3

Note: <LDINV>=1 inverts all output data on the LD bus. However, the LDIV signal that indicates the inversion of output data by auto bus inversion remains unchanged.

LCD Size Setting Register

LCDSIZE (0284H)

	7	6	5	4	3	2	1	0
bit Symbol	COM3	COM2	COM1	COM0	SEG3	SEG2	SEG1	SEG0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	Common setting 0000: Reserved 1000: 320 0001: 64 1001: 480 0010: 96 1010: Reserved 0011: 120 1011: Reserved 0100: 128 1100: Reserved 0101: 160 1101: Reserved 0110: 200 1110: Reserved 0111: 240 1111: Reserved				Segment setting 0000: Reserved 1000: Reserved 0001: 64 1001: Reserved 0010: 128 1010: Reserved 0011: 160 1011: Reserved 0100: 240 1100: Reserved 0101: 320 1101: Reserved 0110: 480 1110: Reserved 0111: 640 1111: Reserved			

Note: Although the TMP92CZ26A contains 288 Kbytes of RAM that can be used as display RAM, it may not be enough depending on display size and color mode.

LCD Control 0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
Read/Write	R/W	R/W	R/W	R/W		R/W	R/W	R/W
After reset	0	0	0	0		0	0	0
Function	PIP function 0:Disable 1:Enable	Segment data 0: Normal 1: Always output “0”	FR divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0(Note 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

LCD Control 1 Register

LCDCTL1 (0286H)

	7	6	5	4	3	2	1	0
bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
Read/Write	R/W	R/W	R/W	R/W			R/W	R/W
After reset	1	0	1	0			0	0
Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	

LCD Control 2 Register

LCDCTL2 (0287H)

	7	6	5	4	3	2	1	0
bit Symbol	LGOE2P	LGOE1P	LGOE0P					
Read/Write	R/W	R/W	R/W					
After reset	0	0	0					
Function	LGOE2 phase 0: Rising 1: Falling	LGOE1 phase 0: Rising 1: Falling	LGOE0 phase 0: Rising 1: Falling					

Divide FRM 0 Register

LCDDVM0 (0283H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP3	FMP2	FMP1	FMP0	FML3	FML2	FML1	FML0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 3-0)				LHSYNC DVM (bits 3-0)			

Divide FRM 1 Register

LCDDVM1 (0288H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP7	FMP6	FMP5	FMP4	FML7	FML6	FML5	FML4
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 7-4)				LHSYNC DVM (bit 7-4)			

LCD LHSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDHSP (028AH)	bit Symbol	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 7–0)							
		7	6	5	4	3	2	1	0
(028BH)	bit Symbol	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 15-8)							

LCD V SYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDVSP (028CH)	bit Symbol	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LVSYNC period (bits 7-0)							
		7	6	5	4	3	2	1	0
(028DH)	bit Symbol							LVP9	LVP8
	Read/Write							W	
	After reset							0	0
	Function							LVSYNC period (bits 9-8)	

LCD LVSYNC Pre Pulse Register

		7	6	5	4	3	2	1	0
LCDPRVSP (028EH)	bit Symbol		PLV6	PLV5	PLV4	PLV3	PLV2	PLV1	PLV0
	Read/Write		W						
	After reset		0	0	0	0	0	0	0
	Function		Front dummy LVSYNC (bits 6-0)						

LCDHSDLY (028FH)

	7	6	5	4	3	2	1	0
bit Symbol		HSD6	HSD5	HSD4	HSD3	HSD2	HSD1	HSD0
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		LHSYNC delay (bits 6-0)						

LCDLDDLY (0290H)

	7	6	5	4	3	2	1	0
bit Symbol	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
Read/Write	R/W	W						
After reset	0	0	0	0	0	0	0	0
Function	Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

LCDO0DLY (0291H)

	7	6	5	4	3	2	1	0
bit Symbol		OE0D6	OE0D5	OE0D4	OE0D3	OE0D2	OE0D1	OE0D0
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		OE0 delay (bits 6-0)						

LCDO1DLY (0292H)

	7	6	5	4	3	2	1	0
bit Symbol		OE1D6	OE1D5	OE1D4	OE1D3	OE1D2	OE1D1	OE1D0
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		OE1 delay (bits 6-0)						

LCDO2DLY (0293H)

	7	6	5	4	3	2	1	0
bit Symbol		OE2D6	OE2D5	OE2D4	OE2D3	OE2D2	OE2D1	OE2D0
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		OE2 delay (bits 6-0)						

LCDHSW (0294H)

	7	6	5	4	3	2	1	0
bit Symbol	HSW7	HSW6	HSW5	HSW4	HSW3	HSW2	HSW1	HSW0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LHSYNC width (bits 7-0)							

LCDLDW (0295H)

	7	6	5	4	3	2	1	0
bit Symbol	LDW7	LDW6	LDW5	LDW4	LDW3	LDW2	LDW1	LDW0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LLOAD width (bits 7-0)							

LCDHO0W (0296H)

	7	6	5	4	3	2	1	0
bit Symbol	O0W7	O0W6	O0W5	O0W4	O0W3	O0W2	O0W1	O0W0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE0 width (bits 7-0)							

LCDHO1W (0297H)

	7	6	5	4	3	2	1	0
bit Symbol	O1W7	O1W6	O1W5	O1W4	O1W3	O1W2	O1W1	O1W0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE1 width (bits 7-0)							

LCDHO2W (0298H)

	7	6	5	4	3	2	1	0
bit Symbol	O2W7	O2W6	O2W5	O2W4	O2W3	O2W2	O2W1	O2W0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE2 width (bits 7-0)							

LCDHWB8 (0299H)

	7	6	5	4	3	2	1	0
bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

LCD Main Area Start Address Register

		7	6	5	4	3	2	1	0
LSAML (02A0H)	bit Symbol	LMSA7	LMSA6	LMSA5	LMSA4	LMSA3	LMSA2	LMSA1	
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	After reset	0	0	0	0	0	0	0	
	Function	LCD main area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSAMM (02A1H)	bit Symbol	LMSA15	LMSA14	LMSA13	LMSA12	LMSA11	LMSA10	LMSA9	LMSA8
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD main area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSAMH (02A2H)	bit Symbol	LMSA23	LMSA22	LMSA21	LMSA20	LMSA19	LMSA18	LMSA17	LMSA16
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	1	0	0	0	0	0	0
	Function	LCD main area start address (A23-A16)							

Note: When assigned internal RAM as VRAM, A1 signal cannot be used. Every 4bytes setting is needed.

LCD Sub Area Start Address Register

		7	6	5	4	3	2	1	0
LSASL (02A4H)	bit Symbol	LSSA7	LSSA6	LSSA5	LSSA4	LSSA3	LSSA2	LSSA1	
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	After reset	0	0	0	0	0	0	0	
	Function	LCD sub area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSASM (02A5H)	bit Symbol	LSSA15	LSSA14	LSSA13	LSSA12	LSSA11	LSSA10	LSSA9	LSSA8
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSASH (02A6H)	bit Symbol	LSSA23	LSSA22	LSSA21	LSSA20	LSSA19	LSSA18	LSSA17	LSSA16
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	1	0	0	0	0	0	0
	Function	LCD sub area start address (A23-A16)							

Note: When assigned internal RAM as VRAM, A1 signal cannot be used. Every 4bytes setting is needed.

LCD Sub Area HOT Point Register (X-dir)

		7	6	5	4	3	2	1	0
LSAHX	bit Symbol	SAHX7	SAHX6	SAHX5	SAHX4	SAHX3	SAHX2	SAHX1	SAHX0
(02A8H)	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02A9H)	bit Symbol							SAHX9	SAHX8
	Read/Write							R/W	R/W
	After reset							0	0
	Function							LCD sub area HOT point (9-8)	

LCD Sub Area HOT Point Register (Y-dir)

		7	6	5	4	3	2	1	0
LSAHY	bit Symbol	SAHY7	SAHY6	SAHY5	SAHY4	SAHY3	SAHY2	SAHY1	SAHY0
(02AAH)	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02ABH)	bit Symbol								SAHY8
	Read/Write								R/W
	After reset								0
	Function								LCD sub area HOT point (8)

LCD Sub Area Display Segment Size Register

		7	6	5	4	3	2	1	0
LSASS	bit Symbol	SAS7	SAS6	SAS5	SAS4	SAS3	SAS2	SAS1	SAS0
(02ACH)	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area segment size (7-0)							
		7	6	5	4	3	2	1	0
(02ADH)	bit Symbol							SAS9	SAS8
	Read/Write							R/W	R/W
	After reset							0	0
	Function							LCD sub area segment size (9-8)	

LCD Sub Area Display Common Size Register

		7	6	5	4	3	2	1	0
LSACS	bit Symbol	SAC7	SAC6	SAC5	SAC4	SAC3	SAC2	SAC1	SAC0
(02AEH)	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area common size (7-0)							
		7	6	5	4	3	2	1	0
(02AFH)	bit Symbol								SAC8
	Read/Write								R/W
	After reset								0
	Function	LCD sub area common size (8)							

3.19.3 Description of Operation

3.19.3.1 Outline

After the required settings such as the operation mode, display data memory address, color mode, and LCD size are specified, the start register is set to start the LCDC operation.

The LCDC issues a bus request to the CPU. When the bus is granted, the LCDC reads data of the display size from the display RAM, stores the data in the FIFO buffer in the LCDC, and then returns the bus to the CPU.

The display data in the FIFO buffer is transferred to the LCD driver via a dedicated bus (LD pin). At this time, control pins (such as LCP0) that are connected to the LCD driver also output specified waveforms in synchronization with the transfer of display data.

Note: While display RAM data is being read, the CPU operation is halted by the internal BUSREQ signal. Therefore, the CPU stop time must be taken into account in programming.

External SDRAM, SRAM, or internal RAM (288 Kbytes) can be used as the display RAM. Since the internal RAM allows very fast accesses (32-bit bus, 2-1-1-1 read/write), it enables data transfer to the LCD driver (DMA operation) with the minimum CPU stop time. Using the internal RAM also greatly reduces power consumption during LCD display.

3.19.3.2 Display Memory Mapping

Since the number of bits needed to display one pixel varies even for the same display size depending on the selected color mode, the required display RAM size also varies with each color mode. (The color mode can be selected from a range of monochrome to 16777216 colors.)

In monochrome mode, one pixel of display data corresponds to one bit of display RAM data. Likewise, the number of display RAM data used for displaying one pixel in each color mode is as follows:

4-grayscale	1 pixel = 2 bits
16-grayscale	1 pixel = 4 bits
64-grayscale	1 pixel = 6 bits
STN 256-color	1 pixel = 8 bits
STN 4096-color	1 pixel = 12 bits
STN 65536-color	1 pixel = 16 bits
STN 256K-color	1 pixel = 16 bits (not 18 bits)
STN 16M-color	1 pixel = 24 bits

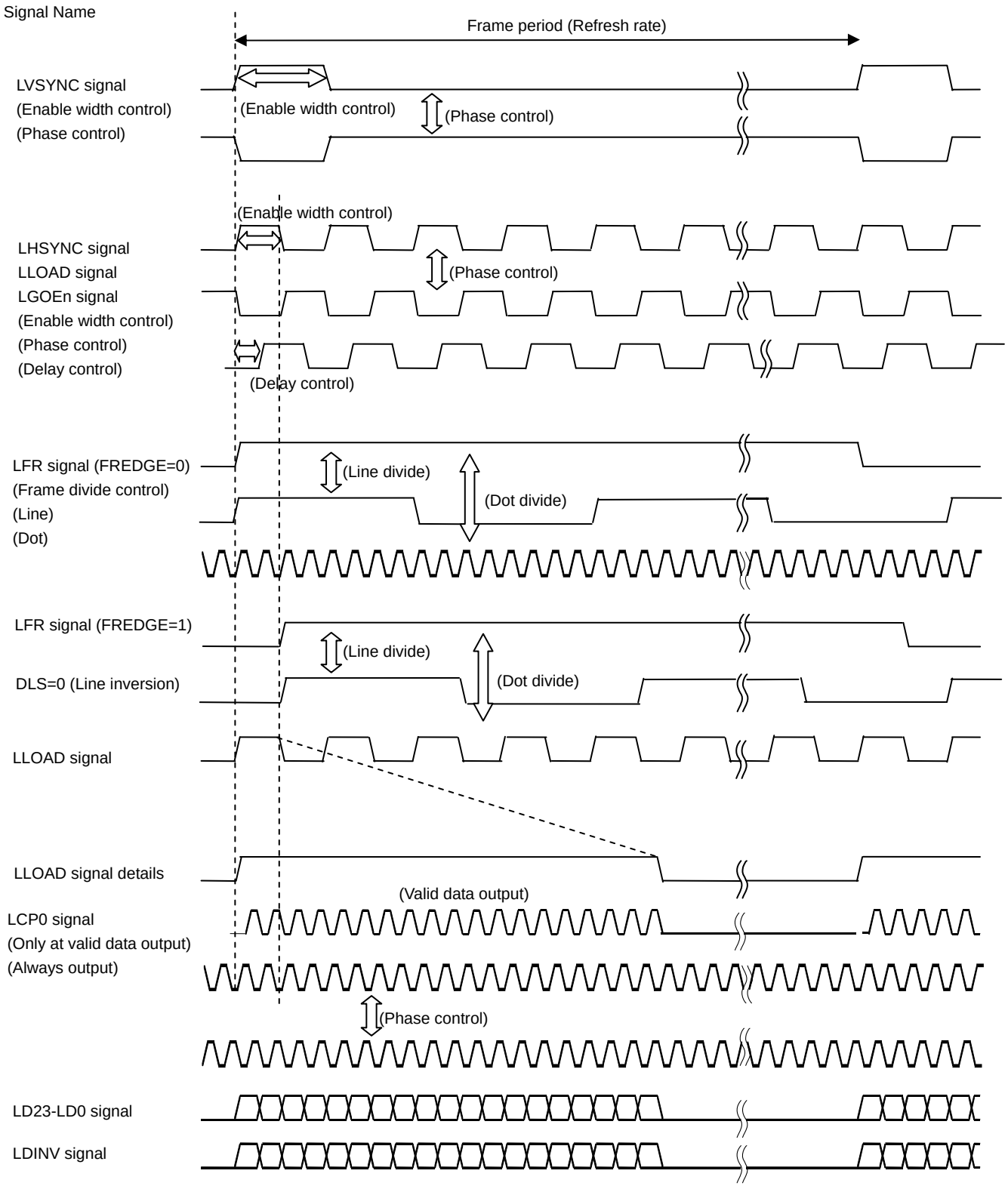
For example, a 320-segment x 240-common display in 4-grayscale mode requires 19200 bytes of display RAM space (320 × 240 × 2 = 152600 bits = 19200 bytes).

For details, refer to “Memory Map Image and Data Output in Each Display Mode” later in this chapter.

3.19.3.3 Basic Operation

The following diagram shows the basic timings of the waveforms generated by the LCDC and adjustable elements. The adjustable elements for each signal include enable time, phase, and delay time.

The signals used and their connections and settings vary with the LCD driver type (STN/TFT) and specifications to be used.



3.19.3.4 Reference Clock LCP0

LCP0 is used as the reference clock for all the signals in the LCDC.

This section explains how to set the frequency (period) of the LCP0 signal.

The LCP0 clock speed (LD bus transfer speed) is determined by selecting TFT or STN and setting LCDMODE0<SCPW1:0> and LCDMODE1<SWPW2>. The clock speed should be selected to meet the characteristics of the LCD driver to be used.

The LCP0 period can be selected from four types: $f_{SYS}/2$, $f_{SYS}/4$, $f_{SYS}/8$, $f_{SYS}/16$, $f_{SYS}/24$ and $f_{SYS}/48$.

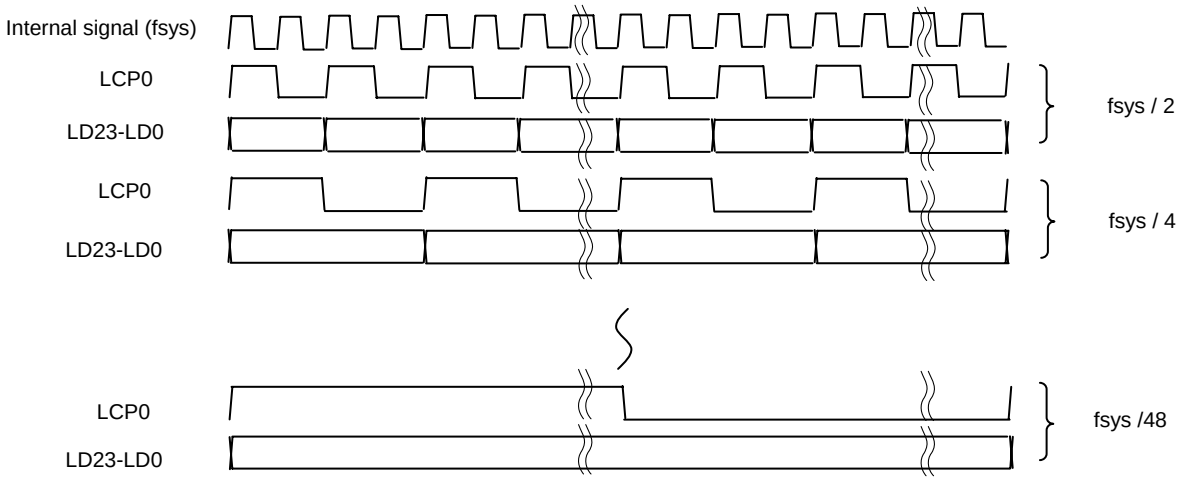


Figure 3.19.1 LCP Frequency Selection

Minimum speed

The LCP0 period needs to be short enough to prevent the next line signal from overlapping the current line signal.

The transfer speed of display data must be set to suit the refresh rate; otherwise data cannot be transferred properly. Set the data transfer speed so that each transfer completes within the LHSYNC period.

STN monochrome/grayscale	:	Segment size / 8 × LCP0 [s: period] < LHSYNC [s: period] STN color
STN color	:	Segment size × 3 / 8 LCP0 [s: period] < LHSYNC [s: period]
TFT	:	Segment size × LCP0 [s: period] < LHSYNC [s: period]

Maximum speed

If the LCP0 period is too short, the data to be transferred to the LCD driver cannot be prepared in time, causing wrong data to be transferred. The maximum transfer speed is limited by the operation mode and display RAM type (bus width, wait condition, and so on). If the data rotation function is used, the transfer speed must be slower.

LCP0 Setting Range Table

Conditions

f_{SYS} : 60 MHz
Display size (color) : up to 320 × 320
Display size (monochrome/grayscale) : up to 640 × 480

Note: This table shows the range of LCP0 settings that can be made under the conditions shown above. If the CPU clock speed, display size, or refresh rate is changed, the LCP0 range also changes.

Display RAM / Display Mode	Internal RAM	SDRAM	External SRAM (0 waits)	External SRAM (N waits)
STN monochrome Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to$f_{SYS}/16$ (up to 2 waits) $f_{SYS}/8$ to $f_{SYS}/16$ (up to 6 waits) $f_{SYS}/16$ (up to 14 waits)
STN 4-grayscale Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/8$ (up to 2 waits) $f_{SYS}/8$ (up to 6 waits)
STN 16-grayscale Refresh cycle = 140 Hz	$f_{SYS}/2$ to $f_{SYS}/8$	$f_{SYS}/2$ to $f_{SYS}/8$	$f_{SYS}/4$ to $f_{SYS}/8$	$f_{SYS}/8$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/16$ (up to 6 waits)
STN 64-grayscale Refresh cycle = 200 Hz	$f_{SYS}/4$	$f_{SYS}/4$	$f_{SYS}/4$	$f_{SYS}/4$ (up to 1 wait)
STN 256-color Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$	$f_{SYS}/8$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/16$ (up to 6 waits)
STN 4K-color Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/8$ to $f_{SYS}/16$ (up to 6 waits) $f_{SYS}/16$ (up to 14 waits)
TFT 4K-color Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ To $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/8$ to $f_{SYS}/16$ (up to 6 waits) $f_{SYS}/16$ (up to 14 waits)
TFT 64K-color Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/8$ to $f_{SYS}/16$ (up to 6 waits) $f_{SYS}/16$ (up to 14 waits)
TFT 64K-color + rotation operation	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/8$ to $f_{SYS}/16$ (up to 6 waits) $f_{SYS}/16$ (up to 14 waits)
TFT 256K-color Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$	$f_{SYS}/8$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/16$ (up to 2 waits)
TFT 16M-color Refresh cycle = 70 Hz	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/2$ to $f_{SYS}/16$	$f_{SYS}/4$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/8$ to $f_{SYS}/16$ (up to 2 waits) $f_{SYS}/16$ (up to 2 waits)

Example 1: When f_{SYS} = 10 MHz, STN mode, LCDMODE0<SCPW1:0> = 01

Internal reference clock LCP0 = f_{SYS} / 8 = 10 MHz / 8 = 1.25 [MHz]
LCP0 period = 1 / 1.25 [MHz] = 0.8 [μS]

Example 2: when f_{SYS} = 60 MHz, TFT mode, LCDMODE0<SCPW1:0> = 11

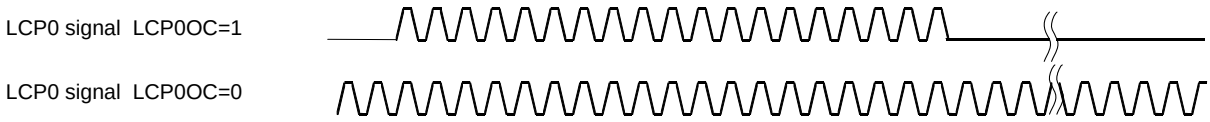
Internal reference clock LCP0 = f_{SYS} / 16 = 60 MHz / 16 = 3.75 [MHz]
LCP0 period = 1 / 3.75 [MHz] = 266 [nS]

LCDMODE0 Register

LCDMODE0 (0280H)

	7	6	5	4	3	2	1	0
bit Symbol	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	1	1	0	0	0	0
Function	Display RAM 00: Internal RAM (32-bit) 01: External SRAM 10: SDRAM 11: Reserved		LD bus transfer speed SCPW2= 0 00: 2-clk 01: 4-clk 10: 8-clk 11: 16-clk SCPW2= 1 00: 6-clk 01: 12-clk 10: 24-clk 11: 48-clk		Mode selection 0000: Reserved 1000: Reserved 0001: SR (mono) 1001: Reserved 0010: SR (4-gray) 1010: TFT (256-color) 0011: Reserved 1011: TFT (4096-color) 0100: SR (16-gray) 1100: TFT (64K-color) 0101: SR (64-gray) 1101: TFT(256K-,16M-color) 0110: STN (256-color) 1110: Reserved 0111: STN (4096-color) 1111: Reserved		

LCDCTL0 <LCP0OC> is used to control the output timing of the LCP0 signal. When <LCP0OC>=0, the LCP0 signal is always output. When <LCP0OC>=1, the LCP0 signal is output only when valid data is output.



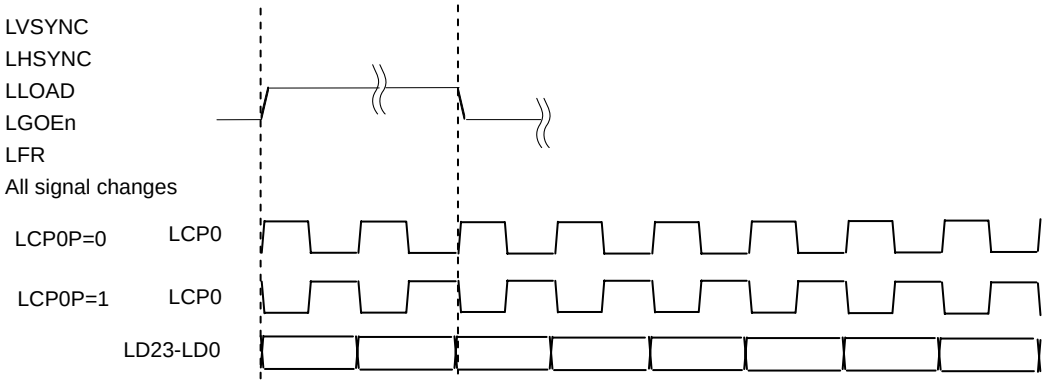
LCD Control 0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
Read/Write	R/W	R/W	R/W	R/W		R/W	R/W	R/W
After reset	0	0	0	0		0	0	0
Function	PIP function 0: Disable 1: Enable	Segment data 0: Normal 1: Always output “0”	Frame divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0: Line 1: LCP0	LCP0 (Note) 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

The phase of the LCP0 signal can be inverted by the setting of LCDCTL1<LCP0P>.



LCD Control 1 Register

LCDCTL1 (0286H)

	7	6	5	4	3	2	1	0
bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
Read/Write	R/W	R/W	R/W	R/W			R/W	R/W
After reset	1	0	1	0			0	0
Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00 : 1 clock of LHSYNC 01 : 2 clocks of LHSYNC 10 : 3 clocks of LHSYNC 11 : Reserved	

3.19.3.5 Refresh Rate

The period of the horizontal synchronization signal LHSYNC is defined as the product of the value set in LCDHSP<LH15:0> and the LCP0 clock period.
The value to be set in LCDHSP<LH15:0> is obtained as follows:

TFT
Segment size + number of dummy clocks (*)

STN
Monochrome/grayscale : (Segment size / 8) + number of dummy clocks (*)
Color : (Segment size × 3 / 8) + number of dummy clocks (*)

LHSYNC [s: period] = LCP0 [s: period] × (<LH15:0> + 1)

LCD LHSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDHSP (028AH)	bit Symbol	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 7–0)							
		7	6	5	4	3	2	1	0
(028BH)	bit Symbol	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 15-8)							

The period of the vertical synchronization signal LVSYNC is defined as the product of the value set in LCDVSP<LV9:0> and the LHSYNC period.
The value to be set in LCDVSP<LV9:0> is obtained as follows:

TFT
Common size + number of dummy clocks (*)

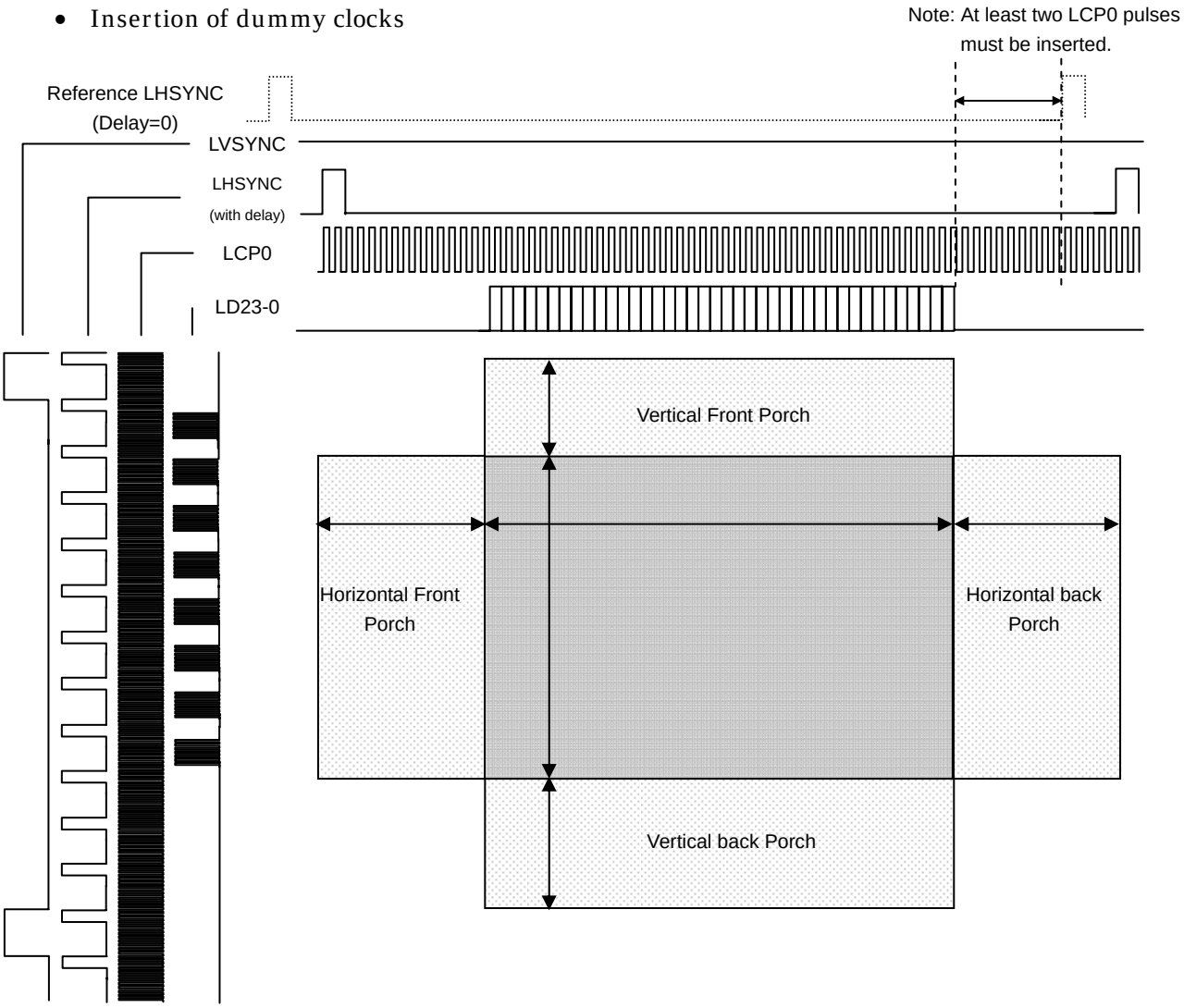
STN
Common size + number of dummy clocks (*)
(A minimum of one dummy clock must be inserted in the back porch.)

LVSYNC [s: period] = LHSYNC [s: period] × (<LV9:0> + 1)
= LCP0 [s: period] × (<LH15:0> + 1) × (<LV9:0> + 1)

LCD V SYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDVSP (028CH)	bit Symbol	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LVSYNC period (bits 7-0)							
		7	6	5	4	3	2	1	0
(028DH)	bit Symbol							LVP9	LVP8
	Read/Write							W	
	After reset							0	0
	Function	LVSYNC period (bits 9-8)							

- Insertion of dummy clocks



The above is a conceptual diagram showing the data (LD23-0), shift clock (LCP0), horizontal synchronization signal (LHSYNC), and vertical synchronization signal (LVSYNC) on the LCD panel.

The front porch and back porch as shown above should be taken into consideration in setting LCDHSP<LH15:0> and LCDVSP<LV9:0> explained earlier.

Note 1: The horizontal back porch must be set so that “data transfer” plus “LCP0 × 2 clocks” are completed within one period of the reference clock LHSYNC (with 0 delay), as defined by the following equation:

Delay time (LLOAD) + number of data transfer times + 2 < LHSYNC (LCP0 pulse count)

Note 2: The vertical back porch must have a minimum of one dummy clock.

(*) TFT driver

The recommended number of dummy clocks is specified by each TFT driver (or LCD module). Refer to the specifications of the TFT driver (LCD module) to be used.

(*) STN driver

For an STN driver, the refresh rate can be set accurately by adjusting the value of the horizontal back porch. If the desired refresh rate cannot be obtained by the horizontal back porch, it can be further adjusted by the vertical back porch. For details, refer to the setting example to be described later in this section.

- Setting method

The front dummy LHSYNC (vertical front porch) not accompanied by valid data in the total of LHSYNC period in the LVSYNC period is defined by the value set in LCDPRVSP<PLV6:0>.

Front dummy LHSYNC (vertical front porch) = <PLV6:0>

The back dummy LHSYNC (vertical back porch) is defined as follows:

(<LVP9:0> + 1) – (valid LHSYNC: common size) – (front dummy LHSYNC: <PLV6:0>)

The vertical back porch must have a minimum of one dummy clock.

The front dummy LCP0 (horizontal front porch) not accompanied by valid data in the total number of LCP0 clocks in the LHSYNC period is defined by the value set in LCDLDDLY<LDD6:0>.

Front dummy LCP0 (horizontal front porch) = <LDD6:0>

The back dummy LCP0 (horizontal back porch) is defined as follows:

(<LH15:0> + 1) – (valid LCP0: segment size) – (front dummy LCP0: <LDD6:0>)

Note 1: The back dummy LCP0 (horizontal back porch) must have a minimum of two LCP0 clocks.

Note 2: The delay time that is set in LCDLDDLY<LDD6:0> is counted based on LHSYNC (with 0 delay).

LCDLDDLY (0290H)

	7	6	5	4	3	2	1	0
bit Symbol	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
Read/Write	R/W	W						
After reset	0	0	0	0	0	0	0	0
Function	Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

Example 1) Setting the refresh rate to 200 Hz under the following conditions:

f_{SYS} = 30 MHz, STN mode, 320-segment × 240-common, 4096-color display,
LCDMODE0<SCPW1:0> = 00

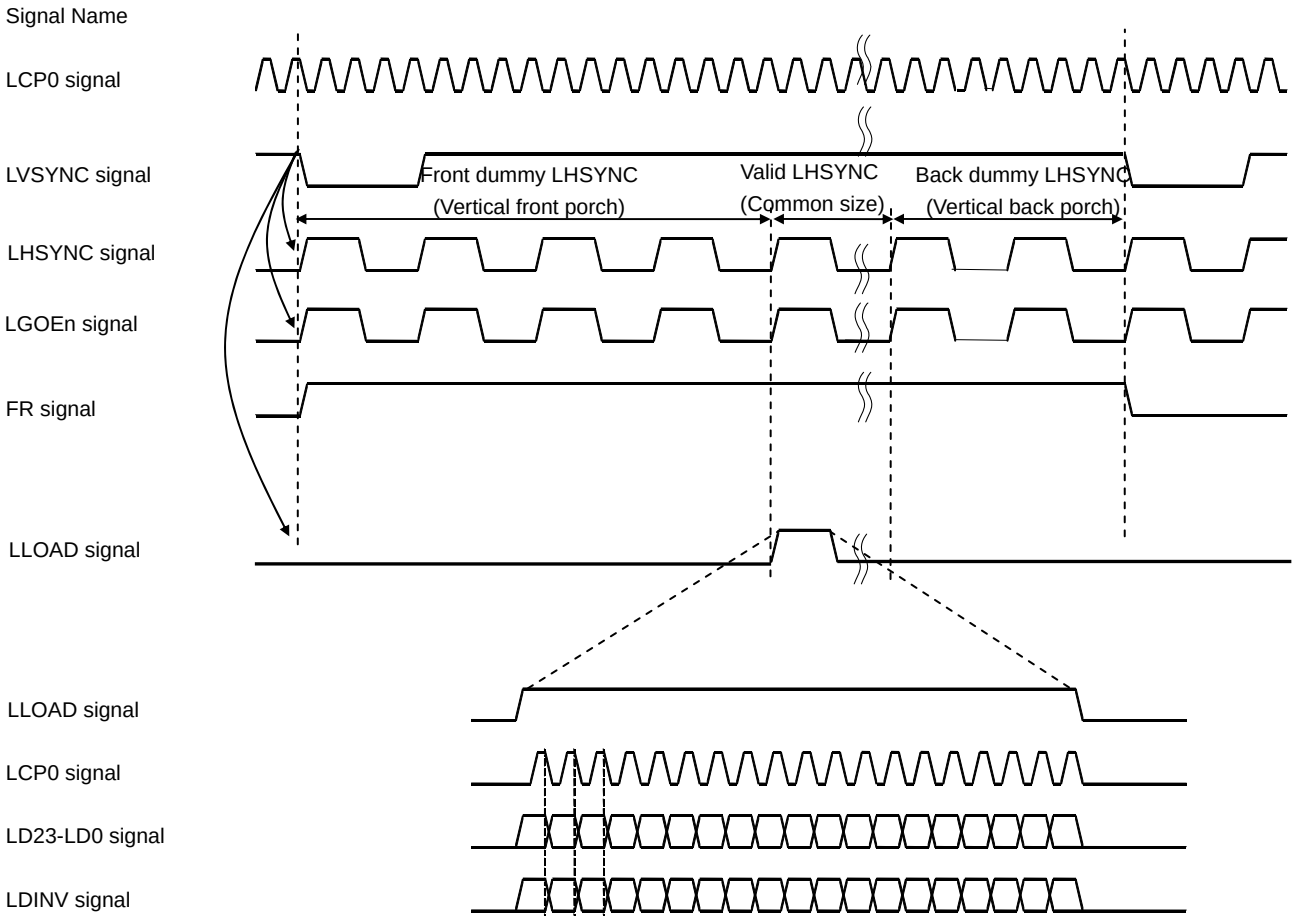
Internal reference clock LCP0 = f_{SYS} / 4 = 30 [MHz] / 4 = 7.5 [MHz]
Therefore, LCP0 period = 1 / 7.5 [MHz] = 0.133 [μS]
Condition 1: Refresh rate = 200 Hz, Refresh cycle = 5 [ms
Condition 2: LH = <LH15:0> ≥ (320×3/8) – 1 = 119
Condition 3: LV = <LVP9:0> ≥ 240 – 1

When <LVP9:0> = 239 (minimum value):

LVSYNC [s: period] = LHSYNC [s: period] × (<LVP9:0> + 1)
= LCP0 [s: period] × (<LH15:0> + 1) × (<LVP9:0> + 1)

5 [ms] = (1 / 7.5 [MHz]) × (LH + 1) ×240
LH + 1 = $(5 \times 10^{-3}) \times (7.5 \times 10^{6}) / 240$
= 156.25

3.19.3.6 Signal Settings



The above diagram shows the typical timings of the signals controlled by the LCDC. This section explains how to control each of these signals.

1. LVSYNC Signal

The period of the vertical synchronization signal LVSYNC indicates the time for each screen update (refresh rate). The LVSYNC period is defined as an integral multiple of the period of the horizontal synchronization signal LHSYNC.

The LVSYNC period is calculated as the product of the value set in LCDVSP<LV 9:0> and the LHSYNC period. The value to be set in LCDVSP<LV9:0> should be "common size + number of dummy clocks" or larger for TFT and STN.

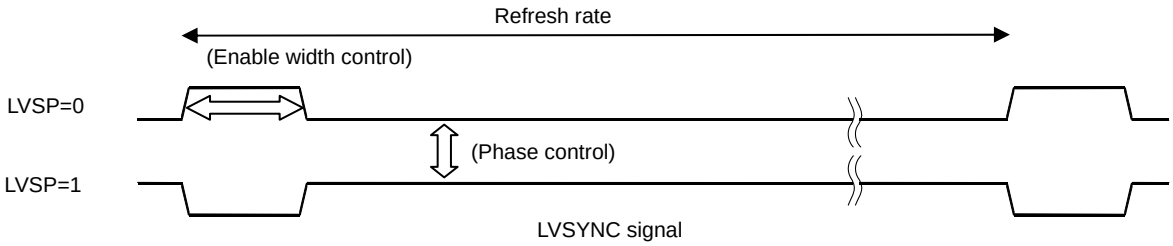
LVSYNC [s: period] = LHSYNC [s: period] × (<LVP9:0> + 1)
= LCP0 [s: period] × (<LH15:0> + 1) × (<LVP9:0> + 1)

LCD V SYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDVSP (028CH)	bit Symbol	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LVSYNC period (bits 7-0)							
		7	6	5	4	3	2	1	0
(028DH)	bit Symbol							LVP9	LVP8
	Read/Write							W	
	After reset							0	0
	Function							LVSYNC period (bits 9-8)	

The enable width of the LVSYNC signal can be specified as 1 clock, 2 clocks, or 3 clocks of LHSYNC in LCDCTL1<LVSW1:0>.

The phase of the LVSYNC signal can be inverted by the setting of LCDCTL1 <LVSP>.



LCD Control 1 Register

		7	6	5	4	3	2	1	0
LCDCTL1 (0286H)	bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
	Read/Write	R/W	R/W	R/W	R/W			R/W	R/W
	After reset	1	0	1	0			0	0
	Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00 : 1 clock of LHSYNC 01 : 2 clocks of LHSYNC 10 : 3 clocks of LHSYNC 11 : Reserved	

2. LHSYNC Signal

The period of the horizontal synchronization signal LHSYNC corresponds to one line of display. The LHSYNC period is defined as an integral multiple of the reference clock signal LCP0.

The LHSYNC period is defined as the product of the value set in LCDHSP<LH15:0 > and the LCP0 clock period. The value to be set in LCDHSP<LH15:0 > should be "segment size + number of dummy clocks" or larger for TFT. In the case of STN, the minimum value of LCDHSP<LH15:0 > is:

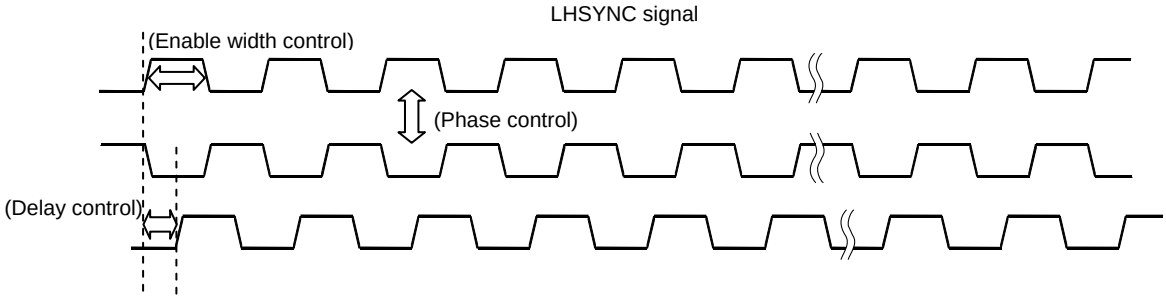
Monochrome/grayscale : (Segment size / 8) + number of dummy clocks
Color : (Segment size × 3 / 8) + number of dummy clocks

LHSYNC [s: period] = LCP0 [s: period] × (<LH15:0> + 1)

LCD LHSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDHSP (028AH)	bit Symbol	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 7-0)							
		7	6	5	4	3	2	1	0
(028BH)	bit Symbol	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 15-8)							

The enable width of the LHSYNC signal can be specified by LCDHSW<HSW9:0>. It is also possible to set the delay time for the LVSYNC signal in units of LCP0 pulses.

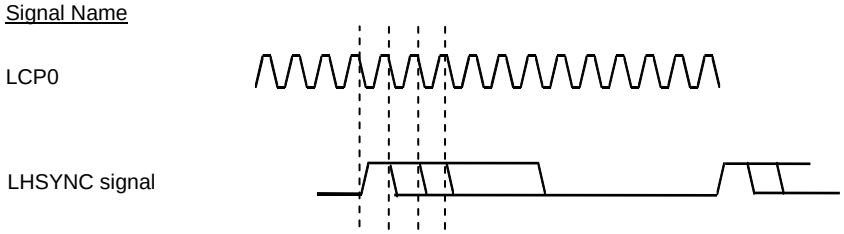


The enable width of the LHSYNC signal is set using LCDHSW<HSW8:0>. It can be specified in a range of 1 to 512 pulses of the LCP0 clock.

The enable width is represented by the following equation:

Enable width = <HSW8:0> + 1

Thus, when LCDHSW<HSW8:0> is set to "0", the enable width is set as one pulse of the LCP0 clock.



High width setting
LCP0 clock = 1, 2, 3 … 512 pulses

LCDHSW Register

LCDHSW (0294H)

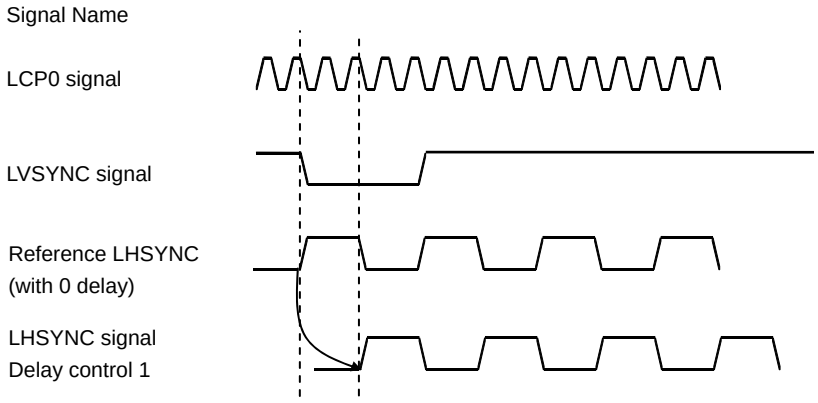
	7	6	5	4	3	2	1	0
bit Symbol	HSW7	HSW6	HSW5	HSW4	HSW3	HSW2	HSW1	HSW0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LHSYNC width (bits 7-0)							

LCDHWB8 (0299H)

	7	6	5	4	3	2	1	0
bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

As shown in the diagram below, delay time of 0 to 127 pulses of the LCP0 clock can be inserted in the LHSYNC signal.

Delay time = <HSD6:0>

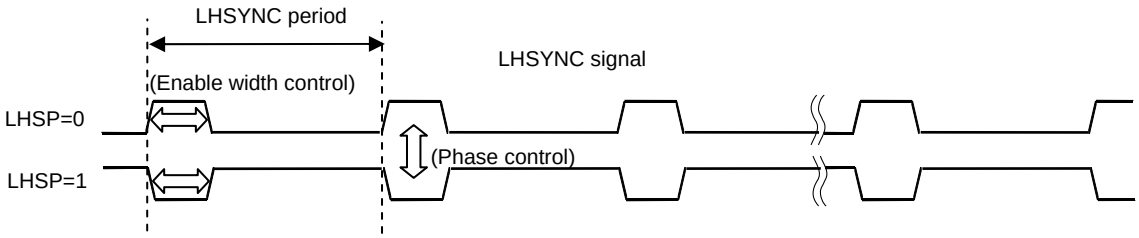


LCDHSDLY Register

LCDHSDLY (028FH)

	7	6	5	4	3	2	1	0
bit Symbol		HSD6	HSD5	HSD4	HSD3	HSD2	HSD1	HSD0
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		LHSYNC delay (bits 6-0)						

The phase of the LHSYNC signal can be inverted by the setting of LCDCTL1 <LVSP>.



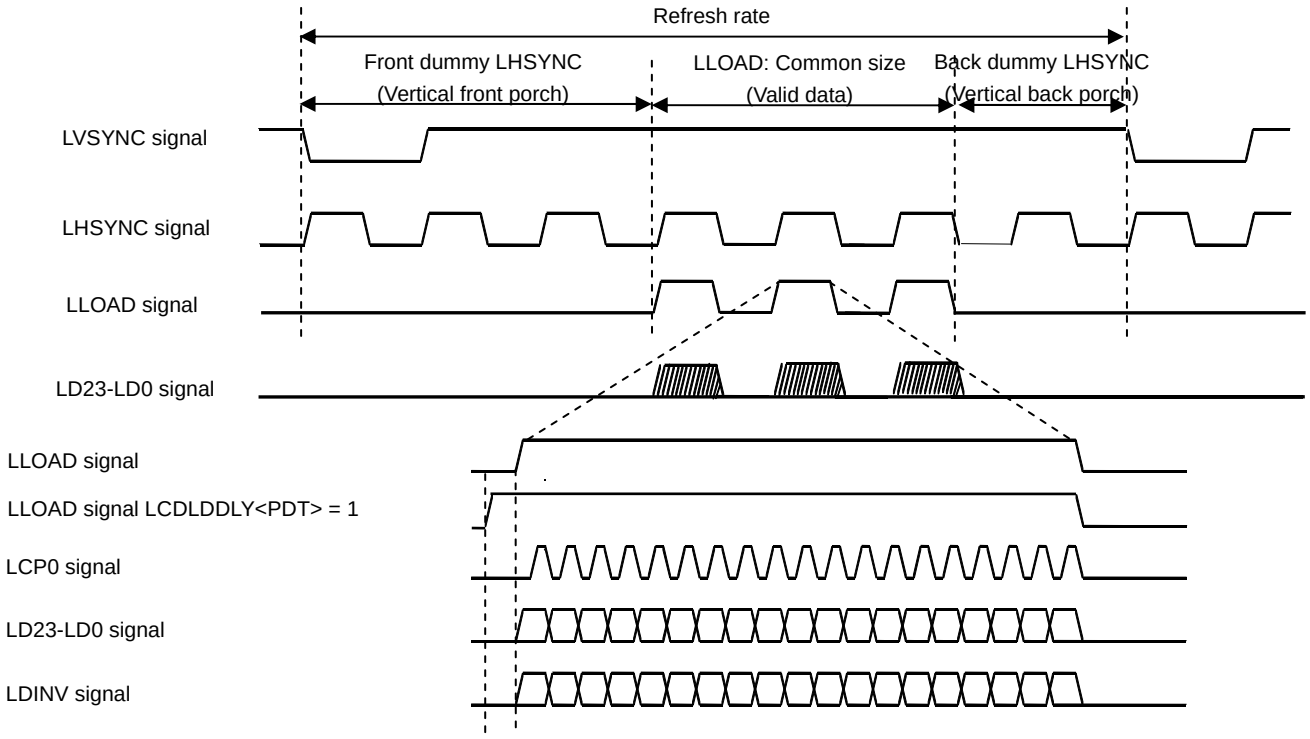
LCD Control 1 Register

LCDCTL1 (0286H)

	7	6	5	4	3	2	1	0
bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
Read/Write	R/W	R/W	R/W	R/W			R/W	R/W
After reset	1	0	1	0			0	0
Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00 : 1 clock of LHSYNC 01 : 2 clocks of LHSYNC 10 : 3 clocks of LHSYNC 11 : Reserved	

3. LLOAD Signal

The LLOAD signal is used to control the timing for the LCD driver to receive display data. The period of the LLOAD signal synchronizes to one line of display. It is defined as an integral multiple of the reference clock LCP0.



The LHSYNC signal and LLOAD signal differs in that the LHSYNC signal is output all the time whereas the LLOAD signal is output only at valid data lines (commons).
Display data is output in synchronization with the LLOAD signal. Therefore, if a delay is inserted in the LLOAD signal through the LCDLDDLY register, data output is also delayed.
Also note that when LCDLDDLY<PDT>=1 , data is output one LCP0 clock later than the LLOAD signal.

LCDLDDLY<PDT>=0: Data is output in synchronization with the LLOAD signal.
LCDLDDLY<PDT>=1: Data is output one LCP0 clock later than the LLOAD signal.

The delay time for the LLOAD signal is controlled based on LCDLDDLY<PDT>=1. Therefore, even if the delay time is set to “0” with LCDLDDLY<PDT>=0, the LLOAD signal is output with a delay of one LCP0 clock. Be careful about this point.

The number of pulses in the front dummy LHSYNC (vertical front porch) is specified by LCDPRVSP<PLV6:0>. This delay time can be set in a range of 0 to 127 pulses of the LCP0 clock.

Front dummy LHSYNC = <PLV6:0>

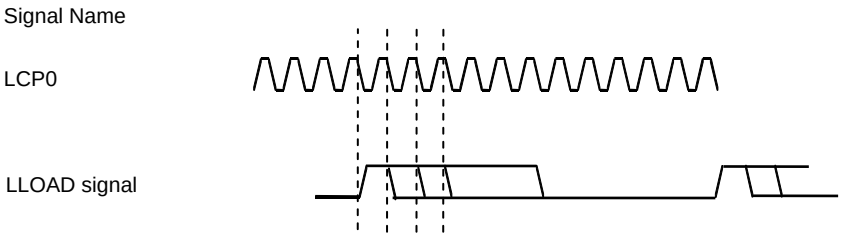
LCD LVSYNC Pre Pulse Register

LCDPRVSP (028EH)

	7	6	5	4	3	2	1	0
bit Symbol		PLV6	PLV5	PLV4	PLV3	PLV2	PLV1	PLV0
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		Front dummy LVSYNC (bits 6-0)						

The back dummy LHSYNC (vertical back porch) is defined as follows:

(<LVP9:0> + 1) – (valid LHSYNC: common size) – (front dummy LHSYNC: <PLV6:0>)



High width setting

LCP0 clock = 1, 2, 3 … 1023 pulses (<PDT>=0) / 1024 pulses (<PDT>=1)

Note: The vertical back porch must be set to “1” or longer in all the cases (STN/TFT).

The enable width of the LLOAD signal is determined depending on the LCDCTL0<LCP0OC> setting, as shown below.

LCDCTL0<LCP0OC> = 0 : Output at setting value in (LCDDLW) <LDW9:0>

LCDCTL0<LCP0OC> = 1 : Output at valid data

LCD Control 0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
Read/Write	R/W	R/W	R/W	R/W		R/W	R/W	R/W
After reset	0	0	0	0		0	0	0
Function	PIP function 0: Disable 1: Enable	Segment data 0: Normal 1: Always output “0”	Frame divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0: Line 1: LCP0	LCP0 (Note) 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

The enable width of the LLOAD signal is specified using LCDLDW<LDW9:0>. It can be set in a range of 0 to 1024 pulses of the LCP0 clock.

The actual enable width is determined depending on the LCDLDDLY<PDT> setting, as shown below.

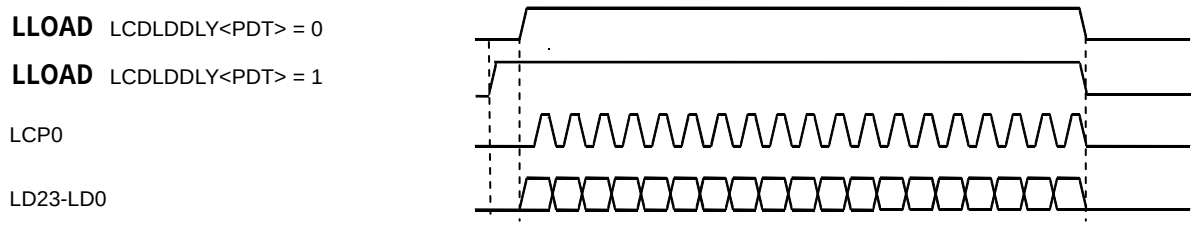
Enable width = <LDW9:0> + 1 (when <PDT> = 1, <LDW9:0>=0 is prohibited)

Enable width = <LDW9:0> (when <PDT> = 0)

LCDLDW Register

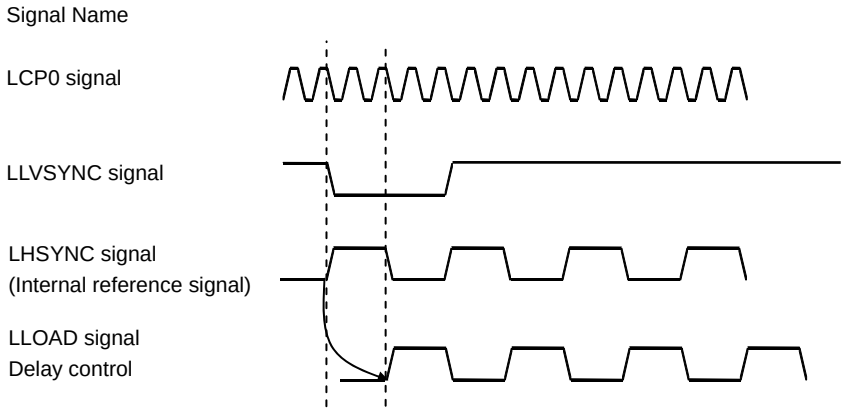
		7	6	5	4	3	2	1	0
LCDLDW (0295H)	bit Symbol	LDW7	LDW6	LDW5	LDW4	LDW3	LDW2	LDW1	LDW0
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LLOAD width (bits 7-0)							
		7	6	5	4	3	2	1	0
LCDHWB8 (0299H)	bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

When LCDCTL0<LCP0OC>=1, the enable width of the LLOAD signal is shown below.



As shown in the diagram below, delay time of 0 to 127 pulses of the LCP0 clock can be inserted in the LLOAD signal.

Delay time = <LDD6:0>

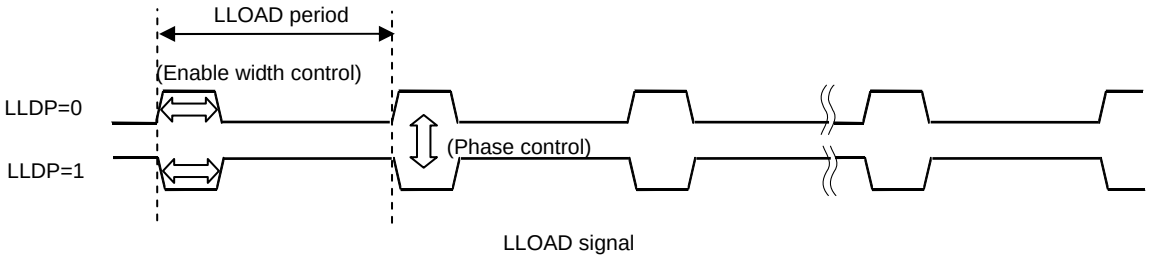


Note: The delay time for the LLOAD signal is controlled based on LCDLDDLY<PDT>=1. Therefore, even if the delay time is set to"0" with LCDLDDLY<PDT>=0, the LLOAD signal is output with a delay of one LCP0 clock. Be careful about this point.

LCDLDDLY Register

		7	6	5	4	3	2	1	0
LCDLDDLY (0290H)	bit Symbol	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
	Read/Write	R/W	W						
	After reset	0	0	0	0	0	0	0	0
	Function	Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

The phase of the LLOAD signal can be inverted by the setting of LCDCTL1 <LLDP>.

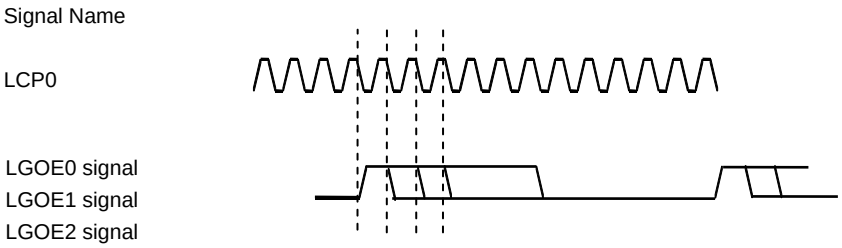


LCD Control 1 Register

		7	6	5	4	3	2	1	0
LCDCTL1 (0286H)	bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
	Read/Write	R/W	R/W	R/W	R/W			R/W	R/W
	After reset	1	0	1	0			0	0
	Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00 : 1 clock of LHSYNC 01 : 2 clocks of LHSYNC 10 : 3 clocks of LHSYNC 11 : Reserved	

4. LGOE0 to LGOE2 Signals

The LCDC has three signals (LGOE0 to LGOE2) that can be controlled like the LHSYNC signal. For these signals, the enable width, delay time, and phase timing can be adjusted as shown below.



High width setting
LGOE0: LCP0 clock = 1, 2, 3 … 512 pulses
LGOE1: LCP0 clock = 1, 2, 3 … 1024 pulses
LGOE2: LCP0 clock = 1, 2, 3 … 1024 pulses

LCDHO0W (0296H)

	7	6	5	4	3	2	1	0
bit Symbol	O0W7	O0W6	O0W5	O0W4	O0W3	O0W2	O0W1	O0W0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE0 width (bits 7-0)							

LCDHO1W (0297H)

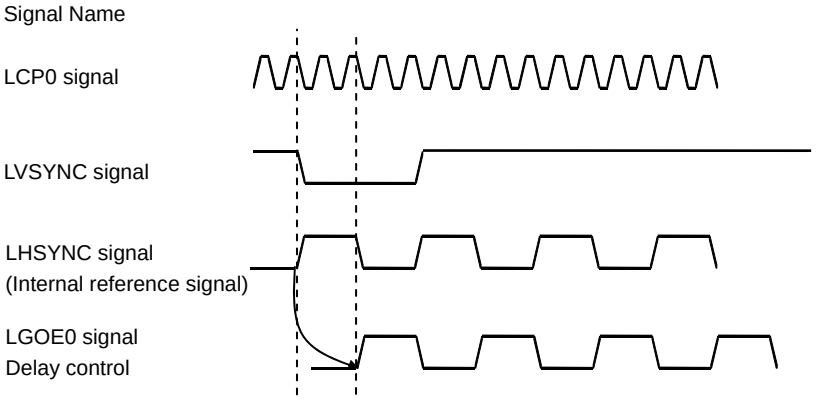
	7	6	5	4	3	2	1	0
bit Symbol	O1W7	O1W6	O1W5	O1W4	O1W3	O1W2	O1W1	O1W0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE1 width (bits 7-0)							

LCDHO2W (0298H)

	7	6	5	4	3	2	1	0
bit Symbol	O2W7	O2W6	O2W5	O2W4	O2W3	O2W2	O2W1	O2W0
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE2 width (bits 7-0)							

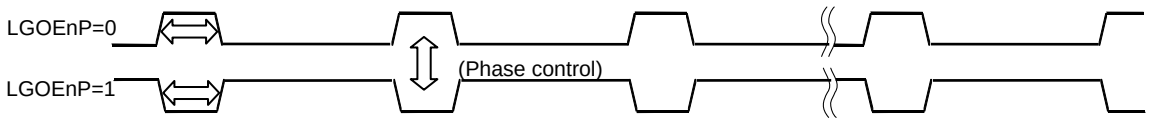
LCDHWB8 (0299H)

	7	6	5	4	3	2	1	0
bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)



		7	6	5	4	3	2	1	0
LCDO0DLY (0291H)	bit Symbol		OE0D6	OE0D5	OE0D4	OE0D3	OE0D2	OE0D1	OE0D0
	Read/Write		W						
	After reset		0	0	0	0	0	0	0
	Function		OE0 delay (bits 6-0)						
		7	6	5	4	3	2	1	0
LCDO1DLY (0292H)	bit Symbol		OE1D6	OE1D5	OE1D4	OE1D3	OE1D2	OE1D1	OE1D0
	Read/Write		W						
	After reset		0	0	0	0	0	0	0
	Function		OE1 delay (bits 6-0)						
		7	6	5	4	3	2	1	0
LCDO2DLY (0293H)	bit Symbol		OE2D6	OE2D5	OE2D4	OE2D3	OE2D2	OE2D1	OE2D0
	Read/Write		W						
	After reset		0	0	0	0	0	0	0
	Function		OE2 delay (bits 6-0)						

LGOEn signal



LCD Control 2 Register

LCDCTL2 (0287H)

	7	6	5	4	3	2	1	0
bit Symbol	LGOE2P	LGOE1P	LGOE0P					
Read/Write	R/W	R/W	R/W					
After reset	0	0	0					
Function	LGOE2 phase 0: Rising 1: Falling	LGOE1 phase 0: Rising 1: Falling	LGOE0 phase 0: Rising 1: Falling					

5. LFR Signal

The LFR (frame) signal is used to control the direction of bias the LCD driver applies on liquid crystal cells. With small screens in monochrome mode, the polarity of the LFR signal is normally inverted in synchronization with each screen display. With large screens or when grayscale or color mode is used, the polarity is inverted at shorter intervals to adjust the display quality.

When LCDCTL0<FRMON>="1" and LCDCTL0<DLS> = "0", the LFR signal is inverted at intervals of "LHSYNC x N" (LHSYNC: internal reference signal with 0 delays). The "N" value is specified in LCDDVM0<FML3:0> and LCDDVM1<FML7:4>.

When <DLS>="0" and <FREDGE>=0, LFR signal synchronous with front edge of LHSYNC signal, and when <DLS>="0" and <FREDGE>=1, LFR signal synchronous with rear edge of LHSYNC signal.

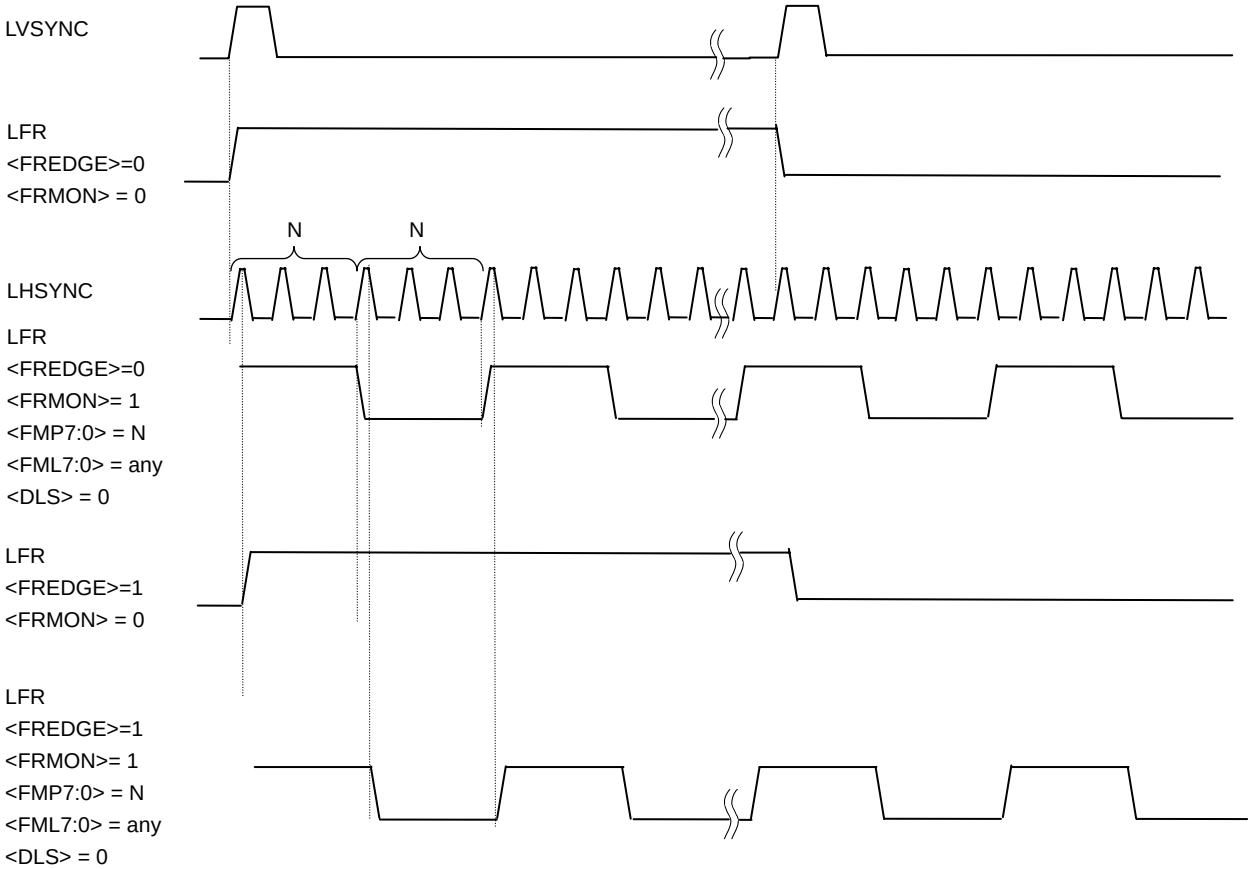
When LCDCTL0<FRMON> is set to "0" to disable the frame divide function, the LFR signal is inverted in synchronization with the LVSYNC period.

Enabling this function does not affect the waveform and timing of the LVSYNC signal. (The refresh rate is not changed.)

Note1: The effect of this function varies with the characteristics of the LCD driver and LCD panel to be used.

Note2: LFR signal delaies synchronous with LHSYNC signal.

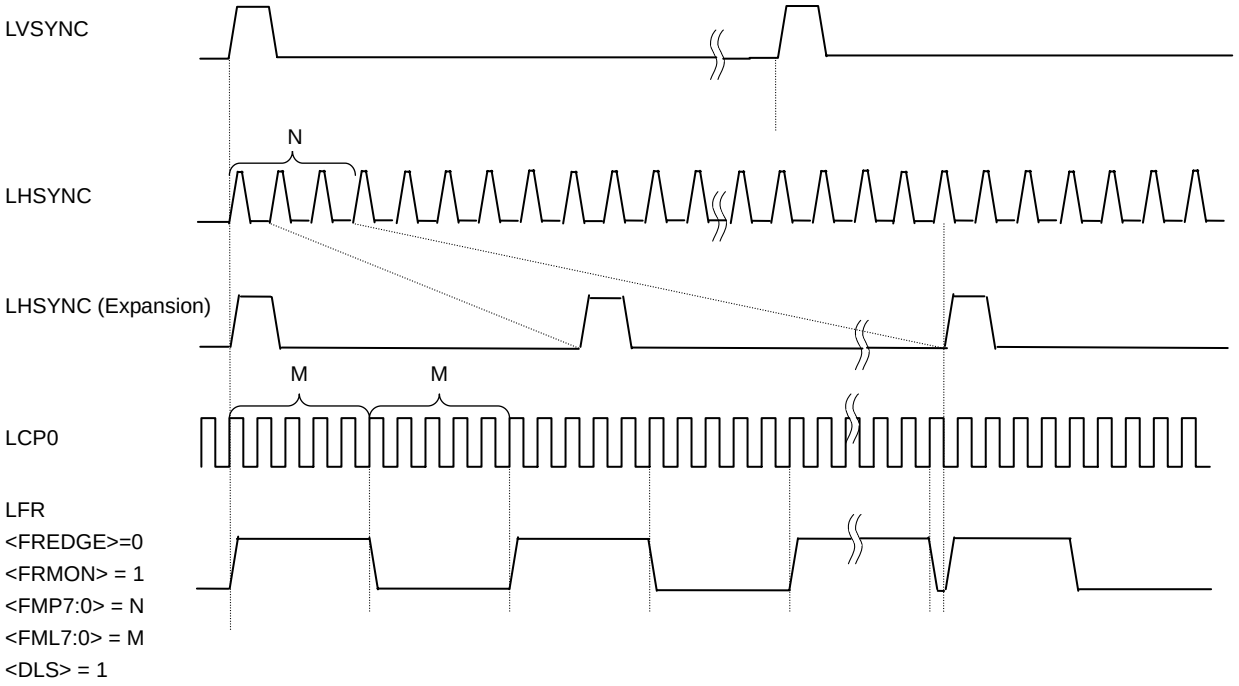
Generally, setting a prime number (3, 5, 7, 11, 13 and so on) as the "N" value produces better results.



When LCDCTL0<FRMON>=1 and LCDCTL0<DLS>=1, frame output is inverted at intervals set in LCDDVM0<FML3:0> and the LFR signal is inverted at intervals of "LCP0 × M". The "M" value is specified in LCDDVM0<FMP7:4>.

When <DLS>="1" LFR signal synchronous with front edge of LHSYNC signal.

So, prohibit to set <FREDGE>=1, always need to set <FREDGE>=0.



Note prohibit to set <FREDGE>=1, always need to set <FREDGE>=0.

LCD Control 0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
Read/Write	R/W	R/W	R/W	R/W		R/W	R/W	R/W
After reset	0	0	0	0		0	0	0
Function	PIP function 0:Disable 1:Enable	Segment data setting 0: Normal 1: Always output “0”	Frame divide setting 0: Disable 1: Enable	Always write “0”		LFR signal LCP0/line selection 0:Line 1:LCP0	LCP0 (Note) 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

Divide FRM 0 Register

LCDDVM0 (0283H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP3	FMP2	FMP1	FMP0	FML3	FML2	FML1	FML0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 3-0) (M)				LHSYNC DVM (bits 3-0) (N)			

LCDDVM1 (0284H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP7	FMP6	FMP5	FMP4	FML7	FML6	FML5	FML4
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits7-4) (M)				LHSYNC DVM (bits 7-4) (N)			

6. LD Bus

The data to be transferred to the LCD driver is output via a dedicated bus (LD23 to LD0). The output format can be selected according to the input method of the LCD driver to be used.

The LCDC reads data of the size corresponding to the specified LCD size from the display RAM and transfers it to the external LCD driver via the data bus pin dedicated to the LCD. Thus, the LCDC automatically issues a bus request to the CPU (to stop CPU operation) when it needs to read data from the display RAM. The bus occupancy rate of the LCDC varies depending on the display mode and the speed at which data is read from the display RAM.

Display RAM	Bus Width	Valid Data Read Time (f_{SYS} clocks/bytes)	Valid Data Read Time t_{LRD}(ns/bytes) at f_{SYS} = 60 MHz
External SRAM	16-bit	(2 + number of waits) / 2	16.6
Internal RAM	32-bit	**1/4	**4.16
External SDRAM	16-bit	*1/2	*8.33

Note: When SDRAM is used, additional 9 clocks are needed as overhead time for reading each common (line) data. When internal RAM is used, additional 1 clock is needed as overhead time for reading each common (line) data. Additional 1 clock of overhead time is also needed when a change of blocks occur in the internal RAM even if the common (line) remains the same.

The time the CPU stops operating while data for one common (line) is being transferred is defined as t_{STOP}, which is represented by the following equation:

$$t_{STOP} = (SegNum \times K / 8) \times t_{LRD}$$

SegNum: Number of display segments
K : Number of bits needed for displaying one pixel

Monochrome display	K=1
4-grayscale display	K=2
16-grayscale display	K=4
256-color display	K=8
4096-color display	K=12
65536-color display	K=16
262144-/16777216-color display	K=24

Note: When SDRAM is used, overhead time is added as follows:

$$t_{STOP}\ [S] = (SegNum \times K / 8) \times t_{LRD} + ((1 / f_{SYS}) \times 8)$$

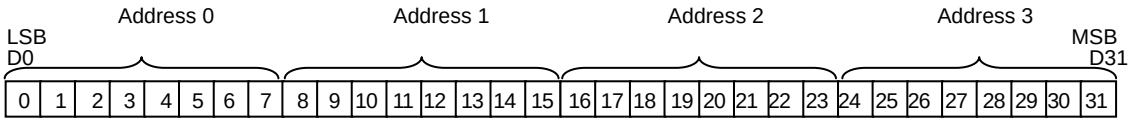
The bus occupancy rate indicates the proportion of the one common (line) update time t_{LP} occupied by t_{STOP} and is calculated by the following equation:

CPU bus occupancy rate = t_{STOP} [s] / LHSYNC [s: period]

- Memory Map Image and Data Output in Each Display Mode

STN monochrome (1-pixel display data = 1-bit memory data)

Display Memory



LD Bus Output

8-bit type

LD0 0 → 8 ···

LD1 1 → 9 ···

LD2 2 → 10 ···

LD3 3 → 11 ···

LD4 4 → 12 ···

LD5 5 → 13 ···

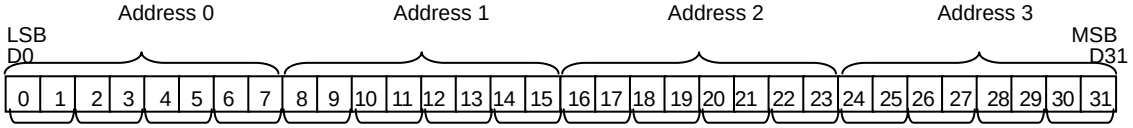
LD6 6 → 14 ···

LD7 7 → 15 ···

Note: When setting 240 segment, 256 segment size of data is required.

●STN 4-grayscale (1-pixel display data = 2-bit memory data)

Display Memory



LD Bus Output

8-bit type

LD0 1 - 0 → 17-16 ···

LD1 3 - 2 → 19-18 ···

LD2 5 - 4 → 21-20 ···

LD3 7- 6 → 23-22 ···

LD4 9- 8 → 25-24 ···

LD5 11-10 → 27-26 ···

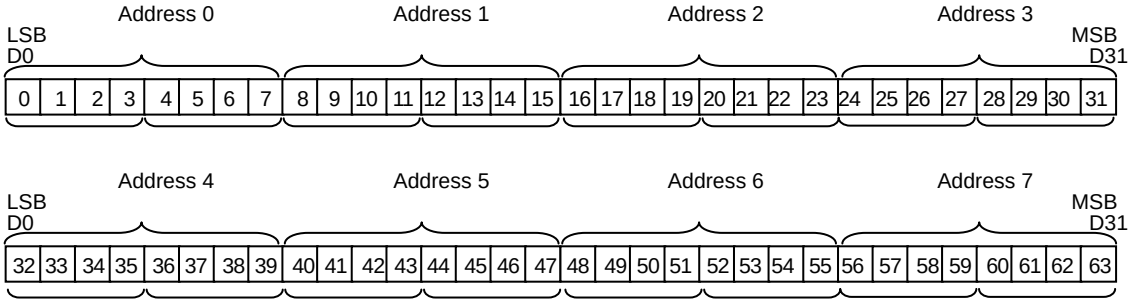
LD6 13-12 → 29-28 ···

LD7 15-14 → 31-30 ···

Figure 3.19.2 Memory Map Image and Data Output in STN Monochrome/4-Grayscale Mode

STN 16-grayscale (1-pixel display data = 4-bit memory data)

Display Memory



LD Bus Output

8-bit type

LD0 3-0 → 35-32 ···

LD1 7-4 → 39-36 ···

LD2 11-8 → 43-40 ···

LD3 15-12 → 47-44 ···

LD4 19-16 → 51-48 ···

LD5 23-20 → 55-52 ···

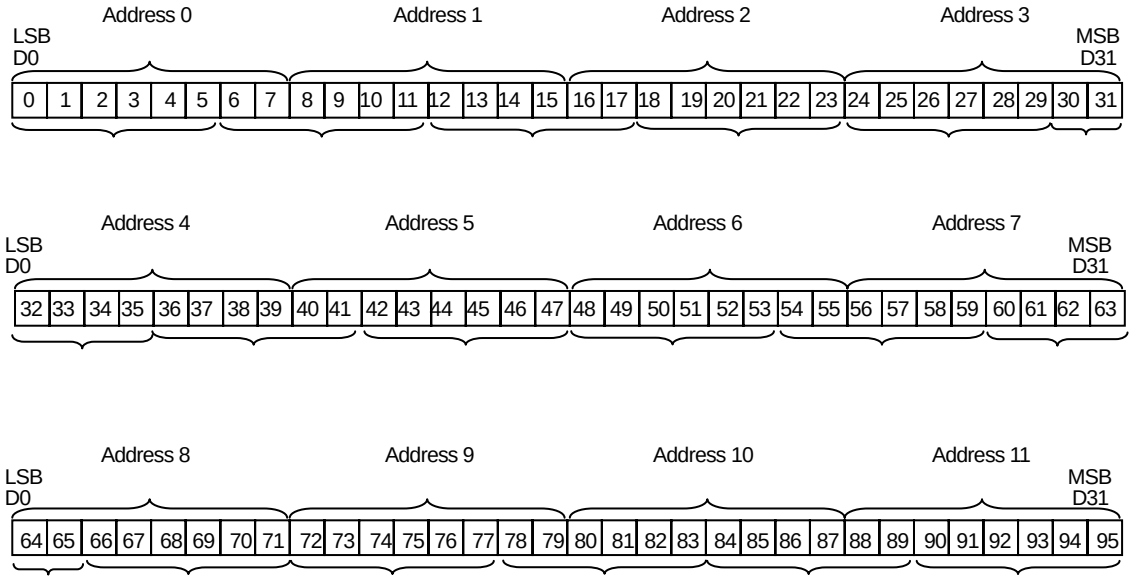
LD6 27-24 → 59-56 ···

LD7 31-28 → 63-60 ···

Figure 3.19.3 Memory Map Image and Data Output in STN 8-/16-Grayscale Mode

STN 64-grayscale (1-pixel display data = 6-bit memory data)

Display Memory



LD Bus Output

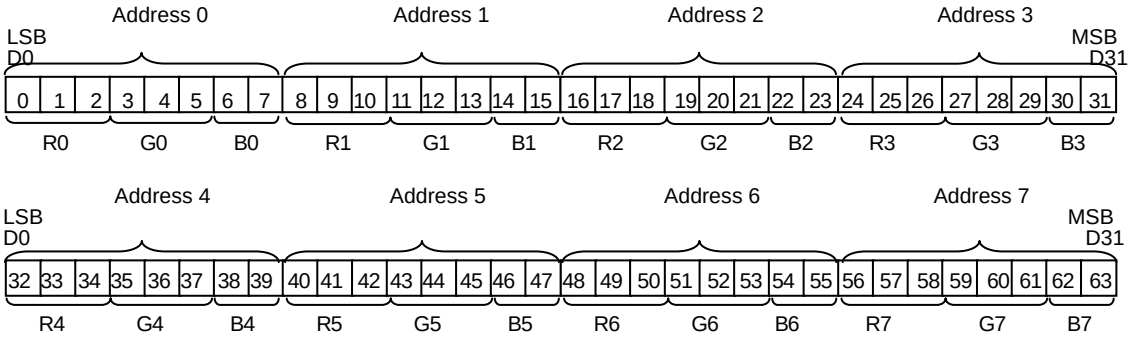
8-bit type

LD0	5-0	→ 53-48
LD1	11-6	→ 59-54
LD2	17-12	→ 65-60
LD3	23-18	→ 71-66
LD4	29-24	→ 77-72
LD5	35-30	→ 83-78
LD6	41-36	→ 89-84
LD7	47-42	→ 95-90

Figure 3.19.4 Memory Map Image and Data Output in STN 64-Grayscale Mode

STN 256-color (1-pixel display data = 8-bit memory data (R: 3 bits, G: 3 bits, B: 2 bits))

Display Memory



LD Bus Output

8-bit type

LD0 2-0(R0) → 23-22(B2) ···

LD1 5-3(G0) → 26-24(R3) ···

LD2 7-6(B0) → 29-27(G3) ···

LD3 10-8(R1) → 31-30(B3) ···

LD4 13-11(G1) → 34-32(R4) ···

LD5 15-14(B1) → 37-35(G4) ···

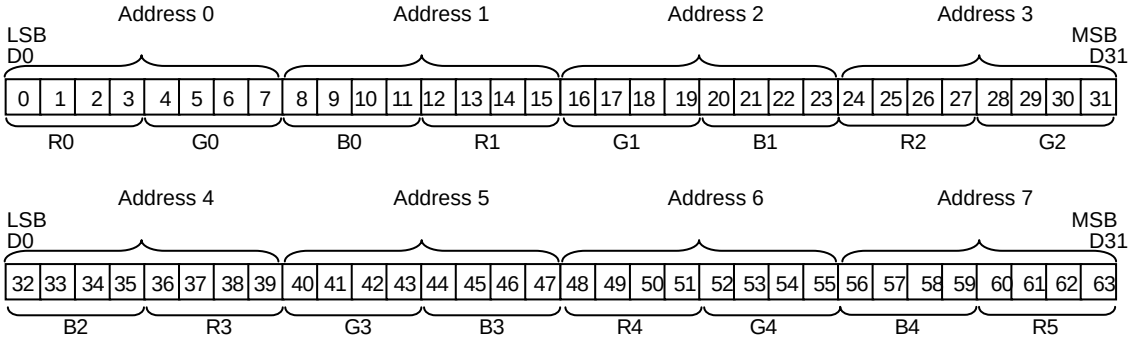
LD6 18-16(R2) → 39-38(B4) ···

LD7 21-19(G2) → 42-40(R5) ···

Figure 3.19.5 Memory Map Image and Data Output in STN 256-Color Mode

STN 4096-color (12 bpp: R: 4 bits, G: 4 bits, B: 4 bits)

Display Memory



LD Bus Output

8-bit type

LD0 3-0(R0) → 35-32(B2)…

LD1 7-4(G0) → 39-36(R3)…

LD2 11-8(B0) → 43-40(G3)…

LD3 15-12(R1) → 47-44(B3)…

LD4 19-16(G1) → 51-48(R4)…

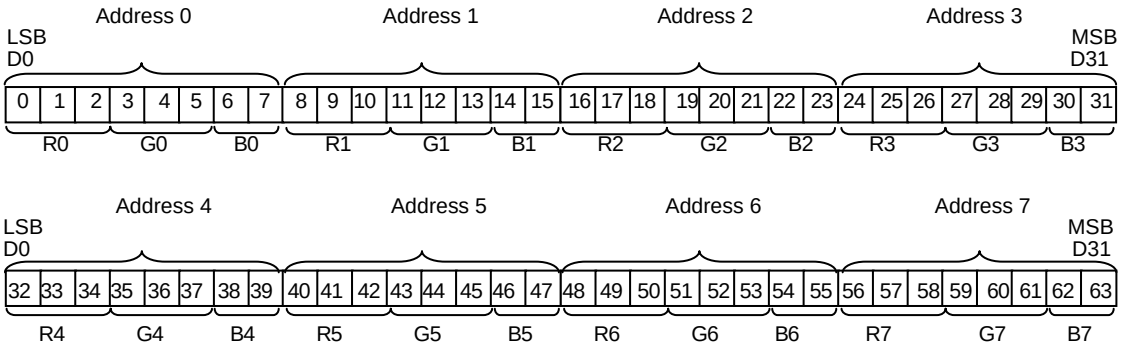
LD5 23-20(B1) → 55-52(G4)…

LD6 27-24(R2) → 59-56(B4)…

LD7 31-28(G2) → 63-60(R5)…

Figure 3.19.6 Memory Map Image and Data Output in STN 4096-Color Mode

TFT 256-color (1-pixel display data = 8-bit memory data (R: 3 bits, G: 3 bits, B: 2 bits)
Display Memory



12bit (TFT)

LD0 0(R0) → 12(R1) ···
LD1 1(R0) → 13(R1) ···
LD2 2(R0) → 14(R1) ···
LD3 3(G0) → 15(G1) ···
LD4 4(G0) → 16(G1) ···
LD5 5(G0) → 17(G1) ···
LD6 6(B0) → 18(B1) ···
LD7 7(B0) → 19(B1) ···

Figure 3.19.7 Memory Map Image and Data Output in TFT 256-Color Mode

TFT 4096-color (1-pixel display data = 12-bit memory data (R: 4 bits, G: 4 bits, B: 4 bits)

Display Memory

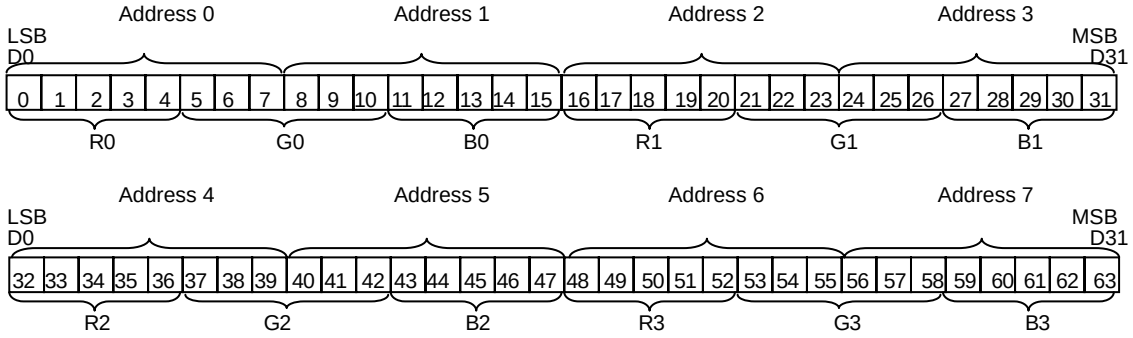
12-bit TFT

LD0 0(R0) → 12(R1) ···
LD1 1(R0) → 13(R1) ···
LD2 2(R0) → 14(R1) ···
LD3 3(R0) → 15(R1) ···
LD4 4(G0) → 16(G1) ···
LD5 5(G0) → 17(G1) ···
LD6 6(G0) → 18(G1) ···
LD7 7(G0) → 19(G1) ···
LD8 8(B0) → 20(B1) ···
LD9 9(B0) → 21(B1) ···
LD10 10(B0) → 22(B1) ···
LD11 11(B0) → 23(B1) ···

Figure 3.19.8 Memory Map Image and Data Output in TFT 4096-Color Mode

TFT 65536-color (16 bpp: R: 5 bits, G: 6 bits, B: 5 bits)

Display Memory



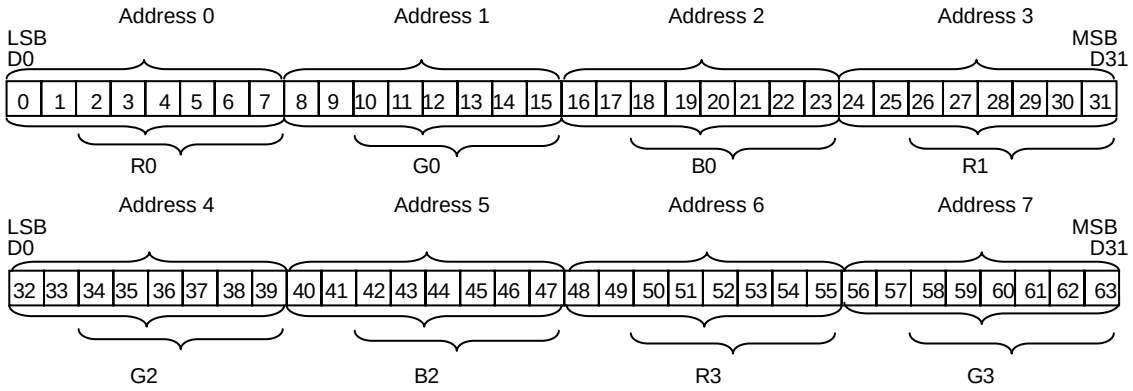
16-bit TFT

LD0 0(R0) → 16(R1) ···
LD1 1(R0) → 17(R1) ···
LD2 2(R0) → 18(R1) ···
LD3 3(R0) → 19(R1) ···
LD4 4(R0) → 20(R1) ···
LD5 5(G0) → 21(G1) ···
LD6 6(G0) → 22(G1) ···
LD7 7(G0) → 23(G1) ···
LD8 8(G0) → 24(G1) ···
LD9 9(G0) → 25(G1) ···
LD10 10(G0) → 26(G1) ···
LD11 11(B0) → 27(B1) ···
LD12 12(B0) → 28(B1) ···
LD13 13(B0) → 29(B1) ···
LD14 14(B0) → 31(B1) ···
LD15 15(B0) → 32(B1) ···

Figure 3.19.9 Memory Map Image and Data Output in TFT 65536-Color Mode

TFT 262144-/16777216-color (24 bpp: R: 8 bits, G: 8 bits, B: 8 bits)

Display Memory



24-bit TFT		18-bit TFT	
LD18	0(R0) → 24(R1) ···		
LD19	1(R0) → 25(R1) ···		
LD0	2(R0) → 26(R1) ···	LD0	2(R0) → 26(R1) ···
LD1	3(R0) → 27(R1) ···	LD1	3(R0) → 27(R1) ···
LD2	4(R0) → 28(R1) ···	LD2	4(R0) → 28(R1) ···
LD3	5(R0) → 29(R1) ···	LD3	5(R0) → 29(R1) ···
LD4	6(R0) → 30(R1) ···	LD4	6(R0) → 30(R1) ···
LD5	7(R0) → 31(R1) ···	LD5	7(R0) → 31(R1) ···
LD20	8(G0) → 32(G1) ···		
LD21	9(G0) → 33(G1) ···		
LD6	10(G0) → 34(G1) ···	LD6	10(G0) → 34(G1) ···
LD7	11(G0) → 35(G1) ···	LD7	11(G0) → 35(G1) ···
LD8	12(G0) → 36(G1) ···	LD8	12(G0) → 36(G1) ···
LD9	13(G0) → 37(G1) ···	LD9	13(G0) → 37(G1) ···
LD10	14(G0) → 38(G1) ···	LD10	14(G0) → 38(G1) ···
LD11	15(G0) → 39(G1) ···	LD11	15(G0) → 39(G1) ···
LD22	16(B0) → 40(B1) ···		
LD23	17(B0) → 41(B1) ···		
LD12	18(B0) → 42(B1) ···	LD12	18(B0) → 42(B1) ···
LD13	19(B0) → 43(B1) ···	LD13	19(B0) → 43(B1) ···
LD14	20(B0) → 44(B1) ···	LD14	20(B0) → 44(B1) ···
LD15	21(B0) → 45(B1) ···	LD15	21(B0) → 45(B1) ···
LD16	22(B0) → 46(B1) ···	LD16	22(B0) → 46(B1) ···
LD17	23(B0) → 47(B1) ···	LD17	23(B0) → 47(B1) ···

Note: The display RAM data format for 18 bpp is the same as that for 24 bpp. When 18 bpp is used, the least significant bit should be disabled by port setting.

Figure 3.19.10 Memory Map Image and Data Output in TFT 262144-/16777216-Color Mode

7. LDIV Signal

The <LDINV> and <AUTOINV> bits of the LCDMODE1 register are used to control the LDIV signal as well as data output. The LDIV signal indicates the inversion of all the LD bus signals.

When LCDMODE1<LDINV>=1, all display data is forcefully inverted and the LDIV signal is also driven high. When LCDMODE1<AUTOINV>=1, the data that has just been transferred and the data to be transferred next are compared. If there are more changed bits than unchanged bits (for example, 7 or more bits are changed when using a 12-bit bus, and 5 or more bits are changed when using a 8-bit bus), the data is inverted and the LDIV signal is also driven high. This function can be used with TFT source drivers having the data inversion function to reduce radiated noise and power consumption due to high-speed data inversion.

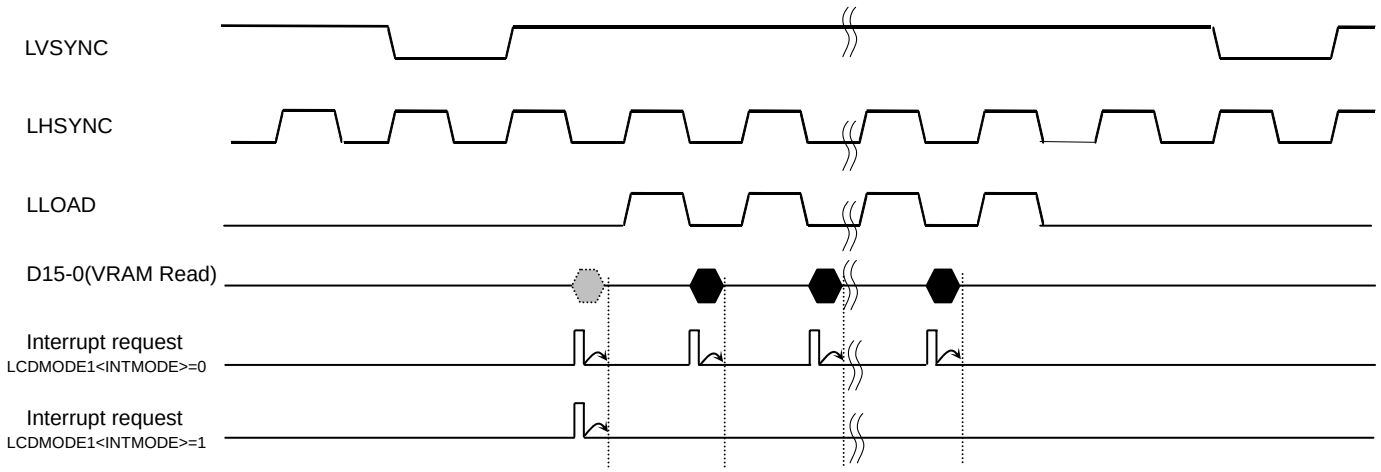
If <LDINV> and <AUTOINV> are both set to "1" at the same time, <LDINV> is given priority and <AUTOINV> is disabled.

3.19.4 Interrupt Function

The LCDC has two types of interrupts.

One is generated synchronous with the LLOAD signal and the other is generated synchronous with the LLOAD signal that is output immediately after the LVSYNC signal.

LCDMODE1<INTMODE> is used to switch between these two types of interrupts.



When LCDMODE1<INTMODE>=0, an interrupt request is generated at the start of each VRAM read before the LLOAD generates (once in each LLOAD period).

When LCDMODE1<INTMODE>=1, an interrupt request is generated at the start of VRAM read before the first LLOAD generates (once in each LVSYNC period).

**The interrupt request generates when reading the data from VRAM at once. Since reading from VRAM is executed by DMA with bus request to the CPU, DMA operation is given priority. Thus CPU accepts interrupt immediately after reading the data from VRAM.

LCDMODE1 Register

	7	6	5	4	3	2	1	0
bit Symbol	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	W	W
After reset	0	0	0	0	0	0	0	0
Function	Data rotation function (Supported for 64K-color: 16 bps only) 000: Normal 100: 90-degree 001: Horizontal flip 101: Reserved 010: Vertical flip 110: Reserved 011: Vertical & horizontal flip 111: Reserved			LD bus inversion 0: Normal 1: Invert	Auto bus inversion 0: Disable 1: Enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	LFR edge 0: LHSYNC Front Edge 1:LHSYNC Rear Edge	LD bus Trance Speed 0: normal 1: 1/3

LCDMODE1 (0281H)

Note: The LCDMODE1<INTMODE> setting must not be changed while the LCDC is operating. Be sure to set LCDCTL0<START> to “0” to stop the LCDC operation before changing the interrupt setting.

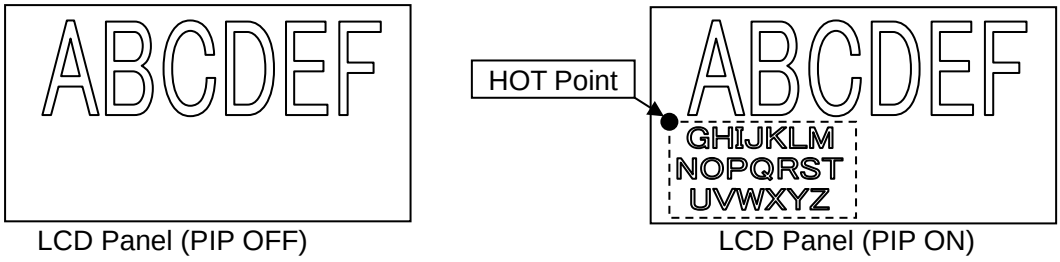
3.19.5 Special Functions

3.19.5.1 PIP (Picture in Picture) Function

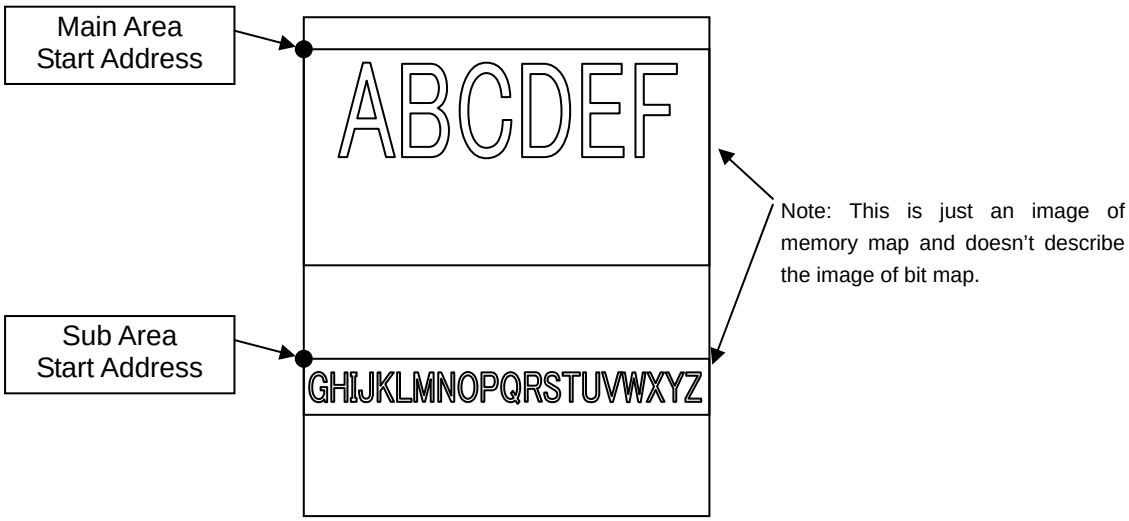
The TMP92CZ26A includes a PIP (Picture in Picture) function that allows a different screen to be displayed over the screen currently being displayed on the LCD.

The PIP function manages the address space of display memory by dividing it into “main screen” and “sub screen”. For the main screen, the display size and start address are specified as in the case of the normal screen display. For the sub screen, the display size and start address are also specified for determining the position and size of the sub screen.

When the HOT point (upper-left corner) and segment/common size are set for the sub screen and the PIP function is enabled by setting LCDCTL0 <PIPE> to “1”, the sub screen is displayed over the main screen.



LCD Panel (PIP OFF) LCD Panel (PIP ON)



VRAM Memory Map

The table below shows the HOT point locations that can be specified.

	*VRAM Access	HOT_Point(Y_dir)	HOT_Point(X_dir)
Monochrome display	16bit	In units of 16 dots	In units of 1 line
	32bit	In units of 32 dots	
4-grayscale display	16bit	In units of 8 dots	
	32bit	In units of 16 dots	
16-grayscale display	16bit	In units of 4 dots	
	32bit	In units of 8 dots	
64-grayscale display	16bit	In units of 8 dots	
	32bit	In units of 16 dots	
256-color display	16bit	In units of 2 dots	
	32bit	In units of 4 dots	
4K-color display	16bit	In units of 4 dots	
	32bit	In units of 8 dots	
64K-color display	16bit	In units of 1 dots	
	32bit	In units of 2 dots	
STN 256K-color display	16bit	In units of 8 dots	
	32bit	In units of 16 dots	
TFT 256k/16M-color display	16bit	In units of 2 dots	
	32bit	In units of 4 dots	

Note 1: The “VRAM Access” colomn shows the bus size for accessing the display RAM. When external RAM is used, the bus size depends on the bit width of the external RAM to be used. When the internal RAM is used VRAM is always accessed via a 32-bit bus.

Note 2: The same RAM must be used for both the main and sub areas.

The table below shows the HOT point segment and common sizes that can be specified.

	*VRAM Access	Segment size		Common size
		Minimum size	units	
Monochrome display	16bit	32 dots	In units of 16 dots	In units of 1 line
	32bit	64 dots	In units of 32 dots	
4-grayscale display	16bit	16 dots	In units of 8 dots	
	32bit	32 dots	In units of 16 dots	
16-grayscale display	16bit	8 dots	In units of 4 dots	
	32bit	16 dots	In units of 8 dots	
64-grayscale display	16bit	16 dots	In units of 8 dots	
	32bit	32 dots	In units of 16 dots	
256-color display	16bit	4 dots	In units of 2 dots	
	32bit	8 dots	In units of 4 dots	
4K-color display	16bit	8 dots	In units of 4 dots	
	32bit	16 dots	In units of 8 dots	
64K-color display	16bit	2 dots	In units of 1 dots	
	32bit	4 dots	In units of 2 dots	
STN 256K-color display	16bit	16 dots	In units of 8 dots	
	32bit	32 dots	In units of 16 dots	
TFT 256k/16M-color display	16bit	4 dots	In units of 2 dots	
	32bit	8 dots	In units of 4 dots	

LCD Main Area Start Address Register

		7	6	5	4	3	2	1	0
LSAML (02A0H)	bit Symbol	LMSA7	LMSA6	LMSA5	LMSA4	LMSA3	LMSA2	LMSA1	
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	After reset	0	0	0	0	0	0	0	
	Function	LCD main area start address (A7-A0)							
		7	6	5	4	3	2	1	0
LSAMM (02A1H)	bit Symbol	LMSA15	LMSA14	LMSA13	LMSA12	LMSA11	LMSA10	LMSA9	LMSA8
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD main area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSAMH (02A2H)	bit Symbol	LMSA23	LMSA22	LMSA21	LMSA20	LMSA19	LMSA18	LMSA17	LMSA16
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	1	0	0	0	0	0	0
	Function	LCD main area start address (A23-A16)							

LCD Sub Area Start Address Register

		7	6	5	4	3	2	1	0
LSASL (02A4H)	bit Symbol	LSSA7	LSSA6	LSSA5	LSSA4	LSSA3	LSSA2	LSSA1	
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	After reset	0	0	0	0	0	0	0	
	Function	LCD sub area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSASM (02A5H)	bit Symbol	LSSA15	LSSA14	LSSA13	LSSA12	LSSA11	LSSA10	LSSA9	LSSA8
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSASH (02A6H)	bit Symbol	LSSA23	LSSA22	LSSA21	LSSA20	LSSA19	LSSA18	LSSA17	LSSA16
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	1	0	0	0	0	0	0
	Function	LCD sub area start address (A23-A16)							

LCD Sub Area HOT Point Register (X-dir)

		7	6	5	4	3	2	1	0
LSAHX (02A8H)	bit Symbol	SAH7	SAH6	SAH5	SAH4	SAH3	SAH2	SAH1	SAH0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02A9H)	bit Symbol							SAH9	SAH8
	Read/Write							R/W	R/W
	After reset							0	0
	Function	LCD sub area HOT point (9-8)							

LCD Sub Area HOT Point Register (Y-dir)

		7	6	5	4	3	2	1	0
LSAHY (02AAH)	bit Symbol	SAHY7	SAHY6	SAHY5	SAHY4	SAHY3	SAHY2	SAHY1	SAHY0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02ABH)	bit Symbol								SAHY8
	Read/Write								R/W
	After reset								0
	Function								LCD sub area HOT point (9-8)

Note: The HOT point should be set in units of the specified number of dots, which is determined by the display color mode and display RAM access data bus width.

LCD Sub Area Display Segment Size Register

		7	6	5	4	3	2	1	0
LSASS (02ACH)	bit Symbol	SAS7	SAS6	SAS5	SAS4	SAS3	SAS2	SAS1	SAS0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area segment size (7-0)							
		7	6	5	4	3	2	1	0
(02ADH)	bit Symbol							SAS9	SAS8
	Read/Write							R/W	R/W
	After reset							0	0
	Function	LCD sub area segment size (9-8)							

Note: The segment size should be set in units of the specified number of dots, which is determined by the display color mode and display RAM access data bus width.

LCD Sub Area Display Common Size Register

		7	6	5	4	3	2	1	0
LSACS (02AEH)	bit Symbol	SAC7	SAC6	SAC5	SAC4	SAC3	SAC2	SAC1	SAC0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	LCD sub area common size (7-0)							
		7	6	5	4	3	2	1	0
(02AFH)	bit Symbol								SAC8
	Read/Write								R/W
	After reset								0
	Function	LCD sub area common size (8)							

Note: The common size should be set in units of 1 line.

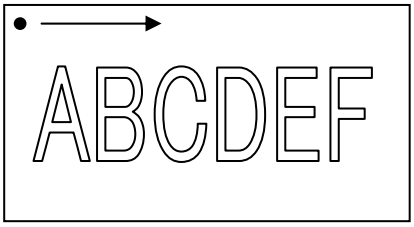
3.19.5.2 Display Data Rotation Function

When display RAM data is output to the LCD driver (LCDD), the data output direction can be automatically rotated by hardware to meet the specifications of the LCDD (or LCD module) to be used.

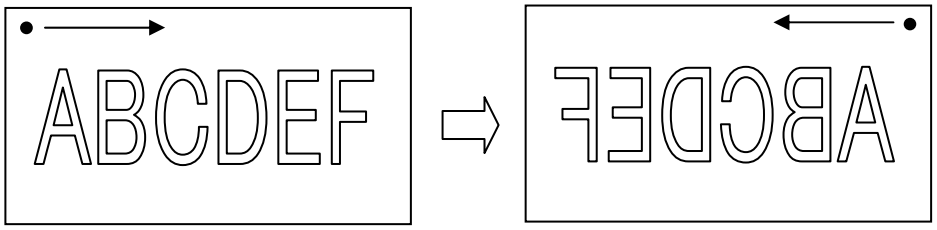
Table 3.19.2 Operation Conditions

Item	Vertical/Horizontal Flip Function	90-Degree Rotation Function
Display size	320 × 240	320×240 → 240 × 320
Color mode	64K colors (16 bpp)	64K colors (16 bpp)
Supported LCDD	TFT, STN	TFT, STN
Display RAM	Internal RAM, external SRAM	Internal RAM, external SRAM

1. Horizontal and Vertical Flip Function

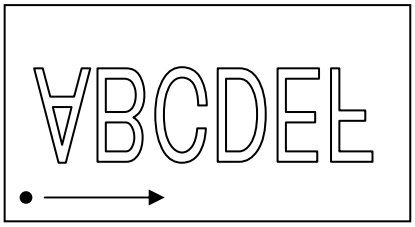


Display RAM image

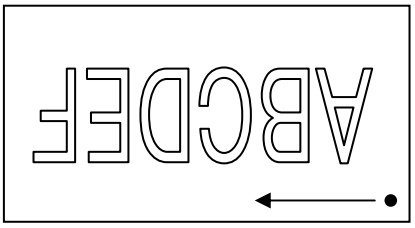


Normal display

Horizontally flipped



Vertically flipped



Horizontally and vertically flipped

The display RAM image shown above uses the data scan method for the normal display screen so that data is read from the display RAM and written to the LCDD from left to right and top to bottom.

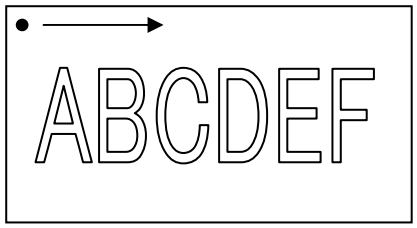
The data on the LCD screen appears as “horizontally flipped” if data is read from the display RAM from left to right and top to bottom and written to the LCDD from right to left and top to bottom.

Likewise, the data on the LCD screen appears as “vertically flipped” if data is written to the LCDD from left to right and bottom to top, or as “horizontally and vertically flipped” if the data is written to the LCDD from right to left and bottom to top.

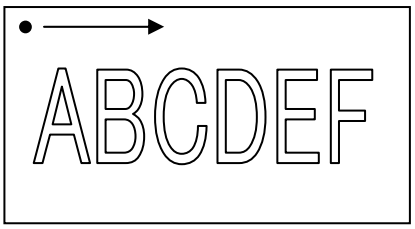
The horizontal and vertical flip function enables the output of display data to meet the specifications of each LCDD without the need to rearrange the display RAM data. In other words, the screen display can be flipped horizontally and vertically without

the need to rewrite the display RAM data.

2. 90-Degree Rotation Function



Display RAM Image (QVGA 320×240)



QVGA (320×240)

Portrait-type QVGA (240×320)
(when this function is used)

The display RAM image above shows typical data of QVGA size (320 segments × 240 commons: landscape type). If the LCDD to be used is of landscape type, the data can be written to the LCDD without any problem.

If the LCDD to be used is of portrait type (240 segments × 320 commons), the data cannot be displayed properly.

This function enables the orientation of each display image to be rotated 90 degrees without the need to change the display RAM data.

3. Setting Method

The <LDC2:0> bits in the LCDMODE1 register are used to set the display data rotation function.

LCDMODE1 Register

LCDMODE1 (0281H)

	7	6	5	4	3	2	1	0
bit Symbol	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	W	W
After reset	0	0	0	0	0	0	0	0
Function	Data rotation function (Supported for 64K-color: 16 bps only) 000: Normal 100: 90-degree 001: Horizontal flip 101: Reserved 010: Vertical flip 110: Reserved 011: Horizontal and vertical flip 111: Reserved			LD bus inversion 0: Normal 1: Invert	Auto bus inversion 0: Disable 1: Enable (Valid for TFT only)	Interrupt selection 0:LLOAD 1:LVSYNC	LFR edge 0: LHSYNC Front Edge 1:LHSYNC Rear Edge	LD bus Trance Speed 0: normal 1: 1/3

Note: The <LDC2:0> setting must not be changed while the LCDC is operating. Be sure to set LCDCTL0<START> to "0" to stop the LCDC operation before changing <LDC2:0>.

When the horizontal and vertical flip function or 90-degree rotation function is used, the display RAM start address of main/sub area should be set differently from when in normal mode, as shown in the table below.

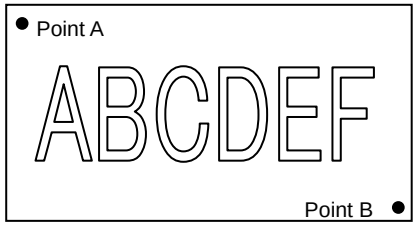
Mode	Setting Point	Display RAM Start Address Setting Example
Normal	Point A	00000h
90-degree rotation	Point B	257FEh
Horizontal flip	Point A	00000h
Vertical flip	Point B	257FEh
Horizontal and vertical flip	Point B	257FEh

How to calculate the point B address:

$(320 \times 240 \times 16/8) - 2 = 153600 - 2$

$= 153598$ [decimal]

$= 257FE$ [hex]



Display RAM Image (QVGA 320 × 240)

3.19.5.3 Considerations for Using the LCDC

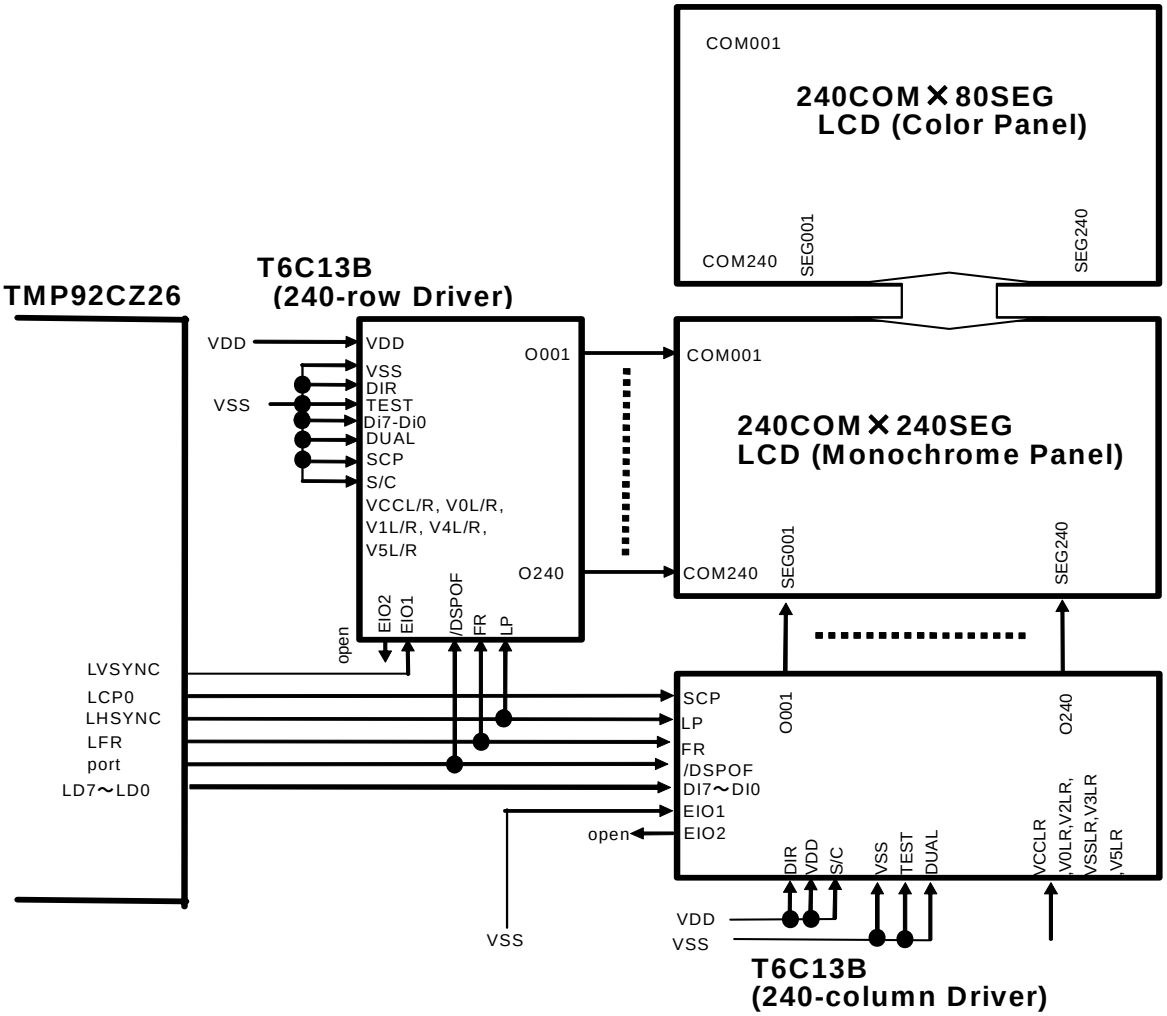
1. If the operation mode is changed while the LCDC is operating, a maximum of one frame may not be displayed properly. Although this degree of disturbance does not normally pose any problem (e.g. no response on LCD, display not visible to human eyes), the actual operation largely depends on the conditions such as the LCD driver, LCD panel, and frame frequency to be used. It is therefore recommended that operation checks be performed under the actual conditions.

2. The LCDMODE1<LDC2:0> setting must not be changed while the LCDC is operating. Be sure to set LCDCTL0<START> to “0” to stop the LCDC operation before changing <LDC2:0>.

3. The LCDC obtains the bus from the CPU when it has some operation to perform. Since the TMP92CZ26A includes other units that act as bus masters such as HDMA and SDRAMC, it is necessary to estimate the bus occupancy rate of each bus master in advance. For details, see the chapter on HDMA.

3.19.6 Setting Example

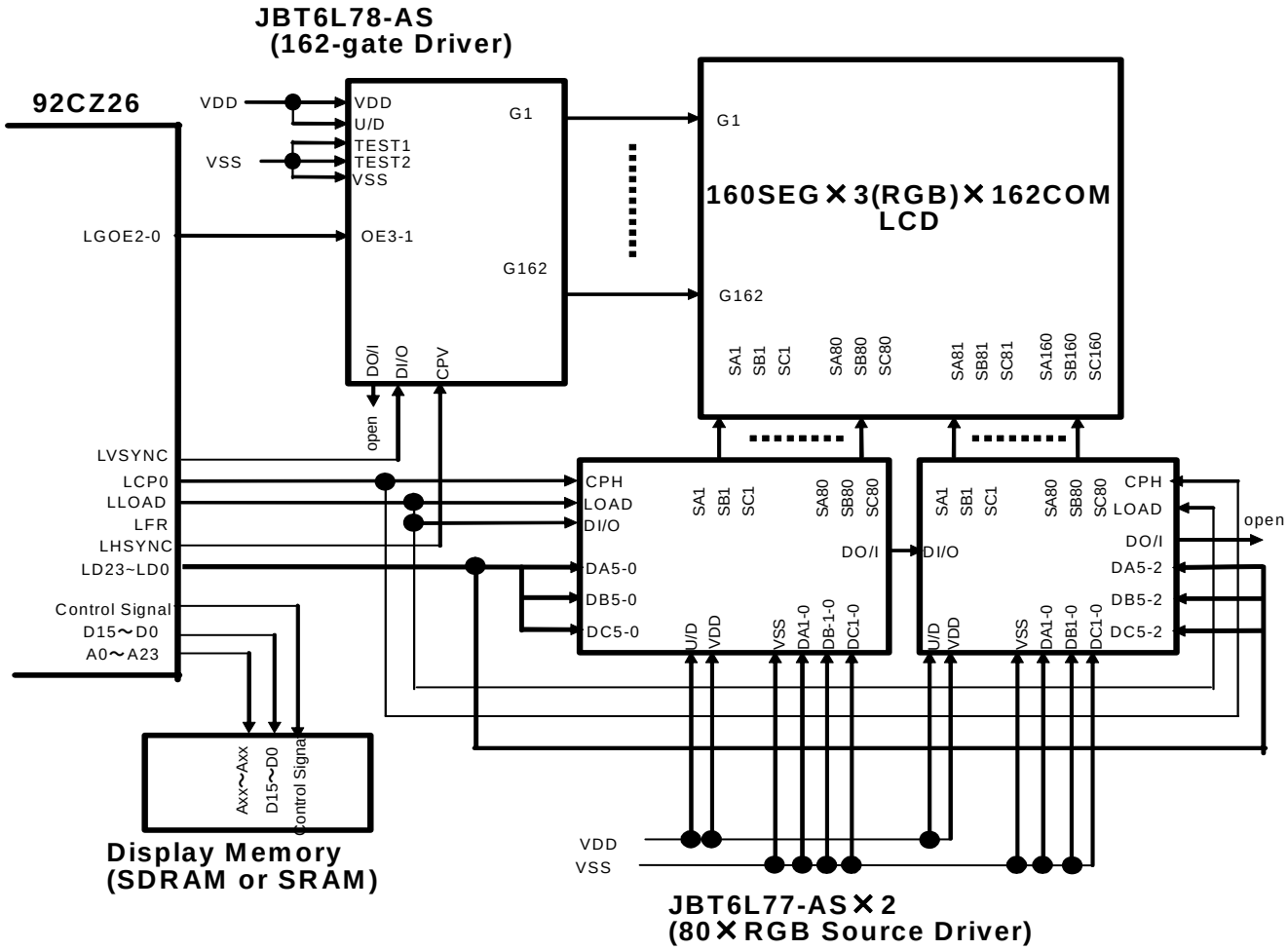
- STN



Note: The LCD drive power for LCD display must be supplied from an external circuit.

Figure 3.19.11 STN-Type LCD Driver Connection Example

- TFT



Note: The LCD drive power for LCD display must be supplied from and external circuit.

Figure 3.19.12 TFT-Type LCD Driver Connection Example

3.20 Touch Screen Interface (TSI)

The TMP92CZ26A has an interface for 4-terminal resistor network touch-screen.

This interface supports two procedures: an X/Y position measurement and touch detection.

Each procedure is performed by setting the TSI control register (TSICR0 and TSICR1) and using an internal AD converter.

3.20.1 Touch-Screen Interface Module Internal/External Connection

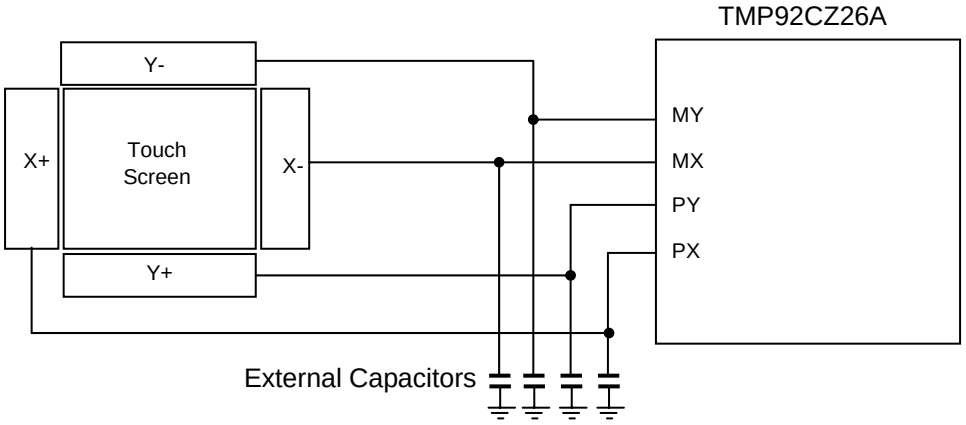


Figure 3.20.1External connection of TSI

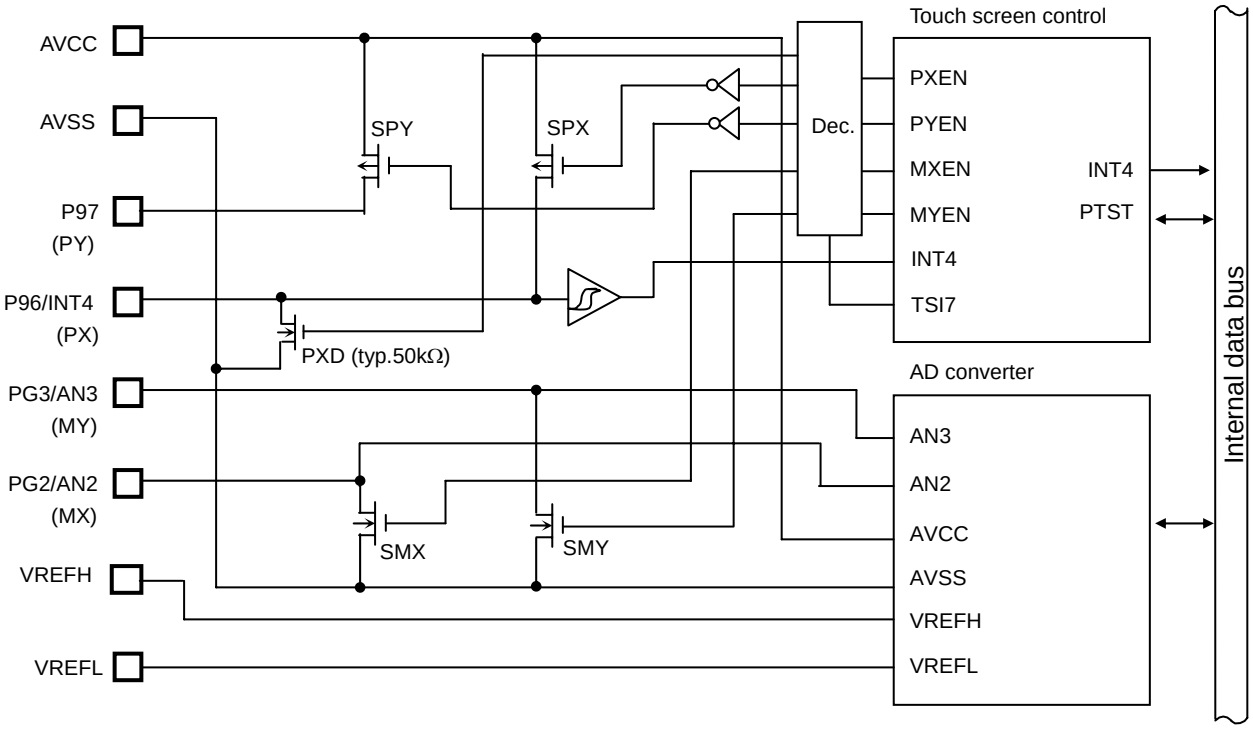


Figure 3.20.2 Internal block diagram of TSI

3.20.2 Touch Screen Interface (TSI) Control Register

TSI control register

TSICR0 (01F0H)

	7	6	5	4	3	2	1	0
bit Symbol	TSI7	INGE	PTST	TWIEN	PYEN	PXEN	MYEN	MXEN
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	0: Disable 1: Enable	Input gate control of Port 96,97 0: Enable 1: Disable	Detection condition 0: no touch 1: touch	INT4 interrupt control 0: Disable 1: Enable	SPY 0 : OFF 1 : ON	SPX 0 : OFF 1 : ON	SMY 0 : OFF 1 : ON	SMX 0 : OFF 1 : ON

PXD (internal pull-down resistor) ON/OFF setting

<PXEN> / <TSI7>	0	1
0	OFF	OFF
1	ON	OFF

De-bounce time setting register

TSICR1 (01F1H)

	7	6	5	4	3	2	1	0
bit Symbol	DBC7	DB1024	DB256	DB64	DB8	DB4	DB2	DB1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	0: Disable 1: Enable	1024	256	64	8	4	2	1
		De-bounce time is set by "(N*64-16) / f_{SYS}"-formula. "N" is sum of number which is set to "1" in bit6 to bit 0. Note3:						

Note1: Since an internal clock is used for de-bounce circuit, when IDLE1, STOP mode or PCM condition, the de-bounce circuit don't operate and also interrupt which through this circuit is not generated. When IDLE1, STOP mode or PCM condition, set this circuit to disable (Write "0" to TSICR1<DBC7>) before entering HALT state.If de-bounce time is set to "0", signal is received after counting the 6-system clock (f_{SYS}) from the condition that this circuit is set to disable.

Note2: During converting the analog input-data by using AD converter, the current flow to the normal C-MOS input-gate. Therefore, provide its current by setting TSICR0<INGE>.If the middle voltage is inputted, cut the input-signal to C-MOS logic (P96,P97) by settig this bit.

Note3: TSICR0<PTST> is that confirming initial pen-touch. When the input-signal to C-MOS logic is blocked by TSICR0<INGE>, this bit is always "1". Please be careful.

Ex:

TSICR1=95H $\rightarrow$N = 64 + 4 + 1 = 69

3.20.3 Touch detection procedure

A Touch detection procedure shows procedure until a pen is touched by the screen and it is detected.

By touching, TSI generates interrupt (INT4) and this procedure will terminate. After an X/Y position measuring procedure is terminated, return to this procedure and wait for next touch.

When touch is waiting, set SPY-switch to ON, and set other 3 switches (SMY, SPX and SMX) to OFF. The pull-down resistor that is connected to P96/INT4/PX pin is set to ON.

During waiting a touch, the internal resistors of X and Y-direction are not connected. Therefore, P96/INT4/PX pin's level is set to Low by internal pull-down register (PXD) and INT4 isn't generated.

When pen was touched, the internal resistors of X and Y-direction are connected. Therefore, P96/INT4/PX pin's level is set to High by internal pull-down register (PXD) and INT4 is generated.

And the de-bounce-circuit is prepared for avoiding that INT4 of plural times generate by one-time touch. When de-bounce-time is set to TSICR1 register, the pulse of time less than its time is ignored.

The circuit detects the rising of signal, counts-up the time of the counter which is set, after count, receive the signal internal. During counting, when the signal is set to Low, counter is cleared. And the state become to state of waiting a rising edge.

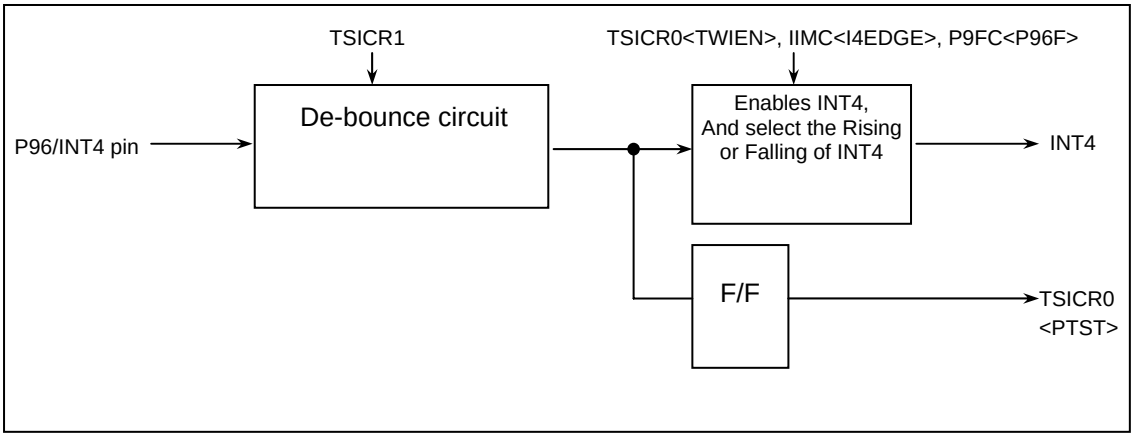


Figure 3.20.3 Block diagram of de-bounce circuit

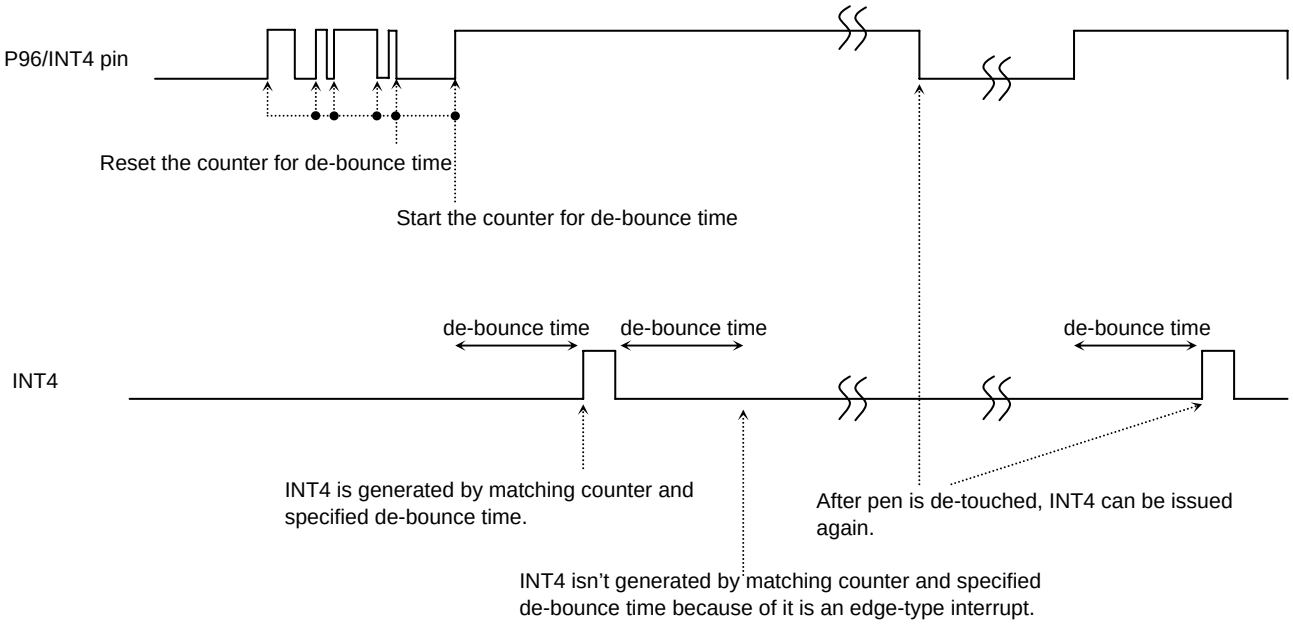


Figure 3.20.4 Timing diagram of de-bounce circuit

3.20.4 X/Y position measuring procedure

In the INT4 routine, execute an X/Y position measuring procedure like below.

<X position measurement>

At first, set both SPX, SMX-switches to ON, and set SPY, SMY-switches to OFF.

By this setting, analog-voltage which shows the X-position will be inputted to PG3/MY/AN3 pin.

The X-position can be measured by converting this voltage to digital code with AD converter.

<Y position measurement>

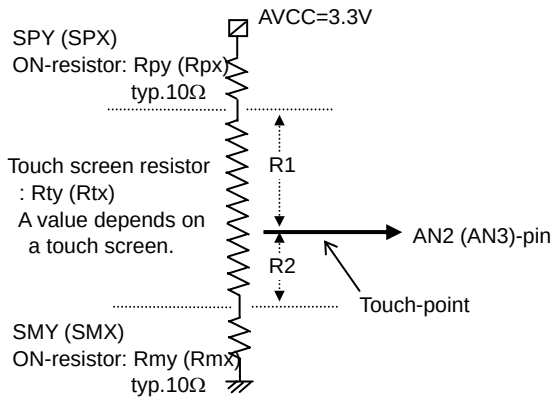
Next, set both SPY, SMY-switches to ON, and set SPX, SMX-switches to OFF.

By this setting, analog-voltage that shows the Y-position will be inputted to PG2/MX/AN2 pin.

The Y-position can be measured by converting this voltage to digital code with AD converter.

The above analog-voltage that is inputted to AN3 or AN2-pin can be calculated. It is a ratio between resistance-value in TMP92CZ26A and resistance-value in touch screen shown in Figure 3.20.5.

Therefore, if the pen touches a corner area on touch screen, analog-voltage will not be to 3.3V or 0.0V. As a notice, since each resistor has an uneven, consider about it. And it is recommended that an average code among a few times AD conversion will be adopted as a correct code.



[Calculation for analog input voltage to AN2 or AN3-pin : (E1)]

E1 = ((R2+Rmy) / (Rpy+Rty+Rmy)) × AVCC [V]

Ex.) The case of AVCC=3.3V, Rpy=Rmy=10Ω, R1=400Ω and R2=100Ω

E1 = ((100+10) / (10+400+100+10) × 3.3

= 0.698V

Note1: A Y-position can be calculated in the same way though above formula is for X-position.

Note2: Rty = R1+R2.

Figure 3.20.5 Calculation analog voltage

3.20.5 Flow chart for TSI

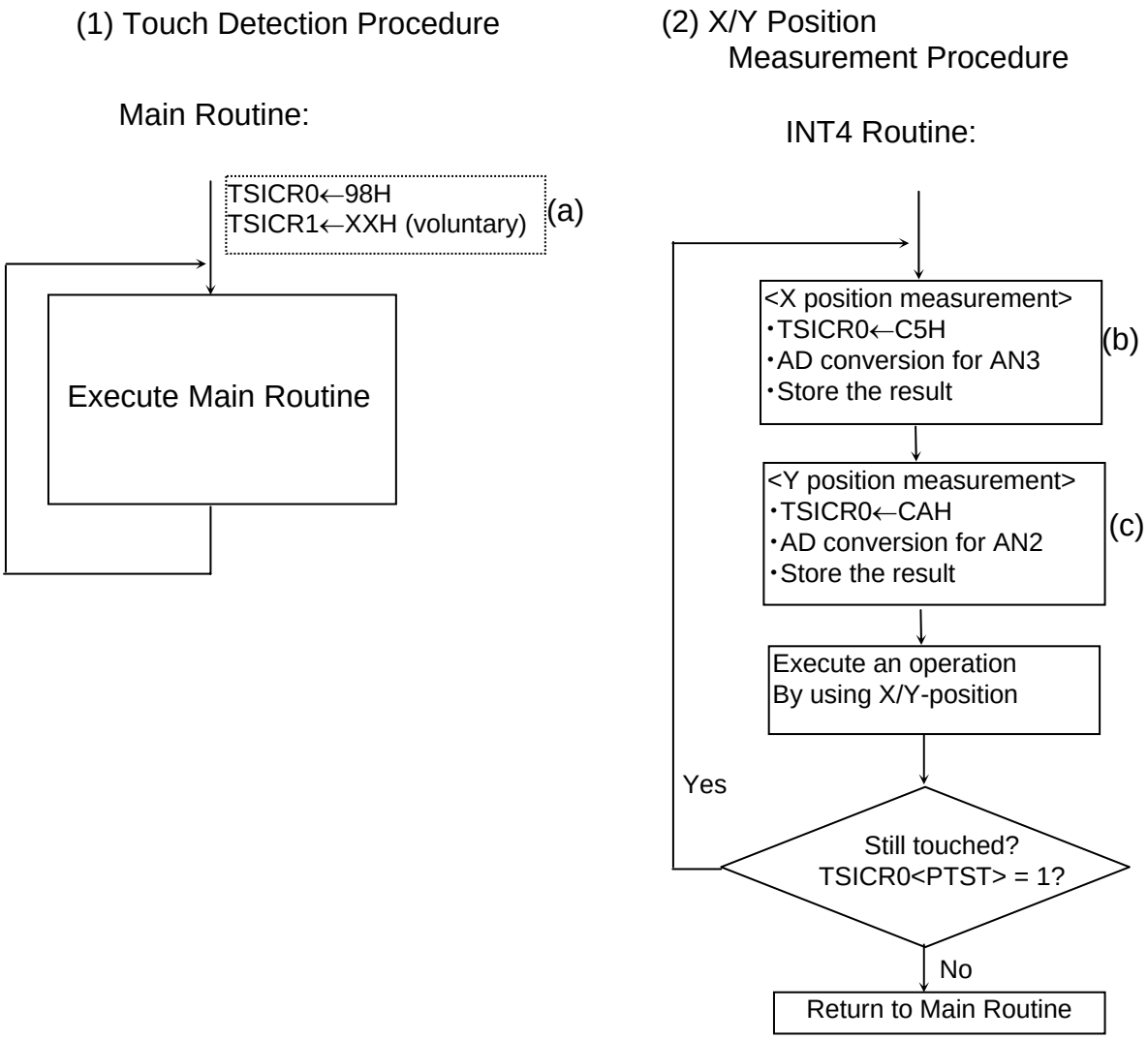
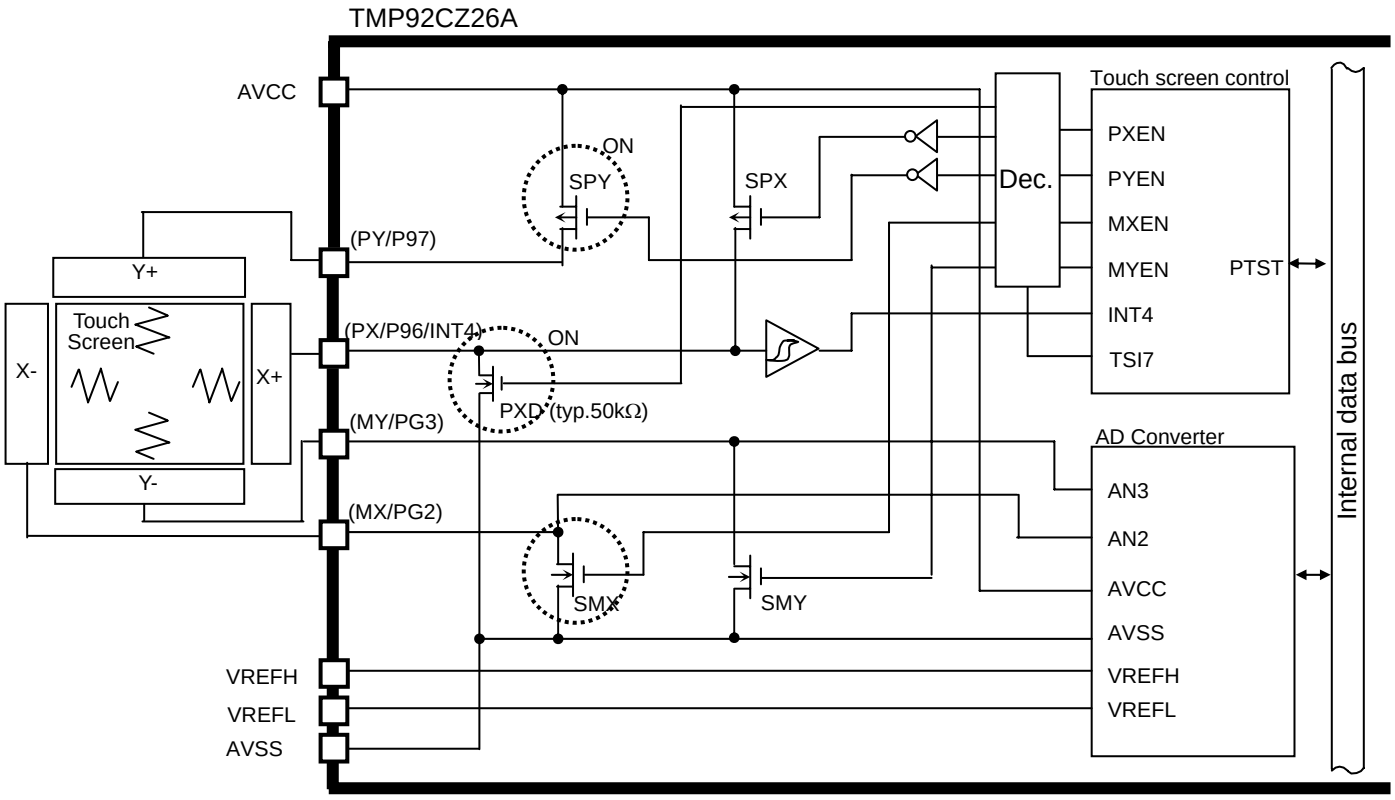


Figure 3.20.6 Flow chart for TSI

Following pages explain each circuit condition of (a), (b) and (c) in above flow chart.

(a) Main routine (condition of waiting INT4 interrupt)

(p9fc)<P96F>, <P97F>= "1"	:	Set P96 to int4/PX, set P97 to PY
(inte34)	:	Set interrupt level of INT4
(tsicr0)=98h	:	Pull-down resistor on, SPY on, Interrupt-set<TWIEN>
ei	:	Enable interrupt

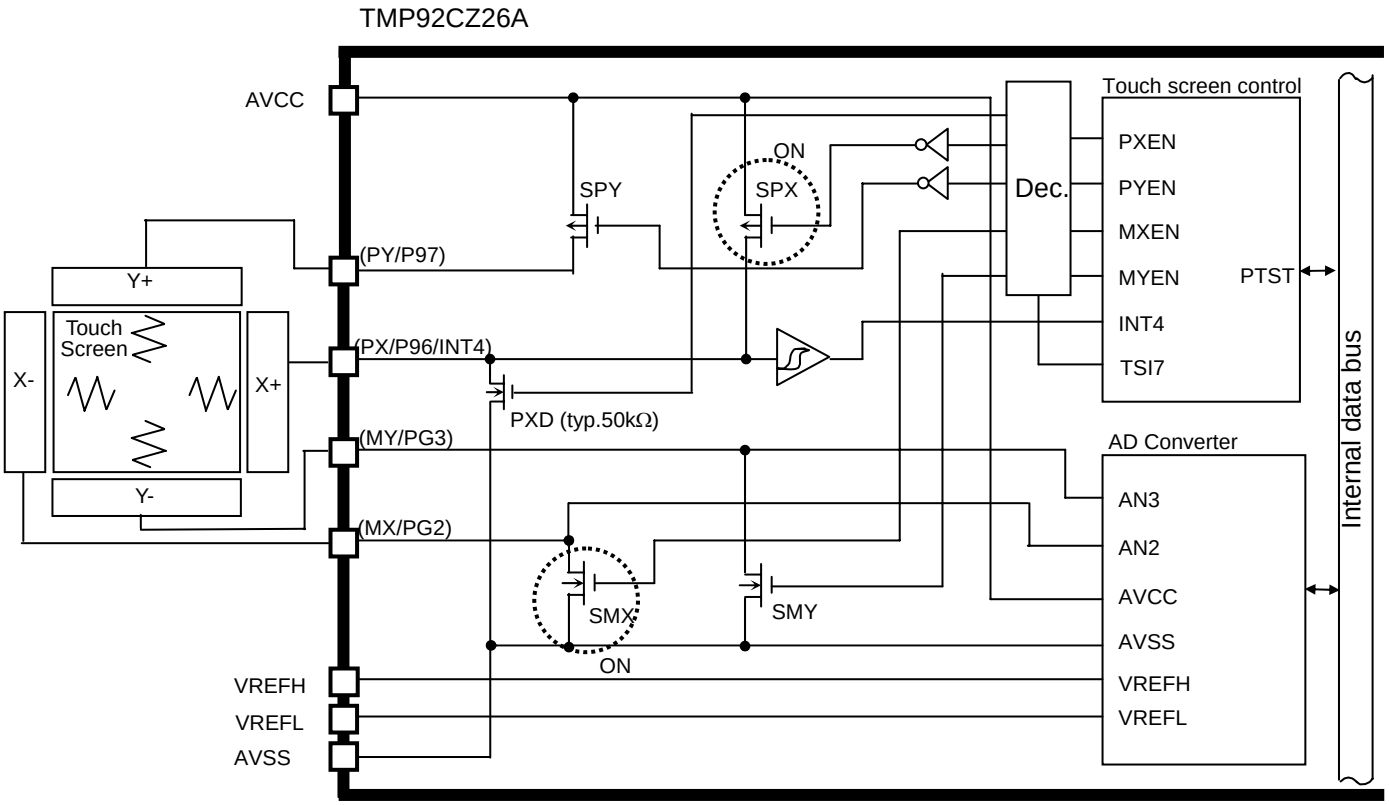


(b) X position measurement (Start AD conversion)

(tsicr0)=c5h : Set SMX, SPX to ON. Set the input gate of P97, P96 to OFF.

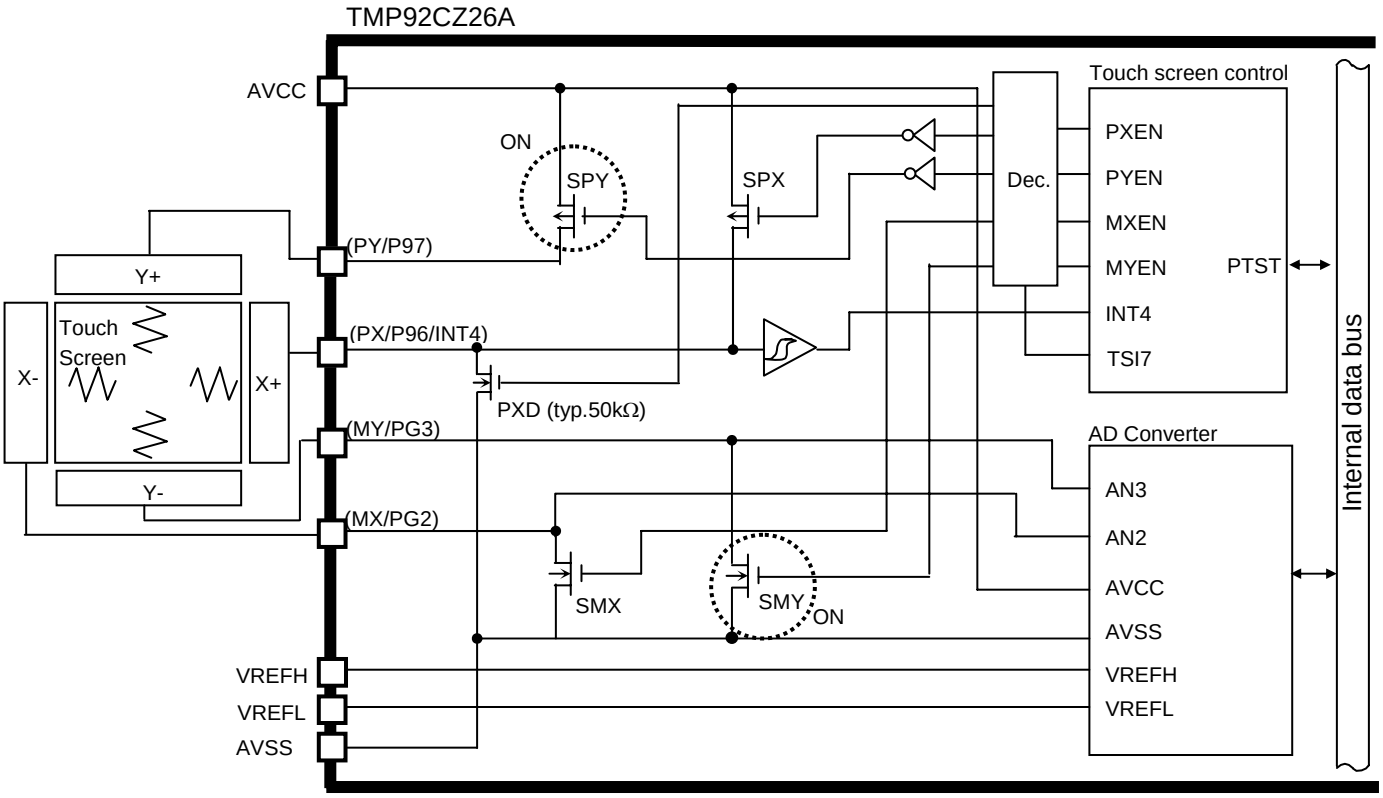
(admod1)=b0h : Set to AN3.

(admod0)=08h : Start AD conversion.



(c) Y position measurement(Start AD conversion)

(tsicr0)=cah : Set SMX, SPX to ON. Set the input gate of P97, P96 to OFF.
(admod1)=a0h : Set to AN2.
(admod0)=08h : Start AD conversion.



3.20.6 Note

1. De-bounce circuit

   The system clock of CPU is used in de-bounce circuit. Therefore, de-bounce circuit is not operated when clock is not supplied to CPU (IDLE1, STOP mode or PCM mode). And, an interrupt which through the de-bounce circuit is not generated.

   When started from IDLE1, STOP or PCM mode by using TSI, set the de-bounce circuit to disable before a condition become to HALT or PCM mode. (TSICR1<DBC7>="0")

2. Port setting

   During conversion the middle voltage of 0V~AVcc by using AD converter, the middle voltage is inputted to a normal C-MOS input-gate (P96 and P97), too.

   Therefore, provide the flow current for P96 and P97 by using TSICR0<INGE>. In this case (TSICR0<INGE>="1"), when the input to C-MOS logic is cut, TSICR0<PTST> for confirming a first pen touch is always set to “1”. Please be careful.

3.21 Real time clock (RTC)

3.21.1 Function description for RTC

1) Clock function (hour, minute, second)
2) Calendar function (month and day, day of the week, and leap year)
3) 24 or 12-hour (AM/PM) clock function
4) +/- 30 second adjustment function (by software)
5) Alarm function (Alarm output)
6) Alarm interrupt generate

3.21.2 Block diagram

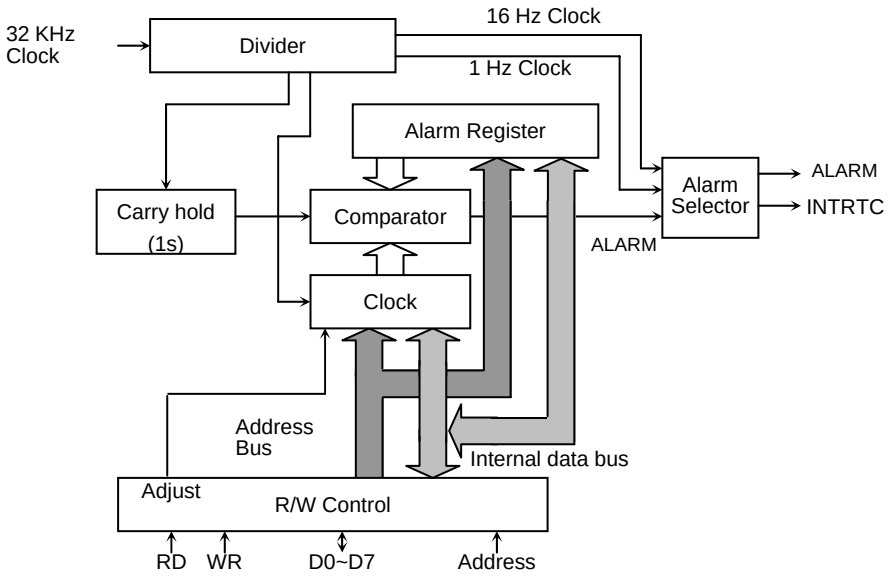


Figure 3.21.1 RTC block diagram

Note1: The Christian era year column:
This product has year column toward only lower two columns. Therefore the next year in 99 works as 00 years. In system to use it, please manage upper two columns with the system side when handle year column in the Christian era.

Note2: Leap year:
A leap year is the year, which is divisible with 4, but the year, which there is exception, and is divisible with 100, is not a leap year. However, the year is divisible with 400, is a leap year. But there is not this product for the correspondence to the above exception. Because there are only with the year that is divisible with 4 as a leap year, please cope with the system side if this function is problem.

3.21.3 Control registers

Table 3.21.1 PAGE 0 (Timer function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	1320H		40 sec	20 sec	10 sec	8 sec	4 sec	2 sec	1 sec	Second column	R/W
MINR	1321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	1322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	1323H						W2	W1	W0	Day of the week column	R/W
DATER	1324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	1325H				Oct.	Aug.	Apr.	Feb.	Jan.	Month column	R/W
YEARR	1326H	Year 80	Year 40	Year 20	Year 10	Year 8	Year 4	Year 2	Year 1	Year column (Lower two columns)	R/W
PAGER	1327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	1328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write “0”				Reset register	W only

Note: As for SECR, MINR, HOURR, DAYR, MONTHR, YEARR of PAGE0, current state is read read it.

Table 3.21.2 PAGE1 (Alarm function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	1320H										R/W
MINR	1321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	1322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	1323H						W2	W1	W0	Day of the week column	R/W
DATER	1324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	1325H								24/12	24-hour clock mode	R/W
YEARR	1326H							LEAP1	LEAP0	Leap-year mode	R/W
PAGER	1327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	1328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write “0”				Reset register	W only

Note: As for SECR, MINR, HOURR, DAYR, MONTHR, YEARR of PAGE1, current state is read read it.

3.21.4 Detailed explanation of control register

RTC is not initialized by reset. Therefore, all registers must be initialized at the beginning of the program.

(1) Second column register (for PAGE0 only)

SECR (1320H)

	7	6	5	4	3	2	1	0
Bit symbol		SE6	SE5	SE4	SE3	SE2	SE1	SE0
Read/Write		R/W						
After reset		Undefined						
Function	"0" is read.	40 sec. column	20 sec. column	10 sec. column	8 sec. column	4 sec. column	2 sec. column	1 sec. column

0	0	0	0	0	0	0	0 sec
0	0	0	0	0	0	1	1 sec
0	0	0	0	0	1	0	2 sec
0	0	0	0	0	1	1	3 sec
0	0	0	0	1	0	0	4 sec
0	0	0	0	1	0	1	5 sec
0	0	0	0	1	1	0	6 sec
0	0	0	0	1	1	1	7 sec
0	0	0	1	0	0	0	8 sec
0	0	0	1	0	0	1	9 sec
0	0	1	0	0	0	0	10 sec
			:				
0	0	1	1	0	0	1	19 sec
0	1	0	0	0	0	0	20 sec
			:				
0	1	0	1	0	0	1	29 sec
0	1	1	0	0	0	0	30 sec
			:				
0	1	1	1	0	0	1	39 sec
1	0	0	0	0	0	0	40 sec
			:				
1	0	0	1	0	0	1	49 sec
1	0	1	0	0	0	0	50 sec
			:				
1	0	1	1	0	0	1	59 sec

Note: Do not set the data other than showing above.

(2) Minute column register (for PAGE0/1)

MINR (1321H)

	7	6	5	4	3	2	1	0
Bit symbol		MI6	MI5	MI4	MI3	MI2	MI1	MI0
Read/Write		R/W						
After reset		Undefined						
Function	"0" is read.	40 min, column	20 min, column	10 min, column	8 min, column	4 min, column	2 min, column	1 min, column

0	0	0	0	0	0	0	0 min
0	0	0	0	0	0	1	1 min
0	0	0	0	0	1	0	2 min
0	0	0	0	0	1	1	3 min
0	0	0	0	1	0	0	4 min
0	0	0	0	1	0	1	5 min
0	0	0	0	1	1	0	6 min
0	0	0	0	1	1	1	7 min
0	0	0	1	0	0	0	8 min
0	0	0	1	0	0	1	9 min
0	0	1	0	0	0	0	10 min
			:				
0	0	1	1	0	0	1	19 min
0	1	0	0	0	0	0	20 min
			:				
0	1	0	1	0	0	1	29 min
0	1	1	0	0	0	0	30 min
			:				
0	1	1	1	0	0	1	39 min
1	0	0	0	0	0	0	40 min
			:				
1	0	0	1	0	0	1	49 min
1	0	1	0	0	0	0	50 min
			:				
1	0	1	1	0	0	1	59 min

Note: Do not set the data other than showing above.

(3) Hour column register (for PAGE0/1)

1. In case of 24-hour clock mode (MONTHR<MO0>= “1”)

HOURR (1322H)

	7	6	5	4	3	2	1	0
Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
Read/Write			R/W					
After reset			Undefined					
Function	"0" is read.		20 hour column	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o’clock
0	0	0	0	0	1	1 o’clock
0	0	0	0	1	0	2 o’clock
			:			
0	0	1	0	0	0	8 o’clock
0	0	1	0	0	1	9 o’clock
0	1	0	0	0	0	10 o’clock
			:			
0	1	1	0	0	1	19 o’clock
1	0	0	0	0	0	20 o’clock
			:			
1	0	0	0	1	1	23 時

Note: Do not set the data other than showing above.

2. In case of 24-hour clock mode (MONTHR<MO0>= “0”)

HOURR (1322H)

	7	6	5	4	3	2	1	0
Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
Read/Write			R/W					
After reset			Undefined					
Function	"0" is read.		PM/AM	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o’clock (AM)
0	0	0	0	0	1	1 o’clock
0	0	0	0	1	0	2 o’clock
			:			
0	0	1	0	0	1	9 o’clock
0	1	0	0	0	0	10 o’clock
0	1	0	0	0	1	11 o’clock
1	0	0	0	0	0	0 o’clock (PM)
1	0	0	0	0	1	1 o’clock

Note: Do not set the data other than showing above.

(4) Day of the week column register (for PAGE0/1)

DAYR (1323H)

	7	6	5	4	3	2	1	0
Bit symbol						WE2	WE1	WE0
Read/Write						R/W		
After reset						Undefined		
Function	"0" is read.					W2	W1	W0

0	0	0	Sunday
0	0	1	Monday
0	1	0	Tuesday
0	1	1	Wednesday
1	0	0	Thursday
1	0	1	Friday
1	1	0	Saturday

Note: Do not set the data other than showing above.

(5) 日桁レジスタ (PAGE0/1)

DATER (1324H)

	7	6	5	4	3	2	1	0
Bit symbol			DA5	DA4	DA3	DA2	DA1	DA0
Read/Write			R/W					
After reset			Undefined					
Function	"0" is read.		Day 20	Day 10	Day 8	Day 4	Day 2	Day 1

0	0	0	0	0	0	0
0	0	0	0	0	1	1st day
0	0	0	0	1	0	2nd day
0	0	0	0	1	1	3rd day
0	0	0	1	0	0	4th day
:						
0	0	1	0	0	1	9th day
0	1	0	0	0	0	10th day
0	1	0	0	0	1	11th day
:						
0	1	1	0	0	1	19th day
1	0	0	0	0	0	20th day
:						
1	0	1	0	0	1	29th day
1	1	0	0	0	0	30th day
1	1	0	0	0	1	31st day

Note1: Do not set the data other than showing above.

Note2: Do not set the day which is not existed. (ex: 30th Feb)

(6) Month column register (for PAGE0 only)

MONTHR (1325H)

	7	6	5	4	3	2	1	0
Bit symbol				MO4	MO4	MO2	MO1	MO0
Read/Write				R/W				
After reset				Undefined				
Function	"0" is read.			10 months	8 months	4 months	2 months	1 month

0	0	0	0	1	January
0	0	0	1	0	February
0	0	0	1	1	March
0	0	1	0	0	April
0	0	1	0	1	May
0	0	1	1	0	June
0	0	1	1	1	July
0	1	0	0	0	August
0	1	0	0	1	September
1	0	0	0	0	October
1	0	0	0	1	November
1	0	0	1	0	December

Note: Do not set the data other than showing above.

(7) Select 24-hour clock or 12-hour clock (for PAGE1 only)

MONTHR (1325H)

	7	6	5	4	3	2	1	0
Bit symbol								MO0
Read/Write								R/W
After reset								Undefined
Function	"0" is read.							1: 24-hour 0: 12-hour

(8) Year column register (for PAGE0 only)

YEARR (1326H)

	7	6	5	4	3	2	1	0
Bit symbol	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
Read/Write	R/W							
After reset	Undefined							
Function	80 Years	40 Years	20 Years	10 Years	8 Years	4 Years	2 Years	1 Year

1	0	0	1	1	0	0	1	99 years
0	0	0	0	0	0	0	0	00 years
0	0	0	0	0	0	0	1	01 years
0	0	0	0	0	0	1	0	02 years
0	0	0	0	0	0	1	1	03 years
0	0	0	0	0	1	0	0	04 years
0	0	0	0	0	1	0	1	05 years
				:				
1	0	0	1	1	0	0	1	99 years

Note: Do not set the data other than showing above.

(9) Leap-year register (for PAGE1 only)

YEARR (1326H)

	7	6	5	4	3	2	1	0
Bit symbol							LEAP1	LEAP0
Read/Write							R/W	
After reset							Undefined	
Function	"0" is read.						00: leap-year 01: one year after leap-year 10: two years after leap-year 11: three years after leap-year	

0	0	Current year is leap-year
0	1	Current is next year of a leap year
1	0	Current is two years of a leap year
1	1	Current is three years of a leap year

(10) PAGE register (for PAGE0/1)

		7	6	5	4	3	2	1	0
PAGER (1327H)	Bit symbol	INTENA			ADJUST	ENATMR	ENAALM		PAGE
	Read/Write	R/W			W	R/W			R/W
	After reset	0			Undefined	Undefined			Undefined
Read-modify write instruction are prohibited	Function	(Note) Interrupt 1: Enable 0: Disable	"0" is read.		1: Adjust	TIMER 1: Enable 0: Disable	ALARM 1: Enable 0: Disable	"0" is read.	PAGE selection

Note: Pleas keep the setting order below and don't set same time.
(Set difference time to Clock/Alarm setting and interrupt setting)

(Example) Clock setting/Alarm setting

ld (pager), 0ch : Clock, Alarm enable

ld (pager), 8ch : Interrupt enable

PAGE	0	Select Page0
	1	Select Page1

ADJUST	0	Don't care
	1	Adjust sec. counter. When set this bit to "1" the sec. counter become to "0" when the value of sec. counter is 0 – 29. And in case that value of sec. counter is 30-59, min. counter is carried and become sec. counter to "0". Output Adjust signal during 1 cycle of f_{SYS}. After being adjusted once, Adjust is released automatically. (PAGE0 only)

(11) Reset register (for PAGE0/1)

		7	6	5	4	3	2	1	0
RESTR (1328H)	Bit symbol	DIS1Hz	DIS16Hz	RSTTMR	RSTALM	RE3	RE2	RE1	RE0
	Read/Write	W							
	After reset	Undefined							
Read-modify write instruction are prohibited	Function	0: 1 Hz	0: 16 Hz	1:Clock reset	1: Alarm reset	Always write "0"			

RSTALM	0	Unused
	1	Reset alarm register

RSTTMR	0	Unused
	1	Reset timer register

<DIS1HZ>	<DIS1HZ>	(PAGER) <ENAALM>	Source signal
1	1	1	Alarm
0	1	0	1Hz
1	0	0	16Hz
Others			Output "0"

3.21.5 Operational description

(1) Reading timer data

There is the case, which reads wrong data when carry of the inside counter happens during the operation which clock data reads. Therefore please read two times with the following way for reading correct data.

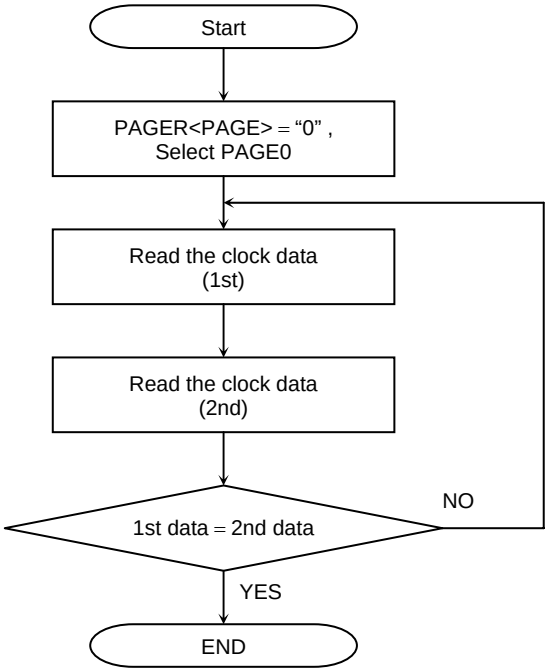


Figure 3.21.2 Flowchart of timer data read

(2) Timing of INTRTC and Clock data

When time is read by interrupt, read clock data within 0.5s(s) after generating interrupt. This is because count up of clock data occurs by rising edge of 1Hz pulse cycle.

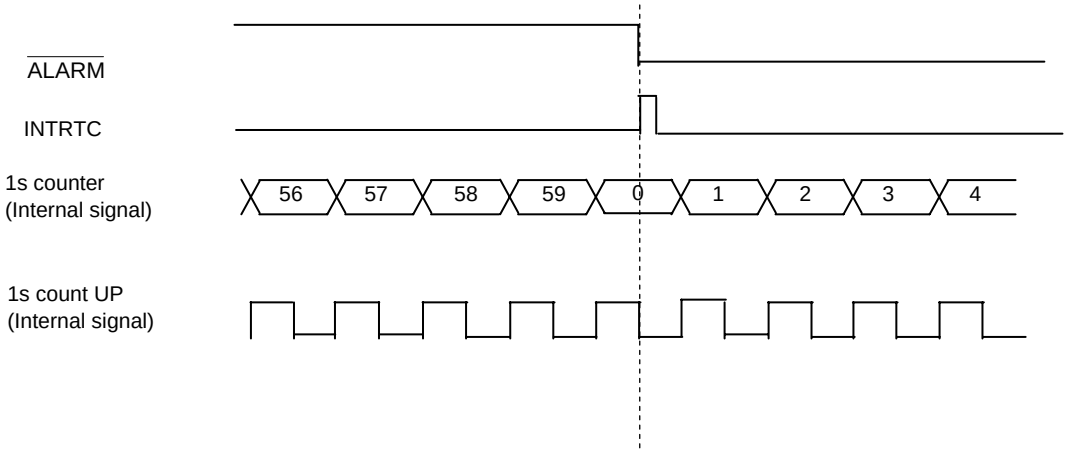


Figure 3.21.3 Timing of INTRTC and Clock data

(3) Writing timer data

When there is carry on the way of write operation, expecting data can not be wrote exactly. Therefore, in order to write in data exactly please follow the below way.

1. Resetting a divider

In RTC inside, there are 15-stage dividers, which generates 1Hz clock from 32,768 KHz. Carry of a timer is not done for one second when reset this divider. So write in data at this interval.

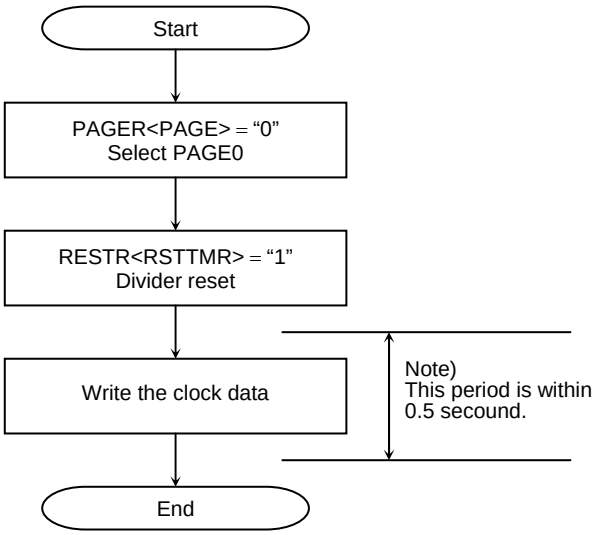


Figure 3.21.4 Flowchart of data write

2. Disabling the timer

Carry of a timer is prohibited when write “0” to PAGER<ENATMR> and can prevent malfunction by 1s Carry hold circuit. During a timer prohibited, 1s Carry hold circuit holds one sec. carry signal, which is generated from divider. After becoming timer enable state, output the carry signal to timer and revise time and continue operation. However, timer is late when timer-disabling state continues for one second or more. During timer disabling, pay attention with system power is downed. In this case the timer is stopped and time is delayed.

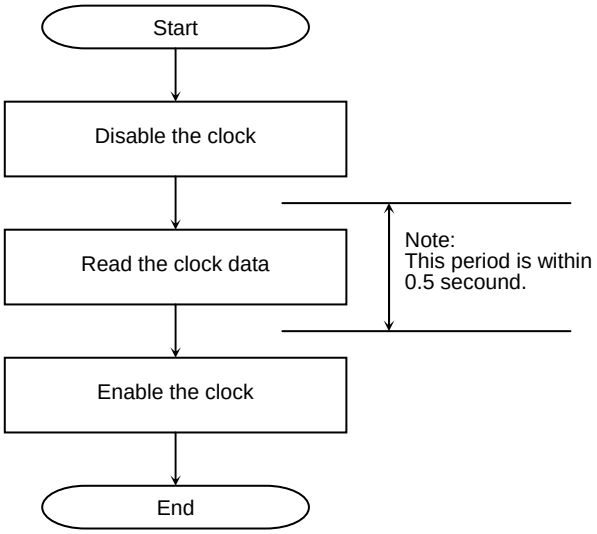


Figure 3.21.5 Flowchart of Clock disable

3.21.6 Explanation of the interrupt signal and alarm signal

Can use alarm function by setting of register of PAGE1 and output either of three signals from $\overline{\text{ALARM}}$ pin as follows by write "1" to PAGER<PAGE>. INTRTC outputs 1shot pulse when the falling edge is detected. RTC is not initializes by RESET. Therefore, when clock or alarm function is used, clear interrupt request flag in INTC (interrupt controller).

(1) In accordance of alarm register and the timer, output "0".

(2) Output clock of 1Hz.

(3) Output clock of 16Hz.

(1) In accordance with alarm register and a clock, output "0"

When value of a clock of PAGE0 accorded with alarm register of PAGE1 with a state of PAGER<ENAALM>= "1", output "0" to $\overline{\text{ALARM}}$ pin and occur INTRTC.

Follows are ways using alarm.

Initialization of alarm is done by writing in "1" at RESTR<RSTALM>, setting value of all alarm becomes don't care. In this case, always accorded with value of a clock and request INTRTC interrupt if PAGER<ENAALM> is "1".

Setting alarm min., alarm hour, alarm day and alarm the day week are done by writing in data at each register of PAGE1.

When all setting contents accorded, RTC generates INTRTC interrupt, if PAGER<INTENA><ENAALM> is "1". However, contents (don't care state) which does not set it up is considered to always accord.

The contents, which set it up once, cannot be returned to don't care state in independence. Initialization of alarm and resetting of alarm register set to don't care.

The following is an example program for outputting alarm from $\overline{\text{ALARM}}$ -pin at noon (PM12:00) every day.

```
    LD      (PAGER), 09H        ;   Alarm disable, setting PAGE1
    LD      (RESTR), D0H        ;   Alarm initialize
    LD      (DAYR), 01H         ;   W0
    LD      (DATAR),01H             1 day
    LD      (HOURR), 12H        ;   Setting 12 o'clock
    LD      (MINR), 00H         ;   Setting 00 min
                                ;   Set up time 31 μs (Note)
    LD      (PAGER), 0CH        ;   Alarm enable
(   LD      (PAGER), 8CH        ;   Interrupt enable )
```

When CPU is operated by high frequency oscillation, it may take a maximum of one clock at 32 kHz (about 30us) for the time register setting to become valid. In the above example, it is necessary to set 31us of set up time between setting the time register and enabling the alarm register.

Note: This set up time is unnecessary when you use only internal interruption.

(2) When output clock of 1Hz

RTC outputs clock of 1Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “0”, <DIS16HZ>= “1”. And RTC generates INTRC interrupt by falling edge of the clock.

(3) When output clock of 16Hz

RTC outputs clock of 16Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “1”, <DIS16HZ>= “0”. And RTC generates INTRC interrupt by falling edge of the clock.

3.22 Melody / Alarm generator (MLD)

TMP92CZ26A contains melody function and alarm function, both of which are output from the MLDALM pin. Five kind of fixed cycles interrupt is generate by using 15bit counter, which is used for alarm generator.

Features are as follows.

1) Melody generator

The Melody function generates signals of any frequency (4Hz- 5461Hz) based on low-speed clock (32.768 KHz) and outputs the signals from the MLDALM pin.

By connecting a loud speaker outside, Melody tone can easily sound.

2) Alarm generator

The Alarm function generates eight kinds of alarm waveform having a modulation frequency (4096Hz) determined by the low-speed clock (32.768 KHz). And this waveform is able to invert by setting a value to a register.

By connecting a loud speaker outside, Alarm tone can easily sound.

Five kinds of fixed cycles (1Hz, 2Hz, 64Hz, 512Hz, 8192Hz) INTERRUPT are generated by using a counter that is used for alarm generator.

This section is constituted as follows.

3.22.1 Block Diagram

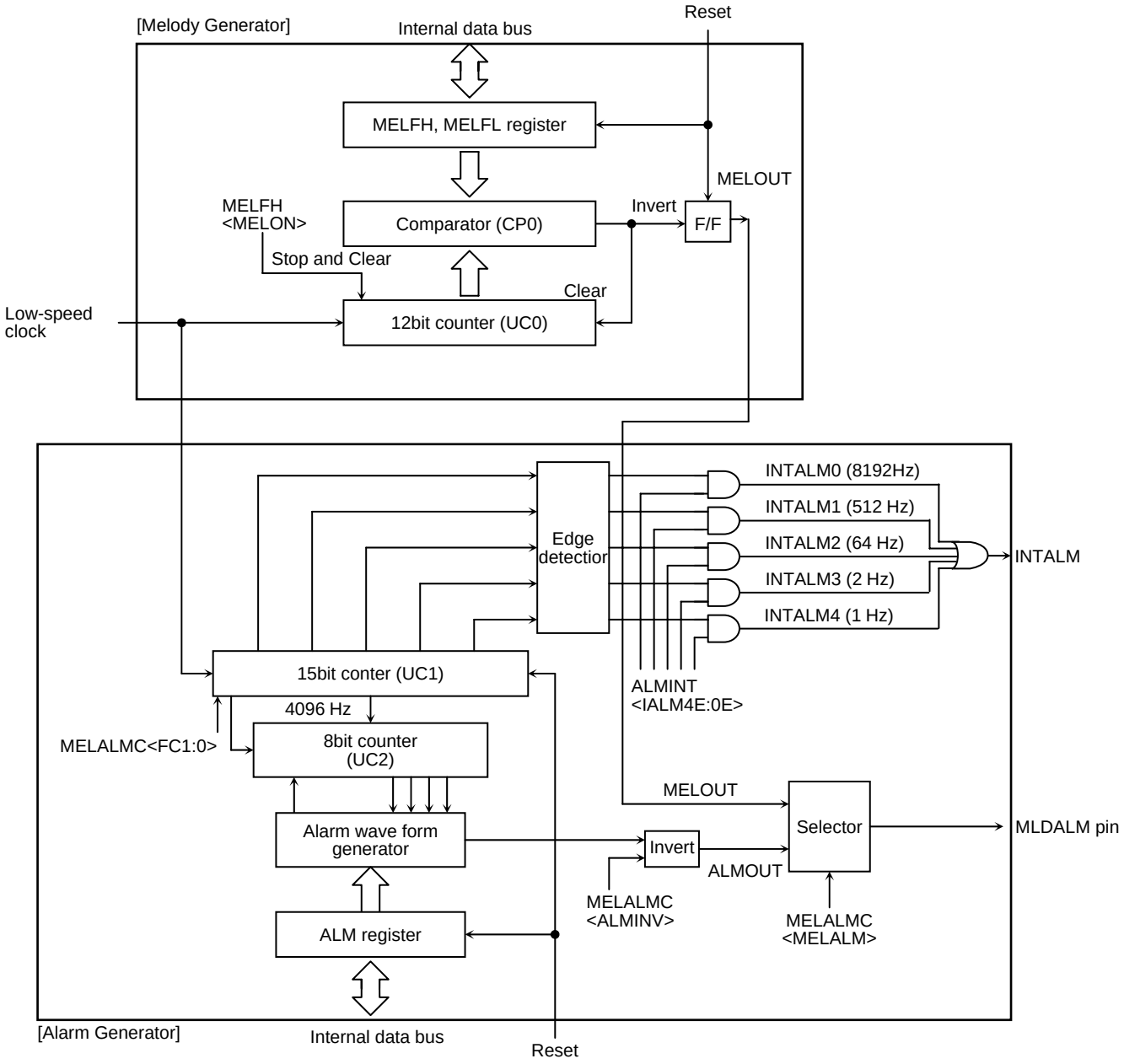


Figure 3.22.1MLD Block Diagram

3.22.2 Control registers

ALM register

ALM (1330H)

	7	6	5	4	3	2	1	0
bit Symbol	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Setting alarm pattern							

MELALMC register

MELALMC (1331H)

	7	6	5	4	3	2	1	0
bit Symbol	FC1	FC0	ALMINV	–	–	–	–	MELALM
Read/Write	R/W		R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	Free-run counter control 00: Hold 01: Restart 10: Clear 11: Clear & Start		Alarm Waveform invert 1:INVERT	Always write “0”				Select Output Waveform 0: Alarm 1: Melody

Note1: MELALMC<FC1> is read always “0”.

Note2: When setting MELALMC register except <FC1:0> during the free-run counter is running, <FC1:0> is kept “01”.

MELFL register

MELFL (1332H)

	7	6	5	4	3	2	1	0
bit Symbol	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Setting melody frequency (lower 8bit)							

MELFH register

MELFH (1333H)

	7	6	5	4	3	2	1	0
bit Symbol	MELON				ML11	ML10	ML9	ML8
Read/Write	R/W				R/W			
After reset	0				0	0	0	0
Function	Control melody counter 0: Stop & Clear 1: Start				Setting melody frequency(upper 4bit)			

ALMINT register

ALMINT (1334H)

	7	6	5	4	3	2	1	0
bit Symbol			–	IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
Read/Write			R/W	R/W				
After reset			0	0	0	0	0	0
Function			Always write “0”	1:INTALM4 (1Hz) enable	1:INTALM3 (2Hz) enable	1:INTALM2 (64Hz) enable	1:INTALM1 (512Hz) enable	1:INTALM0 (8192Hz) enable

Note: INTALM0 to INTALM4 prohibit that set to enable at same time. If setting to enable, set only 1.

3.22.3 Operational Description

3.22.3.1 Melody generator

The Melody function generates signals of any frequency (4Hz-5461Hz) based on low-speed clock (32.768KHz) and outputs the signals from the MLDALM pin.

By connecting a loud speaker outside, Melody tone can easily sound.

(Operation)

At first, MELALMC<MELALM> have to be set as "1" in order to select melody waveform as output waveform from MLDALM. Then melody output frequency has to be set to 12-bit register MELFH, MELFL.

Followings are setting example and calculation of melody output frequency.

(Formula for calculating of melody waveform frequency)

@fs = 32.768 [kHz]

Melody output waveform $f_{MLD}[Hz] = 32768/ (2 \times N + 4)$

Setting value for melody $N = (16384/ f_{MLD}) - 2$

(Note: N = 1~4095 (001H~FFFH), 0 is not acceptable)

(Example program)

In case of outputting "A" musical scale (440Hz)

```
LD    (MELALMC), --XXXXX1B    ;  Select melody waveform
LD    (MELFL), 23H            ;  N = 16384/440 – 2 = 35.2 = 023H
LD    (MELFH), 80H            ;  Start to generate waveform
```

(Refer: Basic musical scale setting table)

Scale	Frequency [Hz]	Register Value: N
C	264	03CH
D	297	035H
E	330	030H
F	352	02DH
G	396	027H
A	440	023H
B	495	01FH
C	528	01DH

3.22.3.2 Alarm generator

The Alarm function generates eight kinds of alarm waveform having a modulation frequency 4096Hz determined by the low-speed clock (32.768 KHz). And this waveform is reversible by setting a value to a register.

By connecting a loud speaker outside, Alarm tone can easily sound.

Five kind of fixed cycles (1Hz, 2Hz, 64Hz, 512Hz, 8192Hz) interrupt be generate by using a counter which is used for alarm generator.

(Operation)

At first, MELALMC<MELALM> have to be set as "0" in order to select alarm waveform as output waveform from MLDALM. Then "10" be set on MELALMC<FC1:0> register, and clear internal counter. Finally alarm pattern has to be set on 8-bit register of ALM. If it is inverted output-data, set <ALMINV> as invert.

Followings are example program, setting value of alarm pattern and waveform of each setting value.

(Setting value of alarm pattern)

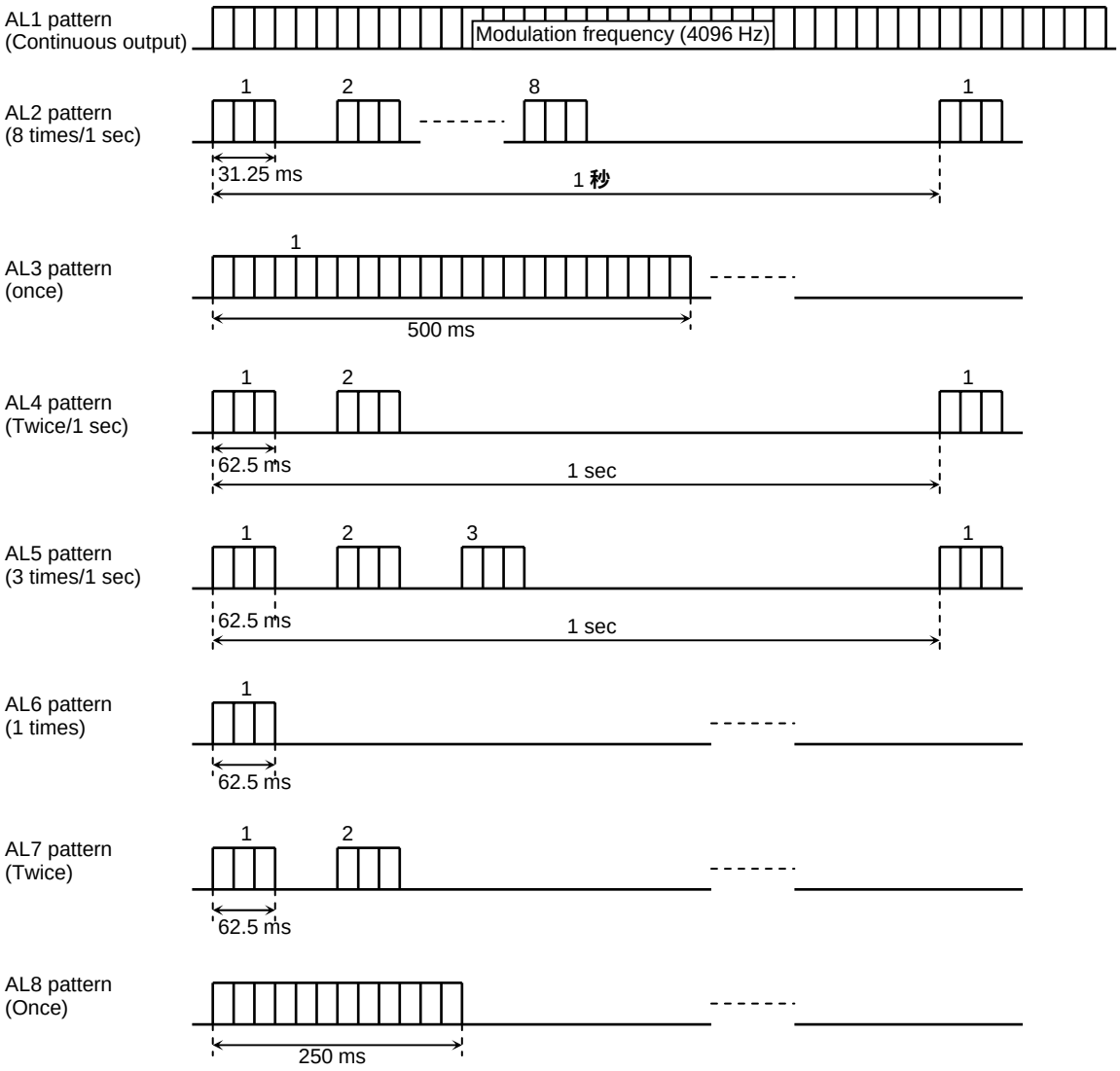
Setting value for ALM register	Alarm waveform
00H	"0" fixed
01H	AL1 pattern
02H	AL2 pattern
04H	AL3 pattern
08H	AL4 pattern
10H	AL5 pattern
20H	AL6pattern
40H	AL7 pattern
80H	AL8 pattern
Other	Undefined (Do not set)

(Example program)

In case of outputting AL2 pattern (31.25ms/8 times/1sec)

```
LD      (MELALMC), C0H      ;  Set output alarm waveform
                            ;  Free-run counter start
LD      (ALM), 02H          ;  Set AL2 pattern, start
```

Example: Waveform of alarm pattern for each setting value: not invert)



3.23 Analog-Digital Converter (ADC)

This LSI has a 6-channel, multiplexed-input, 10-bit successive-approximation Analog-Digital converter (ADC).

Figure 3.23.1 shows a block diagram of the AD converter.

The 6-analog input channels (AN0-AN5) can be used as general-purpose inputs.

Note1: Ensure that the AD converter has halted before executing HALT instruction to place the TMP92CZ26A in IDLE2, IDLE1, STOP or PCM mode to reduce power consumption current. Otherwise, the TMP92CZ26A might go into a standby mode while the internal analog comparator is still enable state.

Note2: The power consumption current is reduced by setting ADMOD1<DACON> to "0" in the ADC has been stopped.

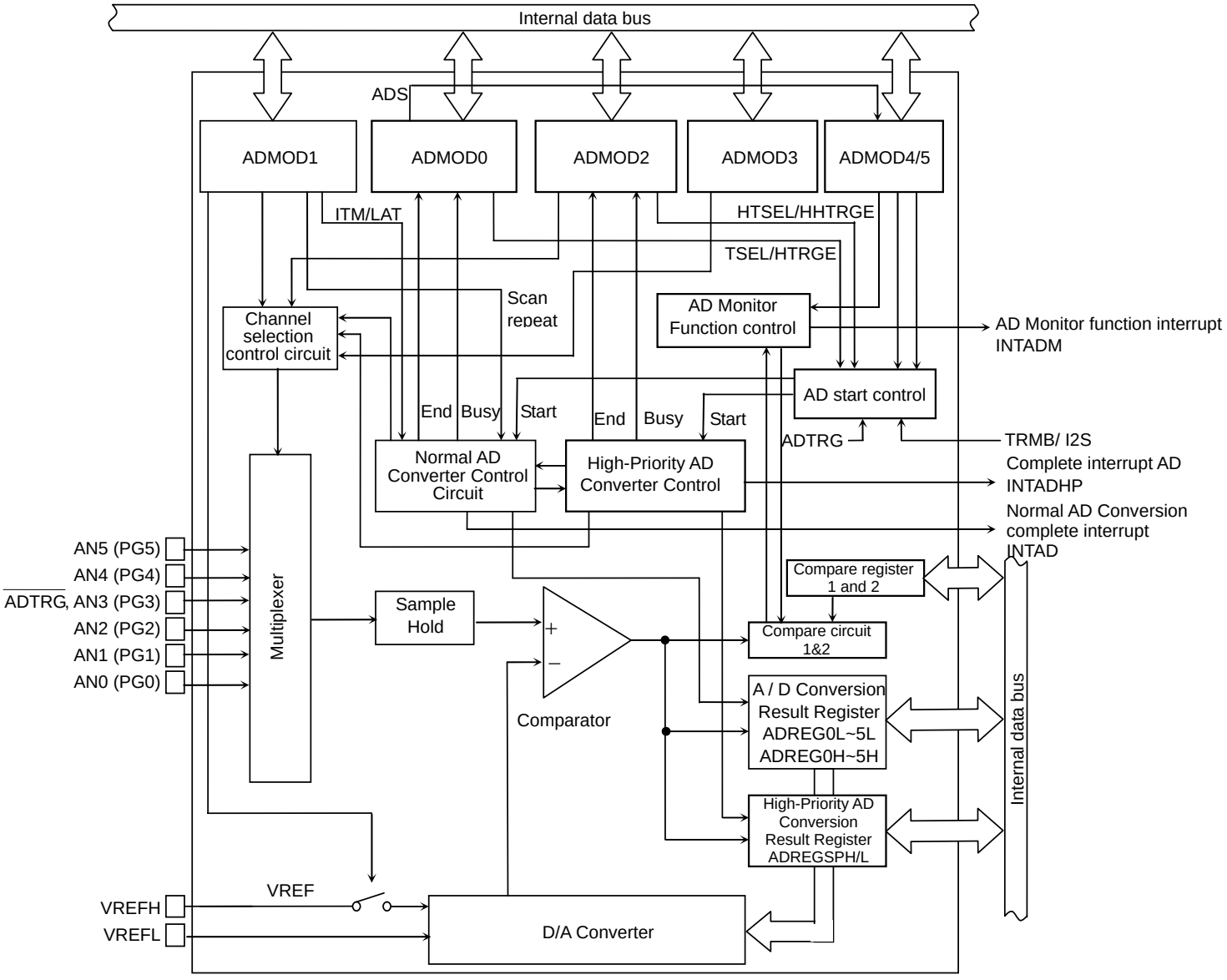


Figure 3.23.1 ADC Block Diagram

3.23.1 Control register

The AD converter has 6-mode control registers (ADMOD0, ADMOD1, ADMOD2, ADMOD3, ADMOD4 and ADMOD5) and 6-conversion result high/low register pairs (ADREG0H/L ~ ADREG5H/L). The results of high-priority AD conversion are stored in the ADREGSPH/L.

Figure 3.23.2 to Figure 3.23.11 show the registers available in the AD converter.

AD Mode Control Register 0 (Normal conversion control)

ADMOD0 (12B8H)

	7	6	5	4	3	2	1	0
bit Symbol	EOS	BUSY		I2AD	ADS	HTRGE	TSEL1	TSEL0
Read/Write	R	R		R/W				
After reset	0	0		0	0	0	0	0
Function	Normal AD conversion end flag 0:During conversion sequence or before starting 1:Complete conversion sequence	Normal AD conversion BUSY Flag 0:Stop conversion 1:During conversion		AD conversion when IDLE2 mode 0: Stop 1: Operate	Start Normal AD conversion 0: Don't Care 1:Start AD conversion Always read as"0".	Normal AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB00 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: Reserved	

Figure 3.23.2 AD Conversion Registers

AD Mode Control Register 1 (Normal conversion control)

ADMOD1 (12B9H)

	7	6	5	4	3	2	1	0
bit Symbol	DACON	ADCH2	ADCH1	ADCH0	LAT	ITM	REPEAT	SCAN
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	DAC and VREF application control	Analog input channel select			Latency 0: No Wait 1:Start after reading conversion result store Register of last channel	Interrupt specification when conversion channel fixed repeat mode	Repeat mode specification 0: Single conversion 1: Repeat conversion	Scan mode specification 0: Channel fixed mode 1: Channel scan mode

Specify AD conversion interrupt for Channel Fixed Repeat Conversion mode

	Channel Fixed Repeat Conversion Mode <SCAN> = "0", <REPEAT> = "1"
0	Generates interrupt every conversion
1	Generated interrupt every fourth conversion

Next SCAN start timing control for the channel scan repeat mode

Channel Scan Repeat mode (<SCAN> = 1, <REPEAT> = 1)	
0	No Wait
1	Start after read last of conversion result store Register

Analog input channel select

<SCAN> / <ADCH2:0>	0: Channel fixed	1: Channel scanned
000	AN0	AN0
001	AN1	AN0→AN1
010	AN2	AN0→AN1→AN2
011	AN3(note)	AN0→AN1→AN2→AN3 (note)
100	AN4	AN0→AN1→AN2→AN3→AN4 (note)
101	AN5	AN0→AN1→AN2→AN3→AN4→AN5 (note)
110	Reserved	
111	Reserved	

Note: When using PG3 pin as $\overline{\text{ADTRG}}$, it cannot be set.

DAC & VREF application control

0	DAC & VREF off (Set before into STOP mode)
1	DAC & VREF on (Set to "1" before starting conversion)

Figure 3.23.3 AD Converter Related Register

AD Mode Control Register 2 (High-priority conversion control)

ADMOD2 (12BAH)

	7	6	5	4	3	2	1	0
bit Symbol	HEOS	HBUSY			HADS	HHTRGE	HTSEL1	HTSEL0
Read/Write	R	R			R/W			
After reset	0	0			0	0	0	0
Function	High-priority AD conversion sequence FLAG 0: During conversion sequence or before starting 1: Complete conversion sequence	High-priority AD conversion BUSY Flag 0:Stop conversion 1:During conversion			Start High-priority AD conversion 0: Don't Care 1: Start AD conversion Always read as"0".	High-priority AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB10 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: I2S Sampling Counter Output	

AD Mode Control Register 3 (High-priority conversion control)

ADMOD3 (12BBH)

	7	6	5	4	3	2	1	0
bit Symbol	–	HADCH2	HADCH1	HADCH0				–
Read/Write	R/W	R/W						R/W
After reset	0	0	0	0				0
Function	Always write "0".	High-priority analog input channel select						Always write "0".

Analog input channel select

<HADCH2:0>	Analog input channel when High-priority conversion
000	AN0
001	AN1
010	AN2
011	AN3(note)
100	AN4
101	AN5
110	Reserved
111	Reserved

Note: When using PG3 pin as $\overline{\text{ADTRG}}$, it cannot be set.

Figure 3.23.4 AD Conversion Registers

AD Mode Control Register 4 (AD Monitor function control)

ADMOD4 (12BCH)

	7	6	5	4	3	2	1	0
bit Symbol	CMEN1	CMEN0	CMP1C	CMP0C	IRQEN1	IRQEN0	CMPINT1	CMPINT0
Read/Write	R/W	R/W	R/W				R	R
After reset	0	0	0	0	0	0	0	0
Function	AD Monitor function1 0: Disable 1: Enable	AD Monitor function0 0: Disable 1: Enable	Generation condition of AD monitor function interrupt 1 0: less than 1: Greater than or Equal	Generation condition of AD monitor function interrupt 0 0: less than 1: Greater than or Equal	AD monitor function interrupt 1 0: Disable 1: Enable (Note)	AD monitor function interrupt 0 0: Disable 1: Enable (Note)	Status of AD monitor function interrupt 1 0: No generation 1: Generation	Status of AD monitor function interrupt 0 0: No generation 1: Generation

Note: When AD monitor function interrupts generate, it is cleared automatically and it is set to disable condition.

AD Mode Control Register 5 (AD Monitor function control)

ADMOD5 (12BDH)

	7	6	5	4	3	2	1	0
bit Symbol		CMCH2	CM1CH1	CM1CH0		CM0CH2	CM0CH1	CM0CH0
Read/Write		R/W				R/W		
After reset		0	0	0		0	0	0
Function		Select analog channel for AD monitor function 1 000: AIN0 100: AN4 001: AIN1 101: AN5 010: AIN2 110: Reserved 011: AN3 111: Reserved				Select analog channel for AD monitor function 0 000: AIN0 100: AN4 001: AIN1 101: AN5 010: AIN2 110: Reserved 011: AN3 111: Reserved		

Note1: When converting AD in hard ware trigger by setting <HHTRGE> and <HTRGE>to “1”, set PGFC<PG3F> to “1” (as ADTRG) in case of external TRG before enabling it. When using an INTTBx0 of 16-bit timer, first set the <TSEL1:0> or <HTSEL1:0> bit to “00” when the timer is not operating. Then, set the <HHTRGE> and <HTRGE> to “1” and enable trigger operation. Finally, operate the timer so that AD conversion will be initiated at constant intervals.

Note 2: When disabling an external trigger ($\overline{\text{ADTRG}}$) for AD conversion, first clear the <HHTRGE> or <HTRGE> bit to “0”, and clear the PGFC<PG3F> to “0”, thus configuring port G as a general-purpose port.

Note 3: When starting AD by using external trigger (ADTRG), it can be started after enabling (<HHTRGE> = “1” or <HTRGE> = “1”) and 3 clock at f_{SYS} was executed. AD is not started when before that time.

Note 4: When chaging compare register value of AD Monitor function, change it after setting AD Monitor function to disable(ADMOD4<CMEN1:0>=”0”).

Figure 3.23.5 AD Conversion Registers

AD Conversion Result Register 0 Low

ADREG0L (12A0H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR01	ADR00					OVR0	ADR0RF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of AN0 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 0 High

ADREG0H (12A1H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN0 AD conversion result							

AD Conversion Result Register 1 Low

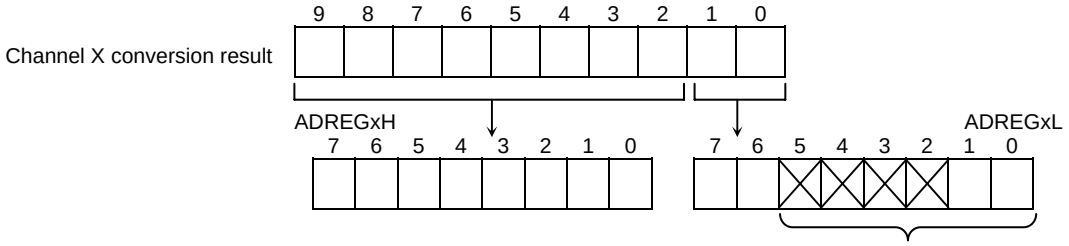
ADREG1L (12A2H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR11	ADR10					OVR1	ADR1RF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of AN1 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 1 High

ADREG1H (12A3H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN1 AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

Figure 3.23.6 AD Conversion Registers

AD Conversion Result Register 2 Low

ADREG2L (12A4H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR21	ADR20					OVR2	ADR2RF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of AN2 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 1 High

ADREG2H (12A5H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN2 AD conversion result							

AD Conversion Result Register 3 Low

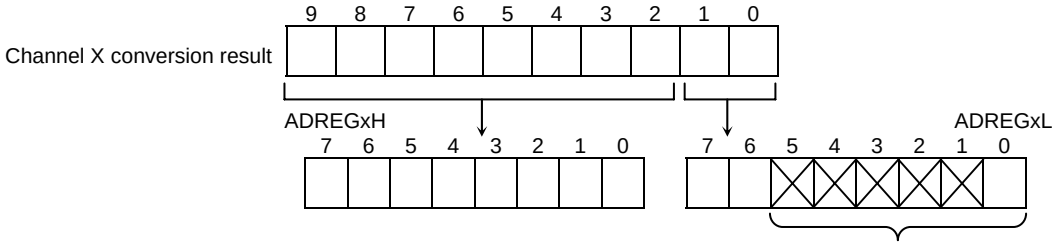
ADREG3L (12A6H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR31	ADR30					OVR3	ADR3RF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of AN3 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 3 High

ADREG3H (12A7H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN3 AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

Figure 3.23.7 AD Conversion Registers

AD Conversion Result Register 4 Low

ADREG4L (12A8H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR41	ADR40					OVR4	ADR4RF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of AN4 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 4 High

ADREG4H (12A9H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN4 AD conversion result							

AD Conversion Result Register 5 Low

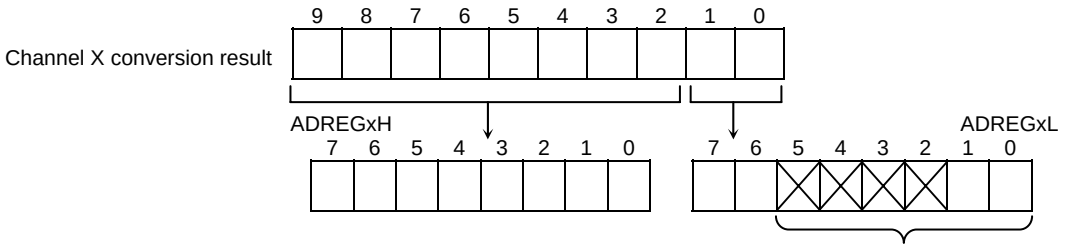
ADREG5L (12AAH)

	7	6	5	4	3	2	1	0
bit Symbol	ADR51	ADR50					OVR5	ADR5RF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of AN5 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 5 High

ADREG5H (12ABH)

	7	6	5	4	3	2	1	0
bit Symbol	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN5 AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

Figure 3.23.8 AD Conversion Registers

High-priority AD Conversion Result Register SP Low

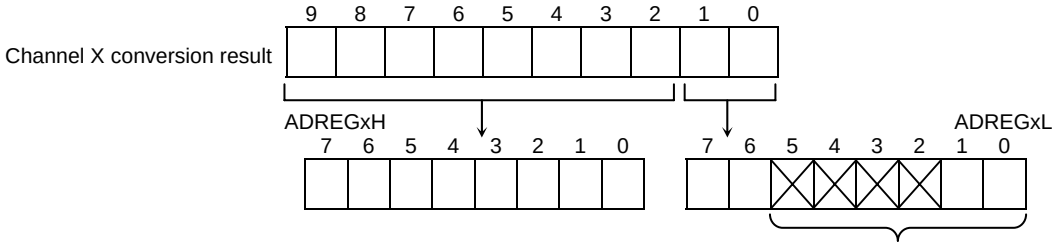
ADREGSPL (12B0H)

	7	6	5	4	3	2	1	0
bit Symbol	ADRSP1	ADRSP0					OVSRP	ADRSPRF
Read/Write	R						R	R
After reset	0	0					0	0
Function	Store Lower 2 bits of an AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

High-priority AD Conversion Result Register SP High

ADREGSPH (12B1H)

	7	6	5	4	3	2	1	0
bit Symbol	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of an AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

Figure 3.23.9 AD Conversion Registers

AD Conversion Result Compare Criterion Register 0 Low

ADCM0REGL (12B4H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR21	ADR20						
Read/Write	R/W							
After reset	0	0						
Function	Store Lower 2 bits of an AD conversion result compare criterion							

AD Conversion Result Compare Criterion Register 0 High

ADCM0REGH (12B5H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of an AD conversion result compare criterion							

AD Conversion Result Compare Criterion Register 1 Low

ADCM1REGL (12B6H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR21	ADR20						
Read/Write	R/W							
After reset	0	0						
Function	Store Lower 2 bits of an AD conversion result compare criterion							

AD Conversion Result Compare Criterion Register 1 High

ADCM1REGH (12B7H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of an AD conversion result compare criterion							

Note: Disable the AD monitor function (ADMOD4<CMEN> = “0”) before attempting to set or modify the value of these registers.

Figure 3.23.10 AD Conversion Registers

AD Conversion Clock Setting Register

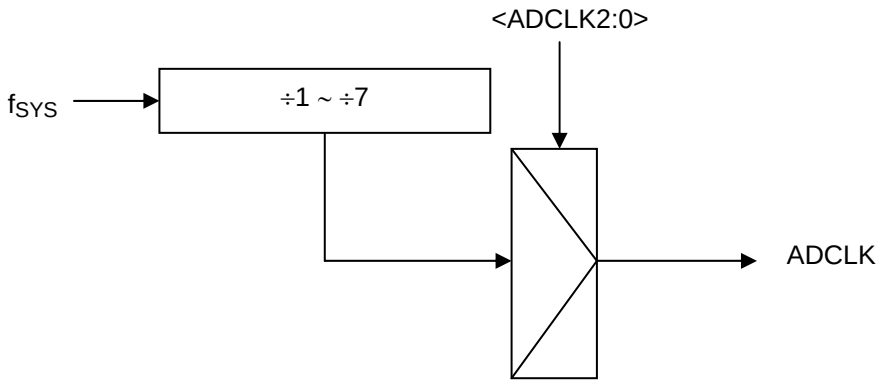
ADCCLK (12BFH)

	7	6	5	4	3	2	1	0
bit Symbol					–	ADCLK2	ADCLK1	ADCLK0
Read/Write					R/W	R/W	R/W	R/W
After reset					0	0	0	0
Function					Always write “0”	Select clock for AD conversion 000 : Reserved 100 : $f_{IO}/4$ 001 : $f_{IO}/1$ 101 : $f_{IO}/5$ 010 : $f_{IO}/2$ 110 : $f_{IO}/6$ 011 : $f_{IO}/3$ 111 : $f_{IO}/7$		

Note1: AD conversion is executed at the clock frequency selected in the above register. To assure conversion accuracy, however, the conversion clock frequency must not exceed 12MHz MHz.

Note2: Don ‘t change the clock frequency while AD conversion is in progress.

Figure 3.23.11 AD Conversion Registers



$f_{IO}(f_{SYS}/2)$	<ADCLK2:0>	ADCLK	AD conversion speed
40MHz	100($f_{IO}/4$)	10.0MHZ	12 μsec
	101($f_{IO}/5$)	8MHZ	15 μsec
30MHz	011($f_{IO}/3$)	10.0MHZ	12 μsec
	100($f_{IO}/4$)	7.5MHZ	16 μsec

AD conversion speed can be calculated by following.

Conversion speed = 120 × (1/ADCLK)

3.23.2 Operation

3.23.2.1 Analog Reference Voltages

The VREFH and VREFL pins provide the analog reference voltages for the ADC.

3.23.2.2 Analog Input Channel(s) selection

The Analog input channels used for AD conversion are selected as follows:

(1) Normal AD conversion

- Analog Input Channel Fixed mode (ADMOD1<SCAN> = “0”)

Setting ADMOD1<ADCH2:0> selects one of the input pins AN0 to AN5 as the input channel.

- Analog Input Channel Scan Mode (ADMOD1<SCAN> = “1”)

Setting ADMOD1<ADCH2:0> selects one of the six scan modes.

(2) High-priority AD conversion

Setting ADMOD3<HADCH2:0> selects one of the eight input pins AN0 ~ AN5.

On a Reset, ADMOD1<SCAN> is set to “0”, and ADMOD1<ADCH2:0> is initialized to “000”. Thus pin AN0 is selected as the fixed input channel. Pins that are not used as analog input channels can be used as standard input port pins.

If a high-priority AD conversion is triggered while a normal AD conversion is in progress, the normal AD conversion sequence is suspended after converting data for the current channel, to perform a high-priority AD conversion. After a high-priority AD conversion is performed, the normal AD conversion sequence is resumed with that channel.

3.23.2.3 Starting an AD Conversion

The ADC supports two types of AD conversion: normal AD conversion and high-priority AD conversion. The ADC initiates a normal AD conversion by software when the ADMOD0<ADS> is set to "1". It initiates a high-priority AD conversion by software when the ADMOD2<HADS> is set to "1". For a normal AD conversion, ADMOD1<REPEAT, SCAN> select one of four conversion modes. For a high-priority AD conversion, the ADC only supports Fixed-Channel Single Conversion mode.

The ADMOD0<TSEL1:0> and ADMOD2<HTSEL1:0> enable a hardware trigger for a normal and high-priority AD conversion, respectively. When these bits are set to "10", a normal or high-priority AD conversion is triggered by a falling edge applied to $\overline{\text{ADTRG}}$ pin. When ADMOD0<TSEL1:0> is set to "00", a normal AD conversion is triggered by INTTB00 of 16-Bit Timer interrupt. When ADMOD2<HTSEL1:0> is set to "00", a high-priority AD conversion is triggered by INTTB10 of 16-Bit Timer interrupt. If this bit is "11", it is triggered by I2S sampling block. Even when a hardware trigger is enabled, software starting can be used.

When a normal AD conversion starts, the Busy flag (ADMOD0<BUSY>) is set to "1". When a high-priority AD conversion starts, the high-priority AD conversion busy flag (ADMOD2<HBUSY>) is set to "1". During a normal AD conversion, if a high-priority AD conversion start, ADMOD0<BUSY> holds "1".

When an AD conversion complete, ADMOD0<EOS> and ADMOD2<HEOS> is set to "1". These flags are cleared to "0" by reading these flags only.

During a normal AD conversion, writing a "1" to ADMOD0<ADS> causes the ADC to abort any ongoing conversion immediately, and restart.

During a normal AD conversion, if normal AD conversion starting is enabled by hard ware trigger, normal AD conversion is restarted when start condition from hard ware trigger is satisfied. When restart is set, normal AD conversion is aborted immediately.

During a normal AD conversion, if a high-priority AD conversion starts(writing a "1" to ADMOD2<HADS> or a hard ware trigger occurs), the ADC aborts any ongoing conversion immediately, and then start a high-priority AD conversion for the channel specified by ADMOD3<HADCH2:0>. Upon the completion of the high-priority conversion, the ADC stores the conversion result to ADREGSPH/L, and then resumes the suspended normal conversion with that channel.

Note: It cannot overlap with three or more AD conversions.

Prohibition example 1 : In FIRST normal AD conversion

→ (Before finished FIRST normal AD conversion) Started SECOND normal AD conversion

→ (Before finished SECOND normal AD conversion) Started THIRD normal AD conversion

Prohibition example 2 : In FIRST normal AD conversion

→ (Before finished FIRST normal AD conversion) Started SECOND normal AD conversion

→ (Before finished SECOND normal AD conversion) Started THIRD high-priority AD conversion

3.23.2.4 AD Conversion Modes and AD Conversion-End Interrupts

The ADC supports the following four conversion modes. For a normal AD conversion, ADMOD0<1:0> select one of the four conversion modes. For a high-priority AD conversion, the ADC only supports Channel Fixed Single Conversion mode.

a. Channel Fixed Single Conversion mode
b. Channel Scan Single Conversion mode
c. Channel Fixed Repeat Conversion mode
d. Channel Scan Repeat Conversion mode

(1) Normal AD conversion

ADMOD0<REPEAT, SCAN> select the conversion mode. Once a conversion is started, the ADMOD0<BUSY> is set to "1". The ADC generates the AD Conversion End interrupt (INTAD) and sets the ADMOD0<EOS> to "1" at the end of the specified conversion process.

a. Channel Fixed Single Conversion mode

This mode is selected by programming ADMOD0<REPEAT, SCAN> to "00".

In this mode, the ADC performs a single conversion on a single selected channel. When a conversion is completed, the ADC sets the ADMOD0<EOS>, and generates the INTAD interrupt. ADMOD0<EOS> is cleared to "0" when it is read.

b. Channel Scan Single Conversion mode

This mode is selected by programming ADMOD0<REPET, SCAN> to "01". In this mode, the ADC performs a single conversion on each of a selected group of channels. When a single conversion sequence is completed, ADMOD0<EOS> is set to "1", and generates the INTAD interrupt. ADMOD0<EOS> is cleared to "0" by reading this bit only.

c. Channel Fixed Repeat Conversion mode

This mode is selected by programming ADMOD0<REPET, SCAN> to "10". In this mode, the ADC repeatedly converts a single selected channel. When a conversion process is completed, ADMOD0<EOS> is set to "1".

ADMOD1<ITM> control INTAD interrupts generation in this mode. The timing when ADMOD0<EOS> is set also depends on the ADMOD1<ITM>. The EOCF bit is cleared when it is read. ADMOD0<EOS> is cleared to "0" by reading this bit only.

If ADMOD1<ITM> is set to "0", the ADC generates an interrupt after each conversion. The results of conversion are always stored in the ADREGxH/L register. The ADMOD0<EOS> is set to "1" when the ADC stores the results to the ADREGxH/L.

If ADMOD1<ITM> is set to “1”, the ADC generates an interrupt after every four conversions. The results of conversions are sequentially stored in the ADREG0H/L to ADREG3H/L registers, in that order. The ADMOD0<EOS> is set to “1” when the ADC stores the results in the ADREG3H/L. The next conversion results are again stored in the ADREG0, and so on. The ADMOD0<EOS> is cleared to “0” by reading this bit only.

d. Channel Scan Repeat Conversion mode

This mode is selected by programming ADMOD0 <REPEAT, SCAN> to “11”. In this mode, the ADC repeatedly converts the selected group of channels. When a single conversion sequence is completed, the ADC sets ADMOD0<EOS> to “1”, and generates the INTAD interrupt. ADMOD0<EOS> is cleared to “0” by reading this bit only.

In continuous conversion modes (3) and 4)), clearing the ADMOD1<REPEAT> stops the conversion sequence after the ongoing scan conversion process is completed.

Shift to a standby mode (IDLE2 Mode with ADMOD0<I2AD> = “0”, IDLE1 Mode or STOP Mode) immediately stops operation of the AD converter even if AD conversion is still in progress. Therefore, ADC may consume current even if operation is stopped, depending on stop condition of ADC that switches to standby mode. For avoiding this problem, Stop ADC before switching to standby mode.

(2) High-priority AD conversion

For a high-priority AD conversion, the ADC only supports Channel Fixed Single Conversion mode, regardless of the settings of ADMOD1<REPEAT, SCAN>.

When a conversion start condition is satisfied, the ADC performs a single conversion on a single selected channel, which is specified with the ADMOD3<HADCH2:0>. When a conversion is completed, the ADC sets ADMOD2<HEOS>to “1”. HEOS Flag is cleared to “0” by reading this bit only.

Interrupt Generation Timing and Flag Setting in Each AD Conversion Mode

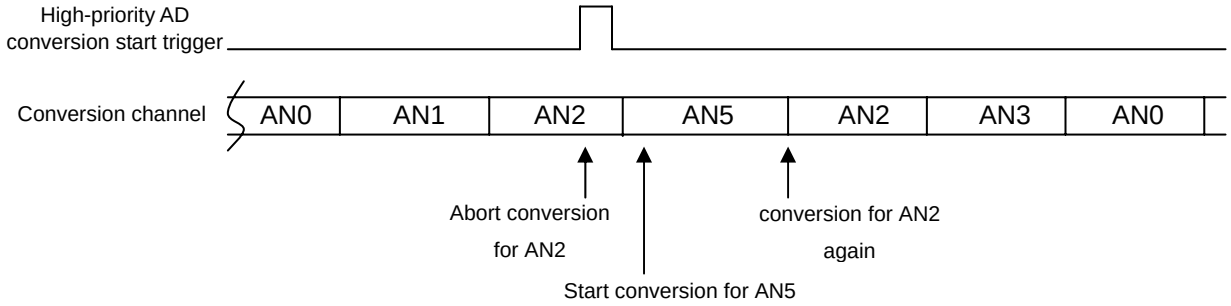
Conversion mode	Interrupt Generation Timing	EOS set timing (Note)	ADMOD1		
			ITM	REPEAT	SCAN
Channel Fixed Single Conversion Mode	After a conversion	After a conversion	–	0	0
Channel Fixed Repeat Conversion Mode	After every conversion	After every conversion	0	1	0
	After every four conversions	After every four conversions	1		
Channel Scan Single Conversion Mode	After a scan conversion sequence	After a scan conversion sequence	–	0	1
Channel Repeat Single Conversion Mode	After each scan conversion sequence	After each scan conversion sequence	–	1	1

Note: EOS is cleared to “0” by reading this bit only.

3.23.2.5 High-Priority Conversion Mode

The ADC can perform a high-priority AD conversion while it is performing a normal AD conversion sequence. A high-priority AD conversion can be started at software by setting the ADMOD2<HADS> to "1". It is also triggered by a hardware trigger if so enabled using ADMOD2<HTSEL1:0>. If a high-priority AD conversion is triggered during a normal AD conversion, the ADC aborts any ongoing conversion immediately, and then begins a single high-priority AD conversion for the channel specified with the ADMOD3<HADC2:0>. Upon the completion of the high-priority AD conversion, the ADC stores the results of the conversion in the ADREGSPH/L, generates the high-priority AD conversion interrupt (INTADHP), and then resumes the suspended normal conversion with that channel. While a high-priority conversion is being performed, a trigger for another high-priority conversion is ignored.

Example: In the case of a high-priority AD conversion for AN5 (ADMOD3<HADCH2:0>="101") is started durring a normal AD conversion in channel scan repeat mode for AN0 to AN3 (ADMOD1<REPEAT,SCAN> = "11"、ADMOD1<ADCH2:0> = "011")



3.23.2.6 AD Monitor Function

When ADMOD4<CMEN1:0> is set to "1", the AD monitor function is enabled. This function generates an interrupt depending on condition of IRQEN1:0, when the finished AD conversion of the channel which specified with the ADMOD5 register, if the value of the AD conversion result register pair is greater or less (specified with CMP1C:0C) than the value of the compare criterion register 0/1(ADCMxREGH/L). The ADC performs this comparison each times it stores results to the specified AD conversion result register and generate interrupt (INTADM) when condition is satisfied. The conversion result register used for the AD monitor function is usually not read in the program, so mind that its overrun flag <OVRn> and conversion result store flag <ADRnRF> are always set.

If these are assigned to different channel, 2-analog channel can monitor "less" or "grater". And if these are assigned same analog channel, the watch that sets the range of the voltage is possible.

3.23.2.7 AD Conversion Time

AD conversion of one time is 120 clocks that include sampling clock. The AD conversion clock can be selected from 1/1 to 1/7 of f_{IO} by ADCLK<ADCLK2:0>. To assure conversion accuracy, the AD conversion clock frequency need to select 12 MHz and under, i.e., AD conversion time need to select 10 μs and over.

3.23.2.8 Storing and Reading the AD Conversion Result

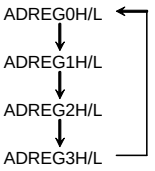
Conversion results are stored into AD conversion result high/low register (ADREG0H/L to ADREG5H/L).

In Channel Fixed Repeat Conversion mode, conversion results are stored into the ADREG0H/L to ADREG3H/L sequentially.

In other modes, the AD conversion result of channel AN0, AN1, AN2, AN3, AN4, and AN5 is stored in ADREG0H/L, ADREG1H/L, ADREG2H/L, ADREG3H/L, ADREG4H/L, and ADREG5H/L respectively.

Table 3.23.1 shows the relationships between the analog input channels and the AD conversion result registers.

Table 3.23.1 Relationships between Analog Input Channels and AD Conversion Result Registers

Analog Input Channel (Port G)	AD Conversion Result Registers	
	Conversion Modes other than at right	Channel Fixed Repeat Conversion Mode (every fourth conversion)
AN0	ADREG0H/L	
AN1	ADREG1H/L	
AN2	ADREG2H/L	
AN3	ADREG3H/L	
AN4	ADREG4H/L	
AN5	ADREG5H/L	

Note: For detect a overrun error thoroughly, read the AD conversion result register high at first and read the AD conversion result register low at second. If OVRn="0" and ADRnRF="1", a correct conversion result was obtained.

3.23.2.9 Data Polling

When the results of AD conversion are processed by means of data polling without using interrupts, ADMOD0<EOS> should be polled. After confirming ADMOD0 <EOS>="1", read the AD conversion result register.

Setting example:

1. Convert the analog input voltage on the AN3 pin and write the result to memory address 2800H using the AD interrupt(INTAD) processing routine.

Main routine

		7	6	5	4	3	2	1	0	
INTEAD	←	1	1	0	0	–	–	–	–	Enable INTAD and set it to interrupt level 4.
ADMOD1	←	1	1	0	0	0	0	1	1	Set pin AN3 to be the analog input channel.
ADMOD0	←	X	X	0	0	0	0	0	1	Start conversion in channel fixed single conversion mode.

Interrupt routine processing example

WA	←	ADREG3	Read value of ADREG3L and ADREG3H into 16-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2800H)	←	WA	Write contents of WA to memory address 2800H.

2. This example repeatedly converts the analog input voltages on the three pins AN0, AN1 and AN2, using channel scan repeat conversion mode.

INTEAD	←	1	0	0	0	–	–	–	–	Disable INTAD.
ADMOD1	←	1	1	0	0	0	0	1	0	Set pins AN0 to AN2 to be the analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Start conversion in channel scan repeat conversion mode.

3. Convert the analog input voltage on the AN2 pin as a high-priority AD conversion, and write the result to memory address 2A00H using the High-priority AD interrupt(INTADHP) processing routine.

Main routine

INTEAD	←	1	1	0	1	–	–	–	–	Enable INTADHP and set it to interrupt level 6.
ADMOD1	←	1	0	0	0	0	0	0	0	DAC On.
ADMOD3	←	0	0	1	0	0	0	0	0	Set pin AN2 to be the analog input channel.
ADMOD2	←	0	0	0	0	1	0	0	0	Start a high-priority AD conversion by software.

Interrupt routine processing example

WA	←	ADREGSP	Read value of ADREGSPL and ADREGSPH into 16-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2A00H)	←	WA	Write contents of WA to memory address 2A00H.

4. Convert the analog input voltage on the AN4 pin as a normal AD conversion of a channel fixed single conversion mode. And then if its conversion result is greater or equal than the value of (ADCM0REGL/H), write the result to memory address 2C00H using the AD monitor function interrupt (INTADM) processing routine.

Main routine

INTEAD	←	–	–	–	–	1	0	1	1	Enable INTAD and set it to interrupt level 3.
ADMOD5	←	0	0	0	0	1	0	0	0	Set the analog input channel AN4 for AD monitor function 0.
ADMOD4	←	0	0	1	0	0	0	0	0	Enable the AD monitor function0 and AD monitor function interrupt 0. Set “a conversion result ≥ AD conversion result compare criterion register” for generation condition of monitor function interrupt 0.
ADMOD1	←	1	0	1	0	0	0	0	0	Set pin AN4 to be the analog input channel.
ADMOD0	←	0	0	0	0	1	0	0	0	Start a normal AD conversion by software.

Interrupt routine processing example

WA	←	ADREG4	Read value of ADREG4L and ADREG4H into 16-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2C00H)	←	WA	Write contents of WA to memory address 2C00H.

X : Don't care, – : No change

3.24 Watchdog Timer (Runaway detection timer)

The TMP92CZ26A contains a watchdog timer of runaway detecting.

The watchdog timer (WDT) is used to return the CPU to the normal state when it detects that the CPU has started to malfunction (runaway) due to causes such as noise. When the watchdog timer detects a malfunction, it generates a non-maskable interrupt INTWD to notify the CPU of the malfunction.

Connecting the watchdog timer output to the reset pin internally forces a reset.

(The level of external $\overline{\text{RESET}}$ pin is not changed.)

3.24.1 Configuration

Figure 3.24.1 is a block diagram of the watchdog timer (WDT).

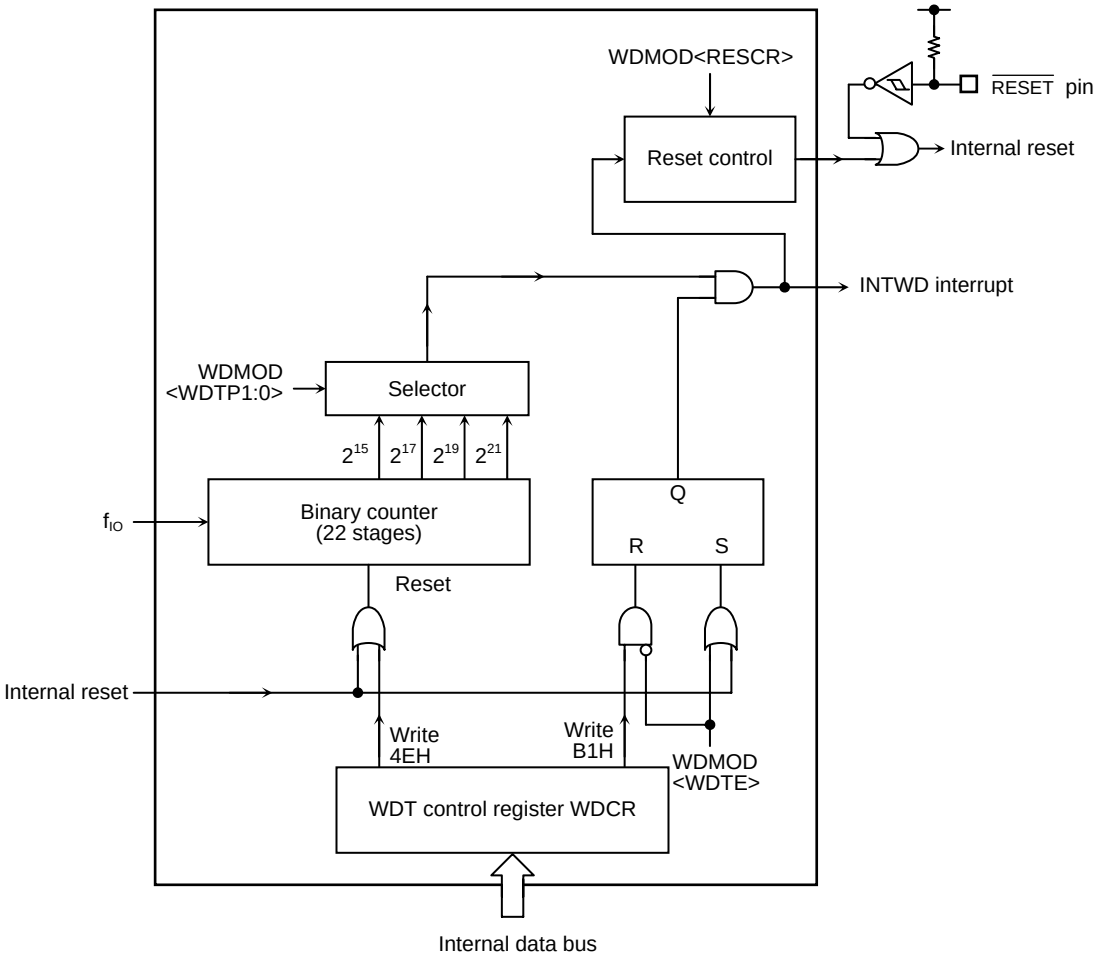


Figure 3.24.1 Block Diagram of Watchdog Timer

Note: It needs to care designing the total machine set, because Watchdog timer can't operate completely by external noise.

3.24.2 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1:0> has elapsed. The watchdog timer must be cleared “0” in software before an INTWD interrupt will be generated. If the CPU malfunctions (e.g., if runaway occurs) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter will overflow and an INTWD interrupt will be generated. The CPU will detect malfunction (runaway) due to the INTWD interrupt and in this case it is possible to return to the CPU to normal operation by means of an anti-malfunction program.

The watchdog timer begins operating immediately on release of the watchdog timer reset.

The watchdog timer is halted in IDLE1 or STOP mode. The watchdog timer counter continues counting during bus release (when $\overline{\text{BUSAK}}$ goes low).

When the device is in IDLE2 mode, the operation of WDT depends on the WDMOD<I2WDT> setting. Ensure that WDMOD<I2WDT> is set before the device enters IDLE2 mode.

The watchdog timer consists of a 22-stage binary counter which uses the clock (f_{IO}) as the input clock. The binary counter can output $2^{15}/f_{IO}$, $2^{17}/f_{IO}$, $2^{19}/f_{IO}$ and $2^{21}/f_{IO}$. Selecting one of the outputs using WDMOD<WDTP1:0> generates a watchdog timer interrupt when an overflow occurs.

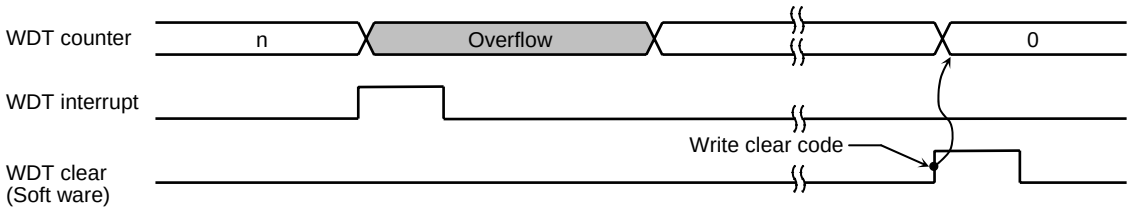


Figure 3.24.2 Normal Mode

The runaway detection result can also be connected to the reset pin internally.

In this case, the reset time will be 32 clocks (102.4 μs at f_{OSCH} = 10 MHz) as shown in Figure 3.24.3. After a reset, the clock f_{IO} is divided f_{SYS} by two, where f_{SYS} is generated by dividing the high-speed oscillator clock (f_{OSCH}) by sixteen through the clock gear function

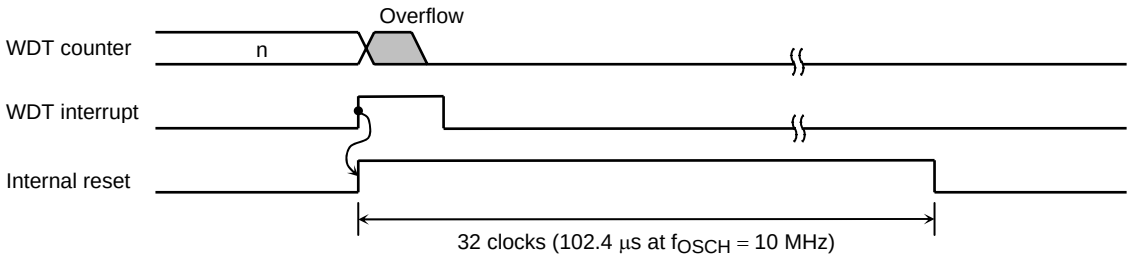


Figure 3.24.3 Reset Mode

3.24.3 Control Registers

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

(1) Watchdog timer mode registers (WDMOD)

1. Setting the detection time for the watchdog timer in <WDTP1:0>

   This 2-bit register is used for setting the watchdog timer interrupt time used when detecting runaway.

   On a reset this register is initialized to WDMOD<WDTP1:0> = 00.

   The detection time for WDT is $2^{15}/f_{IO}$ [s]. (The number of system clocks is approximately 65, 536.)

2. Watchdog timer enable/disable control register <WDTE>

   At reset, the WDMOD<WDTE> is initialized to "1", enabling the watchdog timer.

   To disable the watchdog timer, it is necessary to clear this bit to "0" and to write the disable code (B1H) to the watchdog timer control register (WDCR). This makes it difficult for the watchdog timer to be disabled by runaway.

   However, it is possible to return the watchdog timer from the disabled state to the enabled state merely by setting <WDTE> to "1".

3. Watchdog timer out reset connection <RESCR>

   This register is used to connect the output of the watchdog timer with the RESET terminal internally. Since WDMOD<RESCR> is initialized to 0 at reset, a reset by the watchdog timer will not be performed.

(2) Watchdog timer control registers (WDCR)

This register is used to disable and clear the binary counter for the watchdog timer.

- Disable control

  The watchdog timer can be disabled by clearing WDMOD<WDTE> to 0 and then writing the disable code (B1H) to the WDCR register.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).
WDMOD	←	0	–	–	X	X	–	–	0	Clear WDMOD <WDTE> to "0".
WDCR	←	1	0	1	1	0	0	0	1	Write the disable code (B1H).

- Enable control

  Set WDMOD<WDTE> to "1".

- Watchdog timer clear control

  To clear the binary counter and cause counting to resume, write the clear code (4EH) to the WDCR register.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).

Note1: If it is used disable control, set the disable code (B1H) to WDCR after write the clear code (4EH) once. (Please refer to setting example.)

Note2: If it is changed Watchdog timer setting, change setting after set to disable condition once.

WDMOD (1300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0			I2WDT	RESCR	–
Read/Write	R/W					R/W		
After reset	1	0	0			0	0	0
Function	WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{IO}$ 01: $2^{17}/f_{IO}$ 10: $2^{19}/f_{IO}$ 11: $2^{21}/f_{IO}$				IDLE2 0: Stop 1: Operate	1: Internally connects WDT out to the reset pin	Always write “0”

Watchdog timer out control

0	–
1	Connects WDT out to a reset

IDLE2 control

0	Stop
1	Operation

Watchdog timer detection time

00	$2^{15}/f_{IO}$ (Approximately 819.2 μs at f_{IO} = 40 MHz)
01	$2^{17}/f_{IO}$ (Approximately 3.276 ms at f_{IO} = 40 MHz)
10	$2^{19}/f_{IO}$ (Approximately 13.107 ms at f_{IO} = 40 MHz)
11	$2^{21}/f_{IO}$ (Approximately 52.428 ms at f_{IO} = 40 MHz)

Watchdog timer enable/disable control

0	Disabled
1	Enabled

Figure 3.24.4 Watchdog Timer Mode Register

WDCR (1301H)

Read -modify -write instruction is prohibited

	7	6	5	4	3	2	1	0
Bit symbol	–							
Read/Write	W							
After reset	–							
Function	B1H: WDT disable code 4EH: WDT clear code							

WDT disable/clear control

B1H	Disable code
4EH	Clear code
Others	Don’t care

Figure 3.24.5 Watchdog Timer Control Register

3.25 Power Management Circuit (PMC)

The TMP92CZ26A incorporates a power management circuit (PMC) for managing power supply in standby state as protective measures against leak current in fine-process products. The following six power supply rails are available.

・Analog power supply	:	AVCC & AVSS (for ADC)
・3V-A, 3V-B digital I/O power supply	:	DVCC3A, 3B & DVSSCOM (for general pins)
・1.5V-A digital internal power supply	:	DVCC1A & DVSSCOM (for general circuits)
・1.5V-B digital internal power supply	:	DVCC1B & DVSSCOM (for RTC, PMC)
・1.5V-C oscillation power supply	:	DVCC1C & DVSS1C (for high-frequency oscillator, PLL)

Each power supply rail is independent of one another (VSS is partially shared).

Of the six power supply rails, those that are supplied in Power Cut Mode are the power supply rail for external pins (DVCC-3A, DVCC-3B), the power supply rail for ADC (AVCC), and the power supply rail for RTC and backup RAM (DVCC-1B). DVCC1A and DVCC1C power supply rails are isolated internally with their signals cut off so that no flow-through current will be generated in the LSI when the power is turned off.

- DVCC-3A, DVCC-3B

  This 3V rail supplies power for holding external pins, controlling ON/OFF of external power supplies, and interrupt input to release standby state.

- AVCC

  This 3V rail supplies power in the touch panel interface for interrupt input to release standby state.

- DVCC-1B

  This 1.5V rail supplies power to the RTC, 16 Kbytes of RAM, and PMC.

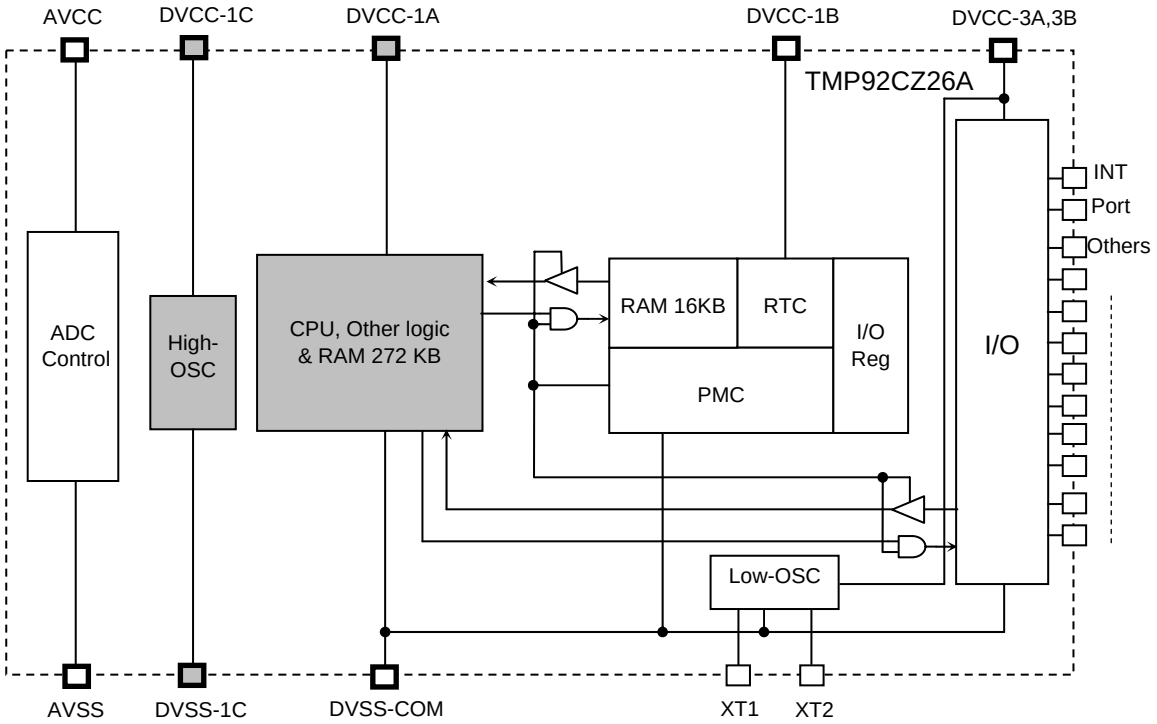


Figure 3.25.1 Power Supply System

3.25.1 SFR

PMCCTL (02F0H)

	7	6	5	4	3	2	1	0
bit symbol	PCM_ON					–	WUTM1	WUTM0
Read/Write	R/W					W	R/W	R/W
After system reset	0					0	0	0
After hot reset	Data retained					–	Data retained	Data retained
Function	Power Cut Mode 0: Disable 1: Enable					Always write “0” Always read as “0”	Warm-up time 00: 29 (15.625 ms) 01: 210 (31.25 ms) 10: 211 (62.5 ms) 11: 212 (125 ms)	

Note: After wake-up interruption, internal wake-up timer count setting register value:<WUTM1:0>, and after about 77us, external PWE terminal change from low level to high level. Additionally after more about 92us, internal reset signal will be released. We recommend to confirm actual performance on final set, because the time to be stable all voltage level and power supply circuit are difference characteristics every final set.

The following operations are affected by the setting of the <PCM_ON> bit.

	PCM_ON = 1	PCM_ON = 0
External interrupt input	No interrupt HOT_RESET signal assert	Interrupt
Operation after reset	–	Startup depending on the AM1 and AM0 pins
Operation after hot reset	Startup from boot ROM regardless of the AM1 and AM0 pins and jump to internal RAM area.	–
Warm-up counter	A change in the PWE pin level is used as a trigger to start counting the low-frequency clock for releasing HOT_RESET.	Counter stopped

3.25.2 Detailed Description of Operation

This section explains the procedures for entering and exiting the Power Cut Mode.

- Entering the Power Cut Mode

When to enter the Power Cut Mode, the CPU needs to be operating in the internal RAM.

Low frequency clock (XT) must be enable condition.

It is also necessary to disable interrupt requests, stop DMA operations, WDT and AD converter. Next, set the output pins to function as ports through the Pn, PnCR and PnDR registers. At this time, PM7 should be set as PWE. Of the external interrupt pins, those to be used for waking up from the Power Cut Mode should be set as input pins with interrupt enabled.

About trigger of interruption, only rising edges are effective among selectable interruption pins. When INT4 is used as TSI, the de-bounce circuit should be disabled.

Then, set the warm-up time for waking up from Power Cut Mode in PMCCTL<WUTM1:0>. Write the wake-up program at addresses from 46000H to 49FFFH in the internal RAM.

Should be written all initialize sequence including WDT in this program.

Finally, stop the PLL and set PMCCTL<PCM_ON> to “1” to enter the Power Cut Mode.

At this time, the RESET (HOT_RESET) signal is asserted for all the circuits excluding external I/O and PMC.

Note: As soon as PMCCTL<PCM_ON> is set to “1”, the power management signal (PWE) changes from “1” to “0” and external power supplies are turned off.

1. Prepare to shift Power Cut Mode

(1) Set the warm-up time: PMCCTL<WUTM1:0>

After wake-up interruption, internal wake-up timer count setting register value:<WUTM1:0>, and after about 77us, external PWE terminal change from low level to high level. Additionally after more about 92us, internal reset signal will be released. We recommend to confirm actual performance on final set, because the time to be stable all voltage level and power supply circuit are difference characteristics every final set.

Warm-up time can be selected among 15.625ms, 31.25ms, 62,5ms and 125ms.

(2) Prepare the initial program after Warm-up (46000H~49FFFH)

After wake-up, jump to Boot ROM, and Boot ROM process distinguish only bit 7 of PMCCTL register. All initialize setting including WDT setting must be written in fixed RAM area (46000H~49FFFH).

(3) Control of low frequency clock (XT)

Power Management Circuit operates by low frequency clock. Low frequency clock (XT) must be enable condition

2. Operation Sequence

(1) Execution area of program must shift to internal RAM area.

Before shifting Power Cut Mode, it must stop all the source which might be disturbed to shift Power Cut Mode.

a. Disable Watch Dog Timer operation
b. Disable A/D converter operation
c. Disable all DMA function
   - Disable LCDC
   - Auto refresh of SDRAM (We recommend to use self refresh mode)
   - Disable DMAC

(2) Fix to port condition (Pn, PnCR, PnFC,PnDR)

Fix port condition and set external interrupt mode to wake-up trigger.

When INT4 is used as TSI, the de-bounce circuit should be disabled.

(3) Disable interruption (DI)

(4) Stop PLL operation

(5) Shift to Power Cut Mode (PMCCTL<PCM_ON>= “1”)

- Exiting the Power Cut Mode

The Power Cut Mode can be exited by external or internal interruption. (It inhibits to exit the Power Cut Mode by reset when DVCC1A is cut off. Reset must be asserted after supplying power to DVCC1A and waiting for its voltage to fully stabilize.) The interrupts that can be used to exit the Power Cut Mode are RTC interrupt, INT0 to INT7 (TSI interrupt) and INTKEY interrupt.

Table 3.25.1 Wake-up triggers

Source	Symbol	Note
RTC	INTRTC	
External	INT0	Only support "Rising Edge"
	INT1	Only support "Rising Edge"
	INT2	Only support "Rising Edge"
	INT3	Only support "Rising Edge"
	INT4	When TSI, need to disable de-bounce circuit Only support "Rising Edge"
	INT5	Only support "Rising Edge"
	INT6	Only support "Rising Edge"
	INT7	Only support "Rising Edge"
Key	INTKEY	KI0~KI8 Only support "Falling Edge"

When an interrupt request is accepted, the power management signals (PWE) changes from "0" to "1" and power is supplied to each block that has been cut off. After the warm-up time set in PMCCTL<WUTM1:0> has elapsed, HOT_RESET is automatically released and the CPU starts up from the internal boot ROM regardless of the external AM pin state. All external ports retain the state before entering the Power Cut Mode except for the PnDR setting which is released upon release of HOT_RESET.

* Output pin Hi-Z state → "1" or "0" output
* Input pin input gate OFF → Input pin input gate ON

The internal boot ROM first checks the PMCCTL <PCM_ON> bit in the PMC. If this bit is set to "1", execution jumps to address 46000H in the internal RAM before making all initial settings. The <PCM_ON> bit in the PMC is cleared to "0" by software.

Note 1: The interrupt that released the Power Cut Mode, whichever it is, does not activate any interrupt operation. Nor is it possible to identify which interrupt released the Power Cut Mode.

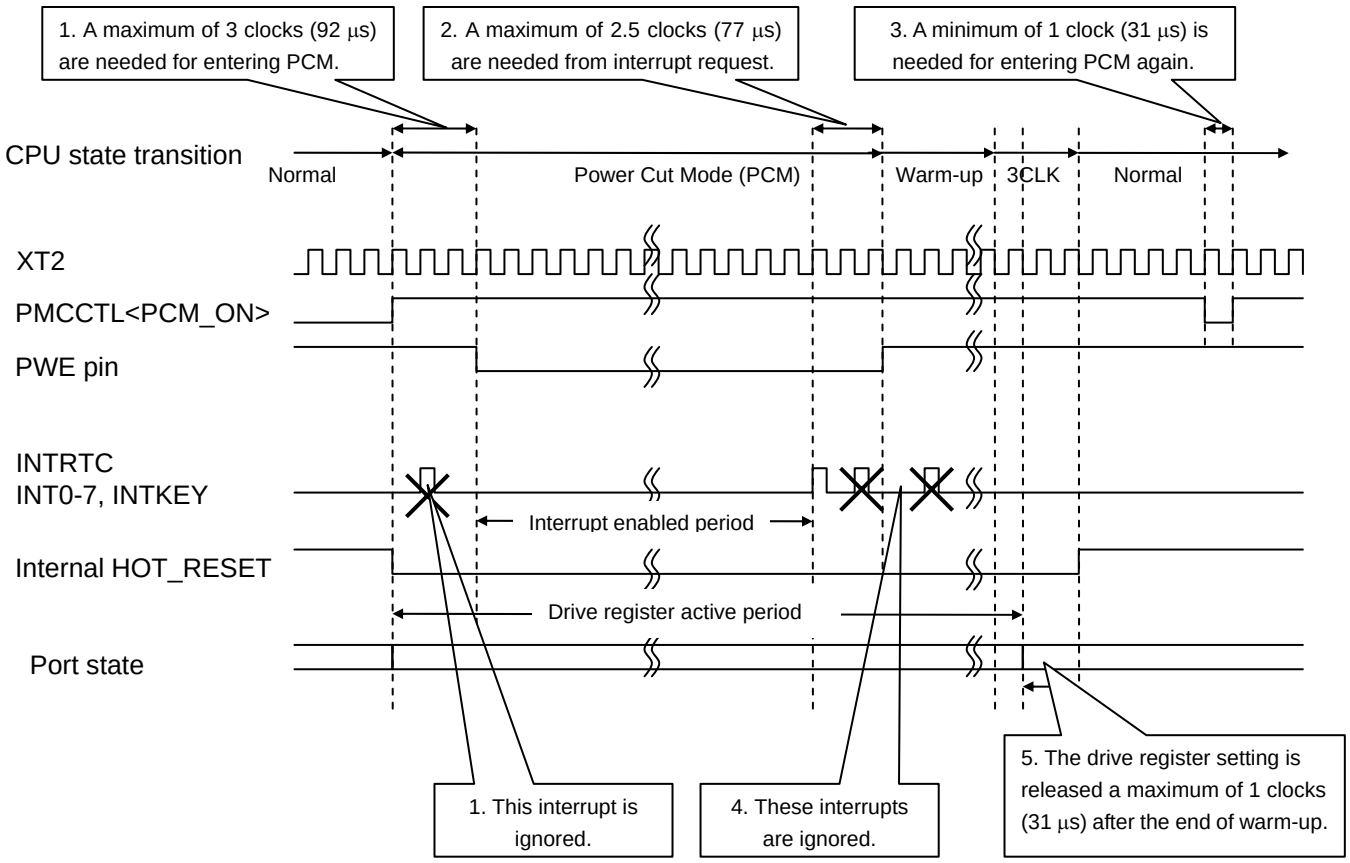
Note 2: Once the PMCCTL<PCM_ON> bit is set to "1", it remains in this state. To re-enter the Power Cut Mode, it is necessary to set this bit to "0" once and then to "1" again. At this time, a minimum of 31 us must be inserted between setting <PCM_ON> to "0" and "1".

Note 3: Since the Power Cut Mode is exited using the boot ROM, some settings must be made by software. Be careful about this point.

BROMCR (016CH)

	7	6	5	4	3	2	1	0
Bit symbol						CSDIS	ROMLESS	VACE
Read/Write						R/W		
After reset						1	0	1
Function						NAND Flash area CS output 0: Enable 1: Disable	Boot ROM 0: Used 1: Not used	Vector address conversion 0: Disable 1: Enable

3.25.3 Detailed Description of Timing



Internal HOT_RESET assert to dead circuit only. (DVCC1A &DVCC1C circuit)

1. If it is set PMCCTL<PCM_ON>="1", shift the Power Cut Mode, however, it spends 3-clock times maximum (around 92μS) to shift from normal mode to Power Cut Mode. And the wake-up triggers during this 3-clock times, are ignored.
2. It spends 2.5-clock times maximum (around 77μS) from the trigger to wake-up to rise-up the PWE terminal.
3. After wake-up from Power Cut Mode, reset to "0" the PMCCTL<PCM_ON> bit by soft ware. If you want to shift Power Cut Mode again, need to wait 1-clock time minimum (around 31μS).
4. The wake-up triggers during waking-up, are ignored.
5. After Warm-up count, and spend 1-clock time (around 31μS), release the DRV setting of every ports. After that, spends 2-clock time (around 62μS), release internal RESET (Hot_Reset).

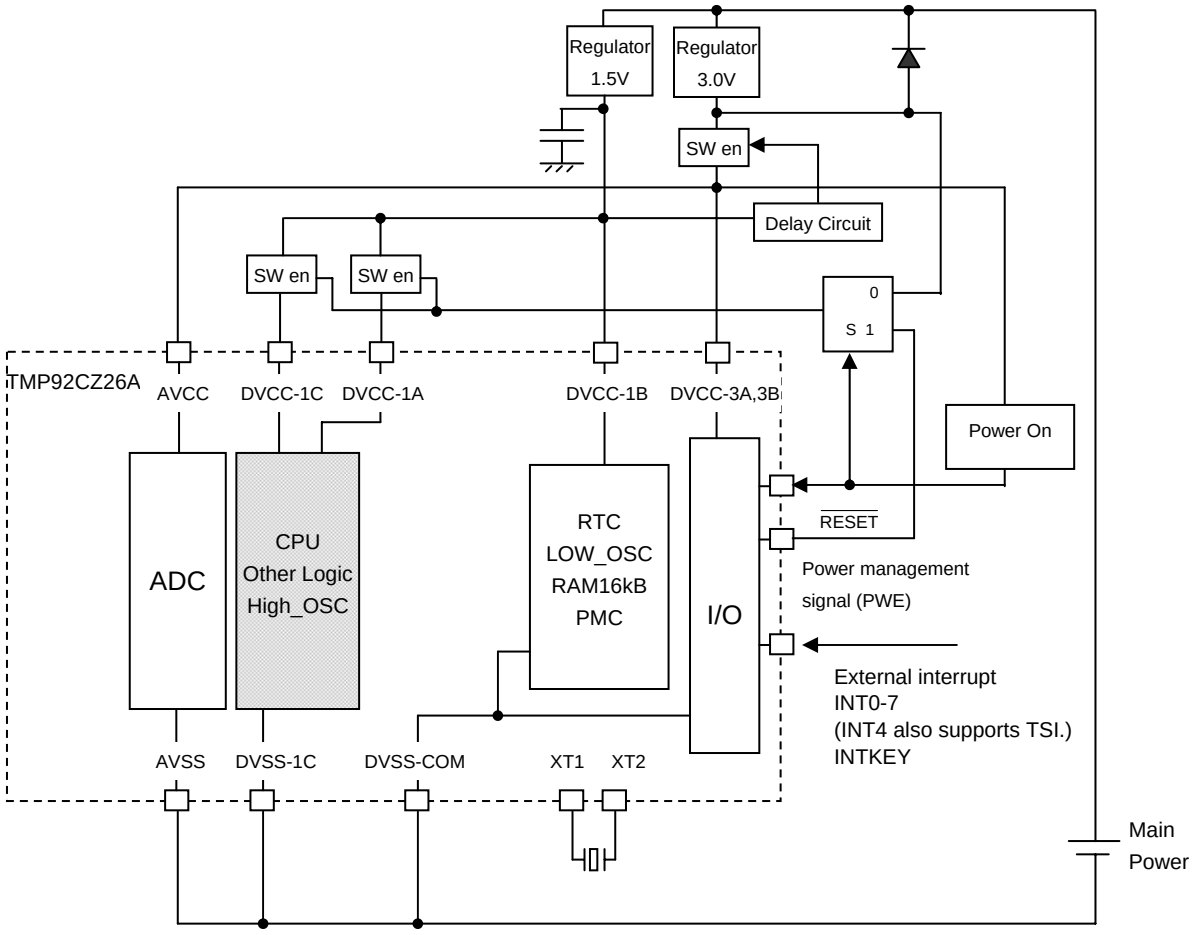


Figure3.25.2 Example External Circuitry for Using the PMC

Figure3.25.2 shows an example of external circuitry for using the PMC.

In normal mode, the power management pin (PWE) outputs “1” and power is supplied to all the blocks in the TMP92CZ26A.

In the Power Cut Mode, the power management pin (PWE) outputs “0” and power is cut off for the internal circuitry excluding the CPU, part of internal RAM, AD converter and RTC to reduce leak current. In the Power Cut Mode, power is supplied to only the I/O (including the AD pins), TSI circuit, 16 Kbytes of internal RAM, low-frequency oscillation circuit, RTC and PMC.

3.25.4 Notes of Power sequence

- Power ON/Power OFF Sequence (Initial Power ON/Complete Power OFF)

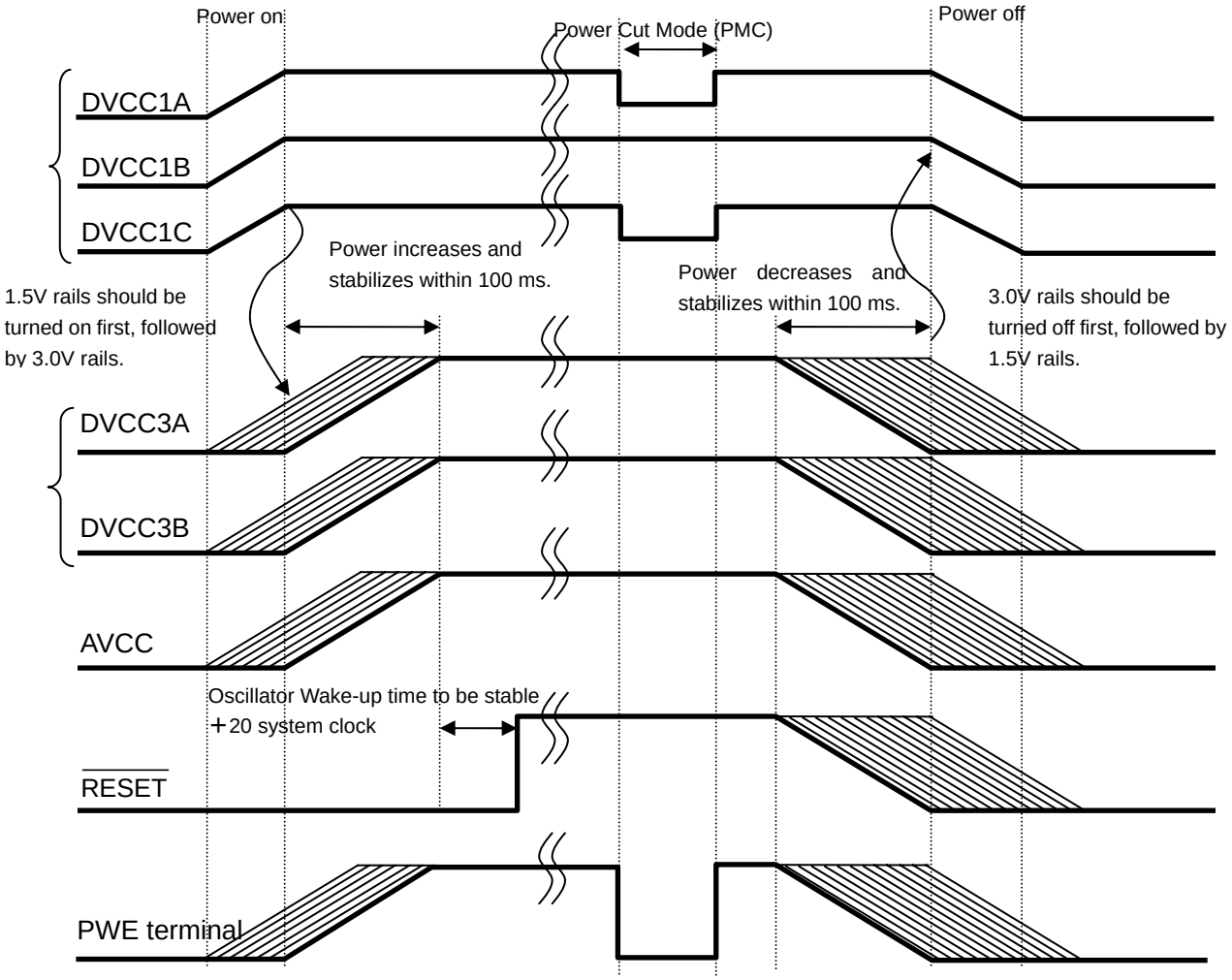
In the power ON sequence (initial power ON), power must be supplied to internal circuits first and then to external circuits, as shown below. In the power OFF sequence (complete power OFF), power must be turned off from external circuits so that internal circuits are turned off last.

Power ON

(DVCC1A, DVCC1B, DVCC1C) → (DVCC3A, DVCC3B, AVCC)

Power OFF

(AVCC, DVCC3A, DVCC3B) → (DVCC1C, DVCC1B, DVCC1A)



Note1: Although it is possible to turn on or off 1.5V and 3.0V rails simultaneously, external pins may temporarily become unstable in this case. Therefore, if there is any possibility that this would affect external devices connected with the TMP92CZ26A, external power supplies should be turned on or off while internal power supplies are stable, as shown in the diagram above.

Note2: In the power ON sequence, 3V rails must not be turned on before 1.5V rails. In the power OFF sequence, 3V rails must not be turned off after 1.5V rails.

3.25.5 Setting Example

```
Condition: Wake-up trigger=INT4(TSI)
        org     002000h

        ld      (syscr0),40h    ;   Enable low frequency clock
        ldw     (wdmod),0b100h  ;   Disable WDT
        ldw     (admod0),0000h  ;
        ldw     (admod2),0000h  ;   Disable AD converter
        ldw     (admod4),0000h  ;
        ld      (lcdctl0),00h   ;   Disable DMA operation
        ld      (pmfc),80h      ;   Set PM7 port to PWE function

        ld      (p9fc),40h      ;
                                    Set INT4 and set level
        ld      (inte34),50h    ;
        ld      (tsicr1),00h    ;   Disable de-bounce circuit

        ld      (pllcr0), 00h   ;   Change CPU clock from PLL to fOSCH
        ld      (pllcr1), 00h   ;   Stop the PLL circuit
        ld      (pmcctl),00h    ;   Set Warm-up time
        di                      ;
        ld      (pmcctl),80h    ;   Enable <PCM_ON> = 1
                                    Shift to Power Cut Mode

; After Wake-up
        org     046000h

        ld      (pmcctl),00h    ;   Disable <PCM_ON> = 0
```

3.26 Multiply and Accumulate Calculation Unit (MAC)

The TMP92CZ26A includes a multiply-accumulate unit (MAC) capable of 32-bit × 32-bit + 64-bit arithmetic operations at high speed. The MAC has the following features:

- One-cycle execution for all MAC operations (excluding register access time)
- Three operation modes : 1) 64-bit + 32-bit × 32-bit
  2) 64-bit – 32-bit × 32-bit
  3) 32-bit × 32-bit – 64-bit
- Support for signed/unsigned operations
- Support for integer operations only

3.26.1 Registers

The MAC in the TMP92CZ26A has one control register and three data registers. These registers are connected to the CPU via a 32-bit bus and can be accessed in one system clock (f_{SYS}).

3.26.1.1 Control Register

The control register is used to control the operation of the MAC.

MAC Control Register

MACCR (1BFCH)

Prohibit Read-modify-write

	7	6	5	4	3	2	1	0
bit Symbol	MOVF	MOPST	MSTTG2	MSTTG1	MSTTG0	MSGMD	MOPMD1	MOPMD0
Read/Write	R/W	W	R/W			R/W	R/W	
After reset	0	0	0	0	0	0	0	0
Function	Overflow flag 0: No overflow 1: Overflow occurred	Calculation soft start 0:Don't care 1:Start calculation	Calculation start trigger 000: Write to MACMA<7:0> 001: Write to MACMB<7:0> 010: Write to MACMOR<7:0> 011: Write to MACMOR<39:32> 1xx: Write of "1" to <MOPST>			Sign mode 0: Unsigned 1: Signed	Calculation mode 00: 64 + 32×32 01: 64 – 32×32 10: 32×32 – 64 11: Reserved	

Note 1: <MOPST> is write-only and it is read as "0".

Note 2: Writing "1xx" to <MSTTG2:0> and writing "1" to <MOPST> can be executed in the same write cycle.

Note 3: <MOVF> is fixed two system clocks (f_{SYS}) after calculation is started.

3.26.1.2 Data Registers

The data registers are arranged as shown below.

	Data Registers							
	Bits<63:56>	Bits<55:48>	Bits<47:40>	Bits<39:32>	Bits<31:24>	Bits<23:16>	Bits<15:8>	Bits<7:0>
Multiplier A Register					(1BE3H)	(1BE2H)	(1BE1H)	MACMA (1BE0H)
Multiplier B Register					(1BE7H)	(1BE6H)	(1BE5H)	MACMB (1BE4H)
MAC Register	(1BEFH)	(1BEEH)	(1BEDH)	MACORH (1BECH)	(1BEBH)	(1BEAH)	(1BE9H)	MACORL (1BE8H)

Note 1: After reset, all the registers are cleared to “0”.

Note 2: Read-modify-write instructions can be used on all the registers.

Note 3: All the registers can be accessed in long word, word, or byte units.

Note 4: When MACCR<MSTTG2:0> is set to “0”, “001”, “010” or “011” and the registers are written in word or byte units, the <7:0> bits of each register must be written last.

Note 5: The MACORL register is fixed one system clock (f_{SYS}) after calculation is started, and the MACORH register is fixed two system clocks (f_{SYS}) after calculation is started. Therefore, to read the MACOR register immediately after calculation, be sure to read the MACORL register first.

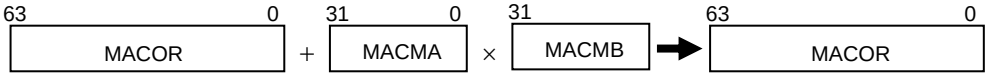
3.26.2 Description of Operation

(1) Calculation mode

The MAC has the following three types of calculation mode. The calculation mode to be used is specified in MACCR<MOPMD1:0>. MACCR<MSMD> is used to select unsigned or signed mode. The operation of each calculation mode is explained below.

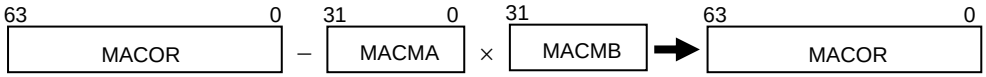
(a) 64 + 32 × 32 mode

In this mode, the contents of the MACMA register and the MACMB register are multiplied and the result is added to the contents of the MACOR register. Then, the result is stored back in the MACOR register.



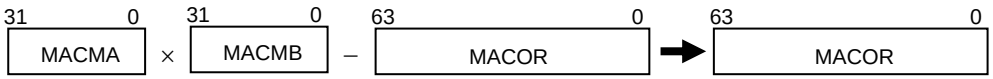
(b) 64 – 32 × 32 mode

In this mode, the contents of the MACMA register and the MACMB register are multiplied and the result is subtracted from the contents of the MACOR register. Then, the result is stored back in the MACOR register.



(c) 32 × 32 – 64 mode

In this mode, the contents of the MACMA register and the MACMB register are multiplied and the contents of the MACOR register are subtracted from the result. Then, the result is stored back in the MACOR register.



(d) Sign mode

Both multiply-accumulate and multiply-subtract operations can be executed in unsigned or signed mode.

In signed mode, the MACMA, MACMB, and MACOR registers become signed registers, and the most significant bit is treated as the sign bit and the data set in each register is treated as a two's complement value. Table 3.26.1 shows the range of values that can be represented in each sign mode.

Table 3.26.1 Data Range in Unsigned/Signed Mode

	MACMA, MACMB Registers	MACOR Register
Unsigned	$0 \sim 2^{32}-1$	$0 \sim 2^{64}-1$
Signed	$-2^{31} \sim +2^{31}-1$	$-2^{63} \sim +2^{63}-1$

Use signed mode when the values to be set in the MACMA and MACMB registers are signed (two's complement) data. Even in unsigned mode it is possible to set signed (two's complement) data in the MACOR register to perform additions and subtractions in signed mode.

(2) Calculation start trigger

As a trigger to start calculation, writing to the MACMA, MACMB or MACOR register or soft start (MACCR<MOPST>=1) can be selected in MACCR<MSTTG2:0>.

(3) Overflow flag

When an overflow occurs in the calculation result (see Table 3.26.2), MACCR<MOVF> is set to "1". Once an overflow occurs, MACCR<MOVF> is held at "1" regardless of subsequent calculation results. Since the overflow flag is not automatically cleared by a read operation, it is necessary to write "0" to clear this flag.

Table 3.26.2 Overflow Definitions

Sign Mode	Calculation Result (MACOR register value)	MACCR<MOVF>
Signed	$MACOR > 2^{64}-1$	1
	$0 \le MACOR \le 2^{64}-1$	0
	$MACOR < 0$	1
Unsigned	$MACOR > 2^{63}-1$	1
	$-2^{63} \le MACOR \le 2^{63}-1$	0
	$MACOR < -2^{63}$	1

3.26.3 Operation Examples

(1) Unsigned multiply-accumulate operation

The following shows a setting example for calculating "33333333 + 11111111 × 22222222":

```
ld      (MACCR), 0x08       ; Unsigned multiply-accumulate mode
                              Start calculation by write to MACMB.
ld      xde, 0x00000000
ld      xhl, 0x33333333
ld      xix, 0x11111111
ld      xiy, 0x22222222
ld      (MACORL), xhl       ; Write 33333333 to MACORL.
ld      (MACORH), xde       ; Clear MACORH.
ld      (MACMA), xix        ; Write 11111111 to MACMA.
ld      (MACMB), xiy        ; Write 22222222 to MACMB.   <-- Calculation start
ld      xhl, (MACORL)       ; Read lower result 0x41FDB975.
bit     7, (MACCR)                  ; Check over-flow error
jp      nz, ERROR           ; Go to error routine, if there is over-flow error
ld      xde, (MACORH)       ; Read upper result 0x02468ACF.
```

(2) Signed multiply-subtract operation

The following shows a setting example for calculating "33333333 – 11111111 × –22222222":

```
ld      (MACCR), 0x25       ; Signed multiply-subtract mode
                              Start calculation by write of "1" to <MOPST>.
ld      xde, 0x00000000
ld      xhl, 0x33333333
ld      xix, 0x11111111
ld      xiy, 0xDDDDDDDE     ; −22222222
ld      (MACORL), xhl       ; Write 33333333 to MACORL.
ld      (MACORH), xde       ; Clear MACORH.
ld      (MACMA), xix        ; Write 11111111 to MACMA.
ld      (MACMB), xiy        ; Write −22222222 to MACMB.
set     5, (MACCR)                  ;            <-- Calculation start
ld      xhl, (MACORL)       ; Read lower result 0x41FDB975.
bit     7, (MACCR)                  ; Check over-flow error
jp      nz, ERROR           ; Go to error routine, if there is over-flow error
ld      xde, (MACORH)       ; Read upper result 0x02468ACF.
```

(3) Unsigned multiply-accumulate operation (two multiply-accumulate operations)

The following shows a setting example for calculating "(33333333 + 11111111 × 22222222) + (11111111 × 44444444)":

```
ld      (MACCR), 0x08       ; Unsigned multiply-accumulate mode
                              Start calculation by write to MACMB.
ld      xde, 0x00000000
ld      xhl, 0x33333333
ld      xix, 0x11111111
ld      xiy, 0x22222222
ld      xiz, 0x44444444
ld      (MACORL), xhl       ; Write 33333333 to MACORL.
ld      (MACORH), xde       ; Clear MACORH.
ld      (MACMA), xix        ; Write 11111111 to MACMA.
ld      (MACMB), xiy        ; Write 22222222 to MACMB.   <-- Calculation start
ld      (MACMB), xiz        ; Write 44444444 to MACMB.   <-- Calculation start
ld      xhl, (MACORL)       ; Read lower result 0x5F92C5F9.
bit     7, (MACCR)                  ; Check over-flow error
jp      nz, ERROR           ; Go to error routine, if there is over-flow error
ld      xde, (MACORH)       ; Read upper result 0x06D3A06D.
```

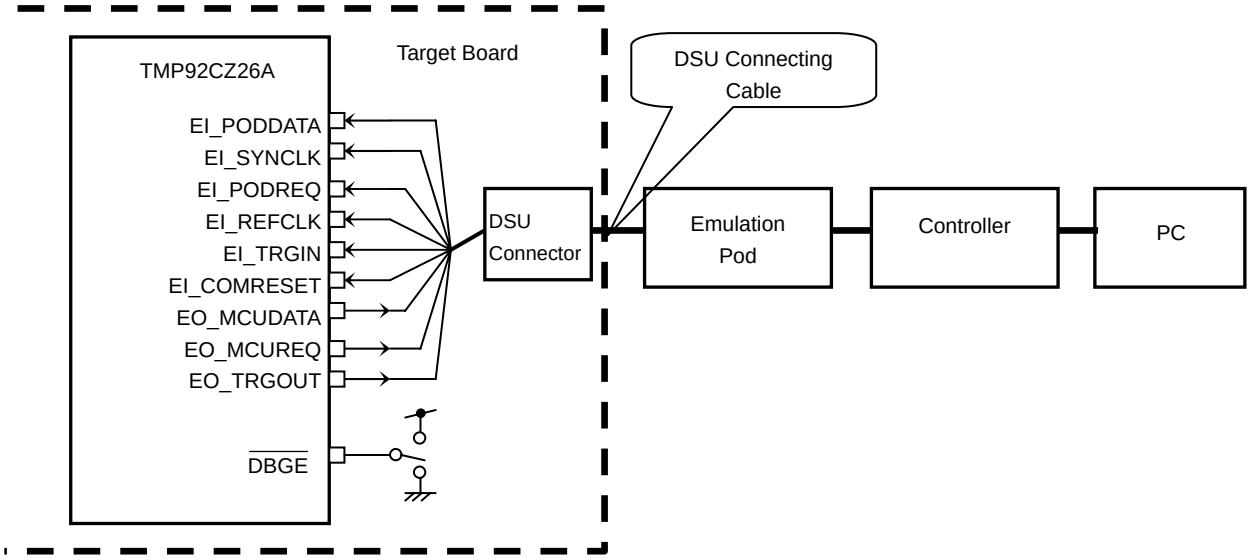
3.27 Debug Mode

The TMP92CZ26A includes a debug support unit (DSU) for enabling on-board debugging.

The DSU has 9 debug pins for interfacing with an external emulator via a DSU connector to be mounted on the target board and a DSU connecting cable. For details about debugging, please refer to the instruction manual of the emulation pod to be used.

This section provides product-specific explanations related to debug mode.

(1) Connection method



Note: When connecting the TMP92CZ26A and an emulator in debug mode, place the DSU connector on the target board as near (less than 5cm) to the TMP92CZ26A as possible. It is desirable that all the signals are same length.

Recommend connector: SAMTEC FTSH-110-01-DV-EJ

(2) How to enter debug mode

Debug mode can be entered by setting the $\overline{\text{DBGE}}$ pin to Low. To return to normal mode from debug mode, be sure to set the $\overline{\text{DBGE}}$ pin to High and then reset the system using the $\overline{\text{RESET}}$ pin. In details of debus mode, refer the manual of emulation POD.

(3) Limitations in debug mode

Debug mode has the following limitations:

1) Target reset

While debugging is being performed, the system reset ($\overline{\text{RESET}}$ pin) of the target (microcontroller) must not be used to reset the controller and microcontroller. Instead, reset should be performed from the controller. (For details, please refer to the instruction manual of the emulation pod to be used.)

* If reset from the microcontroller by the $\overline{\text{RESET}}$ pin may clash the register information and internal RAM data in the CPU, including not only programs but also breakpoint and trace information.

2) Pins

In debug mode, a total of 9 pins (PZ0 to PZ7 in Port Z and PU7 in Port U) are used to connect the TMP92CZ26A with an emulator via a DSU probe for communicating with the controller. For this reason, these 9 pins cannot be debugged. Therefore, if the port control register of each pin is changed in debug mode, the register contents are changed but the function of each pin remains the same.

Port Z Register

PZ (0068H)

	7	6	5	4	3	2	1	0
bit Symbol	PZ7	PZ6	PZ5	PZ4	PZ3	PZ2	PZ1	PZ0
Read/Write	R/W							
After reset	External pin data (Output latch is reset to "0".)							

Port Z Control Register

PZCR (006AH)

	7	6	5	4	3	2	1	0
bit Symbol	PZ7C	PZ6C	PZ5C	PZ4C	PZ3C	PZ2C	PZ1C	PZ0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port Z Function Register

PZFC (006BH)

	7	6	5	4	3	2	1	0
bit Symbol	PZ7F	PZ6F	PZ5F	PZ4F	PZ3F	PZ2F	PZ1F	PZ0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port							

Port Z Drive Register

PZDR (009AH)

	7	6	5	4	3	2	1	0
bit Symbol	PZ7D	PZ6D	PZ5D	PZ4D	PZ3D	PZ2D	PZ1D	PZ0D
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Input/output buffer drive register for standby mode							

Note: Although it is possible to write to shaded bits, writing to these bits has no effect (the DSU communication function is given a higher priority).

Port U Register

PU (00A4H)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
Read/Write		R/W						
After reset		External pin data (Output latch is reset to “0”.)						

Port U Control Register

PUCR (00A6H)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7C	PU6C	PU5C	PU4C	PU3C	PU2C	PU1C	PU0C
Read/Write		W						
After reset	0	0	0	0	0	0	0	0
Function		0: Input 1: Output						

Port U Function Register

PUFC (00A7H)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7F	PU6F	PU5F	PU4F	PU3F	PU2F	PU1F	PU0F
Read/Write		W						
After reset	0	0	0	0	0	0	0	0
Function		0: Port 1: Data bus for LCDC (LD23 to LD16) Note: When LD23 to LD16 are used, set <PUnC> to “1”.						

Port U Drive Register

PUDR (009CH)

	7	6	5	4	3	2	1	0
Bit Symbol	PU7D	PU6D	PU5D	PU4D	PU3D	PU2D	PU1D	PU0D
Read/Write		R/W						
After reset	1	1	1	1	1	1	1	1
Function		Input/output buffer drive register for standby mode						

Note: Although it is possible to write to shaded bits, writing to these bits has no effect (the DSU communication function is given a higher priority).

3) Boot function

In this LSI, we support boot function, however, this boot function is not available in debug mode. (It is inhibit to set $\overline{\text{DBGE}}$ ="0", AM0="1" and AM1="1" at the same time.)

4) PMC function

In debug mode, the PMC function for cutting off the power supply to internal circuitry and reducing standby current is not also available.

BROMCR Register Specifications in Debug Mode

BROMCR (016CH)

	7	6	5	4	3	2	1	0
Bit symbol						CSDIS	ROMLESS	VACE
Read/Write							R/W	
After reset						1	1*	1/0
Function						NAND Flash area CS output 0: Enable 1: Disable	Boot ROM 0: Used 1: Not used	Vector address conversion 0: Disable 1: Enable

PMCCTL (02F0H)

	7	6	5	4	3	2	1	0
bit symbol	PCM_ON					–	WUTM1	WUTM0
Read/Write	R/W					W	R/W	R/W
After system reset	0					0	0	0
After hot reset	Data retained					–	–	–
Function	Power Cut Mode 0: Disable 1: Enable					Always write "0". Always read "0".	Warm-up time 00: 2^9 (15.625 ms) 01: 2^{10} (31.25 ms) 10: 2^{11} (62.5 ms) 11: 2^{12} (125 ms)	

Note: Even if the <PCM_ON> bit is set to "1", the Power Cut Mode cannot be entered (the external PWE pin is not set to "0").

5) Data bus occupancy

The TMP92CZ26A includes three controllers (LCD controller, SDRAM controller and DMAC) that function as bus masters apart from the CPU. Therefore, it is necessary to estimate the bus occupancy time of each bus master and control each function accordingly to ensure proper operation of each function. (For details, please refer to the chapter on the DMA controller.)

In debug mode, in addition to the operations of these bus masters, a steal program that runs in the background must also be taken into account in programming. When the program stops at a breakpoint (including step execution), the CPU operation is halted but the LCD controller, SDRAM controller and DMA controller remain active. At this time, the steal program also runs in the background. Once the steal program obtains the bus, it occupies the bus for 80 times of debug transmission clock (LH_SYNCLK) maximum. Therefore, in some cases, other DMA operations (LCD display, DMAC data transfer, SDRAM refresh) may not be performed at desired timing.

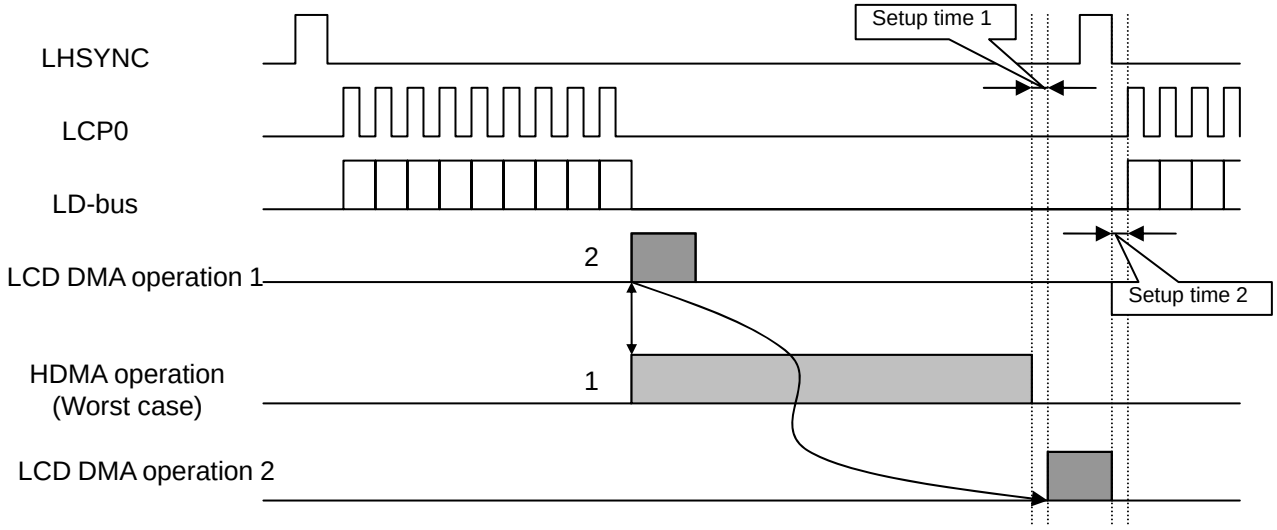


Figure 3.27.1 Example of Data Bus Occupancy Timing in Non-Debug Mode

Figure 3.27.1 shows an example of data bus occupancy timing in non-debug mode, depicting the LHSYNC signal, LCP0 signal, and LD-bus signal for transferring data from the LCD controller to the LCD driver, and the LCD DMA operation timing for reading data from the display RAM.

If HDMA is asserted immediately before the DMA operation for the LCD (LCD DMA operation 1) is started, this operation must wait until HDMA is finished before it can be performed (LCD DMA operation 2).

Taking the above into account, it is necessary to ensure that each LCD DMA operation is finished before the next LCD driver output is started.

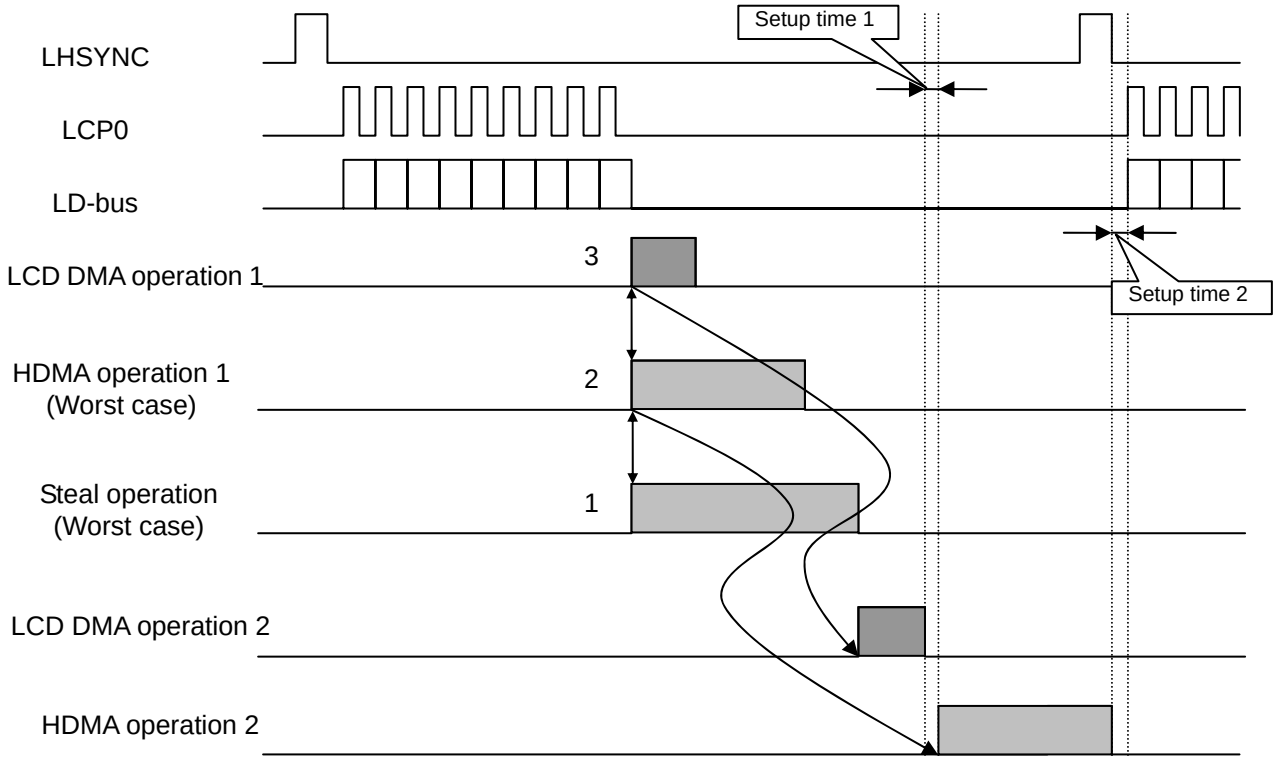


Figure 3.27.2 Example of Data Bus Occupancy Timing in Debug Mode

Figure 3.27.2 shows an example of data bus occupancy timing in debug mode. If the steal program issues a wait request immediately before the DMA operation for the LCD (LCD DMA operation 1) and HDMA (HDMA operation 1) are asserted, these operations must wait until the steal program is finished before they can be performed. (LCD DMA is given a higher priority than HDMA in bus arbitration. This means that bus requests is

sued for LCD DMA and HDMA while the steal program is running are processed in the order of LCD and HDMA (LCD DMA operation 2 → HDMA operation 2) regardless of the order in which they are issued.)

Taking the above into account, it is necessary to ensure that each LCD DMA or HDMA operation is finished before the next LCD driver output is started.

In other words, to avoid abnormal operation in debug mode, the maximum duration of HDMA operation time must be set so that it does not interfere with LCD DMA operation. Alternatively, the LHSYNC period should be adjusted to accommodate a wait request by the steal program (80 times of transmission for debug clock: LH_SYNCLK), although this slightly reduces the LCD display quality.

4. Electrical Characteristics (Tentative)

4.1 Maximum Ratings

Symbol	Contents	Rating	Unit
DVCC3A DVCC3B	Power Supply Voltage	-0.3 to 3.9	V
DVCC1A DVCC1B DVCC1C		-0.3 to 3.0	
AVCC		-0.3 to 3.9	
V_{IN}	Input Voltage	-0.3 ～DVCC3A/3B+0.3 (Note1) -0.3 to AVCC + 0.3 (Note2)	V
IOL	Output Current (1pin)	15	mA
IOH	Output Current (1pin)	-15	mA
Σ_{IOL}	Output Current (total)	80	mA
Σ_{IOH}	Output Current (total)	-50	mA
P_D	Power Dissipation (Ta = 85°C)	600	mW
T_{SOLDER}	Soldering Temperature (10s)	260	°C
T_{STG}	Storage Temperature	-65 to 150	°C
T_{OPR}	Operation Temperature	-0 to 70	°C
T_{OPR}	Operation Temperature (80MHz)	-0 to 50	°C

Note1: If setting it, don't exceed the Maximum Ratings of DVCC3A (PV port and PW port are DVCC3B).

Note2: In PG0 to PG5, P96,P97,VREFH,VREFL maximum ratings for AVCC is applied.

Note3: The 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 maximum rating is exceeded, a 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 designing products that include this device, ensure that no maximum rating value will ever be exceeded.

Point of note about solderability of lead free products (attach "G" to package name)

Test parameter	Test condition	Note
Solderability	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%
	Solder bath temperature =245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux (use of lead free)	

4.2 DC Electrical Characteristics

Symbol	Parameter	Min	Typ.	Max	Unit	Condition	
DVCC 3A	General I/O Power Supply Voltage (DVCC=AVCC) (DVSSCOM=AVSS=0V)	3.0	3.3	3.6	V	X1=6 to 10MHz CPU CLK (60MHz) CPU CLK (80MHz)	XT1=30 to 34KHz
DVCC 1A	Internal Power A	1.4	1.5	1.6	V		
DVCC 1B	Internal Power B						
DVCC 1C	High CLK oscillator and PLL Power						
VIL0	Input Low Voltage for D0 to D7 P10 to P17 (D8 to 15), P60 to P67 P71 to P76, P90 PC4 to PC7, PF0 to PF5 PG0 to PG5, PJ5 to PJ6 PN0 to PN7, PP1 to PP2 PR0 to PR3, PT0 to PT7 PU0 to PU7, PX5, PX7	-0.3	–	0.3×DVCC3A	V	3.0≦DVCC3A≦3.6	
VIL1	Input Low Voltage for PV0 to PV2, PV6 to PV7, PW0 to PW7		–	0.3×DVCC3B		3.0≦DVCC3B≦3.6	
VIL2	Input Low Voltage for P91 to P92, P96 to P97, PA0 to PA7 PC0 to PC3, PP3 to PP5, PZ0 to PZ7, $\overline{\text{RESET}}$		–	0.25× DVCC3A		3.0≦DVCC3A≦3.6	
VIL3	Input Low Voltage for AM0 to AM1, $\overline{\text{DBGE}}$		–	0.1×DVCC3A		3.0≦DVCC3A≦3.6	
VIL4	Input Low Voltage for X1		–	0.1×DVCC1C		1.4≦DVCC1C≦1.6	
VIL5	Input Low Voltage for XT1		–	0.15 ×DVCC3A		3.0≦DVCC3A≦3.6	

Note: Above power supply range is premised that all power supply of same system is equal.
(DVCC1A = DVCC1B = DVCC1C or DVCC3A = DVCC3B=AVCC)

Symbol	Parameter	Min	Typ.	Max	Unit	Condition
VIH0	Input High Voltage for D0 to D7 P10 to P17 (D8 to 15), P60 to P67 P71 to P76, P90 PC4 to PC7, PF0 to PF5 PG0 to PG5, PJ5 to PJ6 PN0 to PN7, PP1 to PP2 PR0 to PR3, PT0 to PT7 PU0 to PU7, PX5, PX7	0.7 × DVCC3A	–	DVCC3A + 0.3	V	3.0≦DVCC3A≦3.6
VIH1	Input High Voltage for PV0 to PV2, PV6 to PV7, PW0 to PW7	0.7 × DVCC3B	–	DVCC3B + 0.3		3.0≦DVCC3B≦3.6
VIH2	Input High Voltage for P91 to P92, P96 to P97, PA0 to PA7 PC0 to PC3, PP3 to PP5, PZ0 to PZ7, $\overline{\text{RESET}}$	0.75× DVCC3A	–	DVCC3A + 0.3		3.0≦DVCC3A≦3.6
VIH3	Input High Voltage for AM0 to AM1 , $\overline{\text{DBGE}}$	0.9× DVCC3A	–	DVCC3A + 0.3		3.0≦DVCC3A≦3.6
VIH4	Input High Voltage for X1	0.9×DVCC1C	–	DVCC1C + 0.3		1.4≦DVCC1C≦1.6
VIH5	Input High Voltage for XT1	0.85× DVCC3A	–	DVCC3A + 0.3		3.0≦DVCC3A≦3.6

Symbol	Parameter	Min	Typ.	Max	Unit	Condition	
VOL1	Output Low Voltage1 P90 to P92, PC0 to PC3, PC7 PF0 to PF5, PK1 to PK7 PM1 to PM2, PM7 PN0 to PN7, PP1 to PP7 PV0 to PV7, PW0 to PW7, PX5, PX7	–	–	0.4	V	IOL = 0.5mA, 3.0≦DVCC3A	
VOL2	Output Low Voltage2 Except VOL1 output pin					IOL = 2mA, 3.0≦DVCC3A	
VOH1	Output High Voltage1 P90 to P92, PC0 to PC3, PC7 PF0 to PF7, PK1 to PK7 PM1 to PM2, PM7 PN0 to PN7, PP1 to PP7 PV0 to PV7, PW0 to PW7 PX5, PX7	2.4	–	–		IOH = -0.5mA, 3.0≦DVCC3A	
VOH2	Output High Voltage2 Except VOL1 output pin					IOH = -2mA, 3.0≦DVCC3A	
VOL(T)	Output Low Voltage for P96(PX), P97(PY)-pins	–	–	0.2		IOL(T)=6.6mA	3.0≦DVCC3A≦3.6
VOH(T)	Output High Voltage for P96(PX), P97(PY)-pins	VCC-0.2	–	–		IOH(T)=-6.6mA	
ILI	Input Leakage Current	–	0.02	±5	μA	0.0 ≦ Vin ≦ DVCC3A	
ILO	Output Leakage Current	–	0.05	±10	μA	0.2 ≦ Vin ≦ DVCC3A-0.2V	
RRST	Pull Up/Down Resistor for $\overline{\text{RESET}}$, PA0 to PA7, P96	30	50	70	KΩ		
CIO	Pin Capacitance	–	–	10	pF	fc=1MHz	
VTH	Schmitt Width for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PP3 to PP5, PZ0 to PZ7, $\overline{\text{RESET}}$	0.6	0.8	1.0	V	3.0≦DVCC3A≦3.6	

Note1 : Typical values are value that when Ta = 25°C and Vcc = 3.3 V unless otherwise noted.

Note2 : This data shows exept "debug mode"

Symbol	Parameter	Min	Typ.	Max	Unit	Condition	
ICC	NORMAL (note2)	–	15	30	mA	PLL_ON f_{SYS}=80MHz	DVCC3A,3B = 3.6V
			45	60			DVCC1A,1B,1C = 1.6V
	IDLE2	–	0.5	1			DVCC3A,3B = 3.6V
			28	45			DVCC1A,1B,1C = 1.6V
	NORMAL (note2)	–	12	23		PLL_ON f_{SYS}=60MHz	DVCC3A,3B = 3.6V
			34	45			DVCC1A,1B,1C = 1.6V
	IDLE2	–	0.4	0.8			DVCC3A,3B = 3.6V
			21	34			DVCC1A,1B,1C = 1.6V
	IDLE1	–	12	45	μA	PLL_OFF f_{SYS} =10MHz	DVCC3A,3B = 3.6V
			200	3200			DVCC1A,1B,1C = 1.6V
	Power Cut Mode (WITH PMC function)	–	6	35	μA	Ta ≦ 70℃	DVCC3A=3.6V DVCC3B=3.6V AVCC=3.6V
				30		Ta ≦ 50℃	
			2	50		Ta ≦ 70℃	DVCC1A=0V DVCC1B=1.6V DVCC1C=0V XT=32KHz X=OFF
				35		Ta ≦ 50℃	
	STOP	–	6	35		Ta ≦ 70℃	DVCC3A=3.6V DVCC3B=3.6V AVCC3.6V
				30		Ta ≦ 50℃	
			200	800		Ta ≦ 70℃	DVCC1A=1.6V DVCC1B=1.6V DVCC1C=1.6V XT=OFF X=OFF
				600		Ta ≦ 50℃	

Note1 : Typical values are value that when Ta = 25°C and Vcc = 3.3 V unless otherwise noted.

Note2 : ICC measurement conditions (NORMAL, SLOW):

All functions are operational; output pins except bus pin are open, and input pins are fixed. Bus pin CL=50pF

(Access toexternal memory at 8-waitsetting)

Note3: This data shows exept "debug mode"

4.3 AC Characteristics

The Following all AC regulation is the measurement result in following condition, if unless otherwise noted.

AC measuring condition

- Clock of top column in above table shows system clock frequency, and "T" shows system clock period [ns].
- Output level: High = $0.7 \times 3AV_{CC}$, Low = $0.3 \times 3AV_{CC}$
- Input level: High = $0.9 \times 3AV_{CC}$, Low = $0.1 \times 3AV_{CC}$

Note: In table, "Variable" shows the regulation at DVCC3A=3.0V~3.6V, DVCC1A=DVCC1B=DVCC1C=1.4~1.6V.

4.3.1 Basic Bus Cycle

Read cycle

No.	Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
			Min	Max			
1	OSC period (X1/X2)	t_{OSC}	100	166.6	–	–	ns
2	System clock period (= T)	t_{CYC}	12.5	266.6	12.5	16.6	
3	SDCLK low width	t_{CL}	0.5T – 3		3.25	5.3	
4	SDCLK high width	t_{CH}	0.5T – 3		3.25	5.3	
5-1	A0 ~ A23 valid → D0 ~ D15 input at 0 waits	t_{AD}		2.0T – 18.0	7	15.3	
5-2	A0 ~ A23 valid → D0 ~ D15 input at 4 waits/6 waits	t_{AD6}		6.0T – 18.0	–	82	
		t_{AD7}		8.0T - 18.0	82	–	
6-1	$\overline{RD}$ falling → D0 ~ D15 input at 0 waits	t_{RD}		1.5T – 18.0	–	7	
		t_{RD}		1.5T – 18.0	0.75	–	
6-2	$\overline{RD}$ falling → D0 ~ D15 input at 4 waits/6waits	t_{RD6}		5.5T – 18.0	–	73.6	
		t_{RD7}		5.5T – 18.0	50.75	–	
7-1	$\overline{RD}$ low width at 0 waits	t_{RR}	1.5T – 10		8.75	14.9	
7-2	$\overline{RD}$ low width at 4 waits	t_{RR6}	5.5T – 10		58.75	81.3	
8	A0 ~ A23 valid → $\overline{RD}$ falling	t_{AR}	0.5T – 5		1.25	3.3	
9	$\overline{RD}$ falling → SDCLK rising	t_{RK}	0.5T – 5		1.25	3.3	
10	A0 ~ A23 valid → D0 ~ D15 hold	t_{HA}	0		0	0	
11	$\overline{RD}$ rising → D0 ~ D15 hold	t_{HR}	0		0	0	
12	$\overline{WAIT}$ setup time	t_{TK}	3		3	5	
13	$\overline{WAIT}$ hold time	t_{KT}	2		2	3	
14	Data byte control access time	t_{SBA}		1.5T – 18.0	0.75	7	
15	$\overline{RD}$ high width	t_{RRH}	0.5T – 5		1.25	3.3	

AC measuring condition

- Data_bus, Address_bus, various function control signal capacitance C_L = 50 pF

Note: The operation guarantee temprature: 80MHz: Ta=0~50°C, less than 60MHz: Ta=0~70°C

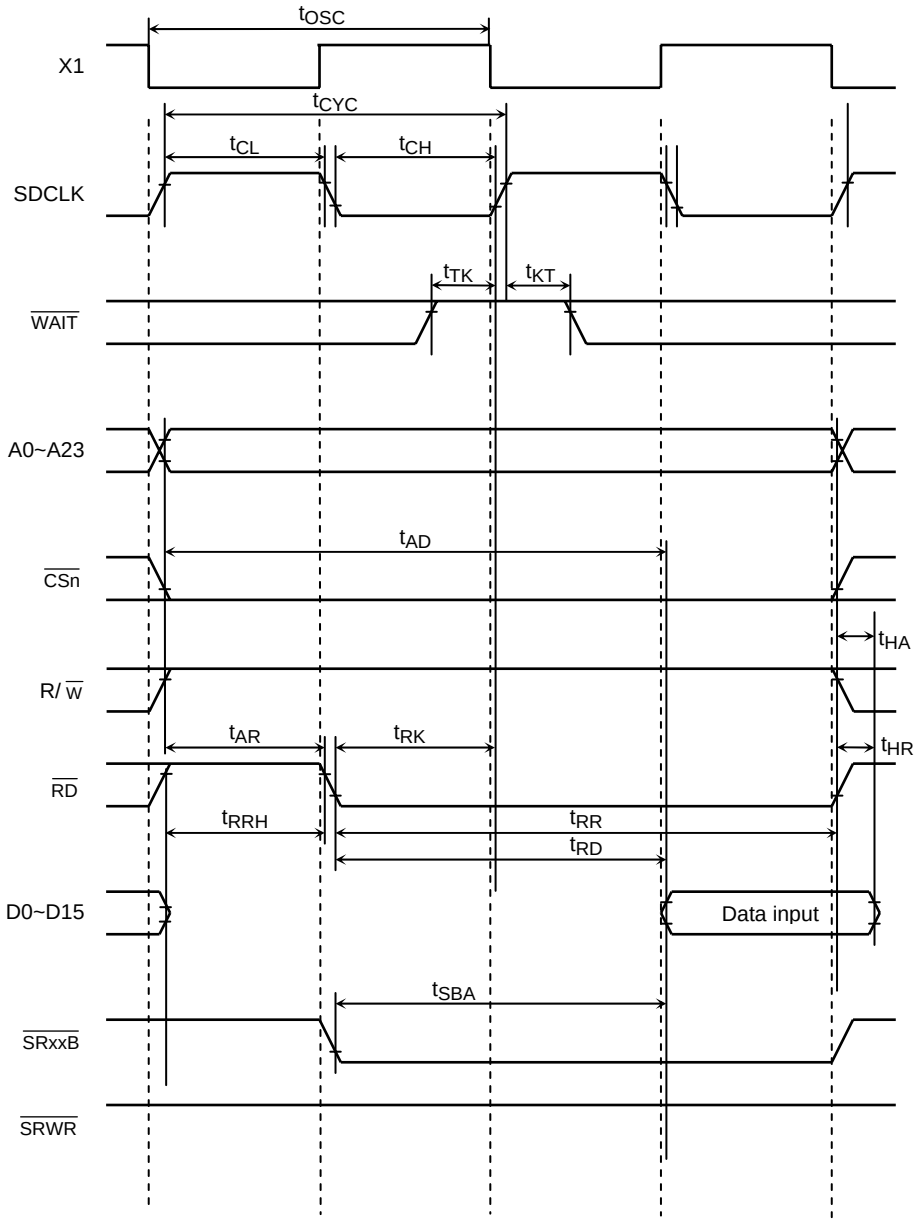
Write cycle

No.	Parameter	Symbol	Variable		80MHz	60MHz	Unit
			Min	Max			
16-1	D0 ~ D15 valid → $\overline{WR}$ xx rising at 0 waits	t_{DW}	1.0T – 10.0		–	6.6	ns
		t_{DW}	1.0T – 6.0		6.5	–	
16-2	D0 ~ D15 valid → $\overline{WR}$ xx rising at 2 waits/4 waits	t_{DW4}	3.0T – 10.0		–	39.8	
		t_{DW6}	5.0T – 6.0		56.5	–	
17-1	$\overline{WR}$ xx low width at 0 waits	t_{WW}	1.0T – 7.0		–	9.6	
		t_{WW}	1.0T – 4.0		8.5	–	
17-2	$\overline{WR}$ xx low width at 2 waits/4 waits	t_{WW4}	3.0T – 7.0		–	42.8	
		t_{WW6}	5.0T – 4.0		58.5	–	
18	A0 ~ A23 valid → $\overline{WR}$ falling	t_{AW}	0.5T – 5.0		1.25	3.3	
19	$\overline{WR}$ xx falling → SDCLK rising	t_{WK}	0.5T – 5.0		1.25	3.3	
20	$\overline{WR}$ xx rising → A0 ~ A23 hold	t_{WA}	0.5T – 5.0		1.25	3.3	
21	$\overline{WR}$ xx rising → D0 ~ D15 hold	t_{WD}	0.5T – 5.0		1.25	3.3	
22-1	$\overline{RD}$ rising → D0 ~ D15 output	t_{RDO}	0.5T – 2.0		–	6.3	
		t_{RDO}	0.5T – 1.0		5.25	–	
22-2	$\overline{RD}$ rising → D0 ~ D15 output	t_{RDO}	1.5T – 2.0		–	22.9	
		t_{RDO}	2.5T – 1.0		30.25	–	
23	Write width for SRAM	t_{SWP}	1.0T – 7.0		–	9.6	
		t_{SWP}	1.0T – 4.0		8.5	–	
24	Data byte control ~ end of write for SRAM	t_{SBW}	1.0T – 7.0		–	9.6	
		t_{SBW}	1.0T – 4.0		8.5	–	
25	Address setup time for SRAM	t_{SAS}	0.5T – 5.0		1.25	3.3	
26	Write recovery time for SRAM	t_{SWR}	0.5T – 5.0		1.25	3.3	
27	Data setup time for SRAM	t_{SDS}	1.0T – 10.0		–	6.6	
		t_{SDS}	1.0T – 6.0		6.5	–	
28	Data hold time for SRAM	t_{SDH}	0.5T – 5.0		1.25	3.3	

AC measuring condition

Note: The operation guarantee Temperature: 80MHz: Ta=0~50°C, less than 60MHz: Ta=0~70°C

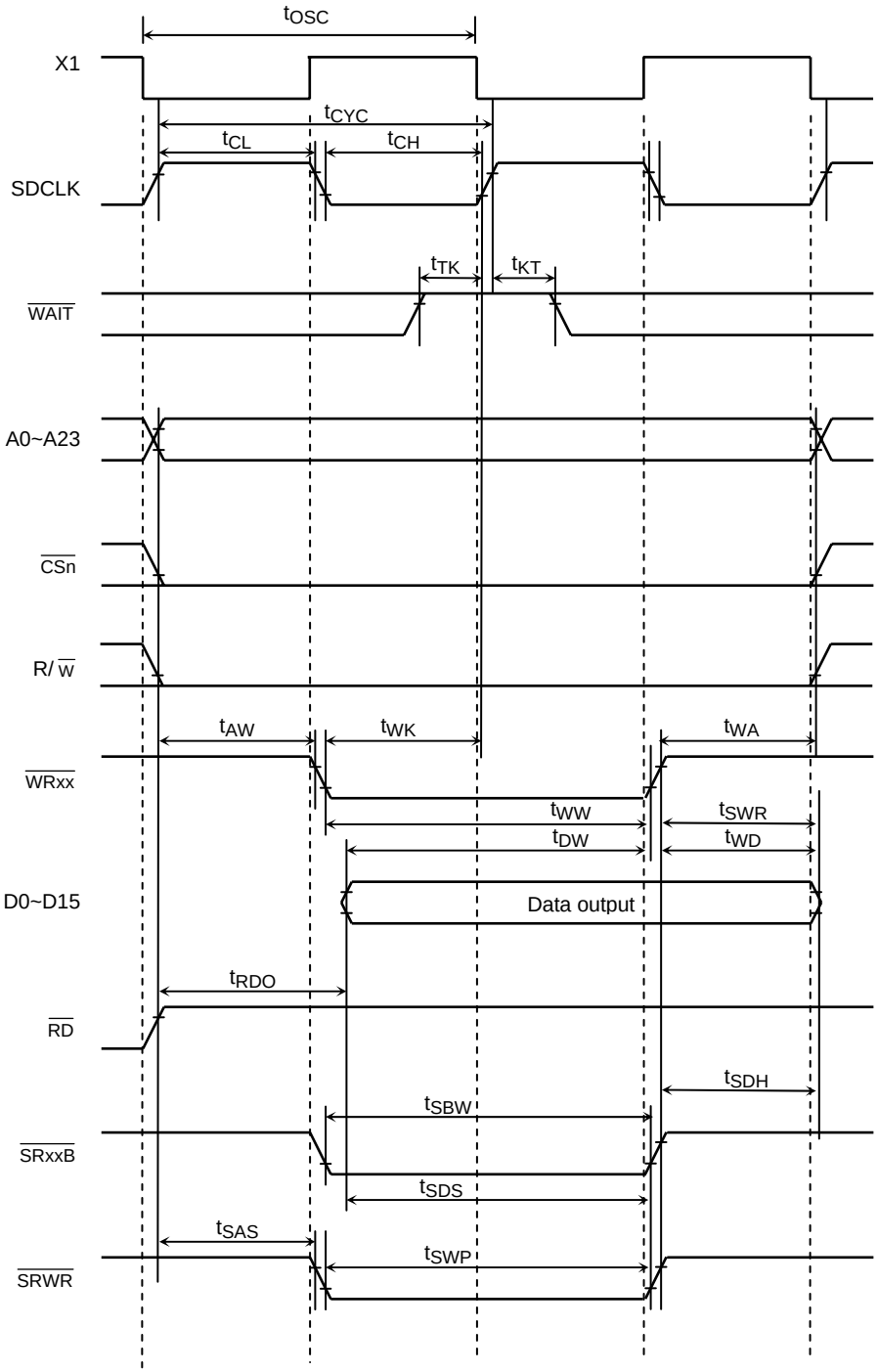
(1) Read cycle (0 waits)



Note1: The phase relation between X1 input signal and the other signals is undefined.

Note2: The above timing chart show an example of basic bus timing. The $\overline{CSn}$, R/$\overline{W}$, $\overline{RD}$, $\overline{WRxx}$, $\overline{SRxxB}$, $\overline{SRWR}$ pins timing can be adjusted by memory controller timing adjust function.

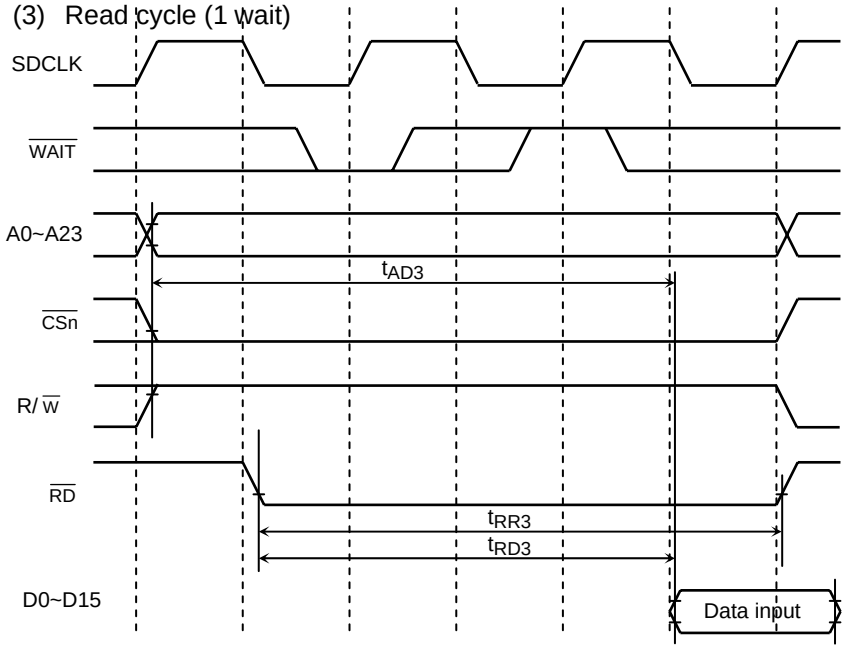
(2) Write cycle (0 waits)



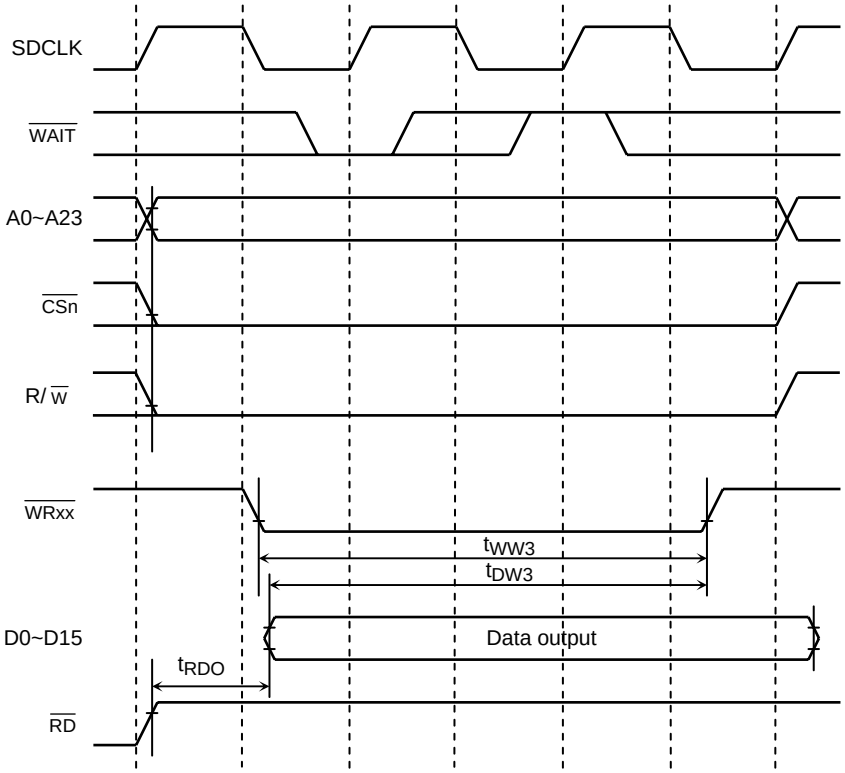
Note1: The phase relation between X1 input signal and the other signals is undefined.

Note2: The above timing chart show an example of basic bus timing. The $\overline{CSn}$, R/$\overline{W}$, $\overline{RD}$, $\overline{WRxx}$, $\overline{SRxxB}$, $\overline{SRWR}$ pins timing can be adjusted by memory controller timing adjust function.

(3) Read cycle (1 wait)



(4) Write cycle (1 wait)



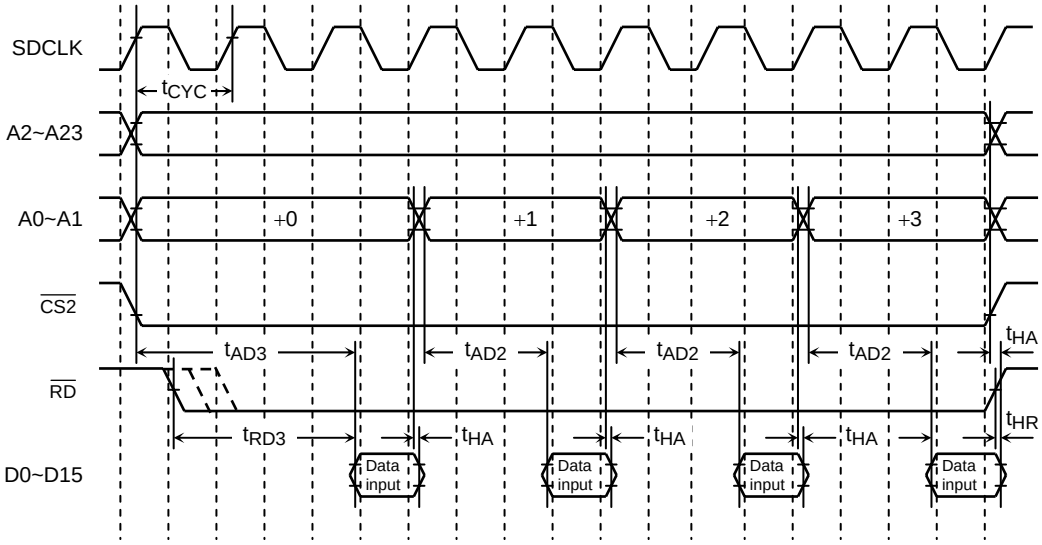
4.3.2 Page ROM Read Cycle

(1) 3-2-2-2 mode

	Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
			Min	Max			
1	System clock period (= T)	t_{CYC}	12.5	266.6	12.5	16.6	ns
2	A0, A1 → D0 ~ D15 input	t_{AD2}		2.0T – 18	7	15.2	
3	A2 ~ A23 → D0 ~ D15 input	t_{AD3}		3.0T – 18	19.5	31.8	
4	$\overline{RD}$ falling → D0 ~ D15 input	t_{RD3}		2.5T – 18	13	24	
5	A0 ~ A23 Invalid → D0 ~ D15 hold	t_{HA}	0		0	0	
6	$\overline{RD}$ rising → D0 ~ D15 hold	t_{HR}	0		0	0	

AC measuring condition

Note: The (a), (b) and (c) of “Symbol” in above table depend on the falling timing of $\overline{RD}$ pin. The falling timing of $\overline{RD}$ pin is set by MEMCR0<RDTMG1:0> in memory controller. If MEMCR0<RDTMG1:0> is set to “00”, it correspond with (a) in above table, and “01” is (b), “10” is (c).



4.3.3 SDRAM controller AC Characteristics

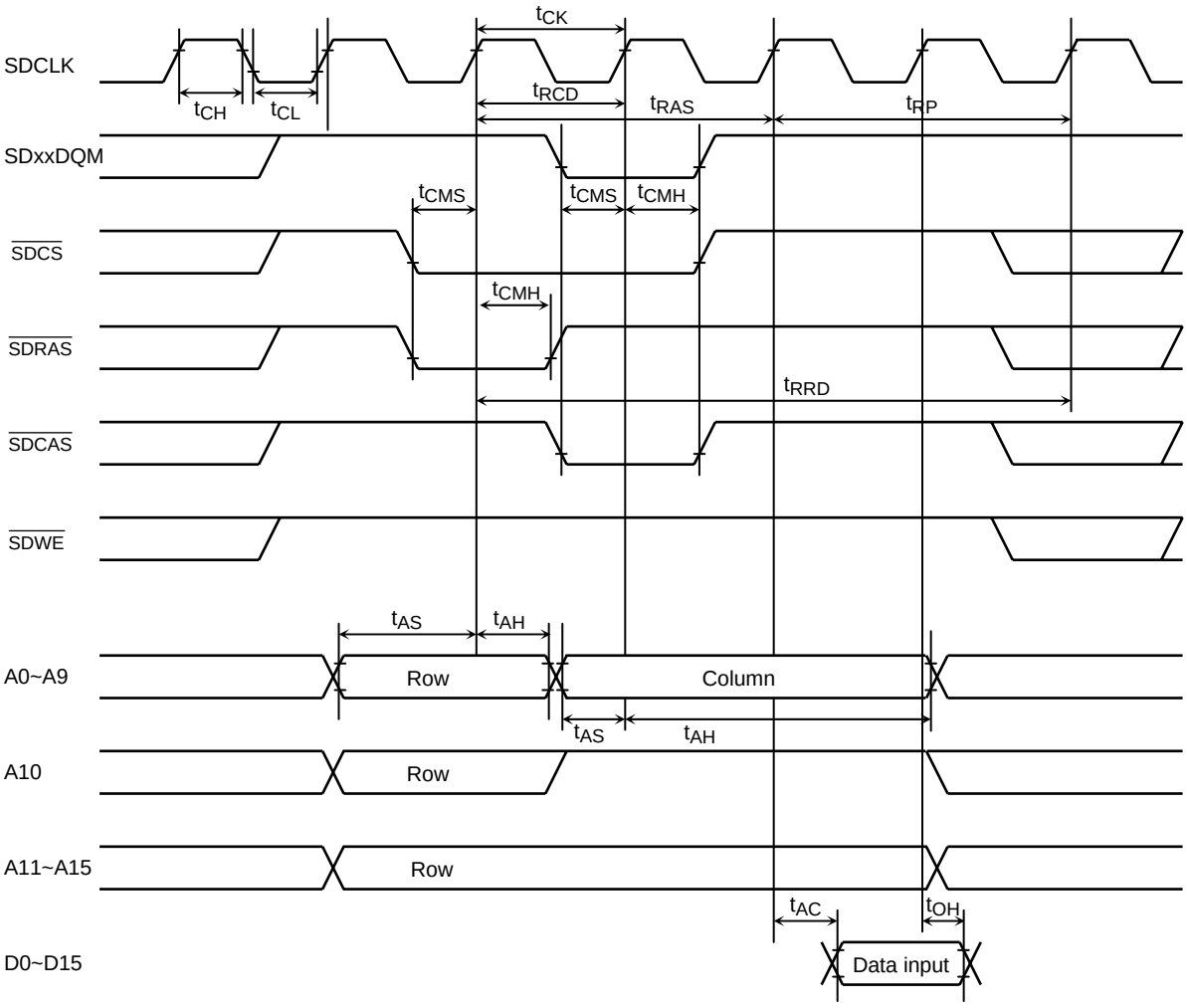
	Parameter		Symbol	Variable		80 MHz	60 MHz	Unit
				Min	Max			
1	Ref/Active to ref/active command period	<STRC[2:0]>=000	t_{RC}	T		12.5	16.6	ns
		<STRC[2:0]>=110		7T		87.5	116.2	
2	Active to precharge command period	<STRC[2:0]>=000	t_{RAS}	2T	12210	25.0	33.2	
		<STRC[2:0]>=110		7T		87.5	116.2	
3	Active to read/write command delay time	<STRCD>=0	t_{RCD}	T		12.5	16.6	
		<STRCD>=1		2T		25.0	33.2	
4	Precharge to active command period	<STRP>=0	t_{RP}	T		12.5	16.6	
		<STRP>=1		2T		25.0	33.2	
5	Active to active command period	<STRC[2:0]>=000	t_{RRD}	3T		37.5	49.8	
		<STRC[2:0]>=110		7T		87.5	116.2	
6	Write recovery time	<STWR>=0	t_{WR}	T		12.5	16.6	
		<STWR>=1		2T		25.0	33.2	
7	CLK cycle time		t_{CK}	T		12.5	16.6	
8	CLK high level width		t_{CH}	0.5T – 5		–	3.3	
				0.5T – 3		3.25	–	
9	CLK low level width		t_{CL}	0.5T – 5		–	3.3	
				0.5T – 3		3.25	–	
10-1a	Access time from CLK(CL* =2)		t_{AC}		T – 16	–	0.6	
10-1b	<SRDS>=0(Read data shift OFF)				T – 16	- 3.5	–	
10-2a	Access time from CLK(CL* =2)		t_{AC}		T – 6.5	–	10.1	
10-2b	<SRDS>=1(Read data shift ON)				T – 6.5	6	–	
11	Output data hold time		t_{OH}	0		0	0	
12	Data-in set-up time	1Word/Single	t_{DS}	0.5T – 4		2.25	3.3	
		Burst	t_{DS}	0.5T – 4		2.25	3.3	
13	Data-in hold time	1Word/Single	t_{DH}	T – 10		2.5	6.6	
		Burst	t_{DH}	0.5T – 6		–	2.3	
			t_{DH}	0.5T – 4		2.25	–	
14	Address set-up time		t_{AS}	0.5T – 4		2.25	4.3	
15	Address hold time		t_{AH}	0.5T – 6		–	2.3	
				0.5T – 4		2.25	–	
16	CKE set-up time		t_{CKS}	0.5T – 5		–	3.3	
				0.5T – 3		3.25	–	
17	Command set-up time		t_{CMS}	0.5T – 5		–	3.3	
				0.5T – 3		3.25	–	
18	Command hold time		t_{CMH}	0.5T – 6		–	2.3	
				0.5T – 4		2.25	–	
19	Mode register set cycle time		t_{RSC}	T		12.5	16.6	

*CL: CAS latency

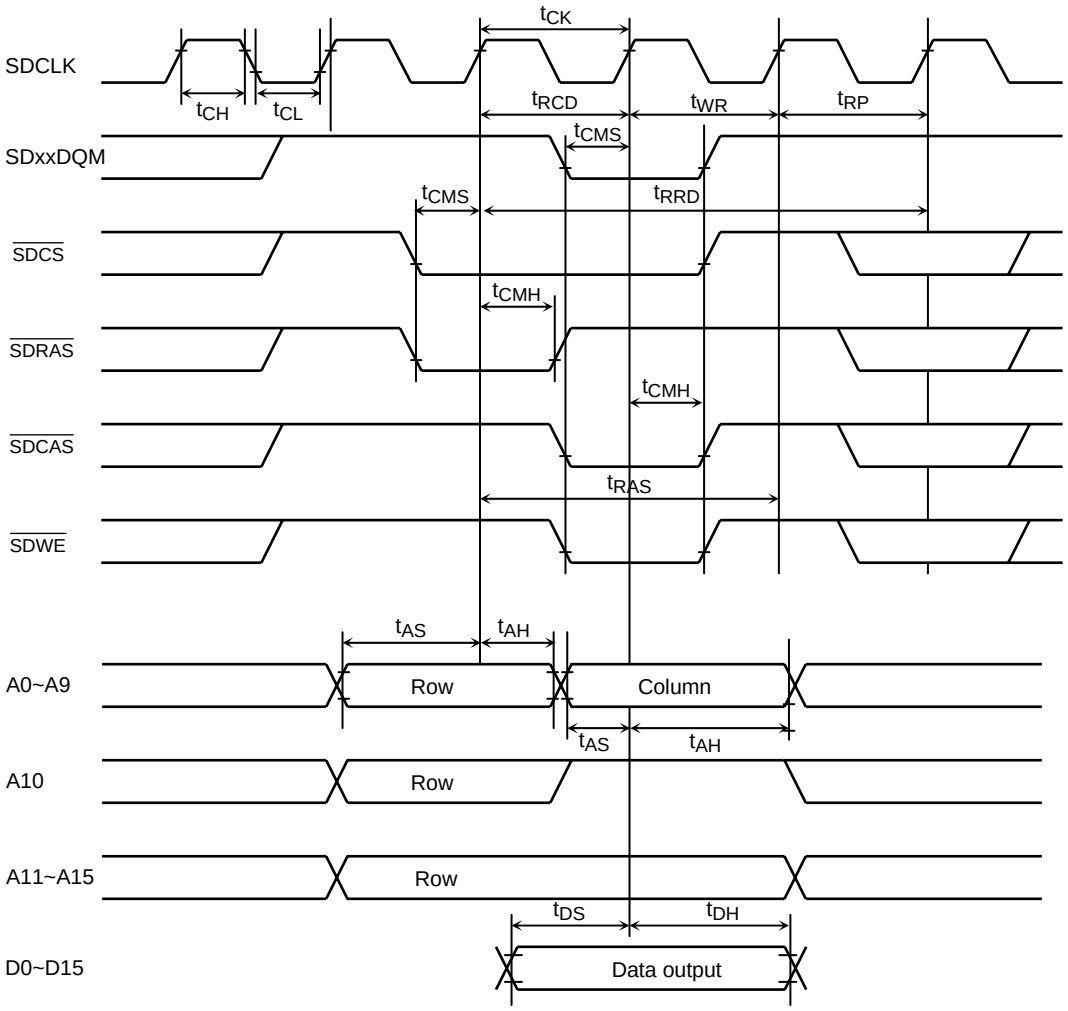
AC measuring condition

SDCLK pin $C_L = 30$ pF, Other pins $C_L = 50$ pF

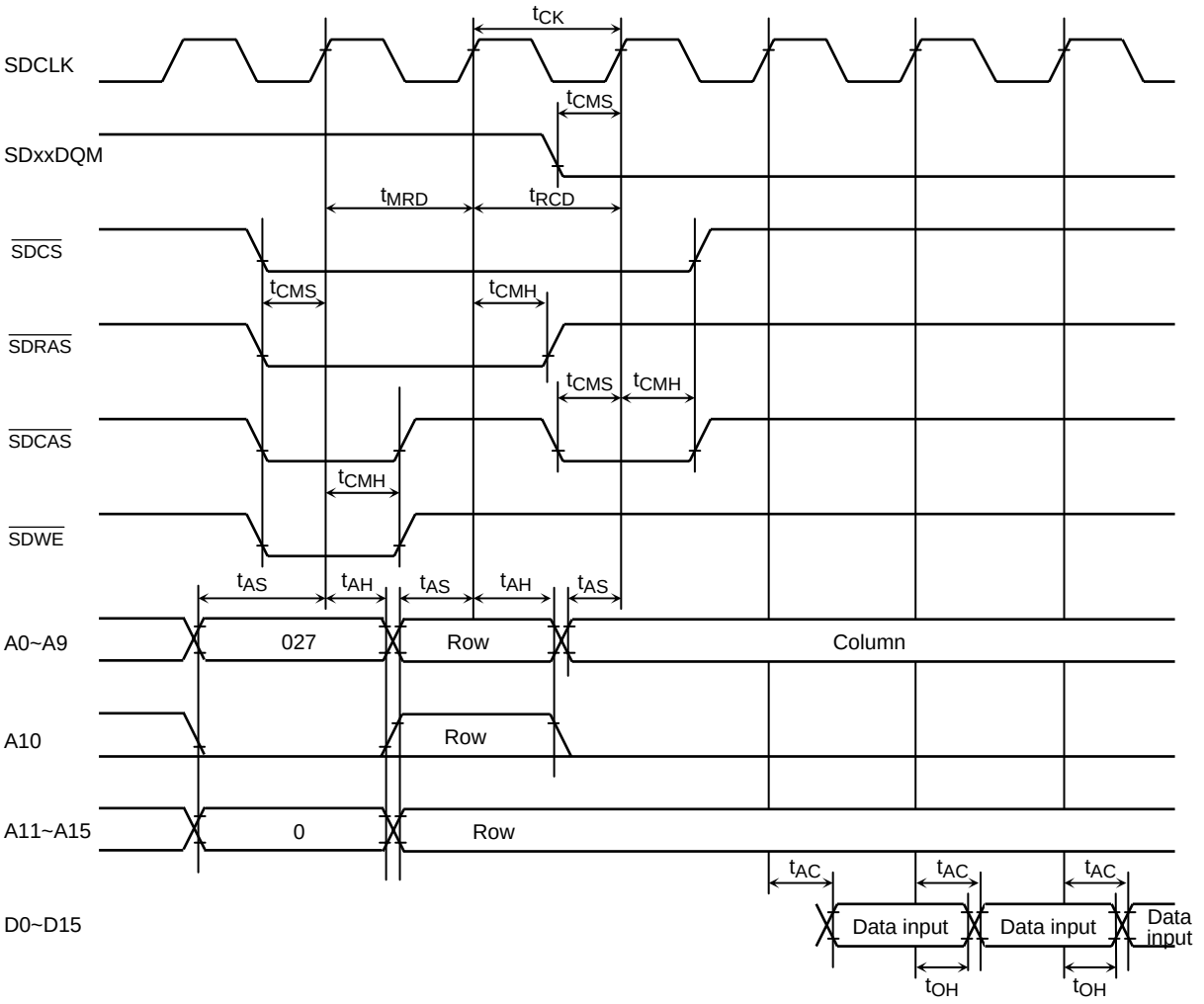
(1) SDRAM read timing (1Word length read mode, <SPRE>=1)



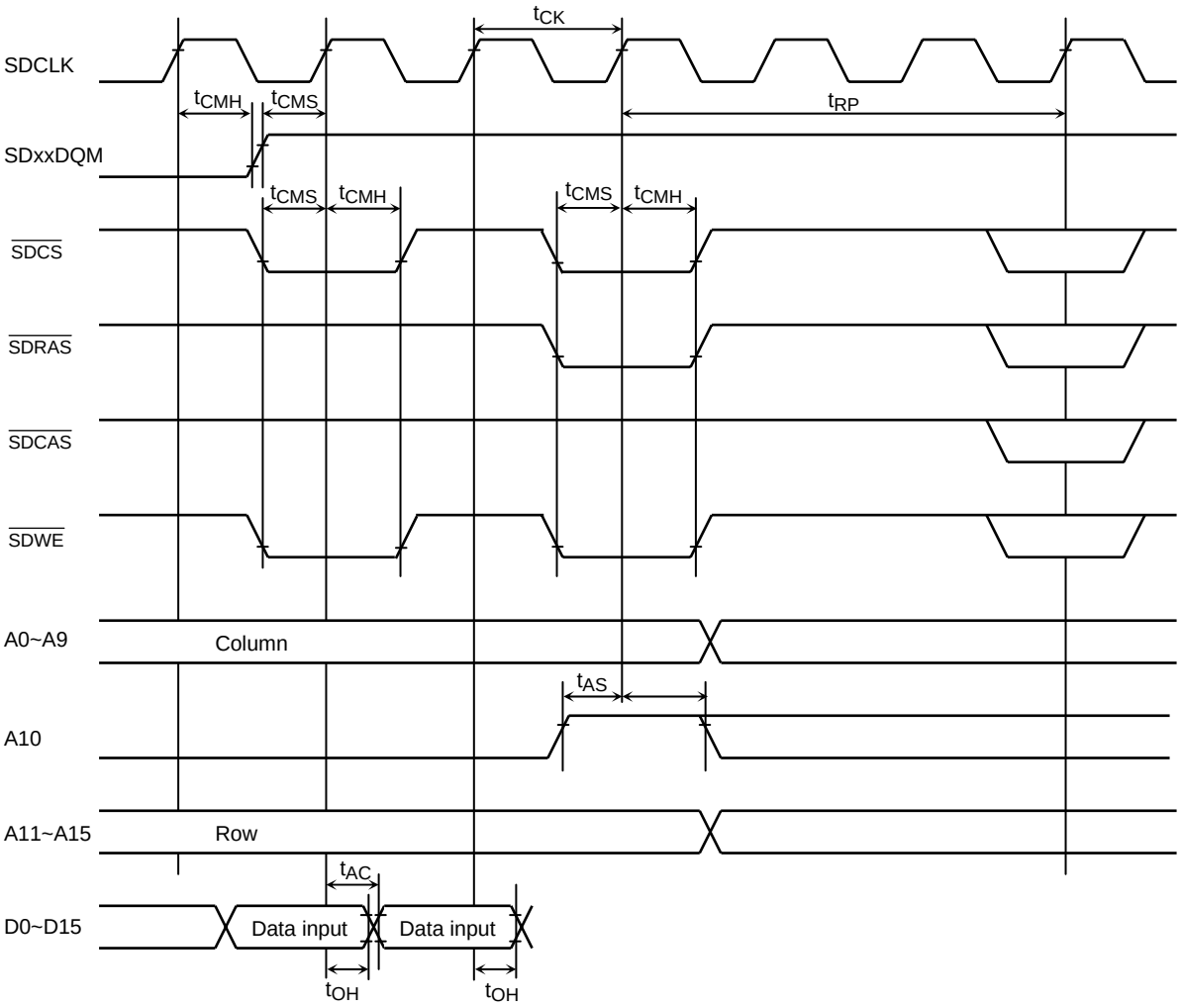
(2) SDRAM write timing (Single write mode, <SPRE>=1)



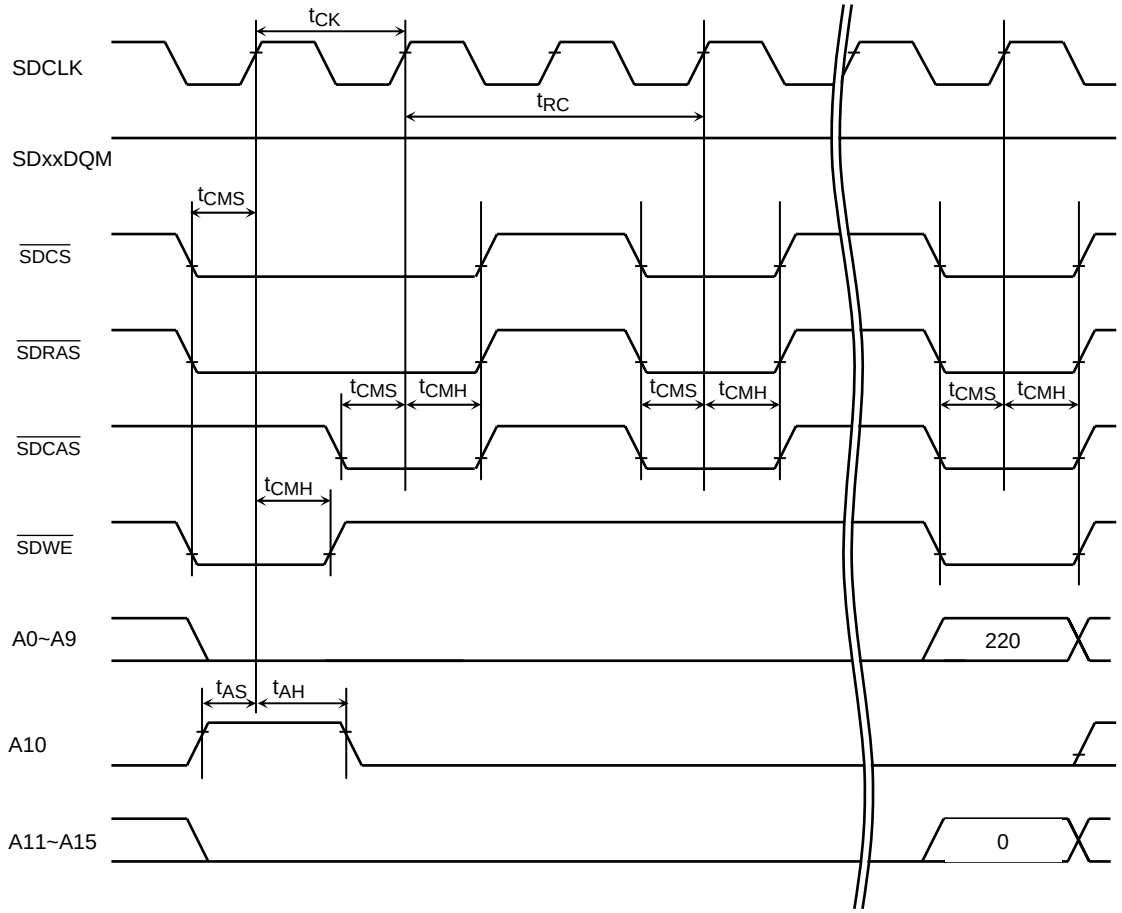
(3) SDRAM burst read timing (Start burst cycle)



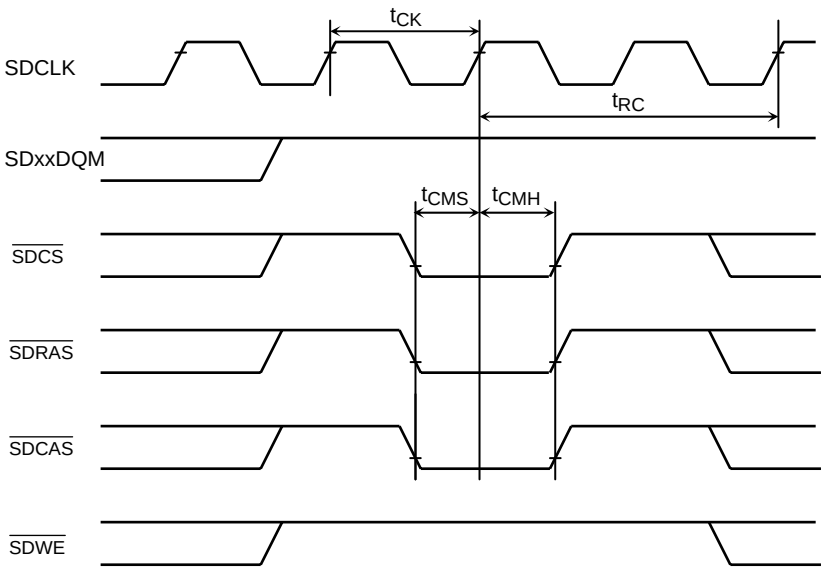
(4) SDRAM burst read timing (End burst timing)



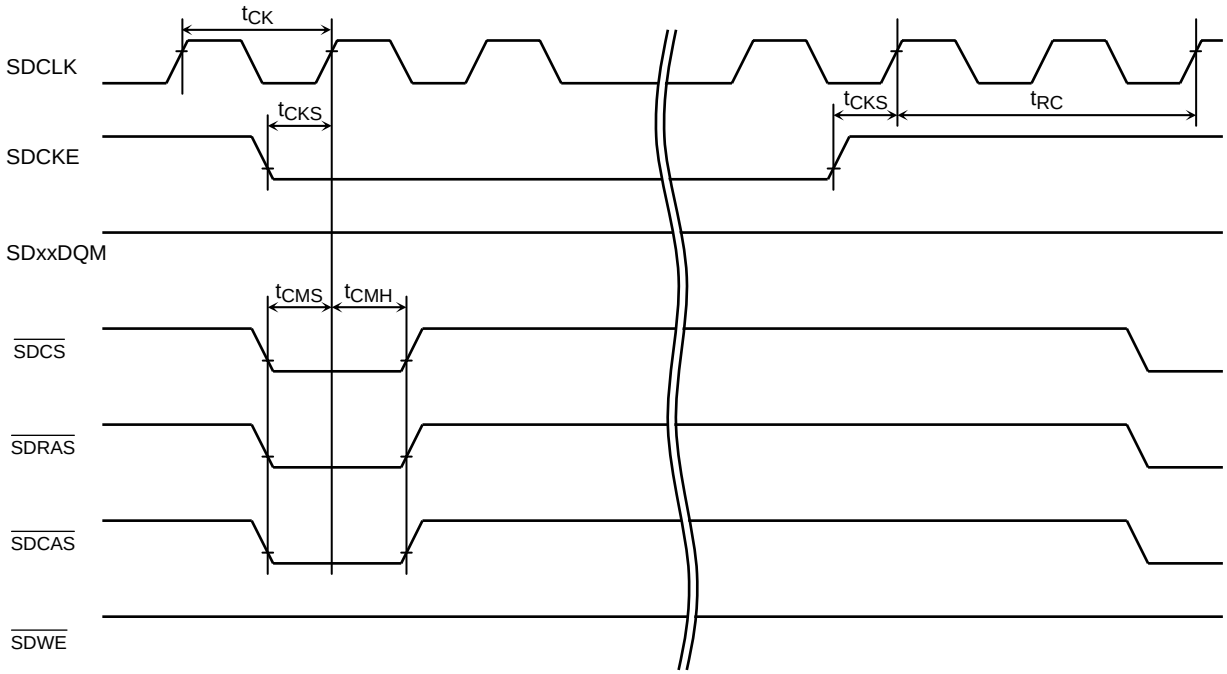
(5) SDRAM initializes timing



(6) SDRAM refreshes timing



(7) SDRAM self refresh timing



4.3.4 NAND Flash Controller AC Characteristics

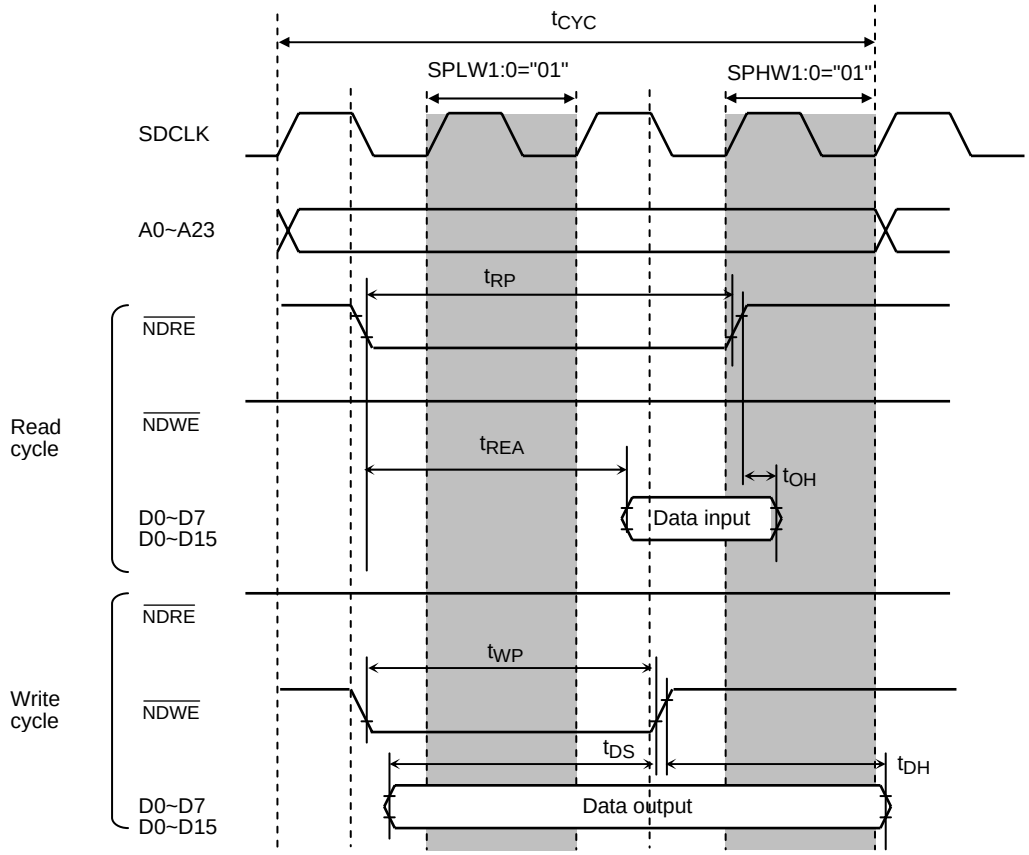
No.	Symbol	Parameter	Variable		80 MHz (n=3) (m=3)	60 MHz (n=3) (m=3)	Unit
			Min	Max			
1	t_{NC}	Access cycle	(2 + n + m) T		100	132	ns
2	t_{RP}	$\overline{NDRE}$ low level width	(1.5+ n) T – 12		45	63	
3	t_{REA}	$\overline{NDRE}$ data access time		(1.5 + n) T – 15	41	60	
4	t_{OH}	Read data hold time	0		0	0	
5	t_{WP}	$\overline{NDWE}$ low level width	(1.0 + n) T – 20		30	47	
6	t_{DS}	Write data setup time	(1.0 + n) T – 20		30	47	
7	t_{DH}	Write data hold time	(0.5 + m) T – 2		42	56	

AC measuring condition

Note1: The “n” in “Variable” means wait-number which is set to NDFMCR0<SPLW1:0>, and “m” means number which is set to NDFMCR0<SPHW1:0>.

Example: If NDFMCR0<SPLW1:0> is set to “01”, n=1, t_{RP} = (1.5 + n) T – 12 = 2.5T – 12

Note2: In above variable, the setting that result is minus can not use.



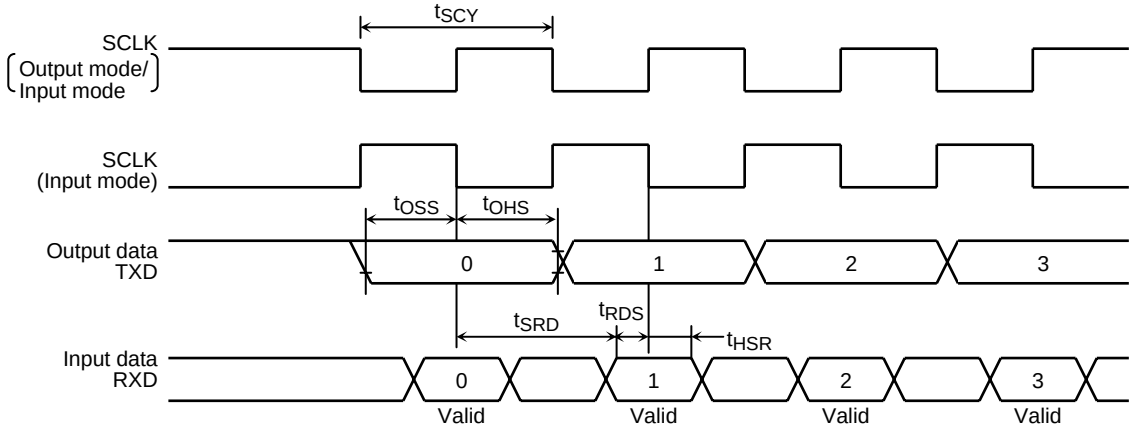
4.3.5 Serial channel timing

(1) SCLK input mode (I/O interface mode)

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
SCLK cycle	t_{SCY}	16T		200	266	ns
Output data → SCLK rising/ falling	t_{OSS}	$t_{SCY}/2 - 4T - 30$		20	36.4	
SCLK rising/ falling → Output data hold	t_{OHS}	$t_{SCY}/2 + 2T - 20$		105	146	
SCLK rising/ falling → Input data hold	t_{HSR}	$2T + 10$		35	43	
SCLK rising/ falling → Input data valid	t_{SRD}		$t_{SCY} - 20$	180	246	
Input data valid → SCLK rising/ falling	t_{RDS}	20		20	20	

(2) SCLK output mode (I/O interface mode)

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
SCLK cycle (Programmable)	t_{SCY}	16T	8192T	200	266	ns
Output data → SCLK rising/ falling	t_{OSS}	$t_{SCY}/2 - 40$		60	93	
SCLK rising/ falling → Output data hold	t_{OHS}	$t_{SCY}/2 - 40$		60	93	
SCLK rising/ falling → Input data hold	t_{HSR}	0		0	0	
SCLK rising/ falling → Input data valid	t_{SRD}		$t_{SCY} - 1T - 50$	137.5	199	
Input data valid → SCLK rising/ falling	t_{RDS}	$1T + 50$		62.5	66	



4.3.6 Timer input pulse (TA0IN, TA2IN, TB0IN0, TB1IN0)

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
Clock cycle	t_{VCK}	8T+100		200	234	ns
Low level pulse width	t_{VCKL}	4T + 40		90	107	
High level pulse width	t_{VCKH}	4T + 40		90	107	

4.3.7 Interrupt Operation

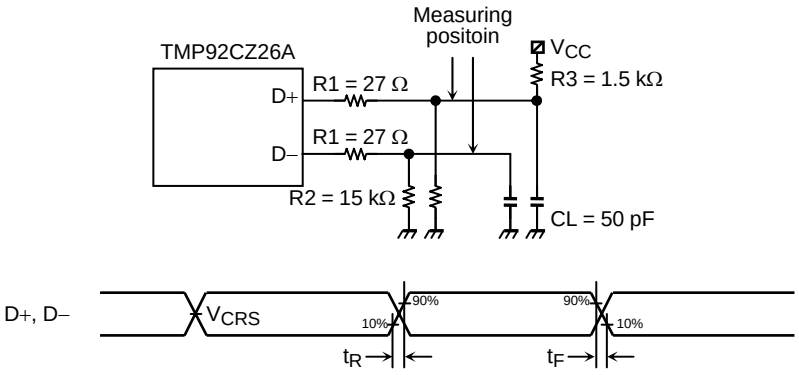
Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
INT0~INT7 low width	t_{INTAL}	2T + 40		65	74	ns
INT0~INT7 high width	t_{INTAH}	2T + 40		65	74	

4.3.8 USB Timing (Full-speed)

V_{CC} = 3.3 ± 0.3 V/f_{USB} = 48 MHz/Ta = 0 ~ 70°C

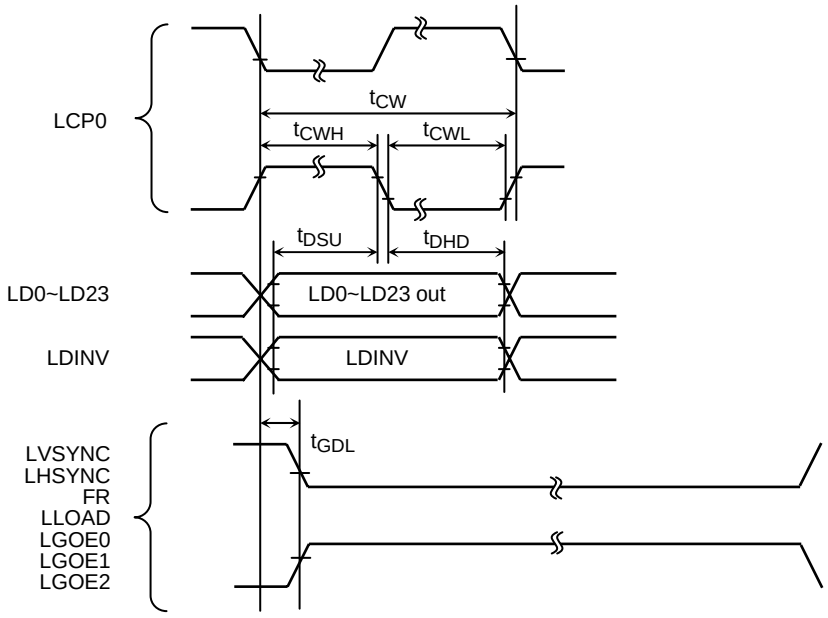
Parameter	Symbol	Min	Max	Unit
D+, D– rising time	t_R	4	20	ns
D+, D– falling time	t_F	4	20	
Output signal crossover voltage	V_{CRS}	1.3	2.0	V

AC measuring condition



4.3.9 LCD Controller

Parameter	Symbol	Variable		80 MHz (n=0)	60 MHz (n=0)	Unit
		Min	Max			
LCP0 clock period	t_{CW}	2T(n+1)		25	33.3	ns
LCP0 high width (Include phase inversion)	t_{CWH}	T(n+1) – 5		7.5	11.6	
LCP0 low width (Include phase inversion)	t_{CWL}	T(n+1) – 5		7.5	11.6	
Data valid → LCP0 falling (Include phase inversion)	t_{DSU}	T(n+1) – 7.5		5	9.1	
LCP0 falling → Data hold (Include phase inversion)	t_{DHD}	T(n+1) – 7.5		5	9.1	
Signal delay from LCP0 basic changing point (Include phase inversion)	t_{GDL}	-20	20	±20	±20	



AC measuring condition

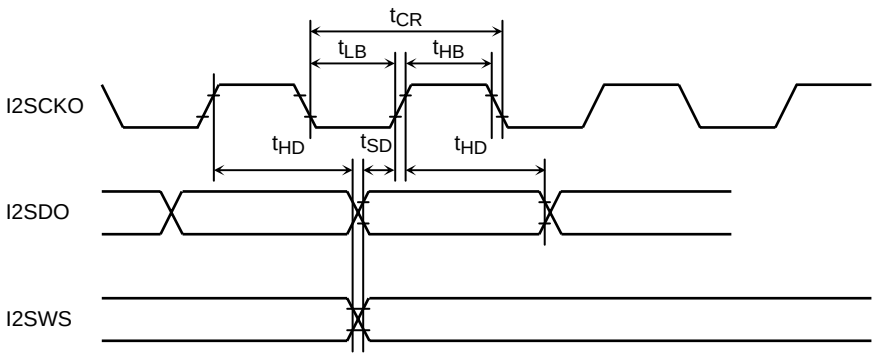
- C_L = 50 pF (LCP0 only C_L = 30 pF)

Note: The “n” in “Variable” show value that is set to LCDMODE0<SCPW1:0>.

Example: If LCDMODE0<SCPW1:0> = “01”, n=1, t_{RWP} = 2T(n+1) = 2T

4.3.10 I^2S Timing

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
I2SCKO clock period	t_{CR}	t_{IC}		100	100	ns
I2SCKO high width	t_{HB}	$0.5\,t_{CR} - 15$		35	35	
I2SCKO low width	t_{LB}	$0.5\,t_{CR} - 15$		35	35	
I2SDO, I2SWS setup time	t_{SD}	$0.5\,t_{CR} - 15$		35	35	
I2SDO, I2SWS hold time	t_{HD}	$0.5\,t_{CR} - 8$		42	42	



Note: The Maximum operation frequency of I2SCKO in I2S circuit is 10MHz. Don't set I2SCKO to value more than 10MHz.

AC measuring condition

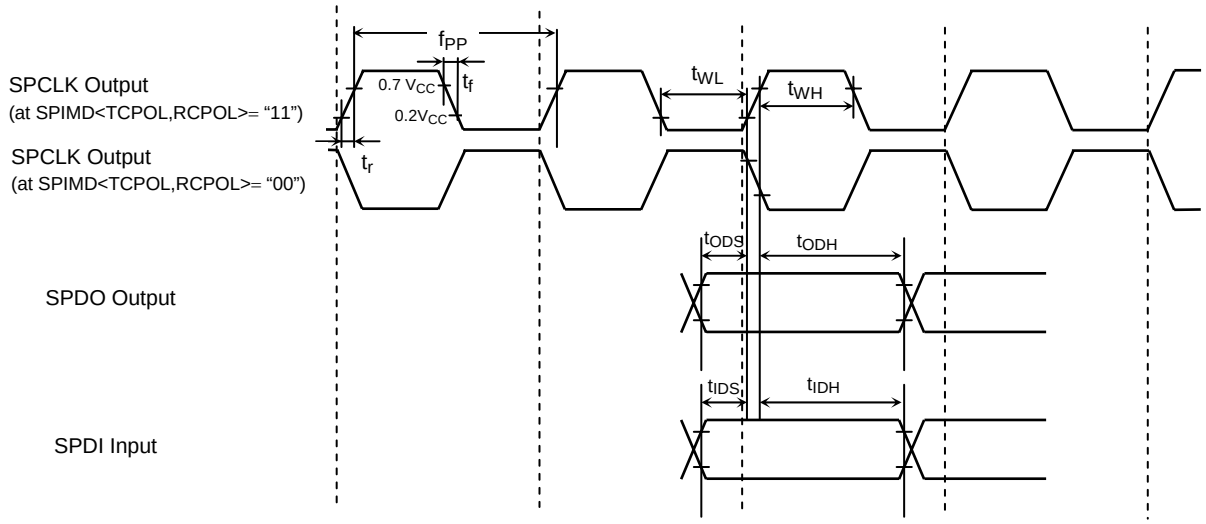
- I2SCKO, I2SDO and I2SWS pins C_L = 30 pF

4.3.11 SPI Controller

Parameter	Symbol	Variable		80MHz	60 MHz	Unit
		Min	Max			
SPCLK frequency (= 1/S)	f_{PP}		20	20	15	MHz
SPCLK rising time	t_r		6	6	6	ns
SPCLK falling time	t_f		6	6	6	
SPCLK low width	t_{WL}	0.5S – 6		19	28	
SPCLK high width	t_{WH}	0.5S – 6		19	28	
Output data valid → SPCLK rising	t_{ODS}	0.5S – 18		7	15	
SPCLK rising/ falling → Output data hold	t_{ODH}	0.5S – 10		15	23.4	
Input data valid → SPCLK rising/ falling	t_{IDS}	5		5	5	
SPCLK rising/ falling → Input data valid	t_{IDH}	5		5	5	

AC measuring condition

- Clock of top column in above table shows system clock frequency, and “S” in “Variable” show SPCLK clock cycle [ns].
- C_L = 25 pF



4.4 AD Conversion Characteristics

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage (+)	VREFH		AVCC − 0.2	AVCC	AVCC	V
Analog reference voltage (−)	VREFL		DVSS	DVSS	DVSS + 0.2	
AD converter power supply voltage	AVCC		DVCC3A/3B	DVCC3A/3B	DVCC3A/3B	
AD converter ground	AVSS		DVSS	DVSS	DVSS	
Analog input voltage	AVIN		VREFL		VREFH	
Analog current for analog reference voltage	IREFON	<VREFON> = 1		0.38	0.45	mA
	IREFOFF	<VREFON> = 0		1	5	μA
Total error (Quantize error of ±0.5 LSB is included)	E_T	Conversion speed at 12uS		±2.0	±4.0	LSB

Note1: 1 LSB = (VREFH − VREFL)/1024[V]

Note2: Minimum frequency for operation

Minimum clock for AD converter operate is 3MHz. (Clock frequency that is seleted by Clock gear ≥ f_{SYS} = 3MHz)

Note3: The power supply current from AVCC pin is included in the power supply current of VCC pin (ICC).

5. Table of Special function registers (SFRs)

The SFRs include the I/O ports and peripheral control registers allocated to the 8-Kbyte address space from 000000H to 001FF0H.

(1) I/O Port
(2) Interrupt control
(3) Memory controller
(4) TSI(Touch screen I/F)
(5) SDRAM controller
(6) LCD controller
(7) PMC
(8) USB controller
(9) SPI controller
(10) MMU
(11) NAND-Flash controller
(12) DMA controller
(13) Clock gear, PLL
(14) 8-bit timer
(15) 16-bit timer
(16) SIO
(17) SBI
(18) AD converter
(19) Watchdog timer
(20)RTC(Real time clock)
(21)MLD(Melody/alarm generator)
(22)I^2S
(23) MAC

Table layout

Symbol	Name	Address	7	6			1	0

Bit Symbol
Read/Write
Initial value after reset
Remarks

Note: “Prohibit RMW” in the table means that you cannot use RMW instructions on these register.
Example: When setting bit0 only of the register PxCR, the instruction “SET 0, (PxCR)” cannot be used. The LD (transfer) instruction must be used to write all eight bits.

Read/Write

R/W:	Both read and write are possible.
R:	Only read is possible.
W:	Only write is possible.
W*:	Both read and write are possible (when this bit is read as1)
Prohibit RMW:	Read modify write instructions are prohibited. (The EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instruction are read modify write instructions.)
R/W*:	Read modify write is prohibited when controlling the pull-up resistor.

Table 5.1 I/O Register Address Map

[1] Port (1/2)

Address	Name
0000H	
1H	
2H	
3H	
4H	P1
5H	
6H	P1CR
7H	P1FC
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0010H	P4
1H	
2H	
3H	P4FC
4H	P5
5H	
6H	
7H	P5FC
8H	P6
9H	
AH	P6CR
BH	P6FC
CH	P7
DH	
EH	P7CR
FH	P7FC

Address	Name
0020H	P8
1H	P8FC2
2H	
3H	P8FC
4H	P9
5H	P9FC2
6H	P9CR
7H	P9FC
8H	PA
9H	
AH	
BH	PAFC
CH	
DH	
EH	
FH	

Address	Name
0030H	PC
1H	
2H	PCCR
3H	PCFC
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	PF
DH	
EH	PFCR
FH	PFFC

Address	Name
0040H	PG
1H	
2H	
3H	PGFC
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	PJ
DH	
EH	PJCR
FH	PJFC

Address	Name
0050H	PK
1H	
2H	
3H	PKFC
4H	PL
5H	
6H	
7H	PLFC
8H	PM
9H	
AH	
BH	PMFC
CH	PN
DH	
EH	PNCR
FH	PNFC

Address	Name
0060H	PP
1H	
2H	PPCR
3H	PPFC
4H	PR
5H	
6H	PRCR
7H	PRFC
8H	PZ
9H	
AH	PZCR
BH	
CH	
DH	
EH	
FH	

Address	Name
0070H	Reserved
1H	Reserved
2H	Reserved
3H	Reserved
4H	Reserved
5H	Reserved
6H	Reserved
7H	Reserved
8H	Reserved
9H	Reserved
AH	Reserved
BH	Reserved
CH	Reserved
DH	Reserved
EH	Reserved
FH	Reserved

[1] Port (2/2)

Address	Name
0080H	
1H	P1DR
2H	
3H	
4H	P4DR
5H	P5DR
6H	P6DR
7H	P7DR
8H	P8DR
9H	P9DR
AH	PADR
BH	
CH	PCDR
DH	
EH	
FH	PFDR

Address	Name
0090H	PGDR
1H	
2H	
3H	PJDR
4H	PKDR
5H	PLDR
6H	PMDR
7H	PNDR
8H	PPDR
9H	PRDR
AH	PZDR
BH	PTDR
CH	PUDR
DH	PVDR
EH	PWDR
FH	PXDR

Address	Name
00A0H	PT
1H	
2H	PTCR
3H	PTFC
4H	PU
5H	
6H	PUCR
7H	PUFC
8H	PV
9H	PVFC2
AH	PVCR
BH	PVFC
CH	PW
DH	
EH	PWCR
FH	PWFC

Address	Name
00B0H	PX
1H	
2H	PXCR
3H	PXFC
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[2] INTC

Address	Name
00D0H	INTE12
1H	INTE34
2H	INTE56
3H	INTE7
4H	INTETA01
5H	INTETA23
6H	INTETA45
7H	INTETA67
8H	INTETB0
9H	INTETB1
AH	
BH	INTES0
CH	
DH	
EH	
FH	

Address	Name
00E0H	INTESBIADM
1H	INTESPI
2H	Reserved
3H	INTEUSB
4H	Reserved
5H	INTEALM
6H	Reserved
7H	
8H	INTERTC
9H	INTEKEY
AH	INTELCD
BH	INTEI2S01
CH	INTENDFC
DH	Reserved
EH	INTEP0
FH	INTEAD

Address	Name
00F0H	INTE0
1H	INTETC01 /INTEDMA01
2H	INTETC23 /INTEDMA23
3H	INTETC45 /INTEDMA45
4H	INTETC67
5H	SIMC
6H	IIMC0
7H	INTWDT
8H	INTCLR
9H	
AH	IIMC1
BH	
CH	
DH	
EH	
FH	Reserved

Address	Name
0100H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	DMA4V
5H	DMA5V
6H	DMA6V
7H	DMA7V
8H	DMAB
9H	DMAR
AH	DMASEL
BH	
CH	
DH	
EH	
FH	

[3] MEMC

Address	Name
0140H	B0CSL
1H	B0CSH
2H	MAMR0
3H	MSAR0
4H	B1CSL
5H	B1CSH
6H	MAMR1
7H	MSAR1
8H	B2CSL
9H	B2CSH
AH	MAMR2
BH	MSAR2
CH	B3CSL
DH	B3CSH
EH	MAMR3
FH	MSAR3

Address	Name
0150H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	BEXCSL
9H	BEXCSH
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0160H	
1H	
2H	
3H	
4H	
5H	
6H	PMEMCR
7H	
8H	CSTMGCR
9H	WRTMGCR
AH	RDTMGCR0
BH	RDTMGCR1
CH	BROMCR
DH	RAMCR
EH	
FH	

[4] TSI

Address	Name
01F0H	TSICR0
1H	TSICR1
2H	Reserved
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[5] SDRAMC

Address	Name
0250H	SDACR
1H	SDCISR
2H	SDRCR
3H	SDCMM
4H	SDBLS
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[6] LCDC

Address	Name
0280H	LCDMODE0
1H	LCDMODE1
2H	
3H	LCDDVM0
4H	LCDSIZE
5H	LCDCTL0
6H	LCDCTL1
7H	LCDCTL2
8H	LCDDVM1
9H	
AH	LCDHSP
BH	LCDHSP
CH	LCDVSP
DH	LCDVSP
EH	LCDPRVSP
FH	LCDHSDLY

Address	Name
0290H	LCDHSDLY
1H	LCDO0DLY
2H	LCDO1DLY
3H	LCDO2DLY
4H	LCDHSW
5H	LCDLDW
6H	LCDHO0W
7H	LCDHO1W
8H	LCDHO2SW
9H	LCDHWB8
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
02A0H	LSAML
1H	LSAMM
2H	LSAMH
3H	
4H	LSASL
5H	LSASM
6H	LSASH
7H	
8H	LSAHX
9H	LSAHX
AH	LSAHY
BH	LSAHY
CH	LSASS
DH	LSASS
EH	LSACS
FH	LSACS

[7] PMC

Address	Name
02F0H	PMCCTL
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[8] USBC (1/2)

Address	Name
0500H	Descriptor
to	RAM
067FH	(384 byte)

Address	Name
0780H	ENDPOINT0
1H	ENDPOINT1
2H	ENDPOINT2
3H	ENDPOINT3
4H	
5H	
6H	
7H	
8H	
9H	EP1_MODE
AH	EP2_MODE
BH	EP3_MODE
CH	
DH	
EH	
FH	

Address	Name
0790H	EP0_STATUS
1H	EP1_STATUS
2H	EP2_STATUS
3H	EP3_STATUS
4H	
5H	
6H	
7H	
8H	EP0_SIZE_L_A
9H	EP1_SIZE_L_A
AH	EP2_SIZE_L_A
BH	EP3_SIZE_L_A
CH	
DH	
EH	
FH	

Address	Name
07A0H	
1H	EP1_SIZE_L_B
2H	EP2_SIZE_L_B
3H	EP3_SIZE_L_B
4H	
5H	
6H	
7H	
8H	
9H	EP1_SIZE_H_A
AH	EP2_SIZE_H_A
BH	EP3_SIZE_H_A
CH	
DH	
EH	
FH	

Address	Name
07B0H	
1H	EP1_SIZE_H_B
2H	EP2_SIZE_H_B
3H	EP3_SIZE_H_B
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
07C0H	bmRequestType
1H	bRequest
2H	wValue_L
3H	wValue_H
4H	wIndex_L
5H	wIndex_H
6H	wLength_L
7H	wLength_H
8H	SetupReceived
9H	Current_Config
AH	Standard Request
BH	Request
CH	DATASET1
DH	DATASET2
EH	USB STATE
FH	EOP

Address	Name
07D0H	COMMAND
1H	EPx_SINGLE1
2H	Reserved
3H	EPx_BCS1
4H	Reserved
5H	
6H	INT_Control
7H	
8H	Standard Request Mode
9H	Request Mode
AH	
BH	
CH	
DH	
EH	ID_CONTROL
FH	ID_STATE

Note: Do not access no allocated name address.

[8] USBC (2/2)

Address	Name
07E0H	Port Status
1H	FRAME_L
2H	FRAME_H
3H	ADDRESS
4H	
5H	
6H	USBREADY
7H	
8H	Set Descriptor STALL
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
07F0H	USBINTFR1
1H	USBINTFR2
2H	USBINTFR3
3H	USBINTFR4
4H	USBINTMR1
5H	USBINTMR2
6H	USBINTMR3
7H	USBINTMR4
8H	USBCR1
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[9] SPIC

Address	Name
0820H	SPIMD
1H	SPIMD
2H	SPICT
3H	SPICT
4H	SPIST
5H	SPIST
6H	SPICR
7H	SPICR
8H	
9H	
AH	
BH	
CH	SPIIE
DH	SPIIE
EH	
FH	

Address	Name
0830H	SPITD0
1H	SPITD0
2H	SPITD1
3H	SPITD1
4H	SPIRD0
5H	SPIRD0
6H	SPIRD1
7H	SPIRD1
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] MMU

Address	Name
0880H	LOCALPX
1H	LOCALPX
2H	LOCALPY
3H	LOCALPY
4H	LOCALPZ
5H	LOCALPZ
6H	
7H	
8H	LOCALLX
9H	LOCALLX
AH	LOCALLY
BH	LOCALLY
CH	LOCALLZ
DH	LOCALLZ
EH	
FH	

Address	Name
0890H	LOCALRX
1H	LOCALRX
2H	LOCALRY
3H	LOCALRY
4H	LOCALRZ
5H	LOCALRZ
6H	
7H	
8H	LOCALWX
9H	LOCALWX
AH	LOCALWY
BH	LOCALWY
CH	LOCALWZ
DH	LOCALWZ
EH	
FH	

Address	Name
08A0H	LOCALESX
1H	LOCALESX
2H	LOCALESY
3H	LOCALESY
4H	LOCALESZ
5H	LOCALESZ
6H	
7H	
8H	LOCALEDX
9H	LOCALEDX
AH	LOCALEDY
BH	LOCALEDY
CH	LOCALEDZ
DH	LOCALEDZ
EH	
FH	

Address	Name
08B0H	LOCALOSX
1H	LOCALOSX
2H	LOCALOSY
3H	LOCALOSY
4H	LOCALOSZ
5H	LOCALOSZ
6H	
7H	
8H	LOCALODX
9H	LOCALODX
AH	LOCALODY
BH	LOCALODY
CH	LOCALODZ
DH	LOCALODZ
EH	
FH	

Note: Do not access no allocated name address.

[11] NAND-Flash controller

Address	Name
08C0H	NDFMCR0
1H	NDFMCR0
2H	NDFMCR1
3H	NDFMCR1
4H	NDECCRD0
5H	NDECCRD0
6H	NDECCRD1
7H	NDECCRD1
8H	NDECCRD2
9H	NDECCRD2
AH	NDECCRD3
BH	NDECCRD3
CH	NDECCRD4
DH	NDECCRD4
EH	
FH	

Address	Name
08D0H	NDRSCA0
1H	NDRSCA0
2H	NDRSCD0
3H	
4H	NDRSCA1
5H	NDRSCA1
6H	NDRSCD1
7H	
8H	NDRSCA2
9H	NDRSCA2
AH	NDRSCD2
BH	
CH	NDRSCA3
DH	NDRSCA3
EH	NDRSCD3
FH	

Address	Name
1FF0H	NDFDTR0
1H	NDFDTR0
2H	NDFDTR1
3H	NDFDTR1
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[12] DMAC

Address	Name
0900H	HDMAS0
1H	HDMAS0
2H	HDMAS0
3H	
4H	HDMAD0
5H	HDMAD0
6H	HDMAD0
7H	
8H	HDMACA0
9H	HDMACA0
AH	HDMACB0
BH	HDMACB0
CH	HDMAM0
DH	
EH	
FH	

Address	Name
0910H	HDMAS1
1H	HDMAS1
2H	HDMAS1
3H	
4H	HDMAD1
5H	HDMAD1
6H	HDMAD1
7H	
8H	HDMACA1
9H	HDMACA1
AH	HDMACB1
BH	HDMACB1
CH	HDMAM1
DH	
EH	
FH	

Address	Name
0920H	HDMAS2
1H	HDMAS2
2H	HDMAS2
3H	
4H	HDMAD2
5H	HDMAD2
6H	HDMAD2
7H	
8H	HDMACA2
9H	HDMACA2
AH	HDMACB2
BH	HDMACB2
CH	HDMAM2
DH	
EH	
FH	

Address	Name
0930H	HDMAS3
1H	HDMAS3
2H	HDMAS3
3H	
4H	HDMAD3
5H	HDMAD3
6H	HDMAD3
7H	
8H	HDMACA3
9H	HDMACA3
AH	HDMACB3
BH	HDMACB3
CH	HDMAM3
DH	
EH	
FH	

Address	Name
0940H	HDMAS4
1H	HDMAS4
2H	HDMAS4
3H	
4H	HDMAD4
5H	HDMAD4
6H	HDMAD4
7H	
8H	HDMACA4
9H	HDMACA4
AH	HDMACB4
BH	HDMACB4
CH	HDMAM4
DH	
EH	
FH	

Address	Name
0950H	HDMAS5
1H	HDMAS5
2H	HDMAS5
3H	
4H	HDMAD5
5H	HDMAD5
6H	HDMAD5
7H	
8H	HDMACA5
9H	HDMACA5
AH	HDMACB5
BH	HDMACB5
CH	HDMAM5
DH	
EH	
FH	

Address	Name
0970H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	Reserved
DH	Reserved
EH	HDMAE
FH	HDMATR

[13] CGEAR, PLL

Address	Name
10E0H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	EMCCR0
4H	EMCCR1
5H	EMCCR2
6H	Reserved
7H	
8H	PLLCR0
9H	PLLCR1
AH	
BH	
CH	
DH	
EH	
FH	

[14] 8-bit timer

Address	Name
1100H	TA01RUN
1H	
2H	TA0REG
3H	TA1REG
4H	TA01MOD
5H	TA1FFCR
6H	
7H	
8H	TA23RUN
9H	
AH	TA2REG
BH	TA3REG
CH	TA23MOD
DH	TA3FFCR
EH	
FH	

Address	Name
1110H	TA45RUN
1H	
2H	TA4REG
3H	TA5REG
4H	TA45MOD
5H	TA5FFCR
6H	
7H	
8H	TA67RUN
9H	
AH	TA6REG
BH	TA7REG
CH	TA67MOD
DH	TA7FFCR
EH	
FH	

[15] 16-bit timer

Address	Name
1180H	TB0RUN
1H	
2H	TB0MOD
3H	TB0FFCR
4H	
5H	
6H	
7H	
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

Address	Name
1190H	TB1RUN
1H	
2H	TB1MOD
3H	TB1FFCR
4H	
5H	
6H	
7H	
8H	TB1RG0L
9H	TB1RG0H
AH	TB1RG1L
BH	TB1RG1H
CH	TB1CP0L
DH	TB1CP0H
EH	TB1CP1L
FH	TB1CP1H

[16] SIO

Address	Name
1200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	SIRCR
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[17] SBI

Address	Name
1240H	SBI0CR1
1H	SBI0DBR
2H	I2C0AR
3H	SBI0CR2/SBI0SR
4H	SBI0BR0
5H	
6H	
7H	SBI0CR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[18] 10-bit ADC

Address	Name
12A0H	ADREG0L
1H	ADREG0H
2H	ADREG1L
3H	ADREG1H
4H	ADREG2L
5H	ADREG2H
6H	ADREG3L
7H	ADREG3H
8H	ADREG4L
9H	ADREG4H
AH	ADREG5L
BH	ADREG5H
CH	Reserved
DH	Reserved
EH	Reserved
FH	Reserved

Address	Name
12B0H	ADREGSPL
1H	ADREGSPH
2H	Reserved
3H	Reserved
4H	ADCM0REGL
5H	ADCM0REGH
6H	ADCM1REGL
7H	ADCM1REGH
8H	ADMOD0
9H	ADMOD1
AH	ADMOD2
BH	ADMOD3
CH	ADMOD4
DH	ADMOD5
EH	
FH	ADCCLK

[19] WDT

Address	Name
1300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[20] RTC

Address	Name
1320H	SECR
1H	MINR
2H	HOURR
3H	DAYR
4H	DATER
5H	MONTHR
6H	YEARR
7H	PAGER
8H	RESTR
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[21] MLD

Address	Name
1330H	ALM
1H	MELALMC
2H	MELFL
3H	MELFH
4H	ALMINT
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[22] I²S

Address	Name
1800H	I2S0BUF
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	I2S0CTL
9H	I2S0CTL
AH	I2S0C
BH	I2S0C
CH	
DH	
EH	
FH	

Address	Name
1810H	I2S1BUF
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	I2S1CTL
9H	I2S1CTL
AH	I2S1C
BH	I2S1C
CH	
DH	
EH	
FH	

[23] MAC

Address	Name
1BE0H	MACMA
1H	MACMA
2H	MACMA
3H	MACMA
4H	MACMB
5H	MACMB
6H	MACMB
7H	MACMB
8H	MACORL
9H	MACORL
AH	MACORL
BH	MACORL
CH	MACORH
DH	MACORH
EH	MACORH
FH	MACORH

Address	Name
1BF0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	MACCR
DH	
EH	
FH	

Note: Do not access no allocated name address.

(1) I/O ports (1/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1	PORT1	0004H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
P4	PORT4	0010H	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			0	0	0	0	0	0	0	0
P5	PORT5	0014H	P57	P56	P55	P54	P53	P52	P51	P50
			R/W							
			0	0	0	0	0	0	0	0
P6	PORT6	0018H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
P7	PORT7	001CH		P76	P75	P74	P73	P72	P71	P70
				R/W						
				Data from external port (Output latch register is set to “1”)		Data from external port (Output latch register is cleared to “0”)		Data from external port (Output latch register is set to “1”)		1
P8	PORT8	0020H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			1	1	1	1	1	0	1	1
P9	PORT9	0024H	P97	P96				P92	P91	P90
			R					R/W		
			Data from external port					Data from external port (Output latch register is set to “1”)		
PA	PORTA	0028H	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R							
			Data from external port							
PC	PORTC	0030H	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
			R/W							
			Data from external port (Output latch register is set to “1”)							
PF	PORTF	003CH	PF7		PF5	PF4	PF3	PF2	PF1	PF0
			R/W		R/W					
			1		Data from external port (Output latch register is set to “1”)					
PG	PORTG	0040H			PG5	PG4	PG3	PG2	PG1	PG0
					R					
					Data from external port					
PJ	PORTJ	004CH	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
			R/W							
			1	Data from external port (Output latch register is set to “1”)		1	1	1	1	1
PK	PORTK	0050H	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
			R/W							
			0	0	0	0	0	0	0	0
PL	PORTL	0054H	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
			R/W							
			0	0	0	0	0	0	0	0
PM	PORTM	0058H	PM7					PM2	PM1	
			R/W					R/W		
			1					1	1	
PN	PORTN	005CH	PN7	PN6	PN5	PN4	PN3	PN2	PN1	PN0
			R/W							
			Data from external port (Output latch register is cleared to “1”)							
PP	PORTP	0060H	PP7	PP6	PP5	PP4	PP3	PP2	PP1	
			R/W							
			0	0	Data from external port (Output latch register is cleared to “0”)					

(1) I/O ports (2/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PR	PORTR	0064H					PR3	PR2	PR1	PR0
							R/W			
							Data from external port (Output latch register is cleared to “0”)			
PT	PORTT	00A0H	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
PU	PORTU	00A4H	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
PV	PORTV	00A8H	PV7	PV6		PV4	PV3	PV2	PV1	PV0
			R/W			R/W				
			Data from external port (Output latch register is cleared to “0”)			Data from external port (Output latch register is cleared to “0”)				
PW	PORTW	00ACH	PW7	PW6	PW5	PW4	PW3	PW2	PW1	PW0
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
PX	PORTX	00B0H	PX7		PX5	PX4				
			R/W		R/W					
			Data from external port (Output latch register is cleared to “0”)							
PZ	PORTZ	0068H	PZ7	PZ6	PZ5	PZ4	PZ3	PZ2	PZ1	PZ0
			R/W							
			Data from external port (Output latch register is cleared to “0”)							

(1) I/O ports (3/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1CR	PORT1 control register	0006H (Prohibit RMW)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1:Output							
P1FC	PORT1 function register	0007H (Prohibit RMW)								P1F
										W
										0/1
										0: Port 1: Data bus (D8~D15)
P4FC	PORT4 function register	0013H (Prohibit RMW)	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			0: Port 1: Address bus (A0~A7)							
P5FC	PORT5 function register	0017H (Prohibit RMW)	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			0: Port 1: Address bus (A8~A15)							
P6CR	PORT6 control register	001AH (Prohibit RMW)	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P6FC	PORT6 function register	001BH (Prohibit RMW)	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			0: Port 1: Address bus (A16~A23)							
P7CR	PORT7 control register	001EH (Prohibit RMW)		P76C	P75C	P74C	P73C	P72C	P71C	
				W	W	W	W	W	W	
				0	0	0	0	0	0	
				0: Input port, $\overline{WAIT}$ 1:Output port	0: Input port, NDR/$\overline{B}$ 1: Output port, R/$\overline{W}$	0: Input port 1: Output port, EA25	0: Input port 1: Output port, EA24	0: Input port 1: Output port, $\overline{NDWE}$ @ <P72> = 0, $\overline{WRLU}$ @ <P72> = 1	0: Input port 1: Output port, $\overline{NDRE}$ @ <P71> = 0, $\overline{WRLL}$ @ <P71> = 1	
P7FC	PORT7 function register	001FH (Prohibit RMW)		P76F	P75F	P74F	P73F	P72F	P71F	P70F
				W						
				0	0	0	0	0	0	0
				0: Port 1: $\overline{WAIT}$	0: Port 1:NDR/$\overline{B}$, R/$\overline{W}$	0:Port 1: EA25	0:Port 1: EA24	0: Port 1: $\overline{NDWE}$ @ <P72> = 0, $\overline{WRLU}$ @ <P72> = 1	0: Port 1: $\overline{NDRE}$ @ <P71> = 0, $\overline{WRLL}$ @ <P71> = 1	0: Port 1: $\overline{RD}$

(1) I/O ports (4/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P8FC	PORT8 function register	0023H (Prohibit RMW)	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: <P87F2>	0: Port 1: <P86F2>	0: Port 1: $\overline{CSZC}$	0: Port 1: $\overline{CSZB}$	0: Port 1: $\overline{CS3}$, $\overline{CSXA}$	0: Port, $\overline{CSZA}$ 1: $\overline{CS2}$, $\overline{SDCS}$	0: Port 1: $\overline{CS1}$	0: Port 1: $\overline{CS0}$
P8FC2	PORT8 function fegister2	0021H (Prohibit RMW)	P87F2	P86F2			P84F2	P82F2	P81F2	
			W				W			
			0	0			0	0	0	
			0: $\overline{CSXB}$ 1: $\overline{ND1CE}$	0: $\overline{CSZD}$ 1: $\overline{ND0CE}$			0: Port, $\overline{CS3}$ 1: $\overline{CSXA}$	0: Output port, $\overline{CS2}$ 1: $\overline{CSZA}$, $\overline{SDCS}$	0: <P81F> 1: $\overline{SDCS}$	
P9CR	PORT9 control register	0026H (Prohibit RMW)						P92C	P91C	P90C
								W		
								0	0	0
								0 Input port, $\overline{CTS0}$ 1: Output port, SCLK0	0: Input port, RXD0 1: Output port,	0: Input port, 1: Output port, TXD0
P9FC	PORT9 function register	0027H (Prohibit RMW)		P96F				P92F		P90F
				W				W		W
				0				0		0
				0: Input port, 1:INT4				0:Port, $\overline{CTS0}$ 1:SCLK0		0:Port 1:TXD0
P9FC2	PORT9 function register2	0025H (Prohibit RMW)	–					–		P90FC2
			W					W		W
			0					0		0
			Always write “0”					Always write “0”		0: CMOS 1:Open -Drain

(1) I/O ports (5/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PAFC	PORTA function register	002BH (Prohibit RMW)	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
			W							
			0	0	0	0	0	0	0	0
			0: Key-in disable 1: Key-in enable							
PCCR	PORTC control register	0032H (Prohibit RMW)	PC7C	PC6C	PC5C	PC4C	PC3C	PC2C	PC1C	PC0C
			W							
			0	0	0	0	0	0	0	0
			0: Input port, 1: Output port, KO output (Open -drain)	0: Input port, EA28 1: Output port	0: Input port, EA27 1: Output port	0: Input port, EA26 1: Output port	0: Input port, INT3 1: Output port, TA2IN	0: Input port, INT2 1: Output port,	0: Input port, INT1 1: Output port, TA0IN	0: Input port, INT0 1: Output port,
PCFC	PORTC function register	0033H (Prohibit RMW)	PC7F	PC6F	PC5F	PC4F	PC3F	PC2F	PC1F	PC0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1:KO output (Open -Drain)	0: Port 1:EA28,	0:Port 1:EA27	0:Port 1:EA26	0:Port 1:INT3, TA2IN	0: Port 1: INT2	0: Port 1: INT1, TA0IN	0: Port 1:INT0
PFCR	PORTF control register	003EH (Prohibit RMW)			PF5C	PF4C	PF3C	PF2C	PF1C	PF0C
					W					
					0	0	0	0	0	0
					0: Input, 1: Output					
PFFC	PORTF function register	003FH (Prohibit RMW)	PF7F		PF5F	PF4F	PF3F	PF2F	PF1F	PF0F
			W		W					
			1		0	0	0	0	0	0
			0: Output port, 1: SDCLK		0:Port 1:I2S1WS	0:Port 1:I2S1DO	0:Port 1:I2S1CKO	0:Port 1:I2S0WS	0:Port 1:I2S0DO	0:Port 1:I2S1CKO

(1) I/O ports (6/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PGFC	PORTG function register	0043H (Prohibit RMW)					PG3F			
							W			
							0			
							0:Input port, AN3 1: $\overline{\text{ADTRG}}$			
PJCR	PORTJ control register	004EH (Prohibit RMW)		PJ6C	PJ5C					
				W						
				0	0					
				0:Input	1: Output					
PJFC	PORTJ function register	004FH (Prohibit RMW)	PF7F	PF6F	PF5F	PF4F	PF3F	PF2F	PF1F	PF0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: SDCKE	0: Port 1:NDCLE	0: Port 1: NDALE	0: Port 1: SDLUDQM	0: Port 1: SDLLDQM	0: Port 1: $\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$	0: Port 1: $\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$	0: Port 1: $\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$
PKFC	PORTK function register	0053H (Prohibit RMW)	PK7F	PK6F	PK5F	PK4F	PK3F	PK2F	PK1F	PK0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: LGOE2	0: Port 1: LGOE1	0: Port 1: LGOE0	0: Port 1: LHSYNC	0: Port 1: LVSYNC	0: Port 1: LFR	0: Port 1: LLOAD	0: Port 1: LCP0
PLFC	PORTL function register	0057H (Prohibit RMW)	PL7F	PL6F	PL5F	PL4F	PL3F	PL2F	PL1F	PL0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: Data bus for LCDC (LD7~LD0)							
PMFC	PORTM function register	005BH (Prohibit RMW)	PM7F					PM2F	PM1F	
			W					W	W	
			0					0	0	
			0: Port 1:PWE					0: Port 1: $\overline{\text{ALARM}}$, $\overline{\text{MLDALM}}$	0: Port 1:MLDALM ,TA1OUT	

(1) I/O ports (7/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PNCR	PORTN control register	005EH (Prohibit RMW)	PN7C	PN6C	PN5C	PN4C	PN3C	PN2C	PN1C	PN0C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
PNFC	PORTN function register	005FH (Prohibit RMW)	PN7F	PN6F	PN5F	PN4F	PN3F	PN2F	PN1F	PN0F
			W							
			0	0	0	0	0	0	0	0
			0:CMOS output 1:Open-Drain output							
PPCR	PORTP control register	0062H (Prohibit RMW)			PP5C	PP4C	PP3C	PP2C	PP1C	
					W					
					0	0	0	0	0	
					0: Input 1: Output					
PPFC	PORTP function register	0063H (Prohibit RMW)	PP7F	PP6F	PP5F	PP4F	PP3F	PP2F	PP1F	
			W							
			0	0	0	0	0	0	0	
			0: Port 1: TB1OUT0	0: Port 1: TB0OUT0	0: Port 1: TB1IN0@ <PP5C>=1 INT7@ <PP5C>=0	0: Port 1: TB0IN0@ <PP4C>=1 INT6@ <PP4C>=0	0: Port 1: TA7OUT@ <PP3C>=1 INT5@ <PP3C>=0	0: Port 1: TA5OUT	0: Port 1: TA3OUT	
PRCR	PORTR control register	0066H (Prohibit RMW)					PR3C	PR2C	PR1C	PR0C
							W			
							0	0	0	0
							0: Input, 1: Output			
PRFC	PORTR function register	0067H (Prohibit RMW)					PR3F	PR2F	PR1F	PR0F
							W			
							0	0	0	0
							0: Port 1: SPCLK	0: Port 1: $\overline{SPCS}$	0: Port 1: SPDO	0: Port 1: SPDI
PTCR	PORTT control register	00A2H (Prohibit RMW)	PT7C	PT6C	PT5C	PT4C	PT3C	PT2C	PT1C	PT0C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
PTFC	PORTT function register	00A3H (Prohibit RMW)	PT7F	PT6F	PT5F	PT4F	PT3F	PT2F	PT1F	PT0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: Data bus for LCDC (LD15~LD8)							

(1) I/O ports (8/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PUCR	PORTU control register	00A6H (Prohibit RMW)	PU7C	PU6C	PU5C	PU4C	PU3C	PU2C	PU1C	PU0C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
PUFC	PORTU function register	00A7H (Prohibit RMW)	PU7F	PU6F	PU5F	PU4F	PU3F	PU2F	PU1F	PU0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: LD23	0: Port 1: LD22	0: Port 1: LD21@ <PU5C>=1	0: Port 1: LD20	0: Port 1: LD19	0: Port 1: LD18	0: Port 1: LD17	0: Port 1: LD16
PVCR	PORTV control register	00AAH (Prohibit RMW)	PV7C	PV6C				PV2C	PV1C	PV0C
			W					W		
			0	0				0	0	0
			0: Input 1: Output					0: Input 1: Output		
PVFC	PORTV function register	00ABH (Prohibit RMW)	PV7F	PV6F				PV2F	PV1F	PV0F
			W					W		
			0	0				0	0	0
			0: Port 1: SCL	0: Port 1: SDA				0: Port 1:Reserved	0: Port 1:Reserved	0: Port 1: SCLK0@ <PV0C>=1
PWCR	PORTW control register	00AEH (Prohibit RMW)	PW7C	PW6C	PW5C	PW4C	PW3C	PW2C	PW1C	PW0C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
PWFC	PORTW function register	00AFH (Prohibit RMW)	PW7F	PW6F	PW5F	PW4F	PW3F	PW2F	PW1F	PW0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: Reserved							
PXCR	PORTX control register	00B2H (Prohibit RMW)	PX7C		PX5C					
			W		W					
			0		0					
			0: Input 1: Output							
PXFC	PORTX function register	00B3H (Prohibit RMW)	PX7F		PX5F	PX4F				
			W		W	W				
			0		0	0				
			0:Port 1:Reserved		0:Port 1: X1USB input	0:Port 1: CLKOUT @<PX4>=0 LDIV @<PX4>=1				
PZCR	PORTZ control register	006AH (Prohibit RMW)	PZ7C	PZ6C	PZ5C	PZ4C	PZ3C	PZ2C	PZ1C	PZ0C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							

(1) I/O ports (9/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1DR	PORT1 drive register	0081H	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P2DR	PORT2 drive register	0082H	P27D	P26D	P25D	P24D	P23D	P22D	P21D	P20D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P3DR	PORT3 drive register	0083H	P37D	P36D	P35D	P34D	P33D	P32D	P31D	P30D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P4DR	PORT4 drive register	0084H	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P5DR	PORT5 drive register	0085H	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P6DR	PORT6 drive register	0086H	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P7DR	PORT7 drive register	0087H		P76D	P75D	P74D	P73D	P72D	P71D	P70D
				R/W						
				1	1	1	1	1	1	1
				Input/Output buffer drive register for standby mode						
P8DR	PORT8 drive register	0088H	P87D	P86D	P85D	P84D	P83D	P82D	P81D	P80D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P9DR	PORT9 drive register	0089H	P97D	P96D				P92D	P91D	P90D
			R/W					R/W		
			1	1				1	1	1
			Input/Output buffer drive register for standby mode					Input/Output buffer drive register for standby mode		
PADR	PORTA drive register	008AH	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PCDR	PORTC drive register	008CH	PC7D	PC6D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PFDR	PORTF drive register	008FH	PF7D		PF5D	PF4D	PF3D	PF2D	PF1D	PF0D
			R/W		R/W					
			1		1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							

(1) I/O ports (10/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PGDR	PORTG drive register	0090H					PG3D	PG2D		
							R/W			
							1	1		
							Input/Output buffer drive register for standby mode			
PJDR	PORTJ drive register	0093H	PJ7D	PJ6D	PJ5D	PJ4D	PJ3D	PJ2D	PJ1D	PJ0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PKDR	PORTK drive register	0094H	PK7D	PK6D	PK5D	PK4D	PK3D	PK2D	PK1D	PK0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PLDR	PORTL drive register	0095H	PL7D	PL6D	PL5D	PL4D	PL3D	PL2D	PL1D	PL0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PMDR	PORTM drive register	0096H	PM7D					PM2D	PM1D	
			R/W					R/W		
			1					1	1	
			Input/Output buffer drive register for standby mode							
PNDR	PORTN drive register	0097H	PN7D	PN6D	PN5D	PN4D	PN3D	PN2D	PN1D	PN0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PPDR	PORTP drive register	0098H	PP7D	PP6D	PP5D	PP4D	PP3D	PP2D	PP1D	
			R/W							
			1	1	1	1	1	1	1	
			Input/Output buffer drive register for standby mode							
PRDR	PORTR drive register	0099H					PR3D	PR2D	PR1D	PR0D
							R/W			
							1	1	1	1
							Input/Output buffer drive register for standby mode			
PTDR	PORTT drive register	009BH	PT7D	PT6D	PT5D	PT4D	PT3D	PT2D	PT1D	PT0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PUDR	PORTU drive register	009CH	PU7D	PU6D	PU5D	PU4D	PU3D	PU2D	PU1D	PU0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PVDR	PORTV drive register	009DH	PV7D	PV6D		PV4D	PV3D	PV2D	PV1D	PV0D
			R/W			R/W				
			1	1		1	1	1	1	1
			Input/Output buffer drive register for standby mode							

(1) I/O ports (11/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PWDR	PORTW drive register	009EH	PW7D	PW6D	PW5D	PW4D	PW3D	PW2D	PW1D	PW0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PXDR	PORTX drive register	009FH	PX7D		PX5D	PX4D				
			R/W							
			1		1	1				
			Input/Output buffer drive register for standby mode							
PZDR	PORTZ drive register	009AH	PZ7D	PZ6D	PZ5D	PZ4D	PZ3D	PZ2D	PZ1D	PZ0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							

(2) Interrupt control (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0	INT0 enable	00F0H	–				INT0			
			–	–	–	–	I0C	I0M2	I0M1	I0M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTE12	INT1 & INT2 enable	00D0H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	00D1H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	00D2H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE7	INT7 enable	00D3H	–				INT7			
			–	–	–	–	I7C	I7M2	I7M1	I7M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	00D4H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	00D5H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA45	INTTA4 & INTTA5 enable	00D6H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA67	INTTA6 & INTTA7 enable	00D7H	INTTA7 (TMRA7)				INTTA6 (TMRA6)			
			ITA7C	ITA7M2	ITA7M1	ITA7M0	ITA6C	ITA6M2	ITA6M1	ITA6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB0	INTTB00 & INTTB01 enable	00D8H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB1	INTTB10 & INTTB11 enable	00D9H	INTTB11 (TMRB1)				INTTB10 (TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES0	INTRX0 & INTTX0 enable	00DBH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESBI ADM	INTSBI & INTADM enable	00E0H	INTADM				INTSBI			
			IADM0C	IADMM2	IADMM1	IADMM0	ISBI0C	ISBIM2	ISBIM1	ISBIM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESPI	INTSPI enable	00E1H	INTSPITX				INTSPIRX			
			ISPITC	ISPITM2	ISPITM1	ISPITM0	ISPIRC	ISPIRM2	ISPIRM1	ISPIRM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

(2) Interrupt control (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTEUSB	INTUSB enable	00E3H	–				INTUSB			
			–	–	–	–	IUSBC	IUSBM2	IUSBM1	IUSBM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEALM	INTALM enable	00E5H	–				INTALM			
			–	–	–	–	IALMC	IALMM2	IALMM1	IALMM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTERTC	INTRTC enable	00E8H	–				INTRTC			
			–	–	–	–	IRC	IRM2	IRM1	IRM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEKEY	INTKEY enable	00E9H	–				INTKEY			
			–	–	–	–	IKC	IKM2	IKM1	IKM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTELCD	INTLCD enable	00EAH	–				INTLCD			
			–	–	–	–	ILCD1C	ILCDM2	ILCDM1	ILCDM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEI2S01	INTI2S0 & INTI2S1 enable	00EBH	INTI2S1				INTI2S0			
			II2S1C	II2S1M2	II2S1M1	II2S1M0	II2S0C	II2S0M2	II2S0M1	II2S0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTENDFC	INTRSC & INTRDY enable	00ECH	INTRSC				INTRDY			
			IRSCC	IRSCM2	IRSCM1	IRSCM0	IRDYC	IRDYM2	IRDYM1	IRDYM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 enable	00EEH	–				INTP0			
			–	–	–	–	IP0C	IP0M2	IP0M1	IP0M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEAD	INTAD & INTADHP enable	00EFH	INTADHP				INTAD			
			IADHPC	IADHPM2	IADHPM1	IADHPM0	IADC	IADM2	IADM1	IADM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

(2) Interrupt control (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETC01 /INTEDMA01	INTTC0/INTDMA0 & INTTC1/INTDMA1 enable	00F1H	INTTC1/INTDMA1				INTTC0/INTDMA0			
			ITC1C /IDMA1C	ITC1M2 /IDMA1M2	ITC1M1 /IDMA1M1	ITC1M0 /IDMA1M0	ITC0C /IDMA0C	ITC0M2 /IDMA0M2	ITC0M1 /IDMA0M1	ITC0M0 /IDMA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23 /INTEDMA23	INTTC2/INTDMA2 & INTTC3/INTDMA3 enable	00F2H	INTTC3/INTDMA3				INTTC2/INTDMA2			
			ITC3C /IDMA3C	ITC3M2 /IDMA3M2	ITC3M1 /IDMA3M1	ITC3M0 /IDMA3M0	ITC2C /IDMA2C	ITC2M2 /IDMA2M2	ITC2M1 /IDMA2M1	ITC2M0 /IDMA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC45 /INTEDMA45	INTTC4/INTDMA4 & INTTC5/INTDMA5 enable	00F3H	INTTC5/INTDMA5				INTTC4/INTDMA4			
			ITC5C /IDMA5C	ITC5M2 /IDMA5M2	ITC5M1 /IDMA5M1	ITC5M0 /IDMA5M0	ITC4C /IDMA4C	ITC4M2 /IDMA4M2	ITC4M1 /IDMA4M1	ITC4M0 /IDMA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC67	INTTC6 & INTTC7 enable	00F4H	INTTC7 (DMA7)				INTTC6 (DMA6)			
			ITC7C	ITC7M2	ITC7M1	ITC7M0	ITC6C	ITC6M2	ITC6M1	ITC6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
SIMC	SIO interrupt mode control	00F5H (Prohibit RMW)	–	–						IR0LE
			W	W						W
			0	0						1
			Always write “0”	Always write “0”						0: INTRX0 edge mode 1: INTRX0 level mode
IIMC0	Interrupt input mode control 0	00F6H (Prohibit RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	–
			W	W	W	W	W	W	R/W	R/W
			0	0	0	0	0	0	0	0
			INT5 edge 0: Rising 1: Falling	INT4 edge 0: Rising 1: Falling	INT3 edge 0: Rising 1: Falling	INT2 edge 0: Rising 1: Falling	INT1 edge 0: Rising 1: Falling	INT0 edge 0: Rising 1: Falling	0: INT0 edge mode 1:INT0 level mode	Always write “0”
INTWDT	INTWD enable	00F7H	–				INTWD			
			–	–	–	–	ITCWD	–	–	–
			–	–			R	–	–	–
			Always write “0”				0	–	–	–
INTCLR	Interrupt clear control	00F8H (Prohibit RMW)	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt vector							
IIMC1	Interrupt input mode control 1	00FAH (Prohibit RMW)							I7EDGE	I6EDGE
									W	W
									0	0
									INT7 edge 0: Rising 1: Falling	INT6 edge 0: Rising 1: Falling

(2) Interrupt control (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	0100H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 start vector	0101H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 start vector	0102H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 start vector	0103H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
DMA4V	DMA4 start vector	0104H			DMA4V5	DMA4V4	DMA4V3	DMA4V2	DMA4V1	DMA4V0
					R/W					
					0	0	0	0	0	0
					DMA4 start vector					
DMA5V	DMA5 start vector	0105H			DMA5V5	DMA5V4	DMA5V3	DMA5V2	DMA5V1	DMA5V0
					R/W					
					0	0	0	0	0	0
					DMA5 start vector					
DMA6V	DMA6 start vector	0106H			DMA6V5	DMA6V4	DMA6V3	DMA6V2	DMA6V1	DMA6V0
					R/W					
					0	0	0	0	0	0
					DMA6 start vector					
DMA7V	DMA7 start vector	0107H			DMA7V5	DMA7V4	DMA7V3	DMA7V2	DMA7V1	DMA7V0
					R/W					
					0	0	0	0	0	0
					DMA7 start vector					
DMAB	DMA burst	0108H	DBST7	DBST6	DBST5	DBST4	DBST3	DBST2	DBST1	DBST0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request on burst mode							
DMAR	DMA request	0109H (Prohibit RMW)	DREQ7	DREQ6	DREQ5	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request in software							
DMASEL	Micro DMA/HDMA Select	010AH			DMASEL5	DMASEL4	DMASEL3	DMASEL2	DMASEL1	DMASEL0
					R/W					
					0	0	0	0	0	0
					0:Micro DMA5 1:HDMA5	0: Micro DMA4 1:HDMA4	0: Micro DMA3 1:HDMA3	0: Micro DMA2 1:HDMA2	0: Micro DMA1 1:HDMA1	0: Micro DMA0 1:HDMA0

(3) Memory controller (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B0CSL	BLOCK0 CS/WAIT control register low	0140H (Prohibit RMW)	B0WW3	B0WW2	B0WW1	B0WW0	B0WR3	B0WR2	B0WR1	B0WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved		
B0CSH	BLOCK0 CS/WAIT control register high	0141H (Prohibit RMW)	B0E			B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
			R/W			R/W				
			0			0	0	0	0	0
			CS select 0: Disable 1: Enable			Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	
B1CSL	BLOCK1 CS/WAIT control register low	0144H (Prohibit RMW)	B1WW3	B1WW2	B1WW1	B1WW0	B1WR3	B1WR2	B1WR1	B1WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved		
B1CSH	BLOCK1 CS/WAIT control register high	0145H (Prohibit RMW)	B1E			B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
			R/W			R/W				
			0			0	0	0	0	0
			CS select 0: Disable 1: Enable			Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: SDRAM		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	
B2CSL	BLOCK2 CS/WAIT control register low	0148H (Prohibit RMW)	B2WW3	B2WW2	B2WW1	B2WW0	B2WR3	B2WR2	B2WR1	B2WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved		
B2CSH	BLOCK2 CS/WAIT control register high	0149H (Prohibit RMW)	B2E	B2M		B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
			R/W			R/W				
			1	0		0	0	0	0	1
			CS select 0: Disable 1: Enable	0: 16 MB 1: Sets area		Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: SDRAM		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	

(3) Memory controller (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B3CSL	BLOCK3 CS/WAIT control register low	014CH (Prohibit RMW)	B3WW3	B3WW2	B3WW1	B3WW0	B3WR3	B3WR2	B3WR1	B3WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved		
B3CSH	BLOCK3 CS/WAIT control register high	014DH (Prohibit RMW)	B3E			B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
			R/W			R/W				
			0			0	0	0	0	0
			CS select 0: Disable 1: Enable			Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	
BEXCSL	BLOCK EX CS/WAIT control register low	0158H (Prohibit RMW)	BEXWW3	BEXWW2	BEXWW1	BEXWW0	BEXWR3	BEXWR2	BEXWR1	BEXWR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 waits 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved		
BEXCSH	BLOCK EX CS/WAIT control register high	0159H (Prohibit RMW)				BEXREC	BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
						R/W				
						0	0	0	0	0
						Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	

(3) Memory controller (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MAMR0	Memory address mask register 0	0142H	M0V20	M0V19	M0V18	M0V17	M0V16	M0V15	M0V14-9	M0V8
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR0	Memory start address register 0	0143H	M0S23	M0S22	M0S21	M0S20	M0S19	M0S18	M0S17	M0S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR1	Memory address mask register 1	0146H	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	MV15-9	M1V8
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR1	Memory start address register 1	0147H	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR2	Memory address mask register 2	014AH	M2V22	M2V21	M2V20	M2V19	M2V18	M2V17	M2V16	M2V15
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR2	Memory start address register 2	014BH	M2S23	M2S22	M2S21	M2S20	M2S19	M2S18	M2S17	M2S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR3	Memory address mask register 3	014EH	M3V22	M3V21	M3V20	M3V19	M3V18	M3V17	M3V16	M3V15
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR3	Memory start address register 3	014FH	M3S23	M3S22	M3S21	M3S20	M3S19	M3S18	M3S17	M3S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							

(3) Memory controller (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PMEMCR	Page ROM control register	0166H				OPGE	OPWR1	OPWR0	PR1	PR0
						R/W				
						0	0	0	1	0
						ROM page access 0: Disable 1: Enable	Wait number on page 00: 1 CLK (n-1-1-1 mode) 01: 2 CLK (n-2-2-2 mode) 10: 3 CLK (n-3-3-3 mode) 11: Reserved		Byte number in a page 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	
CSTMGC	Adjust for Timing of control signal	0168H			TACSEL1	TACSEL0			TAC1	TAC0
					R/W				R/W	
					0	0			0	0
					Select area to change timing 00:CS0 01:CS1 10:CS2 11:CS3				Select delay time(TAC) 00:0 × f_{SYS} 01:1 × f_{SYS} 10:2 × f_{SYS} 11:Reserved	
WRTMGCRR	Adjust for Timing of control signal	0169H			TCWSEL1	TCWSEL0	TCWS1	TCWS0	TCWH1	TCWH0
					R/W					
					0	0	0	0	0	0
					Select area to change timing 00:CS0 01:CS1 10:CS2 11:CS3		Select delay time(TCWS) 00:0.5 × f_{SYS} 01:1.5 × f_{SYS} 10:2.5 × f_{SYS} 11:3.5 × f_{SYS}		Select delay time(TCWH) 00:0.5 × f_{SYS} 01:1.5 × f_{SYS} 10:2.5 × f_{SYS} 11:3.5 × f_{SYS}	
RDTMGCR0	Adjust for Timing of control signal	016AH	B1TCRS1	B1TCRS0	B1TCRH1	B1TCRH0	B0TCRS1	B0TCRS0	B0TCRH1	B0TCRH0
			R/W							
			0	0	0	0	0	0	0	0
			Select delay time(TCRS) 00:0.5 × f_{SYS} 01:1.5 × f_{SYS} 10:2.5 × f_{SYS} 11:3.5 × f_{SYS}		Select delay time(TCRH) 00:0 × f_{SYS} 01:1 × f_{SYS} 10:2 × f_{SYS} 11:3 × f_{SYS}		Select delay time(TCRS) 00:0.5 × f_{SYS} 01:1.5 × f_{SYS} 10:2.5 × f_{SYS} 11:3.5 × f_{SYS}		Select delay time(TCRH) 00:0 × f_{SYS} 01:1 × f_{SYS} 10:2 × f_{SYS} 11:3 × f_{SYS}	
RDTMGCR1	Adjust for Timing of control signal	016BH	B3TCRS1	B3TCRS0	B3TCRH1	B3TCRH0	B2TCRS1	B2TCRS0	B2TCRH1	B2TCRH0
			R/W							
			0	0	0	0	0	0	0	0
			Select delay time(TCRS) 00:0.5 × f_{SYS} 01:1.5 × f_{SYS} 10:2.5 × f_{SYS} 11:3.5 × f_{SYS}		Select delay time(TCRH) 00:0 x× f_{SYS} 01:1 × f_{SYS} 10:2 × f_{SYS} 11:3 × f_{SYS}		Select delay time(TCRS) 00:0.5 × f_{SYS} 01:1.5 × f_{SYS} 10:2.5 × f_{SYS} 11:3.5 × f_{SYS}		Select delay time(TCRH) 00:0 × f_{SYS} 01:1 × f_{SYS} 10:2 × f_{SYS} 11:3 × f_{SYS}	
BROMCR	Boot Rom Control register	016CH						CSDIS	ROMLESS	VACE
								R/W		
								1	0/1	1/0
								Nand-Flash Area CS Output 0:enable 1:disable	Boot ROM 0: Use 1: No use	Vector address 0: Disable 1: Enable
RAMCR	RAM Control register	016DH								–
										R/W
										1
										Always write “1”

(4) TSI

Symbol	Name	Address	7	6	5	4	3	2	1	0
TSICR0	TSI control register0	01F0H	TSI7	INGE	PTST	TWIEN	PYEN	PXEN	MYEN	MXEN
			R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
			0	0	0	0	0	0	0	0
			0: Disable 1: Enable	Input gate control of Port 96,97 0: Enable 1: Disable	Detection condition 0: no touch 1: touch	INT4 interrupt control 0: Disable 1: Enable	SPY 0 : OFF 1 : ON	SPX 0 : OFF 1 : ON	SMY 0 : OFF 1 : ON	SMX 0 : OFF 1 : ON
TSICR1	TSI control register1	01F1H	DBC7	DB1024	DB256	DB64	DB8	DB4	DB2	DB1
			R/W							
			0	0	0	0	0	0	0	0
			0: Disable 1: Enable	1024	256	64	8	4	2	1
				De-bounce time is set by “(N*64-16) / f_{SYS}”-formula. “N” is sum of number which is set to “1” in bit6 to bit 0.						

(5) SDRAM controller

Symbol	Name	Address	7	6	5	4	3	2	1	0
SDACR	SDRAM access control register	0250H	SRDS	–	SMUXW1	SMUXW0	SPRE			SMAC
			R/W							R/W
			1	0	0	0	0			0
			Read data shift function 0: Disable 1: Enable	Always write “0”	Address multiplex type 00: Type A (A9-) 01: Type B (A10-) 10: Type C (A11-) 11: Reserved		Read/Write commands 0: Without auto pre-charge 1: With auto precharge			SDRAM controller 0: Disable 1: Enable
SDCISR	SDRAM Command Interval Setting Register	0251H		STMRD	STWR	STRP	STRCD	STRC2	STRC1	STRC0
				R/W						
				1	1	1	1	1	0	0
				TMRD 0: 1 CLK 1: 2 CLK	TWR 0: 1 CLK 1: 2 CLK	TRP 0: 1 CLK 1: 2 CLK	TRCD 0: 1 CLK 1: 2 CLK	TRC 000: 1 CLK 001: 2 CLK 010: 3 CLK 011: 4 CLK 100: 5 CLK 101: 6 CLK 110: 7 CLK 111: 8 CLK		
SDRCR	SDRAM refresh control register	0252H	–			SSAE	SRS2	SRS1	SRS0	SRC
			R/W			R/W				
			0			1	0	0	0	0
			Always write “0”			Self Refresh auto exit function 0:Disable 1:Enable	Refresh interval 000: 47 states 001: 78 states 010: 156 states 011: 312 states 100: 468 states 101: 624 states 110: 936 states 111: 1248 states			Auto Refresh 0:Disable 1:Enable
SDCMM	SDRAM command register	0253H						SCMM2	SCMM1	SCMM0
								R/W		
								0	0	0
								Command issue 000: Don’t care 001: Initialization sequence a. Precharge All command b. Eight Auto Refresh commands c. Mode Register Set command 010: Precharge All command 100: Reserved 101: Self Refresh Entry command 110: Self Refresh Exit command Others: Reserved		
SDBLS	SDRAM HDRAM burst length register	0254H			SDBL5	SDBL4	SDBL3	SDBL2	SDBL1	SDBL0
					0	0	0	0	0	0
					For HDMA5	For HDMA4	For HDMA3	For HDMA2	For HDMA1	For HDMA0
					HDMA burst length 0:1 Word Read / Single Write 1:Full Page Read / Burst Write					

(6) LCD controller (1/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCD MODE0	LCD mode0 register	0280H	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
			R/W							
			0	0	1	1	0	0	0	0
			Display RAM 00: Internal RAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transfer speed SCPW2= 0 00: 2-clock 01: 4-clock 10: 8-clock 11: 16-clock SCPW2= 1 00: 6-clock 01: 12-clock 10: 24-clock 11: 48-clock		Mode setting 0000 : Reserved 0001 : SR (mono) 0010 : SR (4Gray) 0011 : Reserved 0100 : SR (16Gray) 0101 : SR (64Gray) 0110 : STN (256 color) 0111 : STN (4096 color)		1000 : Reserved 1001 : Reserved 1010 : TFT (256 color) 1011 : TFT (4096 color) 1100 : TFT (64k color) 1101 : TFT256k,16M (color) 1110 : Reserved 1111 : Reserved	
LCD MODE1	LCD mode1 register	0281H	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
			R/W						W	W
			0	0	0	0	0	0	0	0
			Data rotation function (Supported for 64K-color: 16bps only) 000: Normal 001: Horizontal flip 010: Vertical flip 011: Horizontal & vertical flip		100: 90-degree 101: Reserved 110: Reserved 111: Reserved	LD bus Inversion 0: Normal 1: Inversion	Auto bus inversion 0: Disable 1: enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	FR edge 0: LHSYNC front edge 1:LHSYNC back edge	LD bus transfer speed 0: normal 1: 1/3
LCDDVM0	LCD divide frame0 register	0283H	FMP3	FMP2	FMP1	FMP0	FML3	FML2	FML1	FML0
			R/W							
			0	0	0	0	0	0	0	0
			LCP0 DVM (bits 3-0)				LHSYNC DVM (bits 3-0)			
LCDDVM1	LCD divide frame1 register	0288H	FMP7	FMP6	FMP5	FMP4	FML7	FML6	FML5	FML4
			R/W							
			0	0	0	0	0	0	0	0
			LCP0 DVM (bits 7-4)				LHSYNC DVM (bit 7-4)			
LCDSIZE	LCD size register	0284H	COM3	COM2	COM1	COM0	SEG3	SEG2	SEG1	SEG0
			R/W				R/W			
			0	0	0	0	0	0	0	0
			Common setting 0000 : reserved 0001 : 64 0010 : 96 0011 : 120 0100 : 128 0101 : 160 0110 : 200 0111 : 240		1000 : 320 1001 : 480 1010 : Reserved 1011 : Reserved 1100 : Reserved 1101 : Reserved 1110 : Reserved 1111 : Reserved		Segment setting 0000 : Reserved 0001 : 64 0010 : 128 0011 : 160 0100 : 240 0101 : 320 0110 : 480 0111 : 640		1000 : Reserved 1001 : Reserved 1010 : Reserved 1011 : Reserved 1100 : Reserved 1101 : Reserved 1110 : Reserved 1111 : Reserved	
LCDCTL0	LCD control0 register	0285H	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
			R/W					R/W		
			0	0	0	0		0	0	0
			PIP function 0:Disable 1:Enable	Segment Data 0:Normal 1: Always output “0”	FR divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

(6) LCD controller (2/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDCTL1	LCD control1 register	0286H	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
			R/W	R/W	R/W	R/W			R/W	R/W
			1	0	1	0			0	0
			LCP0 phase 0:Rising 1:Falling	LHSYNC phase 0:Rising 1: Falling	LVSYNC phase 0:Rising 1: Falling	LLOAD phase 0:Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	
LCDCTL2	LCD control2 register	0287H	LGOE2P	LGOE1P	LGOE0P					
			R/W							
			0	0	0					
			LGOE2 phase 0: Rising 1: Falling	LGOE1 phase 0: Rising 1: Falling	LGOE0 phase 0: Rising 1: Falling					
LCDHSP	LHSYNC Pulse register	028AH	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
			W							
			0	0	0	0	0	0	0	0
			LHSYNC period (bits 7-0)							
LCDHSP	LHSYNC Pulse register	028BH	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
			W							
			0	0	0	0	0	0	0	0
			LHSYNC period (bits 15-8)							
LCDVSP	LVSYNC Pulse register	028CH	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
			W							
			0	0	0	0	0	0	0	0
			LVSYNC period (bits 7-0)							
LCDVSP	LVSYNC Pulse register	028DH							LVP9	LVP8
									W	
									0	0
									LVSYNC period (bits 9-8)	
LCDPRVSP	LVSYNC Pre Pulse register	028EH		PLV6	PLV5	PLV4	PLV3	PLV2	PLV1	PLV0
				W						
				0	0	0	0	0	0	0
				Front dummy LVSYNC (bits 6-0)						
LCDHSDLY	LHSYNC Delay register	028FH		HSD6	HSD5	HSD4	HSD3	HSD2	HSD1	HSD0
				W						
				0	0	0	0	0	0	0
				LHSYNC delay (bits 6-0)						
LCDLDDLY	LLOAD Delay register	0290H	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
			R/W	W						
			0	0	0	0	0	0	0	0
			Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

(6) LCD controller (3/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDO0DLY	LGOE0 Delay register	0291H		OE0D6	OE0D5	OE0D4	OE0D3	OE0D2	OE0D1	OE0D0
				W						
				0	0	0	0	0	0	0
				OE0 delay (bits 6-0)						
LCDO1DLY	LGOE1 Delay register	0292H		OE1D6	OE1D5	OE1D4	OE1D3	OE1D2	OE1D1	OE1D0
				W						
				0	0	0	0	0	0	0
				OE1 delay (bits 6-0)						
LCDO2DLY	LGOE2 Delay register	0293H		OE2D6	OE2D5	OE2D4	OE2D3	OE2D2	OE2D1	OE2D0
				W						
				0	0	0	0	0	0	0
				OE2 delay (bits 6-0)						
LCDHSW	LHSYNC Width register	0294H	HSW7	HSW6	HSW5	HSW4	HSW3	HSW2	HSW1	HSW0
			W							
			0	0	0	0	0	0	0	0
			Setting bit7-0 for LHSYNC Width							
LCDLDW	LLOAD width register	0295H	LDW7	LDW6	LDW5	LDW4	LDW3	LDW2	LDW1	LDW0
			W							
			0	0	0	0	0	0	0	0
			LHSYNC width (bits 7-0)							
LCDHO0W	LGOE0 width register	0296H	O0W7	O0W6	O0W5	O0W4	O0W3	O0W2	O0W1	O0W0
			W							
			0	0	0	0	0	0	0	0
			LLOAD width (bits 7-0)							
LCDHO1W	LGOE1 width register	0297H	O1W7	O1W6	O1W5	O1W4	O1W3	O1W2	O1W1	O1W0
			W							
			0	0	0	0	0	0	0	0
			LGOE1 width (bits 7-0)							
LCDHO2W	LGOE2 width register	0298H	O2W7	O2W6	O2W5	O2W4	O2W3	O2W2	O2W1	O2W0
			W							
			0	0	0	0	0	0	0	0
			LGOE2 width (bits 7-0)							
LCDHWB8	Bit8,9 for signal width register	0299H	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
			W							
			0	0	0	0	0	0	0	0
			LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

(6) LCD controller (4/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LSAML	Start address register LCD main-L	02A0H	LMSA7	LMSA6	LMSA5	LMSA4	LMSA3	LMSA2	LMSA1	
			R/W							
			0	0	0	0	0	0	0	
			LCD main area start address (A7-A1)							
LSAMM	Start address register LCD main-M	02A1H	LMSA15	LMSA14	LMSA13	LMSA12	LMSA11	LMSA10	LMA9	LMSA8
			R/W							
			0	0	0	0	0	0	0	0
			LCD main area start address (A15-A8)							
LSAMH	Start address register LCD main-H	02A2H	LMSA23	LMSA22	LMSA21	LMSA20	LMSA19	LMSA18	LMSA17	LMSA16
			R/W							
			0	1	0	0	0	0	0	0
			LCD main area start address (A23-A16)							
LSASL	Start address register LCD sub-L	02A4H	LSSA7	LSSA6	LSSA5	LSSA4	LSSA3	LSSA2	LSSA1	
			R/W							
			0	0	0	0	0	0	0	
			LCD sub area start address (A7-A1)							
LSASM	Start address register LCD sub -M	02A5H	LSSA15	LSSA14	LSSA13	LSSA12	LSSA11	LSSA10	LSSA9	LSSA8
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area start address (A15-A8)							
LSASH	Start address register LCD sub -H	02A6H	LSSA23	LSSA22	LSSA21	LSSA20	LSSA19	LSSA18	LSSA17	LSSA16
			R/W							
			0	1	0	0	0	0	0	0
			LCD sub area start address (A23-A16)							
LSAHX	Hot point register LCD sub -X	02A8H	SAHX7	SAHX6	SAHX5	SAHX4	SAHX3	SAHX2	SAHX1	SAHX0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area HOT point (7-0)							
LSAHX	Hot point register LCD sub -X	02A9H							SAHX9	SAHX8
									R/W	
									0	0
									LCD sub area HOT point (9-8)	
LSAHY	Hot point register LCD sub -Y	02AAH	SAHY7	SAHY6	SAHY5	SAHY4	SAHY3	SAHY2	SAHY1	SAHY0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area HOT point (7-0)							
LSAHY	Hot point register LCD sub -Y	02ABH								SAHY8
										R/W
										0
										LCD sub area HOT point (9-8)
LSASS	Segment size register LCD sub	02ACH	SAS7	SAS6	SAS5	SAS4	SAS3	SAS2	SAS1	SAS0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area segment size (7-0)							
LSASS	Segment size register LCD sub	02ADH							SAS9	SAS8
									R/W	
									0	0
									LCD sub area segment size (9-8)	
LSACS	Common size register LCD sub	02AEH	SAC7	SAC6	SAC5	SAC4	SAC3	SAC2	SAC1	SAC0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area common size (7-0)							
LSACS	Common size register LCD sub	02AFH								SAC8
										R/W
										0
										LCD sub area common size (8)

(7) PMC

Symbol	Name	Address	7	6	5	4	3	2	1	0
PMCCTL	PMC Control Register	02A0H	PCM_ON					–	WUTM1	WUTM0
			R/W					W	R/W	R/W
		After system reset	0					0	0	0
		After Hot reset	Data retained					–	Data retained	Data retained
			Power Cut Mode 0: Disable 1: Enable					Always write “0” Always read as “0”	Warm-up time 00: 29 (15.625 ms) 01: 210 (31.25 ms) 10: 211 (62.5 ms) 11: 212 (125 ms)	

(8) USB controller (1/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
Descriptor RAM0	Descriptor RAM 0 register	0500H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM1	Descriptor RAM 1 register	0501H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM2	Descriptor RAM 2 register	0502H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM3	Descriptor RAM 3 register	0503H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
:	:	:	:							
:	:	:					:			
Descriptor RAM381	Descriptor RAM 381 register	067DH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM382	Descriptor RAM 382 register	067EH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM383	Descriptor RAM 383 register	067FH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint0	Endpoint 0 register	0780H	EP0_DATA7	EP0_DATA6	EP0_DATA5	EP0_DATA4	EP0_DATA3	EP0_DATA2	EP0_DATA1	EP0_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint1	Endpoint 1 register	0781H	EP1_DATA7	EP1_DATA6	EP1_DATA5	EP1_DATA4	EP1_DATA3	EP1_DATA2	EP1_DATA1	EP1_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint2	Endpoint 2 register	0782H	EP2_DATA7	EP2_DATA6	EP2_DATA5	EP2_DATA4	EP2_DATA3	EP2_DATA2	EP2_DATA1	EP2_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint3	Endpoint 3 register	0783H	EP3_DATA7	EP3_DATA6	EP3_DATA5	EP3_DATA4	EP3_DATA3	EP3_DATA2	EP3_DATA1	EP3_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
EP1_MODE	Endpoint 1 mode register	0789H			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0
EP2_MODE	Endpoint 2 mode register	078AH			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0
EP3_MODE	Endpoint 3 mode register	078BH			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0

(8) USB controller (2/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
EP0_STATUS	Endpoint 0 status register	0790H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP1_STATUS	Endpoint 1 status register	0791H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP2_STATUS	Endpoint 2 status register	0792H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP3_STATUS	Endpoint 3 status register	0793H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP0_SIZE_L_A	Endpoint 0 size register Low A	0798H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP1_SIZE_L_A	Endpoint 0 size register Low A	0799H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP2_SIZE_L_A	Endpoint 2 size register Low A	079AH	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP3_SIZE_L_A	Endpoint 3 size register Low A	079BH	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP1_SIZE_L_B	Endpoint 1 size register Low B	07A1H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP2_SIZE_L_B	Endpoint 2 size register Low B	07A2H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP3_SIZE_L_B	Endpoint 3 size register Low B	07A3H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP1_SIZE_H_A	Endpoint 1 size register High A	07A9H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP2_SIZE_H_A	Endpoint 2 size register High A	07AAH						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP3_SIZE_H_A	Endpoint 3 size register HighA	07ABH						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0

(8) USB controller (3/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
EP1_SIZE_H_B	Endpoint 1 size register High B	07B1H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP2_SIZE_H_B	Endpoint 2 size register High B	07B2H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP3_SIZE_H_B	Endpoint 0 size register High B	07B3H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
bmRequestType	bmRequest-Type register	07C0H	DIRECTION	REQ_TYPE1	REQ_TYPE0	RECIPIENT4	RECIPIENT3	RECIPIENT2	RECIPIENT1	RECIPIENT0
			R							
			0	0	0	0	0	0	0	0
bRequest	bRequest register	07C1H	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
			R							
			0	0	0	0	0	0	0	0
wValue_L	wValue register Low	07C2H	VALUE_L7	VALUE_L6	VALUE_L5	VALUE_L4	VALUE_L3	VALUE_L2	VALUE_L1	VALUE_L0
			R							
			0	0	0	0	0	0	0	0
wValue_H	wValue register High	07C3H	VALUE_H7	VALUE_H6	VALUE_H5	VALUE_H4	VALUE_H3	VALUE_H2	VALUE_H1	VALUE_H0
			R							
			0	0	0	0	0	0	0	0
wIndex_L	wIndex register Low	07C4H	INDEX_L7	INDEX_L6	INDEX_L5	INDEX_L4	INDEX_L3	INDEX_L2	INDEX_L1	INDEX_L0
			R							
			0	0	0	0	0	0	0	0
wIndex_H	wIndex register High	07C5H	INDEX_H7	INDEX_H6	INDEX_H5	INDEX_H4	INDEX_H3	INDEX_H2	INDEX_H1	INDEX_H0
			R							
			0	0	0	0	0	0	0	0
wLength_L	wLength register Low	07C6H	LENGTH_L7	LENGTH_L6	LENGTH_L5	LENGTH_L4	LENGTH_L3	LENGTH_L2	LENGTH_L1	LENGTH_L0
			R							
			0	0	0	0	0	0	0	0
wLength_H	wLength register High	07C7H	LENGTH_H7	LENGTH_H6	LENGTH_H5	LENGTH_H4	LENGTH_H3	LENGTH_H2	LENGTH_H1	LENGTH_H0
			R							
			0	0	0	0	0	0	0	0

(8) USB controller (4/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SetupReceived	SetupRecei-ved register	07C8H	D7	D6	D5	D4	D3	D2	D1	D0
			W							
			0	0	0	0	0	0	0	0
Current_Config	Current_ Config register	07C9H	REMOTEWAKEUP		ALTERNATE[1]	ALTERNATE[0]	INTERFACE[1]	INTERFACE[0]	CONFIG[1]	CONFIG[0]
			R		R					
			0		0	0	0	0	0	0
Standard Request	Standard-Request register	07CAH	S_INTERFACE	G_INTERFACE	S_CONFIG	G_CONFIG	G_DESCRIPT	S_FEATURE	C_FEATURE	G_STATUS
			R							
			0	0	0	0	0	0	0	0
Request	Request register	07CBH		SOFT_RESET	G_PORT_STS	G_DEVICE_ID	VENDOR	CLASS	ExSTANDARD	STANDARD
				R						
				0	0	0	0	0	0	0
DATASET1	DATASET 1 register	07CCH	EP3_DSET_B	EP3_DSET_A	EP2_DSET_B	EP2_DSET_A	EP1_DSET_B	EP1_DSET_A		EP0_DSET_A
			R							R
			0	0	0	0	0	0		0
DATASET2	DATASET 2 register	07CDH	EP7_DSET_B	EP7_DSET_A	EP6_DSET_B	EP6_DSET_A	EP5_DSET_B	EP5_DSET_A	EP4_DSET_B	EP4_DSET_A
			R							
			0	0	0	0	0	0	0	0
USB_STATE	USB state register	07CEH						Configured	Addressed	Default
								R/W	R	
								0	0	1
EOP	EOP register	07CFH	EP7_EOPB	EP6_EOPB	EP5_EOPB	EP4_EOPB	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
			W							
			1	1	1	1	1	1	1	1
COMMAND	Command register	07D0H		EP[2]	EP[1]	EP[0]	Command[3]	Command[2]	Command[1]	Command[0]
				W						
				0	0	0	0	0	0	0
EPx_SINGLE1	Endpoint 1 single register	07D1H	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	
			R/W				R/W			
			0	0	0		0	0	0	
EPx_BCS1	Endpoint 1 BCS register	07D3H	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_BCS	EP2_BCS	EP1_BCS	
			R/W				R/W			
			0	0	0		0	0	0	
INT_Control	Interrupt control register	07D6H								Status_nak
										R/W
										0
Standard Request Mode	Standard Request mode register	07D8H	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
			R/W							
			0	0	0	0	0	0	0	0
Request Mode	Request mode register	07D9H		Soft_Reset	G_Port_Sts	G_DeviceId				
				R/W						
				0	0	0				

(8) USB controller (5/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
Port Status	Port status register	07E0H	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
			W							
			0	0	0	1	1	0	0	0
FRAME_L	Frame register Low	07E1H	–	T[6]	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
			R							
			0	0	0	0	0	0	0	0
FRAME_H	Frame register H	07E2H	T[10]	T[9]	T[8]	T[7]		CREATE	FRAME_STS1	FRAME_STS0
			R					R		
			0	0	0	0		0	1	0
ADDRESS	Address register	07E3H		A6	A5	A4	A3	A2	A1	A0
				R						
				0	0	0	0	0	0	0
USBREADY	USB ready register	07E6H								USBREADY
										R/W
										0
Set Descriptor STALL	Set-Descriptor stall register	07E8H								S_D_STALL
										W
										0
USBINTFR1	USB interrupt flag register 1	07F0H (Prohibit RMW)	INT_URST_STR	INT_URST_END	INT_SUS	INT_RESUME	INT_CLKSTOP	INT_CLKON		
			R/W							
			0	0	0	0	0	0		
			When read 0: Not generate interrupt 1: Generate interrupt			When write 0: Clear flag 1: –				
USBINTFR2	USB interrupt flag register 2	07F1H (Prohibit RMW)	EP1_FULL_A	EP1_Empty_A	EP1_FULL_B	EP1_Empty_B	EP2_FULL_A	EP2_Empty_A	EP2_FULL_B	EP2_Empty_B
			R/W							
			0	0	0	0	0	0	0	0
			When read 0: Not generate interrupt 1: Generate interrupt			When write 0: Clear flag 1: –				
USBINTFR3	USB interrupt flag register 3	07F2H (Prohibit RMW)	EP3_FULL_A	EP3_Empty_A	EP3_FULL_B	EP3_Empty_B				
			R/W							
			0	0	0	0				
			When read 0:Not generate interrupt 1:Generate interrupt When write 0: Clear flag 1: –							
USBINTFR4	USB interrupt flag register 4	07F3H (Prohibit RMW)	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	
			R/W							
			0	0	0	0	0	0	0	
			When read 0: Not generate interrupt 1: Generate interrupt			When write 0: Clear flag 1: –				

(8) USB controller (6/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
USBINTMR1	USB interrupt mask register 1	07F4H	MSK_URST_STR	MSK_URST_END	MSK_SUS	MSK_RESUME	MSK_CLKSTOP	MSK_CLKON		
			R/W							
			1	1	1	1	1	1		
			0: Be not masked 1: Be masked							
USBINTMR2	USB interrupt mask register 2	07F5H	EP1_MSK_FA	EP1_MSK_EA	EP1_MSK_FB	EP1_MSK_EB	EP2_MSK_FA	EP2_MSK_EA	EP2_MSK_FB	EP2_MSK_EB
			R/W							
			1	1	1	1	1	1	1	1
			0: Be not masked 1: Be masked							
USBINTMR3	USB interrupt mask register 3	07F6H	EP3_MSK_FA	EP3_MSK_EA						
			R/W							
			1	1						
			0: Be not masked 1: Be masked							
USBINTMR4	USB interrupt mask register 4	07F7H	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	
			R/W							
			1	1	1	1	1	1	1	
			0: Be not masked 1: Be masked							
USBCR1	USB control register 1	07F8H	TRNS_USE	WAKEUP					SPEED	USBCLKE
			R/W						R/W	
			0	0					1	0
			Transceiver 0:disable 1:enble	Wake up 0: – 1:Start						

(9) SPIC (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SPIMD	SPI Mode Setting register	0820H (Prohibit RMW)	SWRST	XEN				CLKSEL2	CLKSEL1	CLKSEL0
			W	R/W				R/W		
			0	0				1	0	0
			Software reset 0: don't care 1: Reset	SYSCK 0: disable 1: enable				Select Baud Rate 000:Reserved 100: $f_{SYS}/8$ 001: $f_{SYS}/2$ 101: $f_{SYS}/16$ 010: $f_{SYS}/3$ 110: $f_{SYS}/64$ 011: $f_{SYS}/4$ 111: $f_{SYS}/256$		
		0821H (Prohibit RMW)	LOOPBACK	MSB1ST	DOSTAT		TCPOL	RCPOL	TDINV	RDINV
			R/W				R/W			
			0	1	1		0	0	0	0
			LOOPBACK Test mode 0:disbale 1:enable	Start bit for Transmit / Receive 0:LSB 1:MSB	SPDO pin state (no transmit) 0:fixed to "0" 1:fixed to "1"		Synchronous clock edge during transmitting 0: fall 1: rise	Synchronous clock edge during receiving 0: fall 1: rise	Invert data During transmitting 0: disable 1: enable	Invert data During receiving 0: disable 1: enable
SPICT	SPI Control register	0822H	CEN	SPCS_B	UNIT16	TXMOD	TXE	FDPXE	RXMOD	RXE
			R/W							
			0	1	0	0	0	0	0	0
			communication control 0: disable 1: enable	$\overline{SPCS}$ pin 0: output "0" 1: output "1"	Data length 0: 8bit 1: 16bit	Transmit mode 0: UNIT 1: Sequential	Transmit control 0: disable 1: enable	Alignment in Full duplex 0: disable 1: enable	Receive Mode 0: UNIT 1: Sequential	Receive control 0: disable 1: enable
		0823H	CRC16_7_B	CRCRX_TX_B	CRCRESET_B					
			R/W							
			0	0	0					
			CRC select 0: CRC7 1: CRC16	CRC data 0: Transmit 1: receive	CRC calculate register 0:Reset 1:Release Reset					
SPIST	SPI Status register	0824H					TEMP		TEND	REND
							R		R	
							1		1	0
							Transmit FIFO Status 0: no space 1: having space		Transmit Status 0: during transmission or having transmission data 1: finish	Receive Status 0: during receiving or not having receiving data 1: finish or not having space
		0825H								
SPIIE	SPI Interrupt enable register	082CH					TEMPIE	RFULIE	TENDIE	RENDIE
							R/W			
							0	0	0	0
							TEMP interrupt 0:enable 1:disable	RFUL interrupt 0:enable 1:disable	TEND interrupt 0:enable 1:disable	REND interrupt 0:enable 1:disable
		082DH								

(9) SPIC (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SPICR	SPI CRC register	0826H	CRCD7	CRCD6	CRCD5	CRCD4	CRCD3	CRCD2	CRCD1	CRCD0
			R							
			0	0	0	0	0	0	0	0
			CRC result register [7:0]							
		0827H	CRCD15	CRCD14	CRCD13	CRCD12	CRCD11	CRCD10	CRCD9	CRCD8
			R							
			0	0	0	0	0	0	0	0
			CRC result register [15:8]							
SPITD0	SPI transmission data0 register	0830H	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [7:0]							
		0831H	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [15:8]							
SPITD1	SPI transmission data1 register	0832H	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [7:0]							
		0833H	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [15:8]							
SPIRD0	SPI receive data0 register	0834H	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
			R							
			0	0	0	0	0	0	0	0
			Receive data register [7:0]							
		0835H	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
			R							
			0	0	0	0	0	0	0	0
			Receive data register [15:8]							
SPIRD1	SPI receive data1 register	0836H	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
			R							
			0	0	0	0	0	0	0	0
			Receive data register [7:0]							
		0837H	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
			R							
			0	0	0	0	0	0	0	0
			Receive data register [15:8]							

(10) MMU (1/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALPX	LOCALX register for program	0880H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)							
LOCALPX	LOCALX register for program	0881H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALPY	LOCALY register for program	0882H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)					
LOCALPY	LOCALY register for program	0883H	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALPZ	LOCALZ register for program	0884H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)							
LOCALPZ	LOCALZ register for program	0885H	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (2/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALLX	LOCALX register for LCD	0888H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALLX	LOCALX register for LCD	0889H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALLY	LOCALY register for LCD	088AH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALLY	LOCALY register for LCD	088BH	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALLZ	LOCALZ register for LCD	088CH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALLZ	LOCALZ register for LCD	088DH	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (3/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALRX	LOCALX register for read	0890H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALRX	LOCALX register for read	0891H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALRY	LOCALY register for read	0892H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALRY	LOCALY register for read	0893H	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALRZ	LOCALZ register for read	0894H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALRZ	LOCALZ register for read	0895H	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (4/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALWX	LOCALX register for write	0898H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALWX	LOCALX register for write	0899H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALWY	LOCALY register for write	089AH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALWY	LOCALY register for write	089BH	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALWZ	LOCALZ register for write	089CH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALWZ	LOCALZ register for write	089DH	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (5/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALESX	LOCALX register for DMA source	08A0H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALESX	LOCALX register for DMA source	08A1H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALESY	LOCALY register for DMA source	08A2H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALESY	LOCALY register for DMA source	08A3H	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALESZ	LOCALZ register for DMA source	08A4H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALESZ	LOCALZ register for DMA source	08A5H	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (6/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALEDX	LOCALX register for DMA destination	08A8H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALEDX	LOCALX register for DMA destination	08A9H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALEDY	LOCALY register for DMA destination	08AAH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALEDY	LOCALY register for DMA destination	08ABH	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALEDZ	LOCALZ register for DMA destination	08ACH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALEDZ	LOCALZ register for DMA destination	08ADH	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (7/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALOSX	LOCALX register for DMA source	08B0H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALOSX	LOCALX register for DMA source	08B1H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALOSY	LOCALY register for DMA source	08B2H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALOSY	LOCALY register for DMA source	08B3H	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALOSZ	LOCALZ register for DMA source	08B4H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALOSZ	LOCALZ register for DMA source	08B5H	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(10) MMU (8/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALODX	LOCALX register for DMA destination	08B8H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-X (“0” is disabled because of overlapped with Common-area.)							
LOCALODX	LOCALX register for DMA destination	08B9H	LXE							X8
			R/W							R/W
			0							0
			LOCALX BANK 0:disable 1:enable	Set BANK number for LOCAL-X X8-X0 setting and CS 000000000~011111111 CSXA 100000000~111111111 CSXB						
LOCALODY	LOCALY register for DMA destination	08BAH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Set BANK number for LOCAL-Y (“3” is disabled because of overlapped with Common-area.)					
LOCALODY	LOCALY register for DMA destination	08BBH	LYE							
			R/W							
			0							
			LOCALY BANK 0:disable 1:enable							
LOCALODZ	LOCALZ register for DMA destination	08BCH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Set BANK number for LOCAL-Z (“3” is disabled because of overlapped with Common-area.)							
LOCALODZ	LOCALZ register for DMA destination	08BDH	LZE							Z8
			R/W							R/W
			0							0
			LOCALZ BANK 0:disable 1:enable	Set BANK number for LOCAL-Z Z8-Z0 setting and CS 000000000~001111111 CSZA 100000000~101111111 CSZC 010000000~011111111 CSZB 110000000~111111111 CSZD						

(11) NAND-Flash controller (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDFMCR0	NANDF Control0 Register	08C0H (Prohibit RMW)	WE	ALE	CLE	CE0	CE1	ECCE	BUSY	ECCRST
			R/W							
			0	0	0	0	0	0	0	0
			WE enable 0: Disable 1: Enable	ALE control 0: "L" out 1: "H" out	CLE control 0: "L" out 1: "H" out	$\overline{CE0}$ control 0: "H" out 1: "L" out	$\overline{CE1}$ control 0: "H" out 1: "L" out	ECC circuit control 0: Disable 1: Enable	NAND Flash state 1: Busy 0: Ready	ECC reset control 0: – 1: Reset *Always read as "0".
		08C1H (Prohibit RMW)	SPLW1	SPLW0	SPHW1	SPHW0	RSECCL	RSEDN	RSESTA	RSECGW
			R/W						W	R/W
			0	0	0	0	0	0	0	0
			Strobe pulse width ($\overline{Low}$ width of $\overline{NDRE}$, $\overline{NDWE}$) Inserted width = (f_{SYS}) × (set value)		Strobe pulse width ($\overline{High}$ width of $\overline{NDRE}$, $\overline{NDWE}$) Inserted width = (f_{SYS}) × (set value)		Reed-Solomon ECC latch 0: Disable 1: Enable	Reed-Solomon operation 0: Encode (Write) 1: Decode (Read)	Reed-Solomon error calculation start 0: – 1: Start *Always read as "0".	Reed-Solomon ECC generator write control 0: Disable 1: Enable
NDFMCR1	NANDF Control1 Register	08C2H	INTERDY	INTRSC				BUSW	ECCS	SYSCKE
			R/W	R/W				R/W	R/W	R/W
			0	0				0	0	0
			Ready interrupt 0: Disable 1: Enable	Reed-Solomon calculation end interrupt 0: Disable 1: Enable				Data bus width 0: 8-bit 1: 16-bit	ECC calculation 0:Hamming 1: Reed-Solomon	Clock control 0: Disable 1: Enable
		08C3H	STATE3	STATE2	STATE1	STATE0	SEER1	SEER0		
			R							
			0	0	0	0	Undefined	Undefined		
			Status read (See the table below.)							
NDECCRD0	NANDF Code ECC Register0	08C4H	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0) (7-0)							
		08C5H	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0) (15-8)							
NDECCRD1	NANDF Code ECC Register1	08C6H	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0) (7-0)							
		08C7H	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0) (15-8)							

(11) NAND-Flash controller (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDECCRD2	NANDF Code ECC Register2	08C8H	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08C9H	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							
NDECCRD3	NANDF Code ECC Register3	08CAH	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08CBH	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							
NDECCRD4	NANDF Code ECC Register4	08CCH	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08CDH	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							

(11) NAND-Flash controller (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDRSCA0	NANDF read solomon Result address Register0	08D0H	RS0A7	RS0A6	RS0A5	RS0A4	RS0A3	RS0A2	RS0A1	RS0A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08D1H							RS0A9	RS0A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD0	NANDF read solomon Result data Register0	08D2H	RS0D7	RS0D6	RS0D5	RS0D4	RS0D3	RS0D2	RS0D1	RS0D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
NDRSCA1	NANDF read solomon Result address Register1	08D4H	RS1A7	RS1A6	RS1A5	RS1A4	RS1A3	RS1A2	RS1A1	RS1A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08D5H							RS1A9	RS1A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD1	NANDF read solomon Result data Register1	08D6H	RS1D7	RS1D6	RS1D5	RS1D4	RS1D3	RS1D2	RS1D1	RS1D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
NDRSCA2	NANDF read solomon Result address Register2	08D8H	RS2A7	RS2A6	RS2A5	RS2A4	RS2A3	RS2A2	RS2A1	RS2A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08D9H							RS2A9	RS2A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD2	NANDF read solomon Result data Register2	08DAH	RS2D7	RS2D6	RS2D5	RS2D4	RS2D3	RS2D2	RS2D1	RS2D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							

(11) NAND-Flash controller (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDRSCA3	NANDF read solomon Result address Register3	08DCH	RS3A7	RS3A6	RS3A5	RS3A4	RS3A3	RS3A2	RS3A1	RS3A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08DDH							RS3A9	RS3A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD3	NANDF read solomon Result data Register3	08DEH	RS2D7	RS2D6	RS2D5	RS2D4	RS2D3	RS2D2	RS2D1	RS2D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
NDFDTR0	NANDF Data Register0	1FF0H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (7-0)							
		1FF1H	D15	D14	D13	D12	D11	D10	D9	D8
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (15-8)							
NDFDTR1	NANDF Data Register1	1FF2H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (7-0)							
		1FF3H	D15	D14	D13	D12	D11	D10	D9	D8
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (15-8)							

(12) DMAC (1/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS0	DMA source address Register0	0900H	D0SA7	D0SA6	D0SA5	D0SA4	D0SA3	D0SA2	D0SA1	D0SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA0 (7:0)							
		0901H	D0SA15	D0SA14	D0SA13	D0SA12	D0SA11	D0SA10	D0SA9	D0SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA0 (15:8)							
		0902H	D0SA23	D0SA22	D0SA21	D0SA20	D0SA19	D0SA18	D0SA17	D0SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA0 (23:16)							
HDMAD0	DMA destination address Register0	0904H	D0DA7	D0DA6	D0DA5	D0DA4	D0DA3	D0DA2	D0DA1	D0DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA0 (7:0)							
		0905H	D0DA15	D0DA14	D0DA13	D0DA12	D0DA11	D0DA10	D0DA9	D0DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA0 (15:8)							
		0906H	D0DA23	D0DA22	D0DA21	D0DA20	D0DA19	D0DA18	D0DA17	D0DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA0 (23:16)							
HDMACA0	DMA Transfer count number A Register0	0908H	D0CA7	D0CA6	D0CA5	D0CA4	D0CA3	D0CA2	D0CA1	D0CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [7:0] for DMA0							
		0909H	D0CA15	D0CA14	D0CA13	D0CA12	D0CA11	D0CA10	D0CA9	D0CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA0							
HDMACB0	DMA Transfer count number B Register0	090AH	D0CB7	D0CB6	D0CB5	D0CB4	D0CB3	D0CB2	D0CB1	D0CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [7:0] for DMA0							
		090BH	D0CB15	D0CB14	D0CB13	D0CB12	D0CB11	D0CB10	D0CB9	D0CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [15:8] for DMA0							
HDMAM0	DMA transfer Mode Register0	090CH				D0M4	D0M3	D0M2	D0M1	D0M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (2/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS1	DMA source address Register1	0910H	D1SA7	D1SA6	D1SA5	D1SA4	D1SA3	D1SA2	D1SA1	D1SA0
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA1 (7:0)							
		0911H	D1SA15	D1SA14	D1SA13	D1SA12	D1SA11	D1SA10	D1SA9	D1SA8
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA1 (15:8)							
		0912H	D1SA23	D1SA22	D1SA21	D1SA20	D1SA19	D1SA18	D1SA17	D1SA16
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA1 (23:16)							
HDMAD1	DMA destination address Register1	0914H	D1DA7	D1DA6	D1DA5	D1DA4	D1DA3	D1DA2	D1DA1	D1DA0
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA1 (7:0)							
		0915H	D1DA15	D1DA14	D1DA13	D1DA12	D1DA11	D1DA10	D1DA9	D1DA8
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA1 (15:8)							
		0916H	D1DA23	D1DA22	D1DA21	D1DA20	D1DA19	D1DA18	D1DA17	D1DA16
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA1 (23:16)							
HDMACA1	DMA Transfer count number A Register1	0918H	D1CA7	D1CA6	D1CA5	D1CA4	D1CA3	D1CA2	D1CA1	D1CA0
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number A for DMA1 (7:0)							
		0919H	D1CA15	D1CA14	D1CA13	D1CA12	D1CA11	D1CA10	D1CA9	D1CA8
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number A for DMA1 (15:8)							
HDMACB1	DMA Transfer count number B Register1	091AH	D1CB7	D1CB6	D1CB5	D1CB4	D1CB3	D1CB2	D1CB1	D1CB0
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number B for DMA1 (7:0)							
		091BH	D0CB15	D0CB14	D0CB13	D0CB12	D0CB11	D0CB10	D0CB9	D0CB8
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number B for DMA1 (15:8)							
HDMAM1	DMA transfer Mode Register1	091CH				D1M4	D1M3	D1M2	D1M1	D1M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (3/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS2	DMA source address Register2	0920H	D2SA7	D2SA6	D2SA5	D2SA4	D2SA3	D2SA2	D2SA1	D2SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA2 (7:0)							
		0921H	D2SA15	D2SA14	D2SA13	D2SA12	D2SA11	D2SA10	D2SA9	D2SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA2 (15:8)							
		0922H	D2SA23	D2SA22	D2SA21	D2SA20	D2SA19	D2SA18	D2SA17	D2SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA2 (23:16)							
HDMAD2	DMA destination address Register2	0924H	D2DA7	D2DA6	D2DA5	D2DA4	D2DA3	D2DA2	D2DA1	D2DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA2 (7:0)							
		0925H	D2DA15	D2DA14	D2DA13	D2DA12	D2DA11	D2DA10	D2DA9	D2DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA2 (15:8)							
		0926H	D2DA23	D2DA22	D2DA21	D2DA20	D2DA19	D2DA18	D2DA17	D2DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA2 (23:16)							
HDMACA2	DMA Transfer count number A Register2	0928H	D2CA7	D2CA6	D2CA5	D2CA4	D2CA3	D2CA2	D2CA1	D2CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [7:0] for DMA2							
		0929H	D2CA15	D2CA14	D2CA13	D2CA12	D2CA11	D2CA10	D2CA9	D2CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA2							
HDMACB2	DMA Transfer count number B Register2	092AH	D2CB7	D2CB6	D2CB5	D2CB4	D2CB3	D2CB2	D2CB1	D2CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [7:0] for DMA2							
		092BH	D2CB15	D2CB14	D2CB13	D2CB12	D2CB11	D2CB10	D2CB9	D2CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA2							
HDMAM2	DMA transfer Mode Register2	092CH				D2M4	D2M3	D2M2	D2M1	D2M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (4/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS3	DMA source address Register3	0930H	D3SA7	D3SA6	D3SA5	D3SA4	D3SA3	D3SA2	D3SA1	D3SA0
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA3 (7:0)							
		0931H	D3SA15	D3SA14	D3SA13	D3SA12	D3SA11	D3SA10	D3SA9	D3SA8
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA3 (15:8)							
		0932H	D3SA23	D3SA22	D3SA21	D3SA20	D3SA19	D3SA18	D3SA17	D3SA16
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA3 (23:16)							
HDMAD3	DMA destination address Register3	0934H	D3DA7	D3DA6	D3DA5	D3DA4	D3DA3	D3DA2	D3DA1	D3DA0
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA3 (7:0)							
		0935H	D3DA15	D3DA14	D3DA13	D3DA12	D3DA11	D3DA10	D3DA9	D3DA8
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA3 (15:8)							
		0936H	D3DA23	D3DA22	D3DA21	D3DA20	D3DA19	D3DA18	D3DA17	D3DA16
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA3 (23:16)							
HDMACA3	DMA Transfer count number A Register3	0938H	D3CA7	D3CA6	D3CA5	D3CA4	D3CA3	D3CA2	D3CA1	D3CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [7:0] for DMA3							
		0939H	D3CA15	D3CA14	D3CA13	D3CA12	D3CA11	D3CA10	D3CA9	D3CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA3							
HDMACB3	DMA Transfer count number B Register3	093AH	D3CB7	D3CB6	D3CB5	D3CB4	D3CB3	D3CB2	D3CB1	D3CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [7:0] for DMA3							
		093BH	D3CB15	D3CB14	D3CB13	D3CB12	D3CB11	D3CB10	D3CB9	D3CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [15:8] for DMA3							
HDMAM3	DMA transfer Mode Register3	093CH				D3M4	D3M3	D3M2	D3M1	D3M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (5/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS4	DMA source address Register4	0940H	D4SA7	D4SA6	D4SA5	D4SA4	D4SA3	D4SA2	D4SA1	D4SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA4 (7:0)							
		0941H	D4SA15	D4SA14	D4SA13	D4SA12	D4SA11	D4SA10	D4SA9	D4SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA4 (15:8)							
		0942H	D4SA23	D4SA22	D4SA21	D4SA20	D4SA19	D4SA18	D4SA17	D4SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA4 (23:16)							
HDMAD4	DMA destination address Register4	0944H	D4DA7	D4DA6	D4DA5	D4DA4	D4DA3	D4DA2	D4DA1	D4DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA4 (7:0)							
		0945H	D4DA15	D4DA14	D4DA13	D4DA12	D4DA11	D4DA10	D4DA9	D4DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA4 (15:8)							
		0946H	D4DA23	D4DA22	D4DA21	D4DA20	D4DA19	D4DA18	D4DA17	D4DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA4 (23:16)							
HDMACA4	DMA Transfer count number A Register4	0948H	D4CA7	D4CA6	D4CA5	D4CA4	D4CA3	D4CA2	D4CA1	D4CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA4							
		0949H	D4CA15	D4CA14	D4CA13	D4CA12	D4CA11	D4CA10	D4CA9	D4CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA4							
HDMACB4	DMA Transfer count number B Register4	094AH	D4CB7	D4CB6	D4CB5	D4CB4	D4CB3	D4CB2	D4CB1	D4CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [7:0] for DMA4							
		094BH	D4CB15	D4CB14	D4CB13	D4CB12	D4CB11	D4CB10	D4CB9	D4CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [15:8] for DMA4							
HDMAM4	DMA transfer Mode Register4	094CH				D4M4	D4M3	D4M2	D4M1	D4M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (6/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS5	DMA source address Register5	0950H	D5SA7	D5SA6	D5SA5	D5SA4	D5SA3	D5SA2	D5SA1	D5SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA5 (7:0)							
		0951H	D5SA15	D5SA14	D5SA13	D5SA12	D5SA11	D5SA10	D5SA9	D5SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA5 (15:8)							
		0952H	D5SA23	D5SA22	D5SA21	D5SA20	D5SA19	D5SA18	D5SA17	D5SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA5 (23:16)							
HDMAD5	DMA destination address Register5	0954H	D5DA7	D5DA6	D5DA5	D5DA4	D5DA3	D5DA2	D5DA1	D5DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA5 (7:0)							
		0955H	D5DA15	D5DA14	D5DA13	D5DA12	D5DA11	D5DA10	D5DA9	D5DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA5 (15:8)							
		0956H	D5DA23	D5DA22	D5DA21	D5DA20	D5DA19	D5DA18	D5DA17	D5DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA5 (23:16)							
HDMACA5	DMA Transfer count number A Register5	0958H	D5CA7	D5CA6	D5CA5	D5CA4	D5CA3	D5CA2	D54CA1	D5CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [7:0] for DMA5							
		0959H	D5CA15	D5CA14	D5CA13	D5CA12	D5CA11	D5CA10	D5CA9	D5CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A [15:8] for DMA5							
HDMACB5	DMA Transfer count number B Register5	095AH	D5CB7	D5CB6	D5CB5	D5CB4	D5CB3	D5CB2	D5CB1	D5CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [7:0] for DMA5							
		095BH	D5CB15	D5CB14	D5CB13	D5CB12	D5CB11	D5CB10	D5CB9	D5CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B [15:8] for DMA5							
HDMAM5	DMA transfer Mode Register5	095CH				D5M4	D5M3	D5M2	D5M1	D5M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (7/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAE	DMA enable Register	097EH			DMAE5	DMAE4	DMAE3	DMAE2	DMAE1	DMAE0
					R/W					
					0	0	0	0	0	0
					DMA channel operation 0: Disable 1: Enable					
HDMATR	DMA timer Register	097FH	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
			R/W							
			0	0	0	0	0	0	0	0
			Timer operation 0: Disable 1: Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by "Maximum bus occupancy time / (256/f_{SYS})". "00H" cannot be set.						

(13) Clock gear, PLL

Symbol	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System clock control register0	10E0H		XTEN	USBCLK1	USBCLK0		WUEF		PRCK
				R/W				R/W		R/W
				1	0	0		0		0
				Low -frequency oscillator circuit (fs) 0: Stop 1: Oscillation	Select the clock of USB(f_{USB}) 00: Disable 01: Reserved 10: X1USB 11: f_{PLLUSB}			Warm-up timer		Select Prescaler clock 0: f_{SYS}/2 1: f_{SYS}/8
SYSCR1	System clock control register1	10E1H						GEAR2	GEAR1	GEAR0
								R/W		
								1	0	0
								Select gear value of high frequency (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8 101: (Reserved) 110: (Reserved) 111: (Reserved) 100: fc/16		
SYSCR2	System clock control register2	10E2H	–	CKOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0		
			R/W							
			0	0	1	0	1	1		
			Always write “0”.	Select CLKOUT 0: f_{SYS} 1: fs	Warm-Up Timer 00: Reserved 01: 2^8/inputted frequency 10:2^{14}/inputted frequency 11:2^{16}/inputted frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			
EMCCR0	EMC control register0	10E3H	PROTECT				–	EXTIN	DRVOSCH	DRVOSCL
			R				R/W	R/W	R/W	R/W
			0				0	0	1	1
			Protect flag 0: OFF 1: ON				Always write “0”.	1: External clock	fc oscillator drive ability 1: NORMAL 0: WEAK	fs oscillator drive ability 1: NORMAL 0: WEAK
EMCCR1	EMC control register1	10E4H	Switching the protect ON/OFF by write to following 1st-KEY,2nd-KEY 1st-KEY: EMCCR1=5AH,EMCCR2=A5H in succession write 2nd-KEY: EMCCR1=A5H,EMCCR2=5AH in succession write							
EMCCR2	EMC control register2	10E5H								
PLLCR0	PLL control register0	10E8H		FCSEL	LUPFG					
				R/W	R					
				0	0					
				Select fc clock 0 : f_{OSCH} 1 : f_{PLL}	Lock-up timer Status flag 0 : not end 1 : end					
PLLCR1	PLL control register1	10E9H	PLL0	PLL1	LUPSEL					PLLTIMES
			R/W							R/W
			0	0	0					0
			PLL0 for CPU 0: Off 1: On	PLL1 for USB 0: Off 1: On	Select stage of Lock up counter 0: 12 stage (for PLL0) 1:13 stage (for PLL1)					Select the number of PLL 0: ×12 1: ×16

(14) 8-bit timer (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA01RUN	TMRA01 RUN register	1100H	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA01 prescaler 0: Stop and clear 1: Run (Count up)	Up counter (UC1)	Up counter (UC0)
TA0REG	8-bit timer register 0	1102H (Prohibit RMW)	–							
			W							
			0							
TA1REG	8-bit timer register 1	1103H (Prohibit RMW)	–							
			W							
			0							
TA01MOD	TMRA01 MODE register	1104H	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA0 00: TA0IN pin 01: ϕT1 10: ϕT4 11: ϕT16	
TA1FFCR	TMRA1 Flip-Flop control register	1105H (Prohibit RMW)					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don't care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1: TMRA1
TA23RUN	TMRA23 RUN register	1108H	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA23 prescaler 0: Stop and clear 1: Run (Count up)	Up counter (UC3)	Up counter (UC2)
TA2REG	8-bit timer register 2	110AH (Prohibit RMW)	–							
			W							
			0							
TA3REG	8-bit timer register 3	110BH (Prohibit RMW)	–							
			W							
			0							
TA23MOD	TMRA23 MODE register	110CH	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA3 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA2 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA3FFCR	TMRA3 Flip-Flop control register	110DH (Prohibit RMW)					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don't care		TA3FF control for inversion 0: Disable 1: Enable	TA3FF inversion select 0: TMRA2 1: TMRA3

(14) 8-bit timer (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA45RUN	TMRA45 RUN register	1110H	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA45 prescaler 0: Stop and clear 1: Run (Count up)	Up counter (UC5)	Up counter (UC4)
TA4REG	8-bit timer register 4	1112H (Prohibit RMW)	–							
			W							
			0							
TA5REG	8-bit timer register 5	1113H (Prohibit RMW)	–							
			W							
			0							
TA45MOD	TMRA45 MODE register	1114H	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA4 00: 32KHz clock 01: ϕT1 10: ϕT4 11: ϕT16	
TA5FFCR	TMRA5 Flip-Flop control register	1115H (Prohibit RMW)					TA5FFC1	TA5FFC0	TA5FFIE	TA5FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF control for inversion 0: Disable 1: Enable	TA5FF inversion select 0: TMRA4 1: TMRA5
TA67RUN	TMRA67 RUN register	1118H	TA6RDE				I2TA67	TA67PRUN	TA7RUN	TA6RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA67 prescaler 0: Stop and clear 1: Run (Count up)	Up counter (UC7)	Up counter (UC6)
TA6REG	8-bit timer register 2	111AH (Prohibit RMW)	–							
			W							
			0							
TA7REG	8-bit timer register 3	111BH (Prohibit RMW)	–							
			W							
			0							
TA67MOD	TMRA67 MODE register	111CH	TA67M1	TA67M0	PWM61	PWM60	TA7CLK1	TA7CLK0	TA6CLK1	TA6CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA7 00: TA6TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA6 00: 32KHz clock 01: ϕT1 10: ϕT4 11: ϕT16	
TA7FFCR	TMRA7 Flip-Flop control register	111DH (Prohibit RMW)					TA7FFC1	TA7FFC0	TA7FFIE	TA7FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA7FF 01: Set TA7FF 10: Clear TA7FF 11: Don't care		TA7FF control for inversion 0: Disable 1: Enable	TA7FF inversion select 0: TMRA6 1: TMRA7

(15) 16-bit timer (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB0RUN	TMRB0 RUN register	1180H	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffer 0: disable 1: enable	Always write “0”.			IDLE2 0: Stop 1: Operate	TMRB0 prescaler 0: Stop and clear 1: Run (Count up)		Up counter (UC10)
TB0MOD	TMRB0 MODE register	1182H (Prohibit RMW)	–	–	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			Always write “00”.		Software capture control 0: Execute 1: Undefined	Capture timing 00: Disable INT6 occurs at rising edge 01: TB0IN0 ↑ INT6 occurs at rising edge 10: TB0IN0 ↑ TB0IN0 ↓ INT6 occurs at falling edge 11: TA1OUT ↑ TA1OUT ↓ INT6 occurs at rising edge		Control Up counter 0:Clear Disable 1:Clear Enable	TMRB1 source clock 00: TB0IN0 input 01: φT1 10: φT4 11: φT16	
TB0FFCR	TMRB0 Flip-Flop control register	1183H (Prohibit RMW)	–	–	TB0CT1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			Always write “11”. *Always read as “11”.		TB1FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB1FF0 00: Invert 01: Set 10: Clear 11: Don’t care * Always read as “11”.	
					When capture UC10 to TB0CP1H/L	When capture UC10 to TB0CP0H/L	When UC10 matches with TB0RG1H/L	When UC10 matches with TB0RG0H/L		
TB0RG0L	16 bit timer register 0 low	1188H (Prohibit RMW)	–							
			W							
			0							
TB0RG0H	16 bit timer register 0 high	1189H (Prohibit RMW)	–							
			W							
			0							
TB0RG1L	16 bit timer register low	118AH (Prohibit RMW)	–							
			W							
			0							
TB0RG1H	16 bit timer register 1 high	118BH (Prohibit RMW)	–							
			W							
			0							
TB0CP0L	Capture register 0 low	118CH	–							
			R							
			Undefined							
TB0CP0H	Capture register 0 high	118DH	–							
			R							
			Undefined							
TB0CP1L	Capture register 1 low	118EH	–							
			R							
			Undefined							
TB0CP1H	Capture register 1 high	118FH	–							
			R							
			Undefined							

(15) 16-bit timer (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB1RUN	TMRB1 RUN register	1190H	TB1RDE	–			I2TB1	TB1PRUN		TB1RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffer 0: disable 1: enable	Always write “0”.			IDLE2 0: Stop 1: Operate	TMRB0 prescaler		Up counter (UC12)
								0: Stop and clear 1: Run (Count up)		
TB1MOD	TMRB1 MODE register	1192H (Prohibit RMW)	–	–	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			Always write “00”.		Software capture control 0: Execute 1: Undefined	Capture timing 00: Disable INT7 occurs at rising edge 01: TB1IN0 ↑ INT7 occurs at rising edge 10: TB1IN0 ↑ TB1IN0 ↓ INT7 occurs at falling edge 11: TA3OUT ↑ TA3OUT ↓ INT7 occurs at rising edge		Control Up counter 0:Clear Disable 1:Clear Enable	TMRB1 source clock 00: TB1IN0 input 01: φT1 10: φT4 11: φT16	
TB1FFCR	TMRB1 Flip-Flop control register	1193H (Prohibit RMW)	–	–	TB1CT1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			Always write “11”. *Always read as “11”.		TB1FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB1FF0 00: Invert 01: Set 10: Clear 11: Don’t care * Always read as “11”.	
					When capture UC12 to TB1CP1H/L	When capture UC12 to TB0CP0H/L	When UC12 matches with TB1RG1H/L	When UC12 matches with TB1RG0H/L		
TB1RG0L	16 bit timer register 0 low	1198H (Prohibit RMW)	–							
			W							
			0							
TB1RG0H	16 bit timer register 0 high	1199H (Prohibit RMW)	–							
			W							
			0							
TB1RG1L	16 bit timer register low	119AH (Prohibit RMW)	–							
			W							
			0							
TB1RG1H	16 bit timer register 1 high	119BH (Prohibit RMW)	–							
			W							
			0							
TB1CP0L	Capture register 0 low	119CH	–							
			R							
			Undefined							
TB1CP0H	Capture register 0 high	119DH	–							
			R							
			Undefined							
TB1CP1L	Capture register 1 low	119EH	–							
			R							
			Undefined							
TB1CP1H	Capture register 1 high	119FH	–							
			R							
			Undefined							

(16) UART/Serial channels

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial channel 0 buffer register	1200H (Prohibit RMW)	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R (Receive) /W (Transmission)							
			Undefined							
SC0CR	Serial channel 0 control register	1201H (Prohibit RMW)	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to 0 when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK0↑ 1: SCLK0↓	0:baud rate generator 1: SCLK0 pin input
SC0MOD0	Serial channel 0 mode 0 register	1202H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer data bit 8	0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wake up 0: Disable 1: Enable	00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		00: TA0TRG 01: Baud rate generator 10: Internal clock ϕ1 11: External clock (SCLK0 input)	
BR0CR	Serial channel 0 baud rate control register	1203H	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W							
			0	0	0	0	0	0	0	0
			Always write “0”.	(16–K) /16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency “N” setting 0~F			
BR0ADD	Serial channel 0 K setting register	1204H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
							0	0	0	0
							Sets frequency divisor “K” (1~F)			
SC0MOD1	Serial channel 0 mode 1 register	1205H	I2S0	FDPX0						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full						
SIRCR	IrDA control register	1207H	PLSEL	RXSEL	TXEN	RXEN	SIRWD3	SIRWD2	SIRWD1	SIRWD0
			R/W							
			0	0	0	0	0	0	0	0
			Select transmit pulse width 0: 3/16 1: 1/16	Receive data 0:“H” pulse 1: “L” pulse	Transmit 0: Disable 1: Enable	Receive 0: Disable 1: Enable	Select receive pulse width Set the valid SIRR×D pulse width for equal or more than 2x × (setting value + 1) + 100ns Can be set: 1~14 Can not be set: 0, 15			

(17) SBI

Symbol	Name	Address	7	6	5	4	3	2	1	0
SBICR1	Serial bus interface control register 1	1240H (Prohibit RMW)	BC2	BC1	BC0	ACK	–	SCK2	SCK1	SCK0 /SWRMON
			R/W			R/W	R	R/W		R/W
			0	0	0	0	1	0	0	0/1
			Number of transfer bits 000: 8 001: 1 010: 2 011: 3 100: 4 101: 5 110: 6 111: 7			Acknowledge mode specification 0: Disable 1: Enable	Always read as “1”.	Setting for the divisor value “n” (When writing) 000: 4 001: 5 010: 6 011: 7 100:8 101: 9 110: 10 111: (Reserved)		
SBIDBR	SBI buffer register	1241H (Prohibit RMW)	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
			R (receive)/W (Transmit)							
			Undefined							
I2CAR	I2C BUS Address register	1242H (Prohibit RMW)	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
			R/W							
			0	0	0	0	0	0	0	0
			Slave Address setting							Address recognition 0: Enable 1: Disable
SBISR When read	Serial bus interface status register	1243H (Prohibit RMW)	MST	TRX	BB	PIN	AL/SBIM1	AAS/SBIM0	AD0/ SWRST1	LRB/ SWRST0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	1	0	0	0	0
			Master/ Slave status monitor 0:Slave 1:Master	Transmitter / Receiver status monitor 0: Receiver 1: Transmitter	I^2C bus status monitor 0: Free 1: Busy	INTSBI request monitor 0: Request 1: Cancel	Arbitration lost detection monitor 0: – 1: Detected	Slave Address match detection monitor 0: Undetected 1: Detected	General call detection monitor 0: Undetected 1: Detected	Last receive bit monitor 0: “0” 1: “1”
SBICR2 When write	Serial bus interface control register 2				Start/Stop condition 0: Stop condition 1: Busy condition	Cancel INTSBI interrupt request 0:Don’t care 1:Cancel interrupt request	Serial bus interface operation mode selection 00: Port mode 01: (Reserved) 10: I^2C bus mode 11: (Reserved)		Software reset generate write “10” and “01”, then an internal reset signal is generated.	
SBIBR0	Serial bus interface baud rate register 0	1244H (Prohibit RMW)	–	I2SBI	–	–	–	–	–	–
			W	R/W	R					R/W
			0	0	1	1	1	1	1	0
			Always read “0”	IDLE2 0: Stop 1: Operate	Always read as “1”.					Always write “0”.
SBICR0	Serial bus interface control register 0	1247H (Prohibit RMW)	SBIEN	–	–	–	–	–	–	–
			R/W	R						
			0	0	0	0	0	0	0	0
			SBI operation 0:disable 1:enable	Always read as “0”.						

(18) AD converter (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADREG0L	AD conversion result register 0 low	12A0H	ADR01	ADR00					OVR0	ADR0RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN0 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG0H	AD conversion result register 0 high	12A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN0 conversion result							
ADREG1L	AD conversion result register 1 low	12A2H	ADR11	ADR10					OVR1	ADR1RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN1 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG1H	AD conversion result register 1 high	12A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN1 conversion result							
ADREG2L	AD conversion result register 2 low	12A4H	ADR21	ADR20					OVR2	ADR2RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN2 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG2H	AD conversion result register 2 high	12A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN2 conversion result							
ADREG3L	AD conversion result register 3 low	12A6H	ADR31	ADR30					OVR3	ADR3RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN3 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG3H	AD conversion result register 3 high	12A7H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN3 conversion result							
ADREG4L	AD conversion result register 4 low	12A8H	ADR4	ADR4					OVR4	ADR4F
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN4 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG4H	AD conversion result register 4high	12A9H	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN4 conversion result							
ADREG5L	AD conversion result register 5 low	12AAH	ADR5	ADR5					OVR5	ADR5F
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN5 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG5H	AD conversion result register 5 high	12ABH	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN5 conversion result							

(18) AD converter (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADREGSPL	High priority Conversion Register SP low	12B0H	ADRSP1	ADRSP0					OVSRP	ADRSPRF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of an AD conversion result						Overrun 1: Generate	AD conversion result store flag 1: Stored
ADREGSPH	High priority Conversion Register SP high	12B1H	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AD conversion result							
ADCM0REGL	AD Conversion Result Compare Criterion Register 0 Low	12B4H	ADR21	ADR20						
			R/W							
			0	0						
			Store Lower 2 bits of an AD conversion result compare criterion							
ADCM0REGH	AD Conversion Result Compare Criterion Register 0 High	12B5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R/W							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AD conversion result compare criterion							
ADCM1REGL	AD Conversion Result Compare Criterion Register 1 Low	12B6H	ADR21	ADR20						
			R/W							
			0	0						
			Store Lower 2 bits of an AD conversion result compare criterion							
ADCM1REGH	AD Conversion Result Compare Criterion Register 1 High	12B7H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R/W							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AD conversion result compare criterion							
ADCCLK	AD Conversion Clock Setting Register	12BFH					–	ADCLK2	ADCLK1	ADCLK0
							R/W	R/W	R/W	R/W
							0	0	0	0
							Always write “0”	Select clock for AD conversion 000 : Reserved 100 : $f_{IO}/4$ 001 : $f_{IO}/1$ 101 : $f_{IO}/5$ 010 : $f_{IO}/2$ 110 : $f_{IO}/6$ 011 : $f_{IO}/3$ 111 : $f_{IO}/7$	

(18) AD converter (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD0	AD mode control register 0	12B8H	EOS	BUSY		I2AD	ADS	HTRGE	TSEL1	TSEL0
			R			R/W				
			0	0		0	0	0	0	0
			Normal AD conversion end flag 0:During conversion sequence or before starting 1:Complete conversion sequence	Normal AD conversion BUSY Flag 0:Stop conversion 1:During conversion		AD conversion when IDLE2 mode 0: Stop 1: Operate	Start Normal AD conversion 0: Don't Care 1:Start AD conversion Always read as"0".	Normal AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB00 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: Reserved	
ADMOD1	AD mode control register 1	12B9H	DACON	ADCH2	ADCH1	ADCH0	LAT	ITM	REPEAT	SCAN
			R/W							
			0	0	0	0	0	0	0	0
			DAC and VREF application control	Analog input channel select			Latency 0: No Wait 1:Start after reading conversion result store Register of last channel	Interrupt specification when conversion channel fixed repeat mode	Repeat mode specification 0: Single conversion 1: Repeat conversion	Scan mode specification 0: Channel fixed mode 1: Channel scan mode
ADMOD2	AD mode control register 2	12BAH	HEOS	HBUSY			HADS	HHTRGE	HTSEL1	HTSEL0
			R				R/W			
			0	0			0	0	0	0
			High-priority AD conversion sequence FLAG 0: During conversion sequence or before starting 1: Complete conversion sequence	High-priority AD conversion BUSY Flag 0:Stop conversion 1:During conversion			Start High-priority AD conversion 0: Don't Care 1: Start AD conversion Always read as"0".	High-priority AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB10 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: I2S Sampling Counter Output	
ADMOD3	AD mode control register 3	12BBH	–	HADCH2	HADCH1	HADCH0				–
			R/W							R/W
			0	0	0	0				
			Always write "0".	High-priority analog input channel select						Always write "0".
ADMOD4	AD mode control register 4	12BCH	CMEN1	CMEN0	CMP1C	CMP0C	IRQEN1	IRQEN0	CMPINT1	CMPINT0
			R/W							
			0	0	0	0	0	0	0	0
			AD Monitor function1 0: Disable 1: Enable	AD Monitor function0 0: Disable 1: Enable	Generation condition of AD monitor function interrupt 1 0: less than 1: Greater than or Equal	Generation condition of AD monitor function interrupt 0 0: less than 1: Greater than or Equal	AD monitor function interrupt 1 0: Disable 1: Enable (Note)	AD monitor function interrupt 0 0: Disable 1: Enable (Note)	Status of AD monitor function interrupt 1 0: No generation 1: Generation	Status of AD monitor function interrupt 0 0: No generation 1: Generation
ADMOD5	AD mode control register 5	12BDH		CMCH2	CM1CH1	CM1CH0		CM0CH2	CM0CH1	CM0CH0
				R/W				R/W		
				0	0	0		0	0	0
				Select analog channel for AD monitor function 1 000: AIN0 100: AN4 001: AIN1 101: AN5 010: AIN2 110: Reserved 011: AN3 111: Reserved				Select analog channel for AD monitor function 1 000: AIN0 100: AN4 001: AIN1 101: AN5 010: AIN2 110: Reserved 011: AN3 111: Reserved		

(19) Watchdog timer

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	WDT mode register	1300H	WDTE	WDTP1	WDTP0			I2WDT	RESCR	–
			R/W					R/W		
			1	0	0			0	0	0
			WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{IO}$ 01: $2^{17}/f_{IO}$ 10: $2^{19}/f_{IO}$ 11: $2^{21}/f_{IO}$				IDLE2 0: Stop 1: Operate	1:Internally connects WDT out to the reset pin	Always write “0”.
WDCR	WDT control register	1301H (Prohibit RMW)	–							
			W							
			–							
			B1H: WDT disable code 4E: WDT clear code							

(20) RTC (Real-Time Clock)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SECR	Second register	1320H		SE6	SE5	SE4	SE3	SE2	SE1	SE0
				R/W						
				Undefined						
			"0" is read	40 sec.	20 sec.	10 sec.	8 sec.	4 sec.	2 sec.	1 sec.
MINR	Minute register	1321H		MI6	MI5	MI4	MI3	MI2	MI1	MI0
				R/W						
				Undefined						
			"0" is read	40 min.	20 min.	10 min.	8 min.	4 min.	2 min.	1 min.
HOURR	Hour register	1322H			HO5	HO4	HO3	HO2	HO1	HO0
					R/W					
					Undefined					
			"0" is read		20 hours (PM/AM)	10 hours	8 hours	4 hours	2 hours	1 hour
DAYR	Day register	1323H						WE2	WE1	WE0
								R/W		
								Undefined		
			"0" is read					W2	W1	W0
DATER	Date register	1324H			DA5	DA4	DA3	DA2	DA1	DA0
					R/W					
					Undefined					
			"0" is read		20 days	10 days	8 days	4 days	2 days	1 day
MONTHR	Month register	1325H				MO4	MO3	MO2	MO1	MO0
						R/W				
						Undefined				
		PAGE0	"0" is read			10 month	8 month	4 month	2 month	1 month
		PAGE1	"0" is read							0: Indicator for 12 hours 1: Indicator for 24 hours
YEARR	Year register	1326H	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
			R/W							
			Undefined							
		PAGE0	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 year
		PAGE1	"0" is read						Leap year setting 00: Leap year 01: One year after 10: Two years after 11: Three years after	
PAGER	Page register	1327H (Prohibit RMW)	INTENA			ADJUST	ENATMR	ENAALM		PAGE
			R/W			W	R/W			R/W
			0			Undefined	Undefined			Undefined
			Interrupt 1: Enable 0: Disable	"0" is read		1: Adjust	TIMER 1: Enable 0: Disable	ALARM 1: Enable 0: Disable	"0" is read.	PAGE selection
RESTR	Reset register	1328H (Prohibit RMW)	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	RE3	RE2	RE1	RE0
			W							
			Undefined							
			0: 1 Hz	0: 16 Hz	1:Clock reset	1: Alarm reset	Always write "0"			

(21) Melody/alarm generator

Symbol	Name	Address	7	6	5	4	3	2	1	0
ALM	Alarm-pattern register	1330H	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
			R/W							
			0	0	0	0	0	0	0	0
			Alarm pattern setting							
MELALMC	Melody/ alarm control register	1331H	FC1	FC0	ALMINV	–	–	–	–	MELALM
			R/W							
			0	0	0	0	0	0	0	0
			Free run counter control 00: Hold 01: Restart 10: Clear 11: Clear and start		Alarm frequency invert 1: Invert	Always write “0”.				Output frequency 0: Alarm 1: Melody
MELFL	Melody frequency L-register	1332H	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
			R/W							
			0	0	0	0	0	0	0	0
			Melody frequency set (Low 8bit)							
MELFH	Melody frequency H-register	1333H	MELON				ML11	ML10	ML9	ML8
			R/W				R/W			
			0				0	0	0	0
			Melody counter control 0: Stop and clear 1: Start				Melody frequency set (Upper 4 bits)			
ALMINT	Alarm interrupt enable register	1334H			–	IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
					R/W					
					0	0	0	0	0	0
					Always write “0”.	1:INTALM4 (1Hz) enable	1:INTALM3 (2Hz) enable	1:INTALM2 (64Hz) enable	1:INTALM1 (512Hz) enable	1:INTALM0 (8192Hz) enable

(22) I²S (1/2)

Symbol	Name	Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2S0BUF	I2S Transmission Buffer Register0	1800H (Prohibit RMW)	B015	B014	B013	B012	B011	B010	B009	B008	B007	B006	B005	B004	B003	B002	B001	B000
			W															
			Undefined															
			Transmission buffer register (FIFO)															
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			B031	B030	B09	B028	B027	B026	B025	B024	B023	B022	B021	B020	B019	B018	B017	B016
			W															
			Undefined															
			Transmission buffer register (FIFO)															
I2S1BUF	I2S Transmission Buffer Register1	1810H (Prohibit RMW)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			B115	B114	B113	B112	B111	B110	B109	B108	B107	B106	B105	B104	B103	B102	B101	B100
			W															
			Undefined															
			Transmission buffer register (FIFO)															
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			B131	B130	B129	B128	B127	B126	B125	B124	B123	B122	B121	B120	B119	B118	B117	B116
			W															
			Undefined															
			Transmission buffer register (FIFO)															

(22) I²S (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
I2S0CTL	I2S Control Register0	1808H	TXE0	*CNTE0		DIR0	BIT0	DTFMT01	DTFMT00	SYSCKE0
			R/W	R/W		R/W	R/W	R/W	R/W	R/W
			0	0		0	0	0	0	0
			Transmit 0: Stop 1: Start	Counter control 0: Clear 1: Start		Transmission start BIT 0:MSB 1:LSB	Bit length 0: 8 bits 1:16 bits	Output format 00: I2S 10: Right 01: Left 11:Reserved		System clock 0:Disable 1:Enable
		1809H	CLKS0			FSEL0	TEMP0	WLVL0	EDGE0	CLKE0
			R/W			R/W	R	R/W	R/W	R/W
			0			0	1	0	0	0
			Source clock 0: f_{SYS} 1: f_{PLL}			Stereo /monaural 0: Stereo 1: Monaural	Condition of transmission FIFO 0: data 1: None data	WS level 0:low left 1:high left	Clock edge for data output 0:Rising 1:Falling	Clock enable (After transmission) 0:Operate 1:Stop
I2S0C	I2S0 Divider Value Setting Register	180AH	CK07	CK06	CK05	CK04	CK03	CK02	CK01	CK00
			R/W							
			0	0	0	0	0	0	0	0
			Divider value for CK signal (8-bit counter)							
		180BH			WS05	WS04	WS03	WS02	WS01	WS00
			R/W							
			0	0	0	0	0	0	0	0
			Divider value for WS signal (6-bit counter)							
I2S1CTL	I2S Control Register1	1818H	TXE1	*CNTE1		DIR1	BIT1	DTFMT11	DTFMT10	SYSCKE1
			R/W	R/W		R/W	R/W	R/W	R/W	R/W
			0	0		0	0	0	0	0
			Transmit 0: Stop 1: Start	Counter control 0: Clear 1: Start		Transmission start BIT 0:MSB 1:LSB	Bit length 0: 8 bits 1:16 bits	Output format 00: I2S 10: Right 01: Left 11:Reserved		System clock 0:Disable 1:Enable
		1819H	CLKS1			FSEL1	TEMP1	WLVL1	EDGE1	CLKE1
			R/W			R/W	R	R/W	R/W	R/W
			0			0	1	0	0	0
			Source clock 0: f_{SYS} 1: f_{PLL}			Stereo /monaural 0: Stereo 1: Monaural	Condition of transmission FIFO 0: data 1: None data	WS level 0:low left 1:high left	Clock edge for data output 0:Rising 1:Falling	Clock enable (After transmission) 0:Operate 1:Stop
I2S1C	I2S1 Divider Value Setting Register	181AH	CK17	CK16	CK15	CK14	CK13	CK12	CK11	CK10
			R/W							
			0	0	0	0	0	0	0	0
			Set divide frequency for CK signal (8-bit counter)							
		181BH			WS15	WS14	WS13	WS12	WS11	WS10
			R/W							
			0	0	0	0	0	0	0	0
			Set divided frequency for WS signal (6-bit counter)							

(23) MAC (1/2)

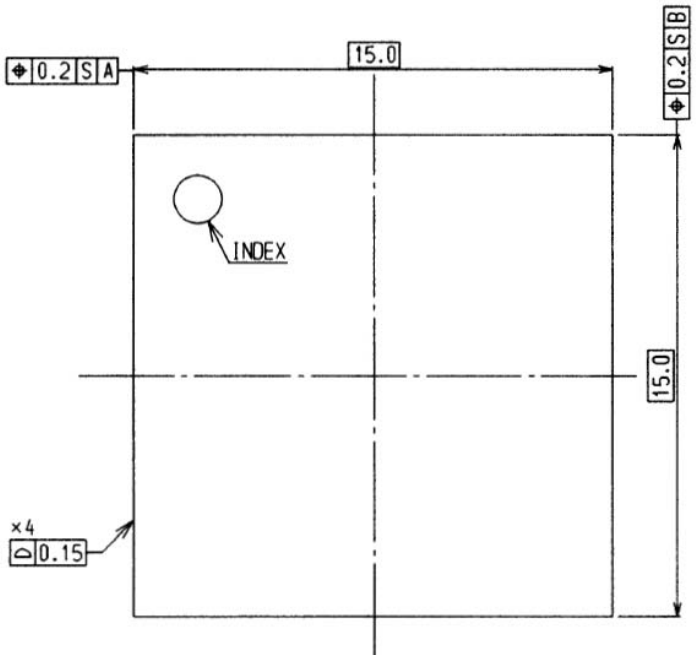
Symbol	Name	Address	7	6	5	4	3	2	1	0
MACMA_LL	Data register Multiplier A-LL	1BE0H	MA7	MA6	MA5	MA4	MA3	MA2	MA1	MA0
			R/W							
			Undefined							
			Multiplier A data register [7:0]							
MACMA_LH	Data register Multiplier A-LH	1BE1H	MA15	MA14	MA13	MA12	MA11	MA10	MA9	MA8
			R/W							
			Undefined							
			Multiplier A data register [15:8]							
MACMA_HL	Data register Multiplier A-HL	1BE2H	MA23	MA22	MA21	MA20	MA19	MA18	MA17	MA16
			R/W							
			Undefined							
			Multiplier A data register [23:16]							
MACMA_HH	Data register Multiplier A-HH	1BE3H	MA31	MA30	MA29	MA28	MA27	MA26	MA25	MA24
			R/W							
			Undefined							
			Multiplier A data register [31:24]							
MACMB_LL	Data register Multiplier B-LL	1BE4H	MB7	MB6	MB5	MB4	MB3	MB2	MB1	MB0
			R/W							
			Undefined							
			Multiplier B data register [7:0]							
MACMB_LH	Data register Multiplier B-LH	1BE5H	MB15	MB14	MB13	MB12	MB11	MB10	MB9	MB8
			R/W							
			Undefined							
			Multiplier B data register [15:8]							
MACMB_HL	Data register Multiplier B-HL	1BE6H	MB23	MB22	MB21	MB20	MB19	MB18	MB17	MB16
			R/W							
			Undefined							
			Multiplier B data register [23:16]							
MACMB_HH	Data register Multiplier B-HH	1BE7H	MB31	MB30	MB29	MB28	MB27	MB26	MB25	MB24
			R/W							
			Undefined							
			Multiplier B data register [31:24]							
MACOR_LLL	Data register Multiply and Accumulate -LLL	1BE8H	OR7	OR6	OR5	OR4	OR3	OR2	OR1	OR0
			R/W							
			Undefined							
			Multiply and Accumulate data register [7:0]							
MACOR_LLH	Data register Multiply and Accumulate -LLH	1BE9H	OR15	OR14	OR13	OR12	OR11	OR10	OR9	OR8
			R/W							
			Undefined							
			Multiply and Accumulate data register [15:8]							
MACOR_LHL	Data register Multiply and Accumulate -LGL	1BEAH	OR23	OR22	OR21	OR20	OR19	OR18	OR17	OR16
			R/W							
			Undefined							
			Multiply and Accumulate data register [23:16]							
MACOR_LHH	Data register Multiply and Accumulate -LHH	1BEBH	OR31	OR30	OR29	OR28	OR27	OR26	OR25	OR24
			R/W							
			Undefined							
			Multiply and Accumulate data register [31:24]							

(23) MAC (2/2)

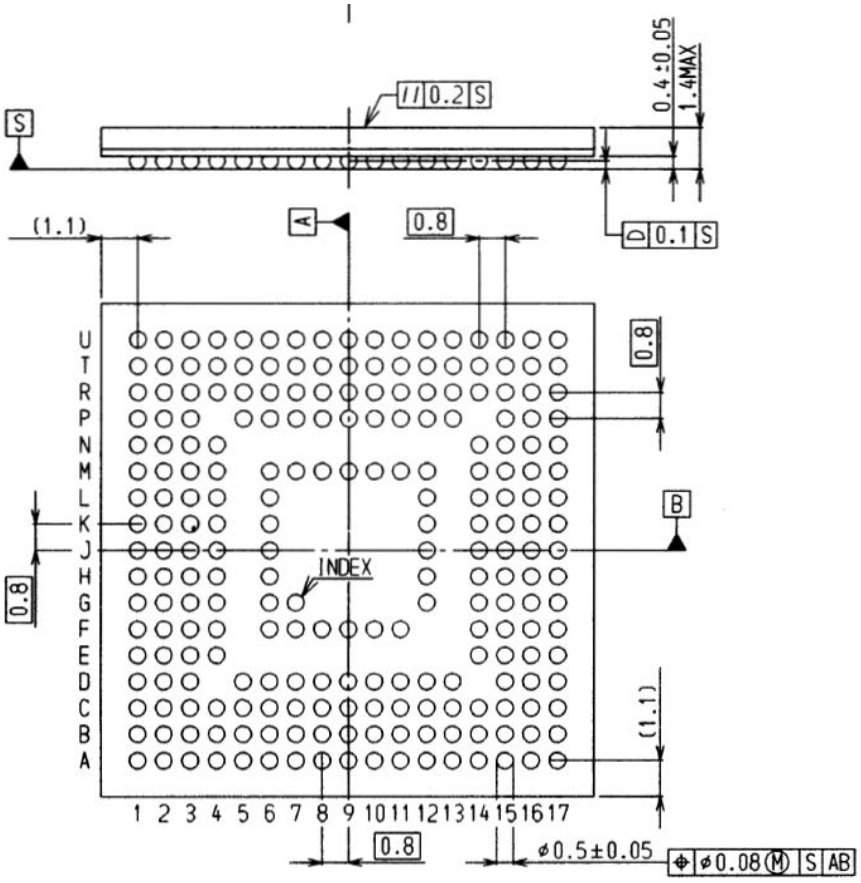
Symbol	Name	Address	7	6	5	4	3	2	1	0
MACOR_HLL	Data register Multiply and Accumulate -HLL	1BECH	OR39	OR38	OR37	OR36	OR35	OR34	OR33	OR32
			R/W							
			Undefined							
			Multiply and Accumulate data register [39:32]							
MACOR_HLH	Data register Multiply and Accumulate -HLH	1BEDH	OR47	OR46	OR45	OR44	OR43	OR42	OR41	OR40
			R/W							
			Undefined							
			Multiply and Accumulate data register [47:40]							
MACOR_HHL	Data register Multiply and Accumulate -HHL	1BEEH	OR55	OR54	OR53	OR52	OR51	OR50	OR49	OR48
			R/W							
			Undefined							
			Multiply and Accumulate data register [55:48]							
MACOR_HHH	Data register Multiply and Accumulate -HHH	1BEFH	OR63	OR62	OR61	OR60	OR59	OR58	OR57	OR56
			R/W							
			Undefined							
			Multiply and Accumulate data register [63:56]							
MACCR	MAC Control Register	1BFCH	MOVF	MOPST	MSTTG2	MSTTG1	MSTTG0	MSGMD	MOPMD1	MOPMD0
			R/W	W	R/W			R/W	R/W	
			0	0	0	0	0	0	0	0
			Over flow flag 0:no over flow 1:generate over flow	Start calculation control 0:don't care 1: Start calculation	Select the trigger of start calculation 000: Write to MACMA[7:0] 001: Write to MACMB[7:0] 010: Write to MACMOR[7:0] 011: Write to MACMOR[39:32] 1xx: Write "1" to <MOPST>			Sign mode 0:Unsigned 1:Signed	Calculation Mode 00: 64 + 32 × 32 01: 64 − 32 × 32 10: 32 × 32 − 64 11: Reserved	

6. Package

P-FBGA228-1515-0.80A5



TOP VIEW



BOTTOM VIEW