TOSHIBA

Data Book

32bit Micro controller

TLCS-900/H1 series

TMP92CH21FG

TENTATIVE Rev0.92 11th/Jul./2002

Since this Revision 0.92 is still under working, there may be some mistakes in it.
When you will start to design, please order the latest one.

--------- Contents--------

1. Outline and Device Characteristics ····· 92CH21-1
2. Pin Assignment and Functions ····· 92CH21-5
2.1 Pin Assignment ····· 92CH21-5
2.2 Pin name and functions ····· 92CH21-6
3. Oepration ····· 92CH21-12
3.1 CPU ····· 92CH21-12
3.2 Memory Map ····· 92CH21-14
3.3 Clock Function and Standby Function ····· 92CH21-15
3.4 Interrupts ····· 92CH21-36
3.5 Function of Ports ····· 92CH21-56
3.5.1 Port 1 ····· 92CH21-
3.5.2 Port 2 ····· 92CH21-
3.5.3 Port 3 ····· 92CH21-
3.5.4 Port 4 ····· 92CH21-
3.5.5 Port 5 ····· 92CH21-
3.5.6 Port 6 ····· 92CH21-
3.5.7 Port 7 ····· 92CH21-
3.5.8 Port 8 ····· 92CH21-69
3.5.9 Port 9 ····· 92CH21-
3.5.10 Port A ····· 92CH21-
3.5.11 Port C ····· 92CH21-
3.5.12 Port F ····· 92CH21-
3.5.13 Port G ····· 92CH21-
3.5.14 Port J ····· 92CH21-
3.5.15 Port K ····· 92CH21-
3.5.16 Port L ····· 92CH21-
3.5.17 Port M ····· 92CH21-
3.6 Memory Controller ····· 92CH21-93
3.7 8-bit Timers (TMRA) ····· 92CH21-111
3.8 MMU ····· 92CH21-132
3.9 Serial Channel ····· 92CH21-143
3.10 USB Controller ····· 92CH21-177
3.11 Analog/Digital Converter ····· 92CH21-281
3.12 Watch Dog Timer ····· 92CH21-291
3.13 RTC ····· 92CH21-296
3.14 LCD Controller ····· 92CH21-310
3.15 Melody / Alarm Generator ····· 92CH21-351
3.16 SDRAM Controller ····· 92CH21-357
3.17 NAND-Flash Controller ····· 92CH21-367

3.18 16-bit Timers (TMRB) ····· 92CH21-385
3.19 Touch Screen interface ····· 92CH21-395
3.20 I^2S ····· 92CH21-401
3.21 Boot ROM ····· 92CH21-408
3.22 Power Supply Backup ····· 92CH21-426
4. Electrical Charactoristics ····· 92CH21-428
5. Table of Speial function registers (SFRs) ····· 92CH21-449
6. Package ····· 92CH21-492

DataBook Modification History

Rev/Data	Page	Modification item	Reason
0.91 /20-Apr-2002		New release	
0.92 /10-Jul-2002 (Shown by Blue-color)	1,3,6	Product name : JTMP92CH21 explanation	New added
	4,7,431	Modified Pin assignment for S0ALEL/H,S1ALEL/H	Mistake
	5	Deleted Port0 in Pin assignment	
	9	Modified attribute(I/O) for MX,MY	
	13	Modified Figure 3.1	
	71	Separated Figure 3.5.11	
	80	Deleted PC7F,PC6F explanation	
	98	Added (Note) for Bus sizing	New added
	105	Added connection example for NOR-Flash	
	110	Added Note the NAND-Flash area setting	
	114	Modified Table 3.7.2	Mistake
	120,121	Modified Figure 3.7.8, 3.7.9	
	136	Added explanation for MMU	New added
	141	Added example how to use MMU	
	145,146	Modified Figure 3.9.2	Mistake
	147	Modified Table 3.9.2	
	149	Modified baud rate calculation formula	
	150	Added (Note)	
	151	Modified (3)Seirial clock generation circuit	
	157,158	Modified Figure 3.9.7, 3.9.8	
	167,169,171	Modified setting example	
	175	Added (6) Using IrDA 115.2Kbps with USB	New added
	311	Modified Table 3.14.1	Mistake
	313	Added (Note)	New added
	315,316	Added (Note)	Mistake
	323	Modified bit number for LD-bus	
	325	Added page, explanation of LD-bus for 4096 STN color	New added
	336	Modified diagram example	
	337	Added program example	
	358	Added (Note3) for SDACR1	Mistake
	358	Modified SDACR2<SDRS1 to 0> explanation	
	366	Modified Figure 3.16.2(2)	
	374	Modified comment in Note1	
	387	Modified Table 3.8.2	
	404,406	Added comment in Setting example	
	408-425	Modified 3.21 Boot-ROM	New added
	435,436	Modified pins for shmitt-level	Mistake
	475	Modified explanation in BnCSL-registers	

CMOS 32-bit Micro-controller

TMP92CH21FG / JTMP92CH21

1. Outline and Device Characteristics

TMP92CH21 is high-speed advanced 32-bit micro-controller developed for controlling equipment which processes mass data.

TMP92CH21 is a micro-controller which has a high-performance CPU (900/H1 CPU) and various built-in I/Os.

TMP92CH21FG is housed in a 144-pin flat package. JTMP92CH21 is a chip form product.

Device characteristics are as follows:

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

- Compatible with TLCS-900/L1 instruction code
- 16Mbytes of linear address space
- General-purpose register and register banks
- Micro DMA : 8channels (250ns/4bytes at fc = 40MHz, best case)

(2) Minimum instruction execution time : 50ns(at fc=40MHz)

(3) Internal memory

- Internal RAM : 16K-byte (can be used for program and data memory)
- Internal ROM : 8K-byte (used as boot-prgram)

  Possible downloading user's download-program through either USB, UART or NAND-Flash.

(4) External memory expansion

- Expandable up to 512M bytes (shared program/data area)
- Can simultaneously support 8/16/32-bit width external data bus
  ⋯ Dynamic data bus sizing
- Separate bus system

(5) Memory controller

- Chip select output : 4 channel

(6) 8-bit timers: 4 channels

(7) 16-bit timer/event counter : 1 channel

980508TBA1

- For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance / Handling Precautions.
- TOSHIBA is continually working to improve the quality and the reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing TOSHIBA products, to observe standards of safety, and to avoid situations in which a malfunction or failure of a TOSHIBA product could cause loss of human life, bodily injury or damage to property. In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent products specifications. Also, please keep in mind the precautions and conditions set forth in the TOSHIBA Semiconductor Reliability Handbook.
- The products described in this document are subject to foreign exchange and foreign trade laws.
- The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA CORPORATION for any infringements of intellectual property or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any intellectual property or other rights of TOSHIBA CORPORATION or others.
- The information contained herein is subject to change without notice.

(8) General-purpose serial interface: 2 channels

- UART/Synchronous mode: 2 channels (ch.0 and 1)
- IrDA Ver.1.0(115kbps) mode selectable: 1channel (ch.0)

(9) USB(Universal Serial Bus) Controller : 1 channel

- Compliant with USB rev1.1
- Full-speed (12MHz) (LOW-spped is not supported.)
- Endpoints spec

  Endpoint-0 : Control 64Bytes*1 -FIFO
  Endpoint-1 : BULK(out) 64Bytes*2 -FIFO
  Endpoint-2 : BULK(in) 64Bytes*2 -FIFO
  Endpoint-3 : Interrupt(in) 8Bytes*1 -FIFO

- Descripter-RAM : 384Bytes

(10) I^2S(Inter-IC Sound) interface: 1 channel

- I^2Sbus-mode/SIO-mode selectable (Master, transmittion only)
- 32-byte FIFO buffer

(11) LCD controller

- Supported up to 4096-color for TFT, 256-color,16,8,4Gray-levels and B/W for STN
- Shift register/built-in RAM LCD driver

(12) SDRAM Controller: 1 channel

- Supported 16M,64M,128M,256M even 512Mbit SDR-SDRAM
- Possible to execute instruction on SDRAM

(13) Timer for real-time clock (RTC)

(14) Key-on wake up (Interrupt key input)

(15) 10-bit A/D converter: 4 channels

(16) Touch screen interface

- Available to reduce external components

(17) Watch dog timer

(18) Melody/alarm generator

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

(19) MMU

- Expandable up to 512M bytes (3 local area/bank method)
- Independent bank for each Program,Read-data,Write-data and LCD-display-data

(21) Interrupts: 51 interrupts

- 9 CPU interrupts: Software interrupt instruction and illegal instruction
- 34 internal interrupts: Seven selectable priority levels
- 7 external interrupts: Seven selectable priority levels(6-edge selectable)

(22)Input/output ports: 126 pins

(23) NAND Flash interface: 2 channel

- Available to connect directly with NAND-Flash
- ECC calculation

(24)Stand-by function

- Three Halt modes: Idle2 (programmable), Idle1, Stop
- Isolated Power supply : RTCVCC controlled by /BE-pin
- Each pin status programmable for stand-by mode

(25) Triple-clock controller

- Clock doubler (PLL) supplies 48MHz for USB, 36MHz system-clock for others
- Clock gear function: Select a High-frequency clock fc to fc/16
- RTC (fs=32.768kHz)

(26)Operating voltage:

- VCC = 3.0 V to 3.6 V (fc max = 40MHz)
- VCC = 2.7 V to 3.6 V (fc max = 27MHz)

(27) Package:

- 144-pin QFP (P-LQFP144 -1616 - 0.40)
- 144-pin Chip form is also available. For details, contact your local Toshiba sales representative.

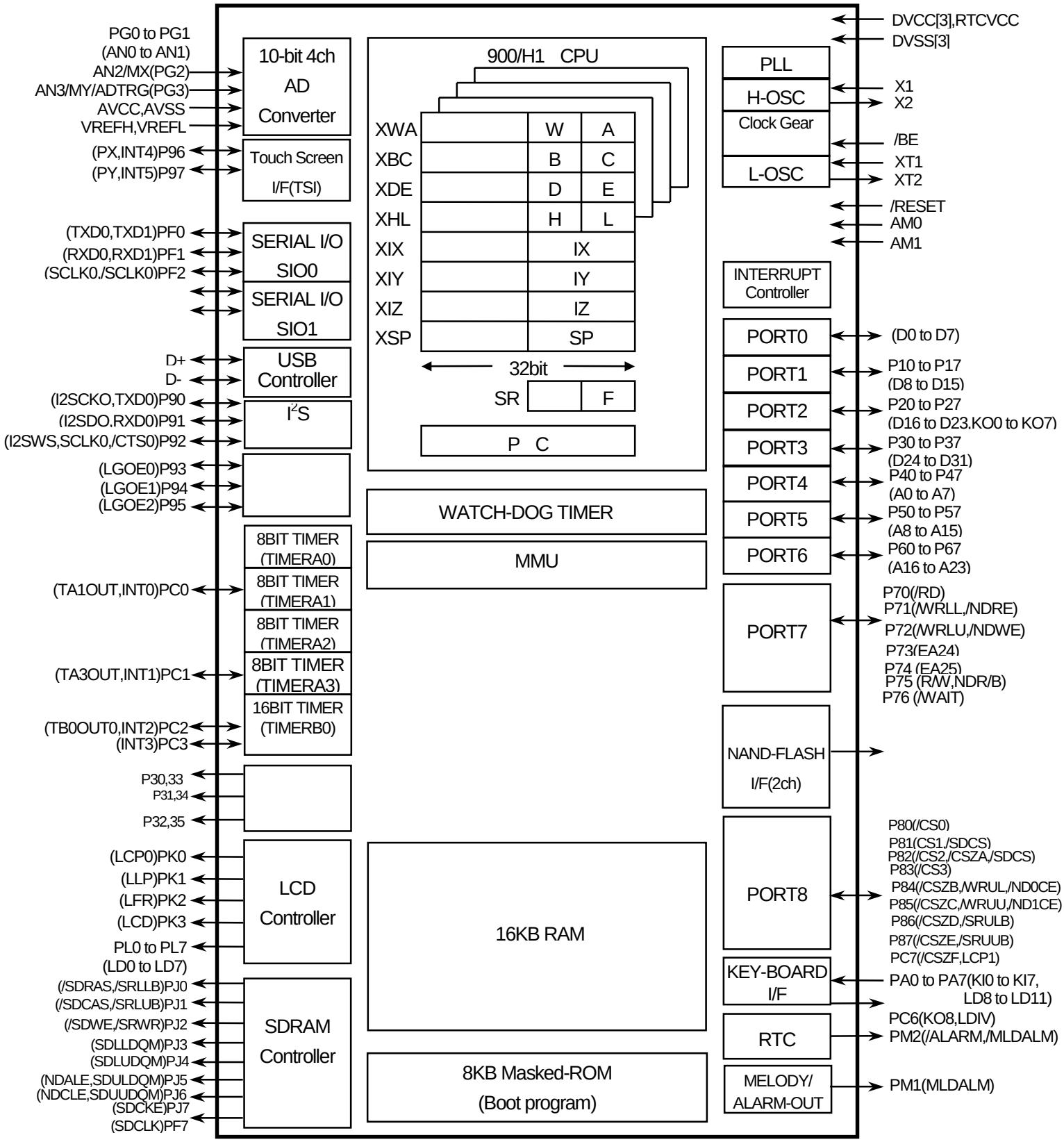


Figure 1.1 TMP92CH21 block diagram

2. Pin Assignment and Functions

The assignment of input/output pins for the TMP92CH21FG, their names and functions are as follows:

2.1 Pin Assignment

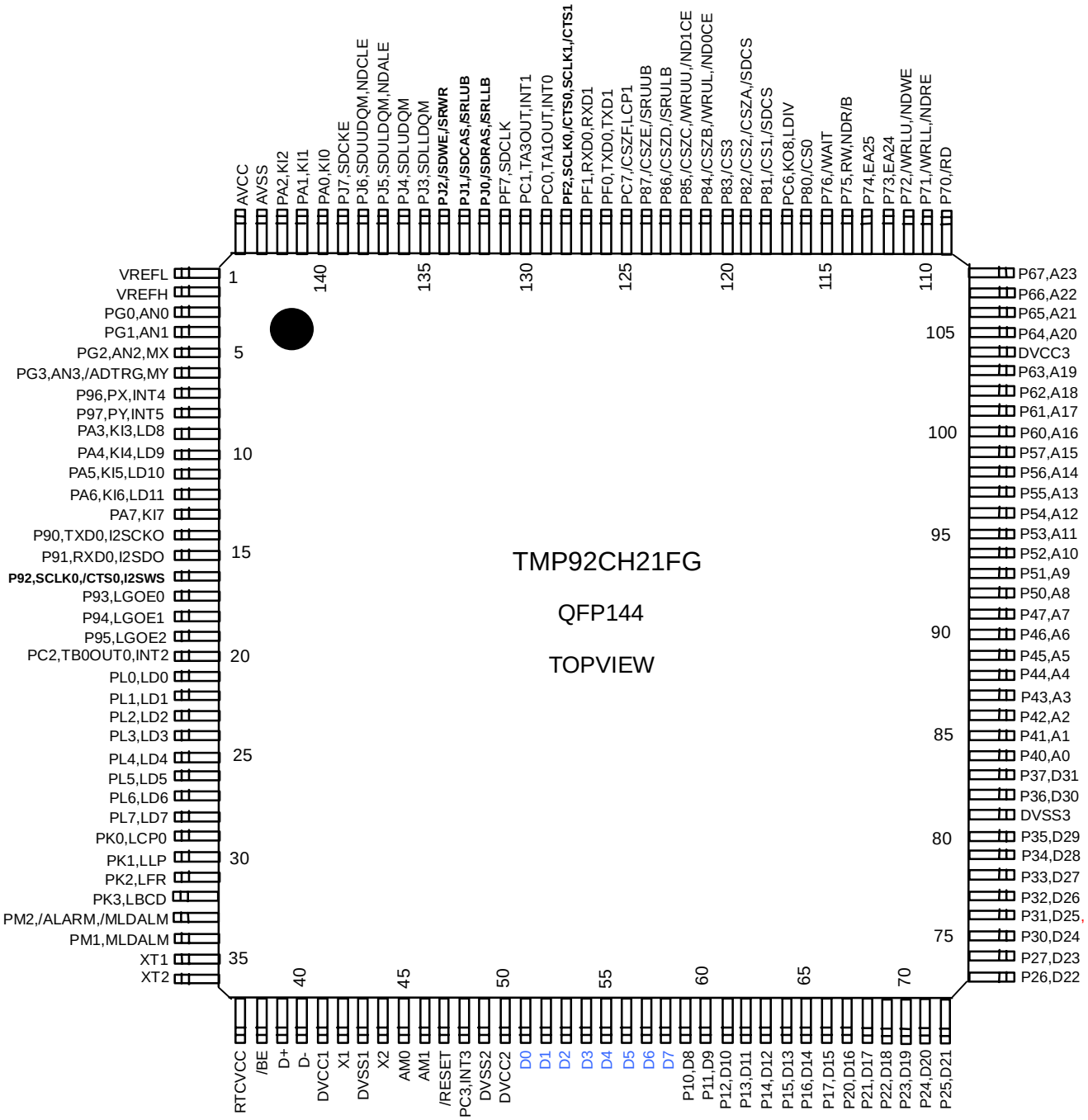


Figure2.1.1 Pin assignment diagram (144-pin QFP)

2.2 PAD Assignment

(Chip size 5.98mm × 6.42mm)

Table2.2.1 Pad assignment diagram (144-pin chip) Unit: μm

PIN No	Name	X point	Y point	PIN No	Name	X point	Y point	PIN No	Name	X point	Y point
1	VREFL	-2852	2671	49	DVSS2	-488	-3072	97	P55	2848	815
2	VREFH	-2852	2546	50	DVCC2	-338	-3072	98	P56	2848	941
3	PG0	-2852	2421	51	D0	-200	-3072	99	P57	2848	1066
4	PG1	-2852	2296	52	D1	-75	-3072	100	P60	2848	1191
5	PG2	-2852	2171	53	D2	49	-3072	101	P61	2848	1316
6	PG3	-2852	2045	54	D3	174	-3072	102	P62	2848	1441
7	P96	-2852	1920	55	D4	300	-3072	103	P63	2848	1566
8	P97	-2852	1795	56	D5	425	-3072	104	DVCC3	2848	1692
9	PA3	-2852	1270	57	D6	550	-3072	105	P64	2848	1823
10	PA4	-2852	1145	58	D7	675	-3072	106	P65	2848	1974
11	PA5	-2852	1020	59	P10	800	-3072	107	P66	2848	2130
12	PA6	-2852	895	60	P11	925	-3072	108	P67	2848	2292
13	PA7	-2852	769	61	P12	1050	-3072	109	P70	2460	3065
14	P90	-2852	644	62	P13	1176	-3072	110	P71	2295	3065
15	P91	-2852	519	63	P14	1301	-3072	111	P72	2127	3065
16	P92	-2852	394	64	P15	1426	-3072	112	P73	1964	3065
17	P93	-2852	269	65	P16	1551	-3072	113	P74	1807	3065
18	P94	-2852	144	66	P17	1676	-3072	114	P75	1654	3065
19	P95	-2852	18	67	P20	1801	-3072	115	P76	1506	3065
20	PC2	-2852	-106	68	P21	1927	-3072	116	P80	1361	3065
21	PL0	-2852	-231	69	P22	2052	-3072	117	PC6	1226	3065
22	PL1	-2852	-356	70	P23	2177	-3072	118	P81	1101	3065
23	PL2	-2852	-481	71	P24	2303	-3072	119	P82	976	3065
24	PL3	-2852	-606	72	P25	2460	-3072	120	P83	851	3065
25	PL4	-2852	-732	73	P26	2848	-2279	121	P84	726	3065
26	PL5	-2852	-857	74	P27	2848	-2138	122	P85	600	3065
27	PL6	-2852	-982	75	P30	2848	-1982	123	P86	475	3065
28	PL7	-2852	-1107	76	P31	2848	-1831	124	P87	350	3065
29	PK0	-2852	-1232	77	P32	2848	-1687	125	PC7	225	3065
30	PK1	-2852	-1357	78	P33	2848	-1562	126	PF0	100	3065
31	PK2	-2852	-1482	79	P34	2848	-1437	127	PF1	-24	3065
32	PK3	-2852	-1608	80	P35	2848	-1311	128	PF2	-150	3065
33	PM2	-2852	-1892	81	DVSS3	2848	-1186	129	PC0	-275	3065
34	PM1	-2852	-2017	82	P36	2848	-1061	130	PC1	-400	3065
35	XT1	-2852	-2142	83	P37	2848	-936	131	PF7	-525	3065
36	XT2	-2852	-2444	84	P40	2848	-811	132	PJ0	-650	3065
37	RTCVCC	-2465	-3072	85	P41	2848	-686	133	PJ1	-775	3065
38	/BE	-2339	-3072	86	P42	2848	-560	134	PJ2	-901	3065
39	D+	-2062	-3072	87	P43	2848	-435	135	PJ3	-1026	3065
40	D-	-1875	-3072	88	P44	2848	-310	136	PJ4	-1151	3065
41	DVCC1	-1598	-3072	89	P45	2848	-185	137	PJ5	-1276	3065
42	X1	-1472	-3072	90	P46	2848	-60	138	PJ6	-1401	3065
43	DVSS1	-1347	-3072	91	P47	2848	65	139	PJ7	-1526	3065
44	X2	-1126	-3072	92	P50	2848	190	140	PA0	-1652	3065
45	AM0	-1001	-3072	93	P51	2848	315	141	PA1	-1777	3065
46	AM1	-876	-3072	94	P52	2848	440	142	PA2	-1902	3065
47	/RESET	-750	-3072	95	P53	2848	565	143	AVSS	-2275	3065
48	PC3	-625	-3072	96	P54	2848	690	144	AVCC	-2400	3065

2.3 Pin names and functions

The following table shows the names and functions of the input/output pins

Table2.3.1 Pin names and functions (1/5)

Pin Name	Number of pins	I/O	Function
D0 to D7	8	I/O	Data: Data bus 0 to 7
P10 to P17 D8 to D15	8	I/O I/O	Port 1: I/O port Input or output specifiable in units of bits Data: Data bus 8 to 15
P20 to P27 D16 to D23 KO0 to KO7	8	I/O I/O Output	Port 2: I/O port Input or output specifiable in units of bits Data: Data bus 16 to 23 Key Output 0 to 7 : pins used of Key-scan strobe (Open drain-output programmable)
P30 D24	1	I/O I/O	Port 30: I/O port Data24: Data bus 24
P31 D25	1	I/O I/O	Port 31: I/O port Data25: Data bus 25
P32 D26	1	I/O I/O	Port 32: I/O port Data26: Data bus 26
P33 D27	1	I/O I/O	Port 33: I/O port Data27: Data bus 27
P34 D28	1	I/O I/O	Port 34: I/O port Data28: Data bus 28
P35 D29	1	I/O I/O	Port 35: I/O port Data29: Data bus 29
P36 to P37 D30 to D31	2	I/O I/O	Port 3: I/O port Input or output specifiable in units of bits Data: Data bus 30o 31
P40 to P47 A0 to A7	8	I/O Output	Port 4: I/O port Input or output specifiable in units of bits Address: Address bus 0 to 7
P50 to P57 A8 to A15	8	I/O Output	Port 5: I/O port Input or output specifiable in units of bits Address: Address bus 8 to 15
P60 to P67 A16 to A23	8	I/O Output	Port 6: I/O port Input or output specifiable in units of bits Address: Address bus 16 to 23
P70 /RD	1	Output Output	Port70: Output port Read: Outputs strobe signal to read external memory
P71 /WRLL /NDRE	1	I/O Output Output	Port 71: I/O port Write: Output strobe signal for writing data on pins D0 to D7 NAND-Flash read: Outputs strobe signal to read external NAND-Flash
P72 /WRLU /NDWE	1	I/O Output Output	Port 72: I/O port Write: Output strobe signal for writing data on pins D8 to D15 Write Enable for NAND-Flash
P73 EA24	1	I/O Output	Port 73: Output port Extended Address 24
P74 EA25	1	I/O Output	Port 74: Output port Extended Address 25
P75 R/W NDR/B	1	Output Output Input	Port 75: Output port Read/Write: 1 represents Read or Dummy cycle; 0 represents write cycle NAND-Flash Ready(1)/Busy(0) input
P76 /WAIT	1	I/O Input	Port 76: I/O port Wait: Signal used to request CPU bus wait

Table2.3.1 Pin names and functions (2/5)

Pin Name	Number of pins	I/O	Function
P80 /CS0	1	Output Output	Port80:Output port Chip select 0: Outputs “Low” when address is within specified address area
P81 /CS1 /SDCS	1	Output Output Output	Port81:Output port Chip select 1: Outputs “Low” when address is within specified address area Chip Select for SDRAM: Outputs “0” when address is within SDRAM -address area
P82 /CS2 /CSZA /SDCS	1	Output Output Output Output	Port82:Output port Chip select 2: Outputs “Low” when address is within specified address area Expand Chip Select: ZA: outputs “0” when address is within specified address area Chip Select for SDRAM: Outputs “0” when address is within SDRAM -address area
P83 /CS3	1	Output Output	Port83:Output port Chip select 3: Outputs “Low” when address is within specified address area
P84 /WRUL /CSZB /ND0CE	1	Output Output Output Output	Port84:Output port Write: Output strobe signal for writing data on pins D16 to D23 Expand Chip Select: ZB: outputs “0” when address is within specified address area Chip select for NAND-Flash0: Outputs “0” when NAND-Flash-0 is accessed
P85 /WRUU /CSZC /ND1CE	1	Output Output Output Output	Port85: Output port Write: Output strobe signal for writing data on pins D24 to D31 Expand Chip Select: ZC: outputs “0” when address is within specified address area Chip select for NAND-Flash1: Outputs “0” when NAND-Flash-1 is accessed
P86 /CSZD /SRULB	1	Output Output Output	Port86: Output port Expand Chip Select: ZD: outputs “0” when address is within specified address area Data enable for SRAM on pins D16 to D23
P87 /CSZE /SRUUB	1	Output Output Output	Port87: Output port Expand Chip Select: ZE: outputs “0” when address is within specified address area Data enable for SRAM on pins D24 to D31
P90 TXD0 I2SCKO	1	I/O Output Output	Port90: I/O port Serial 0 send data: Open drain-output programmable I2S Clock Output
P91 RXD0 I2SDO	1	I/O Input Output	Port91: I/O port Serial 0 receive data I2S Data Output
P92 SCLK0 /CTS0 I2SWS	1	I/O I/O Input Output	Port92: I/O port Serial 0 clock I/O Serial 0 data send enable (Clear to Send) I2S Word Select Output
P93 LGOE0	1	I/O Output	Port93: I/O port Output Enable-0 for external TFT-LCD Driver
P94 LGOE1	1	I/O Output	Port94: I/O port Output Enable-1 for external TFT-LCD Driver
P95 LGOE2	1	I/O Output	Port95: Output port Output Enable-2 for external TFT-LCD Driver
P96 INT4 PX	1	I/O Input Output	Port 96: Input port Interrupt request pin4:Interrupt request with programmable rising/falling edge X-Plus : Pin connectted to X+ for Touch Screen Panel
P97 INT5 PY	1	I/O Input Output	Port 97: Input port Interrupt request pin5:Interrupt request with programmable rising/falling edge Y-Plus : Pin connectted to Y+ for Touch Screen Panel
PA0 to PA2 KI0 to KI2	3	Input Input	Port: A0 to A2 port: Pin used to input ports Key input 0 to 2: Pin used of Key on wake-up 0 to 2 (shummit input, with pull-up register)
PA3 to PA6 KI3 to KI6 LD8 to LD11	4	Input Input Output	Port: A3 to A6 port: Pin used to input ports Key input 3 to 6: Pin used of Key on wake-up 3 to 6 (shummit input, with pull-up register) Data Bus for LCD-driver
PA7 KI7	1	Input Input	Port: A7 port: Pin used to input ports Key input 7: Pin used of Key on wake-up 7 (shummit input, with pull-up register)

Table2.3.1 Pin names and functions (3/5)

Pin Name	Number of pins	I/O	Function
PC0 INT0 TA1OUT	1	I/O Input Output	PortC0: I/O port Interrupt request pin0 : Interrupt request pin with programmable level / rising /falling edge 8bit timer 1 output: Timer 1 output
PC1 INT1 TA3OUT	1	I/O Input Output	PortC1: I/O port Interrupt request pin1 : Interrupt request pin with programmable rising /falling edge 8bit timer 3 output: Timer 3 output
PC2 INT2 TB0OUT0	1	I/O Input Output	PortC2 : I/O port Interrupt request pin2: Interrupt request pin with programmable rising /falling edge Timer B0 output
PC3 INT3	1	I/O Input	PortC3: I/O port Interrupt request pin3 : Interrupt request pin with programmable rising /falling edge
PC6 KO8 LDIV	1	I/O Output Output	PortC6 : I/O port Key Output 8 : pin used of Key-scan strobe (Open drain-output programmable) Data Invert enable for external TFT-LCD Driver
PC7 /CSZF LCP1	1	I/O Output Output	PortC7 : I/O port Expand Chip Select: ZF: outputs “0” when address is within specified address area Shift-clock-1 for external TFT-LCD Driver
PF0 TXD0 TXD1	1	I/O Output Output	PortF0: I/O port Serial 0 send data: Open drain-output programmable Serial 1 send data: Open drain-output programmable
PF1 RXD0 RXD1	1	I/O Input Input	PortF1: I/O port Serial 0 receive data Serial 1 receive data
PF2 SCLK0 /CTS0 SCLK1 /CTS1	1	I/O I/O Input I/O Input	PortF2: I/O port Serial 0 clock I/O Serial 0 data send enable (Clear to Send) Serial 1 clock I/O Serial 1 data send enable (Clear to Send)
PF7 SDCLK	1	Output Output	PortF7: Output port Clock for SDRAM
PG0 to PG1 AN0 to AN1	2	Input Input	Port G0 to G1 port: Pin used to input ports Analog input 0 to 1: Pin used to Input to A/D conveter
PG2 AN2 MX	1	Input Input Output	Port G2 port: Pin used to input ports Analog input 2: Pin used to Input to A/D conveter X-Minus: Pin connectted to X- for Touch Screen Panel
PG3 AN3 MY /ADTRG	1	Input Input Output Intput	Port G3 port: Pin used to input ports Analog input 3: Pin used to Input to A/D conveter Y-Minus: Pin connectted to Y- for Touch Screen Panel A/D trigger: Signal used to request A/D start

Table2.3.1 Pin names and functions (4/5)

Pin Name	Number of pins	I/O	Function
PJ0 /SDRAS /SRLLB	1	Output Output Output	PortJ0: Output port Row Address Storobe for SDRAM Data enable for SRAM on pins D0 to D7
PJ1 /SDCAS /SRLUB	1	Output Output Output	PortJ1: Output port Column Address Storobe for SDRAM Data enable for SRAM on pins D8to D15
PJ2 /SDWE /SRWR	1	Output Output Output	PortJ2: Output port Write Enable for SDRAM Write for SRAM: strobe signal for writing data
PJ3 SDLLDQM	1	Output Output	PortJ3: Output port Data enable for SDRAM on pins D0 to D7
PJ4 SDLUDQM	1	Output Output	PortJ4: Output port Data enable for SDRAM on pins D8 to D15
PJ5 SDULDQM NDALE	1	Output Output Output	PortJ5: I/O port Data enable for SDRAM on pins D16 to D23 Address Latch Enable for NAND-Flash
PJ6 SDUUDQM NDCLE	1	Output Output Output	PortJ6: I/O port Data enable for SDRAM on pins D24 to D31 Command Latch Enable for NAND-Flash
PJ7 SDCKE	1	Output Output	PortJ7: Output port Clock Enable for SDRAM
PK0 LCP0	1	Output Output	PortK0: Output port LCD driver output pin
PK1 LLP	1	Output Output	PortK1: Output port LCD driver output pin
PK2 LFR	1	Output Output	PortK2: Output port LCD driver output pin
PK3 LBCD	1	Output Output	PortK3: Output port LCD driver output pin
PL0 to PL3 LD0 to LD3	4	Output Output	Port L0 to L7: Output port Data Bus for LCD-driver
PL4 to PL7 LD4 to LD7	4	I/O Output	Port L0 to L7: I/O port Data Bus for LCD-driver
/BE	1	Input	Back-up enable
PM1 MLDALM	1	Output Output	PortM1: Output port Melody / Alarm output pin
PM2 /ALARM /MLDALM	1	Output Output Output	PortM2: Output port RTC alarm output pin Melody / Alarm output pin (inverted)

Table2.3.1 Pin names and functions (5/5)

Pin Name	Number of pins	I/O	Function
D+, D-	2	I/O	USB-data Connecting pin
AM0,AM1	2	in	Operation mode: Fix to AM1="0",AM0="1" for 16-bit external bus starting Fix to AM1="1",AM0="0" for 32-bit external bus starting Fix to AM1="1",AM0="1" for BOOT(32-bit internal-MROM) starting
X1/X2	2	I/O	High-frequency oscillator connection pins
XT1/XT2	2	I/O	Low-frequency oscillator connection pins
/RESET	1	in	Reset: initializes TMP92CH21 (With pull-up resistor)
VREFH	1	in	Pin for reference voltage input to AD converter (H)
VREFL	1	in	Pin for reference voltage input to AD converter (L)
AVCC	1	-	Power supply pin for AD converter
AVSS	1	-	GND pin for AD converter (0 V)
DVCC	3	-	Power supply pins (All Vcc pins should be connected with the power Supply pin)
DVSS	3	-	GND pins (0 V) (All pins shuold be connected with GND(0V)
RTCVCC	1	-	Power supply pin for RTC, low-frequency oscillator and portM

3. OPERATION

This section describes the basic components, functions and operation of the TMP92CH21.

3.1 CPU

The TMP92CH21 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 of 900/H1 CPU are as follows:

Parameter	900/H1 CPU
Width of CPU Address Bus	24-bit
Width of CPU Data Bus	32-bit
Internal Operating Frequency	max.20MHz (at fc=40MHz)
Minimum Bus Cycle	1-clock access(50ns)
Internal RAM	32-bit 1-clock access
Internal Boot-ROM	32-bit 2-clock access
Internal I/O	8/16-bit 2-clock access 8/16-bit 5 to 6-clock access
External Device	8/16/32-bit 2-clock access (can insert some waits)
Minimum Instruction Execution Cycle	1-clock(50ns)
Conditional Jump	2-clock(100ns)
Instruction Queue Buffer	12-byte
Instruction Set	Compatible with TLCS-900/L1,TLCS-900/H2 (LDX instruction is deleted)
CPU mode	Only maximum mode
Micro DMA	8-channel

3.1.2 Reset Operation

When resetting the TMP92CH21 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 /RESET input Low for at least 20 system clocks(16μs at 40MHz).

At reset, since the clock doubler(PLL) is bypassed and clock-gear is set to 1/16,

system clock operates at 1.25MHz(fc=40MHz).
When the Reset has been accepted, the CPU performs the following:

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

  PC<0~7> ← data in location FFFF00H
  PC<8~15> ← data in location FFFF01H
  PC<16~23> ← data in location FFFF02H
- Sets the Stack Pointer (XSP) to 00000000H.
- Sets bits <IFF0 to IFF2> of the Status Register (SR) to 111 (thereby setting the Interrupt Level Mask Register to level 7).
- Clears bits <RFP0 to RFP1> 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. CPU internal registers not mentioned above do not change when the Reset is released.

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.
- Sets the port pins, including the pins that also act as internal I/O, to General-Purpose Input or Output Port Mode.

Internal reset is released as soon as external reset is released.

The operation of memory controller cannot be insured until power supply becomes stable after power-on reset. The external RAM data provided before turning on the TMP92CH21 may be spoiled because the control signals are unstable until power supply becomes stable after power on reset.

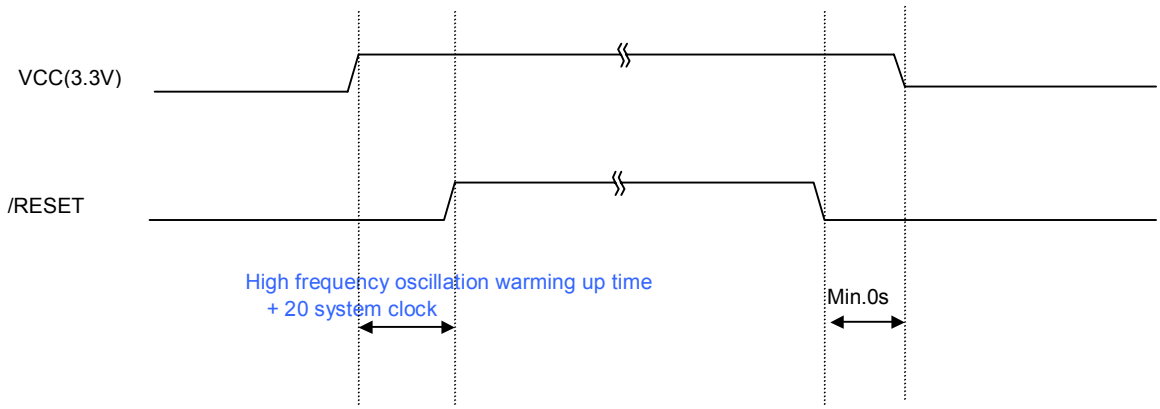


Figure 3.1 Power on Reset Timing Example

3.1.3 Setting of AM0 and AM1

Set AM1 and AM0 pins like Table3.1 according to system usage.

Table 3.1 Operation Mode Setup Table

Operation Mode	Mode Setup input pin		
	/RESET	AM1	AM0
16-bit external bus starting	↑	0	1
32-bit external bus starting		1	0
BOOT(32-bit internal-MROM) starting		1	1

3.2 Memory Map

Figure 3.2 is a memory map of the TMP92CH21.

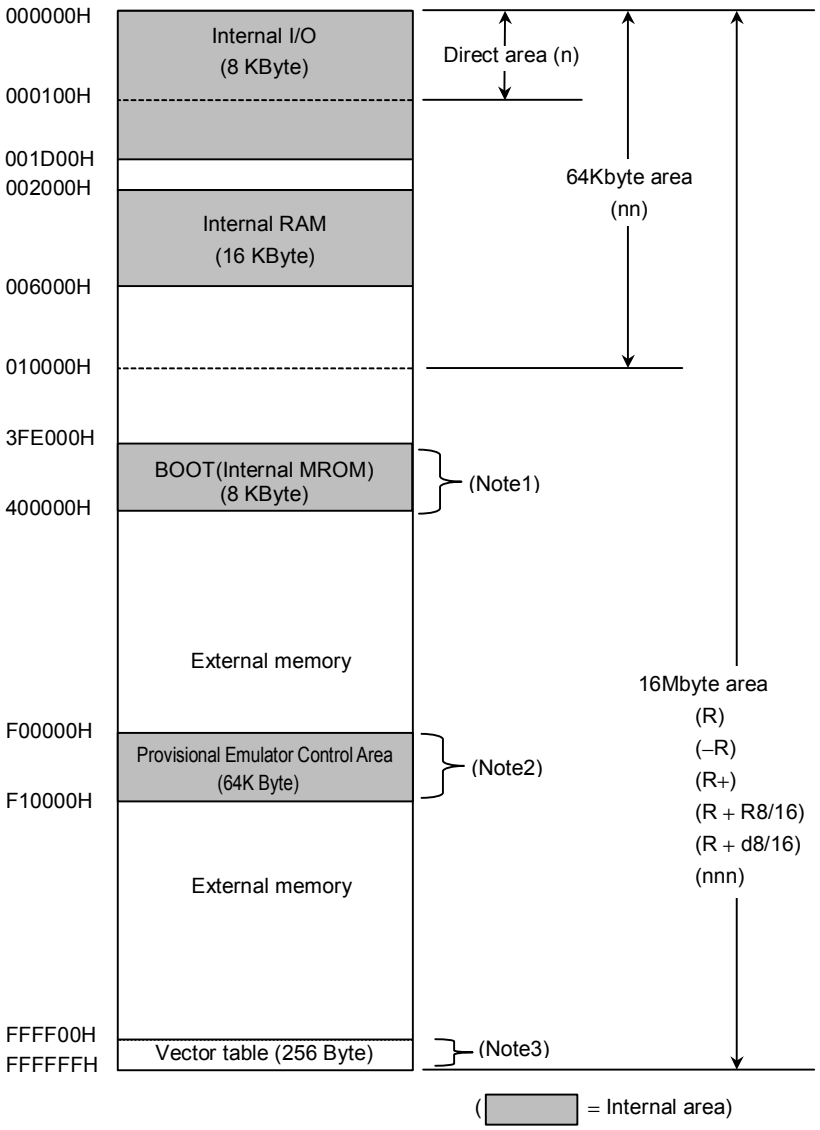


Figure 3.2 Memory Map

Note1: BOOT-prgram(Internal MROM) is mapped only for BOOT-mode. At another starting mode, its area(3FE000H to 3FFFFFH) is mapped to external-memory.

Note2: Provisional emulator control area is for an emulator, it is mapped F00000H to F0FFFFH after reset.

Note3: Don't use the last 16-byte area (FFFFF0H to FFFFFFH). This area is reserved for an emulator.

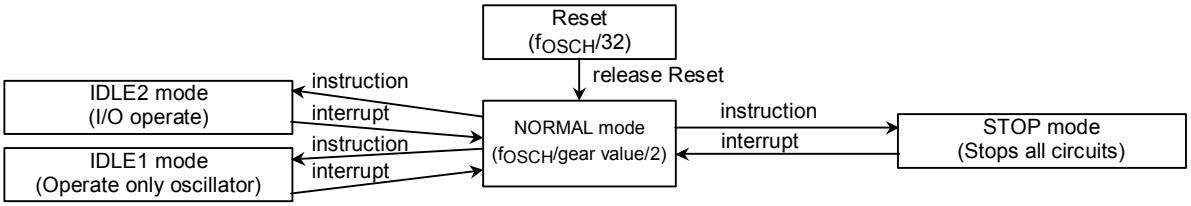
Note4: On emulator /WR signal and /RD signal are asserted, when provisional emulator control area is accessed.
Be carefull to use external memory.

3.3 Clock Function and Standby Function

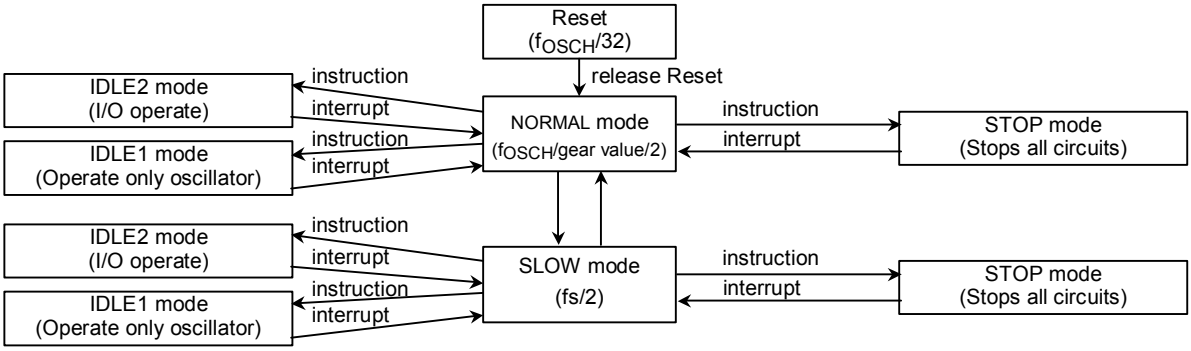
TMP92CH21 contains (1)clock gear, (2)clock doubler(PLL), (3) standby controller and (4) noise-reducing circuit. They are used for low-power, low-noise systems.
This chapter is organized as follows:

3.3.1 Block diagram of system clock
3.3.2 SFRs
3.3.3 System clock controller
3.3.4 Prescaler clock controller
3.3.5 Noise-reducing circuit
3.3.7 Standby controller

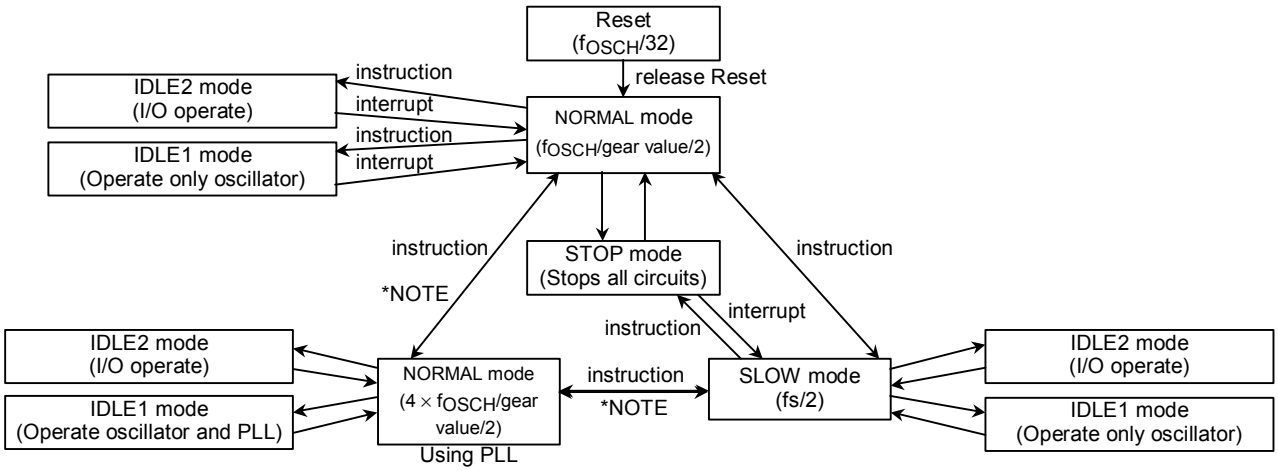
The clock operating modes are as follows: (a) Single Clock Mode (X1, X2 pins only), (b) Dual Clock Mode (X1, X2, XT1 and XT2 pins) and (c) Triple Clock Mode (X1, X2, XT1 and XT2 pins and PLL). Figure 3.3.1 shows a transition figure.



(a) Single clock mode transition figure



(b) Dual clock mode transition figure



(c) Triple clock mode transition Figure

***NOTE)**

- It's prohibited to control PLL in SLOW mode when shifting from SLOW mode to NORMAL mode with use of PLL. (PLL Start up/Stop/Change Write to PLLCR0<PLLON>,PLLCR1<FCSEL> resister)
- If you shift from NORMAL mode with use of PLL to NORMAL mode, execute following setting in the same order.
  1) Change CPU clock(PLLCR0<FCSEL> <- "0" 2)Stop PLL circuit(PLLCR1<PLLON><-"0")
- It's prohibited to shift from NORMAL mode with use of PLL to STOP mode directly.
  You should set NORMAL mode once, and then shift to STOP mode.(You should stop high frequency oscillator after you stop PLL.)

Figure 3.3.1 System clock block diagram

The clock frequency input from the X1 and X2 pins is called fc and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<SYSCK> is called the system clock f_{FPH}. The system clock f_{SYS} is defined as the divided clock of f_{FPH}, and one cycle of f_{SYS} is defined to as one state.

3.3.1 Block diagram of system clock

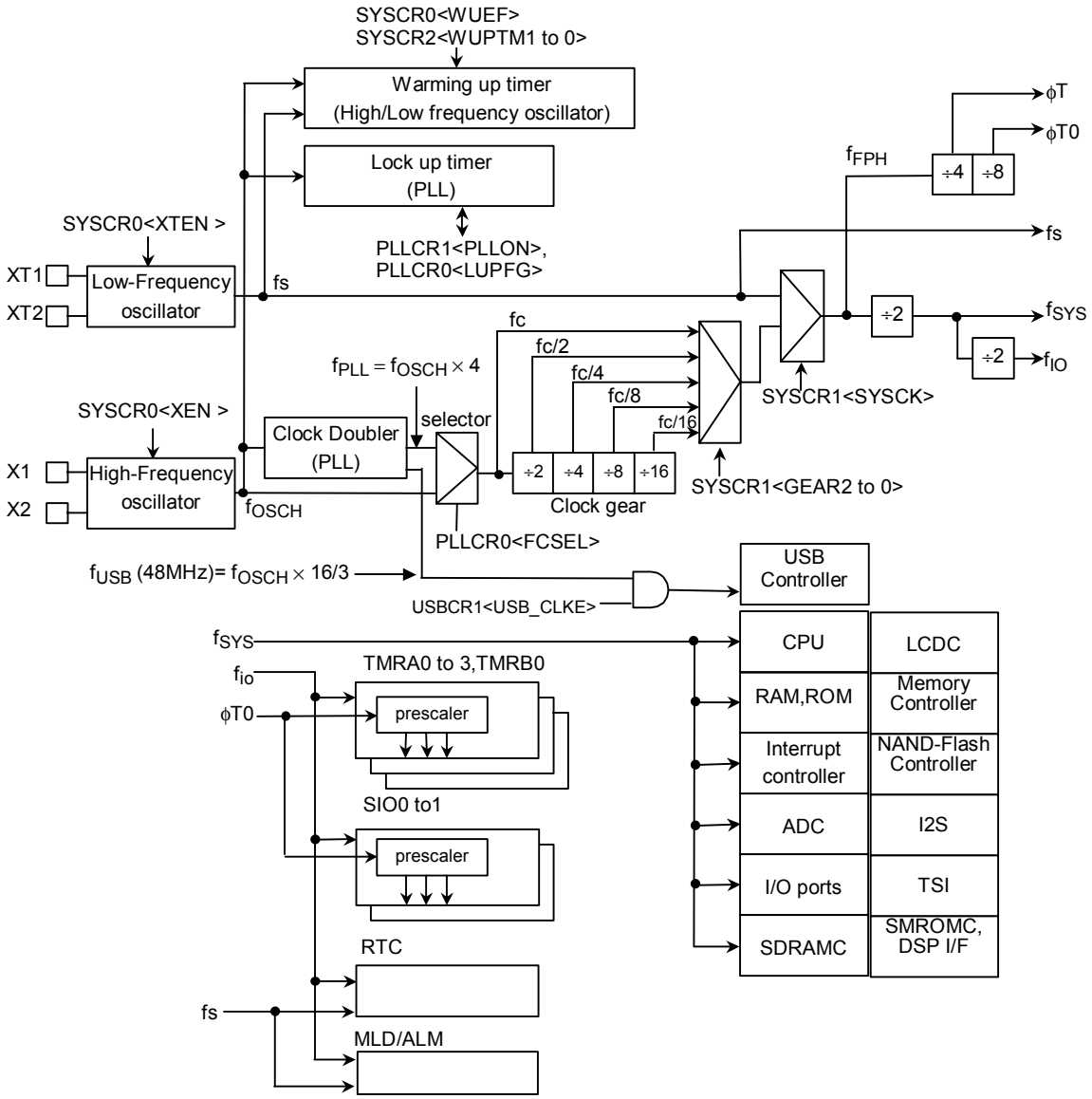


Figure 3.3.2 Block Diagram of System clock

Table 3.3.1 Selection example for f_{OSCH}

Selection example for f_{OSCH}	High-frequency Oscillation : f_{OSCH}	System-Clock : f_{SYS}	USB-Clock : f_{USB}
(a) Needed USB with PLL	9.0MHz	18MHz	48MHz
(b) No-needed USB with PLL	max.10.0MHz	max.20MHz	---
(c) No-needed USB without PLL	max.40.0MHz	max.20MHz	---

(Note1) To use USB, High-frequency Oscillator should be 9.0MHz.

3.3.2 SFR

SYSCR0 (10E0H)

	7	6	5	4	3	2	1	0
bit Symbol	XEN	XTEN				WUEF		
Read/Write	R/W					R/W		
After reset	1	1				0		
Function	High-frequency oscillator (fc) 0: Stop 1: Oscillation	Low-frequency oscillator (fs) 0: Stop 1: Oscillation				Warm-up Timer 0: Write Don't care 1: Write start timer 0: Read end warm-up 1: Read do not end warm-up		

SYSCR1 (10E1H)

	7	6	5	4	3	2	1	0
bit Symbol					SYSCK	GEAR2	GEAR1	GEAR0
Read/Write					R/W	R/W		
After reset					0	1	0	0
Function					Select system clock. 0:fc 1:fs	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			WUPTM1	WUPTM0	HALTM1	HALTM0		
Read/Write	R/W		R/W	R/W	R/W	R/W		
After reset	0		1	0	1	1		
Function	Write '0' .		Warm-Up Timer 00: reserved 01: 2^8/inputted frequency 10:2^{14} 11:2^{16}		HALT mode 00: reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			

Note1: The unassigned register,SYSCR0<bit5 to 3>,SYSCR0<bit1 to 0>,SYSCR1<bit 7 to 4>, and SYSCR2<bit6,bit1 to 0> are read as undefined-value.

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

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
	After reset	0					0	1	1
	Function	Protect flag 0: OFF 1: ON					1: External clock	fc oscillator driverability 1: NORMAL 0: WEAK	fs oscillator driverability 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								

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	PLLON							
Read/Write	R/W							
After reset	0							
Function	Control on/off 0: Off 1: On							

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							
Function	Output/Input buffer drive-register for standby-mode							

(Purpose and How to use)

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 according to the expected pin-status. They will be effective after CPU executed "HALT" instruction.
It is not depend on stand-by mode(IDLE2,IDLE1 or STOP).
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. It contains two oscillation circuits and a clock gear circuit for high-frequency (fc) operation. The register SYSCR1<SYSCK> changes the system clock to either fc or fs, SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling of each oscillator, and SYSCR1<GEAR2 to 0> sets the high-frequency clock gear to either 1, 2, 4, 8 or 16 (fc, fc/2, fc/4, fc/8 or 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 cause the system clock (f_{SYS}) to be set to fc/32 (fc/16 × 1/2) after reset.

For example, f_{SYS} is set to 1.25 MHz when the 40MHz oscillator is connected to the X1 and X2 pins.

(1) Switching from Normal Mode to Slow Mode

When the resonator is connected to the X1 and X2 pins, or to the XT1 and XT2 pins, the warm-up timer can be used to change the operation frequency after stable oscillation has been attained.

The warm-up time can be selected using SYSCR2<WUPTM1 to 0>.

This warm-up timer can be programmed to start and stop as shown in the following examples 1 and 2.

Table 3.3.2 shows the warm-up time.

Note 1: When using an oscillator (other than a resonator) with stable oscillation, a warm-up timer is not needed.

Note 2: The warm-up timer is operated by an oscillation clock. Hence, there may be some variation in warm-up time.

Table 3.3.2 Warming-up times

at f_{OSCH} = 40MHz, fs = 32.768 kHz

Warming-up Time SYSCR2 <WUPTM1,WUPTM0>	Change to Normal Mode	Change to Slow Mode
01 (2^8 / frequency)	6.4 (μs)	7.8 (ms)
10 (2^{14} / frequency)	409.6 (μs)	500 (ms)
11 (2^{16} / frequency)	1.638 (ms)	2000 (ms)

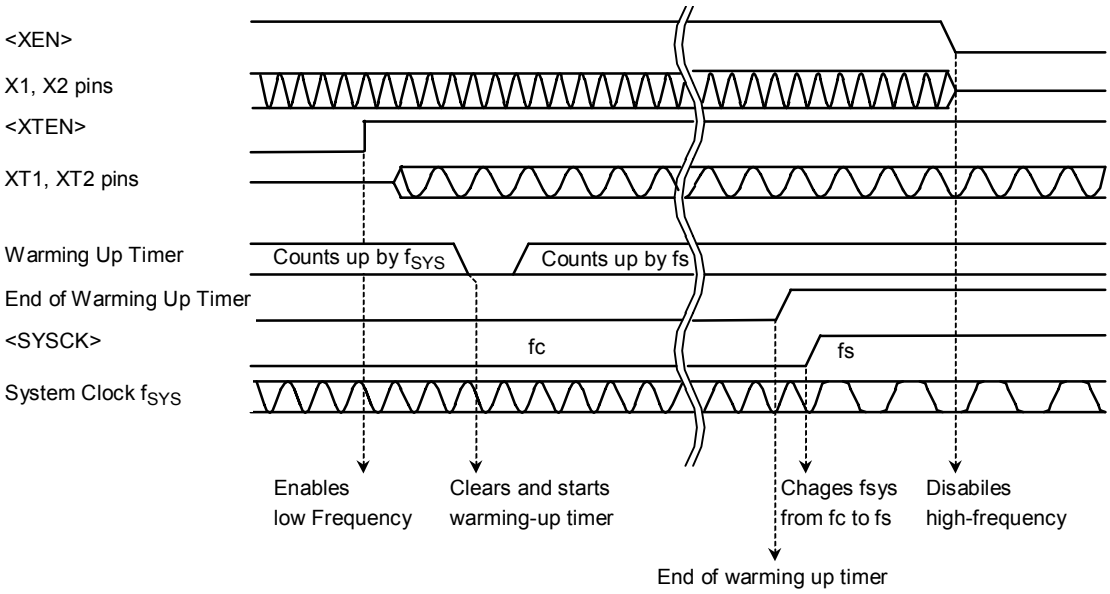
Example 1-Setting the clock

Changing from high frequency (fc) to low frequency (fs).

```
SYSCR0    EQU   10E0H
SYSCR1    EQU   10E1H
SYSCR2    EQU   10E2H
          LD    (SYSCR2), 0X11---XXB  ;  Sets warm-up time to 2^16/fs.
          SET   6, (SYSCR0)           ;  Enables low-frequency oscillation.
          SET   2, (SYSCR0)           ;  Clears and starts warm-up timer.
WUP:      BIT   2, (SYSCR0)           ;  } Detects stopping of warm-up timer.
          JR    NZ, WUP               ;  }
          SET   3, (SYSCR1)           ;  Changes f_SYS from fc to fs.
          RES   7, (SYSCR0)           ;  Disables high-frequency oscillation.
```

(Note) "x" means don't care

"-" means no change



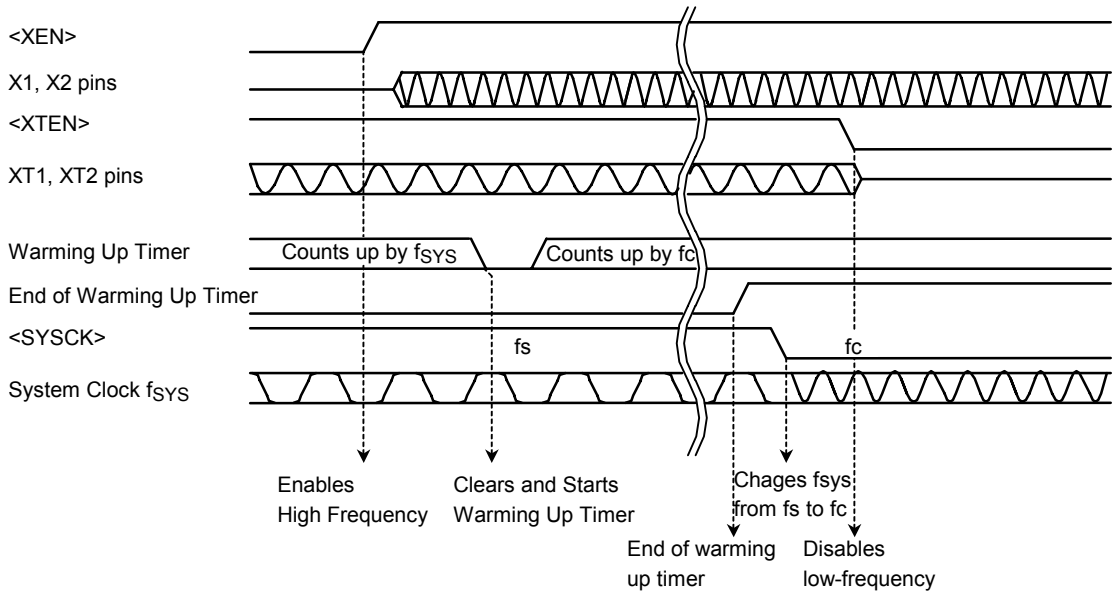
Example 2-Setting the clock

Changing from low frequency (fs) to high frequency (fc).

```
SYSCR0    EQU   10E0H
SYSCR1    EQU   10E1H
SYSCR2    EQU   10E2H
          LD    (SYSCR2), 0X10---XXB  ;  Sets warm-up time to 2^14/fc.
          SET   7, (SYSCR0)           ;  Enables high-frequency oscillation.
          SET   2, (SYSCR0)           ;  Clears and starts warm-up timer.
WUP:      BIT   2, (SYSCR0)           ;  } Detects stopping of warm-up timer.
          JR    NZ, WUP               ;  }
          RES   3, (SYSCR1)           ;  Changes fSYS from fs to fc.
          RES   6, (SYSCR0)           ;  Disables low-frequency oscillation.
```

(Note) "x" means don't care

"-" means no change



(2) Clock gear controller

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

Example 3

Changing to a high-frequency gear

```
SYSCR1    EQU    10E1H

          LD     (SYSCR1), XXXX0100B    ;    Changes fSYS to fc/32.
```

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 warmming 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), XXXX0001B    ;    Changes fSYS to fc/4.
          LD     (DUMMY), 00H           ;    Dummy instruction
Instruction to be executed after clock gear has changed
```

3.3.4 Clock doubler (PLL)

PLL outputs the f_{PLL} clock signal, which is four times as fast as f_{OSCH}. It can use the low-frequency oscillator, even though the internal clock is high-frequency.

A Reset initializes PLL to Stop status, setting to PLLCR0,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 16-stage binary counter. Lock-up time is about 1.6ms at f_{OSCH} = 10MHz.

(note-1) Input frequency limitation for PLL

The limitation of input frequency(High frequency oscillation) for PLL is following.

f_{OSCH} = 6 ~ 10MHz (Vcc = 3.0~ 3.6V)

(note-2) PLLCR0<LUPFG>

The logic of PLLCR0<LUPFG> is different from 900/L1's DFM.

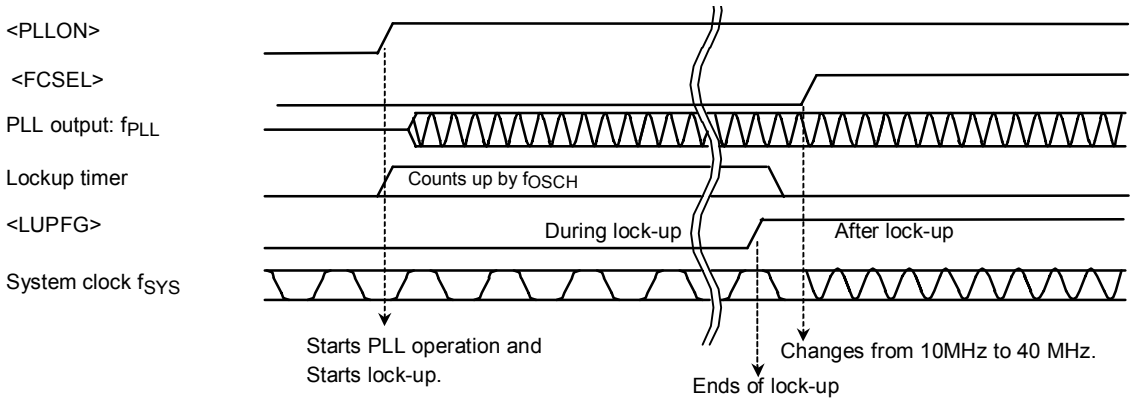
Be careful to judge an end of lock-up time.

The following is an setting example for PLL-starting and PLL-stopping.

(example-1) PLL-starting

```
PLLCR0    EQU    10E8H
PLLCR1    EQU    10E9H
          LD     (PLLCR1), 1XXXXXXXXB  ;  Enables PLL operation and starts lock-up.
LUP:      BIT    5, (PLLCR0)           ;  Detects end of lock-up
          JR     Z, LUP                ;
          LD     (PLLCR0), X1XXXXXXB   ;  Changes fc from 10 MHz to 40 MHz.
```

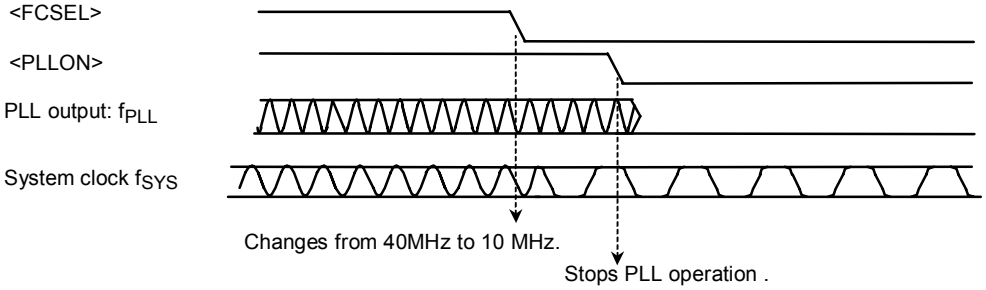
X: Don't care



(example-2) PLL-stopping

```
PLLCR0    EQU    10E8H
PLLCR1    EQU    10E9H
          LD     (PLLCR0), X0XXXXXXB   ;  Changes fc from 40 MHz to10 MHz.
          LD     (PLLCR1), 0XXXXXXXB   ;  Stop PLL.
```

X: Don't care



Limitation point on the use of PLL

1. It's prohibited to execute PLL enable/disable control in the SLOW mode(fs) (writing to PLLCR0 and PLLCR1). You should control PLL in the NORMAL mode.

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

```
LD      (PLLCR0), 00H    ;  Change the clock fPLL to fOSCH
LD      (PLLCR1), 00H    ;  PLL stop
```

3. If you stop high frequency oscillator during using PLL, you should stop PLL before you stop high frequency oscillator.

Examples of settings are below.

(1) Start Up / Change Control

(OK) Low frequency oscillator operation mode(f_s) (high frequency oscillator STOP)
→ High frequency oscillator start up → High frequency oscillator operation mode(f_{OSCH})→ PLL start up → PLL use mode (f_{PLL})

```
        LD      (SYSCR0),  11---1--B    ; High frequency oscillator start/ Warming up start
WUP:    BIT     2,(SYSCR0)              ;
        JR      NZ,WUP                  ; Check for the flag of warming up end
        LD      (SYSCR1),  ----0---B    ; Change the system clock fs to fOSCH
        LD      (PLLCR1),1-------B      ; PLL start up / lock up start
LUP:    BIT     5, (PLLCR0)             ;
        JR      Z,LUP                   ; Check for the flag of lock up end
        LD      (PLLCR0),-1------B      ; Change the system clock fOSCH to fPLL
```

(OK) Low frequency oscillator operation mode(f_s) (high frequency oscillator Operate)
→ High frequency oscillator operation mode(f_{OSCH})→PLL start up → PLL use mode (f_{PLL})

```
        LD      (SYSCR1),  ----0---B    ; Change the system clock fs to fOSCH
        LD      (PLLCR1), 1-------B     ; PLL start up / lock up start
LUP:    BIT     5, (DFMCR0)             ;
        JR      Z,LUP                   ; Check for the flag of lock up end
        LD      (PLLCR0),-1------B      ; Change the system clock fOSCH to fPLL
```

(NG) Low frequency oscillator operation mode(f_s) (high frequency oscillator STOP)
→ High frequency oscillator start up → PLL start up → PLL use mode (f_{PLL})

```
        LD      (SYSCR0),11---1--B      ; High frequency oscillator start/ Warming up start
WUP:    BIT     2,(SYSCR0)              ;
        JR      NZ,WUP                  ; Check for the flag of warming up end
        LD      (PLLCR1),1-------B      ; PLL start up / lock up start
LUP:    BIT     5, (PLLCR0)             ;
        JR      Z,LUP                   ; Check for the flag of lock up end
        LD      (PLLCR0),-1------B      ; Change the internal clock fOSCH to fPLL
        LD      (SYSCR1), ---0---B      ; Change the system clock fS to fPLL
```

(2) Change / Stop Control

(OK) PLL use mode (f_{PLL})→ High frequency oscillator operation mode(f_{OSCH})→ PLL Stop → Low frequency oscillator operation mode(f_s)→High frequency oscillator stop

```
LD      (PLLCR0),-0------B      ; Change the system clock fPLL to fOSCH
LD      (PLLCR1),0-------B      ; PLL stop
LD      (SYSCR1),  ----1---B    ; Change the system clock fOSCH to fs
LD      (SYSCR0),  0-------B    ; High frequency oscillator stop
```

(NG) PLL use mode (f_{PLL})→ Low frequency oscillator operation mode(f_s)→ PLL stop →High frequency oscillator stop

```
LD      (SYSCR1),  ----1---B    ; Change the system clock fPLL to fS
LD      (PLLCR0),   -0------B   ; Change the internal clock (fC) fPLL to fOSCH
LD      (PLLCR1),   0-------B   ; PLL stop
LD      (SYSCR0),  0-------B    ; High frequency oscillator stop
```

(OK) PLL use mode (f_{PLL})→Set the STOP mode →High frequency oscillator operation mode (f_{OSCH})→ PLL stop→ HALT(High frequency oscillator stop)

```
LD      (SYSCR2),  ----01--B    ; Set the STOP mode
                                (This command can execute before use of PLL)
LD      (PLLCR0),-0------B      ; Change the system clock fPLL to fOSCH
LD      (PLLCR1), 0------B      ; PLL stop
HALT                            ; Shift to STOP mode
```

(NG) PLL use mode (f_{PLL})→Set the STOP mode→ HALT(High frequency oscillator stop)

```
LD      (SYSCR2),  ----01--B    ; Set the STOP mode
                                (This command can execute before use of PLL)
HALT                            ; Shift to STOP mode
```

3.3.5 Noise reduction circuits

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

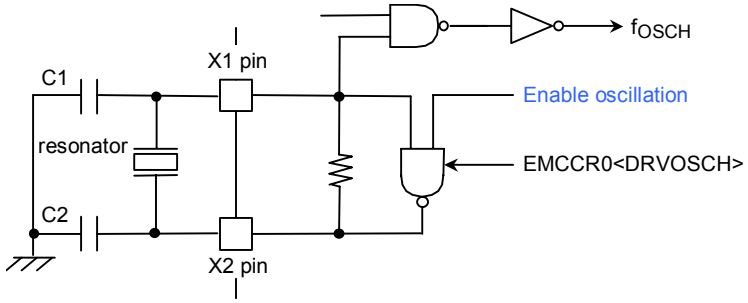
(1) Reduced drivability for high-frequency oscillator
(2) Reduced drivability for low-frequency oscillator
(3) Single drive for high-frequency oscillator
(4) SFR protection of register contents

(1) Reduced drivability for high-frequency oscillator

(Purpose)

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

(Block diagram)



(Setting method)

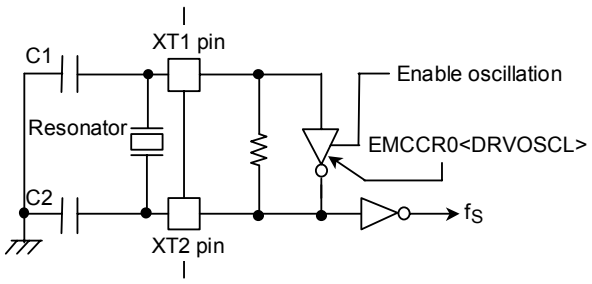
The drivability of the oscillator is reduced by writing"0" to EMCCR0<DRVOSCH> register. By reset, <DRVOSCH> is initialized to "1" and the oscillator starts oscillation by normal-drivability when the power-supply is on.

(2) Reduced drivability for low-frequency oscillator

(Purpose)

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

(Block diagram)



(Setting method)

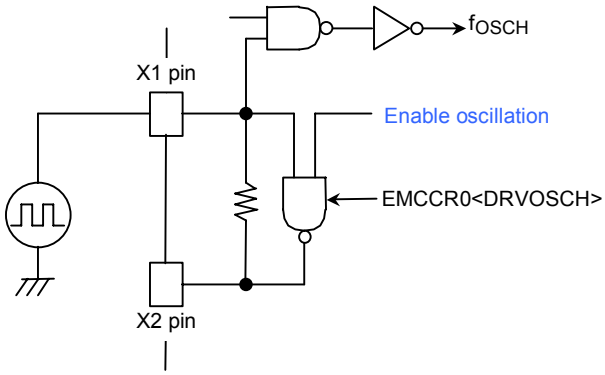
The drivability of the oscillator is reduced by writing 0 to the EMCCR0<DRVOSCL> register. By Reset, <DRVOSCL> is initialized to “1”.

(3) Single drive for high-frequency oscillator

(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”.

(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 it 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,
   BROMCR
2. MMU
   LOCALPX/PY/PZ, LOCALLX/LY/LZ,
   LOCALRX/RY/RZ, LOCALWX/WY/WZ,
3. Clock gear
   SYSCR0, SYSCR1, SYSCR2, EMCCR0
4. PLL
   PLLCR0,PLLCR1

(Operation explanation)

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

（Double key)

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

2^{nd}-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							
Function	Output/Input buffer drive-register for standby-mode							

(Purpose and How to use)
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 according to the expected pin-status. They will be effective after CPU executed "HALT" instruction.
It is not depend on stand-by mode(IDLE2,IDLE1 or STOP).
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:

① 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 3.3.3 SFR seting operation during IDLE2 mode

Internal I/O	SFR
TMRA01	TA01RUN<I2TA01>
TMRA23	TA23RUN<I2TA23>
TMRB0	TB0RUN<I2TB0>
SIO0	SC0MOD1<I2S0>
SIO1	SC1MOD1<I2S1>
A/D converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

② IDLE1: Only the oscillator,RTC (real-time clock) ,USBC and MLD continue to operate.

③ STOP: All internal circuits stop operating.

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

Table 3.3.4 I/O operation during Halt Modes

Halt Mode		IDLE2	IDLE1	STOP
SYSCR2 <HALTM1 to 0>		11	10	01
Block	CPU	Stop		
	I/O ports	Depend on PxDR-register setting		
	TMRA,TMRB	Available to select operation block	Stop	
	SIO			
	A/D converter			
	WDT			
	I2S,LCDC,SDRAMC, Interrupt controller	Operate		
	USBC,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 to 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~INT3, INTKEY and RTC 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 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 necessry enough resetting time (see table 3.3.5) 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 3.3.5 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	◎	×	×*1	–	–	–*1
		INT0 to 4 (Note1)	◎	◎	◎*1	○	○	○*1
		INTALM0 to 4	◎	◎	×	○	○	×
		INTTA0 to 3,INTTB00 to 01	◎	×	×	×	×	×
		INTRX0 to 1,TX0 to 1	◎	×	×	×	×	×
		INTTBO0,INTI2S	◎	×	×	×	×	×
		INTAD	◎	×	×	×	×	×
		INTKEY	◎	◎	◎*1	○	○	○*1
		INTRTC	◎	◎	◎*1	○	○	○*1
		INTUSB	◎	◎	◎*1	○	○	○*1
		INTLCD	◎	×	×	×	×	×
	RESET		◎	◎	◎	◎	◎	◎

◎: After clearing the Halt mode, CPU starts interrupt processing. (RESET initializes the microcont.)

○: 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.

Note 1: 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
8203H   LD     (IIMC), 00H          ; Selects INT0 interrupt rising edge.
8206H   LD     (INTE0AD), 06H       ; Sets INT0 interrupt level to 6.
8209H   EI     5                    ; Sets interrupt level to 5 for CPU.
820BH   LD     (SYSCR2), 28H        ; Sets Halt mode to Idle1 Mode.
820EH   HALT                        ; Halts CPU.

INT0 ──┐/──────────────────────────────► INT0 interrupt routine
                                                    │
                                                  RETI
820FH   LD     XX, XX  ◄────────────────────────────┘
```

(3) Operation

① 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 3.3.5 illustrates an example of the timing for clearance of the IDLE2 Mode Halt state by an interrupt.

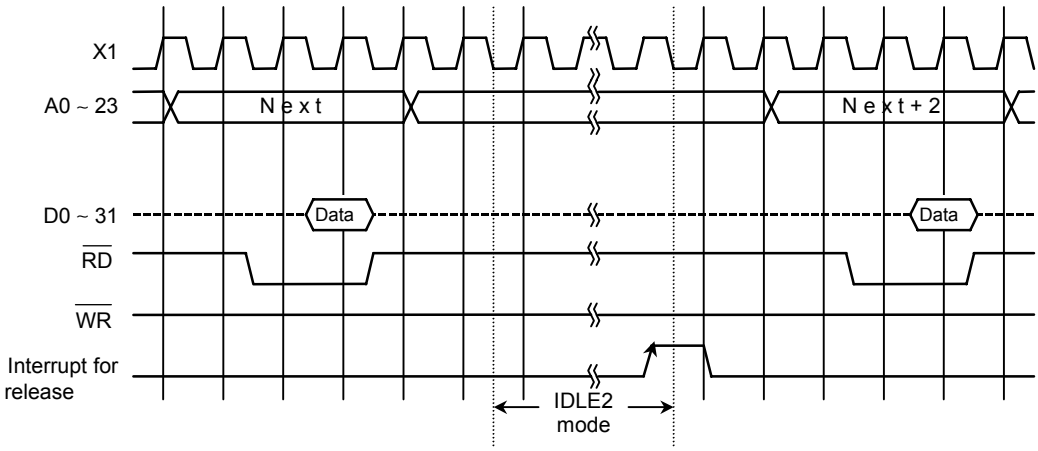


Figure 3.3.7 Timing chart for IDLE2 Mode Halt state cleared by interrupt

② IDLE1 Mode

In IDLE1 Mode, only the internal oscillator and the RTC,USBC 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 3.3.6 illustrates the timing for clearance of the IDLE1 Mode Halt state by an interrupt.

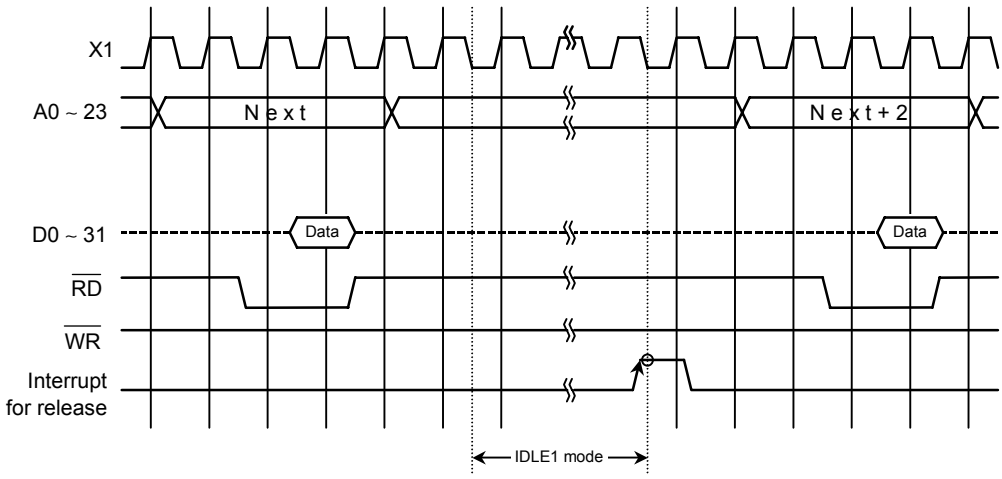


Figure 3.3.8 Timing chart for IDLE1 Mode Halt state cleared by interrupt

③ 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 3.3.7 illustrates the timing for clearance of the STOP Mode Halt state by an interrupt.

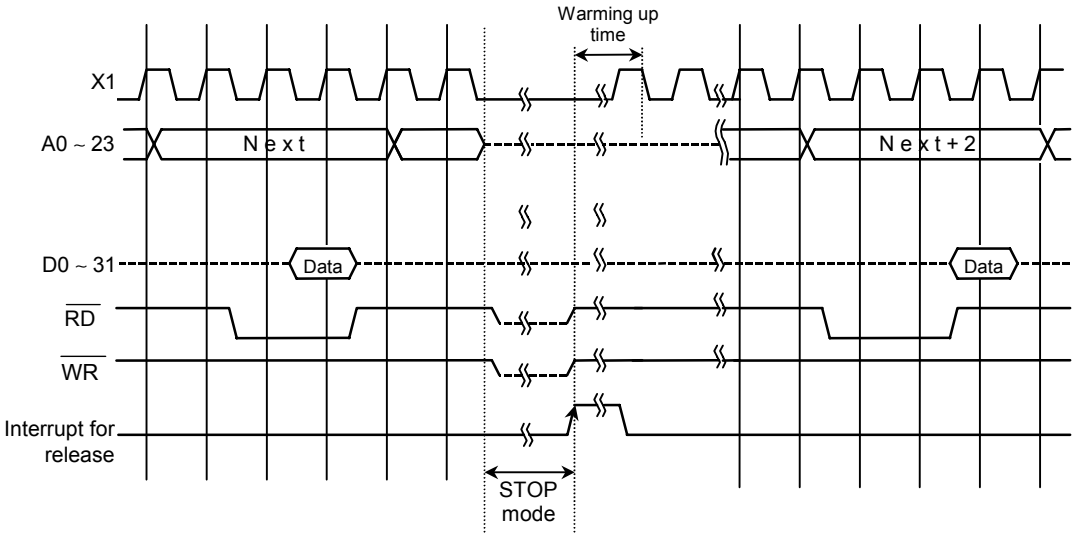


Figure 3.3.9 Timing chart for STOP Mode Halt state cleared by interrupt

Table 3.3.6 Example of warming-up time after releasing STOP-mode

@f_{OSCH} = 16 MHz, fs = 32.768 kHz

SYSCR1 <SYSCK>	SYSCR2<WUPTM1,0>		
	01 (2^8)	10 (2^{14})	11 (2^{16})
0 (fc)	16 us	1.024 ms	4.096 ms
1 (fs)	7.8 ms	500 ms	2000 ms

3.4 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.

The TMP92CH21 has a total of 51 interrupts divided into the following five types:

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

Internal interrupts: 34 sources
- Internal I/O interrupts: 26 sources
- Micro DMA Transfer End interrupts: 8 sources

External interrupts: 7 sources
- Interrupts on external pins (INT0 to INT5, 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 to 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 are processed irrespective of the value in <IFF2 to 0>.

The value in the Interrupt Mask Register <IFF2 to 0> can be changed using the EI instruction (EI num sets <IFF2 to 0> to num). For example, the command EI 3 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 EI 0 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 EI 1).

The DI instruction (sets <IFF2 to 0> to 7) is exactly equivalent to the EI 7 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.

In Micro DMA Mode the CPU automatically transfers data in one-byte, two-byte or four-byte blocks; this mode allows high-speed data transfer to and from internal and external memory and internal I/O ports.

In addition, the TMP92CH21 also has a software start function in which micro DMA processing is requested in software rather than by an interrupt.

Figure 3.4.1 is a flowchart showing overall interrupt processing.

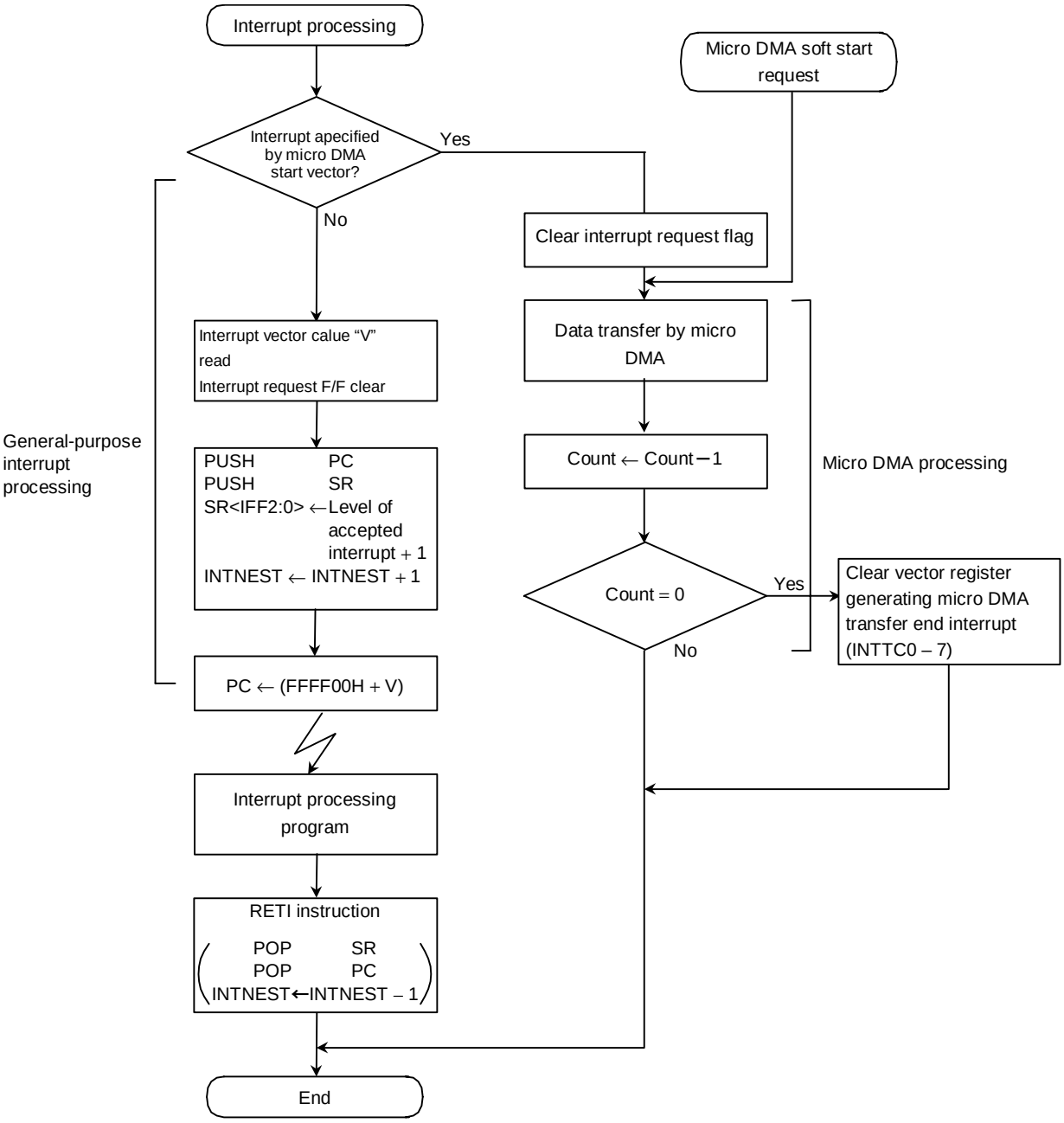


Figure 3.4.1 Interrupt and micro DMA processing sequence

3.4.1 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 to 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 to 0>, the CPU will accept the interrupt. The CPU Interrupt Mask Register <IFF2 to 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 to 0> to 111, disabling all maskable interrupts.

Table 3.4.1 shows the TMP92CH21 interrupt vectors and micro DMA start vectors. FFFF00H~FFFFFFH (256 bytes) is designated as the interrupt vector area.

Table 3.4.1 TMP92CH21 interrupt vectors and micro DMA start vectors

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value	Address refer to Vector	Micro DMA 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	-	-	-
11		INT0: INT0 pin input	0028H	FFFF28H	0AH (Note1)
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		INTALM0: ALM0(8KHz)	003CH	FFFF3CH	0FH
17		INTALM1: ALM1(512Hz)	0040H	FFFF40H	10H
18		INTALM2: ALM2(64Hz)	0044H	FFFF44H	11H
19		INTALM3: ALM3(2Hz)	0048H	FFFF48H	12H
20		INTALM4: ALM4(1Hz)	004CH	FFFF4CH	13H
21		INTP0: Protect0 (WR to special SFR)	0050H	FFFF50H	14H
22		(reserved)	0054H	FFFF54H	15H
23		INTTA0: 8-bit timer 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 wake up	0070H	FFFF70H	1CH
30		INTRTC: RTC(alarm interrupt)	0074H	FFFF74H	1DH
31		INTTBO0: 16-bit timer 0 (overflow)	0078H	FFFF78H	1EH
32		INTLCD: LCDC/LP pin	007CH	FFFF7CH	1FH
33		INTRX0: Serial receive (Channel 0)	0080H	FFFF80H	20H (Note1)
34		INTTX0: Serial transmission (Channel 0)	0084H	FFFF84H	21H
35		INTRX1: Serial receive (Channel 1)	0088H	FFFF88H	22H (Note1)
36		INTTX1: Serial transmission (Channel 1)	008CH	FFFF8CH	23H
37		(reserved)	0090H	FFFF90H	24H (Note1)
38		(reserved)	0094H	FFFF94H	25H
39		INT5: INT5 pin input	0098H	FFFF98H	26H
40		INTI2S: I2S(Channel 0)	009CH	FFFF9CH	27H
41		INTNDF0(NAND Flash Controller Channel 0)	00A0H	FFFFA0H	28H
42		INTNDF1(NAND Flash Controller Channel 1)	00A4H	FFFFA4H	29H
43		(reserved)	00A8H	FFFFA8H	2AH
44		(reserved)	00ACH	FFFFACH	2BH
45		(reserved)	00B0H	FFFFB0H	2CH
46		(reserved)	00B4H	FFFFB4H	2DH
47		(reserved)	00B8H	FFFFB8H	2EH
48		INTUSB: USB	00BCH	FFFFBCH	2FH
49		(reserved)	00C0H	FFFFC0H	30H
50		(reserved)	00C4H	FFFFC4H	31H

51	Maskable	(reserved)	00C8H	FFFFC8H	32H
52		INTAD: AD conversion end	00CCH	FFFFCCH	33H
53		INTTC0: Micro DMA end (Channel 0)	00D0H	FFFFD0H	34H
54		INTTC1: Micro DMA end (Channel 1)	00D4H	FFFFD4H	35H
55		INTTC2: Micro DMA end (Channel 2)	00D8H	FFFFD8H	36H
56		INTTC3: Micro DMA end (Channel 3)	00DCH	FFFFDCH	37H
57		INTTC4: Micro DMA end (Channel 4)	00E0H	FFFFE0H	38H
58		INTTC5: Micro DMA end (Channel 5)	00E4H	FFFFE4H	39H
59		INTTC6: Micro DMA end (Channel 6)	00E8H	FFFFE8H	3AH
60		INTTC7: Micro DMA end (Channel 7)	00ECH	FFFFECH	3BH
- to -		(reserved)	00F0H : 00FCH	FFFFF0H : FFFFFCH	- to -

Note: Micro DMA default priority

If an interrupt request is generated by micro DMA, the interrupt has a higher priority than any other maskable interrupt (irrespective of default channel priority).

Note1 : When standing-up micro DMA, set at edge detect mode.

Note2 : Micro DMA processing cannot be applied.

3.4.2 Micro DMA processing

In addition to general-purpose interrupt processing, the TMP92CH21 also includes a micro DMA function. 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 stand-by by HALT instruction, the requirement of micro DMA will be ignored (pending).

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

(1)Micro DMA operation

When an interrupt request is generated by an interrupt source specified by the Micro DMA Start Vector Register, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request. The eight micro DMA channels allow micro DMA processing to be set for up to eight types of interrupt at once.

When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared. Data in one-byte or two-byte or four-byte 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 start vector register is cleared to 0, the next micro DMA operation is disabled and micro DMA processing terminates.

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).

If an interrupt request is triggered for the interrupt source in use during the interval between the time at which the micro DMA 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 (and not as a general-purpose interrupt), the interrupt level should first be set to 0 (i.e. 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.

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 eight bits of a 32-bit address are not valid).

Three micro DMA transfer modes are supported: one-byte transfers, two-byte (one-word) transfer and four-byte transfer. 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 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 3.4.2 (1), 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 34 different interrupts – the 33 interrupts shown in the micro DMA start vectors in Table 3.4.1 and a micro DMA soft start.
Figure 3.4.2 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: external 8-bit bus, 0 waits, and even-numbered transfer source and transfer destination addresses).

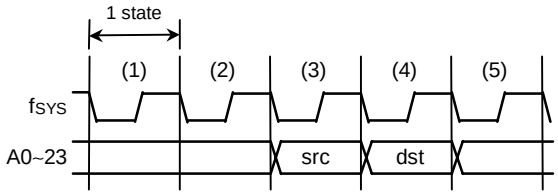


Figure 3.4.2 Timing for micro DMA cycle

State (1),(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 TMP92CH21 can initiate micro DMA either with an interrupt or by using the micro DMA soft start function, in which micro DMA is initiated by a Write cycle which writes to the register DMAR.

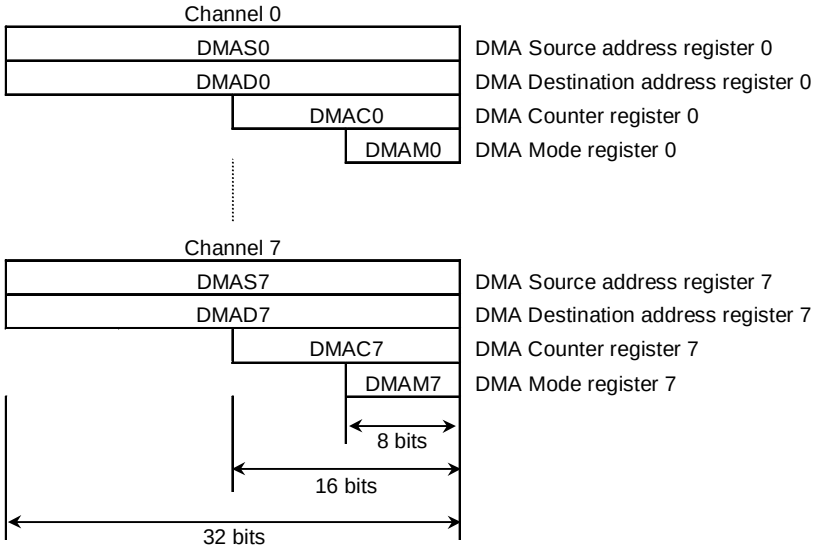
Writing 1 to any bit of the register DMAR causes micro DMA to be performed once. On completion of the transfer, the bits of DMAR which support the end channel are automatically cleared to 0.

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.

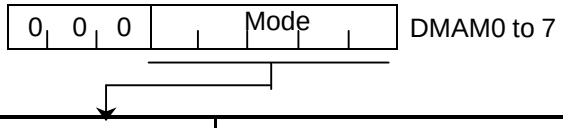
Symbol	NAME	Address	7	6	5	4	3	2	1	0
DMAR	DMA Request	109h (no RMW)	DREQ7	DREQ6	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1	DREQ0
			R/W							
			0	0	0	0	0	0	0	0

(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 State number
0 0 0 z z	Destination INC mode (DMADn +) ← (DMASn) DMACn ← DMACn - 1 if DMACn = 0 then INTTCn	5states
0 0 1 z z	Destination DEC mode (DMADn -) ← (DMASn) DMACn ← DMACn - 1 if DMACn = 0 then INTTCn	5states
0 1 0 z z	Source INC mode (DMADn) ← (DMASn +) DMACn ← DMACn - 1 if DMACn = 0 then INTTCn	5states
0 1 1 z z	Source DEC mode (DMADn) ← (DMASn -) DMACn ← DMACn – 1 if DMACn = 0 then INTTCn	5states
1 0 0 z z	Source and Destination INC mode (DMADn +) ← (DMASn +) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	6states
1 0 1 z z	Source and Destination DEC mode (DMADn -) ← (DMASn -) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	6states
1 1 0 z z	Source and Destination Fixed mode (DMADn) ← (DMASn) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	5states
1 1 1 z z	Counter mode DMASn ← DMASn + 1 DMACn ← DMACn – 1 if DMACn = 0 then INTTCn	5states

ZZ: 00 = 1-byte transfer
01 = 2-byte transfer
10 = 4-byte transfer
11 = (reserved)

Note: 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.

The Transfer Mode Register should not be set to any value other than those listed above.

3.4.3 Interrupt controller operation

The block diagram in Figure 3.4.3 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 52 interrupt channels there is an interrupt request flag (consisting of a flip-flop), an interrupt priority setting register and a micro DMA start vector register. The interrupt request flag latches interrupt requests from the peripherals. The flag is cleared to zero 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 a micro DMA burst transfer is terminated, and when an instruction that clears the interrupt for that channel is executed (by writting 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. INTE0AD 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. The priority of non-maskable interrupt (Watchdog Timer interrupts) is fixed at 7. 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 to 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 to 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 to 0> (i.e. interrupts with a priority higher than the interrupt being processed) will be accepted.
When interrupt processing has been completed (i.e. after execution of a RETI instruction), the CPU restores to SR<IFF2 to 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 start vector. Writing the start vector of the interrupt source for the micro DMA processing (see Table 3.4.1), enables the corresponding interrupt to be processed by micro DMA processing. The values must be set in the micro DMA parameter registers (e.g. DMAS and DMAD) prior to micro DMA processing.

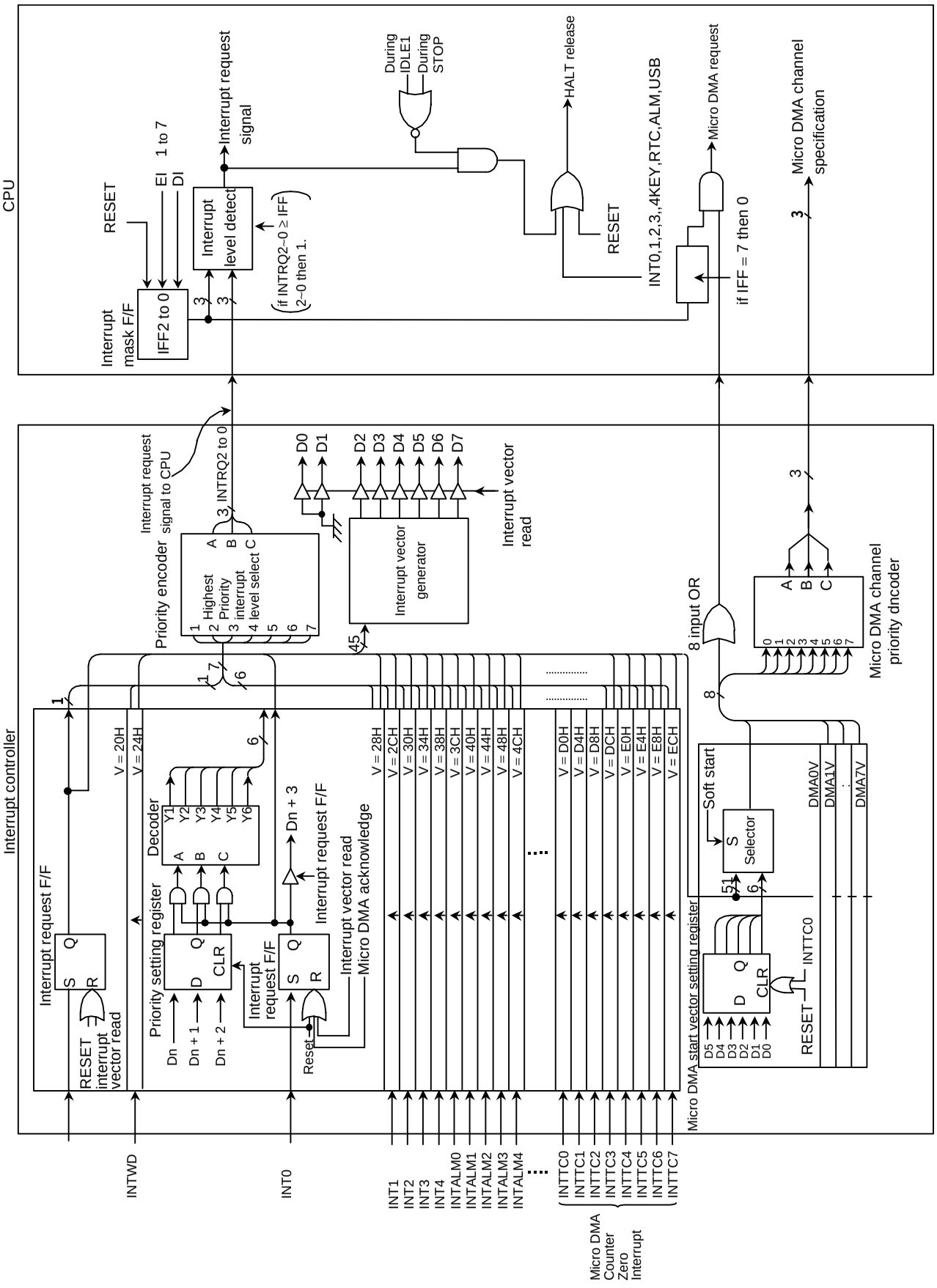
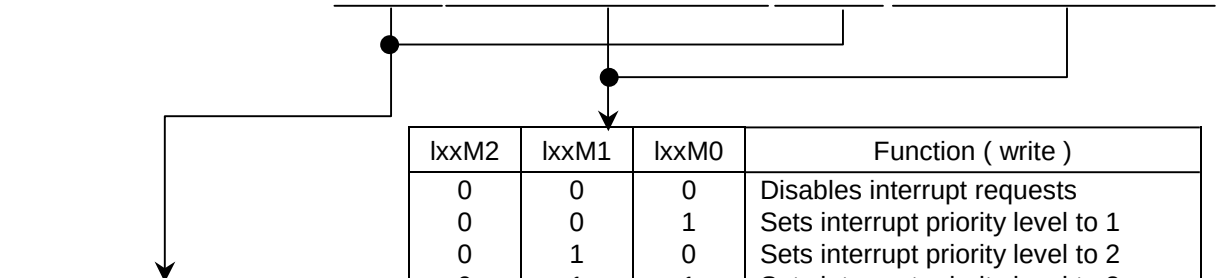


Figure 3.4.3 Block Diagram of Interrupt Controller

(1) Interrupt level setting registers

Symbol	NAME	Address	7	6	5	4	3	2	1	0
INTE0AD	INT0 & INTAD Enable	F0h	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			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			
INTE34	INT3 & INT4 Enable	D1h	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0				0			
INTE5I2S	INT5 & INTI2S Enable	EBh	INTI2S				INT5			
			II2SC	II2SM2	II2SM1	II2SM0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0				0			
INTETA01	INTTA0 & INTTA1 Enable	D4h	INTTA1(Timer1)				INTTA0(Timer0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0				0			
INTETA23	INTTA2 & INTTA3 Enable	D5h	INTAT3(Timer3)				INTAT2(Timer2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0				0			
INTETB01	INTTB0 & INTTB1 Enable	D8h	INTTB1(Timer4)				INTTB0(Timer4)			
			ITB1C	ITB1M2	ITB1M1	ITB1M0	ITB0C	ITB0M2	ITB0M1	ITB0M0
			R	R/W			R	R/W		
			0				0			
INTETBO0	INTTBO0 (Overflow) Enable	DAh	-				INTTBO0			
			-	-	-	-	ITBO0C	ITBO0M2	ITBO0M1	ITBO0M0
							R	R/W		
			Note: Always fixed to 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			
INTES1	INTRX1 & INTTX1 Enable	DCh	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0				0			
INTEUSB	INTUSB Enable	E3h	-				INTUSB			
			-	-	-	-	IUSB0C	IUSBM2	IUSBM1	IUSBM0
							R	R/W		
			Note: Always fixed to 0				0			
INTEALM01	INTALM0 & INTALM1 Enable	E5h	INTALM1				INTALM0			
			IA1C	IA1M2	IA1M1	IA1M0	IA0C	IA0M2	IA0M1	IA0M0
			R	R/W			R	R/W		
			0				0			
INTEALM23	INTALM2 & INTALM3 Enable	E6h	INTALM3				INTALM2			
			IA3C	IA3M2	IA3M1	IA3M0	IA2C	IA2M2	IA2M1	IA2M0
			R	R/W			R	R/W		
			0				0			

Symbol	NAME	Address	7	6	5	4	3	2	1	0
INTEALM4	INTALM4 Enable	E7h	-				INTALM4			
			-	-	-	-	IA4C	IA4M2	IA4M1	IA4M0
			R				R/W			
			Note: Always fixed to 0				0			
INTERTC	INTRTC Enable	E8h	-				INTRTC			
			-	-	-	-	IRC	IRM2	IRM1	IRM0
			R				R/W			
			Note: Always fixed to 0				0			
INTECKEY	INTKEY Enable	E9h	-				INTKEY			
			-	-	-	-	IKC	IKM2	IKM1	IKM0
			R				R/W			
			Note: Always fixed to 0				0			
INTLCD	INTLCD Enable	EAh	-				INTLCD			
			-	-	-	-	ILCD1C	ILCDM2	ILCDM1	ILCDM0
			R				R/W			
			Note: Always fixed to 0				0			
INTEND01	INTNDF0 & INTNDF1 Enable	ECh	INTNDF1				INTNDF0			
			IN1C	IN1M2	IN1M1	IN1M0	IN0C	IN0M2	IN0M1	IN0M0
			R	R			R	R/W		
			0				0			
INTEP0	INTP0 Enable	EEh	-				INTP0			
			-	-	-	-	IP0C	IP0M2	IP0M1	IP0M0
			R				R/W			
			Note: Always fixed to 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
INTETC01	INTTC0 & INTTC1 Enable	F1h	INTTC1(DMA1)				INTTC0(DMA0)			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0				0			
INTETC23	INTTC2& INTTC3 Enable	F2h	INTTC3(DMA3)				INTTC2(DMA2)			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0				0			
INTETC45	INTTC4 & INTTC5 Enable	F3h	INTTC5(DMA5)				INTTC4(DMA4)			
			ITC5C	ITC5M2	ITC5M1	ITC5M0	ITC4C	ITC4M2	ITC4M1	ITC4M0
			R	R/W			R	R/W		
			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			
INTWDT	INTWD Enable	F7h	-				INTWD			
			-	-	-	-	ITCWD	-	-	-
							R			
			Note: Always fixed to 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

(2) External interrupt control

Symbol	NAME	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt Input Mode Control	F6H (no 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	0:INT0 edge mode 1:INT0 level mode	Write "0"

Note:
*INT0 Level Enable

0	Rising edge detect INT
1	"H"level INT

Note 1 : Disable INT0 request before changing INT0 pin mode from level-sense to edge-sense.

Setting example:

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

Note: X = Don't care; "-" = No change.

Note 2 : See electrical characteristics in section 4 for external interrupt input pulse width.

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> = 1
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	P97	Rising Edge	<I5EDGE> = 0
		Falling Edge	<I5EDGE> = 1

(3) SIO Receive interrupt control

Symbol	NAME	Address	7	6	5	4	3	2	1	0
SIMC	SIO Interrupt Mode Control	F5H (no RMW)	-	-	-	-	-	-	IR1LE	IR0LE
									W	W
			0	-	-	-	-		1	1
			Write '0'						0:INTRX1 edge mode 1:INTRX1 level mode	0:INTRX0 edge mode 1:INTRX0 level mode

INTRX1 Level Enable

0	Rising edge detect INTRX1
1	"H"level INTRX1

INTRX0 rising edge Enable

0	Rising 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 start vector, as given in Table 3.4 (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 (no RMW)	-	-	-	-	-	-	-	-
			W							
			0	0	0	0	0	0	0	0
			Interrupt Vector							

(5)Micro DMA start vector registers

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

When the micro DMA transfer counter value reaches zero, the micro DMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA start vector register is cleared, and the micro DMA start source for the channel is cleared. Therefore, in order for micro DMA processing to continue, the micro DMA start vector register must be set again during processing of the micro DMA transfer end interrupt.

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

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

(6) 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 zero. 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

(7)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.

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 (IIMC), 00H; Switches from level to edge.
	LD (INTCLR), 0AH; Clears interrupt request flag.
	EI

INTRX	In edge mode(the register SIMC<IRxLE> set to “0”),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

3.5 Function of Ports

TMP92CH21 has I/O port pins that are shown in table 3.5.1 In addition to functioning as general-purpose I/O ports, these pins are also used by internal CPU and I/O functions. Table 3.5.2 lists I/O registers and their specifications.

Table 3.5.1 Port Functions (1/2)

(R: PU= with programmable pull-up resistor, 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 2	P20 to P27	8	I/O	–	Bit	D16 to D23,KO0 to KO7
Port 3	P30 to P37	8	I/O	–	Bit	D24 to D31,
Port 4	P40 to P47	8	O	–	(Fixed)	A0 to A7
Port 5	P50 to P57	8	O	–	(Fixed)	A8 to A15
Port 6	P60 to P67	8	I/O	–	Bit	A16 to A23
Port 7	P70	1	Output	–	(Fixed)	/RD
	P71	1	I/O	–	Bit	/WRLL,/NDRE
	P72	1	I/O	–	Bit	/WRLU,/NDWE
	P73	1	Output	–	(Fixed)	EA24
	P74	1	Output	–	(Fixed)	EA25
	P75	1	I/O	–	Bit	R/W,NDR/B
	P76	1	I/O	–	Bit	/WAIT
Port 8	P80	1	Output	–	(Fixed)	/CS0
	P81	1	Output	–	(Fixed)	/CS1,/SDCS
	P82	1	Output	–	(Fixed)	/CS2,/CSZA,/SDCS
	P83	1	Output	–	(Fixed)	/CS3
	P84	1	Output	–	(Fixed)	/CSZB,/WRUL,/ND0CE
	P85	1	Output	–	(Fixed)	/CSZC,/WRUU,/ND1CE
	P86	1	Output	–	(Fixed)	/CSZD,/SRULB
	P87	1	Output	–	(Fixed)	/CSZE,/SRUUB
Port 9	P90	1	I/O	–	Bit	TXD0,I2SCKO
	P91	1	I/O	–	Bit	RXD0,I2SDO
	P92	1	I/O	–	Bit	SCLK0,/CTS0,I2SWS
	P93	1	I/O	–	Bit	LGOE0
	P94	1	I/O	–	Bit	LGOE1
	P95	1	Output	–	(Fixed)	LGOE2
	P96	1	Input	PD	(Fixed)	INT4,PX
	P97	1	Input	–	(Fixed)	INT5,PY
Port A	PA0 to PA2	3	Input	U	(Fixed)	KI0 to KI2
	PA3 to PA6	4	I/O	U	Bit	LD8 to LD11, KI3 to KI6
	PA7	1	Input	U	(Fixed)	KI7
Port C	PC0	1	I/O	–	Bit	INT0,TA1OUT
	PC1	1	I/O	–	Bit	INT1,TA3OUT
	PC2	1	I/O	–	Bit	INT2,TB0OUT0
	PC3	1	I/O	–	Bit	INT3
	PC6	1	I/O	–	Bit	KO8,LDIV
	PC7	1	I/O	–	Bit	/CSZF,LCP1
Port F	PF0	1	I/O	–	Bit	TXD0,TXD1
	PF1	1	I/O	–	Bit	RXD0,RXD1
	PF2	1	I/O	–	Bit	SCLK0,/CTS0, SCLK1,/CTS1
	PF7	1	Output	–	(Fixed)	SDCLK

Table 3.5.1 Port Functions (2/2)

(R: PU= with programmable pull-up resistor, 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 G	PG0 to PG1	2	Input	–	(Fixed)	AN0 toAN1
	PG2	1	Input	–	(Fixed)	AN2,MX
	PG3	1	Input	–	(Fixed)	AN3, /ADTRG,MY
Port J	PJ0	1	Output	–	(Fixed)	/SDRAS,/SRLLB
	PJ1	1	Output	–	(Fixed)	/SDCAS,SRLUB
	PJ2	1	Output	–	(Fixed)	/SDWE,/SRWR
	PJ3	1	Output	–	(Fixed)	SDLLDQM
	PJ4	1	Output	–	(Fixed)	SDLUDQM
	PJ5	1	I/O	–	Bit	SDULDQM,NDALE
	PJ6	1	I/O	–	Bit	SDUUDQM,NDCLE
	PJ7	1	Output	–	(Fixed)	SDCKE
Port K	PK0	1	Output	–	(Fixed)	LCP0
	PK1	1	Output	–	(Fixed)	LLP
	PK2	1	Output	–	(Fixed)	LFR
	PK3	1	Output	–	(Fixed)	LBCD
Port L	PL0 to PL3	4	Output	–	(Fixed)	LD0 to LD3
	PL4 to PL7	4	I/O	–	Bit	LD4 to LD7
Port M	PM1	1	Output	–	(Fixed)	MLDALM
	PM2	1	Output	–	(Fixed)	/ALARM,/MLDALM

Table 3.5.2 I/O Registers and Specifications (1/3) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port 1	P10 to P17	Input port	X	0	0	None
		Output port	X	1		
		D8 to D15 bus	X	X	1	
Port 2	P20 to P27	Input port	X	0	0	0
		Output port	X	1		
		D16 to D23 bus	X	X	1	
		KO0 to KO7	X	1	0	1
Port 3	P30 to P37	Input port	X	0	0	None
		Output port	X	1		
		D24 to D31 bus	X	X	1	
Port 4	P40 to P47	Output port	X	None	0	None
		A0 to A7 output	X		1	
Port 5	P50 to P57	Output port	X	None	0	
		A8 to A15 output	X		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 P72 P75 to P76	Input port	X	0	0	
	P70	/RD output	X	None	1	
	P71	/WRLL output	1	1		
		/NDRE output	0			
	P72	/WRLU output	1			
		/NDWE output	0			
	P73	EA24 output	X	None		
	P74	EA25 output				
	P75	R/W output	X	1	1	
		NDR/B input	X	0	1	
	P76	/WAIT input	X		1	
Port 8	P80 to P87	Output Port	X	None	0	0
	P80	/CS0 output	X		1	0
	P81	/CS1 output	X		1	0
		/SDCS output	X		X	1
	P82	/CS2 output	X		1	0
		/CSZA Output	X		0	1
		/SDCS output	X		1	1
	P83	/CS3 output	X		1	0
	P84	/CSZB output	X		1	0
		/WRUL output	X		0	1
		/ND0CE output	X		1	1
	P85	/CSZC output	X		1	0
		/WRUU output	X		0	1
		/ND1CE output	X		1	1
	P86	/CSZD output	X		1	0
		/SRULB output	X		X	1
	P87	/CSZE output	X		1	0
		/SRUUB output	X		X	1

Table 3.5.2 I/O Registers and Specifications (2/3) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port 9	P90 to P97	Input port	X	0	0	0
	P90 to P95	Output port	X	1	0	
	P90	TXD0 output	X	1	1	
		I2SCKO output	X	0	1	
		TXD0 output(Open Drain)	X	1	1	1
	P91	RXD0 input	X	0	0	None
		I2SDO output	X	0	1	
	P92	SCLK0 output	X	1	1	
		I2SWS output	X	0	1	
		SCLK0,/CTS0 input (note1)	X	0	0	
		LGOE0 output	X	0	1	
		LGOE1 output	X	0	1	
		LGOE2 output	X	0	1	
	P96	INT4 input	X	None	1	
	P97	INT5 input	X	None	1	
Port A	PA0 to PA7	Input port	None	0	0	None
		KI0 to KI7 input			1	
	PA3 to PA6	LD8 to LD11 output		1	0	None
Port C	PC0 to PC3 PC6 to PC7	Input port	X	0	0	None
		Output port	X	1	0	
	PC0	INT0 input	X	0	1	
		TA1OUT output	X	1	1	None
	PC1	INT1 input	X	0	1	None
		TA3OUT output	X	1	1	
	PC2	INT2 input	X	0	1	None
		TB0OUT0 output	X	1	1	None
	PC3	INT3 input	X	0	1	None
	PC6	LDIV output	X	1	1	
		KO8 output (Open drain)	X	0	1	
	PC7	LCP1 output	X	1	1	
		/CSZF output	X	0	1	
Port F	PF0 to PF2	Input port	X	0	0	0
	PF0 to PF2,PF7	Output port	X	1	0	
	PF0	TXD0 output	X	1	1	0
		TXD1 output	X	0	1	
		TXD0/TXD1 output (Open Drain)	X	1/0	1	1
	PF1	RXD0 input	X	0	0	None
		RXD1 input				
	PF2	SCLK0 output	X	1	1	
		SCLK1 output	X	0	1	
		SCLK0,/CTS0 input	X	0	0	
		SCLK1,/CTS1 input			0	
	PF7	SDCLK output	X	None	1	

(note1) To use P92-pin as SCLK0-input or /CTS0-input, set "1" to PF<PF2>.

Table 3.5.2 I/O Registers and Specifications (3/3) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port G	PG0 to PG3	Input port	X	None	None	None
		AN0 to AN3 input				
	PG3	/ADTRG input				
	PG2	MX output				
	PG3	MY outout				
Port J	PJ0 to PJ7	Output port	X	1	0	None
	PJ5 to PJ6	Input port	X	0	0	
	PJ0	/SDRAS,/SRLLB output	X	None	1	
	PJ1	/SDCAS,/SRLUB output	X		1	
	PJ2	/SDWE,/SRWR output	X		1	
	PJ3	SDLLDQM output	X		1	
	PJ4	SDLUDQM output	1		1	
	PJ5	SDULDQM output	1	1	1	
		NDALE output	0			
	PJ6	SDUUDQM output	1			
		NDCLE output	0			
	PJ7	SDCKE output	X	None	1	
Port K	PK0 to PK3	Output port	X	None	0	None
	PK0	LCP0 output	X		1	
	PK1	LLP output	X		1	
	PK2	LFR output	X		1	
	PK3	LBCD output	X		1	
Port L	PL4 to PL7	Input Port	X	0	0	None
	PL0 to PL7	Output Port	X	1	0	
	PL0 to PL7	LD0 to LD7 output	X	1	1	
Port M	PM1 to PM2	Output Port	X	None	0	None
	PM1	MLDALM output	X		1	
	PM2	/MLDALM output	0		1	
		/ALARM output	1		1	

After a Reset, the port pins listed below function as general-purpose I/O port pins.

A Reset sets I/O pins, which can be programmed for either input, or output to be input ports pins. Setting the port pins for internal function use must be done in software.

3.5.1 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).

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

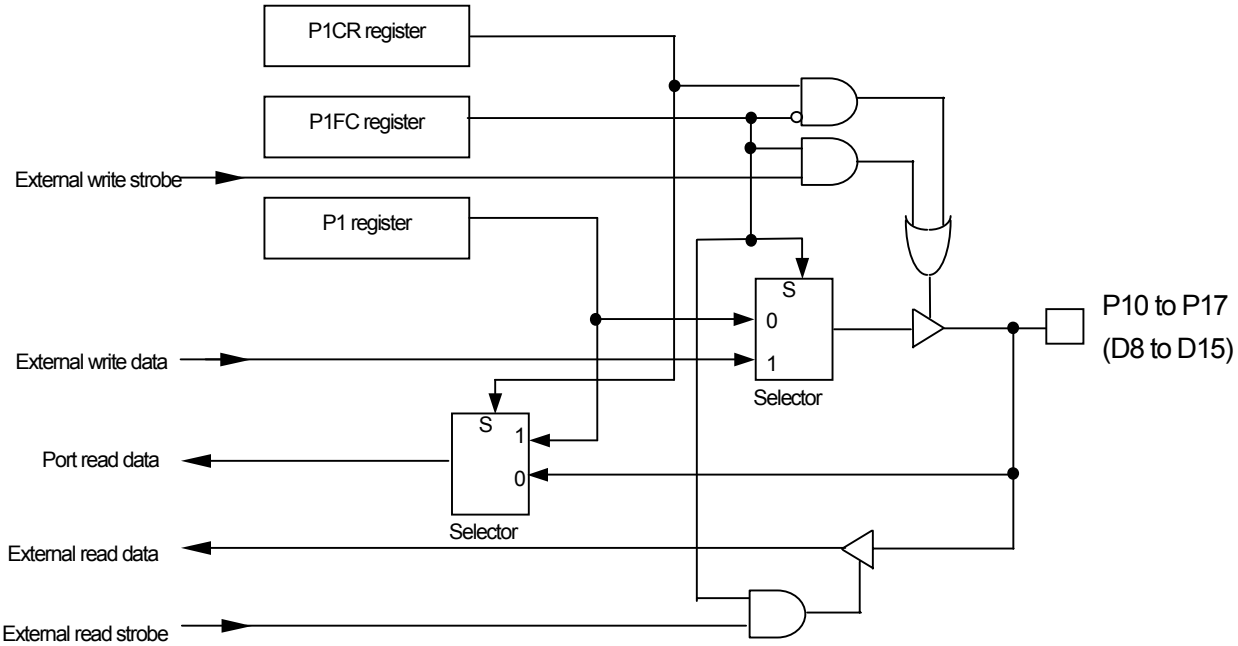


Figure 3.5.1 Port1

Table 3.5.3 Port1 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P1	PORT1	0004H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			0	0	0	0	0	0	0	0
			Input/Output							
P1CR	PORT1 Control Register	0006H	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	-	-	-	-	-	-	-	P1F
										W
										0/1
			0:PORT 1:Data Bus(D8 to D15)							
P1DR	PORT1 Drive Register	0081H	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

3.5.2 Port 2 (P20 to P27)

Port2 is an 8-bit general-purpose I/O port. Bits can be individually set as either inputs or outputs by control register P2CR and function register P2FC.

In addition to functioning as a general-purpose I/O port, port2 can also function either as a data bus (D16 to D23) or key-board interface pin KO0 to KO7 which can set to open drain output buffer.

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Input port
1	0	Data Bus (D16 to D23)
1	1	Input port

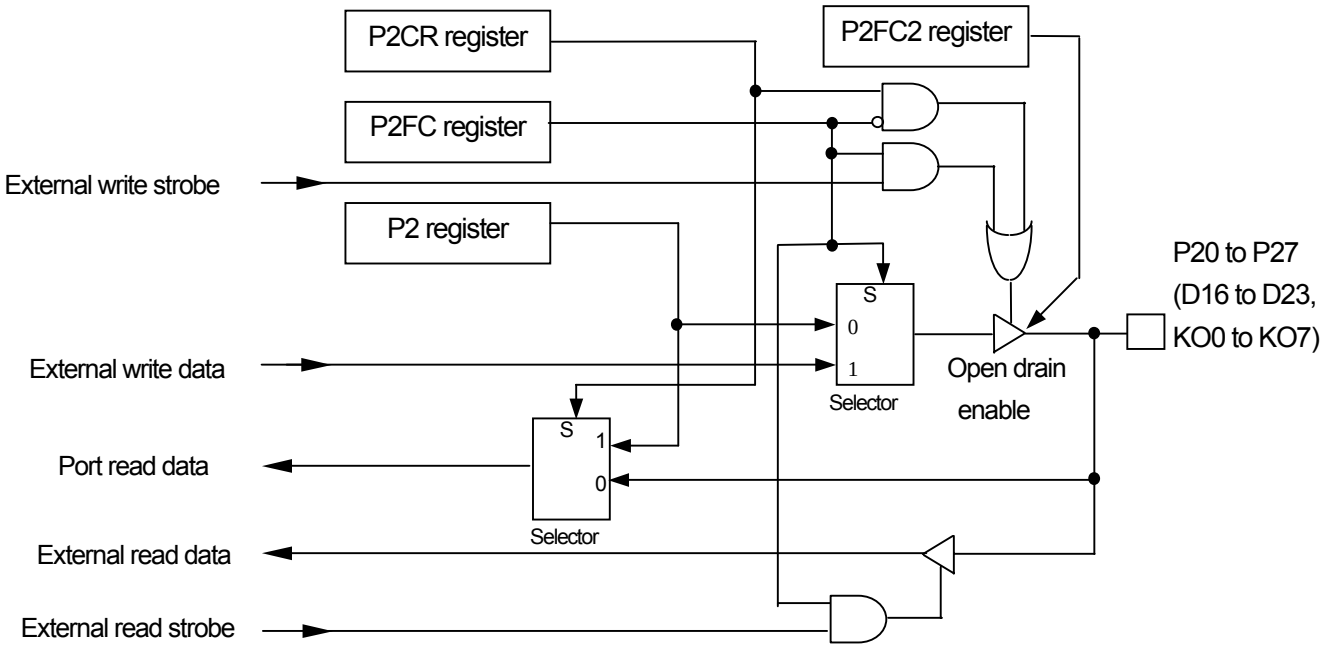


Figure 3.5.2 Port2

Table 3.5.4 Port2 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P2	PORT2	0008H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			0	0	0	0	0	0	0	0
			Input/Output							
P2CR	PORT2 Control Register	000AH	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0:Input 1:Output							
P2FC	PORT2 Function Register	000BH	-	-	-	-	-	-	-	P2F
										W
										0/1
			0:PORT 1:Data Bus(D16 to D23)							
P2FC2	PORT2 Function Register2	0009H	P27F2	P26F2	P25F2	P24F2	P23F2	P22F2	P21F2	P20F2
			W							
			0							
			0:CMOS output, 1:Open drain output							
P2DR	PORT2 Drive Register	0082H	P27D	P26D	P25D	P24D	P23D	P22D	P21D	P20D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

3.5.3 Port 3 (P30 to P37)

Port3 is an 8-bit general-purpose I/O port. Bits can be individually set as either inputs or outputs by control register P3CR and function register P3FC.

In addition to functioning as a general-purpose I/O port, port3 can also function either as a data bus (D24 to D31)

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Input port
1	0	Data Bus (D24 to D31)
1	1	Input port

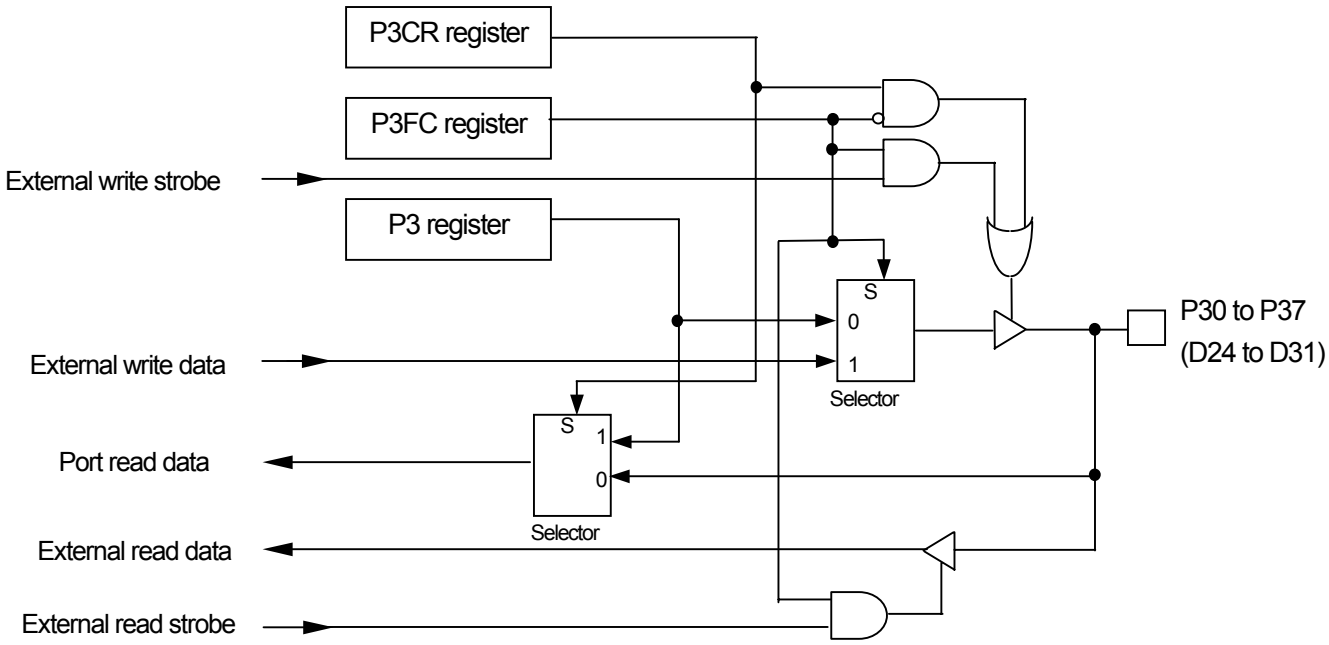


Figure 3.5.3 Port3

Table 3.5.5 Port3 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P3	PORT3	000CH	P37	P36	P35	P34	P33	P32	P31	P30
			R/W							
			0	0	0	0	0	0	0	0
			Input/Output							
P3CR	PORT3 Control Register	000EH	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
			W							
			0	0	0	0	0	0	0	0
			0:Input 1:Output							
P3FC	PORT3 Function Register	000FH	-	-	-	-				P3F
							W	W	W	W
							0	0	0	0/1
			<P3F> 0:PORT, 1:Data Bus(D24 to D31)							
P3DR	PORT3 Drive Register	0083H	P37D	P36D	P35D	P34D	P33D	P32D	P31D	P30D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

3.5.4 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).

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

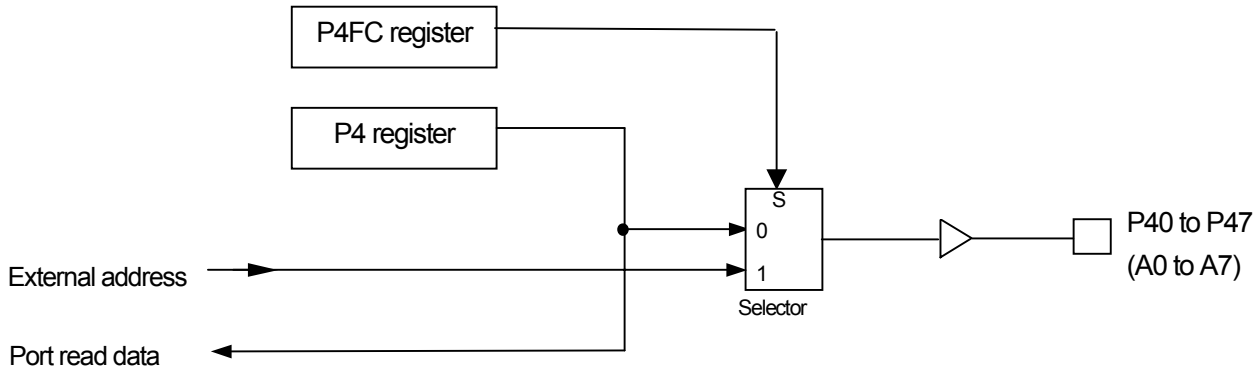


Figure 3.5.4 Port4

Table 3.5.6 Port4 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P4	PORT4	0010H	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			0	0	0	0	0	0	0	0
			Input/Output							
P4FC	PORT4 Function Register	0013H	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
			W							
			0/1							
			0:PORT 1:Address Bus(A0 to A7)							
P4DR	PORT4 Drive Register	0084H	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

3.5.5 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).

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address Bus(A8 to A15)
1	0	Address Bus(A8 to A15)
1	1	Output port

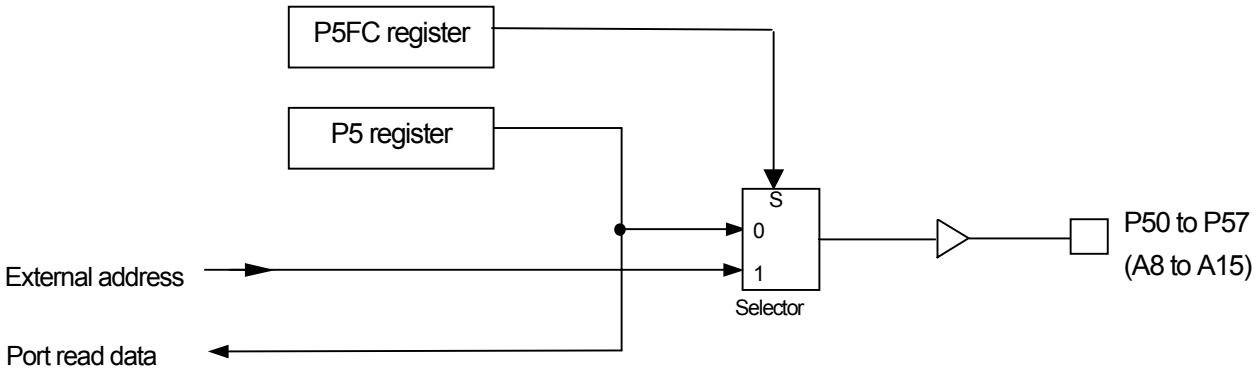


Figure 3.5.5 Port5

Table 3.5.7 Port5 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P5	PORT5	0014H	P57	P56	P55	P54	P53	P52	P51	P50
			R/W							
			0	0	0	0	0	0	0	0
			Input/Output							
P5FC	PORT5 Function Register	0017H	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
			W							
			0/1							
			0:PORT 1:Address Bus(A8 to A15)							
P5DR	PORT5 Drive Register	0085H	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

3.5.6 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 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).

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address Bus(A16 to A23)
1	0	Address Bus(A16 to A23)
1	1	Input port

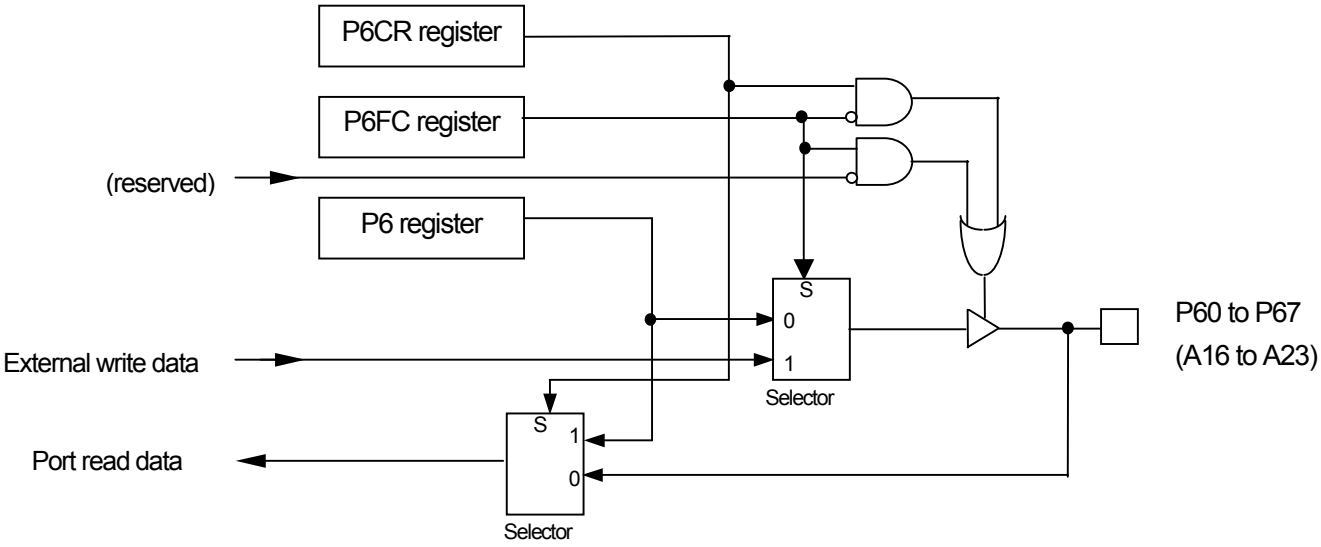


Figure 3.5.6 Port6

Table 3.5.8 Port6 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P6	PORT6	0018H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			0	0	0	0	0	0	0	0
			Input/Output							
P6CR	PORT6 Control Register	001AH	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	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
			W							
			0/1							
			0:PORT 1:Address Bus(A16 to A23)							
P6DR	PORT6 Drive Register	0086H	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

3.5.7 Port 7 (P70 to P76)

Port7 is a 7-bit general-purpose I/O port(P70,P73 and P74 are 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,P73 and P74 pins to output port mode, and P71,P72,P75 and P76 pin to input port mode.

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	/RD pin
1	0	/RD pin
1	1	P70 Output port

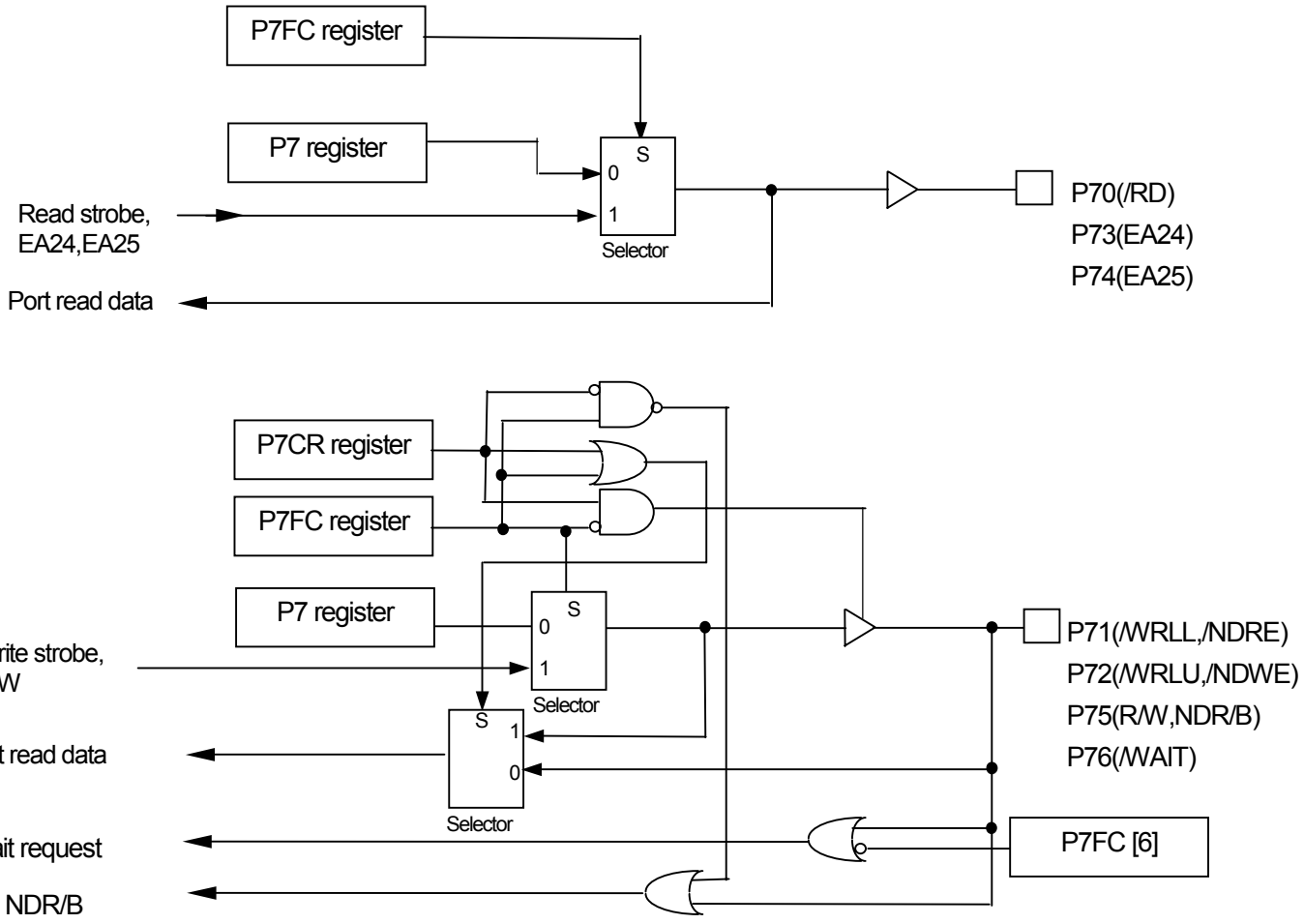


Figure 3.5.7 Port7

Table 3.5.9 Port7 Registers

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P7	PORT7	001CH	-	P76	P75	P74	P73	P72	P71	P70
				R/W						
			-	1	1	0	0	1	1	1
				In/Out		Output		In/Out		Output
P7CR	PORT7 Control Register	001EH	-	P76C	P75C	-	-	P72C	P71C	-
				W	W			W	W	
			-	0	0	-	-	0	0	-
				0:Input 1:Output						
P7FC	PORT7 Function Register	001FH	-	P76F	P75F	P74F	P73F	P72F	P71F	P70F
				W						
			-	0	0	0	0	0	0	0/1
				0:PORT 1:/WAIT	0:PORT 1:R/W, NDR/B	0:PORT 1:EA25	0:PORT 1:EA24	0:PORT 1: /NDWE at <P72>=0 /WRLU at <P72>=1	0:PORT 1: /NDRE at <P71>=0 /WRLL at <P71>=1	0:PORT 1:/RD
P7DR	PORT7 Drive Register	0087H	-	P76D	P75D	P74D	P73D	P72D	P71D	P70D
				R/W						
				1						
				Output/Input buffer drive-register for standby-mode						

P72 setting

<P72C> / <P72F>	0	1
0	Input port	Output port
1	(reserved)	/NDWE out (at <P72>=0) /WRLH out (at <P72>=1)

P71 setting

<P71C> / <P71F>	0	1
0	Input port	Output port
1	(reserved)	/NDRE out (at <P71>=0) /WRLL out (at <P71>=1)

P76 setting

<P76C> / <P76F>	0	1
0	Input port	Output port
1	/WAIT input	(reserved)

P75 setting

<P75C> / <P75F>	0	1
0	Input port	Output port
1	NDR/B input	R/W out

3.5.8 Port 8 (P80 to P87)

Port80 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”.

Port8 also function as interface-pin for external memory.

Above setting is used the function register P8FC.

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

Resetting resets P80F to P87F of P8FC to “0” and P8FC2 to “0”, sets all bits to output ports.

Port82 initial state

AM1	AM0	Function Setting after reset is released
0	0	Don’t use this setting
0	1	“0” Output port
1	0	“0” Output port
1	1	“1” Output port

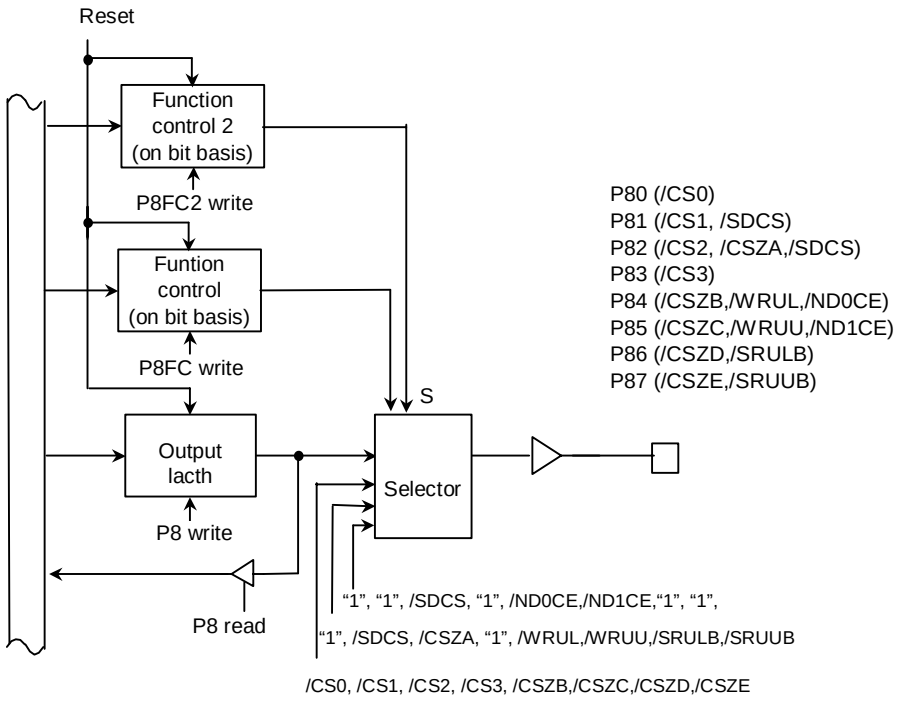


Figure 3.5.9 Port 8

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					0/1	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							
Function	0: PORT 1: /CSZE	0: PORT 1: /CSZD	Refer below table	Refer below table	0: PORT 1: /CS3	0: PORT 1: /CS2	0: PORT 1: /CS1	0: PORT 1: /CS0

Port 8 Function Register 2

P8FC2 (0021H)

	7	6	5	4	3	2	1	0
bit Symbol	P87F2	P86F2	P85F2	P84F2	P83F2	P82F2	P81F2	P80F2
Read/Write	W							
After reset	0							
Function	0: <P87F> 1:/SRUUB	0: <P86F> 1:/SRULB	Refer below table	Refer below table	Write “0”	0: <P82F> 1:/CSZA	0: <P81F> 1:/SDCS	Write “0”

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							
Function	Output/Input buffer drive-register for standby-mode							

P85 setting

<P85F> / <P85F2>	0	1
0	Output port	/CSZC out
1	/WRUU out	/ND1CE out

P84 setting

<P84F> / <P84F2>	0	1
0	Output port	/CSZB out
1	/WRUL out	/ND0CE out

P82 setting

<P82F> / <P82F2>	0	1
0	Output port	/CS2 out
1	/CSZA out	/SDCS out

(Note) Read-modify-write is prohibited for P8FC and P8FC2 .

Figure 3.5.10 Register for Port 8

3.5.9 Port9 (P90 to P97)

P90 to P94 are 5-bit general-purpose I/O port. I/O can be set on bit basis using the control register. Resetting sets P90 to P94 to input port and all bits of output latch to”1”.

P95 is 1-bit general-purpose output port and 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) Port90 (TXD0,I2SCKO), Port91 (RXD0,I2SDO), Port 92 (SCLK0,/CTS0,I2SWS)

Port90 to 92 are general-purpose I/O port. They are also used either SIO0 or I2S. Each pins are below.

	SIO-mode (SIO0 module)	UART,IrDA-mode (SIO0 module)	I2S-mode (I2S module)	SIO-mode (I2S module)
P90	TXD0 (data output)	TXD0 (data output)	I2SCKO (clock output)	I2SCKO (clock output)
P91	RXD0 (data input)	RXD0 (data input)	I2SDO (data output)	I2SDO (data output)
P92	SCLK0 (clock input or output)	/CTS0 (Clear to send)	I2SWS (word-select output)	(No use)

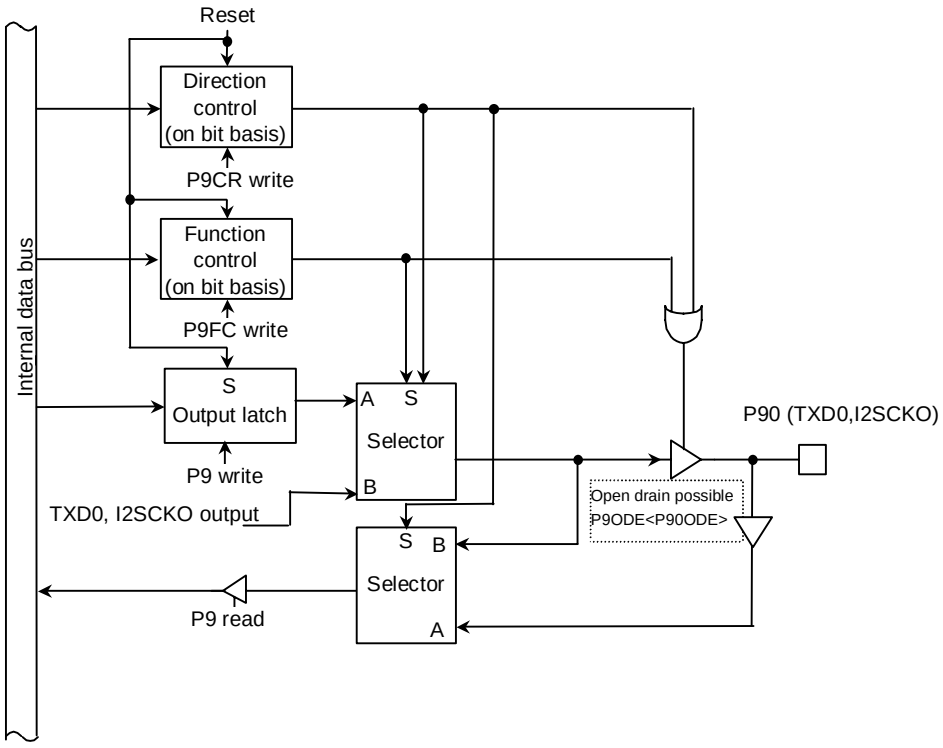


Figure 3.5.11(1) P90

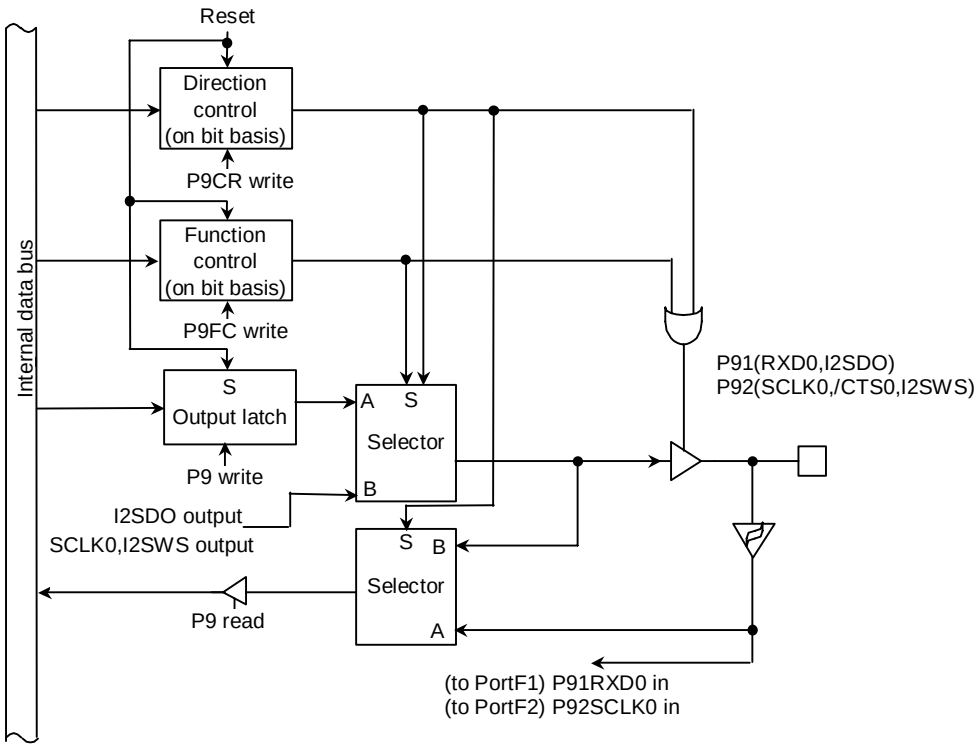


Figure 3.5.11(2) P91 and P92

(2) P93 (LGOE0), P94(LGOE1), P95(LGOE2)

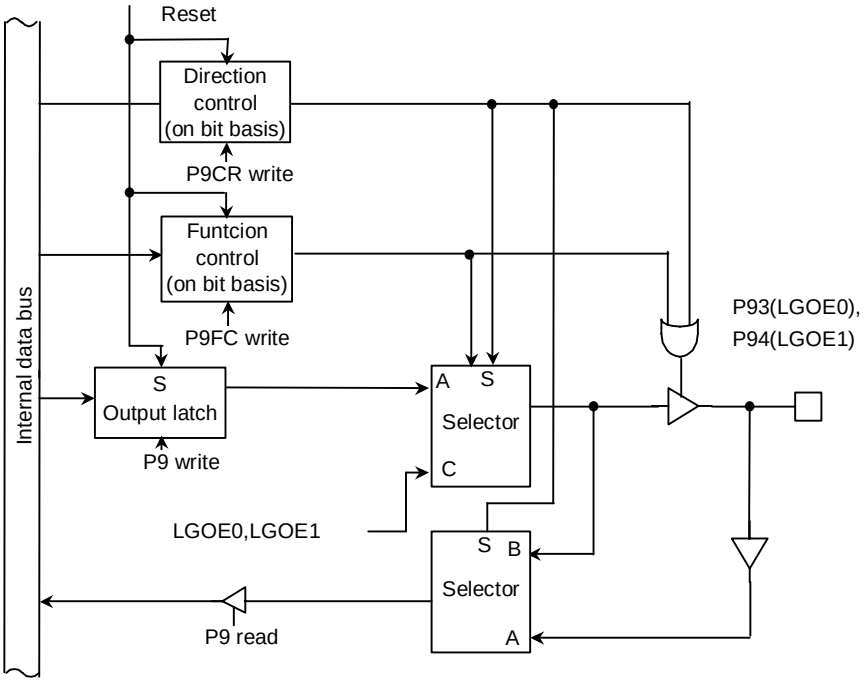


Figure 3.5.12 Port 93 and 94

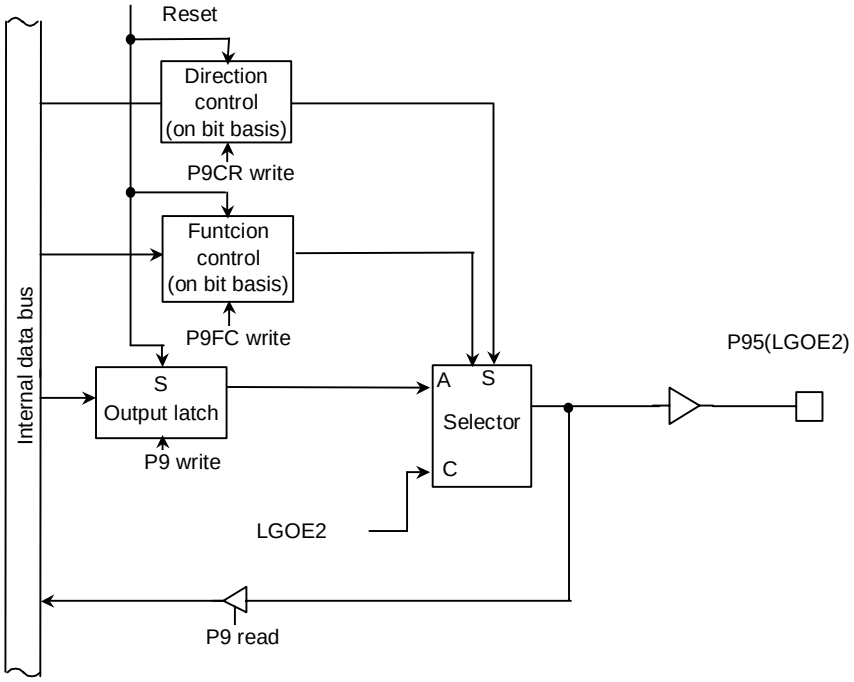


Figure 3.5.13 Port95

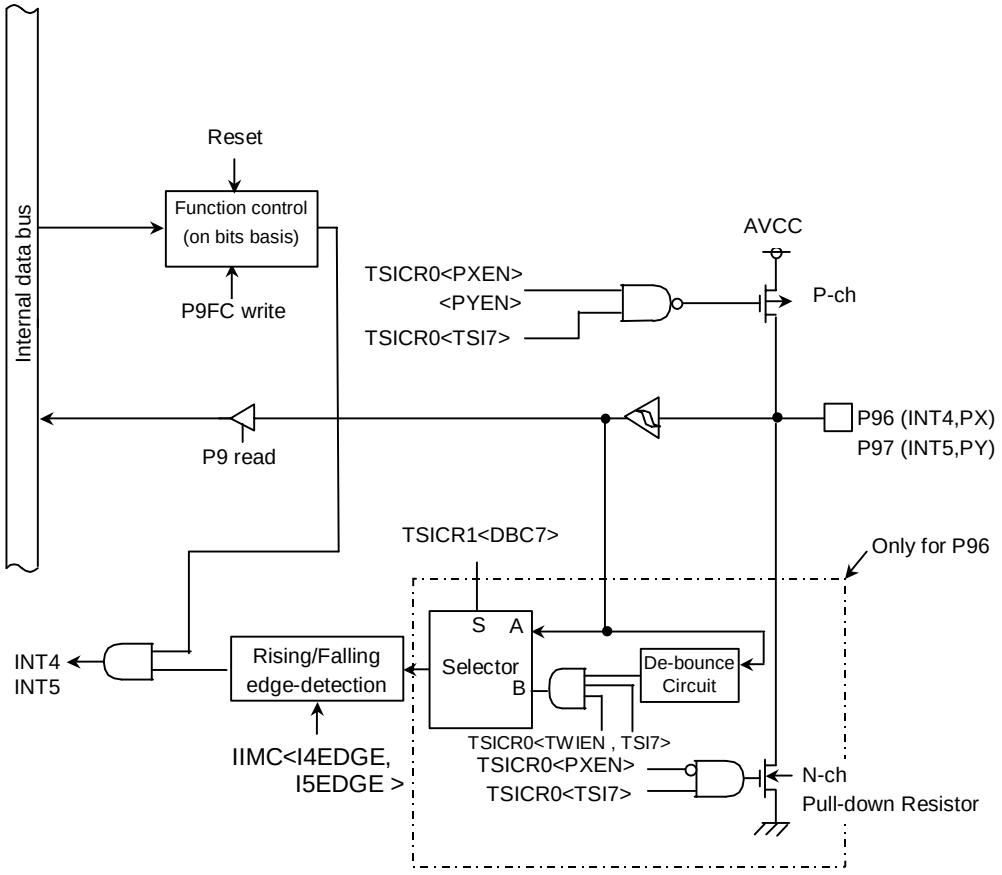


Figure 3.5.14 Port96,97

Port 9 Register

P9 (0024H)

	7	6	5	4	3	2	1	0
bit Symbol	P97	P96	P95	P94	P93	P92	P91	P90
Read/Write	R		R/W					
After reset	Input mode							
	-		0	1				

Port 9 Control Register

P9CR (0026H)

	7	6	5	4	3	2	1	0
bit Symbol			P95C	P94C	P93C	P92C	P91C	P90C
Read/Write			W					
After reset			0					
Function								

Port 9 Function Register

P9FC (0027H)

	7	6	5	4	3	2	1	0
bit Symbol	P97F	P96F	P95F	P94F	P93F	P92F	P91F	P90F
Read/Write	W							
After reset	0							
Function	0: Input port 1: INT5	0: Input port 1: INT4						

P92 setting

<P92C> / <P92F>	0	1
0	input port SCLK0,/CTS0 in	output port
1	I2SWS out	SCLK0 out

P91 setting

<P91C> / <P91F>	0	1
0	input port RXD0 in	output port
1	I2SDO out	(reserved)

P90 setting

<P90C> / <P90F>	0	1
0	input port	output port
1	I2SCKO out	TXD0 out

P95 setting

<P95C> / <P95F>	0	1
0	output port	
1	LGOE2 out	-

P94 setting

<P94C> / <P94F>	0	1
0	input port	output port
1	LGOE1 out	-

P93 setting

<P93C> / <P93F>	0	1
0	input port	output port
1	LGOE0 out	-

Port 9 Function Register 2

P9FC2 (0025H)

	7	6	5	4	3	2	1	0
bit Symbol								P90F2
Read/Write								W
After reset								0
Function								0:CMOS 1:OpenDrain

Port 9 Drive Register

P9DR (0089H)

	7	6	5	4	3	2	1	0
bit Symbol	P97D	P96D	P95D	P94D	P93D	P92D	P91D	P90D
Read/Write	R/W							
After reset	1							
Function	Output/Input buffer drive-register for standby-mode							

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

Figure 3.5.14 Register for Port 9

3.5.10 Port A (PA0 to PA7)

Port A0 to A7 are 8-bit 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.

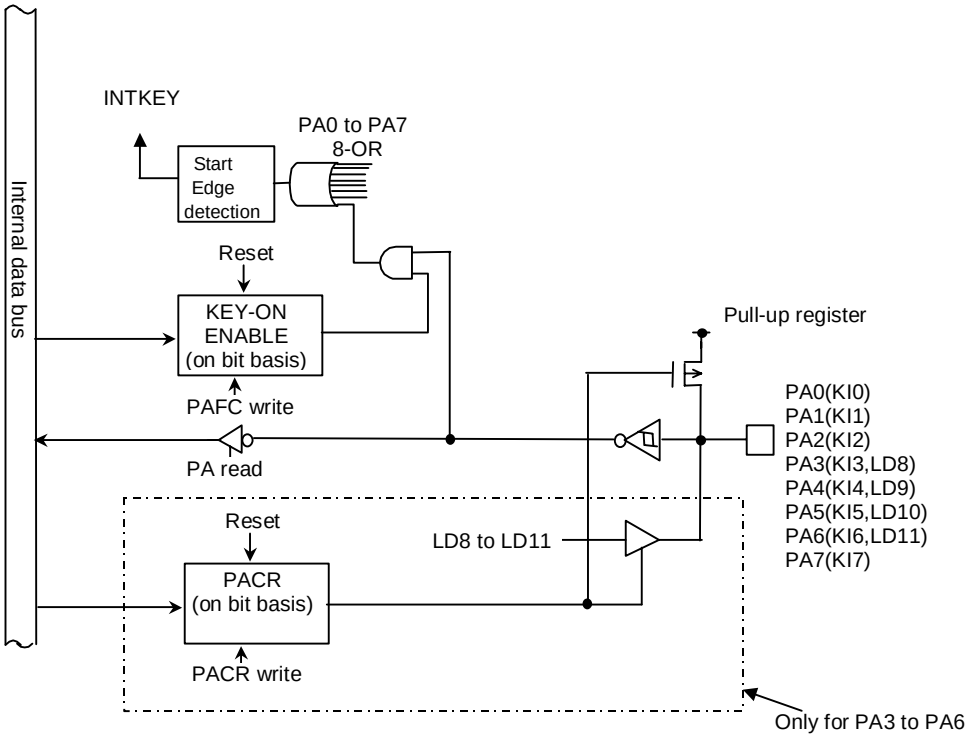


Figure 3.5.15 Port A

When PAFC=”1”, if either of input of KI0-KI7 pins falls down, INTKEY interrupt is generated. INTKEY interrupt can be used 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	Input Mode							

Port A Function Register

PAFC (002BH)

	7	6	5	4	3	2	1	0
bit Symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	W							
After reset	0							
	0: KEY IN disable 1: KEY IN enable							

Port A Control Register

PACR (002AH)

	7	6	5	4	3	2	1	0
bit Symbol		PA6C	PA5C	PA4C	PA3C			
Read/Write		W						
After reset		0						
		0: input port or KEY IN 1: LD11-LD8 output						

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							
Function	Output/Input buffer drive-register for standby-mode							

Note 1: Read-Modify-Write is prohibited for the registers PACR and PAFC .

Figure 3.5.16 Port A register

3.5.11 Port C (PC0 to PC3,PC6 to PC7)

PC0 to PC3,PC6 and PC7 are 6-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets Port C to an input port.

In addition to functioning as a general-purpose I/O port, Port C can also function as output pin for timers (TA1OUT, TA3OUT and TB0OUT0), input pin for external interruption (INT0 to INT3) , output pin for memory(/CSZF) , output pin for Key(KO8) and output pin for LCD Driver(LDIV,LCP1). 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,TA1OUT)

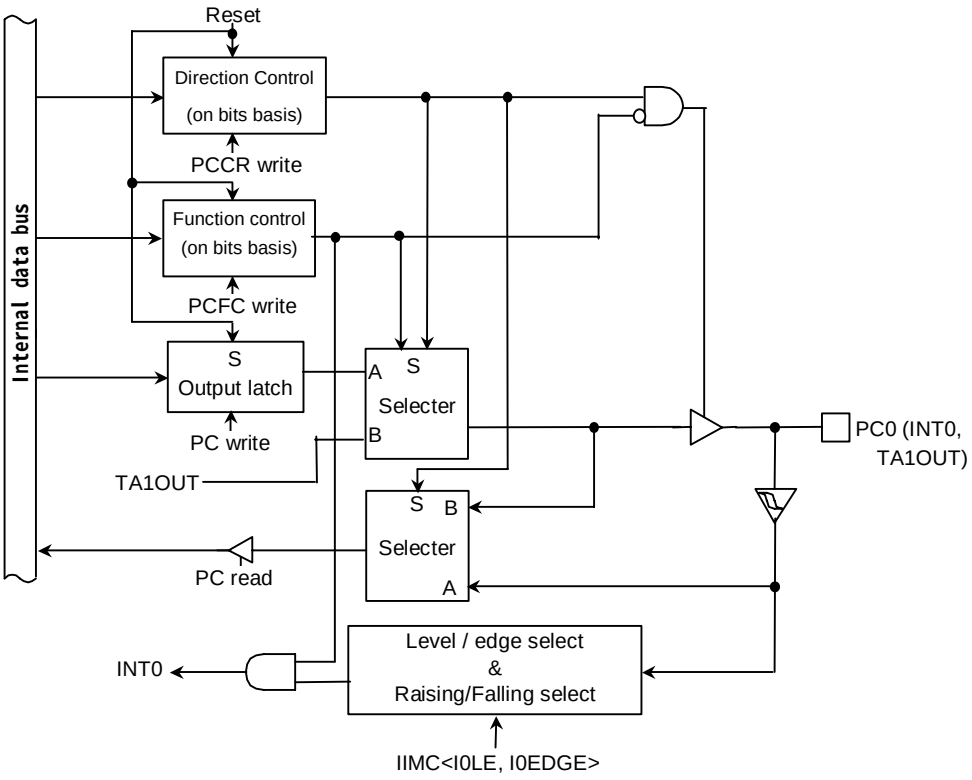


Figure 3.5.17 Port C0

(2) PC1(INT1,TA3OUT), PC2(INT2,TB0OUT0) , PC3(INT3,TB0OUT1)

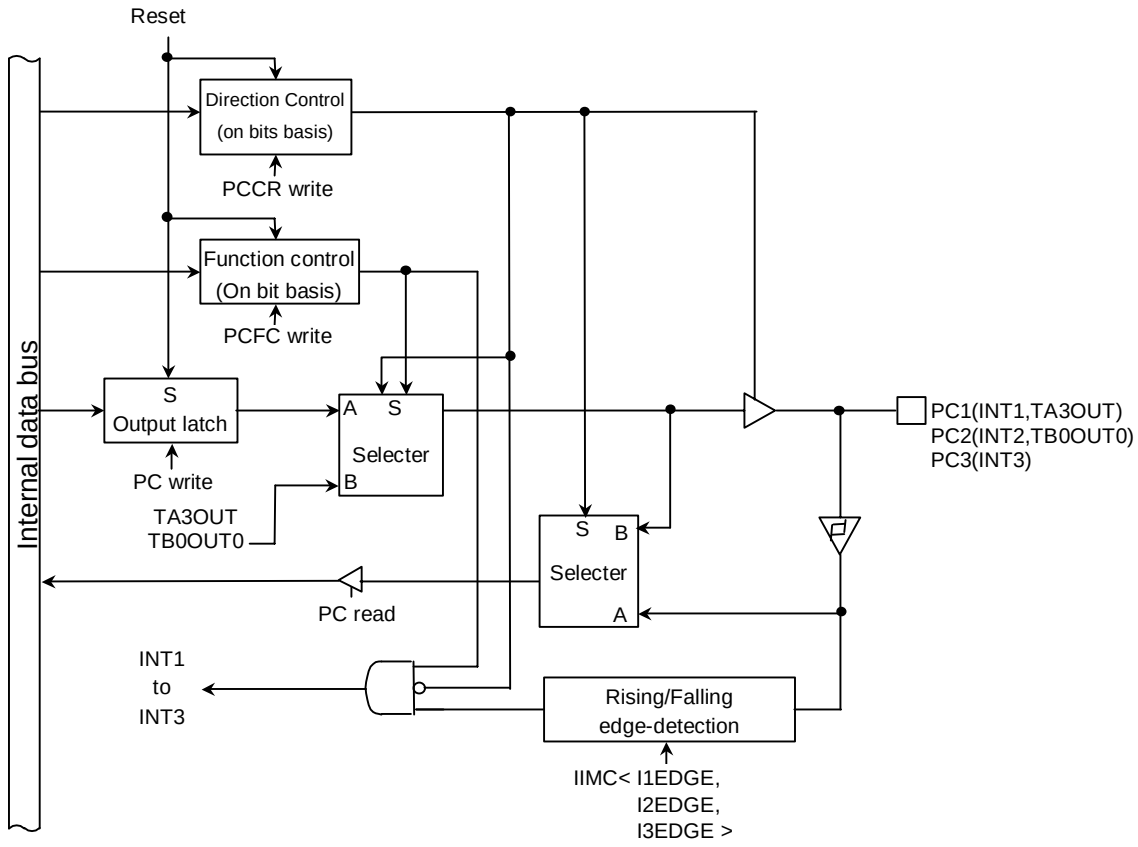


Figure 3.5.18 Port C1, C2, C3

(3) PC6 (KO8,LDIV)

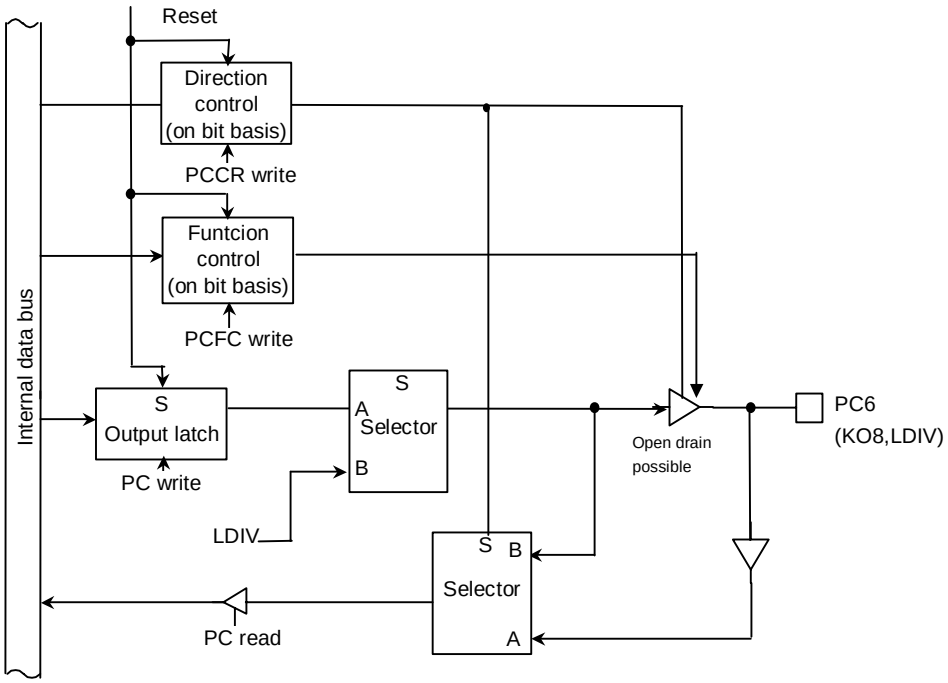


Figure 3.5.19 Port C6

(4) PC7 (/CSZF,LCP1)

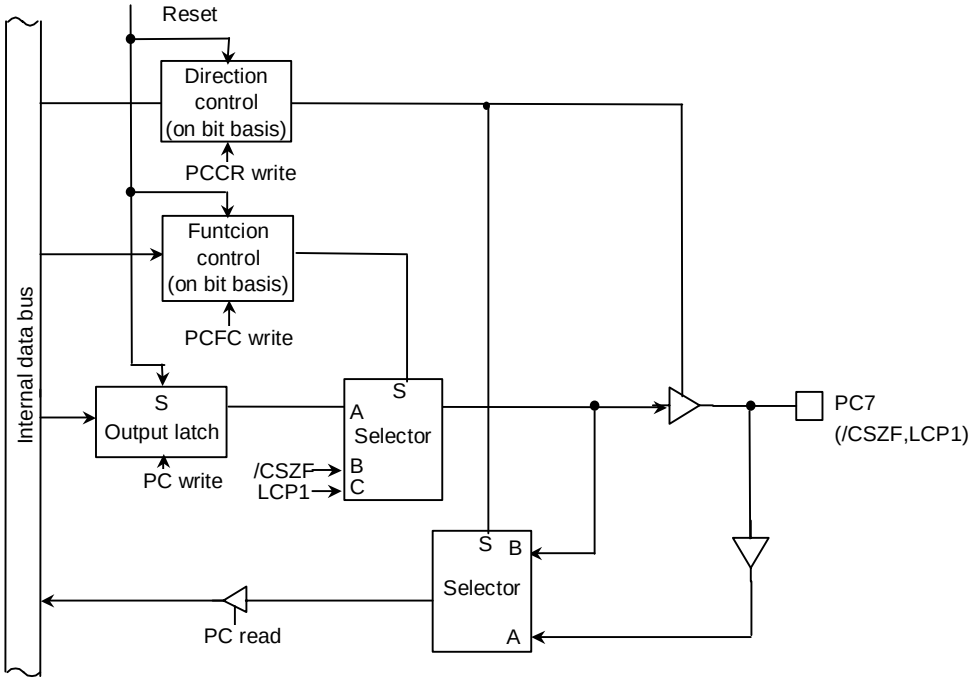


Figure 3.5.19 Port C7

Port C Register

PC (0030H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7	PC6			PC3	PC2	PC1	PC0
Read/Write	R/W				R/W			
After Reset	Input Mode				Input Mode			
	1				1			

Port C Control Register

PCCR (0032H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7C	PC6C			PC3C	PC2C	PC1C	PC0C
Read/Write	W				W			
After Reset	0				0			
	0: IN 1: OUT				0: IN 1: OUT			

Port C Function Register

PCFC (0033H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7F	PC6F			PC3F	PC2F	PC1F	PC0F
Read/Write	W				W			
After Reset	0				0			
Function								

PC2 setting

<PC2C> / <PC2F>	0	1
0	input port	output port
1	INT2	TB0OUT0

PC1 setting

<PC1C> / <PC1F>	0	1
0	input port	output port
1	INT1	TA3OUT

PC0 setting

<PC0C> / <PC0F>	0	1
0	input port	output port
1	INT0	TA1OUT

PC7 setting

<PC7C> / <PC7F>	0	1
0	input port	output port
1	/CSZF out	LCP1 out

PC6 setting

<PC6C> / <PC6F>	0	1
0	input port	output port
1	KO8(open drain)	LDIV out

PC3 setting

<PC3C> / <PC3F>	0	1
0	input port	output port
1	INT3	(reserved)

Port C Drive Register

PCDR (008CH)

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

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

Figure 3.5.20 Register for Port C

3.5.12 Port F (PF0 to PF2,PF7)

Port F0 to F2 are 3-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets PF0 to PF2 to be an 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 PF2 can also function as the I/O for serial channels 0 and 1. A pin can be enabled for I/O by writing a “1” to the corresponding bit of the Port F Function Register (PFFC).

PortF7 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 (TXD0,TXD1), F1(RXD0,RXD1), F2(SCLK0,/CTS0,SCLK1,/CTS1)

PortF0 to F2 are general-purpose I/O port. They are also used either SIO0 or SIO1. Each pins are below.

	SIO-mode (SIO0 module)	UART,IrDA-mode (SIO0 module)	SIO-mode (SIO1 module)	UART -mode (SIO1 module)
PF0	TXD0 (data output)	TXD0 (data output)	TXD1 (data output)	TXD1 (data output)
PF1	RXD0 (data input)	RXD0 (data input)	RXD1 (data input)	RXD1 (data input)
PF2	SCLK0 (clock input or output)	/CTS0 (Clear to send)	SCLK1 (clock input or output)	/CTS1 (Clear to send)

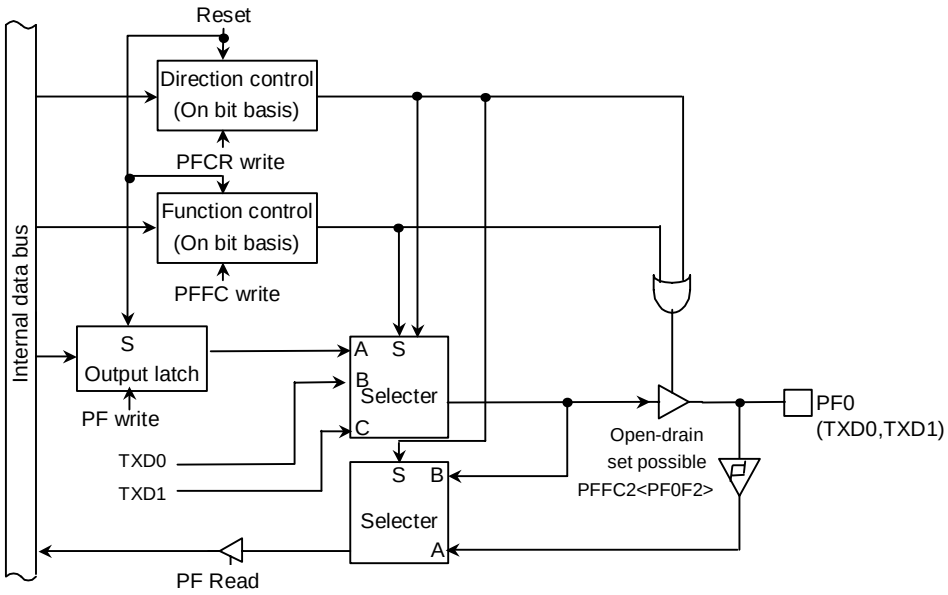


Figure 3.5.21 Port F0

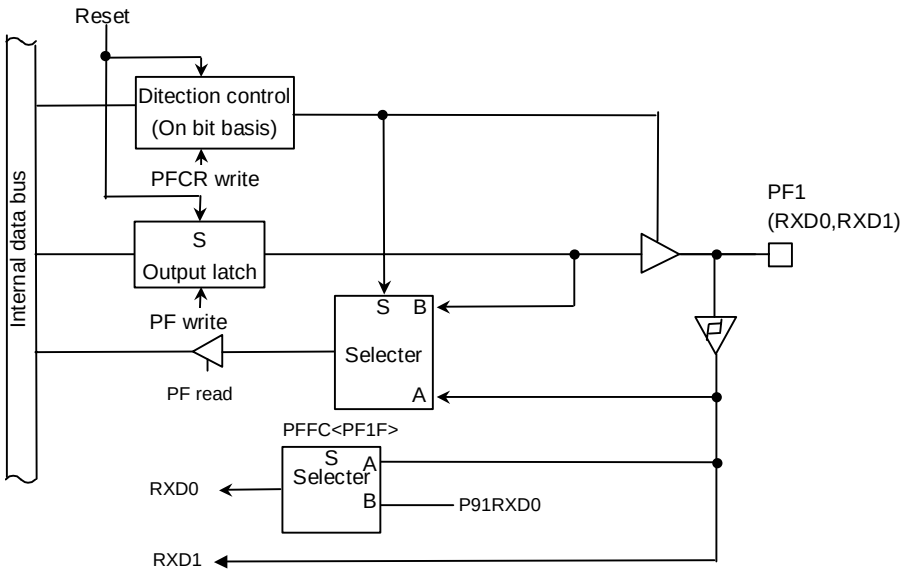


Figure 3.5.22 Port F1

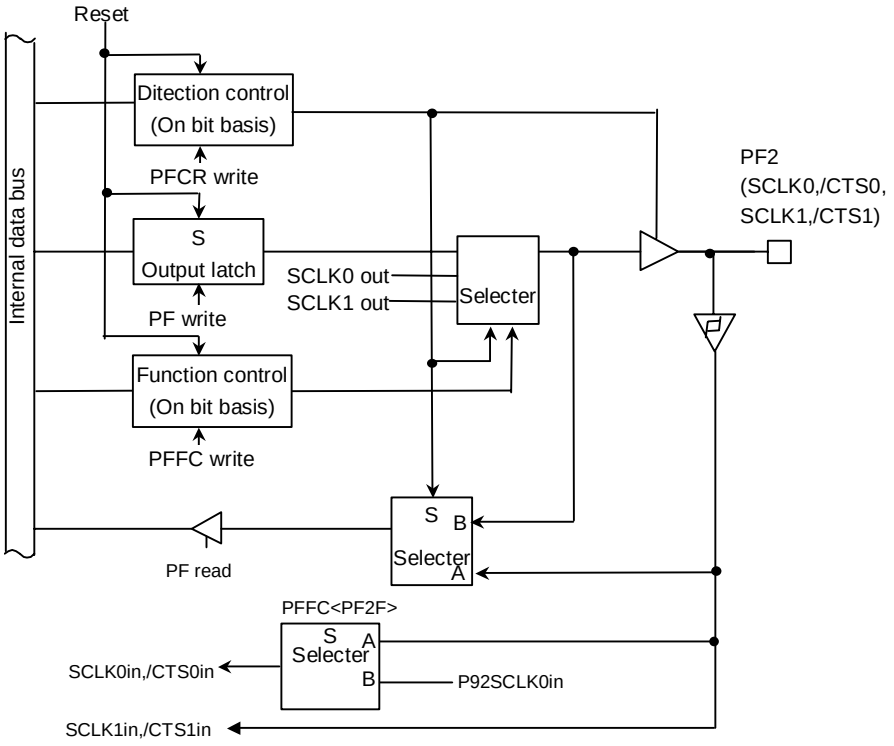


Figure 3.5.23 Port F2

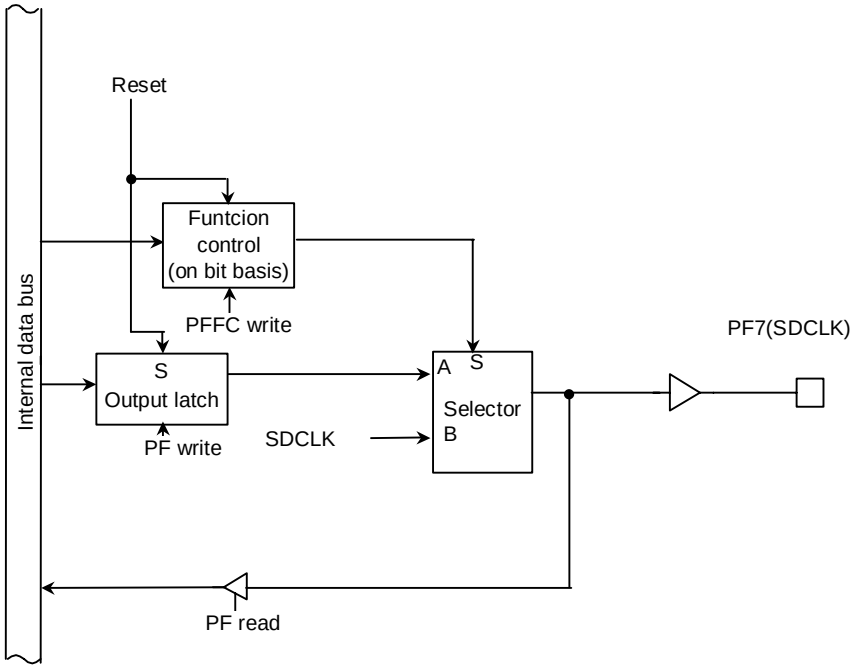


Figure 3.5.26 Port F7

Port F Register

PF (003CH)

	7	6	5	4	3	2	1	0
bit Symbol	PF7					PF2	PF1	PF0
Read/Write	R/W					R/W		
After Reset	1					1		

Port F Control Register

PFCR (003EH)

	7	6	5	4	3	2	1	0
bit Symbol						PF2C	PF1C	PF0C
Read/Write						W		
After Reset						0		

Port F Functon Register

PFFC (003FH)

	7	6	5	4	3	2	1	0
bit Symbol	PF7F					PF2F	PF1F	PF0F
Read/Write	W					W		
After Reset	1					0		
Function							RXD0 pin Selection 0: PortF1 1: Port91	

PF2 setting

<PF2C> / <PF2F>	0	1
0	input port or SCLK1,/CTS1 in or SCLK0,/CTS0 in From PF2-pin at <PF2>=0 From P92-pin at <PF2>=1	output port
1	SCLK1 out	SCLK0 out

PF1 setting

<PF1C> / <PF1F>	0	1
0	input port or RXD0/RXD1 in	output port
1		

PF0 setting

<PF0C> / <PF0F>	0	1
0	input port	output port
1	TXD1 out	TXD0 out

PF7 setting

<PF7F>	
0	output port
1	SDCLK out

Port F Functon Register2

PFFC2 (003DH)

	7	6	5	4	3	2	1	0
bit Symbol								PF0F2
Read/Write								W
After Reset								0
Function								Output buffer 0: CMOS 1: OpenDrain

Port F Drive Register

PFDR (008FH)

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

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

Figure 3.5.24 Register for Port F

3.5.13 Port G (PG0 to PG3)

PG0 to PG3 are 4-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 A/D converter.

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

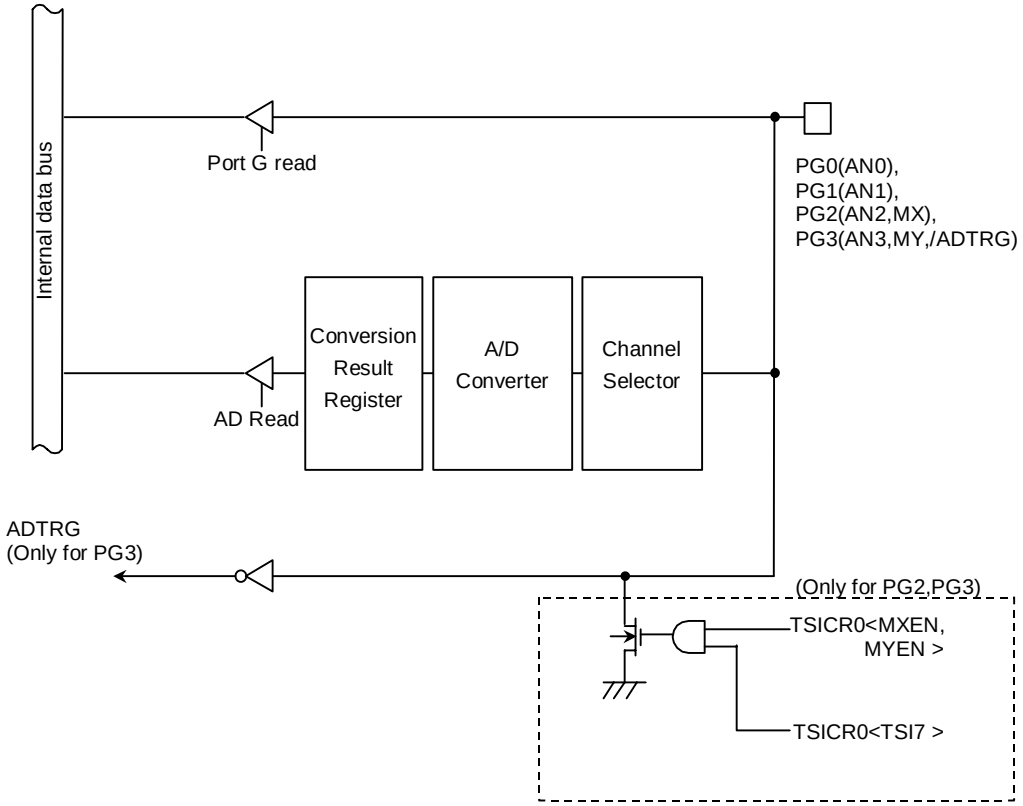


Figure 3.5.25 Port G

Port G Register

PG (0040H)

	7	6	5	4	3	2	1	0
Bit Symbol					PG3	PG2	PG1	PG0
Read/Write					R			
After reset					Input mode			

Note: The input channel selection of A/D Converter and the permission of ADTRG input are set by A/D Converter mode register ADMOD1.

Port G Drive Register

PGDR (0090H)

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

Figure 3.5.26 Register for Port G

3.5.14 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 (/SDRAS, /SDCAS, /SDWE, SDLLDQM, SDLUDQM, SDULDQM, SDUUDQM and SDCKE) ,SRAM(/SRWR, /SRLLB and /SRLUB) and NANDFlash(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.

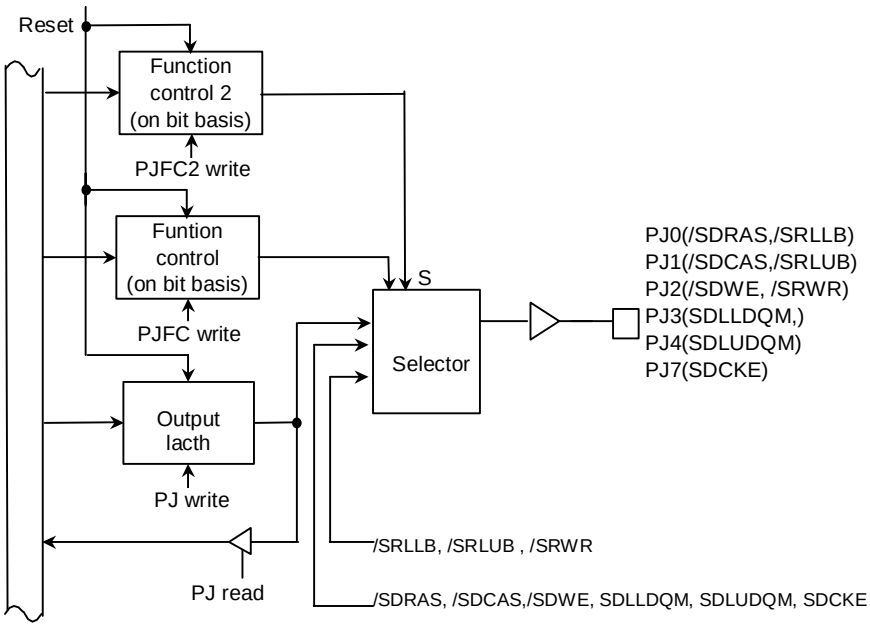


Figure3.5.27 Port J0,J1,J2,J3,J4 and J7

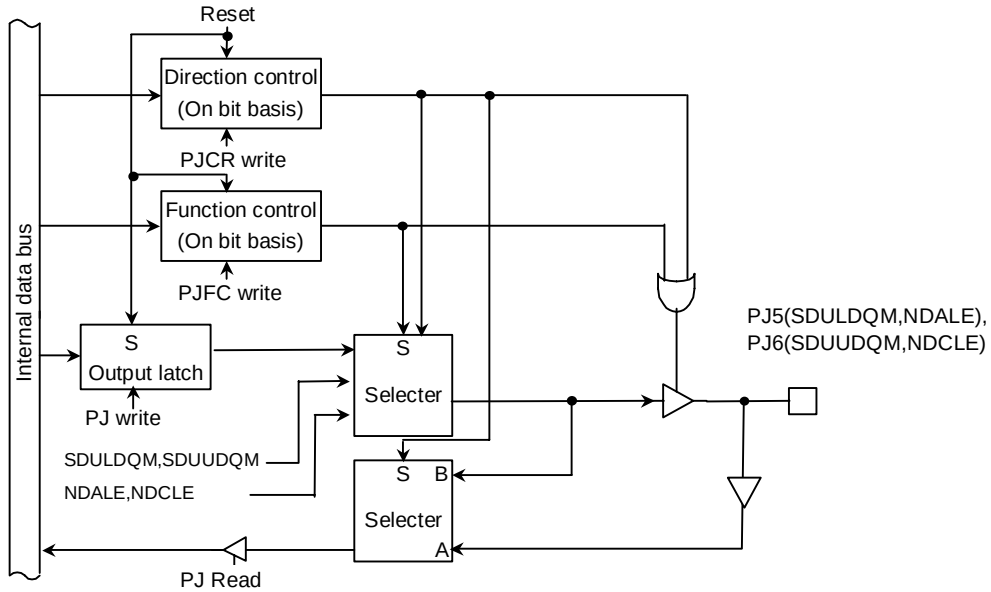


Figure3.5.28 Port J5 and J6

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							

Port J control register

PJCR (004EH)

	7	6	5	4	3	2	1	0
bit Symbol		PJ6C	PJ5C					
Read/Write		W						
After Reset		0						
Function		0: IN , 1: OUT						

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							
Function	0: PORT 1: SDCKE	0: PORT 1: NDCLE at <PJ6>=0, SDUUDQM at <PJ6>=1	0: PORT 1: NDALE at <PJ5>=0, SDULDQM at <PJ5>=1	0: PORT 1: SDLUDQM	0: PORT 1: SDLLDQM	0: PORT 1: /SDWE, /SRWR	0: PORT 1: /SDCAS, /SRLUB	0: PORT 1: /SDRAS, /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							
Function	Output/Input buffer drive-register for standby-mode							

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

Figure3.5.28 Register for PortJ

3.5.15 Port K (PK0 to PK3)

PortK are 4-bit output ports. Resetting sets the output latch PK to “0”, and PK0 to PK3 pins output “0”.

In addition to functioning as output ports, PortK also function as output pins for LCD controller (LCP0, LLP, LFR and LBCD).

Above setting is used the function register PKFC.

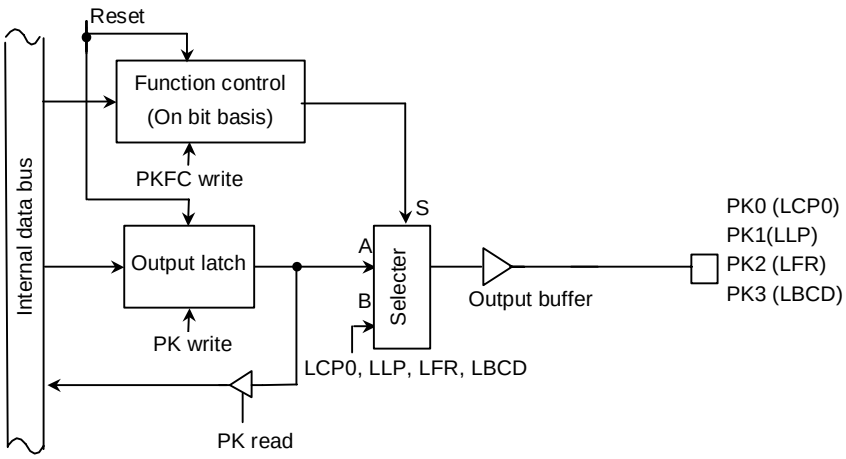


Figure3.5.29 Port K

Port K register

PK (0050H)	7	6	5	4	3	2	1	0
bit Symbol					PK3	PK2	PK1	PK0
Read/Write					R/W			
After Reset					0			

Port K function register

PKFC (0053H)	7	6	5	4	3	2	1	0
bit Symbol					PK3F	PK2F	PK1F	PK0F
Read/Write					W			
After Reset					0			
Function					0: PORT 1: LBCD	0: PORT 1: LFR	0: PORT 1: LLP	0: PORT 1: LCP0

Port K Drive register

PKDR (0094H)	7	6	5	4	3	2	1	0
bit Symbol					PK3D	PK2D	PK1D	PK0D
Read/Write					R/W			
After reset					1			
Function					Output/Input buffer drive-register for standby-mode			

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

Figure3.5.31 Register for Port K

3.5.16 Port L (PL0 to PL7)

PL0 to PL3 are 4-bit output ports. Resetting sets the output latch PL to “0”, and PL0 to PL3 pins output “0”.

PL4 to PL7 are 4-bit general-purpose I/O ports. Each bit can be set individually for input or output using the control register PLCR. Resetting, the control register PLCR to “0” and sets PL4 to PL7 to input ports. In addition to functioning as a general-purpose I/O port, Port L can also function as a data bus for LCD controller (LD0 to LD7). Above setting is used the function register PLFC.

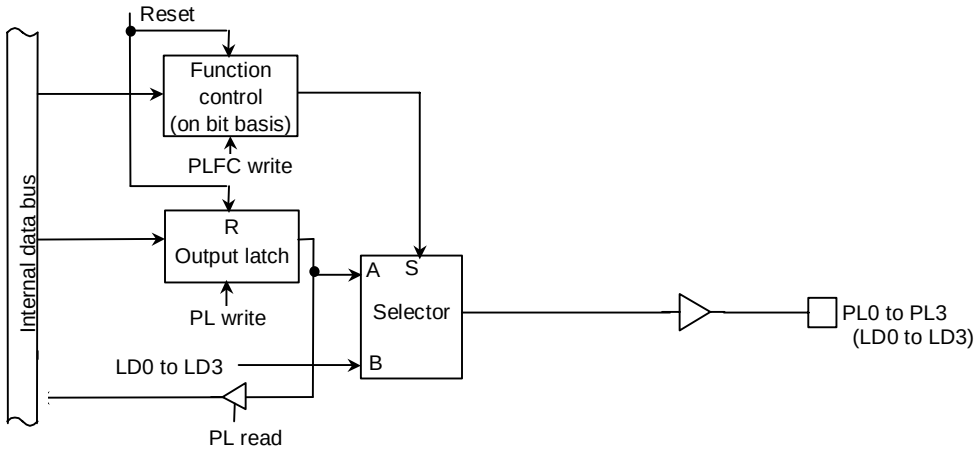


Figure3.5.32 Register for Port L0 to L3

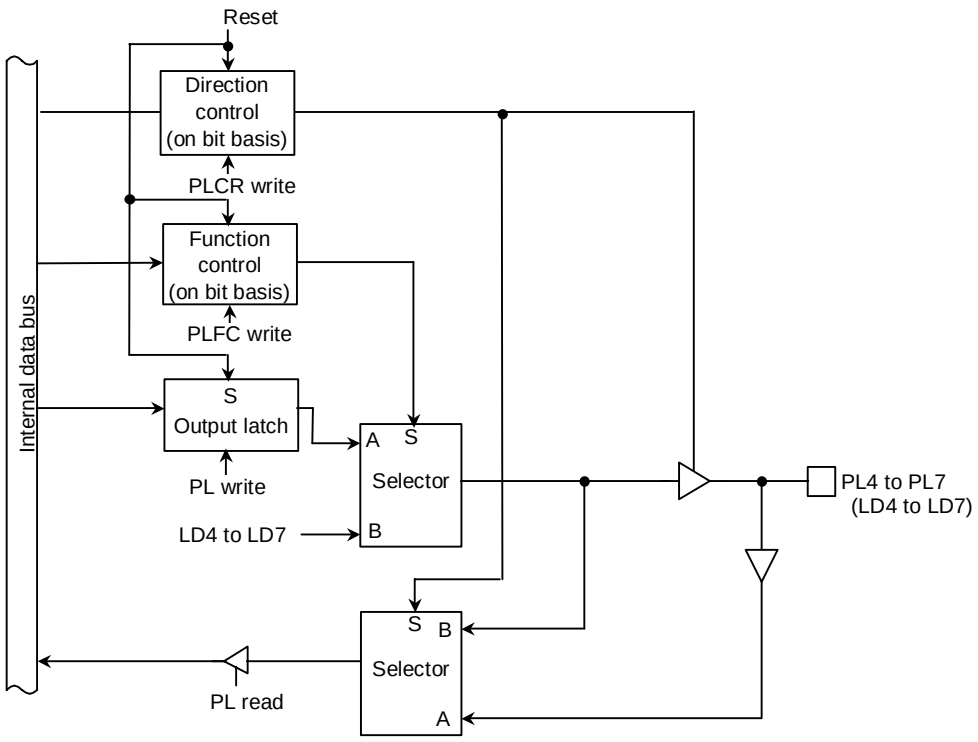


Figure3.5.33 Register for Port L4 to L7

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							

Port L Control Register

PLCR (0056H)

	7	6	5	4	3	2	1	0
bit Symbol	PL7C	PL6C	PL5C	PL4C				
Read/Write	W							
After Reset	0							
Function	0: IN 1: OUT							

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							
Function	0: Port 1: data bus for LCDC (LD7 to LD0)							

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							
Function	Output/Input buffer drive-register for standby-mode							

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

Figure3.5.33 Port L register

3.5.17 Port M(PM1 to PM2)

PM1 to PM2 are 2-bit output ports. Resetting sets the output latch PM to “1”, and PM1 to PM2 pins output “1”.

In addition to functioning as ports, PortM also function as output pins for RTC alarm (/ALARM) , output pin for melody/alarm generator (MLDALM, /MLDALM).

Above setting is used the function register PMFC.

Only PM2 has two output function which /ALARM and /MLDALM. This selection is used PM<PM2>.

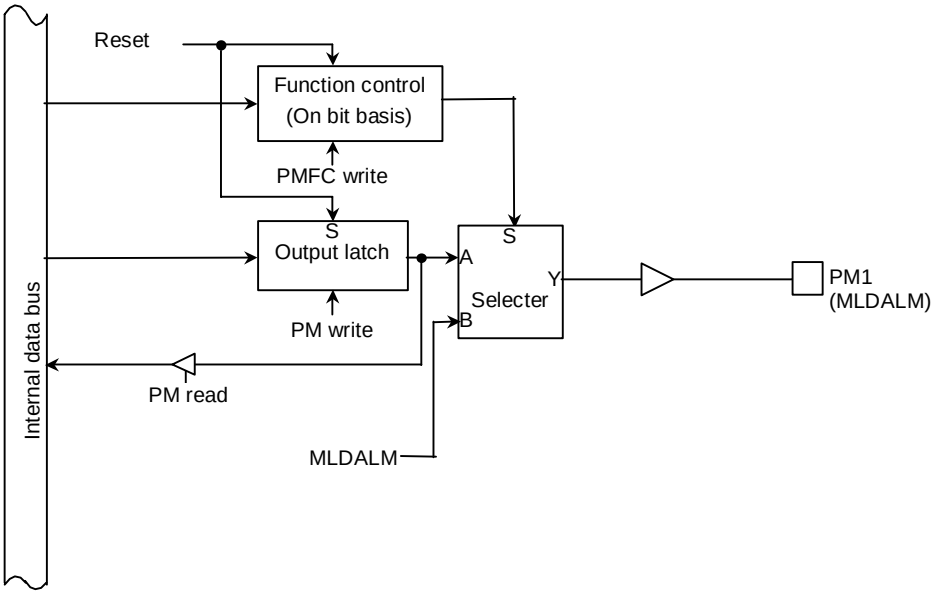


Figure3.5.35 Port M1

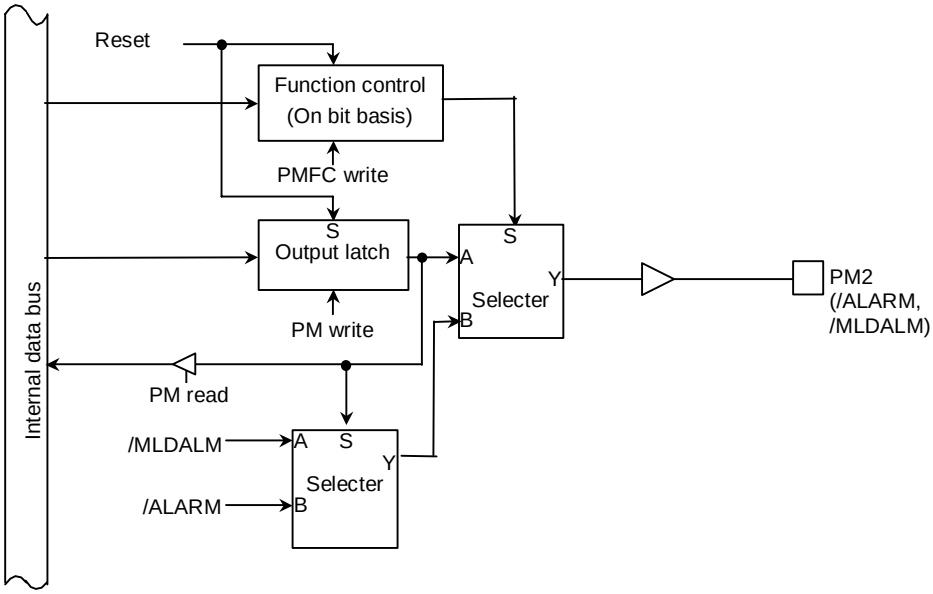


Figure3.5.36 Port M2

Port M register

PM (0058H)

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

Port M function register

PMFC (005BH)

	7	6	5	4	3	2	1	0
bit Symbol						PM2F	PM1F	
Read/Write						W		
After Reset						0		
Function						0: PORT 1: /ALARM at <PM2>="1" 1: /MLDALM at<PM2>="0"	0: port 1: MLDALM out	

Port M Drive register

PMDR (0096H)

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

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

Figure3.5.31 Register for Port M

3.6 Memory Controller

3.6.1 Functions

TMP92CH21 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 : Only either CS1 or CS2-blocks is supported.
* Page-ROM : Only CS2-blocks is supported.
* NAND-Flash : CS0 is recommended for NAND-Flash(ND0/1FDTR, 001D00H-001EFFH), RAM built-in LCD-driver(001FE0H-001FEFH)

(About NAND Flash area, refer to 3.6.6(2) .)

(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, 16bit or 32bit 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. Read cycle and write cycle can specify the number of waits individually. The number of waits is controlled in 6 mode mentioned below.

0 wait, 1wait, 2 wait, 3 wait, 4wait N wait(controls with /WAIT pin)

3.6.2 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 Table3.6(1),(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 access Page-ROM.
- Internal-Boot ROM control register: BROMCR
  Sets to access Boot-ROM.

Table 3.6(1) Control Register

		7	6	5	4	3	2	1	0
B0CSL (0140H)	Bit symbol	-	B0WW2	B0WW1	B0WW0	-	B0WR2	B0WR1	B0WR0
	Read/Write	W							
	After Reset	-	0	1	0	-	0	1	0
B0CSH (0141H)	Bit Symbol	B0E	-	-	B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
	Read/Write	W							
	After Reset	0	0(Note)	0 (Note)	0 (Note)	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	-	B1WW2	B1WW1	B1WW0	-	B1WR2	B1WR1	B1WR0
	Read/Write	W							
	After Reset	-	0	1	0	-	0	1	0
B1CSH (0145H)	Bit Symbol	B1E	-	-	B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
	Read/Write	W							
	After Reset	0	0 (Note)	0 (Note)	0 (Note)	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	-	B2WW2	B2WW1	B2WW0	-	B2WR2	B2WR1	B2WR0
	Read/Write	W							
	After Reset	-	0	1	0	-	0	1	0
B2CSH (0149H)	Bit Symbol	B2E	B2M	-	B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
	Read/Write	W							
	After Reset	1	0	0 (Note)	0 (Note)	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	-	B3WW2	B3WW1	B3WW0	-	B3WR2	B3WR1	B3WR0
	Read/Write	W							
	After Reset	-	0	1	0	-	0	1	0
B3CSH (014DH)	Bit Symbol	B3E	-	-	B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
	Read/Write	W							
	After Reset	0	0 (Note)	0 (Note)	0 (Note)	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

Note: Fix to “0”.

Table 3.6(2) Control Register

		7	6	5	4	3	2	1	0
BEXCSH (0159H)	Bit Symbol	-	-	-	-	BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
	Read/Write	W							
	After Reset	-	-	-	-	0	0	0	0
BEXCSL (0158H)	Bit Symbol	-	BEXWW2	BEXWW1	BEXWW0	-	BEXWR2	BEXWR1	BEXWR0
	Read/Write	W							
	After Reset	-	0	1	0	-	0	1	0
PMEMCR (0166H)	Bit Symbol	-	-	-	OPGE	OPWR1	OPWR0	PR1	PR0
	Read/Write				R/W				
	After Reset	-	-	-	0	0	0	1	0
BROMCR (0167H)	Bit Symbol	-	-	-	-	-	-	ROMLESS	VACE
	Read/Write							R/W	
	After Reset	-	-	-				0/1	1/0

(2) Operation after reset release

The start data bus width is determined depending on state of AM1/AM0 pins just after reset release. Then, the external memory is accessed as follows

AM1	AM0	Start mode
0	0	Don't use this setting
0	1	Start with 16-bit data bus (note)
1	0	Start with 32-bit data bus (note)
1	1	Start with BOOT(32-bit internal-MROM)

(note) A memory to be used as starting after reset is either NOR-Flash, Masked-ROM.

NAND-Flash and SDRAM can't be used.

AM1/AM0 pins are valid only just after reset release. In the other cases, the data bus width is set to the value set to BnBUS bit of the control register.

By reset, only control register (B2CSH/B2CSL) of the block address area 2 is automatically effective (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 reset release, the block address areas are specified by the memory start address register(MSARn) and the memory address mask register(MAMRn). Then the control register (BnCS) is set.

Set the enable bit(BnE) of the control register to "1" to enable the setting.

3.6.3 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 area is specified by two registers.

The memory start address register(MSARn) sets the start address of the block address areas. The memory controller compares between the register value and the address every bus cycles. The address bit which is masked by the memory address mask register(MAMRn) is not compared by the memory controller. The block address area size is determined by setting the memory address mask register. The set value in the register is compared with the block address area on the bus. If the compared result is a match, the memory controller sets the chip select signal(CSn) to "Low".

(i) Setting memory start address register

The MS23 to 16 bits of the memory start address register respectively correspond with addresses A23 to A16. The lower start address A15 to 0 are always set to address 0000H.

Therefore the start address of the block address area are set to addresses 000000H to FF0000H every 64K bytes.

(ii) Setting memory address mask registers

The memory address mask register sets whether an address bit is compared or not. Set the register to "0" to compare, or to "1" not to compare.

The address bit to be set is depended on the block address area.

Block address area 0 : A20 to A8

Block address area 1 : A21 to A8

Block address area 2 to 3 : A22 to A15

The above-mentioned bits are always compared. The block address area size is determined by the compared result.

The size to be set depending on the block address area is as follows.

Size (bytes) / CS area	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	○	○	○	○	○		
CS1	○	○		○	○	○	○	○	○	○	
CS2 to 3			○	○	○	○	○	○	○	○	○

Note: After reset release, 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

(iii) Example of register setting

To set the block address area 64Kbytes from address 110000H, set the register as follows.

MSAR1 Register

bit	7	6	5	4	3	2	1	0
bit Symbol	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
Specified value	0	0	0	1	0	0	0	1

M1S23 to 16 bits of the memory start address register MSAR1 correspond with address A23 to A16.

A15 to 0 are set to “0”. Therefore setting MSAR1 to the above-mentioned value specifies the start address of the block address area to address 110000H.

The start address is set as it is in the other block address areas.

MAMR1 Register

bit	7	6	5	4	3	2	1	0
bit Symbol	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	M1V15 to 9	M1V8
Specified value	0	0	0	0	0	0	1	1

M1V21 to 16 and M1V8 bits of the memory address mask register MAMR1 set whether address A21 to 16 and A8 are compared or not. Set the register to “0” to compare, or to “1” not to compare. M1V15-9 bits set whether address A15 to A9 are compared or not with 1 bit. A23 and A22 are always compared.

Setting the above-mentioned compares A23 to A9 with the values set as the start addresses. Therefore 512k bytes of addresses 110000H to 1101FFH are set as the block address area 1, and compared with the addresses on the bus. If the compared result is a match, the chip select signal CS1 is set to “LOW”.

The other block address area sizes are specified like this.

A23 and A22 are always compared in the block address area 0. Whether A20 to A8 are compared or not is set to register.

Similarly, A23 is always compared in block address areas 2 to 5. Whether A22 to A15 are compared or not is set to register.

Note: 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

Note: 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 the BnOM1 to 0 bit of the control register (BnCSH) specifies the memory type to be connected with the block address areas. The interface signal is output according to the set memory as follows

BnOM1,BnOM0 bit (BnCSH Regsiter)

BnOM1	BnOM0	Function
0	0	SRAM/ROM (Default)
0	1	(Reserved)
1	0	-
1	1	SDRAM

(Note1) SDRAM should be set to block either 1 or 2.

(Note2) Set “00” for NAND-Flash,RAM-built-in LCDD

(3) Data Bus Width Specification

The data bus width is set for every block address area. The bus size is set by the BnBUS1 and BnBUS0 bits of the control register (BnCSH) as follows.

BnBUS bit (BnCSH Regsiter)

BnBUS1	BnBUS0	Function
0	0	8-bit bus mode (Default)
0	1	16-bit bus mode
1	0	32-bit bus mode
1	1	Don’t use this setting

(Note1) SDRAM should be set to either “01”(16-bit bus) or “10”(32-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.

(Note)

Since there is a possibility of abnormal writing/reading of the data
if bus width put the different memories in consecutive address, do not execute
a access which is placed on both memories with one command.

Operand data size (bit)	Operand start address	Memory data size (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 a read, data input to the bus ignored. At write, the bus is at high impedance and the write strobe signal remains non to active.

(4) Wait control

The external bus cycle completes a wait of two states at least (100ns at f_{SYS}=20MHz). Setting the <BnWW2 to 0> and <BnWR2 to 0> of BnCSL specifies the number of waits in the read cycle and the write cycle. BnWW is set with the same method as BnWR.

BnWW/BnWR bit (BnCSL Regsiter)

BnWW2 BnWR2	BnWW1 BnWR1	BnWW0 BnWR0	Function
0	0	1	2states(0 wait) access fixed mode
0	1	0	3states(1 wait) access fixed mode(Default)
1	0	1	4states(2 wait) access fixed mode
1	1	0	5states(3 wait) access fixed mode
1	1	1	6states(4 wait) access fixed mode
0	1	1	/WAIT pin input mode
Others			(Reserved)

(Note1) For SDRAM, above setting is no meaning. So, refer 3.16 SDRAM Controller.

(Note2) For NAND-Flash, this setting is no-meaning.

For RAM-built-in LCDD, this setting is effective.

(i) Waits number fixed mode

The bus cycle is completed with the set states. The number of states is selected from 2 states(0WAIT) to 6states(4WAIT).

(ii) /WAIT pin input mode

This mode samples the /WAIT input pins. It continuously samples the /WAIT pin state and inserts a wait if the pin is active. The bus cycle is minimum 2 states. The bus cycle is completed when the wait signal is non-active(“High” level) at 2 states. The bus cycle extends if the wait signal is active at 2 states and more.

(5) Recovery(Data-hold) cycle control

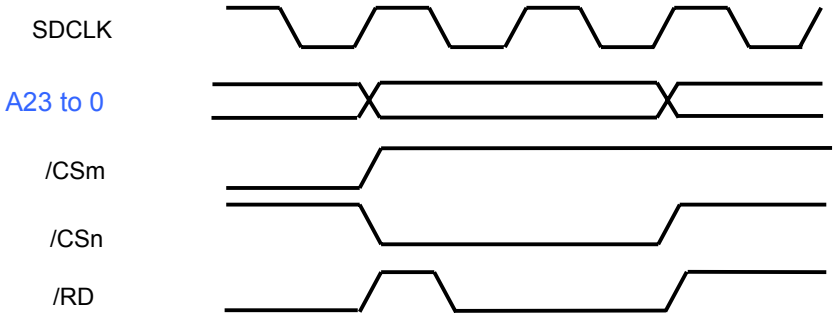
Some memory have an A.C specification about data-hold time from /CE or /OE for read cycle and a data-confliction problem occurs. To avoid this problem, 1-dummy cycle can be inserted after CSm-block access cycle by setting “1” to BmCSH<BmREC>-register.

This 1-dummy cycle is inserted when the next cycle is for another CS-block.

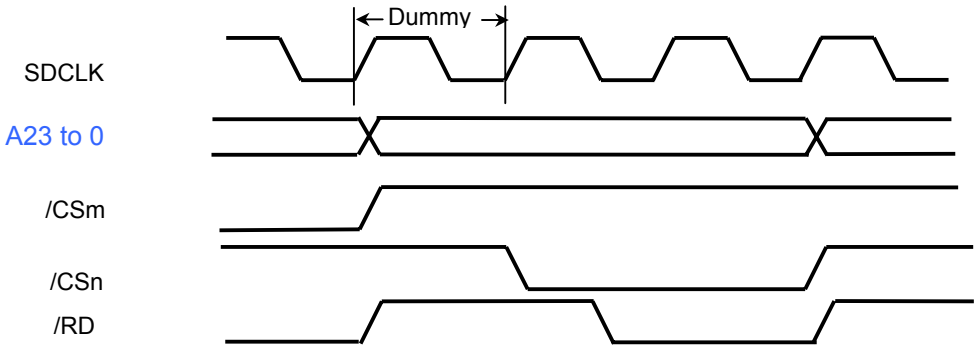
BnREC bit (BnCSH register)

0	No dummy cycle is inserted (default).
1	Dummy cycle is inserted.

- When not inserting a dummy cycle(0 wait)

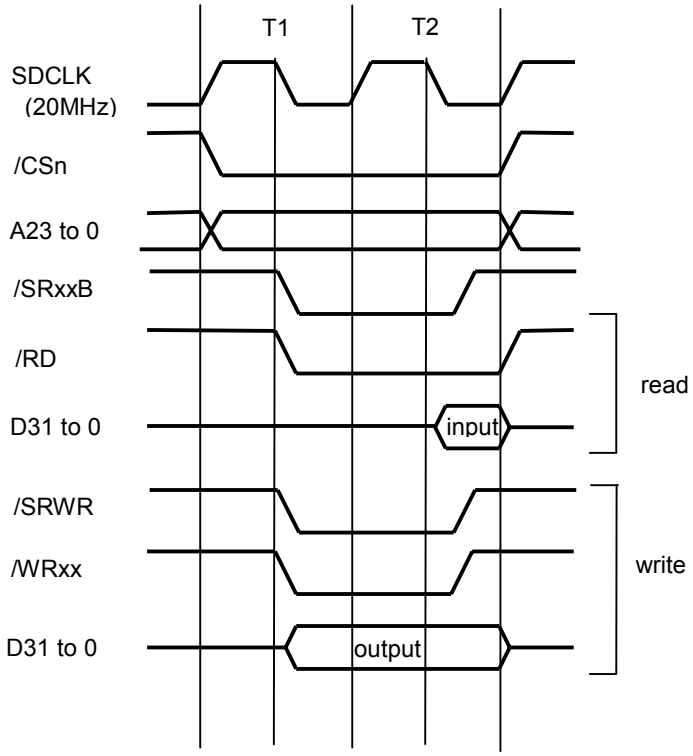


- When inserting a dummy cycle(0 wait)

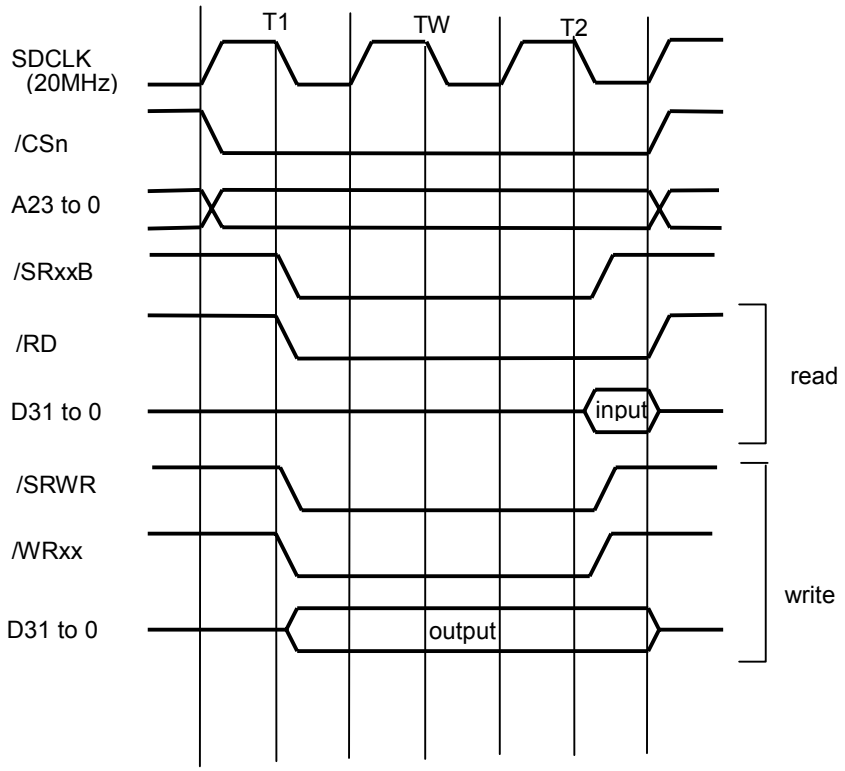


(6) Basic bus timing

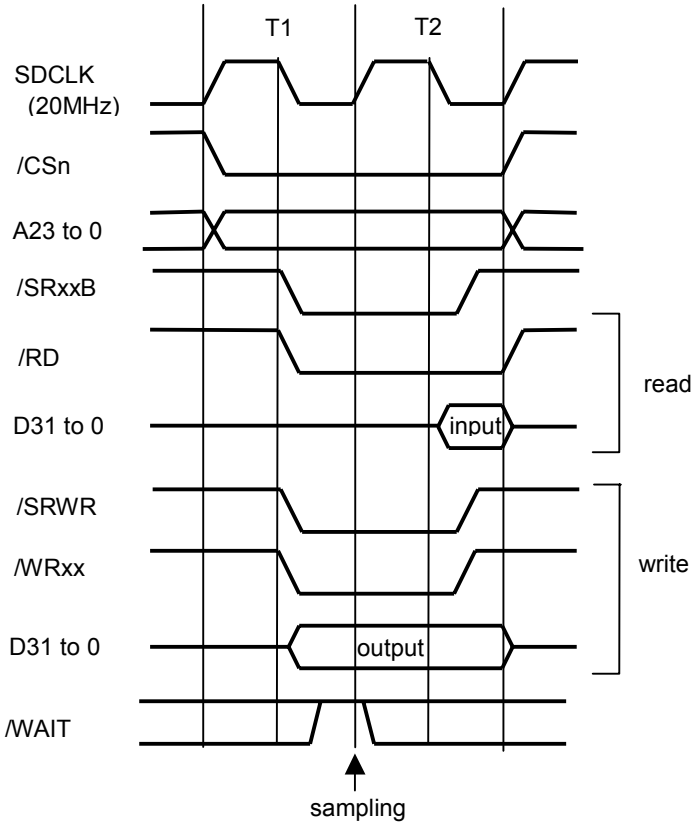
(a) External Read / Write Cycle (0 WAIT)



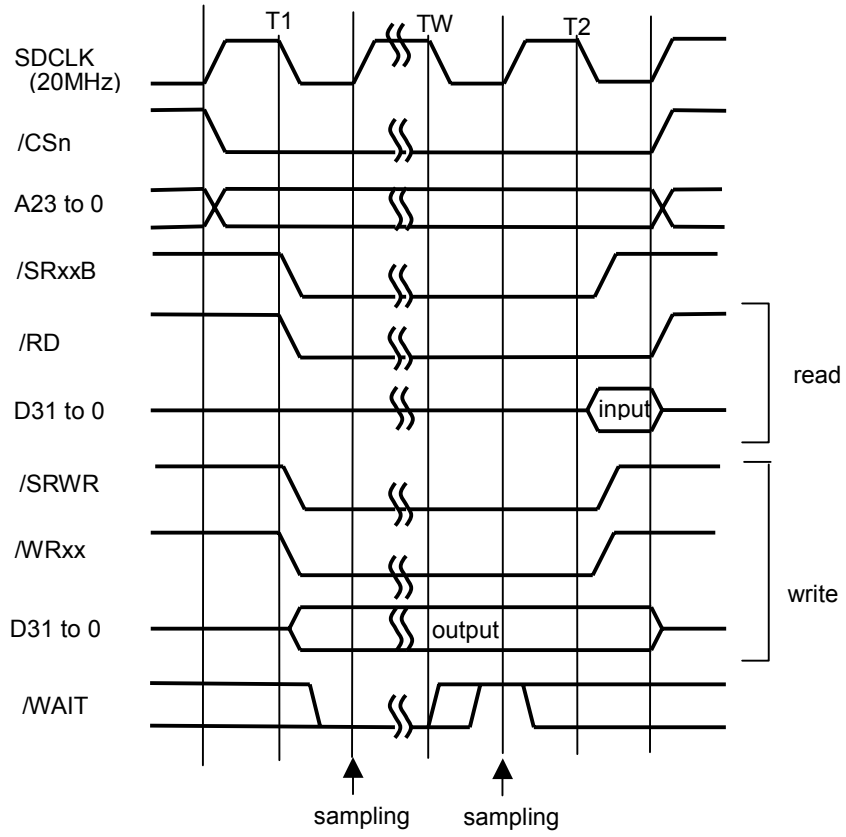
(b) External Read / Write Cycle (1 WAIT)



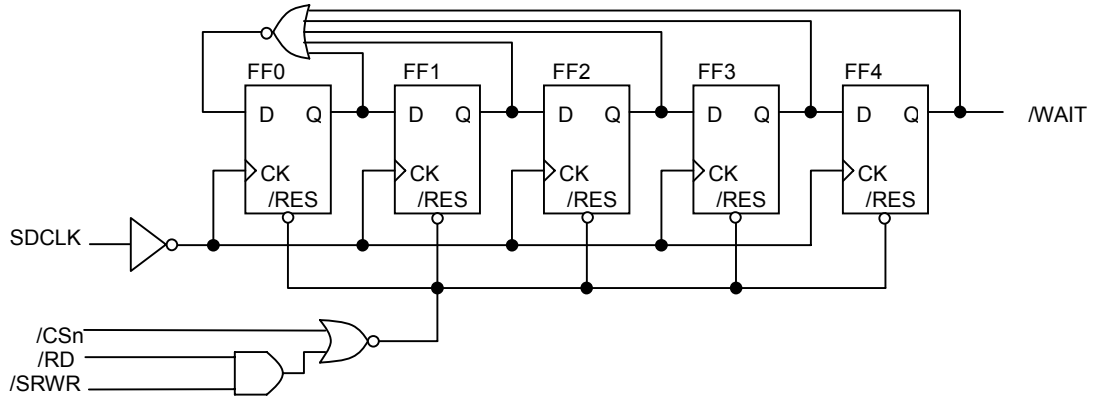
(c) External Read / Write Cycle (0 WAIT at /WAIT pin input mode)

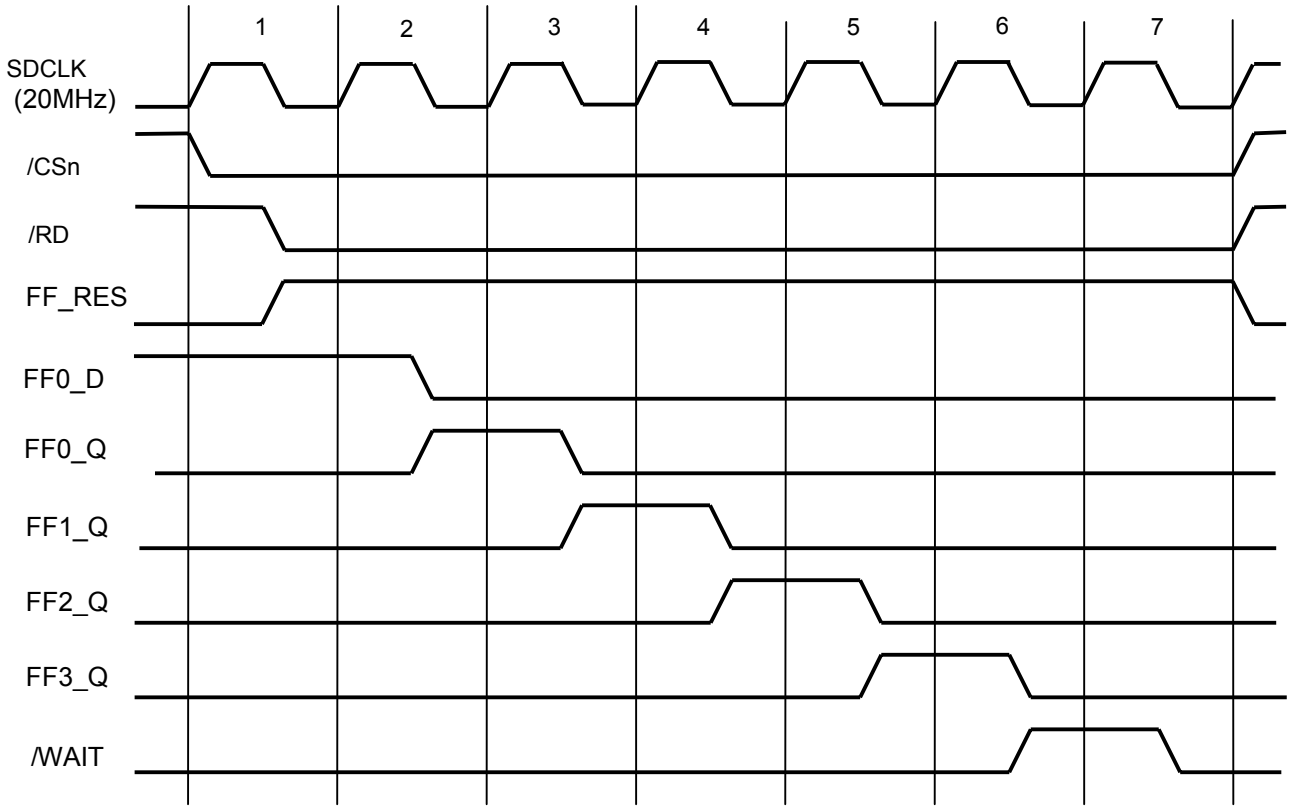


(d) External Read / Write Cycle (n WAIT at /WAIT pin input mode)



Example of WAIT Input Cycle (5WAIT)





(7) Connecting external memory

Figure 3.6.3 shows an example of how to connect external 16bit-SRAM and 16-bit NOR-Flash to the TMP92CH21.

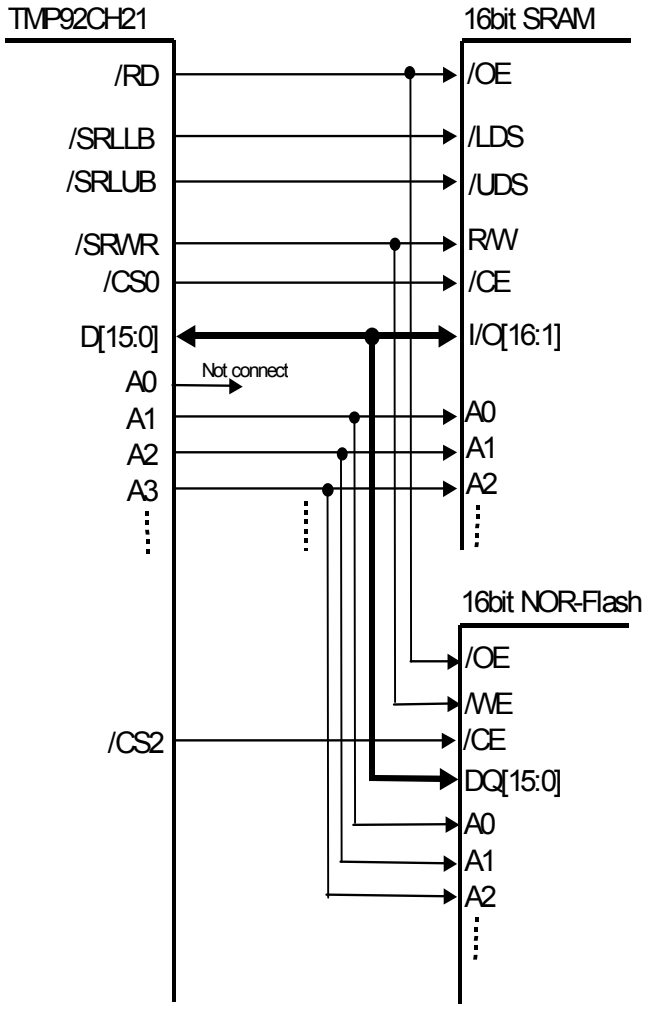


Figure 3.6.3 Example of external 16-bit SRAM and NOR-Flash connection

3.6.4 ROM Control (Page Mode)

This section describes ROM page mode accessing and how to set registers. ROM page mode is set by the page ROM control register.

(1) Operation And How To Set The Resgisters

TMP92CH21 supports ROM access of the page mode. The ROM access of the page mode is specified only in the block address area 2.

ROM page mode is set by the page ROM control register (PMEMCR). Setting <OPGE> of the PMEMCR register to “1” sets the memory access of the block address area to ROM page mode access.

The number of read cycles is set by the <OPWR1 to 0> bit of the PMEMCR register.

OPWR1/OPWR0 bit (PMEMCR Register)

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	(Reserved)

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 the <PR1 to 0> of the PMEMCR register. 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.

PR1/PR0 bit (PMEMCR Register)

PR1	PR0	ROM Page Size
0	0	64Bytes
0	1	32Bytes
1	0	16Bytes (default)
1	1	8Bytes

For the signal timing pulse, see ROM Read Cycle in section 4.3..2.

3.6.5 Internal Boot-ROM Control

This section describes about built-in Boot-ROM.

For the specification of S/W in Boot-ROM, refer 3.20 Boot-ROM section.

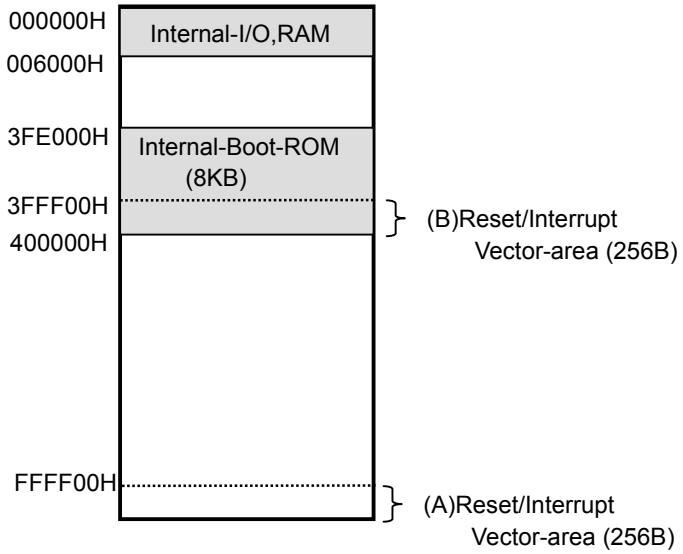
(1) Boot-mode

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	Start with 32-bit data bus
1	1	Start with BOOT(32-bit internal-MROM)

(2) Boot-ROM memory map

Boot-ROM is consist of 8Kbyte-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 of 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 aplication-system program.

This <VACE> is initialized to "1" in Boot-mode. At another starting mode, this-register has no-meaning.

(note1) The last 16-byte area (FFFFF0H – 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-register is initialized to “1”.

If this register has been set to “1”, writing “0” is disabled.

BROMCR (0167H)

	7	6	5	4	3	2	1	0
bit Symbol							ROMLESS	VACE
Read/Write							R/W	
After reset							0/1	1/0
Function							Boot-ROM 0: Use 1: No-use	Vector Address Conversion 0: disable 1: enable

3.6.6 Cautions

(1) Note the timing between /CS and /RD

If the parasitic capacitance of the /RD(read signal) is greater than that of the /CS(Chip Select Signal), it is possible that an uninttended 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 3.6(1).

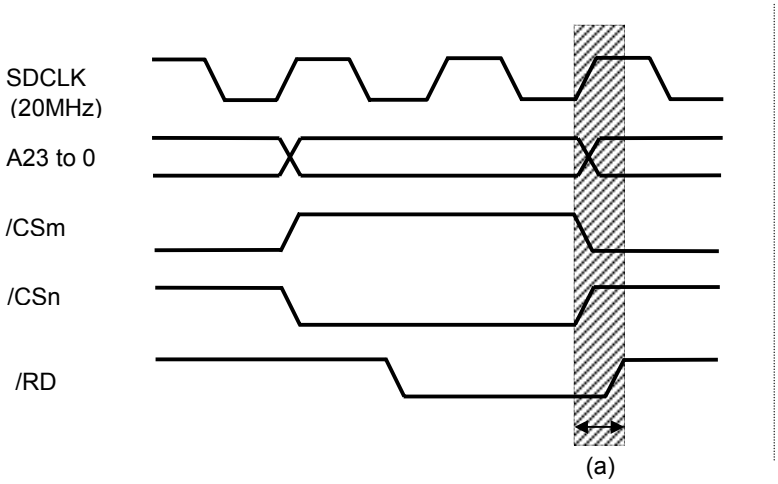


Figure 3.6(1) Read Signal Delay Read cycle

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 3.6(2), an unitended read cycle like the one shown in (b) may occur.

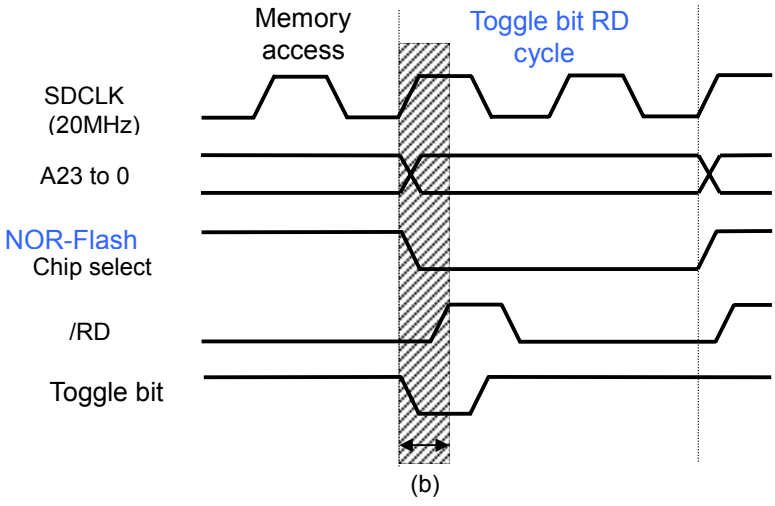


Figure 3.6(2) NOR-Flash Toggle Bit Read cycle

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.

(2) Note the NAND-Flash area setting

Figure 3.6(3) shows a memory map for NAND-Flash, RAM built-in LCD-driver.

And since CS3-area is recommended to assign address from 000000H to 3FFFFFH, this case is explained.

In this case, “NAND-Flash, RAM built-in LCD-driver” and CS3-area are overlapped. But each control circuit to access in TMP92CH21 operates independently.

So, if a program on CS3-area accesses e.g. NAND-Flash, both /CS3 and NAND-Flash will be accessed at the same time and some problem like data confliction will be occurred.

To avoid this phenomenon, CS0-area is recommended to assign address from 000000H to 007FFFH of 32Kbytes though /CS0-pin will be not needed.

Since the priority of CS0 is higher than CS3, only NAND-Flash will be accessed correctly by this setting.

(Note1) In this case, the address from 000000H to 007FFFH of 32Kbytes in CS3’s memory can’t be used.

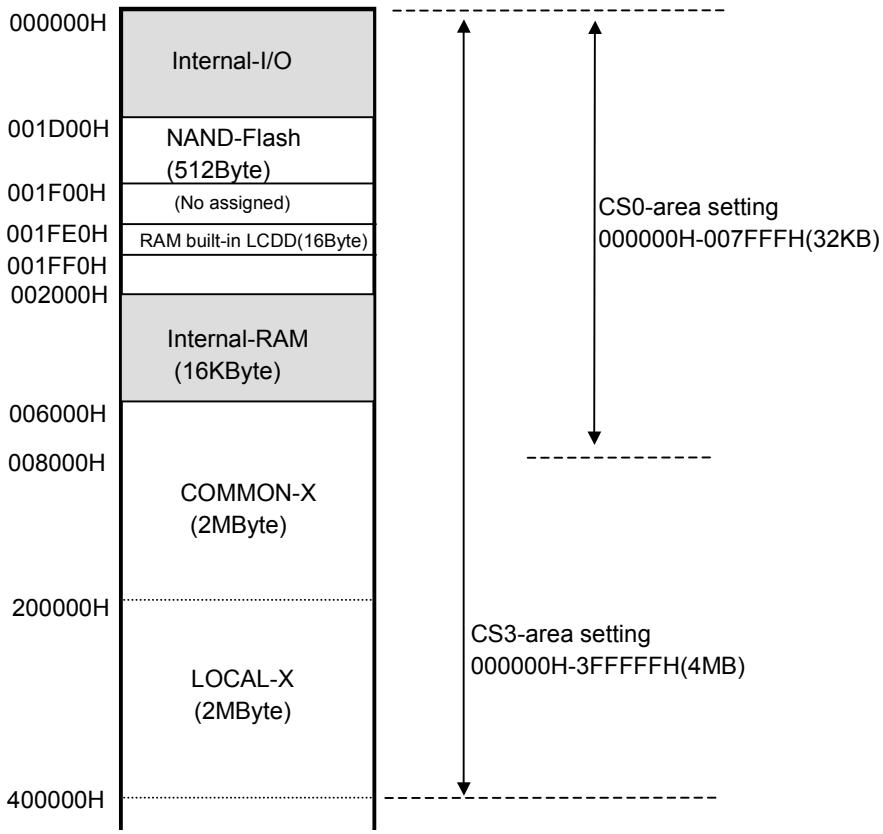


Figure 3.6(3) Recommended CS3 and CS0 setting

3.7 8-bit Timers (TMRA)

The TMP92CH21 features 4 built-in 8-bit timers.
These timers are paired into four modules: TMRA01 and TMRA23. Each module consists of two channels and can operate in any of the following four 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 3.7.1 to Figure 3.7.2 Show block diagrams for TMRA01 and TMRA23.
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 five controls SFR (special-function registers).
Each of the two modules (TMRA01 and TMRA23) 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.

3.7.1 Block diagrams
3.7.2 Operation of each circuit
3.7.3 SFR
3.7.4 Operation in each mode
(1) 8-Bit Timer Mode
(2) 16-Bit Timer Mode
(3) 8-Bit PPG (programmable pulse generation) Output Mode
(4) 8-Bit PWM (pulse width modulation) Output Mode
(5) Mode settings

Table 3.7.1 Registers and pins for each module

Module		TMRA01	TMRA23
External pin	Input pin for external clock	No	No
	Output pin for timer flip-flop	TA1OUT (shared with PC0)	TA3OUT (shared with PC1)
SFR (address)	Timer run register	TA01RUN (1100H)	TA23RUN (1108H)
	Timer register	TA0REG (1102H) TA1REG (1103H)	TA2REG (110AH) TA3REG (110BH)
	Timer mode register	TA01MOD (1104H)	TA23MOD (110CH)
	Timer flip-flop control register	TA1FFCR (1105H)	TA3FFCR (110DH)

3.7.1 Block diagrams

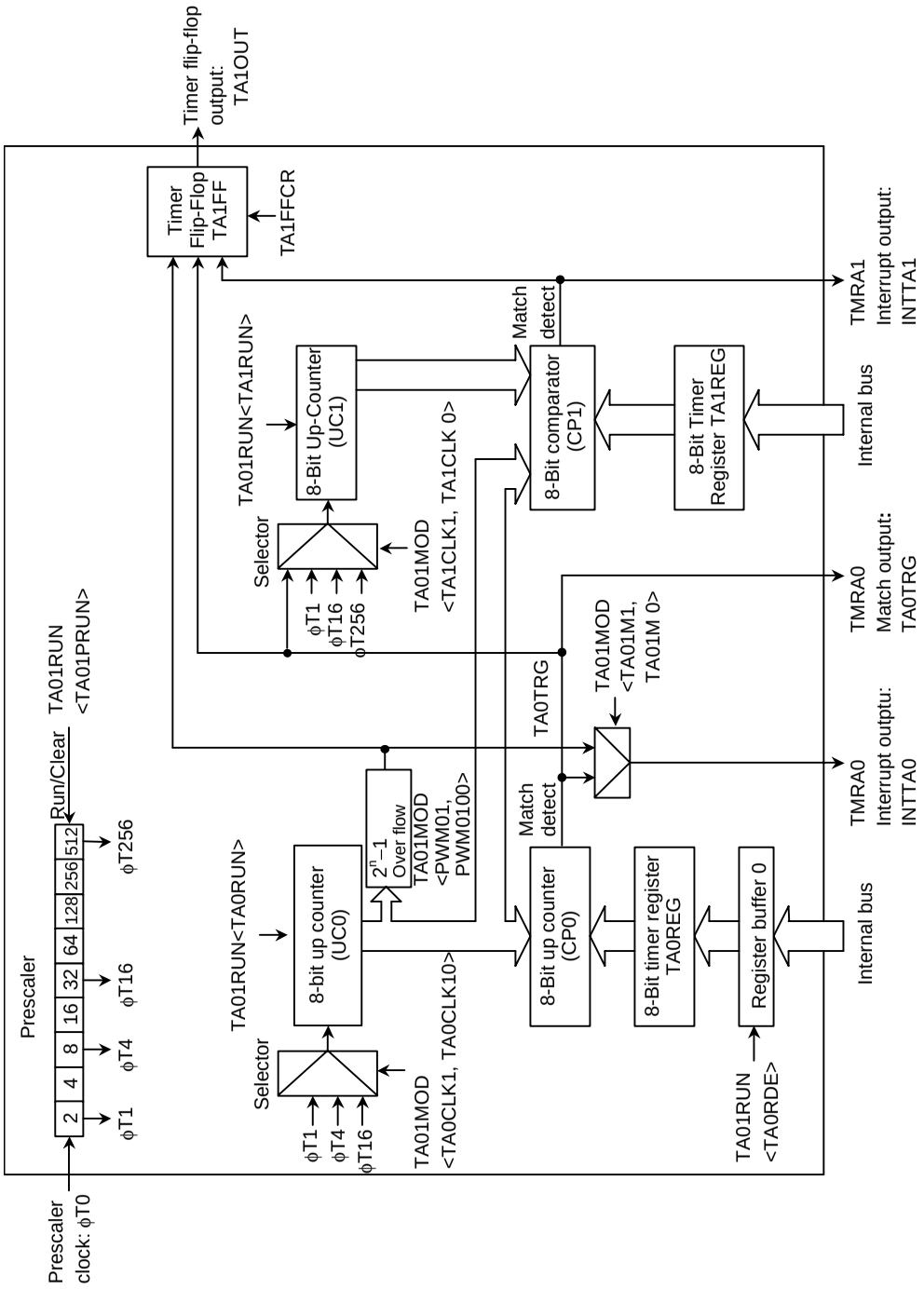


Figure 3.7.1 TMRA01 block diagram

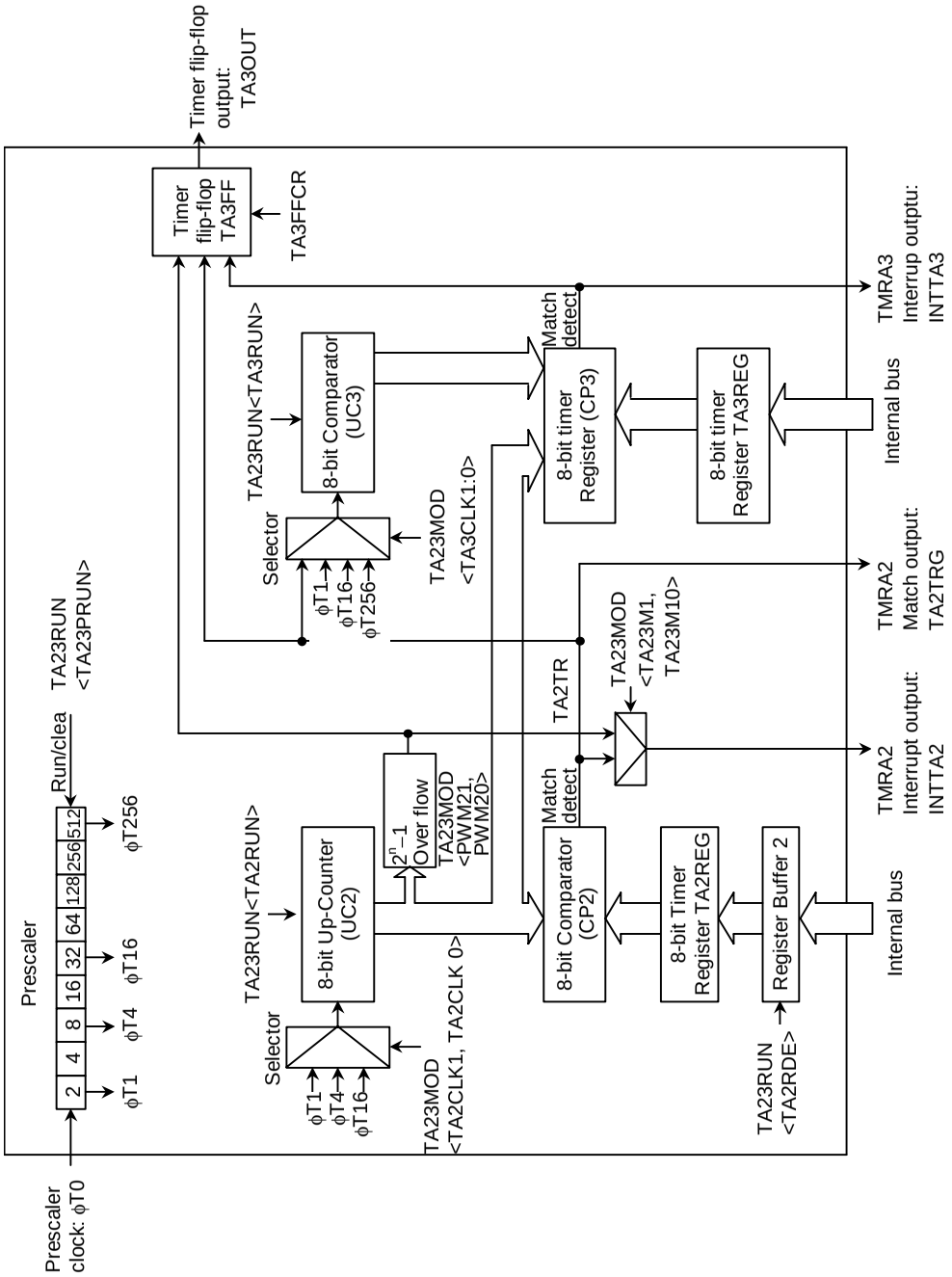


Figure 3.7.2 TMRA23 block diagram

3.7.2 Operation of each circuit

(1) Prescalers

A 9-bit prescaler generates the input clock to TMRA01.
The clock φT0 is divided by 4 the CPU clock fsys and input to this prescaler.

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 zero and stops operation. Table 3.7 (2) shows the various prescaler output clock resolutions. SYS

Table 3.7.2 Prescaler output clock resolution

System Clock Selection <SYSCK>	Gear Value <GEAR2 to0>	Prescaler Output Clock Resolution			
		φT1	φT4	φT16	φT256
1 (fs)	XXX	$fs/2^3$	$fs/2^5$	$fs/2^7$	$fs/2^{11}$
0 (fc)	000 (fc)	$f_{SYS}/2^3$	$f_{SYS}/2^5$	$f_{SYS}/2^7$	$f_{SYS}/2^{11}$
	001 ($fc_{/2}$)	$f_{SYS}/2^4$	$f_{SYS}/2^6$	$f_{SYS}/2^8$	$f_{SYS}/2^{12}$
	010 ($fc_{/4}$)	$f_{SYS}/2^5$	$f_{SYS}/2^7$	$f_{SYS}/2^9$	$f_{SYS}/2^{13}$
	011 ($fc_{/8}$)	$f_{SYS}/2^6$	$f_{SYS}/2^8$	$f_{SYS}/2^{10}$	$f_{SYS}/2^{14}$
	100 ($fc_{/16}$)	$f_{SYS}/2^7$	$f_{SYS}/2^9$	$f_{SYS}/2^{11}$	$f_{SYS}/2^{15}$

xxx: Don't care

(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,TA01CLK0>.
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.

(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-1 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.

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 3.7.7.4 show the configuration of TA0REG.

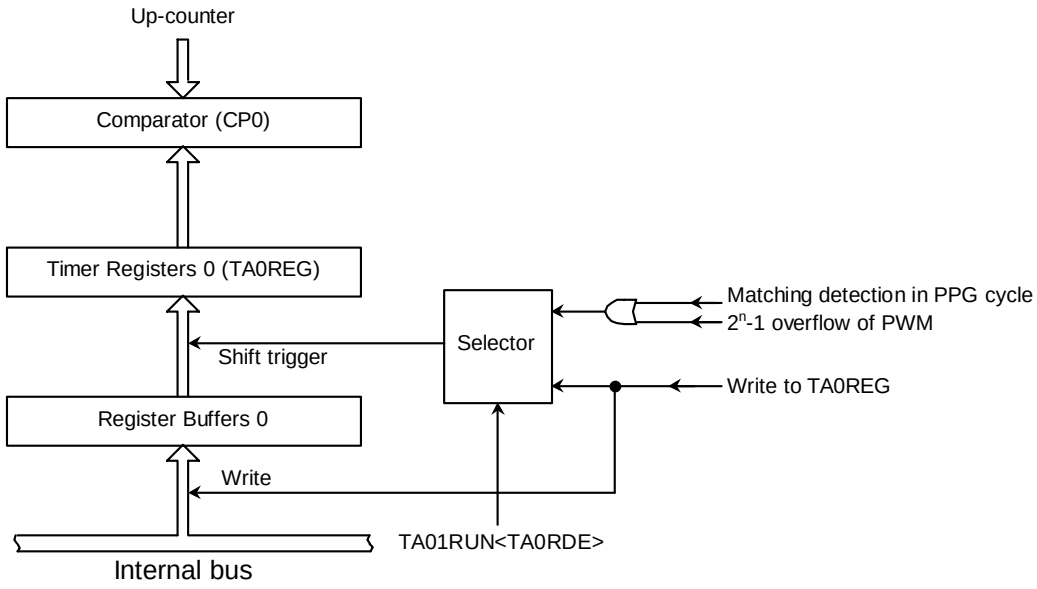


Figure 3.7.4 Configuration of TA0REG

Note: The same memory address is allocated to the timer register and the register buffer. When <TA0RDE> = 0, the same value is written to the register buffer and the timer register; when <TA0RDE> = 1, only the register buffer is written to.

The address of each timer register is as follows.

TA0REG: 001102H TA1REG: 001103H

TA2REG: 00110AH TA3REG: 00110BH

All these registers are write-only and cannot be read.

(4) Comparator (CP0)

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 zero 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.

(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<TAFF1IE> in the Timer flip-flops Control Register A Reset clears the value of TA1FF to “0”. Writing“01”or”10”to TA1FFCR<TAFF1C1, TAFF1C0>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 (which can also be used as PC0). When this pin is used as the timer output, the timer flip-flop should be set beforehand using the Port C Function Register PCCR, PCFC.

3.7.3 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				IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

TA0REG double buffer control

0	Disable
1	Enable

Timer Run/Stop control

0	Stop & Clear
1	Run (count up)

I2TA01: Operation in IDLE2 Mode

TA01PRUN: Run prescaler

TA1RUN: Run Timer 1

TA0RUN: Run Timer 0

Note: The values of bits 4 to 6 of TA01RUN are undefined 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				IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

TA2REG double buffer control

0	Disable
1	Enable

Timer Run/Stop control

0	Stop & Clear
1	Run (count up)

I2TA23: Operation in IDLE2 Mode

TA23PRUN: Run prescaler

TA3RUN: Run Timer 3

TA2RUN: Run Timer 2

Note: The values of bits 4 to 6 of TA23RUN are undefined when read.

Figure 3.7.5 Register for TMRA

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: 2^6-1 10: 2^7-1 11: 2^8-1		Source clock for TMRA1 00: TA0TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA0 00: (reserved) 01: φT1 10: φT4 11: φT16	

TMRA0 source clock selection

00	reserved
01	φT1 (prescaler)
10	φT4 (prescaler)
11	φT16 (prescaler)

TMRA1 source clock selection

	TA01MOD <TA01M1–TA01M0>≠01	TA01MOD <TA01M1–TA01M 0>=01
00	Comparator output from TMRA0	Overflow output from TMRA0
01	φT1	
10	φT16	
11	φT256	(16-Bit Timer Mode)

PWM cycle selection

00	reserved
01	$(2^6\text{-}1) \times$ clock source
10	$(2^7\text{-}1) \times$ clock source
11	$(2^8\text{-}1) \times$ clock source

TMRA01 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA0) + 8-bit timer (TMRA1)

Figure 3.7.6 TMRA registers

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: 2^6-1 10: 2^7-1 11: 2^8-1		TMRA3 clock for TMRA3 00: TA2TRG 01: φT1 10: φT16 11: φT256		TMRA2 clock for TMRA2 00: reserved 01: φT1 10: φT4 11: φT16	

TMRA2 source clock selection

00	Do not set
01	φT1 (prescaler)
10	φT4 (prescaler)
11	φT16 (prescaler)

TMRA3 source clock selection

	TA23MOD <TA23M1~TA23M0>≠01	TA23MOD <TA23M1~TA23M0>=01
00	Comparator output from TMRA2	Overflow output from TMRA2
01	φT1	
10	φT16	
11	φT256	(16-Bit Timer Mode)

PWM cycle selection

00	reserved
01	$(2^6\text{-}1) \times$ clock source
10	$(2^7\text{-}1) \times$ clock source
11	$(2^8\text{-}1) \times$ clock source

TMRA23 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA2) + 8-bit timer (TMRA3)

Figure 3.7.7 TMRA registers

TMRA1 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA1FFCR (1105H)	Bit symbol					TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
	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

Inverse signal for Timer Flop-Flop 1 (TA1FF)
(Don't care except in 8-Bit Timer Mode)

0	Inversion by TMRA0
1	Inversion by TMRA1

Inversion of TA1FF

0	Disabled
1	Enabled

Control of TA1FF

00	Inverts the value of TA1FF
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 undefined when read.

Figure3.7.8 TMRA registers

TMRA3 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA3FFCR (110DH)	Bit symbol					TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
	Read/Write					R/W		R/W	
	After Reset					1	1	0	0
Read-Modify-Write instructions are prohibited.	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

Inverse signal for Timer Flip-Flop 3 (TA3FF)
(Don't care except in 8-Bit Timer Mode)

0	Inversion by TMRA2
1	Inversion by TMRA3

Inversion of TA3FF

0	Disabled
1	Enabled

Control of TA3FF

00	Inverts the value of TA3FF
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 undefined when read.

Figure3.7.9 TMRA register

Timer Register

Symbol	Address	7	6	5	4	3	2	1	0
TA0REG	1102H	-							
		W							
		undefined							
TA1REG	1103H	-							
		W							
		undefined							
TA2REG	110AH	-							
		W							
		undefined							
TA3REG	110BH	-							
		W							
		undefined							

Figure 3.7.10 Register for 8-bit Timers

3.7.4 Operation in each mode

(1) 8-Bit Timer Mode

Both timer 0 and timer 1 can be used independently as 8-bit interval timers.

① Generating interrupts at a fixed interval (using TMRA1)

To generate interrupts at constant intervals using timer 1 (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 40 μseconds at fsys = 20 MHz, set each register as follows:

		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	-	-	Select 8-Bit Timer Mode and select φT1 (0.4 μs at fsys = 20 MHz) as the input clock.
TA1REG	←	0	1	1	0	0	1	0	0	Set TREG1 to 40 μs ÷ φT1 = 100 = 64H
INTETA01	←	X	1	0	1	–	–	–	–	Enable INTTA1 and set it to Level 5.
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA11 counting.

Note: X = Don't care; "–" = No change

Select the input clock using Table 3.7 2.

Table 3.7.2 Selecting Interrupt Interval and the Input Clock Using 8-Bit Timer

Input Clock	Interrupt Interval (at fsys = 20 MHz)	Resolution
φT1 (8/fsys)	0.4 μs to 102.4 μs	0.4 μs
φT4 (32/fsys)	1.6 μs to 409.6 μs	1.6 μs
φT16 (128/fsys)	6.4 μs to 1.639 ms	6.4 μs
φT256 (2048/fsys)	102.4 μs to 26.22 ms	102.4 μs

Note: The input clocks for TMRA0 and TMRA1 differ as follows:
TMRA0: Uses TMRA 0 input (TA0IN) and can be selected from φT1, φT4 or φT16
TMRA1: Match output of TMRA0 (TA0TRG) and can be selected from φT1, φT16, φT256

② Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF1) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 2.4-μs square wave pulse from the TA1OUT pin at fsys = 20 MHz, use the following procedure to make the appropriate register settings. This example uses timer 1; however, either timer 0 or timer 1 may be used.

		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	–	–	Select 8-Bit Timer Mode and select ϕT1 (0.4 μs at fsys = 20 MHz) as the input clock.
TA1REG	←	0	0	0	0	0	0	1	1	Set the timer register to 2.4 μs ÷ ϕT1 ÷ 2 = 3
TA1FFCR	←	X	X	X	X	1	0	1	1	Clear TA1FF to 0 and set it to invert on the match detect signal from timer 1.
PCCR	←	–	–	–	–	–	–	–	1	Set PC0 to function as the TA1OUT pin.
PCFC	←	–	–	–	–	–	–	–	1	
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

Note: X = Don't care; "–" = No change

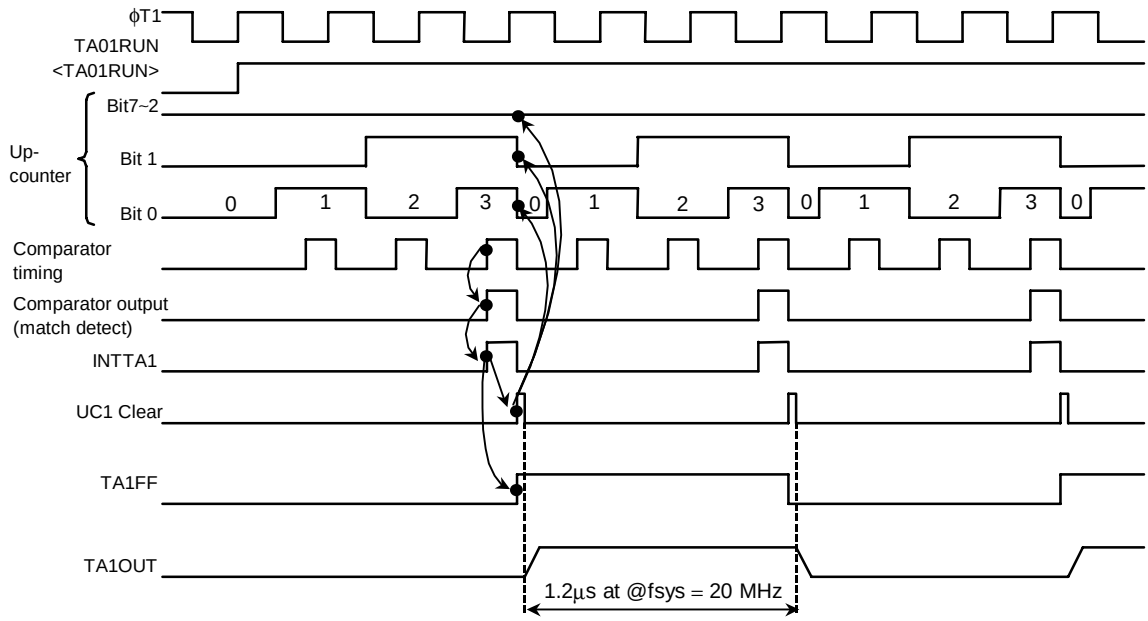


Figure 3.7.11 Square wave output timing chart (50% Duty)

③ 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.

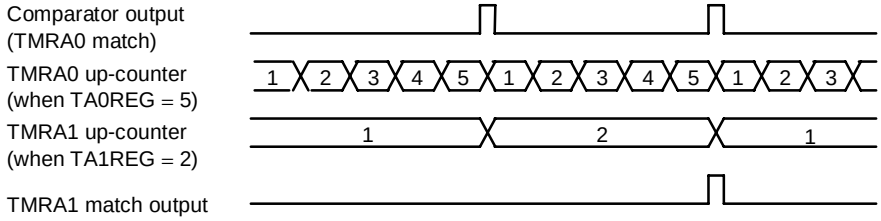


Figure 3.7.12 TMRA1 Count Up on Signal from TMRA0

(2) 16-Bit Timer Mode

A 16-bit interval timer is configured by pairing the two 8-bit timers TMRA0 and TMRA1.

To make a 16-bit interval timer in which TMRA0 and TMRA1 are cascaded together, set TA01MOD <TA01M1,TA01M0> 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,TA011CLK0>. Table 3.7.2 shows the relationship between the timer (interrupt) cycle and the input clock selection.

To set the timer interrupt interval, set the lower eight bits in timer register TA0REG and the upper eight bits in TA1REG. Be sure to set TA0REG first (as entering data in TA0REG temporarily disables the compare, while entering data in TA1REG starts the compare).

Setting example: To generate an INTTA1 interrupt every 0.4 seconds at fsys = 20 MHz, set the timer registers TA0REG and TA1REG as follows:

If ϕT16 (6.4 μs at fsys = 20 MHz) is used as the input clock for counting, set the following value in the registers: 0.4 s ÷ 6.4 μs = 62500 = F424H; i.e. set TA1REG to F4H and TA0REG to 24H.

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

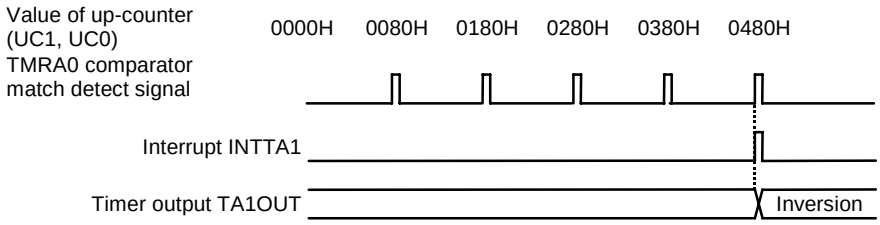


Figure3.7.13 Timer output by 16-Bit Timer Mode

(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 (which can also be used as PC0).

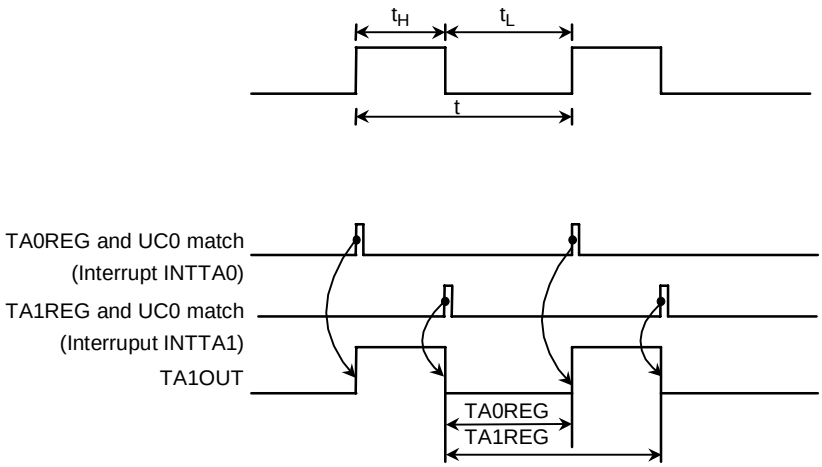


Figure3.7.14 8 bit PPG output waveforms

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 shows a block diagram representing this mode.

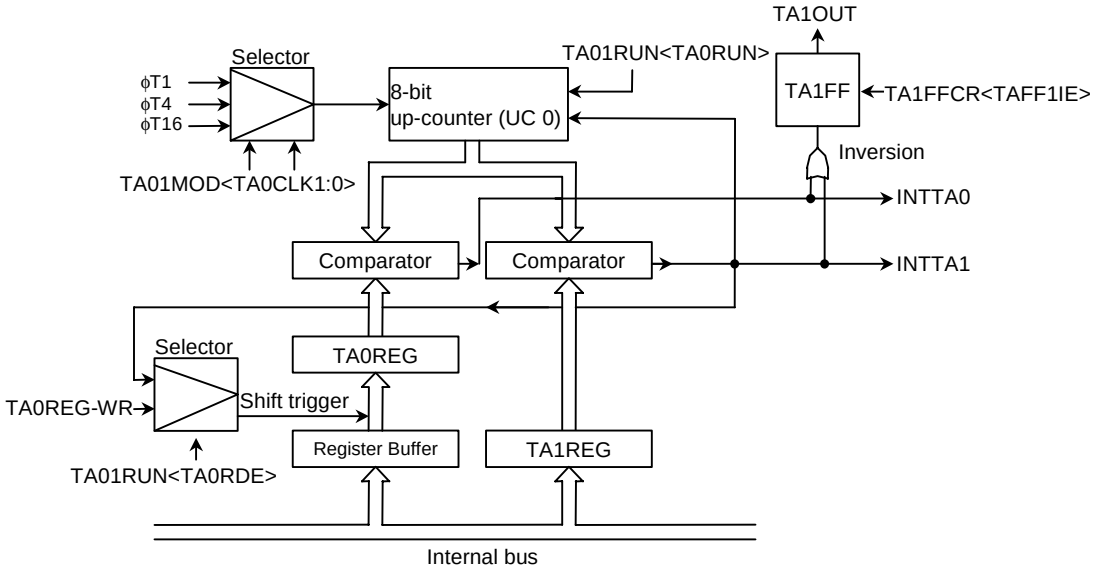


Figure3.7.15 Block diagram of 8-Bit PPG Output Mode

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).

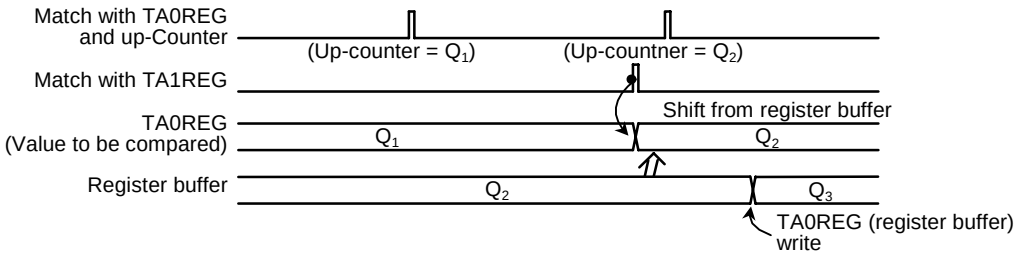
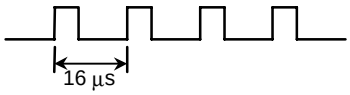


Figure3.7.16 Operation of register buffer

Example: To generate 1/4-duty 62.5 kHz pulses (at fsys = 20 MHz):



Calculate the value which should be set in the timer register.

To obtain a frequency of 62.5 kHz, the pulse cycle t should be: t = 1/62.5 kHz = 16 μs

φT1 = 0.4 μs (at 20 MHz);

16 μs ÷ 0.4 μs = 40

Therefore set TA1REG to 40 (28H)

The duty is to be set to 1/4: t × 1/4 = 16 μs × 1/4 = 4 μs

4 μs ÷ 0.4 μs = 10

Therefore, set TA0REG = 10 = 0AH.

		7	6	5	4	3	2	1	0	
TA01RUN	←	0	X	X	X	–	0	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 0AH
TA1REG	←	0	0	1	0	1	0	0	0	Write 28H
TA1FFCR	←	X	X	X	X	0	1	1	X	Set TA1FF, enabling both inversion and the double buffer.
										10 generates a negative logic pulse.
PCCR	←	–	–	–	–	–	–	–	1	Set PC0 as the TA1OUT pin.
PCFC	←	–	–	–	–	–	–	–	1	
TA01RUN	←	1	X	X	X	–	1	1	1	Start TMRA0 and TMRA1 counting.

Note: X = Don’t care; “–” = No change

(4) 8-Bit PWM 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 (which is also used as PC1). 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-1 counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01 to PWM00>). The up-counter UC0 is cleared when 2^n-1 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-1 counter overflow

Value set in TA0REG $\neq$ 0

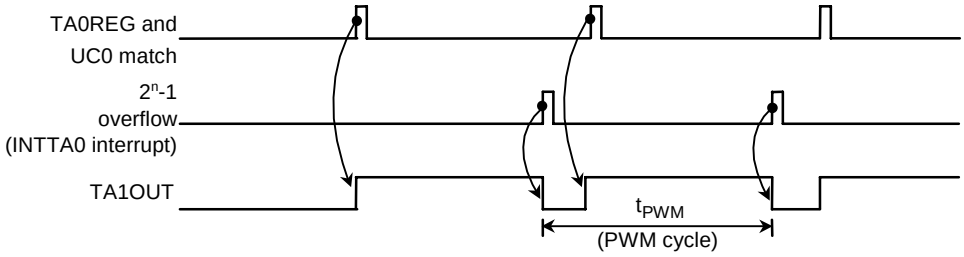


Figure 3.4.17 8-bit PWM waveforms

Figure3.7.1 shows a block diagram representing this mode.

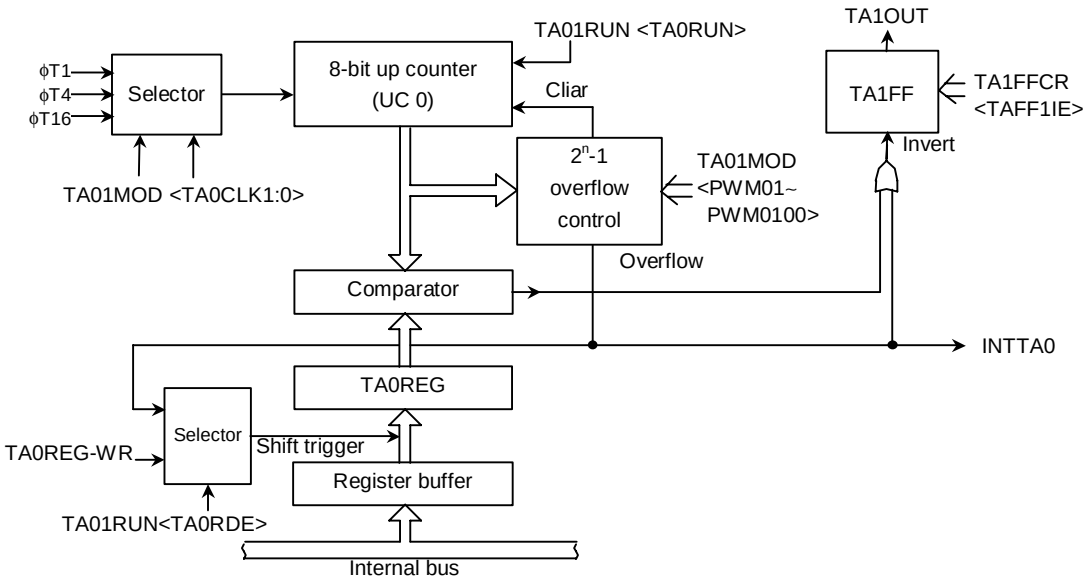


Figure3.7.18 Block diagram of 8-Bit PWM Mode

In this mode the value of the register buffer will be shifted into TA0REG if $2^n - 1$ overflow is detected when the TA0REG double buffer is enabled.
Use of the double buffer facilitates the handling of low duty ratio waves.

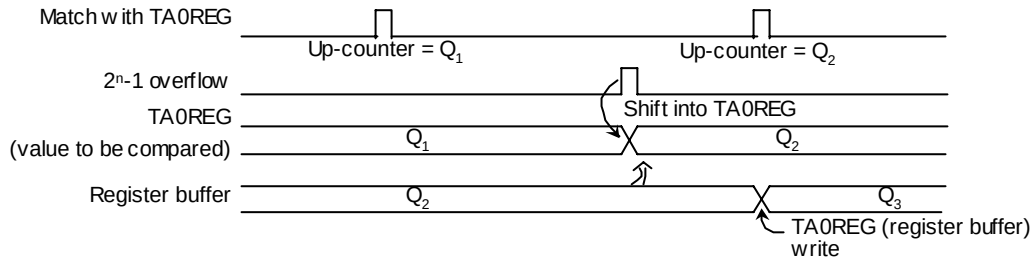
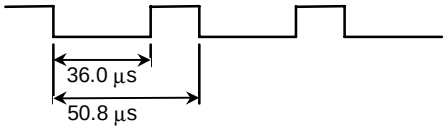


Figure 3.7.19 Register buffer operation

Example: To output the following PWM waves on the TA1OUT pin at fsys = 20 MHz:



To achieve a 50.8-µs PWM cycle by setting ϕT1 to 0.4 µs (at fsys = 20 MHz):
50.8 µs ÷ 0.4 µs = 127
$2^n - 1 = 127$
Therefore n should be set to 7.
Since the low-level period is 36.0 µs when ϕT1 = 0.4 µs,
set the following value for TREG0:
36.0 µs ÷ 0.4 µs = 90 = 5AH

		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	–	–	0	1	Select 8-Bit PWM Mode (cycle: 2^7-1) and select ϕT1 as the input clock.
TA0REG	←	0	1	0	1	1	0	1	0	Write 5AH.
TA1FFCR	←	X	X	X	X	1	0	1	X	Clear TA1FF to 0, enable the inversion and double buffer.
PCCR	←	–	–	–	–	–	–	–	1	Set PC0 as the TA1OUT pin.
PCFC	←	–	–	–	–	–	–	–	1	
TA01RUN	←	1	X	X	X	–	1	-	1	Start TMRA0 counting.

Note: X = Don't care; "–" = No change

Table 3.7.3 PWM cycle

	PWM Interval (at fsys = 20MHz)		
	φT1	φT4	φT16
$2^6 - 1$	25.2 μs (39.7 kHz)	100.8 μs (9.92 kHz)	403.2 μs (2.48 kHz)
$2^7 - 1$	50.8 μs (19.6 kHz)	203.2 μs (4.92 kHz)	810 μs (1.23 kHz)
$2^8 - 1$	102 μs (9.80 kHz)	408 μs (2.45 kHz)	1.63 ms (0.61 kHz)

(5) Settings for each mode

Table 3.7.4 shows the SFR settings for each mode.

Table 3.7.4 Timer mode setting registers

Register name	TA01MOD				TA1FFCR
<Bit Symbol>	<TA01M1:0>	<PWM01:00>	<TA1CLK1:0>	<TA0CLK1:0>	<TAFF1IS>
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	$2^6 - 1$, $2^7 - 1$, $2^8 - 1$ (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

Note:"–" = Don't care

3.8. External memory extension function (MMU)

This is MMU function which can expand program / data area to 512M byte by having 3 local area.

The recommendation address memory map is shown in Fig. 3.8.1 (1), (2).

However, when 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 setting of bank is called LOCAL-area. Since the address for LOCAL area is fixed, it cannot be changed.

A program is disabled to branch between different bank of same local-area.

TMP92CH21 has following external pins to connect memory-LSI .

Address bus : EA25,EA24 and A23 to A0
Chip Select : /CS0 to /CS3,/CSZA to /CSZF, /SDCS, /ND0CE and /ND1CE
Data bus : D31 to D0

3.8.1. Recommended memory map

Fig. 3.8.1 (1) shows one of recommendation address memory map. It can be expanded to maximum memory size and it's for the system which internal Boot-ROM with NAND-Flash is not needed.

Fig. 3.8.1 (2) also shows one of recommendation address memory map. It's for a simple memory system like internal Boot-ROM with NAND-Flash and SDRAM.

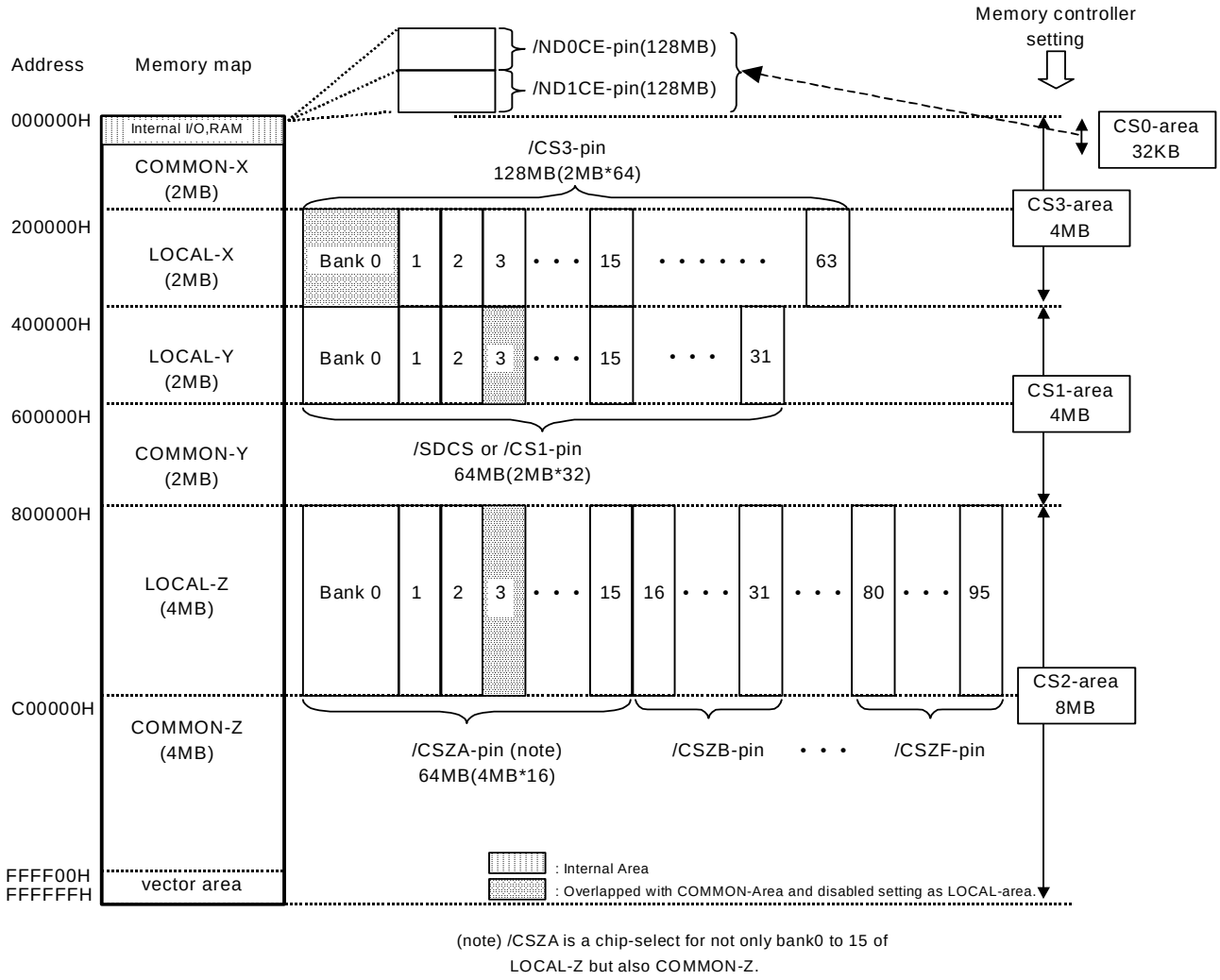


Figure 3.8.1.(1) –a Recommendation memory map for maximum specification (Logical)

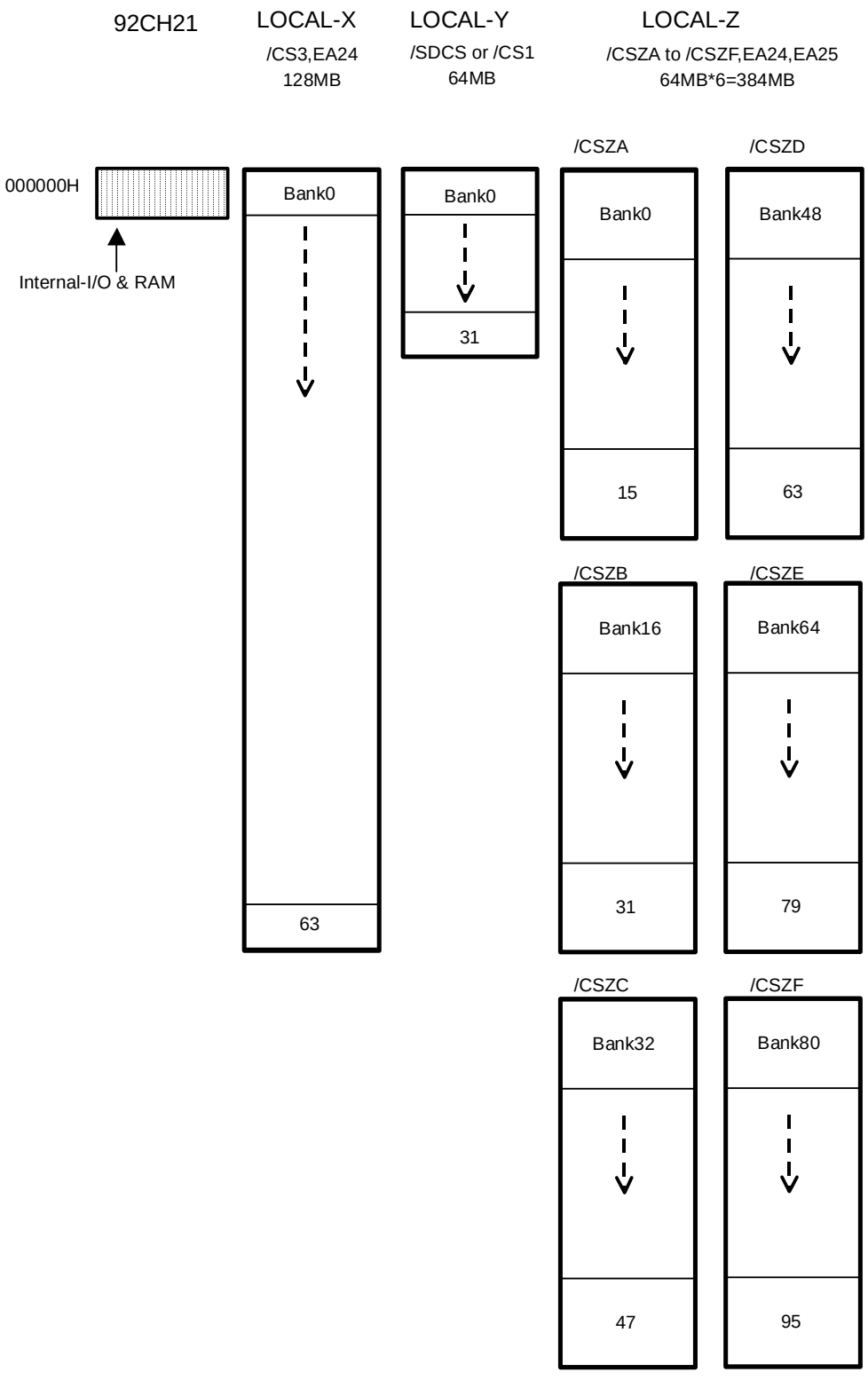


Figure 3.8.1(1) –b Recommendation memory map for maximum specification (Physical)

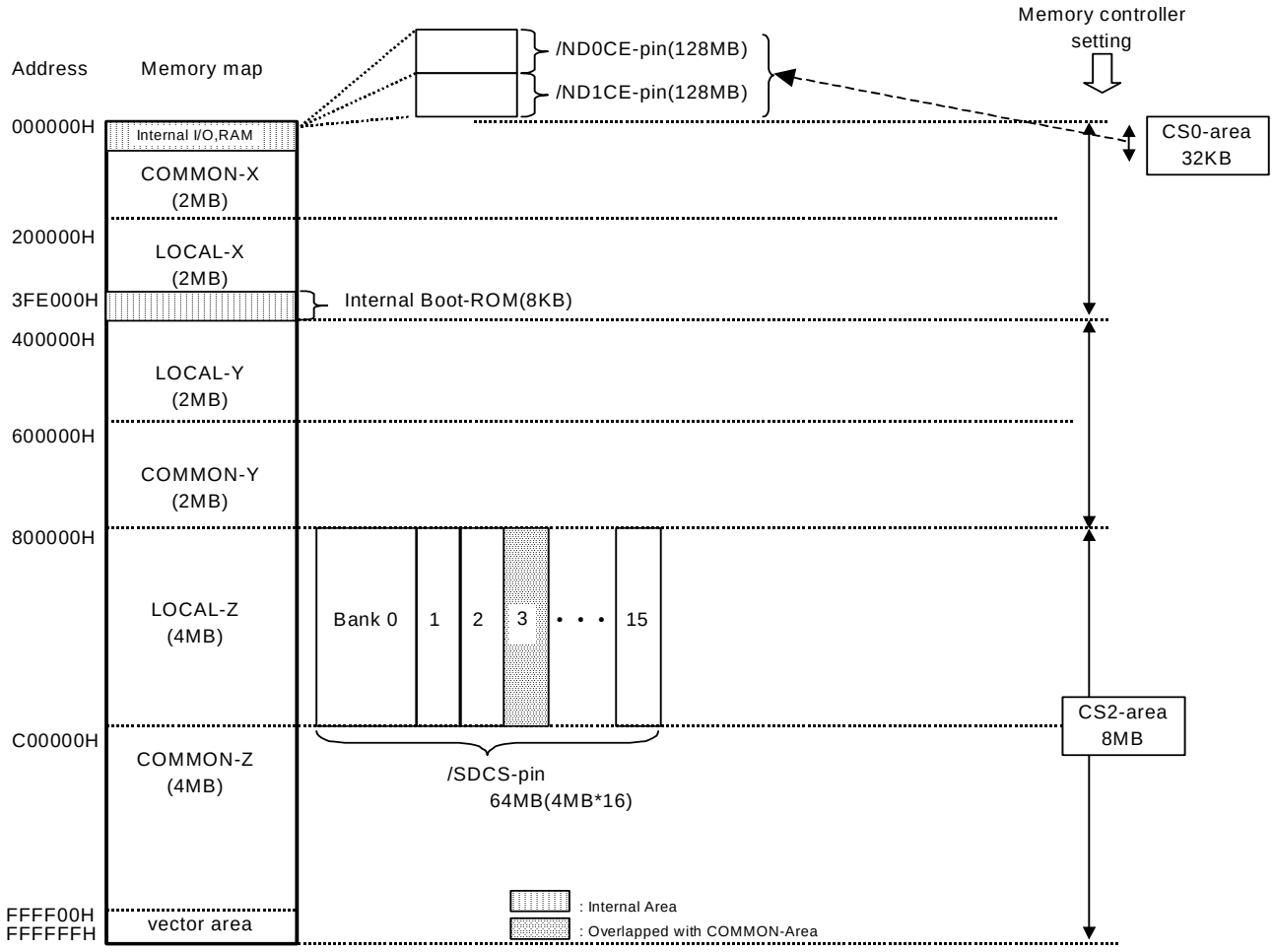


Figure 3.8.1(2) –a Recommendation memory map for simple system (Logical)

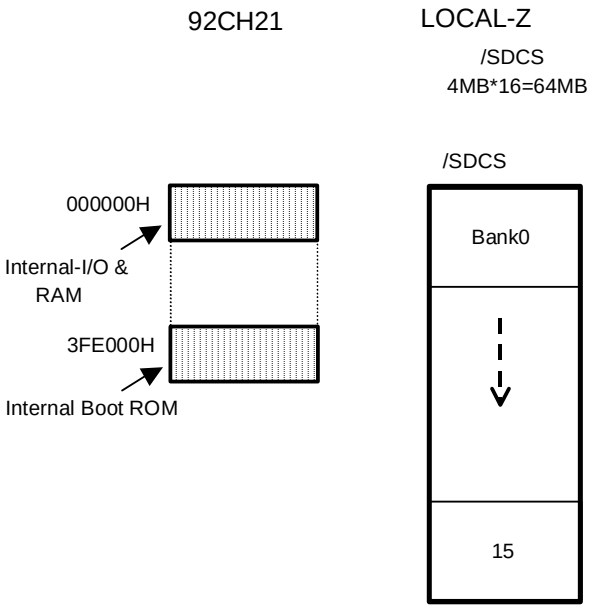


Figure 3.8.1(2) –b Recommendation memory map for for simple system (Physical)

3.8.2 Control registers

There are 12-registers for MMU. They are prepared for 4-purpose using (as Program, data-Read, data-Write and LCDC-display data), and 3-local area(Local-X,Y and Z).

By these 4-purpose registers, a data can be accessed easily.

(How to use)

At first, set enable register and bank-number of each LOCAL register.

In that case, a combination pin and memory setting also should be set to the Ports and Memory controller.

After that, if CPU or LCDC outputs logical address of the local area, MMU converts and outputs it to physical address according to the bank number. The physical address bus is output to the external address bus pin. By this operation, accessing to external memory becomes possible.

(Note)

1) Since the common area is disabled to use as local area, don't set bank number which is overlapped with common area to LOCAL register.
2) In the LOCAL-area, changing Program bank number (LOCALPX, Y or Z) is disabled.
3) After data bank number (LOCALRn, LOCALWn or LOCALLn; "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    ;
ld   (localrx),81h  ;  data bank number is set
ld   wa,(localrx)   ;  <---- Inserted Dummy instruction which accesses SFR
ld   wa,(xix)       ;  instruction which reads bank1 of local-X area.
```

LOCAL-X register for Program

LOCALPX (01D0H)

	7	6	5	4	3	2	1	0
bit Symbol	LXE		X5	X4	X3	X2	X1	X0
Read/Write	R/W		R/W					
After reset	0		0					
Function	Use BANK for LOCALX 0: not use 1: use		Setting BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)					

LOCAL-Y register for Program

LOCALPY (01D1H)

	7	6	5	4	3	2	1	0
bit Symbol	LYE			Y4	Y3	Y2	Y1	Y0
Read/Write	R/W			R/W				
After reset	0			0				
Function	Use BANK for LOCALY 0: not use 1: use			Setting BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)				

LOCAL-Z register for Program

LOCALPZ (01D3H)

	7	6	5	4	3	2	1	0
bit Symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
Read/Write	R/W	R/W						
After reset	0	0						
Function	Use BANK for LOCALZ 0: disable 1: enable	Setting BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)						

LOCAL-X register for LCDC-display data

LOCALLX (01D4H)

	7	6	5	4	3	2	1	0
bit Symbol	LXE		X5	X4	X3	X2	X1	X0
Read/Write	R/W		R/W					
After reset	0		0					
Function	Use BANK for LOCALX 0: not use 1: use		Setting BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)					

LOCAL-Y register for LCDC-display data

LOCALLY (01D5H)

	7	6	5	4	3	2	1	0
bit Symbol	LYE			Y4	Y3	Y2	Y1	Y0
Read/Write	R/W			R/W				
After reset	0			0				
Function	Use BANK for LOCALY 0: not use 1: use			Setting BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)				

LOCAL-Z register for LCDC-display data

LOCALLZ (01D7H)

	7	6	5	4	3	2	1	0
bit Symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
Read/Write	R/W	R/W						
After reset	0	0						
Function	Use BANK for LOCALZ 0: disable 1: enable	Setting BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)						

LOCAL-X register for Read data

LOCALRX (01D8H)

	7	6	5	4	3	2	1	0
bit Symbol	LXE		X5	X4	X3	X2	X1	X0
Read/Write	R/W		R/W					
After reset	0		0					
Function	Use BANK for LOCALX 0: not use 1: use		Setting BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)					

LOCAL-Y register for Read data

LOCALRY (01D9H)

	7	6	5	4	3	2	1	0
bit Symbol	LYE			Y4	Y3	Y2	Y1	Y0
Read/Write	R/W			R/W				
After reset	0			0				
Function	Use BANK for LOCALY 0: not use 1: use			Setting BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)				

LOCAL-Z register for Read data

LOCALRZ (01DBH)

	7	6	5	4	3	2	1	0
bit Symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
Read/Write	R/W	R/W						
After reset	0	0						
Function	Use BANK for LOCALZ 0: disable 1: enable	Setting BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)						

LOCAL-X register for Write data

LOCALWX (01DCH)

	7	6	5	4	3	2	1	0
bit Symbol	LXE		X5	X4	X3	X2	X1	X0
Read/Write	R/W		R/W					
After reset	0		0					
Function	Use BANK for LOCALX 0: not use 1: use		Setting BANK number for LOCAL-X ("0" is disabled because of overlapped with Common-area.)					

LOCAL-Y register for Write data

LOCALWY (01DDH)

	7	6	5	4	3	2	1	0
bit Symbol	LYE			Y4	Y3	Y2	Y1	Y0
Read/Write	R/W			R/W				
After reset	0			0				
Function	Use BANK for LOCALY 0: not use 1: use			Setting BANK number for LOCAL-Y ("3" is disabled because of overlapped with Common-area.)				

LOCAL-Z register for Write data

LOCALWZ (01DFH)

	7	6	5	4	3	2	1	0
bit Symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
Read/Write	R/W	R/W						
After reset	0	0						
Function	Use BANK for LOCALZ 0: disable 1: enable	Setting BANK number for LOCAL-Z ("3" is disabled because of overlapped with Common-area.)						

3.8.3 Setting example

The below is a 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)	/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)	/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 (16KB)	---- (32bit,1clk)	Bank2 in LOCAL-Y	002000H to 005FFFH	

(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
		6	ld (p8fc), 02H	; P81: /CS1
		7	ld (p8fc2), 04H	; P82: /CSZA
		8	ld (pjfc), 07H	; PJ2:/SRWR, PJ1:/SRLUB, PJ0:/SRLLB
		9	ld xsp,6000H	; Stack Pointer = 6000H
		10	ld (localpy),80H	; Bank0 in LOCAL-Y is set as Program 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 No.12's instruction.

・No.12 is an instruction to call sub routine. When CPU outputs 400000H address, this MMU will convert and output 000000H 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 brunch to sub-routine.

(Note this example is based on sub routine program is already written on SRAM.)

(c) Sub routine (Bank-0 in LOCAL-Y)

Logical address	Physical address	No	Instruction	Comment
		16	org 400000H	;
400000H	000000H	17	ld (localwy),81H	; Bank1 in LOCAL-Y is set as Write data for LCD Display RAM
4000xxH	0000xxH	18	ld (locally), 81H	; Bank1 in LOCAL-Y is set as LCD display data for LCD Display RAM
		19	ld (localrz), 80H	; Bank0 in LOCAL-Z is set as Read data for Character-ROM
		20	ld xiy,800000H	; Index address register to read Character-ROM
		21	ld wa,(xiy)	; Reading Character-ROM
		22	:	; Convert it to display-data
		23	ld (localpy), 82H	;
		24	ld xix, 400000H	; Index address register to write LCD Display data
		25	ld (xix), bc	; Writing LCD Display data
		26	:	; Setting LCD Controller
		27	:	;
		28	ld xiz, 400000H	; Setting 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.

3.9 Serial Channels

TMP92CH21 includes 2serial I/O channels. For each channels either UART Mode (asynchronous transmission) or I/O Interface Mode (synchronous transmission) can be selected.

- I/O Interface Mode ——— Mode 0: For transmitting and receiving I/O data using the synchronizing signal SCLK for extending I/O.
- 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 wake-up function for making the master controller start slave controllers via a serial link (a multi-controller system).
Figure 3.9.2, 3, 4 are block diagrams for each channel.
Each Channel can be used independently.
Each channel operates in the same fashion except for the following points; hence only the operation of Channel 0 is explained below.

Table 3.9.1 Differences between Channels 0 to 1

	Channel 0	Channel 1
Pin Name	TXD0 (P90 or PF0) RXD0 (P91 or PF1) /CTS0,SCLK0 (P92 or PF2)	TXD1 (PF0) RXD1 (PF1) /CTS1,SCLK1 (PF2)
IrDA Mode	Yes	No

This chapter contains the following sections:

3.9.1Block diagram

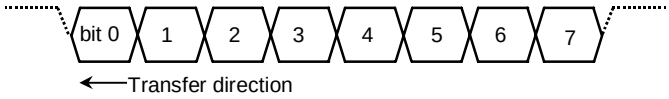
3.9.2Operation of each circuit

3.9.3SFR

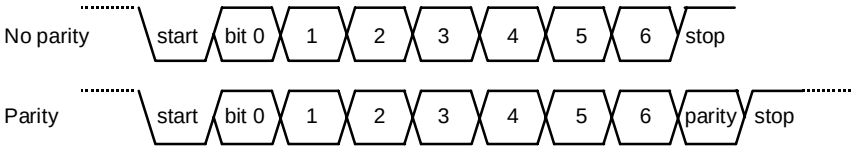
3.9.4Operation in each mode

3.9.5Support for IrDA Mode

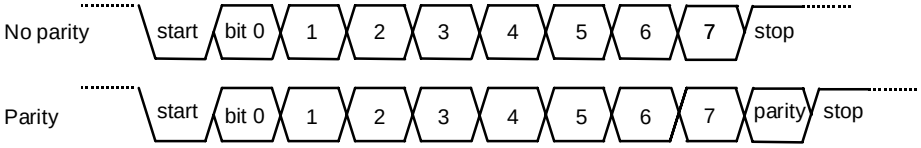
- Mode 0 (I/O Interface Mode)



- Mode 1 (7-Bit UART Mode)



- Mode 2 (8-Bit UART Mode)



- Mode 3 (9-Bit UART Mode)

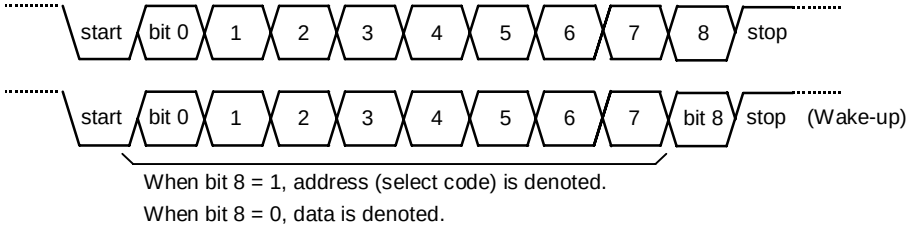


Figure 3.9.1 Data formats

3.9.1 Block diagrams

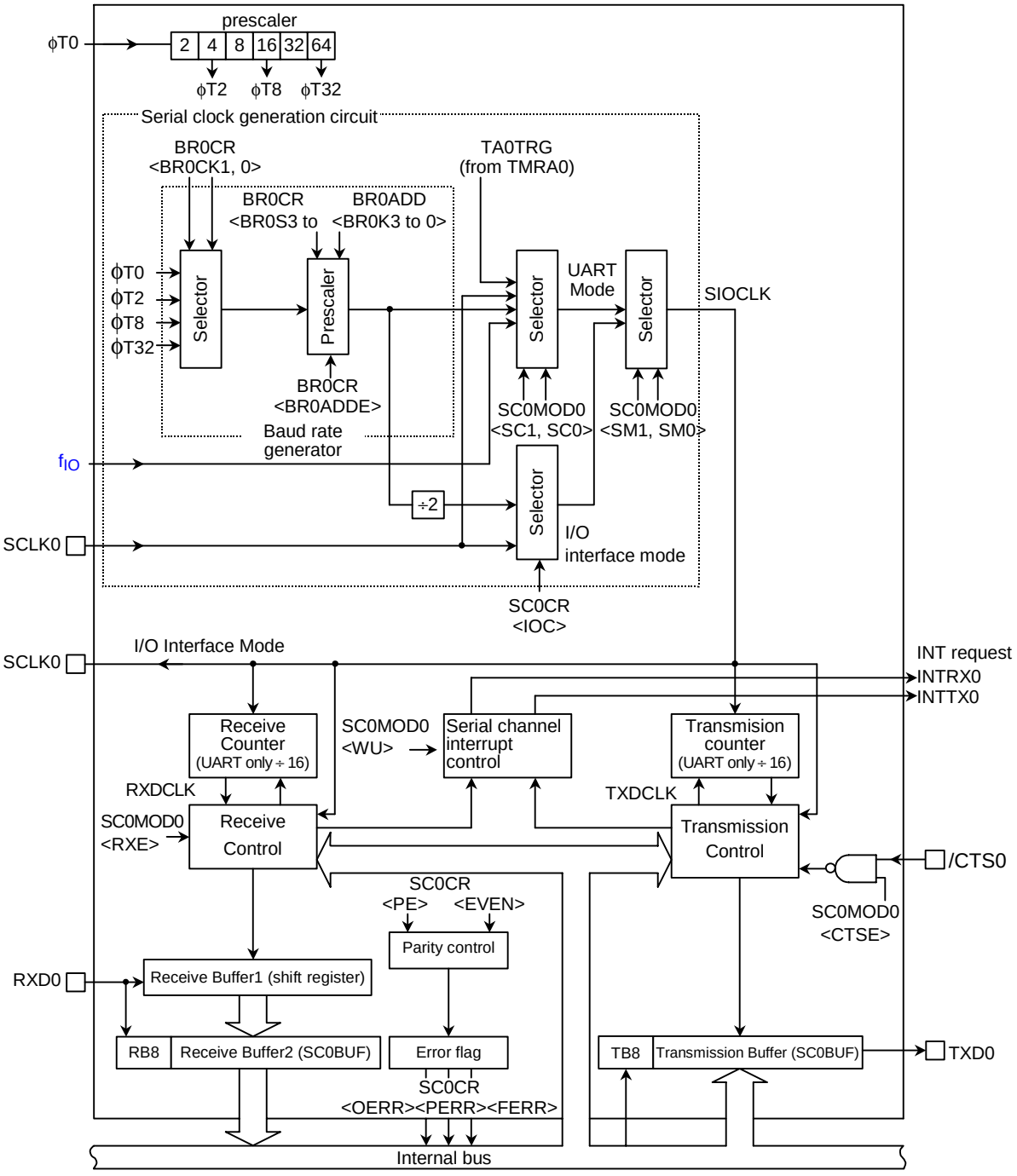


Figure 3.9.2 Block diagram of the Serial Channel 0

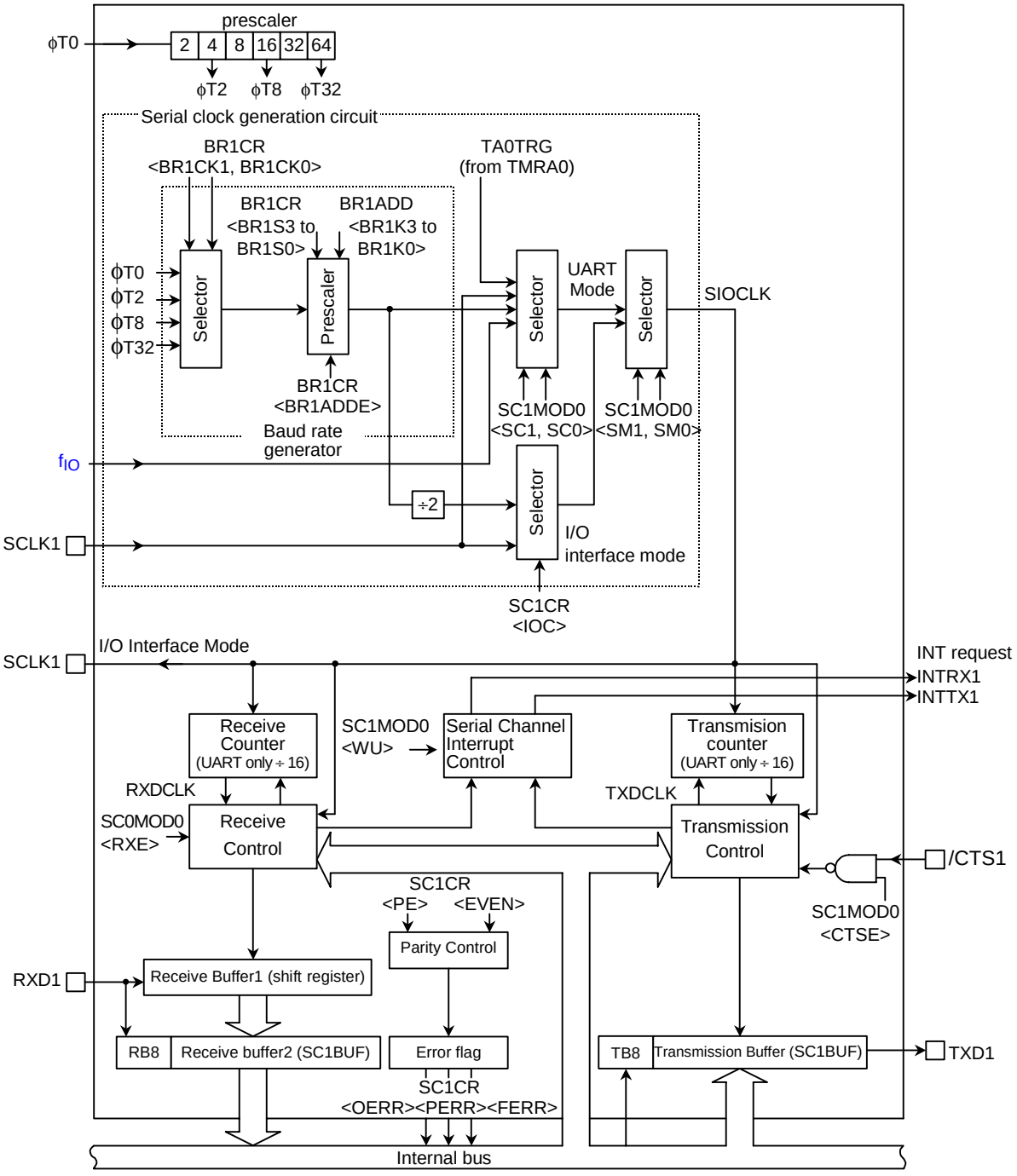


Figure 3.9.3 Block diagram of the Serial Channel 1

3.9.2 Operation for each circuit

(1) Prescaler, Prescaler clock select

There is a 6-bit prescaler for waking serial clock.

The prescaler can be run by selecting the baud rate generator as the waking serial clock. $_{SYS}$

Table 3.9.2 shows prescaler clock resolution into the baud rate generator.

Table 3.9.2 Prescaler Clock Resolution to Baud Rate Generator

System clock <SYSCK>	Clock gear <GEAR2 to 0>	Clock resolution			
		ϕT0	ϕT2	ϕT8	ϕT32
1 (fs)	XXX	$fs/2^2$	$fs/2^4$	$fs/2^6$	$fs/2^8$
0 (fc)	000 (fc)	$f_{SYS}/2^2$	$f_{SYS}/2^4$	$f_{SYS}/2^6$	$f_{SYS}/2^8$
	001 ($fc/2$)	$f_{SYS}/2^3$	$f_{SYS}/2^5$	$f_{SYS}/2^7$	$f_{SYS}/2^9$
	010 ($fc/4$)	$f_{SYS}/2^4$	$f_{SYS}/2^6$	$f_{SYS}/2^8$	$f_{SYS}/2^{10}$
	011 ($fc/8$)	$f_{SYS}/2^5$	$f_{SYS}/2^7$	$f_{SYS}/2^9$	$f_{SYS}/2^{11}$
	100 ($fc/16$)	$f_{SYS}/2^6$	$f_{SYS}/2^8$	$f_{SYS}/2^{10}$	$f_{SYS}/2^{12}$

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 a circuit, which generates transmission and receiving clocks that determine 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 to BR0CK0> 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 or 16 values, determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3 to BR0S0> and BR0ADD<BR0K3 to BR0K0>.

•In UART Mode

(1) When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3 to BR0K0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CK<BR0S3 to BR0S0>. (N = 1, 2, 3 ⋯ 16)

(2) 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 to BR0S0> (N = 2, 3 ⋯ 15) and the value of K set in BR0ADD<BR0K3 to R0K0> (K = 1, 2, 3 ⋯ 15)

Note: If N = 1 or N = 16, the N + (16 – K) / 16 division function is disabled. Set BR0CR<BR0ADDE> to 0.

•In I/O Interface Mode

The N + (16 – K) / 16 division function is not available in I/O Interface Mode. Set 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 (fsys) is 19.6608 MHz, the input clock is φT2 (fsys/16), the frequency divider N (BR0CR<BR0S3 to BR0S0>) = 8, and BR0CR<BR0ADDE> = 0, the baud rate in UART Mode is as follows:

$$\text{Baud Rate} = \frac{\text{fsys}/16}{8} \div 16$$

$$= 19.6608 \times 10^6 \div 16 \div 8 \div 16 = 9600 \text{ (bps)}$$

Note: The N + (16 – K) / 16 division function is disabled and setting BR0ADD<BR0K3 to BR0K0> is invalid.

•N+(16-K)/16 divider (UART Mode only)

Accordingly, when the source clock frequency (fsys) = 15.9744 MHz, the input clock is φT2 (fsys/16), the frequency divider N (BR0CR<BR0S3 to BR0S0>) = 6, K (BR0ADD<BR0K3 to BR0K0>) = 8, and BR0CR <BR0ADDE> = 1, the baud rate in UART Mode is as follows:

$$\text{Baud Rate} = \frac{\text{fsys}/16}{6 + (16 - 8)/16} \div 16$$

$$= 15.9744 \times 10^6 \div 16 \div (6+8/16) \div 16 = 9600 \text{ (bps)}$$

Table 3.9.3 to 4 show examples of UART Mode transfer rates.

Additionally, the external clock input is available in the serial clock. (Serial Channels 0 & 1). 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)> = f_{SYS} / 4

•In I/O Interface Mode

Baud rate = external clock input frequency

It is necessary to satisfy (external clock input cycle) >= f_{SYS} / 16

Table 3.9.3 Selection of Transfer Rate(1)

(when baud rate generator Is used and BR0CR <BR0ADDE> = 0)

Unit (kbps)

fsys [MHz]	Input Clock / Frequency Divider	ϕT0 (4/fsys)	ϕT2 (16/fsys)	ϕT8 (64/fsys)	ϕT32 (256/fsys)
18.432000	15	19.200	4.800	1.200	0.300
19.660800	8	38.400	9.600	2.400	0.600
↑	16	19.200	4.800	1.200	0.300

Note: Transfer rates in I/O Interface Mode are eight times faster than the values given above.

Table 3.9.4 Selection of Transfer Rate(2)

(When TMRA0 with input Clock ϕT1 is used)

Unit (kbps)

fsys / TA0REG	20 MHz	19.6608 MHz	16 MHz
02H		76.8	62.5
04H		38.4	31.25
05H	31.25		
08H		19.2	
10H		9.6	

Method for calculating the transfer rate (when timer 0 is used):

$$\text{Transfer rate} = \frac{\text{fsys}}{\text{TA0REG} \times \underline{8} \times 16}$$

↑ (when TMRA0 (input clock ϕT1) is used)

Note1) The TMRA0 match detect signal cannot be used as the transfer clock in I/O Interface Mode.

Note2) This table is calculated when fc is selected as a system clock, the clock gear is set to fc.

(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, SC0> 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 edge of the shift clock which is output on the SCLK0 pin.
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); this 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.

(7) Transmission counter

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.

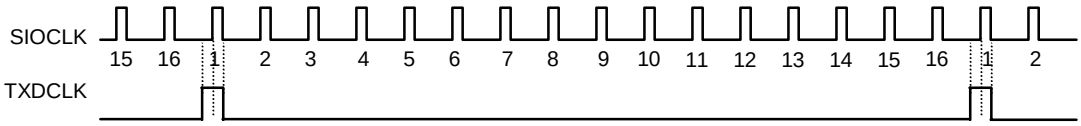


Figure 3.9.4 Generation of the transmission clock

(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 of the shift clock which is output on the SCLK0 pin.

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, generating a transmission shift clock TXDSFT.

Handshake function

Serial Channels 0, 1 each has a /CTS 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 /CTS0 pin foes High on completion of the current data send, data transmission is halted until the /CTS0 pin foes 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 /RTS pin, a handshake function can be easily configured by setting any port assigned to be the /RTS function. The /RTS should be output "High" to request send data halt after data receive is completed by software in the RXD interrupt routine.

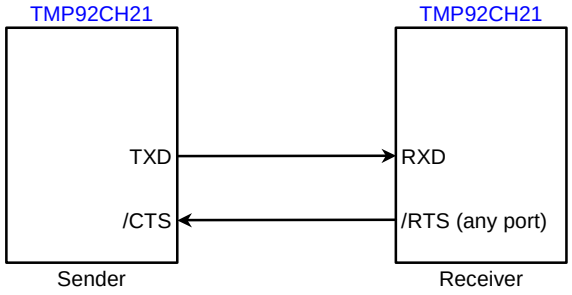
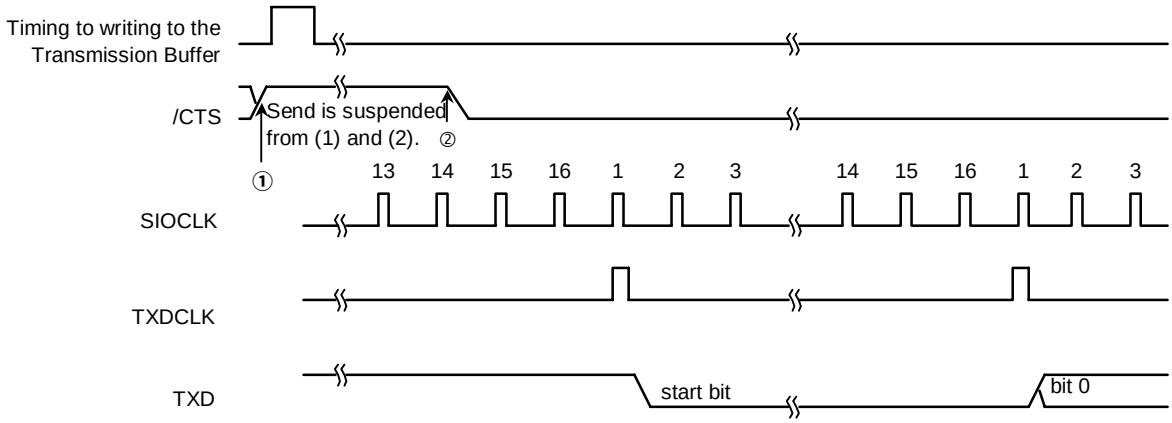


Figure 3.9.5 Handshake function



Note 1: If the /CTS signal goes High during transmission, no more data will be sent after completion of the current transmission.

Note 2: Transmission starts on the first falling edge of the TXDCLK clock after the /CTS signal has fallen.

Figure 3.9.6 /CTS (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

4) Set to disable receiving (write ‘0’ to SC0MOD0<RXE>)

5) Wait to terminate current frame

6) Read receiving buffer

7) Read error flag

8) Set to enable receiving (write ‘1’ to SC0MOD0<RXE>)

9) Request to transmit again

10) Other

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.

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

① 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 last bit (parity bit)
Overrun error timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit

Note: 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.

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

② I/O interface

Transmission Interrupt timing	SCLK Output Mode	Immediately after rise of last SCLK signal. (See figure 3.9 19.)
	SCLK Input Mode	Immediately after rise of last SCLK signal Rising Mode, or immediately after fall in Falling Mode. (See figure 3.9 20.)
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 3.9 21.)
	SCLK Input Mode	Timing used to transfer received data to Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See figure 3.9 22.)

3.9.3 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	Timer 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 when 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) Enable

0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Figure 3.9.7 Serial Mode Control Register (channel 0, SC0MOD0)

SC1MOD0 (120AH)

	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 (for UART)

00	Timer TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{IO}
11	External clock (SCLK1 input)

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 when 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) enable

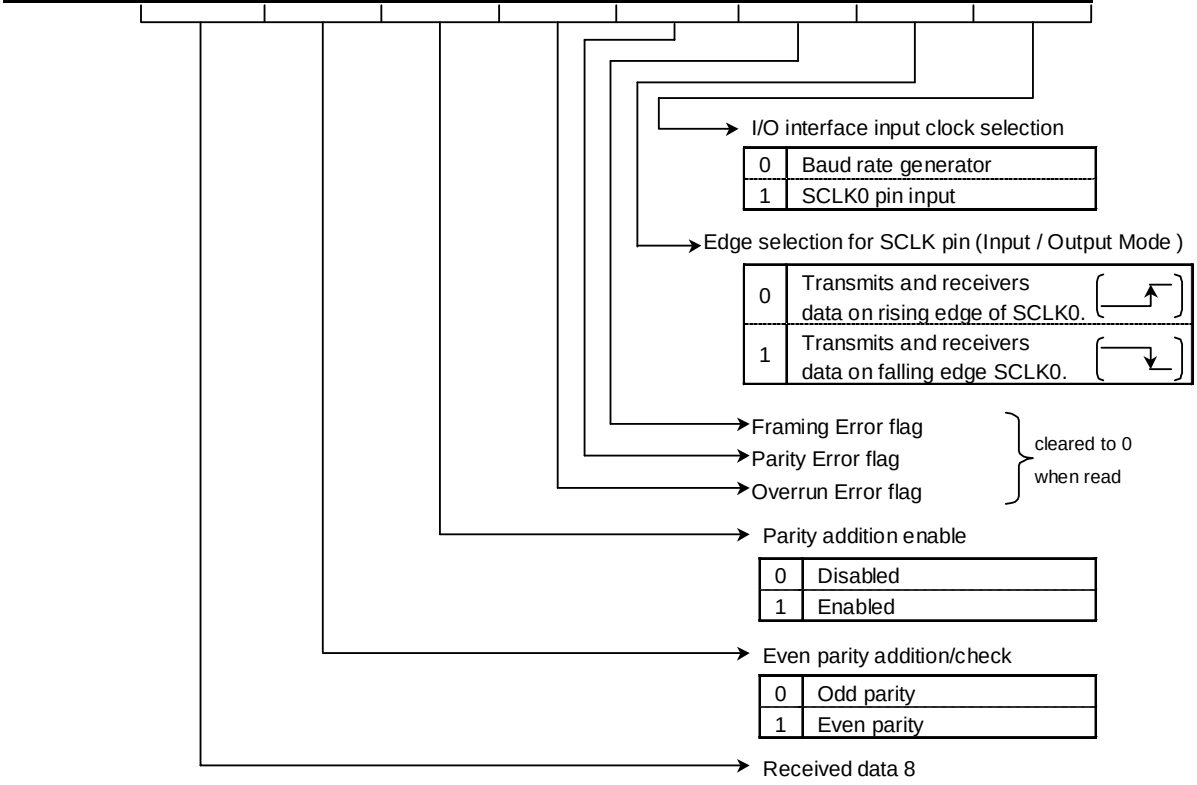
0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Figure 3.9.8 Serial Mode Control Register (channel 1, SC1MOD0)

SC0CR (1201H)

	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		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		



Note: As all error flags are cleared after reading do not test only a single bit with a bit-testing instruction.

Figure 3.9.9 Serial Control Register (channel 0, SC0CR)

SC1CR (1209H)

	7	6	5	4	3	2	1	0
bit symbol	BR8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when)			R/W	
After Reset	0	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: SCLK1 ↑ 1: SCLK1 ↓	0: baud rate generator 1: SCLK1 pin input
				Overrun	Parity	Framing		

I/O interface input clock select

0	Baud rate generator
1	SCLK1 pin input

Edge selection for SCLK pin (input / output mode)

0	Transmits and receives data on rising edge of SCLK1. ↑
1	Transmits and receives data on falling edge of SCLK1. ↓

Framing Error flag
Parity Error flag
Overrun Error flag
} cleared to Zero when read

Parity addition enable

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data bit 8

Note: As all error flags are cleared after reading do not test only a single bit with a bit-testing instruction.

Figure 3.9.10 Serial Control Register (channel 1, SC1CR)

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 fixed to "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

	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
BR0CR <BR0S3:0> / BR0ADD <BR0K3:0>	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	

Note 1: Set BR0CR <BR0ADDE> to "1" after setting K (K = 1 to 15) to BR0ADD<BR0K3 to 0> when + (16 – K)/16 division function is used.

Note 2: + (16 – K)/16 division function is possible to use in only UART mode.
Set BR0CR <BR0ADDE> to "0" and disable + (16 – K)/16 division function in I/O interface mode.

Figure 3.9.11 Baud rate generator control (channel 0, BR0CR, BR0ADD)

BR1CR (120BH)

	7	6	5	4	3	2	1	0
bit Symbol	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	(Note) Always fixed to "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	Disabled
1	Enabled

Input clock selection for baud rate generator

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

BR1ADD (120CH)

	7	6	5	4	3	2	1	0
Bit symbol					BR1K3	BR1K2	BR1K1	BR1K0
Read/Write					R/W			
After Reset					0	0	0	0
Function					Set frequency divisor K (divided by N + (16 – K)/16)			

Baud rate generator frequency divisor setting

BR1CR <BR1S3 to BR1S0> / BR1ADD <BR1K3 to BR1K0>	BR1CR<BR1ADDE> = 1		BR1CR<BR1ADDE> = 0
0000	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	

Note 1: Set BR1CR <BR1ADDE> to "1" after setting K (K = 1 to 15) to BR1ADD <BR1K3 to 0> when + (16 – K) / 16 division function is used.

Note 2: + (16 – K) / 16 division functions is possible to use in only UART mode. Set BR1CR <BR1ADDE> to "0" and disable + (16 – K) / 16 division in I/O interface mode.

Figure 3.9.12 Baud rate generator control (channel 1, BR1CR, BR1ADD)

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	(Reveiving)

Note: Prohibit read modify write for SC0BUF.

Figure 3.9.13 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 3.9.14 Serial Mode Control Register 1 (channel 0, SC0MOD1)

SC1BUF (1208H)

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 SC1BUF.

Figure 3.9.15 Serial Transmission/Receiving Buffer Registers (channel 1, SC1BUF)

SC1MOD1 (120DH)

	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 3.9.16 Serial Mode Control Register 1 (channel 1, SC1MOD1)

3.9.4 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.

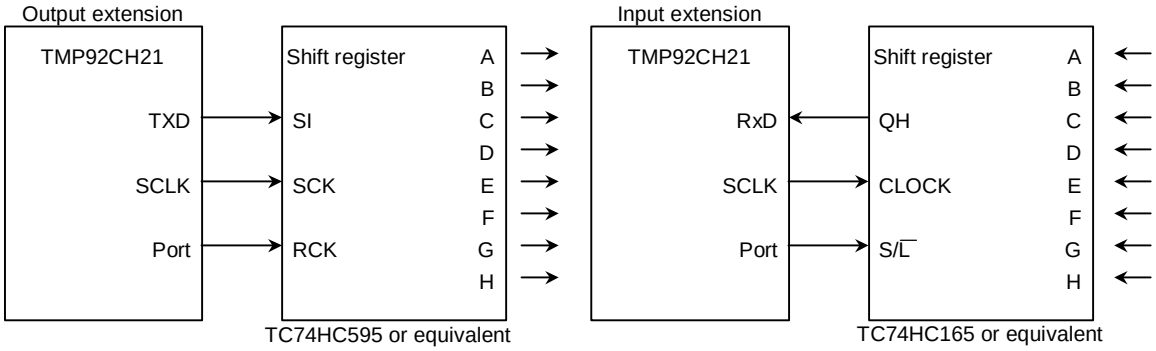


Figure 3.9.17 SCLK Output Mode connection example

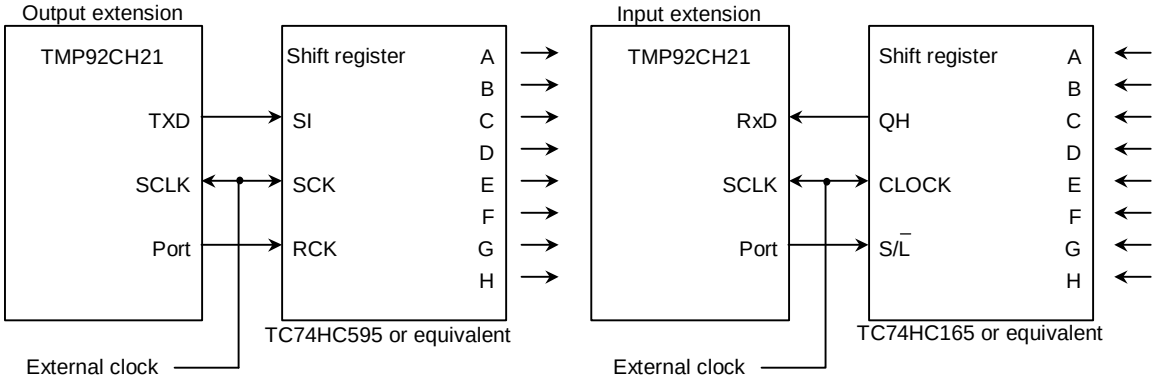


Figure 3.9.18 Example of SCLK Input Mode Connection

① 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.

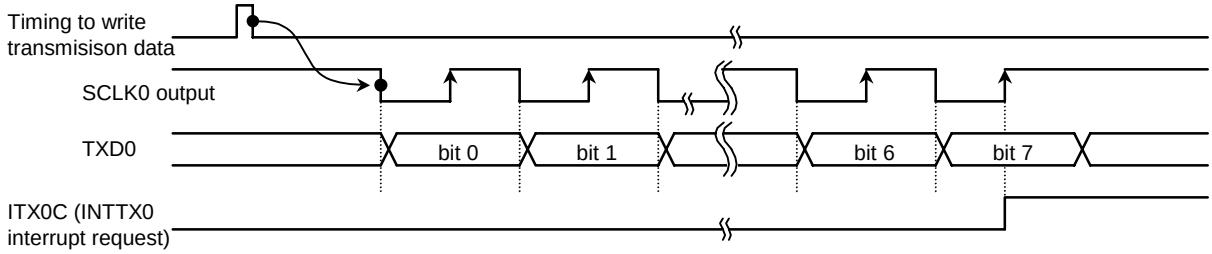


Figure 3.9.19 Transmitting Operation in I/O Interface Mode (SCLK0 Output Mode) (Channel 0)

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.

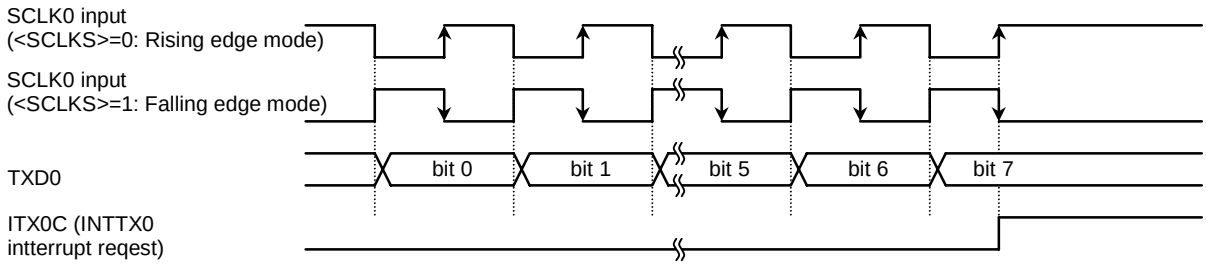


Figure 3.9.20 Transmitting Operation in I/O Interface Mode (SCLK0 Input Mode) (channel 0)

② 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.

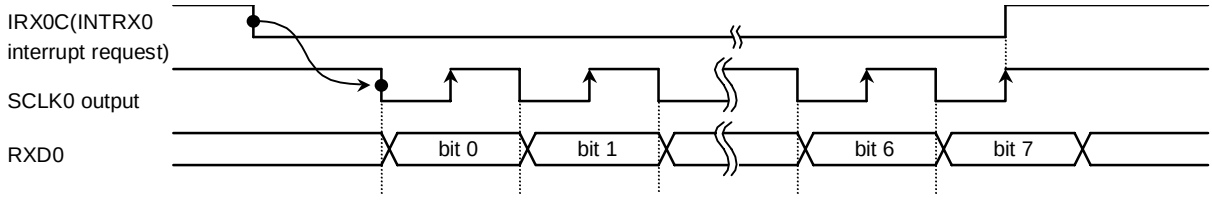


Figure 3.9.27 Receiving operation in I/O Interface Mode (SCLK0 Output Mode)

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.

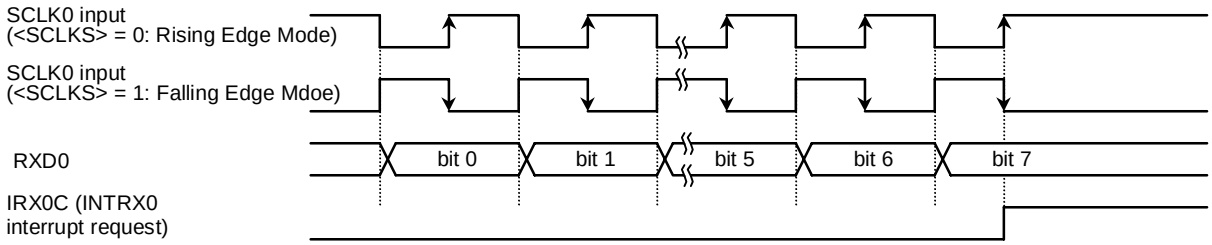


Figure 3.9.28 Receiving Operation in I/O interface Mode (SCLK0 Input Mode)

Note: The system must be put in the Receive Enable state (SC0MOD0<RXE> = 1) before data can be received.

③ 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. 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
fc = 19.6608 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.
PFCR	–	–	–	–	–	1	0	1	Set PF0, PF1 and PF2 to function as the TXD0, RXD0
PFFC	–	–	–	–	–	1	0	1	and SCLK0 pins respectively
SC0MOD0	0	0	0	0	0	0	0	0	select I/O Interface Mode.
SC0MOD1	1	1	0	0	0	0	0	0	Select Full Duplex Mode.
SC0CR	0	0	0	0	0	0	0	0	Sclk_out, transmit on negative edge, receive on positive edge
BR0CR	0	0	0	1	1	0	0	0	Baud rate = 9600 bps
SC0MOD0	0	0	1	0	0	0	0	0	Enable receiving
SC0BUF	*	*	*	*	*	*	*	*	Set the transmit data and start.

INTTX0 interrupt routine

Acc	←	SC0BUF							Read the receiving buffer.
SC0BUF	*	*	*	*	*	*	*	*	Set the next transmit data.

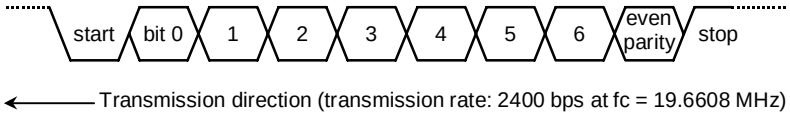
Note: X = Don't care; "–" = No change

(2) Mode 1 (7-bit UART Mode)

7-Bit UART Mode is selected by setting the Serial Channel Mode Register SC0MOD0<SM1,SM0> 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.



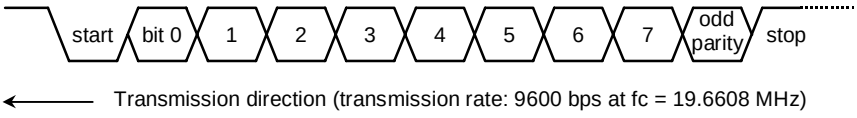
		7	6	5	4	3	2	1	0	
PFCR	←	–	–	–	–	–	–	–	1	Set PF0 to function as the TXD0 pin.
PFFC	←	–	–	–	–	–	–	–	1	
SC0MOD0	←	X	0	–	X	0	1	0	1	Select 7-Bit UART Mode.
SC0CR	←	X	1	1	X	X	X	0	0	Add even parity.
BR0CR	←	0	0	1	0	1	0	0	0	Set the transfer rate to 2400 bps.
INTES0	←	X	1	0	0	–	–	–	–	Enable the INTTX0 interrupt and set it to Interrupt Level 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

Note: 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.



Main settings

		7	6	5	4	3	2	1	0	
PFCR	←	–	–	–	–	–	–	0	–	Set PF1 to function as the RXD0 pin.
PFFC		–	–	–	–	–	–	0	_	
SC0MOD0	←	–	0	1	X	1	0	0	1	Enable receiving in 8-Bit UART Mode.
SC0CR	←	X	0	1	X	X	X	0	0	Add odd parity.
BR0CR	←	0	0	0	1	1	0	0	0	Set the transfer rate to 9600 bps.
INTES0	←	–	–	–	–	X	1	0	0	Enable the INTTX0 interrupt and set it to Interrupt Level 4.

Interrupt processing

Acc	← SC0CR AND 00011100	Check for errors.
if Acc	≠ 0 then ERROR	
Acc	← SC0BUF	Read the received data.

Note: X = Don't care; "–" = No change

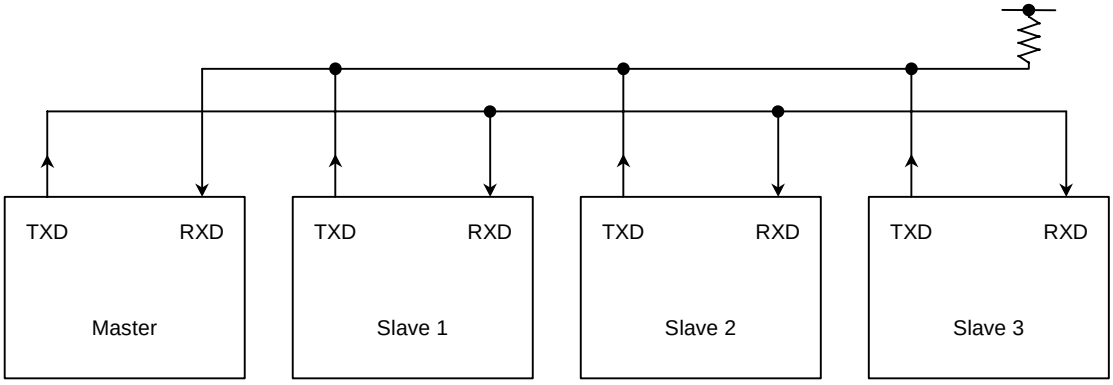
(4) Mode 3 (9-Bit UART Mode)

9-Bit UART Mode is selected by setting SC0MOD0<SM1,SM0> 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.

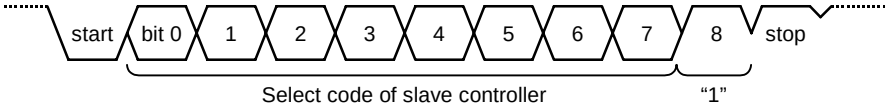
Figure 3.9.29 Serial Link using Wake-up function

Protocol

① Select 9-Bit UART Mode on the master and slave controllers.

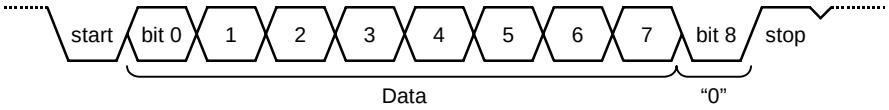
② Set the SC0MOD0<WU> bit on each slave controller to 1 to enable data receiving.

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.



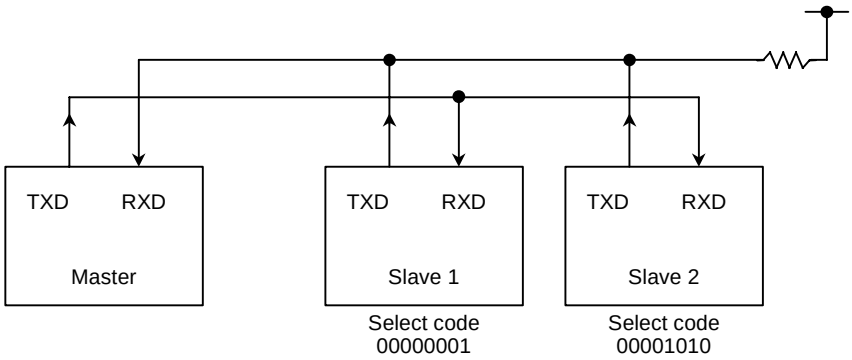
④ Each slave controller receives the above frame. Each controller checks the above select code against its own select code. The controller whose code matches clears its <WU> bit to 0.

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.



Since Serial Channels 0 and 1 operate in exactly the same way, Channel 0 only is used for the purposes of this explanation.

•Setting the master controller

Main

```
PFCR    ← – – – – – – 0 1   } Set PF0 and PF1 to function as the TXD0 and RXD0 pins respectively.
PFFC    ← – – – – – – 0 1   }
INTES0  ← 1 1 0 0 1 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 fIO 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.

INTTX0 interrupt

SC0MOD0 ← 0 – – – – – – –   Set TB8 to 0.
SC0BUF  ← * * * * * * * *   Set data for transmission.
```

•Setting the slave controller

Main

```
PFCR    ← – – – – – – 0 1   } Select PF1 and PF0 to function as the RXD0 and TXD0 pins
PFFC    ← – – – – – – 0 1   } respectively (open-drain output).
PFFC2   ← X X X X X X X 1   }
INTES0  ← 1 1 0 1 1 1 1 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 fSYS as the
                            transfer clock.

INTRX0 interrupt

Acc ← SC0BUF
if Acc = select code
Then SC0MOD0 ← - - - 0 - - - - Clear <WU> to 0.
```

3.9.5 Support for IrDA

SIO0 includes support for the IrDA 1.0 infrared data communication specification. Figure 3.9.30 shows the block diagram.

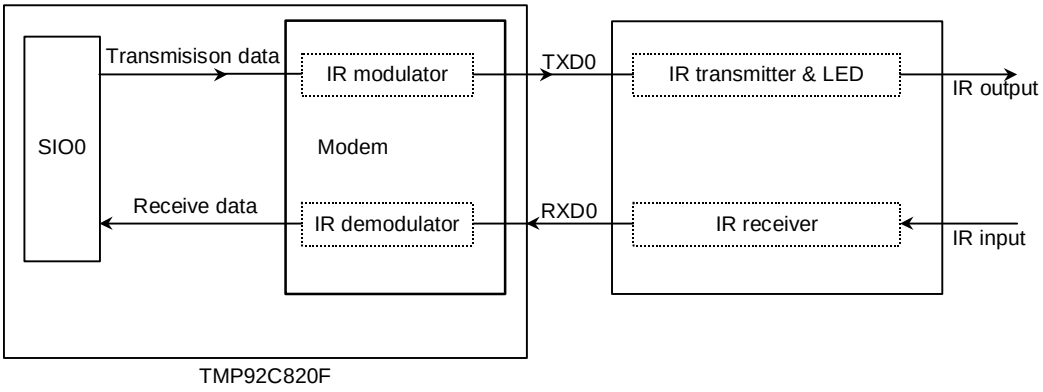


Figure 3.9.30 Block Diagram

(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.

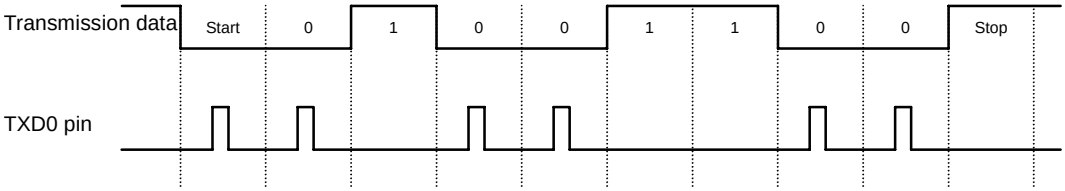


Figure 3.9.31 Transmission example

(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>.

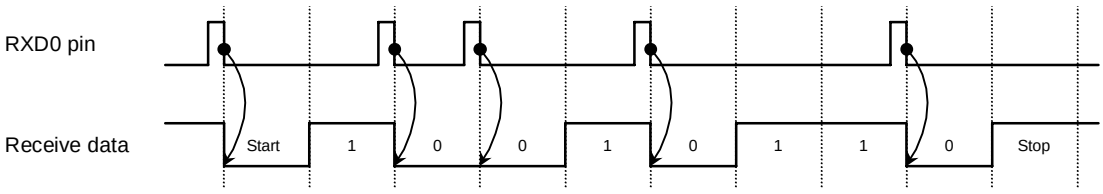


Figure 3.9.32 Receiving example

(3) Data format

The data format is fixed as follows:

- Data length: 8-bit
- Parity bits: none
- Stop bits: 1bit

(4) SFR

Figure 3.9.21 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 and receiving for SIO0 ; The modem operates as follows:
• SIO0 starts transmitting.
• IR receiver starts receiving.

(5) Notes

1. Baud rate for IrDA

When IrDA is operated, set 01 to SC0MOD0<SC1 to 0> to generate baud-rate.

The setting except above(TA0TRG, f_{IO} and SCLK0-input) can not be used.

2. The pulse width for transmittion

The IrDA 1.0 specification is defined in Table 3.9.2.

Table 3.9.2 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 pulse width is defined 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 TMP92CH21 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 except a special condition which is explained (6) Using IrDA 115.2Kbps with USB .

Also when the 38.4 kbps and 1/16 pulse width, + (16-K)/16 division function can not be used.

Table 3.9.3 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

×: Can not be used (16-K)/16 division function

–: Can not be set to 1/16 pulse width

(note): Can be used (16-K)/16 division function at a special condition

(6) Using IrDA 115.2Kbps with USB

When the USB is used in the system, f_{OSCH} should be 9.0MHz. In this case, the IrDA can not be realized 115.2Kbps without using (16-K)/16 division function.

Therefore, only for this case, following condition is allowed to use.

(Setting condition)

- f_{OSCH} = 9.0MHz, PLL on ----→ f_{FPH} = 36MHz, f_{USB} = 48MHz
- Clock for baud rate generator = φT0
- Divided value for baud rate generator = 2 + (16-9)/16
- Pulse width = 3/16

(Calculation result)

- Baud rate = 36MHz / 128 / (2+7/16) = 115.38Kbps
  This baud rate include +0.156% error, but IrDA specficication is within ±0.87%.
- Pulse width = (1/281.25Kbps)*((2*(1/16) + 3*(2/16)) = 1.777us
  This pulse width is greater than 1.41us(IrDA specficication) .

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 4x × (value + 1) Can be set : 1 to 4 Can not be set : 0, 15			

Select receive pulse width

Formula: Effective pulse width ≥ 4x × (value + 1)

$x = 1/f_{FPH}$

0000	Cannot be set
0001	Equal or more than 8x + 100ns
to	
1110	Equal or more than 60x + 100ns
1111	Can not be set

Receive operation

0	Disabled
1	Enabled

Transmit operation

0	Disabled
1	Enabled

Select transmit pulse width

0	3/16
1	1/16

Figure 3.9.21 IrDA Control Register

3.10 USB Controller

3.10.1 Outline

This USB Controller (USBC) is designed for various serial links to construct USB-system.

The outline is as follows:

(1) Compliant with USB rev1.1

(2) Full-speed (12Mbps) (not supported Low-speed (1.5Mbps))

(3) Auto Bus Enumeration with 384Byte Descriptor-RAM

(4) Supported 3 kinds of transfer type: Control, Interrupt and Bulk

Endpoint-0: Control 64Bytes*1 -FIFO
Endpoint-1: BULK(out) 64Bytes*2 -FIFO
Endpoint-2: BULK(in) 64Bytes*2 -FIFO
Endpoint-3: Interrupt(in) 8Bytes*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

(Note) TMP92CH21 don't have a pull-up pin for D+-pin. So, add switch at external and control it by using voluntary output port and software.

3.10.1.1 System Configuration

The USB Controller (USBC) is consist 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-RAM
3) USB Tranceiver

About above 1) is explained at 3.10.2, and 2) is 3.10.3.

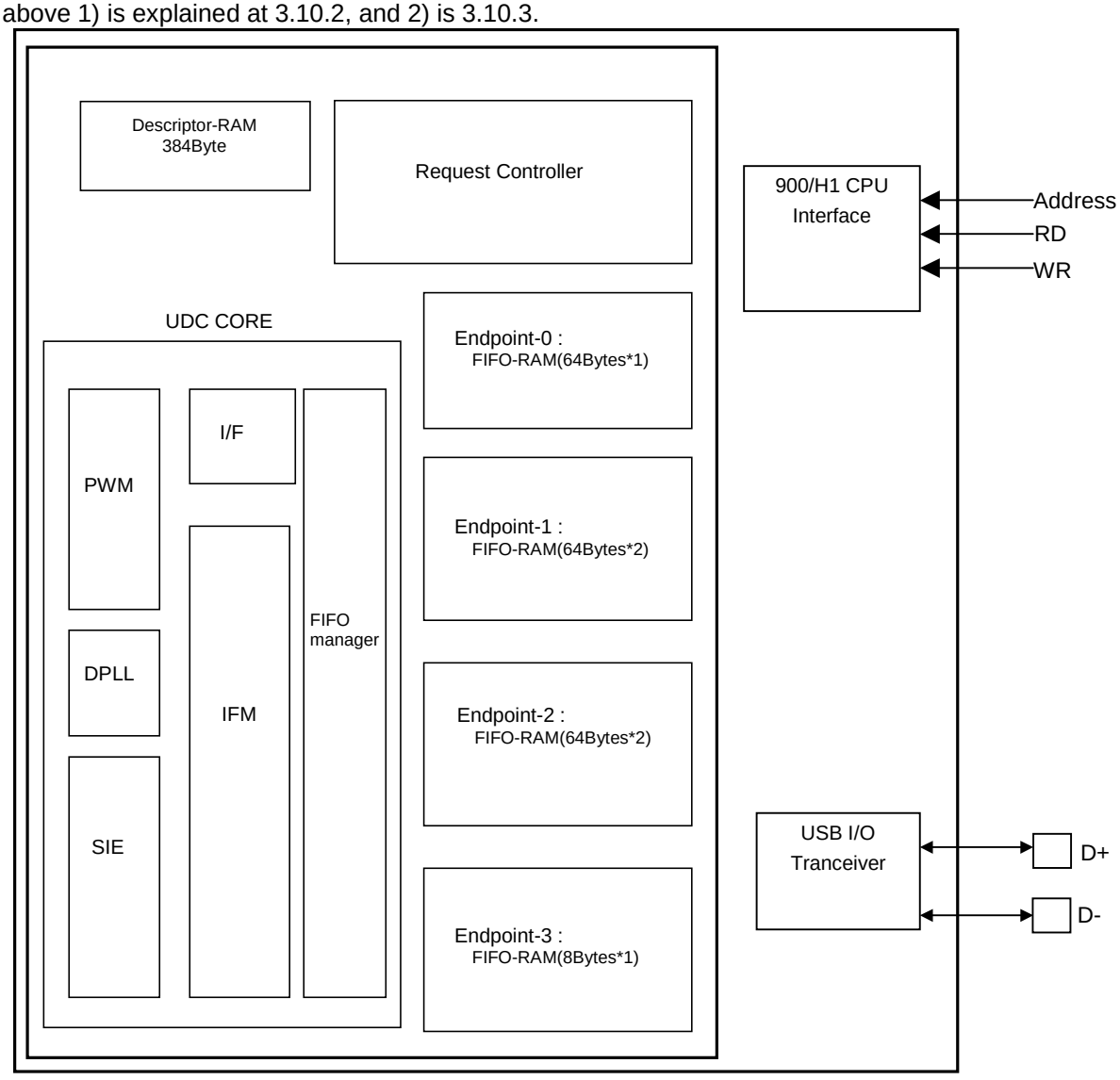


Figure3.10.1 USBC Block diagram

3.10.2 900/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 USB Controller) generation
- A bridge for SFR
- USB clock control (48MHz)

3.10.2.1 SFRs

The 900/H1 CPU I/F have following SFRs to control UDC and USB-tranceiver.

- 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 3.10.1 900/H1 CPU I/F SFRs

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

3.10.2.2 USBCR1 register

This register is used to set USB-clock enable, tranceiver enable etc.

USBCR1 (07F8H) After reset: 00***010B

	7	6	5	4	3	2	1	0
bit symbol	TRNS_USE	WAKEUP	-	-	-	-	SPEED	USB CLKE
R/W	R/W	R/W				Write '0'	R/W	R/W

・TRNS_USE (bit7)

0: Disable internal tranceiver
1: Enable internal tranceiver

Set to “1” for TMP92CH21.

・WAKEUP (bit6)

0: ---
1: Start remort wake up function

When the remort wakeup function is needed, at first check the Current_Config<REMOTE WAKEUP>. If the <REMOTE WAKEUP>=”1”, Write “1” and “0” to <WAKEUP> after checking EP0, 1,2,3_STATUS<SUSPEND>=”1”(SUSPEND status). By this, remort 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 (12MHz)
0:reserved

This bit selects USB speed.
Set to “1” for TMP92CH21.

・USBCLKE (bit0)

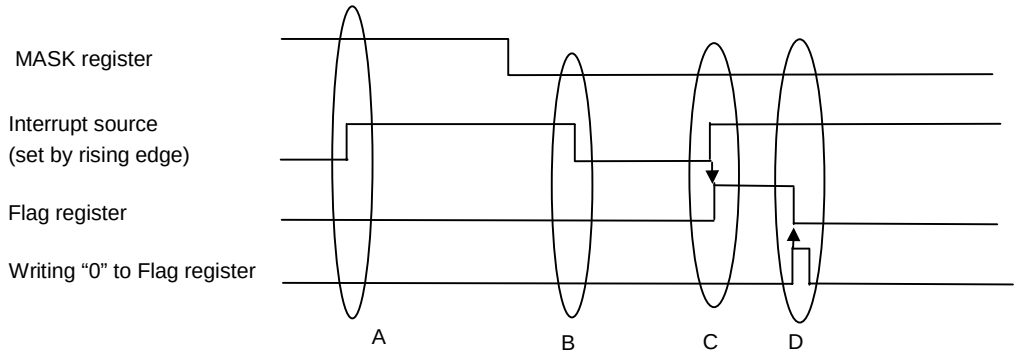
0: Disable USB clock
1: Enable USB clock

This bit controls to supply USB clock.
Enable USB clock after PLL is locked up.
When both USB clock and PLL are stopped, stop USB clock at first.

3.10.2.3 USBINTFRn,MRn register

These SFR controls to generate INTUSB(only one interrupt to CPU) because UDC outputs 25 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 bit=”0” and interrupt souce changes “0”→”1”.
D：The flag register is reset to “0” by writing “0” to Flag register.

Note1) Both “INTUSB generated number” and “Bit number which is set to FLAG-reg” are not always equal.
Therefore, there is a case once-INTUSB and many numbers interrupt source.
In the INTUSB-interrupt routine, after checking FLAG-reg, clear FLAG-reg and clear interrupt request flag in INTEUSB-register.

Note2) When USBINTMRn or USBINTFRn is written, disable INTUSB (write 00H to INTEUSB-register) before.

USBINTFR1 (07F0H) After reset: 0000 00**B

	7	6	5	4	3	2	1	0
bit symbol	NT_URST_ST	NT_URST_EN	INT_SUS	INT_RESUME	NT_CLKSTOP	INT_CLKON	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W		

・INT_URST_STR (bit7)
0:INT_URST_STR not generated
1:INT_URST_STR generated

This is a flag register for INT_URST_STR (USB starting interrupt by USB_RESET).
This is set to “1” when USB controller started to receive “bus reset” signal from USB-host.
An application program has to initialize whole USBC by this interrupt.

・INT_URST_END (bit6)
0:INT_URST_END not generated
1:INT_URST_END generated

This is a flag register for INT_URST_END (USB termination interrupt by USB_RESET).
This is set to “1” when USB controller received “bus reset termination” signal from USB-host.

・INT_SUS (bit5)
0:INT_SUS not generated
1:INT_SUS generated
This is a flag register for INT_SUS (USB suspend interrupt).
This is set to “1” when USB changed to “suspend status”.

・INT_RESUME (bit4)
0:INT_RESUME not generated
1:INT_RESUME generated
This is a flag register for INT_RESUME (USB resume interrupt).
This is set to “1” when USB changed to “resume status”.

・INT_CLKSTOP (bit3)
0:INT_CLKSTOP not generated
1:INT_CLKSTOP generated
This is a flag register for INT_CLKSTOP (USB enabled to stop clock supplying interrupt).
This is set to “1” when USB changed to “suspend status” and enabled to stop clock supplying.

・INT_CLKON (bit2)
0:INT_CLKON not generated
1:INT_CLKON generated
This is a flag register for INT_CLKON (USB enabled to start clock supplying interrupt).
This is set to “1” when USB changed to “resume status” and enabled to start clock supplying.

USBINTFR2 (07F1H) After reset: 0000 0000B

	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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

USBINTFR3 (07F2H) After reset: 00** ****B

	7	6	5	4	3	2	1	0
bit symbol	EP3_FULL_A	EP3_Empty_A	-	-	-	-	-	-
R/W	R/W	R/W						

EPx_FULL_A/B：

(When transmitting)

This is set to “1” when CPU wrote data to FIFO_A/B.

(When receiving)

This is set to “1” when USBC received data to FIFO_A/B.

EPx_Empty_A/B：

(When transmitting)

This is set to “1” when FIFO became empty after transmittion.

(When receiving)

This is set to “1” when FIFO became empty after CPU read all data from FIFO.

USBINTFR4 (07F3H) After reset: 0000 000*B

	7	6	5	4	3	2	1	0
bit symbol	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

・INT_SETUP (bit7)

0: INT_SETUP not generated

1: INT_SETUP generated

This is a flag register for INT_SETUP (USB setup interrupt).
This is set to “1” when USB received request for needed S/W(software) control from USB host.
By S/W(INT_SETUP routine), at first, read device request of 8-bytes from USBC and execute operation according to each request.

・INT_EP0 (bit6)

0: INT_EP0 not generated

1: INT_EP0 generated

This is a flag register for INT_EP0 (USB received data interrupt on the data phase for Control transfer type).
This is set to “1” when USB received data on the data phase for Control transfer type.
At the Control write transfer type, data reading from FIFO is needed if this interrupt occurred.
At the Control read transfer type, transmission data writing to FIFO is needed if this interrupts occurred.
And some kinds of host may don’t assert “ACK” for last packet in the data stage. In that case, this interrupt can not be generated. So, ignore this interrupt of after last packet data was written in the data stage because the data number to transmit is specified by the host or it depends on the capacity of the device.

• INT_STAS (bit5)

0: INT_STAS not generated

1: INT_STAS generated

This is a flag register for INT_STAS (USB status stage terminated interrupt).
This is set to “1” when the status stage terminated.
If this interrupt is generated, it means the request terminated successfully.
If this interrupt is not generated and INT_SETUP is generated, EP0_STATUS <STAGE_ERR> is set to “1” and it means the request didn’t terminate successfully.

• INT_STASN (bit4)

0: INT_ STASN not generated

1: INT_ STASN generated

This is a flag register for INT_STASN (USB host changed to status stage - interrupt).
This is set to “1” when the USB host changed to status stage at the Control read transfer type.
This interrupt is needed in case of data length is less than wLength(specified by the host).
But if the USB host changed to status stage, this interrupt is always generated because of this signal is designed by using NAK for first packet. So, to avoid this always interrupt, use mask register USBINTMRn. Disable this interrupt before data of last payload will be written.

• INT_EPxN (bit3,2,1)

0: INT_ EPxN not generated

1: INT_ EPxN generated

This is a flag register for INT_EPxN (NAK acknowladge to the USB host - interrupt).
This is set to “1” when the Endpoint1,2 and 3 transmitted NAK.

USBINTMR1 (07F4H) After reset: 1111 11**B

	7	6	5	4	3	2	1	0
bit symbol	SK_URST_ST	SK_URST_EN	MSK_SUS	MSK_RESUME	MSK_CLKSTO	MSK_CLKON	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W		

• MSK_URST_STR (bit7)
0：USBINTFR1<INT_URST_STR> is not masked.
1：USBINTFR1<INT_USRT_STR> is masked.
This is a mask register for USBINTFR1<INT_URST_STR>.

• MSK_URST_END (bit6)
0：USBINTFR1 <INT_URST_END> is not masked.
1：USBINTFR1 <INT_URST_END> is masked.
This is a mask register for USBINTFR1 <INT_URST_END> .

• MSK_SUS (bit5)
0：USBINTFR1<INT_SUS> is not masked.
1：USBINTFR1<INT_SUS> is masked.
This is a mask register for USBINTFR1<INT_SUS> .

• MSK_RESUME (bit4)
0：USBINTFR1<INT_RESUME> is not masked.
1：USBINTFR1<INT_RESUME> is masked.
This is a mask register for USBINTFR1<INT_RESUME> .

• MSK_CLKSTOP (bit3)
0：USBINTFR1<INT_CLKSTOP> is not masked.
1：USBINTFR1<INT_CLKSTOP> is masked.
This is a mask register for USBINTFR1<INT_CLKSTOP>.

• MSK_CLKON (bit2)
0：USBINTFR1 <INT_CLKON> is not masked.
1：USBINTFR1 <INT_CLKON> is masked.
This is a mask register for USBINTFR1 <INT_CLKON> .

USBINTMR2 (07F5H) After reset: 1111 1111B

	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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

USBINTMR3 (07F6H) After reset: 11** ****B

	7	6	5	4	3	2	1	0
bit symbol	EP3_MSK_FA	EP3_MSK_EA	-	-	-	-	-	-
R/W	R/W	R/W						

EPx_MSK_FA/FB/EA/EB：

1: Each flag resgiter is not masked.

0: Each flag resgiter is masked.

Note1) The flag registers of EPx_FULL_A/B and EPx_Empty_A/B are not status flag.
Therefore, check DATASET register if the FIFO-status is needed.

USBINTMR4 (07F7H) After reset: 1111 111*B

	7	6	5	4	3	2	1	0
bit symbol	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

• MSK_SETUP (bit7)
0：USBINTFR4<INT_SETUP> is not masked.
1：USBINTFR4<INT_SETUP> is masked.
This is a mask register for USBINTFR4<INT_SETUP> .

• MSK_EP0 (bit6)
0：USBINTFR4<INT_EP0> is not masked.
1：USBINTFR4<INT_EP0> is masked.
This is a mask register for USBINTFR4<INT_EP0> .

• MSK_STAS (bit5)
0：USBINTFR4<INT_STAS> is not masked.
1：USBINTFR4<INT_STAS> is masked.
This is a mask register for USBINTFR4<INT_STAS>.

• MSK_STASN (bit4)
0：USBINTFR4<INT_STASN> is not masked.
1：USBINTFR4<INT_STASN> is masked.
This is a mask register for USBINTFR4<INT_STASN>.

• MSK_EP1N (bit3)
0：USBINTFR4<INT_EP1N> is not masked.
1：USBINTFR4<INT_EP1N> is masked.
This is a mask register for USBINTFR4<INT_EP1N>.

• MSK_EP2N (bit2)
0：USBINTFR4<INT_EP2N> is not masked.
1：USBINTFR4<INT_EP2N> is masked.
This is a mask register for USBINTFR4<INT_EP2N>.

• MSK_EP3N (bit1)
0：USBINTFR4<INT_EP3N> is not masked.
1：USBINTFR4<INT_EP3N> is masked.
This is a mask register for USBINTFR4<INT_EP3N>.

3.10.3 UDC CORE

3.10.3.1 SFRs

The UDC CORE has following SFRs to control UDC and USB-tranceiver.

a) FIFO-RAM
- Endpoint 0-3 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) Set up

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		

Table3.10.1 UDC CORE SFRs (1/2)

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
0789H	R/W	EP1_MODE
078AH	R/W	EP2_MODE
078BH	R/W	EP3_MODE
0790H	R	EP0_STATUS
0791H	R	EP1_STATUS
0792H	R	EP2_STATUS
0793H	R	EP3_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
07A1H	R	EP1_SIZE_L_B
07A2H	R	EP2_SIZE_L_B
07A3H	R	EP3_SIZE_L_B
07A9H	R	EP1_SIZE_H_A
07AAH	R	EP2_SIZE_H_A
07ABH	R	EP3_SIZE_H_A
07B1H	R	EP1_SIZE_H_B
07B2H	R	EP2_SIZE_H_B
07B3H	R	EP3_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	StandardRequest
07CBH	R	Request
07CCH	R	DataSet1
07CDH	R	DataSet2
07CEH	R	USB_STATE
07CFH	W	EOP
07D0H	W	Command
07D1H	R/W	EPx_SINGLE1
07D3H	R/W	EPx_BCS1

07D6H	R/W	INT_Control

Table3.10.1 UDC CORE SFRs (2/2)

07D8H	R/W	StandardRequestMode
07D9H	R/W	RequestMode
07DEH	W	ID_CONTROL
07DFH	R	ID_STATE
07E0H	R/W	Port_Status
07E1H	R	FRAME_L
07E2H	R	FRAME_H
07E3H	R	ADDRESS
07E4H	R/W	Reserved
07E6H	R/W	USBREADY
07E8H	W	Set Descriptor STALL

3.10.3.2 ENDPOINTx register (x:0-3)

This register is prepared for each endpoint.
This is the window register from or to FIFO-RAM.
In the Auto Bus Enumeration, the Request Controller in USBC set mode, which is defined at endpoint descriptor for each endpoint automatically. By this, each endpoint is set to voluntary direction.

1) 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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

2) 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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

3) 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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

4) 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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note1) 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-byte or 4-byte.

The device request received from the USB host is stored to following 8-bytes registers.
The 8-bytes registers are bmRequestType, bRequest, wValue_L, wValue_H, wIndex_L, wIndex_H, wLength_L and wLength_H. They are updated at every new setup-token from the host receiving.
When the USBC received without error, INT_SETUP-interrupt is asserted and it means the new device request has been received.
And there may be a request which is operated automatically by USBC. It depends on received request. In that case, the USBC don't assert INT_SETUP-interrupt. A request which the USBC is operating now can be checked by reading STANDARD_REQUEST_FLAG and REQUEST_FLAG.

3.10.3.3 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
R/W	R	R	R	R	R	R	R	R

- DIRECTION（bit7） 0 : from Host to Device
  1：from Device to Host
- REQ_TYPE (bit6-5) 0：Standard
  1：Class
  2：Vendor
  3：(reserved)
- RECEPIENT (bit4-0) 0：Device
  1：Interface
  2：Endpoint
  3：etc.
  4-31：(reserved)

3.10.3.4 bRequest register

This register shows the bRequestType-field of device request.

bRequest (07C1H)

	7	6	5	4	3	2	1	0
bit symbol	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
R/W	R	R	R	R	R	R	R	R

(Standard)
0：GET_STATUS
1：CLEAR_FEATURE
2：(reserved)
3：SET_FEATURE
4：(reserved)
5：SET_ADDRESS
6：GET_DESCRIPTOR
7：SET_DESCRIPTOR
8：GET_CONFIGURATION
9：SET_CONFIGURATION
A：GET_INTERFACE
B：SET_INTERFACE
C：SYNCH_FRAME

(Printer class)
0：GET_DEVICE_ID
1：GET_PORT_STATUS
2：SOFT_RESET

3.10.3.5 wValue register

There are 2-registers, the wValue_L register shows the lower-byte of wValue -field of device request and wValue_H register shows upper-byte.

wValue_H (07C3H)
wValue_L (07C2H)

	7	6	5	4	3	2	1	0
bit symbol	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
R/W	R	R	R	R	R	R	R	R

3.10.3.6 wIndex register

There are 2-registers, 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_H (07C5H)
wIndex_L (07C4H)

	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
R/W	R	R	R	R	R	R	R	R

3.10.3.7 wLength register

There are 2-registers, 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_H (07C7H)
wLength_L (07C6H)

	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
R/W	R	R	R	R	R	R	R	R

3.10.3.8 Setup Received register

This register is used for which an application program tells USBC to recognized INT_SETUP interrupt.

Setup Received (07C8H)

	7	6	5	4	3	2	1	0
bit symbol	D7	D6	D5	D4	D3	D2	D1	D0
R/W	W	W	W	W	W	W	W	W

If this register is accessed by an application program, the USBC removes to disabling access to EP0's FIFO-RAM because the USBC 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.

3.10.3.9 Current Config register

This register shows the current value of CONFIG,INTERFACE set by SET_CONFIGURATION and SET_INTERFACE.

Current Config (07C9H)

	7	6	5	4	3	2	1	0
bit symbol	REMOTEWAKI	-	ALTERNATE[1]	ALTERNATE[0]	NTERFACE[1]	NTERFACE[0]	CONFIG[1]	CONFIG[0]
R/W	R	-	R	R	R	R	R	R

• CONFIG(bit1-0)
0：UNCONFIGURED Set to UNCONFIGURED by the host
1：CONFIGURED 1 Set to CONFIGURED 1 by the host
2：CONFIGURED 2 Set to CONFIGURED 2 by the host

• INTERFACE (bit3-2)
0：INTERFACE 0 Set to INTERFACE 0 by the host
1：INTERFACE 1 Set to INTERFACE 1 by the host
2：INTERFACE 2 Set to INTERFACE 2 by the host

• ALTERNATE(bit5-4)
0：ALTERNATE 0 Set to ALTERNATE 0 by the host
1：ALTERNATE 1 Set to ALTERNATE 1 by the host
2：ALTERNATE 2 Set to ALTERNATE 2 by the host

• REMOTE WAKEUP (bit7)
0：disabled Disabled RemoteWakeup by the host
1：enabled Enabled RemoteWakeup by the host

3.10.3.10 Standard Request register

This register shows the Standard Request which is executing now.
A bit which is set to “1” shows current executing request.

Standard Request (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
R/W	R	R	R	R	R	R	R	R

- Bit number
  - 0：GET_STATUS
  - 1：CLEAR_FEATURE
  - 2：SET_FEATURE
  - 3：GET_DESCRIPTOR
  - 4：GET_CONFIGURATION
  - 5：SET_CONFIGURATION
  - 6：GET_INTERFACE
  - 7：SET_INTERFACE

3.10.3.11 Request register

This register shows the Device Request which is executing now.
A bit which is set to “1” shows current 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
R/W	-	R	R	R	R	R	R	R

- Bit number
  - 0：Standard request
  - 1：Standard request(No-support auto Bus Enumeration(SET_DESCRIPTOR, SYNCH_FRAME)
  - 2：Class request
  - 3：Vendor request
  - 4：GET_DEVICE_ID
  - 5：GET_PORT_STATUS
  - 6：SOFT_RESET

3.10.3.12 DATASET register

This register shows whether the data exist or not in FIFO-RAM.
The application program can check it by accessing this register.
In the receiving status, this each bit is set to “1” with generating interrupt after the USBC terminates to receive data from the host, and is reset to “0” after the 1-packet data are read from FIFO-RAM.
In the transmitting status, this each bit is set to “1” after the 1-packet data are written to FIFO-RAM, and is reset to “0” with generating interrupt after the USBC terminates to transmitt data to host.

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
R/W	R	R	R	R	R	R	—	R

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
R/W	R	R	R	R	R	R	R	R

(Note) DATASET1<EP3_DSET_B>,DATASET2 registers are not used at TMP92CH21.

- Single packet mode (DATASET1:Bit0,2,4,6 DATASET2:Bit0,2,4,6)

  These bits show whether the data exist or not in each FIFO-RAM.
  In the receiving mode, a bit which is set to “1” shows that the data to read exist in FIFO-RAM. Read data from FIFO-RAM. The data number to read are shown by reading EPx_SIZE register.
  In the transimitting mode, a bit which is cleared to “0” shows that the CPU is available to write data for payload or less to FIFO-RAM. If each bit is “1”, wait until each one is cleared to “0”. For the case of transmitting short-packet, access EOP register after writing transmittion data to each endpoint.

- Dual packet mode (DATASET1:Bit3,5,7 DATASET2:1,3,5,7)

  These bit are effective in the dual packet mode. This mode have 2 packet FIFO-RAM.
  Each packet(called packet-A,packet-B) has DATASET-bit.
  When the transfer type is isochronous, the packet which is available to access at current frame is shown. This is different from above one. In this case, the other one of A or B is set to “1” and is renew at every new frame.

Note1) In the receiving mode, if the both A and B bits are “1”, read data of the 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 there are spce in FIFO-RAM. So, write data for payload or less to FIFO-RAM.
For the case of transmitting short-packet, write “0” to EOP<EPn_EOPB> after writing data to the FIFO-RAM.
The maximum size which is available to write each packet is same with maximim payload size. If the both A and B bits are “0”, continuous writing of double maximum payload size are available.

Note3) In the transmitting mode,if both A and B packet are empty and EOP<EPn_EOPB> is written to “0”, the NULL-data Is set to FIFO-RAM.
In the single mode, the NULL-data Is also set to FIFO-RAM if the above operation is executed.

3.10.3.13 EPx_STATUS register（x: 0-3）

These are status registers for each endpoint. The <SUSPEND> is common for all endpoint.

EP0,1,2,3_STATUS (0790H,0791H,0792H,0793H)

	7	6	5	4	3	2	1	0
bit symbol	-	TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
R/W	-	R	R	R	R	R	R	R

• TOGGLE Bit (bit6) shows status of toggle sequence bit.
0：TOGGLE BIT0
1：TOGGLE BIT1

• SUSPEND (bit5) shows status of power management of USBC.
0：RESUME In the SUSPEND-status, some limitation about accessing to USBC is needed.
1：SUSPEND For the detail, refer 3.10.9.

• ENDPOINT STATUS (bit4-2) shows status of endpoint of USBC.
The status show about available or not transferring or the result of transferring. These are depend on transfer type.
(For the isochronous transfer type, refer 3.10.6.)

0：READY Receiving:
shows available to receive.
For the endpoint 1 to 3, this register is initialized to “READY” by setting transfer type at SET_CONFIGURATION.
For the endpoint 0, this register is initialized to “READY” by detecting reset from the host.
This is also initialized to “READY” by terminating without error of the status-stage.

Transmitting:
Basically, this is same with “Receiving”.
But only for transmitting, the “READY” is kept by transmitting data correctly to the host and received “NAK”. In this case, EPx_Empty_A or EPx_Empty_B interrupt is generated to show terminating transmittion correctly.

1：DATAIN is set with generating EPx_FULL_A or EPx_FULL_B interrupt when the data is received without error from the host.

2：FULL Refer 3.10.8 (2) Details for the STATUS register.

3：TX_ERR is set when an “ACK” is not received after transmitting data responded IN-token from the host. In this case, an interrupt is not generated. The host will transmitt IN-token to re-try.

4：RX_ERR is set without transmitting “ACK” when an error(like CRC-error) is detected in the data of a received token. In this case, an interrupt is not generated. The host will transmitt data to re-try.

5：BUSY is used only for the control transfer type and set when a token of status-stage is received from the host after terminating data-stage.
When the USBC is ready to terminate status-stage, the USBC returns to READY.

6：STALL shows each endpoint is STALL status.
In this status, the USBC transmits STALL-handshake except for SETUP-token.
In the control endpoint, the USBC returns to READY after received SETUP-token.

In the other endpoint, the USBC returns to READY after initializing command of The FIFO-RAM.

7：INVALID shows each endpoint is UNCONFIGURED status.
In this status, the USBC has no reaction by receiving token from the host.
By reset, all endpoint are INVALID status. Only endpoint 0 returns to READY by receiving USB-reset.
Each endpoint returns to READY by configured.

• FIFO_DISABLE (bit1) shows FIFO-RAM status except for EP0.
0：FIFO enabled If the FIFO-RAM is set to disabled, the USBC transmits NAK
1：FIFO disabled handshake forcibly for the all tranfer. Disabled or enabled is set by COMMAND register. This bit is cleared to “0” when tranfer type is changed.

• STAGE_ERR（bit0） shows a status stage is not terminated correctly.
0：SUCCESS ERROR is set when a status stage is not terminated correctly and
1：ERROR new SETUP-token is received.
This bit is cleared to “0” by reading EP0_STATUS register.
The USBC changes to status stage and wait terminating after EOP register is written. In this case, if an application is needed to confirm that a status stage is terminated correctly, this bit is effective when a new request flag is received.
And this bit is effective to judge that a last request is canceled or not when a new request flag is asserted.

3.10.3.14 EPx_SIZE register（x: 0-3）

These registers have following function.

a) In the receiving, showing data number for 1-packet which was received correctly.
b) In the transmitting, showing payload size. But showing length for the short packet. This register is not needed to read when the USBC is transmitting.
c) Showing dual-packet mode and effective packet.

Each endpoint has H(High)-register which 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-RAM.
And each H/L-register has 2-set for dual-packet mode.
By reset, these are initialized to maximum payload size.

EP1,2,3_SIZE_H_A (07A9H,07AAH,07ABH)
EP1,2,3_SIZE_H_B (07B1H,07B2H,07B3H)

	7	6	5	4	3	2	1	0
bit symbol	—	—	—	—	—	DATASIZE9	DATASIZE8	DATASIZE7
R/W	-	-	-	—	—	R	R	R

EP0,1,2,3_SIZE_L_A (0798H,0799H,079AH,079BH)

	7	6	5	4	3	2	1	0
bit symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
R/W	R	R	R	R	R	R	R	R

EP1,2,3_SIZE_L_B (07A1H,07A2H,07A3H)

	7	6	5	4	3	2	1	0
bit symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
R/W	R	R	R	R	R	R	R	R

(Note) EP3_SIZE_H_B , EP3_SIZE_L_Bregisters are not used at TMP92CH21.

- DATASIZE9-7 (H bit2-0)
- DATASIZE6-0 (L bit6-0)

In receiving, data number for 1-packet received from the host is shown. This is renewed when a data from the host is received with no-error.

- PACKET ACTIVE（L bit7）

In dual-packet mode, an enabled packet for accessing is shown.
In this case, the USBC accesses packet-A and B mutually.
Check this bit after accessing FIFO-RAM.

1：OUT_ENABLE
0：OUT_DISABLE

In receiving endpoint mode, a data is read if this bit is “1”.
In single-packet mode, this bit is no-meaning. The packet-A is always used.

3.10.3.15 FRAME register

This register shows frame-number which is issued with SOF-token from the host and is used for isochronous transfer type.

Each H and L-register shows upper and lower bits.

whether the data exist or not in FIFO-RAM.

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_STS1
R/W	R	R	R	R	-	R	R	R

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]
R/W	R	R	R	R	R	R	R	R

• T[10:7] (H bit7-4)
• T[6:0] (L bit6-0) shows frame-number and renewed when SOF-token is received.

• CREATE (H bit2) shows enabled or disabled of function for SOF auto-generation.
0：DISABLE It is used for the case of receiving-error of SOF-token.
1：ENABLE This function is set by COMMAND-register.
By reset, this bit is initialized to “0”.

• FRAME STATUS (H bit1,0) show the status whether a frame-number shown FRAME-register
0：BEFORE is correct or not. At the LOST-status, a correct frame-number
1：VALID is invalid.
2：LOST If this register is “VALID”, it means a value in FRAME-register is correct.
If this register is “BEFORE”, when using SOF auto-generation, BEFOREcondition displies it from USB-host-contreoller inside that from SOF generation time to receive SOF-token. Right value is it in frame-number that value advanced one value from value displied to FRAME register.

3.10.3.16 ADDRESS register

This register shows device-address which is specified by the host in Bus Enumeration.
By reading this register, a current address can be confirmed.

ADDRESS (07E3H)

	7	6	5	4	3	2	1	0
bit symbol	-	A6	A5	A4	A3	A2	A1	A0
R/W	-	R	R	R	R	R	R	R

• ADDRESS6-0（bit6-0） The USBC compares this register and address in all packet-ID and judges whether it is an effective transaction.
This is initialized to “00H” by USB-reset.

3.10.3.17 EOP register

This register is used when a data-phase of control-transfet type is terminating or when a short-packet 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
R/W	W	W	W	W	W	W	W	W

(Note) EOP<EP7_EOPB, EP6_EOPB, EP5_EOPB, EP4_EOPB> registers are not used at TMP92CH21.

In a data-phase of control-transfet type, write “0” to <EP0_EOPB> after writing all transmittion data to the FIFO-RAM or reading all received data from the FIFO-RAM. By this, the USBC will change to the status-stage terminating.

In a bulk-IN or interrupt-IN endpoint and in case of short-packet transmitting, write “0” to <EP0_EOPB> to terminate writing transmittion data.

Write “1” to another bit.

3.10.3.18 Port Status register

This register is used when a request of printer-class is receiving.
In case of request of GET_PORT_STATUS, the USBC operates automatically by using this data.

Port Staus (07E0H)

	7	6	5	4	3	2	1	0
bit symbol	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
R/W	W	W	W	W	W	W	W	W

The data should be written before receiving request.
Write “0” to the <Reserved> bit. This register is initialized to “18H” by reset.

3.10.3.19 Standard Request Mode register

This register is used to select auto Bus Enumeration for each kind of request.
By writing “0”, an auto Bus Enumeration is enabled.
By writing “1”, an auto Bus Enumeration is disabled and needed to control by a program.

This register set answer automatically to standard-request in hardware or control in software. Each bit mean kind of request.
This register is auto-answer at hardware by to reset object bit to ”0”, and it is control in software by to set object bit to “1”. If this register received request in hardware-control, this register disable interrupt-signal（INT_SETUP、INT_ENDPOINT0,INT_STATUS,INT_STATUSNAK）.If this register received request in software-control, this register assert interrupt-signal to outside, and it shift control-right to software.

Standard Request Mode (07D8H)

	7	6	5	4	3	2	1	0
symbol	S_Interface	G_Interfac	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 0： GET_STATUS
1： CLEAR_FEATURE
2： SET_FEATURE
3： GET_DESCRIPTOR
4： GET_CONFIGURATION
5： SET_CONFIGURATION
6： GET_INTERFACE
7： SET_INTERFACE

3.10.3.20 Request Mode register

This register set answer automatically to class-request in hardware or control in software. Each bit mean kind of request. Each bit mean kind of request.
This register is auto-answer at hardware by to reset object bit to ”0”, and it is control in software by to set object bit to “1”. If this register received request in hardware-control, this register disable interrupt-signal（INT_SETUP、INT_ENDPOINT0,INT_STATUS,INT_STATUSNAK）.If this register received request in software-control, this register assert interrupt-signal to outside, and it shift control-right to software.

Request Mode (07D9H)

	7	6	5	4	3	2	1	0
symbol	–	Soft_Reset	G_Port_Sts	G_DeviceId	–	–	–	–
R/W	–	R/W	R/W	R/W	–	–	–	–

Bit 0-3 : Reserve
4 : GET_DEVICE_ID
5 : GET_PORT_STATUS
6 : SOFT_RESET
7 : Reserve

Note1) It supports only auto-answer that about SET_ADDRESSrequest.
Note2) It is control in only SYNCH_FRAME-software that about SET_DESCRIPTOR.
Note3) It is control in only software that about class-request except for Vendor-request, printerclass.
If you hold PRINTER_CLASS-signal to ”0”, Bit4, 5,6 don’t write, and it is control in only software.
Note4) If you shift control in auto-answer and software, you must shift it before from reset to connect with USB host.
If you shift control in auto-answer and software while connection, this is like below.
• Auto-answer －> Software-contorol
If this conditon at receive request, conditon is switched software-contorol by next request.
• Software-contorol －> Auto-answer
There is possibility of wrong work. Therefore prohibit above shifting.
Note5) Interrupt of INT_SETUP, ENDPOINT0, STATUS and STATUSNAK assert only software-control.

3.10.3.21 COMMAND register

This register set COMMAND at each endpoint. This register can set select of endpoint in bit6-4 and select of kindof COMMAND in bit3-0.
It is ignored that outputted COMMAND to don't support endpoint.

COMMAND (07D0H)

	7	6	5	4	3	2	1	0
symbol		EP [2]	EP [1]	EP [0]	Command [3]	Command[2]	Command [1]	Command [0]
R/W	—	W	W	W	W	W	W	W

• EP [2:0] (bit 6-4)
0： Select endpoint0
1： Select endpoint1
2： Select endpoint2
3： Select endpoint3

• COMMAND [3:0]（bit 3-0）
0： Reserve
1： Reserve

2： SET_DATA0 This COMMAND clear toggle-sequence-bit in applicable endpoint. (EP0-3)
If you input this COMMAND, this command compulsory set "0" to toggle-sequence-bit in applicable endpoint.
Renewal data-toggle at transfer renew automatically by USBC.
If you compulsoly return "0" to toggle-sequence-bit in endpoint, you must execute this COMMAND.
You don't have to execute this COMMND in contorol-transfer-type and isochronous-transfer-type by controlling all in hardware.

3： RESET This COMMAND reset applicable endpoint.（EP0-3）
If you input this COMMAND, this COMMAND initializes applicable endpoint.
If you clear STALL of endpoint in CLEAR_FEATURE-request, you execute this COMMAND.（This executive don't affect transfer-mode）
Initialized item by this COMMAND is below.
- Clear toggle-sequence-bit of applicable endpoint
- Clear STALL of applicable endpoint
- Set to FIFO_ENABELEcondition

4： STALL This COMMAND is STALL applicable endpoint.（EP0-3）
If must return STALL handshake as answer to device-request, you execute this COMMAND.

5： INVALID This COMMAND is condition prohibit making use of applicable endpoint（EP1-3）.
If USBC detect in USB_RESET-signal from USB host, USBC prohibit using it automatically that all endpoint except for endpoint0. If config and interface is changed by devicerequest, you must prohibit using don't use endpoint.

6： CREATE_SOF This COMMAND inable quasi-SOF-generation-function inable.(EP0)
Default set to disable, it need using isochronous-transfer.

7： FIFO_DISABLE This COMMAND disable FIFO-RAM of applicable endpoint.（EP1-3）
If this COMMAND is set from outside, transfer in applicable endpoint send NAK. If this is reciving packet while set from outside, this is valid in next token. This executive don't affect transfering packet.

8： FIFO_ENABLE This COMMAND inable FIFO-RAM of applicable endpoint.（EP1-3）
If you disable FIFO-RAM by FIFO_DISABLE-COMMAND, this function use for cancel disables condition. At this point, If during receiving packet,this is valid in next token.
If during you detect in USB_RESET from host and you set transfer-mode by SET_CONFIG,SET_INTERFACE-request and you execute RESET-COMMAND, applicable endpoint is FIFO_ENABLE condition.

9： INIT_DESCRIPTOR This COMMAND use writed descriptor-RAM during running system.（EP0）
If USBC detect in USB_RESET from host controller, USBC read in content of descriptor-RAM automatically, and it set various setting.
If you change descriptor-RAM at during running system, it must read setting again. Therefore you executethis command. Case of connects in USB host, this function read in automatically. Therefore you don't have to execute COMMAND.

10： FIFO_CLEAR This COMMAND initialize FIFO-RAM of applicable endpoint.（EP1-3）
But this COMMAND doesn't initialize TOGGLE.
If reset in software, you execute this COMMAND.
Initialized content by this COMMAND is below.
- Clear STALL of applicable endpoint
- Set to FIFO_ENABELE-condition

11： STAL_CLEAR This COMMAND clear STALL applicable endpoint.（EP1-3）
If you clear only STALL of endpoint,you execute this COMMAND.

Note1) If you write this register, you must use recovery-system of number of 3clock in 12MHz.
If you go on to WR,you must consider.

3.10.3.22 INT_Control register

INT_STATUS_NAK-interrupt is disabled and inabled by writing value to this register.
This is initialized to disable by outside reset. When received setup-packet,this is disable.

INT_Control (07D6H)

	7	6	5	4	3	2	1	0
symbol	—	—	—	—	—	—	—	Status_nak
R/W	—	—	—	—	—	—	—	R/W

If host finish data-phase in it that data-length smaller than data-length appointed in wLength, it don't balance device side with stage management in contorol-read-transfer.
Therefore INT_STATUSNAK-interrupt was added for inform switching to status-stage.
But If don't need it, you can disable by this interrupt is asserted whenever status-stage every time.

- STATUS_NAK (bit0)
  0：INT_STATUS_NAKinterrupt disable
  1：INT_STATUS_NAKinterrupt inable

3.10.3.23 USB STATE register

This register output now state of device in connection with USB host.

USB STATE (07CEH)

	7	6	5	4	3	2	1	0
symbol	—	—	—	—	—	Configured	Addressed	Default
R/W	—	—	—	—	—	R／W	R	R

Inside USBC refer to this Configured,Addressed,Defaultbit, and manage answer to each device-request. If you excute transaction of SET_CONFIGrequest in software, you must write now state to this register. If appointed config0 from host, this is Unconfiged. And it must return to Addressed-state. Therefore, if appointed config0, you must write"0" to bit2.
When writed "0" to Configured bit (bit2), set automatically Addressedbit (bit1) in hardware. If appointed it from host that config value supported by device, device must execute mode-setting of each endpoint by value appointed at endpoint-descriptor in the config-descriptor. After finish mode setting, you set "1" to this Configuredbit (bit2) before access EOP register. When this bit was set to "1", Addressedbit(bit1) is reseted "0" automatically.

- Bit2-0
  1： Default
  2： Addressed
  4： Configured

3.10.3.24 EPx_MODE register (x is1-3)

This register set transfer-mode of Endpoint.（EP1-3）
If you set contolrol transaction of SET_CONFIG, SET_INTERFACE in software, you have to compose in appointed config or interface. At this point, you access this register, and you have to set mode.

EP1,2,3_MODE (0789H,078AH,078BH)

	7	6	5	4	3	2	1	0
symbol	−	−	Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
R/W	−	−	R/W	R/W	R/W	R/W	R/W	R/W

If you set transaction at SET_CONFIG, SET_INTERFACErequest to software-contorol, after receiced INT_SETUP interrupt, and you must finish weiting to during access to EOP register. This transaction prohibits writing in other timing, and it is ignore.

• DIRECTION (bit0)
0：OUT direction of from host to device
1：IN direction of from device to host

• MODE (bit1,2)
0：Contoroltransfer-type
1：Isochronous-transfer-type
2 : Balk-transfer-type or interrupt-transfer-type
3：Interrupt（Not toggle）

• PAYLOAD (bit3,4,5)
0： 8byte
1： 16byte
2： 32byte
3： 64byte
4： 128byte
5： 256byte
6： 512byte
7： 1023byte（Note1,2）

(Note1) max-packet-size of isochronous-transfer-type is 1023byte.
(Note2) If you appoint except for 8,16,…,1023 to wMaxPacketSize of Descriptor,payload big more than Descriptor value set by automatically answer of Set_Configuration,Set_Interface.

• Others（bit6,7） Reservation

3.10.3.25 EPx_SINGLE register

This register set mode of FIFO-RAM in each Endpoint.

EPx_SINGLE1 (07D1H)

	7	6	5	4	3	2	1	0
symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT	-	EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	-
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	-

(Note) Endpoint3 is only SINGLE-mode at TMP92CH21.

Bit 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_SELECTbit is “1”, EPx_SINGLEbit is validated below content.
0：DUAL-mode 1：SINGLE-mode
If you want to validate to content of EPx_SELECTbit, EPx_BCSbit set to “1”.

0：invalidate 1：validate

3.10.3.26 EPx_BCS register

This register is validate mode CPU-access-mode to FIFO-RAM in Each Endpoint

EPx_BCS1 (07D3H)

	7	6	5	4	3	2	1	0
symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT	-	EP3_BCS	EP2_BCS	EP1_BCS	-
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	-

Bit 0：No use
1：EP1_BCS
2：EP2_BCS
3：EP3_BCS
4：No use
5：EP1_SELECT
6：EP2_SELECT
7：EP3_SELECT

EPx_BCSbit have to set to “1”.
0：reserbvation 1：CPU access
If you want to validate to content of EPx_SELECTbit, EPx_BCSbit set to “1”.
0：invalidate 1：validate

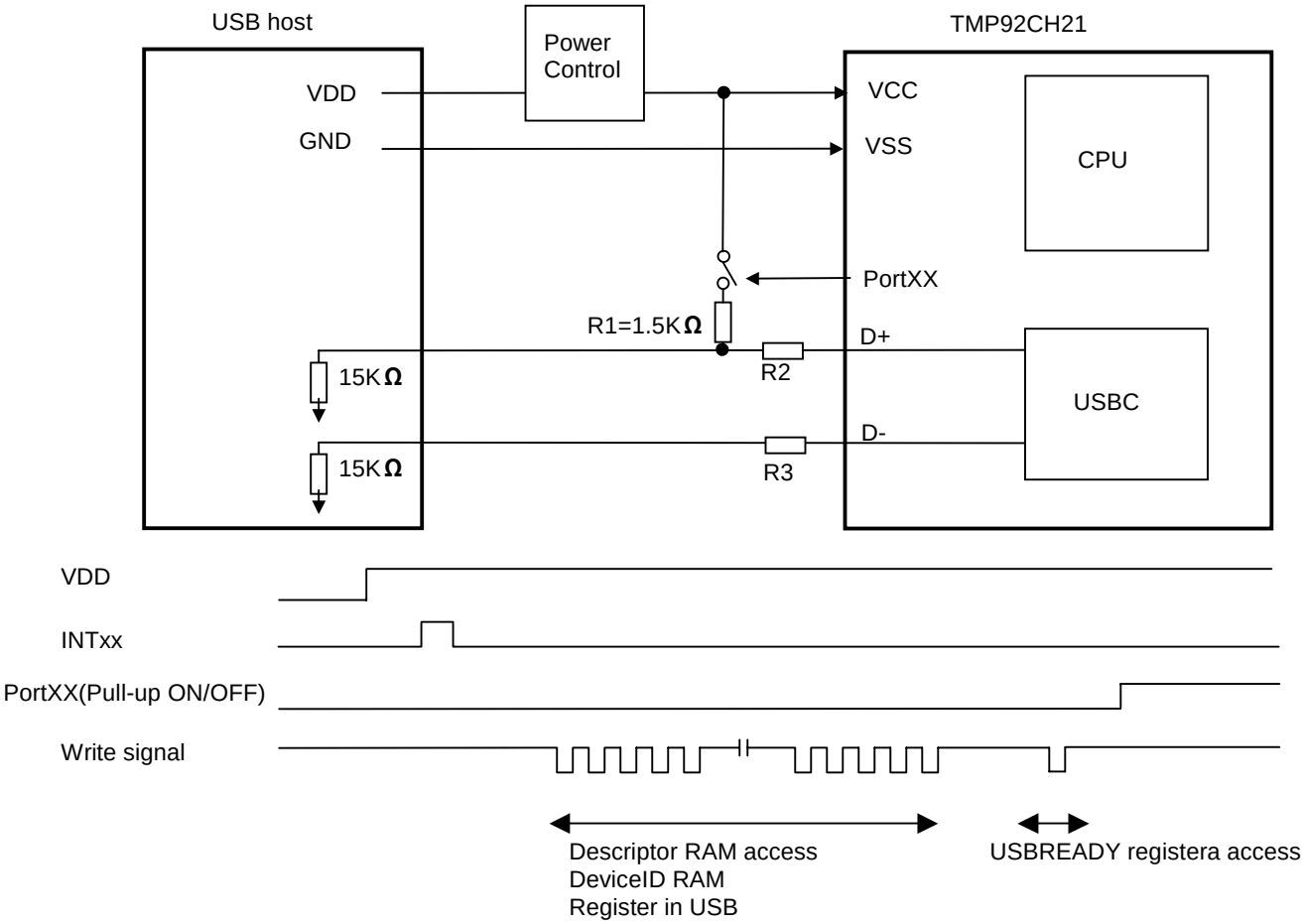
3.10.3.27 USBREADY register

This register informs USBC on finishing writing data to Descriptor RAM.
After store in Descriptor RAM you must write “0” to bit0.

USBREADY (07E6H)

	7	6	5	4	3	2	1	0
symbol	-	-	-	-	-	-	-	USBREADY
R/W	-	-	-	-	-	-	-	R/W

Bit0：USBREADY
0：Finish write to Descriptor RAM.
1：You can write to Descriptor RAM
（But you must not write to Descriptor RAM when it connection host.）



You detect in level of VDDsignal from USB cable, and execute initialize sequence.
USB disable detecting in USB_RESET signal until it writes “0” to USBREADY register.
In case of contorol pull-up resistance on D+signal to coontorol-signal, when pull-upresistance is connected host in OFFcondition, this condition is equivalent condition with USB_RESETsignal bypull-down resistance in host side. Therefore this function isn’t detected in USB_RESET until write “0” to USBREADY register.

Note1) Pull-up resistance and contorol-switch are needed at external of TMP92CH21.

3.10.3.28 Set Descriptor STALL register

This register set Set-Descriptor-request whether returns STALL automatically in data stage or status stage.

Set Descriptor STALL (07E8H)

	7	6	5	4	3	2	1	0
symbol	-	-	-	-	-	-	-	S_D_STALL
R/W	-	-	-	-	-	-	-	W

Bit0：S_D_STALL
0：Software contorol （default）
1：Automatically STALL

3.10.3.29 Descriptor RAM

This register use for store of Descriptor in RAM. Size of Descriptor is 384byte.But you must write to following format to Descriptor RAM-sturucture-sample when store Descriptor.

Descriptor RAM (0500H to 067FH)

	7	6	5	4	3	2	1	0
symbol	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Possible timing in RD／WR is only before detect in USB_RESETand performing SET_DESCRIPTOR request.

This SET_DESCRIPTOR-request-transaction-time is until access EOP register since INT_SETUP assert. If there is rewriting request of Descriptorin in SET_DESCRIPTOR, you process request belowsequence.

1) Read out every packet Descriptor transfer in SET_DESCRIPTOR request.
2) When finish reading out Descriptor number of last packet, you write all Descriptor RAM to Descriptor.
3) When finish writing, you execute INIT_DESCRIPTOR of command register.
4) When finish all transaction, access EOP register and finish status stage.
5) When recive into INT_STATUS, it is finished status stage.

You don’t have to execute INIT_DESCRIPTOR command when connect host. It is reading automatically start detect in USB_RESET.

3.10.4 Descriptor RAM

This area store Descriptor defined in USB. Device, Config, Interface, Endpoint, String-Descriptor has to set RAM in below Format.

Device Descriptor	18byte
Config1 Descriptor (Interfaces、Endpoints)	Under 255byte
Config2 Descriptor (Interfaces、Endpoints)	Under 255byt
String0 Length	1byte
String1 Length	1byte
String2 Length	1byte
String3 Length	1byte
String0 Descriptor	Under 63byte
String1 Descriptor	Under 63byte
String2 Descriptor	Under 63byte
String3 Descriptor	Under 63byte

Note1) If it isn't support String-Descriptor,String **x** Length area is size 0.
No support String-Descriptor return STALL.

Note2) Description in Config Descriptor refers to Descriptor sample.

Note3) USBC decides config-number, interface-number and endpoint-number. Therefore if supporting endpoint-number is small, close up and assign address.

Note4) This function effective only in case of store Descriptor as RAM.

Note5) Descriptor is total 384byte in device, config and string.

Note6) Possible timing in RD/WR of Descriptor RAM is only before detect in USB_RESET, and processing SET_DESCRIPTOR request.（Must not access except for this timing.）
Writing must finish before connection to USB host and processing SET_DESCRIPTOR request.
This SET_DESCRIPTOR request-transaction-time is during from INT_SETUP assert to 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 byte
515H	00H	WTotalLength(H)	
516H	01H	bNumInterfaces	
517H	01H	bConfigurationValue	
518H	00H	iConfiguration	
519H	A0H	bmAttributes	bus powered - remote wakeup
51AH	31H	MaxPower	98mA
Interface0 Descriptor AlternateSetting0			
51BH	09H	bLength	
51CH	04H	bDescriptorType	Interface Descriptor
51DH	00H	bInterfaceNumber	
51EH	00H	bAlternateSetting	AlternateSetting0
51FH	01H	bNumEndpoints	
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 byte
529H	00H	WMaxPacketSize(H)	
52AH	00H	bInterval	
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	
Endpoint1 Descriptor			
534H	07H	bLength	
535H	05H	bDescriptorType	Endpoint Descriptor
536H	01H	bEndpointAddress	OUT
537H	02H	bmAttributes	BULK
538H	40H	WMaxPacketSize(L)	64 byte
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 byte
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 byte
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 byte
557H	00H	WMaxPacketSize(H)	
558H	00H	bInterval	
Endpoint3 Descriptor			
559H	07H	bLength	
55AH	05H	bDescriptorType	Endpoint Descriptor
55BH	83H	bEndpointAddress	IN
55CH	03H	bmAttributes	INTERRUPT
55DH	08H	WMaxPacketSize(L)	8 byte
55EH	00H	WMaxPacketSize(H)	
55FH	01H	bInterval	1ms
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			

3.10.5 Device Request

3.10.5.1 Standard • Request

(1) GET_STATUS Request

This request returns automatically status of appointed Reseption side.

BmRequestType	BRequest	wValue	wIndex	wLength	Data
10000000B 10000001B 10000010B	GET_STATUS	0	0 Interface Endpoint	2	Device,Interface or Endpoint status

Return below information to according to priority of little-endian in request to Device.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote Wakeup	SelfPower
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- RemoteWakeup　Return now RemoteWakeup setting.
  This bit is set or reseted by SET_FEATURE or CLEAR_FEATURE request. Default is value setting bmAttributes field in Config Descriptor.
- SelfPower　Return now power supply setting. Return self or BusPower according to setting value bmAttributes field in Config Descriptor.

Request to interface returns 00H of number of 2byte.

Request to Endpoint returns in accoding to priority of little-endian below 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　Return HALT status of selected point.

(2) CLEAR_FEATURE Request

This request clear Particuler function or disable Particuler function.

BmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	CLEAR__ FEATURE	Feature Selector	0 Interface Endpoint	0	None

- Reception side Device
  FeatureSelector：1 Disable now RemoteWakeup setting.
  FeatureSelector：This is STALL except for 1.
- Reseption side interface
  This is STALL.
- Reseption side End point
  FeatureSelector：0 Clear HALT of applicable Endpoint.
  FeatureSelector：This is STALL except for 0.

Note1) If request absent Endpoint, it is stall.

(3) SET_FEATURE Request

This request is set Particuler function or inable Particuler function.

BmRequestType	bRequest	wValue	Windex	wLength	Data
00000000B 00000001B 00000010B	SET__FEATURE	Feature Selector	0 Interface Endpoint	0	None

- Reseption side Device
  FeatureSelector：1 Inable now RemoteWakeup setting.
  FeatureSelector：This is STALL except for 1.
- Reseption side Interface
  This is STALL.
- Reseption side End point
  FeatureSelector：0 Halted applicable Endpoint.
  FeatureSelector：This is STALL except for 0.

Note1) If request absent Endpoint, it is stall.

(4) SET_ADDRESS Request

This request is setting DeviceAddress.Following request answer this setting DeviceAddless.

Answer on previous Device Address until StatusStage 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 send 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 Send Device Descriptor stored Descriptor RAM. When you use Descriptor ROM, prepare idProduct and bcdDevice field as register. Therefore if determine Descriptor-data as ROM,you can writing only this area. If you use it, access to this register before connection to USB host, and determined data.
- Config — Config Send Config Descriptor stored Descriptor RAM. At this point, send not only Config Descriptor but also interface, Endpoint Descriptor.
- String — String Send String Descriptor of index appointed lower byte in wValue field.

Note1) Send Decriptor of short data length in wLength and Descriptor length by automatically answer of Get_Descriptor.

(6) SET_DESCRIPTOR Request

This request set Particuler function or inable Particuler 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 is no support.

According to INT_SETUP interrupt, If receiving request discerned SET_DESCRIPTOR request, take back data after EP0_DSET_Abit of DATASET register confirmed “1”. When finishing, access to EOP register and write “0” to EP0_EOPBbit. And finish Status Stage. Transaction is same Vendor Request.This detail refer to section Vendorrequest.

(7) GET_CONFIGURATION Request

This request returns configuration value of now Device.

BmRequestType	BRequest	WValue	wIndex	wLength	Data
10000000B	GET_ CONFIG	0	0	1	Configuration Value

If no compound, return to “0”. If compound, return to configuration value.

(8) SET_CONFIGURATION Request

This request set device compound.

BmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ CONFIG	Configuration Value	0	0	None

This is configuration value appointed inferiority byte of WValue field.

When this value is “0”, it is no compound.

(9) GET_INTERFACE Request

This request returns Alternate Setting value set appointed interface.

BmRequestType	bRequest	wValue	wIndex	wLength	Data
10000001B	GET_ INTERFACE	0	Interface	1	Alternate Setting

If there is no appointed interface, it is STALL.

(10) SET_INTERFACE Request

This request select AlternateSetting in appoint interface.

BmRequestType	bRequest	wValue	wIndex	wLength	Data
00000001B	SET_ INTERFACE	Alternate Setting	Interface	0	None

When there is no appointed interface, it is STALL.

(11) SYNCH_FRAME Request

This request sends same frame of Endpoint.

BmRequestType	bRequest	wValue	wIndex	WLength	Data
10000010B	SYNCH_FRAME	0	Endpoint	2	Frame No

Automatically answer of this request is no support.

According to INT_SETUP interrupt, If receiving request discerned SYNCH_FRAME, write data of 2byte in Frame No after EP0_DSET_Abit of DATASET register confirmed “0”. When finishing, access to EOP-register and write “0” to EP0_EOPBbit, and finish Status Stage. This transaction can useing only in case of Endpoint support isochronous-transfer-type, and in case of support this request. Transaction way is same vendor-request. This detail refers to section vendor-request.

3.10.5.2 Printer Class Request

USBC supports automatically answer about request in Printer class.
If it is class except for for printer, you process class request according to INT_SETUP interrupt. This transaction is same Vendor Request.

(1) GET_PORT_STATUSRequest
This request sends Port Status to host.

BmRequestType	bRequest	wValue	wIndex	WLength	Data
10100001B	GET_ PORT_STATUS	0	Interface	1	Port Status

USBC have within Port_Status register. You must write port information at this register. When you receive this request, this sends PortStauts register data within USBC. Application has to set port information beforehand PortStauts register within USBC. Port information possible for sending is only 1type. Therefore wIndex value is ignored.
Each bit of this register is below.

D7	D6	D5	D4	D3	D2	D1	D0
Reserved	Reserved	PaperError	Select	NotError	Reserved	Reserved	Reserved

(2) SOFT_RESET Request
This request receives Soft Reset.

BmRequestType	bRequest	wValue	wIndex	WLength	Data
00100011B	SOFT_RESET	0	Interface	0	None

When it receives Soft Reset, set SOFT_RESET flag. When finish Status Stage, reset SOFT_RESET flag. This Request receive flag is only one type. Therefore WIndex field value is ignored.

(3) Vendor request （class request）

Automatically answers Vendorrequest in USBC don't support.

According to INT_SETUPinterrupt, access to register store of Device Request, and discern receiving request. If this request is Vendor request, massage USBC in outside, and you has to process according to it.

Below is explanation case of data-phase is sending(Contorolread),and case of data-phase is receiving(Contorolwrite).

(a) Contorol Request

BmRequestType	bRequest	wValue	wIndex	wLength	Data
110000xxB	Vender peculiar	Vendor peculier	Vendor peculier	Vendor peculier (expire0)	Vendor Data

When it receive into INT_SETUP, Application judge content of device request by bmRequestType、bRequest,wValue,wIndex,wLength register and you process adequate it. Application access to "Setup_Received register" after judged request. And application has to inform USBC to recognize into INT_SETUP.

If you prepare sending data in application side, you access DATASET register and you must confirm "0" in EP0_DSET_Abit. And you write data at FIFO-RAM of endpoint 0.If you send data more than payload, write data after you confirm whether a bit of EP0_DSET_A in DATASET register is "0".（You may use interrupt signal of INT_ENDPOINT0.）If you finish a writing all data, you write "0" at bit of EP0 on EOP register.When USBC received it ,it close automatically status stage. And when USBC normally close status stage, it asserts interrupt INT_STATUS.If out side application recognize normally close status stage, you management stage by this interrupt signal. Maybe you don't normally close status stage, and maybe you receive new SETUPTokeninto the stage. When assert INT_SETUP interrupt signal, put "1" to STAGE_ERROR bit in EP0_STATUS register. This operation inform Status Stage can't finish normally to outside.

And there is case of dataphasefinish in data number is short than value showed to wLength in protocol in Contorol-read-transfer-type of USB. If composed transaction of only wLength value in application program, this function can't meet in case of host change Status Stage nothing arrive at expect data number. At this point, you can inform change of Status Stage by INT_STATUSNAK interrupt signal. (But you have to cancel mask of STATUS_NAKbit in interrupt Contorol register.) As it is, case of Vendorrequest, it set receiving buffer size to Host Contoroller in driverside. Therefore we think don't occur thing like this. (Data sending in Payload of 8byte from device recognaized Short Packet until grasp Payload Size of deviceside by host. If you contorol Standard Request, you have to regard to software by may be abobe case on the exterior.)

(b) Contorol Write/Request

• There is no dataphase

BmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendorpeculier	Vendorpeculier	Vendorpeculier	0	None

When application receive INT_SETUPinterrupt, Application judge content of Request to bmRequestType, bRequest, wValue, wIndex and wLength register. And you process according to this Request. After judge Request, applicaton access Setup_Received register, and it have to inform to USBC to recognized INT_SETUPinterrupt.

If transaction finish in application side, write “0” to bit of EP0 in EOP register. USBC receive this,and it automatically finish Stataus Stage.

• There is dataphase

BmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendorpeculier	Vendorpeculier	Vendorpeculier	Vendorpeculier (except for 0)	Vendor Data

If application receives INT_SETUP interrupt, you judge content of Device Request to bmRequestType, bRequest, wValue, wIndex, wLength register, and you process according to this request. After application judge Request, it access Setup_Received register, and it have to inform recognization INT_SETUP interrupt to USBC. If you are ready receiving data in application side, you access DATASET register, and EP0_DATASET confirming “1”. And you read out data to FIFO-RAM of Endpoint 0. If receive data more than Payload, sence bit of Dataset registerEP0_DSET_A, and after confirmed ”1”, have to read out data of next packet. (You can make use of INT_ENDPOINT0 interrupt signal.) After read out all data, write “0” to bit of EP0 in EOP register. USBC receive this, and it automatically finishes Status Stage.

When USBC normal finish Status Stage, it assert INT_STATUS interrupt. If application in outside grasp normally finish of Status Stage, you manage make use of this interrupt signal.there is case of can’t normally finish Status Stage or receiving new SETUPToken in same Stage.At this point, in assert INT_SETUP interrupt signal, it put “1” to STAGE_ERRORbit of EP0_STATUS register, and inform can’t normally finish Status Stage to outside.

Below is Contorol flow in USBC watch from application.

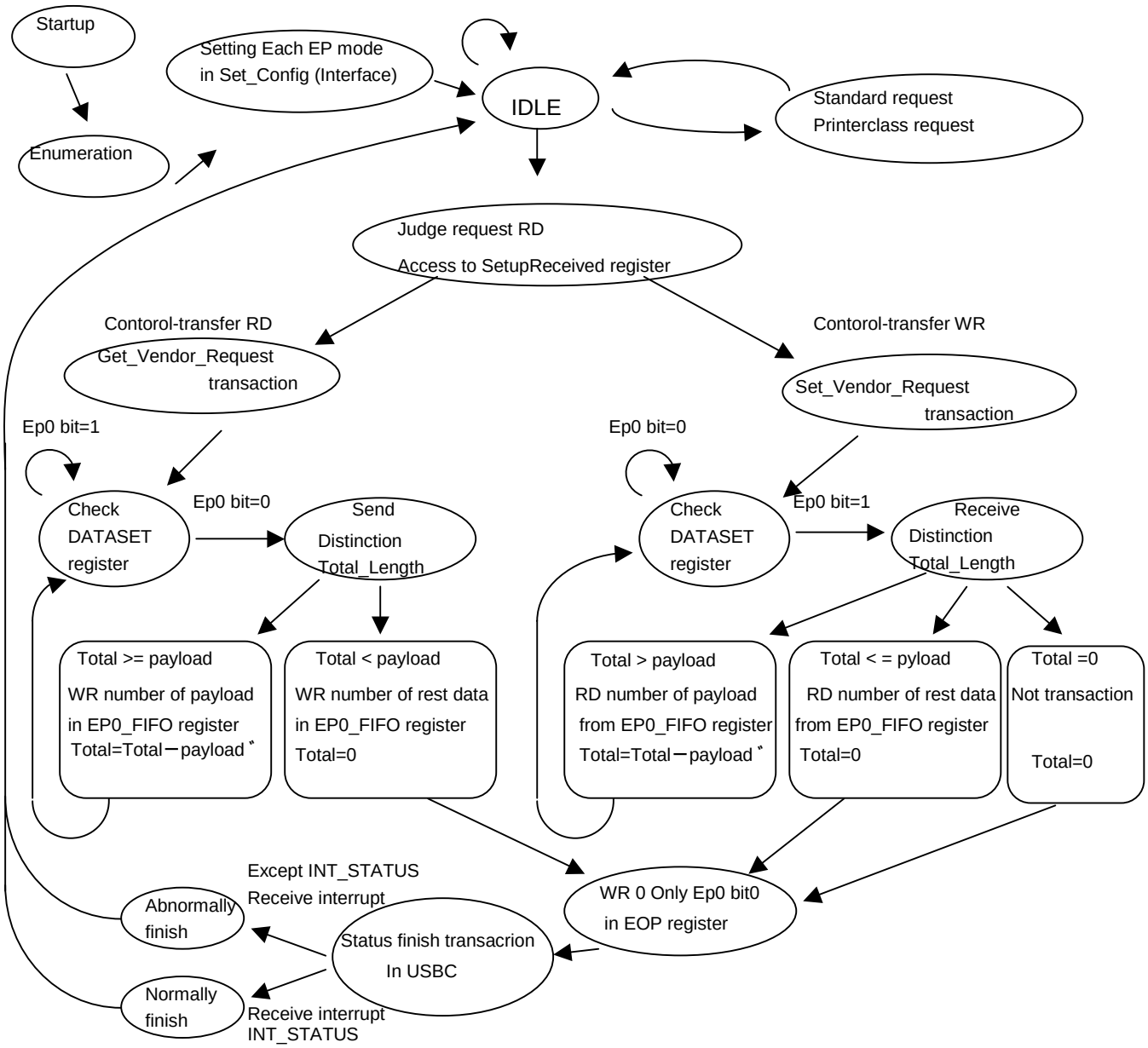


Figure 3.10.2 Contorol flow in USBC watch from application

Note1) Thre is no such special case as overlap receive SETUP packet.
Hull flow is contorol transfer chapter 4.5.2.3.

Note2) This flow is a series of flow of request. But a process can be separated to module to process every each interrupt.

3.10.6 Transfer-mode and protocol transaction

USBC perform automatically in hardware as follows;

Receiveing packet
Judgement of address, endpoint and transfer-mode
Error process
Confirmation of togglebit and CRC in receiving data packet
Generation of togglebit and CRC in sending data packet
Handshake Answer

(1)Protocol outline

Below is definition format of USB packet. This is processed during send and receives hardware into USBC.

- SYNC field

  Always exist first of each packet, and get same time input data and in CLK in USBC.

- Packet identification field (PID)

  This PID follow on SYNC field every USBpacket.USBC discern PID type,and judge transfer-type by decoding this cord.

- Address field

  USBC confirm whether this function aapointed to host by making use of this field. USBC compare with address set ADDRESS register. If an address accords, USBC continue process. If an address doesn't accord, USBC ignore this Token.

- Endpoint field

  If want more than two subs channel in field of 4bit,it deside this function. USBC can support Endpoint to max seven except for Contorol. Token isn't permited Endpoint ignores.

- Frame number field

  Field of 11bit is added +1 at every frame by host. This field follows first sent SOFToken of each frame, and frame number is appointed it. USBC read content of this field when receiving SOFToken, and set frame number to FRAME register.

- Data field

  This field is data of unit byte in 0 to 1023byte. When receiving it, USBC transfer only part of this data to FIFO, after confirm CRC, assert interrupt signal. And inform finishing loading data to FIFO. When sending it, following INToken, transfer data of FIFO. Finally, attach data of CRC field.

- CRC function

  Token is attached 5bit, data is attached CRC of 15bit. USBC compare automatically CRC of receive data and attached. When sending it, send generate CRC automatically. This function dosen't compare by various transfer-mode.

(2) Transfer-mode

USBC support transfer-mode in FULL speed.

- FULL speed device

  contorol-transfer-type

  Interrupt-transfer-type

  Balk-transfer-type

  Isochronous-transfer-type

Following is explanation movement of USBC in each transfer-mode. Movement of each transfer explains flow of data until FIFO.

(a) Balk –transfer-type

Balk-transfer-type warrants transfering no error between host and function by detect error and retry. Make use of 3phasein Token, data and handshake fundamentally, but flow Contorol and STALLcondition is 2phase by to dataphase changed for handshakephase. USBC hold Status of every each Endpoint, and it contorol Flow Contorol in hardware. Each Endpoint condition can confirm to EPx_STATUS register.

(a-1) Sending balk mode

Below is transaction • format sending loading balk. It has to follow below.

- Token : IN
- Data : DATA0/DATA1, NAK, STALL
- Handshake : ACK

Contorol flow

Below is Contorol flow in USBC receiving INToken.

1) Receive Token paket. Besides this confirm address • Endpoint number • error, and it check whether conform applicable Endpoint-transfer-mode and IN Token. If don't conform, return IDLE.

2) Confirm condition of EPx_STATUS register.

- INVALI condition : Return IDLE.
- STAL condition : Return stall Handshake, and return IDLE.

Confirm FIFO condition, if nothing prepared data number of 1packet data, return NAKHandshake, and return IDLE.

If data Number of 1packet is prepard to FIFO, switch to 3).

3) Switch to generation data packet.

It generate data packet make use of togglebit register into the USBC.

Next it transfer data to SIE from FIFO into the USBC and it generate data packet. At this point, it confirms transfered data number. And if there is more than max payload size of each endpoint, it is finished loading after Bitstuff Error occurs. And regard STATUS as STALL.

4) Attach CRCbit counted to last loading data of FIFO to last.

5) When it received ACK handshake in host,

- Clear FIFO.
- Clear DATASETregister.
- Renewtogglebit,and prepare for next.
- Set STATUS to READY.

USBC close normally. FIFO can receive next data.

If it was Time out not receive ACK from host,

- Regard STATUS as TX_ERR.
- Put back adlesspointer of FIFO.

You execute them. And you wait to next retry leave keeping FIFO data.

This flow is figure 5.1.

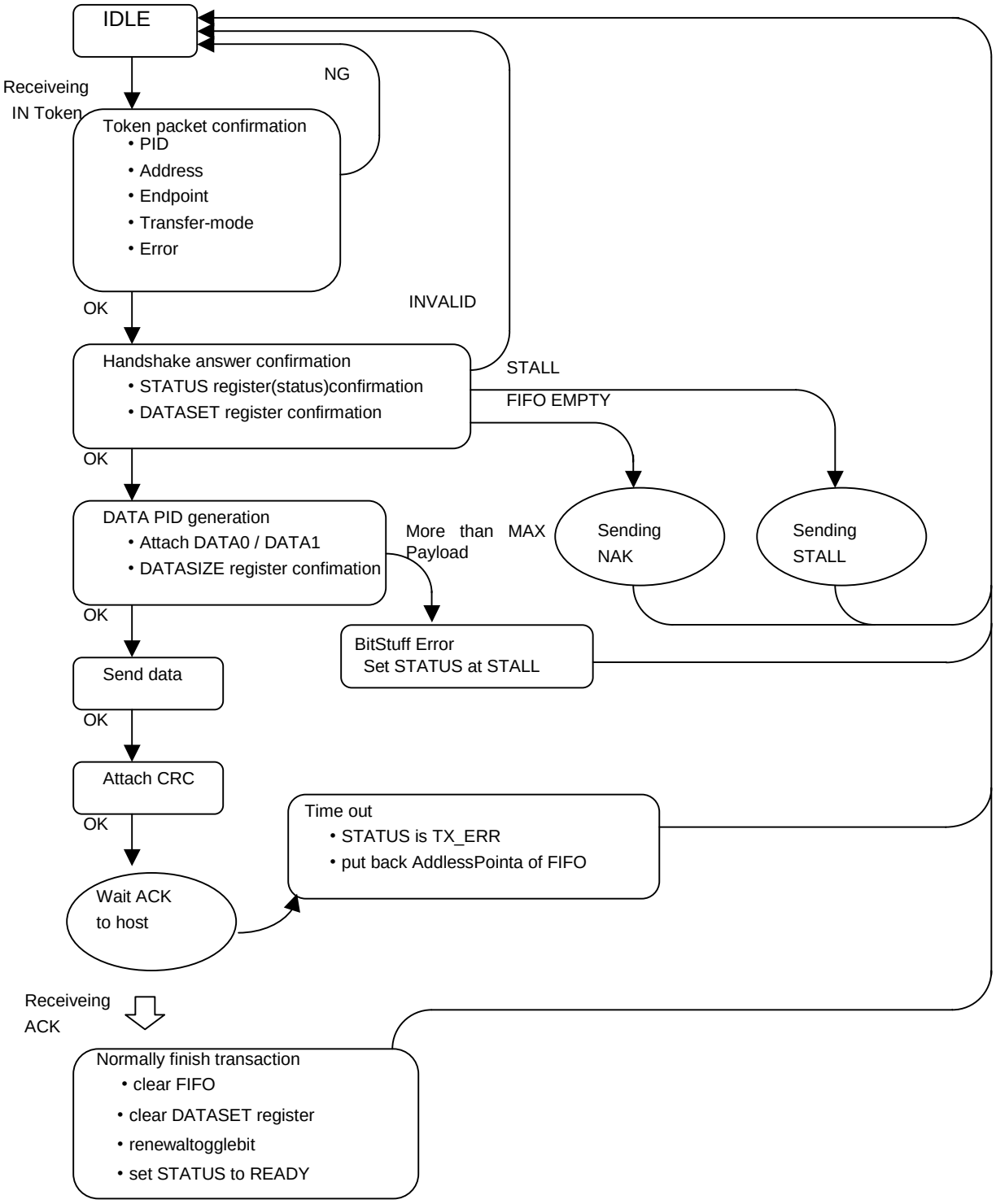


Figure 3.10.3 Contorol flow in USBC（balk-tranfer-type (sending)／interrupt-transfer-type (sending)）

(a-2) Receiving balk mode

Below is transaction format receiving balk-transfer-type. It has to follow below.

- Token: OUT
- Data: DATA0/DATA1
- Handshake: ACK,NAK,STALL

Contorol-flow

Below is Contorol-flow into USBC receiving INToken.

1) Receive Tokenpaket. Besides this confirm address,endpoint number and error and this check whether conform applicable endpoint transfer-mode and IN Token. If can't confirmation, return IDLE.

2) Confirm condition Status register

Invalid condition: Return IDLE.

Stall condition: When finished dataphase, return STALL-Handshake and return IDLE. Data cancel.

Confirm FIFO condition, If can't preparation store of data number of 1packet, clear now loaded data, and return NAK Handshake following dataphase, and return IDLE.

3) Receive Data packet.

Load data from SIE into the USBC to FIFO.At this point, it confirms loaded data number. And if there is more than max-payload-size of each endpoint, it returns IDLE regarded STATUS as STALL. ACK Handshake doesn't return.

4) After loaded last data, if it doesn't accord counting CRC compare with loaded CRC, it set STATUS at RX_ERR and returns IDLE. At this point it dosen't return ACK.

After retry, when it can normally receving next data, STATUS change 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-addless-pointer return. And it can receive next data.

5) If it finishes normally accord CRC compare withtoggle, return ACK-hand-shake.

Bellow is process into the USBC.

- Set transfer-data-number to DATASIZE register.
- Set DATASET register.
- Renewtogglebit,and prepare for next.
- Set STATUS to READY.

USBC is normally finishing it.

This flow is figure 5.2.

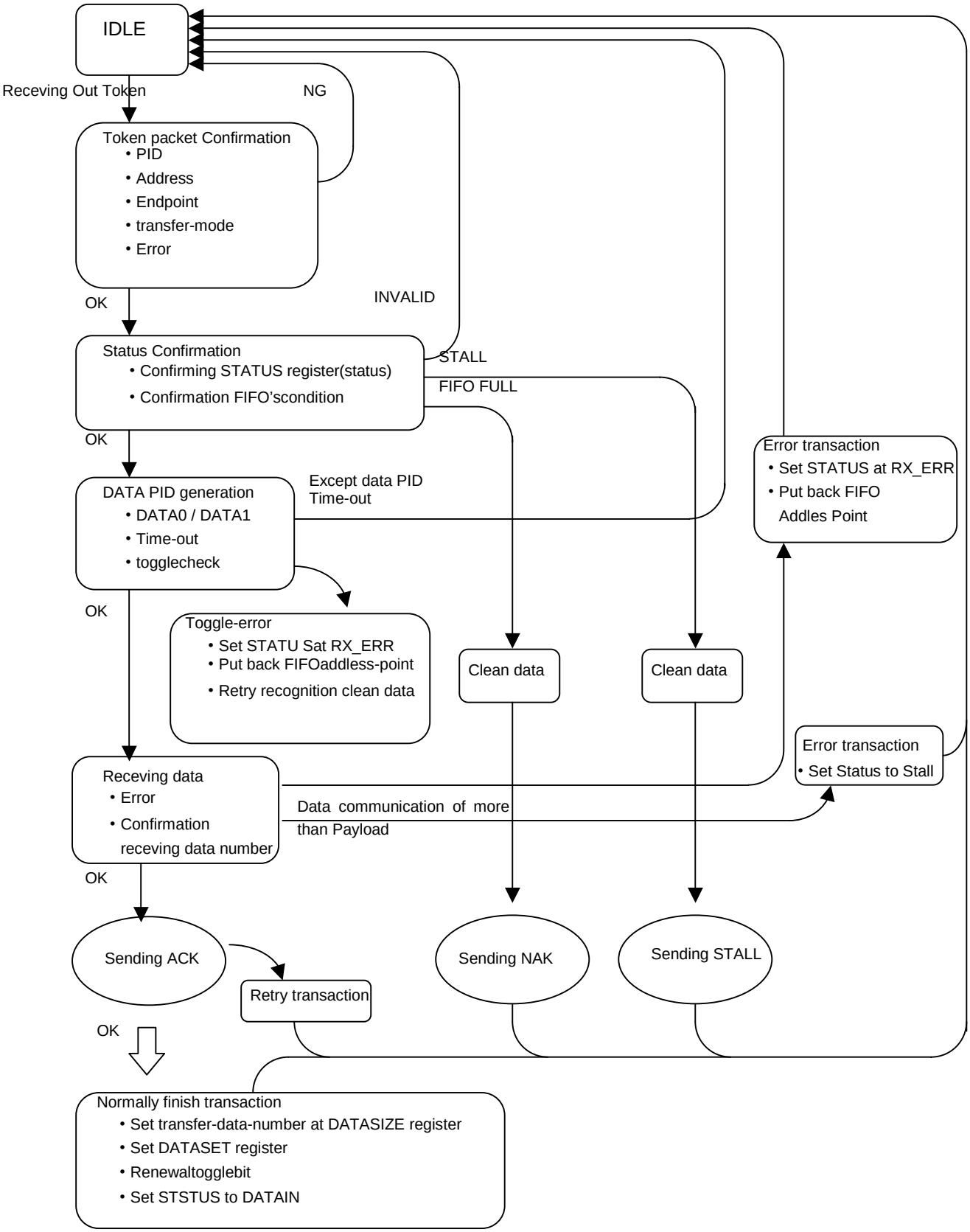


Figure 3.10.4 Contorol flow in USBC（balk-transfer-type(receiving)）

(b) Interrupt-transfer-type

Interrupt-transfer-type makes use of transaction-format same sending-balk-transfer.

Setting and answer in USBC Hardware transfering Usingtogglebit are same transfering sending balk. It can transfer dosen't usetogglebit ininterrupt-transfer. In this case, if it receive ACK Handshake to host, renewal toggle-bit, and normally finish. USBC clear FIFO for next transfer.

(b-1) interrupt-sending-mode（toggle-mode）

USBC movement is same balk sending mode. Please refer to (a).

(b-2) interrupt sending mode（Not toggle-mode）

This is same balk sending mode fundamentally. But if can't receive ACK Handshake from host, transaction is differnt.

After sending datapacket,

When receving ACK Handshake from host

- Clear FIFO.
- Clear DATASET register.
- Renewtogglebit and prepare for next.
- Set STATUS to READY.

USBC normally finish by above transaction. FIFO cans receving next data.

If it was Time-out nothing receving ACK from host,

- Clear FIFO.
- Clear DATASET register.
- Renew toggle-bit and prepare for next.
- Set STATUS to READY.

You process above transaction. And it is same except for changing STATUS.

(c) Contorol-transfer-type

Contorol-transfer-type have below three stages.

- Set-up stage
- Data stage
- Status stage

Data stage is skipped sometimes. Each stage has in one or plural transaction. USBC perform each transaction while managing of three stages in Hardware. Contorol-transfer-type has below 3type by whether there is Data Stage or not, or direction.

- Contorol • read-transfer-type
- Contorol • write-transfer-type
- Contorol • write-transfer-type（not Data Stage）

3 transfer-sequence is figure 5.4, figure 5.5,and figure 5.6.

USB answer automatically about standard-request, printerclass-request in hardware. Class-request, Vendor-request except for Standard-request, printerclass have to intervening CPU on contoroling USBC.

Below is Contorol flow in USBC and Contorol flow in intervening CPU.

(c-1)Setup stage

Setup stage is same sending balk transaction except for in case of Token ID is SETUP. But Contorol-flow in USBC deffer it.

- Token : SETUP
- Data : DATA0
- Handshake : ACK

Contorol-flow

Below is Contorol-flow into the USBC receving SETUP-Token.

Receving Set up-Token-packet and confirm address • Endpoint-number • error.

Check the Contorol-transfer-mode with applicable endpoint.

1) Confirm STATUS register situation.

Return IDLE only INVALID situation.

This transaction permit receiving data by STATUS-register-value and FIFO condition inbalk-transfer-mode. But dataSETUP is return STATUS to READY, and prohibit access from CPU to FIFO, and clear in FIFO in Endpoint0 in case of every, and prepare for after dataphase.

If CPU access Setup Received registers USBC, it recognized receiving Device-request. And this function cancels access-prohibition of FIFO in EP0 from CPU.

There is this function for receiving it if coming new request in during normally finish before one device-request.

3) Receive Data-packet.
Load number of 8byte device-request from SIE in USBC to below request-register.

- bmRequestType register
- bRequest register
- wValue register
- wIndex register
- wLength register

4) After loaded last data, if it doesn't accord counting CRC compare with loaded CRC, it set STATUS at RX_ERR.And doesn't return ACK-Hand-shake to host. And returns. IDLE. Host retry.

5) If it finish normally accord CRC compare with toggle, return ACK-hand-shake to host.

- Discern right to contorol receiving Device-request,if request need contorol in softwar, assert INT_SETUPinterrupt,and inform receiving request in outside.This function don't assert INT_SETUPinterrupt in case of automatically answer in hardware.
- According to stage Contorol flow prepare next stage.
- Set STATUS to DATAIN.
- Set toggle-bit to "1".

Setup stage finish above.

This flow is figure 5.3.

8 byte data Loaded this SETUP stage is Device-Request.。

CPU must process correspond it devicerequest. USBC detect in content of next from data of 8byte,and it manages stage-management in hardware.

- There is Data-stage or not
- Data-stage direction

It judge Contorol • read-transfer-type, Contorol • write-transfer-type Contorol • write-transfer-type（not dataphase）by them.

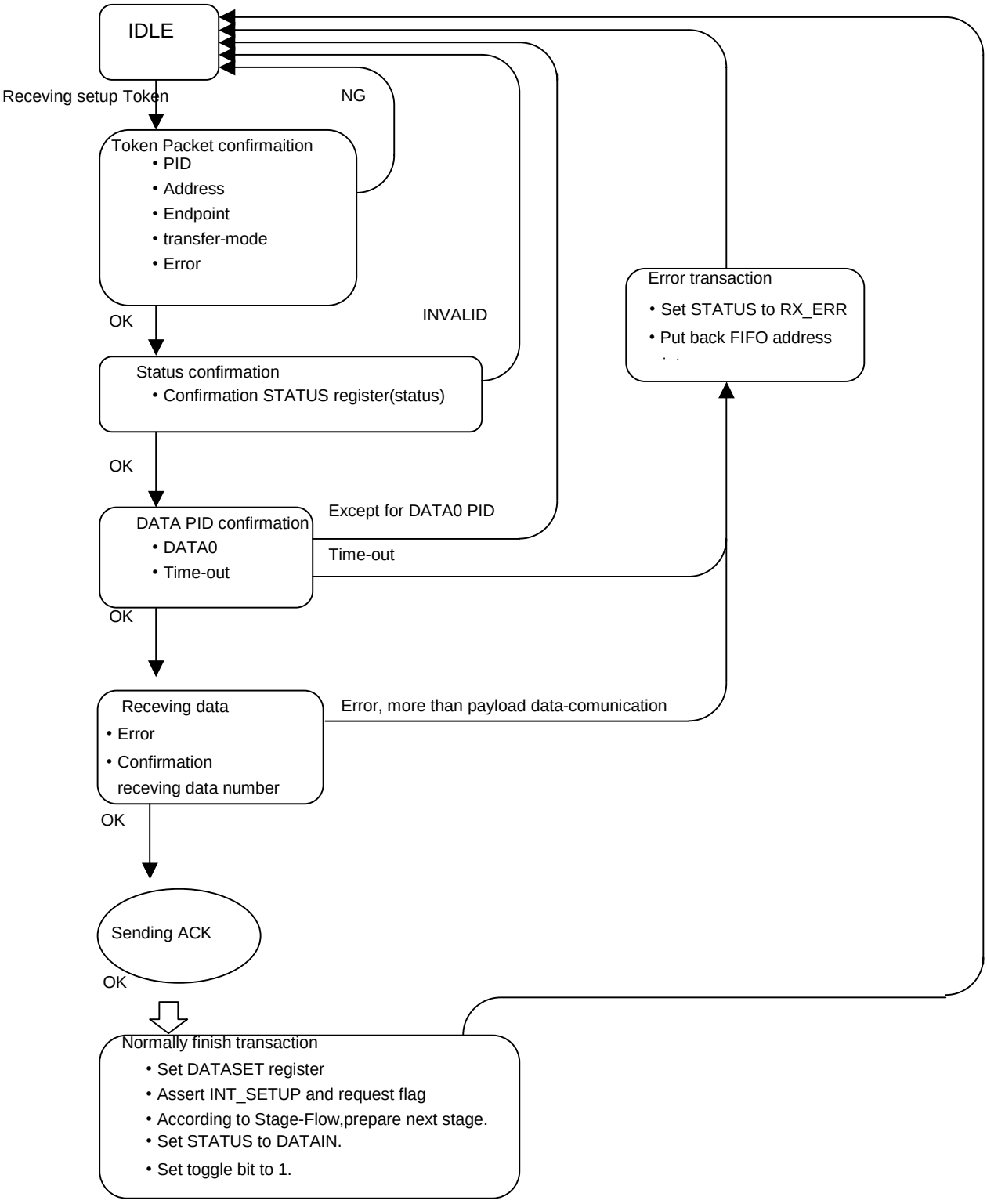


Figure 3.10.5 Contorol Flow in USBC（Setup stage）

(c-2) Data-Stage

Data-Stage composed base on togglesequenceby one or plural transaction.
Transactionis same format sending or receving balk • transaction.
Below is difference.

- toggle bit start from ”1” by SETUP-stage.
- It judge whether right that IN and OUTToken compare to course-bit of performing Device-Request. If receive Token reserse transfer-direction, recognive it as Status-stage.
- Assart INT_ENDPOINT0 interrupt.

(c-3)Status-stage

Status-stage compose 0data length packet with DATA1’sPID1 and handshake behind IN o r OUT Token.it use transaction different course preceding stage.
This is combination below.

- Contorol/read-transfer-type : OUT
- Contorol/write-transfer-type : IN
- Contorol/write-transfer-type（not data-phase） : IN

USBC process Status-stage to base of Contorol-flow in Contorol-transfer-type. At this point, CPU has to write “0” to EP0bit of EOP-register in transaction last for Status-stage normally finish.
Below is detail of Status-stage.

(c-3-1) In status stage

Below is IN status stage transaction format.

- Token : IN
- Data : DATA1（0data length）,NAK, STALL
- Handshake : ACK

Contor-flow

Below is transaction-flow of IN status • stage in USBC.

1)After receve Token-packet, confirm receving address • endpoint-number • error. If it dosen’t conform, return IDLE. If it is permited status stage base of stage contorol-flow in USBC, advance next stage.
2) Confirmation STATUS registers condition.

- INVALID condition : Return to IDLE.
- STALLcondition : Return to STALL Hand shake and put back IDLE.

Confirm access EOP register from out side. If can’t access it, return NAK handshake for continue contorol-transfer. And return to IDLE.
3) If it Confirm EOP register accessed, it send 0data length data packet and CRC.
4) If receive and shake

- Set STATU at READY.
- Assart INT_STATUS interrupt.

It is normally finish by above transaction.
If it was Time-out dosen’t receive ACK Hand shalke from host,

- Set STATUS register at TX_ERR and Return IDLE. And wait to retry Status stage.

At this point, If start new SETUP stage close abnormally status stage, USBC set error to STATUS register.

(c-3-2) OU Status-stage
Below is OUTStatus-stage transactionformat.

- Token : OUT
- Data : DATA1（0data length）
- Handshake : ACK, NAK, STALL

Contorol flow

Below is transaction flow of OUT status stage into the USBC.

1) Receive Token packet. Confirm receving address • endpoint number • error. If it dosen't conform, return IDLE. If status stage permited base of stage contorol flow into the USBC, advance next stage.
2) Confirm status register condition.
   - INVALID condition : Return IDLE.
   - STALL condition : Clean datapacket,return STALL hand shake and return IDLE.

   Confirm access EOP register from out side. If don`t access to EOP register, return to NAK handshake for continue Contorol-transfer. And condition return to IDLE.
3) If confirm access to EOP register, receive 0datapacket and CRC.
4) If there was no error data, send ACK handshake to host.
   - set STATUS at READY.
   - Assert INT_STATUS interrupt.

This is normally closing.
If there is error data, dosen't return ACK handshake.

- Set STATUS register at RX_ERR and return IDLE.It wait to retry status stage.

At this point, if status-stage finish abnormally and it started new SETUP stage, USBC set STATUS register to stage error. Sequence of this protocol refers to section supplement.

(c-4)Stage management

USBC manage hardware progress of each stage in Contorol-transfer.

Receiving Token from SB host or CPU access register performs each stage. Each stage in Contorol-transfer-type has to perform combination software. Below is USBC detecting from 8 byte data in SETUP stage. It discern from type of Contorol-transfer-type, and manege stage.

- There is data-stage or not
- data-stage direction

Judge Contorol-read-transfer-type, Contorol-write-transfer-type, Contorol-write-transfer-type（no data stage） by them.

Below are various conditions for changing stage in contorol-transfer.

If receving correspond to next stage from host before switch next satge from state into the USBC, it return NAK handshake and inform USBC host of BUSY. If receive SETUP Token from host to every condition In All contorol-transfer-type, this function stop now transaction, and switch SETUP-stage in USBC. If CPU is processing previous contorol-transfer-type, react it by receiving new INT_SETUPinterrupt.

Stage change condition of Contorol • Read-transfer-type

＊Receive SETUPToken from host

• start setup • stage in USBC.

• Receving normally data in request and discern. And assert outside INT_SETUP interrupt.

• Change data • stage into the USBC.

＊Receive INToken from host

• CPU take back request from request-register when it react INT_SETUP interrupt.

•Discern request and access SetupReceivedregister for inform USBC to recognized INT_SETUP interrupt.

• According to Content device-request, monitor EP0bi of DATASETregister write data to FIFO.

• If USBC is set data of payload to FIFO or it was ordered CPU shortpacke-transfer-type t in EOPregiste,set DATASET register.

• USBC react INtoken setting data in FIFO and it load to host.

• When CPU finish transaction, it write ”0” to EP0bit of EOPregister.

• Change statusstage into the USBC.

＊Receive OUTtoken from hst.

• Return ACK at OUTtoken and chabge IDLE condition into the USBC.

• Assert INT_STATUSinterrupt in outside.

These change condition is figure 5.4.

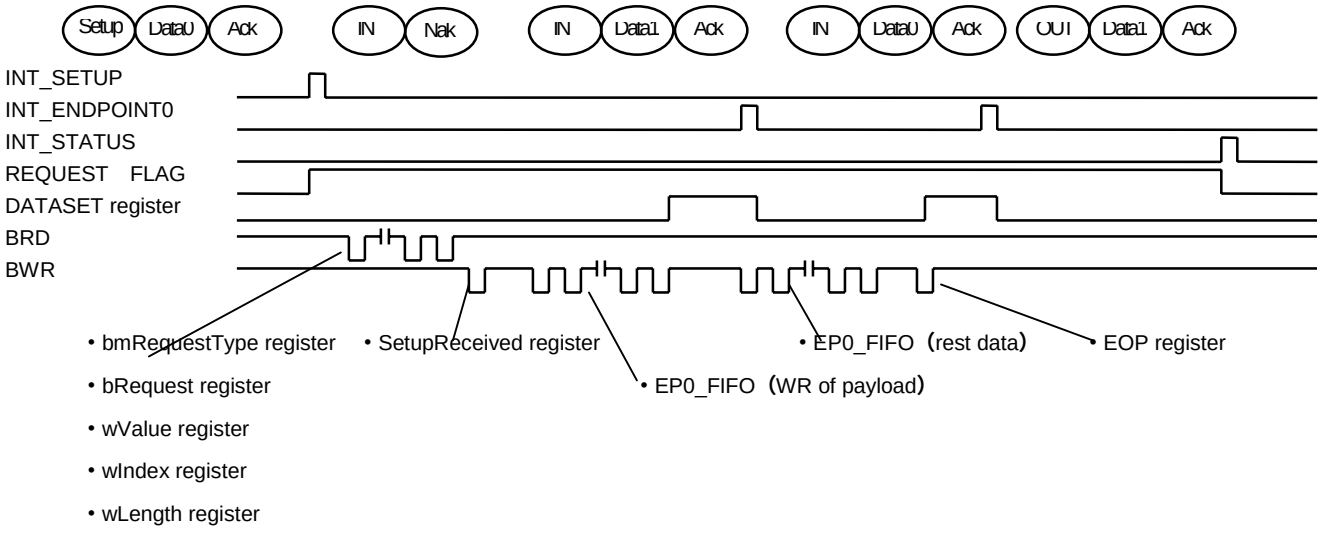


Figure 3.10.6 Figure the contorolflow in USBC（Contorol • read-transfer-type）

Stage-change conditon of contorol • write- transfer-type

＊Receive SETUPtoken from host.

- Start setup • stage in USBC.
- Receving normally data in request and discern. And assert outside INT_SETUP interrupt.
- Change data • stage into the USBC.

＊Receive OUTToken from host.

- CPU take back request from request-register when it react INT_SETUP interrupt.
- Discern request and access SetupReceivedregister for inform USBC to recognized INT_SETUP interrupt.
- Normally receive dataphase data and set EP0bit in DATASETregister.
- CPU take back data in FIFO by set DATASET register.
- CPU perform transaction receiving data by device-request.
- When CPU finish transaction, it write ”0” to EP0bit of EOPregister.
- Change statusstage into the USBC.

＊Recive INToken from host.

- Return datapacket of 0data at INtoken and change IDLE condition into the USBC.
- When receive ACK at 0datapacket, assert INT_STATUSinterrupt in outside.

These change condition is figure 5.5.

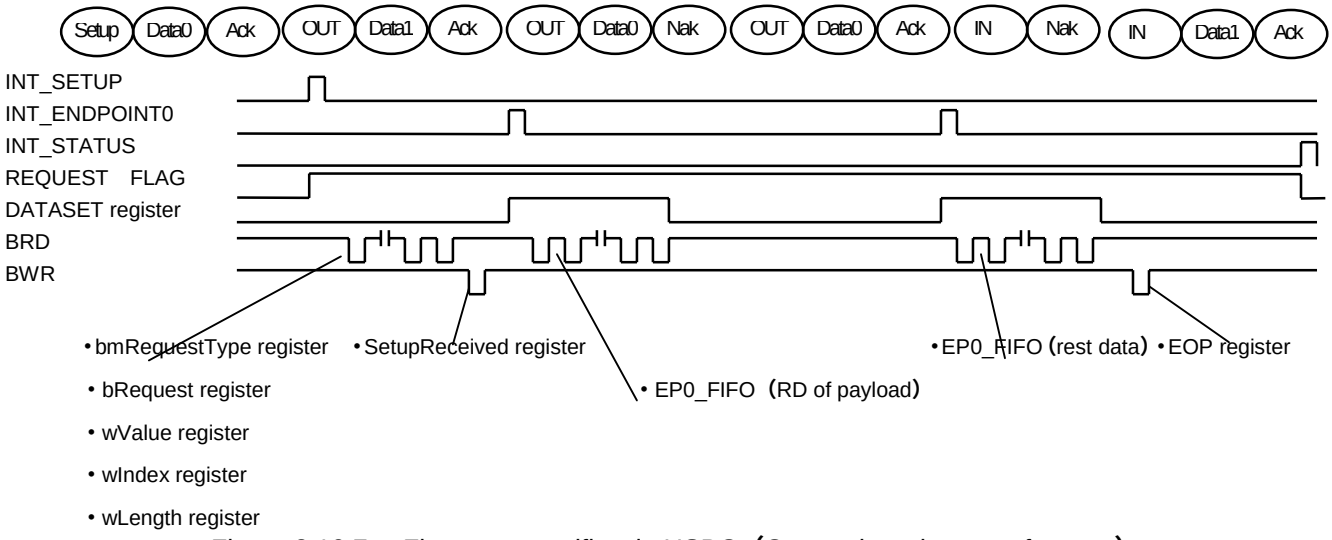


Figure 3.10.7 Figure contorolflow in USBC（Contorol • write- transfer-type）

In Contorol • read- transfer-type, transaction-number of Data-stage transaction not always accord with appointed data-number in device-request.Therefore CPU can advancing transaction make use of INT_STATUSNAK interrupt. But when make use of class and Vendorrequest, if agreeabled wLength value and data transfer-number in dataphase, must not make use of this interrupt. If don’t know datastage data, access DATASIZEregister and can confirming now receiving data-number.

Stage-change conditon of Contorol • write（no datastage）transfer-type

＊Receive SETUPtoken from host

- start setup • stage in USBC.
- receving normally data in request and discern. And assert outside INT_SETUP interrupt.
- Change data • stage into the USBC.

＊Receive INTokenfrom host

- CPU take back request from request-register when it react INT_SETUP interrupt.
- Discern request and access SetupReceivedregister for inform USBC to recognized INT_SETUP interrupt
- CPU perform transaction receiving data by device-request.
- When CPU finish transaction, it write ”0” to EP0bit of EOPregister.
- Change statusstage into the USBC.
- Return datapacket of 0data at INtoken and change IDLE condition into the USBC.
- When receive ACK at 0datapacket, assert INT_STATUSinterrupt in outside.

These change condition is figure 5.6.

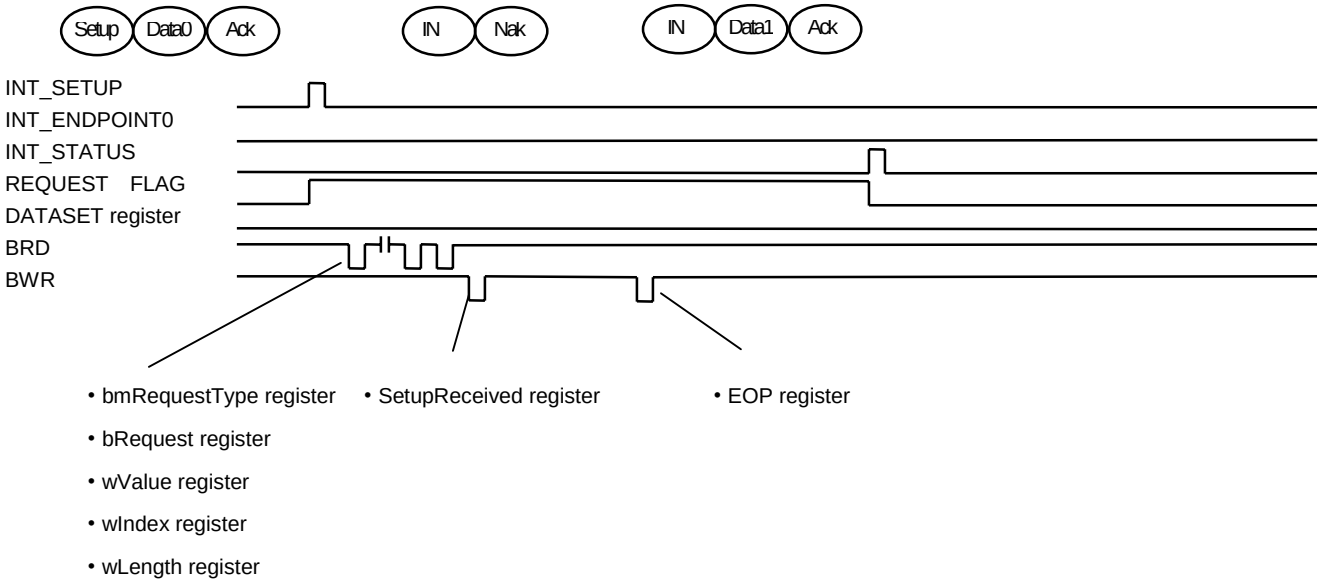


Figure 3.10.8 Figure contorolflow in USBC（Contorol • write-transfer-type • not data-phase）

(d) Isochronous-transfer-type

Isochronous-transfer-type is guaranteed loading limitted data-number each frame.

But it doesn't retry transfer by differing other three forwrding when error occurs. Therefore isochronous-transfer-type load only 2phaseof token, data and it dosen't use handshakephase. And dataphase is always DATA0 for this transaction doesn't support toggle-sequence. Therefore USBC dosen't confirm when dataPID is receiving mode.

Isochronous-transfer-type is prosecc data in every frame data, it process all transaction of finish loading by receiving SOFToken. USBC make use of dividing inside FIFO into two similarly dual-packet-mode in Isochronous-transfer-type.

(d-1) isochronous-transfer-mode

isochronous-transfer-type follow below transaction • format.

- Token : IN
- Data : DATA0

Contorolflow

Isochronous-transfer-type is frame management. And write data to FIFO in endpoint sent by IN－Token in nextframe.

Below are two conditions in FIFO isochronous sending and loading.

X. FIFO has sending data to host in now frame

(DATASET registerbit＝condition of 1)

Y． FIFO has data for sending host in next frame

(DATASET registerbit＝condition of 0)

Dividing FIFO into two (packet A, B) is fellow to condition of two condition (X, Y). Below flow is Xconditon (packetA), Ycondition (packetB) in now frame.

ConditionX and Y change one after the other by receiving SOF.

Below is contorolflow into the USBC receiving INtoken.

1) Receive tokenpacket and confirm address • endpoint-number. And transfer-mode of applicable-endpoint check wether confirm with INtoken. If it dosen't confirm, return IDLE.
2) Confirm condition of statusregister.
   - INVALIDcondition : Return to IDLE.
3) Change generation datapacket.

   Generation datapacket. At this point dataPID have to attach DATA0. Next, loading from FIFO (Xcondition) of packetA into the USBC to SIE. And generation DATApacket.。
4) Attach CRCbit counted to last transfer data of FIFO to last.
5) Below is transaction receving SOFtoken from host.
   - Change the packetA's FIFO from Xcondition to Ycondition. And clear data.
   - Change the packetB from Ycondition to Xcondition.
   - Set framenumber to frame-register.
   - Assert SOF and inform to frame increment in outside.

- DATASETregister clear packetAbit and it set packetBbit arrangement loading in now frame.
- Set STATUS to READY.

USBC finish normally.

PacketA'sFIFO can receving nextdata.

Interchange packetB'sFIFO in renewed frame and it is loaded in same flow. If don't receive SOFtoken error and so on, this data is lost by don't renewal frame. Nothing problem recieveing PID, if receving frame-data accompany CRC-error, USB set LOST to STATUS in FRAME register, and don't renew frame-number. But assert SOF and renew FIFO condition. If it receives SOFtoken nothing sending and loading of isochronous in frame, USBC clear FIFO (Xcondition) and set STATUS at FULL.

These are figure 5.7.

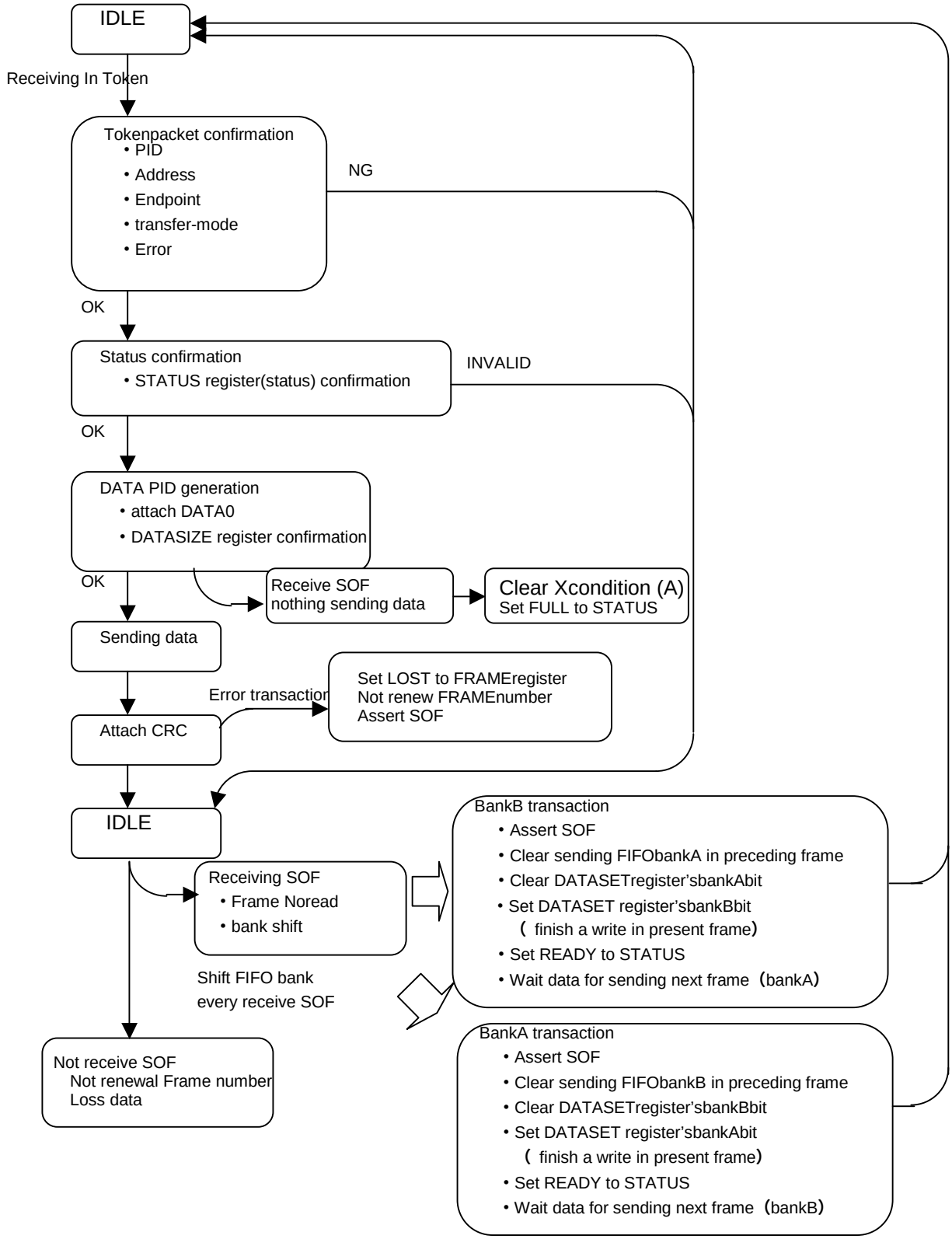


Figure 3.10.9 Contorol flow in USBC (isochronous-transfer-type (sending))

(d-2)isochronousreceiving mode

receiving isochronous-transfer-type follow transaction • format in below.

- Token : OUT
- Data : DATA0

Contorol flow

Isochronous-transfer-type is frame management. And it is taken back CPU in next frame that data writed FIFO in OUT Token.

FIFO of isochronousforawrding receving has two conditions below.

X.FIFO (FIFO has sending data to host in present frame)

(DATASET registerbit=condition of 0)

Y. FIFO(FIFO has data received from host in now frame)

(DATASET registerbit=condition of 1)

Dividing FIFO into two (packet A, B) is fellow to condition of two condition (X, Y). Below flow is Xconditon (packetA), Ycondition (packetB) in now frame.

ConditionX and Y change one after the other by receiving SOF.

Below is contorolflow into the USBC is receiving OUTtoken.

All transaction performs hardware.

1) Receive tokenpacket and confirm address • endpoint-number. And transfer-mode of applicable-endpoint check wether confirm with INtoken. If it dosen't confirm, return to IDLE.
2) Confirm condition of statusregister.
   - INVALIDcondition : Return to IDLE.
3) Receive datapacket.

   Loading data from SIE into the USBC to packetA'sFIFO (Xcondition).
4) After load last data to FIFO, it compare calculate CRC with loaded CRC.

   When finish loading, result reflects to STATUS. Data store in FIFO, set receive transfer-data-number to packetA to DATASIZE-register in packetA.
5) If receive SOFtoken from host, it process below.
   - Change packetA'sFIFO from Xcondition to Ycondition.
   - Change packetB from Ycondition to Xcondition and clear data. And prepare next loading.
   - Set FRAME register to frame number.
   - Assert SOF and it inform that frame increment outside.
   - DATASET register set packetA's bit and clear packetB's bit to store receiving data now frame.
   - If CRC comparison result agree it, it set DATAIN to STATUS.

     If doesn't agree result, set RX_ERR to STATUS.

   USBC finish normally. CPU takes back packetA's data.

This renewal stage interchanges part of packetA's FIFO and packetB's FIFO. And load same flow. If it receive SOF Token by error and so on, this data loss for renew frame. If nothing problem receiving PID USBC and receving frame-data accompany CRC-error, USBC set LOST to STATUS in FRAME register, don't renew frame-number. But in this case, SOF is asserted and FIFO condition renewal. If receive nothing transfer sending isochronous in frame, USBC clear FIFO (Xcondition), and it set READY to STATUS.

These are figure 5.8.

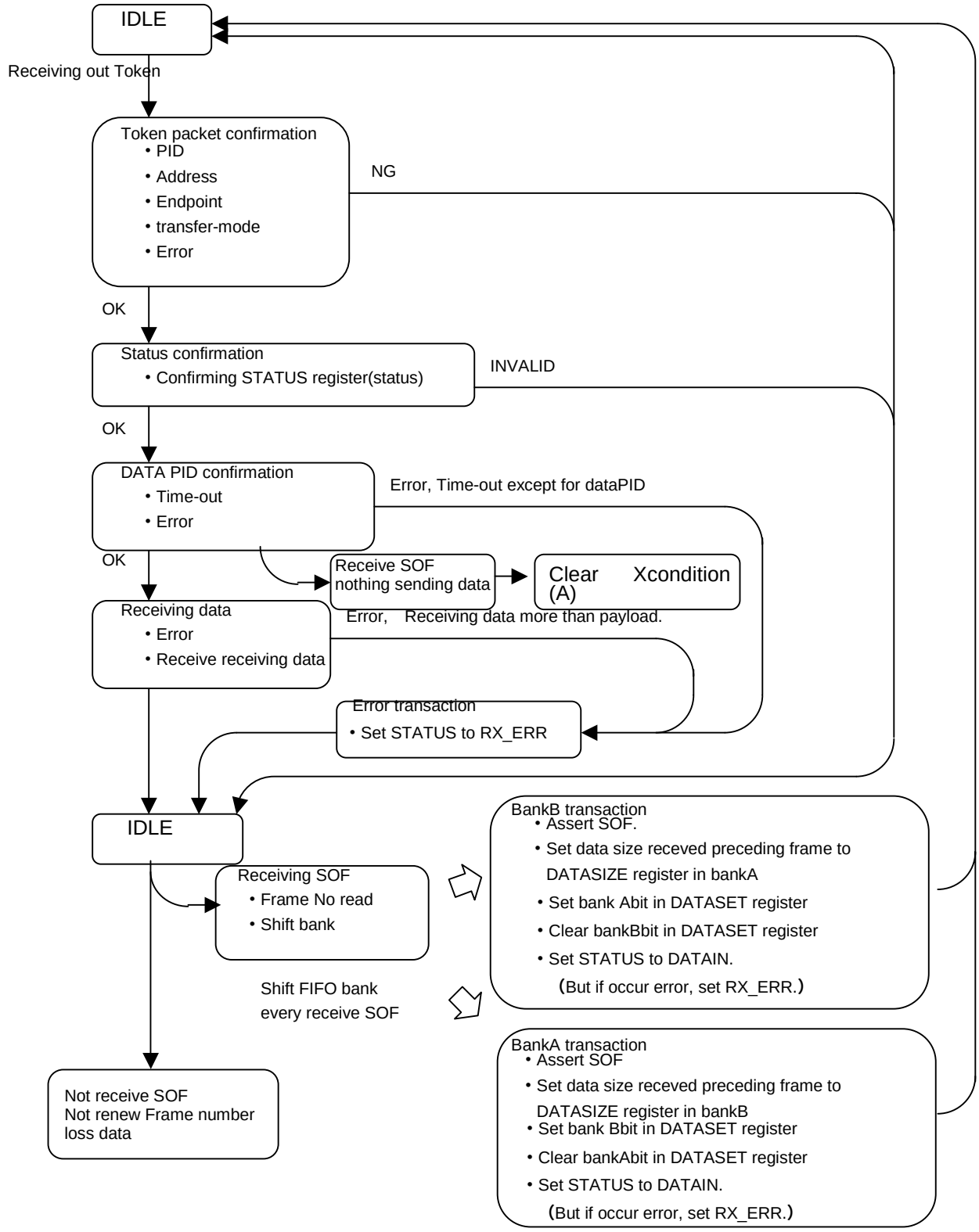


Figure 3.10.9 Contorolflow in USBC（isochronous-transfer-type（receiving））

3.10.7 Businterface and access to FIFO

(1) CPUbus • interface

USBC prepare two type FIFO-access single packet, dual packet. Single packet mode is using mode implemented FIFOcapacity as big FIFO in hardware. Dual packet mode divides FIFO capacity into two FIFO. And it use as independent FIFO. Even if USBC is sending and receiving to USB host, it can use buseefficient by to possible load to FIFO.But Contorol-transfer-type receive only single packet mode.

You must hold 0 in Epx_SINGLE signal of Endpoint using dual packet mode. If this signal hold 1,FIFO register run single mode.

Sample：If you use Endpoint 1 to dual packet of payload64byte

EP1_FIFO size ：Prepare 128byte
EP1_SINGLE signal ：Hold 0
EP1Descriptor setting
Diection ：Optional
MAXpayload size ：64byte
Transfer-mode ：Optional

(a) Single packet mode

This is data sequencein single packet mode in using CPUbusI/F.

Figure 6.1 is receivingsequence,figure 6.2 is sendingsequence.Main of this chapter is access to FIFO. Datasequencewith USB host show chapter 5.

Endpoint 0 can't change mode by exclusive single packet mode. Shift in single packet and dual packet of Endpoint 1-3 can changing by setting Epx_SINGLE register. Don't shift loading.

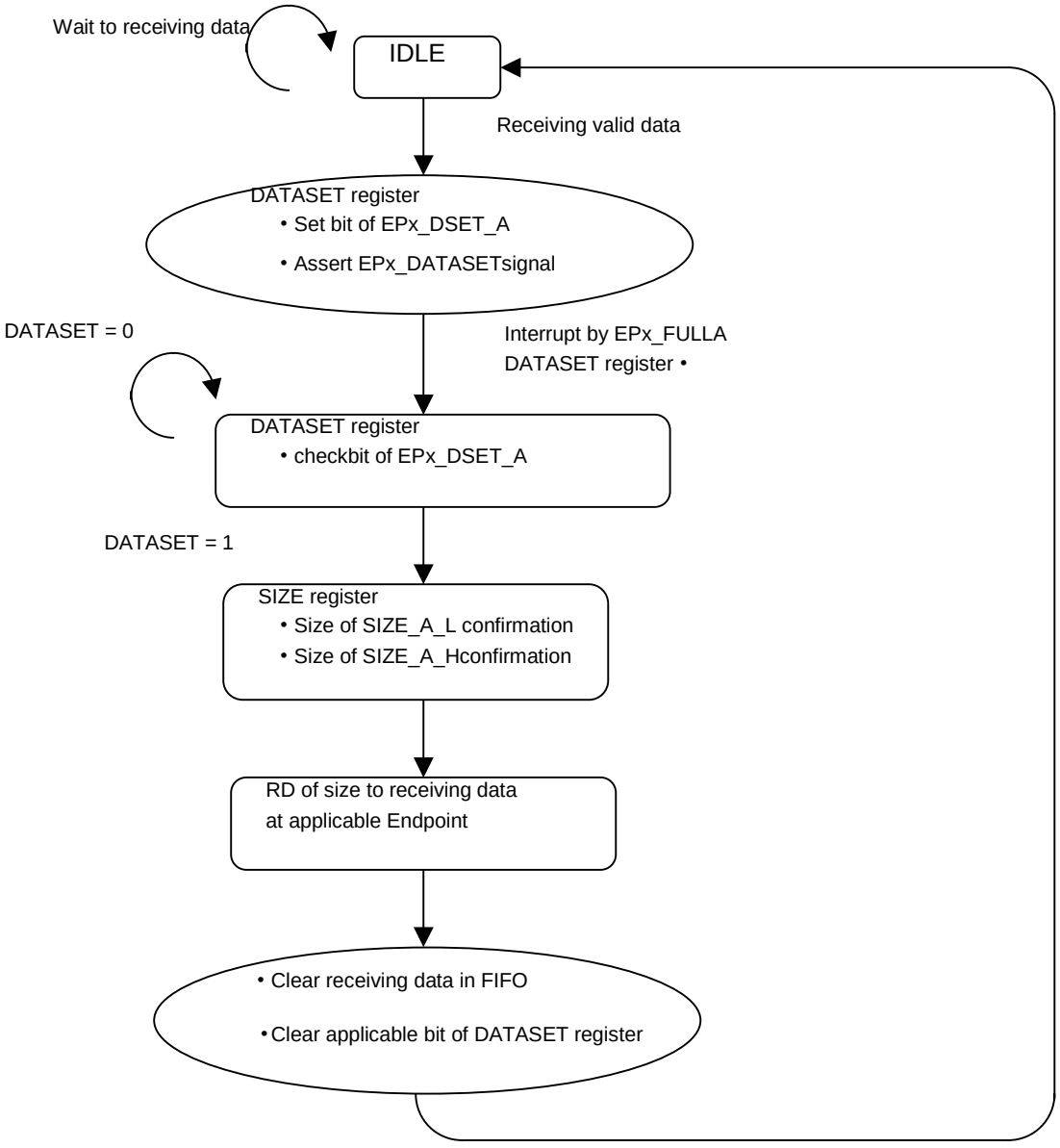


Figure 3.10.10 Receiving sequencein single packet mode

Below is sending sequencein single packet mode.

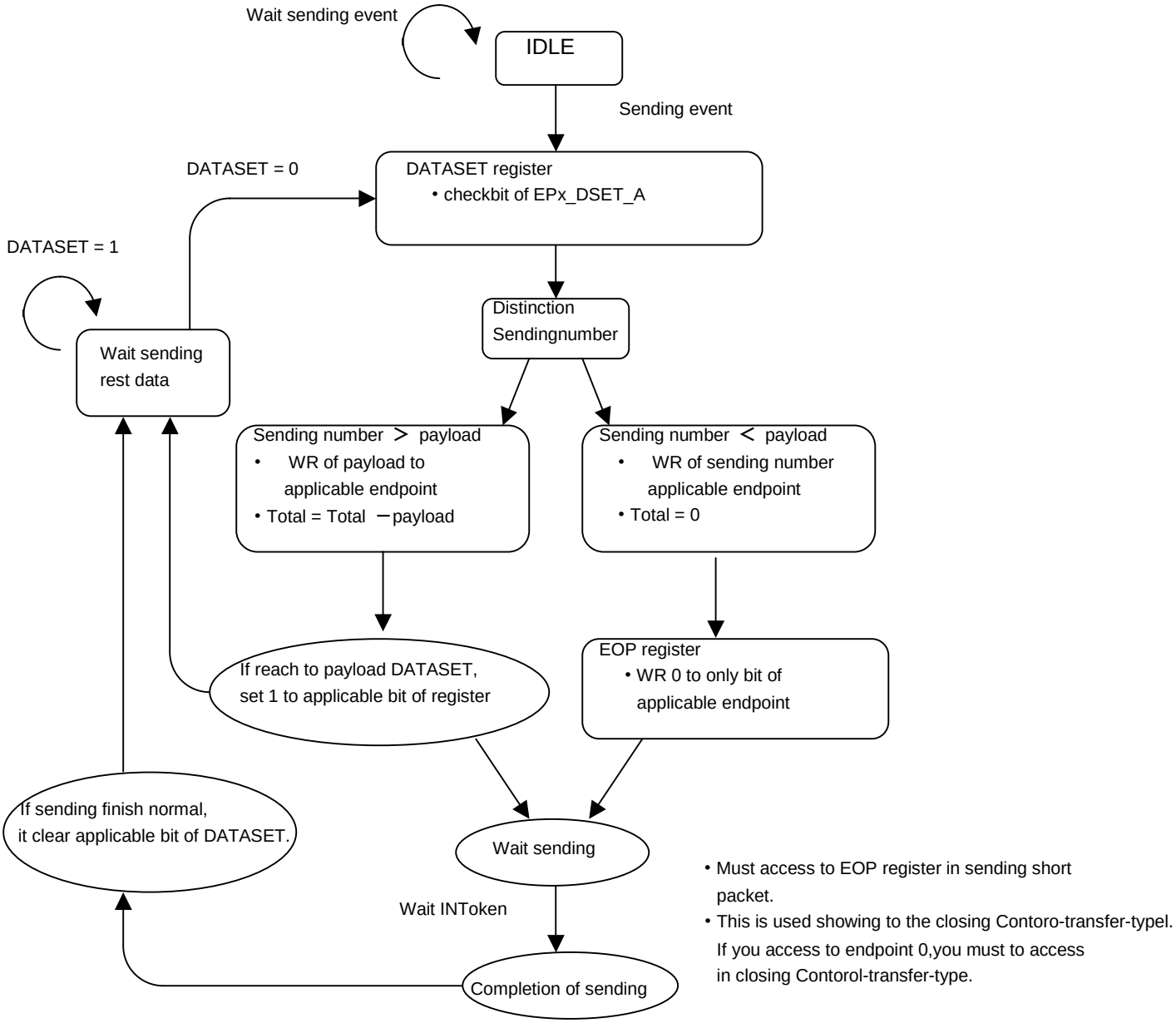


Figure 3.10.11 Sending sequencein single packet mode

(b)Dual-packet-mode

Dual-packet-mode is contorol-mode according to priority control in hardware divide FIFO into two independent packets A and B and. It can perform at once sending data and receiveing data to USB host and exchange to outside. When it read out data from FIFO for receiving, you confirm conditon of two packets, and you must to consider the order of priority. If it has received data to two packets, USBC out put from first received data received by access possible FIFO common two packets. DATASIZE register is prepared every A, Bpacket. CPU must grasp data number of first received packet make use of PACKET_ACTIVEbit. First receive packet is PACKET_ACTIVEbit set 1. Apacket and Bpacket set data turn about always.

Below is thissequence.

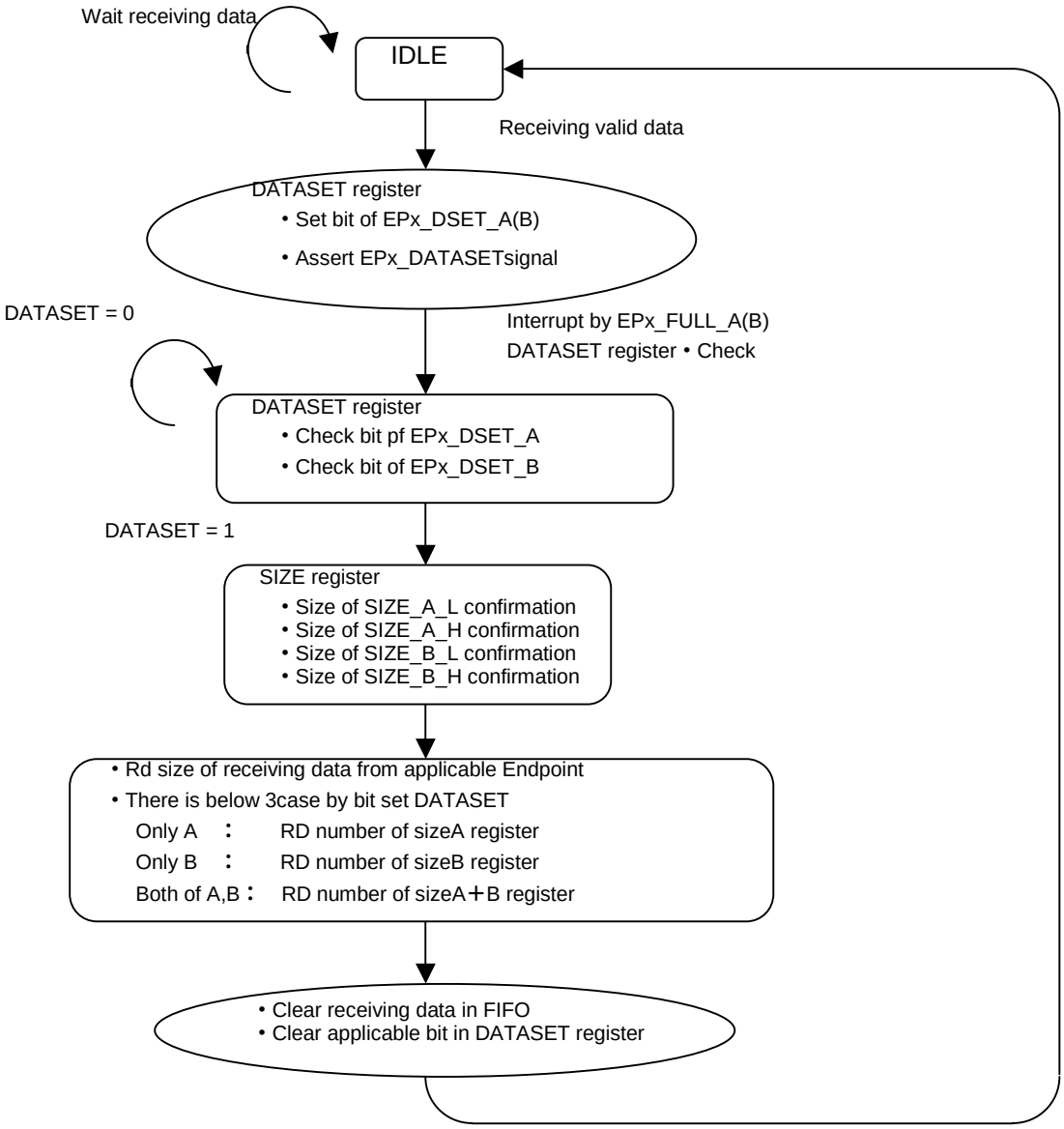


Figure 3.10.12 Receiving sequencein dual packet mode

When you write data to FIFO in sending, you confirm condition of two packets, and you must load after you consider order of loading packetA, B. When you set loading data number, set to which A，Bpacket, you judge by showing PACKET_ACTIVEbit. This packet of Bit0 is loading packet now.

You must caution that logic of PACKET_ACTIVEbit in receiving and sending is reverse.

Below is this sequence.

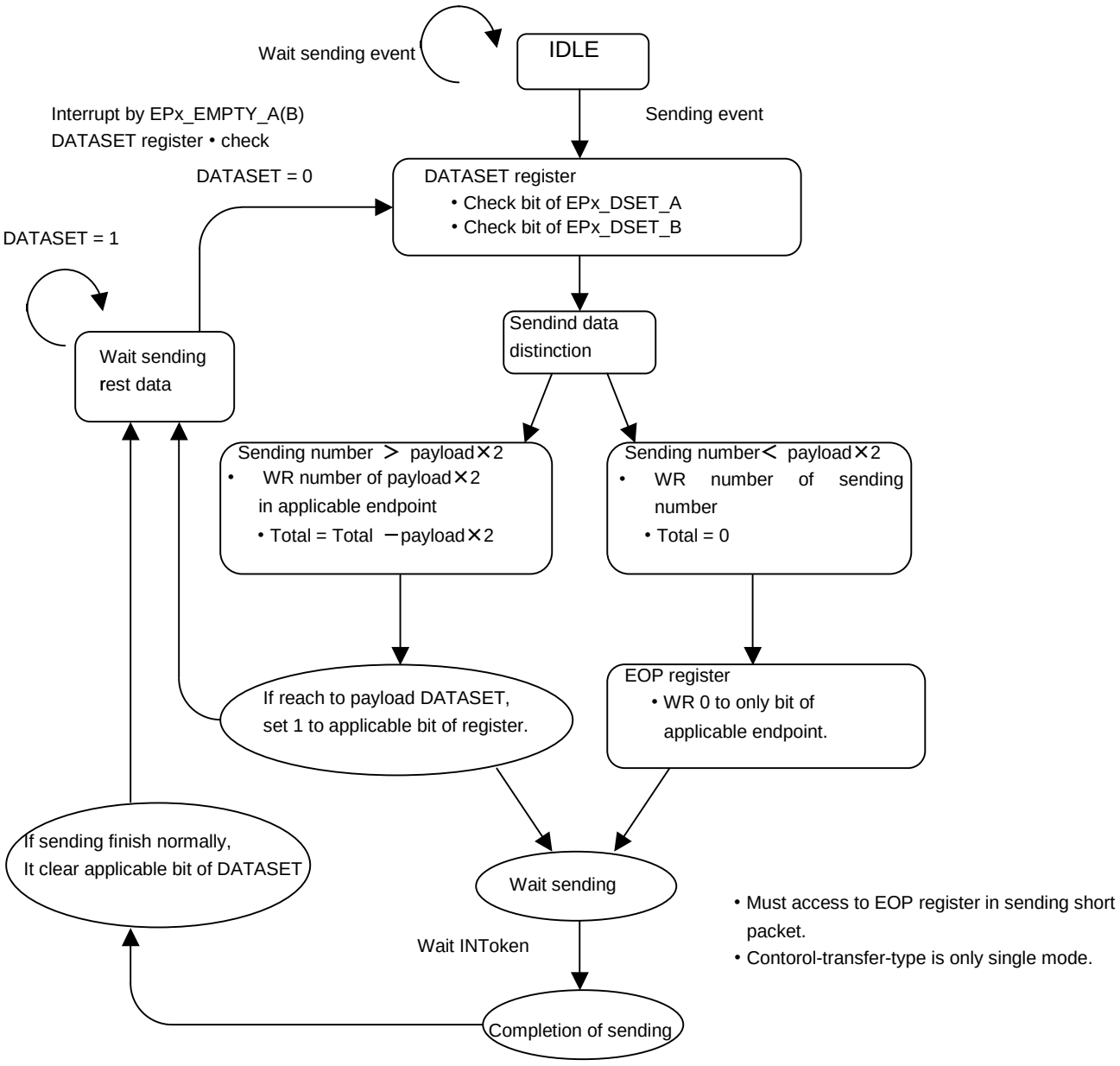


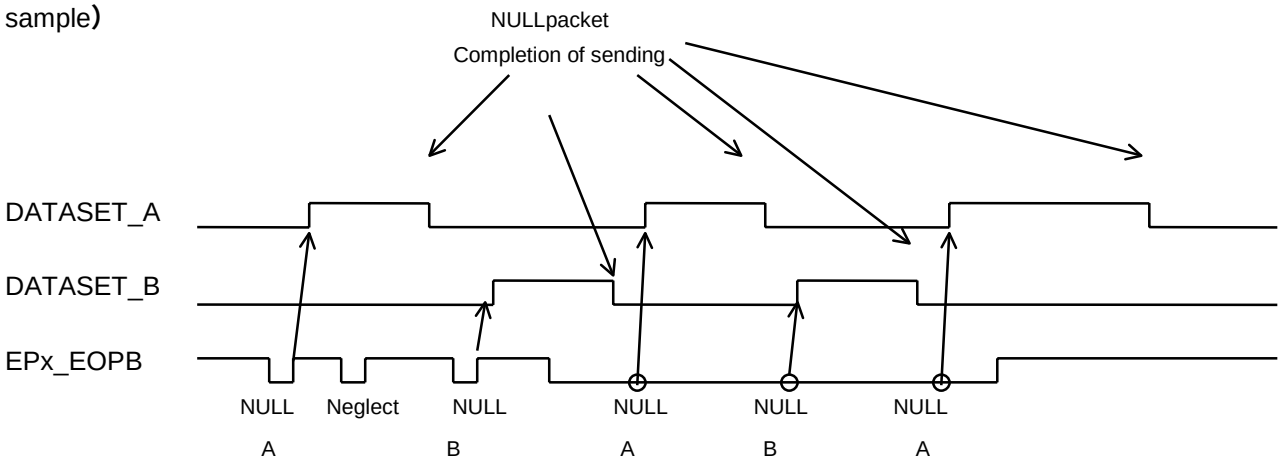
Figure 3.10.13 Sending sequencein dual packet mode

(c) Issuance of NULLpacket

if sending NULLpacket,by input L pulse from EPx_EOPB signal, data of 0rength set to FIFO, and you can sending NULLpacket to INToken.

But If set NULLdata to FIFODATA, it valid only SET signal Llevel condition（occasion of don't have data in FIFO）. If answer with every NULLpacket to receiving IN Token in certain period, it can to answer by keep EPx_EOPBsignal to Llevel.

sample）



(2) Interrupt contorol

Interrupt signal is prepared. This function makes use of adept system.

Detail refers to 3.10.2 900/H1 CPU I/F.

3.10.8 USB device-answer

USB controller (USBC) set various register and initialization in USBC in detecting of hardware reset, detecting of USBbusreset, and enumeration answer.

Below is explaining about each condition.

(1) Condition in detect in busreset.

USBC initialize in register detect in busreset on USB signal line and it prepare enumerationmovement from USB host. After detect in USB reset, USBC set ENDPOINT0 to Contorol-transfer-type 8byte payload and default address by to enable using default pipe. And set prohibintion condition of endpoint except for it.

Register name	Initial value
ENDPOINT STATUS EP0	40H
Except for EP0	5CH

(2) Detail of STATUS register

Prepared ststus register every endpoint mean condition of every endpoint in USBC.

Each condition affects loading various USB. Change condition in each transfer-type refers to chapter 5.

Endpoint • status register value express below condition in 0-3.Expression to 0-4 is result of various loading. It can confirm on the eve loading result to endpoint by confirming to outside USBC.

0 READY
1 DATAIN
2 FULL
3 TX_ERR
4 RX_ERR

These condition mean to normally run endpoint. Differ meaning of expression by every transfer-mode. Therefore you 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)
In Initial	READY	READY
Not transfer	READY	FULL
Normally finish	DATAIN	READY
Detect in error	RXERR	TXERR

Transfer-mode of except for ISO transfer

This is result previous transfer. When transfer finish,it is renewed.

	OUT,SETUP	IN
In Initial	READY	READY
Normally transfer finish	DATAIN	READY
Status-stage finish	READY	READY
Transfer error	RXERR	TXERR

In initial is while it renews RESET, USB reset and Current Config register. On detect in error, it dosen't occur EPx_DATASET except for toggle-transfer-mode and isochronous-transfer-mode in interrupt.

5-7 in expression of status register mean to endpoint is special condition.

5 BUSY BUSY occur only endpoint perform Contorol-transfer-type. When USBC perform contorol-write-transfer, in CPU dosen't finish condition with enumerationtransaction .And If it is receving ID of status stage from USB host USB, it is set. CPU finish enumerationtransaction, and STATUS is BUSY until written 0 to EP0bit of EOP register in USBC. CPU finish enumerationtransaction, and written 0 to EP0bit of EOP register,and if it normally finish status stage from USB host,it display READY.

Please refer to 5.2.3 in chapter 5.

6 STALL STALL mean to endpoint is STALLcondition.

This condition occur case of violate protocolor or error in busenumeration. If return Endpoint to normally transfer possible condition, want device-request of USB. This request returns normally condition. But Contorol endpoint returns normally condition by receiveing SETUP Token. And into SETUP stage.

7	INVALID	This condition mean can’t using endpoint.USBC set INVALID condition to isn’t apointed ENDPOINT, and it ignore all of Token to this endpoint. When initialized it, this conditionoccur always. When USBC detect in hardware reset, it set INVALIDconditionto all endpoint. Next, if receiving USB reset, it is renewed READY onlyendpoint0. Other endpoint defined on descriptor if that normally finish, that is renewed.

3.10.9 Power management

USB contoroller (USBC) can switch from optional resume condition(turn on the power supply condition) to suspend(suspension)condition,and return from suspend condition to turn on the power supply condition.

This function can having low electricity consumption by oparating CLK suppling for USBC.

(1) Switch to Suspend condition

USB hostca setting suspend condition to USB device by keep on IDLE state. USBC switch to suspend condition by below process.

- USBC switch to suspend condition if detect in successive IDLE state of more than 3ms on USB signal. At thi point, set 1 to SUSOENDbit of STATUS register. After switch to suspend condition, if besides pass away 2ms, USBC renew from 1 from 0 to USBINTFR1<INT_SUS>. USBINTFR1＜After INT_CLKSTOP＞ renew 1 from 0, setting 0 to USBCR1<USBCLKE>,and stopped into the supply of CLK(USB_CLK).
- In this condition, register value into the USBC keep on all of value,but access from outside except for reading of STATUS register, Current Config register,and USBINTFR1,2, USBINTMR1,2, USBCR1 can't perform.

(2) Return from suspend condition by host resume

Way to USBC change from suspend condition to resumecondition have two type. instance of resumecondition output from USB host and instance of remote wakeup.

When restore to activity of bus on USB signal by resumecondition output from USB, USBC reset 0 from 1 to SUSPEND output,and it reset 0 to SUSPENDbit of STATUS register. And it resumed system.resumecondition output from this host keep on no less than during 10ms. Therefore effectiveprotocoloccuring on USBsignal line is after pass away this time.

(3) Return from suspend condition by remote wakeup

remote wakeup is system for prompt toresumefrom suspending USBdevice to USB host. Remote wakeup isn't supported by condition. And remote wakeup is limited using from USB host by busenumeration.

Function of remote wakeup in USBC can use when it is permited.

Setting remote wakeup by bus can confirm bit7 of Current Config register. When this bit is 1, remote wakeup can use. Remote wakeup dosen't disable in this bit. Therefore if this bit express disable,you must not perform remote wakeup. If it fill these condition, output resumecondition output to USB host by to WR "1→0" to USBCR1<WAKEUP> of USBC in suspend condition. and it prompt toresume from USBC to host. After USBC change to suspend condition, during 2ms ignore WAKEUPinput. Therefore remote wakeup is effective by to USBINTFR1<INT_SUS> is 1.

(4) Low power consumption by contorol of CLK input signal

When USBC switch to suspend condition,it stop in CLK and switch to low power consumption condition. But this function enable besides low power consumption by stop to source of CLK supplied to outside as system. CLK supply to USBC can contorol clock supply to USB makes use of USBINTFR1<INT_SUS> and <INT_CLKSTOP>.

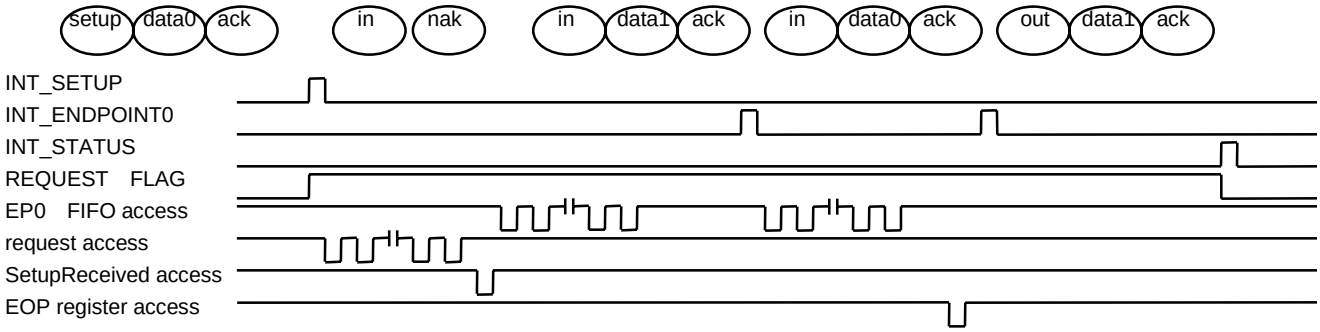
You must move like next. If USBC switch to suspend condition,USBINTFR1<INT_SUS> is 1,and <INT_CLKSTOP > is 1. After confirmation, stop supply CLK (USBCLK) by setting “0” to USBCR1<USBCLKE>. By resumefrom host if SUSPEND signal is 0, you must supply normally CLK to USBC within 3ms.

And whe you use remote wakeup, you must supply stable CLK to USBC before using.

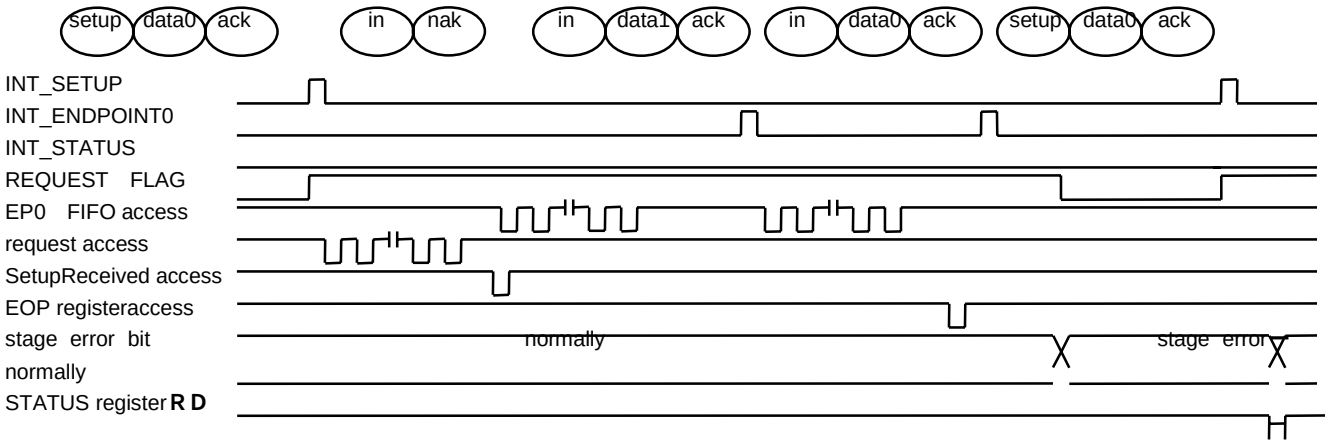
3.10.10 Supplement

(1) Outside 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		Register name	Beginning value Outside reset	Beginning value USB_RESET
BmRequestType	0x00	0x00		INT_Control	0x00	0x00
Brequest	0x00	0x00		USBBUFF_TEST	0x00	Hold
wValue_L	0x00	0x00		USB_STATE	0x01	0x01
wValue_H	0x00	0x00		EPx_MODE	0x00	0x00
wIndex_L	0x00	0x00		EPx_Status	0x1C	0x1C
wIndex_H	0x00	0x00		EPx_Size_L_A	0x88	0x88
wLength_L	0x00	0x00		EPx_Size_L_B	0x08	0x08
wLength_H	0x00	0x00		EPx_Size_H_A	0x00	0x00
Current_Config	0x00	0x00		EPx_Size_H_B	0x00	0x00
Standard_Request	0x00	0x00		Frame_L	0x00	0x00
Request	0x00	0x00		Frame_H	0x02	0x02
DataSet	0x00	0x00		Address	0x00	0x00
Port_Status	0x18	Hold		EPx_SINGLE	0x00	Hold
Standard_Request_Mode	0x00	Hold		EPx_BCS	0x00	Hold
Request_Mode	0x00	Hold		ID_STATE	0x01	0x00

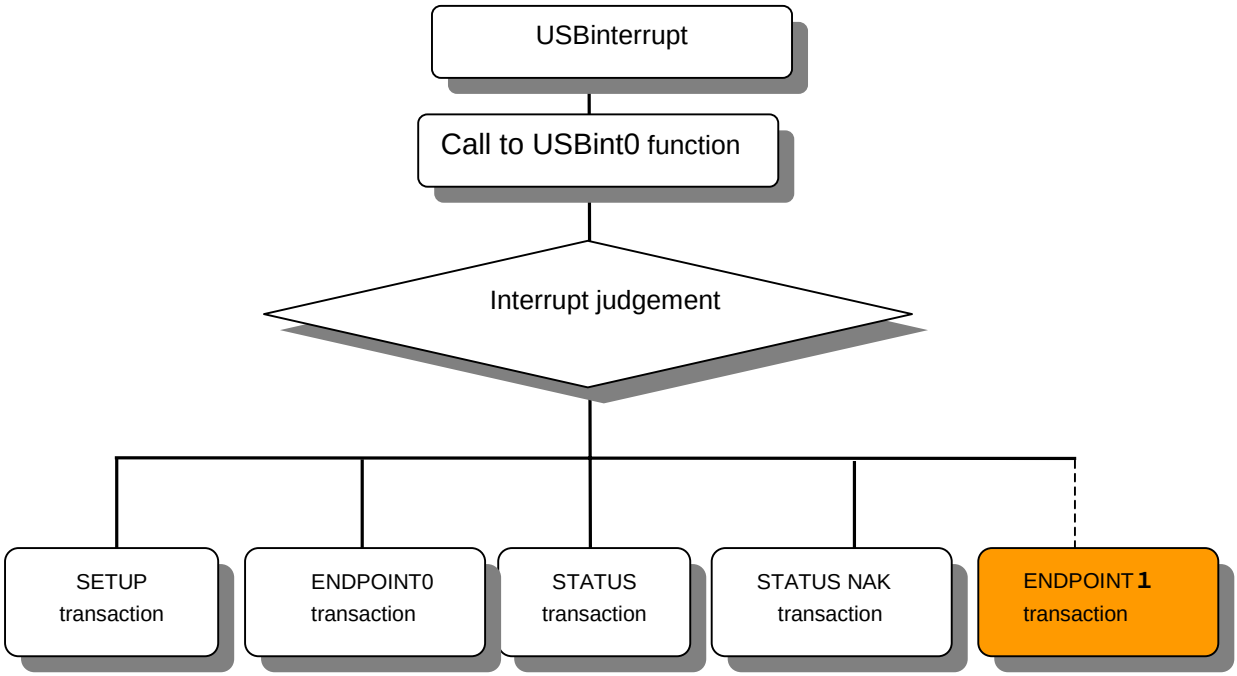
Note1) Above beginning value is initialized value by outside reset, USB_RESET.This value differs indicate value by various condition.
Please refer to register structure of chapter 2.

Note2) Beginning value of EPx_Size_L_A, EPx_Size_L_B, EPx_Size_H_A, EPx_Size_H_B register differ by size of FIFO.
EP0_STATUS register is initialized 0x00 after receive USB_RESET.

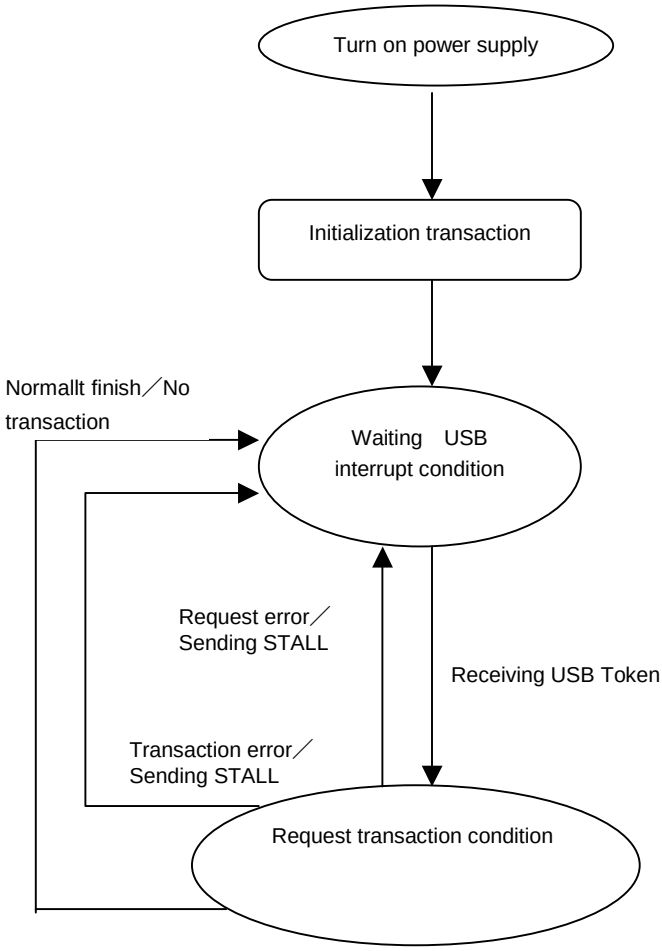
Note3) Beginning value of ID_STATE register is initialized value by outside reset. When rececе USB_RESETsignal to host,it isinitialized 0x00.

(3) USB contorol flow chart

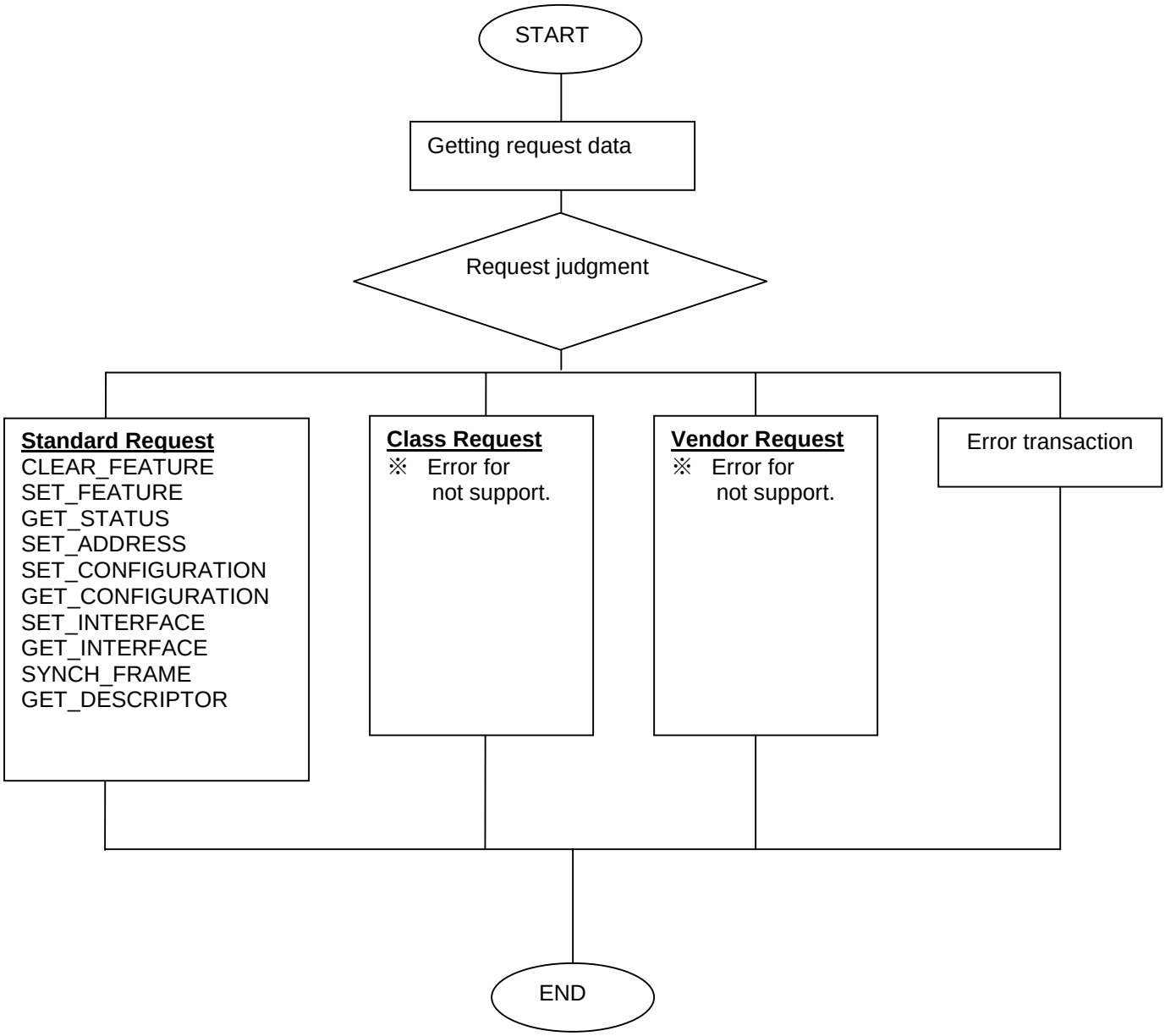
(a) Transaction to standard request (outline flowchart (exsample))



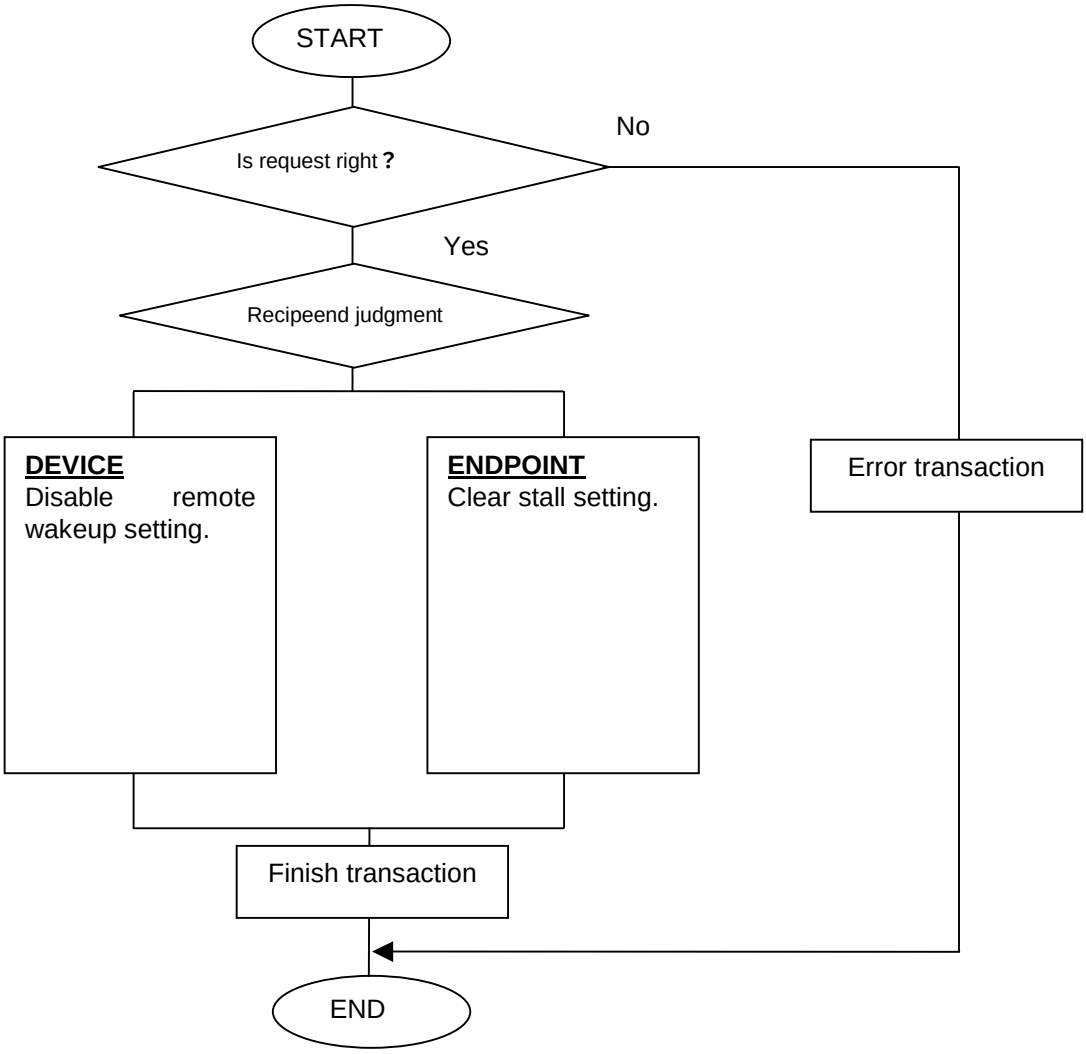
(b) Condition change



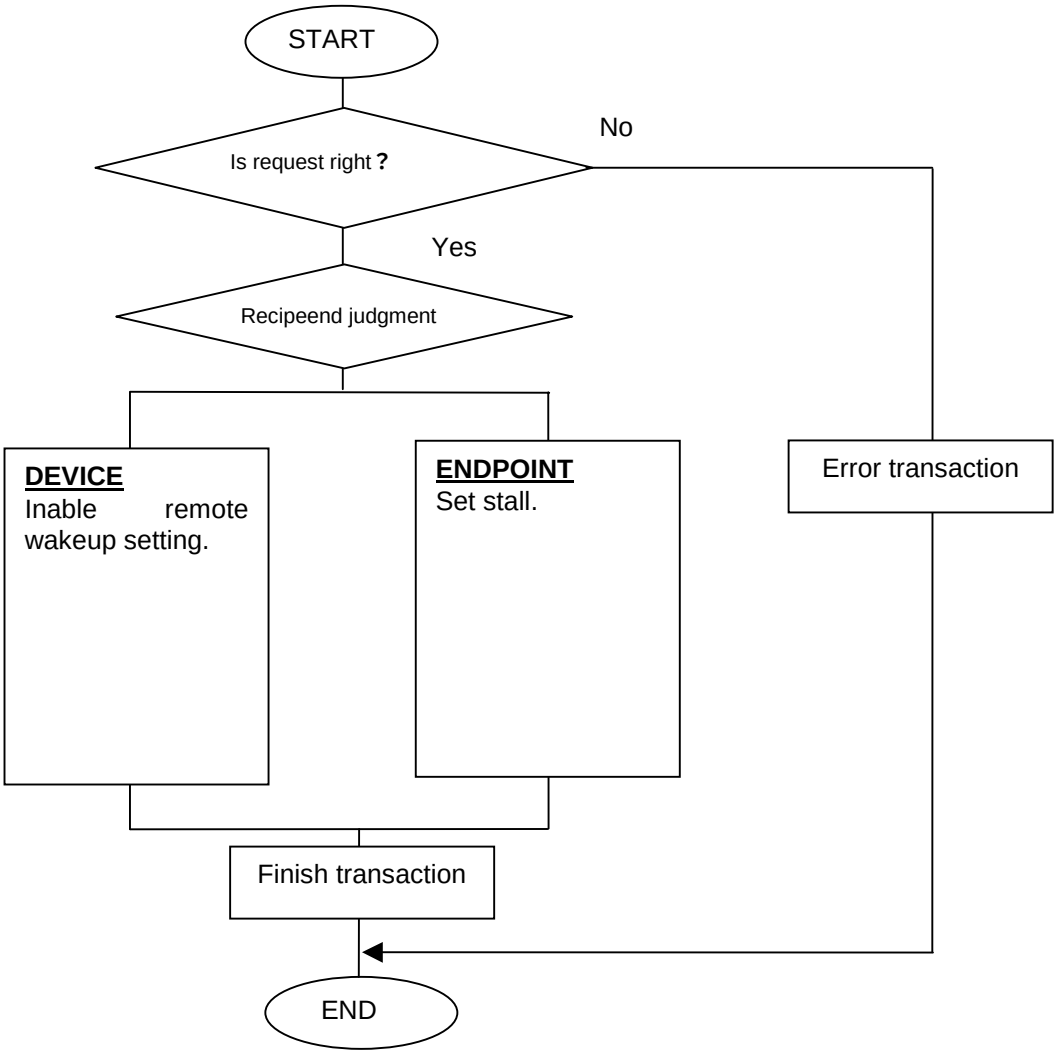
(c) Getting device request and various request judgment



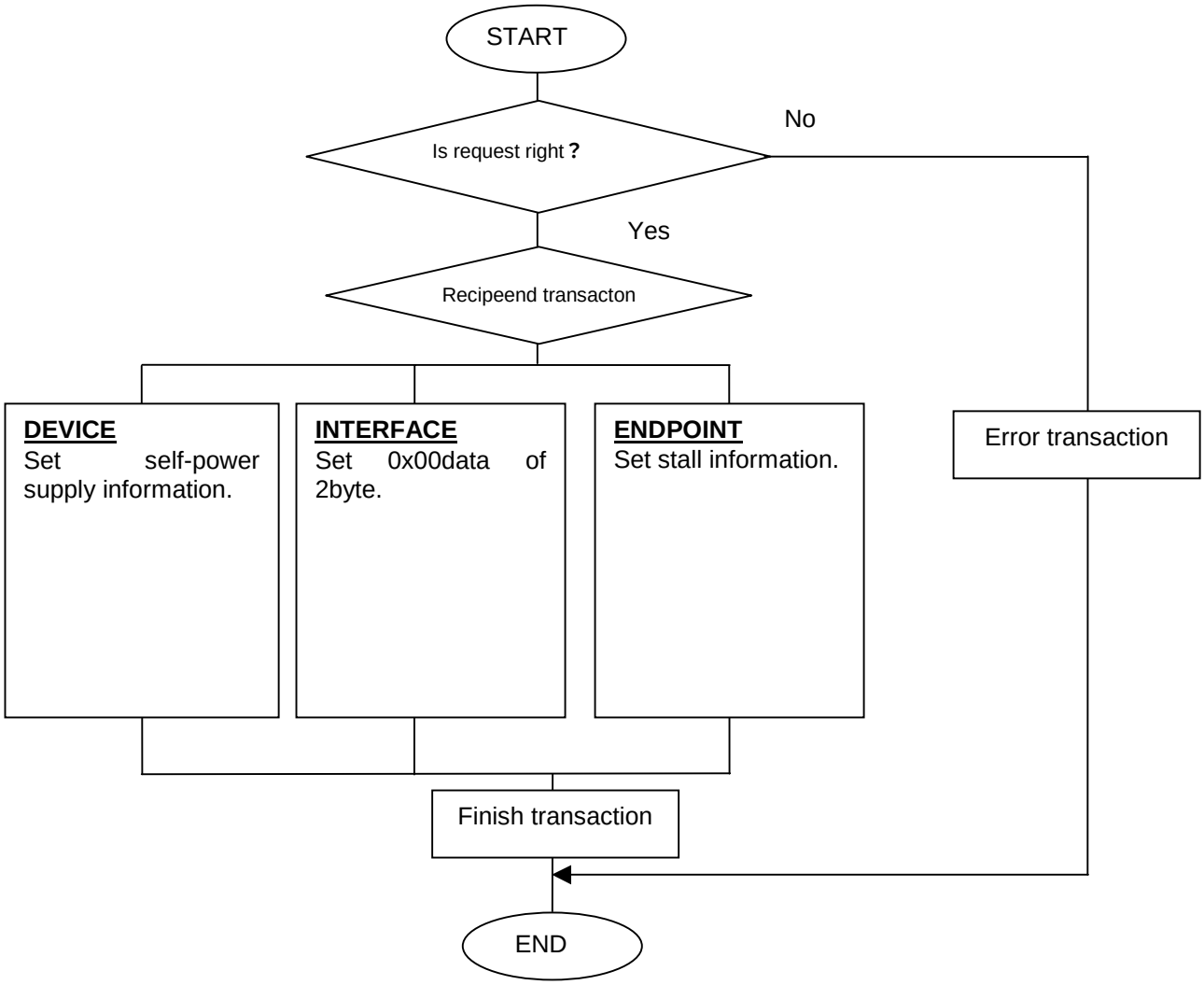
(c-1) Clear Featurerequest transaction



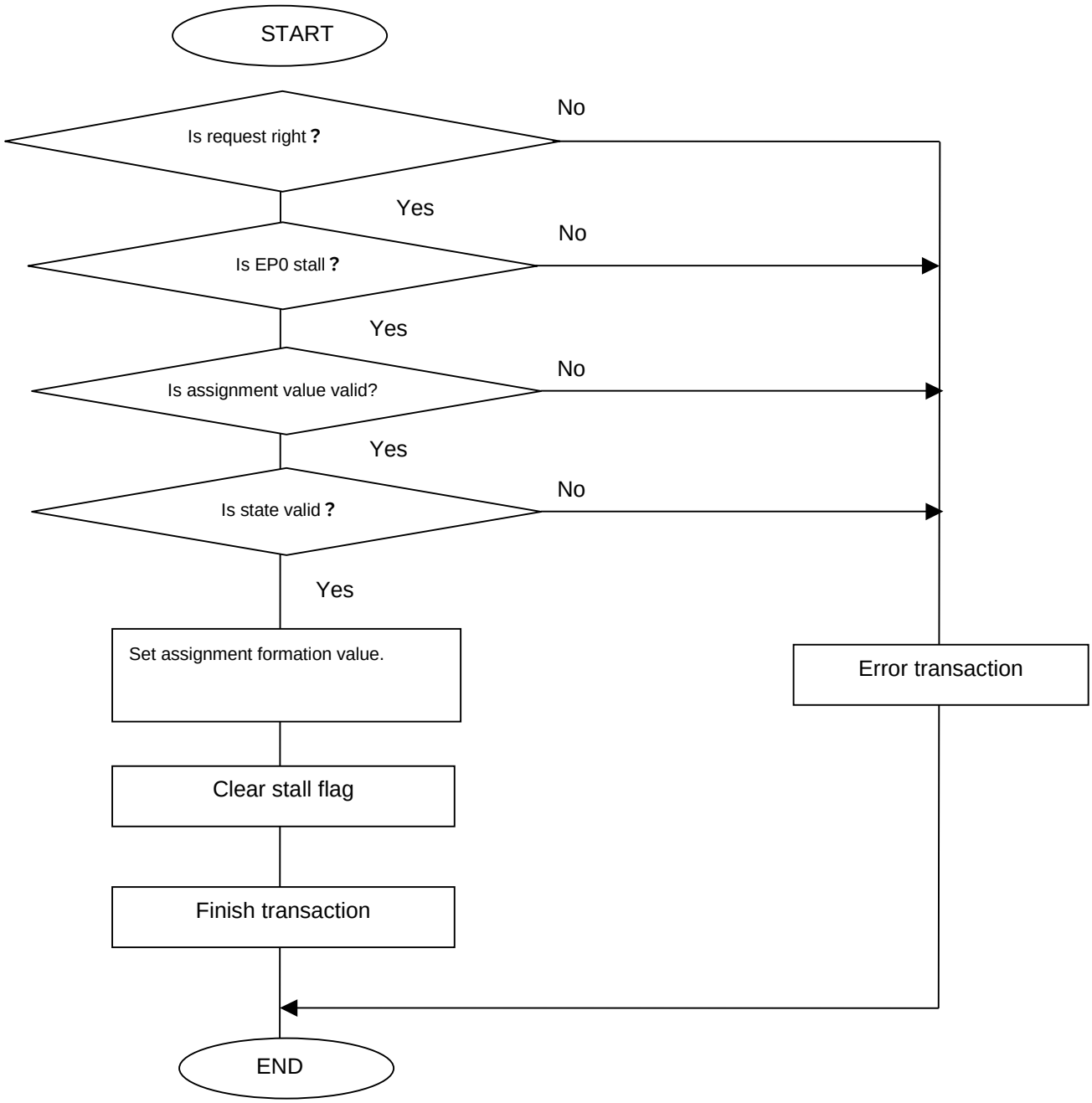
(c-2) Set Featurerequest transaction



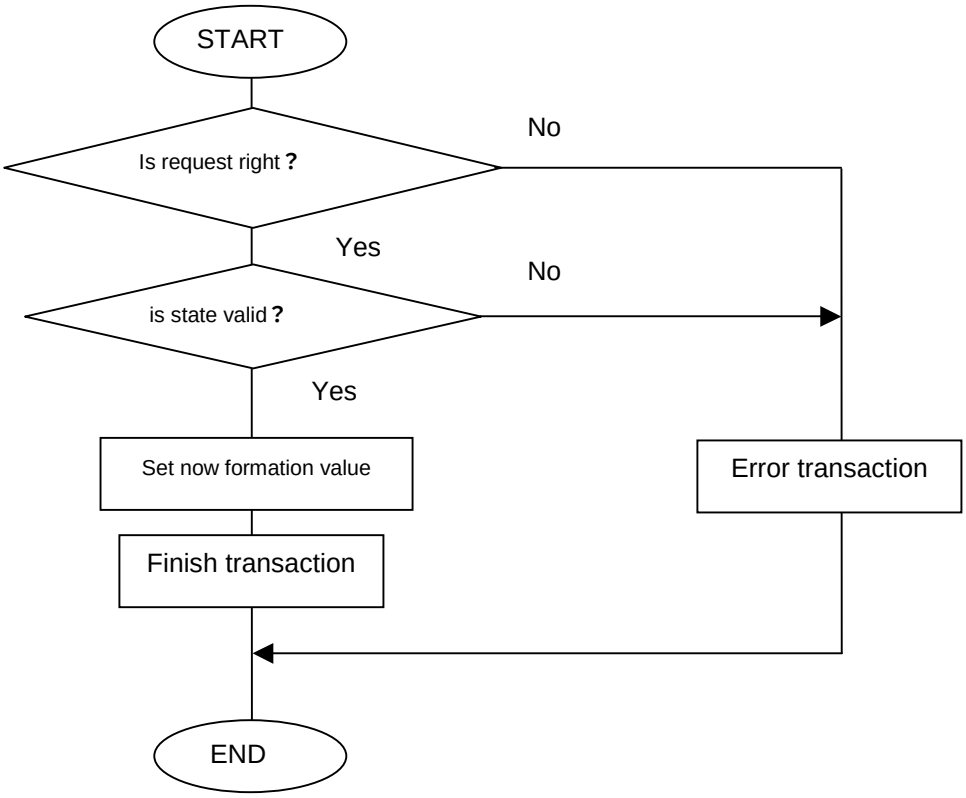
(c-3) Get Status-request transaction



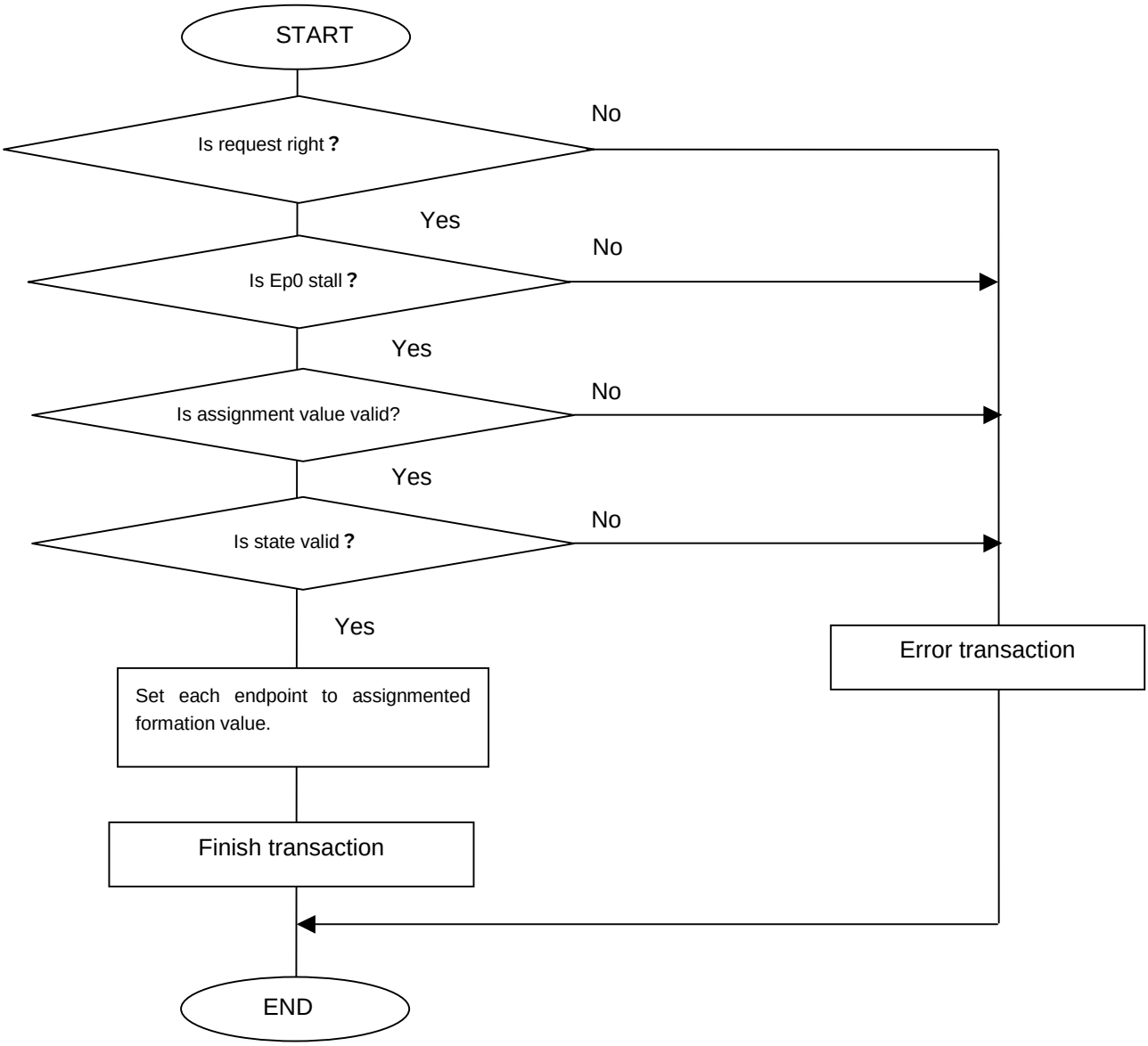
(c-4) Set Configration-request transaction



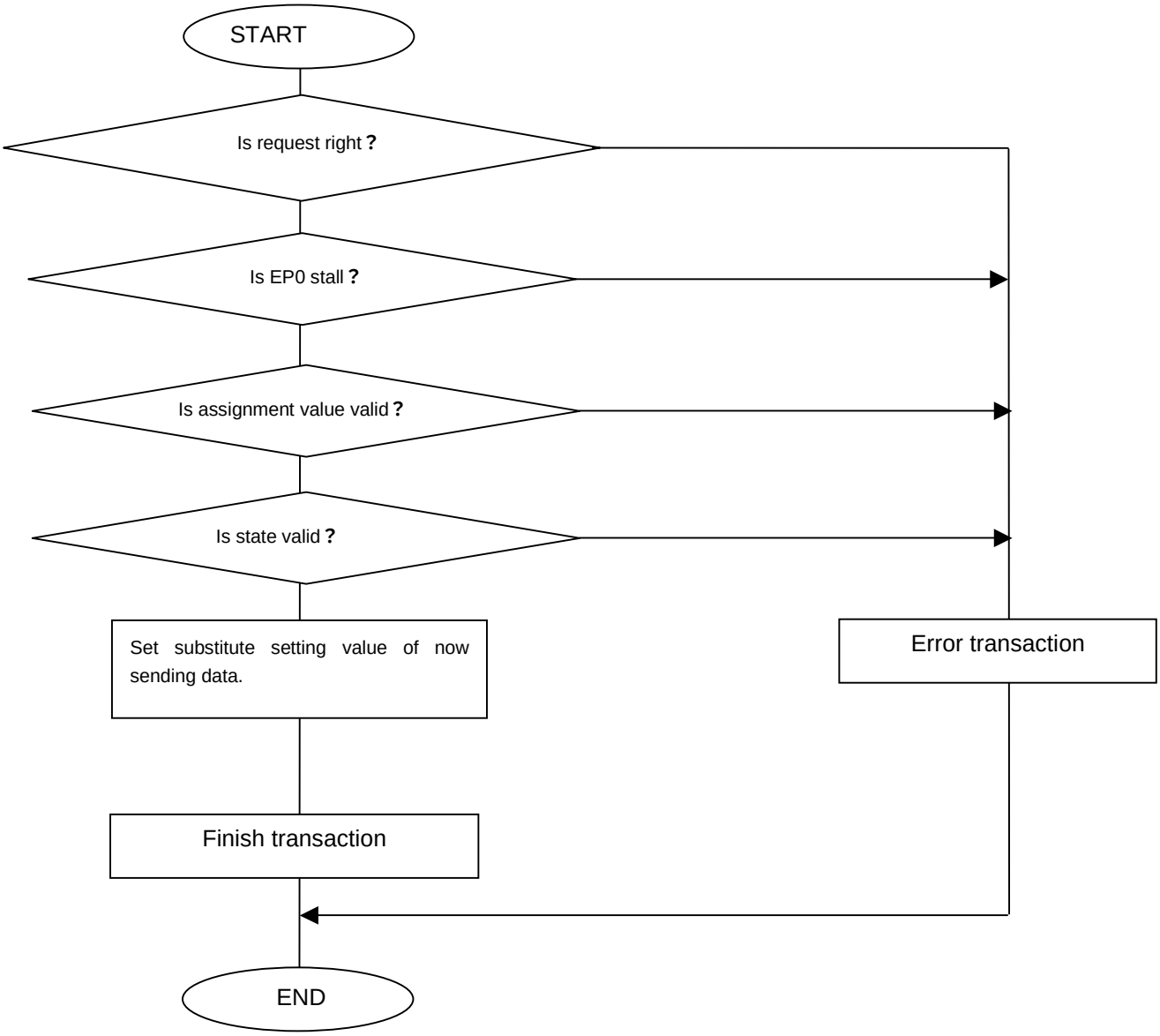
(c-5) Get Configration-request transaction



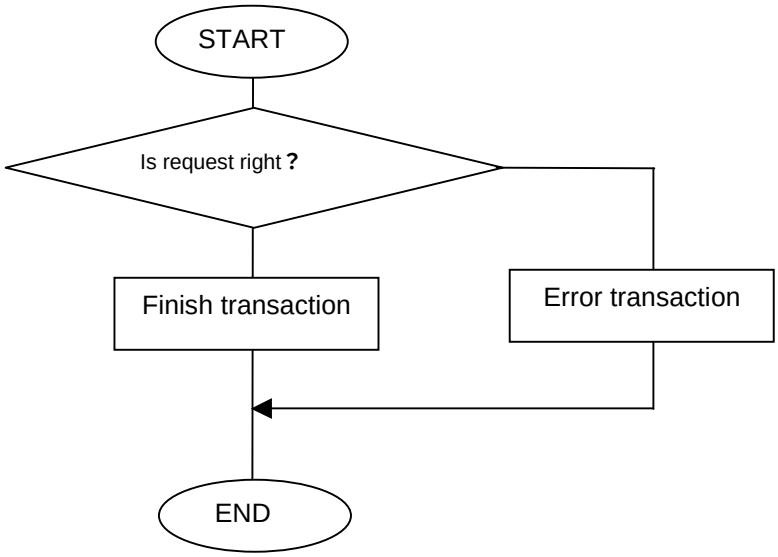
(c-6) Set Interface-request transaction



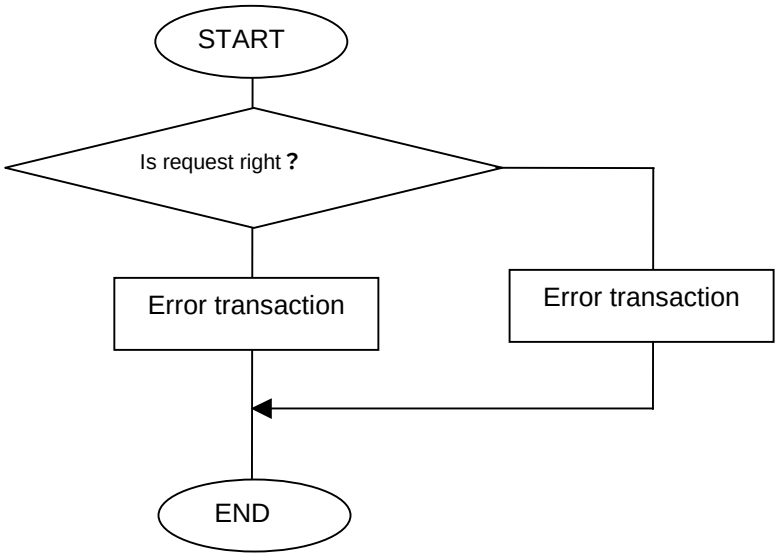
(c-7) Synch Framerequest transaction



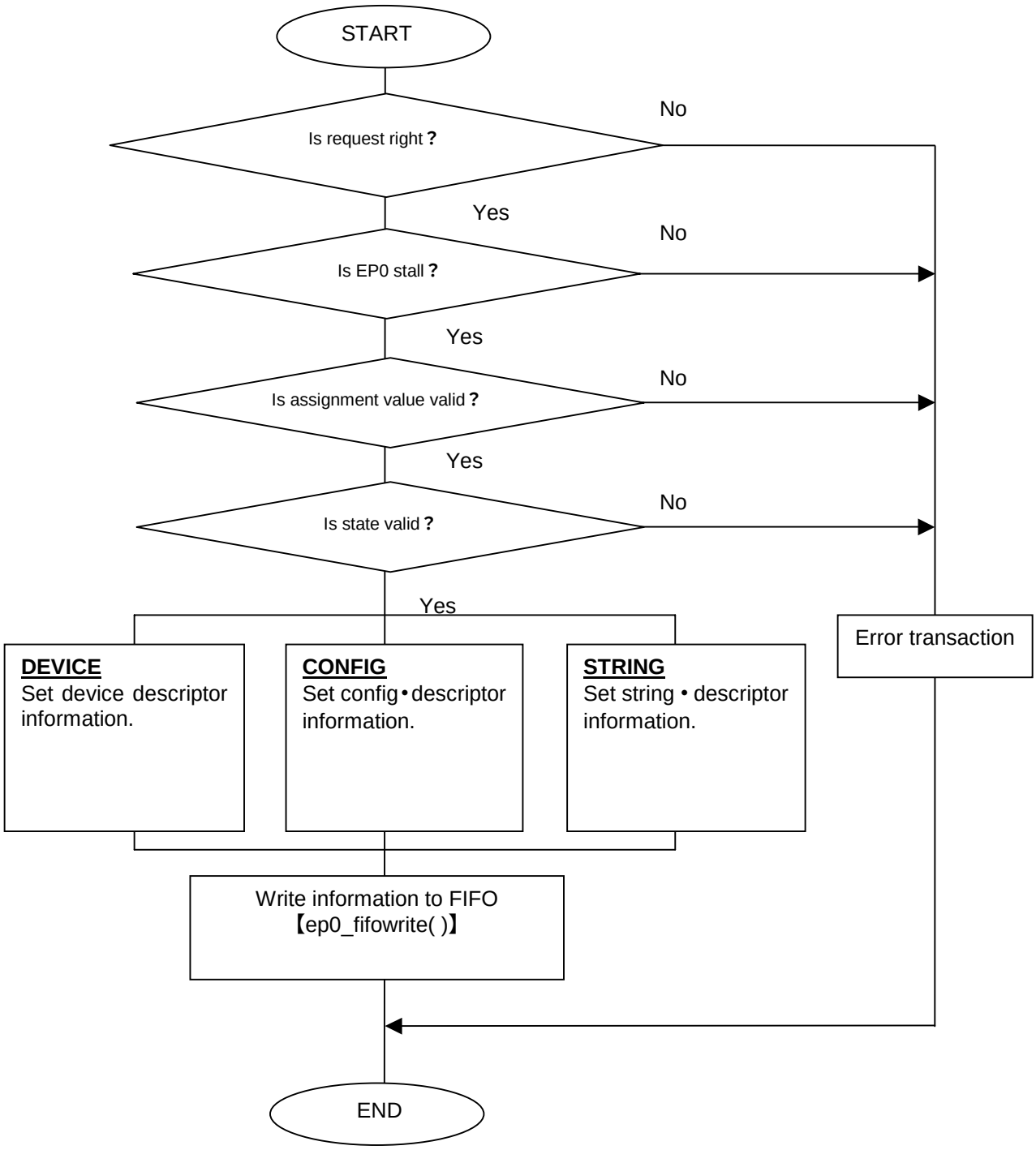
(c-8)Synch Frame-request transaction



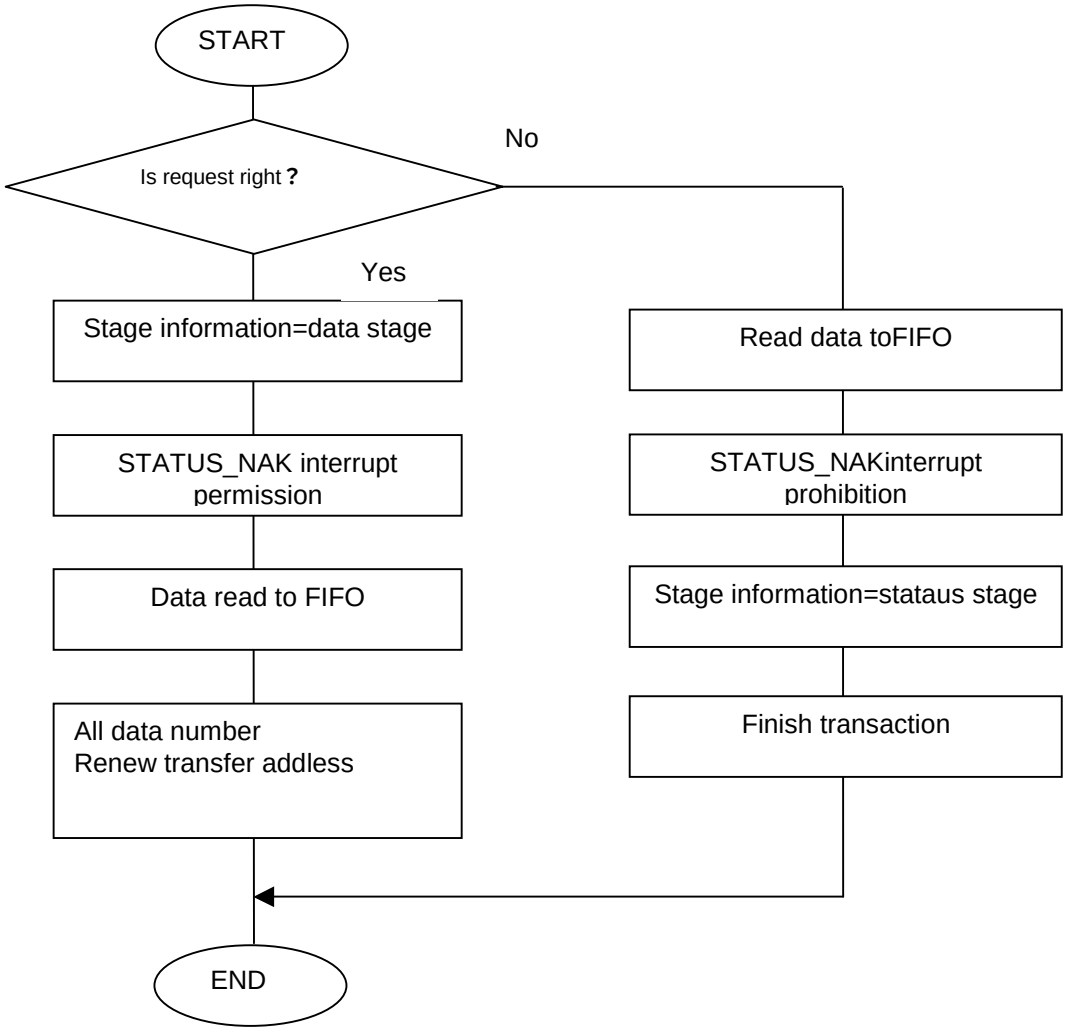
(c-9) SET DESCRIPTOR-request transaction



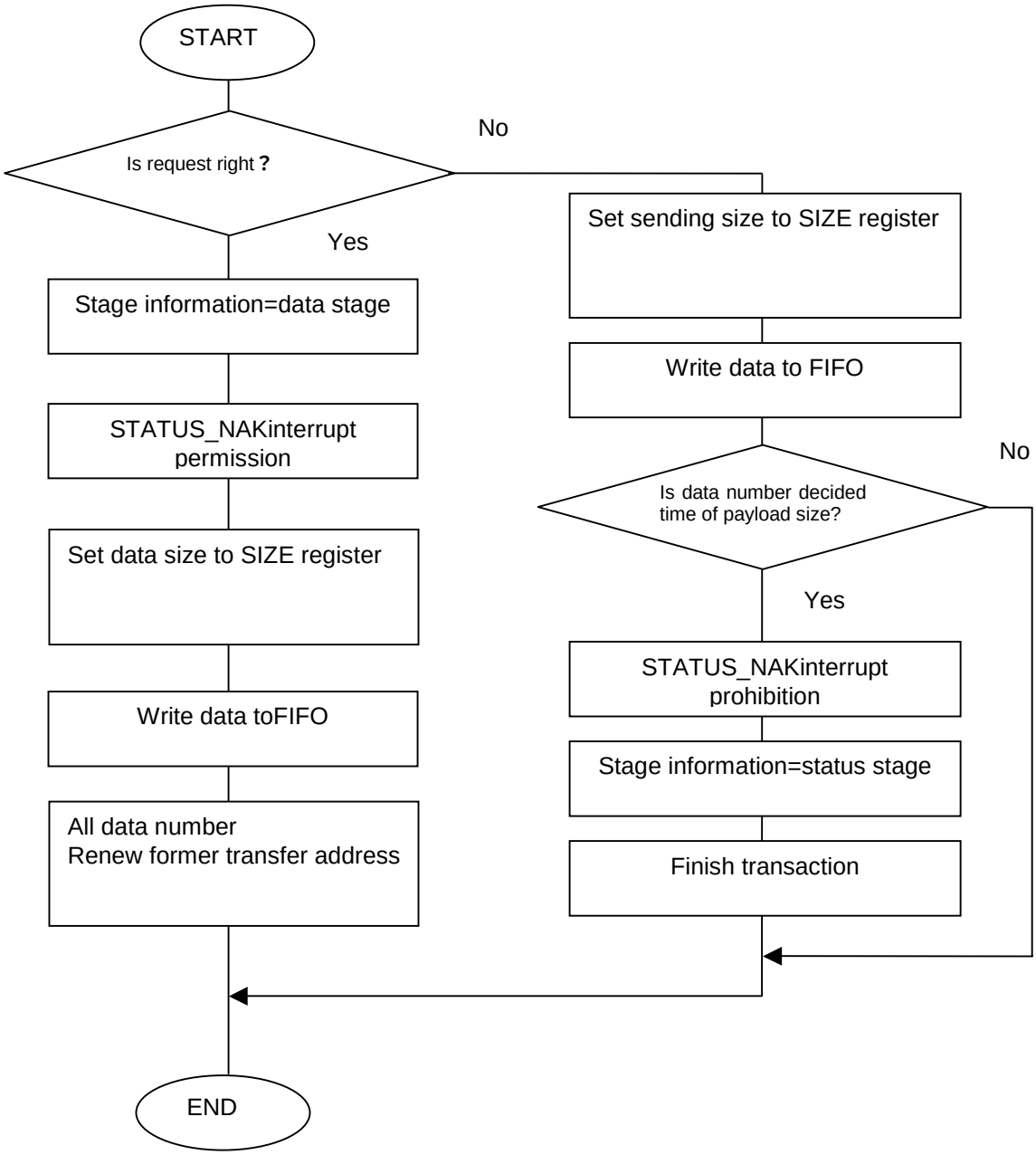
(c-10) Get Descriptorrequest transaction



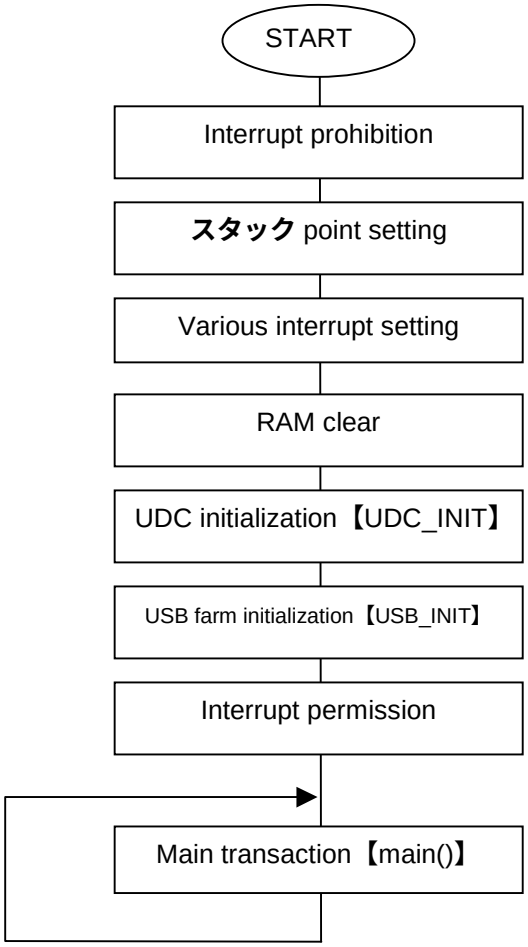
(c-11) Data • read transaction to FIFO by EP0



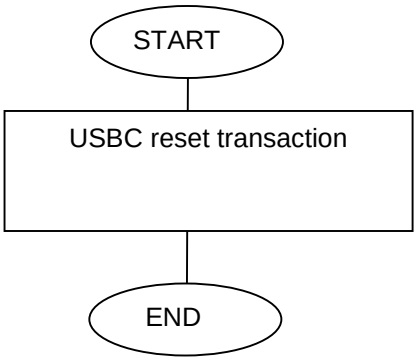
(c-12) Data • write transaction to FIFO by EP0



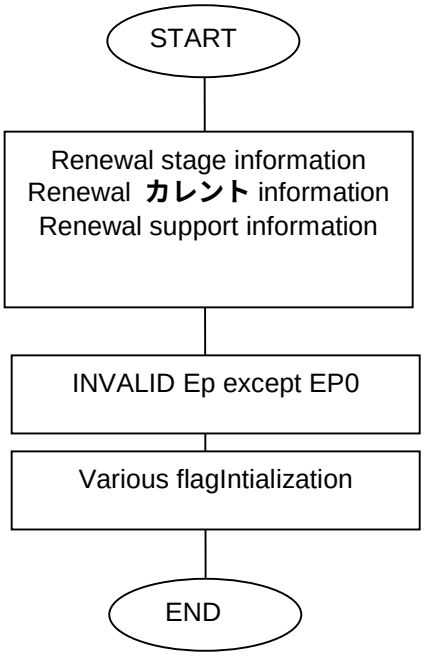
(c-13) Beginning setting transaction of microcontoroller



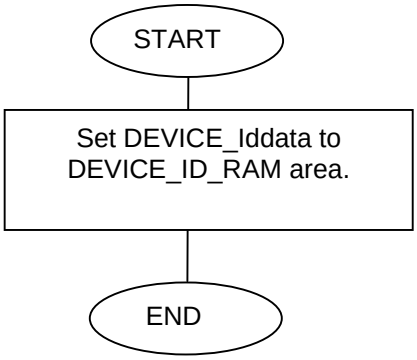
(c-14) Begining setting transaction of USBC



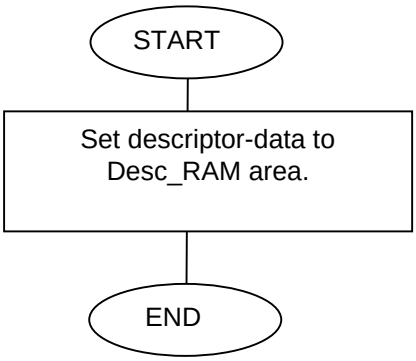
(c-15) Beginning transaction of USB farm chaging number



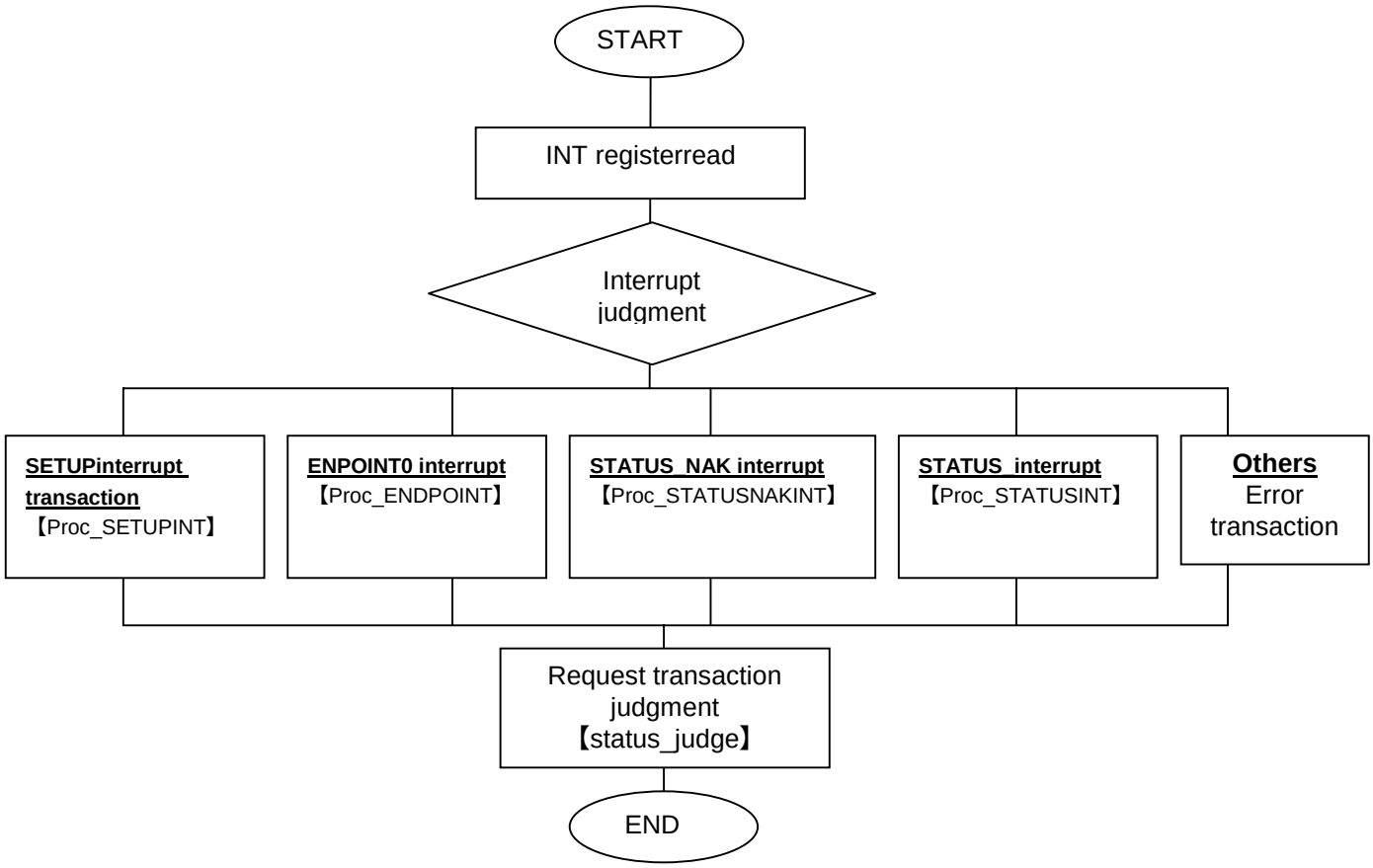
(c-16)Set DEVICE_Iddata to DEVICE_ID of UDC.



(c-17)Descriptor dataset transaction

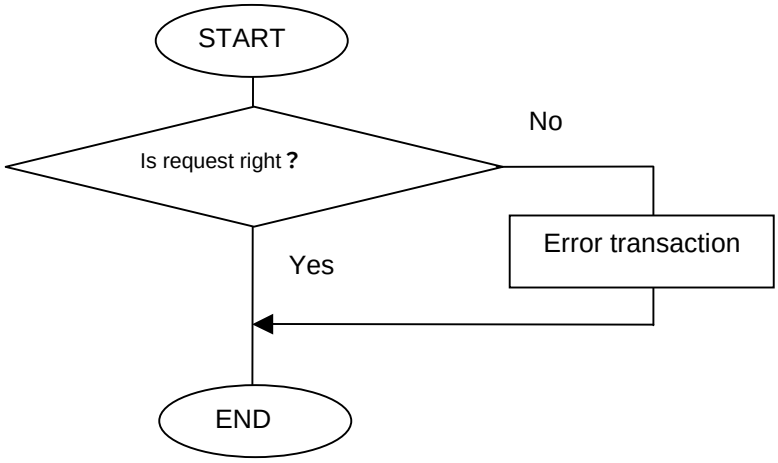


(c-18) USB interrupt transaction

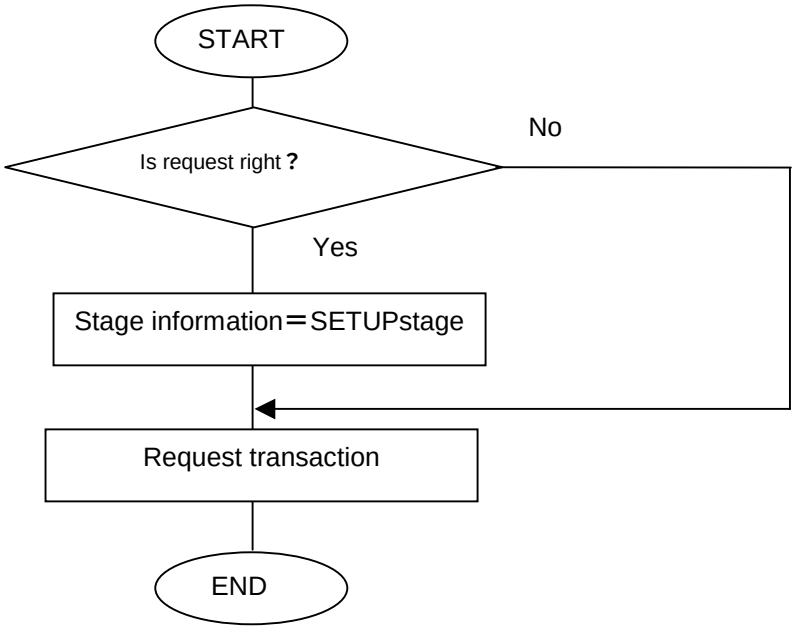


(c-19) dummy function for not using **マスカブル** interrupt.
※Transaction perform nothing, therefore outline-flow is skip.

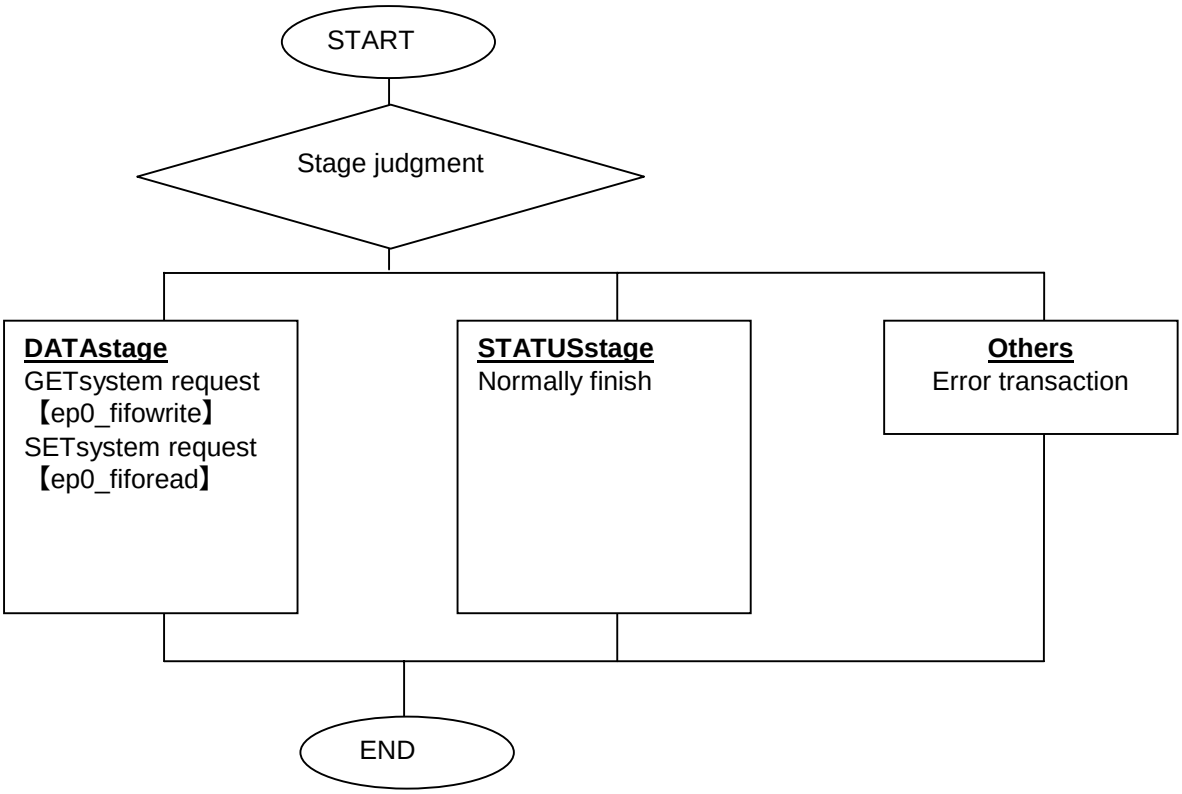
(c-20) request judgment transaction. If transaction result is error, it put put STALL command.



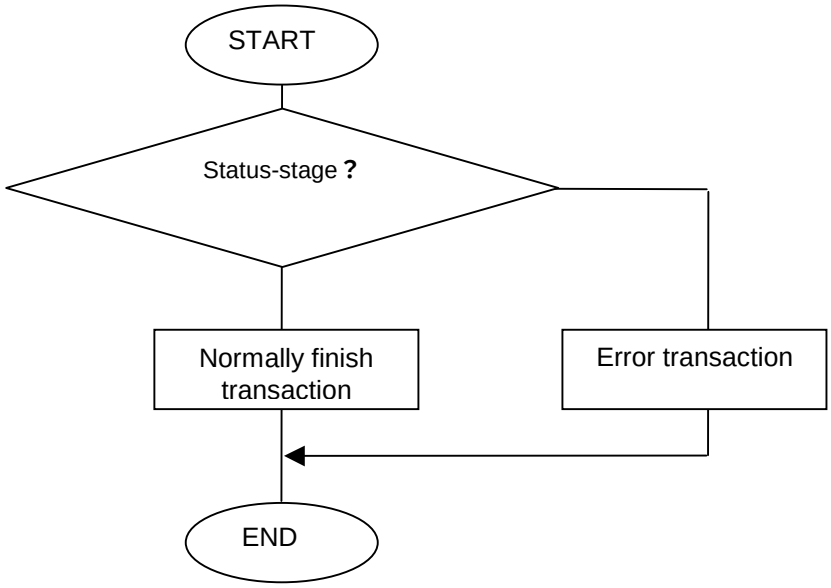
(c-21) SETUPstage transaction



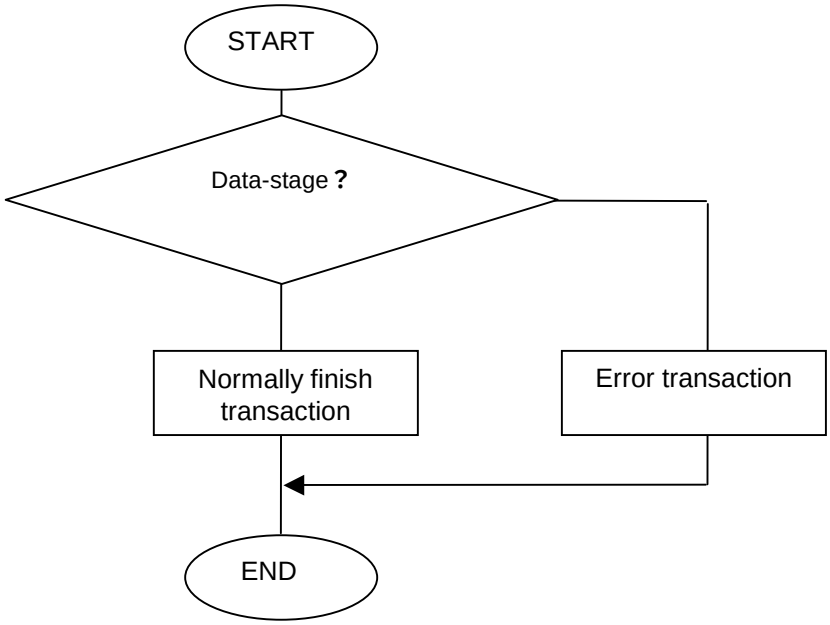
(c-22) Perform Endpoint0 transaction in except for SETUPstage.



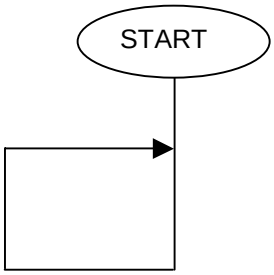
(c-23)Status • stage interrupt transaction



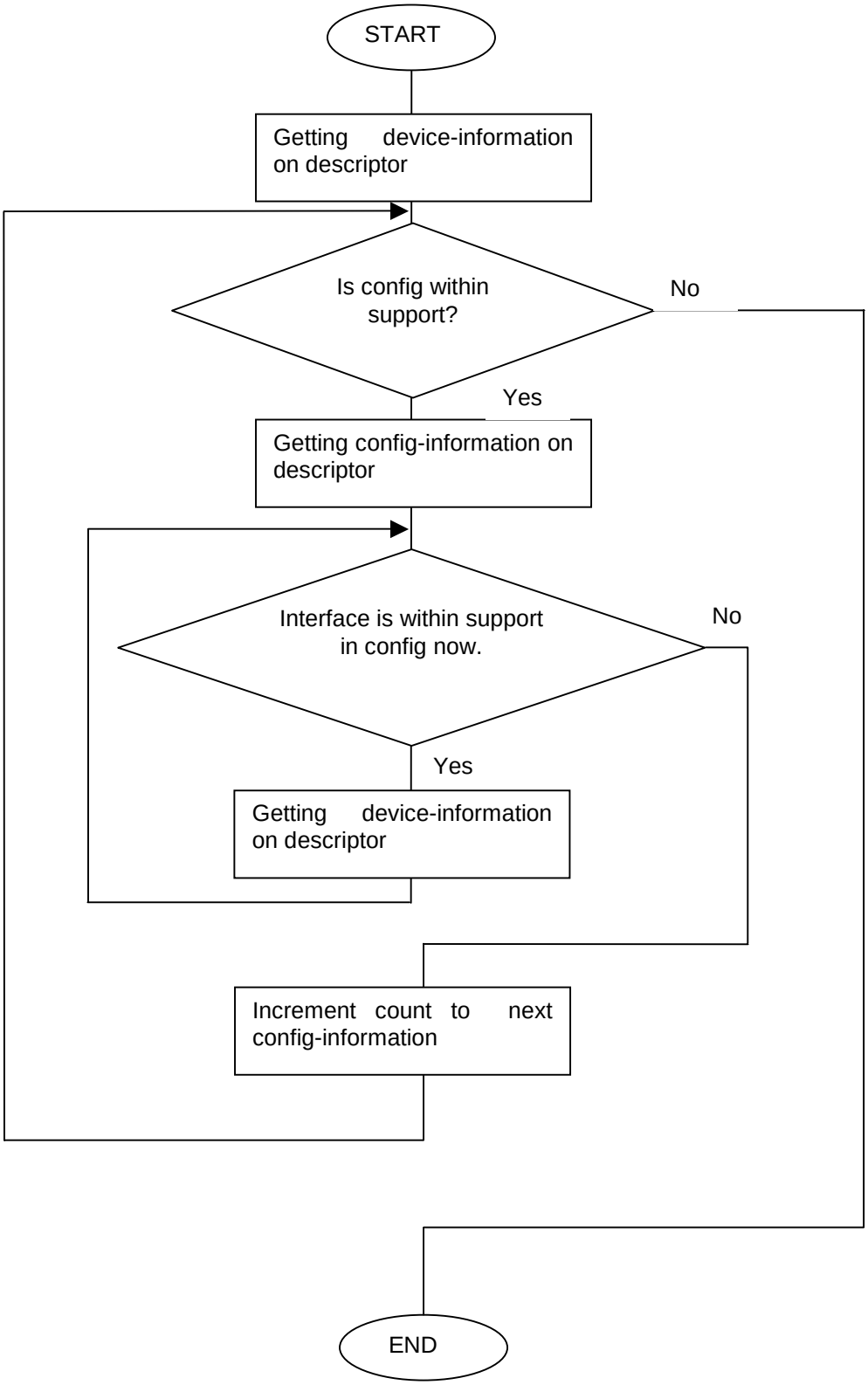
(c-24) STATUS NAKinterrupt transaction



(c-25)This transaction is no transaction by USB transaction perfprm in interrupts.



(c-26) Getting Descriptor information (reration of standarrd request)



3.11 Analog/Digital Converter

The TMP92CH21 incorporates a 10-bit successive approximation-type analog/digital converter (AD converter) with 4-channel analog input.

Figure 3.11.1 is a block diagram of the AD converter. The 4-channel analog input pins (AN0~AN3) are shared with the input-only port Port G so they can be used as an input port.

Note: When IDLE2, IDLE1 or STOP Mode is selected, as to reduce the power, with some timings the system may enter a standby mode even though the internal comparator is still enabled. Therefore be sure to check that AD converter operations are halted before a HALT instruction is executed.

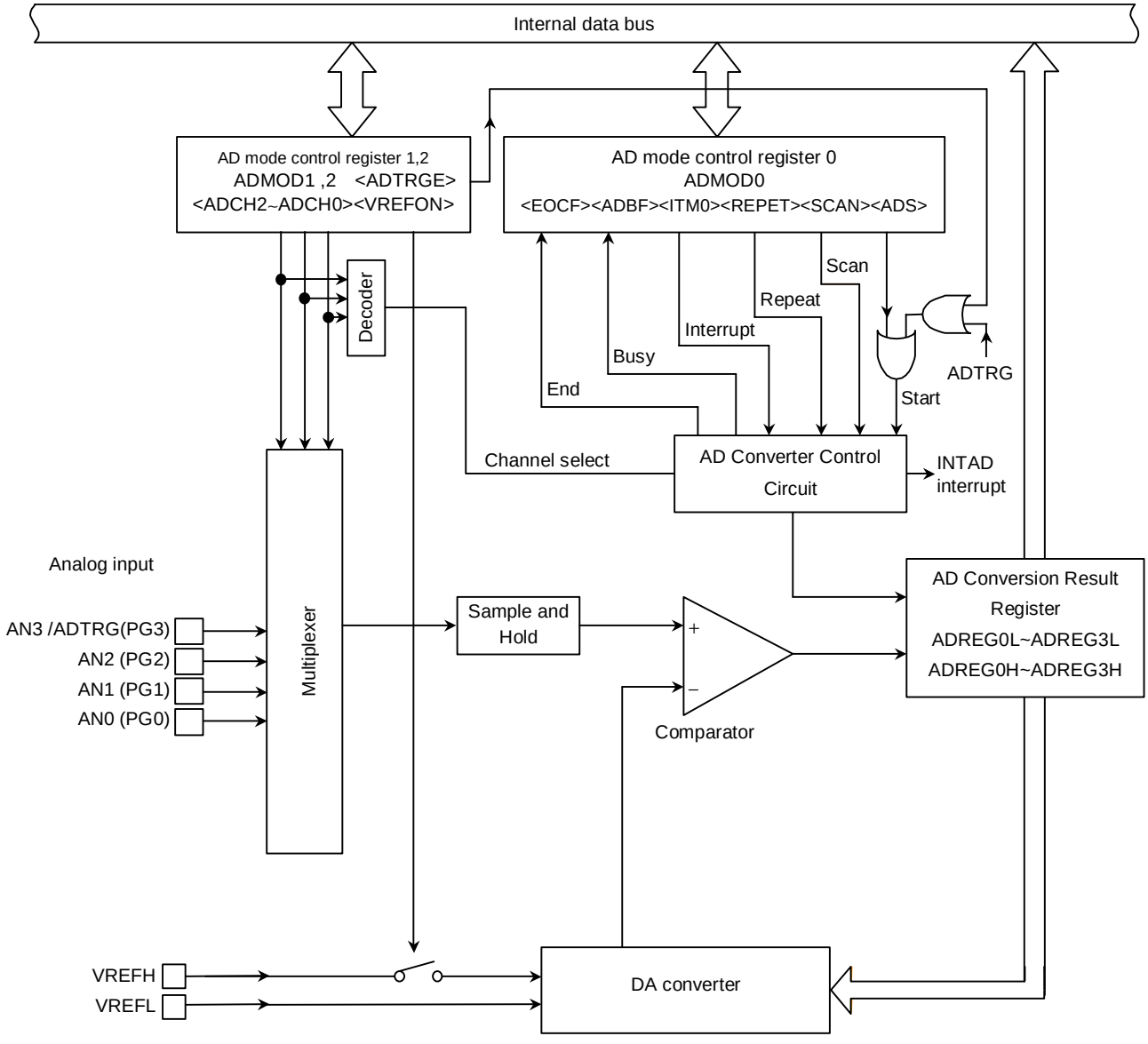


Figure 3.11.1 Block diagram of AD converter

3.11.1 Analog/Digital converter registers

The AD converter is controlled by the three AD Mode Control Registers: ADMOD0, ADMOD1 and ADMOD2. The five AD Conversion Data Result Registers (ADREG0H/L to ADREG3H/L) store the results of AD conversion.

Figure 3.11.2 shows the registers related to the AD converter.

AD Mode Control Register 0

ADMOD0 (12B8H)	7	6	5	4	3	2	1	0
Bit symbol	EOCF	ADBF	–	–	ITM0	REPEAT	SCAN	ADS
Read/Write	R				R/W			
After Reset	0	0	0	0	0	0	0	0
Function	AD Conversion End flag 0: conversion in progress 1: conversion complete	A D Conversion Busy flag 0: conversion stopped 1: Conversion in progress	Note: Always fixed to 0	Note: Always fixed to 0	Interrupt specification in conversion channel fixed repeat mode 0: every conversion 1: every fourth conversion	Repeat mode specification 0: Single Conversion 1: Repeat Conversion Mode	Scan mode specification 0: Conversion Channel Fixed Mode 1: Conversion Channel Scan Mode	AD conversion start 0: Don't care 1: start conversion Always 0 when read

AD conversion start

0	Don't care
1	Start AD conversion

Note: Always read as 0.

AD scan mode setting

0	AD Conversion Channel Fixed Mode
1	AD Conversion Channel Scan Mode

AD repeat mode setting

0	AD Single Conversion Mode
1	AD Repeat Conversion 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	Generates interrupt every fourth conversion.

AD Conversion Busy flag

0	AD conversion stopped
1	AD conversion in progress

AD Conversion End flag

0	Before or during AD conversion
1	AD conversion complete

Figure 3.11.2 AD Converter Related Register

AD Mode Control Register 1

ADMOD1 (12B9H)	7	6	5	4	3	2	1	0
Bit symbol	VREFON	I2AD	–	–	–	–	ADCH1	ADCH0
Read/Write	R/W	R/W				R/W		
After Reset	0	0	0	0	0	0	0	0
Function	VREF application control 0: OFF 1: ON	IDLE2 0: Stop 1: Operate	Note: Always fixed to 0	Note: Always fixed to 0	Note: Always fixed to 0	Note: Always fixed to 0	Analog input channel selection	

Analog input channel selection

<ADCH1, ADCH0> \ <SCAN>	0 (channel fixed)	1 (channel scanned)
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011(Note)	AN3	AN0 → AN1 → AN2 → AN3

IDLE2 control

0	Stopped
1	In operation

Control of application of reference voltage to AD converter

0	OFF
1	ON

Before starting conversion (before writing 1 to ADMOD0 <ADS>), set the <VREFON> bit to 1.

AD Mode Control Register 2

ADMOD2 (12BAH)	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	–	–	–	ADTRGE
Read/Write								R/W
After Reset	0	0	0	0	0	0	0	0
Function	Note: Always fixed To 0	Note: Always fixed To 0	Note: Always fixed To 0	Note: Always fixed to 0	Note: Always fixed to 0	Note: Always fixed to 0	Note: Always fixed to 0	AD external trigger start control 0: disable 1: enable

AD conversion start control by external trigger ($\overline{\text{ADTRG}}$ input)

0	Disabled
1	Enabled

Note: As pin AN3 also function as the $\overline{\text{ADTRG}}$ input pin, do not set < ADCH2:0 > = “011, 100” when using $\overline{\text{ADTRG}}$ with < ADTRGE > set to “1”.

Figure 3.11.3 AD Converter Related Register

AD Conversion Result Register 0 Low

ADREG0L (12A0H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR01	ADR00						ADR0RF
Read/Write	R							R
After Reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD Conversion Data Storage flag 1: Conversion result 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	Undefined							
Function	Stores upper eight bits AD conversion result.							

AD Conversion Result Register 1 Low

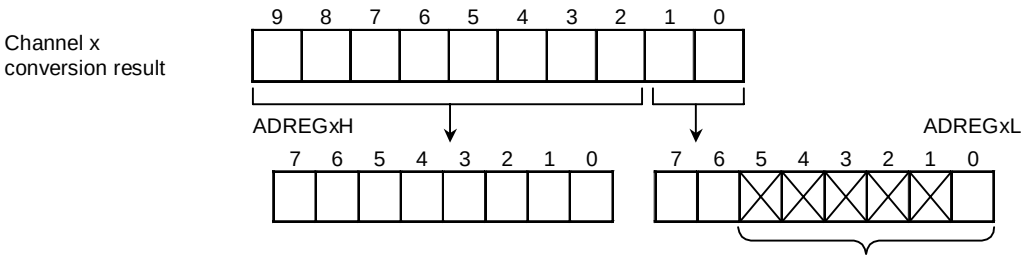
ADREG1L (12A2H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10						ADR1RF
Read/Write	R							R
After Reset	Undefined							0
Function	stores lower 2 bits of AD conversion result							AD Conversion Result flag 1: Conversion result 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	Undefined							
Function	Stores upper eight bits of AD conversion result.							



- Bits 5-1 are always read as 1.
- Bit 0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.4 AD Converter Related Registers

AD Conversion Result Register 2 Low

ADREG2L (12A4H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR21	ADR20						ADR2RF
Read/Write	R							R
After Reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result.							A/D conversion data storage flag 1: Conversion result stored

AD Conversion Result Register 2 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	Undefined							
Function	Stores upper eight bits of AD conversion result.							

AD Conversion Result Register 3 Low

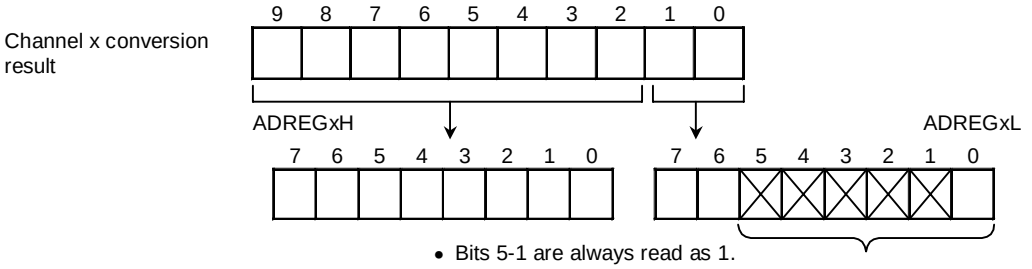
ADREG3L (12A6H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR31	ADR30						ADR3RF
Read/Write	R							R
After Reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result.							AD Conversion Data Storage flag 1: conversion result 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	Undefined							
Function	Stores upper eight bits of AD conversion result.							



- Bits 5-1 are always read as 1.
- Bit 0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.5 AD Converter Related Registers

3.11.2 Description of operation

(1) Analog reference voltage

A high-level analog reference voltage is applied to the VREFH pin; a low-level analog reference voltage is applied to the VREFL pin. To perform AD conversion, the reference voltage, the difference between VREFH and VREFL, is divided by 1024 using string resistance. The result of the division is then compared with the analog input voltage.
To turn off the switch between VREFH and VREFL, write a 0 to ADMOD1<VREFON> in AD Mode Control Register 1. To start AD conversion in the OFF state, first write a 1 to ADMOD1<VREFON>, wait 3 μs until the internal reference voltage stabilizes (this is not related to fc), then set ADMOD0<ADS> to 1.

(2) Analog input channel selection

The analog input channel selection varies depends on the operation mode of the AD converter.

- In Analog Input Channel Fixed Mode (ADMOD0<SCAN> = 0)
  Setting ADMOD1<ADCH1~ADCH0> selects one of the input pins AN0~AN3 as the input channel.
- In Analog Input Channel Scan Mode (ADMOD0<SCAN> = 1)
  Setting ADMOD1<ADCH1~ADCH0> selects one of the twelve scan modes.

Table 3.11.1 illustrates analog input channel selection in each operation mode.

On a Reset, ADMOD0<SCAN> is set to 0 and ADMOD1<ADCH2~ADCH0> is initialized to 000. Thus pin AN0 is selected as the fixed input channel. Pins not used as analog input channels can be used as standard input port pins.

Table 3.11.1 Analog input channel selection

<ADCH2~0>	Channel fixed <SCAN> = "0"	Channel scan <SCAN> = "1"
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011	AN3	AN0 → AN1 → AN2 → AN3

(3) Starting AD Conversion

To start AD conversion, write a 1 to ADMOD0<ADS> in AD Mode Control Register "0" or ADMOD2<ADTRGE> in A/D Mode Control Register 2, and input falling edge on $\overline{\text{ADTRG}}$ pin. When AD conversion starts, the AD Conversion Busy flag ADMOD0<ADBF> will be set to 1, indicating that AD conversion is in progress.
During A/D conversion, a falling edge input on the $\overline{\text{ADTRG}}$ pin will be ignored.

(4) AD conversion modes and the AD Conversion End interrupt

The four AD conversion modes are:

- Channel Fixed Single Conversion Mode
- Channel Scan Single Conversion Mode
- Channel Fixed Repeat Conversion Mode
- Channel Scan Repeat Conversion Mode

The ADMOD0<REPET> and ADMOD0<SCAN> settings in AD Mode Control Register 0 determine the AD mode setting.

Completion of AD conversion triggers an INTAD AD Conversion End interrupt request. Also, ADMOD0<EOCF> will be set to 1 to indicate that AD conversion has been completed.

① Channel Fixed Single Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 00 selects Conversion Channel Fixed Single Conversion Mode.
In this mode data on one specified channel is converted once only. When the conversion has been completed, the ADMOD0<EOCF> flag is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt request is generated.

② Channel Scan Single Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 01 selects Conversion Channel Scan Single Conversion Mode.
In this mode data on the specified scan channels is converted once only. When scan conversion has been completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt request is generated.

③ Channel Fixed Repeat Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 10 selects Conversion Channel Fixed Repeat Conversion Mode.

In this mode data on one specified channel is converted repeatedly. When conversion has been completed, ADMOD0<EOCF> is set to 1 and ADMOD0<ADBF> is not cleared to 0 but held at 1. INTAD interrupt request generation timing is determined by the setting of ADMOD0<ITM0>.

Setting <ITM0> to 0 generates an interrupt request every time an AD conversion is completed.

Setting <ITM0> to 1 generates an interrupt request on completion of every fourth conversion.

④ Channel Scan Repeat Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 11 selects Conversion Channel Scan Repeat Conversion Mode.

In this mode data on the specified scan channels is converted repeatedly. When each scan conversion has been completed, ADMOD0<EOCF> is set to 1 and an INTAD interrupt request is generated. ADMOD0<ADBF> is not cleared to 0 but held at 1.

To stop conversion in a repeat conversion mode (i.e. in cases ③ and ④), write a 0 to ADMOD0<REPET>. After the current conversion has been completed, the repeat conversion mode terminates and ADMOD0<ADBF> is cleared to 0.

Switching to a halt state (IDLE2 Mode with ADMOD1<I2AD> cleared to 0, IDLE1 Mode or STOP Mode) immediately stops operation of the AD converter even when AD conversion is still in progress. In repeat conversion modes (i.e. in cases ③ and ④), when the halt is released, conversion restarts from the beginning. In single conversion modes (i.e. in cases ① and ②), conversion does not restart when the halt is released (the converter remains stopped).

Table 3.11.2 shows the relationship between the AD conversion modes and interrupt requests.

Table 3.11.2 Relationship Between AD Conversion Modes and Interrupt Requests

Mode	Interrupt Request Generation	ADMOD0		
		<ITM0>	<REPEAT>	<SCAN>
Channel Fixed Single Conversion Mode	After completion of conversion	X	0	0
Channel Scan Single Conversion Mode	After completion of scan conversion	X	0	1
Channel Fixed Repeat Conversion Mode	Every conversion	0	1	0
	Every forth conversion	1		
Channel Scan Repeat Conversion Mode	After completion of every scan conversion	X	1	1

X: Don't care

(5) AD conversion time

132state (6.6 μs @ f_{SYS} = 20 MHz) are required for the AD conversion of one channel.

(6) Storing and reading the results of AD conversion

The AD Conversion Data Upper and Lower Registers (ADREG0H/L to ADREG4H/L) store the results of AD conversion. (ADREG0H/L to ADREG4H/L are read-only registers.) In Channel Fixed Repeat Conversion Mode, the conversion results are stored successively in registers ADREG0H/L to ADREG3H/L. In other modes the AN0, AN1, AN2, AN3, AN4 conversion results are stored in ADREG0H/L, ADREG1H/L, ADREG2H/L and ADREG3H/L respectively.

Table 3.11.3 shows the correspondence between the analog input channels and the registers which are used to hold the results of AD conversion.

Table 3.11.3 Correspondence Between Analog Input Channels and AD Conversion Result Registers

Analog input channel (Port G)	AD Conversion Result Register	
	Conversion modes other than at right	Channel fixed repeat conversion mode (every 4 th conversion)
AN0	ADREG0H/L	ADREG0H/L → ADREG1H/L → ADREG2H/L → ADREG3H/L → ADREG0H/L
AN1	ADREG1H/L	
AN2	ADREG2H/L	
AN3	ADREG3H/L	

<ADRxRF>, bit 0 of the AD conversion data lower register, is used as the AD conversion data storage flag. The storage flag indicates whether the AD conversion result register has been read or not. When a conversion result is stored in the AD conversion result register, the flag is set to 1. When either of the AD conversion result registers (ADREGxH or ADREGxL) is read, the flag is cleared to 0.

Reading the AD conversion result also clears the AD Conversion End flag ADMOD0<EOCF> to 0.

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	
INTE0AD	←	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-bit 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.

		7	6	5	4	3	2	1	0	
INTE0AD	←	1	0	0	0	-	-	-	-	Disable INTAD.
ADMOD1	←	1	1	0	0	0	0	1	0	Set pins AN0~AN2 to be the analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Start conversion in Channel Scan Repeat Conversion Mode.

Note: X = Don't care; "–" = No change

3.12 Watchdog Timer (Runaway Detection Timer)

The TMP92CH21 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.

3.12.1 Configuration

Figure 3.12.1(1) is a block diagram of the watchdog timer (WDT).

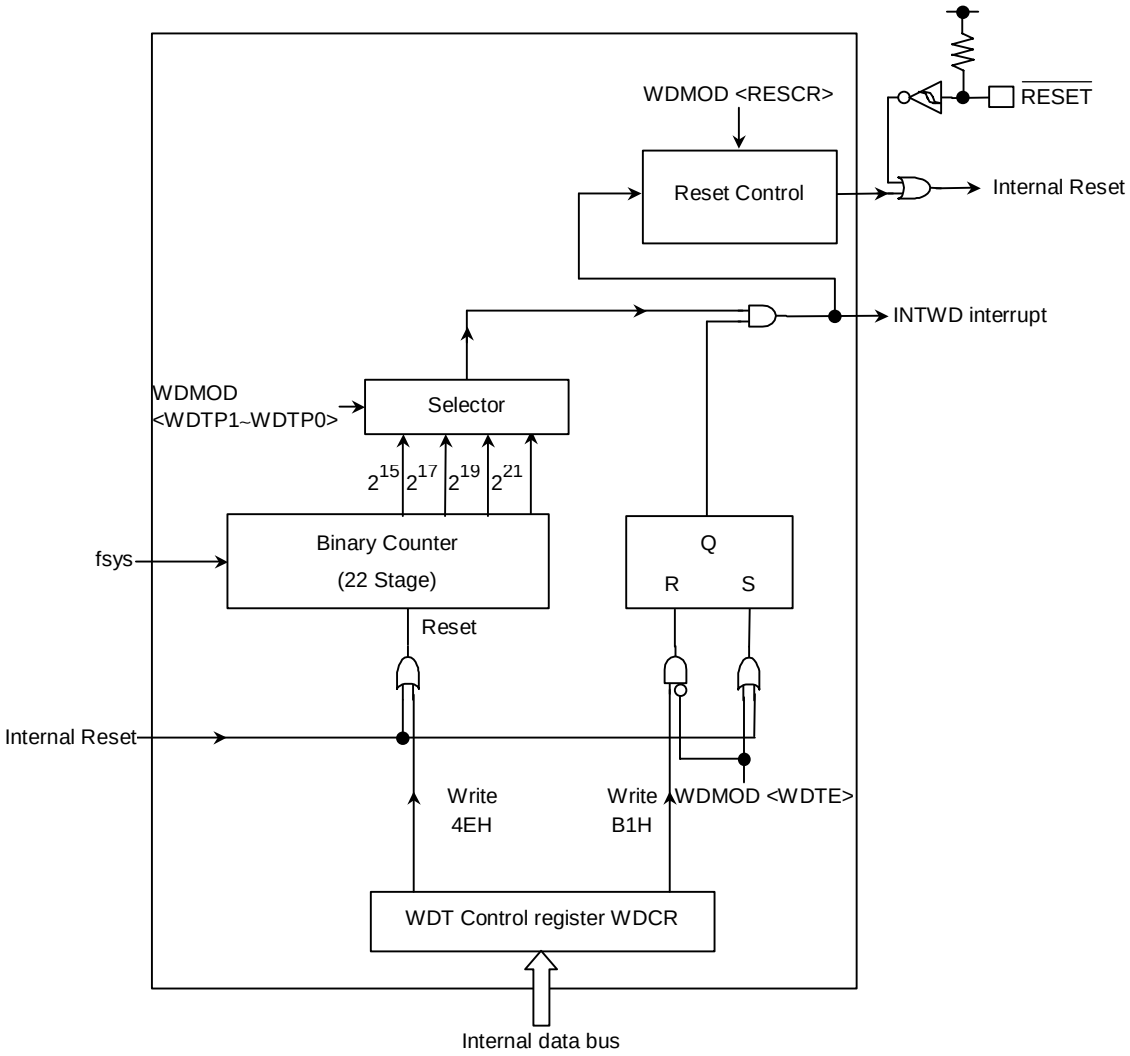


Figure 3.12.1. (1) Block Diagram of Watchdog Timer

The watchdog timer consists of a 22-stage binary counter which uses the clock **ϕ**(2/sys) as the input clock. The binary counter can output 2^{16}/fsys, 2^{18}/fsys, 2^{20}/fsys and 2^{22}/fsys. Selecting one of the outputs using WDMOD<WDTP1,WDTP0> generates a watchdog timer interrupt when an overflow occurs.

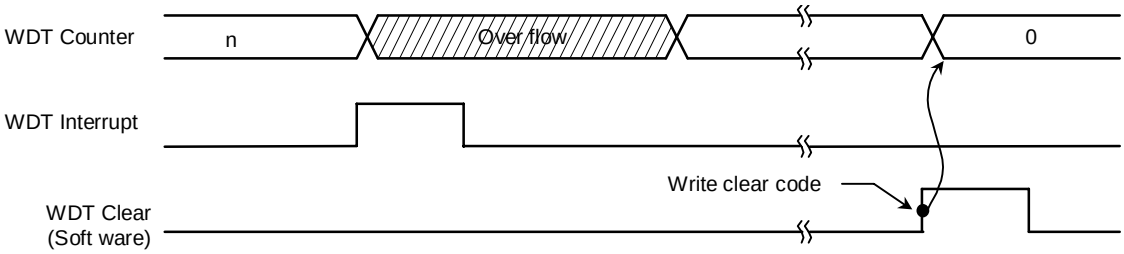


Figure 3.12.1. (2) Normal Mode

The runaway detection result can also be connected to the Reset pin internally.
In this case, the reset time will be between 44 and 58 system clocks (2.2~2.9 µs @ f_{sys} = 20 MHz) as shown in figure 3.12.1(3).

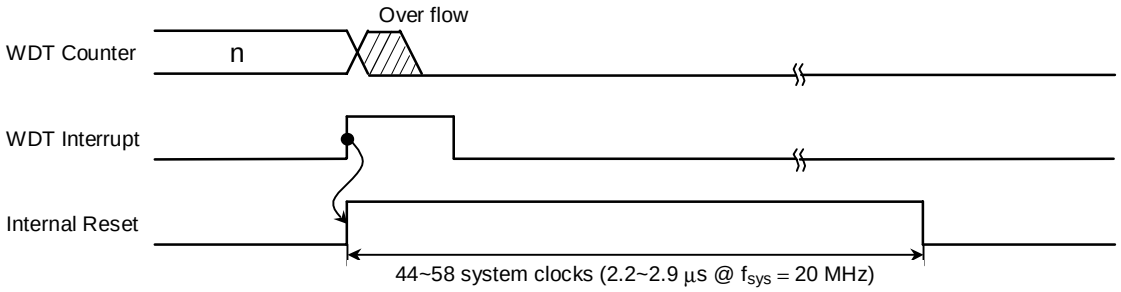


Figure 3.12.1. (3) Reset Mode

3.12.2 Control registers

The watchdog timer WDT is controlled by tow control registers WDMOD and WDCR .

(1) Watchdog Timer Mode Register (WDMOD)

① Setting the detection time for the watchdog timer in <WDTP1,WDTP0>

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,WDTP0> = 00.
The detection times for WDT is 2^{16}/fsys [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 set 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 Register (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.

  ```
  WDMOD  ← 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).
  ```

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	0
Function	WDT control 1: enable	Select detecting time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$			Always fixed to "0".	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	$2^{16}/f_{sys}$ (approximately 3.28ms @ fsys = 20MHz)
01	$2^{18}/f_{sys}$ (approximately 13.1ms @ fsys = 20MHz)
10	$2^{20}/f_{sys}$ (approximately 52.4ms @ fsys = 20MHz)
11	$2^{22}/f_{sys}$ (approximately 210ms @ fsys = 20MHz)

Watchdog timer enable/disable control

0	Disabled
1	Enabled

Figure 3.12.2. (1) Watchdog Timer Mode Register

WDCR (1301H)

	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 3.12.2. (2) Watchdog Timer Control Register

3.12.3 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1,WDTP0> has elapsed. The watchdog timer must be zero-cleared in software before an INTWD interrupt will be generated. If the CPU malfunctions (i.e. 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-mulfunction program.

The watchdog timer begins operating immediately on release of the watchdog timer reset.

The watchdog timer is reset and halted in IDLE1 or STOP Modes. The watchdog counter continues counting during bus release (When /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.

Example: ① Clear the binary counter.

```
WDCR    ← 0 1 0 0 1 1 1 0    Write the clear code (4EH).
```

② Set the watchdog timer detection time to $2^{18}/ f_C$.

```
WDMOD   ← 1 0 1 - - - - -
```

③ Disable the watchdog timer.

```
WDMOD   ← 0 - - X - - - -    Clear <WDTE> bit to 0.
WDCR    ← 1 0 1 1 0 0 0 1    Write the disable code (B1H).
```

3.13 Real time clock (RTC)

3.13.1 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
7) Divided power supply.

3.13.2 Block diagram

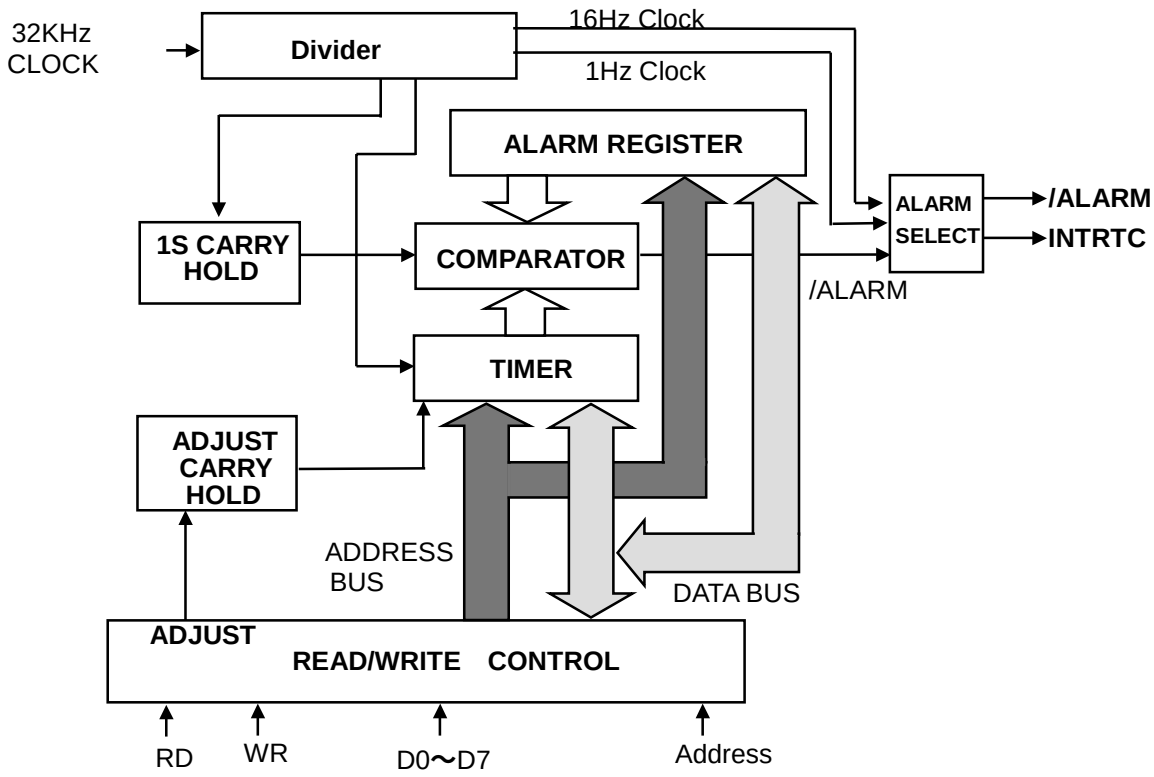


Figure 3.13(1) RTC block diagram

(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 which is divisible with 4 as a leap year, please cope with the system side if this function is problem.

3.13.3 Control registers

Table 3.13(1) PAGE 0 (Timer function) registers

Symbol	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	Bit0	Function	Read/Writ
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	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	INT ENA			ADJUST	ENATMR	ENAALM		PAGE	PAGE register	W,R/W
RESTR	1328h	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	0	0	0	0	Reset register	Write only

(Note) As for SECR, MINR, HOURR, DAYR, MONTHR, YEER of PAGE0, current state is read when read it.

Table 3.13(2) PAGE 1(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 for Alarm	R/W
HOURR	1322h			20 hours	10 hours	8 hours	4 hours	2 hours	1 hours	Hour column for Alarm	R/W
DAYR	1323h						W2	W1	W0	Day of the week column for Alarm	R/W
DATER	1324h			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column for Alarm	R/W
MONTHR	1325h								24/12	24-hour clock mode	R/W
YEARR	1326h							LEAP 1	LEAP 0	Leap-year mode	R/W
PAGER	1327h	**INT ENA**			ADJUST	ENATMR	ENAALM		PAGE	PAGE register	W,R/W
RESTR	1328h	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	0	0	0	0	Reset register	Write only

3.13.4 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	10sec. Column	8 sec. column	4 sec. column	2sec. column	1sec. 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	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.

(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	20min, column	10min, column	8 min. column	4 min. column	2 min, column	1min, 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.

(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 o'clock

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/$\overline{\text{AM}}$	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

(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

(5) Day column register (for 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

(6) Month column register (for PAGE0 only)

MONTHR (1325H)

	7	6	5	4	3	2	1	0
bit Symbol				MO4	MO3	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

(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 year
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

(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	Present is next year of a leap year
1	0	Present is two years after a leap year
1	1	Present is three years after leap year

(10) Setting PAGE register(for PAGE0/1)

PAGER (1327H)

	7	6	5	4	3	2	1	0
bit Symbol	INTENA			ADJUST	ENA TMR	ENAALM		PAGE
Read/Write	R/W			W	R/W			R/W
After reset	0			—	Undefined			Undefined
Function	1:INT enable (note)	"0" is read.		1:ADJUST	TIMER 1:ENABLE 0:DISABLE	ALARM 1:ENABLE 0:DISABLE		PAGE select

Prohibit Read Modify Write

(Note) When INTRTC is used, set following,

ld (pager), 0ch
ld (pager), 8ch

0	Select Page0
1	Select Page1

0	—
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".

(11) Setting reset register (for PAGE0/1)

RESTR (1328H)

	7	6	5	4	3	2	1	0
bit Symbol	DIS1Hz	DIS16Hz	RSTTMR	RSTALM	RE3	RE2	RE1	RE0
Read/Write	W							
After reset	Undefined							
Function	0:1HZ	0:16HZ	1:TIMER RESET	1:ALARM RESET	Please fix to "0"			

Prohibit Read Modify Write

0	—
1	Reset alarm register

0	—
1	Reset timer register

0	Enable 16Hz clock (/ALARM output , INTRTC)
1	Disable 16Hz clock (/ALARM output , INTRTC)

0	Enable 1Hz clock (/ALARM output , INTRTC)
1	Disable 1Hz clock (/ALARM output , INTRTC)

3.13.5 Operational description

(1) Reading timer data

① There is the case, which reads wrong data when carry of the inside counter happens during the operation which timer data reads. Therefore please read two times with the following way for reading correct data.

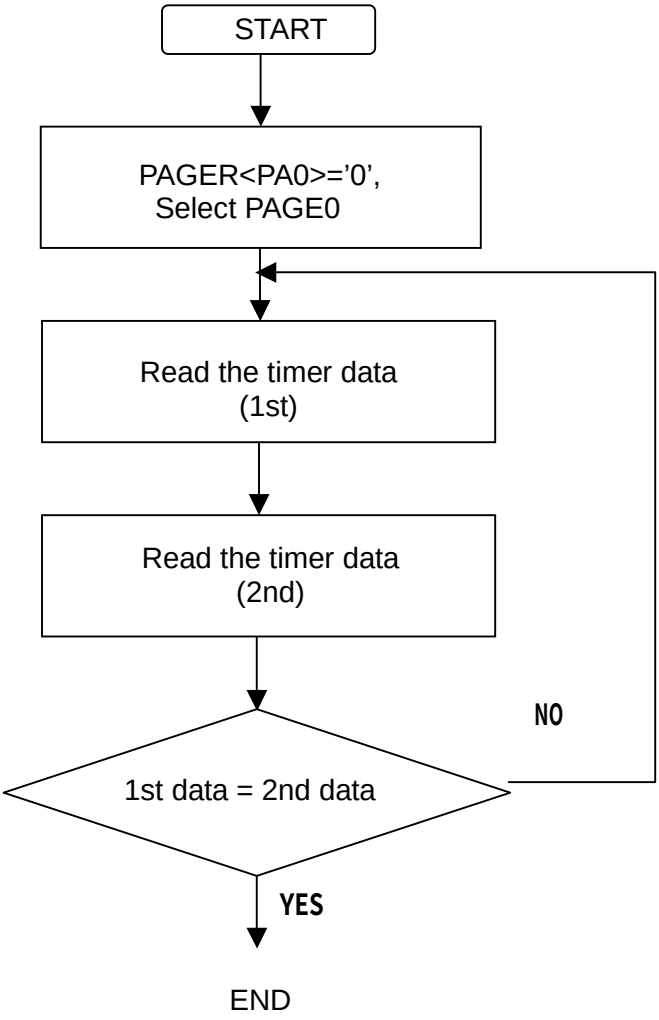
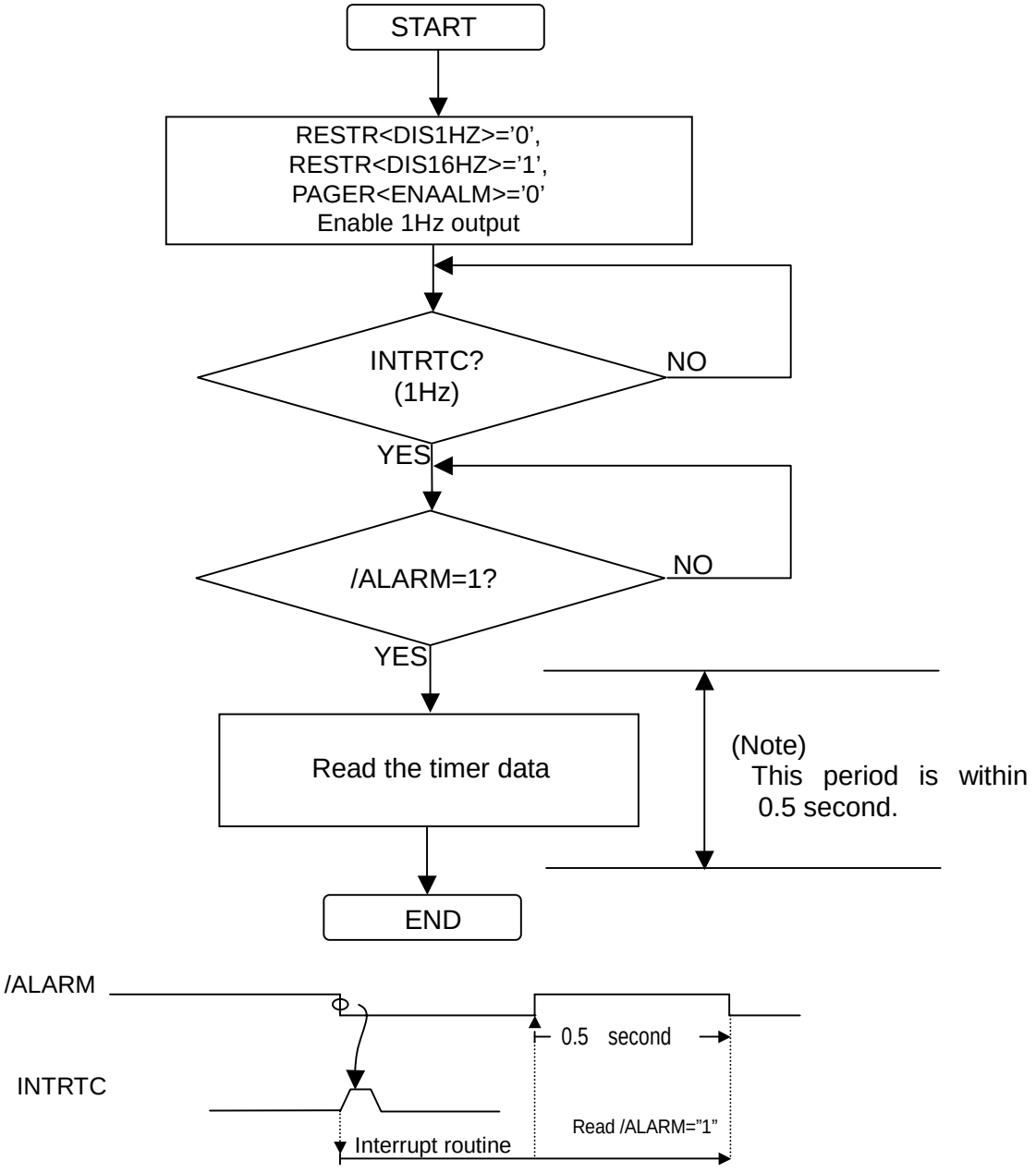


Figure 3.13(2) Flowchart of timer data read

② Readout of timer data that used /ALARM output

Timer data can be read with rising edge of /ALARM output by detecting /ALARM='1' with interrupt routine of INTRTC of 1 Hz



The reason why read a timer of RTC after reading PORT in interrupt routine of /ALARM=1 is that carry of RTC timer occurs with rising edge of pulse period of 1 Hz. By reading timer during 0.5second after carry happening, right data (a timer value) can be read.

Figuire3.11(3) Readout of the timer table used /ALARM output

(2) 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 32,768KHz. Carry of a timer is not done for one second when reset this divider. So write in data at this interval.

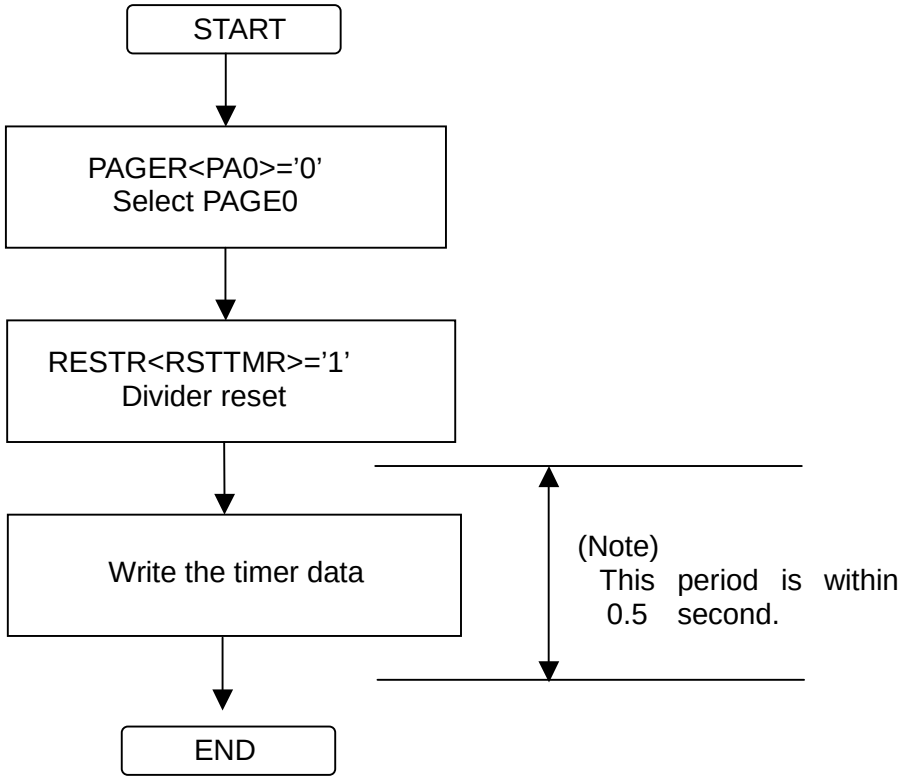


Figure 3.13(4) Flowchart of data write

② Disabling the timer

Carry of a timer is prohibited when write in '0' at PAGER<ENATMR> and can prevent malfunction by CLOCK HOLD circuit. During a timer prohibited, CLOCK HOLD circuits 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.

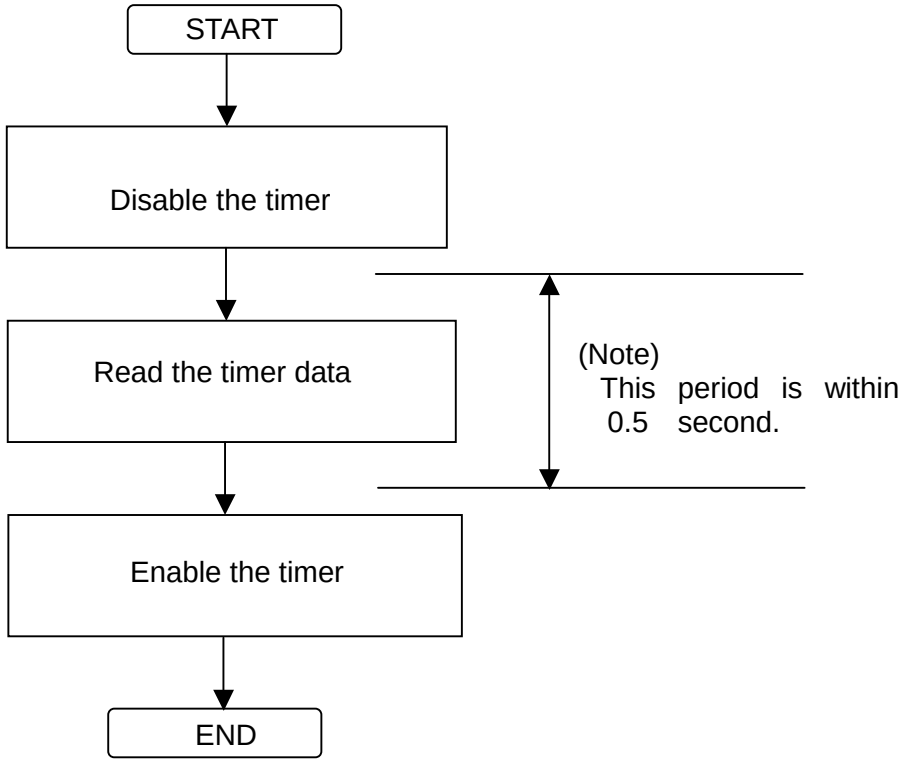


Figure 3.13(5) Flowchart of timer disable

3.13.6 Explanation of the alarm function

Can use alarm function by setting of register of PAGE1 and output either of three signal to /ALARM pin as follows.

(1) In accordance of alarm register and the timer, output '0'.

(2) Output clock of 1Hz.

(3) Output clock of 16Hz.

Resetting does not initialize RTC. So the case of using INTRTC, clear interrupt request flag INTERCKEY <IRC> after reset.

(1) In accordance of alarm register and a timer, output '0'.

When value of a timer of PAGE0 accorded with alarm register of PAGE1 with a state of PAGER<ENAALM>='1', output '0' to /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 timer and occur INTRTC interrupt if PAGER<ENAALM> is '1'.

Setting alarm min., alarm hour, alarm day and alarm the day week is done by writing in data at each register of PAGE1.

When all setting contents accorded, RTC generates INTRTC interrupt, if PAGER<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 are necessary.

The following is an example program for outputting alarm from /ALARM-pin at noon

(PM12: 00) every day.

```
        LD      (PAGER), 09H    ; Alarm disable, setting PAGE1
        LD      (RESTR), D0H    ; Alarm initialize
        LD      (MONTHR), 01H   ; 24-hour clock mode
        LD      (HOURR), 12H    ; setting 12 o'clock
        LD      (MINR), 00H     ; setting 00 min.
                                ; Set up time 31us (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 under SLOW mode or when you use only internal interruption.

(2) When output clock of 1Hz

RTC outputs clock of 1Hz to /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 /ALARM pin by setting up PAGER<ENAALM>='0', RESTR<DIS1HZ>='1', <DIS16HZ>='0'. And RTC generates INTRC interrupt by falling edge of the clock.

3.14 LCD Controller

The TMP92CH21 incorporates two types liquid crystal display driving circuit for controlling LCD Driver LSI. One circuit handles a RAM build-in type LCD driver that can store display data in the LCD driver itself, and the other circuit handles a shift-register type LCD driver that must serially transfer the display data to LCD driver for each display picture.

Software programmable screen size to support monochrome, 4,8,16 level gray scale, color STN panels, and color TFT panels.

This LCDC support for 2 bpp (bit per pixel:4 gray), 3 bpp (8 gray), 4 bpp (16 gray) 8 bpp, (256 colors) and 12 bpp (4096 colors) for dot matrix panels. In passive matrix STN mode, it support for 8 bpp (256 colors) out of a palette of 4096 colors. And active matrix TFT mode, it support for 8 bpp (256 colors) and 12 bpp (4096 colors).

Support for data bus width for 8- or 12-bit TFT panels and 8-, and 4-bit wide LCD panel data bus for STN panels. Hardware panning (soft horizontal and vertical scrolling)

1) Shift-register type LCD driver control mode (SR mode)

This mode is for monochrome STN or color STN panel. Set the mode of operation, start address of source data save memory and LCD size to control register before setting start register.

After set start register LCDC outputs bus release request to CPU and read data from source memory.

After that LCDC transmits data of volume of LCD size to external LCD driver through special LCDC data bus (LD11-LD0). At this time, control signals connected LCD driver output specified waveform synchronizes with data transmission. After finish data reading from RAM for display, LCDC cancels the bus release request and CPU will re-start.

As the DISPLAY RAM, external SRAM and SDRAM burst mode can be used in TMP92CH21. And TMP92CH21 has 16KB SRAM, this internal RAM can available to use for display RAM. Internal RAM access is very fast (32bit bus width, 1-SYSCLK read/write), disturbance time of CPU is able to set minimum time at LCDC DMA.

2) Color display mode for TFT panel

Data transmission process is same to above SR mode. For control to TFT source driver, there are signals; LD11~0; 8 bpp RGB (R:3, G:3, B:2) and 12 bpp RGB (R:4,G:4,B:4), LCP0, LFR, LLP, and LDIV: invert data line. And for control to TFT gate driver, there are signals; LCP1, LBCD and OE: control to detail output timing.

3) RAM built-in type LCD driver control mode (RAM mode)

Data transmission to LCD driver is executed by move instruction of CPU. After setting mode of operation to control register, when moves instruction of CPU is executed LCDC outputs chip select signal to LCD driver connected to the outside from control pin (LCP0 etc.) . Therefore control of data transmission numbers corresponding to LCD size is controlled by instruction of CPU.

3.14.1 Feature of LCDC of each mode

Each feature and operation of pin is as follows.

Table 3.14.1 Feature of LCDC of each mode

LCDD		Shift- register type LCD driver control mode		RAM built-in type LCD driver control mode
		TFT	STN	
Display Color		256 color, 4096 color	Monochrome, 4, 8, 16 level gray 256 color, 4096 color	Depend on LCDD
The number of picture elements can be handled		Row (Common): 64,128,160,200,240,320 Column (Segment): 64,128,160,256,320	**Monochrome/ 4/8/16Gray** Row (Common): 64,128,160,200,240,320,480 Column (Segment): 64,128,160,256,320,480,640 **Color (256,4096)** Row (Common): 64,128,160,200,240,320 Column (Segment): 64,128,160,256,320	Depend on LCDD
		Internal SRAM: 256-128×128 max,4096-128×64 max		
Data bus width (display: RAM,SDRAM)		16,32bit	16,32bit	Depend on CS/WAIT controller (Same to normal memory access)
Data bus width (destination: LCDD)		8,12bit	4,8bit	
Maximum transmission rate (@ f_{SYS} = 20[MHz])		12.5ns/Byte@SDRAM/BURST 12.5nS/Byte@internal RAM, 25ns/1Byte@external SRAM		-
Pan function		Available to use		Depend on LCDD
External Pins	LCD Data Bus LD11 to LD0	Connect to data bus of LCDD 4bit:LD3 to LD0 8bit:LD7 to LD0 12bit:LD11 to LD0 (only use TFT panel)		Not use LD bus Use normal data bus
	Bus state R/W	Not use		Connect to /WR terminal of LCDD
	Address Bus A0	Not use		Connect to D/I terminal of LCDD For distinction of data or instruction
	LCP0	Horizontal shift clock for source driver of TFT panel	Shift clock 0 for column LCDD Connect to CP terminal of column LCDD. LD bus data is latched at falling edge of this signal.	Chip enable signal for column LCDD Connect to /CE terminal of 1'st column LCDD
	LCP1	Vertical shift clock for gate driver of TFT panel	Shift clock 1 for column LCDD Connect to CP terminal of column LCDD. LD bus data is latched at falling edge of this signal.	Not use
	LLP	Data load signal for source driver of TFT panel	Latch pulse output for column and row LCDD. Connect to LP terminal of column and row LCDD. Display data is latched to output buffer at rising edge of this signal.	Chip enable signal for column LCDD Connect to /CE terminal of 2'nd column LCDD
	LGOE2 to 0	Output enable signal for gate driver of TFT panel	Not use	Not use
	LFR	Alternating signal for LCD display control. Connect to FR terminal of LCDD	Alternating signal for LCD display control. Connect to FR terminal of LCDD	Chip enable signal for column LCDD Connect to /CE terminal of 3'rd column LCDD
	LBCD		Refresh rate signal	Chip enable signal for row LCDD Connect to /LE terminal of row LCDD
	LDIV	Connect to LDIV terminal for source driver of TFT panel. This signal shows output data inversion.	Not use	

3.14.2 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	0	0	0	0	0
Function	Display RAM 00: Internal SRAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transmission speed 00: Reserved 01: 2-clock 10: 4-clock 11: 8-clock		Mode setring 0000: Built-in RAM type 0001: SR 1bpp (mono) 0010: SR 2bpp (4gray) 0011: SR 3bpp (8gray) 0100: SR 4bpp (16gray)		0101: STN 8bpp(256) 0110: STN 12bpp(4K) 0111: reserved 1000: TFT 8bpp(256) 1001: TFT 12bpp(4K) other: reserved	

NOTE) SDRAM access is only supported “Burst 1clk Access”

LCDMODE1 Register

LCDMODE1 (0281H)

	7	6	5	4	3	2	1	0
bit Symbol			LLPMODE	LDINV	AUTOINV	INTMODE	LDO1	LDO0
Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
After reset			0	0	0	0	0	0
Function			LLP mode 0: Mode1 1: Mode2	LD bus Inversion 0: normal 1:inversion	Auto LD bus inversion 0: disable 1: enable (valid in TFT mode)	Interrupt Select 0: LP 1: BCD	LD bus width control 00: 4bit width A_type 01: 4bit width B_type 10: 8bit width A_type 11: 8bit width B_type other reserved	

When using TFT mode, LD bus width is fixed below setting.
256 color 8bit width A_type
4096 color 12bit width

NOTE) When set <LDINV>=1, “auto LD bus inversion function” doesn’t work.
If you want to use “auto LD bus inversion function”, it need to set <LDINV>=0.

LCD fFP Register

LCDFFP (0282H)

	7	6	5	4	3	2	1	0
bit Symbol	FP7	FP6	FP5	FP4	FP3	FP2	FP1	FP0
Read/Write	R/W							
After Reset	0							
Function	Setting bit7-0 for f FP							

Divide FRM Register

LCDDVM (0283H)

	7	6	5	4	3	2	1	0
bit Symbol	FMN7	FMN6	FMN5	FMN4	FMN3	FMN2	FMN1	FMN0
Read/Write	R/W							
After Reset	0							
Function	Setting DVM bit7-0							

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	Row setting 0000 : reserve 0101 : 200 0001 : 64 0110 : 240 0010 : 120 0111 : 320 0011 : 128 1000 : 480 0100 : 160 other : reserved				Column setting 0000 : 32 0101 : 320 0001 : 64 0110 : 480 0010 : 128 0111 : 640 0011 : 160 1000 : 768 0100 : 256 1001 : 960 other : reserved			

NOTE1) In case of color mode (STN,TFT), maximum size is 320*320
And it set to internal SRAM for display RAM, maximum size is below.

1 bpp (Monochrome):	640*200
2 bpp (4gray):	320*200
4 bpp (16gray):	256*128
8 bpp (256 color):	128*128
12 bpp (4096 color):	128*64

NOTE2) TMP92CH21 don't support 240-segment size, but if it choose cascade type segment driver, it can use 256-segment size. Then it needs to ready 256 segment display area.

LCD Control-0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol		ALL0	FRMON	EXOR	FP9	MMULCD	FP8	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		Column Data setting 0: normal 1:All display data '0'	FR divide setting 0: disable 1: enable	EXOR setting 0: off 1: on	f FP setting bit9	Built in RAM LCDD setting 0: sequential access 1: random access	f FP setting bit8	LCDC start 0:STOP 1:START

LCD Control-1 Register

LCDCTL1 (0286H)

	7	6	5	4	3	2	1	0
bit Symbol	LCP0P	LCP1P	LBCDP				LBCDW1	LBCDW0
Read/Write	R/W	R/W	R/W				R/W	R/W
After Reset	1	0	1				0	0
Function	LCP0 phase 0: rise 1: fall	LCP1 Phase 0: rise 1: fall	LBCD phase 0:Low Enable 1:Hi Enable				LBCD width control 00 : LCP1_1CLK 01 : LCP1_2CLK 10 : LCP1_3CLK 11 : Reserved	

LCDC Source Clock Counter Register

LCDSCC (0287H)

	7	6	5	4	3	2	1	0
bit Symbol	SCC7	SCC6	SCC5	SCC4	SCC3	SCC2	SCC1	SCC0
Read/Write	R/W							
After Reset	0							
Function	LCDC Source Clock Counter bit7-0							

LCD Clock Counter Register0

LCDCCR0 (0288H)

	7	6	5	4	3	2	1	0
bit Symbol						PCPV2	PCPV1	PCPV0
Read/Write						R/W	R/W	R/W
After Reset						0	0	0
Function						Pre LCP1 CLK: Dummy clock number to valid clock of LCP1		

LCD Clock Counter Register1

LCDCCR1 (0289H)

	7	6	5	4	3	2	1	0
bit Symbol				TLDE4	TLDE3	TLDE2	TLDE1	TLDE0
Read/Write				R/W	R/W	R/W	R/W	R/W
After Reset				0	0	0	0	0
Function	Setting register of LLP enable time							

LCD Clock Counter Register2

LCDCCR2 (028AH)

	7	6	5	4	3	2	1	0
bit Symbol	LLPSU7	LLPSU6	LLPSU5	LLPSU4	LLPSU3	LLPSU2	LLPSU1	LLPSU0
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	Set-up time of LLP							

LCD RED Palette Register

		7	6	5	4	3	2	1	0
LCDRP10 (0291H)	bit Symbol	1R3	1R2	1R1	1R0	0R3	0R2	0R1	0R0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	0	0	1	0	0	0	0	0
	Function	256 color STN mode RED1 level setting				256 color STN mode RED0 level setting			
		7	6	5	4	3	2	1	0
LCDRP32 (0292H)	bit Symbol	3R3	3R2	3R1	3R0	2R3	2R2	2R1	2R0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	0	1	1	0	0	1	0	0
	Function	256 color STN mode RED3 level setting				256 color STN mode RED1 level setting			
		7	6	5	4	3	2	1	0
LCDRP54 (0293H)	bit Symbol	5R3	5R2	5R1	5R0	4R3	4R2	4R1	4R0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	1	0	1	0	1	0	0	0
	Function	256 color STN mode RED5 level setting				256 color STN mode RED4 level setting			
		7	6	5	4	3	2	1	0
LCDRP76 (0294H)	bit Symbol	7R3	7R2	7R1	7R0	6R3	6R2	6R1	6R0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	1	1	1	0	1	1	0	0
	Function	256 color STN mode RED7 level setting				256 color STN mode RED6 level setting			

Note: In case using TFT mode, it's prohibit to change above palette setting

LCD GREEN Palette Register

		7	6	5	4	3	2	1	0
LCDGP10 (0295H)	bit Symbol	1G3	1G2	1G1	1G0	0G3	0G2	0G1	0G0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	0	0	1	0	0	0	0	0
	Function	256 color STN mode GREEN1 level setting				256 color STN mode GREEN0 level setting			
		7	6	5	4	3	2	1	0
LCDGP32 (0296H)	bit Symbol	3G3	3G2	3G1	3G0	2R3	2G2	2G1	2G0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	0	1	1	0	0	1	0	0
	Function	256 color STN mode GREEN3 level setting				256 color STN mode GREEN2 level setting			
		7	6	5	4	3	2	1	0
LCDGP54 (0297H)	bit Symbol	5G3	5G2	5G1	5G0	4G3	4G2	4G1	4G0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	1	0	1	0	1	0	0	0
	Function	256 color STN mode GREEN5 level setting				256 color STN mode GREEN4 level setting			
		7	6	5	4	3	2	1	0
LCDGP76 (0298H)	bit Symbol	7G3	7G2	7G1	7G0	6G3	6G2	6G1	6G0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	1	1	1	0	1	1	0	0
	Function	256 color STN mode GREEN7 level setting				256 color STN mode GREEN6 level setting			

Note: In case using TFT mode, it's prohibit to change above palette setting

LCD BLUE Palette Register

		7	6	5	4	3	2	1	0
LCDBP10 (0299H)	bit Symbol	1R3	1R2	1R1	1R0	0R3	0R2	0R1	0R0
	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	256 color STN mode BLUE1 level setting				256 color STN mode BLUE0 level setting			
		7	6	5	4	3	2	1	0
LCDBP32 (029AH)	bit Symbol	3R3	3R2	3R1	3R0	2R3	2R2	2R1	2R0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After Reset	1	1	0	0	1	0	0	0
	Function	256 color STN mode BLUE3 level setting				256 color STN mode BLUE2 level setting			

Note: In case using TFT mode, it's prohibit to change above palette setting

LCD OE0 control Register

LCDOE00 (02B0H)

	7	6	5	4	3	2	1	0
bit Symbol	OE007	OE006	OE005	OE004	OE003	OE002	OE001	OE000
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	OE0 CONTROL OF GATE DRIVER OF TFT PANEL							

LCDOE01 (02B1H) to LCDOE04 (02B4H)

LCDOE05 (02B5H)

	7	6	5	4	3	2	1	0
bit Symbol	OE057	OE056	OE055	OE054	OE053	OE052	OE051	OE050
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	OE0 CONTROL OF GATE DRIVER OF TFT PANEL							

LCD OE1 control Register

LCDOE10 (02C0H)

	7	6	5	4	3	2	1	0
bit Symbol	OE107	OE106	OE105	OE104	OE103	OE102	OE101	OE100
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	OE1 CONTROL OF GATE DRIVER OF TFT PANEL							

LCDOE11 (02C1H) to LCDOE14 (02C4H)

LCDOE15 (02C5H)

	7	6	5	4	3	2	1	0
bit Symbol	OE157	OE156	OE155	OE154	OE153	OE152	OE151	OE150
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	OE1 CONTROL OF GATE DRIVER OF TFT PANEL							

LCD OE2 control Register

LCDOE20 (02D0H)

	7	6	5	4	3	2	1	0
bit Symbol	OE207	OE206	OE205	OE204	OE203	OE202	OE201	OE200
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	OE2 CONTROL OF GATE DRIVER OF TFT PANEL							

LCDOE21 (02D1H) to LCDOE24 (02D4H)

LCDOE25 (02D5H)

	7	6	5	4	3	2	1	0
bit Symbol	OE257	OE256	OE255	OE254	OE253	OE252	OE251	OE250
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	OE2 CONTROL OF GATE DRIVER OF TFT PANEL							

	Start address register			ROW number setting register		
	H (bit23-16)	M (bit15-8)	L (bit7-1)	H (bit8)	L (bit7-0)	-----
A-area	LSARAH (02A2H) 40H	LSARAM (02A1H) 00H	LSARAL (02A0H) 00H	CMNAH (02A4H) 00H	CMNAL (02A3H) 00H	-----
B-area	LSARBH (02A8H) 40H	LSARBM (02A7H) 00H	LSARBL (02A6H) 00H	CMNBH (02AAH) 00H	CMNBL (02A9H) 00H	-----
C-area	LSARCH (02AEH) 40H	LSARCM (02ADH) 00H	LSARCL (02ACH) 00H	-----	-----	-----

NOTE) all register can RMW

LCDC0L/LCDC0H/LCDC1L/LCDC1H/LCDC2L/LCDC2H/LCDR0L/LCDR0H register

	7	6	5	4	3	2	1	0
bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
Read/Write	Depend on External LCDD specification							
After Reset	Depend on External LCDD specification							
Function	Depend on External LCDD specification							

address	function	Chip enable terminal
3C0000H to 3CFFFFH	Built-in RAM LCDD1	LCP0
3D0000H to 3DFFFFH	Built-in RAM LCDD2	LLP
3E0000H to 3EFFFFH	Built-in RAM LCDD3	LFR
3F0000H to 3FFFFFH	Built-in RAM LCDD4	LBCD

3.14.3 Shift-register type LCD driver control mode (SR mode, STN color)

3.14.3.1 Description of operation

Set the mode of operation, start address of source data save memory, gray-scale level and LCD size to control registers before setting start register.

After set start register LCDC outputs bus release request to CPU and read data from source memory. After finish data read from source data, LCDC cancels the bus release request and CPU will re-start. After that LCDC transmits data of volume of LCD size to external LCD driver through LD bus (Special data bus for only LCDD). At this time, control signals (LCP0 etc.) connected LCD driver output specified waveform synchronizes with data transmission.

SR mode LCDC, during data read from source memory (during DMA operation), CPU is stopped by internal BUSREQ signal.

LCD controller generates some control signals (LFR, LBCD, LLP etc) from base clock: LCDSCC. LCDSCC is base clock for LCD controller, made from system clock: $f_{SYS.}$

TMP92CH21 has special clock generator for LCDC. Detail refresh rate of LCD frame can set from this special generator. This generator made by 8-bit counter and 1/16 speed clock of system clock.

NOTE1) SR mode LCDC, during data read from source memory (during DMA operation), CPU is stopped by internal BUSREQ signal. When using SR mode LCDC, programmer need to care the CPU performance.

NOTE2) TMP92CH21 has 16KB SRAM, this internal RAM can available to use for display RAM. Internal RAM access is very fast (32bit bus width, 1-SYSCLK read/write), disturbance time of CPU is able to set minimum time at LCDC DMA.

This LCDC support monochrome, 2bpp (4gray), 3bpp (8gray), 4bpp (16gray), 8bpp (256 color) and 12bpp (4096 color. Display RAM is supported external SDRAM, SRAM and internal RAM (16KB).

It is automatically set to suitable condition data correction against interference between pixels in panels. It doesn't need special adjustment.

In passive matrix STN mode, it support for 8 bpp (256 colors) out of a palette of 4096 colors. And it support for 4096 color out of a pallet of 4096 colors.

Selectable data output width in 4bit or 8bit. And selectable data output sequence in 2 mode.

We'll explain about setting to control SR type LCDD from next page.

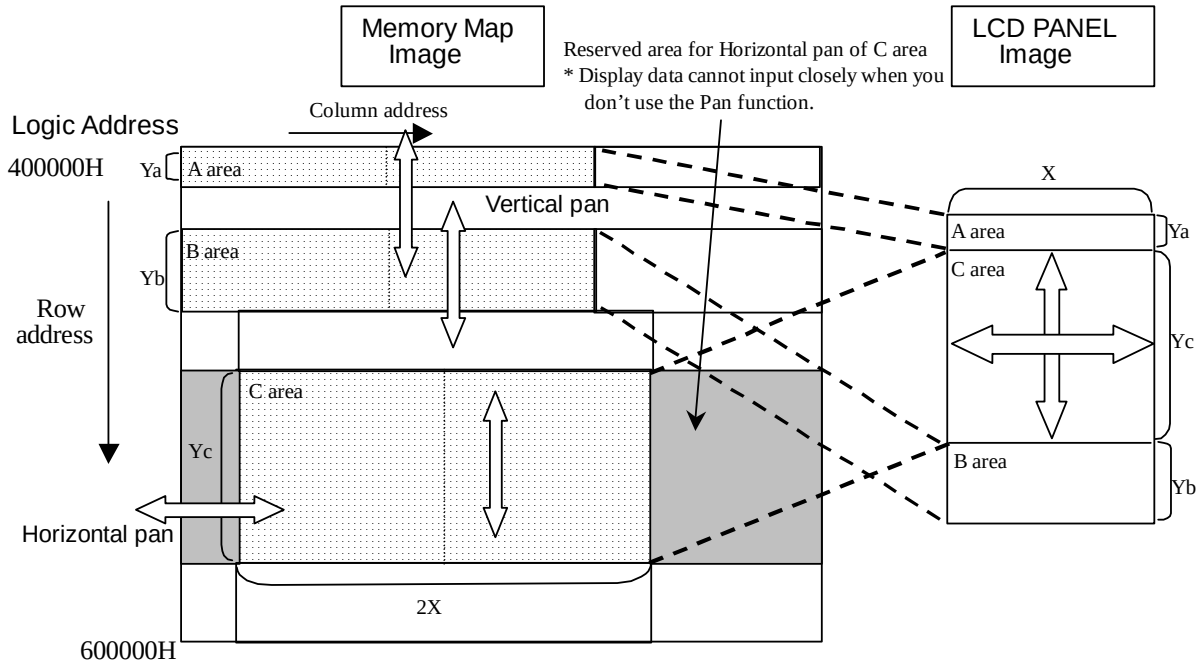
3.14.3.2 Memory Space (Common spec. SR and TFT mode)

The LCDC can display the LCD panel image which is divided horizontally into 3 parts; upper, middle and lower. Each area calls A, B and C area that has some characteristics showing below.

Start/End address of each area in the physical memory space can be defined in the LCD Start/End address registers. C area can be defined only start address.

A and B areas are programmable visibility and they are set enable or not in START ADDRESS register and ROW NUMBER register. When ROW NUMBER register of A and B area set to 0, the C area take over all panel space.

When the size of A or B area is greater than LCD panel, the area of the panel is all C area because the displaying priority is A > B > C. If the A area set to enable while the panel area is defined as all C area (that is A and B area are disable), C area is shifted to under the LCD panel and A area is inserted from the top of the LCD panel. Similarly if the B area set to enable while the panel area is defined as all C area , B area is inserted from the bottom of the C area overlapping .



Memory mapping from physical memory to LCD panel

3.14.3.3 Display memory mapping and Panning Function (Common spec. SR and TFT)

LCDC can change the panel window if only you change each start address of A, B and C area. Display area can be vertical and horizontal panned by changing row and column address.

This LCDC can select many display mode: 1bpp(monochrome), 2bpp(4gray), 3bpp(8gray), 4bpp(16gray), 8bpp(256 color) and 12bpp(4096 color) and 1-line(row) data volume is different each display mode. If use panning function, it need to care and calculate address each display mode. As Details, refer the follows page “Relation of Memory map image and Output data”. And this LCDC can support external SDRAM, SRAM and internal SRAM for display RAM.

When you use SDRAM for display RAM, data from one line to the next line cannot be input continuously in display RAM, even if you don’t use the panning function. One Row address of display SDRAM corresponds to 1’st line of display panel. Now display data of 2’nd line cannot be set within the 1’st row address of display RAM even if the necessary data for the size you want to display do not fill the capacity of 1’st row address of display SDRAM. Adding the one line to display panel is equal to adding one address to row address of display SDRAM. In other words, when using SDRAM for display RAM, address calculation for panning is easy.

When you use SRAM for display RAM, data from one line to the next line must be input continuously in display RAM. But address calculation for panning is complex and horizontal panning function is not support.

3.14.3.4 Data transmission

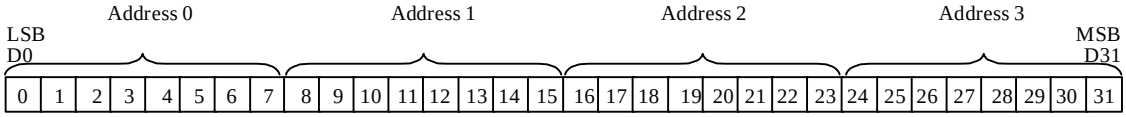
TMP91CH21 has LD bus(LD7 to LD0): special data bus for LCDD. And about that bus width can support 4, 8bit and each bus width can select 2 formats. 2 type formats of 8bit bus width can support only STN color mode (256, 4096 color). 12bit bus width is supported only TFT mode.

And LD bus data invert function is supported. By setting LCDMODE2<LDINV>=1, All LD bus data invert. There is <AUTOINV>bit in this LCDMODE2 register, but this automatic data invert function is for only TFT mode.

Relation of Memory map image and Output data

Monochrome (1 bpp (bit per pixel))

Display memory image

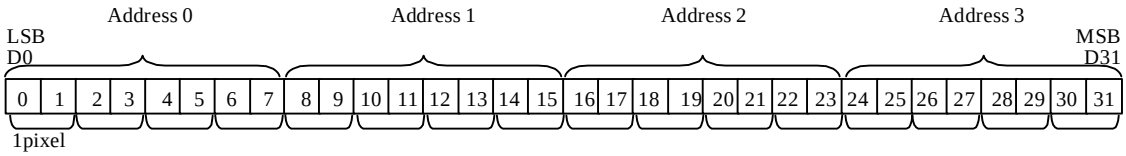


LD bus output sequence

4bitA type		4bitB type		8bit	
LD0	0 to 4 to 8 to 12 ~	LD0	4 to 0 to 12 to 8 ~	LD0	0 to 8 ~
LD1	1 to 5 to 9 to 13 ~	LD1	5 to 1 to 13 to 9 ~	LD1	1 to 9 ~
LD2	2 to 6 to 10 to 14 ~	LD2	6 to 2 to 14 to 10 ~	LD2	2 to 10 ~
LD3	3 to 7 to 11 to 15 ~	LD3	7 to 3 to 15 to 11 ~	LD3	3 to 11 ~
LD4	not use	LD4	not use	LD4	4 to 12 ~
LD5	not use	LD5	not use	LD5	5 to 13 ~
LD6	not use	LD6	not use	LD6	6 to 14 ~
LD7	not use	LD7	not use	LD7	7 to 15 ~

4Gray (2 bpp)

Display memory image



LD bus output sequence

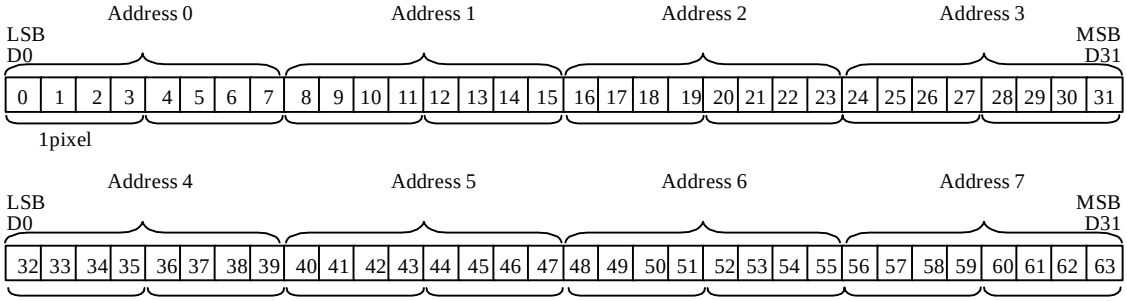
4bitA type		4bitB type		8bit	
LD0	1-0 to 9- 8 to 17-16 ~	LD0	9- 8 to 1-0 to 25-24~	LD0	1 - 0 to 17-16 ~
LD1	3-2 to 11-10 to 19-18 ~	LD1	11-10 to 3-2 to 27-26 ~	LD1	3 - 2 to 19-18 ~
LD2	5-4 to 13-12 to 21-20 ~	LD2	13-12 to 5-4 to 29-28 ~	LD2	5 - 4 to 21-20 ~
LD3	7-6 to 15-14 to 23-22 ~	LD3	15-14 to 7-6 to 31-30 ~	LD3	7- 6 to 23-22 ~
LD4	not use	LD4	not use	LD4	9- 8 to 25-24 ~
LD5	not use	LD5	not use	LD5	11-10 to 27-26 ~
LD6	not use	LD6	not use	LD6	13-12 to 29-28 ~
LD7	not use	LD7	not use	LD7	15-14 to 31-30 ~

Relation of Memory map image and Output data (1)

Relation of Memory map image and Output data

8/16Gray (4 bpp: 8gray case, valid data is 3bit but data space need to 4bit)

Display memory image



LD bus output sequence

4bitA type			4bitB type			8bitA type		
LD0	3-0	to 19-16 ~	LD0	19-16	to 3-0 ~	LD0	3-0	to 35-32 ~
LD1	7-4	to 23-20 ~	LD1	23-20	to 7-4 ~	LD1	7-4	to 39-36 ~
LD2	11-8	to 27-24 ~	LD2	27-24	to 11-8 ~	LD2	11-8	to 43-40 ~
LD3	15-12	to 31-28~	LD3	31-28	to 15-12 ~	LD3	15-12	to 47-44 ~
LD4	not use		LD4	not use		LD4	19-16	to 51-48 ~
LD5	not use		LD5	not use		LD5	23-20	to 55-52 ~
LD6	not use		LD6	not use		LD6	27-24	to 59-56 ~
LD7	not use		LD7	not use		LD7	31-28	to 63-60 ~

*8Gray data format is same to 16gray, 1pixel need to 4bit space. LSB bit is invalid data.

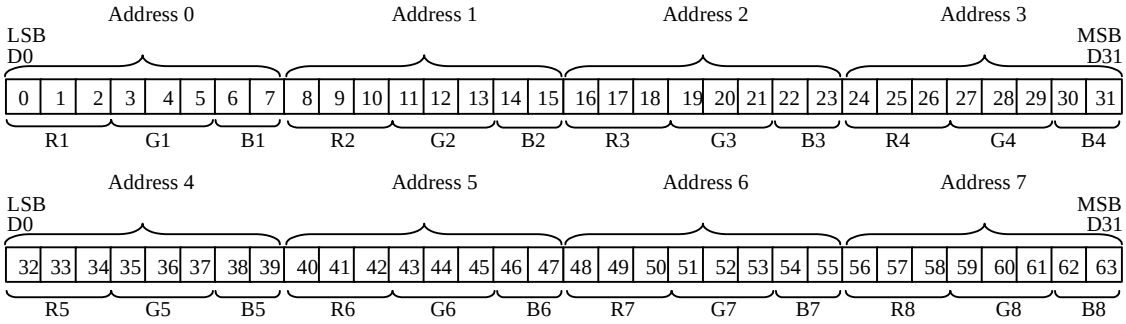
This mode don't support 8bitB type

Relation of Memory map image and Output data (2)

Relation of Memory map image and Output data

256 Color (8 bpp; R:3bit,G:3bit,B:2bit)

Display memory image



LD bus output sequence

4bitA type		4bitB type	
LD0	2-0(R1) to 13-11(G2) ~	LD0	13-11(G2) to 2-0(R1) ~
LD1	5-3(G1) to 15-14(B2) ~	LD1	15-14(B2) to 5-3(G1) ~
LD2	7-6(B1) to 18-16(R3) ~	LD2	18-16(R3) to 7-6(B1) ~
LD3	10-8(R2) to 21-19(G3) ~	LD3	21-19(G3) to 10-8(R2) ~
LD4	not use	LD4	not use
LD5	not use	LD5	not use
LD6	not use	LD6	not use
LD7	not use	LD7	not use

LD bus output sequence

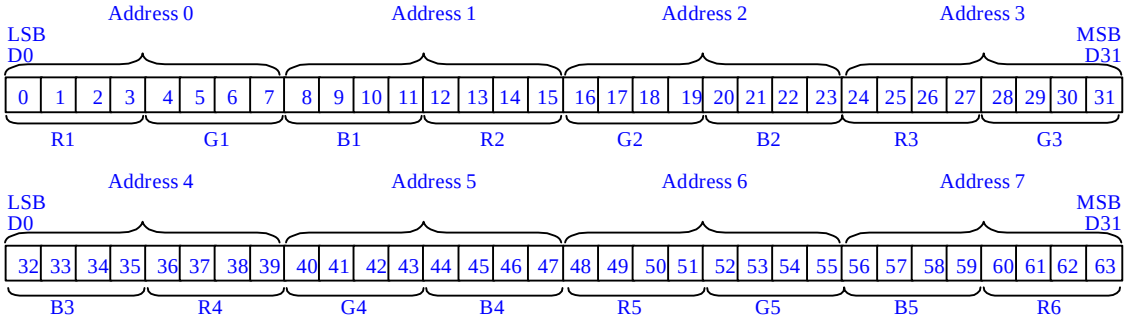
8bitA type		8bitB type	
LD0	2-0(R1) to 23-22(B3) ~	LD0	2-0(R1) to 7-6(G1) to 45-43(G6) to 47-46(B6)~
LD1	5-3(G1) to 26-24(R4) ~	LD1	5-3(B1) to 10-8(R2) to 50-48(R7) to 53-51(G7)~
LD2	7-6(B1) to 29-27(G4) ~	LD2	13-11 (G2) to 15-14(B2) to 55-54(B7) to 58-56(R8)~
LD3	10-8(R2) to 31-30(B4) ~	LD3	18-16(R3) to 21-19(G3) to 61-59(G8) to 63-62(B8)~
LD4	13-11(G2) to 34-32(R5) ~	LD4	23-22(B3) to 26-24(R4) to 66-64(R9) to 69-67(G9)~
LD5	15-14(B2) to 37-35(G5) ~	LD5	29-27(G4) to 31-30(B4) to 71-70(B9) to 74-72(R10)~
LD6	18-16(R3) to 39-38(B5) ~	LD6	34-32(R5) to 37-35(G5) to 77-75(G10) to 79-78(B10) ~
LD7	21-19(G3) to 42-40(R6) ~	LD7	39-38(B5) to 42-40(R6) to 82-80(R11) to 85-83(G11) ~

Relation of Memory map image and Output data (3)

Relation of Memory map image and Output data

4096 Color (12 bpp: R:4bit, G:4bit, B:4bit)

Display memory image



LD bus output sequence

4bitA type		4bitB type		8bitA type	
LD0	3-0(R1) to 19-16(G2) ~	LD0	19-16(G2) to 3-0(R1) ~	LD0	3-0(R1) to 35-32(B3) ~
LD1	7-4(G1) to 23-20(B2) ~	LD1	23-20(B2) to 7-4(G1) ~	LD1	7-4(G1) to 39-36(R4) ~
LD2	11-8(B1) to 27-24(R3) ~	LD2	27-24(R3) to 11-8(B1) ~	LD2	11-8(B1) to 43-40(G4) ~
LD3	15-12(R2) to 31-28(G3)~	LD3	31-28(G3) to 15-12(R2) ~	LD3	15-12(R2) to 47-44(B4)~
LD4	not use	LD4	not use	LD4	19-16(G2) to 51-48(R5)~
LD5	not use	LD5	not use	LD5	23-20(B2) to 55-52(G5)~
LD6	not use	LD6	not use	LD6	27-24(R3) to 59-56(B5)~
LD7	not use	LD7	not use	LD7	31-28(G3) to 63-60(R6)~

*8Gray data format is same to 16gray, 1pixel need to 4bit space. LSB bit is invalid data.

This mode don't support 8bitB type

Relation of Memory map image and Output data (4)

3.14.3.5 Refresh rate setting

Frame cycle (refresh rate) is generated from setting of LSCC and FFP. LBCD terminal out one pulse every this cycle and LFR normally out invert signal every this cycle. But when DEVIDE FRAME function is used, LFR signal change to special signal for high quality display.

(1) Basic clock setting

TMP91CH21 has special clock generator for basic source clock used in LCD controller. This generator can set detail refresh rate for LCDC.

This generator is made by divide by 16 of System clock and 8bit counter.

It shows that the way of setting and calculation.

f_{BCD} [Hz] : Frame rate (refresh rate: Frequency of LBCD signal)

FP : FP [9:0] setting value of FFP register

SCC : SCC [7:0] setting value of LSCC register

$$f_{BCD}\ [Hz] = f_{SYS}\ [Hz] / (SCC * 16 * FP)$$

Ex.) f_{SYS} [Hz] = 20MHz, 480COM (FP = 480), target refresh rate = 140Hz

140 [Hz] = 20000000 [Hz] / (SCC * 16 * 480)
SCC = 20000000 / (140 * 16 * 480) = 18.60

Value of setting to register is only integer, SCC=18. It cut off the floating value.

This time, refresh rate comes to 144.6 [Hz]

LCDC Source Clock Counter Register

		7	6	5	4	3	2	1	0
LCDSCC (0287H)	bit Symbol	SCC7	SCC6	SCC5	SCC4	SCC3	SCC2	SCC1	SCC0
	Read/Write	R/W							
	After Reset	0							
	Function	LCDC Source Clock Counter bit7-0							

* Usually it should be write from 1-hex to FFFF-hex data in above register. If set to “0”, can’t operate.
* If set the refresh rate too fast, it may not in time to trance display data. tLP time is decided by SCC.

$$t_{LP}\ [s] = (1 / f_{SYS}\ [Hz]\ ) * 16 * SCC$$

tLP is shows 1-line (ROW) display time. It must need to finish data transmission of 1-line data during tLP cycle time. About data transmission time, refer from “Data transmission and bus occupation”.

(2) Refresh rate adjust function (correct function)

In this function, LBCD frequency: refresh rate is generated by setting by LCDSCC<SCC[7:0]> and FFP<FP[9:0]> register. FFP value is normally set same value of ROW number, but this value can use for correction of BCD frequency: refresh rate.

This function always set bigger value than ROW number, set to slower frequency. If set smaller value than ROW number, LCDC can't operate correctly.

It shows setting example as follows.

Ex.) f_{SYS} [Hz] = 20MHz, 480COM (FP = 480), target refresh rate = 140Hz

140 [Hz] = 20000000 [Hz] / (SCC * 16 * 480)
SCC = 20000000 / (140 * 16 * 480) = 18.60

Value of setting to register is only integer, SCC=18. It cut off the floating value.

This time, refresh rate comes to 144.6 [Hz]

f_{BCD} [Hz] = f_{SYS} [Hz] / (SCC * 16 * FP)

FP value is adjusted to set SCC=18 in above equation again.

140 [Hz] = 20000000 / (18 * 16 * FP)

FP = 496.03

Value of setting to register is only integer, FP=496.

This time, refresh rate comes to 140.0 [Hz]

LCD fFP Register

LCDFFP (0282H)

	7	6	5	4	3	2	1	0
bit Symbol	FP7	FP6	FP5	FP4	FP3	FP2	FP1	FP0
Read/Write	R/W							
After Reset	0							
Function	Setting bit7-0 for f FP							

Ref.) We recommend Refresh rate value for around …

Monochrome: 70 [Hz]

4, 8, 16gray and color: 140 to 200 [Hz]

(3) Divide frame adjust function

We call divide frame adjust function that can available to decrease uneven display in big size LCD panel.

When this function is enabled by set to <FRMON>=1, LFR signal change out level high and low, every product of LLP cycle and setting value in follows LCDDVM register.

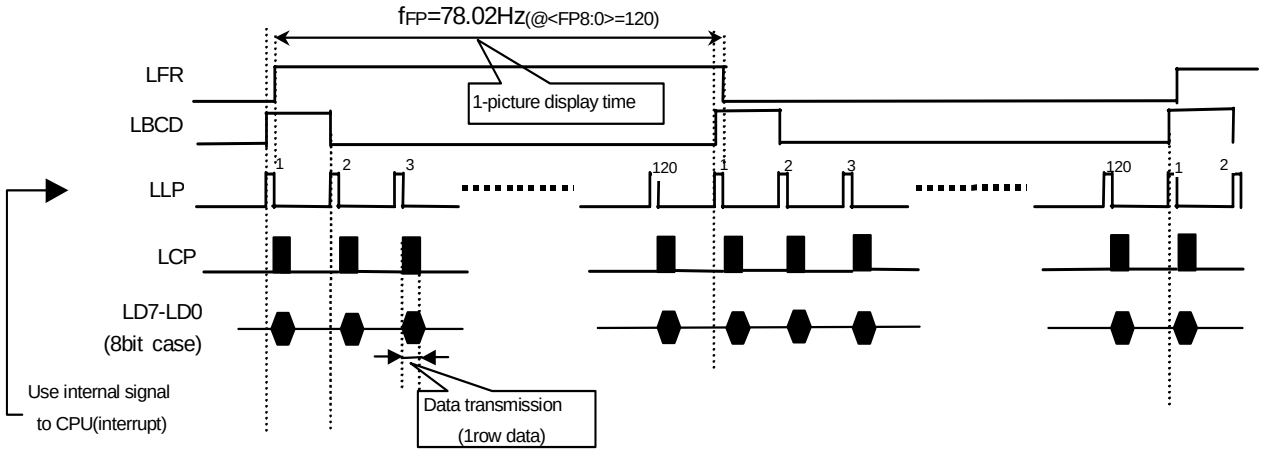
When this function is disabled by set to <FRMON>=0, LFR signal change out level high and low, every LBCD cycle. This function is no relation with LBCD's timing.

NOTE) Availability of this function is depend on actual using LCD driver or LCD panel. We recommend checking that register's value in using real environment.

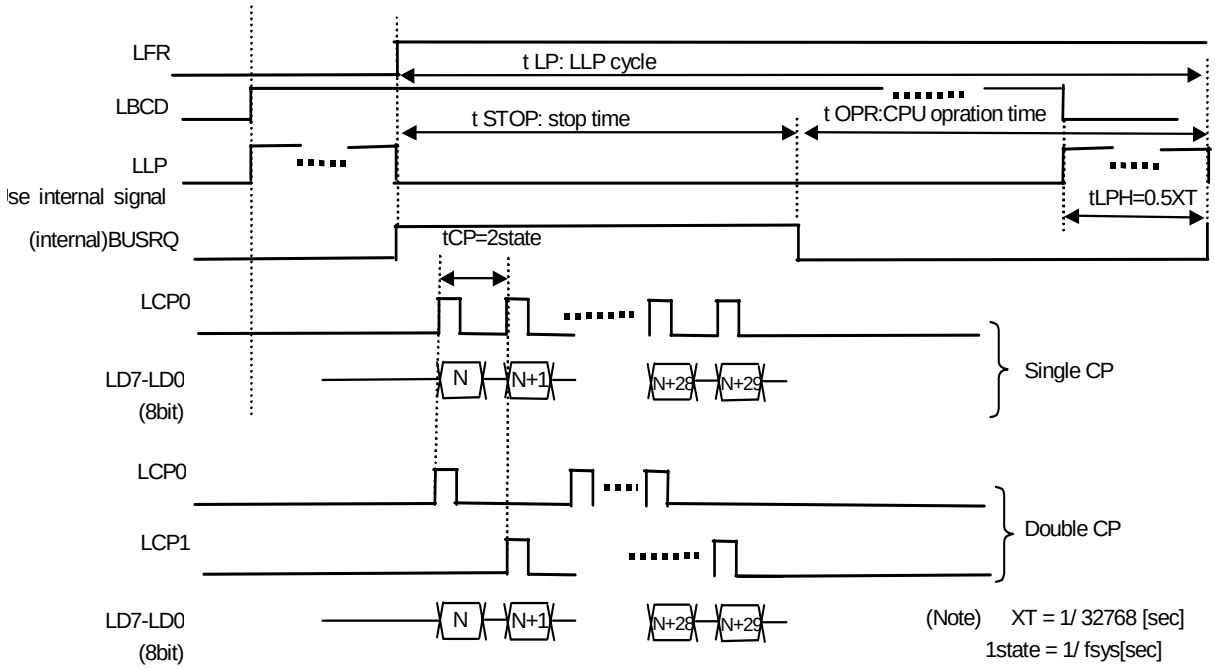
Divide FRM Register

		7	6	5	4	3	2	1	0
LCDDVM (0283H)	bit Symbol	FMN7	FMN6	FMN5	FMN4	FMN3	FMN2	FMN1	FMN0
	Read/Write	R/W							
	After Reset	0							
	Function	Setting DVM bit7-0							

Ref.) In generally, the value of LCDDVM register is good for prime number (3,5,7,11,13 etc).

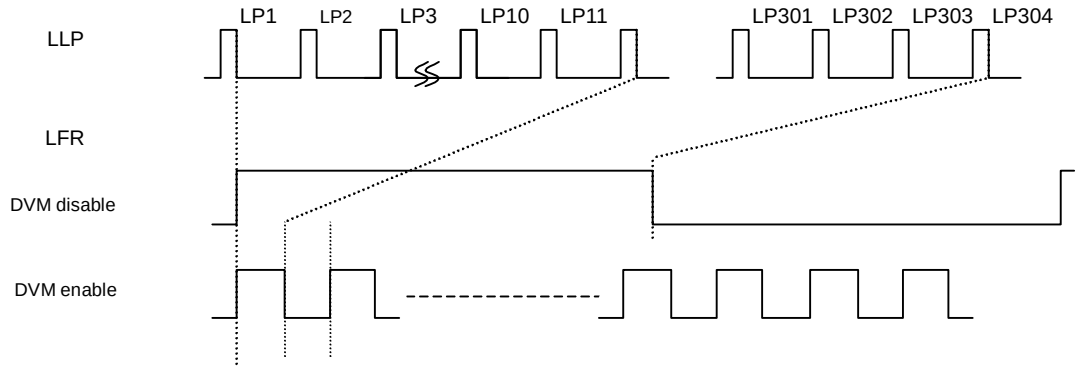


Whole timing diagram of SR mode



Detail timing diagram of SR mode

Condition: FFP[9:0] setting=240(COM)+63, LCDDVM<FMN7:0>=0B_hex



Waveform of LLP, LFR

3.14.3.6 LCD data transmission speed and data bus occupation rate

After set start register LCDC outputs bus release request to CPU and read data from source memory. After that LCDC transmits data of volume of LCD size to external LCD driver through special LCDC data bus (LD11-LD0). At this time, control signals connected LCD driver output specified waveform synchronizes with data transmission. After finish data reading from RAM for display, LCDC cancels the bus release request and CPU will re-start.

During data read from source memory (during DMA operation), CPU is stopped by internal BUSREQ signal. When using SR mode LCDC, programmer need to care the CPU performance. This occupation rate of data bus is depend on data size, transmission speed (CPU clock speed) and using display RAM type.

Display RAM	Bus width	Valid data reading time (fSYS Clock/Byte)	Valid data reading time TLRD (nS/Byte) @fSYS=20MHz
External SRAM	16bit	2／2	50
	32bit	2／4	25
Internal RAM	32bit	1／4	12.5
External SDRAM	16bit	*1／2	*25
	32bit	*1／4	*12.5

NOTE) When using SDRAM for display RAM, it needs over head time (+ 8clk) every 1 ROW data reading.

We call tSTOP: CPU stop time during transmission of 1 ROW data. tSTOP is calculated by below equation each display mode.

tSTOP = (ColNum*K / 8) * tLRD ;except SDRAM use

ColNum: Number of column
K: bit number per pixel (bpp)

Monochrome	K=1
4 Gray	K=2
8/16Gray	K=4
256 color	K=8
4096 color	K=12

In case of SDRAM use, need more over head time.

tSTOP = (ColNum * K / 8) * tLRD + ((1 / fSYS) * 8) ;SDRAM use

Data bus occupation rate shows percentage of tSTOP time in tLP time.

Data bus occupation rate = tSTOP / tLP

NOTE) About tLP time, refer “Refresh rate setting”.

3.14.3.7 Timing Diagram of LD bus

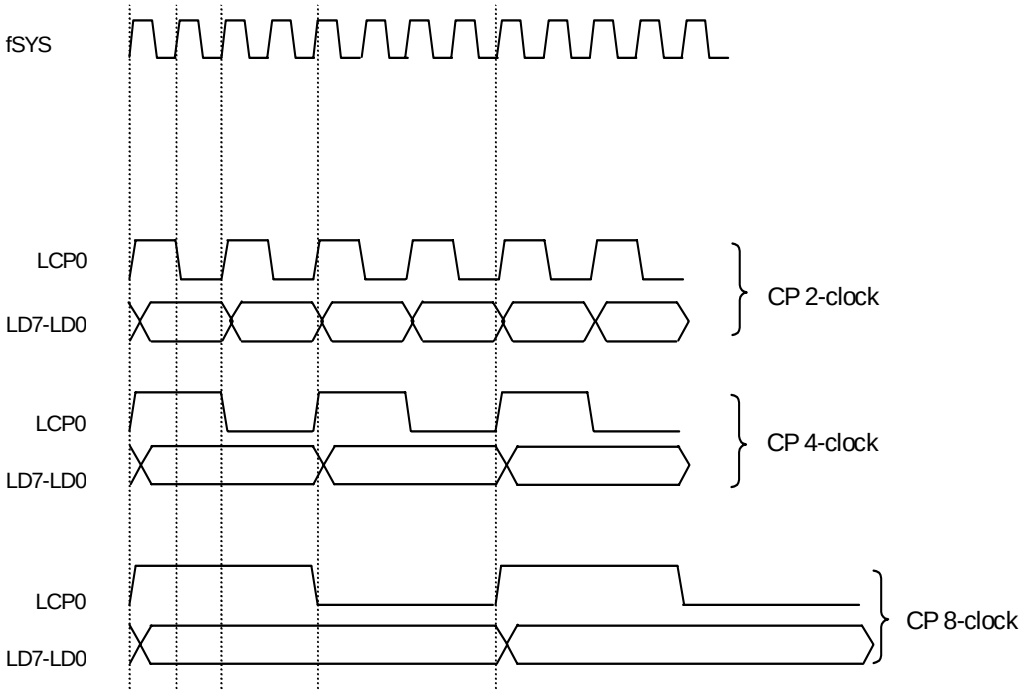
TMP92CH21 can select to display RAM for external SRAM: available to set WAIT, internal SRAM and external SDRAM: 16, 32, 64, 128, 256 and 512Mbit.

And as 480Bytt FIFO buffer is built in this LCDC, LD bus speed can control.

That speed can select in 3 kind of cycle: (fsys/2, fsys/4, fsys/8)

LD bus data:LD11 to LD0 is out at rising edge of LCP0, LCDD receive at falling edge of LCP0.

NOTE) It need to care that LCP cycle is too slow to trance correctly.



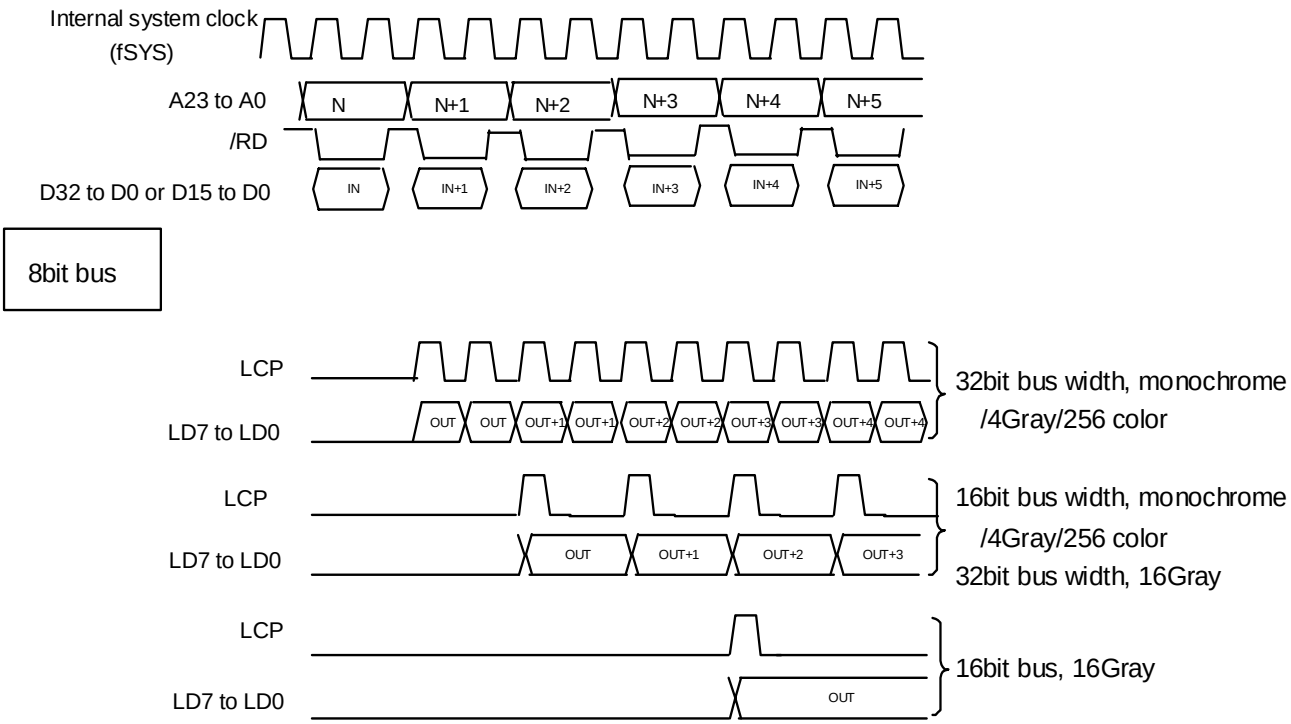
Selection of LCP cycle

If LCP cycle is not set for suitable speed against refresh rate, LD bus data is not able to trance correctly. tLP time shows below equation.

$$t_{LP}\ [s] = (1 / f_{SYS}\ [Hz]\ ) * 16 * SCC$$

Data transmission must finish in tLP time. Set to SCC clock and LCP0 speed to be less than tLP time. About the setting of SCC, refer “Basic clock setting” of “refresh rate setting”.

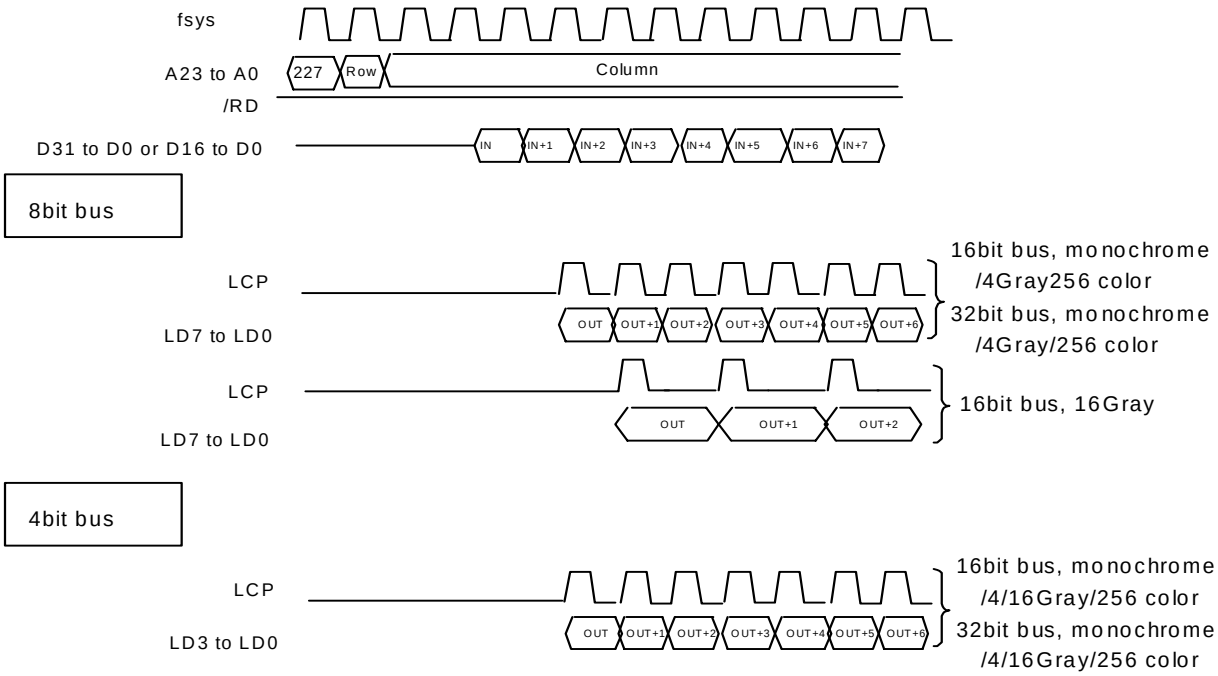
The kind of display memory and display mode decide LCP speed. In other words, if it set too fast setting case: not enough the transmission data in FIFO, LCD data can’t trans correctly.



Fastest timing diagram for external SRAM, 0-WAIT

* When using internal SRAM, it always select 32bit bus width and 0-wait, 1CLK access.

Timing diagram for internal SRAM



Timing diagram of SDRAM burst run

3.14.3.8 Setting of color palette

TMP92CH21 can support monochrome, 4,8,16 level gray scale, color STN panels, and color TFT panels. It shows as follows how to set each mode.

Monochrome

No need to special setting, it's just select monochrome mode by LCDMODE1<MODE[3:0]> register.

4, 8, 16gray

No need to special setting as same as monochrome mode, it's just select monochrome mode by LCDMODE1<MODE[3:0]> register.

In case of 8 and 16Gray mode, both setting needs 4bit data per 1 pixel. Even if set to 8Gray mode, LSB bit of display data is invalid.

STN256 color

Firstly, select STN256 mode by LCDMODE1<MODE[3:0]> and next set the detail contrast adjustment. In case of 256STN mode, 8bit display data is divided to 3bit (Red), 3bit (Green) and 2bit (Blue). Red and green are 8 level contrast and blue is 4 level contrast, each level can adjust from serious 16 level contrast.

Red contrast level can select by LCDRP10, LCDRP32, LCDRP54 and LCDRP76, in these total 4 registers, incase of green, LCDGP10, LCDGP32, LCDGP54 and LCDGP76, in case of blue, LCDBP10, LCDBP32.

At the result, it support for 8bpp:256 colors out of a palette of 4096 colors.

Color palette setting (RED)

Selectable 8bit	<<<							contrast							>>>
Data	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
7	*	*	*	*	*	*	*	*	*	*	*	*	*	*	@	*
6	*	*	*	*	*	*	*	*	*	*	*	*	@	*	*	*
5	*	*	*	*	*	*	*	*	*	*	@	*	*	*	*	*
4	*	*	*	*	*	*	*	*	@	*	*	*	*	*	*	*
3	*	*	*	*	*	*	@	*	*	*	*	*	*	*	*	*
2	*	*	*	*	@	*	*	*	*	*	*	*	*	*	*	*
1	*	*	@	*	*	*	*	*	*	*	*	*	*	*	*	*
0	@	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Color palette setting (GREEN)

Selectable 8bit	<<<							contrast							>>>
Data	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
7	*	*	*	*	*	*	*	*	*	*	*	*	*	*	@	*
6	*	*	*	*	*	*	*	*	*	*	*	*	@	*	*	*
5	*	*	*	*	*	*	*	*	*	*	@	*	*	*	*	*
4	*	*	*	*	*	*	*	*	@	*	*	*	*	*	*	*
3	*	*	*	*	*	*	@	*	*	*	*	*	*	*	*	*
2	*	*	*	*	@	*	*	*	*	*	*	*	*	*	*	*
1	*	*	@	*	*	*	*	*	*	*	*	*	*	*	*	*
0	@	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Color palette setting (BULE)

Selectable 8bit	<<<							contrast							>>>
Data	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3	*	*	*	*	*	*	*	*	*	*	*	*	@	*	*	*
2	*	*	*	*	*	*	*	*	@	*	*	*	*	*	*	*
1	*	*	*	*	@	*	*	*	*	*	*	*	*	*	*	*
0	@	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

*: selectable contrast point

@: Initial setting point

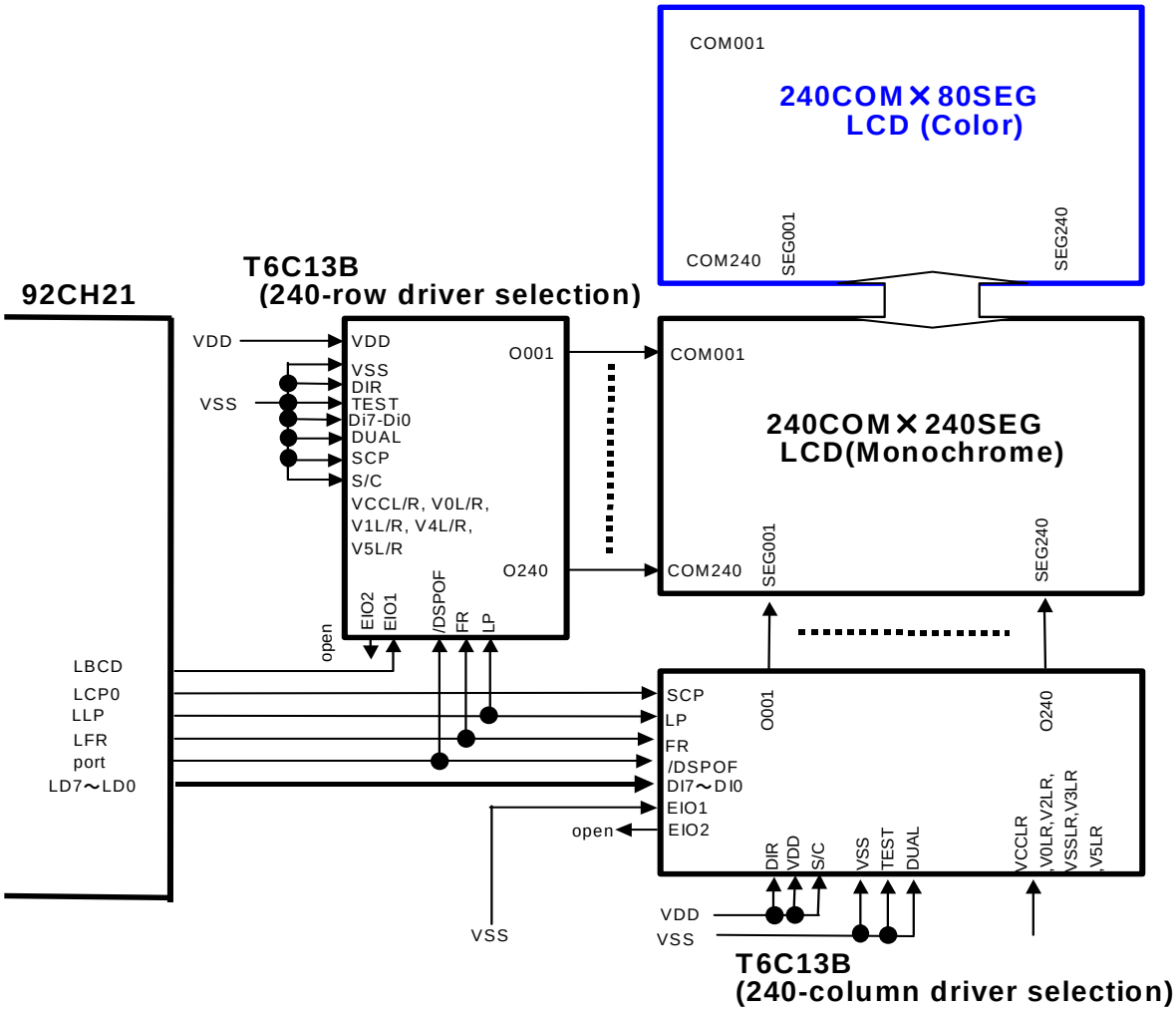
Palette setting of each basic color (RGB)

STN4096 color

It just select STN4096 mode by LCDMODE1<MODE[3:0]>.

This LCDC is 4096 color palette maximum. If select STN4096 color mode, it can’t adjust each color contrast level.

3.14.3.9 Example to connect SR type LCDD



(Note) Other circuit is necessary for LCD drive power supply for LCD driver display.

Interface example for Shift register type LCD DRIVER

3.14.3.10 Program example (4K color STN)

```
;******** PORT settings *********
        ld      (plfc),0ffh         ; LD7-LD0  set
        ld      (plcr),0f0h         ; output mode
        ld      (pkfc),0fh          ; LBCD,LLP,LCP0)

;******** LCD settings *********
        ld      (LCDSCC),31     ; counter for LP (Refresh Rate:220Hz@27MHz)
        ld      (LCDCCR0),01h   ;
        ld      (LCDCCR1),01h   ; SCP Negedge
        ld      (LCDCCR2),02h   ;

        ld      (LCDSIZE),64h   ; 240comx256seg
        ld      (LCDFFP),240    ; 240com
        ld      (LCDMODE0),096h         ; SDRAM, STN:4K
        ld      (LCDMODE1),02h          ; 8bit A type

        ld      (LSARCL),00h    ; C area
        ld      (LSARCM),00h    ;
        ld      (LSARCH),40h    ;

        ld      (LSARAL),00h    ; A area
        ld      (LSARAM),00h    ;
        ld      (LSARAH),00h    ;

        ld      (LSARBL),00h    ; B area
        ld      (LSARBM),00h    ;
        ld      (LSARBH),00h    ;

        ld      (CMNAL),00h     ; A area Row number
        ld      (CMNAH),00h     ;

        ld      (CMNBL),00h     ; B area Row number
        ld      (CMNBH),00h     ;

        ld      (LCDCTL1),0e0h  ; SCP0,SCP1:negedge, BCD:
        ld      (LCDDVM),3
        ld      (LCDCTL0),01h   ; START (FP bit8=0)

end
```

3.14.4 TFT color display mode

3.14.4.1 Description of operation

It is basically same setting to SR mode.

Set the mode of operation, start address of source data save memory, color level and LCD size to control registers before setting start register.

After set start register, LCDC outputs bus release request to CPU and read data from source memory. After finish data read from source data, LCDC cancels the bus release request and CPU will re-start. After that LCDC transmits data of volume of LCD size to external LCD driver through LD bus (Special data bus for only LCDD). At this time, control signals (LCP0 etc.) connected LCD driver output specified waveform synchronizes with data transmission.

TFT mode LCDC, during data read from source memory (during DMA operation), CPU is stopped by internal BUSREQ signal.

LCD controller generates some control signals (LFR, LBCD, LLP etc) from base clock: LCDSCC. LCDSCC is base clock for LCD controller, made from system clock: f_{SYS}.

For TFT source driver, support these signals: 8bitRGB or 4bit*RGB special data bus and LCP0, LFR, LLP and LDIV.

And for TFT gate driver control, LCP1, LBCD and LGOE2 to 0.

3.14.4.2 Memory space

Memory space setting is same to SR mode. Refer to SR mode.

3.14.4.3 Mapping of display memory and panning function

Panning function and display memory mapping are same to SR mode. Refer to SR mode.

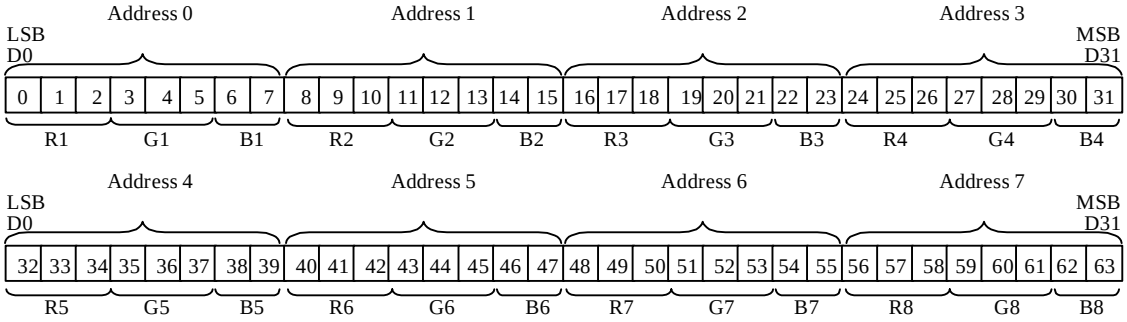
3.14.4.4 Data transmission

TMP91CH21 out display data form special bus for LCDD. And we can select that width for input width of LCDD. 8bit and 12bit width is supported.

Relation of Memory map image and Output data

256 Color (8 bpp; R:3bit,G:3bit,B:2bit)

Display memory image



LD bus output sequence

8bit (TFT)

LD0 0(R1) to 8(R2) ~

LD1 1(R1) to 9(R2) ~

LD2 2(R1) to 10(R2) ~

LD3 3(G1) to 11(G2) ~

LD4 4(G1) to 12(G2) ~

LD5 5(G1) to 13(G2) ~

LD6 6(B1) to 14(B2) ~

LD7 7(B1) to 15(B2) ~

*When use 256 color TFT mode, it need to use 8bit LD bus width.

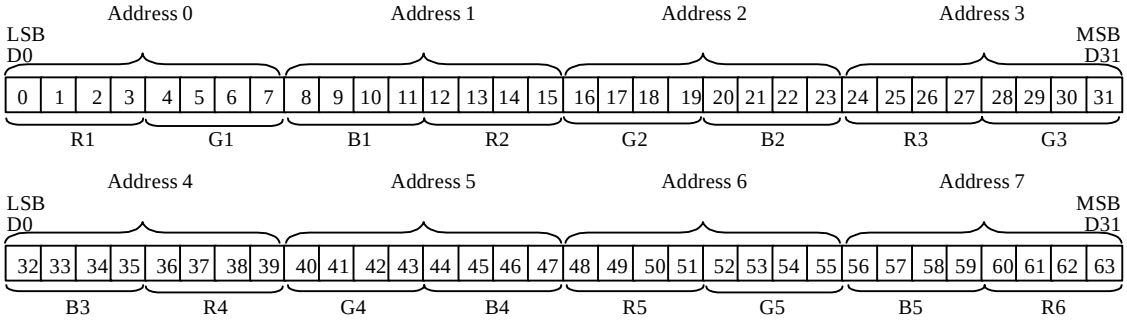
LD8,LD9,LD10 and LD11 terminals can available to use for general port.

Relation of Memory map image and Output data (5)

Relation of Memory map image and Output data

4096 Color (12 bpp; R:4bit,G:4bit,B:4bit)

Display memory image



LD bus output sequence

12bit (TFT)

LD0 0(R1) to 12(R2) ~

LD1 1(R1) to 13(R2) ~

LD2 2(R1) to 14(R2) ~

LD3 3(R1) to 15(R2) ~

LD4 4(G1) to 16(G2) ~

LD5 5(G1) to 17(G2) ~

LD6 6(G1) to 18(G2) ~

LD7 7(G1) to 19(G2) ~

LD8 8(B1) to 20(B2) ~

LD9 9(B1) to 21(B2) ~

LD10 10(B1) to 22(B2) ~

LD11 11(B1) to 23(B2) ~

Relation of Memory map image and Output data (6)

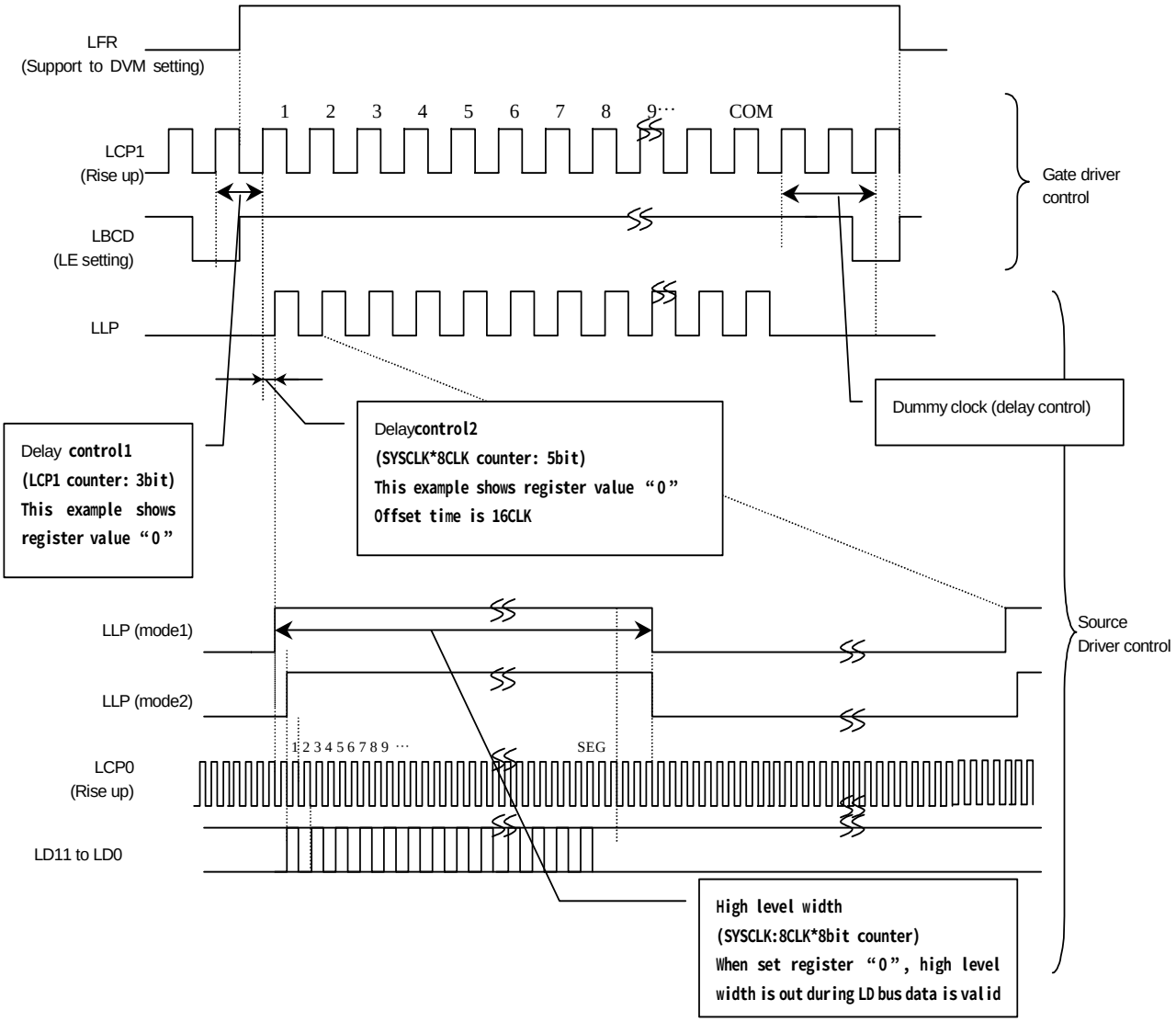
3.14.4.5 Setting each control signals

TFT source driver is controlled by LCP1:base clock (LCP0), data start clock (LFR) and load pulse (LLP). Special data bus: LD11 to 0 is used 8bit of 12bit for suitable LCDD.

Each signal can seriously adjust timing using some control register.

When use TFT driver, much data need to use. So, during wide bus and high speed transmission, some noise may generate. TMP92CH21 has LDIV function. This function set minimum data change method automatically, from inverting data and LDIV signal, to compare one data and one data of one time before. If TFT LCDD is supported data inverting function, it's possible to decrease the noise.

It shows basic timing as follows.



Timing diagram of TFT driver control

LCD Clock Counter Register0

LCDCCR0 (0288H)

	7	6	5	4	3	2	1	0
bit Symbol						PCPV2	PCPV1	PCPV0
Read/Write						R/W	R/W	R/W
After Reset						0	0	0
Function						Pre LCP1 CLK: Dummy clock number to valid clock of LCP1		

LCD Clock Counter Register1

LCDCCR1 (0289H)

	7	6	5	4	3	2	1	0
bit Symbol				TLDE4	TLDE3	TLDE2	TLDE1	TLDE0
Read/Write				R/W	R/W	R/W	R/W	R/W
After Reset				0	0	0	0	0
Function	Setting register of LLP enable time							

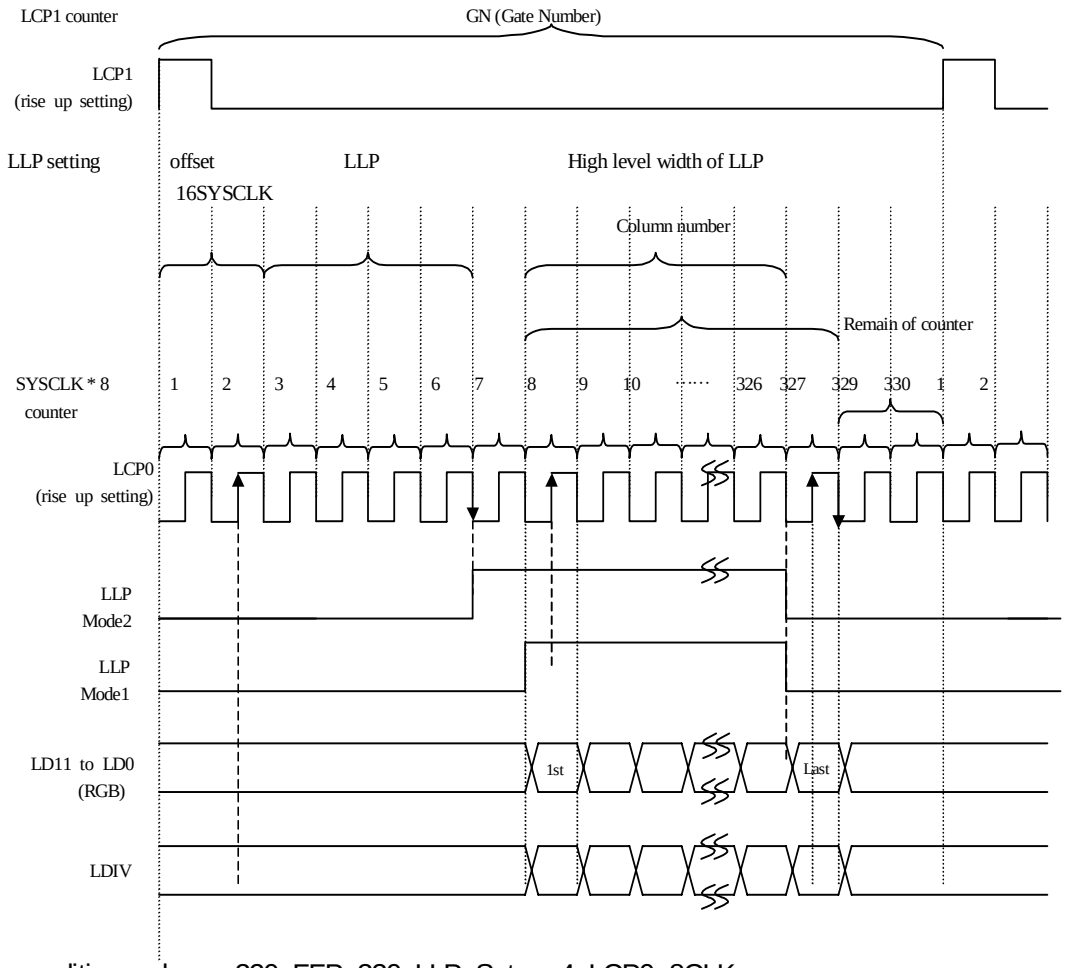
LCD Clock Counter Register2

LCDCCR2 (028AH)

	7	6	5	4	3	2	1	0
bit Symbol	LLPSU7	LLPSU6	LLPSU5	LLPSU4	LLPSU3	LLPSU2	LLPSU1	LLPSU0
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	Set-up time of LLP							

3.14.5 Source driver control

Detail timing is below



condition: column=320, FFP=330, LLP_Setup=4, LCP0=8CLK

Timing example for TFT driver

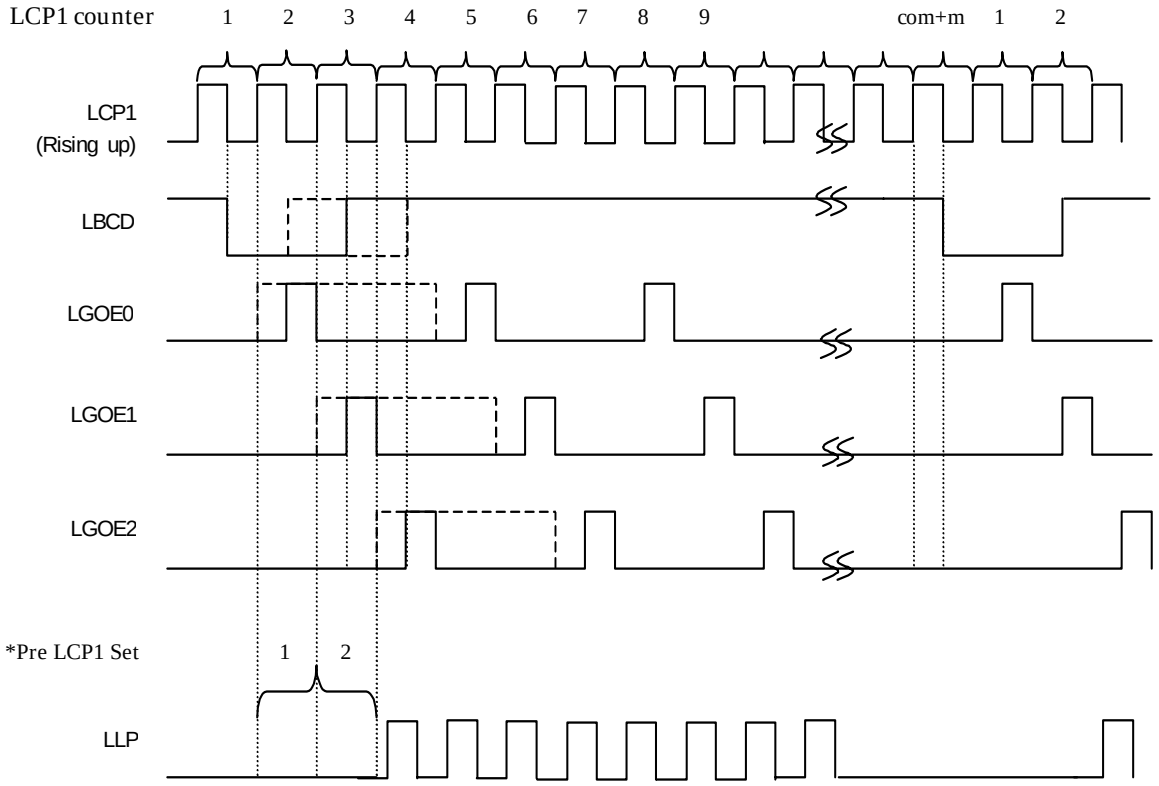
NOTE) Above figure and explanation shows under this condition.

<LCDCTL2>CPHP=1,CPVP=0

3.14.5.1 Gate driver control

TFT gate driver is controlled by LCP1:base clock and LBCD: vertical shift data signal. TMP92CH21 has LGOE2 to LGOE0:3bit output enable signals and these LGOE2 to LGOE0 can available to control individually. TFT gate driver's output can be controlled by this serious timing and can available to make blanking adjustment and to zoom function.

LBCD signal start to out from rising up of LCP1 and TFT gate driver recognize start point of vertical direction. Just after then, LGOE0 out from rising up timing of LCP1 and repeat. LGOE1 out one LCP1 clock delayed. LGOE2 delay one LCP1 clock from LGOE1.By <LCDCTL2>GDIOW[1:0] setting, Width of LBCD can select 1, 2, or 3clk of LCP1.



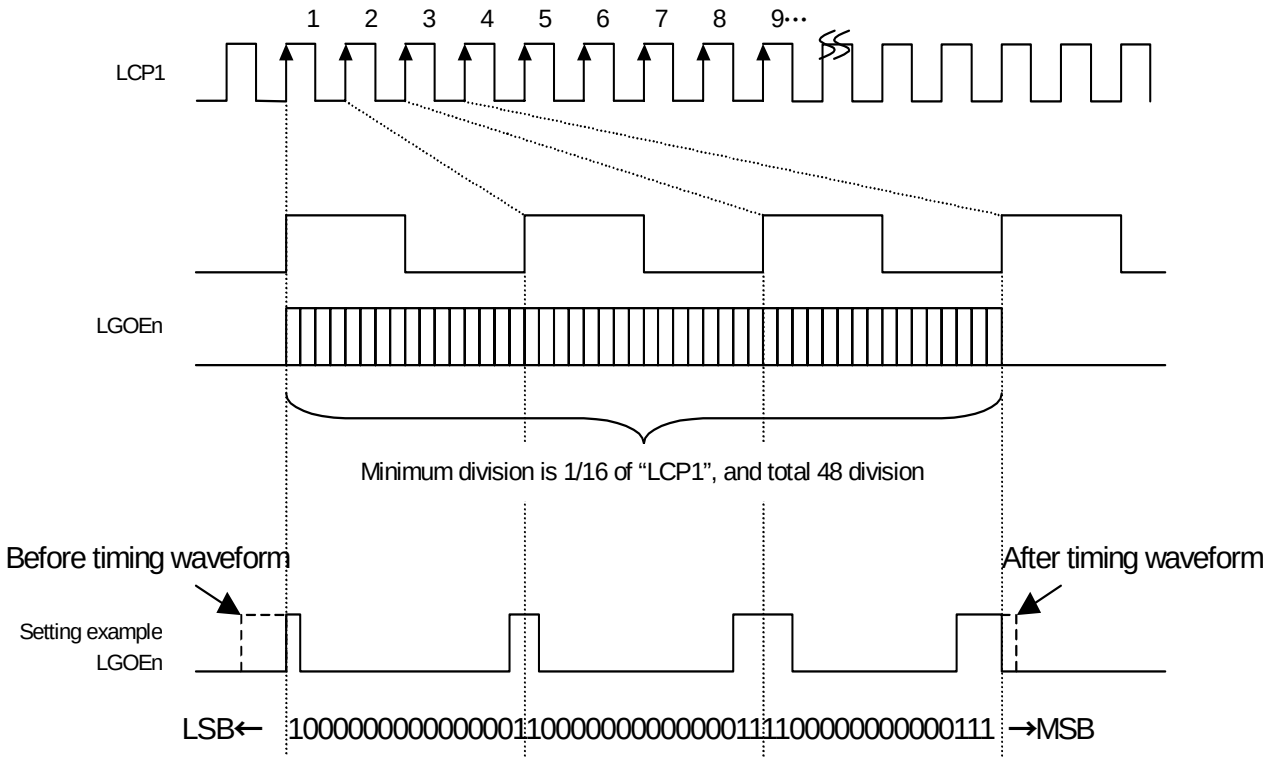
Note1) LCP1 counter: 1024clk maximum

Note2) Pre_LCP1_SET: 8clk of LCP1 maximum

Example of TFT gate driver timing control

TMP92CH21 has LGOE2 to LGOE0 pins to control 3kind of gate output individually. LGOE0 start to out from rising up timing of LCP1's 1'st pulse and repeat out each 3 pulse of LCP1. LGOE1 start to out from LCP1's 2'nd pulse, LGOE2 start to out from LCP1's 3'rd pulse. And all LGOE repeat out. LGOE2 to LGOE0 signal can adjust 1/16 of LCP1 cycle minimum step, and that information from 48bit control register.

TFT gate driver's output can be controlled by this serious timing and can available to make blanking adjustment and to zoom function.



* It can make free waveform to write to 48bit register in LCDOEn5 to LCDn0

(Write to 1, high-level out and write to 0, low-level out. LSB fast)

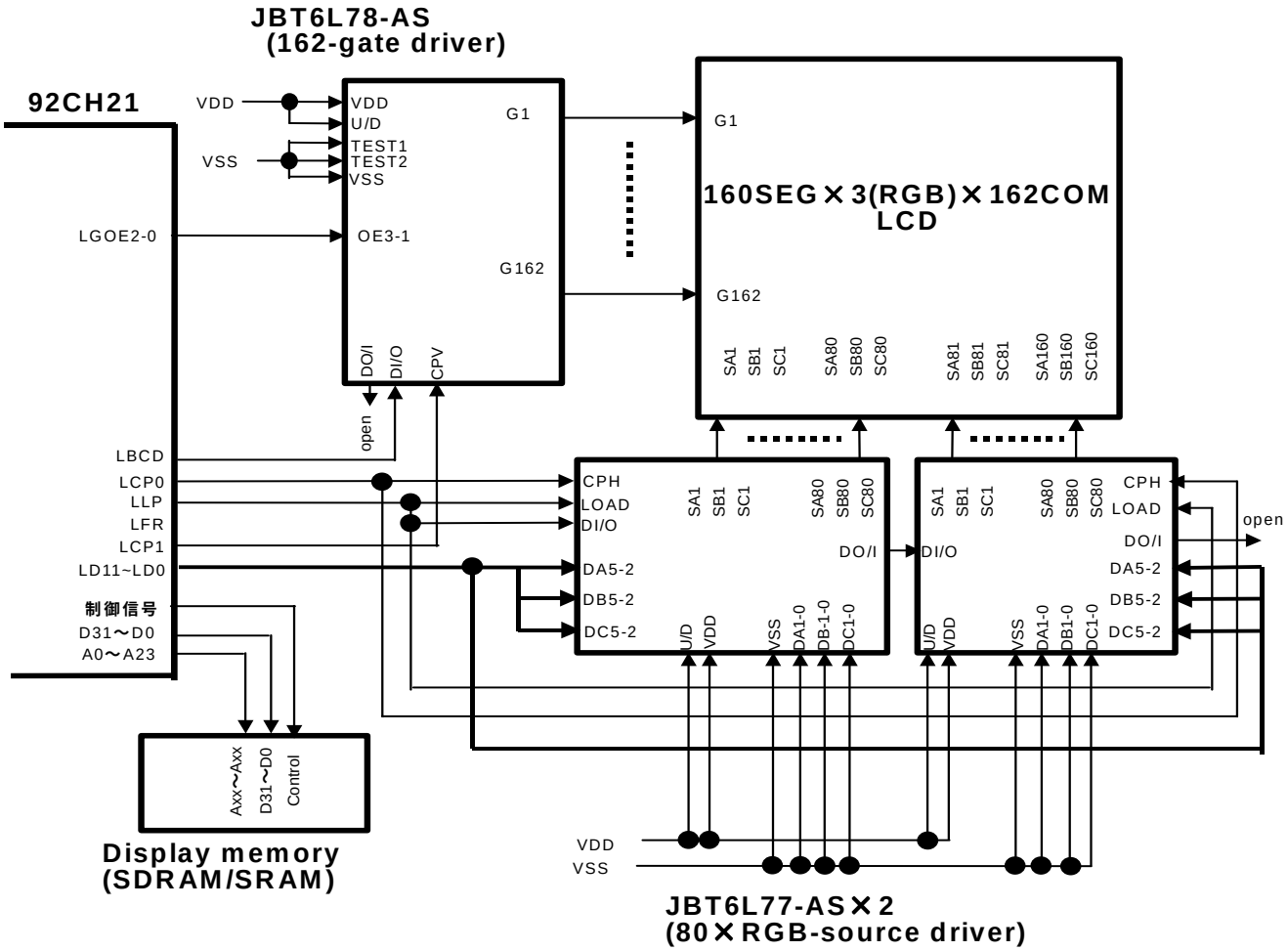
Detail waveform GOEn signal for TFT gate driver

Note) Above explanation and figure are follow setting case.

Condition: <LCDCTL2>CPHP=1,CPVP=0

In case of CPHP=0,CPVP=1 setting, LCP0 and LCP1 phase is invert

3.14.5.2 Example to connect TFT LCDD



(Note): Need other circuit to supply power for driving LCDD

Interface example for TFT type LCD DRIVER

3.14.6 Built-in RAM type LCDD mode

3.14.6.1 Description of operation

Data transmission to LCD driver is executed by move instruction of CPU.

After setting mode of operation to control register, when move instruction of CPU is executed LCDC outputs chip select signal to LCD driver connected to the outside from control pin (LCP0 etc.). Therefore control of data transmission numbers corresponding to LCD size is controlled by instruction of CPU. There are 2 kinds of address of LCD driver in this case, and which is chosen determines by LCDCTL <MMULCD> register.

3.14.6.2 Random access type

It corresponds to address direct writing type LCD driver at the time of <MMULCD> ="1." The transmission place address at this time can also assign the memory area of 3C0000H - 3FFFFF to four area for every 64 K bytes. **(RANDOM ACCESS TYPE)**

About interface and access timing, it's same to normal memory. Refer the memory access timing.

address	function	Chip enable terminal
3C0000H to 3CFFFFH	Built-in RAM LCDD1	LCP0
3D0000H to 3DFFFFH	Built-in RAM LCDD2	LLP
3E0000H to 3EFFFFH	Built-in RAM LCDD3	LFR
3F0000H to 3FFFFFH	Built-in RAM LCDD4	LBCD

3.14.6.3 Sequential access type

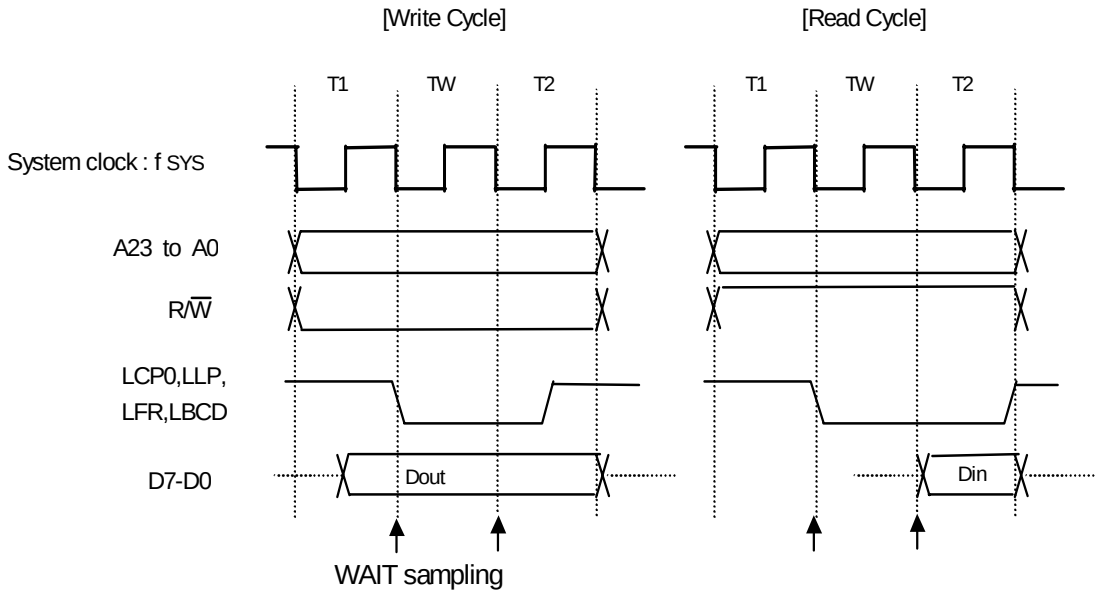
Data transmission to LCD driver is executed by move instruction of CPU.

After setting mode of operation to control register, when move instruction of CPU is executed LCDC outputs chip select signal to LCD driver connected to the outside from control pin (LCP0 etc.). Therefore control of data transmission numbers corresponding to LCD size is controlled by instruction of CPU. There are 2 kinds of address of LCD driver in this case, and which is chosen determines by LCDCTL <MMULCD> register.

It corresponds to LCD driver which has every 1 byte of instruction register and display data register in LCD driver at the time of <MMULCD> ="0." Please make the transmission place address at this time into either of 1FE0H-1FE7H. **(SEQUENTIAL ACCESS TYPE)**

LCDC0L/LCDC0H/LCDC1L/LCDC1H/LCDC2L/LCDC2H/LCDR0L/LCDR0H register

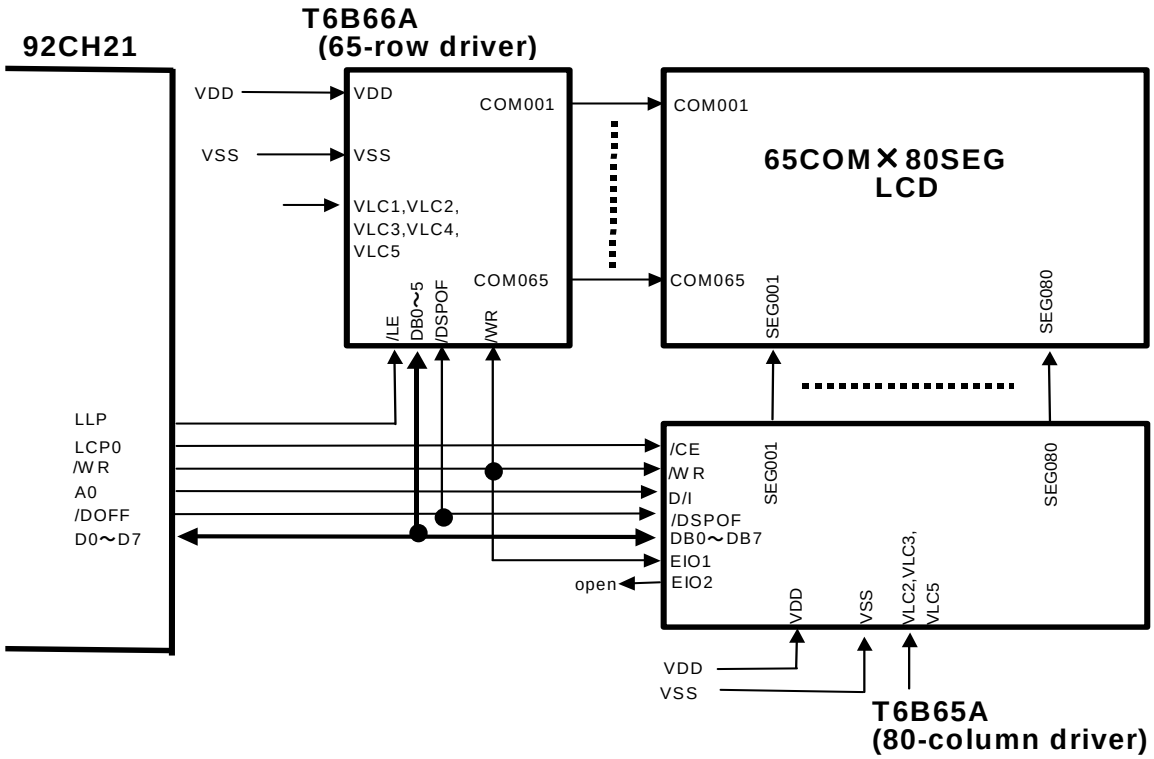
	7	6	5	4	3	2	1	0
bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
Read/Write	Depend on External LCDD specification							
After Reset	Depend on External LCDD specification							
Function	Depend on External LCDD specification							



Note1. This waveform is the case of 3-state access.
Note2. Note the different rising timing for D1BSCP etc.

Example of access timing for RAM built-in type LCD driver (Wait=0)

3.14.6.4 Example to connect built-in RAM LCDD



(Note) Other circuit is necessary for LCD drive power supply for LCD driver display.

Interface example for built-in RAM and sequential access type LCD DRIVER

3.14.6.5 Example for programming

Setting example : In case of use 80SEG X 65COM LCD driver.

Assign external column driver to LCDC1 and row driver to LCDC4.

This example used LD instruction in setting of instruction and used burst function of micro DMA by soft start in setting of display data.

In case of store 650 bytes transfer data to LCD driver.

```
; ***** Setting external terminal *****
        ld      (xxxx),xxh      ; /CE for ,
                                ; /LE for LCDL1,
; ***** Setting for LCDC *****
        ld      (lcdmode0),00h  ; Select RAM mode
        ld      (lcdctl0),00h   ; MMULCD=0 (sequential access mode)

; ***** Setting for mode of LCDC0/LCDR0 *****
        ld      (lcdc1l),xx     ; Setting instruction for LCDC1
        ld      (lcdc4l),xx     ; Setting instruction for LCDC4

;***** Setting for micro DMA and INTTC(ch0) *****
        ld      a,08h           ; Source address INC mode
        ldc     dmam0,a         ;
        ld      wa,650          ; count=650
        ldc     dmac0,wa        ;
        ld      xwa,400000h     ; Source address=400000H
        ldc     dmas0,xwa       ;
        ld      xwa,1fe1h       ;Destination address=1FE1H(LCDC0H)
        ldc     dmad0,xwa       ;
        ld      (intetc01),06H  ; INTTC0  level=6
        ei      6               ;
        ld      (dmab),01h      ; Burst  mode
        ld      (dmar),01h      ; Soft start
```

3.15 Melody / Alarm generator(MLD)

TMP92CH21 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.

- 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.

- Alarm generator

The Alarm function generates eight kinds of alarm waveform having a modulation frequency (4096Hz) determined by the low-speed clock (32.768KHz). 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 kind of fixed cycles (1Hz, 2Hz, 64Hz, 512Hz, 8KHz) INTERRUPT be generated by using a counter which is used for alarm generator.

This section is constituted as follows.

3.15.1 Block diagram
3.15.2 Control registers
3.15.3 Actuating exposition
3.15.3.1 melody generator
3.15.3.2 Alarm generator

3.15.1 Block Diagram

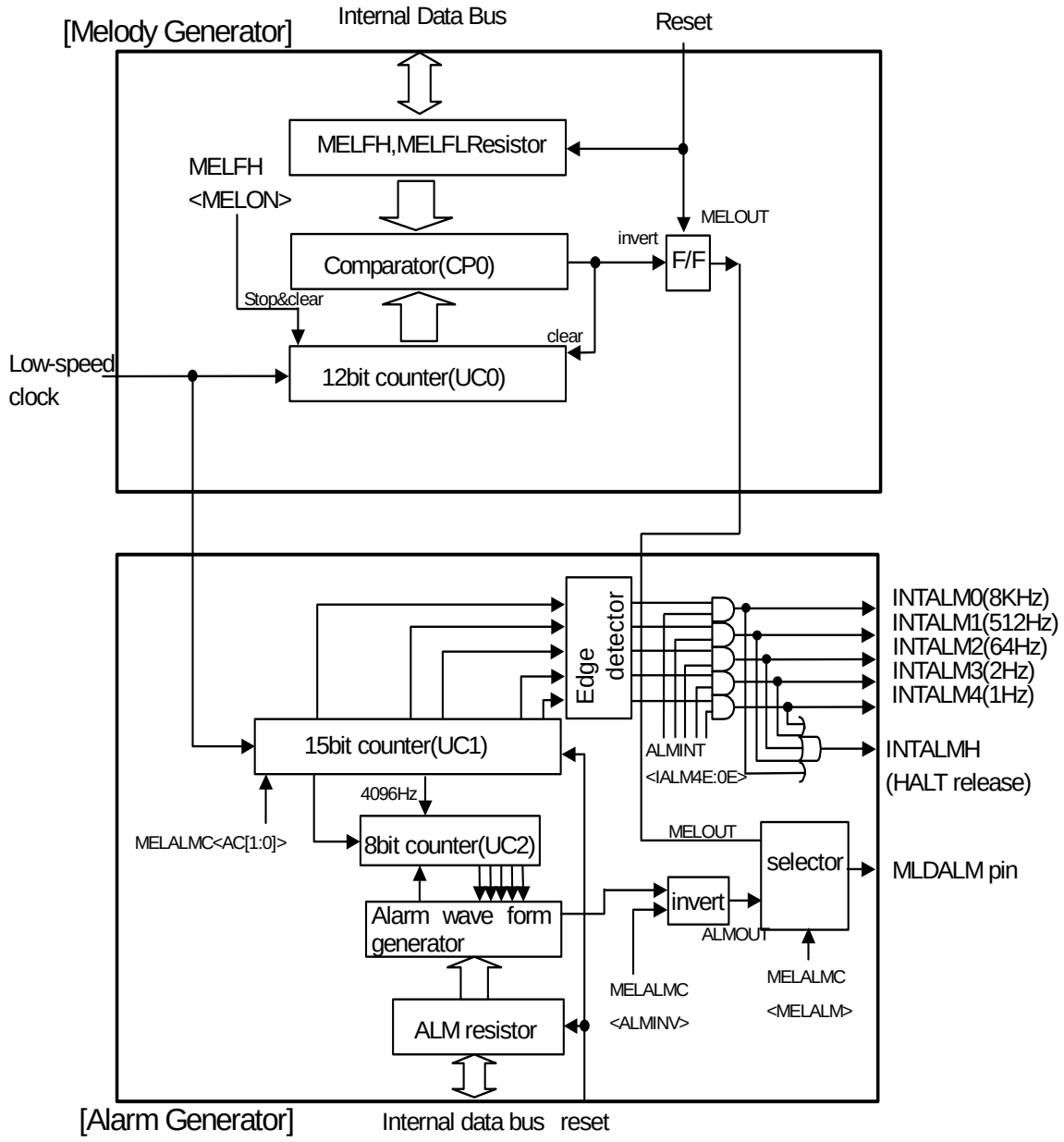


Figure 3.15.1 MLD Block Diagram

3.15.2 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							
Function	Setting alarm pattern							

MLDALMC 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
After reset	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	Write “0”				Output Waveform select 0: Alarm 1: Melody

(Note1) MELALMEC<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							
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			
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				
Function			Write “0”	1:Interrupt enable for INTALM4～INTALM0				

3.15.3 Operational Description

3.15.3.1 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×N+4)

setting value for melody N = (16384 / f_{MLD}) – 2

(notice: 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

3.15.3.2 Alarm generator

The Alarm function generates eight kinds of alarm waveform having a modulation frequency 4096Hz determined by the low-speed clock (32.768KHz). 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, 8KHz) 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 alarm pattern has to be set on 8-bit register of ALM. Finally “10” be set on MLDALMC<AC1: 0> register, and <ALMINV> be set as invert. By setting these values, counter start to generate alarm waveform.

Followings are example program, setting value of alarm pattern and waveform of each setting value.

(Setting value of alarm pattern)

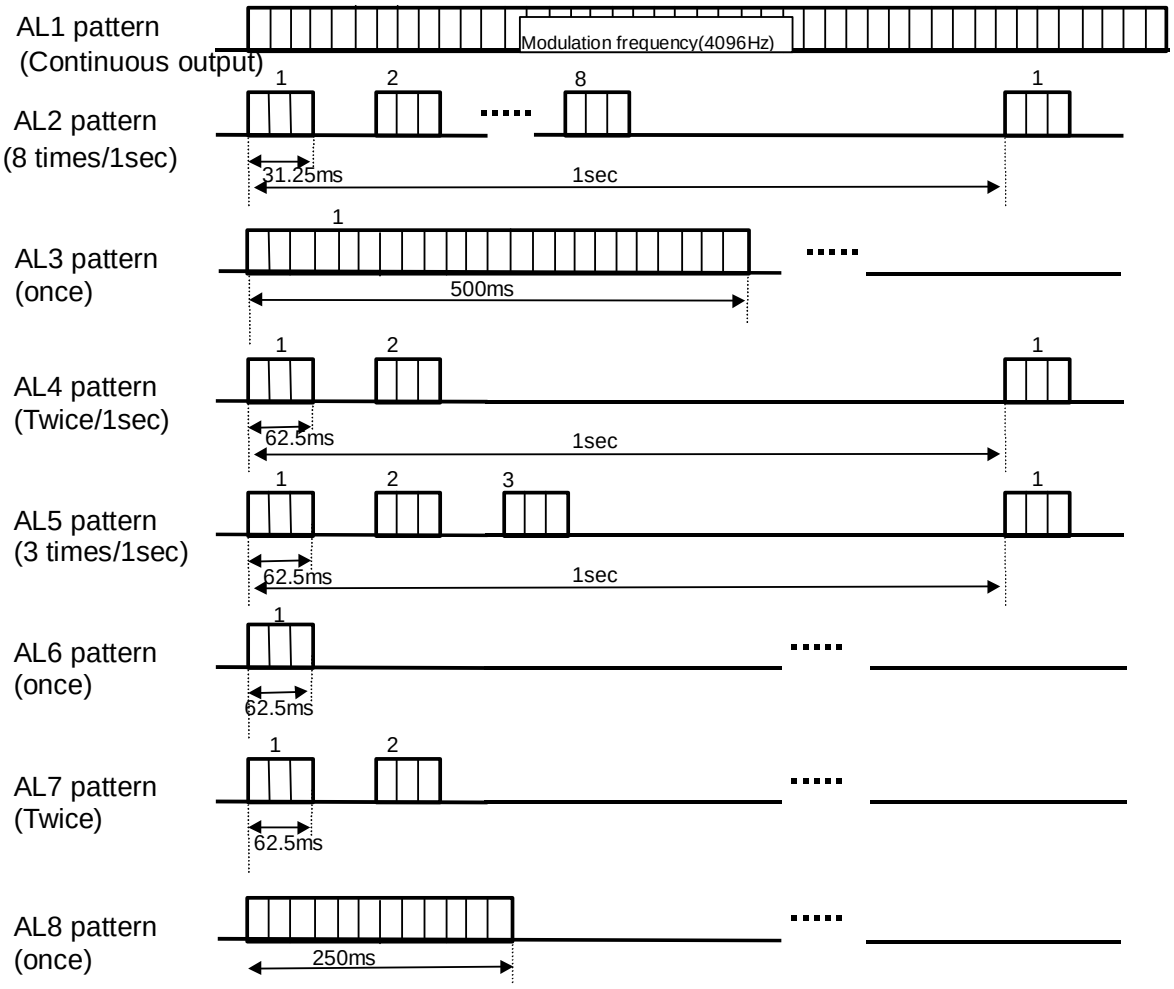
Setting value for ALM register	Alarm waveform
00H	“0” fixed
01H	AL1 pattern
02H	AL2 pattern
04H	AL3 pattern
08H	AL4 pattern
10H	AL5 pattern
20H	AL6pattern
40H	AL7 pattern
80H	AL8 pattern
Other	Undefined (do not set)

(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)



3.16 SDRAM Controller (SDRAMC)

TMP92CH21 includes SDRAM controller which supports SDRAM access by CPU/LCDC. The features are as follows.

(1) Support SDRAM

- Data Rate Type : Only SDR(Single Data Rate) Type
- Bulk of Memory : 16 / 64 / 128 / 256 / 512Mbit
- Number of Bank : 2 / 4 Banks
- Width of Data Bus : 16 / 32 @CPU Access
  16 / 32 @ LCDC Access
- Read Burst Length : 1Word / Full Page
- Write Mode : Single / Burst

(2) Automatic Initialize Function

- All Bank Pre-charge Command Generate
- 8 times Auto Refresh
- Mode Register Set Generate

(3) Access Mode

	CPU Access	LCDC Access
Read Burst Length	1Word / Full Page	Full Page
Addressing Mode	Sequential	Sequential
CAS Latency(clock)	2	2
Write Mode	Single / Burst	-

(4) Access Cycle

- CPU Access (Read/Write)
  Read Cycle : 1Word- 4state / Full Page - 1state
  Write Cycle : Single - 3state / Burst - 1state
  Access Data Width : 8bit/16bit/32bit

- LCDC Burst Access (Read only)
  Read Cycle : 1state(50ns @fsys=20MHz)
  Over head : 4state(200ns @fsys=20MHz)
  Access Data Width : 16bit/32bit

(5) Refresh Cycle Auto Generate

- Auto Refresh is generated during another area access.
- Refresh interval is programmable.
- Self Refresh is supported (mode1 and mode2)

(Notes)

- Display data has to set from the head of each page.
- Condition of SDRAM's area is set by CS1 or CS2 setting of MEMORY CONTROLLER.

3.16.1 Control Registers

Figure 3.16.1(1) shows the SDRAMC control registers. Setting these registers controls the operation of SDRAMC.

SDRAM Access Control Register 1

SDACR1 (0250H)

	7	6	5	4	3	2	1	0
Bit symbol		SASFR	SMRD	SWRC	SBST	SWMOD	SRBL	SMAC
Read/Write	R/W							
After Reset		0	0	0	0	1	0	0
Function	Always Write to "0"	Selecting Self Refresh Mode (note1) 0: mode1 1: mode2	Mode Register Set Delay Time 0: 1clock 1: 2clock	Write Recovery Time 0: 1clock 1: 2clock	Burst Stop Command (note3) 0:Precharge All 1:Burst Stop	Selecting Write Mode (note2) 0: Burst 1: Single	Selecting Read Burst Length (note2) 0: 1word 1: Full Page	SDRAM Controller 0:Disable 1:Enable

(note1) Issue Self Refresh Entry-command after setting <SASFR> to "1". And set this bit when SDRAM is not self reflesh-state.

(note2) Issue Mode Register Set-command after changing <SWMOD> or <SRBL>.

(note3) For the 1st-cut, write "1" to SDACR1<SBST> in case of using SDRAM as display-memory for LCD Controller.

SDRAM Access Control Register2

SDACR2 (0251H)

	7	6	5	4	3	2	1	0
Bit symbol				SBS	SDRS1	SDRS0	SMUXW1	SMUXW0
Read/Write	R/W							
After Reset				0	0	0	0	0
Function				Number of Banks 0: 2Banks 1: 4Banks	Selecting ROW address Size 00: 2048rows (11bits) 01: 4096rows (12bits) 10: 8192rows (13bits) 11: reserved		Selecting Address Multiplex Type 00: TypeA (A9-) 01: TypeB (A10-) 10: TypeC (A11-) 11: reserved	

SDRAM Refresh Control Register

SDRCR (0252H)

	7	6	5	4	3	2	1	0
Bit symbol					SRS2	SRS1	SRS0	SRC
Read/Write					R/W			
After Reset					0	0	0	0
Function					Refresh interval 000: 47state 001: 78state 010: 97state 011:124state 100:156state 101:195state 110:249state 111:312state			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						Issuing Command (note2) (note3) 000: Not Issue 001: Initialize Sequence a. Precharge All Banks b. 8 Times Auto Refresh c. Mode Register Set 100: Mode Register Set 101: Self Refresh Entry 110: Self Refresh Exit other: Reserved		

(note2) <SCMM2-0> is cleared to “000” after a command is issued. But Self Refresh Entry-command is not cleared and it can be cleared by only Self Refresh Exit-command.

(note3) Write command except Self Refresh Exit –command after checking whether they are “000”.

Figure 3.16.1(1) SDRAMC control registers

3.16.2 Operation description

(1) Memory access control

SDRAM controller is enabled when SDACR1<SMAC>=1. And then SDRAM control Signals (SDCS, SDRAS, SDCAS, SDWE, SDLLDQM, SDLUDQM, SDULDQM, SDUUDQM, SDCLK and SDCKE) are outputted during the time CPU or LCDC accesses CS1 or CS2 area.

In the access cycle, Address multiplex outputs row/column address through A0 to A15 pin. And multiplex width is decided by setting SDACR2<SMUXW0 to SMUXW1> of use memory size. The relation between multiplex width and memory size is shown in Table 3.16.2(4).

Table 3.16.2(1) Address multiplex

92CH21 PIN Name	Address of SDRAM accessing cycle				
	Row address			Column address	
	TypeA <SMUXW>"00"	TypeB <SMUXW>"01"	TypeC <SMUXW>"10"	16Bit data bus width B1CSH<BnBUS>="01"	32Bit data bus width B1CSH<BnBUS>="10"
A0	A9	A10	A11	A1	A2
A1	A10	A11	A12	A2	A3
A2	A11	A12	A13	A3	A4
A3	A12	A13	A14	A4	A5
A4	A13	A14	A15	A5	A6
A5	A14	A15	A16	A6	A7
A6	A15	A16	A17	A7	A8
A7	A16	A17	A18	A8	A9
A8	A17	A18	A19	A9	A10
A9	A18	A19	A20	A10	A11
A10	A19	A20	A21	AP	AP
A11	A20	A21	A22	Row address	
A12	A21	A22	A23		
A13	A22	A23	EA24		
A14	A23	EA24	EA25		
A15	EA24	EA15	EA26		

SDRAM read access by CPU can select 1wird or full page mode, And access by LCDC is fixed the full page burst mode. Write access by CPU can select single or burst mode. DRAM access cycle is shown in Fig. 3.16.2 (1),(2).

SDRAM access cycle number is not depend on <B1CSL> and <B2CSL> register setting.
In the full page burst read cycle, a mode setup and a pre-charge cycle are automatically inserted in a cycle front and back.

(2) Instruction executing on SDRAM

CPU can execute instructions on SDRAM. However, Below function is not operated.

a) Executing HALT instruction

b) Performing self-refresh entry and exit command

When the above-mentioned is operated, it is necessary to execute it by another memory such as built-in RAM.

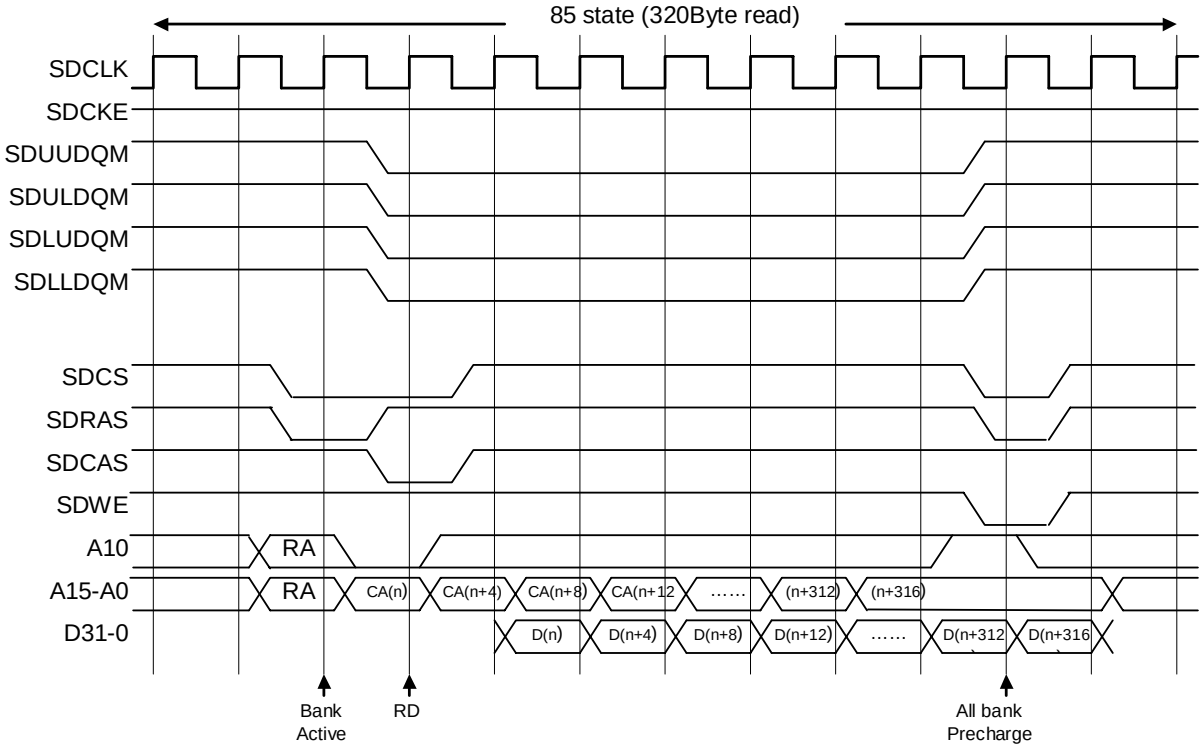


Figure 3.16.2(1) Timing of burst read cycle

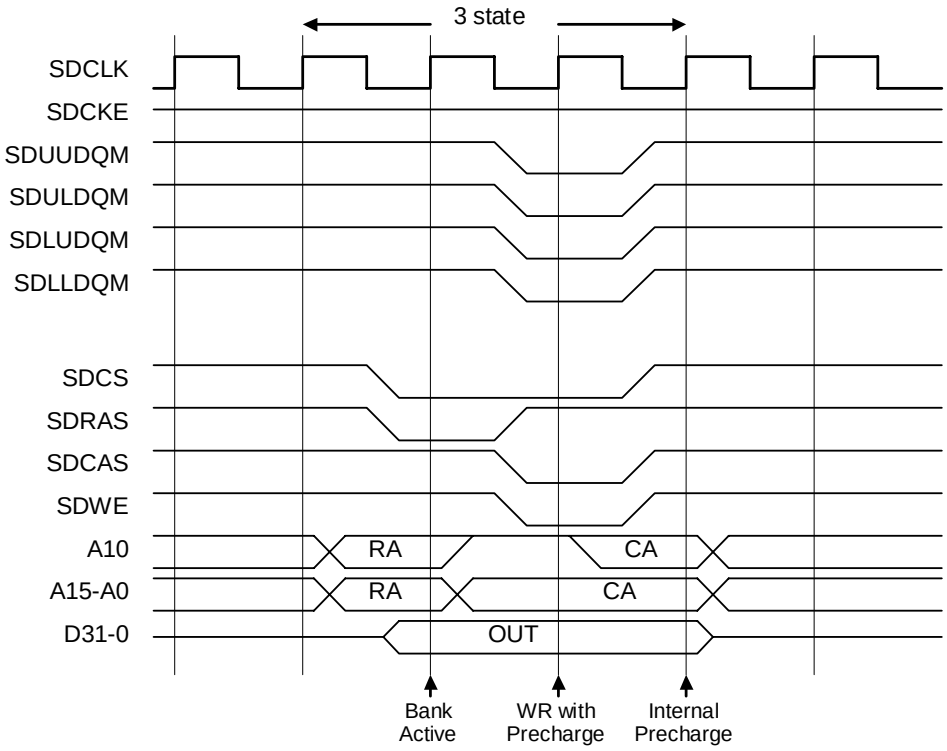


Figure 3.16.2(2) Timing of CPU write cycle (Structure of data bus: 32bit×1, Operand size: 4byte, Address: 4n+0)

(3) Refresh Control

TMP92CH21 supports two Refresh commands of Auto-Refresh and Self-Refresh.

(a) Auto-Refresh

The Auto-Refresh command is automatically generated at intervals set by SDRCR<SRS0-2> by making it to SDRCR<SRFC> to "1". The generation interval can be set between 47-312states (2.4us-15.6us at fsys=20MHz). CPU operation (instruction fetch and execution) stops while performing the Auto-Refresh command. The Auto-Refresh cycle is shown in Fig. 3.16.2(3) and the Auto-Refresh generation interval is shown in Table 3.16.2(2). Auto-Refresh can be used only with CPU Operate or IDLE2.

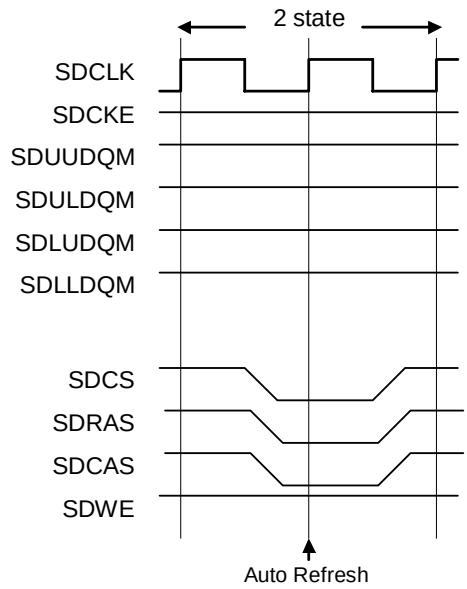


Figure 3.16.2(3) Timing of Auto refresh cycle

SDRCR<SRS0-2>			Insertion interval (state)	Frequency (system clock)					
SRS2	SRS1	SRS0		6MHz	10MHz	12.5MHz	15MHz	17.5MHz	20MHz
0	0	0	47	7.8	4.7	3.8	3.1	2.7	2.4
0	0	1	78	13.0	7.8	6.2	5.2	4.5	3.9
0	1	0	97	16.2	9.7	7.8	6.5	5.5	4.9
0	1	1	124	20.7	12.4	9.9	8.3	7.1	6.2
1	0	0	156	26.0	15.6	12.5	10.4	8.9	7.8
1	0	1	195	32.5	19.5	15.6	13.0	11.1	9.8
1	1	0	247	41.2	24.7	19.8	16.5	14.1	12.4
1	1	1	312	52.0	31.2	25.0	20.8	17.8	15.6

(Unit: us)

Table 3.16.2(2) Refresh cycle insertion interval

(b) Self-Refresh

The Self-Refresh command is generated by making it to SDCMM<SCMM0 to SCMM2> to "101". The Self-Refresh cycle is shown in Fig. 3.16.2(4). During Self-Refresh, Refresh is performed inside SDRAM (an Auto-refresh command is not needed).
The Auto-Refresh command is automatically executed once when Self-Refresh is released, and Refresh is done according to the setting of the Auto-Refresh command after that.

(Note: When standby mode is cancelled by a reset, the I/O registers are initialized, therefore, auto-refresh is not performed.) .
(Note: During Self-Refresh cannot be accessed to SDRAM.)
(Note: After the Self-Refresh Entry command, CPU is shifted to IDLE1 or the STOP mode by the HALT instruction the setup time is necessary. Between instruction which sets SDCMM<SCMM0-2> to "101" with HALT instruction, execute 10 bytes or more of NOP, or other instructions.

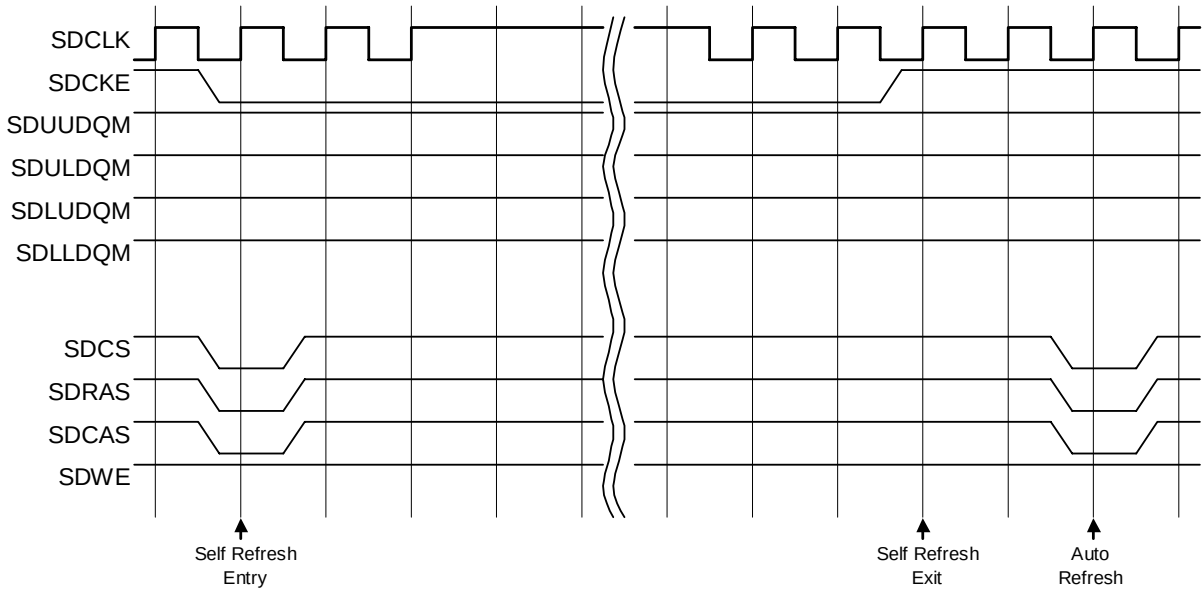


Figure 3.16.2(4) Timing of Self refresh cycle

TMP92CH21 has two modes by the difference of the generation condition of Self-Refresh Entry and the Exit command. The mode is set by SDACR1<SASFR >. Two mode difference is shown in Table3.16.2(3).
Mode2 automatically generates Self-Refresh Entry and the Exit command synchronizing with the accessing of the SDRAM. When it wants to suppress current consumption in SDRAM, an effective mode. However, Over head of number state is needed from the access generation to beginning the access actually to execute the Self-Refresh Exit command. It is necessary to note it because the performance decreases.

	Mode1	Mode2
Self-Refresh Entry Condition	When SDCMM<SCMM0-SCMM2> is set to "101"	When SDCMM<SCMM0-SCMM2> is set to "101", and there is no SDRAM access
Self-Refresh Exit Condition	1) When SDCMM<SCMM0-SCMM2> is set to "101" 2) When HALT is released	1) When SDCMM<SCMM0-SCMM2> Is set to "101" 2) When Accessing to SDRAM note) When the SDRAM access ends, the Self-Refresh Entry command is automatically generated.
Use limitation		Only CPU Operate and the IDLE2 mode can be used.

Table 3.16.2(3) Difference Mode1 and Mode2

(4) SDRAM Initialize

TMP92CH21 can generate the following SDRAM initialize routine after injection power-supply to SDRAM. The cycle is shown in Fig. 3.16.2 (5).

1. Pre-charge of all banks
2. The auto-refresh cycle of 8 cycles
3. The initial configuration to a mode register

The above cycle is generated by setting SDCMM<SCMM0 to SCMM2> to “001”.
While performing this cycle, operation (an instruction fetch, command execution) of CPU is stopped. In addition, before performing an initialization cycle, a port needs to be set as SDRAM control signal and an address signal (A0-A15).

After the initialization cycle is finished, SDCMM< SCMM0 to SCMM2> is set to “000” automatically.

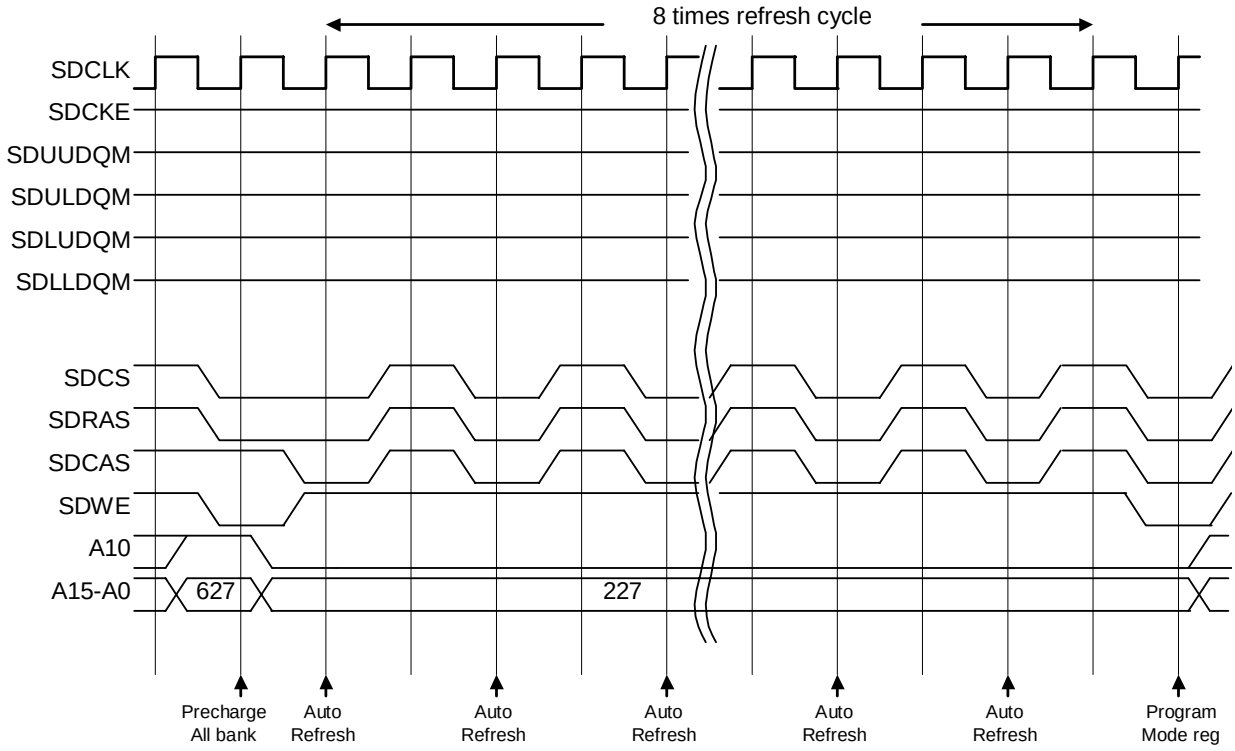


Figure 3.16.2(5) Timing of initialization cycle

(5) Connection Example

The example of connection with SDRAM is shown in table 3.16.2(3) and figure 3.16.2. (9).

Table 3.16.2(4) Connection with SDRAM

92CH21 Pin Name	SDRAM Pin Name												
	Data bus width 16bit					Data bus width 32bit							
	16M	64M	128M	256M	512M	16M ×16bit×2		64M ×16bit×2		128M ×16bit×2		64M ×32bit	128M ×32bit
A0	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0
A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1
A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2
A3	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3
A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4
A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5
A6	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6
A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7
A8	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8
A9	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9
A10	A10	A10	A10	A10	A10	A10	A10	A10	A10	A10	A10	A10	A10
A11	BS	A11	A11	A11	A11	BS	BS	A11	A11	A11	A11	BS0	A11
A12	-	BS0	BS0	A12	A12	-	-	BS0	BS0	BS0	BS0	BS1	BS0
A13	-	BS1	BS1	BS0	BS0	-	-	BS1	BS1	BS1	BS1	-	BS1
A14	-	-	-	BS1	BS1	-	-	-	-	-	-	-	-
A15	-	-	-	-	-	-	-	-	-	-	-	-	-
SDCS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS
SDUUDQM	-	-	-	-	-	UDQM		UDQM		UDQM		DQM3	DQM3
SDULDQM	-	-	-	-	-	LDQM		LDQM		LDQM		DQM2	DQM2
SDLUDQM	UDQM	UDQM	UDQM	UDQM	UDQM		UDQM		UDQM		UDQM	DQM1	DQM1
SDLLDQM	LDQM	LDQM	LDQM	LDQM	LDQM		LDQM		LDQM		LDQM	DQM0	DQM0
SDRAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS
SDCAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS
SDWE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE
SDCKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE	CKE
SDCLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK
SDACR <SMUXW>	00: TypeA	00: TypeA	01: TypeB	01: TypeB	10: TypeC	01: TypeB		01: TypeB		10: TypeC		01: TypeB	01: TypeB

(An): Row Address

: Command address pin of SDRAM

Fig. 3.16.2 (6) Connection with SDRAM(1MWord×16bit)

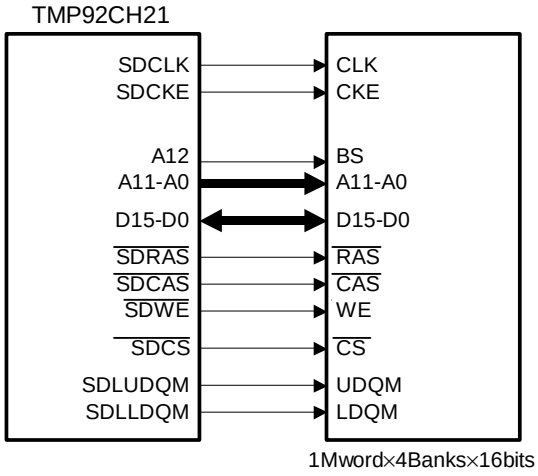


Fig. 3.16.2 (7) Connection with SDRAM(1MWord×16bit×2)

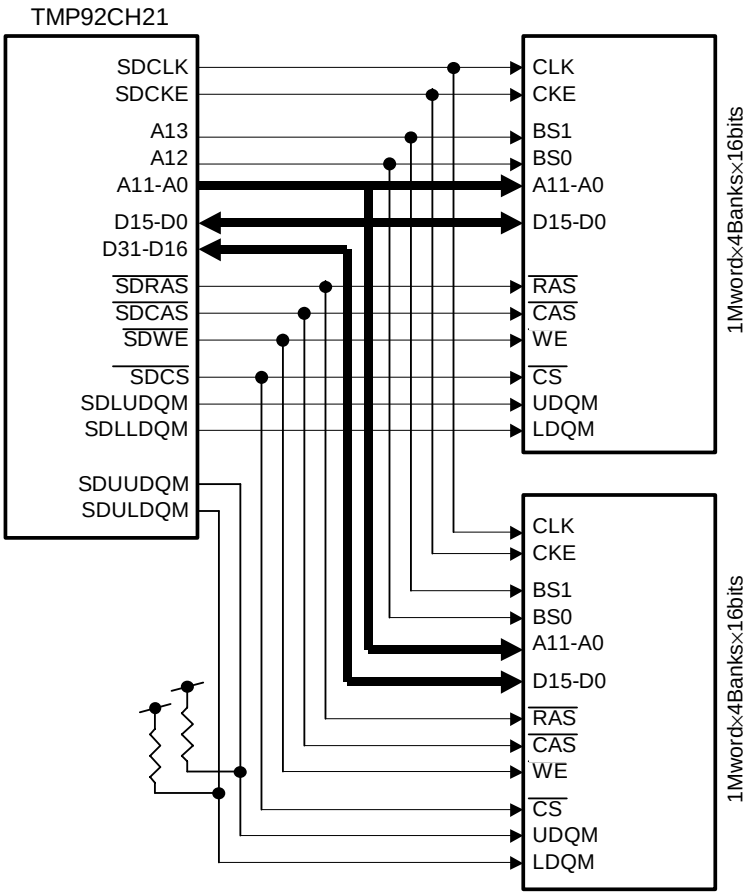
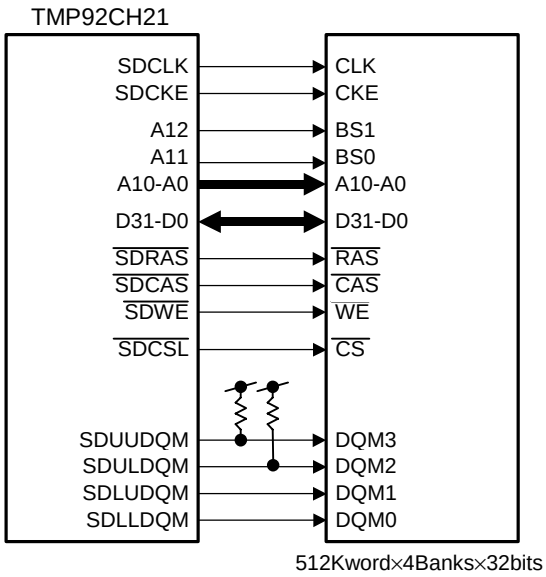


Fig. 3.16.2 (8) Connection with SDRAM(512KWord×32bit)



3.17 NAND Flash Controller

3.17.1 Characteristics

The NAND Flash Controller (NDFC) generates the control signals required to interface with the NAND Flash. It has also the ECC calculating circuits.

This product has 2-v. External pins except Chip-Enable are multiplexed . They are selected by NDCR<CHSEL> register.

Both channels operate in the same manner, and thus only the operation of Channel-0 will be explained below.

The NAND Flash Controller has the following characteristics.

- Controlled NAND Flash interface by setting Registers.
- On-chip ECC calculating circuits

(Note1) The /WP(Write Protect)-pin of NAND-Flash is not supported by this product.
If necessary, prepare it on external-circuit.

(Note2) Though 2-channel NDFC are built-in, don't access both NAND-Flash at the same time.

3.17.2 Block Diagram

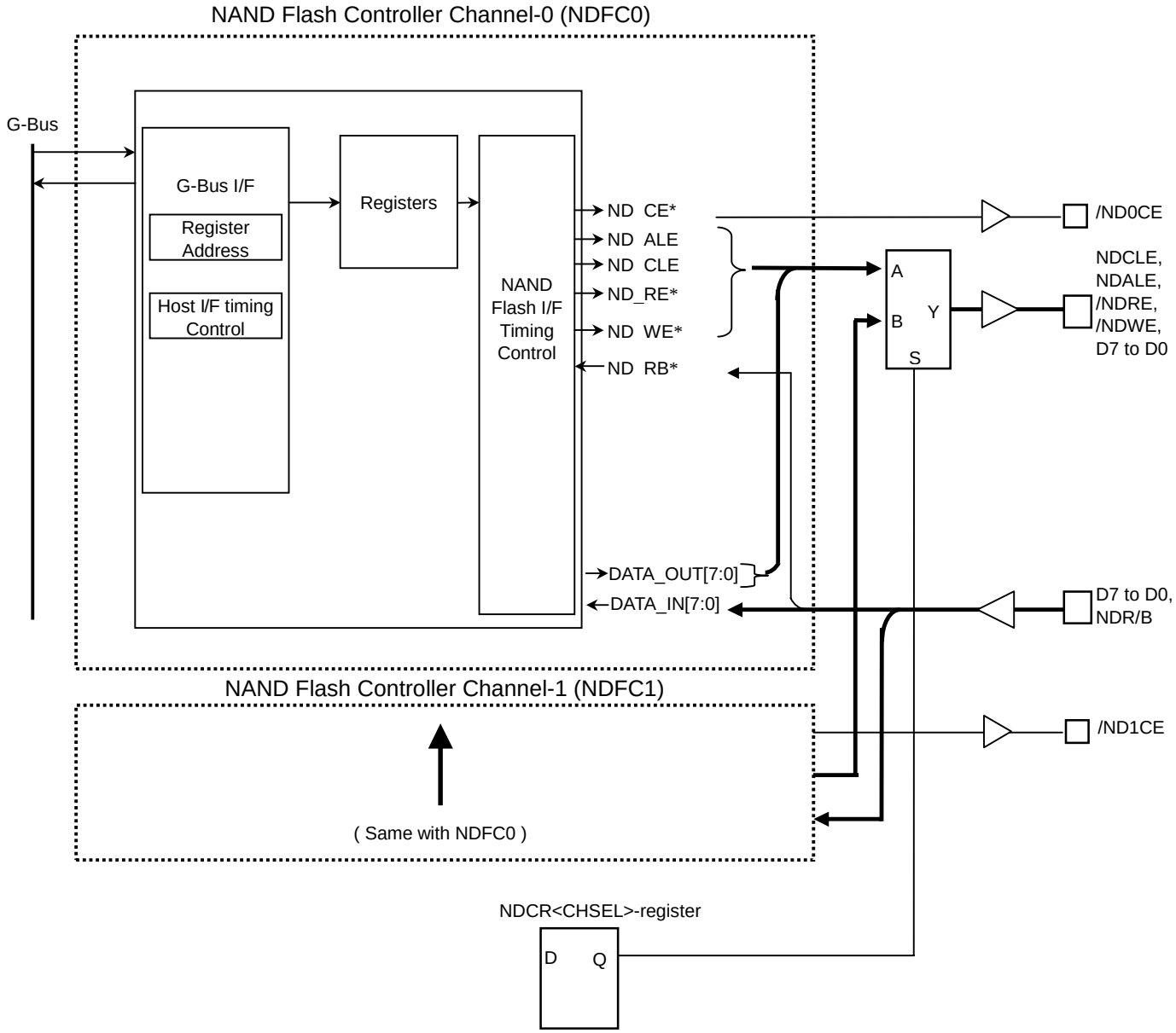


Figure 3.17.2.1 NAND Flash Controller Block Diagram

3.17.3 Detailed Explanation

3.17.3.1 Access to NAND Flash

The NDFC supports the interface between the NAND Flash using register indirect sequence. It has the ECC calculating circuits. Please see 3.17.3.2 in detail of the ECC. This section describes the procedure to access to NAND Flash.

Basically, set the command in ND0FMCR at first and then read or write in ND0FDTR. The read cycle for ND0FDTR is finished after the external read cycle for the NAND Flash is finished. Equally, the write cycle for ND0FDTR is finished after the external write cycle for the NAND Flash is finished.

1) Initialize

The initialize sequence is below.

(1) ND0FSPR :Set the Low pulse width.
(2) ND0FIMR :Set 0x81 if need enable interrupt.

2) Write

The write sequence is below.

(1) ND0FMCR :Set 0x70 to reset ECC data.
(2) NDFC doesn't have /WP signal. It must be high using other logic.
(3) Write 512 bytes
- ND0FMCR :Set 0x91 to assert NDCLE signal and do the command mode.
- ND0FDTR :Set 0x80 to write the Serial Data Input command.
- ND0FMCR :Set 0x92 to assert NDALE signal and do the address mode.
- ND0FDTR :Set A[7:0], A[16:9], and A[24:17]. If need, set A[25].
- ND0FMCR :Set 0xb0 to do the data mode.
- ND0FDTR :Write 512 bytes data.

(4) Read ECC data
- ND0FMCR :Set 0xd0 to do the ECC data read mode.
- NDECCRD :Read 6 bytes ECC data.
  - First data: LPR[7:0]
  - Second data: LPR[15:8]
  - Third data: CPR[5:0], 2'b11
  - Fourth data: LPR[23:16]
  - Fifth data: LPR[31:24]
  - Sixth data: CPR[11:6], 2'b11

(5) Write 16 bytes redundant data

- ND0FMCR :Set 0x90 to do the data mode without ECC.
- ND0FDTR :Write 16 bytes redundant data.

  D520: LPR[23:16]

  D521: LPR[31:24]

  D522: CPR[11:6], 2'b11

  D525: LPR[7:0]

  D526: LPR[15:8]

  D527: CPR[5:0], 2'b11

(6) Run Page Program

- ND0FMCR :Set 0x91 to assert NDCLE signal and do the command mode.
- ND0FDTR :Set 0x10 to write the Page Program command.
- ND0FMCR :Set 0x10 to deassert NDALE signal.
- Wait some states (ex. "NOP" * 10)
- ND0FSR :Check BUSY flag. If it's zero, go to the next. If it's one, wait till it will become zero.

(7) Read Status

- ND0FMCR :Set 0x11 to assert NDCLE signal and do the command mode.
- ND0FDTR :Set 0x70 to write the Read Status command.
- ND0FMCR :Set 0x10 to deassert NDCLE signal.
- ND0FDTR :Read the Status data from the NAND Flash.

(8) Continue from 1 to 7 for all other pages if need.

3) Read

The read sequence is below.

(1) ND0FMCR :Set 0x70 to reset ECC data.

(2) Read 512 bytes

- ND0FMCR :Set 0x11 to assert NDCLE signal and do the command mode.
- ND0FDTR :Set 0x00 to write the Read command.
- ND0FMCR :Set 0x12 to assert NDALE signal and do the address mode.
- ND0FDTR :Set A[7:0], A[16:9], and A[24:17]. If need, set A[25].
- ND0FMCR :Set 0x10 to deassert NDALE signal.
- Wait some states (ex. “NOP” * 10)
- ND0FSR :Check BUSY flag. If it’s zero, go to the next. If it’s one, wait till it will become zero.
- ND0FMCR :Set 0x30 to do the data mode with ECC calculation.
- ND0FDTR :Read 512 bytes data.
- ND0FMCR :Set 0x10 to do the data mode without ECC calculation.
- ND0FDTR :Read 16 bytes redundant data.

(3) Read ECC data

- ND0FMCR :Set 0x50 to do the ECC data read mode.
- NDECCRD :Read 6 bytes ECC data.

First data:	LPR[7:0]
Second data:	LPR[15:8]
Third data:	CPR[5:0], 2’b11
Fourth data:	LPR[23:16]
Fifth data:	LPR[31:24]
Sixth data:	CPR[11:6], 2’b11

(4) Compare ECC data and run the error routine if error occurs by Software.

(5) Read other pages

- ND0FMCR :Set 0x10.
- ND0FSR :Check BUSY flag. If it’s zero, go to the next. If it’s one, wait till it will become zero.
- Continue from 1 to 4 but 2-1 to 2-5 can be skipped when Sequential Read.

4) ID Read

The ID read sequence is below.

(1) ND0FMCR :Set 0x11 to assert NDCLE signal and do the command mode.
(2) ND0FDTR :Set 0x90 to write the ID Read command.
(3) ND0FMCR :Set 0x12 to assert NDALE signal and do the address mode.
(4) ND0FDTR :Set 0x00.
(5) ND0FMCR :Set 0x10 to do the data mode without ECC calculation.
(6) ND0FDTR :Read Maker Code.
(7) ND0FDTR :Read Device Code.

3.17.3.2 ECC Control

NDFC has the ECC calculating circuits. The circuits are controlled by ND0FMCR. The Software compares the ECC data and checks if error or not.

The calculated ECC data can be read from NDECCRD register when ND0FMCR is 0xD0 (Write mode) or 0x50 (Read mode). It is six bytes and six read operations for NDECCRD are needed. The order of the data is following as.

First data: LPR[7:0]
Second data: LPR[15:8]
Third data: CPR[5:0], 2'b11
Fourth data: LPR[23:16]
Fifth data: LPR[31:24]
Sixth data: CPR[11:6], 2'b11

3.17.4 Registers

Table 3.17.4(1)-a NAND Flash Control Registers for Channel-0

Address	Register	Register Name
1D00H (1D00H-1EFFH)	ND0FDTR	NAND Flash Data transfer Register
1CB0H (1CB0H-1CB5H)	ND0ECCRD	NAND Flash ECC-code read Register
1CC4H	ND0FMCR	NAND Flash Mode Control Register
1CC8H	ND0FSR	NAND Flash Status Register
1CCCH	ND0FISR	NAND Flash Interrupt Status Register
1CD0H	ND0FIMR	NAND Flash Interrupt Mask Register
1CD4H	ND0FSPR	NAND Flash Strobe pulse width Register
1CD8H	ND0FRSTR	NAND Flash Reset Register

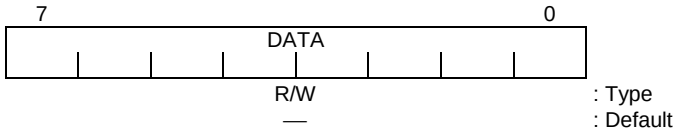
Table 3.17.4(1)-b NAND Flash Control Registers for Channel-1

Address	Register	Register Name
1D00H (1D00H-1EFFH)	ND1FDTR	NAND Flash Data transfer Register
1CB0H (1CB0H-1CB5H)	ND1ECCRD	NAND Flash ECC-code read Register
1CE4H	ND1FMCR	NAND Flash Mode Control Register
1CE8H	ND1FSR	NAND Flash Status Register
1CECH	ND1FISR	NAND Flash Interrupt Status Register
1CF0H	ND1FIMR	NAND Flash Interrupt Mask Register
1CF4H	ND1FSPR	NAND Flash Strobe pulse width Register
1CF8H	ND1FRSTR	NAND Flash Reset Register

Table 3.17.4(1)-c NAND Flash Control Registers

Address	Register	Register Name
01C0H	NDCR	NAND Flash Control Register

3.17.4.1 NAND Flash Data Transfer Register (ND0FDTR, ND1FDTR)



Bit(s)	Mnemonic	Field Name	Description
7 : 0	DATA	DATA	NAND Flash data. Read: Be able to read the data that was read from the NAND Flash. Write: The data is written to the NAND Flash.

(Note1) This register has an 512-address window from 1D00H to 1EFFH. Because a page-size of NAND-Flash is 256 or 512Bytes.
When CPU reads(or writes) from (or to) NAND-Flash and if the block-transfer instruction (“LDIR” instruction) is used, there is a following restriction for 900/H1 CPU.

[Restriction for using the block-transfer instruction]

1) The source address for “LDIR” instruction should be set to (1F00H – read(or write) byte-number) .

Ex.-1) In case of 512-byte read

```
ld    bc,512          ; 512Byte
ld    xix,2000H       ; dst = 2000H
ld    xiy,1D00H       ; src = (1F00H – 512) = 1D00H
ldir  (xix+),(xiy+)   ; block-transfer instruction
```

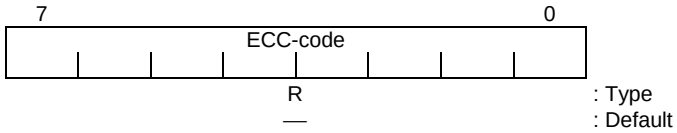
Ex.-2) In case of 16-byte read

```
ld    bc,16           ; 16Byte
ld    xix,2000H       ; dst = 2000H
ld    xiy,1EF0H       ; src = (1F00H – 16) = 1EF0H
ldir  (xix+),(xiy+)   ; block-transfer instruction
```

(Note2) Both ND0FDTR and ND1FDTR are assigned to same address. It depend on NDCR<CHSEL> register whether which channel is accessed.

Figure 3.17.4(1) NAND Flash Data Transfer Register (ND0FDTR, ND1FDTR)

3.17.4.2 NAND Flash ECC-code Read Register (ND0ECCRD, ND1ECCRD)



Bit(s)	Mnemonic	Field Name	Description
7 : 0	ECC-code	ECC-code	NAND Flash calculated ECC data.

(Note1) Both ND0ECCRD and ND1ECCRD are assigned to same address. It depend on NDCR<CHSEL> register whether which channel is accessed.

Figure 3.17.4(2) NAND Flash ECC-code Read Register (ND0ECCRD, ND1ECCRD)

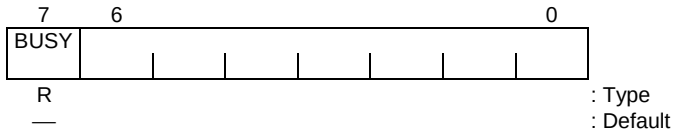
3.17.4.3 NAND Flash Mode Control Register (ND0FMCR, ND1FMCR)

7	6	5	4	3	2	1	0	
WE	ECC1	ECC0	CE	PCNT1	PCNT0	ALE	CLE	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	: Type
0	0	0	0	0	0	0	0	: Default

Bits	Mnemonic	Field Name	Description
7	WE	Write Enable	Write Enable (Default: 0) This bit enables the data write operation. When you write the data to the NAND flash, this bit must be set one. 0: Inhibit write operation 1: Enable write operation
6	ECC1	ECC Control	ECC Control (Default: 00) These bits control the ECC calculating circuits with <CE>(bit4)-register. 11 (at <CE>=X) : Reset ECC circuits. 00 (at <CE>=1) : ECC circuits is disable. 01 (at <CE>=1) : ECC circuits is enable. 10 (at <CE>=1) : Read ECC data calculated by NDFC. 10 (at <CE>=0) : Read ID data.
5	ECC0		
4	CE	Chip Enable	Chip Enable (Default: 0) Enable NAND Flash access. This bit must be set one when access to the NAND Flash. 0: Disable(/NDCE is high.) 1: Enable (/NDCE is low.)
3	PCNT1	Power Control	Power Control (Default: 00) 11: Power On 10: Power Off 01: Power On (continue) 00: reserved
2	PCNT0		
1	ALE	Address Latch Enable	Address Latch Enable (Default: 0) This bit specifies the value of NDALE signal. 0: Low 1: High
0	CLE	Command Latch Enable	Command Latch Enable (Default: 0) This bit specifies the value of NDCLE signal. 0: Low 1: High

Figure 3.17.4(3) NAND Flash Mode Control Register (ND0FMCR ,ND1FMCR)

3.17.4.4 NAND Flash Status Register (ND0FSR, ND1FSR)



Bit	Mnemonic	Field Name	Description
7	BUSY	BUSY	BUSY (Default: Undefined) This bit shows the status of NAND flash. 0: Ready 1: Busy
6 : 0	—	—	Reserved

(Note1) Since a noise-filter for some-states are built-in the NDFC, when NDR/B-pin changes, a <BUSY>-flag don't change at the same time. So, some delay time (ex. "NOP"-instruction * 10) are needed before starting to sense this flag.

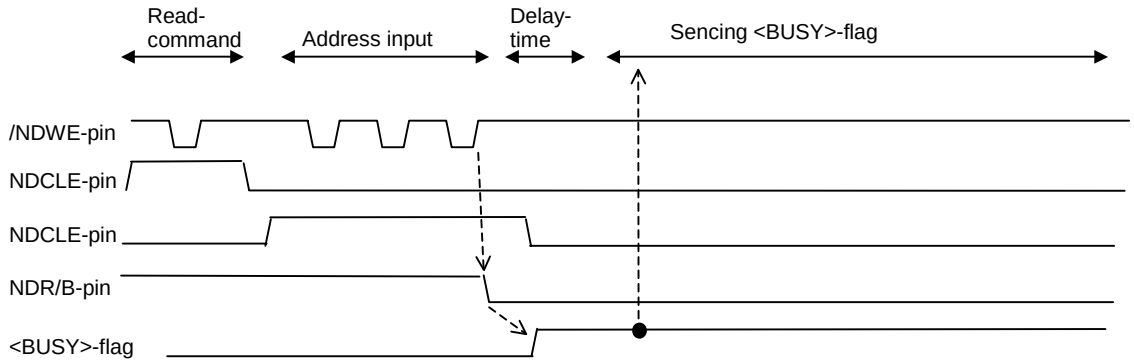
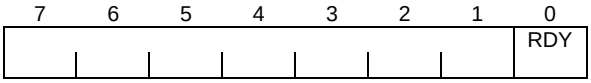


Figure 3.17.4(4) NAND Flash Status Register (ND0FSR ,ND1FSR)

3.17.4.5 NAND Flash Interrupt Status Register (ND0FISR, ND1FISR)

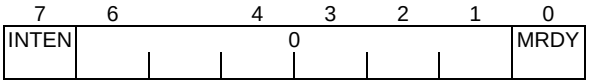


: Type
: Default

Bit	Mnemonic	Field Name	Description
7 : 1	—	—	Reserved
0	RDY	Ready	Ready (Default: 0) This bit is set when NDR/B signal changes from Low(means BUSY) to High(means READY) if NDFIMR<MRDY> is one. Writing “1” clears this bit to zero. Read: 0: None 1: Change NDR/B signal from Busy to Ready. Write: 0: No change 1: Clear to zero

Figure 3.17.4(5) NAND Flash Interrupt Status Register (ND0FISR ,ND1FISR)

3.17.4.6 NAND Flash Interrupt Mask Register (ND0FIMR, ND1FIMR)

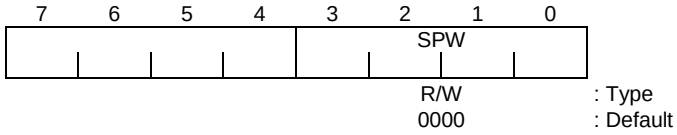


R/W R/W : Type
0 0 : Default

Bit	Mnemonic	Field Name	Description
7	INTEN	Interrupt Enable	Interrupt Enable (Default: 0) When <INTEN> and <MRDY> are set one and NDFISR<RDY> becomes one, the interrupt INTNDFC occurs. 0: Disable 1: Enable
6 : 1	—	—	Reserved
0	MRDY	Mask RDY interrupt	Mask Ready Interrupt (Default: 0) This bit masks the NDFISR<RDY> . If <MRDY> is one, NDFISR<RDY> is set when NDR/B signal changes from Low to High. 0: Disable to set NDFISR<RDY> 1: Enable to set NDFISR<RDY>

Figure 3.17.4(6) NAND Flash Interrupt Mask Register (ND0FIMR ,ND1FIMR)

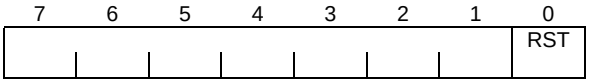
3.17.4.7 NAND Flash Strobe Pulse Width Register (ND0FSPR, ND1FSPR)



Bit	Mnemonic	Field Name	Description
7 : 4	—	—	Reserved
3 : 0	SPW	Strobe Pulse Width	Strobe Pulse Width (Default: 0000) These bits specify the Low pulse width of the /NDRE and /NDWE signals. The low pulse width is the value of this field plus one f_{SYS} clock.

Figure 3.17.4(7) NAND Flash Strobe Pulse Width Register (ND0FSPR ,ND1FSPR)

3.17.4.8 NAND Flash Reset Register (ND0FRSTR ,ND1FRSTR)



R/W : Type
0 : Default

Bit	Mnemonic	Field Name	Description
7 : 1	—	—	Reserved
0	RST	Reset	Reset (Default: 0) Setting this bit reset the NDFC(except NDCR<CHSEL> register). After reset, this bit is cleared automatically. 0: Don't care 1: Reset

(Note1) After writing <RST> register, some waiting- time(about 10-state) are needed to access NDFC.

Figure 3.17.4(8) NAND Flash Reset Register (ND0FRSTR ,ND1FRSTR)

3.17.4.9 NAND Flash Control Register (NDCR)

NDCR (01C0H)

	7	6	5	4	3	2	1	0
bit Symbol	CHSEL							
Read/Write	R/W							
After reset	0							
Function	0:Channel0 1: Channel1							

3.17.5 Timing Diagrams

3.17.5.1 Command and Address Cycle

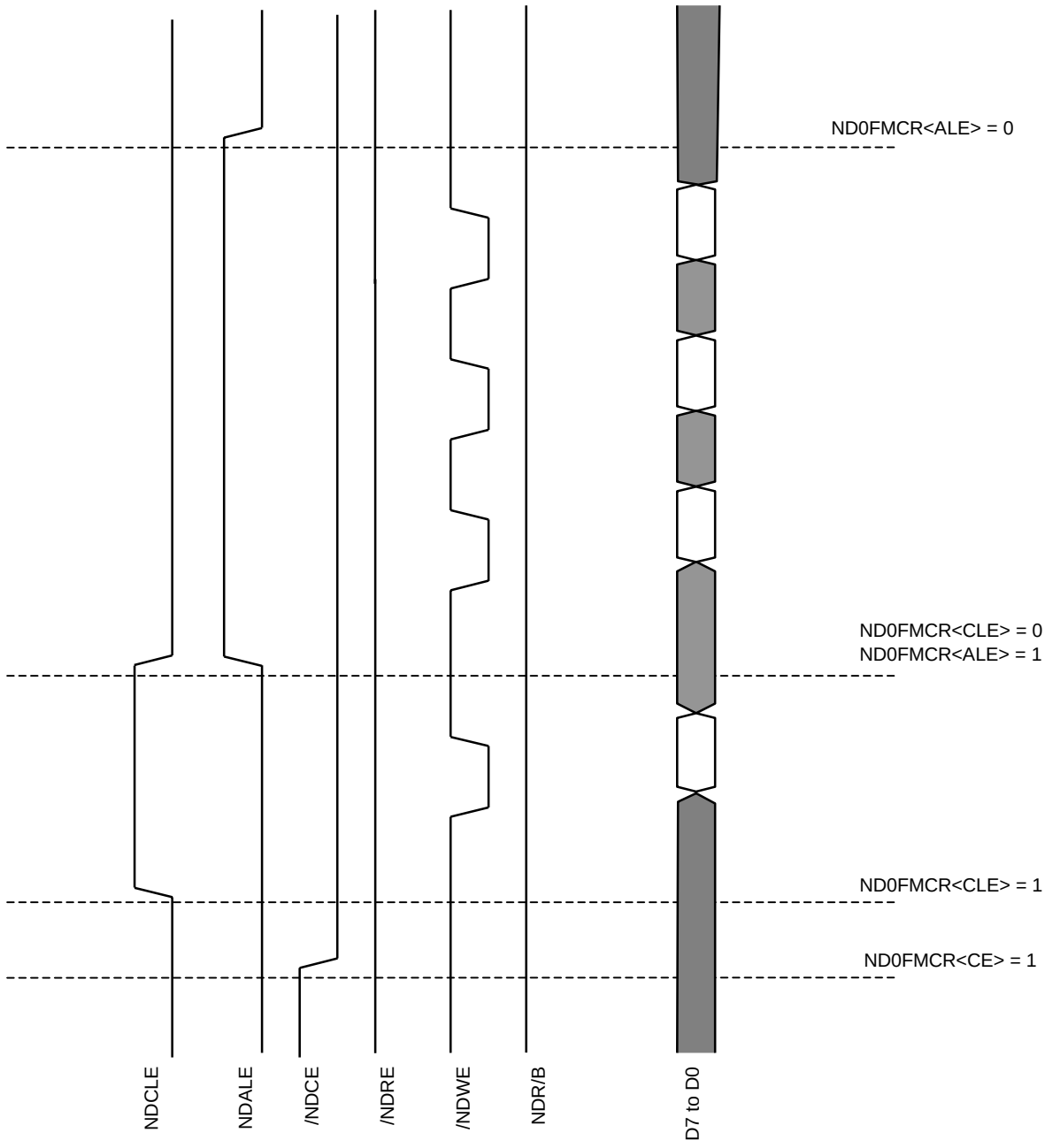


Figure 3.17.5(1) Command and Address Cycle

3.17.5.2 Data Read Cycle

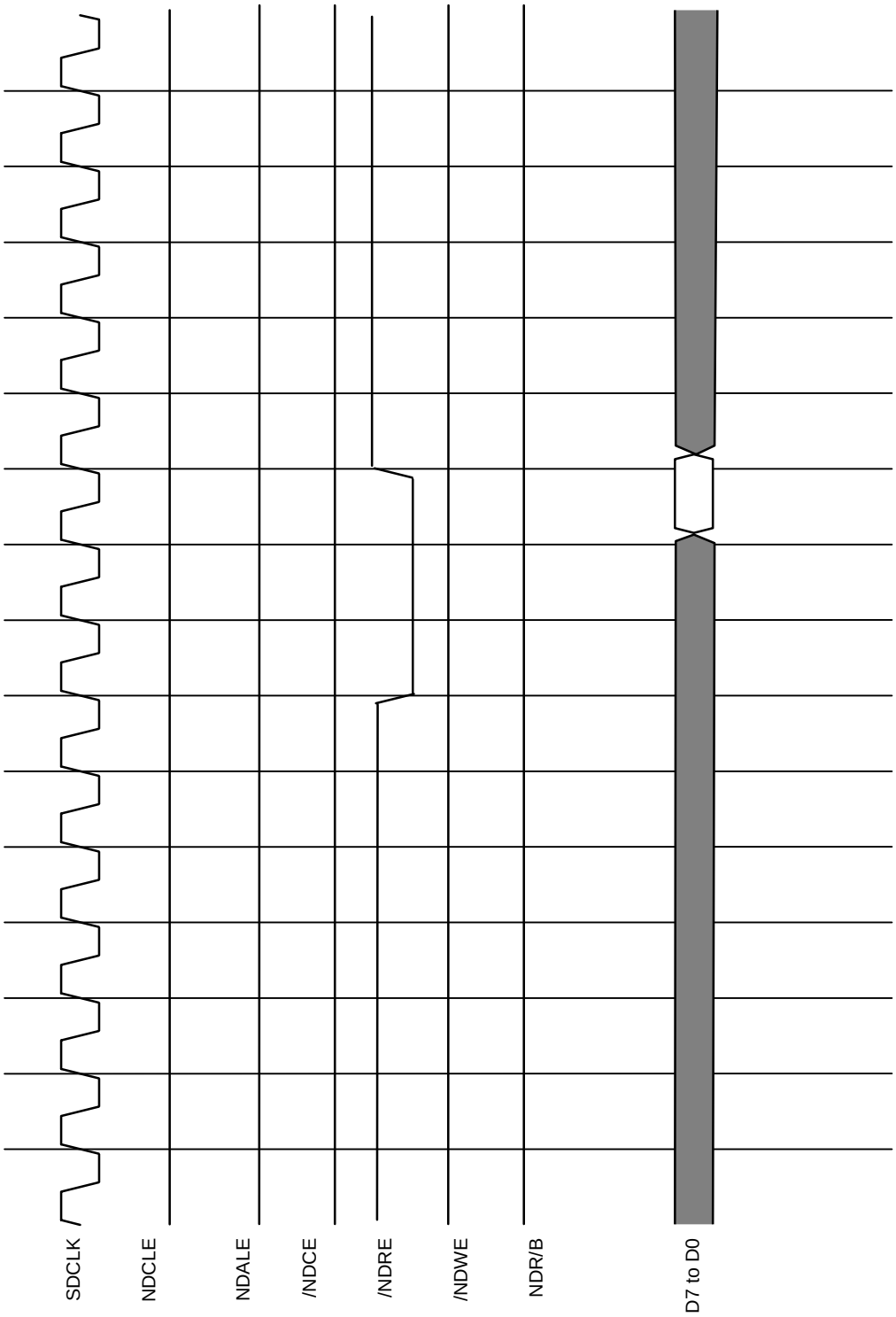


Figure 3.17.5(2) Data Read Cycle (ND0FSPR = 0x02)

3.17.5.3 Data Write Cycle

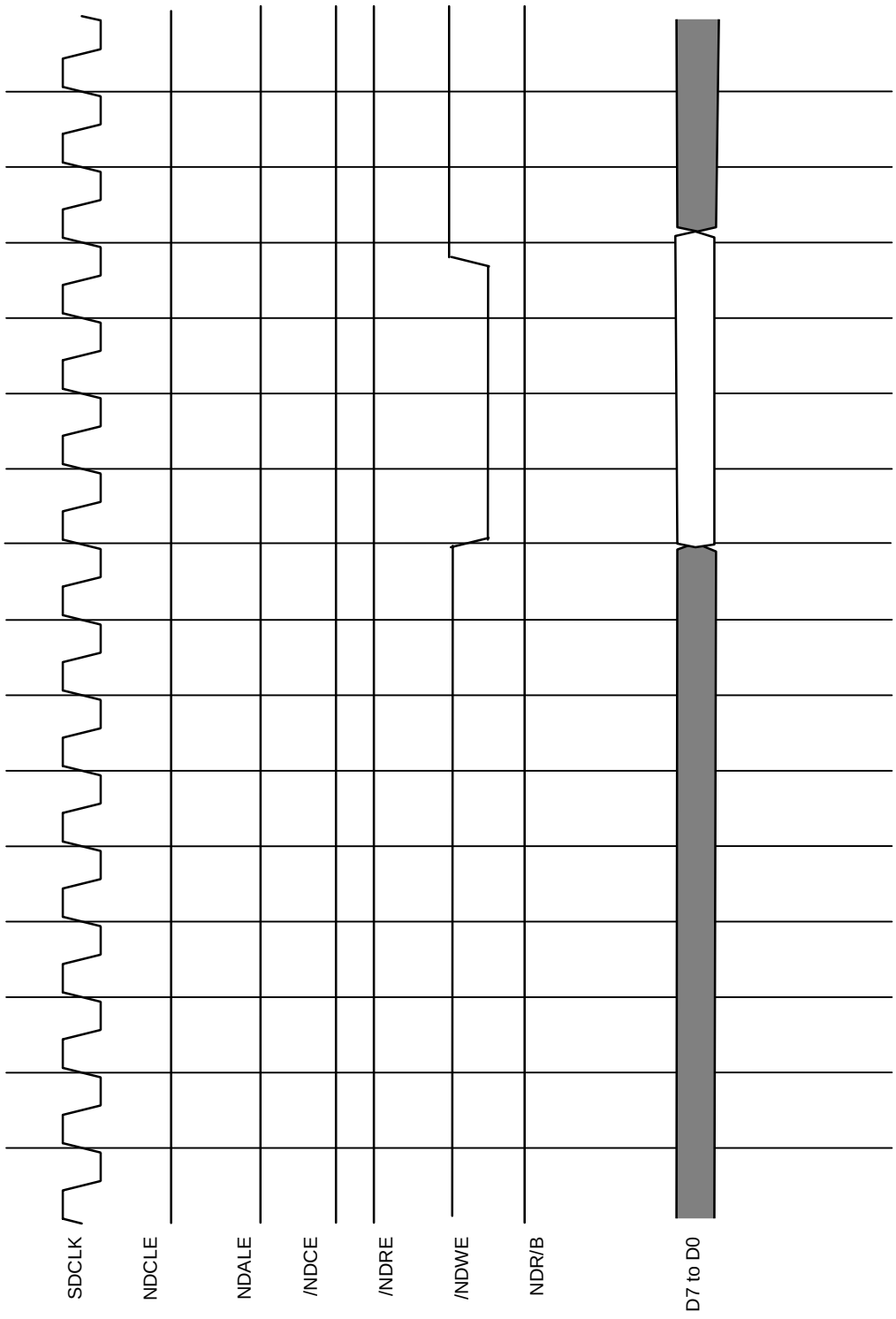
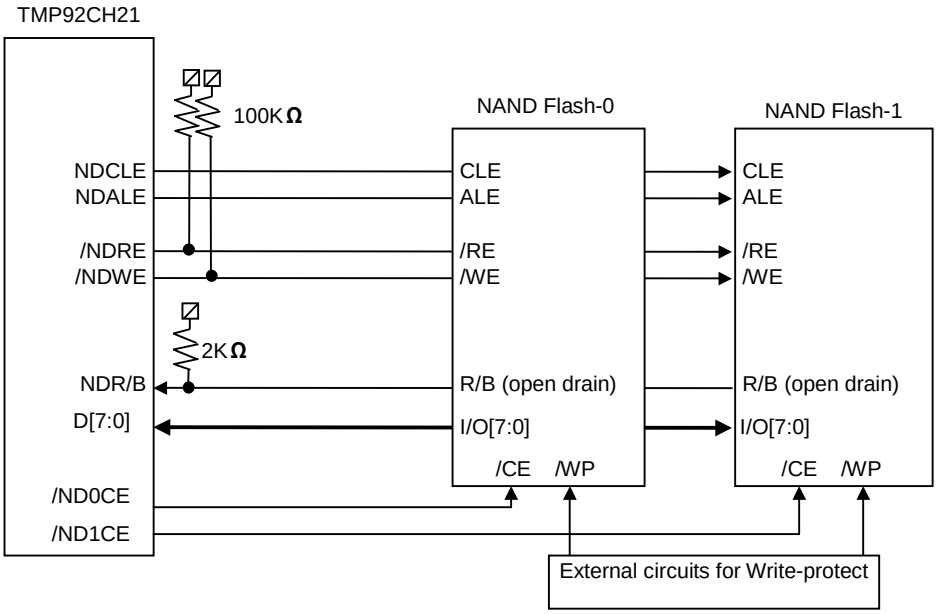


Figure 3.17.5(4) Data Write Cycle (ND0FSPR = 0x03)

3.17.6 Example of using NAND Flash



(Note1) By reset, both /NDRE and /NDWE pins become input port(Port71 and 72).
So, pull-up resisters are needed for them.
(Note2) About value of pull-up resister for NDR/B-pin, it depends on NAND-Flash and stray capacitance on the board. 2KΩ is a typical value.
(Note3) The /WP(Write Protect)-pin of NAND-Flash is not supported by TMP92CH21.
If necessary, prepare it on external-circuit.

Figure 3.17.2 Example of connecting NAND Flash

3.18 16-Bit Timer/Event Counters (TMRB)

The TMP92CH21 incorporates one multifunctional 16-bit timer/event counter (TMRB0) which have the following operation modes:

- 16-Bit Interval Timer Mode
- 16-Bit Event Counter Mode

16-Bit Programmable Pulse Generation (PPG) Mode
Can be used following operation modes by capture function:

- Frequency Measurement Mode
- Pulse Width Measurement Mode
- Time Differential 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.
This chapter consists of the following items:

3.18.1Block diagram

3.18.2Operation of each block

3.18.3SFRs

3.18.4Operation in each mode

(1) 16-Bit Interval Timer Mode

(2) 16-Bit Programmable Pulse Generation (PPG) Mode

Table 3.18.1 Pins and SFR of TMRB0

Spec. \ Channel		TMRB0
External Pins	External clock / Capture trigger input pins	None
	Timer flip-flop output pins	TB0OUT0 (also used as PC2)
SFR (address)	Timer Run Register	TB0RUN (1180H)
	Timer Mode Register	TB0MOD (1182H)
	Timer Flip-Flop Control Register	TB0FFCR (1183H)
	Timer Register	TB0RG0L (1188H) TB0RG0H (1189H) TB0RG1L (118AH) TB0RG1H (118BH)
	Capture Register	TB0CP0L (118CH) TB0CP0H (118DH) TB0CP1L (118EH) TB0CP1H (118FH)

3.18.1 Block diagrams

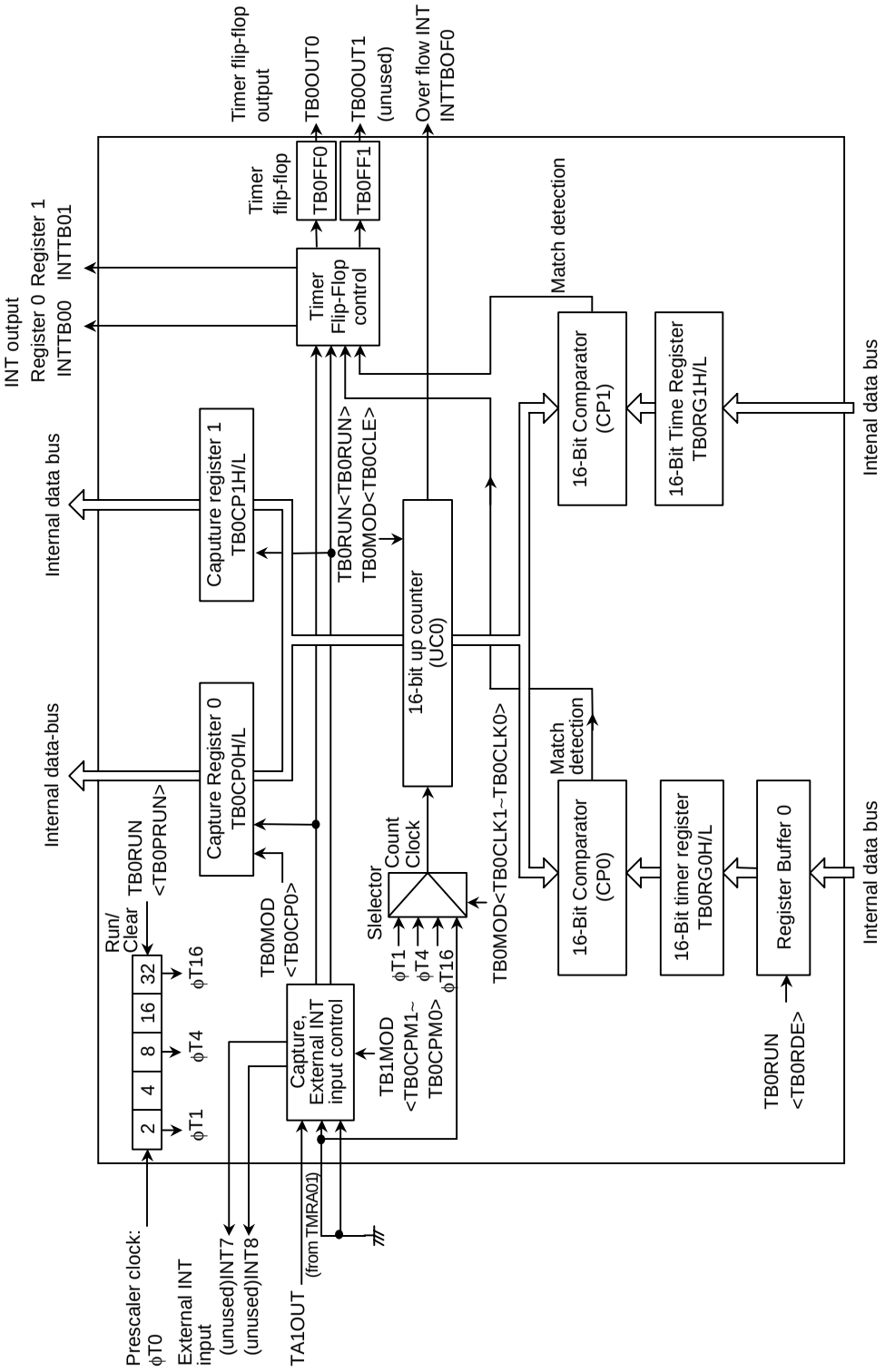


Figure 3.18.1 Block Diagram of TMRB0

3.18.2 Operation

(1) Prescaler

The 5-bit prescaler generates the source clock for timer 0. The prescaler clock (φT0) is divided clock (divided by 4) from system clock f_{SYS}.

This prescaler can be started or stopped using TB0RUN<TB0PRUN>. Counting starts when <TB0PRUN> is set to 1; the prescaler is cleared to zero and stops operation when <TB0PRUN> is set to 0.

Table 3.8.2 Prescaler clock resolution

System Clock Selection <SYSCK>	Gear Value <GEAR2 to0>	Prescaler Output Clock Resolution		
		φT1	φT4	φT16
1 (fs)	XXX	$fs/2^3$	$fs/2^5$	$fs/2^7$
0 (fc)	000 (fc)	$f_{SYS}/2^3$	$f_{SYS}/2^5$	$f_{SYS}/2^7$
	001 ($fc_{/2}$)	$f_{SYS}/2^4$	$f_{SYS}/2^6$	$f_{SYS}/2^8$
	010 ($fc_{/4}$)	$f_{SYS}/2^5$	$f_{SYS}/2^7$	$f_{SYS}/2^9$
	011 ($fc_{/8}$)	$f_{SYS}/2^6$	$f_{SYS}/2^8$	$f_{SYS}/2^{10}$
	100 ($fc_{/16}$)	$f_{SYS}/2^7$	$f_{SYS}/2^9$	$f_{SYS}/2^{11}$

xxx: Don't care

(2) Up-counter (UC0)

UC0 is a 16-bit binary counter which counts up pulses input from the clock specified by TB0MOD <TB0CLK1,TB0CLK0>.

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 & clearing of the counter is controlled by TB0RUN<TB0RUN>.

When clearing is enabled, the up-counter UC0 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.

A Timer Overflow interrupt (INTTBOF0) is generated when UC0 overflow occurs.

(3) Timer registers (TB0RG0 and TB0RG1)

These two 16-bit registers are used to set the interval time. When the value in the up-counter UC0 matches the value set in this timer register, the Comparator Match Detect signal will go Active.

Setting data for timer register is executed using 2 byte data transfer instruction or using 1 byte date transfer instruction twice for lower 8 bits and upper 8 bits in order.

The TB0RG0 timer register has a double-buffer structure, which is paired with register buffer. 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.

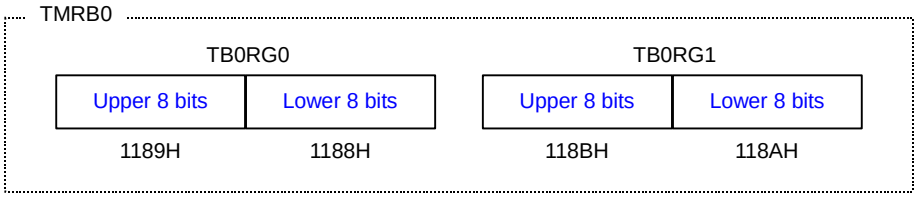
When the double buffer is enabled, data is transferred from the register buffer to the timer register when the values in the up-counter (UC0) and the timer register TB0RG1 match.

After a Reset, TB0RG0 and TB0RG1 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 as shown below.

TB0RG0 and the register buffer both have the same memory addresses (000188H & 000189H) allocated to them. If <TB0RDE> = 0, the value is written to both the timer register and the register buffer. If <TB0RDE> = 1, the value is written to the register buffer only.

The addresses of the Timer Registers are as follows:



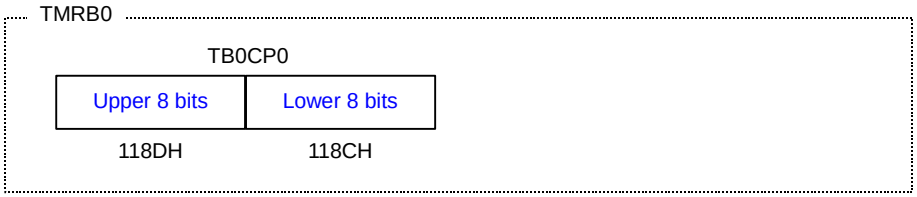
The Timer Registers are write-only registers and thus cannot be read.

(4) Capture Registers (TB0CP0H/L)

These 16-bit registers are used to latch the values in the up-counters.

Data in the Capture Registers should be read 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.

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 control

This circuit controls the timing to latch the value of up-counter UC0 into TB0CP0, TB0CP1

The value in the up-counter can be loaded into a capture register by software. Whenever 0 is written to TB0MOD<TB0CP0>, the current value in the up-counter is loaded into capture register TB0CP0. It is necessary to keep the prescaler in Run Mode (i.e. TB0RUN<TB0PRUN> must be held at a value of 1).

(6) Comparators (CP0 & CP1)

CP0 and CP1 are 16-bit comparators which compare the value in the up-counter UC0 with the value set in TB0RG0 or TB0RG1 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,TB0FF0C0> or <TB0FF1C1,TB0FF1C0>, 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".

The values of TB0FF0 can be output via the Timer Output pins TB0OUT0 (which is shared with PC6). Timer output should be specified using the Port B Function Register.

3.18.3 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	(Note) always fixed to "0"			IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

Count operation

0	Stop and Clear
1	Count

I2TB0: Operation during IDLE2-mode

TB0PRUN: Operation of prescaler

TB0RUN: Operation of TMRB0

Note: The 1, 4 and 5 of TB0RUN are read as underfined value.

Figure 3.18.2 The Registers for TMRB

TMRB0 Mode Register

TB0MOD (1182H)	7	6	5	4	3	2	1	0
Bit symbol		TB0ET1	TB0CP01	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
Read/Write		R/W	W*	R/W				
After Reset	0	0	1	0	0	0	0	0
Function	(Note) always fixed to "0"	TB0FF1 inversion 0: Disable 1: Enable Invert when the UC value matches the value in TB0RG1	Execute software capture 0: Execute 1: Don't care	Capture timing 00: Disable 01:reserved 10:reserved 11:TA1OUT↑ TA1OUT↓		Control up-counter 0: Disable clearing 1: Enable clearing	TMRB0 source clock 00: reserved 01: φT1 10: φT4 11: φT16	

TMRB0 source clock

00	TB0IN0 pin
01	φT1
10	φT4
11	φT16

Up-counter clear control

0	Disable
1	TB0RG1 clearing on match with TB0RG1.

Capture/Interrupt timing

00	Disable
01	reserved
10	reserved
11	CAP0 at TA0TRG rise CAP1 at TA0TRG fall

Software capture

0	The value in the up-counter is captured to TB0CP0.
1	Don't care

Figure 3.18.3 The registers for TMRB

TMRB0 Flip-Flop Control Register

TB0FFCR (1183H)

	7	6	5	4	3	2	1	0
Bit symbol	TB0FF1C1	TB0FF1C0		TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
Read/Write	W*			R/W			W*	
After Reset	1	1	0	0	0	0	1	1
Function	Control TB0FF1 00: Invert 01: Set 10: Clear 11: Don't care * Always read as 11		(Note) Always fixed to "0"	TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger Invert when the UC value is loaded in to TB0CP0.	Invert when the UC value matches the value in TB0RG1.	Invert when the UC value matches the value in TB0RG0.	Control TB0FF0 00: Invert 01: Set 10: Clear 11: Don't care * Always read as 11	

TB0FF0 control

00	Invert
01	Set to 11
10	Clear to 0
11	Don't care

Inverted when the UC value matches the valued in TB0RG0.

0	Disable trigger
1	Enable trigger

Inverted when the UC value matches the valued in TB0RG1.

0	Disable trigger
1	Enable trigger

Inverted when the UC value is loaded in to TB0CP0.

0	Disable trigger
1	Enable trigger

Figure 3.18.4 The Registers for TMRB

3.18.4 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 TB0RG1.

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Stop TMRB0.
INTETB01	←	X	1	0	0	X	0	0	0	Enable INTTB01 and 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 disable the capture function. (** = 01, 10, 11)
TB0RG1	←	*	*	*	*	*	*	*	*	Set the interval time (16 bits).
		*	*	*	*	*	*	*	*	
TB0RUN	←	0	0	X	X	–	1	X	1	Start TMRB0.

Note: X = Don't care; "–" = No change

(2) 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 UC0 with timer register TB0RG0 or TB0RG1 and to be output to TB0OUT0. In this mode the following conditions must be satisfied.

(Value set in TB0RG0) < (Value set in TB0RG1)

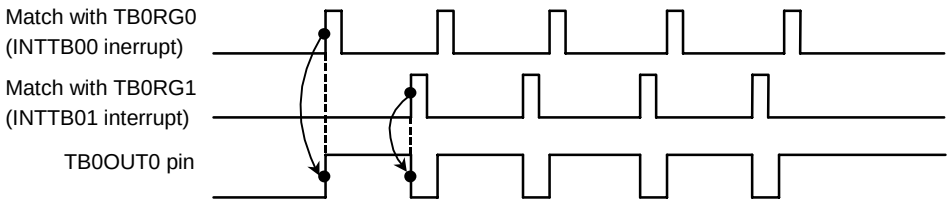


Figure 3.18.5 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0 double buffer is enabled in this mode, the value of Register Buffer 0 will be shifted into TB0RG0 at match with TB0RG1. This feature facilitates the handling of low-duty waves.

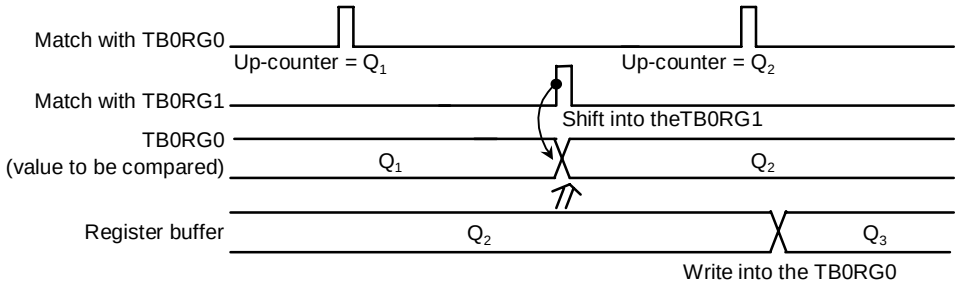


Figure 3.18.6 Operation of Register Buffer

The following block diagram illustrates this mode.

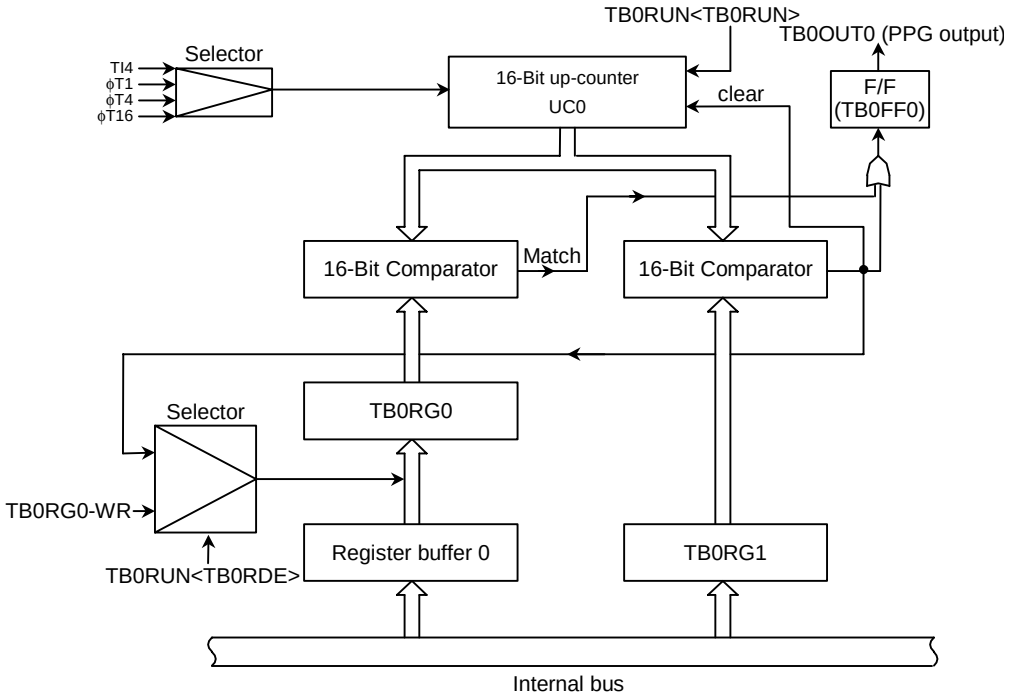


Figure 3.18.7 Block Diagram of 16-BIT Mode

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	–	0	X	0	Disable the TB0RG0 double buffer and stop TMRB0.
TB0RG0	←	*	*	*	*	*	*	*	*	Set the duty ratio (16 bits).
TB0RG1	←	*	*	*	*	*	*	*	*	Set the frequency (16 bits).
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0 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 TB0RG0/TB0RG1. 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)
PCCR	←	X	1	–	X	–	X	–	–	Set PC6 to function as TB0OUT0.
PCFC	←	X	1	–	X	–	X	–	–	
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

Note: X = Don't care; "–" = No change

3.19 Touch Screen Interface (TSI)

The TMP92CH21 has an interface for 4-terminal resistor network touch-screen.

This interface supports two procedure: an X/Y position measurement and a touch detection.

Each procedure can be performed by setting the TSI control register(TSICR0 and TSICR1) and using an internal AD converter.

3.10.1 Touch-Screen Interface Module Internal/External Connection

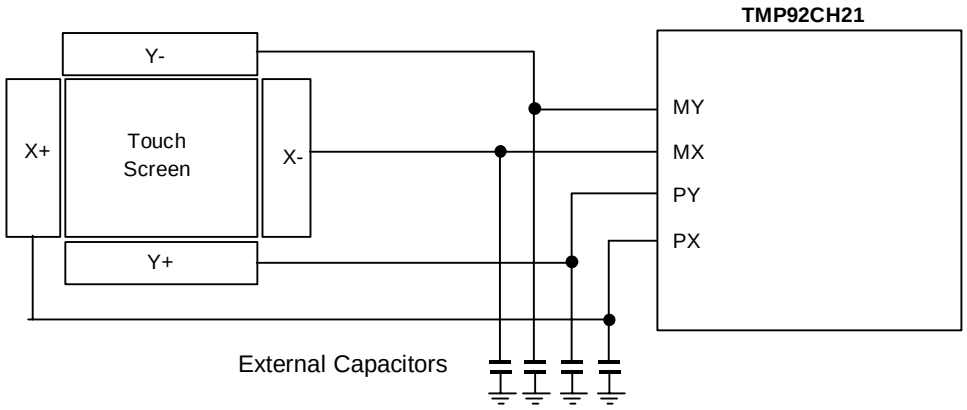


Figure 0.1 (1) External connection of TSI

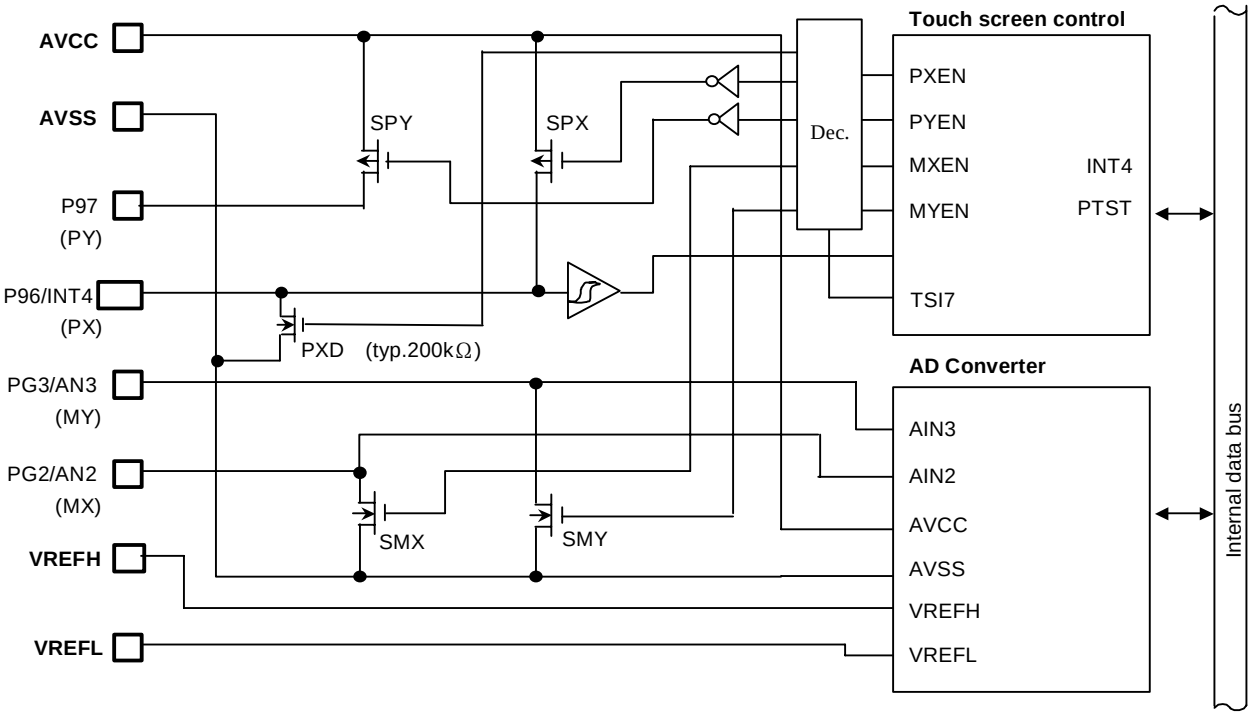


Figure 0.1 (2) Internal block diagram of TSI

3.10.2 Touch Screen Interface (TSI) Control Register

TSI Control Register

TSICR0 (01F0H)

	7	6	5	4	3	2	1	0
bit Symbol	TSI7		PTST	TWIEN	PYEN	PXEN	MYEN	MXEN
Read/Write	R/W		R	R/W	R/W	R/W	R/W	R/W
After reset	0		0	0	0	0	0	0
	0 : Disable 1 : Enable		0 : not touch 1 : touch	0 : Disable 1 : Enable	0 : Disable 1 : Enable SPY:ON	0 : Disable 1 : Enable SPX:ON	0 : Disable 1 : Enable SMY:ON	0 : Disable 1 : Enable SMX:ON

PXD(Pull down resistance) 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
	0 : Disable 1 : Enable	1024	256	64	8	4	2	1
		De-bounce time is set by “(N*64-16) / fSYS” -formula. “N” is sum of number which is set to “1” in bit6 to bit0 .						

(Note) Since an internal clock is used for de-bounce circuit, disable this circuit (Write “0” to TSICR1<DBC7>) before entering IDLE1 or STOP-mode.

3.10.3 Touch detection procedure

A touch detection procedure is a preparing procedure till a pen touches to the screen.

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 the waiting state, ON only SPY-switch and OFF other 3 switches(SMY,SPX and SMX).

The pull down resistor which is connected to P96/INT4/PX pin is ON by SPX-switch is OFF.

During this waiting state, P96/INT4/PX pin's level is L because of the internal resistors between X and Y-direction in the touch screen are not connected and INT4 isn't generated.

If the pen touches, P96/INT4/PX pin's level is H because of the internal resistors between X and Y-direction in the touch screen are connected and INT4 will be generated.

And the de-bounce-circuit like following diagram is prepared to avoid some number's interrupt generation though one time touch.

This can ignore the pulse under the time which is set to TSICR1 register.

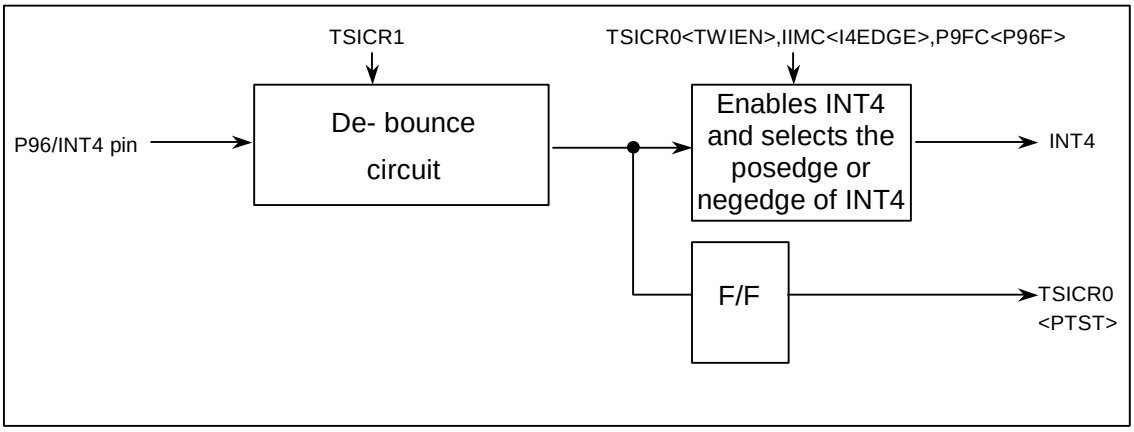


Figure 0.2 (1) Block diagram of de-bounce circuit

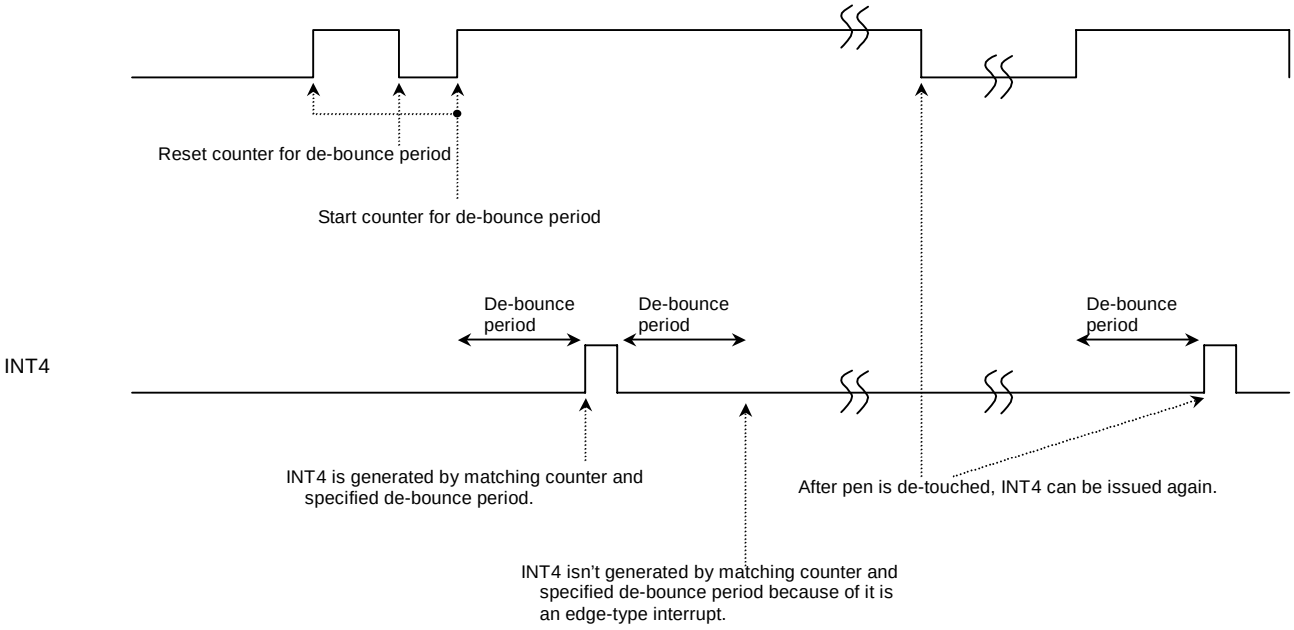


Figure 0.2 (2) Timing diagram of de-bounce circuit

3.10.4 X/Y position measuring procedure

In the INT4 routine, execute an X/Y position measuring procedure like below.

< X position measurement>

At first, ON both SPX and SMX-switches and OFF other 2 switches.

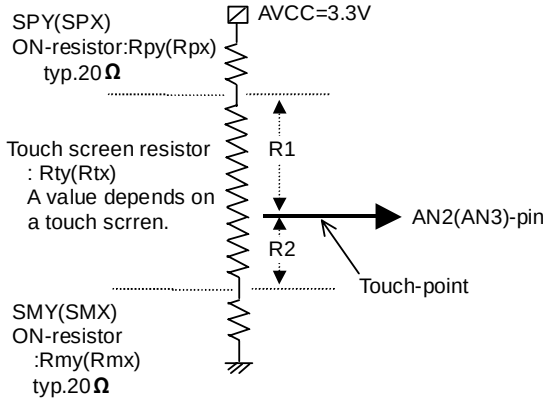
By this setting, analog-voltage which shows the X-position will be inputted to PG3/MY/AN3 pin . The X-position can be measured by converting this voltage to digital code with AD converter .

< Y position measurement>

Next, ON both SPY and SMY-switches and OFF other 2 switches.

By this setting, analog-voltage which 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 which is inputted to AN3 or AN2-pin can be calculated .
It is a ratio between resistance-value in TMP92CH21F and resistance-value in touch screen shown in figure 3.10.2(3) .
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.



[Formula to calculate analog voltage(E1) to AN2 or AN3-pin]

E1 = ((R2+Rmy) / (Rpy+Rty+Rmy)) * AVCC [V]

ex.) The case of AVCC=3.3V,Rpy=Rmy=20Ω,
R1=400Ω and R2=100Ω
E1 = ((100+20) / (20+400+100+20) * 3.3
= 0.733V

Note1) An X-position can be calculated in the same way though above formula is for Y-position.
Note2) Rty = R1+R2 .

Figure3.10.2(3) Calculation analog voltage

3.10.5 Flow chart for TSI

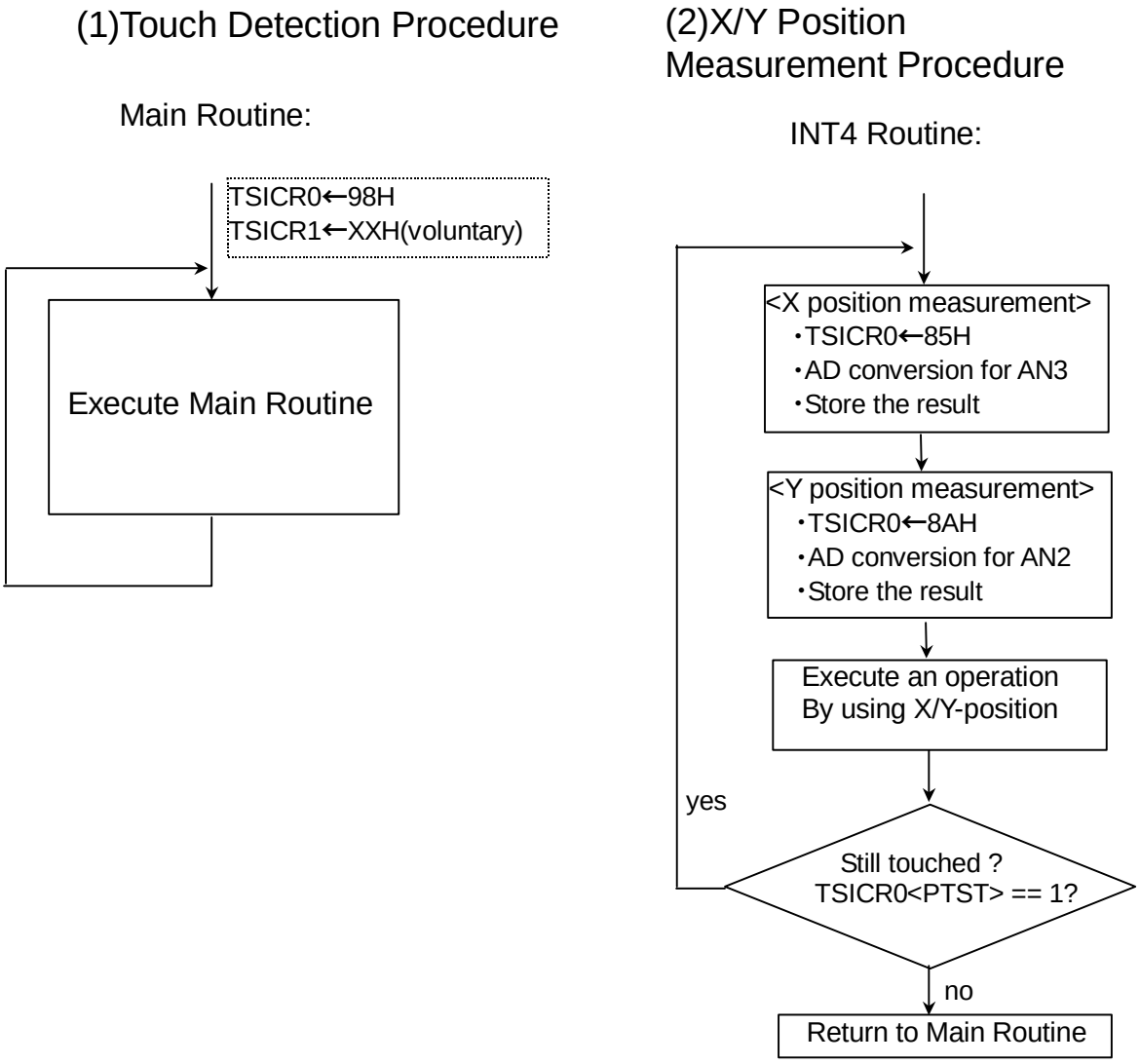


Figure 3.10.5 Flow chart for TSI

3.20 I^2S (Inter-IC Sound)

A serial-output circuit with I^2S-format compatible is built-in.

This product can be used for the digital audio system application by connecting LSI for voice generation (ex. D/A Converter).

This circuit has both I^2S-mode and general SIO-mode. But both mode have only clock-output and data-transmitting function .

Table 3.20.1 shows outline for each mode.

Table 3.20.1 Outline for each mode

	I^2S-mode	SIO-mode
1)Format	I^2S-format compatible (Only Master, transmitting)	General (Only Master, transmitting)
2)Used pin	1. I2SCKO(clock output) 2. I2SDO (clock output) 3. I2SWS (word-select output)	1. I2SCKO(clock output) 2. I2SDO (data output)
3)WS frequency	Selectable either fs/4 or TA1OUT(TMRA1 output)	---
4)Baud-rate (@fc=40MHz)	Selectable either 20,10,5, or 2.5Mbps	
5)Transmittion buffer	16Byte×2ch(Right,Left)	32Byte
6)Direction of data	Selectable either MSB-first or LSB-first	
7)Data length	Selectable either 8bit or 16bit	
8)Edge of clock	Selectable either rising-edge or falling-edge	
9)Interrupt	INTI2S(FIFO-empty interrupt)	

3.20.1 Block diagram

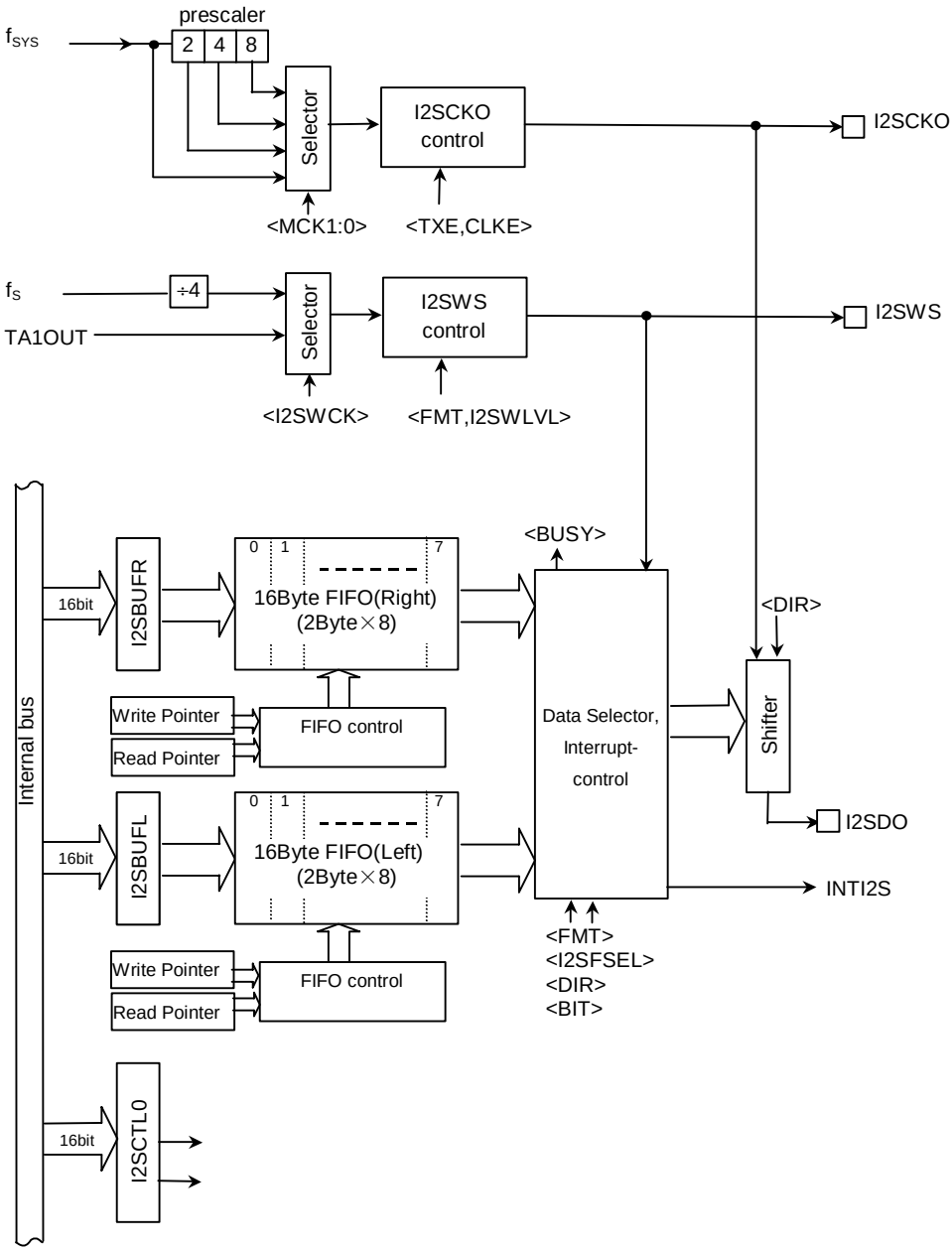


Figure3.20.1 I^2S block diagram

3.20.2 SFR

The following tables are SFR for I2S. This I2S are connected to CPU with 16-bit data bus. So, when CPU access SFR, the 2-byte load-instruction should be used.

I2SCTL0 Register

	7	6	5	4	3	2	1	0
bit Symbol	TXE	FMT	BUSY	DIR	BIT	MCK1	MCK0	I2SWCK
Read/Write	R/W		R	R/W				
After reset	0	0	0	0	0	0	0	0
Function	Transmit 0:stop 1:start	Mode 0: I²S 1:SIO	Status 0:stop 1:under-transmitting	First-BIT 0:MSB 1:LSB	Bit number 0:8bit 1:16bit	Baud-rate 00:fsys 10:fsys/4 01:fsys/2 11:fsys/8		WS clock 0:fs/4 1:TA1OUT

I2SCTL0 (080EH)

Note1. <I2SWCK> is effective only for I²S-mode.

	15	14	13	12	11	10	9	8
bit Symbol	I2SWLVL	EDGE	I2SFSEL	I2SCLKE				SYSCKE
Read/Write	R/W							
After reset	0	0	0	0				0
Function	WS level 0:low left 1:high left	clock edge for data-out 0:falling 1:rising	Select for Stereo 0:Stereo (2-channel) 1:Monaural (1-channel)	Clock-enable (After transmit) 0:operation 1:stop				System-clock 0:disable 1:enable

(080FH)

Note1. <I2SWLVL>,<I2FSEL> and <I2SCLKE> are effective only for I²S-mode.

I2SBUFR Register

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
Read/Write	W															
After reset	undefined															
Function	Register for transmitting buffer(FIFO) (Right-channel)															

I2SBUFR (0800H)

I2SBUFL Register

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	L0
Read/Write	W															
After reset	undefined															
Function	Register for transmitting buffer(FIFO) (Left-channel)															

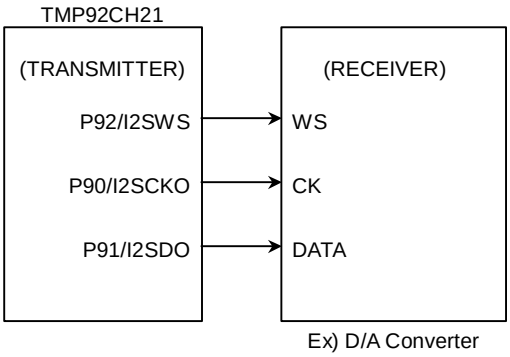
I2SBUFL (0808H)

Figure 3.20.2 I²S SFR

3.20.3 Explanation of I²S-mode

(1) Connection example

Figure 3.20.3(1) shows an example with external-LSI.



Note) Since P90 to P92 become High-impedance, connect pull-up or pull-down resistor if necessary.

Figure 3.20.3(1) example with external-LSI

(2) Procedure

The 32byte-FIFO are built-in and if a FIFO's data became empty, INTI2S-interrupt is occurred.
In the interrupt routine, write next transmittion data to FIFO.
The below shows a setting example and timing diagram.

(Setting example) Transmitting by I²S-mode,I2SWS=8.192KHz,I2SCKO=10MHz, synchronous to rising edge (at f_{SYS}=20MHz)

(Main routine)

	7	6	5	4	3	2	1	0	
INTE5I2S	X	0	0	1	X	-	-	-	Set interrupt level
P9CR	-	-	-	-	-	0	0	0	Set pins to P90(I2SCKO), P91(I2SDO),P92(I2SWS)
P9FC	-	-	-	-	-	1	1	1	
I2SCTL0	0	0	-	0	0	0	1	0	Set I²S-mode,MSB-first,8bit,f_{SYS}/2 clock
	0	1	0	1	0	0	0	1	Set rising-edge, clock-stop
I2SBUFR	**	**	**	**	**	**	**	**	Write 16 byte data to FIFO for Right (8times)
I2SBUFL	**	**	**	**	**	**	**	**	Write 16 byte data to FIFO for Left (8times)
I2SCTL0	1	0	-	0	0	0	1	0	Start transmitting
	0	1	0	1	0	0	0	1	

(INTI2S interruptroutine)

I2SBUFR	**	**	**	**	**	**	**	**	Write 16 byte data to FIFO for Right (8times)
I2SBUFL	**	**	**	**	**	**	**	**	Write 16 byte data to FIFO for Left (8times)

(note) X: don't care -: no change

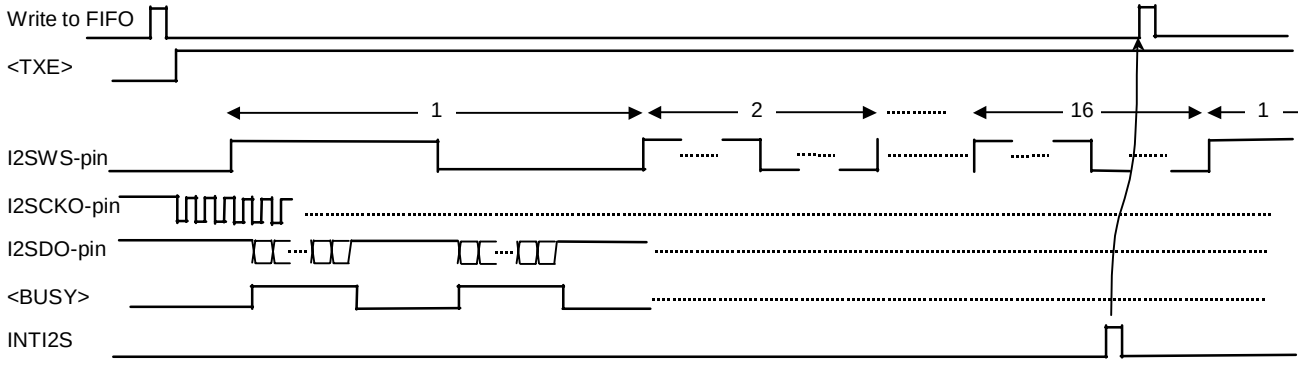


Figure 3.20.3(2) Whole timing diagram

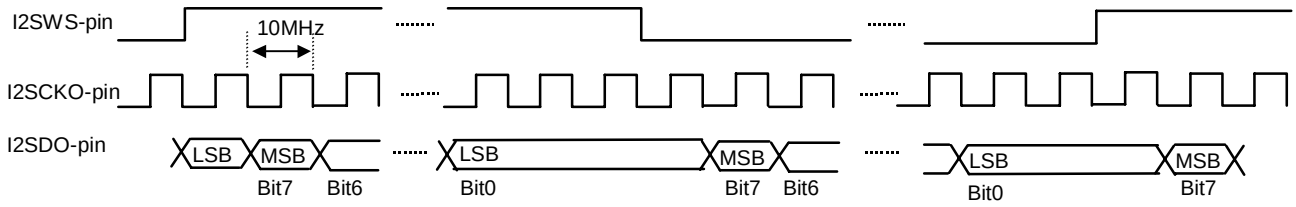


Figure 3.20.3(3) Detail timing diagram

(3) Notes

1) INTI2S timing

INTI2S is generated after the last data in FIFO is loaded to the internal shifter. It means FIFO is empty and shows to write enable next-data.

2) I2SCTL0<TXE>

A transmitting is started by writing “1” to <TXE>-register and stopped by writing “0”. After <TXE> is written “1” once, a transmitting is repeated automatically for Right to Left in order, alternately.

If a transmitting should be stopped, write “0” to <TXE> after <BUSY> changes to “0” in the INTI2S-interrupt routine.

When <TXE> is written “0” during transmitting, transmitting stops immediately.

3) FIFO size

The 16byte FIFO is provided for each Right,Left-channel. All-data don’t have to be used. But use even-number like 2,4, • • ,16.

4) I2SCTL0<I2SFSEL>

Write “1” to <I2SFSEL> and use Right-channel of FIFO in case of “Monaural” . Transmitting-data for Left-channel will be always “0” without writing data to Left-channel of FIFO .

5) Address for I2SBUFR and I2SBUFL

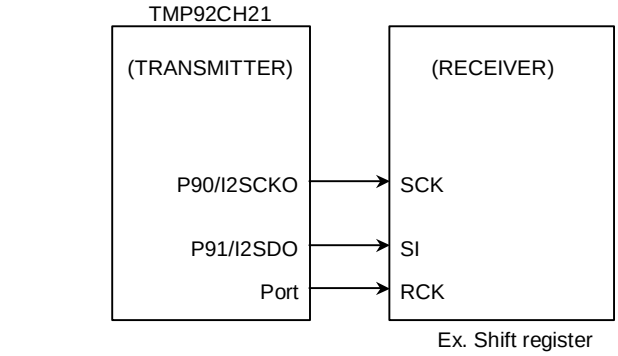
Use “word or long-word data load instruction” to write data to I2SBUFR or I2SBUFL. A “byte data load instruction” can not be used.

The address of I2SBUFR is selected from 0800H to 0803H and I2SBUFL is selected from 0808H to 080BH.

3.20.4 Explanation of SIO-mode

(1) Connection example

Figure 3.20.4(1) shows an example with external-LSI.



Note) Since P90 to P91 become High-impedance, connect pull-up or pull-down resistor if necessary.

Figure 3.20.3(1) example with external-LSI

(2) Procedure

The 32byte-FIFO are built-in and if a FIFO's data became empty, INTI2S-interrupt is occurred. In the interrupt routine, write next transmittion data to FIFO.

The below shows a setting example and timing diagram.

(Setting example) Transmitting by SIO-mode,I2SCKO=10MHz, synchronous to rising edge (at f_{SYS}=20MHz)

(Main routine)

	7	6	5	4	3	2	1	0	
INTE5I2S	X	0	0	1	X	-	-	-	Set interrupt level
P9CR	-	-	-	-	-	-	0	0	Set pins to P90(I2SCKO), P91(I2SDO)
P9FC	-	-	-	-	-	-	1	1	
I2SCTL0	0	1	-	1	0	0	1	-	Set SIO-mode,LSB-first,8bit,f_{SYS}/2 clock
	-	1	-	1	0	0	0	1	Set rising-edge
I2SBUFR	**	**	**	**	**	**	**	**	Write 32 byte data to FIFO (16times)
I2SCTL0	1	1	-	1	0	0	1	-	Start transmitting
	-	1	-	1	0	0	0	1	

(INTI2S interruptroutine)

I2SBUFR	**	**	**	**	**	**	**	**	Write 32 byte data to FIFO (16times)
If <BUSY> == "1" then WAIT else NEXT									Sence and wait for termination trasmit 32byte data
I2SCTL0	1	1	-	1	0	0	1	-	Start transmitting
	-	1	-	1	0	0	0	1	

(note) X: don't care -: no change

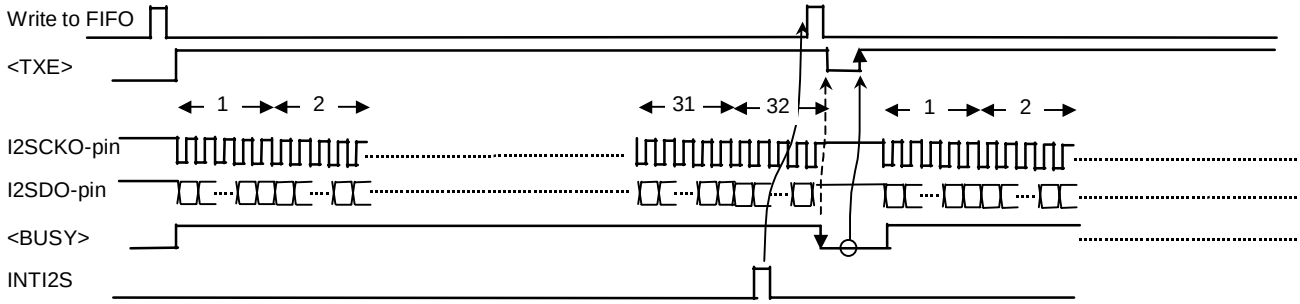


Figure 3.20.4(2) Whole timing

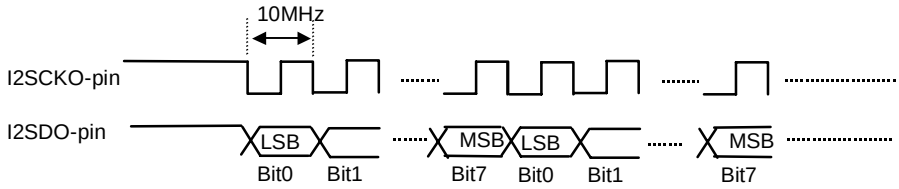


Figure 3.20.4(2) Detail timing

(3) Notes

1) INTI2S timing

INTI2S is generated after the last data in FIFO is loaded to the internal shifter.

It means FIFO is empty and shows to write enable next-data.

2) I2SCTL0<TXE>

A transmitting is started by writing “1” to <TXE>-register and stopped by writing “0”.

<TXE> register is cleared to “0” by <BUSY> changed “1” to “0”.

When <TXE> is written “0” during transmitting, transmitting stops immediately.

3) FIFO size

The 32byte FIFO is provided for SIO-mode. All-data don’t have to be used.

But use even-number like 2,4, • • ,32.

The <BUSY> will be changed to “0” and <TXE> will be cleared to “0” automatically after transmitting of all written data to FIFO. In case of continuous transmitting, write “1” to <TXE> after writing data to FIFO.

The data written number to FIFO is counted automatically and holded by writing “1” to <TXE>.

4) Address for I2SBUFR and I2SBUFL

Use “word or long-word data load instruction” to write data to I2SBUFR(Don’t write to I2SBUFL). A “byte data load instruction” can not be used.

The address of I2SBUFR is selected from 0800H to 0803H.

3.21 Boot ROM for 1st-cut

The Boot ROM is built-in to download user's boot-program.

The 3-kind of downloading way are supported.

(Notes)
There are some limitations for the 1st-cut sample's boot-program.
They are shown by red-color. Please be careful about them.

3.21.1 Operation mode

There are 2 operation mode : MULTI-mode, BOOT-mode. Each mode is set according to the condition of AM1 and AM0 pins when /RESET is asserting.

(1) MULTI mode : After reset, a CPU fetch and execute instruction from external-memory.

(2) BOOT-mode : After reset, a CPU fetch and execute instruction from internal-Boot-ROM. An user-program which execute programming to on-board memory (ex. NOR-Flash) is loaded from any source to internal-RAM and brunch to internal-RAM.
By this operation, an user-program will start to boot.
Table 3.21.2 shows outline of boot-operation.

Table 3.21.1 Operation mode

Mode setting pins			Operation mode	
/RESET	AM1	AM0		
	0	1	MULTI	Start from external 16bit-bus memory
	1	0		Start from external 32bit-bus memory
	1	1	BOOT (Start from internal-BOOT ROM)	
	0	0	TEST (Disabled to set)	

Table 3.21.2 Outline of boot-operation

Name	Order of sensing	Loading			Operation after Loading
		Source	I/F	Destination	
(a)	1	NAND-Flash	Data bus	Internal-RAM	Brunch to Internal-RAM
(b)	2	PC	USB		
(c)	3	PC	UART		

(Notes) For the 1st-cut sample, downloading via (a)UART and (b)USB is not supported.

3.21.2 Hardware specification about internal Boot-ROM

(1) Memory map

Figure 3.21.3 shows a memory map of BOOT-mode.

An 8Kbyte ROM is built-in and it is mapped 3FE000H to 3FFFFFH address.

At MULTI-mode, boot-ROM is not mapped and its area is mapped as external area.

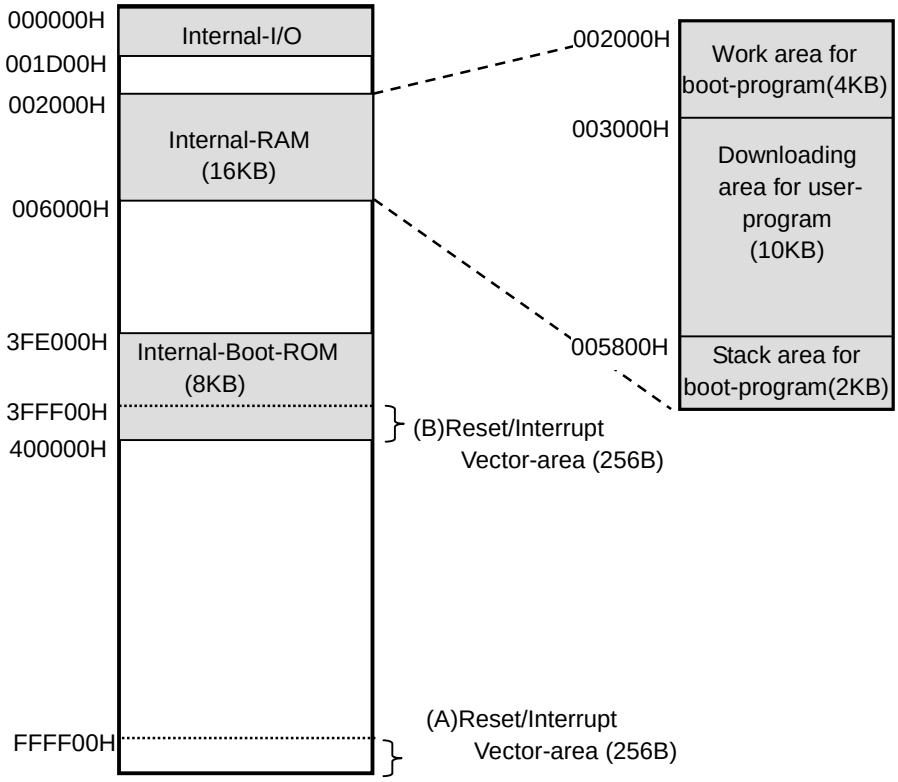


Figure 3.21.3 Memory map of BOOT-mode

(2) Reset/Interrupt-Address conversion circuit

Reset/Interrupt-Vector-address conversion circuit is prepared.

An individual Reset/Interrupt Vector-area can be had by this function. For the detail, refer 3.6.5 Internal Boot-ROM Control.

(3) 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.

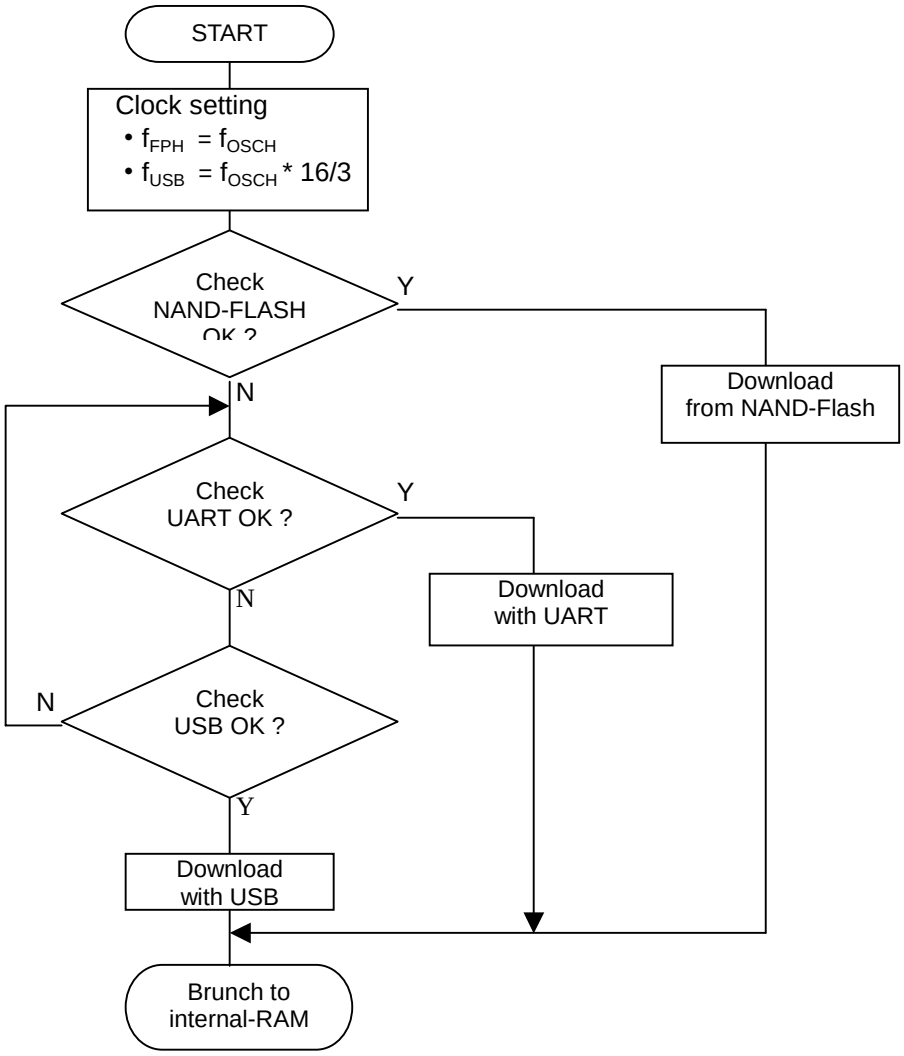
So, an internal Boot-ROM can be disappered by setting BROMCR<ROMLESS> to “1”.

For the detail, refer 3.6.5 Internal Boot-ROM Control.

3.21.3 Outline of boot-operation

There are 3 kinds of downloading way: NAND-Flash, UART and USB.

After reset, a boot-program in the boot-ROM operate like following flow-chart.

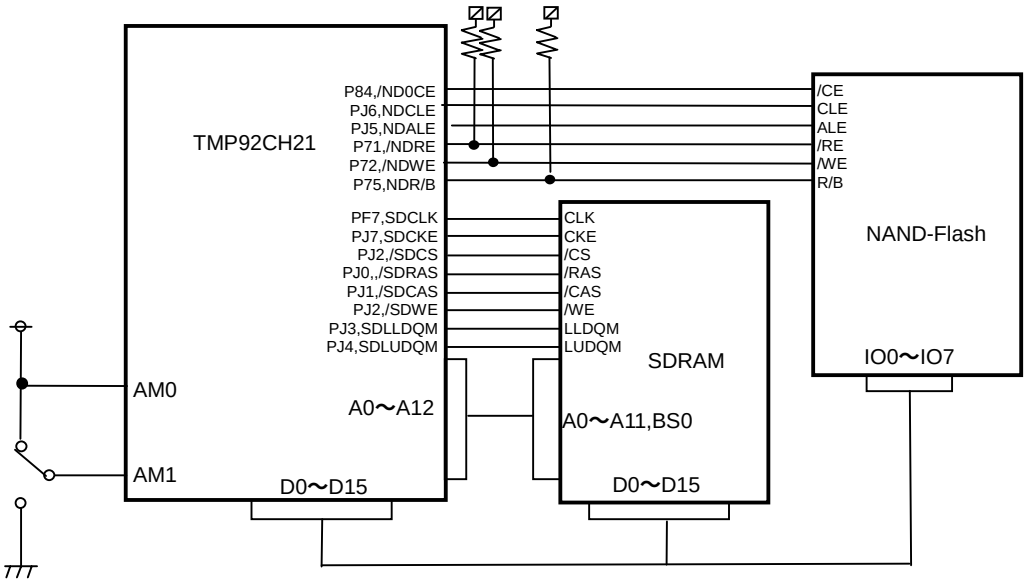


(Note) When a downloading with USB is used, a special USB device-driver is necessary.

3.21.4 Download from NAND-Flash

(1) Connection example

Figure 3.21.4(1) shows an example of NAND-Flash with 16-bit SDRAM.



(Note1) By reset, both /NDRE and /NDWE pins become input port(Port71 and 72). So, pull-up resistors are needed for them.

(Note2) About value of pull-up resistor for NDR/B-pin, it depends on NAND-Flash and stray capacitance on the board.
2KΩ is a typical value.

(Note3) The /WP(Write Protect)-pin of NAND-Flash is not supported by TMP92CH21. If necessary, prepare it on external-circuit.

Figure 3.21.4(1) NAND-Flash connecting example

(2) Usable NAND-Flash

The boot-program is designed based on Smart Media™ physical-format specification Ver1.20. Table 3.21.4(1) shows usable memory devices

Table 3.21.4 (1) Usable memory

Memory size[Mbyte]	NAND-Flash 3.3V Model	Masked ROM 3.3V Model
1	usable	Un-usable
2		usable
4		
8		
16		
32		
64		
128		

(3) Data assignment

The download data are consist of boot identification code(4Byte),user-program size(2Byte) and user-program(max.10Kbyte). They should be assigned(programmed) in NAND-Flash like Figure 3.21.4(2).

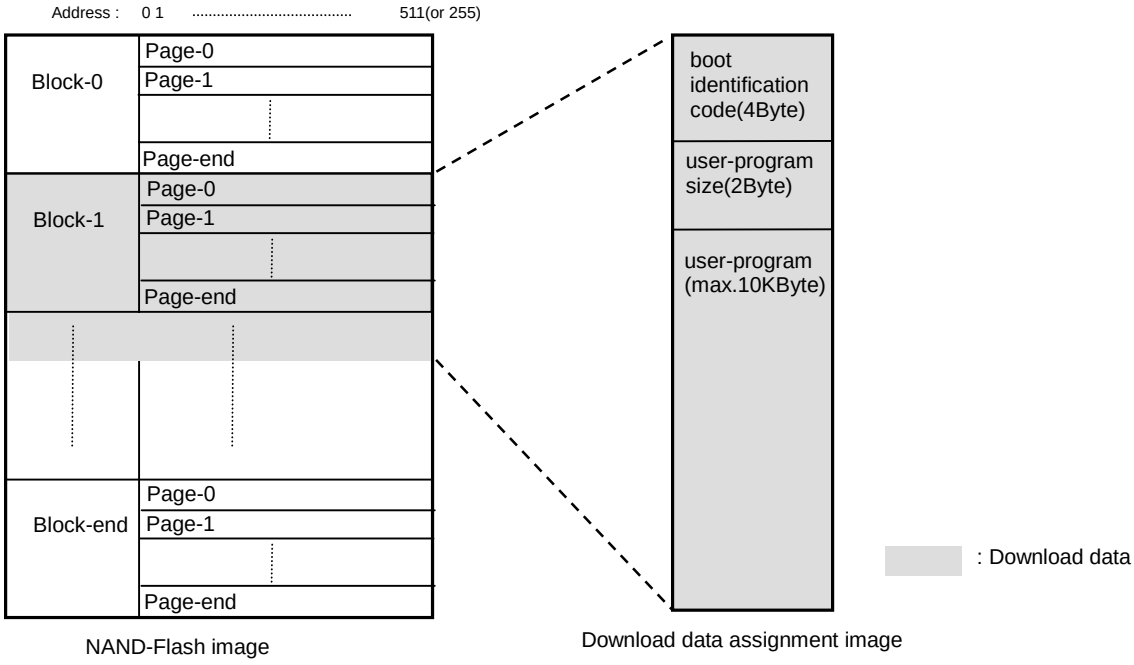


Figure 3.21.4(2) Download data image

a) Boot identification code(4Byte)

At first, a boot-program checks “boot identification code”. If “BOOT” characters in ASCII-code are read from first 4-Byte in Page-0, Block-1 of the NAND-Flash, the boot-program will recognize the booting way is NAND-Flash.

42H(“B”)
4FH(“O”)
4FH(“O”)
54H(“T”)

b) User-program size(2Byte)

A program size should be programmed to next 2-Byte. The first-byte is lower 8-bit and 2nd is upper 8-bit. This size means only user-program size, it doesn’t include “boot identification code(4Byte)“ and “user-program size(2Byte)”.

This must be less than or equal 10Kbyte. So, maximum number is 2800H.

Size(lower 8-bit)
Size(upper 8-bit)

c) User-program (max.10KByte)

An user-program is loaded to internal-RAM.
Be careful the start address is 3000H to make user-program.
Beforehand, program(write) an user-program to NAND-Flash in Binary-format.

The below is an example to explain how to make binary-format file.

(Example) How to convert from extended intel-HEX format file
to Binary-format file

The following is an example of display in text-editor in case of intel-HEX format file is opened.

:103000000607F100030000F201030000B1F16010B7
:00000001FF

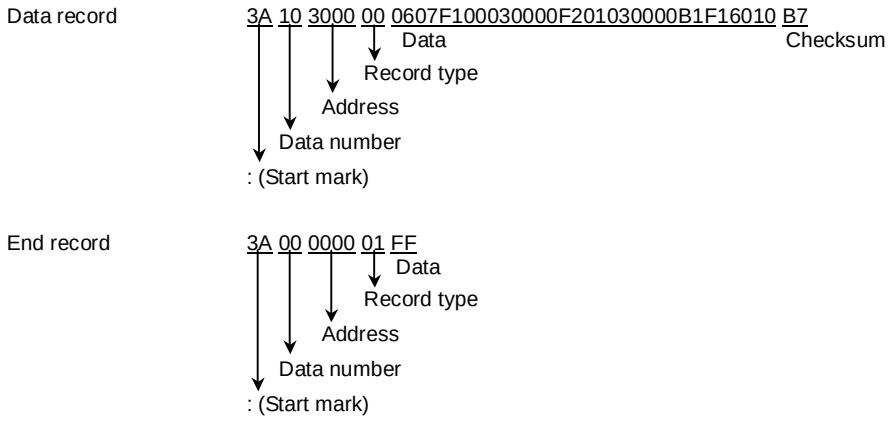
But actually their data are below because of ASCII-code is used for intel-HEX format file.

3A313033303030303030363037463130303033303030304632303130333030303042
31463136303130423703D0A3A3030303030303031464603D0A

So, at first convert above data to binary-format with below table's law .
And next, delete characters except data (Start mark, Data number, Address, Record type and checksum).

Before (ASCII)	After (Binary)
3A	3A（Only 3A should not be converted.）
30 to 39	0 to 9
41or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete it

The intel-HEX format and its meaning are below.



(4) Error check item

The below are checked items by a boot-program.

If any error occurs in each check, a boot-program will cancel downloading from NAND-Flash and skip to next(UART) operation.

a) Usable NAND-Flash

The boot-program reads a device-code from NAND-Flash and checks whether it is usable or not.

b) Boot identification code

c) User-program size

The boot-program checks whether it is less than or equal 10Kbytes.

d) Block status area

The boot-program checks whether each block is normal or not.

e) Data status area

The boot-program checks whether each data status is correct or not.

f) ECC error

The boot-program reads both calculated code from NDFC and ECC-code in NAND-Flash and checks whether they are correctable or not.

g) Ready status of NAND-Flash

The boot-program checks whether Ready-status is normal or not in each action.

The “normal-status” is defined to the time changed to Ready is less than 70[us] at f_{FPH}=40MHz.

(5) ECC-error check

a) Calculation ECC-code

The NDFC (NAND-Flash controller) is used for calculation of ECC-code.

b) ECC-code correction

The boot-program operates like below.

1. Compares both calculated ECC-code from NDFC and ECC-code in NAND-Flash.
2. Judges and corrects according to following each case.

case-(a) : No data-error -> (O.K) Next operation

case-(b) : 1bit data-error -> (O.K) Error correction and next operation

(Notes) For the 1st-cut sample, the case-(b)’s Error correction can not be operated correctly. If an 1-bit data-error occurred, it will be judged to case-(c).

case-(c) : 2bit or more data-error -> (N.G) Termination

case-(d) : ECC-code error -> (O.K) Next operation

(6) Port setting

The boot-program set some ports which are shown in Table 3.21.4(2) to function ports.

Another ports are not set, so they are kept after reset status.

It is needed to care these ports to design an application system.

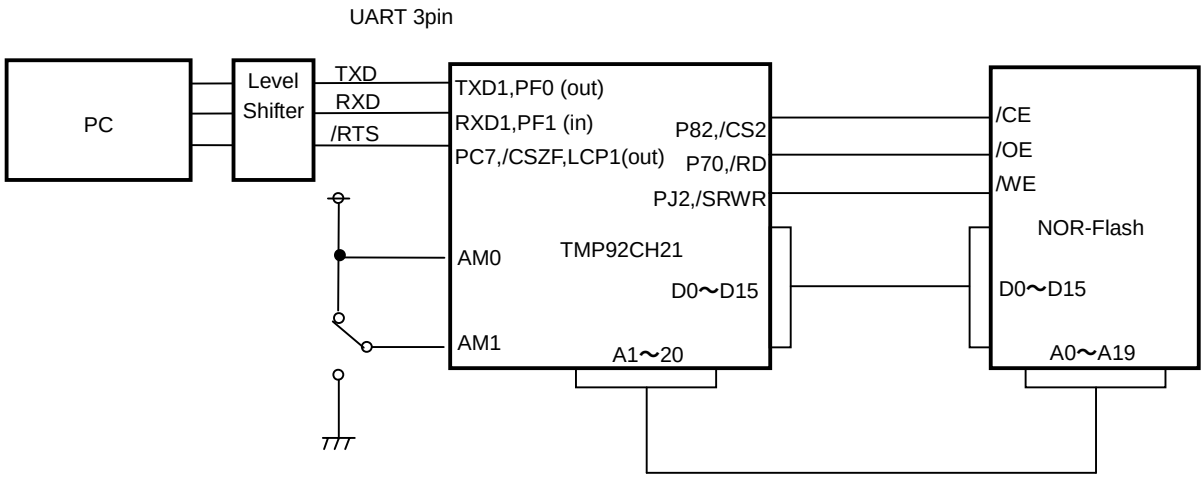
Table 3.21.4(2) Settled Ports

Pin name	After reset in BOOT-mode	Settled function	Note
P70(/RD)	Output port	/RD output	
P71(/NDRE,/WRLL)	Input port	/NDRE output	
P72(/NDWE,/WRLH)	Input port	/NDWE output	
P75(R/B,R/W)	Input port	R/B input	
P82(/CS2,/CSZA,/SDCS)	Output port	/CS2	
P84(/ND0CE,/CSZB,/WRUL)	Output port	/ND0CE	
P86(/SRULB,/CSZD)	Output port	/SRULB	
P87(/SRUUB,/CSZE)	Output port	/SRUUB	
PJ5(NDALE,SDULDQM)	Output port	NDALE	
PJ6(NDCLE,SDUUDQM)	Output port	NDCLE	
PJ5(NDALE,SDULDQM)	Output port	NDALE	
PC7(/CSZF,LCP1)	Input port	Output port for /RTS	
PF0(TXD1,TXD0)	Input port	TXD1 output	Not open-drain port. This port becomes TXD1 after matching data received.
PF1(RXD1,RXD0)	Input port	RXD1 input	

3.21.5 Download with UART

1) Connection example

Figure 3.21.5 shows an example of UART with 16-bit NOR-Flash.



(Note1) By reset, both TXD0, RXD0 and /RTS become input port(PortF0,F1 and PC7).
If necessary, Add pull-up or pull-down resistor.
(Note2) Since PC7(/CSZF,LCP1)-pin is assigned as /RTS-output for UART, their function can not be used in BOOT-mode.

Figure 3.21.5 shows an example of UART 16-bit NOR-Flash .

2) UART interface specification

A SIO chaneel-1 is used to download.

The following shows the UART communication format in BOOT mode.

Before booting, the PC-side must also be set up in the same way.

Note that although the default baud rate is 9600 bps, it can be changed to other values as shown in Table.

Serial transfer mode	: UART(asynchronous communication)mode, full-duplex communication
Data length	: 8-bits
Parity bit	: None
STOP bit	: 1-bit
Handshake	: TMP92CH21(PortC7) -> PC
Baud rate(default)	: 9600 bps

(Notes) For the 1st-cut sample, the baud-rate can not be changed .

3) UART data transfer format

Table 3.21.5(1) to (6) show supported frequency, data transfer format, baud rate modification commands, operation commands, version management information, and frequency measurement result with data storing location, respectively.
Also refer to the description of boot-program operation in the later pages of this chapter as you read these tables.

Table3.21.5(1) Supported frequency (f_{FPH})

16.000 MHz	20.000 MHz	22.579 MHz	25.000 MHz	32.000 MHz	33.868 MHz	36.000 MHz	40.000 MHz

Table3.21.5(2) Transfer format

	Number of Bytes Transferred	Transfer Data from PC to TMP92CH21	Baud Rate	Transfer Data from TMP92CH21 to PC
BOOT ROM	1st byte 2nd byte	Matching data (5AH) —	9600 bps	— (Frequency measurement and baud rate auto set) OK: Echo-back data (5AH) NG: Nothing transmitted
	3rd byte to 6th byte	—		Version management information (See Table3.21.5(5))
	7th byte	—		Frequency information (See Table3.21.5(6))
	8th byte 9th byte	Baud rate modification command (See Table) —		— OK: Echo-back data NG: Error code X 3
	10th byte to n'th -4 byte	User-program Extended Intel Hex format(binary)	Changed new baud rate	NG: Stop operation by checksum error
	n'th -3 byte	—		OK:SUM(High) (See (6) (iii) Notes on SUM)
	n'th -2 byte	—		OK:SUM(Low)
	n'th -1 byte n'th byte	User-program start command (C0H) (See Table3.21.5(4)) —		— OK: Echo-back data (C0H) NG: Error code X 3
RAM	—	JUMP to user-program start address		

"Error code X 3" means sending an error code three times. For example, when error code is 62H, TMP92CH21 sends 62H three times. About error code, see (6)(ii) Error Code.

Table3.21.5(3) Baud rate modification command

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

(Note1) When f_{OSCH} (oscillation frequency) is either 16.00,20.00,20.58 or 25MHz, a 115200bps baud-rate is not supported.

(Notes2) For the 1st-cut sample, the baud-rate can not be changed .

Table3.21.5(4) Operation command

Operation command	Operation
C0H	Start user-program

Table3.21.5(5) Version management information

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

Table3.21.5(6) Frequency measurement result data

Frequency of oscillator (MHz)	16.000	20.000	22.579	25.000	32.000	33.868	36.000	40.000
2000H (RAM store address)	00H	01H	02H	03H	04H	05H	06H	07H

4) Description of UART boot-program operation

The boot-program transfers the data sent from PC in extended intel-hex format into the internal RAM. When the transfer has terminated normally, it calculates the SUM and sends the result to PC before starting to execute an user-program. The execution start address is the received first address. This boot-program provides the user's own way to control on-board programming.

a) Operational procedure

1. Connect the serial cable. Make sure to perform connection before resetting the TMP92CH21.
2. Set both AM1 and AM0 pins to "1" and reset TMP92CH21.
3. The received data in the 1st byte means the matching data. When the boot-program starts in BOOT mode, it goes to a state in which it waits for the matching data to receive. Upon receiving the matching data, it automatically adjusts the serial channels' initial baud rate to 9600 bps. The matching data is 5AH.
4. The 2nd byte is used to echo back 5AH to PC upon completion of the automatic baud rate setting in the first byte. If the device fails in automatic baud rate setting, it goes to an idle state.
5. The 3rd byte to 6th byte 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 of 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 the baud rate modification data. The five kinds of baud rate modification data shown in Table are available. Even when you do not change the baud rate, be sure to send the initial baud rate data (28H;9600 bps). 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 controller when the data received in the 8th byte is one of the baud rate modification data corresponding to the device's operating frequency. Then the baud rate is changed. If the received baud rate data does not correspond to the device's operating frequency, the device goes to an idle state after sending 3 bytes of baud rate modification error code (62H).
9. The receive data in the 10th byte through n'th – 4 byte is received as binary data in Extended Intel Hex format. No received data is echoed back to the controller.
   The boot-program processing routine ignores the received data until it receives the start mark (3AH for ":") in extended intel-hex format. Nor does it send error code to the controller. After receiving the start mark, the routine receives a range of data from the data length to checksum and writes the received data to the specified RAM addresses successively.
   After receiving one record of data from start mark to checksum, the routine goes to a start mark waiting state again.
   If a receive error or checksum error of extended hex format occurs, the device goes to an idle state without returning error code to the controller.
   Because the boot-program processing routine executes a SUM calculation routine upon detecting the end record, the controller should be placed in a SUM waiting state after sending the end record to the device.
10. The n'th – 3 byte and the n'th – 2 byte are the SUM value that is sent to PC in order of upper byte and lower byte. For details on how to calculate the SUM, refer to "Notes on SUM" in the later page of this chapter. The SUM calculation is performed only when no write error, receive error, or extended intel-hex format error has been encountered after detecting the end record. Soon after calculation of SUM, the device sends the SUM data to the controller. The controller should determine whether writing to the RAM has terminated normally depending on whether the SUM value is received after sending the end record to the device.
11. After sending the SUM, the device goes to a state waiting for the user-program start code. If the SUM value is correct, the controller should send the user-program start command to the n'th – 1 byte. The user-program start command is C0H.
12. The n'th byte is used to echo back the user-program start code to the controller. After sending the echo-back to the controller, the stack pointer is set to 5FFFH and the boot-program jumps to the first address that is received as data in extended intel-hex format.
13. If the user-program start code is wrong or a receive error occurs, the device goes to an idle state after returning three bytes of error code to PC.

b) Error code

The boot-program sends the processing status to PC using various code. The error code is listed in the table below.

Table3.21.5(7) Error code

Error Code	Meaning of Error Code
62H	Baud rate modification error occurred.
64H	Operation command error occurred.
A1H	Framing error in received data occurred.
A3H	Overrun error in received data occurred.

*1: When a receive error occurs when receiving the user-program, the device does not send the error code to PC.

*2: After sending the error code, the device goes to an idle state.

c) Notes on SUM

1. Calculation method

SUM consists of byte+byte…..+byte , the sum of which is returned in word as the result. Namely, data is read out in byte and sum of which is calculated, with the result returned in word.

Example:

A1H
B2H
C3H
D4H

If the data to be calculated consists of the four bytes shown to the left, SUM of the data is:

A1H + B2H + C3H + D4H = 02EAH

SUM(HIGH) = 02H

SUM(LOW) = EAH

2. Calculation data

The data from which SUM is calculated is the RAM data from the first address received to the last address received.

The received RAM write data is not the only data to be calculated for SUM. Even when the received addresses are noncontiguous and there are some unwritten areas, data in the entire memory area is calculated. The user-program should not contain unwritten gaps.

d) Notes on extended intel-hex format (binary)

1. After receiving the checksum of a record, the device waits for the start mark (3AH for ":") of the next record. Therefore, the device ignores all data received between records during that time unless the data is 3AH.
2. Make sure that once the controller program has finished sending the checksum of the end record, it does not send anything and waits for two byes of data to be received (upper and lower bytes of SUM). This is because after receiving the checksum of the end record, the boot-program calculates the SUM and returns the calculated SUM in two bytes to the controller.
3. It becomes the cause of incorrect operation to write to areas out of device's internal RAM. Therefore, when an extended record is transmitted, be sure to set a paragraph address to 0000H.
4. Always make sure the first record type is an extended record. Because the initial value of the address pointer is 00H.
5. Address 3000H to 57FFH are assigned to load user-program. So it must be less than 10Kbyte.
6. Beforehand, convert an user-program of intel-hex format from ASCII-code to Binary-format.

   The below is an example to explain how to make binary-format file.

(Example) How to convert from extended intel-HEX format file to Binary-format file

The following is an example of display in text-editor in case of intel-HEX format file is opened.

:103000000607F100030000F201030000B1F16010B7

:00000001FF

But actually their data are below because of ASCII-code is used for intel-HEX format file.

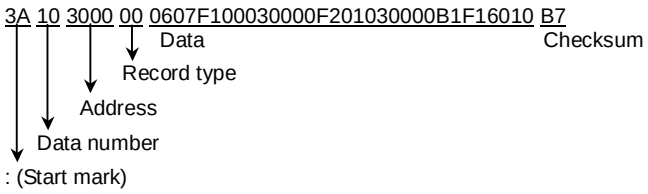
3A313033303030303030363037463130303033303030304632303130333030303042
3146313630313042370D0A3A3030303030303031464 60D0A

Convert above data with below table's law.

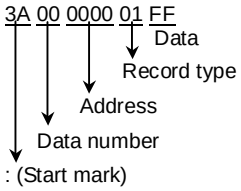
Before (ASCII)	After (Binary)
3A	3A（Only 3A should not be converted.）
30 to 39	0 to 9
41or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete it

The intel-HEX format and its meaning are below.

Data record



End record



e) Error when receiving user-program

If the following errors occur in extended intel-hex format when receiving the user-program, the device goes to an idle state.

- When the record type is not 00H, 01H, 02H
- When a checksum error occurs

f) Error between frequency measurement and baud rate

The boot-program measures the resonator frequency when receiving matching data. If an error is under 3%, the boot-program decides on that frequency. Since there is an overlap between the margin of 3% for 32.000 MHz and 33.868 MHz, the boundary is set at the intermediate value between the two. The baud rate is set based on the measured frequency.

Each baud rate includes a set error shown in Table 3.21.5(8)3.21.5(8).

For example, in the case of 20.000 MHz and 9600 bps, the baud rate is actually set at 9615.38 bps with an error of 0.2%. To establish communication, the sum of the baud rate set error shown in Table 3.21.5(8) and the frequency error need to be under 3%.

Table 3.21.5(8) Set error of each baud rate (%)

	9600 bps	19200 bps	38400 bps	57600 bps	115200 bps
16.000 MHz	0.2	0.2	0.2	–0.8	Not supported
20.000 MHz	0.2	0.2	0.2	1.0	
22.579 MHz	-0.7	-0.7	-0.7	0.1	
25.000 MHz	0.5	-0.8	0.5	0.5	
32.000 MHz	0.2	0.2	0.2	0.7	-0.8
33.868 MHz	0.3	0.3	0.3	-0.7	-0.7
36.000 MHz	0.2	-0.7	0.2	0.2	0.2
40.000 MHz	0.2	0.2	0.2	-0.3	1.0

(Notes) For the 1st-cut sample, the baud-rate error is below table.

	9600 bps
16.000 MHz	0.2
20.000 MHz	0.2
22.579 MHz	–0.7
25.000 MHz	–0.8
32.000 MHz	0.2
33.868 MHz	0.3
36.000 MHz	0.2
40.000 MHz	0.2

5) Port setting

Only ports shown in Table 3.21.4(2)3.21.5(9) are set up in the boot-program.

Another ports are not set, so they are kept after reset status.

It is needed to care these ports to design an application system.

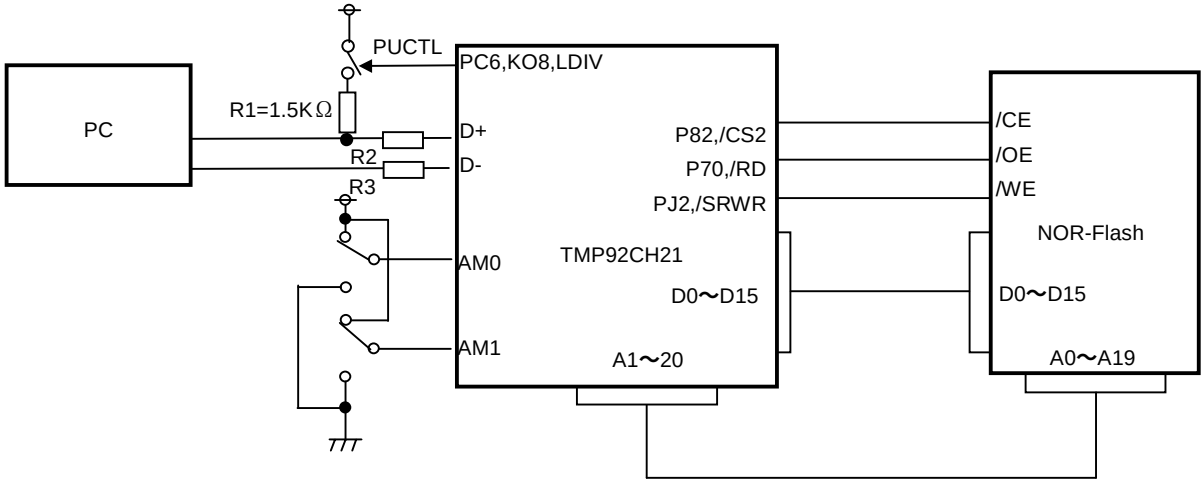
Table 3.21.5(9) Settled Ports

Pin name	After reset in BOOT-mode	Settled function	Note
P70(/RD)	Output port	/RD output	
P71(/NDRE,/WRLL)	Input port	/NDRE output	
P72(/NDWE,/WRLH)	Input port	/NDWE output	
P75(R/B,R/W)	Input port	R/B input	
P82(/CS2,/CSZA,/SDCS)	Output port	/CS2	
P84(/ND0CE,/CSZB,/WRUL)	Output port	/ND0CE	
P86(/SRULB,/CSZD)	Output port	/SRULB	
P87(/SRUUB,/CSZE)	Output port	/SRUUB	
PJ5(NDALE,SDULDQM)	Output port	NDALE	
PJ6(NDCLE,SDUUDQM)	Output port	NDCLE	
PJ5(NDALE,SDULDQM)	Output port	NDALE	
PC7(/CSZF,LCP1)	Input port	Output port for /RTS	
PF0(TXD1,TXD0)	Input port	TXD1 output	Not open-drain port. This port becomes TXD1 after matching data received.
PF1(RXD1,RXD0)	Input port	RXD1 input	

3.21.6 Download with USB

1) Connection example

Figure 3.21.6(1) shows an example of USB with 16-bit NOR-Flash.



(Note1) By reset, PC6 become input port. If necessary, Add pull-up or pull-down resistor.

(Note2) Since PC6(KO8,LDIV)-pin is assigned as PUCTL(Pull-up Control) output for USB, their function can not be used in BOOT-mode.

Figure 3.21.6(1) Example of USB with 16-bit NOR-Flash .

3.22 PSB (Power Supply Backup)

The power supply input of TMP92CH21 is divided into three systems as follows;

Analog Power supply input (AVCC - AVSS)

Digital Power supply input (DVCC - DVSS)

Digital power supply input for RTC (RTCVCC - DVSS)

The individual power supply input is isolated from each other.

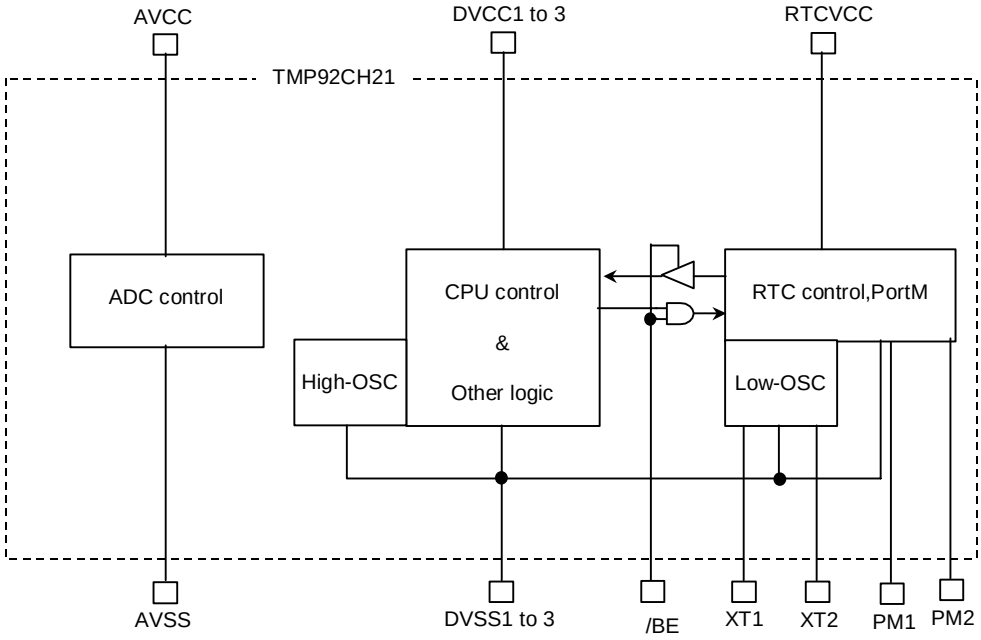


Figure 3.22 (1) Power supply input system

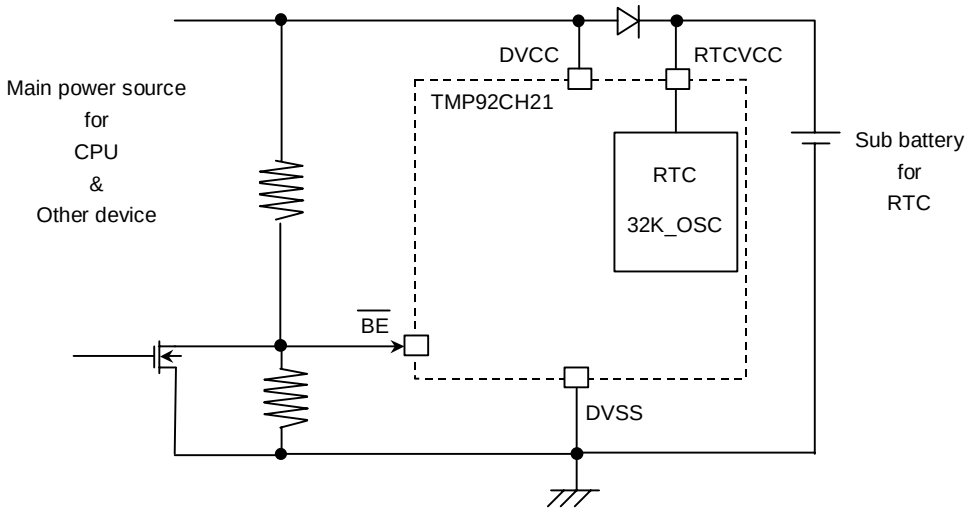


Figure 3.22 (2) Outside circuit example for PSB

The TMP92CH21 has the power supply backup mode which is designed to work for only RTC under sub battery supply. TMP92CH21 enters the power supply backup mode using the /BE(Backup enable signal Pin) and the /RESET.

Figure 3.22 (3), (4) shows the timing diagram of /BE and /RESET.

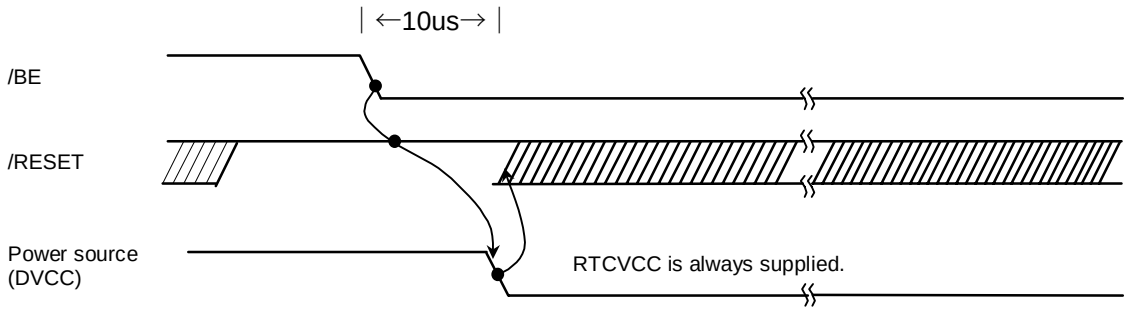


Figure 3.22 (3) Normal Mode to PSB Mode

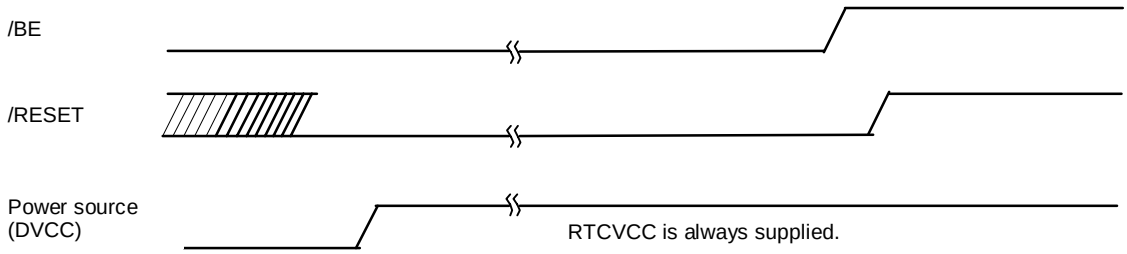


Figure 3.22 (4) Normal Mode from PSB Mode

Backup enable pin (/BE)

RTC can work under dedicated power supply pin which is named as /BE. The /BE pin can inhibit access to RTC registers.

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Power Supply Voltage	V_{CC}	-0.5 to 4.0	V
Input Voltage	V_{IN}	-0.5～VCC+0.5	V
Output Current	IOL	2	mA
Output Current	IOH	-2	mA
Output Current(total)	ΣI_{OL}	80	mA
Output Current(total)	ΣI_{OH}	-80	mA
Power Dissipation(Ta=85℃)	P_D	600	mW
Soldering Temperature(10s)	T_{SOLDER}	260	℃
Storage Temperature	T_{STG}	-65～150	℃
Operation Temperature	T_{OPR}	-20～70	℃

Note: The absolute maximum ratings are rated values that must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, 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 absolute maximum rating value will ever be exceeded.

4.2 DC Electrical Characteristics

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

Symbol	Parameter	Min	Typ.	Max	Unit	Condition
VCC	Power Supply Voltage (DVCC=AVCC=RTCVCC) (DVSS=AVSS=0V)	3.0		3.6	V	X1=8 to 40MHz (Internal 8 to 20MHz) XT1=30 to 34KHz
VIL0	Input Low Voltage for D0 to D7 P10 to P17(D8 to 15) P20 to P27(D16 to 23) P30 to P37(D24 to 31)	-0.3		0.6	V	
VIL1	Input Low Voltage for P40 to P47, P50 to P57, P60 to P67, P71 to P72, P75 to P76, P90, P93 to P94, PC6 to PC7, PF0, PG0 to PG3, PJ5 to PJ6, PL4 to PL7			0.3*VCC		
VIL2	Input Low Voltage for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PF0 to PF2, /BE,/RESET			0.25*VCC		
VIL3	Input Low Voltage for AM0 to AM1			0.3		
VIL4	Input Low Voltage for X1,XT1			0.2*VCC		
VIH0	Input High Voltage for D0 to D7 P10 to P17(D8 to 15) P20 to P27(D16 to 23) P30 to P37(D24 to 31)	2.0		VCC+0.3	V	
VIH1	Input High Voltage for P40 to P47, P50 to P57, P60 to P67, P71 to P72, P75 to P76, P90, P93 to P94, PC6 to PC7, PF0, PG0 to PG3, PJ5 to PJ6, PL4 to PL7	0.7*VCC				
VIH2	Input High Voltage for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PF0 to PF2, /BE,/RESET	0.75*VCC				
VIH3	Input Low Voltage for AM0 to AM1	VCC-0.3				
VIH4	Input Low Voltage for X1, XT1	0.8*VCC				

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

Symbol	Parameter	Min	Typ.	Max	Unit	Condition
VOL	Output Low Voltage			0.45	V	IOL = 1.6mA
VOH1	Output High Voltage	2.4			V	IOH = -400μA
VOH2		0.75*VCC			V	IOH = -100μA
VOH3		0.9*VCC			V	IOH = -20μA
ILI	Input Leakage Current		0.02	±5	μA	0.0 ≦ Vin ≦ VCC
ILO	Output Leakage Current		0.05	±10	μA	0.2 ≦ Vin ≦ VCC-0.2
VSTOP	Power Down Voltage at STOP (for internal RAM back-up)	1.8		3.6	V	VIL2=0.2*Vcc, VIH2=0.8*Vcc
RRST	Pull Up Resistor for /RESET, PA0 to PA7	80		400	KΩ	
RKH	Programmable Pull Down Resistor for P96					
CIO	Pin Capacitance			10	pF	fc=1MHz
VTH	Schmitt Width for P91 to P92, P96 to P97, PA0 to PA7,PC0 to PC3, PF0 to PF2,/BE,/RESET	0.4	1.0		V	
ICC	NORMAL (note2)		TBD	TBD	mA	VCC=3.6V , fc=40MHz
	IDLE2		TBD	TBD		
	IDLE1		TBD	TBD		
	SLOW (note2)		TBD	TBD	μA	VCC=3.6V, fs=32KHz
	IDLE2		TBD	TBD		
	IDLE1		TBD	TBD		
	STOP		TBD	TBD		VCC=3.6V

(note1): Typical values are for when Ta = 25°C and Vcc = 3.3 V unless otherwise noted.

(note2): ICC measurement conditions (NORMAL, SLOW):

All functions are operational; output pins are open and input pins are fixed. Data & address bus CL=30pF loaded.

4.3 AC Characteristics

4.3.1 Basic Bus Cycle

Read cycle

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

No.	Parameter	Symbol	Min	Max	40MHz	36MHz	Unit
1	OSC period (X1/X2)	t_{OSC}	25	31.25	25	27.7	ns
2	System Clock period (=T)	t_{CYC}	50	62.5	50	55.5	
3	SDCLK Low Width	t_{CL}	0.5T-15		10	12.7	
4	SDCLK High Width	t_{CH}	0.5T-15		10	12.7	
5-1	A0 to A23 Valid → D0 to D31 Input at 0WAIT	t_{AD}		2.0T-30	70	81	
5-2	A0 to A23 Valid → D0 to D31 Input at 1WAIT	t_{AD3}		3.0T-30	120	136.5	
6-1	/RD Fall → D0 to D31 Input at 0WAIT	t_{RD}	———	1.5T-30	45	53.3	
6-2	/RD Fall → D0 to D31 Input at 1WAIT	t_{RD3}		2.5T-30	95	108.8	
7-1	/RD Low Width at 0WAIT	t_{RR}	1.5T-20		55	63	
7-2	/RD Low Width at 1WAIT	t_{RR3}	2.5T-20		105	118.8	
8	A0 to A23 Valid → /RD Fall	t_{AR}	0.5T-20		5	7.7	
9	/RD Fall → SDCLK Rise	t_{RK}	0.5T-20		5	7.7	
10	A0 to A23 Valid → D0 to D31 Hold	t_{HA}	0		0	0	
11	/RD Rise → D0 to D31 Hold	t_{HR}	0		0	0	
12	/WAIT Set-up Time	t_{TK}	15		15	15	
13	/WAIT Hold Time	t_{KT}	5		5	5	
14	Data Byte Control Access Time for SRAM	t_{SBA}		1.5T-30	45	53.3	

Write cycle

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

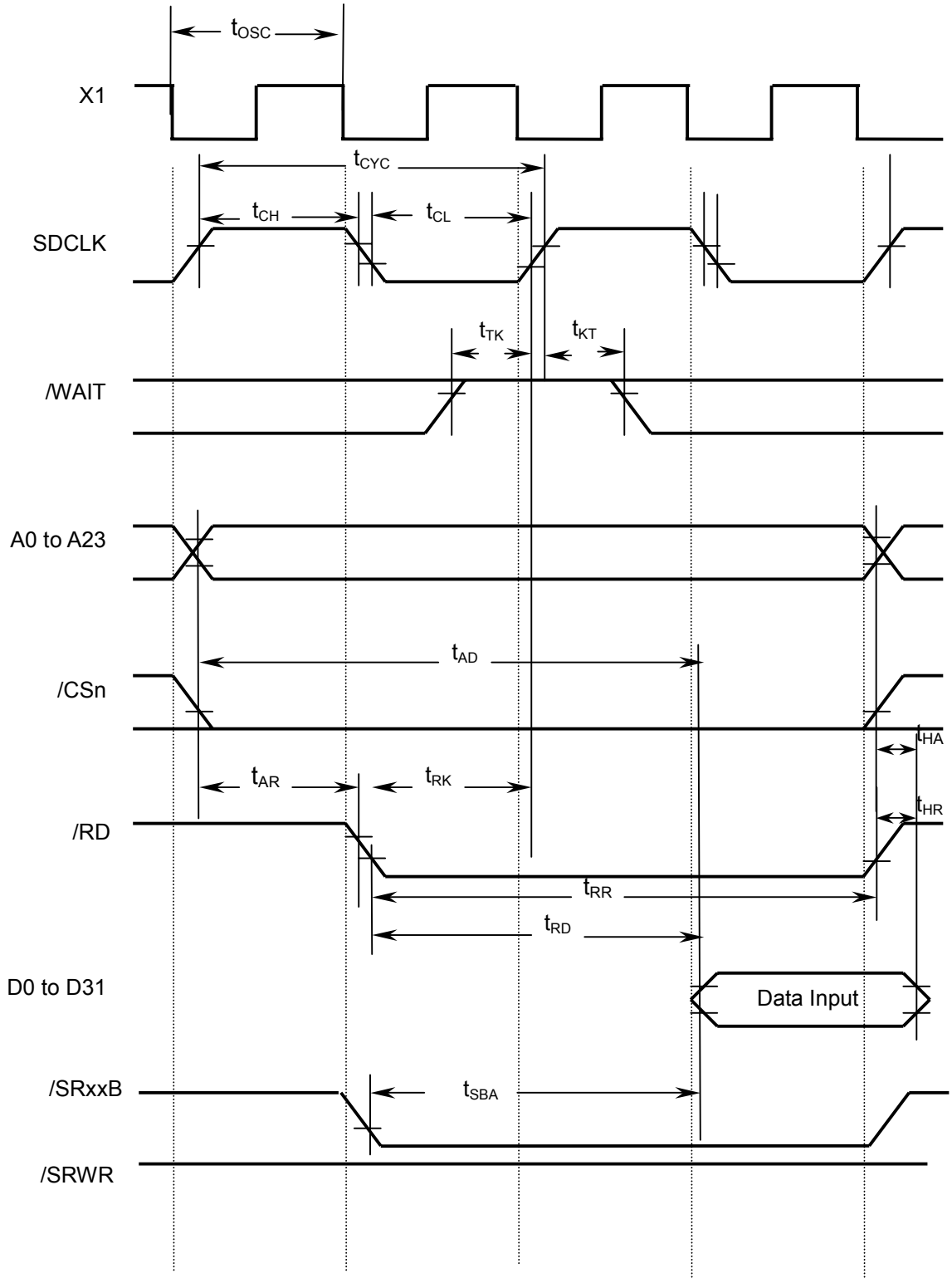
No.	Parameter	Symbol	Min	Max	40MHz	36MHz	Unit
1	OSC period (X1/X2)	t_{OSC}	25	31.25	25	27.7	ns
2	System Clock period (=T)	t_{CYC}	50	62.5	50	55.5	
3	SDCLK Low Width	t_{CL}	0.5T-15		10	12.7	
4	SDCLK High Width	t_{CH}	0.5T-15		10	12.7	
5-1	D0 to D31 Valid → /WRxx Rise at 0WAIT	t_{DW}	1.25T-35		28	34.3	
5-2	D0 to D31 Valid → /WRxx Rise at 1WAIT	t_{DW3}	2.25T-35		78	89.8	
6-1	/WRxx Low Width at 0WAIT	t_{WW}	1.25T-30		33	34.3	
6-2	/WRxx Low Width at 1WAIT	t_{WW3}	2.25T-30		83	89.8	
7	A0 to A23 Valid → /WR Fall	t_{AW}	0.5T-20		5	7.7	
8	/WRxx Fall → SDCLK Rise	t_{WK}	0.5T-20		5	7.7	
9	/WRxx Rise → A0 to A23 Hold	t_{WA}	0.25T-5		8	8.8	
10	/WRxx Rise → D0 to D31 Hold	t_{WD}	0.25T-5		8	8.8	
11	/WAIT Set-up Time	t_{TK}	15		15	15	
12	/WAIT Hold Time	t_{KT}	5		5	5	
13	/RD Rise → D0 to D31 Output	t_{RDO}	0.5T-5		20	22.7	
14	Write Pulse Width for SRAM	t_{SWP}	1.25T-30		33	34.3	
15	Data Byte Control to End of Write for SRAM	t_{SBW}	1.25T-30		33	34.3	
16	Address Setup Time for SRAM	t_{SAS}	0.5T-20		5	7.7	
17	Write Recovery Time for SRAM	t_{SWR}	0.25T-5		8	8.8	
18	Data Setup Time for SRAM	t_{SDS}	1.25T-35		28	34.3	
19	Data Hold Time for SRAM	t_{SDH}	0.25T-5		8	8.8	

AC Condition

●Output ： High = 0.7Vcc, Low = 0.3Vcc, CL = 50pF

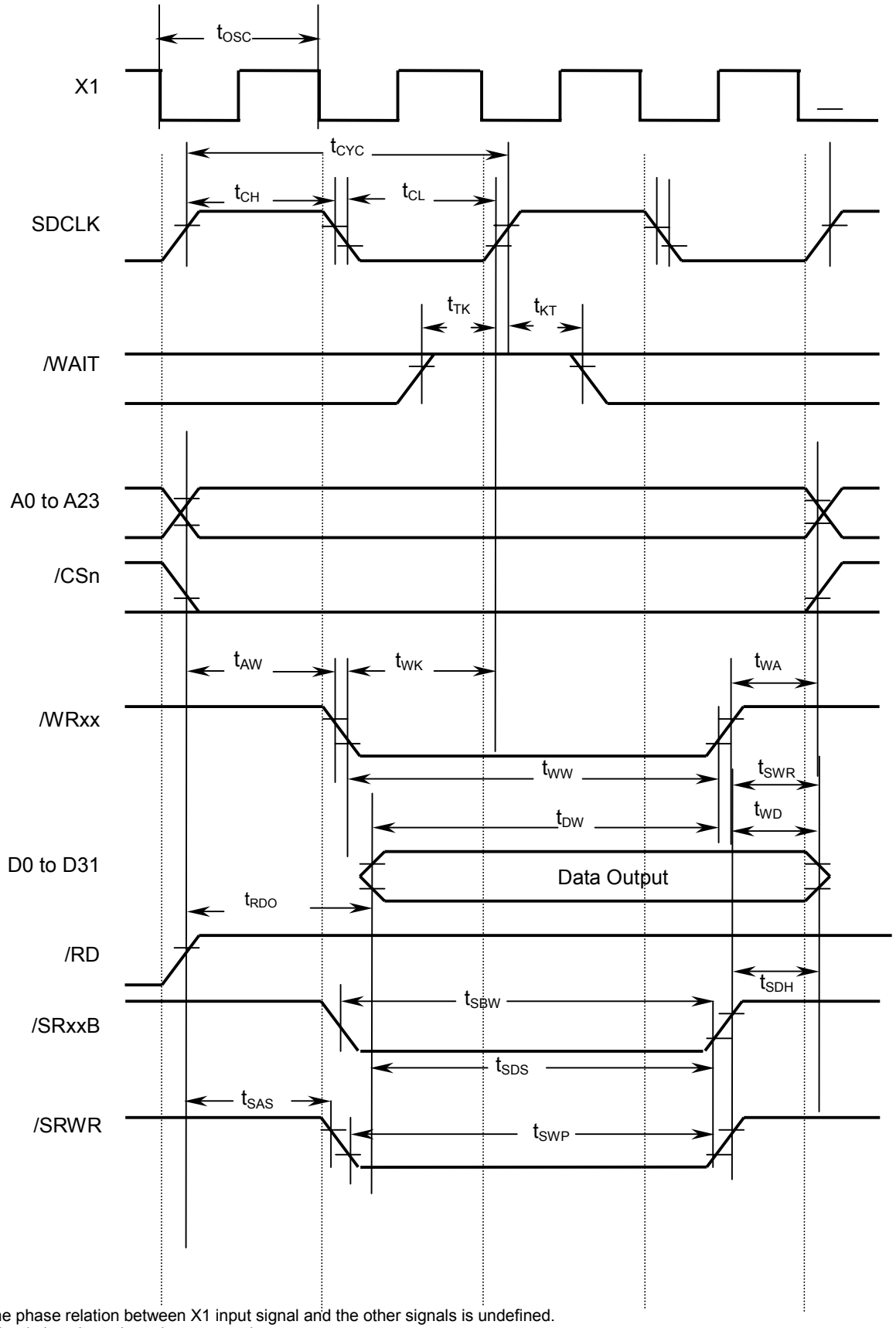
●Input ： High = 0.9Vcc, Low = 0.1Vcc

(1) Read cycle (0 wait)



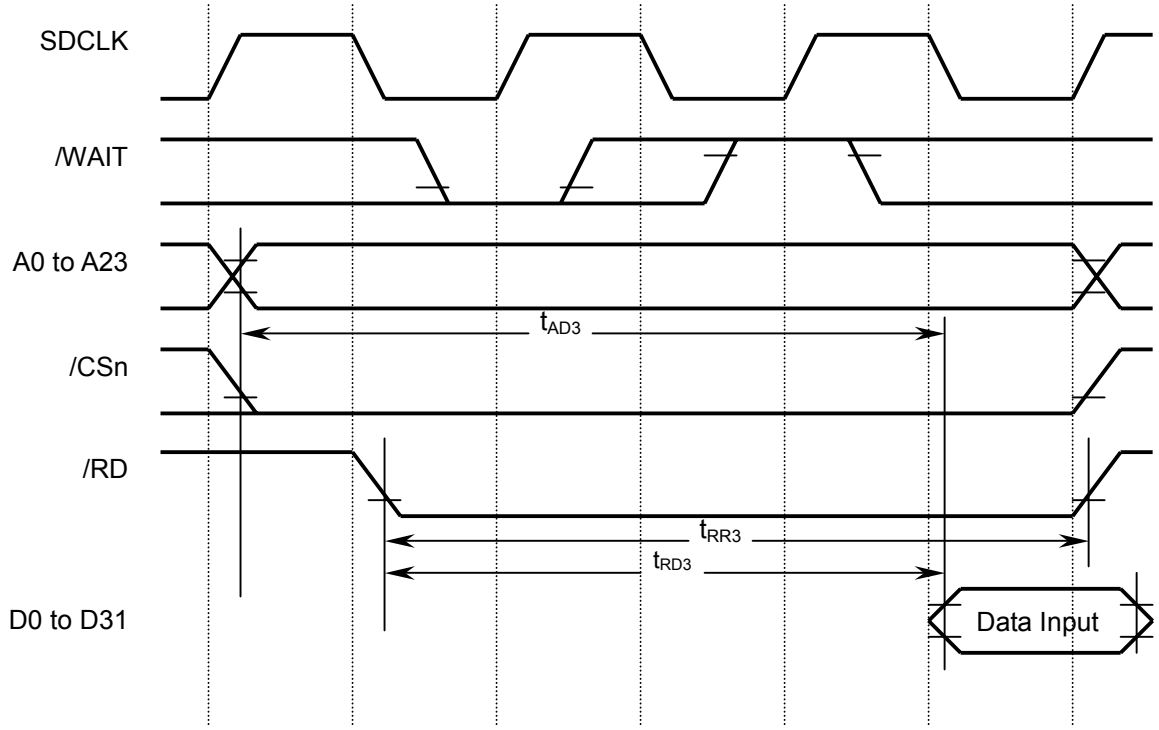
Note: The phase relation between X1 input signal and the other signals is undefined.
The timing chart above is an example.

(2) Write cycle (0 wait)

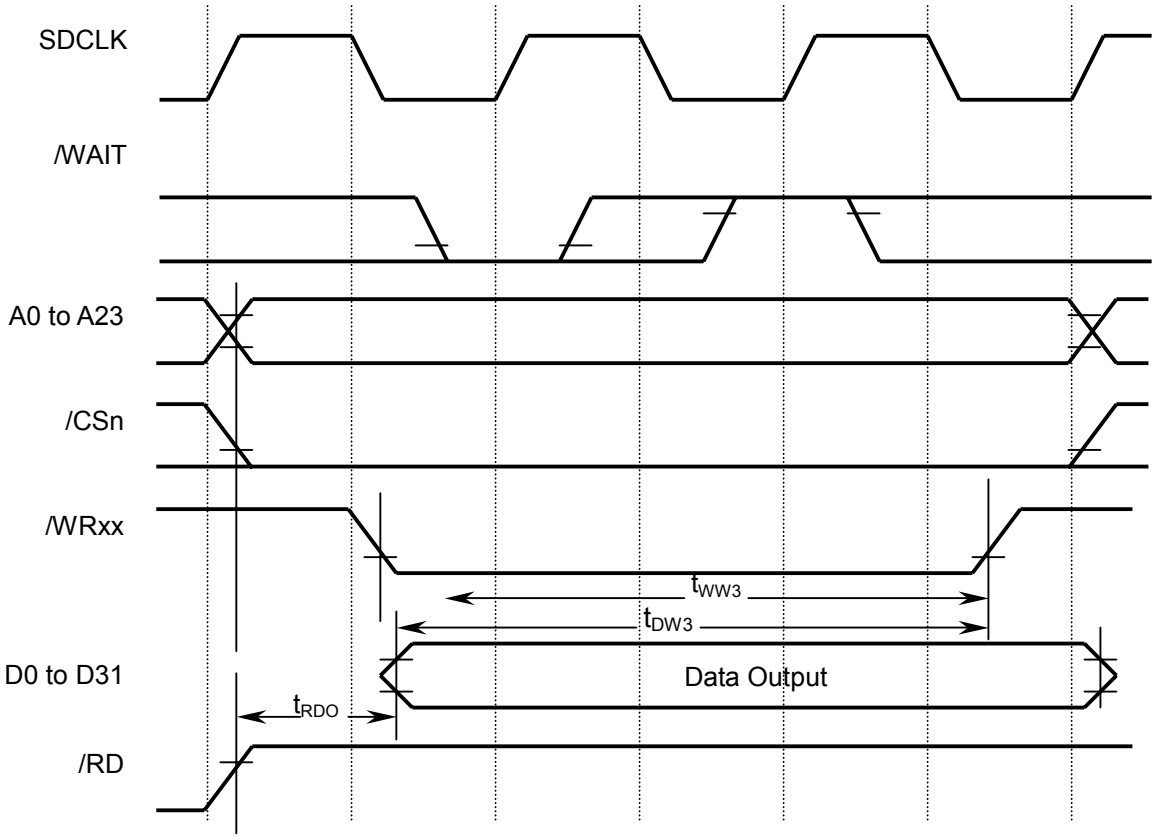


Note: The phase relation between X1 input signal and the other signals is undefined.
The timing chart above is an example.

(3) Read cycle (1 wait)



(4) Write cycle (1 wait)



4.3.2 Page ROM Read Cycle

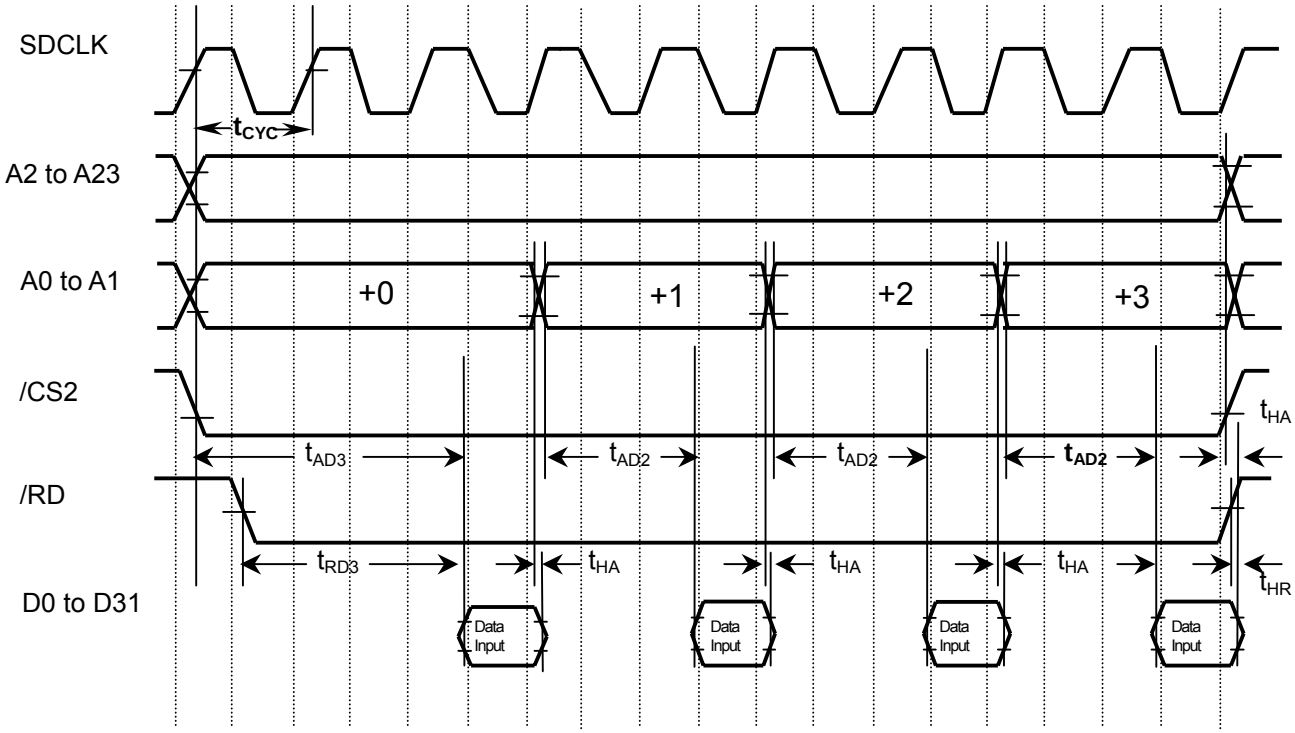
(1)3-2-2-2 mode

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70°C

No.	Symbol	Parameter	MIN	MAX	40MHz	36MHz	Unit
1	t_{CYC}	System Clock Period (=T)	50	62.5	50	55	ns
2	t_{AD2}	A0,A1 → D0 to D31 Input		2.0T-50	50	60	
3	t_{AD3}	A2 to A23 → D0 to D31 Input		3.0T-50	100	116	
4	t_{RD3}	RD fall → D0 to D31 Input		2.5T-45	80	93	
5	t_{HA}	A0 to A23 Invalid → D0 to D31 Hold	0		0	0	
6	t_{HR}	/RD rise → D0 to D31 Hold	0		0	0	

AC Condition
- ●Output：High = 0.7Vcc, Low = 0.3Vcc, CL = 50pF
- ●Input： High = 0.9Vcc, Low = 0.1Vcc

(1) Page ROM Read Cycle (3-2-2-2 mode)



4.3.3 SDRAM Controller AC Characteristics

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

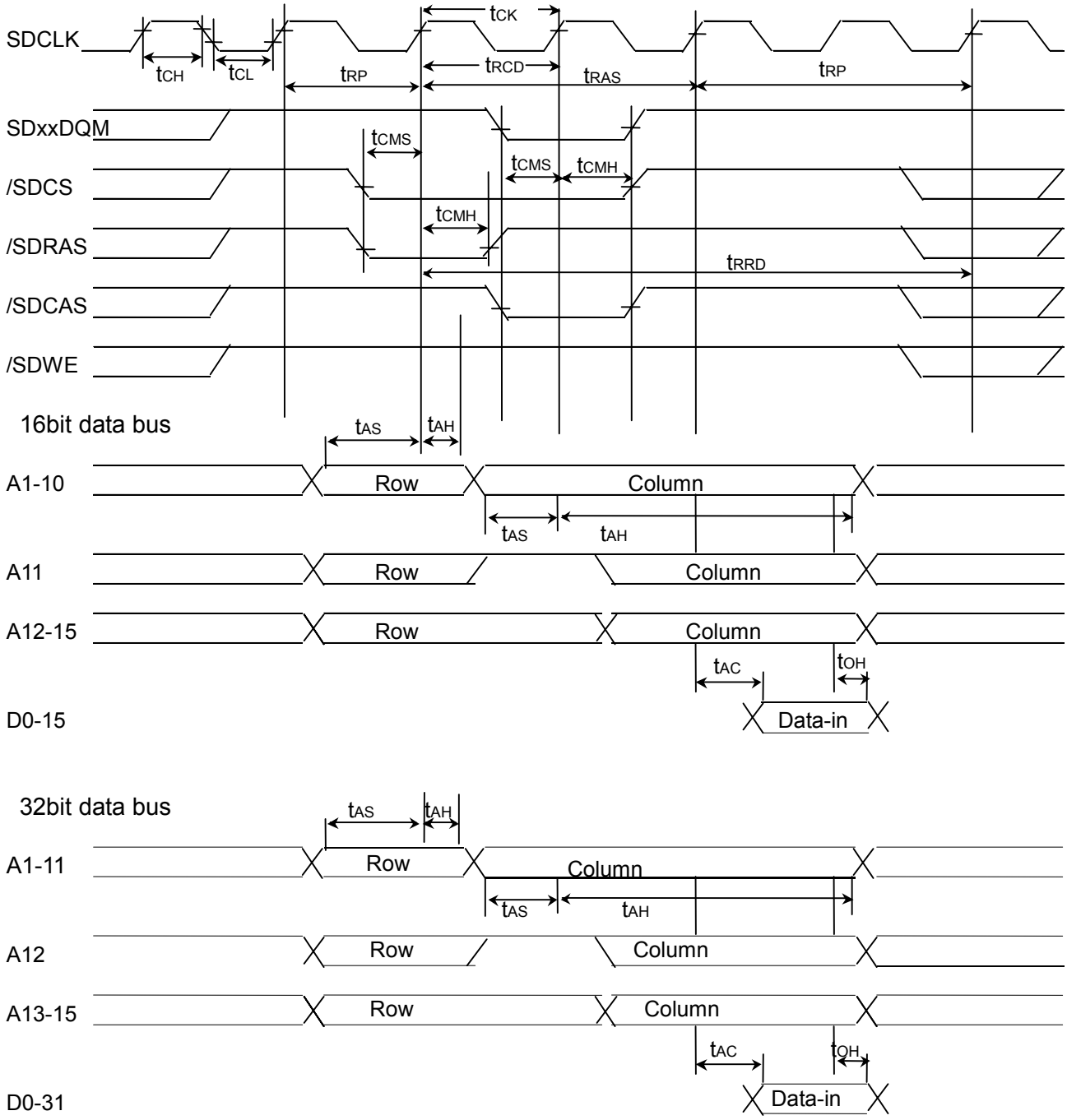
No	Symbol	Parameter	Variable		40MHz		36MHz		Unit
			MIN	MAX	MIN	MAX	MIN	MAX	
1	t_{RC}	Ref/Active to Ref/Active Command Period	2T		100		111		ns
2	t_{RAS}	Active to Precharge Command Period	2T	12210	100	12210	111	12210	
3	t_{RCD}	Active to Read/Write Command Delay Time	T		50		55.5		
4	t_{RP}	Precharge to Active Command Period	T		50		55.5		
5	t_{RRD}	Active to Active Command Period	3T		150		166.5		
6	t_{WR}	Write Recovery Time(CL*=2)	T		50		55.5		
7	t_{CK}	CLK Cycle Time(CL*=2)	T		50		55.5		
8	t_{CH}	CLK High Level Width	0.5T-15		10		12.7		
9	t_{CL}	CLK Low Level Width	0.5T-15		10		12.7		
10	t_{AC}	Access Time from CLK(CL*=2)		T-30		20		25.5	
11	t_{OH}	Output Data Hold Time	0		0		0		
12	t_{DS}	Data-in Set-up Time	T-35		15		20.5		
13	t_{DH}	Data-in Hold Time	T-5		45		50.5		
14	t_{AS}	Address Set-up Time	0.75T-35		2.5		6.6		
15	t_{AH}	Address Hold Time	3		3		3		
16	t_{CKS}	CKE Set-up Time	0.5T-15		10		12.7		
17	t_{CMS}	Command Set-up Time	0.5T-15		10		12.7		
18	t_{CMH}	Command Hold Time	0.5T-15		10		12.7		
19	t_{RSC}	Mode Register Set Cycle Time	T		50		55.5		

*CL is CAS Latency.

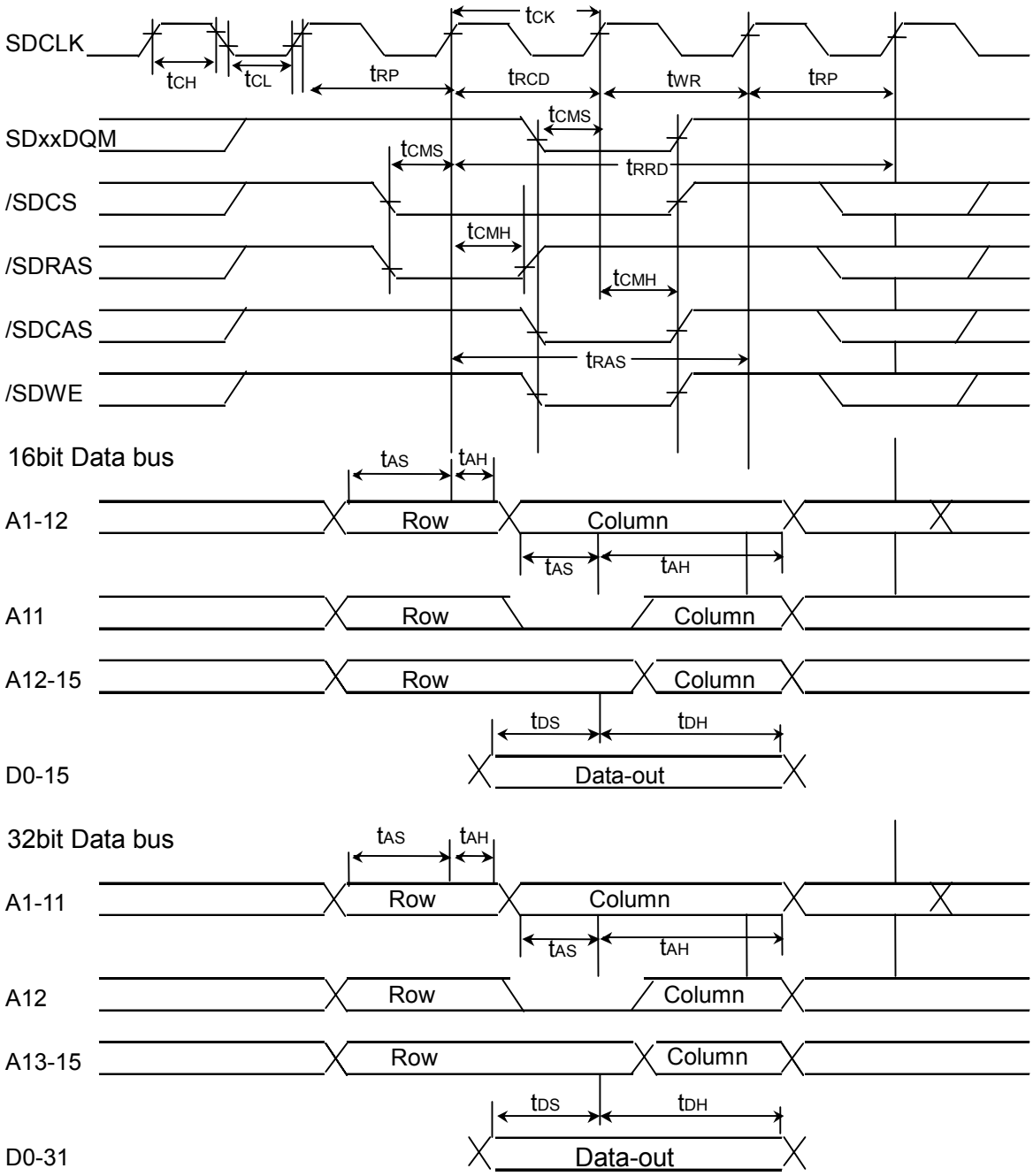
AC Measuring Conditions

- Output Level : High = 0.7 Vcc, Low = 0.3 Vcc, CL = 50 pF
- Input Level : High = 0.9 Vcc, Low = 0.1 Vcc.

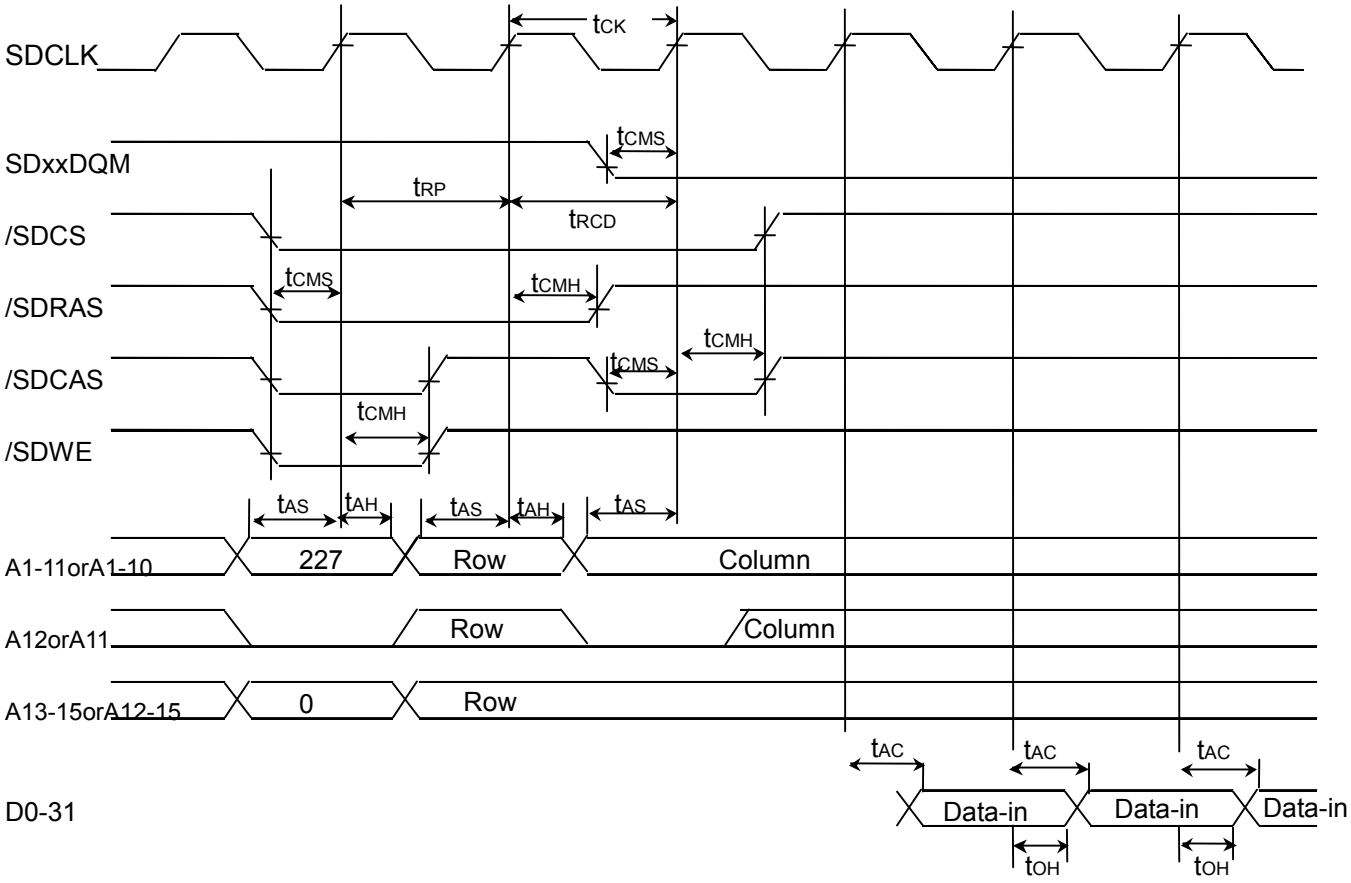
(1) SDRAM Read Timing (CPU Access or LCDC Normal Access)



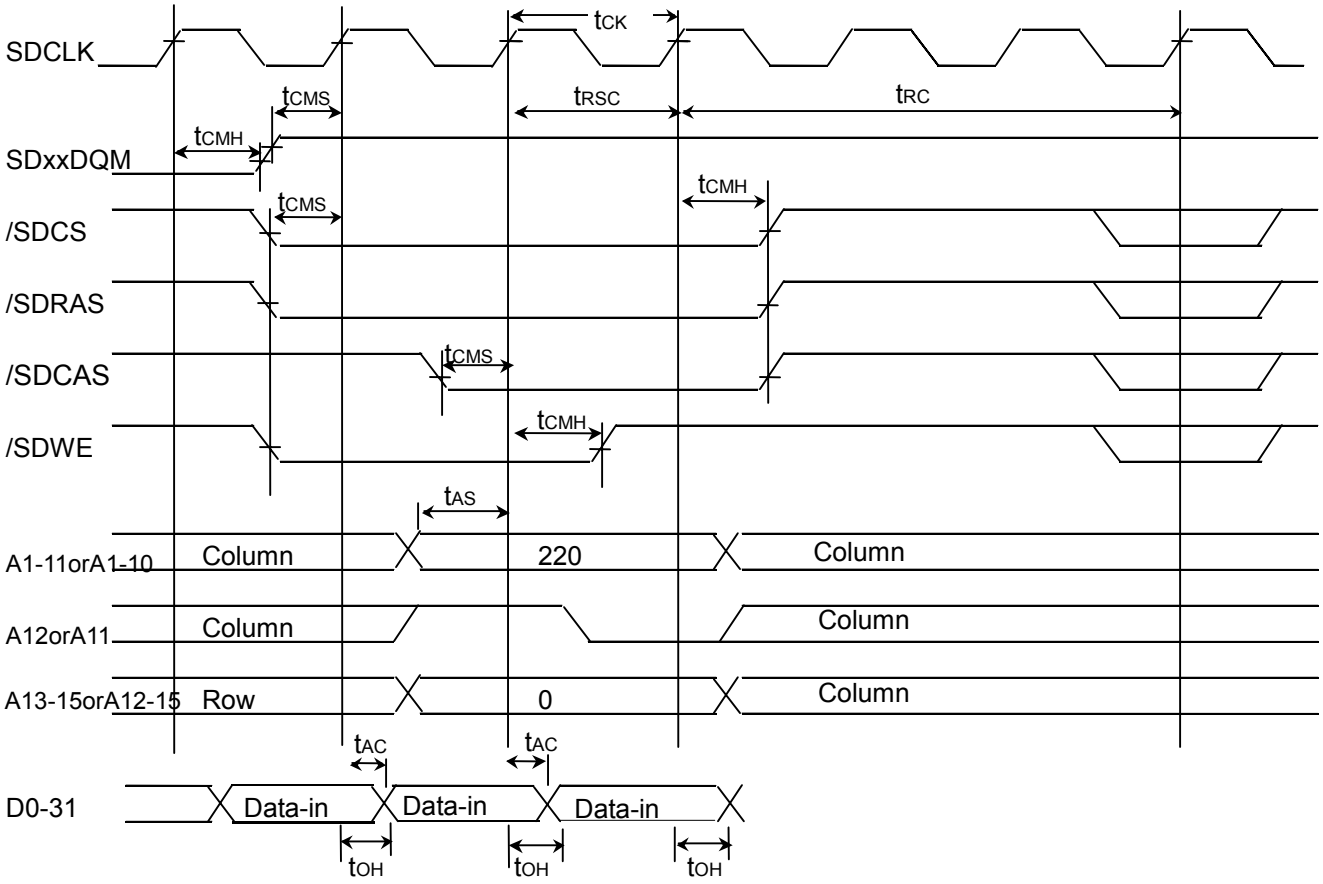
(2) SDRAM Write Timing (CPU Access)



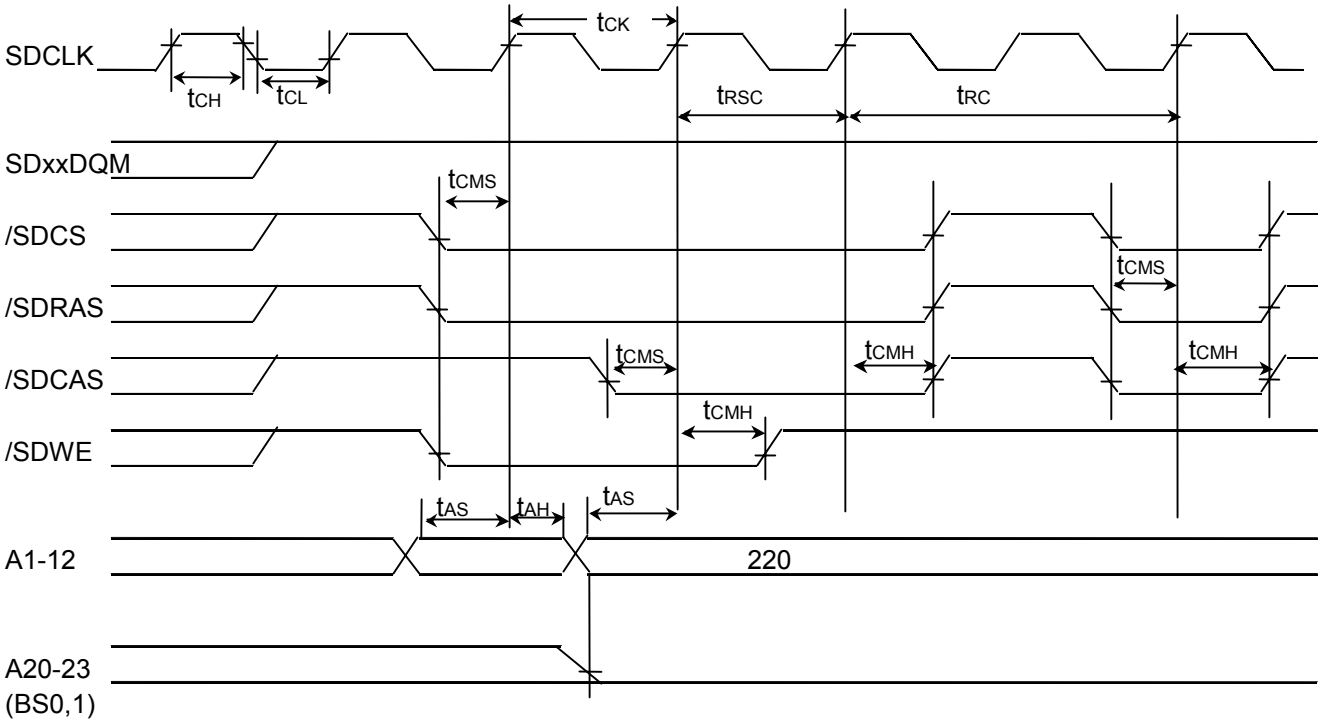
(3) SDRAM Burst Read Timing (Start of Burst Cycle)



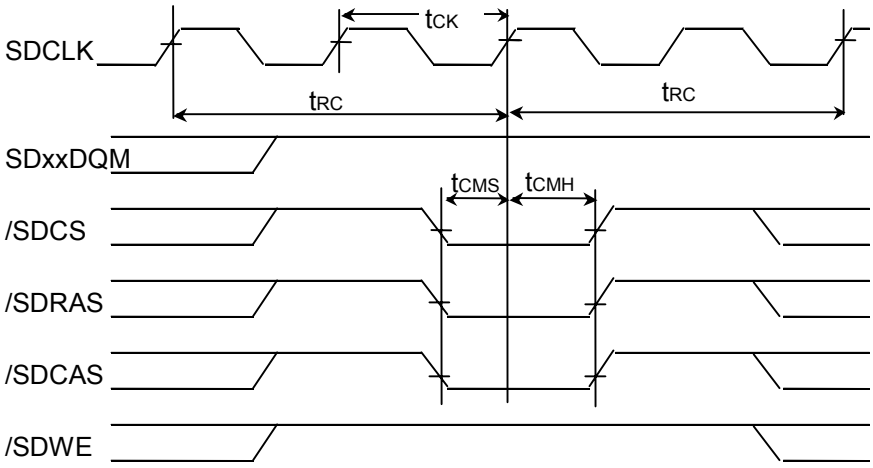
(4) SDRAM Burst Read Timing (End of Burst Cycle)



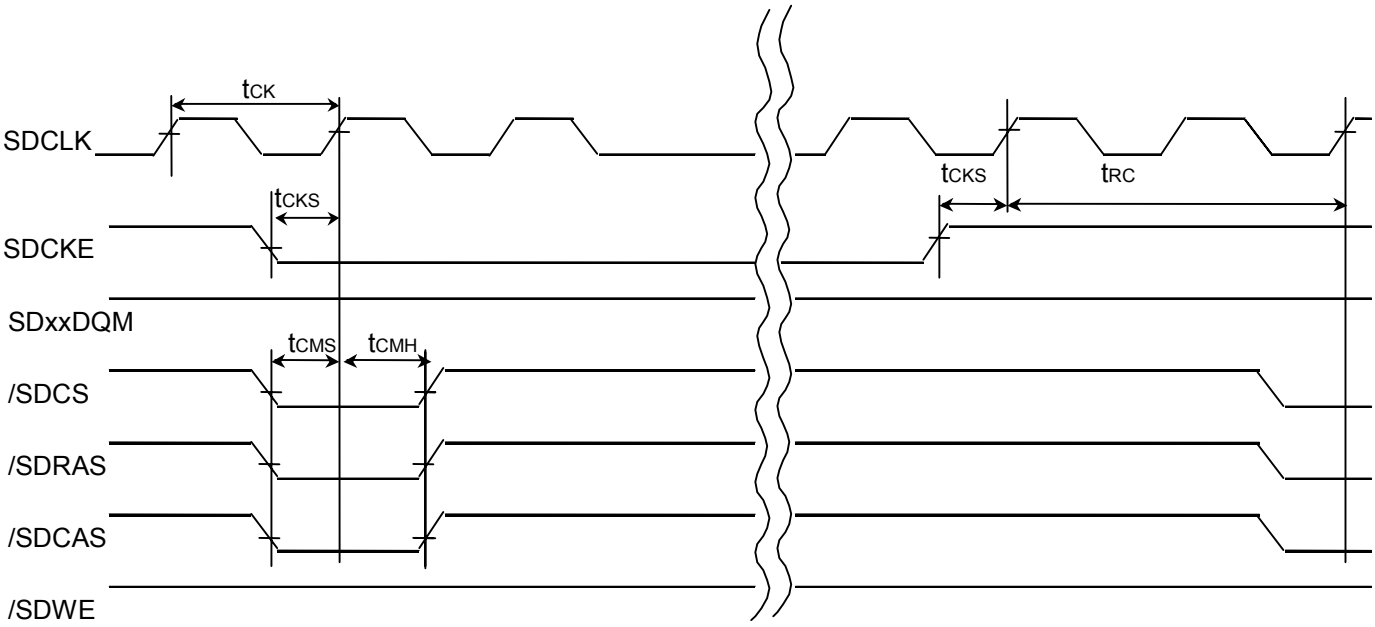
(5) SDRAM Initialize Timing



(6) SDRAM Refresh Timing



(7) SDRAM Self Refresh Timing



4.3.4 NAND-Flash Controller AC Characteristics

Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

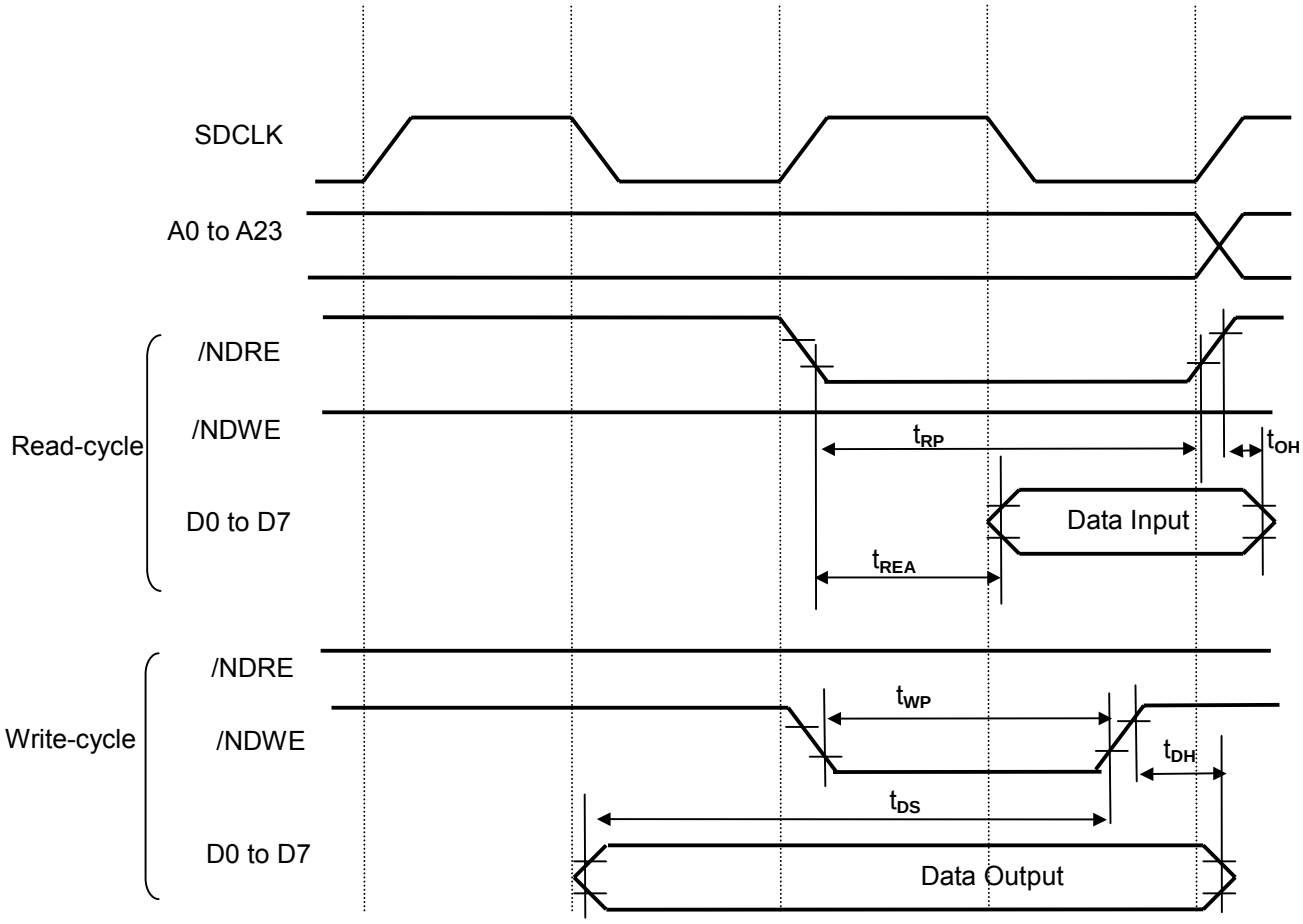
No	Symbol	Parameter	Variable		40MHz		36MHz		Unit
			min	max	min	max	min	max	
1	t_{RP}	/NDRE Low width	T-12		38		43		ns
2	t_{REA}	/NDRE Data access time		T-15		35		40	
3	t_{OH}	Read data hold time	0		0		0		
4	t_{WP}	/NDWE Low width	0.75T-12		25		29		
5	t_{DS}	Write data set-up time	1.25T-17		45		50		
6	t_{DH}	Write data hold time	0.25T-2		10		11		

(Note1) These specification are defined in the case of 0-wait (NDnFSPR<SPW3 to 0>="0000").

All spec. can be calculated by adding (T* number which is set to NDnFSPR<SPW3 to 0>).

AC Measuring Conditions

- Output Level : High = 0.7 Vcc, Low = 0.3 Vcc, CL = 50 pF
- Input Level : High = 0.9 Vcc, Low = 0.1 Vcc.



4.4 AD Conversion Characteristics

Symbol	Parameter	Min	Typ	Max	Unit
VREFH	Analog reference voltage(+)	VCC-0.2	VCC	VCC	V
VREFL	Analog reference voltage(−)	VSS	VSS	VSS+0.2	
AVCC	AD Converter Power Supply Voltage	VCC	VCC	VCC	
AVSS	AD Converter Ground	VSS	VSS	VSS	
AVIN	Analog Input Voltage	VREFL		VREFH	
IREF	Analog Current for analog reference voltage <VREFON> = 1		0.8	1.35	mA
	Analog Current for analog reference voltage <VREFON> = 0		0.02	5.0	μA
E_T	Total error (Quantize error of ±0.5LSB is included)		±1.0	±4.0	LSB

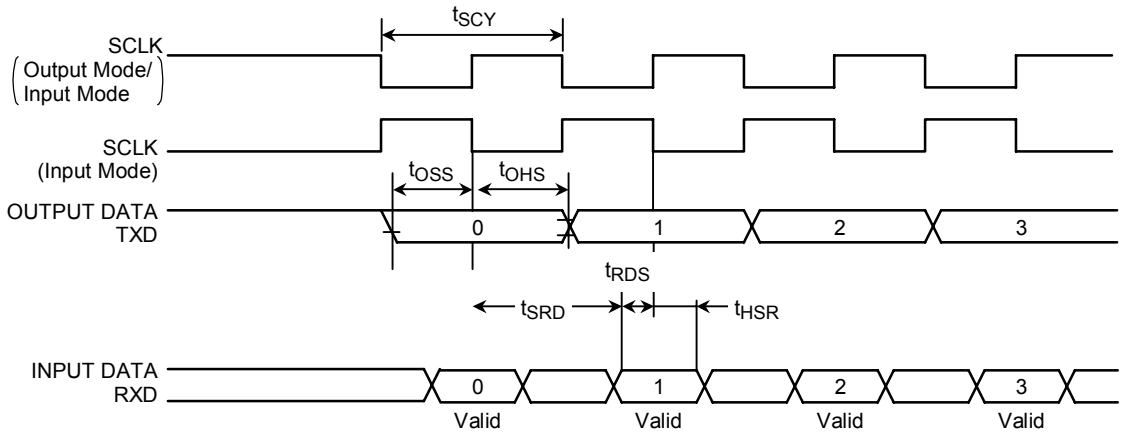
4.5 Serial Channel Timing

(1) SCLK Input mode (I/O Interface mode)

Parameter	Symbol	Variable		40MHz		36MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK Cycle	t_{SCY}	16T		0.8		0.89		μs
Output Data → SCLK Rise	t_{OSS}	t_{SCY}/2-50		350		395		ns
SCLK Rise → Output Data Hold	t_{OHS}	t_{SCY}/2-100		300		345		
SCLK Rise → Input Data Hold	t_{HSR}	0		0		0		
SCLK Rise → Input Data Valid	t_{SRD}		t_{SCY}/2-100		300		345	

(2) SCLK Output mode (I/O Interface mode)

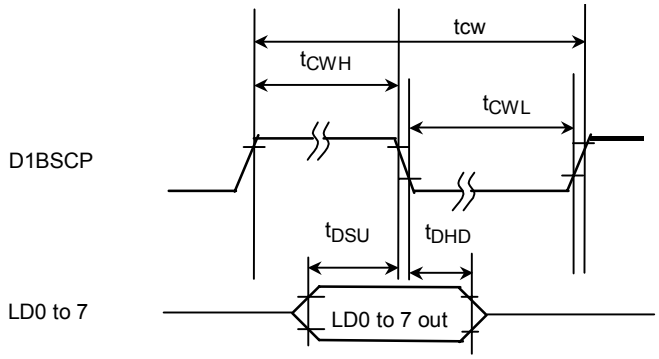
Parameter	Symbol	Variable		40MHz		36MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK Cycle (programmable)	t_{SCY}	16T	8192T	0.8	409.6	0.89	512	μs
Output Data → SCLK Rise	t_{OSS}	t_{SCY}/2-150		250		295		ns
SCLK Rise → Output Data Hold	t_{OHS}	t_{SCY}/2-80		320		365		
SCLK Rise → Input Data Hold	t_{HSR}	0		0		0		
SCLK Rise → Input Data Valid	t_{SRD}		t_{SCY}/2-150		250		295	



4.6 Interrupt Operation

Parameter	Symbol	Variable		40MHz		36MHz		Unit
		Min	Max	Min	Max	Min	Max	
INT0～INT5 Low Width	t_{INTAL}	4T+40		200		260		ns
INT0～INT5 High Width	t_{INTAH}	4T+40		200		260		

4.7 LCD Controller (SR mode)



Vcc = 3.3 ± 0.3V / X1 = 8 to 40MHz / Ta = -20 to 70℃

No.	Symbol	Parameter	Variable		40 MHz (Case: tm=0)		32 MHz (Case: tm=0)		
			Min	Max	Min	Max	Min	Max	
1	tDSU	data Vaild ->D1BSCP Fall	0.5T-20+tm		5		11.25		ns
2	tDHD	D1BSCP Fall->Data Hold	0.5T -5+tm		20		26.25		ns
3	tCWH	D1SBCP->Clock high width	0.5T -10+tm		15		21.25		ns
4	tCWL	D1BSCP->Clock low width	0.5T -10+tm		15		21.25		ns
5	tcw	D1BSCP->Clock cycle	T +2tm		50		62.5		ns

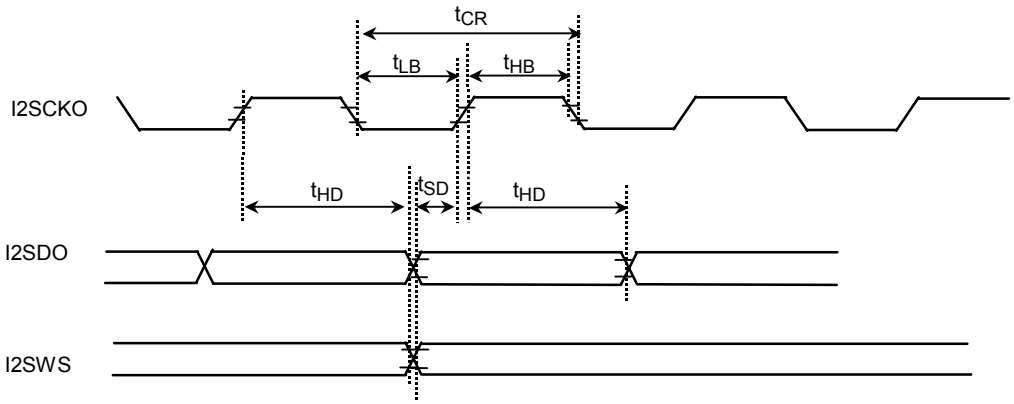
Note: tm=(2^scpw-1)x, ex. If Scpw=3 (8 clock mode) and @20 MHz, tm=(2^3-1)*50=350

4.8 I^2S timing (I^2S,SIO-mode)

Symbol	Parameter	Variable		40 MHz		36 MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_{CR}	I2SCKO Period	2X		50		55		ns
t_{HB}	I2SCKO High width	$t_{CR}/2$ - 10		15		17		
t_{LB}	I2SCKO Low width	$t_{CR}/2$ - 10		15		17		
t_{SD}	I2SDO,I2SWS setup time	$t_{CR}/2$ - 13		12		14		
t_{HD}	I2SDO,I2SWS hold time	$t_{CR}/2$ + 0		25		27		

AC Measuring Conditions

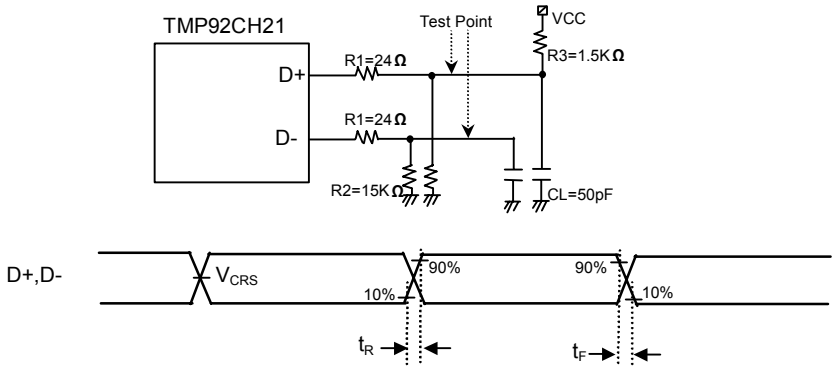
Output Level : High = 0.7 Vcc, Low = 0.3 Vcc, CL = 10 pF



4.9 USB timing (Full speed)

Symbol	Parameter	Min	Max	Unit
t_R	Rising time for D+,D-	4	20	ns
t_F	Falling time for D+,D-	4	20	
V_{CRS}	Output signal crossover voltage	1.3	2.0	V

AC Measuring Conditions

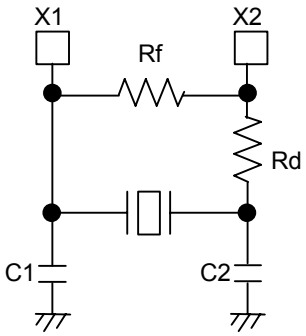


4.12 Recommended Crystal Oscillation Circuit

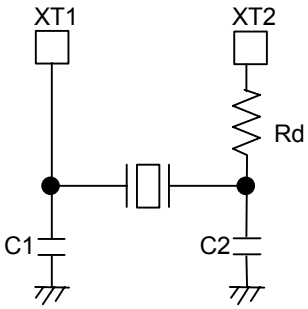
TMP92CH21 is evaluated by below oscillator vender. When selecting external parts, make use of this information..

(note): Total loads value of oscillator is sum of external loads(C1 and C2) and floating loads of actual assemble board. There is a possibility of miss-operating using C1 and C2 value in below table. When designing board, it should design minimum length pattern around oscillator. And we recommend that oscillator evaluation try on your actual using board.

(1) Connection example



High frequency oscillator



Low frequency oscillator

(2) Recommended ceramic oscillator : MURATA co. LTD; JAPAN

MCU	Oscillation Frequency [MHZ]	Item of Oscillator	Parameter of elements				Running Condition	
			C1 [pF]	C2 [pF]	Rf [Ω]	Rd [Ω]	Voltage of Power [V]	Tc [℃]
TMP92CH21	4.00M						2.7 to 3.6	-40 to +85
	6.750M							
	12.50M							
	20.00M							
	27.00M							
	32.00M							

NOTE: In CST ***type oscillator, Capacitance C1,C2 is built in

*After 2001/06,new products will be made, and the old products(now in production) will not be made in MURATA Co LTD ,JAPAN

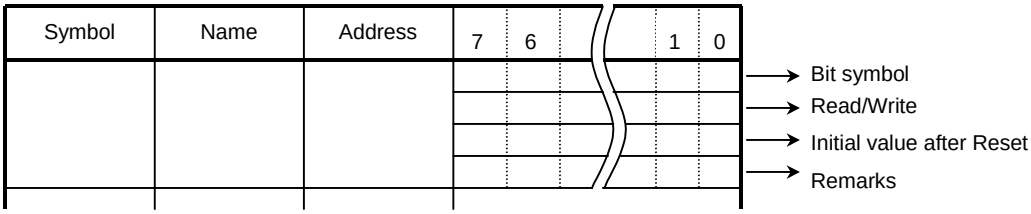
*The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are being changed as occasion arises.. For details, visit the company's home page at http://www.murata.co.jp/search/index.html

5 Table of Special function registers (SFRs)
(SFR ; special function register)

The SFRs include the I/O ports and peripheral control registers allocated to the 4K bytes address space from 000000H to 001FFFH.

(1)I/O Port
(2)Interrupt Control
(3)Memory Controller
(4)MMU
(5)Clock Gear, PLL
(6)LCD Controller
(7)Touch Screen I/F
(8) SDRAM Controller
(9) 8-bit Timer
(10) 16-bit Timer
(11) UART/Serial Channel
(12) USB Controller
(13) AD Converter
(14) Watchdog Timer
(15) RTC (Real–Time Clock)
(16) Melody/Alarm Generator
(17) NAND-Flash Controller
(18) I^2S

Table layout



Note: “Prohibit RMW” in the table means that you cannot use RMW instructions on these register.

Example: When setting bit0 only of the registerP0CR, the instruction “SET 0, (0002G)” 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, TEST, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instruction are read-modify-write instructions.)

Prohibit RMW*; Read-modify-write is prohibited when controlling the pull-up resistor.

Table 5 I/O register address map

[1] PORT

ADDRESS	NAME
0000H	
1H	
2H	
3H	
4H	P1
5H	
6H	P1CR
7H	P1FC
8H	P2
9H	P2FC2
AH	P2CR
BH	P2FC
CH	P3
DH	
EH	P3CR
FH	P3FC

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	PACR
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	PFFC2
EH	PFCR
FH	PFFC

ADDRESS	NAME
0040H	PG
1H	
2H	
3H	
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	PLCR
7H	PLFC
8H	PM
9H	
AH	
BH	PMFC
CH	
DH	
EH	
FH	

ADDRESS	NAME
0080H	
1H	P1DR
2H	P2DR
3H	P3DR
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	
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	
3H	
4H	INTETA01
5H	INTETA23
6H	
7H	
8H	INTETB01
9H	
AH	INTETBO0
BH	INTES0
CH	INTES1
DH	
EH	
FH	

ADDRESS	NAME
00E0H	Reserved
1H	Reserved
2H	Reserved
3H	INTEUSB
4H	Reserved
5H	INTALM01
6H	INTALM23
7H	INTALM4
8H	INTERTC
9H	INTEKEY
AH	INTLCD
BH	INTE5I2S
CH	INTEND01
DH	Reserved
EH	INTEP0
FH	Reserved

ADDRESS	NAME
00F0H	INTE0AD
1H	INTETC01
2H	INTETC23
3H	INTETC45
4H	INTETC67
5H	SIMC
6H	IIMC
7H	INTWDT
8H	INTCLR
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADDRESS	NAME
0100H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	DMA4V
5H	DMA5V
6H	DMA6V
7H	DMA7V
8H	DMAB
9H	DMAR
AH	Reserved
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	BROMCR
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[4] MMU

ADDRESS	NAME
01D0H	LOCALPX
1H	LOCALPY
2H	
3H	LOCALPZ
4H	LOCALLX
5H	LOCALLY
6H	
7H	LOCALLZ
8H	LOCALRX
9H	LOCALRY
AH	
BH	LOCALRZ
CH	LOCALWX
DH	LOCALWY
EH	
FH	LOCALWZ

Note: Do not access no-allocated name address.

[5] 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	

[6] LCDC-1

ADDRESS	NAME
0280H	LCDMODE0
1H	LCDMODE1
2H	LCDFFP
3H	LCDDVM
4H	LCDSIZE
5H	LCDCTL0
6H	LCDCTL1
7H	LCDSCC
8H	LCDCCR0
9H	LCDCCR1
AH	LCDCCR2
BH	
CH	
DH	
EH	
FH	

ADDRESS	NAME
0290H	
1H	LCDRP10
2H	LCDRP32
3H	LCDRP54
4H	LCDRP76
5H	LCDGP10
6H	LCDGP32
7H	LCDGP54
8H	LCDGP76
9H	LCDBP10
AH	LCDBP32
BH	
CH	
DH	
EH	
FH	

[6] LCDC-2

ADDRESS	NAME
02A0H	LSARAL
1H	LSARAM
2H	LSARAH
3H	CMNAL
4H	CMNAH
5H	
6H	LSARBL
7H	LSARBM
8H	LSARBH
9H	CMNBL
AH	CMNBH
BH	
CH	LSARCL
DH	LSARCM
EH	LSARCH
FH	

ADDRESS	NAME
02B0H	LCDOE00
1H	LCDOE01
2H	LCDOE02
3H	LCDOE03
4H	LCDOE04
5H	LCDOE05
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADDRESS	NAME
02C0H	LCDOE10
1H	LCDOE11
2H	LCDOE12
3H	LCDOE13
4H	LCDOE14
5H	LCDOE15
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADDRESS	NAME
02D0H	LCDOE20
1H	LCDOE21
2H	LCDOE22
3H	LCDOE23
4H	LCDOE24
5H	LCDOE25
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no-allocated name address.

[7] TSI

ADDRESS	NAME
01F0H	TSICR0
1H	TSICR1
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[8] SDRAMC

ADDRESS	NAME
0250H	SDACR1
1H	SDACR2
2H	SDRCR
3H	SDCMM
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[9] 8bit Timer

ADDRESS	NAME
1100H	TA01RUN
1H	
2H	TA0REG
3H	TA1REG
4H	TA01MOD
5H	TA01FFCR
6H	
7H	
8H	TA23RUN
9H	
AH	TA2REG
BH	TA3REG
CH	TA23MOD
DH	TA3FFCR
EH	
FH	

[10] 16bit 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

[11] SIO

ADDRESS	NAME
1200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	SIRCR
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	
FH	

Note: Do not access no-allocated name address.

[12] USB Controller (1/2)

ADDRESS	NAME
0500H to 067FH	Descripter-RAM (384Byte)

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	StandardRequest
BH	Request
CH	DataSet1
DH	DataSet2
EH	USB_STATE
FH	EOP

ADDRESS	NAME
07D0H	Command
1H	EPx_SINGLE1
2H	
3H	EPx_BCS1
4H	
5H	
6H	INT_Control
7H	
8H	StandardRequestMode
9H	RequestMode
AH	
BH	
CH	
DH	
EH	ID_CONTROL
FH	ID_STATE

Note: Do not access no-allocated name address.

[12] USB Controller (2/2)

ADDRESS	NAME
07E0H	Port_Status
1H	FRAME_L
2H	FRAME_H
3H	ADDRESS
4H	
5H	
6H	USBREADY
7H	
8H	SetDescriptorSTALL
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.

[13] 10bit ADC

ADDRESS	NAME
12A0H	ADREG0L
1H	ADREG0H
2H	ADREG1L
3H	ADREG1H
4H	ADREG2L
5H	ADREG2H
6H	ADREG3L
7H	ADREG3H
8H	Reserved
9H	Reserved
AH	Reserved
BH	Reserved
CH	Reserved
DH	Reserved
EH	Reserved
FH	Reserved

ADDRESS	NAME
12B0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	ADMOD0
9H	ADMOD1
AH	ADMOD2
BH	Reserved
CH	
DH	
EH	
FH	

[14] WDT

ADDRESS	NAME
1300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[15] 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	

[16] 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.

[17] NAND-Flash Controller-1

ADDRESS	NAME
1CC0H	
1H	
2H	
3H	
4H	ND0FMCR
5H	
6H	
7H	
8H	ND0FSR
9H	
AH	
BH	
CH	ND0FISR
DH	
EH	
FH	

ADDRESS	NAME
1CD0H	ND0FIMR
1H	
2H	
3H	
4H	ND0FSPR
5H	
6H	
7H	
8H	ND0FRSTR
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADDRESS	NAME
1CE0H	
1H	
2H	
3H	
4H	ND1FMCR
5H	
6H	
7H	
8H	ND1FSR
9H	
AH	
BH	
CH	ND1FISR
DH	
EH	
FH	

ADDRESS	NAME
1CF0H	ND1FIMR
1H	
2H	
3H	
4H	ND1FSPR
5H	
6H	
7H	
8H	ND1FRSTR
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADDRESS	NAME
1D00H to 1EFFH	ND0FDTR, ND1FDTR

ADDRESS	NAME
1CB0H to 1CB5H	ND0ECCRD ND1ECCRD

ADDRESS	NAME
01C0H	NDCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no-allocated name address.

[18] I^2S

ADDRESS	NAME
0800H	I2SBUFR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	I2SBUFL
9H	
AH	
BH	
CH	
DH	
EH	I2SCTL0
FH	

Note: Do not access no-allocated name address.

(1) I/O Ports (1/6)

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P1	PORT1	0004H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			0	0	0	0	0	0	0	0
			Input Mode							
P2	PORT2	0008H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			0	0	0	0	0	0	0	0
			Input Mode							
P3	PORT3	000CH	P37	P36	P35	P34	P33	P32	P31	P30
			R/W							
			0	0	0	0	0	0	0	0
			Input Mode							
P4	PORT4	0010H	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			0	0	0	0	0	0	0	0
			Output Mode							
P5	PORT5	0014H	P57	P56	P55	P54	P53	P52	P51	P50
			R/W							
			0	0	0	0	0	0	0	0
			Output Mode							
P6	PORT6	0018H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			0	0	0	0	0	0	0	0
			Input Mode							
P7	PORT7	001CH		P76	P75	P74	P73	P72	P71	P70
				R/W						
				1	1	0	0	1	1	1
				Input Mode	Output		Input Mode			Output
P8	PORT8	0020H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			1	1	1	1	1	0/1	1	1
P9	PORT9	0024H	P97	P96	P95	P94	P93	P92	P91	P90
			R		R/W					
			-	-	0	1	1	1	1	1
			Input Mode							
PA	PORTA	0028H	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R							
PC	PORTC	0030H	PC7	PC6			PC3	PC2	PC1	PC0
			R/W				R/W			
			1				1			
			Input Mode				Input Mode			
PF	PORTF	003CH	PF7			PF4	PF3	PF2	PF1	PF0
			R/W			R/W				
			1			1				
			Output			Input Mode				
PG	PORTG	0040H					PG3	PG2	PG1	PG0
							R			
							Input			
PJ	PORTJ	004CH	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
			R/W							
			1	1	1	1	1	1	1	1
PK	PORTK	0050H					PK3	PK2	PK1	PK0
							R/W			
							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						PM2	PM1	
								R/W		
								1	1	

(1) I/O Ports (2/6)

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P1CR	PORT1 Control Register	0006H	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								P1F
										W
										0/1
			0:PORT 1:Data Bus(D8 to D15)							
P2CR	PORT2 Control Register	000AH	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0:Input 1:Output							
P2FC	PORT2 Function Register	0000BH								P2F
										W
										0/1
			0:PORT 1:Data Bus(D16 to D23)							
P2FC2	PORT2 Function Register2	00009H	P27F2	P26F2	P25F2	P24F2	P23F2	P22F2	P21F2	P20F2
			W							
			0							
			0: 3-STATE, 1:Open Drain							
P3CR	PORT3 Control Register	000EH	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
			W							
			0	0	0	0	0	0	0	0
			0:Input 1:Output							
P3FC	PORT3 Function Register	000FH								P3F
										W
							0	0	0	0/1
							Always write to “0”			
P4FC	PORT4 Function Register	0013H	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
			W							
			0/1							
			0:PORT 1:Address Bus(A0 to A7)							
P5FC	PORT5 Function Register	0017H	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
			W							
			0/1							
			0:PORT 1:Address Bus(A8 to A15)							
P6CR	PORT6 Control Register	001AH	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	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
			W							
			0/1							
			0:PORT 1:Address Bus(A16 to A23)							

(1) I/O Ports (3/6)

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
P7CR	PORT7 Control Register	001EH		P76C	P75C			P72C	P71C	
				W	W			W	W	
				0	0			0	0	
			0:Input 1:Output							
P7FC	PORT7 Function Register	001FH		P76F	P75F	P74F	P73F	P72F	P71F	P70F
			W							
				0	0	0	0	0	0	0/1
				0:PORT 1:/WAIT	0:PORT 1:R/W	0:PORT 1:EA25	0:PORT 1:EA24	0:PORT 1:/WRLU	0:PORT 1:/WRLL	0:PORT 1:/RD
P8FC	PORT8 Function Register	0023H	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
			W							
			0	0	0	0	0	0	0	0
			0: PORT 1: /CSZE	0: PORT 1: /CSZD	0: PORT 1: /CSZC	0: PORT 1: /CSZB	0: PORT 1: /CS3	0: PORT 1: /CS2	0: PORT 1: /CS1	0: PORT 1: /CS0
P8FC2	PORT8 Function Register2	0021H	P87F2	P86F2	P85F2	P84F2		P82F2	P81F2	
			W					W		
			0	0	0	0		0	0	
			0: <P87F> 1:/SRUUB	0: <P86F> 1:/SRULB	0: <P85F> 1: /WRUU	0:<P84F> 1: /WRUL	Write '0'	0: <P82F> 1: /CSZA	0: <P81F> 1:/SDCS	
P9CR	PORT9 Control Register	0026H			P95C	P94C	P93C	P92C	P91C	P90C
					W					
					0	0	0	0	0	0
					0:Input 1:Output					
P9FC	PORT9 Function Register	0027H	P97F	P96F	P95F	P94F	P93F	P92F	P91F	P90F
			W							
			0	0	0	0	0	0	0	0
P9FC2	PORT9 Function Register2	0025H								P90FC2
										W
										0
										0: 3-state, 1:Open-drain
PACR	PORTA Control Register	002AH		PA6C	PA5C	PA4C	PA3C			
				W						
				0						
				0: Input port or Key-in 1: LD11 to LD8 output						
PAFC	PORTA Function Register	002BH	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
			W							
			0							
			0: KEY IN disable 1: KEY IN enable							
PCCR	PORTC Control Register	0032H	PC7C	PC6C			PC3C	PC2C	PC1C	PC0C
			W				W			
			0				0			
			0: IN 1: OUT				0: IN 1: OUT			
PCFC	PORTC Function Register	0033H	PC7F	PC6F			PC3F	PC2F	PC1F	PC0F
			W				W			
			0				0			
			0: port 1: /CSZF	0: 3-state, 1:Open-drain						

(1) I/O Ports (4/6)

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
PFCR	PORTF Control Register	003EH				PF4C	PF3C	PF2C	PF1C	PF0C
						W				
						0				
						0: IN		1: OUT		
PFFC	PORTF Function Register	003FH	PF7F			PF4F	PF3F	PF2F	PF1F	PF0F
			W			W	W	W	W	W
			1			0	0	0	0	0
			0: port 1:SDCLK							
PFFC2	PORTF Function Register2	003DH				PF4F2	PF3F2	PF2F2	PF1F2	PF0F2
						W	W	W	W	W
						0	0	0	0	0
PJCR	PORTJ Control Register	004EH		PJ6C	PJ5C					
				W						
				0						
				0:IN 1: OUT						
PJFC	PORTJ Function Register	004FH	PF7F	PF6F	PF5F	PF4F	PF3F	PF2F	PF1F	PF0F
				W						
				0						
			0: port 1: SDCKE	0: port 1:	0: port 1:	0: port 1:	0: port 1:	0: port 1:	0: port 1:	0: port 1:
PKFC	PORTK Function Register	0053H					PK3F	PK2F	PK1F	PK0F
							W			
							0			
							0: PORT 1: LBCD	0: PORT 1: LFR	0: PORT 1: LLP	0: PORT 1: LCP0
PLCR	PORTL Control Register	0056H	PL7C	PL6C	PL5C	PL4C				
			W							
			0							
			0: IN 1: OUT							
PLFC	PORTL Function Register	0057H	PL7F	PL6F	PL5F	PL4F	PL3F	PL2F	PL1F	PL0F
			W							
			0							
			0: Port 1: data bus for LCDC (LD7 to LD0)							
PMFC	PORTM Function Register	005BH						PM2F	PM1F	
								W	W	
								0	0	
								0:port 1:	0:port 1:MLDALM	

(1) I/O Ports (5/6)

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							
			Output/Input buffer drive-register for standby-mode							
P2DR	PORT2 Drive Register	0082H	P27D	P26D	P25D	P24D	P23D	P22D	P21D	P20D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
P3DR	PORT3 Drive Register	0083H	P37D	P36D	P35D	P34D	P33D	P32D	P31D	P30D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
P4DR	PORT4 Drive Register	0084H	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
P5DR	PORT5 Drive Register	0085H	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
P6DR	PORT6 Drive Register	0086H	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
P7DR	PORT7 Drive Register	0087H		P76D	P75D	P74D	P73D	P72D	P71D	P70D
				R/W						
				1						
				Output/Input buffer drive-register for standby-mode						
P8DR	PORT8 Drive Register	0088H	P87D	P86D	P85D	P84D	P83D	P82D	P81D	P80D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
P9DR	PORT9 Drive Register	0089H	P97D	P96D	P95D	P94D	P93D	P92D	P91D	P90D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
PADR	PORTA Drive Register	008AH	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
PCDR	PORTC Drive Register	008CH	PC7D	PC6D			PC3D	PC2D	PC1D	PC0D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
PFDR	PORTF Drive Register	008FH	PF7D			PF4D	PF3D	PF2D	PF1D	PF0D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
PGDR	PORTG Drive Register	0090H					PG3D	PG2D		
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
PJDR	PORTJ Drive Register	0093H	PJ7D	PJ6D	PJ5D	PJ4D	PJ3D	PJ2D	PJ1D	PJ0D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							

(1) I/O Ports (6/6)

SYMBOL	NAME	Address	7	6	5	4	3	2	1	0
PKDR	PORTK Drive Register	0094H					PK3D	PK2D	PK1D	PK0D
							R/W			
							1			
							Output/Input buffer drive-register for standby-mode			
PLDR	PORTL Drive Register	0095H	PL7D	PL6D	PL5D	PL4D	PL3D	PL2D	PL1D	PL0D
			R/W							
			1							
			Output/Input buffer drive-register for standby-mode							
PMDR	PORTM Drive Register	0086H						PM2D	PM1D	PM0D
								R/W		
								1		
								Output/Input buffer drive-register for standby-mode		

(2) Interrupt Control(1/4)

Symbol	NAME	Address	7	6	5	4	3	2	1	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
INTETA01	INTTA0 & INTTA1 Enable	00D4H	INTTA1(Timer1)				INTTA0(Timer0)			
			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	INTAT3(Timer3)				INTAT2(Timer2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB01	INTTB0 & INTTB1 Enable	00D8H	INTTB1(Timer4)				INTTB0(Timer4)			
			ITB1C	ITB1M2	ITB1M1	ITB1M0	ITB0C	ITB0M2	ITB0M1	ITB0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETBO0	INTTBO0 (Overflow) Enable	00DAH	-				INTTBO0			
			-	-	-	-	ITBO0C	ITBO0M2	ITBO0M1	ITBO0M0
			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
INTES1	INTRX1 & INTTX1 Enable	00DCH	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEUSB	INTUSB Enable	00E3H	-				INTUSB			
			-	-	-	-	IUSBC	IUSBM2	IUSBM1	IUSBM0
							R	R/W		
			Note: Always fixed to 0				0	0	0	0
INTEALM 01	INTALM0 & INTALM1 Enable	00E5H	INTALM1				INTALM0			
			IA1C	IA1M2	IA1M1	IA1M0	IA0C	IA0M2	IA0M1	IA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEALM 23	INTALM2 & INTALM3 Enable	00E6H	INTALM3				INTALM2			
			IA3C	IA3M2	IA3M1	IA3M0	IA2C	IA2M2	IA2M1	IA2M0
			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
INTEALM4	INTALM4 Enable	00E7H	-				INTALM4			
			-	-	-	-	IA4C	IA4M2	IA4M1	IA4M0
			R				R	R/W		
			Note: Always fixed to 0				0	0	0	0
INTERTC	INTRTC Enable	00E8H	-				INTRTC			
			-	-	-	-	IRC	IRM2	IRM1	IRM0
			R				R	R/W		
			Note: Always fixed to 0				0	0	0	0
INTECKEY	INTKEY Enable	00E9H	-				INTKEY			
			-	-	-	-	IKC	IKM2	IKM1	IKM0
			R				R	R/W		
			Note: Always fixed to 0				0	0	0	0
INTLCD	INTLCD Enable	00EAH	-				INTLCD			
			-	-	-	-	ILCD1C	ILCDM2	ILCDM1	ILCDM0
			R				R	R/W		
			Note: Always fixed to 0				0	0	0	0
INTE5I2S	INT5 & INTI2S Enable	00EBH	INTI2S				INT5			
			II2SC	II2SM2	II2SM1	II2SM0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEND01	INTND0 & INTND1 Enable	00ECH	INTND1				INTND0			
			IND1C	IND1M2	IND1M1	IND1M0	IND0C	IND0M2	IND0M1	IND0M0
				R			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 Enable	00EEH	-				INTP0			
			-	-	-	-	IP0C	IP0M2	IP0M1	IP0M0
			R				R	R/W		
			Note: Always fixed to 0				0	0	0	0

(2) Interrupt Control(3/4)

Symbol	NAME	Address	7	6	5	4	3	2	1	0
INTE0AD	INT0 & INTAD Enable	00F0H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC01	INTTC0 & INTTC1 Enable	00F1H	INTTC1(DMA1)				INTTC0(DMA0)			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23	INTTC2& INTTC3 Enable	00F2H	INTTC3(DMA3)				INTTC2(DMA2)			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC45	INTTC4 & INTTC5 Enable	00F3H	INTTC5(DMA5)				INTTC4(DMA4)			
			ITC5C	ITC5M2	ITC5M1	ITC5M0	ITC4C	ITC4M2	ITC4M1	ITC4M0
			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	00F5 (no RMW)	-	-	-	-	-	-	IR1LE	IR0LE
			W						W	W
			0	-	-	-	-	-	1	1
			Write “0”						0:INTRX1 edge mode 1:INTRX1 level mode	0:INTRX0 edge mode 1:INTRX0 level mode
IIMC	Interrupt Input Mode Control	00F6H (no RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	-
			W	W	W	W	W	W	R/W	
			0	0	0	0	0	0	0	0
			INT4EDGE 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	0:INT0 edge mode 1:INT0 level mode	Write “0”
INTWDT	INTWD Enable	00F7H	-				INTWD			
			-	-	-	-	ITCWD	-	-	-
			-				R			
			-	-	-	-	0	-	-	-
INTCLR	Interrupt Clear control	00F8H (no RMW)	-	-	-	-	-	-	-	-
			W							
			0	0	0	0	0	0	0	0
			Interrupt Vector							

(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 (no RMW)	DREQ7	DREQ6	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1	DREQ0
			R/W							
			0	0	0	0	0	0	0	0
			1:DMA request in software							

(3) Memory Controller(1/2)

Symbol	NAME	Address	7	6	5	4	3	2	1	0
B0CSL	BLOCK 0 CS/WAIT Control Register Low	0140H		B0WW2	B0WW1	B0WW0		B0WR2	B0WR1	B0WR0
			W							
				0	1	0		0	1	0
				Write waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved				Read waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved		
B0CSH	BLOCK 0 CS/WAIT Control Register High	0141H	B0E	-	-	B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
			W							
			0	0	0	0	0	0	0/1	0/1
			CS select 0:disable 1:enable	Fix to “0”	Fix to “0”		00:ROM/SRAM 01:Reserved 10:Reserved 11:Reserved		Data Bus width 00:8bit 01:16bit 10:32bit 11:Reserved	
B1CSL	BLOCK 1 CS/WAIT Control Register Low	0144H		B1WW2	B1WW1	B1WW0		B1WR2	B1WR1	B1WR0
			W							
				0	1	0		0	1	0
				Write waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved				Read waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved		
B1CSH	BLOCK 1 CS/WAIT Control Register High	0145H	B1E	-	-	B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
			W							
			0	0	0	0	0	0	0/1	0/1
			CS select 0:disable 1:enable	Fix to “0”	Fix to “0”		00:ROM/SRAM 01:Reserved 10:Reserved 11:SDRAM		Data Bus width 00:8bit 01:16bit 10:32bit 11:Reserved	
B2CSL	BLOCK 2 CS/WAIT Control Register Low	0148H		B2WW2	B2WW1	B2WW0		B2WR2	B2WR1	B2WR0
			W							
				0	1	0		0	1	0
				Write waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved				Read waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved		
B2CSH	BLOCK 2 CS/WAIT Control Register High	0149H	B2E	B2M		B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
			W							
			1	0	0	0	0	0	0/1	0/1
			CS select 0:disable 1:enable	0:16MB 1:Sets area	Fix to “0”		00:ROM/SRAM 01:Reserved 10:Reserved 11: SDRAM		Data Bus width 00:8bit 01:16bit 10:32bit 11:Reserved	
B3CSL	BLOCK 3 CS/WAIT Control Register Low	014CH		B3WW2	B3WW1	B3WW0		B3WR2	B3WR1	B3WR0
			W							
				0	1	0		0	1	0
				Write waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved				Read waits 001:0WAIT 010:1WAIT 101:2WAIT 110:3WAIT 011:1+NWAIT 111:4WAIT Others:Resrved		
B3CSH	BLOCK 3 CS/WAIT Control Register High	014DH	B3E			B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
			W							
			0	0	0	0	0	0	0/1	0/1
			CS select 0:disable 1:enable	Fix to “0”	Fix to “0”		00:ROM/SRAM 01:Reserved 10:Reserved 11:Reserved		Data Bus width 00:8bit 01:16bit 10:32bit 11:Reserved	

(3) Memory Controller(2/2)

Symbol	NAME	Address	7	6	5	4	3	2	1	0
BEXCSL	BLOCK EX CS/WAIT Control Register Low	0158H		BEXWW2	BEXWW1	BEXWW0		BEXWR2	BEXWR1	BEXWR0
			W							
			-	0	1	0	-	0	1	0
				Write waits 001:2WAIT 010:1WAIT 101:2WAIT 110:2WAIT 011:1+NWAIT Others:Resrved				Read waits 001:2WAIT 010:1WAIT 101:2WAIT 110:2WAIT 011:1+NWAIT Others:Resrved		
BEXCSH	BLOCK EX CS/WAIT Control Register High	0159H					BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
			W							
			-	0	0	0	0	0	0/1	0/1
				Fix to “0”	Fix to “0”	Fix to “0”	00:ROM/SRAM 01:Reserved 10:Reserved 11:Reserved		00:8bit 01:16bit 10:32bit 11:Reserved	
PMEMCR	Page ROM Control reg.	0166H				OPGE	OPWR1	OPWR0	PR1	PR0
						R/W				
						0	0	0	1	0
						ROM page access 0:disable 1:enable	wait number on page 00:1CLK(n-1-1-1 mode) 01:2CLK(n-2-2-2 mode) 10:3CLK(n-3-3-3 mode) 11:(Reserved)		Byte number in a page 00:64byte 01:32byte 10:16byte 11:8byte	
MAMR0	Memory Address Mask Register0	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 Register0	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 Register1	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 Register1	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 Register2	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 Register2	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 Register3	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 Register3	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							
BROMCR	Boot ROM Control register	0167H							ROMLESS	VACE
									R/W	
									1/0	0/1

(4) MMU

Symbol	NAME	Address	7	6	5	4	3	2	1	0
LOCALPX	LOCALX Register For Program	01D0H	LXE		X5	X4	X3	X2	X1	X0
			R/W		R/W					
			0		0					
			LOCALX 1: enable		Setting BANK number for LOCALX					
LOCALPY	LOCALY Register For Program	01D1H	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0				
			LOCALY 1: enable			Setting BANK number for LOCALY				
LOCALPZ	LOCALZ Register For Program	01D3H	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0						
			LOCALZ 1: enable	Setting BANK number for LOCAL2						
LOCALLX	LOCALX Register For LCDC	01D4H	LXE		X5	X4	X3	X2	X1	X0
			R/W		R/W					
			0		0					
			LOCALX 1: enable		Setting BANK number for LOCALX					
LOCALLY	LOCALY Register For LCDC	01D5H	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0				
			LOCALY 1: enable			Setting BANK number for LOCALY				
LOCALLZ	LOCALZ Register For LCDC	01D7H	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0						
			LOCALZ 1: enable	Setting BANK number for LOCAL2						
LOCALRX	LOCALX Register For Read	01D8H	LXE		X5	X4	X3	X2	X1	X0
			R/W		R/W					
			0		0					
			LOCALX 1: enable		Setting BANK number for LOCALX					
LOCALRY	LOCALY Register For Read	01D9H	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0				
			LOCALY 1: enable			Setting BANK number for LOCALY				
LOCALRZ	LOCALZ Register For Read	01DBH	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0						
			LOCALZ 1: enable	Setting BANK number for LOCAL2						
LOCALWX	LOCALX Register For Write	01DCH	LXE		X5	X4	X3	X2	X1	X0
			R/W		R/W					
			0		0					
			LOCALX 1: enable		Setting BANK number for LOCALX					
LOCALWY	LOCALY Register For Write	01DDH	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0				
			LOCALY 1: enable			Setting BANK number for LOCALY				
LOCALWZ	LOCALZ Register For Write	01DFH	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0						
			LOCALZ 1: enable	Setting BANK number for LOCAL2						

(5) Clock Gear,PLL

Symbol	NAME	Address	7	6	5	4	3	2	1	0
SYSCR0	System Clock Control Register0	10E0H	XEN	XTEN				WUEF		
			R/W					R/W		
			1	1				0		
			H-OSC(fc) 0: stop 1: oscillation	L-OSC(fs) 0: stop 1: oscillation				Warm-up Timer		
SYSCR1	System Clock Control Register1	10E1H					SYSCK	GEAR2	GEAR1	GEAR0
							R/W	R/W		
							0	1	0	0
							Select system clock 0:fc 1:fs	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	System Clock Control Register2	10E2H			WUPTM1	WUPTM0	HALTM1	HALTM0		
			R/W		R/W					
			0		1	0	1	1		
			Write '0'.		Warm-Up Timer 00: reserved 01: 2^8/inputted frequency 10:2^{14} 11:2^{16}		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
			0					0	1	1
			Protect flag 0: OFF 1: ON					1: External clock	fc oscillator driver ability 1: NORMAL 0: WEAK	fs oscillator driver 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					
PLLCR1	PLL Control Register1	10E9H	PLLON							
			R/W							
			0							
			Control on/off 0: Off 1: On							

(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	0	0	0	0	0
LCD MODE1	LCD Mode1 Register	0281H				LDINV	AUTOINV		LDO1	LDO0
						R/W	R/W		R/W	R/W
						0	0		0	0
LCDFFP	LCD Frame Frequency Register	0282H	FP7	FP6	FP5	FP4	FP3	FP2	FP1	FP0
			R/W							
			0							
			Setting bit7-0 for f FP							
LCDDVM	LCD Divide FRM Register	0283H	FMN7	FMN6	FMN5	FMN4	FMN3	FMN2	FMN1	FMN0
			R/W							
			0							
			Setting DVM bit7-0							
LCDSIZE	LCD Size Register	0284H	COM3	COM2	COM1	COM0	SEG3	SEG2	SEG1	SEG0
			R/W							
			0				0			
			Setting the LCD Common number for SR Mode				Setting the LCD Segment Number for SR Mode			
LCDCTL0	LCD Control0 Register	0285H		ALL0	FRMON	EXOR	FP9	MMULCD	FP8	START
				R/W						
				0	0	0	0	0	0	0
LCDCTL1	LCD Control1 Register	0286H	LCP0P	LCP1P	LBCDP				LBCDW1	LBCDW0
			R/W	R/W	R/W				R/W	R/W
			1	0	0				0	0

(6) LCD Controller(2/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDSCC	LCD Sourcer Clock Counter Register	0287H	SCC7	SCC6	SCC5	SCC4	SCC3	SCC2	SCC1	SCC0
			R/W							
			0							
			LCDC Source Clock Counter bit7-0							
LCDCCR0	LCD Clock Counter Register 0	0288H						PCPV2	PCPV1	PCPV0
								R/W		
								0	0	0
LCDCCR1	LCD Clock Counter Register 1	0289H				TLDE4	TLDE3	TLDE2	TLDE1	TLDE0
						R/W				
						0	0	0	0	0
LCDCCR2	LCD Clock Counter Register 2	028AH			LLPSU5	LLPSU4	LLPSU3	LLPSU2	LLPSU1	LLPSU0
					R/W					
			0	0	0	0	0	0	0	0
LCDRP10	LCD Red Palette Register 10	0291H	1R3	1R2	1R1	1R0	0R3	0R2	0R1	0R0
			R/W				R/W			
			0	0	1	0	0	0	0	0
LCDRP32	LCD Red Palette Register 32	0292H	3R3	3R2	3R1	3R0	2R3	2R2	2R1	2R0
			R/W				R/W			
			0	1	1	0	0	1	0	0
LCDRP54	LCD Red Palette Register 54	0293H	5R3	5R2	5R1	5R0	4R3	4R2	4R1	4R0
			R/W				R/W			
			1	0	1	0	1	0	0	0
LCDRP76	LCD Red Palette Register 76	0294H	7R3	7R2	7R1	7R0	6R3	6R2	6R1	6R0
			R/W				R/W			
			1	1	1	0	1	1	0	0

(6) LCD Controller(3/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDGP10	LCD Green Palette Register 10	0295H	1G3	1G2	1G1	1G0	0G3	0G2	0G1	0G0
			R/W				R/W			
			0	0	1	0	0	0	0	0
LCDGP32	LCD Green Palette Register 32	0296H	3G3	3G2	3G1	3G0	2R3	2G2	2G1	2G0
			R/W				R/W			
			0	1	1	0	0	1	0	0
LCDGP54	LCD Green Palette Register 54	0297H	5G3	5G2	5G1	5G0	4G3	4G2	4G1	4G0
			R/W				R/W			
			1	0	1	0	1	0	0	0
LCDGP76	LCD Green Palette Register 76	0298H	7G3	7G2	7G1	7G0	6G3	6G2	6G1	6G0
			R/W				R/W			
			1	1	1	0	1	1	0	0
LCDBP10	LCD Blue Palette Register 10	0299H	1B3	1B2	1B1	1B0	0B3	0B2	0B1	0B0
			R/W				R/W			
			0	1	0	0	0	0	0	0
LCDBP32	LCD Blue Palette Register 32	029AH	3B3	3B2	3B1	3B0	2B3	2B2	2B1	2B0
			R/W				R/W			
			1	1	0	0	1	0	0	0

(6) LCD Controller(4/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LSARAL	Start Address Register A-Area(L)	02A0H	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
			R/W							
			0	0	0	0	0	0	0	0
			Start Address for A-area(bit7-0)							
LSARAM	Start Address Register A-Area(M)	02A1H	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8
			R/W							
			0	0	0	0	0	0	0	0
			Start Address for A-area(bit15-8)							
LSARAH	Start Address Register A-Area(H)	02A2H	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16
			R/W							
			0	1	0	0	0	0	0	0
			Start Address for A-area(bit23-16)							
CMNAL	Common Number Register A-Area(L)	02A3H	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0
			R/W							
			0	0	0	0	0	0	0	0
			Common Number setting for A-area(bit7-0)							
CMNAH	Common Number Register A-Area(H)	02A4H								CA8
										R/W
										0
										A-area(bit8)
LSARBL	Start Address Register B-Area(L)	02A6H	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
			R/W							
			0	0	0	0	0	0	0	0
			Start Address for B-area(bit7-0)							
LSARBM	Start Address Register B-Area(M)	02A7H	SB15	SB14	SB13	SB12	SB11	SB10	SB9	SB8
			R/W							
			0	0	0	0	0	0	0	0
			Start Address for B-area(bit15-8)							
LSARBH	Start Address Register B-Area(H)	02A8H	SB23	SB22	SB21	SB20	SB19	SB18	SB17	SB16
			R/W							
			0	1	0	0	0	0	0	0
			Start Address for B-area(bit23-16)							
CMNBL	Common Number Register B-Area(L)	02A9H	CB7	CB6	CB5	CB4	CB3	CB2	CB1	CB0
			R/W							
			0	0	0	0	0	0	0	0
			Common Number setting for B-area(bit7-0)							
CMNBH	Common Number Register B-Area(H)	02AAH								CB8
										R/W
										0
										B-area(bit8)
LSARCL	Start Address Register C-Area(L)	02ACH	SC7	SC6	SC5	SC4	SC3	SC2	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Start Address for C-area(bit7-0)							
LSARCM	Start Address Register C-Area(M)	02ADH	SC15	SC14	SC13	SC12	SC11	SC10	SC9	SC8
			R/W							
			0	0	0	0	0	0	0	0
			Start Address for C-area(bit15-8)							
LSARCH	Start Address Register C-Area(H)	02AEH	SC23	SC22	SC21	SC20	SC19	SC18	SC17	SC16
			R/W							
			0	1	0	0	0	0	0	0
			Start Address for C-area(bit23-16)							

(6) LCD Controller(5/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDOE00	LCD OE0 Control Register-0	02B0H	OE007	OE006	OE005	OE004	OE003	OE002	OE001	OE000
			R/W							
			0	0	0	0	0	0	0	0
LCDOE01	LCD OE0 Control Register-1	02B1H	OE017	OE016	OE015	OE014	OE013	OE012	OE011	OE010
			R/W							
			0	0	0	0	0	0	0	0
LCDOE02	LCD OE0 Control Register-2	02B2H	OE027	OE026	OE025	OE024	OE023	OE022	OE021	OE020
			R/W							
			0	0	0	0	0	0	0	0
LCDOE03	LCD OE0 Control Register-3	02B3H	OE037	OE036	OE035	OE034	OE033	OE032	OE031	OE030
			R/W							
			0	0	0	0	0	0	0	0
LCDOE04	LCD OE0 Control Register-4	02B4H	OE047	OE046	OE045	OE044	OE043	OE042	OE041	OE040
			R/W							
			0	0	0	0	0	0	0	0
LCDOE05	LCD OE0 Control Register-5	02B5H	OE057	OE056	OE055	OE054	OE053	OE052	OE051	OE050
			R/W							
			0	0	0	0	0	0	0	0
LCDOE10	LCD OE1 Control Register-0	02C0H	OE107	OE106	OE105	OE104	OE103	OE102	OE101	OE100
			R/W							
			0	0	0	0	0	0	0	0
LCDOE11	LCD OE1 Control Register-1	02C1H	OE117	OE116	OE115	OE114	OE113	OE112	OE111	OE110
			R/W							
			0	0	0	0	0	0	0	0
LCDOE12	LCD OE1 Control Register-2	02C2H	OE127	OE126	OE125	OE124	OE123	OE122	OE121	OE120
			R/W							
			0	0	0	0	0	0	0	0
LCDOE13	LCD OE1 Control Register-3	02C3H	OE137	OE136	OE135	OE134	OE133	OE132	OE131	OE130
			R/W							
			0	0	0	0	0	0	0	0
LCDOE14	LCD OE1 Control Register-4	02C4H	OE147	OE146	OE145	OE144	OE143	OE142	OE141	OE140
			R/W							
			0	0	0	0	0	0	0	0
LCDOE15	LCD OE1 Control Register-5	02C5H	OE157	OE156	OE155	OE154	OE153	OE152	OE151	OE150
			R/W							
			0	0	0	0	0	0	0	0

(6) LCD Controller(6/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDOE20	LCD OE2 Control Register-0	02D0H	OE207	OE206	OE205	OE204	OE203	OE202	OE201	OE200
			R/W							
			0	0	0	0	0	0	0	0
LCDOE21	LCD OE2 Control Register-1	02D1H	OE217	OE216	OE215	OE214	OE213	OE212	OE211	OE210
			R/W							
			0	0	0	0	0	0	0	0
LCDOE22	LCD OE2 Control Register-2	02D2H	OE227	OE226	OE225	OE224	OE223	OE222	OE221	OE220
			R/W							
			0	0	0	0	0	0	0	0
LCDOE23	LCD OE2 Control Register-3	02D3H	OE237	OE236	OE235	OE234	OE233	OE232	OE231	OE230
			R/W							
			0	0	0	0	0	0	0	0
LCDOE24	LCD OE2 Control Register-4	02D4H	OE247	OE246	OE245	OE244	OE243	OE242	OE241	OE240
			R/W							
			0	0	0	0	0	0	0	0
LCDOE25	LCD OE2 Control Register-5	02D5H	OE257	OE256	OE255	OE254	OE253	OE252	OE251	OE250
			R/W							
			0	0	0	0	0	0	0	0

(7) Touch Screen I/F

Symbol	Name	Address	7	6	5	4	3	2	1	0
TSICR0	Touch Screen I/F Control Reg0	01F0H	TSI7		PTST	TWIEN	PYEN	PXEN	MYEN	MXEN
			R/W		R	R/W	R/W	R/W	R/W	R/W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		0: not touch 1: touch	0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable
TSICR1	Touch Screen I/F Control Reg1	01F1H	DBC7	DB1024	DB256	DB64	DB8	DB4	DB2	DB1
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	0	0	0	0	0
			0: Disable 1: Enable	0: Disable 1: Add 1024*96X	0: Disable 1: Add 256*96X	0: Disable 1: Add 64*96X	0: Disable 1: Add 8*96X	0: Disable 1: Add 4*96X	0: Disable 1: Add 2*96X	0: Disable 1: Add 1*96X

(8) SDRAM Controller

Symbol	Name	Address	7	6	5	4	3	2	1	0
SDACR1	SDRAM Access Control Register1	0250H		SASFR	SMRD	SWRC	SBST	SWMOD	SRBL	SMAC
				R/W						
				0				1	0	
			Write “0”	Selecting Self Refresh Mode	Mode Register Set Delay Time	Write Recovery Time	Burst Stop Command	Selecting Write Mode	Selecting Read Burst Length	0:disable 1:enable
SDACR2	SDRAM Access Control Register2	0251H				SBS	SDRS1	SDRS0	SMUXW1	SMUXW0
							R/W			
						0	0	0	0	0
						Number of Banks	Selecting ROW address Size		Selecting Address Multiplex Type	
SDRCR	SDRAM Refresh Control Register	0252H					SRS2	SRS1	SRS0	SRC
							R/W			
							0	0	0	0
							Refresh interval 000: 47state 100:156state 001: 78state 101:295state 010: 97state 110:249state 011:124state 111:312state			Auto Refresh 0:Disable 1:Enable
SDCMM	SDRAM Command Register	0253H						SCMM2	SCMM1	SCMM0
								R/W		
								0	0	0
								Issuing Command		

(9) 8-bit Timer

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	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		
TA0REG	8bit Timer Regisster 0	1102H	-							
			W							
			Undefined							
TA1REG	8bit Timer Regisster 1	1103H	-							
			W							
			Undefined							
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: 2^6-1 10: 2^7-1 11: 2^8-1		Source clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA0 00: reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA1FFCR	TMRA1 Flip-Flop Conttol Register	1105H					TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
							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	TA1RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		
TA2REG	8bit Timer Regisster 2	110AH	-							
			W							
			Undefined							
TA3REG	8bit Timer Regisster 3	110BH	-							
			W							
			Undefined							
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: 2^6-1 10: 2^7-1 11: 2^8-1		Source clock for TMRA1 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA2 00: reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA3FFCR	TMRA3 Flip-Flop Conttol Register	110DH					TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
							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	TA1FF Inversion select 0: TMRA2 1: TMRA3

(10) 16-bit Timer

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 fixed to "0"			IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		
TB0MOD	TMRB0 MODE Register	1182H	-	TB0ET1	TB0CP01	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
				R/W	W			R/W		
			0	0	1	0	0	0	0	0
			always fixed to "0"	TB0FF1 inversion 0: Disable 1: Enable / Invert when the UC value matches the value in TB0RG1	Execute software capture 0: Execute 1: Don't care	Capture timing 00: Disable 01:reserved 10:reserved 11:TA1OUT↑ TA1OUT↓		Control up-counter 0: Disable clearing 1: Enable clearing	TMRB0 source clock 00: reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TB0FFCR	TMRB0 Flip-Flop Control Register	1183H	TB0FF1C1	TB0FF1C0		TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*			R/W			W*	
			1	1	0	0	0	0	1	1
			Control TB0FF1 00: Invert 01: Set 10: Clear 11: Don't care ∗ Always read as 11		(Note) Always fixed to "0"	TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger / Invert when the UC value is loaded in to TB0CP0.	Invert when the UC value matches the value in TB0RG1.	Invert when the UC value matches the value in TB0RG0.	Control TB0FF0 00: Invert 01: Set 10: Clear 11: Don't care ∗ Always read as 11	
TB0RG0L	16bit Timer Regisster 0 Low	1188H	-							
			W							
			Undefined							
TB0RG0H	16bit Timer Regisster 0 High	1189H	-							
			W							
			Undefined							
TB0RG1L	16bit Timer Regisster 1 Low	118AH	-							
			W							
			Undefined							
TB0RG1H	16bit Timer Regisster 1 High	118BH	-							
			W							
			Undefined							
TB0CP0L	Capture Regisster 0 Low	118CH	-							
			R							
			Undefined							
TB0CP0H	Capture Regisster 0 High	118DH	-							
			R							
			Undefined							
TB0CP1L	Capture Regisster 1 Low	118EH	-							
			R							
			Undefined							
TB0CP1H	Capture Regisster 1 High	118FH	-							
			R							
			Undefined							

(11) UART/Serial Channel(1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial Channel 0 Buffer Register	1200H	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R(Receiving) / W(Transmission)							
			Undefined							
SC0CR	Serial Channel 0 Control Register	1201H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Clear 0 after reading)			R/W	
			0	0	0	0	0	0	0	0
			Receive data bit 8	Parity 0:Odd 1:Even	Parity 0:Disable 1:Enable	1:Error Overrun	1:Error Parity	1:Error 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
			Trans-mission data bit 8	0:CTS Disable 1:CTS Enable	0:Receive Disable 1:Receive Enable	Wake up 0:Disable 1:Enable	00:I/O Interface Mode 01:7bit UART Mode 10:8bit UART Mode 11:9bit UART Mode		00:TimerTOTRG 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
			Fix to "0"	(16-K)/16 divided 0:Disable 1:Enable	00:φT0 01:φT2 10:φT8 11:φT32		Set the frequency divisor "N" 0 to F			
BR0ADD	Serial Channel 0 K setting Register	1204H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
			-	-	-	-	0	0	0	0
							Set the frequency divisor "K" (1 to F)			
SC0MOD1	Serial Channel 0 Mode 1 Register	1205H	I2S0	FDPX0						
			R/W	R/W						
			0	0	-	-	-	-	-	-
			IDLE2 0:Stop 1:Operate	I/O interface mode 1:Full duplex 0:Half duplex						
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 effective pulse width for equal or more than 2x × x (value + 1) Can be set : 1 to 4 Can not be set : 0, 15			

(11) UART/Serial Channel(2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC1BUF	Serial Channel 1 Buffer Register	1208H	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R(Receiving) / W(Transmission)							
			Undefined							
SC1CR	Serial Channel 1 Control Register	1209H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Clear 0 after reading)			R/W	
			0	0	0	0	0	0	0	0
			Receive data bit 8	Parity 0:Odd 1:Even	Parity 0:Disable 1:Enable	1:Error Overrun			0:SCLK1 ↑ 1:SCLK1 ↓	0: Baud Rate Generator 1:SCLK1 Pin Input
SC1MOD0	Serial Channel 1 Mode 0 Register	120AH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Trans-mission data bit 8	0:CTS Disable 1:CTS Enable	0:Receive Disable 1:Receive Enable	Wake up 0:Disable 1:Enable	00:I/O Interface Mode 01:7bit UART Mode 10:8bit UART Mode 11:9bit UART Mode		00:TimerTOTRG 01:Baud Rate Generator 10:Internal clock **φ1** 11:External clock (SCLK1 Input)	
BR1CR	Serial Channel 1 Baud Rate Control Register	120BH	-	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0	0	0	0	0	0
			Fix to "0"	(16-K)/16 divided 0:Disable 1:Enable	00:φT0 01:φT2 10:φT8 11:φT32		Set the frequency divisor "N" 0 to F			
BR1ADD	Serial Channel 1 K setting Register	120CH					BR1K3	BR1K2	BR1K1	BR1K0
			R/W							
			-	-	-	-	0	0	0	0
							Set the frequency divisor "K" (1 to F)			
SC1MOD1	Serial Channel 1 Mode 1 Register	120DH	I2S1	FDPX1						
			R/W	R/W						
			0	0	-	-	-	-	-	-
			IDLE2 0:Stop 1:Operate	I/O interface mode 1:Full duplex 0:Half duplex						

(12) USB Controller (1/)

(13) AD Converter(1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD 0	AD Mode Control Register 0	12B8H	EOCF	ADBF	-	-	ITM0	REPET	SCAN	ADS
			R				R/W			
			0	0	0	0	0	0	0	0
			AD Conversion End Flag 1:END	AD Conversion BUSY Flag 1:Busy	Fix to "0"	Fix to "0"	0: Every 1 time 1: Every 4 times	Repeat mode 0:Single mode 1:Repeat mode	Scan mode 0:Fixed channel mode 1:Channel scan mode	AD Conversion start 1:Start Always read as "0"
ADMOD 1	AD Mode Control Register 1	12B9H	VREFON	I2AD	-	-	-	-	ADCH1	ADCH0
			R/W	R/W				R/W		
			0	0	0	0	0	0	0	0
			Ladder resistance 0:OFF 1:ON	IDLE2 0:Stop 1:Operate	Fix to "0"	Fix to "0"	Fix to "0"	Fix to "0"	Input channel 000:AN0 001:AN1 010:AN2 011:AN3	
ADMOD 2	AD Mode Control Register 1	12BAH	-	-	-	-	-	-	-	ADTRG
										R/W
			0	0	0	0	0	0	0	0
			Always fixed to 0	Always fixed to 0	Always fixed to 0	Always fixed to 0	Always fixed to 0	Always fixed to 0	Always fixed to 0	AD external trigger start control 0: disable 1: enable
ADREG 0L	AD Result Register 0 Low	12A0H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
ADREG 0H	AD Result Register 0 High	12A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
ADREG 1L	AD Result Register 1 Low	12A2H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
ADREG 1H	AD Result Register 1 High	12A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
ADREG 2L	AD Result Register 2 Low	12A4H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
ADREG 2H	AD Result Register 2 High	12A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
ADREG 3L	AD Result Register 3 Low	12A6H	ADR31	ADR30						ADR3RF
			R							R
			Undefined							0
ADREG 3H	AD Result Register 3 High	12A7H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			Undefined							

(14) 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	0
			WDT control 1: enable	Select detecting time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$			Always fixed to “0”.	IDLE2 0: Stop 1: Operate	1: Internally connects WDT out to the reset pin	Always write 0
WDCR	WDT Control Register	1301H	-							
			W							
			-							
			B1H:WDT disable code 4E:WDT clear code							

(15) 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"	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"	40 min.	20 min.	10 min.	8 min.	4 min.	2 min.	1min.
HOURR	Hour Register	1322H			HO5	HO4	HO3	HO2	HO1	HO0
					R/W					
					Undefined					
			"0"	"0"	20 hour (PM/AM)	10 hour	8 hour	4 hour	2 hour	1 hour
DAYR	Day Register	1323H						WE2	WE1	WE0
								R/W		
								Undefined		
			"0"	"0"	"0"	"0"	"0"	WE2	WE1	WE0
DATER	Date Register	1324H			DA5	DA4	DA3	DA2	DA1	DA0
					R/W					
					Undefined					
			"0"	"0"	20 day	10 day	8 day	4 day	2 day	1 day
MONTHR	Month Register	1325H				MO4	MO3	MO2	MO1	MO0
						R/W				
						Undefined				
		PAGE0	"0"	"0"	"0"	10 month	8 month	4 month	2 month	1 month
		PAGE1				--	--	--	--	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 year	40 year	20 year	10 year	8 year	4 year	2 year	1 year
		PAGE1	--	--	--	--	--	--	Leap year setting 00:leap year 01:one year after 10:two year after 11 three year afuter	
PAGER	Page Register (Prohibit RMW)	1327H	INTENA			ADJUST	ENATMR	ENAALM		PAGE
			R/W			W	R/W			R/W
			0			--	Undefined			Undefined
			INTRTC ENABLE			ADJUST	TIMER ENABLE	ALARM ENABLE		PAGE setting
RESTR	Reset Register (Prohibit RMW)	1328H	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	RE3	RE2	RE1	RE0
			W							
			Undefined							
			0: 1HZ	0: 16HZ	1: RESET TIMER	1:RESET ALARM	Always write 0.			

(16) Melody/Alarm Generator

Symbol	Name	Address	7	6	5	4	3	2	1	0
ALM	Alarm – Pattern Reg	1330H	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
			R/W							
			0							
			Alarm –Pattern set							
MELALMC	Melody/ Alarm Control Reg	1331H	FC1	FC0	ALMINV					MELALM
			R/W		R/W					R/W
			0		0					0
			Free-run counter Control 00: Hold 01: Restart 10: Clear 11: Clear & Start		Alarm Frequency Invert 1: Invert	Always write 0				Output Frequency 0: Alarm 1: Melody
MELFL	Melody Frequency L-Reg	1332H	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
			R/W							
			0							
			Melody Frequency set (low 8bit)							
MELFH	Melody Frequency H-Reg	1333H	MELON				ML11	ML10	ML9	ML8
			R/W				R/W			
			0				0			
			Melody counter Control 0: Stop and Clear 1: Start				Melody Frequency set (upper 4bit)			
ALMINT	Alarm Interrupt Enable Reg	1334H				IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
						R/W				
						0				
			Always write 0			INTALM4 to INTALM0 Alarm Interrupt Enable				

(17) NAND Flash Controller(1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ND0FDTR	NAND-Flash Data Transfer Register	1D00H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined							
			Data window to read/write NAND Flash							
ND0FMCR	NAND-Flash Mode Control Register	1CC4H	WE	ECC1	ECC0	CE	PCNT1	PCNT0	ALE	CLE
			R/W							
			0	0	0	0	0	0	0	0
ND0FSR	NAND-Flash Status Register	1CC8H	BUSY							
			R							
			Undefined							
			0: ready 1: busy							
ND0FISR	NAND-Flash Interrupt Status Register	1CCCH								RDY
										R/W
										0
ND0FIMR	NAND-Flash Interrupt Mask Register	1CD0H	INTEN							MRDY
			R/W							R/W
			0							0
			0: disable 1: enable							Mask for RDY
ND0FSPR	NAND-Flash Strobe Pulse Width Register	1CD4H					SPW3	SPW2	SPW1	SPW0
							R/W			
							0			
							Pulse width for /NDRE,/NDWE = fSYS * (This reg's value + 1)			
ND0FRSTR	NAND-Flash Reset Register	1CD8H								RST
										R/W
										0
										Reset Controller
NDCR	NAND-Flash Control Register	01C0H	CHSEL							
			R/W							
			0							
			Channel Selection 0: Ch-0 1: Ch-1							
ND0ECCRD	NAND-Flash ECC Code Register	1CB0H	D7	D6	D5	D4	D3	D2	D1	D0
			R							
			Data window to read ECC-code							

(17) NAND Flash Controller(2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ND1FDTR	NAND-Flash Data Transfer Register	1D00H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined							
			Data window to read/write NAND Flash							
ND1FMCR	NAND-Flash Mode Control Register	1CE4H	WE	ECC1	ECC0	CE	PCNT1	PCNT0	ALE	CLE
			R/W							
			0	0	0	0	0	0	0	0
ND1FSR	NAND-Flash Status Register	1CE8H	BUSY							
			R							
			Undefined							
			0: ready 1: busy							
ND1FISR	NAND-Flash Interrupt Status Register	1CECH								RDY
										R/W
										0
ND1FIMR	NAND-Flash Interrupt Mask Register	1CF0H	INTEN							MRDY
			R/W							R/W
			0							0
			0: disable 1: enable							Mask for RDY
ND1FSPR	NAND-Flash Strobe Pulse Width Register	1CF4H					SPW3	SPW2	SPW1	SPW0
							R/W			
							0			
							Pulse width for /NDRE,/NDWE = fSYS * (This reg's value + 1)			
ND1FRSTR	NAND-Flash Reset Register	1CF8H								RST
										R/W
										0
										Reset Controller
ND1ECCRD	NAND-Flash ECC Code Register	1CB0H	D7	D6	D5	D4	D3	D2	D1	D0
			R							
			Data window to read ECC-code							

(18) I^2S

Symbol	Name	Address	7	6	5	4	3	2	1	0
I2SBUFR	I2S FIFO Buffer (R)	0800H	R15/R7	R14/R6	R13/R5	R12/R4	R11/R3	R10/R2	R9/R1	R8/R0
			W							
			Undefined							
			Register for transmitting buffer(FIFO) (Right-channel)							
I2SBUFL	I2S FIFO Buffer (L)	0808H	L15/L7	L14/L6	L13/L5	L12/L4	L11/L3	L10/L2	L9/L1	L8/L0
			W							
			Undefined							
			Register for transmitting buffer(FIFO) (Left-channel)							
I2SCTL0	I2S Control Register0	080EH	TXE	FMT	BUSY	DIR	BIT	MCK1	MCK0	I2SWCK
			R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
			0	0	0	0	0	0	0	0
			Transmit 0:stop 1:start	Mode 0: I^2S 1:SIO	Status 0:stop 1:under-transmitting	First-BIT 0:MSB 1:LSB	Bit number 0:8bit 1:16bit	Baud-rate 00:fsys 01:fsys/2 10:fsys/4 11:fsys/8		WS clock 0:fs/4 1:TA1OUT
			I2SWLVL	EDGE	I2SFSEL	I2SCKE				SYSCKE
			R/W	R/W	R/W	R/W				R/W
			0	0	0	0				0
			WS level 0:low left 1:high left	clock edge 0:falling 1:rising	Select for Stereo 0:Stereo (2-channel) 1:Monaural (1-channel)	Clock-enable (After transmit) 0:operation 1:stop				System-clock 0:disable 1:enable

6. PACKAGE for TMP92CH21FG

Package Name：P-LQFP144-1616-0.40

Unit : mm