TOSHIBA Original CMOS 32-Bit Microcontroller

TLCS-900/H1 Series

TMP92CH21FG

TOSHIBA CORPORATION

Semiconductor Company

Preface

Thank you very much for making use of Toshiba microcomputer LSIs. Before use this LSI, refer the section, “Points of Note and Restrictions”.

CMOS 32-bit Microcontroller

TMP92CH21FG/JTMP92CH21

1. Outline and Device Characteristics

The TMP92CH21 is a high-speed advanced 32-bit Microcontroller developed for controlling equipment which processes mass data.

The TMP92CH21 has a high-performance CPU (900/H1 CPU) and various built-in I/Os.

The TMP92CH21FG is housed in a 144-pin flat package. The 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
- 16 Mbytes of linear address space
- General-purpose register and register banks
- Micro DMA: 8 channels (250 ns/4 bytes at f_{SYS} = 20 MHz, best case)

(2) Minimum instruction execution time: 50 ns (at f_{SYS} = 20 MHz)

(3) Internal memory

- Internal RAM: 16 Kbytes (can be used for program, data and display memory)
- Internal ROM: 8 Kbytes (used as boot program)
  Possible downloading of user program through either USB,
  UART or NAND flash.

(4) External memory expansion

- Expandable up to 512 Mbytes (shared program/data area)
- Can simultaneously support 8,- 16- or 32-bit width external data bus
  ... dynamic data bus sizing
- Separate bus system

(5) Memory controller

- Chip select output: 4 channels

(6) 8-bit timers: 4 channels

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

(8) General-purpose serial interface: 2 channels

- UART/synchronous mode: 2 channels (channel 0 and 1)
- IrDA ver.1.0 (115 kbps) mode selectable: 1 channel (channel 0)

(9) USB (universal serial bus) controller: 1 channel

- Compliant with USB ver.1.1
- Full-speed (12 Mbps) (Low-speed is not supported.)
- Endpoints spec
  Endpoint 0: Control 64 bytes* 1-FIFO
  Endpoint 1: BULK (out) 64 bytes* 2-FIFO
  Endpoint 2: BULK (in) 64 bytes* 2-FIFO
  Endpoint 3: Interrupt (in) 8 bytes* 1-FIFO
- Descriptor RAM: 384 bytes

(10) I^{2}S (Inter-IC sound) interface: 1 channel

- I^{2}S bus mode/SIO mode selectable (Master, transmission only)
- 32-byte FIFO buffer

(11) LCD controller

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

(12) SDRAM controller: 1 channel

- Supports 16 M, 64 M, 128 M, 256 M, and up to 512-Mbit SDR (Single Data Rate)-SDRAM
- Possible to execute instruction on SDRAM

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

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

(15) 10-bit AD converter: 4 channels

(16) Touch screen interface

- Available to reduce external components

(17) Watchdog timer

(18) Melody/alarm generator

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

(19) MMU

- Expandable up to 512 Mbytes (3 local area/8 bank method)
- Independent bank for each program, read data, write data and LCD display data

(20) Interrupts: 50 interrupt

- 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)

(21) Input/output ports: 82 pins (Except Data bus (16bit), Address bus (24bit) and $\overline{\text{RD}}$ pin)

(22) NAND flash interface: 2 channels

- Direct NAND flash connection capability
- ECC calculation (for SLC- type)

(23) Stand-by function

- Three HALT modes: IDLE2 (programmable), IDLE1, STOP
- Each pin status programmable for stand-by mode

(24) Triple-clock controller

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

(25) Operating voltage:

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

(26) Package:

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

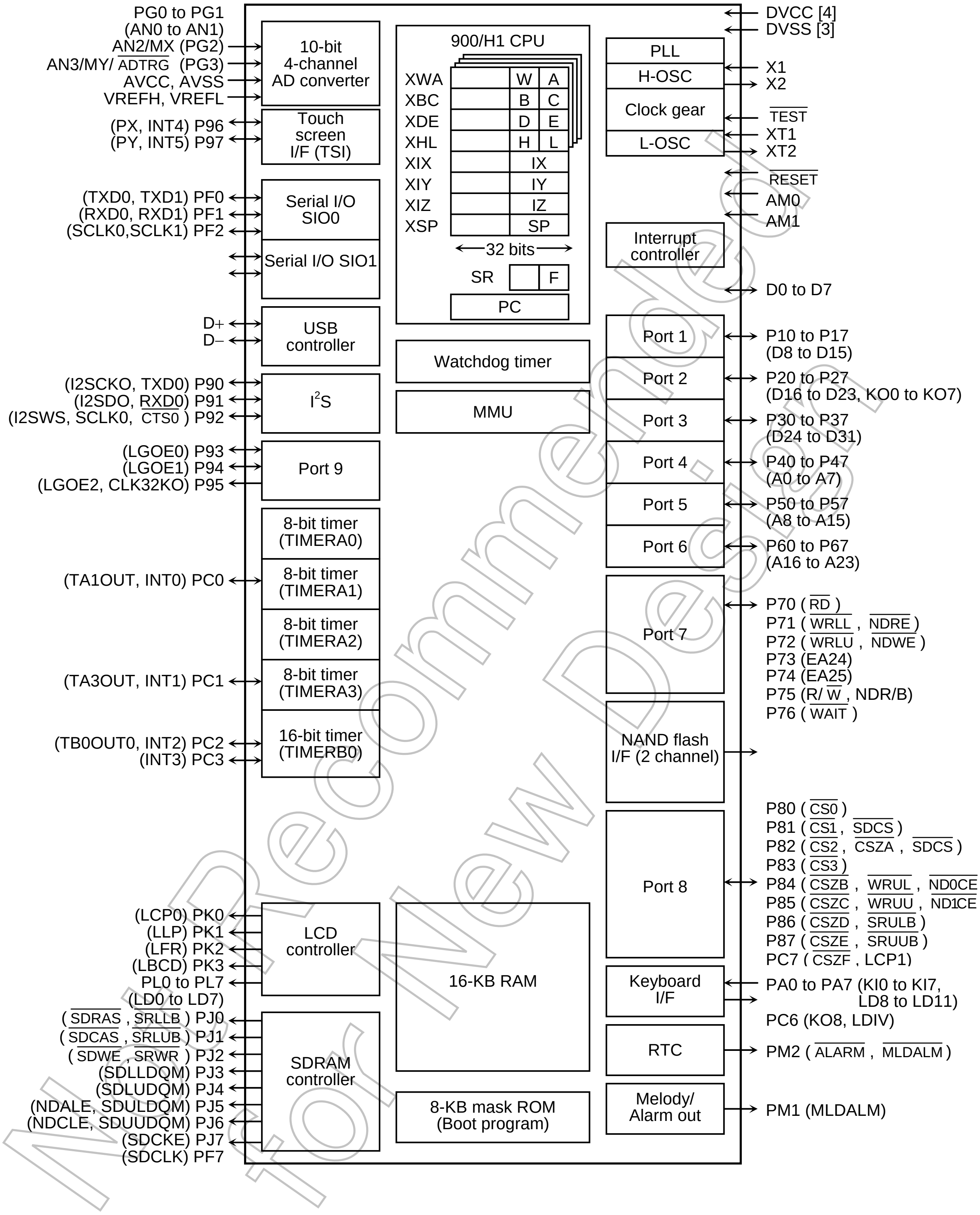


Figure 1.1 TMP92CH21 Block Diagram

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

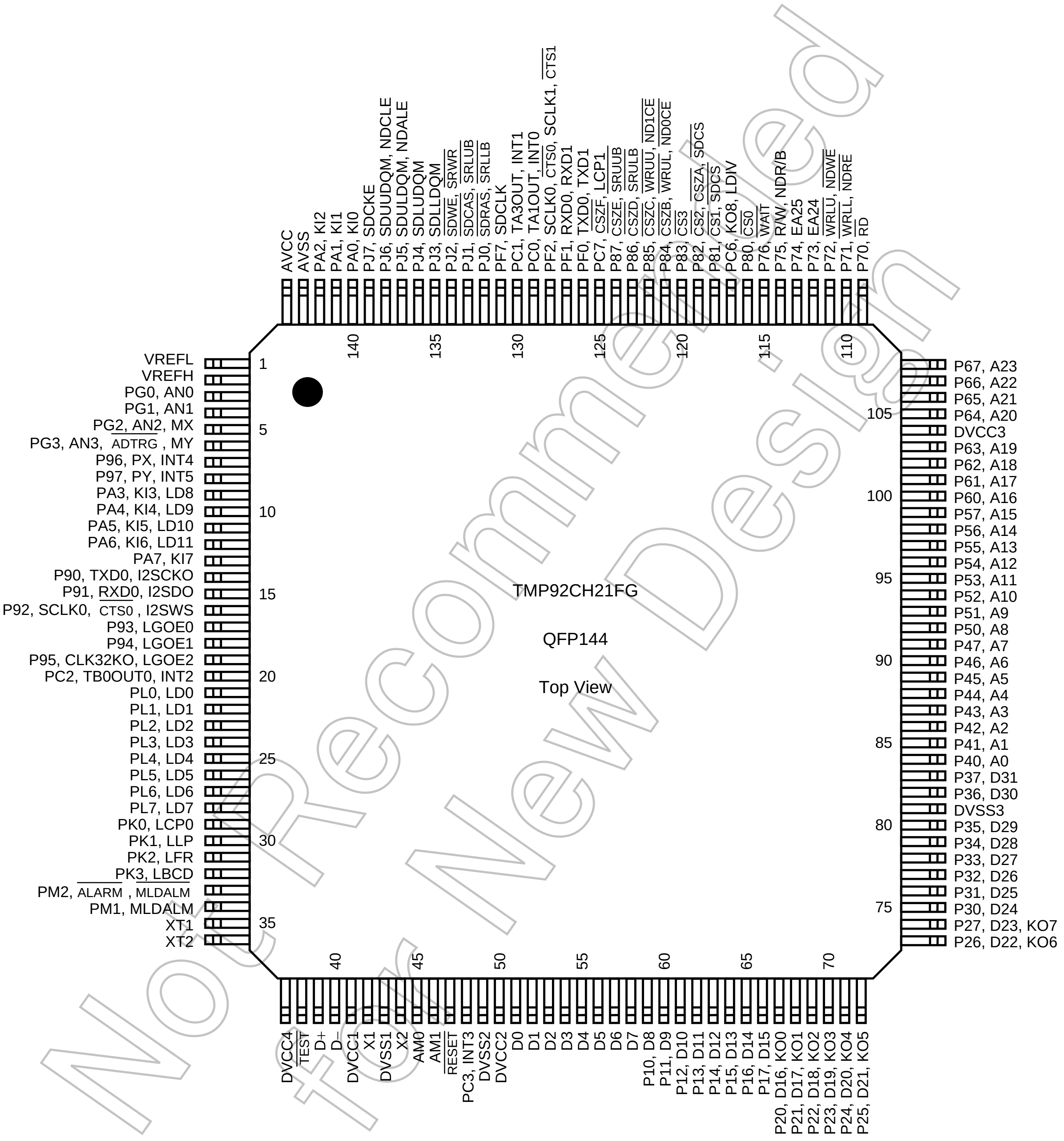


Figure 2.1.1 Pin Assignment Diagram (144-pin QFP)

2.2 PAD Assignment

(Chip size 5.98 mm × 6.42 mm)

Table 2.2.1 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	DVCC4	–2465	–3072	85	P41	2848	–686	133	PJ1	–775	3065
38	$\overline{\text{TEST}}$	–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	$\overline{\text{RESET}}$	–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

Table 2.3.1 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 to P37 D24 to D31	8	I/O I/O	Port 3: I/O port input or output specifiable in units of bits Data24: Data bus 24 to 31
P40 to P47 A0 to A7	8	Output Output	Port 4: Output port Address: Address bus 0 to 7
P50 to P57 A8 to A15	8	Output Output	Port 5: Output port 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 $\overline{\text{RD}}$	1	Output Output	Port70: Output port Read: Outputs strobe signal to read external memory
P71 $\overline{\text{WRLL}}$ $\overline{\text{NDRE}}$	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 $\overline{\text{WRLU}}$ $\overline{\text{NDWE}}$	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	Output Output	Port 73: Output port Extended Address 24
P74 EA25	1	Output Output	Port 74: Output port Extended Address 25
P75 R/$\overline{\text{W}}$ NDR/B	1	I/O Output Input	Port 75: I/O port Read/Write: 1 represents read or dummy cycle; 0 represents write cycle NAND flash ready (1)/Busy (0) input
P76 $\overline{\text{WAIT}}$	1	I/O Input	Port 76: I/O port Wait: Signal used to request CPU bus wait

Table 2.3.2 Pin Names and Functions (2/5)

Pin Name	Number of Pins	I/O	Function
P80 $\overline{\text{CS0}}$	1	Output Output	Port80: Output port Chip select 0: Outputs “low” when address is within specified address area
P81 $\overline{\text{CS1}}$ $\overline{\text{SDCS}}$	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 $\overline{\text{CS2}}$ $\overline{\text{CSZA}}$ $\overline{\text{SDCS}}$	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 $\overline{\text{CS3}}$	1	Output Output	Port83: Output port Chip select 3: Outputs “low” when address is within specified address area
P84 $\overline{\text{WRUL}}$ $\overline{\text{CSZB}}$ $\overline{\text{ND0CE}}$	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 flash 0: Outputs “0” when NAND flash 0 is enabled
P85 $\overline{\text{WRUU}}$ $\overline{\text{CSZC}}$ $\overline{\text{ND1CE}}$	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 flash 1: Outputs “0” when NAND flash 1 is enabled
P86 $\overline{\text{CSZD}}$ $\overline{\text{SRULB}}$	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 $\overline{\text{CSZE}}$ $\overline{\text{SRUUB}}$	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 I^{2}S clock output
P91 RXD0 I2SDO	1	I/O Input Output	Port91: I/O port (Schmitt-input) Serial 0 receive data I^{2}S data output
P92 SCLK0 $\overline{\text{CTS0}}$ I2SWS	1	I/O I/O Input Output	Port92: I/O port (Schmitt-input) Serial 0 clock I/O Serial 0 data send enable (Clear to send) I^{2}S 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 CLK32KO LGOE2	1	Output Output Output	Port95: Output port Output fs (32.768 kHz) clock Output enable-2 for external TFT-LCD driver
P96 INT4 PX	1	Input Input Output	Port 96: Input port (Schmitt-input) Interrupt request pin4: Interrupt request with programmable rising/falling edge X-Plus: Pin connectted to X+ for touch screen panel
P97 INT5 PY	1	Input Input Output	Port 97: Input port (Schmitt-input) 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 (Schmitt input, with pull-up resistor) Key input 0 to 2: Pin used for key-on wakeup 0 to 2
PA3 to PA6 KI3 to KI6 LD8 to LD11	4	Input Input Output	Port: A3 to A6 port: Pin used to input ports (Schmitt input, with pull-up resistor) Key input 3 to 6: Pin used for key-on wakeup 3 to 6 Data bus 8 to 11for LCD driver
PA7 KI7	1	Input Input	Port: A7 port: Pin used to input ports (Schmitt input, with pull-up resistor) Key input 7: Pin used for key-on wakeup 7

Table 2.3.3 Pin Names and Functions (3/5)

Pin Name	Number of Pins	I/O	Function
PC0	1	I/O	Port C0: I/O port (Schmitt-input)
INT0		Input	Interrupt request pin 0: Interrupt request pin with programmable level/rising/falling edge
TA1OUT		Output	8-bit timer 1 output: Timer 1 output
PC1	1	I/O	Port C1: I/O port (Schmitt-input)
INT1		Input	Interrupt request pin 1: Interrupt request pin with programmable rising/falling edge
TA3OUT		Output	8-bit timer 3 output: Timer 3 output
PC2	1	I/O	Port C2: I/O port (Schmitt-input)
INT2		Input	Interrupt request pin 2: Interrupt request pin with programmable rising/falling edge
TB0OUT0		Output	Timer B0 output
PC3	1	I/O	Port C3: I/O port (Schmitt-input)
INT3		Input	Interrupt request pin 3: Interrupt request pin with programmable rising/falling edge
PC6	1	I/O	Port C6: I/O port
KO8		Output	Key Output 8: Pin used of key-scan strobe (Open-drain output programmable)
LDIV		Output	Data invert enable for external TFT-LCD driver
PC7	1	I/O	Port C7: I/O port
$\overline{\text{CSZF}}$		Output	Expand chip select: ZF: Outputs “0” when address is within specified address area
LCP1		Output	Shift-clock-1 for external TFT-LCD driver
PF0	1	I/O	Port F0: I/O port (Schmitt-input)
TXD0		Output	Serial 0 send data: Open-drain output programmable
TXD1		Output	Serial 1 send data: Open-drain output programmable
PF1	1	I/O	Port F1: I/O port (Schmitt-input)
RXD0		Input	Serial 0 receive data
RXD1		Input	Serial 1 receive data
PF2	1	I/O	Port F2: I/O port (Schmitt-input)
SCLK0		I/O	Serial 0 clock I/O
$\overline{\text{CTS0}}$		Input	Serial 0 data send enable (Clear to send)
SCLK1		I/O	Serial 1 clock I/O
$\overline{\text{CTS1}}$		Input	Serial 1 data send enable (Clear to send)
PF7	1	Output	Port F7: Output port
SDCLK		Output	Clock for SDRAM (When SDRAM is not used, SDCLK can be used as system clock)
PG0 to PG1	2	Input	Port G0 to G1 port: Pin used to input ports
AN0 to AN1		Input	Analog input 0 to 1: Pin used to Input to AD conveter
PG2	1	Input	Port G2 port: Pin used to input ports
AN2		Input	Analog input 2: Pin used to Input to AD conveter
MX		Output	X-Minus: Pin connectted to X− for touch screen panel
PG3	1	Input	Port G3 port: Pin used to input ports
AN3		Input	Analog input 3: Pin used to input to AD conveter
MY		Output	Y-Minus: Pin connectted to Y− for touch screen panel
$\overline{\text{ADTRG}}$		Intput	AD trigger: Signal used to request AD start

Table 2.3.4 Pin Names and Functions (4/5)

Pin Name	Number of Pins	I/O	Function
PJ0 $\overline{\text{SDRAS}}$ $\overline{\text{SRLLB}}$	1	Output Output Output	Port J0: Output port Row address strobe for SDRAM Data enable for SRAM on pins D0 to D7
PJ1 $\overline{\text{SDCAS}}$ $\overline{\text{SRLUB}}$	1	Output Output Output	Port J1: Output port Column address strobe for SDRAM Data enable for SRAM on pins D8 to D15
PJ2 $\overline{\text{SDWE}}$ $\overline{\text{SRWR}}$	1	Output Output Output	Port J2: Output port Write enable for SDRAM Write for SRAM: Strobe signal for writing data
PJ3 SDLLDQM	1	Output Output	Port J3: Output port Data enable for SDRAM on pins D0 to D7
PJ4 SDLUDQM	1	Output Output	Port J4: Output port Data enable for SDRAM on pins D8 to D15
PJ5 SDULDQM NDALE	1	I/O Output Output	Port J5: I/O port Data enable for SDRAM on pins D16 to D23 Address latch enable for NAND flash
PJ6 SDUUDQM NDCLE	1	I/O Output Output	Port J6: I/O port Data enable for SDRAM on pins D24 to D31 Command latch enable for NAND flash
PJ7 SDCKE	1	Output Output	Port J7: Output port Clock enable for SDRAM
PK0 LCP0	1	Output Output	Port K0: Output port LCD driver output pin
PK1 LLP	1	Output Output	Port K1: Output port LCD driver output pin
PK2 LFR	1	Output Output	Port K2: Output port LCD driver output pin
PK3 LBCD	1	Output Output	Port K3: Output port LCD driver output pin
PL0 to PL3 LD0 to LD3	4	Output Output	Port L0 to L3: Output port Data bus for LCD driver
PL4 to PL7 LD4 to LD7	4	I/O Output	Port L4 to L7: I/O port Data bus for LCD driver
$\overline{\text{TEST}}$	1	Input	Connect to VCC.
PM1 MLDALM	1	Output Output	Port M1: Output port Melody/alarm output pin
PM2 $\overline{\text{ALARM}}$ $\overline{\text{MLDALM}}$	1	Output Output Output	Port M2: Output port RTC alarm output pin Melody/alarm output pin (inverted)

Note: The output functions SDULDQM, NDALE of PJ5-pin and SDUUDQM, NDCLE of PJ6-pin cannot be used simultaneously. Therefore, 32-bit SDRAM and NAND-Flash cannot be used at the same time.

Table 2.3.5 Pin Names and Functions (5/5)

Pin Name	Number of Pins	I/O	Function
D+, D−	2	I/O	USB-data connecting pin Connect pull-up resistor to both pins to avoid through current when USB is not in use.
AM0, AM1	2	Input	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
$\overline{\text{RESET}}$	1	Input	Reset: Initializes TMP92CH21 (with pull-up resistor, Schmitt input)
VREFH	1	Input	Pin for reference voltage input to AD converter (H)
VREFL	1	Input	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	4	−	Power supply pins (All V_{CC} pins should be connected to the power supply pin)
DVSS	3	−	GND pins (0 V) (All pins should be connected to GND (0 V))

Note: Use a 9.0 MHz oscillator at pins X1/X2 when USB is used.

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 (TLCS-900/H1 CPU)

3.1.1 CPU Outline

The TLCS-900/H1 CPU is a high-speed, high-performance CPU based on the TLCS-900/L1 CPU. The TLCS-900/H1 CPU has an expanded 32-bit internal data bus to process instructions more quickly.

The following is an outline of the CPU:

Table 3.1.1 TMP92CH21 Outline

Parameter	TMP92CH21
Width of CPU address bus	24 bits
Width of CPU data bus	32 bits
Internal operating frequency	Max 20 MHz
Minimum bus cycle	1-clock access (50 ns at f_{SYS} = 20MHz)
Internal RAM	32-bit 1-clock access
Internal boot ROM	32-bit 2-clock access
Internal I/O	8- or 16-bit 2-clock access or 8- or 16-bit 5 to 6-clock access
External SRAM, Masked ROM	8- or 16- or 32-bit 2-clock access (waits can be inserted)
External SDRAM	16- or 32-bit min. 1-clock access
External NAND flash	8-bit min. 4-clock access (waits can be inserted)
Minimum instruction execution cycle	1-clock (50 ns at f_{SYS} =20MHz)
Conditional jump	2-clock (100 ns at f_{SYS} =20MHz)
Instruction queue buffer	12 bytes
Instruction set	Compatible with TLCS-900/L1 (LDX instruction is deleted)
CPU mode	Maximum mode only
Micro DMA	8 channels

3.1.2 Reset Operation

When resetting the TMP92CH21, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input low for at least 20 system clocks (16 μs at fc = 40 MHz).

At reset, since the clock doubler (PLL) is bypassed and the clock-gear is set to 1/16, the system clock operates at 1.25 MHz (fc = 40 MHz).

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 to FFFF02H:

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

- Sets the stack pointer (XSP) to 00000000H.
- Sets bits <IFF2:0> of the status register (SR) to 111 (thereby setting the interrupt level mask register to level 7).
- Clears bits <RFP1:0> of the status register to 00 (there by 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 shown in the “Special Function Register” table 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.

Memory controller operation cannot be ensured until the power supply becomes stable after power-on reset. External RAM data provided before turning on the TMP92CH21 may be corrupted because the control signals are unstable until the power supply becomes stable after power on reset.

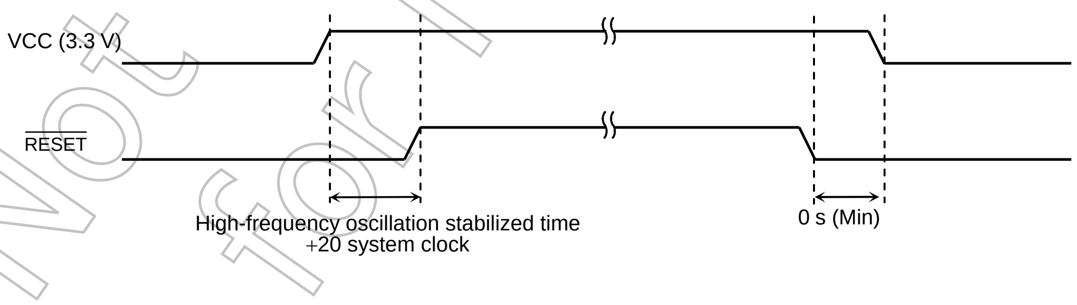


Figure 3.1.1 Power on Reset Timing Example

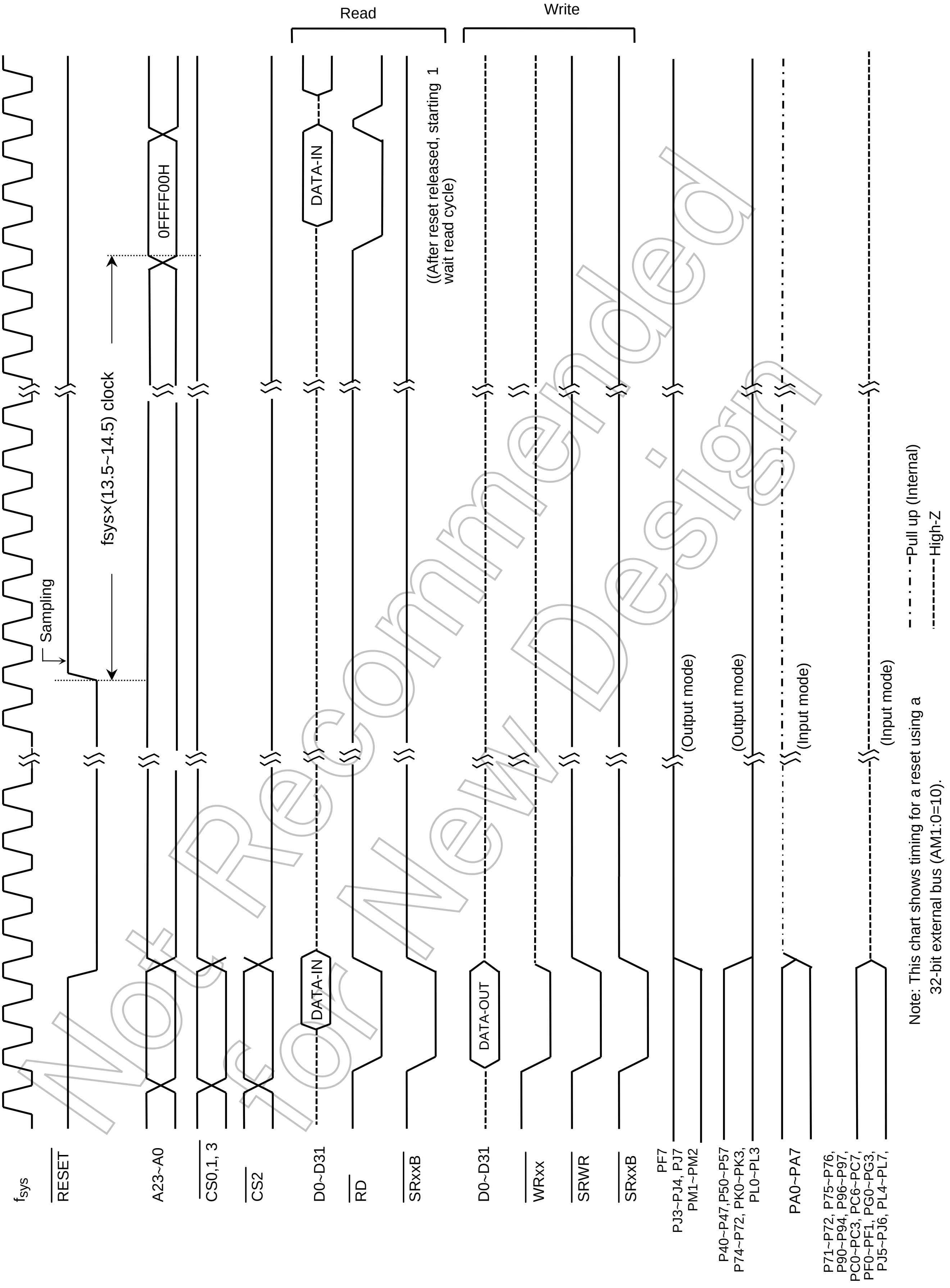
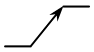


Figure 3.1.2 TMP92CH21 Reset Timing Chart

3.1.3 Setting of AM0 and AM1

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

Table 3.1.2 Operation Mode Setup Table

Operation Mode	Mode Setup Input Pin		
	$\overline{\text{RESET}}$	AM1	AM0
16-bit external bus starting (MULTI 16 mode)		0	1
32-bit external bus starting (MULTI 32 mode)		1	0
Boot (32-bit internal MROM) starting (BOOT mode)		1	1

3.2 Memory Map

Figure 3.2.1 is a memory map of the TMP92CH21.

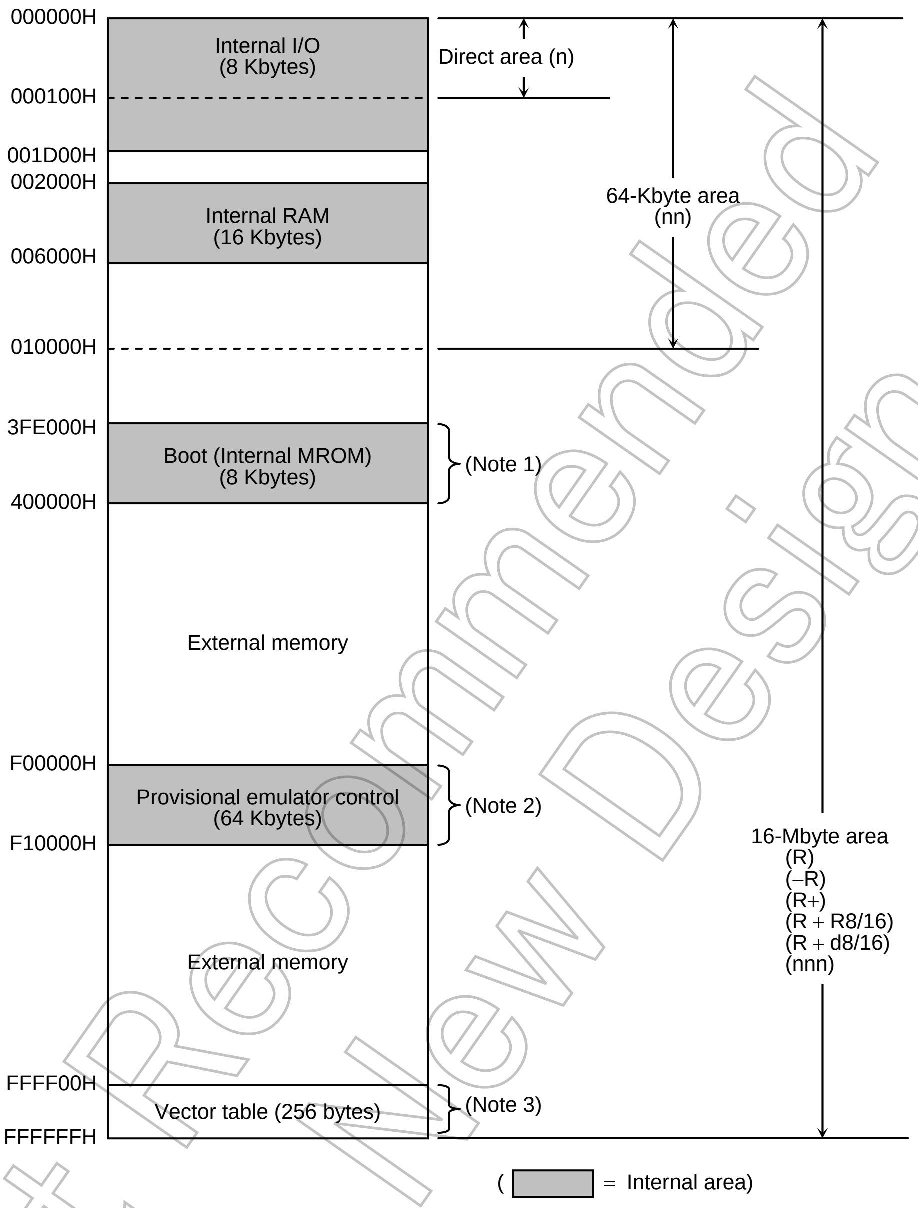


Figure 3.2.1 Memory Map

Note 1: Boot program (Internal MROM) is mapped only for BOOT mode. For other starting modes, its area (3FE000H to 3FFFFFH) is mapped to external-memory.

Note 2: The Provisional emulator control area, mapped F00000H to F0FFFFH after reset, is for emulator use and so is not available. When emulator $\overline{\text{WR}}$ signal and $\overline{\text{RD}}$ signal are asserted, this area is accessed. Ensure external memory is used.

Note 3: Do not use the last 16-byte area (FFFFF0H to FFFFFFH). This area is reserved for an emulator.

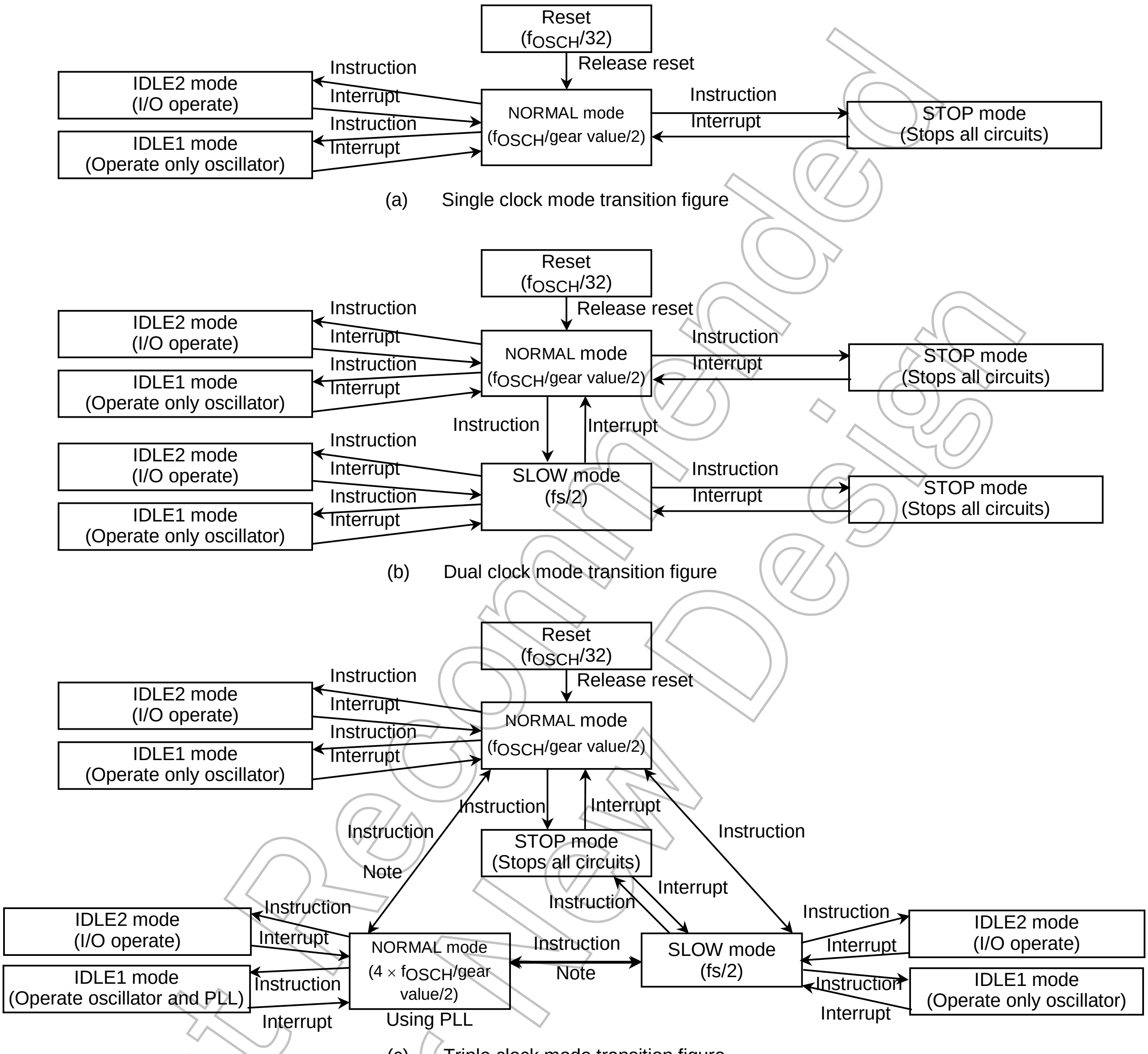
3.3 Clock Function and Stand-by Function

The TMP92CH21 contains (1) clock gear, (2) clock doubler (PLL), (3) stand-by controller and (4) noise reduction circuits. They are used for low power, low noise systems.

This chapter is organized as follows:

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 1: It is not possible 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> register)

Note 2: When shifting from NORMAL mode with use of PLL to NORMAL mode, execute the following setting in the same order.

1) Change CPU clock (PLLCR0<FCSEL> ← “0”)

2) Stop PLL circuit (PLLCR1<PLLON> ← “0”)

Note 3: It is not possible to shift from NORMAL mode with use of PLL to STOP mode directly.

NORMAL mode should be set once before shifting to STOP mode. (Sstop the high-frequency oscillator after stopping 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 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 as one state.

3.3.1 Block Diagram of System Clock

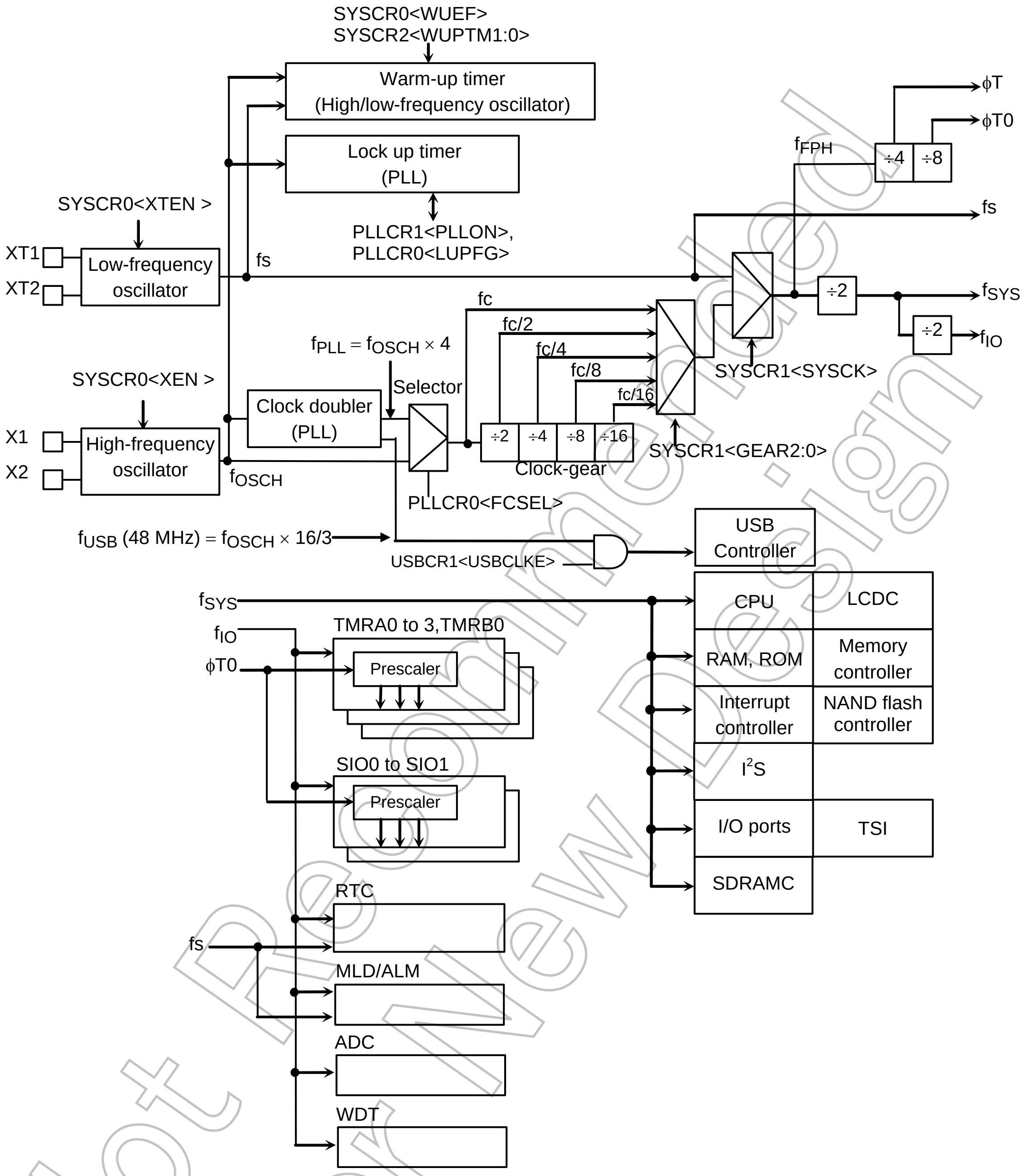


Figure 3.3.2 Block Diagram of System Clock

Table 3.3.1 Selection Example for f_{OSCH}

	High-frequency Oscillation: f_{OSCH}	System Clock: f_{SYS}	USB Clock: f_{USB}
(a) USB in use, with PLL	9.0 MHz	18 MHz	48 MHz
(b) USB not in use, with PLL	10.0 MHz (max)	20 MHz (max)	–
(c) USB not in use, without PLL	40.0 MHz (max)	20 MHz (max)	–

Note: When using USB, the high-frequency oscillator should be 9.0 MHz.

3.3.2 SFR

		7	6	5	4	3	2	1	0
SYSCR0 (10E0H)	Bit symbol	XEN	XTEN				WUEF		
	Read/Write	R/W					R/W		
	Reset state	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		

		7	6	5	4	3	2	1	0
SYSCR1 (10E1H)	Bit symbol					SYSCK	GEAR2	GEAR1	GEAR0
	Read/Write					R/W	R/W		
	Reset state					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)		

		7	6	5	4	3	2	1	0
SYSCR2 (10E2H)	Bit symbol	–		WUPTM1	WUPTM0	HALTM1	HALTM0		
	Read/Write	R/W		R/W	R/W	R/W	R/W		
	Reset state	0		1	0	1	1		
	Function	Always write “0”		Warm-up timer 00: Reserved 01: 2^{8}/input frequency 10: 2^{14}/input frequency 11: 2^{16}/input frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			

Note 1: The unassigned registers, SYSCR0<bit5:3>, SYSCR0<bit1:0>, SYSCR1<bit7:4>, and SYSCR2<bit6, bit1:0> are read as undefined value.

Note 2: Low-frequency oscillator is enabled on reset.

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
	Reset state	0					0	1	1
	Function	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 (10E4H)	Bit symbol	Switch the protect ON/OFF by writing the following to 1st-KEY, 2nd-KEY 1st-KEY: write in sequence EMCCR1 = 5AH, EMCCR2 = A5H 2nd-KEY: write in sequence EMCCR1 = A5H, EMCCR2 = 5AH							
	Read/Write								
	Reset state								
	Function								
EMCCR2 (10E5H)	Bit symbol								
	Read/Write								
	Reset state								
	Function								

Note: When restarting the oscillator from the stop oscillation state (e.g. restarting the oscillator in STOP mode), set EMCCR0<DRVOSCH>, <DRVOSCL>="1".

Figure 3.3.4 SFR for System Clock

PLLCR0
(10E8H)

	7	6	5	4	3	2	1	0
Bit symbol		FCSEL	LUPFG					
Read/Write		R/W	R					
Reset state		0	0					
Function		Select fc clock 0: f_{OSCH} 1: f_{PLL}	Lock up timer status flag 0: Not end 1: End					

Note: Ensure that the 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							
Reset state	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							
Reset state	1	1	1	1	1	1	1	1
Function	Output/input buffer drive-register for stand-by mode							

(Purpose and use)

This register is used to set each pin status at stand-by mode.

All ports have registers of the format shown above. ("x" indicates the 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 the CPU has executed the "Halt" instruction.

This is the case regardless of stand-by mode (IDLE2, IDLE1 or STOP).

The output/input buffer control table is shown below.

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

Note 1: OE denotes an output enable signal before stand-by mode.
Basically, PxCR is used as OE.

Note 2: "n" in PxnD denotes the 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<GEAR2:0> 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 <GEAR2:0> = 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 40 MHz 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<WUPTM1:0>.

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 Warm-up Times

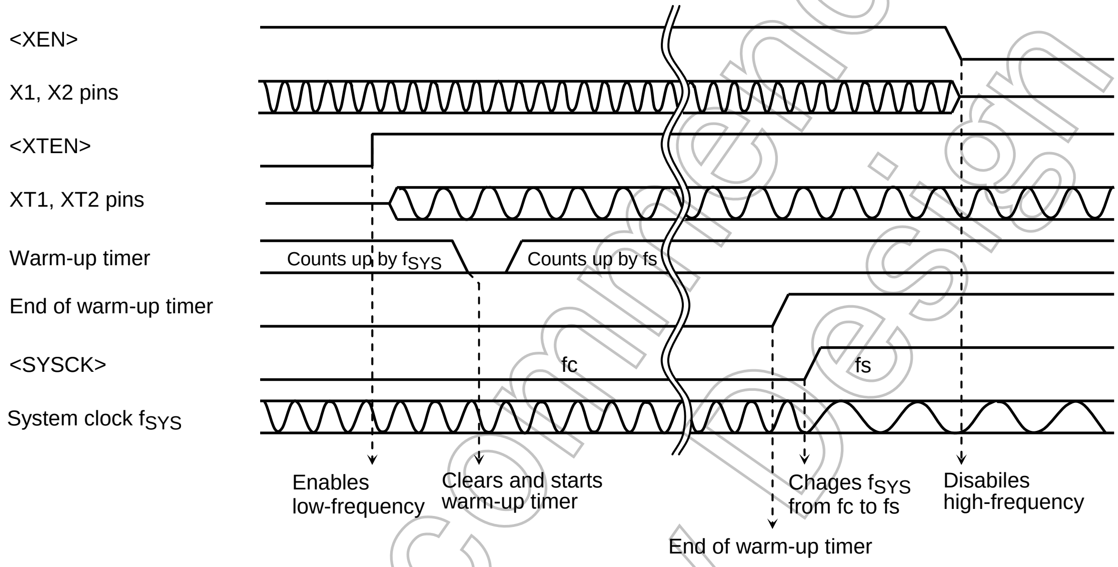
at f_{OSCH} = 40 MHz, fs = 32.768 kHz

Warm-up Time SYSCR2 <WUPTM1:0>	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),  0 X 1 1 – – X X B  ;  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)                   ; }
          JR     NZ, WUP                       ; } Detects stopping of warm-up timer.
          SET    3, (SYSCR1)                   ;  Changes fSYS from fc to fs.
          RES    7, (SYSCR0)                   ;  Disables high-frequency oscillation.
```

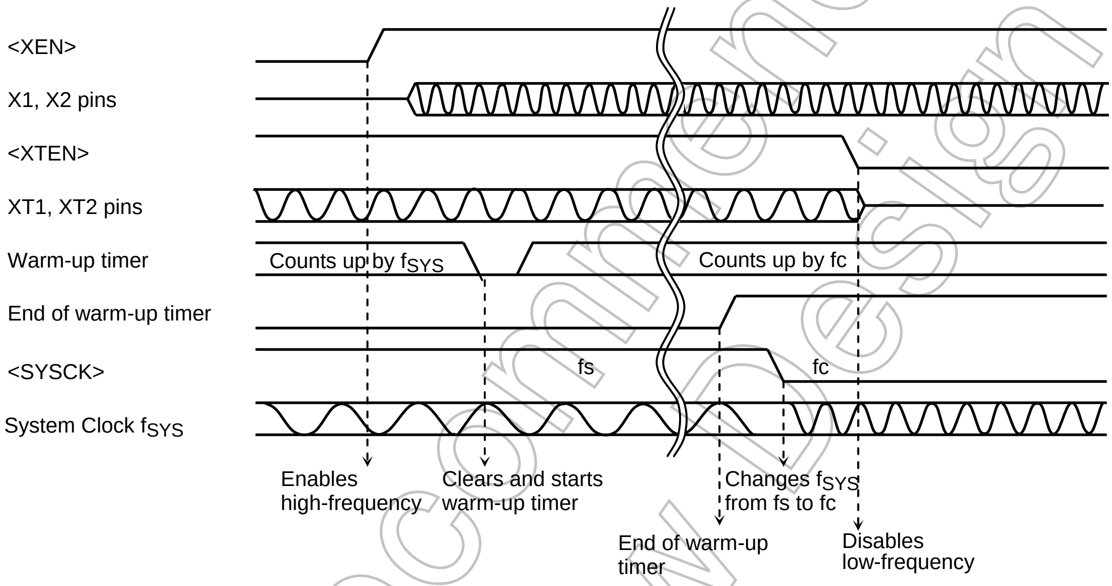
X: Don’t care, –: No change



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),  0 X 1 0 – – X X B  ;    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)                   ; }
           JR     NZ, WUP                       ; }  Detects stopping of warm-up timer.
           RES    3, (SYSCR1)                   ;    Changes fSYS from fs to fc.
           RES    6, (SYSCR0)                   ;    Disables low-frequency oscillation.
```

X: Don't care, –: No change



(2) Clock gear controller

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

Example 3: Changing to a high-frequency gear

```
SYSCR1      EQU     10E1H

            LD      (SYSCR1), XXXX0000B   ;   Changes fSYS to fc/2.
            LD      (DUMMY), 00H          ;   Dummy instruction
      X: Don't care
```

(High-speed clock gear changing)

To change the clock gear, write the register value to the SYSCR1<GEAR2:0> register.It is necessary for the warm-up time to elapse before the change occurs after writing the register value.

There is the possibility that the instruction following the clock gear changing instruction is executed by the clock gear before changing.To execute the instruction following 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}. A low-speed-frequency oscillator can be used, even though the internal clock is high-frequency.

A reset initializes PLL to stop status, so setting to PLLCR0, PLLCR1 register is needed before use.

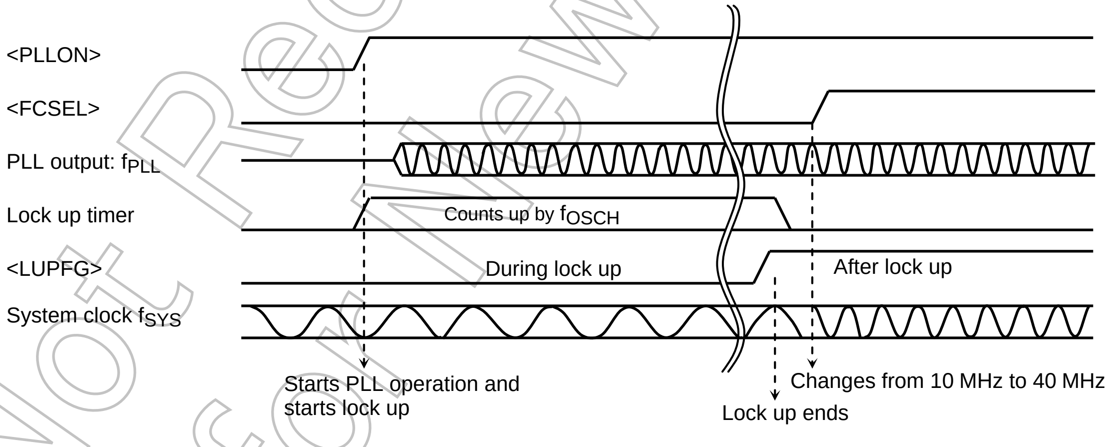
As with an oscillator, this circuit requires time to stabilize. This is called the lock up time and it is measured by a 16-stage binary counter. Lock up time is about 1.6 ms at f_{OSCH} = 10 MHz.

Note 1: Input frequency range for PLL
The input frequency range (High-frequency oscillation) for PLL is as follows:
f_{OSCH} = 6 to 10 MHz (V_{CC} = 3.0 to 3.6 V)

Note 2: PLLCR0<LUPFG>
The logic of PLLCR0<LUPFG> is different from 900/L1's DFM.
Exercise care in determining the end of lock up time.
The following is an example of settings for PLL starting and PLL stopping.

Example 1: PLL starting

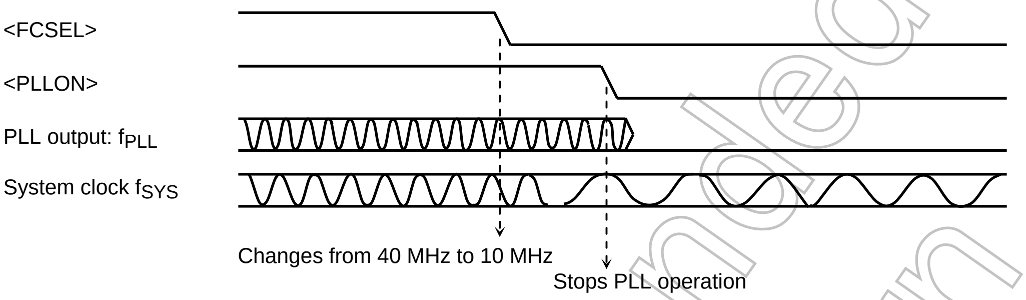
```
PLLCR0      EQU    10E8H
PLLCR1      EQU    10E9H
            LD     (PLLCR1),   1 X X X X X X X B  ;    Enables PLL operation and starts lock up.
LUP:        BIT    5, (PLLCR0)                    ;}
            JR     Z, LUP                         ;}   Detects end of lock up.
            LD     (PLLCR0),   X 1 X X X X X X B  ;    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



Limitations on the use of PLL

1. It is not possible to execute PLL enable/disable control in the SLOW mode (fs) (writing to PLLCR0 and PLLCR1).
   PLL should be controlled in the NORMAL mode.

2. When stopping PLL operation during PLL use, execute the following settings in the same order.

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

3. When stopping the high-frequency oscillator during PLL use, stop PLL before stopping the high-frequency oscillator.

Examples of settings are shown below:

(1) Start up/change control

(OK) Low-frequency oscillator operation mode (fs) (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),     1 1 – – – 1 – – B ;   High-frequency oscillator start/warm-up start
WUP:    BIT     2, (SYSCR0)                     ; }
        JR      NZ, WUP                         ; } Check for warm-up end flag
        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 lock up end flag
        LD      (PLLCR0),     – 1 – – – – – – B ;   Change the system clock fOSCH to fPLL
```

(OK) Low-frequency oscillator operation mode (fs) (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, (PLLCR0)                     ; }
        JR      Z, LUP                          ; } Check for lock up end flag
        LD      (PLLCR0),     – 1 – – – – – – B ;   Change the system clock fOSCH to fPLL
```

(Error) Low-frequency oscillator operation mode (fs) (high-frequency oscillator STOP) → High-frequency oscillator start up → PLL start up → PLL use mode (f_{PLL})

```
        LD      (SYSCR0),     1 1 – – – 1 – – B ;   High-frequency oscillator start/warm-up start
WUP:    BIT     2, (SYSCR0)                     ; }
        JR      NZ, WUP                         ; } Check for warm-up end flag
        LD      (PLLCR1),     1 – – – – – – – B ;   PLL start-up/lock up start
LUP:    BIT     5, (PLLCR0)                     ; }
        JR      Z, LUP                          ; } Check for lock up end flag
        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 (fs) → 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
```

(Error) PLL use mode (f_{PLL}) → Low-frequency oscillator operation mode (fs) → 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),     – – – – 0 1 – – B ;   Set the STOP mode
                                             (This command can be executed 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
```

(Error) PLL use mode (f_{PLL}) → Set the STOP mode → Halt (High-frequency oscillator stop)

```
LD      (SYSCR2),     – – – – 0 1 – – 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

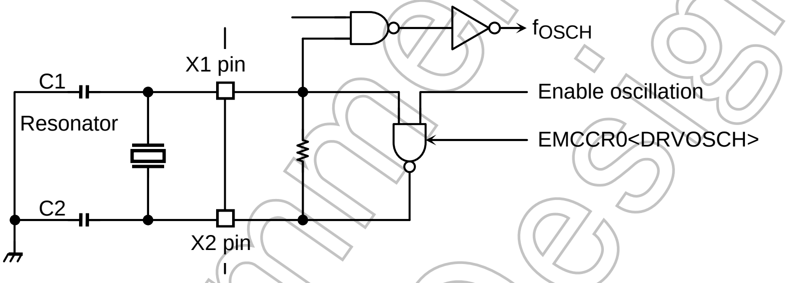
(4) SFR protection of register contents

(1) Reduced drivability for high-frequency oscillator

(Purpose)

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

(Block diagram)



(Setting method)

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

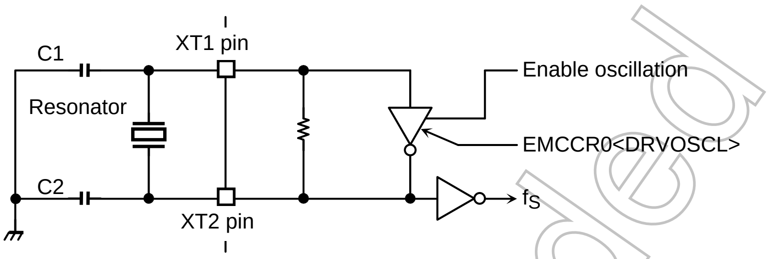
Note: This function (EMCCR0<DRVOSCH> = “0”) is available when f_{OSCH} = 6 to 10 MHz.

(2) Reduced drivability for low-frequency oscillator

(Purpose)

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

(Block diagram)



(Setting method)

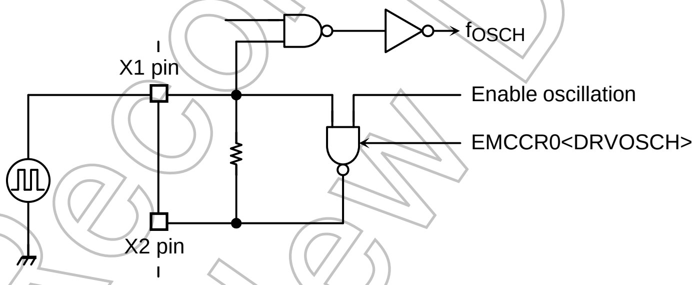
The drive ability of the oscillator is reduced by writing 0 to the EMCCR0<DRVOSCL> register. At reset, <DRVOSCL> is initialized to "1".

(3) Single drive for high-frequency oscillator

(Purpose)

Remove the need for twin drives and prevent operational errors caused by noise input to X2 pin when an 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's output is always "1".

At reset, <EXTIN> is initialized to "0".

(4) Runaway prevention using SFR protection register

(Purpose)

Prevention of program runaway caused by introduction of noise.

Write operations to a specified SFR are prohibited so that the program is protected from runaway caused by stopping of the clock or by changes to the memory control register (memory controller, MMU) which prevent fetch operations.

Runaway error handling is also facilitated 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) becomes possible by setting up a double key to EMCCR1 and EMCCR2 registers.

(Double key)

1st KEY: writes in sequence, 5AH at EMCCR1 and A5H at EMCCR2

2nd KEY: writes in sequence, A5H at EMCCR1 and 5AH at EMCCR2

Protection state can be confirmed by reading EMCCR0<PROTECT>.

At reset, protection becomes OFF.

INTP0 interruption also occurs when a write operation to the specified SFR is executed with protection in the ON state.

3.3.6 Stand-by 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<HALTM1:0> register and each pin-status is set according to the PxDR register, as shown below:

PxDR (xxxxH)

	7	6	5	4	3	2	1	0
Bit symbol	Px7D	Px6D	Px5D	Px4D	Px3D	Px2D	Px1D	Px0D
Read/Write	R/W							
Reset state	1	1	1	1	1	1	1	1
Function	Output/input buffer drive register for stand-by mode							

(Purpose and use)

- This register is used to set each pin status at stand-by mode.
- All ports have this registers of the format shown above. (“x” indicates the 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 the CPU has executed the “Halt” instruction.
- This is the case regardless of stand-by mode (IDLE2, IDLE1 or STOP).
- The Output/Input buffer control table is shown below.

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

Note 1: OE denotes an output enable signal before stand-by mode.
Basically, PxCR is used as OE.

Note 2: “n” in PxnD denotes the 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 3.3.3 shows the register setting operation during IDLE2 mode.

Table 3.3.3 SFR Setting Operation during IDLE2 Mode

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

2. IDLE1: Only the oscillator, RTC (real-time clock) 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 3.3.4.

Table 3.3.4 I/O Operation during HALT Modes

HALT Mode		IDLE2	IDLE1	STOP
SYSCR2<HALTM1:0>		11	10	01
Block	CPU	Stop	Stop	Stop
	I/O ports	Depend on PxDR register setting	Depend on PxDR register setting	Depend on PxDR register setting
	TMRA, TMRB	Available to select operation block	Stop	Stop
	SIO			
	AD converter			
	WDT			
	I2S, LCDC, SDRAMC, Interrupt controller, USBC,	Operate		
	RTC, MLD		Operate	

(2) How to release the HALT mode

These halt states can be released by resetting or requesting an interrupt. The halt release sources are determined by the combination of the states of the interrupt mask register <IFF2:0> and the HALT modes. The details for releasing the halt status are shown in Table 3.3.5.

Release by interrupt requesting

The HALT mode release method depends on the status of the enabled interrupt .When the interrupt request level set before executing the HALT instruction exceeds the value of the interrupt mask register, the interrupt is processed depending on its status after the HALT mode is released, and the CPU status executing the 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, HALT mode release is not executed. (in non-maskable interrupts, interrupt processing is processed after releasing the HALT mode regardless of the value of the mask register.) However only for INT0 to INT4, INTKEY, INTRTC, INTALM and INTUSB interrupts, even if the interrupt request level set before executing the halt instruction is less than the value of the interrupt mask register, HALT mode release is executed. In this case, the interrupt is processed, and the CPU starts executing the instruction following the HALT instruction, but the interrupt request flag is held at “1”.

Release by resetting

Release of all halt statuses is executed by resetting.

When the STOP mode is released by RESET, it is necessary to allow enough resetting time (see Table 3.3.6) for operation of the oscillator to stabilize.

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	INTWD	♦	×	×	–	–	–
		INT0 to INT4 (Note 1)	♦	♦	♦*1	○	○	○*1
		INTALM0 to INTALM4	♦	♦	×	○	○	×
		INTTA0 to INTTA3, INTTB0 to INTTB1	♦	×	×	×	×	×
		INTRX0 to INTRX1, TX0 to TX1	♦	×	×	×	×	×
		INTTBO0, INTI2S	♦	×	×	×	×	×
		INTAD, INT5	♦	×	×	×	×	×
		INTKEY	♦	♦	♦*1	○	○	○*1
		INTRTC	♦	♦	♦*1	○	○	○*1
		INTUSB	♦	♦*2	×	○	○*2	×
		INTLCD	♦	×	×	×	×	×
	RESET		Initialize LSI					

♦: After clearing the HALT mode, CPU starts interrupt processing.

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

×: Cannot 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. This combination is not available.

*1: Release of the HALT mode is executed after warm-up time has elapsed.

*2: 6 interrupts of all 24 INTUSB sources can release Halt state from IDLE1 mode, allowing for the construction of low power dissipation systems. However, the method of use is limited as below.

Shift to IDLE1 mode :

Execute Halt instruction when the flag of INT_SUS or INT_CLKSTOP is "1" (SUSPEND state)

Release from IDLE1 mode :

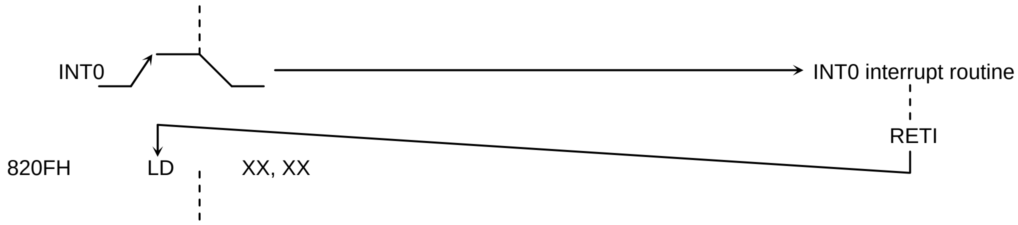
Release Halt state by INT_RESUME or INT_CLKON request (release SUSPEND request)

Release Halt state by INT_URST_STR or INT_URST_END request (RESET request)

Example: Releasing IDLE1 mode

An INT0 interrupt clears the halt state when the device is in IDLE1 mode.

```
Address
8200H    LD      (PCFC), 01H         ;   Sets PC0 to INT0.
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.
```



(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 3.3.7 illustrates an example of the timing for clearance of the IDLE2 mode halt state by an interrupt.

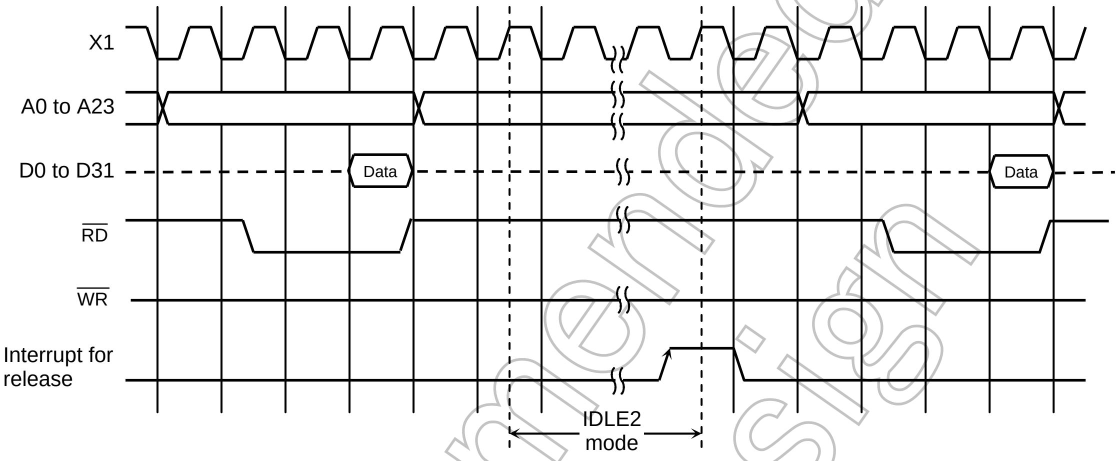


Figure 3.3.7 Timing Chart for IDLE2 Mode Halt State Cleared by Interrupt

2. IDLE1 mode

In IDLE1 mode, only the internal oscillator and the RTC 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 (e.g., restart of operation) is synchronous with it.

Figure 3.3.8 illustrates the timing for clearance of the IDLE1 mode halt state by an interrupt.

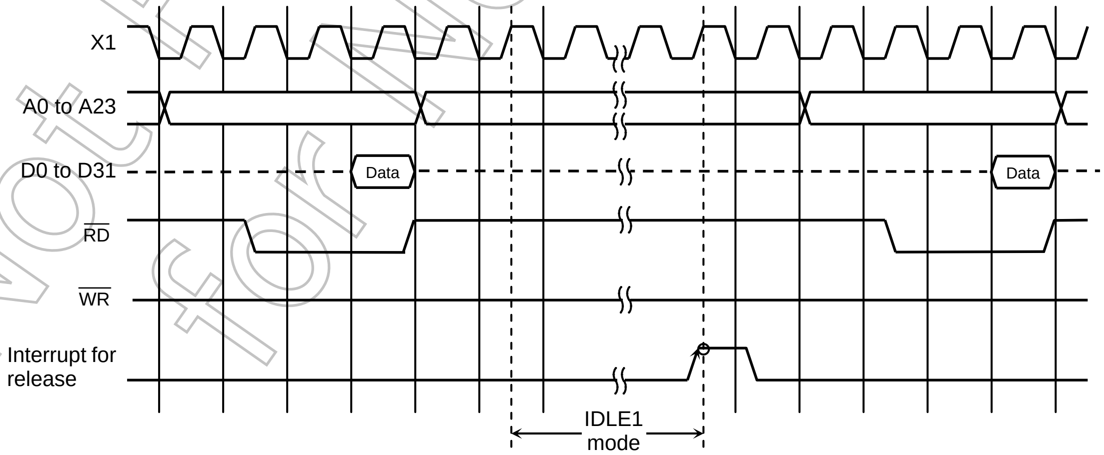


Figure 3.3.8 Timing Chart for IDLE1 Mode Halt State Cleared by Interrupt

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 3.3.9 illustrates the timing for clearance of the STOP mode halt state by an interrupt.

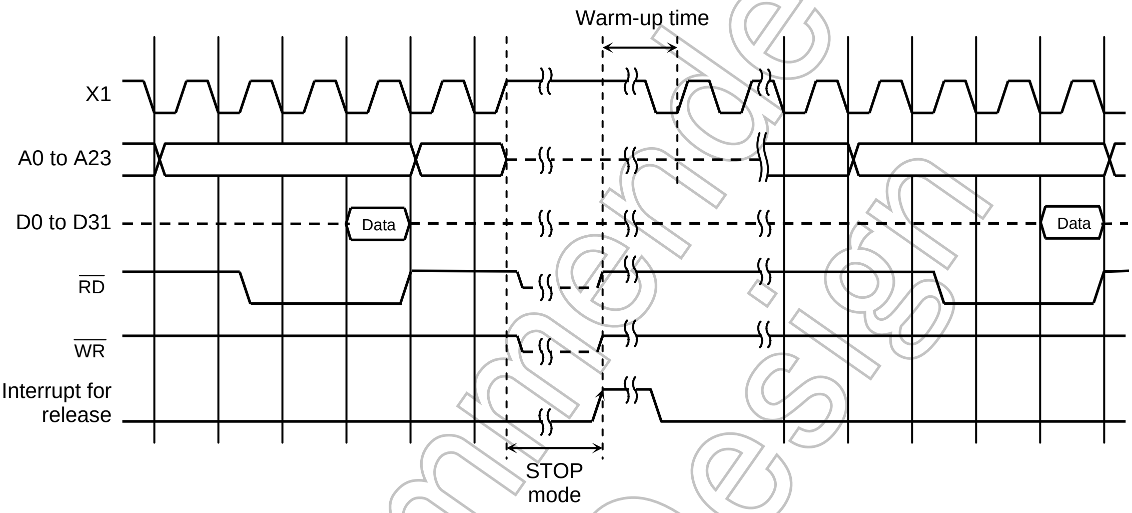


Figure 3.3.9 Timing Chart for STOP Mode Halt State Cleared by Interrupt

Table 3.3.6 Example of Warm-up Time after Releasing STOP Mode

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

SYSCR1 <SYSCK>	SYSCR2<WUPTM1:0>		
	01 (2^8)	10 (2^{14})	11 (2^{16})
0 (fc)	6.4 μs	409.6 μs	1.638 ms
1 (fs)	7.8 ms	500 ms	2000 ms

Table 3.3.7 Input Buffer State Table

Port Name	Input Function Name	Input Buffer State						
		During Reset	When the CPU is operating		In HALT mode (IDLE1/2/STOP)			
					<PxDR> = 1		<PxDR> = 0	
			When used as Function pin	When used as Input pin	When used as Function pin	When used as Input pin	When used as Function pin	When used as Input pin
D0 to D7	D0 to D7	OFF	ON upon external read	–	OFF	–	OFF	–
P10 to P17	D8 to D15	16bit start : OFF 32bit start : OFF Boot start : ON		ON	OFF	ON	OFF	OFF
P20 to P27	D16 to D23	16bit start : ON 32bit start : OFF Boot start : ON						
P30 to P37	D24 to D31							
P60 to P67	–	16bit start : OFF 32bit start : OFF Boot start : ON	–		–		–	
P71 to P72	–	ON						
P75	$\overline{\text{NDRB}}$		ON		ON		OFF	
P76	$\overline{\text{WAIT}}$							
P90	–		–		–		–	
P91	RXD0		ON		ON		OFF	
P92	CTS0, SCLK0							
P93 to P94	–		–		–		–	
P96 [*1]	INT4		ON		ON		OFF	
P97	INT5							
PA0 to PA7[*1]	KI0-KI7							
PC0	INT0							
PC1	INT1							
PC2	INT2							
PC3	INT3							
PC6 to PC7	–		–		–		–	
PF0	–							
PF1	RXD0/1		ON		ON		OFF	
PF2	CTS0/1 SCLK0/1							
PG0 to PG2[*2]	–	OFF	–	ON upon port read	–	OFF	–	
PG3 [*2]	ADTRG		ON		ON		ON	
PJ5 to PJ6	–	ON	–	ON	–	ON	–	
PL4 to PL7	–							

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

OFF: The buffer is always turned off.

–: Not applicable

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

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

Table 3.3.8 Output Buffer State Table (1/2)

Port Name	Output Function Name	Output Buffer State						
		During Reset	When the CPU is operating		In HALT mode (IDLE1/2/STOP)			
					<PxDR>=1		<PxDR>=0	
			When used as Function pin	When used as Output pin	When used as Function pin	When used as Output pin	When used as Function pin	When used as Output pin
D0~D7	D0~D7	OFF	ON upon external write	–	OFF	–	OFF	–
P10~P17	D8~D15			ON		ON		OFF
P20~P27	D16~D23, KO0~KO7							
P30~P37	D24~D31							
P40~P47	A0-A7	ON	ON		ON			
P50~P57	A8~A15							
P60~P67	A16~A23							
P70	$\overline{\text{RD}}$							
P71	$\overline{\text{WRLL}}$, $\overline{\text{NDRE}}$	OFF						
P72	$\overline{\text{WRLU}}$, $\overline{\text{NDWE}}$							
P73	EA24	ON						
P74	EA25							
P75	R/W	OFF						
P76	–		–		–		–	
P80	$\overline{\text{CS0}}$	ON	ON		ON		OFF	
P81	$\overline{\text{CS1}}$, $\overline{\text{SDCS}}$							
P82	$\overline{\text{CS2}}$, $\overline{\text{CSZA}}$, $\overline{\text{SDCS}}$							
P83	$\overline{\text{CS3}}$							
P84	$\overline{\text{CSZB}}$, $\overline{\text{WRUL}}$, $\overline{\text{ND0CE}}$							
P85	$\overline{\text{CSZC}}$, $\overline{\text{WRUU}}$, $\overline{\text{ND1CE}}$							
P86	$\overline{\text{CSZD}}$, $\overline{\text{SRULB}}$							
P87	$\overline{\text{CSZE}}$, $\overline{\text{SRUUB}}$							
P90	TXD0, I2SCKO	OFF						
P91	I2SDO							
P92	SCLK0, I2SWS							
P93	LGOE0							
P94	LGOE1							
P95	LGOE2, CLK32KO	ON						
P96[*1]	PX	OFF		–		–		–
P97	PY							

ON: The buffer is always turned on.

OFF: The buffer is always turned off.

–: Not applicable

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

Table 3.3.9 Output Buffer State Table (2/2)

Port Name	Output Function Name	Output Buffer State						
		During Reset	When the CPU is operating		In HALT mode (IDLE1/2/STOP)			
					<PxDR>=1		<PxDR>=0	
			When used as Function pin	When used as Output pin	When used as Function pin	When used as Output pin	When used as Function pin	When used as Output pin
PA3~PA6[*1]	LD8~LD11	OFF	ON	–	ON	–	OFF	–
PC0	TA1OUT			ON		ON		OFF
PC1	TA3OUT							
PC2	TB0OUT0							
PC3	–		–		–		–	
PC6	KO8, LDIV		ON		ON		OFF	
PC7	$\overline{\text{CSZF}}$, LCP1							
PF0	TXD0, TXD1							
PF1	–		–		–		–	
PF2	SCLK0, SCLK1		ON		ON		OFF	
PF7	SDCLK	ON						
PG2	MX	OFF		–		–		–
PG3	MY							
PJ0	$\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$	ON		ON		ON		OFF
PJ1	$\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$							
PJ2	$\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$							
PJ3	SDLLDQM							
PJ4	SDLUDQM							
PJ5	SDULDQM, NDALE	OFF						
PJ6	SDUUDQM, NDCLE							
PJ7	SDCKE	ON						
PK0	LCP0							
PK1	LLP							
PK2	LFR							
PK3	LBCD							
PL0~PL3	LD0~LD3							
PL4~PL7	LD4~LD7	OFF						
PM1	MLDALM	ON						
PM2	$\overline{\text{MLDALM}}$, $\overline{\text{ALARM}}$							
X2	–			–		–	IDLE2/1:ON, STOP: output ”H”	
XT2	–						IDLE2/1:ON, STOP: output ”HZ”	

ON: The buffer is always turned on.

OFF: The buffer is always turned off.

–: Not applicable

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

3.4 Interrupts

Interrupts are controlled by the CPU Interrupt mask register <IFF2:0> (bits12 to 14 of the status register) and by the built-in interrupt controller.

The TMP92CH21 has a total of 50 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 six 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 is generated simultaneously, the interrupt controller sends the priority value of the interrupt with the highest priority to the CPU. (The highest priority level is 7, the level used for non-maskable interrupts.)

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

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

The value in the interrupt mask register <IFF2:0> can be changed using the EI instruction (EI num sets <IFF2:0> to num). For example, the command 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 <IFF2:0> 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 1 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.

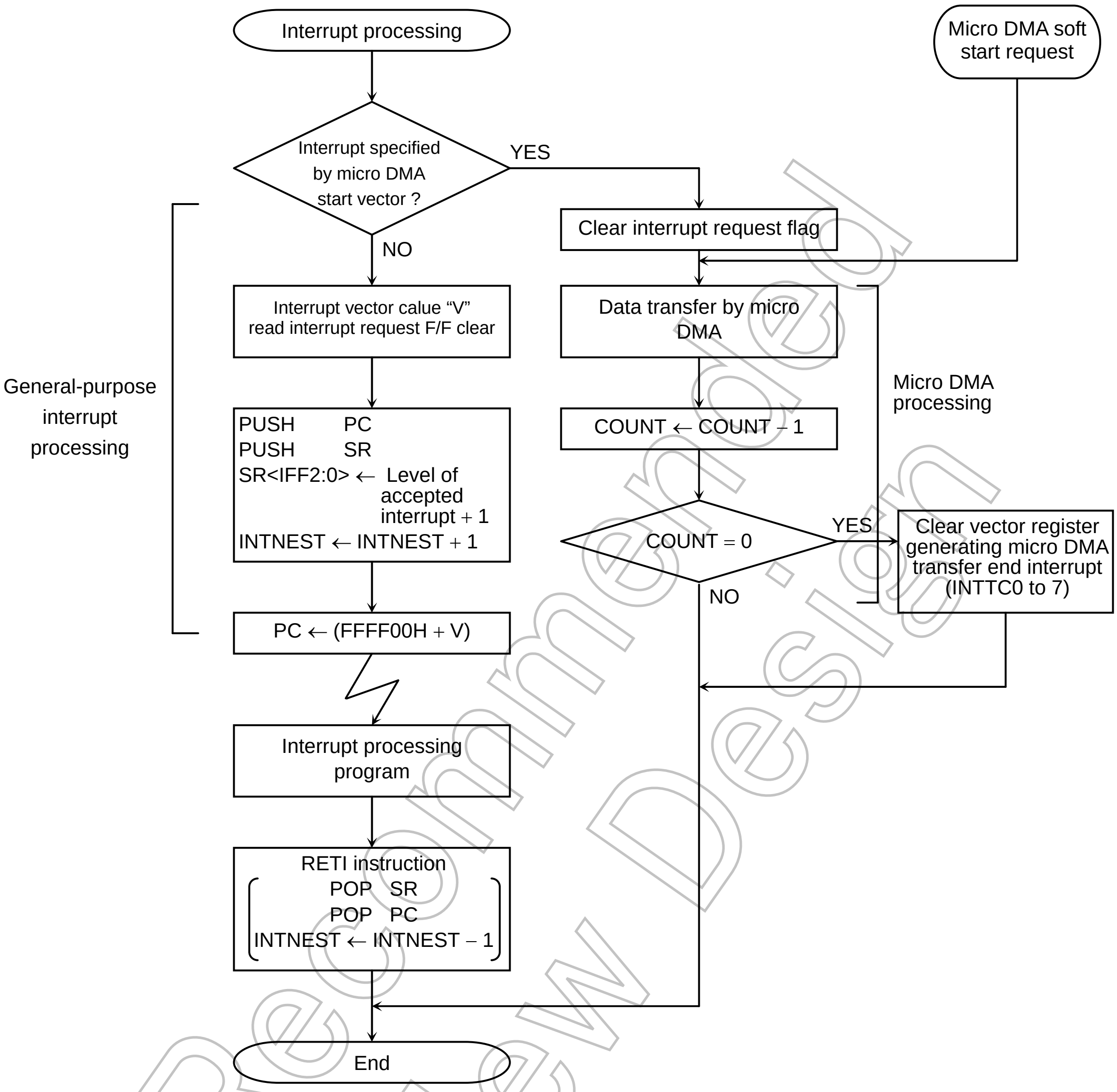

Interrupt processing
Interrupt specified by micro DMA start vector ?
YES
NO
Micro DMA soft start request
Clear interrupt request flag
Data transfer by micro DMA
Interrupt vector calue "V" read interrupt request F/F clear
General-purpose interrupt processing
Micro DMA processing
PUSH PC
PUSH SR
SR<IFF2:0> ← Level of accepted interrupt + 1
INTNEST ← INTNEST + 1
COUNT ← COUNT − 1
COUNT = 0
YES
NO
Clear vector register generating micro DMA transfer end interrupt (INTTC0 to 7)
PC ← (FFFF00H + V)
Interrupt processing program
RETI instruction
POP SR
POP PC
INTNEST ← INTNEST − 1
End

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 has been generated simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt requests.

(The default priority is determined as follows: the smaller the vector value, the higher the priority.)

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

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

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

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

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

Non-maskable interrupts cannot be disabled by a user program. Maskable interrupts, however, can be enabled or disabled by a user program. A program can set the priority level for each interrupt source. (A priority level setting of 0 or 7 will disable an interrupt request.)

If an interrupt request is received for an interrupt with a priority level equal to or greater than the value set in the CPU interrupt mask register <IFF2:0>, the CPU will accept the interrupt. The CPU interrupt mask register <IFF2:0> is then set to the value of the priority level for the accepted interrupt plus 1.

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

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

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

Table 3.4.1 shows the TMP92CH21 interrupt vectors and micro DMA start vectors. FFFF00H to 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	–	–	– (Note1)
11		INT0: INT0 pin input	0028H	FFFF28H	0AH (Note 2)
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 (8192Hz)	003CH	FFFF3CH	0FH
17		INTALM1: ALM1 (512 Hz)	0040H	FFFF40H	10H
18		INTALM2: ALM2 (64 Hz)	0044H	FFFF44H	11H
19		INTALM3: ALM3 (2 Hz)	0048H	FFFF48H	12H
20		INTALM4: ALM4 (1 Hz)	004CH	FFFF4CH	13H
21		INTP0: Protect0 (Write 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-on wakeup	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 (Note 2)
34		INTTX0: Serial transmission (Channel 0)	0084H	FFFF84H	21H
35		INTRX1: Serial receive (Channel 1)	0088H	FFFF88H	22H (Note 2)
36		INTTX1: Serial transmission (Channel 1)	008CH	FFFF8CH	23H
37		(Reserved)	0090H	FFFF90H	24H
38		(Reserved)	0094H	FFFF94H	25H
39		INT5: INT5 pin input	0098H	FFFF98H	26H
40		INTI2S: I^{2}S (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

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value	Address Refer to Vector	Micro DMA Start Vector
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 1: Micro DMA default priority.
Micro DMA initiation takes priority over other maskable interrupts.

Note 2: When initiating micro DMA, set at edge detect mode.

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 is implemented through the CPU, when the CPU is placed in a stand-by state by a Halt instruction, the requirements of the micro DMA will be ignored (pending).

Micro DMA supports 8 channels and can be transferred continuously by specifying the micro DMA burst function as below.

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

(1) Micro DMA operation

When an interrupt request is generated by an interrupt source 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, 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. (Note) In this case, edge triggered interrupts are the only kinds of general interrupts which can be accepted.

Note: If the priority level of micro DMA is set higher than that of other interrupts, CPU operates as follows.
In case INTxxx interrupt is generated first and then INTyyy interrupt is generated between checking “Interrupt specified by micro DMA start vector” (in the Figure 3.4.1) and reading interrupt vector with setting below. The vector shifts to that of INTyyy at the time.
This is because the priority level of INTyyy is higher than that of INTxxx.
In the interrupt routine, CPU reads the vector of INTyyy because cheking of micro DMA has finished.
And INTyyy is generated regardless of transfer counter of micro DMA.
INTxxx: level 1 without micro DMA
INTyyy: level 6 with micro DMA

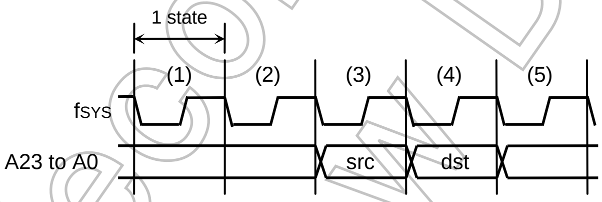
Although the control registers used for setting the transfer source and transfer destination addresses are 32 bits wide, this type of register can only output 24-bit addresses. Accordingly, micro DMA can only access 16 Mbytes (the upper eight bits of a 32-bit address are not valid).

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 memory to memory, from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the various transfer modes, see section 3.4.2 (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: Both source and destination memory are internal RAM and multiples by 4 numbered source and destination addresses.)



Note: In fact, src and dst address are not output to A23 to A0 pins because they are internal RAM address.

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. (If write "0" to each bit, micro DMA doesn't operate). On completion of the transfer, the bits of DMAR which support the end channel are automatically cleared to 0.

Only one channel can be set once for DMA request. (Do not write "1" to plural bits.)

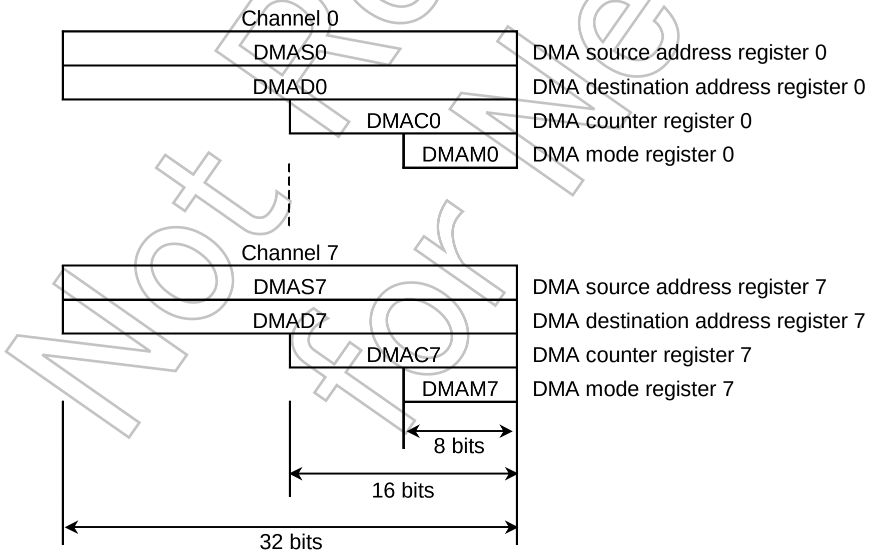
When writing again 1 to the DMAR register, check whether the bit is "0" before writing "1". If read "1", micro DMA transfer isn't started yet.

When a burst is specified by the DMAB register, data is transferred continuously from the initiation of micro DMA until the value in the micro DMA transfer counter is 0. If execatee soft start during micro DMA transfer by interrupt source, micro DMA transfer counter doesn't change. Don't use Read-modify-write instruction to avoid writign to other bits by mistake.

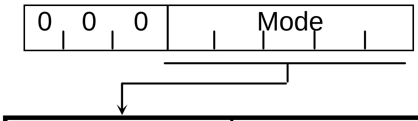
Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA Request	109H (Prohibit RMW)	DREQ7	DREQ6	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1	DREQ0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request in software							

(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



DMAM0 to DMAM7

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

1 1 1 0 0	Counter mode
	DMASn ← DMASn + 1
	DMACn ← DMACn – 1
	If DMACn = 0 then INTTCn
		5 states

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

Note1: 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.

Note2: The transfer mode register should not be set to any value other than those listed above.

Note3: The execution state number shows number of best case (1-state memory access).

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 interrupts 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 writing a micro DMA start vector to the INTCLR register).

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g., INTE0AD or INTE12). 6 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 bit of the interrupt priority setting register indicate the state of the interrupt request flag and thus whether an interrupt request for a given channel has occurred.

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

When interrupt processing has been completed (e.g., after execution of a RETI instruction), the CPU restores to SR<IFF2:0> the priority value which was saved on the stack before the interrupt was generated.

The interrupt controller also includes eight registers which are used to store the micro DMA start vector. Writing the start vector of the interrupt source for the micro DMA processing (see Table 3.4.1), enables the corresponding interrupts 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.

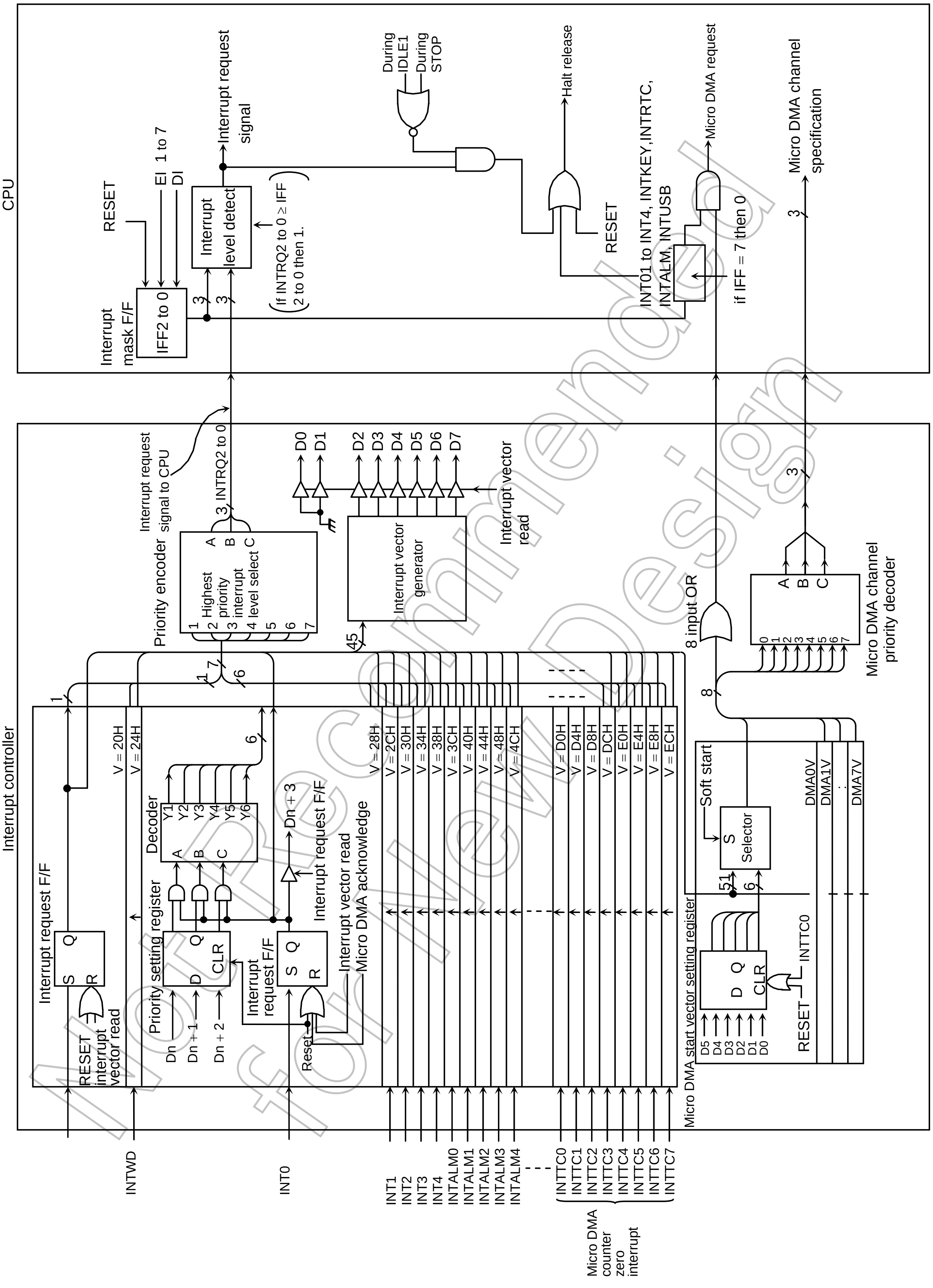
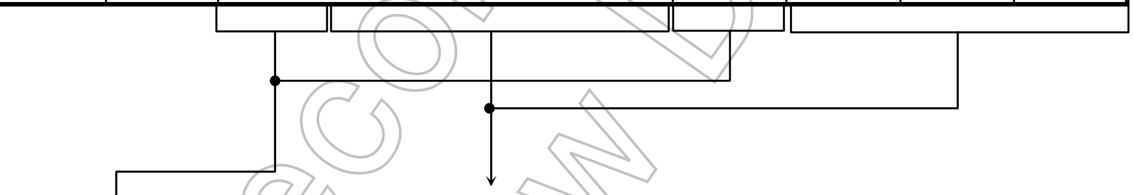


Figure 3.4.3 Block Diagram of Interrupt Controller

(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	0	0	0	0	0	0
INTE12	INT1 & INT2 enable	D0H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	D1H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE5I2S	INT5 & INTI2S enable	EBH	INTI2S				INT5			
			II2SC	II2SM2	II2SM1	II2SM0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	D4H	INTTA1 (TMRA 1)				INTTA0 (TMRA 0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	D5H	INTTA3 (TMRA 3)				INTTA2 (TMRA 2)			
			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	D8H	INTTB1 (TMRA 4)				INTTB0 (TMRA 4)			
			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	DAH	–				INTTBO0			
			–	–	–	–	ITBO0C	ITBO0M2	ITBO0M1	ITBO0M0
							R	R/W		
			Note: Always write 0				0	0	0	0
INTES0	INTRX0 & INTTX0 enable	DBH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES1	INTRX1 & INTTX1 enable	DCH	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	E3H	–				INTUSB			
			–	–	–	–	IUSB0C	IUSBM2	IUSBM1	IUSBM0
							R	R/W		
			Note: Always write 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	0	0	0	0	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	0	0	0	0	0	0

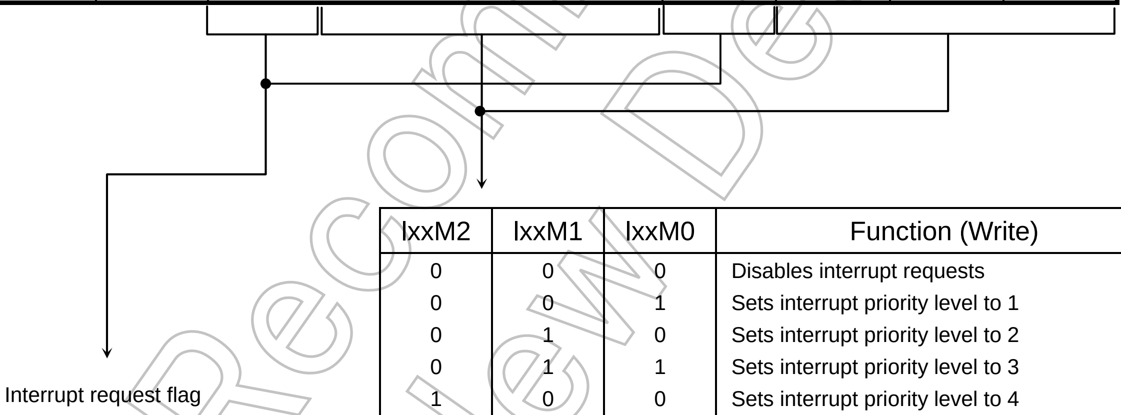
Symbol	Name	Address	7	6	5	4	3	2	1	0
INTEALM4	INTALM4 enable	E7H	–				INTALM4			
			–	–	–	–	IA4C	IA4M2	IA4M1	IA4M0
							R	R/W		
			Note: Always write 0				0	0	0	0
INTERTC	INTRTC enable	E8H	–				INTRTC			
			–	–	–	–	IRC	IRM2	IRM1	IRM0
							R	R/W		
			Note: Always write 0				0	0	0	0
INTEKEY	INTKEY enable	E9H	–				INTKEY			
			–	–	–	–	IKC	IKM2	IKM1	IKM0
							R	R/W		
			Note: Always write 0				0	0	0	0
INTELCD	INTLCD enable	EAH	–				INTLCD			
			–	–	–	–	ILCD1C	ILCDM2	ILCDM1	ILCDM0
							R	R/W		
			Note: Always write 0				0	0	0	0
INTEND01	INTNDF0 & INTNDF1 enable	ECH	INTNDF1				INTNDF0			
			IN1C	IN1M2	IN1M1	IN1M0	IN0C	IN0M2	IN0M1	IN0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 enable	EEH	–				INTP0			
			–	–	–	–	IP0C	IP0M2	IP0M1	IP0M0
							R	R/W		
			Note: Always write 0				0	0	0	0



Interrupt request flag

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

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETC01	INTTC0 & INTTC1 enable	F1H	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	F2H	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	F3H	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	F4H	INTTC7 (DMA7)				INTTC6 (DMA6)			
			ITC7C	ITC7M2	ITC7M1	ITC7M0	ITC6C	ITC6M2	ITC6M1	ITC6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTWDT	INTWD enable	F7H	–				INTWD			
			–	–	–	–	ITCWD	–	–	–
							R			
			Note: Always write 0				0	–	–	–



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

(2) External interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt input mode control	F6H (Prohibit RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	–
			W	W	W	W	W	W	R/W	R/W
			0	0	0	0	0	0	0	0
			INT5EDGE 0: Rising 1: Falling	INT4EDGE 0: Rising 1: Falling	INT3EDGE 0: Rising 1: Falling	INT2EDGE 0: Rising 1: Falling	INT1EDGE 0: Rising 1: Falling	INT0EDGE 0: Rising 1: Falling	0: INT0 edge mode 1: INT0 level mode	Always write "0"

*INT0 level enable

0	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), XXXXXX00B ; Switches from level to edge.
LD      (INTCLR), 0AH     ; Clears interrupt request flag.
NOP                       ; Wait EI execution
NOP
NOP
EI
```

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 (Prohibit RMW)	–						IR1LE	IR0LE
			W						W	W
			0						1	1
			Always write “0” (Note)						0: INTRX1 edge mode 1: INTRX1 level mode	0: INTRX0 edge mode 1: INTRX0 level mode

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

INTRX1 level enable

0	Edge detect INTRX1
1	“H” level INTRX1

INTRX0 rising edge enable

0	Edge detect INTRX0
1	“H” level INTRX0

(4) Interrupt request flag clear register

The interrupt request flag is cleared by writing the appropriate micro DMA 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 (Prohibit RMW)	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt vector							

(5) Micro DMA start vector registers

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

(6) 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
			1: DMA burst request							

(7) Notes

The instruction execution unit and the bus interface unit in this CPU operate independently. Therefore, immediately before an interrupt is generated, if 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 placed after a DI instruction. And in the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing and more than 3–instructions (e.g., “NOP” × 3 times).

If it placed EI instruction without waiting NOP instruction after execution of clearing instruction, interrupt will be enabled before request flag is cleared.

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

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

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

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.

INTRX: Instructions which read the receive buffer.

3.5 Function of Ports

The TMP92CH21 I/O port pins are shown in Table 3.5.1 and Table 3.5.2. In addition to functioning as general-purpose I/O ports, these pins are also used by the internal CPU and I/O functions. Table 3.5.3 to Table 3.5.5 list the I/O registers and their specifications.

Table 3.5.1 Port Functions (1/2)

(R: PD = with programmable pull-down resistor, U = with pull-up resistor)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for Built-in Function
Port 1	P10 to P17	8	I/O	–	Bit	D8 to D15
Port 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	Output	–	(Fixed)	A0 to A7
Port 5	P50 to P57	8	Output	–	(Fixed)	A8 to A15
Port 6	P60 to P67	8	I/O	–	Bit	A16 to A23
Port 7	P70	1	Output	–	(Fixed)	$\overline{\text{RD}}$
	P71	1	I/O	–	Bit	$\overline{\text{WRLL}}$, $\overline{\text{NDRE}}$
	P72	1	I/O	–	Bit	$\overline{\text{WRLU}}$, $\overline{\text{NDWE}}$
	P73	1	Output	–	(Fixed)	EA24
	P74	1	Output	–	(Fixed)	EA25
	P75	1	I/O	–	Bit	R/ $\overline{\text{W}}$, NDR/ $\overline{\text{B}}$
	P76	1	I/O	–	Bit	$\overline{\text{WAIT}}$
Port 8	P80	1	Output	–	(Fixed)	$\overline{\text{CS0}}$
	P81	1	Output	–	(Fixed)	$\overline{\text{CS1}}$, $\overline{\text{SDCS}}$
	P82	1	Output	–	(Fixed)	$\overline{\text{CS2}}$, $\overline{\text{CSZA}}$, $\overline{\text{SDCS}}$
	P83	1	Output	–	(Fixed)	$\overline{\text{CS3}}$
	P84	1	Output	–	(Fixed)	$\overline{\text{CSZB}}$, $\overline{\text{WRUL}}$, $\overline{\text{ND0CE}}$
	P85	1	Output	–	(Fixed)	$\overline{\text{CSZC}}$, $\overline{\text{WRUU}}$, $\overline{\text{ND1CE}}$
	P86	1	Output	–	(Fixed)	$\overline{\text{CSZD}}$, $\overline{\text{SRULB}}$
	P87	1	Output	–	(Fixed)	$\overline{\text{CSZE}}$, $\overline{\text{SRUUB}}$
Port 9	P90	1	I/O	–	Bit	TXD0, I2SCKO
	P91	1	I/O	–	Bit	RXD0, I2SDO
	P92	1	I/O	–	Bit	SCLK0, $\overline{\text{CTS0}}$, I2SWS
	P93	1	I/O	–	Bit	LGOE0
	P94	1	I/O	–	Bit	LGOE1
	P95	1	Output	–	(Fixed)	LGOE2, CLK32KO
	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	$\overline{\text{CSZF}}$, LCP1
Port F	PF0	1	I/O	–	Bit	TXD0, TXD1
	PF1	1	I/O	–	Bit	RXD0, RXD1
	PF2	1	I/O	–	Bit	SCLK0, $\overline{\text{CTS0}}$, SCLK1, $\overline{\text{CTS1}}$
	PF7	1	Output	–	(Fixed)	SDCLK

Table 3.5.2 Port Functions (2/2)

(R: PD = with programmable pull-down resistor, U = with pull-up resistor)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for Built-in Function
Port G	PG0 to PG1	2	Input	–	(Fixed)	AN0 to AN1
	PG2	1	Input	–	(Fixed)	AN2, MX
	PG3	1	Input	–	(Fixed)	AN3, $\overline{\text{ADTRG}}$, MY
Port J	PJ0	1	Output	–	(Fixed)	$\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$
	PJ1	1	Output	–	(Fixed)	$\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$
	PJ2	1	Output	–	(Fixed)	$\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$
	PJ3	1	Output	–	(Fixed)	SDLLDQM
	PJ4	1	Output	–	(Fixed)	SDLUDQM
	PJ5	1	I/O	–	Bit	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)	$\overline{\text{ALARM}}$, $\overline{\text{MLDALM}}$

Table 3.5.3 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	0	
		D8 to D15 bus	X	X	1	
Port 2	P20 to P27	Input port	X	0	0	0
		Output port	X	1	0	0
		D16 to D23 bus	X	X	1	0
		KO0 to KO7	X	1	0	1
Port 3	P30 to P37	Input port	X	0	0	None
		Output port	X	1	0	
		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	None
		A8 to A15 output	X		1	
Port 6	P60 to P67	Input port	X	0	0	None
		Output port	X	1	0	
		A16 to A23 output	X	X	1	
Port 7	P71 to P72 P75 to P76	Input port	X	0	0	None
	P70 to P76	Output port	X	1	0	
	P70	$\overline{\text{RD}}$ output	X	None	1	
	P71	$\overline{\text{WRLL}}$ output	1	1	1	
		$\overline{\text{NDRE}}$ output	0	1	1	
	P72	$\overline{\text{WRLU}}$ output	1	1	1	
		$\overline{\text{NDWE}}$ output	0	1	1	
	P73	EA24 output	X	None	1	
	P74	EA25 output	X		1	
	P75	R/$\overline{\text{W}}$ output	X	1	1	
		NDR/$\overline{\text{B}}$ input	X	0	1	
	P76	$\overline{\text{WAIT}}$ input	X	0	1	
Port 8	P80 to P87	Output Port	X	None	0	0
	P80	$\overline{\text{CS0}}$ output	X		1	0
	P81	$\overline{\text{CS1}}$ output	X		1	0
		$\overline{\text{SDCS}}$ output	X		X	1
	P82	$\overline{\text{CS2}}$ output	X		1	0
		$\overline{\text{CSZA}}$ Output	X		0	1
		$\overline{\text{SDCS}}$ output	X		1	1
	P83	$\overline{\text{CS3}}$ output	X		1	0
	P84	$\overline{\text{CSZB}}$ output	X		1	0
		$\overline{\text{WRUL}}$ output	X		0	1
		$\overline{\text{ND0CE}}$ output	X		1	1
	P85	$\overline{\text{CSZC}}$ output	X		1	0
		$\overline{\text{WRUU}}$ output	X		0	1
		$\overline{\text{ND1CE}}$ output	X		1	1
	P86	$\overline{\text{CSZD}}$ output	X		1	0
		$\overline{\text{SRULB}}$ output	X		X	1
	P87	$\overline{\text{CSZE}}$ output	X		1	0
		$\overline{\text{SRUUB}}$ output	X		X	1

Table 3.5.4 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 P94, P96 to P97	Input port	X	0	0	0
	P90 to P94	Output port	X	1	0	0
	P95		X	0	0	0
	P90	TXD0 output	X	1	1	0
		I2SCKO output	X	0	1	0
		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, $\overline{\text{CTS0}}$ input (Note1)	X	0	0	
	P93	LGOE0 output	X	0	1	
	P94	LGOE1 output	X	0	1	
	P95	LGOE2 output	X	0	1	
		CLK32KO output	X	1	0	
	P96	INT4 input	X	None	1	
	P97	INT5 input	X	None	1	
Port A	PA0 to PA7	Input port	X	0	0	None
		KI0 to KI7 input	X	0	1	
	PA3 to PA6	LD8 to LD11 output	X	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	
		$\overline{\text{CSZF}}$ 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	0
		TXD0/TXD1 output (Open drain)	X	1/0	1	1
	PF1	RXD0 input	X	0	0	None
		RXD1 input	X	0	0	
	PF2	SCLK0 output	X	1	1	
		SCLK1 output	X	0	1	
		SCLK0, $\overline{\text{CTS0}}$ input	X	0	0	
		SCLK1, $\overline{\text{CTS1}}$ input	X	0	0	
	PF7	SDCLK output	X	None	1	

Note: To use P92-pin as SCLK0 input or $\overline{\text{CTS0}}$ input, set "1" to PF<PF2>

Table 3.5.5 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	$\overline{\text{ADTRG}}$ input				
	PG2	MX output				
	PG3	MY output				
Port J	PJ0 to PJ7	Output port	X	1	0	None
	PJ5 to PJ6	Input port	X	0	0	
	PJ0	$\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$ output	X	None	1	
	PJ1	$\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$ output	X		1	
	PJ2	$\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$ output	X		1	
	PJ3	SDLLDQM output	X		1	
	PJ4	SDLUDQM output	1		1	
	PJ5	SDULDQM output	1	1	1	
		NDALE output	0	1	1	
	PJ6	SDUUDQM output	1	1	1	
		NDCLE output	0	1	1	
	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	$\overline{\text{MLDALM}}$ output	0		1	
		$\overline{\text{ALARM}}$ output	1		1	

3.5.1 Port 1 (P10 to P17)

Port 1 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

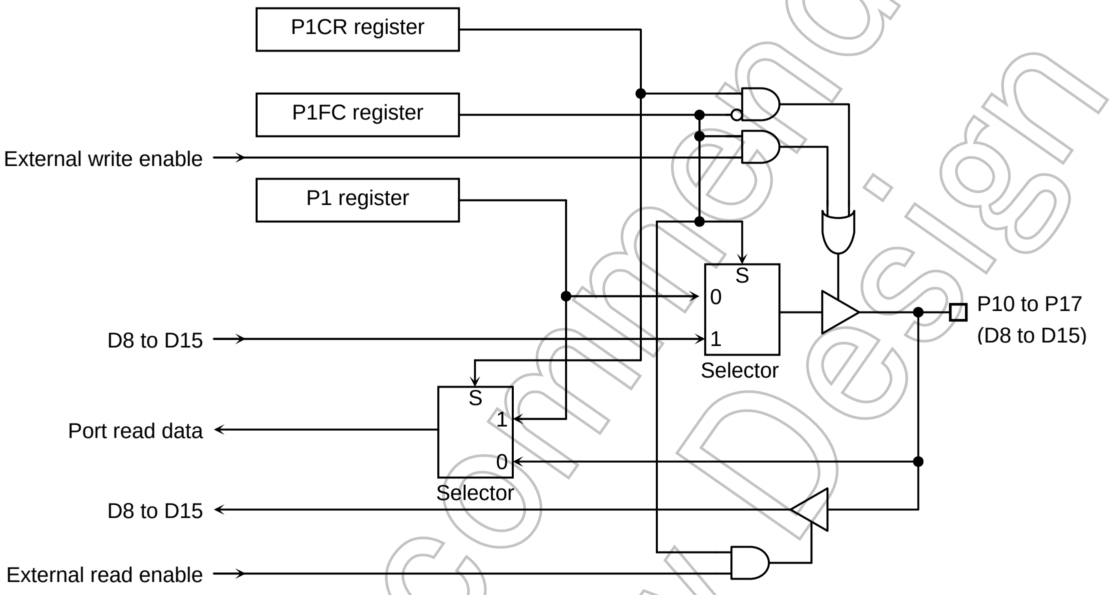


Figure 3.5.1 Port 1

Port 1 register

		7	6	5	4	3	2	1	0
P1	Bit symbol	P17	P16	P15	P14	P13	P12	P11	P10
(0004H)	Read/Write	R/W							
	Reset State	Data from external port (Output latch register is cleared to “0”)							

Port 1 Control register

		7	6	5	4	3	2	1	0
P1CR	Bit symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
(0006H)	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Input 1: Output							

Port 1 Function register

		7	6	5	4	3	2	1	0
P1FC	Bit symbol								P1F
(0007H)	Read/Write								W
	Reset State								0/1 Note 2
	Function								0: Port 1: Data bus (D8 to D15)

Port 1 Drive register

		7	6	5	4	3	2	1	0
P1DR	Bit symbol	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
(0081H)	Read/Write	W							
	Reset State	1	1	1	1	1	1	1	1
	Function	Input/Output buffer drive register for standby mode							

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

Note2: It is set to “Port” or “Data bus” by AM pin setting.

Figure 3.5.2 Register for Port 1

3.5.2 Port 2 (P20 to P27)

Port 2 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, port 2 can also function either as a data bus (D16 to D23) or keyboard interface pin KO0 to KO7 which can be 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

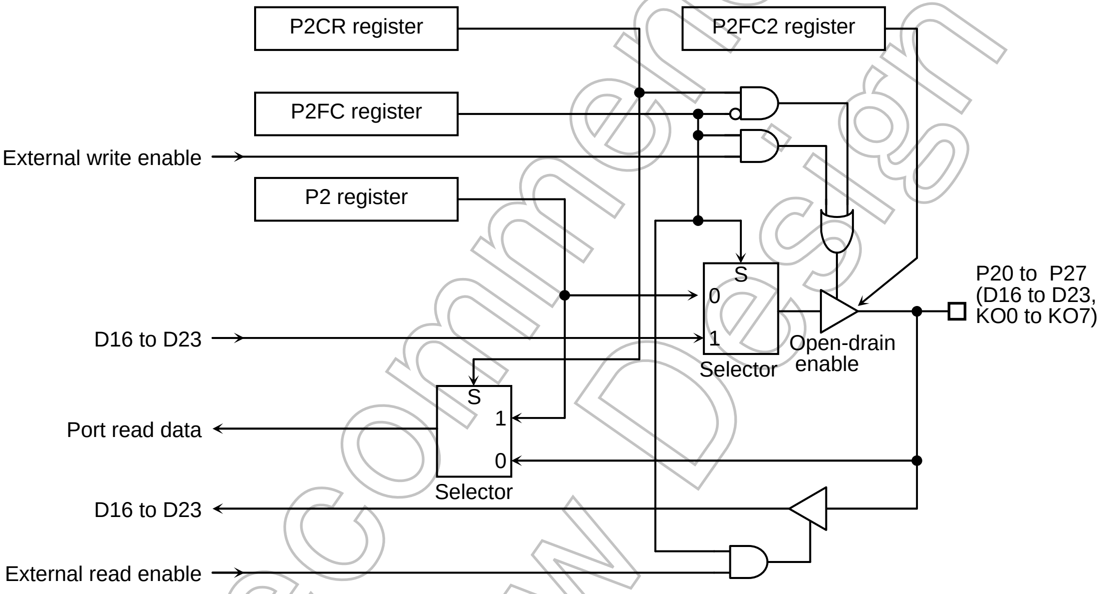


Figure 3.5.3 Port 2

Port 2 register

		7	6	5	4	3	2	1	0
P2	Bit symbol	P27	P26	P25	P24	P23	P22	P21	P20
(0008H)	Read/Write	R/W							
	Reset State	Data from external port (Output latch register is cleared to “0”)							

Port 2 Control register

		7	6	5	4	3	2	1	0
P2CR	Bit symbol	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
(000AH)	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Input 1: Output							

Port 2 Function register

		7	6	5	4	3	2	1	0
P2FC	Bit symbol								P2F
(000BH)	Read/Write								W
	Reset State Note 2								0/1
	Function								0: Port 1: Data bus (D16to D23)

Port 2 Function register 2

		7	6	5	4	3	2	1	0
P2FC2	Bit symbol	P27F2	P26F2	P25F2	P24F2	P23F2	P22F2	P21F2	P20F2
(0009H)	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: CMOS output 1: Open-drain output							

Port 2 Drive register

		7	6	5	4	3	2	1	0
P2DR	Bit symbol	P27D	P26D	P25D	P24D	P23D	P22D	P21D	P20D
(0082H)	Read/Write	W							
	Reset State	1	1	1	1	1	1	1	1
	Function	Input/Output buffer drive register for standby mode							

Note 1: Read-modify-write instruction is prohibited for P2CR, P2FC and P2FC2.

Note 2: It is set to “Port” or “Data bus” by AM pin setting.

Figure 3.5.4 Register for Port 2

3.5.3 Port 3 (P30 to P37)

Port 3 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, port 3 can also function 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

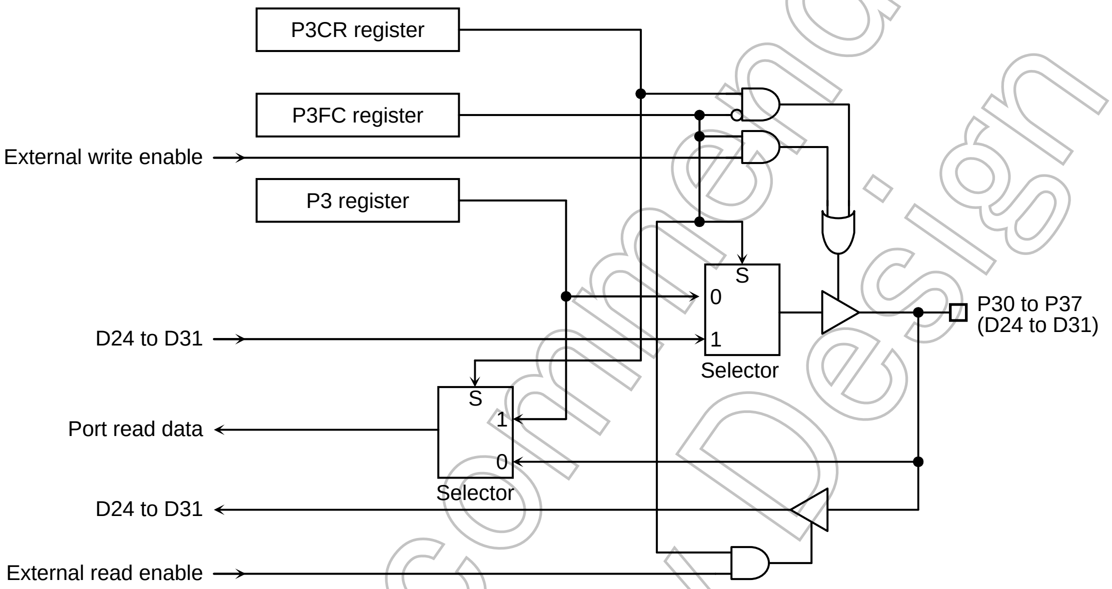


Figure 3.5.5 Port 3

Port 3 register

		7	6	5	4	3	2	1	0
P3 (000CH)	Bit symbol	P37	P36	P35	P34	P33	P32	P31	P30
	Read/Write	R/W							
	Reset State	Data from external port (Output latch register is cleared to "0")							

Port 3 Control register

		7	6	5	4	3	2	1	0
P3CR (000EH)	Bit symbol	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Input 1: Output							

Port 3 Function register

		7	6	5	4	3	2	1	0
P3FC (000FH)	Bit symbol					–	–	–	P3F
	Read/Write					W			W
	Reset State					0	0	0	0/1 Note 2
	Function					Always write "0"			0: Port 1: Data bus (D24 to D31)

Port 3 Drive register

		7	6	5	4	3	2	1	0
P3DR (0083H)	Bit symbol	P37D	P36D	P35D	P34D	P33D	P32D	P31D	P30D
	Read/Write	W							
	Reset State	1	1	1	1	1	1	1	1
	Function	Input/Output buffer drive register for standby mode							

Note 1: Read-modify-write instruction is prohibited for P3CR, P3FC and P3FC2.

Note 2: It is set to "Port" or "Data bus" by AM pin setting.

Figure 3.5.6 Register for Port 3

3.5.4 Port 4 (P40 to P47)

Port 4 is an 8-bit general-purpose output port.

In addition to functioning as a general-purpose output port, port 4 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

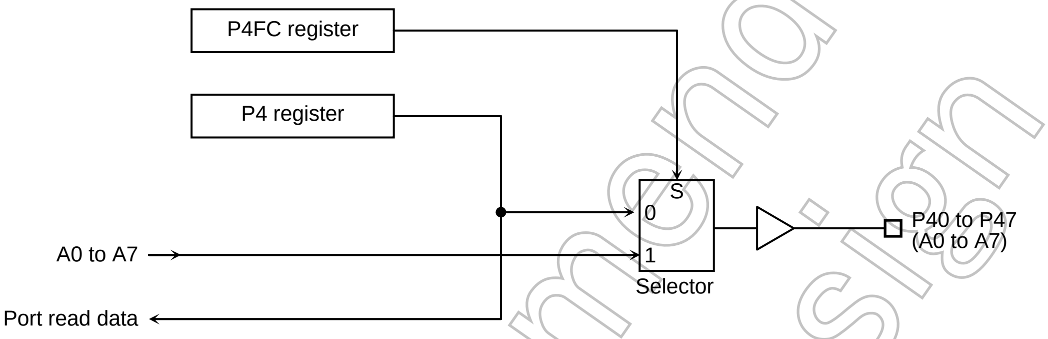


Figure 3.5.7 Port 4

Port 4 register

		7	6	5	4	3	2	1	0
P4 (0010H)	Bit symbol	P47	P46	P45	P44	P43	P42	P41	P40
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0

Port 4 Function register

		7	6	5	4	3	2	1	0
P4FC (0013H)	Bit symbol	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
	Read/Write	W							
	Reset State Note 2	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
	Function	0: Port 1: Address bus (A0 to A7)							

Port 4 Drive register

		7	6	5	4	3	2	1	0
P4DR (0084H)	Bit symbol	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
	Read/Write	W							
	Reset State	1	1	1	1	1	1	1	1
	Function	Input/Output buffer drive register for standby mode							

Note 1: Read-modify-write is prohibited for P4FC.

Note 2: It is set to “Port” or “Address bus” by AM pin setting.

Figure 3.5.8 Register for Port 4

3.5.5 Port 5 (P50 to P57)

Port 5 is an 8-bit general-purpose output port.

In addition to functioning as a general-purpose I/O port, port 5 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

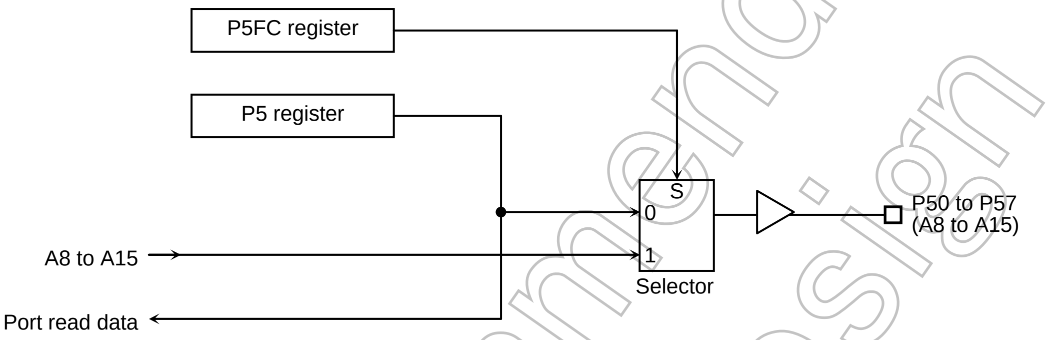


Figure 3.5.9 Port 5

Port 5 register

P5
(0014H)

	7	6	5	4	3	2	1	0
Bit symbol	P57	P56	P55	P54	P53	P52	P51	P50
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0

Port 5 Function register

P5FC
(0017H)

	7	6	5	4	3	2	1	0
Bit symbol	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
Read/Write	W							
Reset State Note 2	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Function	0: Port 1: Address bus (A8 to A15)							

Port 5 Drive register

P5DR
(0085H)

	7	6	5	4	3	2	1	0
Bit symbol	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
Read/Write	W							
Reset State	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note 1: Read-modify-write is prohibited for P5FC.

Note 2: It is set to "Port" or "Address bus" by AM pin setting.

Figure 3.5.10 Register for Port 5

3.5.6 Port 6 (P60 to P67)

Port 6 is an 8-bit general-purpose I/O port. 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, port 6 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

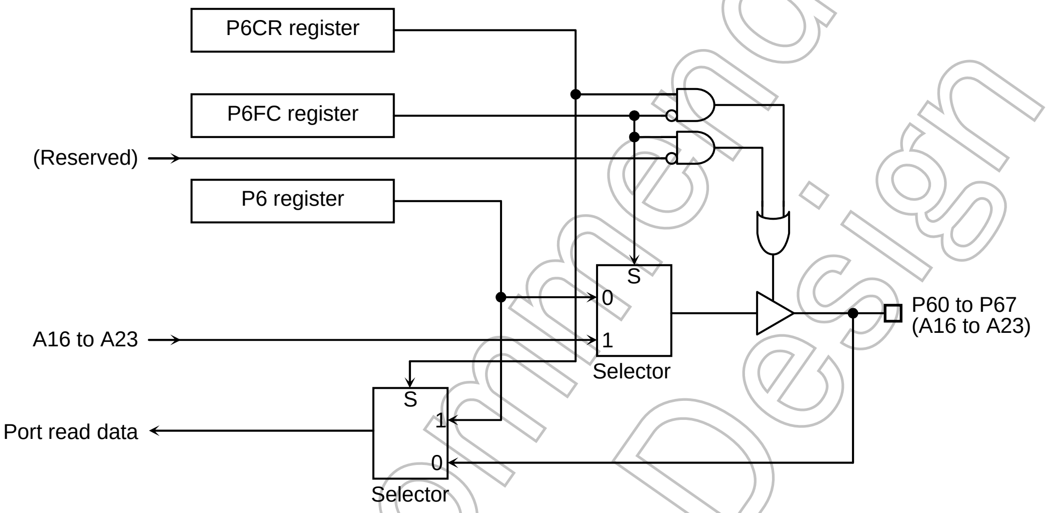


Figure 3.5.11 Port 6

Port 6 register

		7	6	5	4	3	2	1	0
P6 (0018H)	Bit symbol	P67	P66	P65	P64	P63	P62	P61	P60
	Read/Write	R/W							
	Reset State	Data from external port (Output latch register is cleared to “0”)							

Port 6 Control register

		7	6	5	4	3	2	1	0
P6CR (001AH)	Bit symbol	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Input 1: Output							

Port 6 Function register

		7	6	5	4	3	2	1	0
P6FC (001BH)	Bit symbol	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
	Read/Write	W							
	Reset State Note 2	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
	Function	0: Port 1: Address bus (A16 to A23)							

Port 6 Drive register

		7	6	5	4	3	2	1	0
P6DR (0086H)	Bit symbol	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
	Read/Write	W							
	Reset State	1	1	1	1	1	1	1	1
	Function	Input/Output buffer drive register for standby mode							

Note 1: Read-modify-write is prohibited for P6CR and P6FC.

Note 2: It is set to “Port” or “Address bus” by AM pin setting.

Figure 3.5.12 Register for Port 6

3.5.7 Port 7 (P70 to P76)

Port 7 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 as interface pins 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	$\overline{\text{RD}}$ pin
1	0	$\overline{\text{RD}}$ pin
1	1	P70 output port

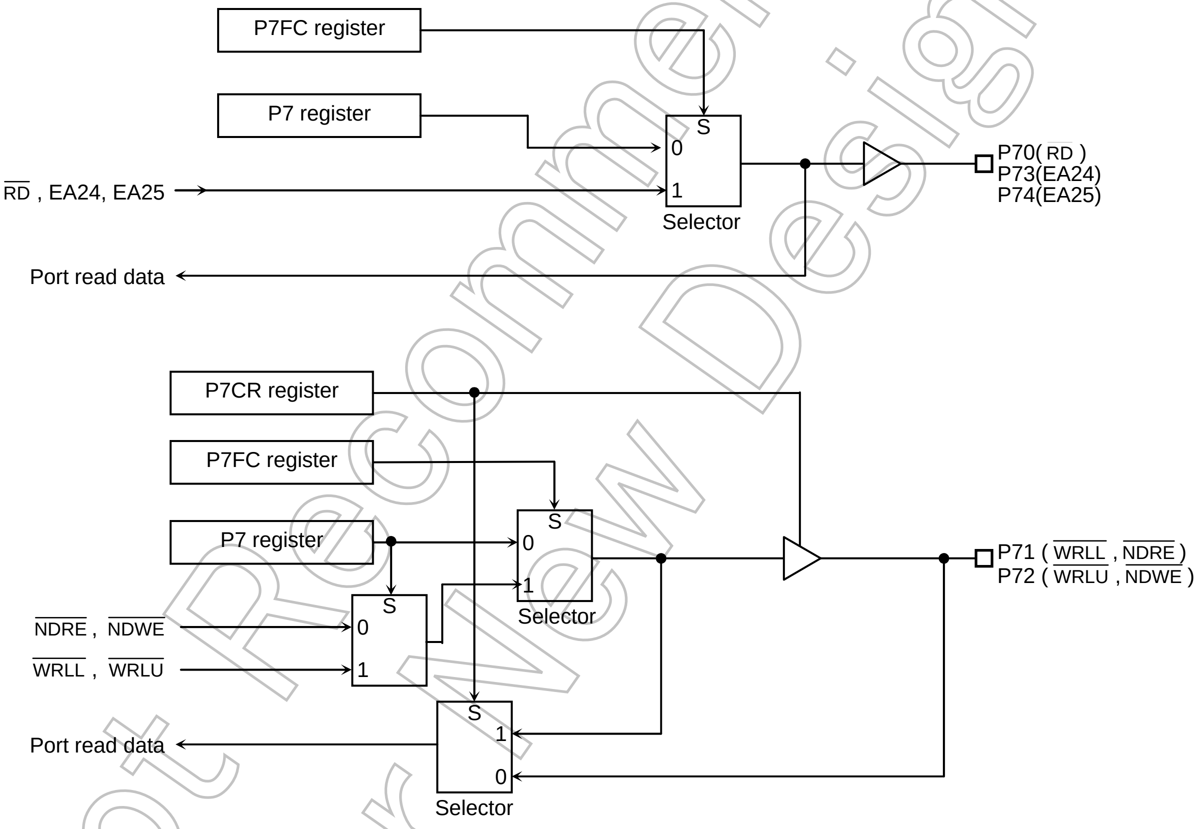


Figure 3.5.13 Port 7

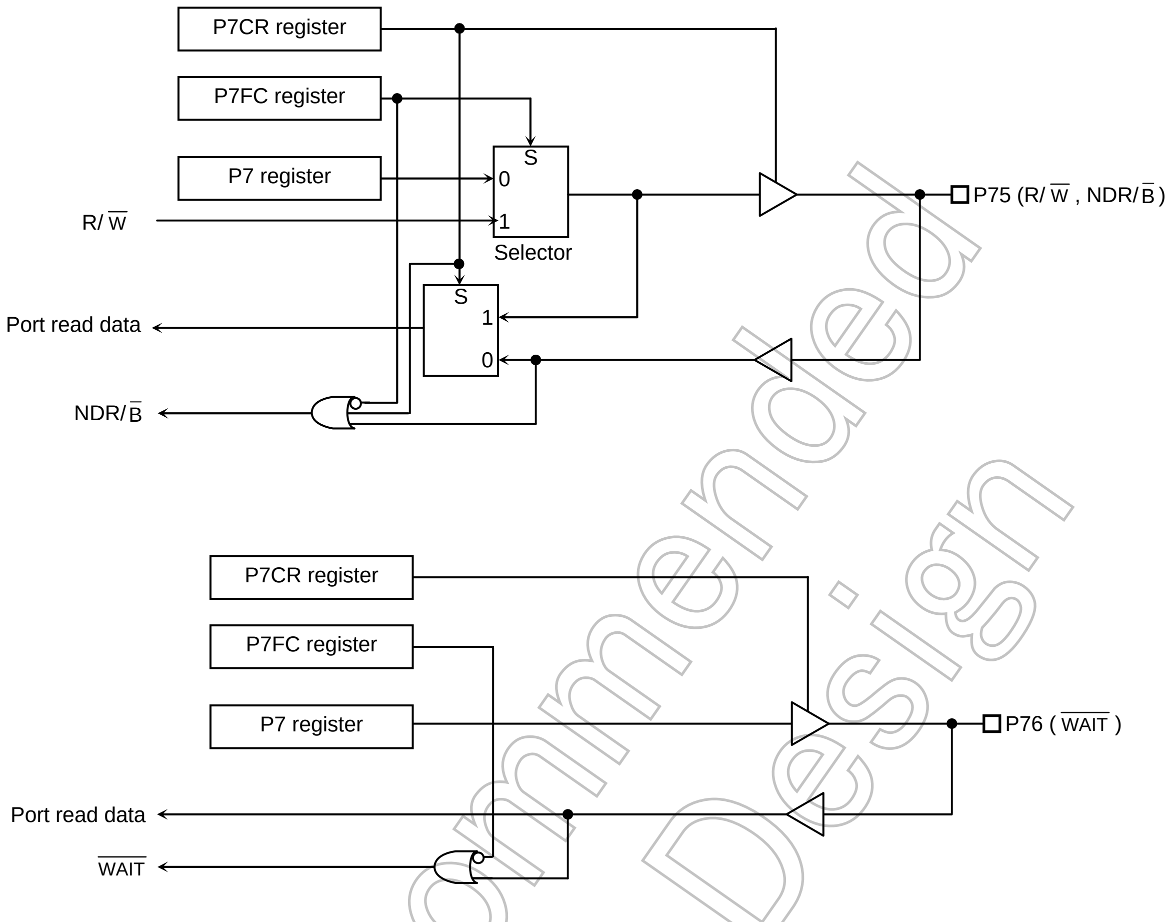


Figure 3.5.14 Port 7

Port 7 register

		7	6	5	4	3	2	1	0
P7 (001CH)	Bit symbol		P76	P75	P74	P73	P72	P71	P70
	Read/Write		R/W						
	Reset State		Data from external port (Output latch register is set to “1”)		0	0	Data from external port (Output latch register is set to “1”)		1

Port 7 Control register

		7	6	5	4	3	2	1	0
P7CR (001EH)	Bit symbol		P76C	P75C			P72C	P71C	
	Read/Write		W				W		
	Reset State		0	0			0	0	
	Function		0: Input port, $\overline{\text{WAIT}}$ 1:Output port	0: Input port, NDR/$\overline{\text{B}}$ 1:Output port, R/$\overline{\text{W}}$			Refer to following table		

Port 7 Function register

		7	6	5	4	3	2	1	0
P7FC (001FH)	Bit symbol		P76F	P75F	P74F	P73F	P72F	P71F	P70F
	Read/Write		W						
	Reset State		0	0	0	0	0	0	0/1 Note 2
	Function		Refer to following table		0: port 1: EA25	0: port 1: EA24	Refer to following table		0: port 1: $\overline{\text{RD}}$

Port 7 Drive register

		7	6	5	4	3	2	1	0
P7DR (0087H)	Bit symbol		P76D	P75D	P94D	P73D	P72D	P71D	P70D
	Read/Write		R/W						
	Reset State		1	1	1	1	1	1	1
	Function		Input/Output buffer drive register for standby mode						

P72 Setting

<P72C> <P72F>	0	1
0	Input port	Output port
1	(Reserved)	$\overline{\text{NDWE}}$ output (at <P72> = 0) $\overline{\text{WRLH}}$ output (at <P72> = 1)

P71 Setting

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

P76 Setting

<P76C> <P76F>	0	1
0	Input port	Output port
1	$\overline{\text{WAIT}}$ input	(Reserved)

P75 Setting

<P75C> <P75F>	0	1
0	Input port	Output port
1	NDR/$\overline{\text{B}}$ input	R/$\overline{\text{W}}$ output

Note 1: Read-modify-write is prohibited for P7CR and P7FC.

Note 2: It is set to “Port” or “ $\overline{\text{RD}}$ ” by AM pin setting.

Note 3: When $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ are used, set registers in the following order to avoid outputting a negative glitch.

Order	Register	Bit2	Bit1
(1)	P7	0	0
(2)	P7FC	1	1
(3)	P7CR	1	1

Figure 3.5.15 Register for Port 7

3.5.8 Port 8 (P80 to P87)

Ports 80 to 87 are 8-bit output ports. Resetting sets the output latch of P82 to “0” and the output latches of P80 to P81, P83 to P87 to “1”.

Port 8 can also be set to function as an interface pin for external memory using function register P8FC.

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

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

Port 82 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

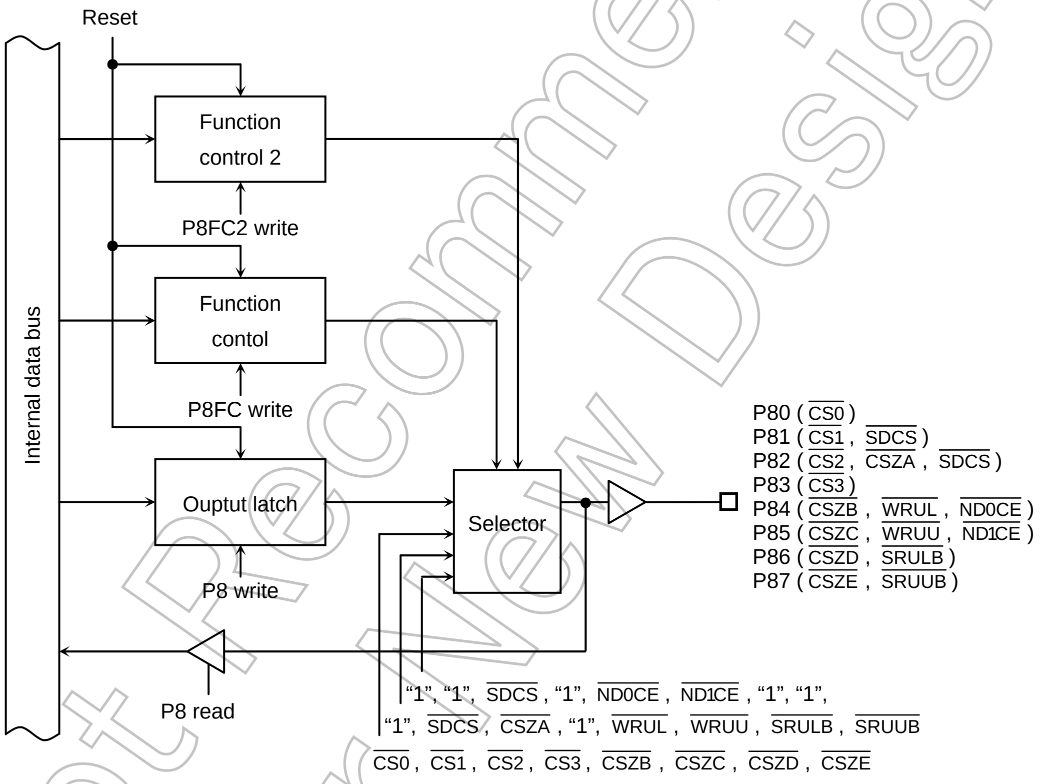


Figure 3.5.16 Port 8

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							
Reset State	1	1	1	1	1	0/1 Note2	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							
Reset State	0	0	0	0	0	0	0	0
Function	0: Port 1: $\overline{\text{CSZE}}$	0: Port 1: $\overline{\text{CSZD}}$	Refer to following table	Refer to following table	0: Port 1: $\overline{\text{CS3}}$	Refer to following table	0: Port 1: $\overline{\text{CS1}}$	0: Port 1: $\overline{\text{CS0}}$

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							
Reset State	0	0	0	0	0	0	0	0
Function	0: <P87F> 1: $\overline{\text{SRUUB}}$	0: <P86F> 1: $\overline{\text{SRULB}}$	Refer to following table	Refer to following table	Always write “0”	Refer to table below	0: <P81F> 1: $\overline{\text{SDCS}}$	Always 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							
Reset State	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

P85 Setting

<P85F> <P85F2>	0	1
0	Output port	$\overline{\text{CSZC}}$ output
1	$\overline{\text{WRUU}}$ output	$\overline{\text{ND1CE}}$ output

P84 Setting

<P84F> <P84F2>	0	1
0	Output port	$\overline{\text{CSZB}}$ output
1	$\overline{\text{WRUL}}$ output	$\overline{\text{ND0CE}}$ output

P82 Setting

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

Note 1: Read-modify-write is prohibited for P8FC and P8FC2.

Note 2: It is set to “0” or “1” by AM pin setting.

Note 3: In MULTI16 or MULTI32 mode, do not write “1” to P8<P82> register before setting P82 pin to $\overline{\text{CS2}}$ or $\overline{\text{CSZA}}$ because, on reset, P82 pin outputs “0” as $\overline{\text{CE}}$ for program memory.

Figure 3.5.17 Register for Port 8

3.5.9 Port 9 (P90 to P97)

P90 to P94 are 5-bit general-purpose I/O ports. I/O can be set on a 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 ports. P90 to P92 function as SIO or I^{2}S, P93 to 95 as output pins for an LCD controller and P96 to P97 as input pins for external interruption (INT4, INT5). In addition, P95 functions as the output pin for a low frequency oscillator, P96 to P97 as PX and PY pins for a touch screen interface.

Setting the corresponding bits of P9CR and P9FC enables the respective functions.

Resetting resets the P9FC to "0", and sets all bits except P95 to input ports.

(1) Port 90 (TXD0, I2SCKO), Port91 (RXD0, I2SDO), Port 92 (SCLK0, $\overline{\text{CTS0}}$ I2SWS)

Ports 90 to 92 are general-purpose I/O ports. They also function as either SIO0 or I^{2}S. Each pin is detailed below.

	SIO mode (SIO0 module)	UART, IrDA mode (SIO0 module)	I^{2}S mode (I^{2}S module)	SIO mode (I^{2}S 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)	$\overline{\text{CTS0}}$ (Clear to send)	I2SWS (Word select output)	(No use)

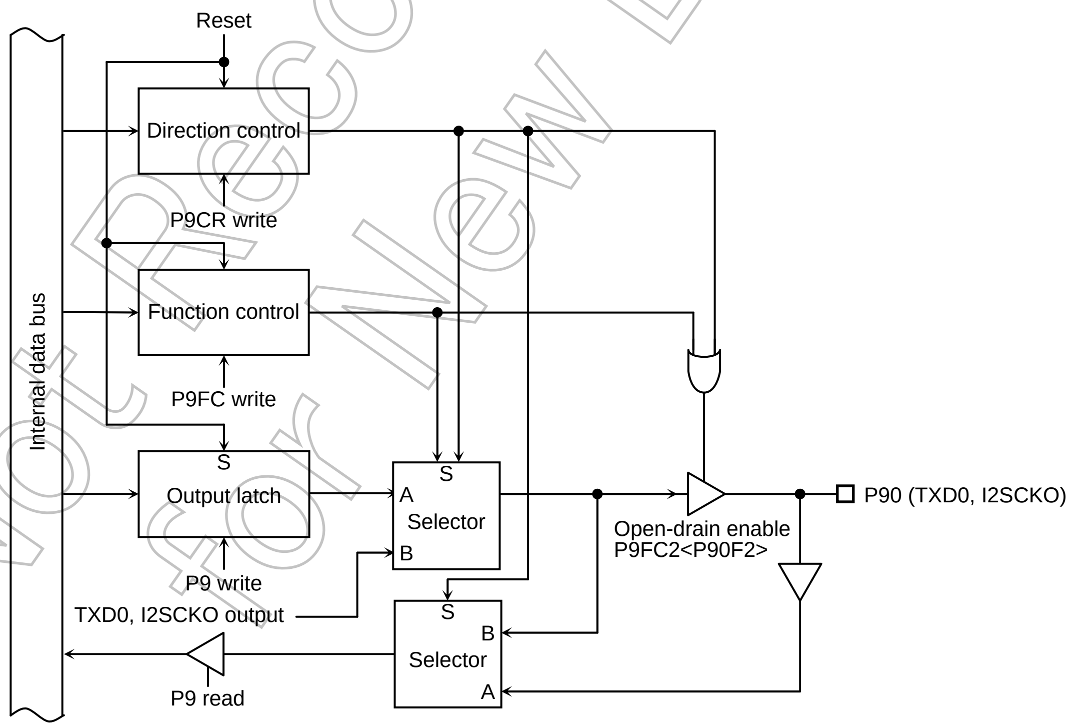


Figure 3.5.18 P90

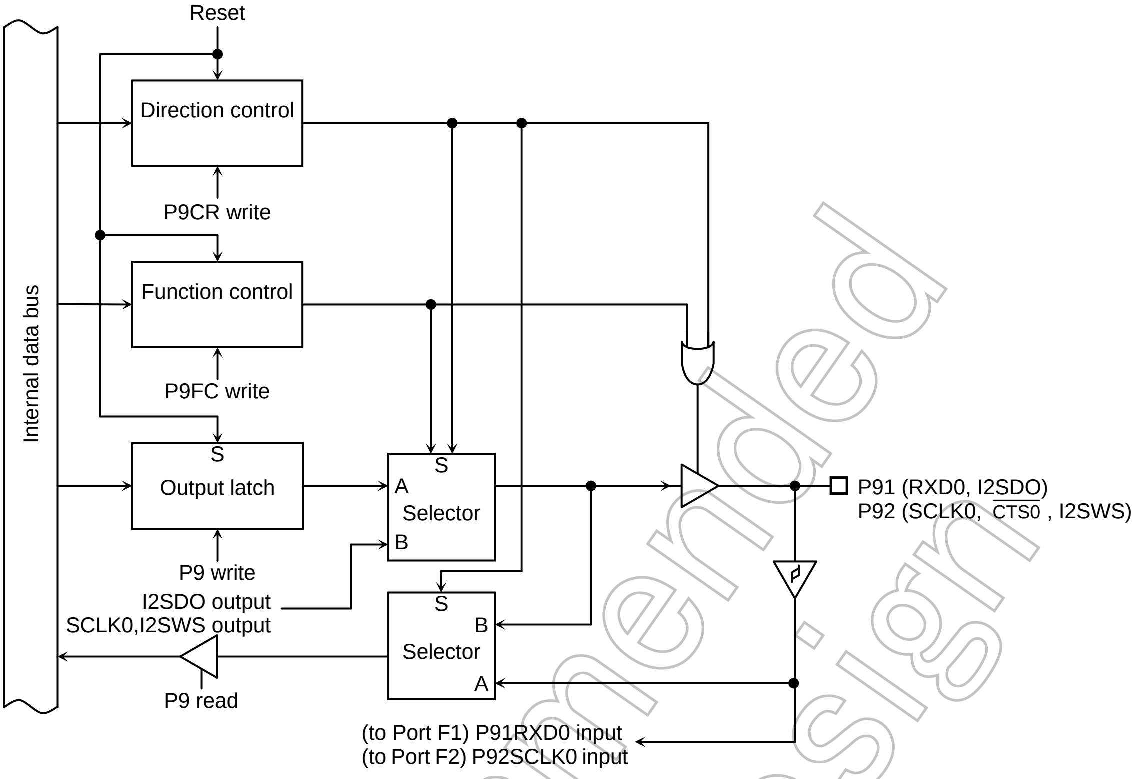


Figure 3.5.19 P91 and P92

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

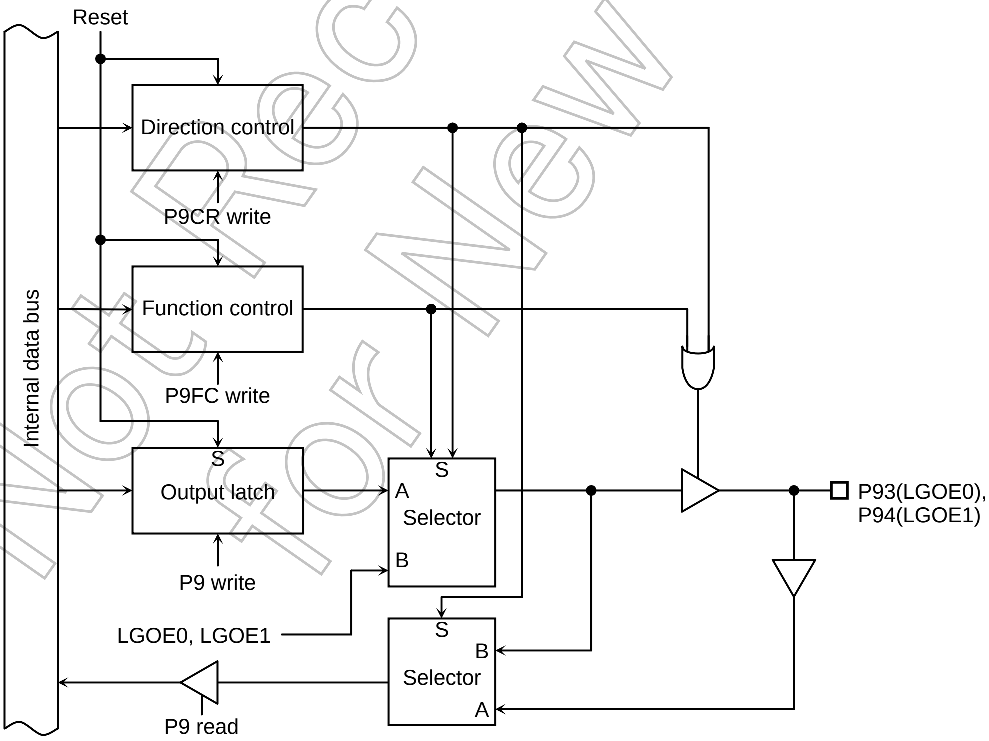


Figure 3.5.20 Port 93 and 94

(3) P95 (CLK32KO, LGOE2)

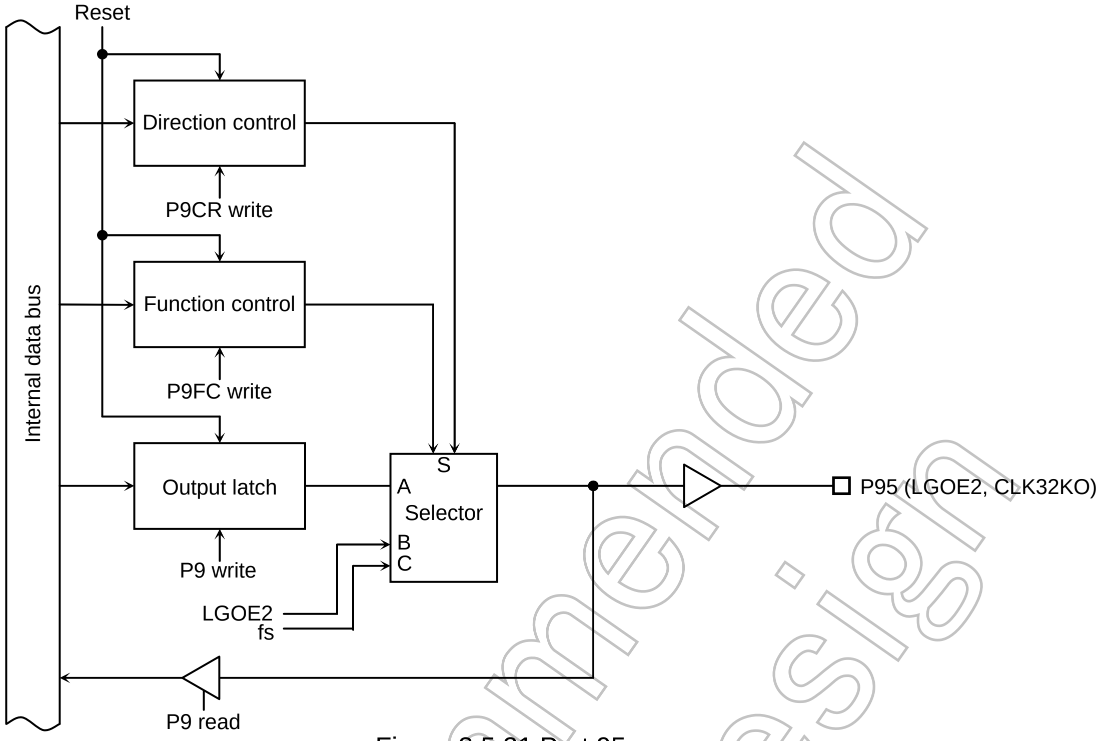


Figure 3.5.21 Port 95

(4) P96 (INT4, PX), P97 (INT5, PY)

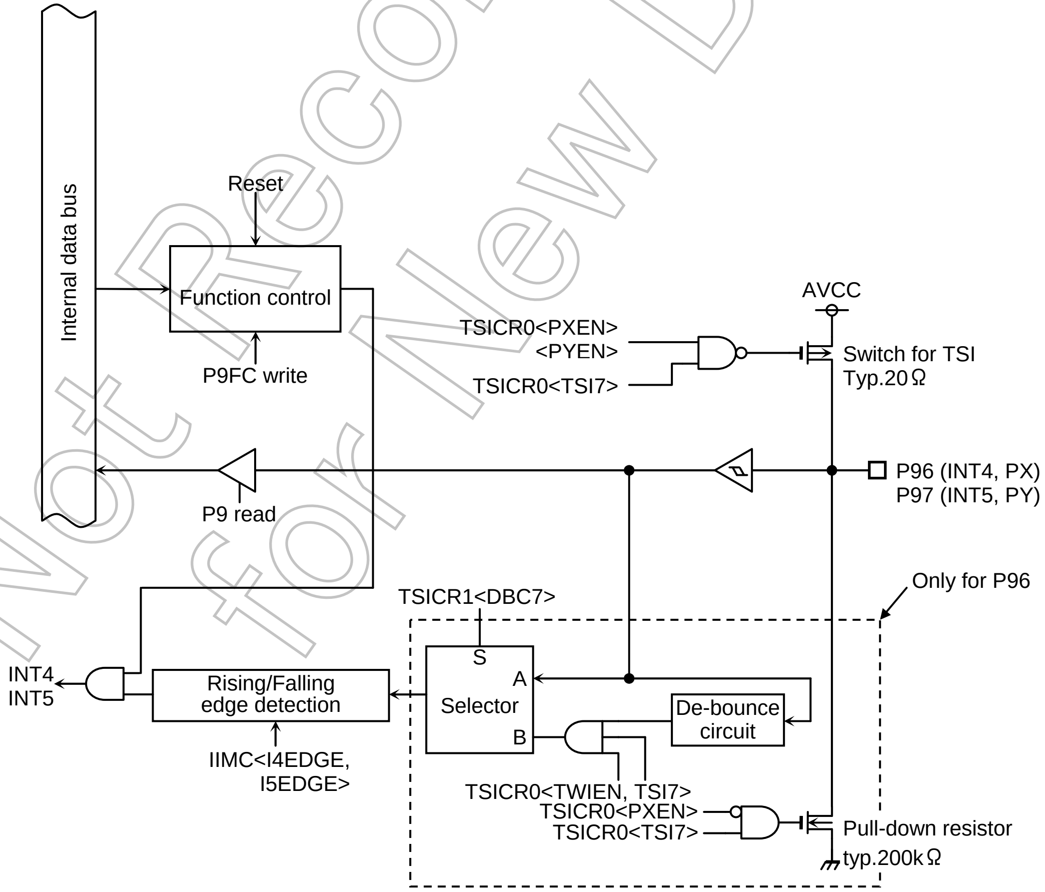


Figure 3.5.22 Port 96, 97

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					
Reset State	Data from external port		0	Data from external port (Output latch register is set to “1”)				

Port 9 Function Register

P9FC (0026H)

	7	6	5	4	3	2	1	0
Bit symbol			P95C	P94C	P93C	P92C	P91C	P90C
Read/Write			W					
Reset State			0	0	0	0	0	0
Function			Refer to following table					

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							
Reset State	0	0	0	0	0	0	0	0
Function	0: Input port 1: INT5	0: Input port 1: INT4	Refer to following table					

P92 Setting

<P92C> <P92F>	0	1
0	Input port SCLK0, $\overline{\text{CTS0}}$ input	Output port
1	I2SWS output	SCLK0 output

P91 Setting

<P91C> <P91F>	0	1
0	Input port RXD0 input	Output port
1	I2SDO output	(Reserved)

P90 Setting

<P90C> <P90F>	0	1
0	Input port	Output port
1	I2SCKO output	TXD0 output

P95 Setting

<P95C> <P95F>	0	1
0	Output port	CLK32KO output
1	LGOE2 output	(Reserved)

P94 Setting

<P94C> <P94F>	0	1
0	Input port	Output port
1	LGOE1 output	(Reserved)

P93 Setting

<P93C> <P93F>	0	1
0	Input port	Output port
1	LGOE0 output	(Reserved)

Port 9 Function Register 2

P9FC2 (0025H)

	7	6	5	4	3	2	1	0
Bit symbol								P90F2
Read/Write								W
Reset State								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							
Reset State	1	1	1	1	1	1	1	1
Function	Output/Input buffer drive register for standby mode							

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

Figure 3.5.23 Register for Port 9

3.5.10 Port A (PA0 to PA7)

Ports A0 to A7 are 8-bit input ports with pull-up resistor. In addition to functioning as general-purpose I/O ports, ports A0 to A7 can also, as a keyboard interface, operate a key-on wakeup function. 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.

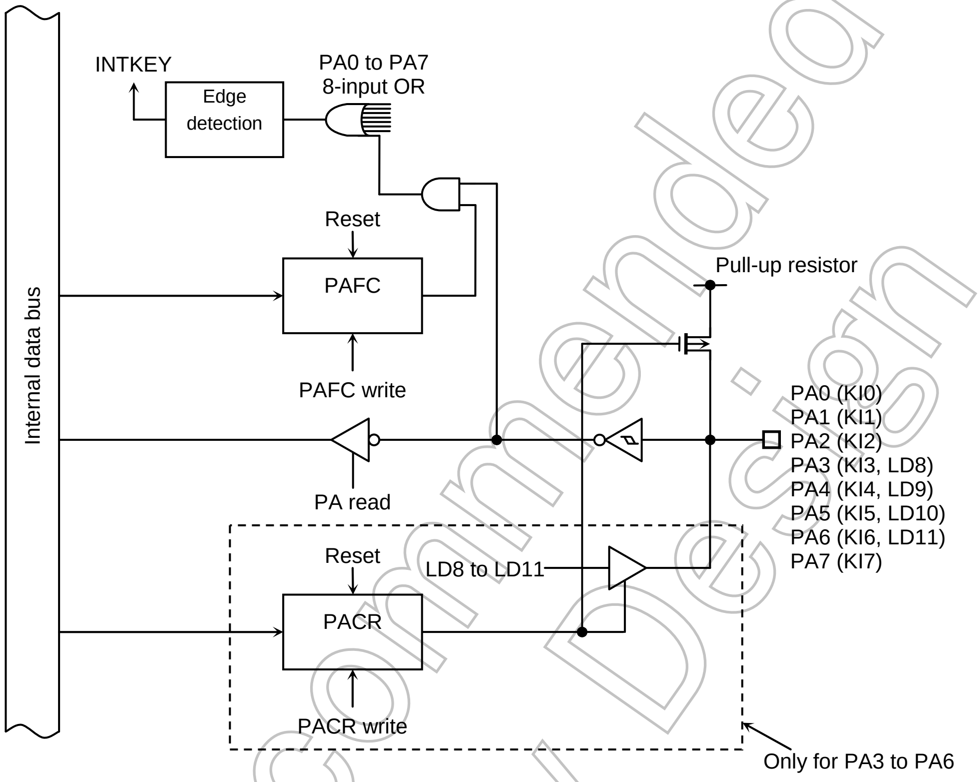


Figure 3.5.24 Port A

When PAFC = “1”, if the input of any of KI0 to KI7 pins fall down, an INTKEY interrupt is generated. An INTKEY interrupt can be used to release all HALT modes.

Port A Register

		7	6	5	4	3	2	1	0
PA (0028H)	Bit symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
	Read/Write	R/W							
	Reset State	Data from external port							

Port A Function Register

		7	6	5	4	3	2	1	0
PAFC (002BH)	Bit symbol	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Key input disable 1: Key input enable							

Port A Control Register

		7	6	5	4	3	2	1	0
PACR (002AH)	Bit symbol		PA6C	PA5C	PA4C	PA3C			
	Read/Write		W						
	Reset State		0	0	0	0			
	Function		0: Input port or Key input 1: LD11 to LD8 output						

Port A Drive register

		7	6	5	4	3	2	1	0
PADR (008AH)	Bit symbol	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
	Read/Write	W							
	Reset State	1	1	1	1	1	1	1	1
	Function	Input/Output buffer drive register for standby mode							

Note: Read-modify-write is prohibited for PACR and PAFC.

Figure 3.5.25 Register for Port A

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

PC0 to PC3, PC6 and PC7 are 6-bit general-purpose I/O ports. 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 an output pin for timers (TA1OUT, TA3OUT and TB0OUT0), input pin for external interruption (INT0 to INT3), output pin for memory ($\overline{\text{CSZF}}$), output pin for key (KO8) and output pin for LCD driver (LDIV, LCP1). These settings are made using the function register PCFC. The edge select for external interruption is determined by the IIMC register in the interruption controller.

(1) PC0 (INT0, TA1OUT)

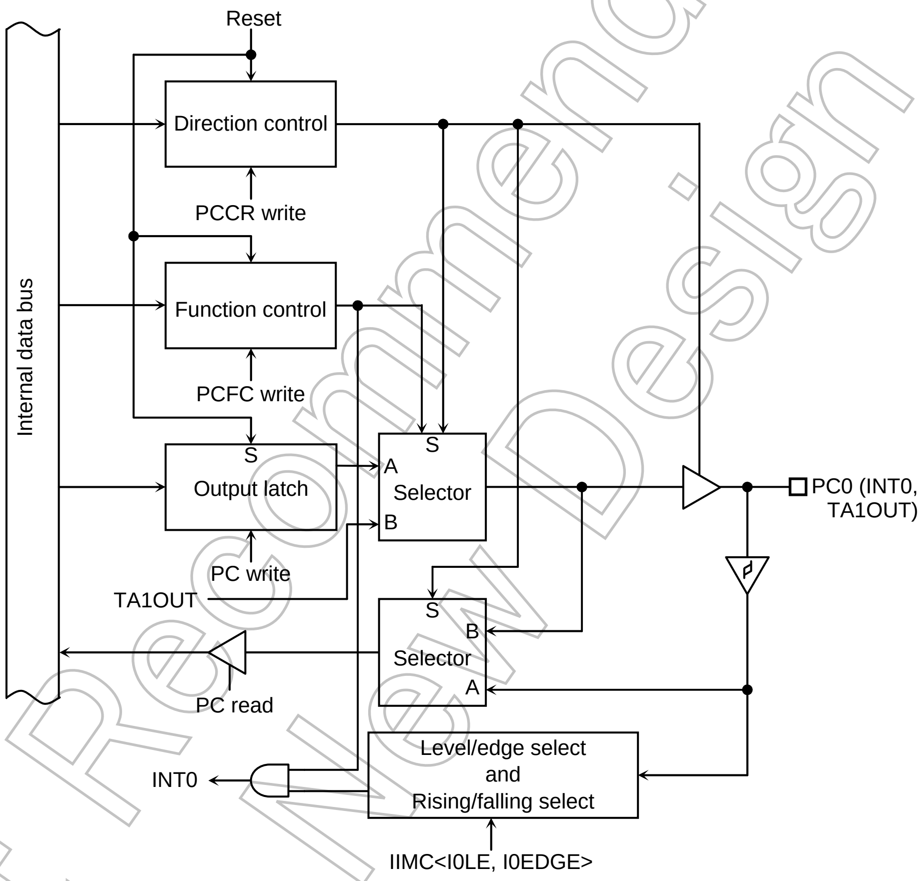


Figure 3.5.26 Port C0

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

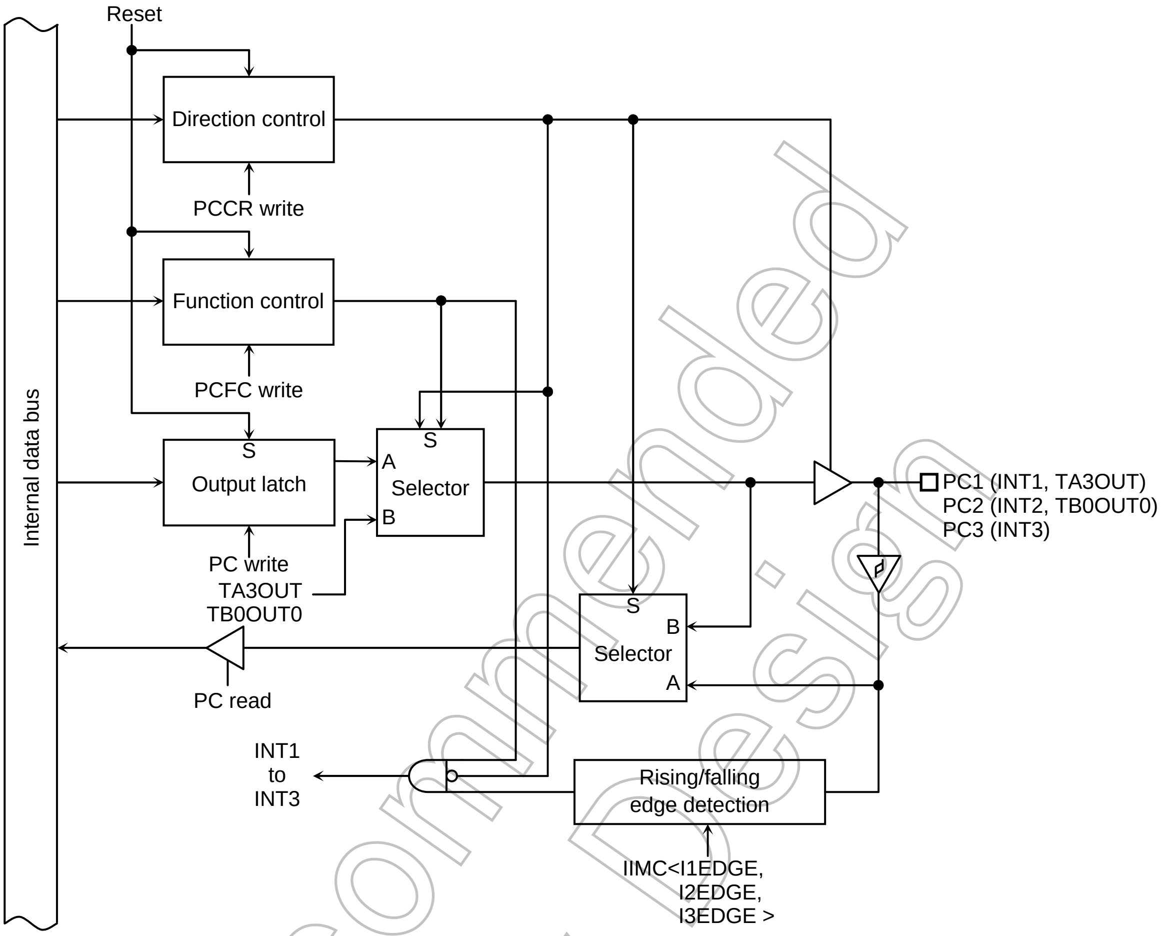


Figure 3.5.27 Port C1, C2, C3

(3) PC6 (KO8, LDIV)

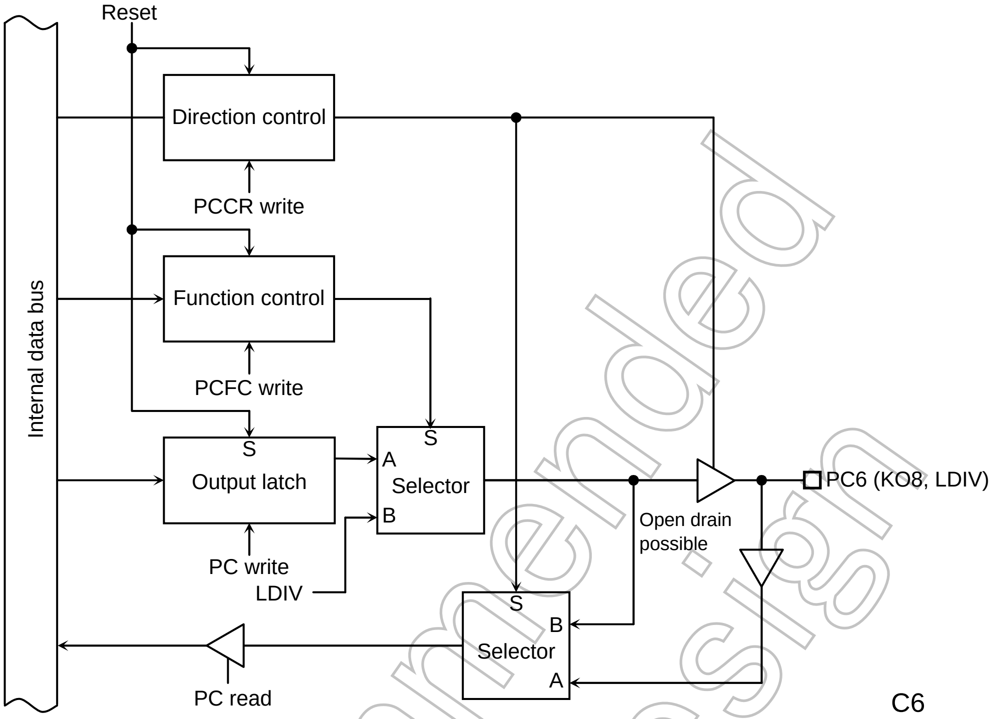


Figure 3.5.28 Port C6

(4) PC7 ($\overline{\text{CSZF}}$, LCP1)

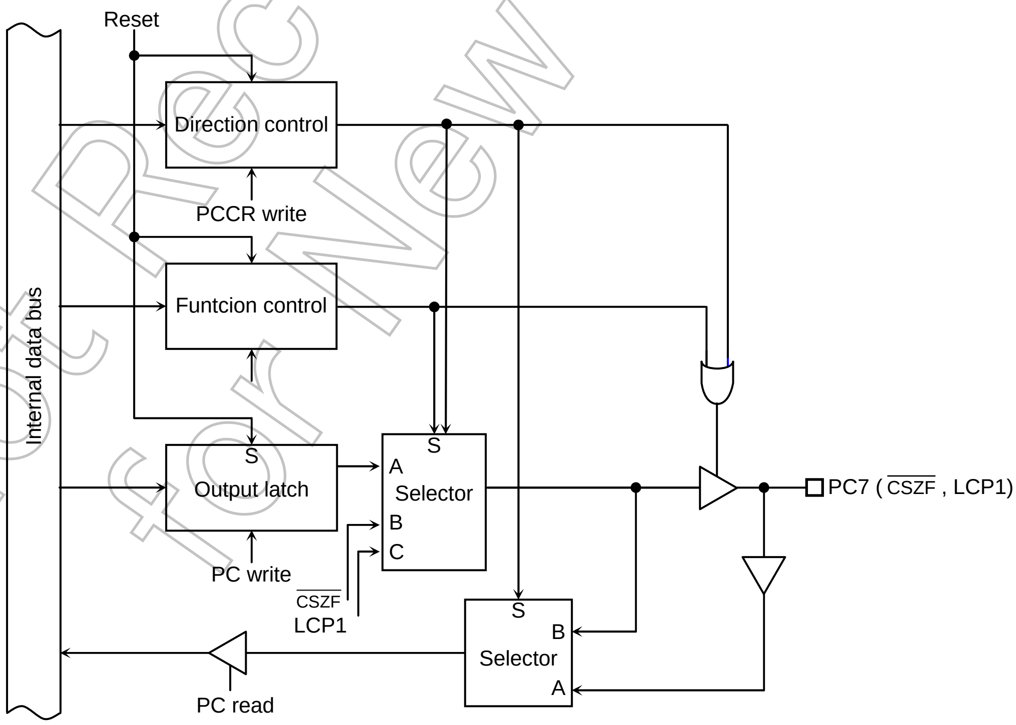


Figure 3.5.29 Port C7

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			
Reset State	Data from external port (Output latch register is set to “1”)				Data from external port (Output latch register is set to “1”)			

Port C Control Register

PCCR
(0032H)

	7	6	5	4	3	2	1	0
Bit symbol	PC7C	PC6C			PC3C	PC2C	PC1C	PC0C
Read/Write	W				W			
Reset State	0	0			0	0	0	0
Function	Refer to following table				Refer to following table			

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			
Reset State	0	0			0	0	0	0
Function	Refer to following table				Refer to following table			

PC2 Setting

<PC2C> <PC2F>	0	1
0	Input port	Output port
1	INT2	TB0OUT

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	$\overline{\text{CSZF}}$ Output	LCP1 Output

PC6 Setting

<PC6C> <PC6F>	0	1
0	Input port	Output port
1	KO8 (Open drain)	LDIV Output

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				R/W			
Reset State	1	1			1	1	1	1
Function	Input/Output buffer drive register for standby mode				Input/Output buffer drive register for standby mode			

Note: Read-modify-write is prohibited for the registers PCCR and PCFC.

Figure 3.5.30 Register for Port C

3.5.12 Port F (PF0 to PF2, PF7)

Ports 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 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).

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

(1) Port F0 (TXD0, TXD1), F1 (RXD0, RXD1), F2 (SCLK0, $\overline{\text{CTS0}}$ SCLK1, $\overline{\text{CTS1}}$)

Ports F0 to F2 are general-purpose I/O ports. They also function as either SIO0 or SIO1. Each pin is detailed 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)	$\overline{\text{CTS0}}$ (Clear to send)	SCLK1 (Clock input or output)	$\overline{\text{CTS1}}$ (Clear to send)

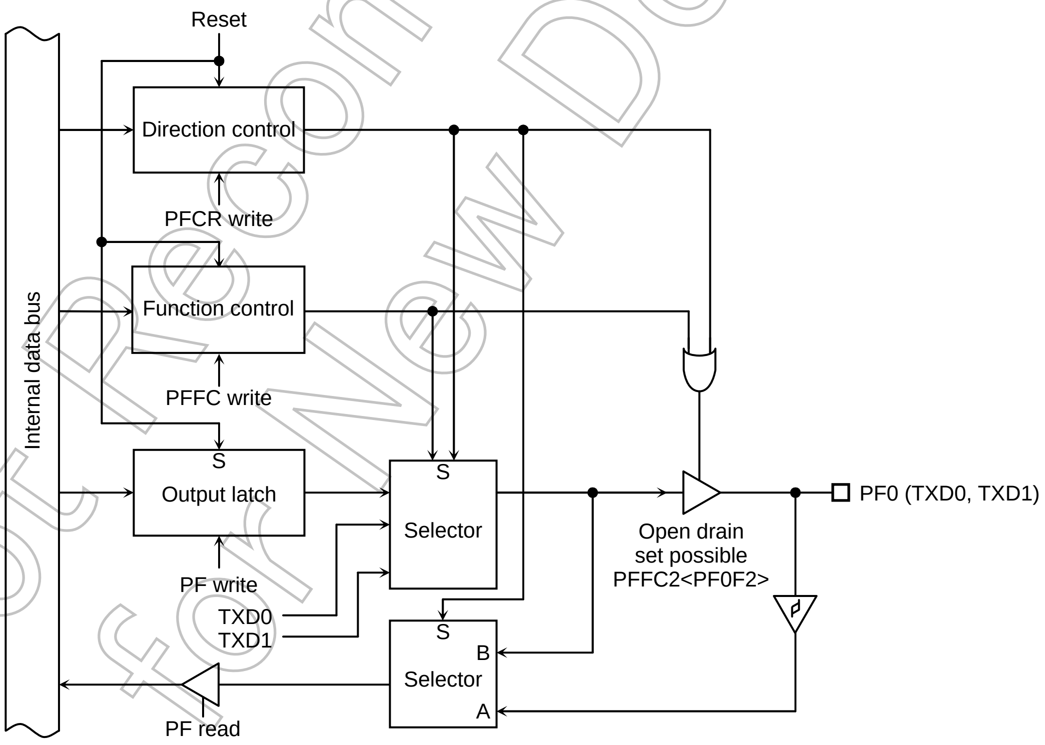


Figure 3.5.31 Port F0

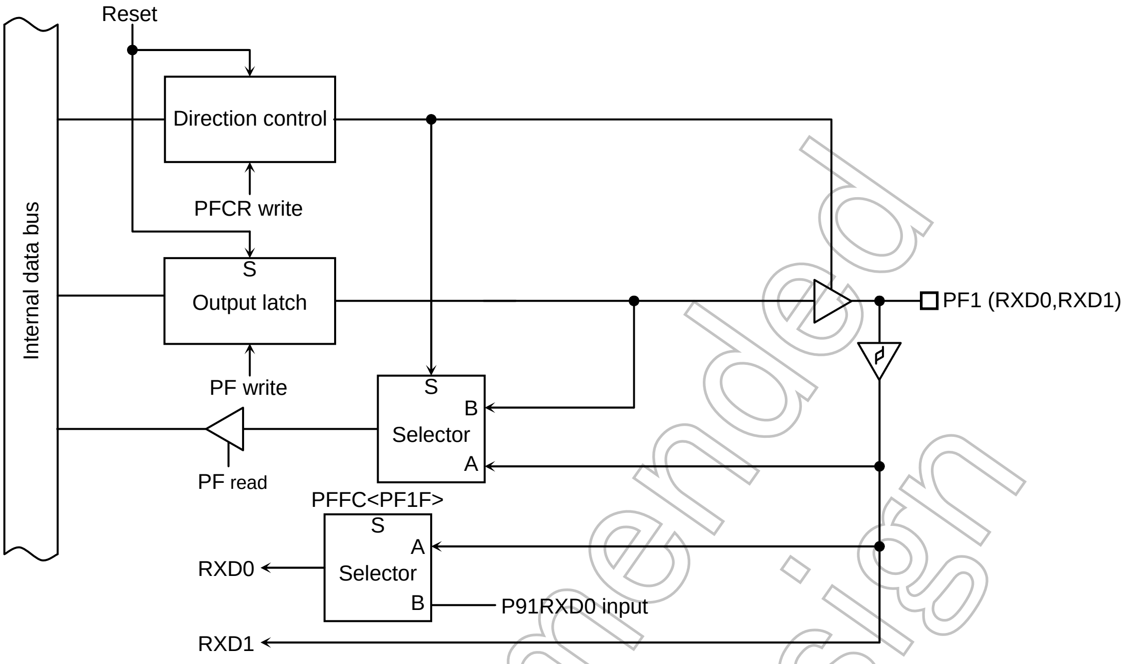


Figure 3.5.32 Port F1

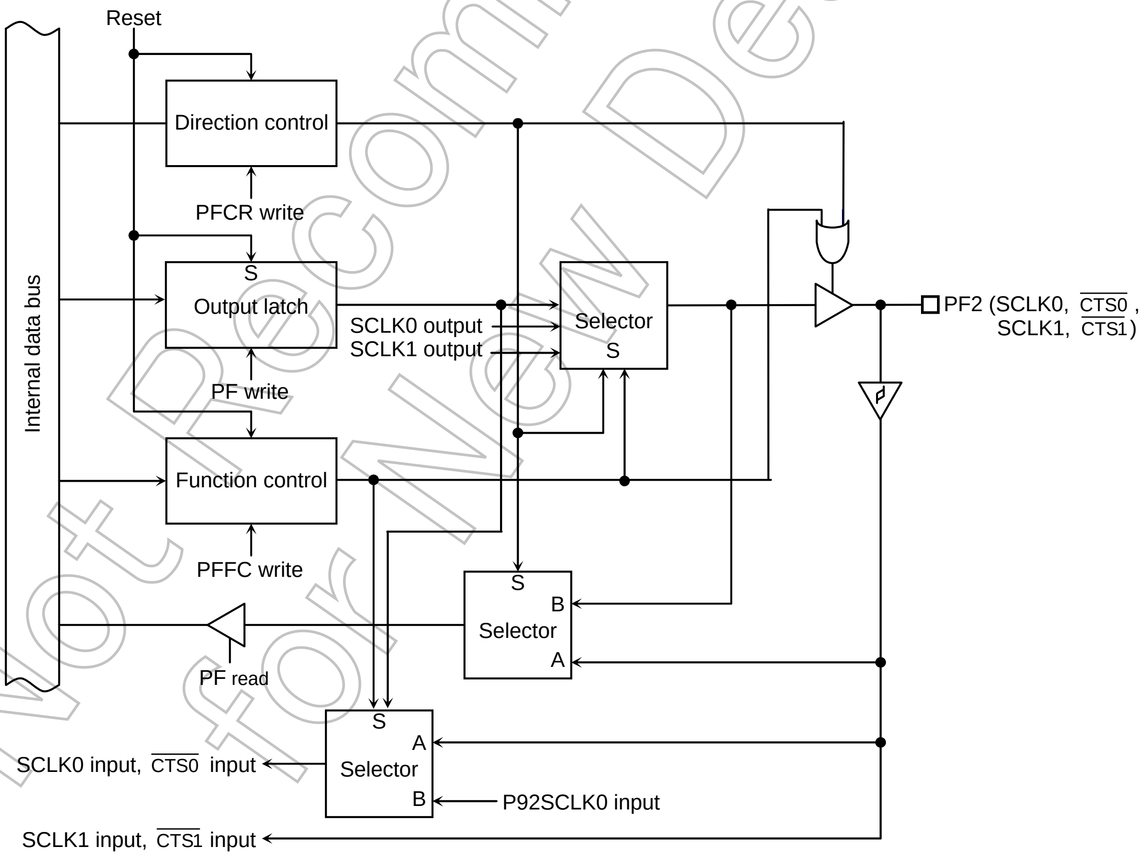


Figure 3.5.33 Port F2

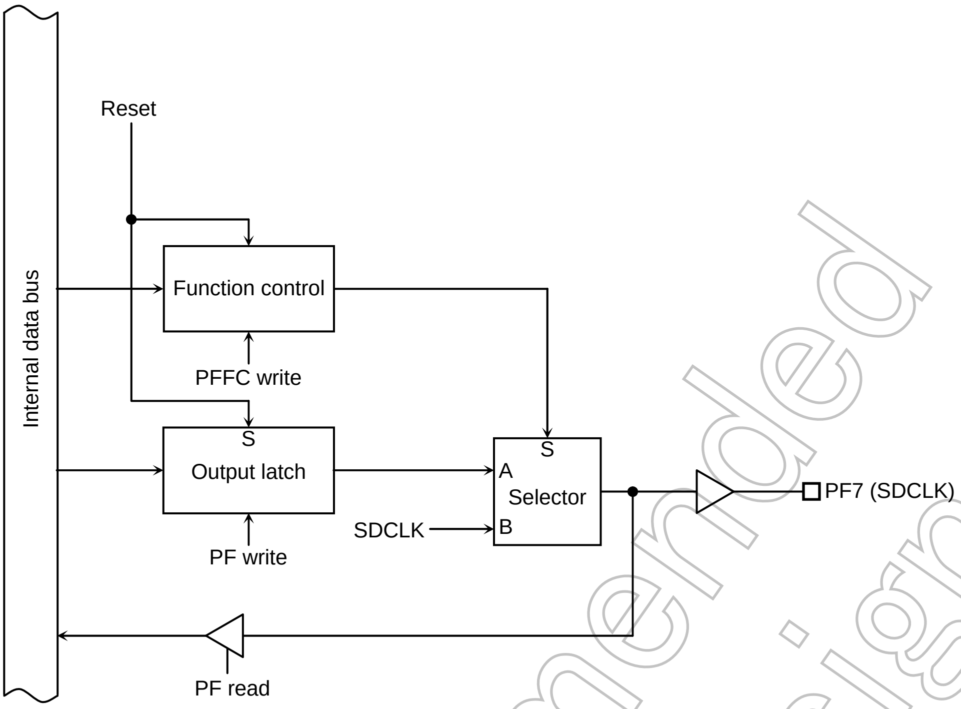


Figure 3.5.34 Port F7

Port F Register

		7	6	5	4	3	2	1	0
PF (003CH)	Bit symbol	PF7					PF2	PF1	PF0
	Read/Write	R/W					R/W		
	Reset State	1					External data (Output latch register is set to “1”)		

Port F Control Register

		7	6	5	4	3	2	1	0
PFCR (003EH)	Bit symbol						PF2C	PF1C	PF0C
	Read/Write						W		
	Reset State						0	0	0
	Function						Refer to following table		

Port F Functon Register

		7	6	5	4	3	2	1	0
PFFC (003FH)	Bit symbol	PF7F					PF2F	PF1F	PF0F
	Read/Write	W					W		
	Reset State	1					0	0	0
	Function	0: Output 1: SDCLK					Refer to following table	RXD0 pin selection 0: Port F1 1: Port 91	Refer to following table

PF2 Setting

<PF2C> <PF2F>	0	1
0	Input port or SCLK1, $\overline{\text{CTS1}}$ input or SCLK0, $\overline{\text{CTS0}}$ input From PF2 pin at <PF2> = 0 From P92 pin at <PF2> = 1	Output port
1	SCLK1 output	SCLK0 output

PF1 Setting

<PF1C> <PF1F>	0	1
0	Input port or RXD0/RXD1 input	Output port
1		

PF0 Setting

<PF0C> <PF0F>	0	1
0	Input port	Output port
1	TXD1 output	TXD0 output

Port F Functon Register 2

		7	6	5	4	3	2	1	0
PFFC2 (003DH)	Bit symbol								PF0F2
	Read/Write								W
	Reset State								0
	Function								Output buffer 0: CMOS 1: Open drain

Port F Drive Register

		7	6	5	4	3	2	1	0
PFDR (008FH)	Bit symbol	PF7D					PF2D	PF1D	PF0D
	Read/Write	R/W					R/W		
	Reset State	1					1	1	1
	Function	Input/Output buffer drive register for standby mode					Input/Output buffer drive register for standby mode		

Note: Read-modify-write is prohibited for the registers PFCR, PFFC and PFFC2.

Figure 3.5.35 Register for Port F

3.5.13 Port G (PG0 to PG3)

PG0 to PG3 are 4-bit input ports and can also be used as the analog input pins for the internal AD converter. PG3 can also be used as the ADTRG pin for the AD converter.

PG2 and PG3 can also be used as the MX and MY pins for a touch screen interface.

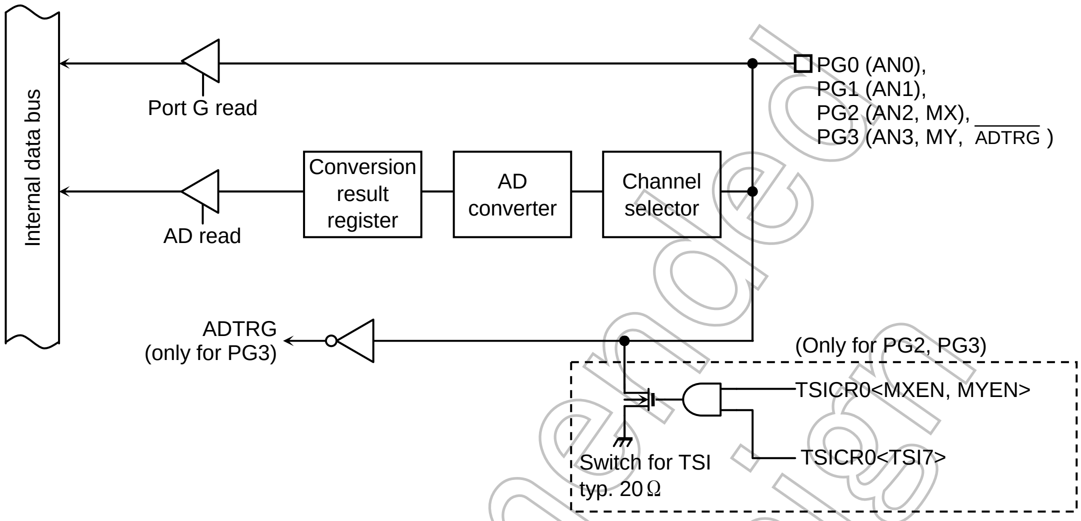


Figure 3.5.36 Port G

Port G Register

		7	6	5	4	3	2	1	0
PG (0040H)	Bit symbol					PG2	PG2	PG1	PG0
	Read/Write					R			
	Reset State					Data from external port			

Note: The input channel selection of the AD converter and the permission for ADTRG input are set by AD converter mode register ADMOD1.

Port G Drive Register

		7	6	5	4	3	2	1	0
PGDR (0090H)	Bit symbol					PG3D	PG2D		
	Read/Write					R/W			
	Reset State					1	1		
	Function					Input/Output buffer drive register for standby mode			

Figure 3.5.37 Register for Port G

3.5.14 Port J (PJ0 to PJ7)

PJ0 to PJ4 and PJ7 are 6-bit output ports. Resetting sets the output latch PJ to “1”, and they output “1”. PJ5 to PJ6 are 2-bit I/O ports.

In addition to functioning as a port, port J also functions as output pins for SDRAM ($\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, $\overline{\text{SDWE}}$, SDLLDQM, SDLUDQM, SDULDQM, SDUUDQM and SDCKE), SRAM ($\overline{\text{SRWR}}$, $\overline{\text{SRLLB}}$ $\overline{\text{SRLUB}}$) and NAND flash (NDALE and NDCLE).

The above settings are made using the function register PJFC.

However, H either SDRAM or SRAM output signals for PJ0 to PJ2 are selected automatically according to the setting of the memory controller.

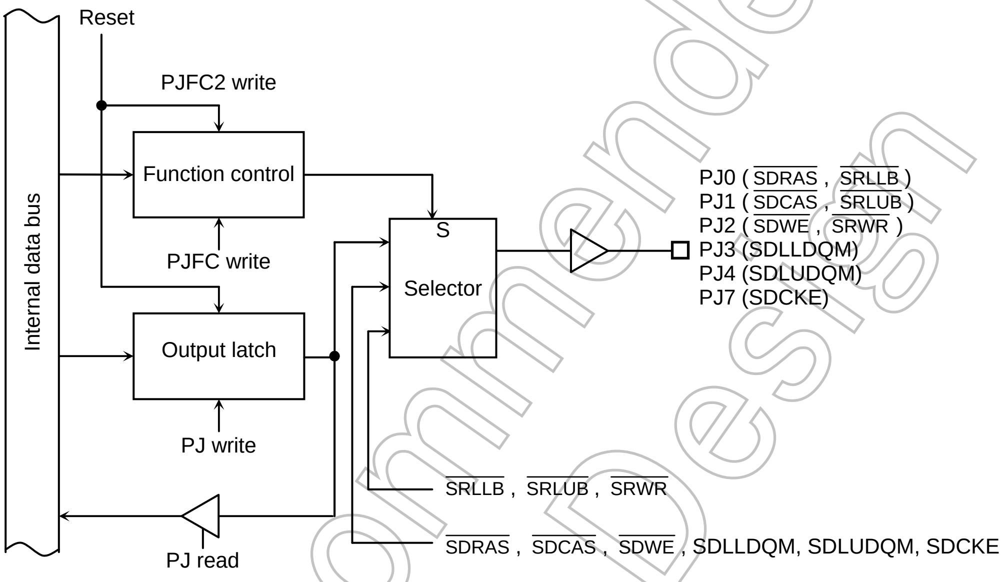

Reset
PJFC2 write
Function control
Internal data bus
PJFC write
Output latch
PJ write
PJ read
S
Selector
PJ0 ($\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$)
PJ1 ($\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$)
PJ2 ($\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$)
PJ3 (SDLLDQM)
PJ4 (SDLUDQM)
PJ7 (SDCKE)
$\overline{\text{SRLLB}}$, $\overline{\text{SRLUB}}$, $\overline{\text{SRWR}}$
$\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, $\overline{\text{SDWE}}$, SDLLDQM, SDLUDQM, SDCKE

Figure 3.5.38 Port J0, J1, J2, J3, J4 and J7

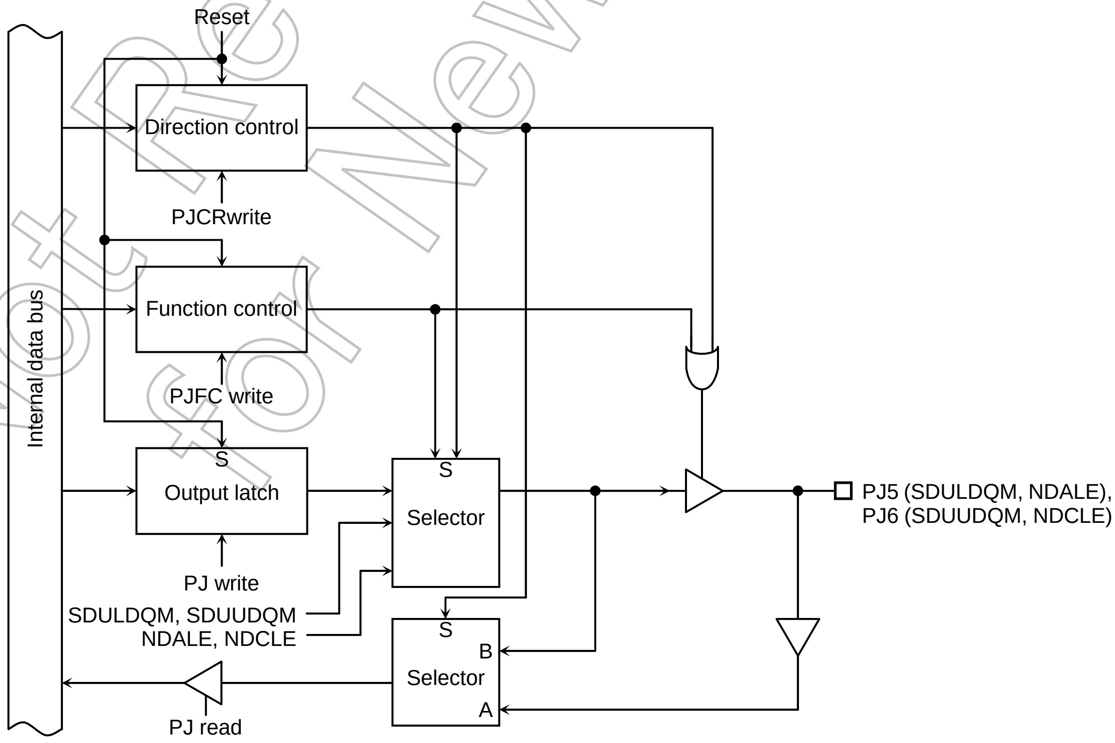


Figure 3.5.39 Port J5 and J6

Port J Register

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

Port J Control Register

		7	6	5	4	3	2	1	0
PJCR (004EH)	Bit symbol		PJ6C	PJ5C					
	Read/Write		W						
	Reset State		0	0					
	Function		0: Input 1: Output						

Port J Function Register

		7	6	5	4	3	2	1	0
PJFC (004FH)	Bit symbol	PJ7F	PJ6F	PJ5F	PJ4F	PJ3F	PJ2F	PJ1F	PJ0F
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	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: $\overline{SDWE}$, $\overline{SDWR}$	0: Port 1: $\overline{SDCAS}$, $\overline{SRLUB}$	0: Port 1: $\overline{SRRAS}$, $\overline{SRLLB}$

Port J Drive Register

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

Note: Read-modify-write is prohibited for the registers PJCR and PJFC.

Figure 3.5.40 Register for Port J

3.5.15 Port K (PK0 to PK3)

Port K is a 4-bit output port. Resetting sets the output latch PK to “0”, and PK0 to PK3 pins output “0”.

In addition to functioning as an output port, port K also functions as output pins for an LCD controller (LCP0, LLP, LFR and LBCD).

The above settings are made using the function register PKFC.

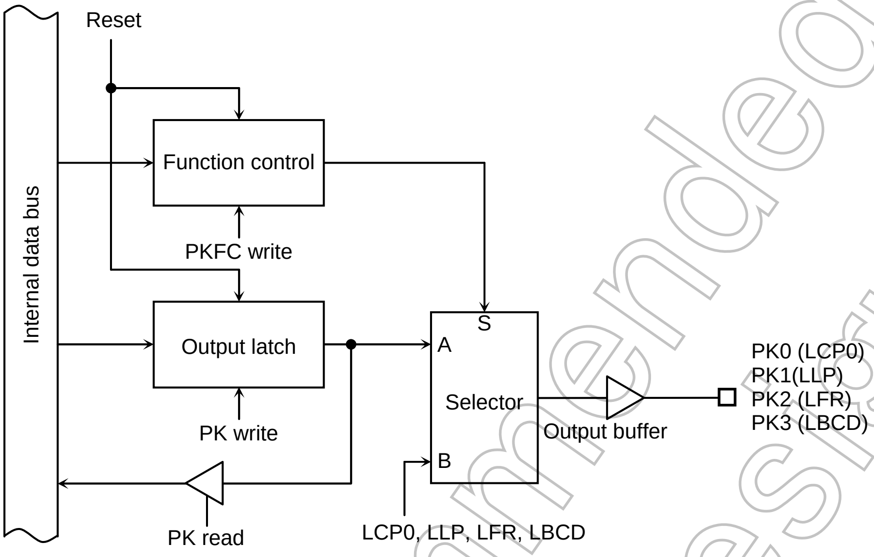


Figure 3.5.41 Port K

Port K Register

		7	6	5	4	3	2	1	0
PK (0050H)	Bit symbol					PK3	PK2	PK1	PK0
	Read/Write					R/W			
	Reset State					0	0	0	0

Port K Function Register

		7	6	5	4	3	2	1	0
PKFC (0053H)	Bit symbol					PK3F	PK2F	PK1F	PK0F
	Read/Write					W			
	Reset State					0	0	0	0
	Function					0: Port 1: LBCD	0: Port 1: LFR	0: Port 1: LLP	0: Port 1: LCP0

Port K Drive Register

		7	6	5	4	3	2	1	0
PKDR (0094H)	Bit symbol					PK3D	PK2D	PK1D	PK0D
	Read/Write					R/W			
	Reset State					1	1	1	1
	Function					Input/Output buffer drive register for standby mode			

Note: Read-modify-write is prohibited for the register PKFC.

Figure 3.5.42 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 resets 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 an LCD controller (LD0 to LD7). The above settings are made using the function register PLFC.

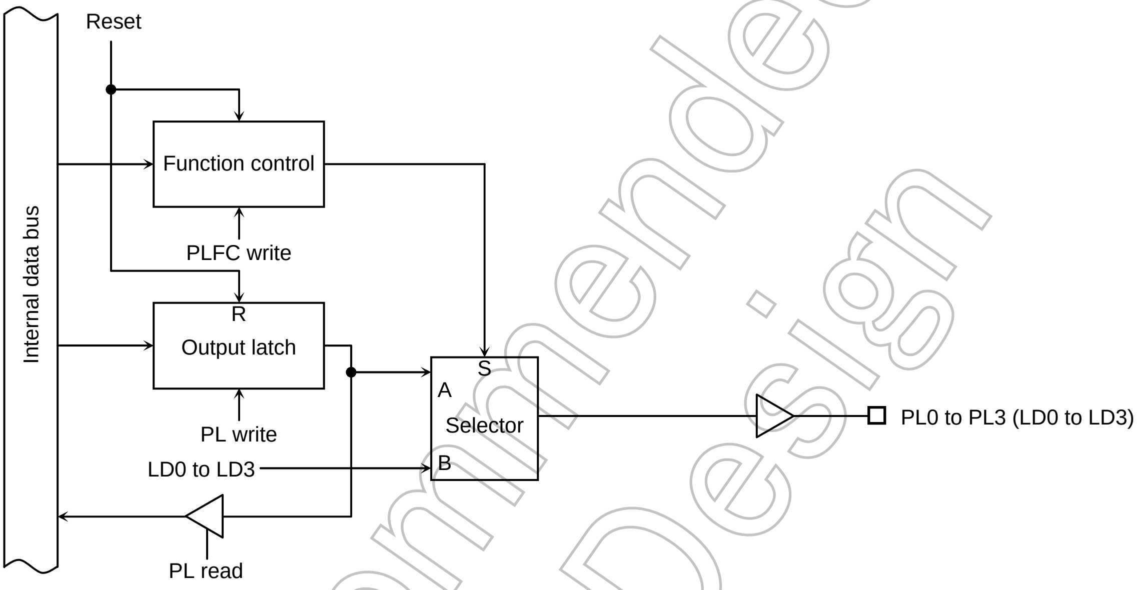


Figure 3.5.43 Register for Port L0 to L3

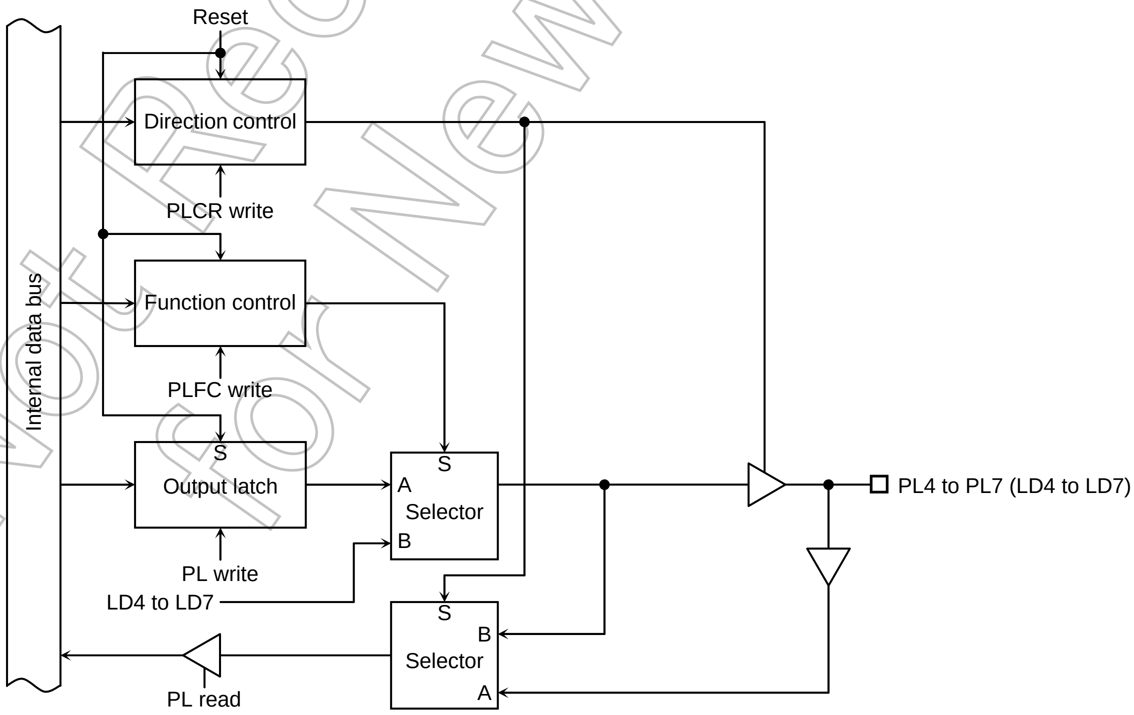


Figure 3.5.44 Register for Port L4 to L7

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							
Reset State	Data from external port (Output latch register is cleared to "0")				0	0	0	0

Port L Control Register

PLCR (0056H)

	7	6	5	4	3	2	1	0
Bit symbol	PL7C	PL6C	PL5C	PL4C				
Read/Write	W							
Reset State	0	0	0	0				
Function	0: Input 1: Output							

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							
Reset State	0	0	0	0	0	0	0	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							
Reset State	1	1	1	1	1	1	1	1
Function	Input/Output buffer drive register for standby mode							

Note: Read-modify-write is prohibited for the registers PLCR and PLFC.

Figure 3.5.45 Port L Register

3.5.17 Port M (PM1 to PM2)

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

In addition to functioning as a port, port M also functions as output pins for the RTC alarm ($\overline{\text{ALARM}}$), and as the output pin for the melody/alarm generator (MLDALM, $\overline{\text{MLDALM}}$).

The above settings are made using the function register PMFC.

Only PM2 has two output functions - $\overline{\text{ALARM}}$ and $\overline{\text{MLDALM}}$. These are selected using PM<PM2>.

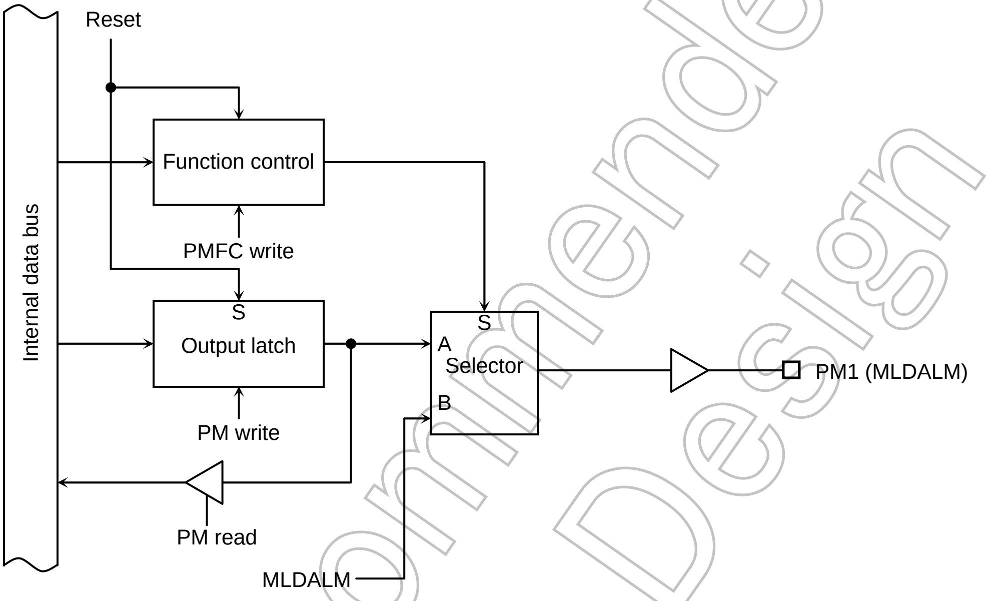


Figure 3.5.46 Port M1

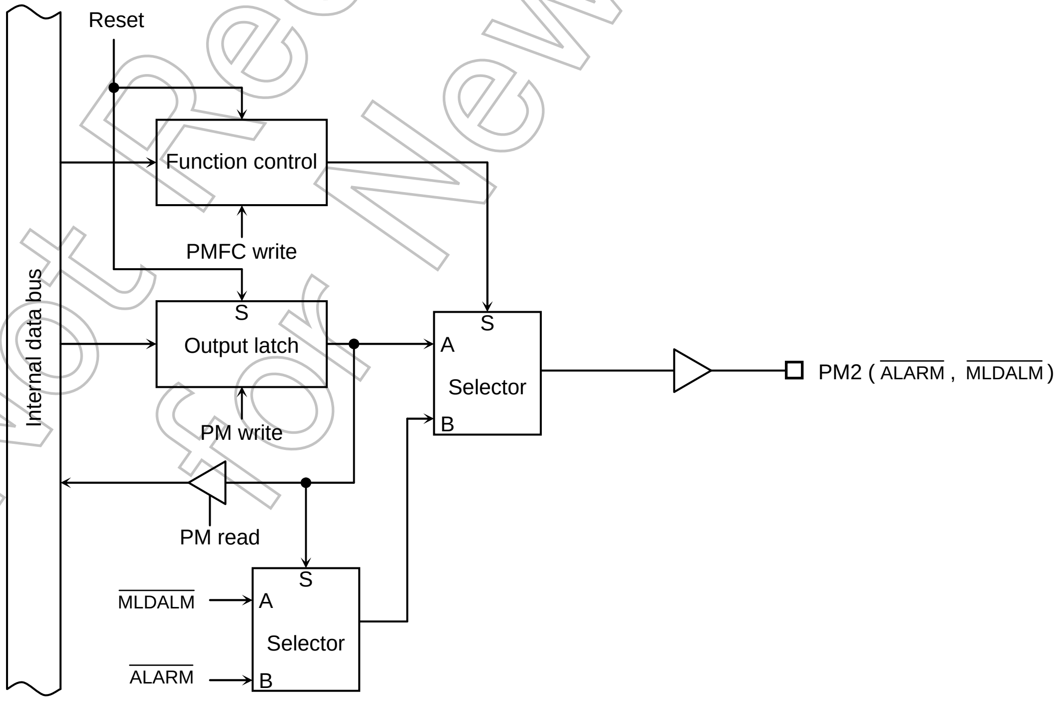

Reset
Internal data bus
Function control
PMFC write
S
Output latch
PM write
PM read
S
A
Selector
B
PM2 ($\overline{\text{ALARM}}$, $\overline{\text{MLDALM}}$)
$\overline{\text{MLDALM}}$
$\overline{\text{ALARM}}$
S
A
Selector
B

Figure 3.5.47 Port M2

Port M Register

PM
(0058H)

	7	6	5	4	3	2	1	0
Bit symbol						PM2	PM1	
Read/Write						R/W		
Reset State						1	1	

Port M Function Register

PMFC
(005BH)

	7	6	5	4	3	2	1	0
Bit symbol						PM2F	PM1F	
Read/Write						W		
Reset State						0	0	
Function						0: Port 1: $\overline{\text{ALARM}}$ at <PM2> = “1” 1: $\overline{\text{MLDALM}}$ at <PM2> = “0”	0: Port 1: MLDALM output	

Port M Drive Register

PMDR
(0096H)

	7	6	5	4	3	2	1	0
Bit symbol						PM2D	PM1D	
Read/Write						R/W		
Reset State						1	1	
Function						Input/Output buffer drive register for standby mode		

Note: Read-modify-write is prohibited for the register PMFC.

Figure 3.5.48 Register for Port M

3.6 Memory Controller

3.6.1 Functions

The 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 the 4-block address area (block 0 to 3).

- SRAM or ROM: All CS blocks (CS0 to CS3) are supported.
- SDRAM : Only either CS1 or CS2 blocks are supported.
- Page ROM : Only CS2 blocks are supported.
- NAND flash : CS0 is recommended for NAND flash (ND0/1FDTR, 001D00H to 001EFFH), RAM built-in LCD driver (001FE0H to 001FEFH). (Regarding NAND flash area, refer to 3.6.6 (2).)

(2) Connecting memory specifications

Specifies SRAM, ROM and SDRAM as memories that connect with the selected address areas.

(3) Data bus width selection

Whether 8 bits, 16 bits or 32 bits is selected as the data bus width of the respective block address areas.

(4) Wait control

Wait specification bit in the control register and $\overline{\text{WAIT}}$ 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 the 6 modes listed below.

0 waits, 1 wait,
2 waits, 3 waits, 4 waits
N waits (controls with $\overline{\text{WAIT}}$ pin)

3.6.2 Control Register and Operation after Reset Release

This section describes the registers that control the memory controller, the state following reset release and the necessary settings.

(1) Control register

The control registers of the memory controller are as follows and in Table 3.6.1 and Table 3.6.2.

- Control register: BnCSH/BnCSL (n = 0 to 3, EX)
  Sets the basic functions of the memory controller; the memory type that is connected, the number of waits which are 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 the method of accessing page ROM.
- Internal boot ROM controls register: BROMCR
  Sets the method of accessing 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				W		
	Reset State		0	1	0		0	1	0
B0CSH (0141H)	Bit symbol	B0E	–	–	B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
	Read/Write	W							
	Reset State	0	0 (Note)	0 (Note)	0	0	0	0	0
MAMR0 (0142H)	Bit symbol	M0V20	M0V19	M0V18	M0V17	M0V16	M0V15	M0V14 to M0V9	M0V8
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR0 (0143H)	Bit symbol	M0S23	M0S22	M0S21	M0S20	M0S19	M0S18	M0S17	M0S16
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
B1CSL (0144H)	Bit symbol		B1WW2	B1WW1	B1WW0		B1WR2	B1WR1	B1WR0
	Read/Write		W				W		
	Reset State		0	1	0		0	1	0
B1CSH (0145H)	Bit symbol	B1E	–	–	B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
	Read/Write	W							
	Reset State	0	0 (Note)	0 (Note)	0	0	0	0	0
MAMR1 (0146H)	Bit symbol	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	M1V15 to M1V9	M1V8
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR1 (0147H)	Bit symbol	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
B2CSL (0148H)	Bit symbol		B2WW2	B2WW1	B2WW0		B2WR2	B2WR1	B2WR0
	Read/Write		W				W		
	Reset State		0	1	0		0	1	0
B2CSH (0149H)	Bit symbol	B2E	B2M	–	B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
	Read/Write	W							
	Reset State	1	0	0 (Note)	0	0	0	0	0
MAMR2 (014AH)	Bit symbol	M2V22	M2V21	M2V20	M2V19	M2V18	M2V17	M2V16	M2V15
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR2 (014BH)	Bit symbol	M2S23	M2S22	M2S21	M2S20	M2S19	M2S18	M2S17	M2S16
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
B3CSL (014CH)	Bit symbol		B3WW2	B3WW1	B3WW0		B3WR2	B3WR1	B3WR0
	Read/Write		W				W		
	Reset State		0	1	0		0	1	0
B3CSH (014DH)	Bit symbol	B3E	–	–	B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
	Read/Write	W							
	Reset State	0	0 (Note)	0 (Note)	0	0	0	0	0
MAMR3 (014EH)	Bit symbol	M3V22	M3V21	M3V20	M3V19	M3V18	M3V17	M3V16	M3V15
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR3 (014FH)	Bit symbol	M3S23	M3S22	M3S21	M3S20	M3S19	M3S18	M3S17	M3S16
	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1

Note 1: Always write “0”.

Note 2:Read-modify-write is prohibited for BnCS0 and BnCSH (n = 0 to 3) registers.

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			
	Reset State					0	0	0	0
BEXCSL (0158H)	Bit symbol		BEXWW2	BEXWW1	BEXWW0		BEXWR2	BEXWR1	BEXWR0
	Read/Write		W				W		
	Reset State		0	1	0		0	1	0
PMEMCR (0166H)	Bit symbol				OPGE	OPWR1	OPWR0	PR1	PR0
	Read/Write				R/W				
	Reset State				0	0	0	1	0
BROMCR (0167H)	Bit symbol							ROMLESS	VACE
	Read/Write							R/W	
	Reset State							0/1	1/0

Note: Read-modify-write is prohibited for BEXCSH and BEXCSL registers.

(2) Operation after reset release

The start data bus width is determined by the state of AM1/AM0 pins just after reset release. The external memory is then 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: The memory to be used on starting after reset must be either NOR flash or masked ROM.
NAND flash and SDRAM cannot be used.

AM1/AM0 pins are valid only just after reset release. In other cases, the data bus width is set by the control register <BnBUS1:0> .

On reset, only the control register (B2CSH/B2CSL) of the block address area 2 becomes effective automatically (B2CSH<B2E> is set to "1" on reset).

The data bus width which is specified by AM1/AM0 pins is loaded to the bit for specification of the bus width of the control register in the block address area 2.

The block address area 2 is set to 000000H to FFFFFFH address on 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). The control register (BnCS) is then set.

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

3.6.3 Basic Functions and Register Setting

This section describes the setting of the block address area, the connecting memory and the number of waits out of the memory controller's functions.

(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 the register value and the address every bus cycle. 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 value that is set to the register is compared with the block address area on the bus. If the result is a match, the memory controller sets the chip select signal (CSn) to "low".

(i) Memory start address register setting

The MS23 to MS 16 bits of the memory start address register correspond with addresses A23 to A16 respectively. The lower start addresses A15 to A0 are always set to address 0000H.

Therefore the start addresses of the block address area are set to all 64 Kbytes of addresses 000000H to FF0000H.

(ii) Memory address mask register setting

The memory address mask register determines whether an address bit is compared or not. In register setting, "0" is "compare", and "1" is "do not compare".

The address bits that can be set depends 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 upper bits are always compared. The block address area size is determined by the result of the comparison.

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 CS3			○	○	○	○	○	○	○	○	○

Note: After reset release, only the control register of the block address area 2 is valid. The control register of block address area 2 has the <B2M> bit. If the <B2M> bit is set to "0", block address area 2 is set to addresses 000000H to FFFFFFH. (This is the state following reset release .) If the <B2M> bit is set to "1", the start address and the address area size are set, as in the other block address areas.

(iii) Example of register setting

To set the block address area 64 Kbytes 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 M1S16 bits of the memory start address register MSAR1 correspond with address A23 to A16.

A15 to A0 are set to “0”. Therefore, if MSAR1 is set to the above mentioned value, the start address of the block address area is set to address 110000H.

MAMR1 Register

Bit	7	6	5	4	3	2	1	0
Bit symbol	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	M1V15 to M1V9	M1V8
Specified value	0	0	0	0	0	0	0	1

M1V21 to M1V16 and M1V8 bits of the memory address mask register MAMR1 are set whether addresses A21 to A16 and A8 are compared or not. In register setting, “0” is “compare”, and “1” is “do not compare”. M1V15 to M1V9 bits determine whether addresses A15 to A9 are compared or not with bit 1. A23 and A22 are always compared.

When set as above, A23 to A9 are compared with the value that is set as the start addresses. Therefore, 512 bytes (addresses 110000H to 1101FFH) are set as block address area 1, and if it is compared with the addresses on the bus, the chip select signal CS1 is set to “low”.

The other block address area sizes are specified in the same way.

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

Similarly, A23 is always compared with block address areas 2 to 5. Whether A22 to A15 are compared or not is determined by the register.

Note 1: 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 2: If an address area other than $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$ is accessed, this area is regarded as $\overline{\text{CSEX}}$. Therefore, wait number and data bus width controls follow the setting of $\overline{\text{CSEX}}$ (BEXCSH, BEXCSL register).

(2) Connection memory specification

Setting the <BnOM1:0> bit of the control register (BnCSH) specifies the memory type that is connected with the block address areas. The interface signal is outputted according to the set memory as follows.

<BnOM1: 0> Bit (BnCSH Register)

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

Note 1: SDRAM should be set to block either 1 or 2.

Note 2: 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 setting the control register (BnCSH)<BnBUS1:0> as follows.

<BnBUS1:0> bit (BnCSH Register)

BnBUS 1	BnBUS 0	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

Note: SDRAM should be set to either "01" (16-bit bus) or "10" (32-bit bus).

This method of changing the data bus width depending on the accessing address is called "dynamic bus sizing". The part of the data bus to which the data is output depends on the data size, baus width and start address.

Note: Since there is a possibility of abnormal writing/reading of the data if two memories with different bus width are put in consecutive addresses, do not execute an access to 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 active.

(4) Wait control

The external bus cycle completes a wait of at least two states (100 ns at f_{SYS} = 20 MHz).

Setting the <BnWW2:0> and <BnWR2:0> of BnCSL specifies the number of waits in the write cycle and the read cycle. <BnWW2:0> is set using the same method as <BnWR2:0>.

<BnWW>/<BnWR> (BnCSL Register)

<BnWW2> <BnWR2>	<BnWW1> <BnWR1>	<BnWW0> <BnWR0>	Function
0	0	1	2 states (0 waits) access fixed mode
0	1	0	3 states (1 wait) access fixed mode (Default)
1	0	1	4 states (2 waits) access fixed mode
1	1	0	5 states (3 waits) access fixed mode
1	1	1	6 states (4 waits) access fixed mode
0	1	1	$\overline{\text{WAIT}}$ pin input mode
Others			(Reserved)

Note 1: For SDRAM, the above setting is ineffective. Refer to 3.16 SDRAM controller.

Note 2: For NAND flash, this setting is ineffective.
For RAM built-in LCDD, this setting is effective.

(i) Waits number fixed mode

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

(ii) $\overline{\text{WAIT}}$ pin input mode

This mode samples the $\overline{\text{WAIT}}$ input pins. In this mode, a wait is inserted continuously while the signal is active. The bus cycle is a minimum 2 states. The bus cycle is completed if the wait signal is non active ("High" level) at the second state. The bus cycle continues if the wait signal is active after 2 states or more.

(5) Recovery (Data hold) cycle control

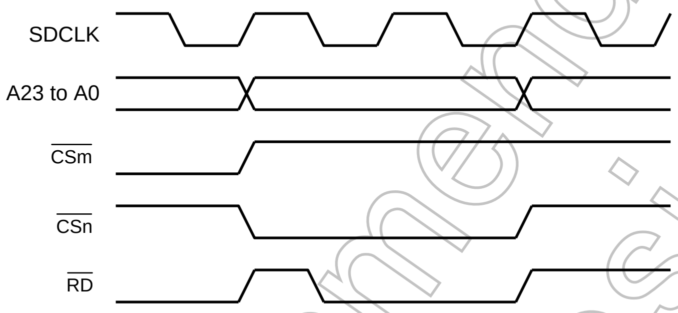
Some memory is defined by AC specification about data hold time by $\overline{\text{CE}}$ or $\overline{\text{OE}}$ for read cycle. Therefore, a data conflict problem may occur. To avoid this problem, 1-dummy cycle can be inserted after CSm-block access cycle by setting "1" to BmCSH<BmREC> register.

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

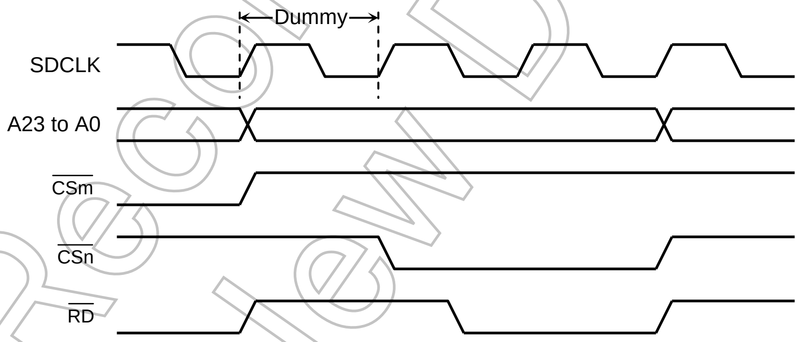
<BnREC> (BnCSH register)

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

- When no dummy cycle is inserted (0 waits)

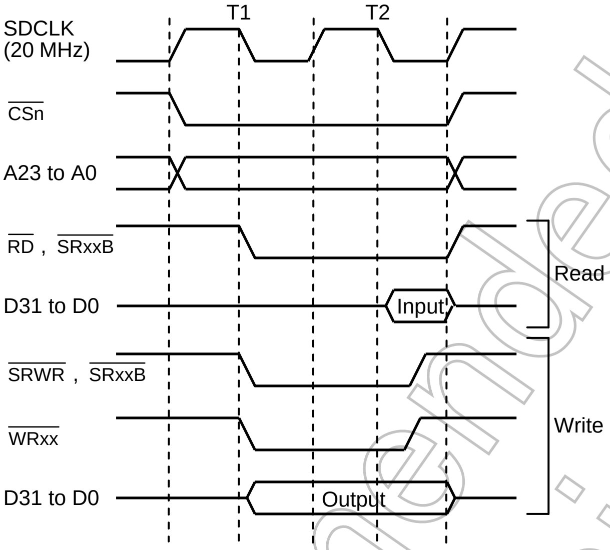


- When inserting a dummy cycle (0 waits)

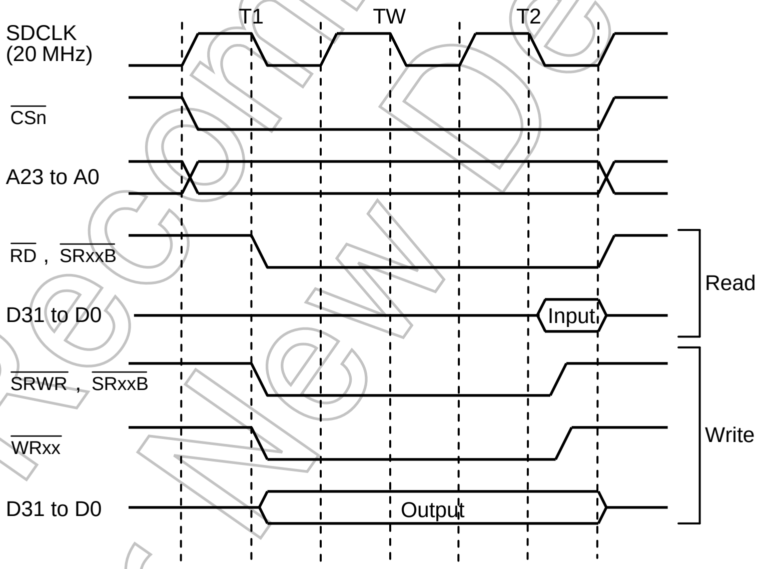


(6) Basic bus timing

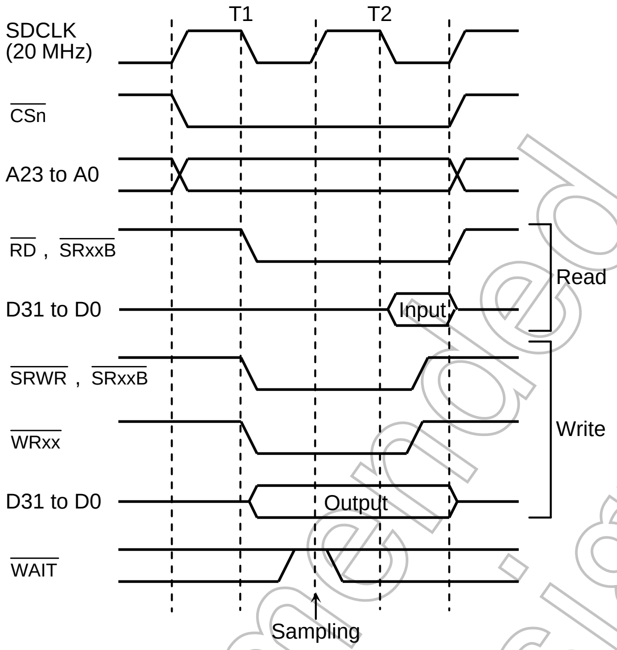
(a) External read/write cycle (0 waits)



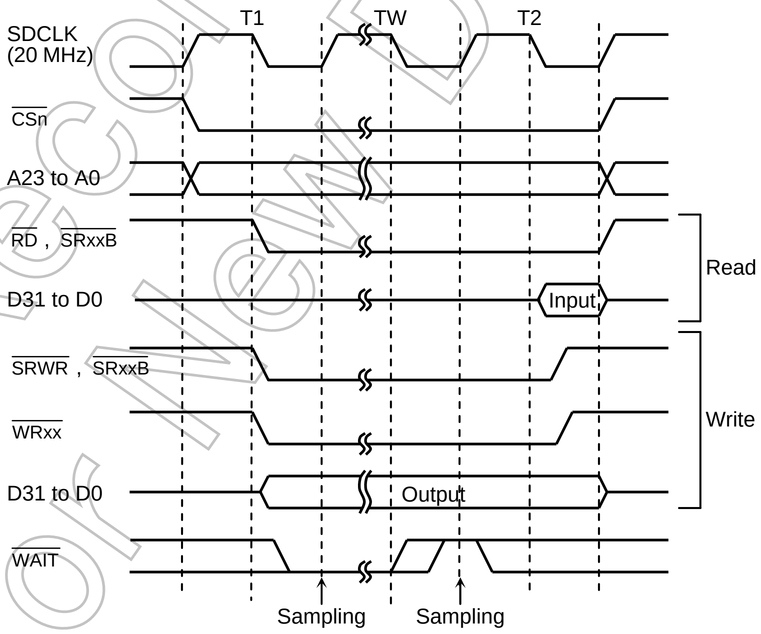
(b) External read/write cycle (1 wait)



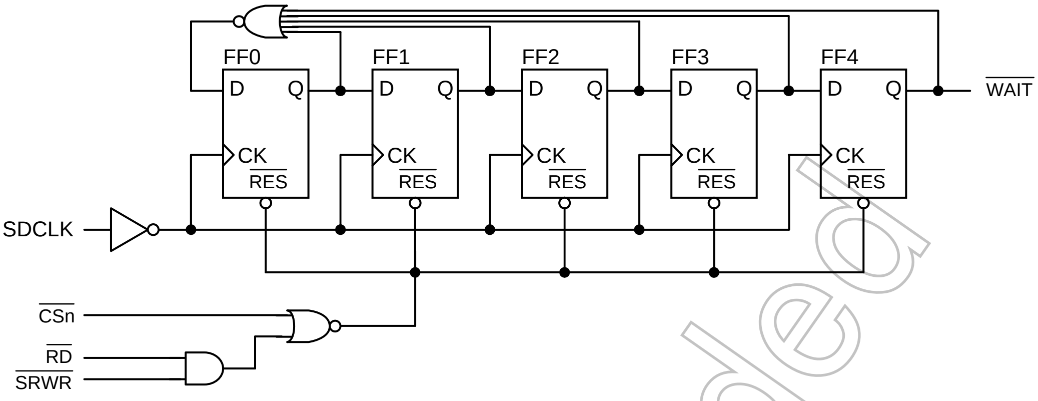
(c) External read/write cycle (0 waits at $\overline{\text{WAIT}}$ pin input mode)

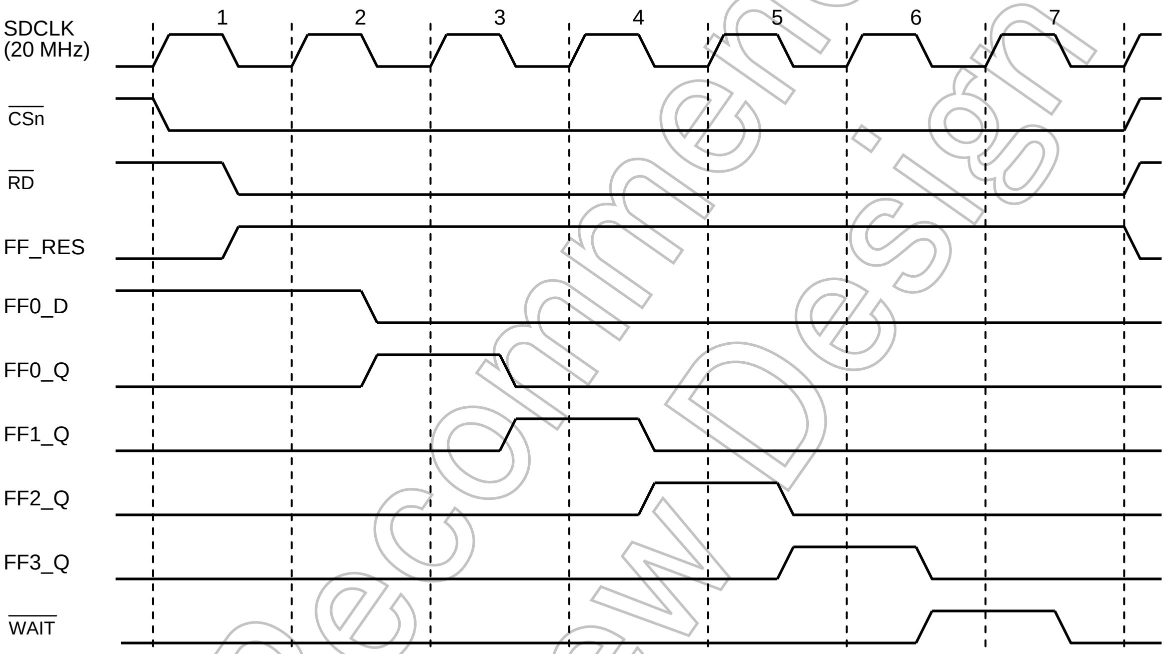


(d) External read/write cycle (n waits at $\overline{\text{WAIT}}$ pin input mode)



Example of wait input cycle (5 waits)





(7) Connecting external memory

Figure 3.6.1 shows an example of how to connect an external 16-bit SRAM and 16-bit NOR flash to the TMP92CH21.

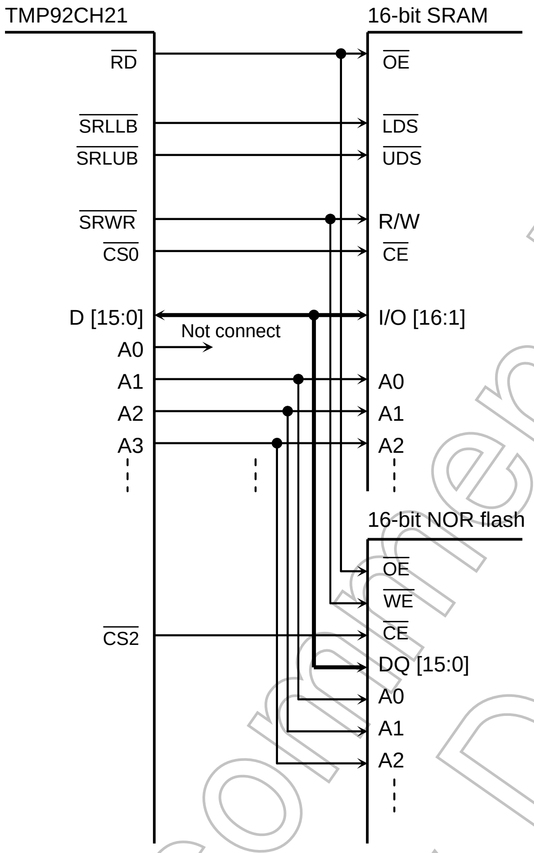


Figure 3.6.1 Example of External 16-Bit SRAM and NOR Flash Connection

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 registers

The TMP92CH21 supports ROM access of the page mode. ROM access of the page mode is specified only in 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 <OPWR1:0> of the PMEMCR register.

<OPWR1:0> (PMEMCR register)

<OPWR1>	<OPWR0>	Number of Cycle in a Page
0	0	1 state (n-1-1-1 mode) ($n \geq 2$)
0	1	2 state (n-2-2-2 mode) ($n \geq 3$)
1	0	3 state (n-3-3-3 mode) ($n \geq 4$)
1	1	(Reserved)

Note: Set the number of waits ("n") using the control register (BnCSL) in each block address area.

The page size (the number of bytes) of ROM in the CPU size is set by the <PR1:0> of the PMEMCR register. When data is read out up to the 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.

<PR1:0> Bit (PMEMCR register)

<PR1>	<PR0>	ROM Page Size
0	0	64 bytes
0	1	32 bytes
1	0	16 bytes (Default)
1	1	8 bytes

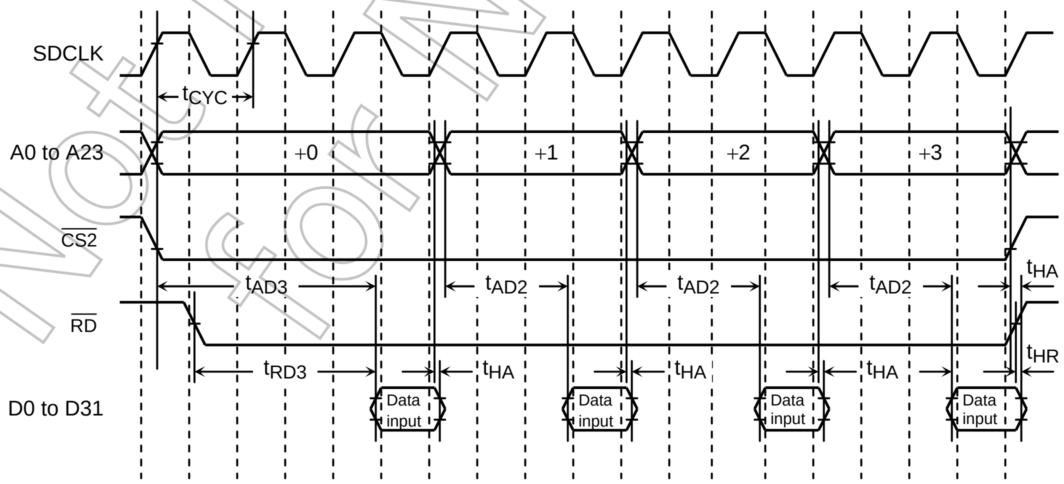


Figure 3.6.2 Page mode access Timing (8-byte example)

3.6.5 Internal Boot ROM Control

This section describes the built-in boot ROM.
For the specification of S/W in boot ROM, refer to 3.20 boot ROM sections.

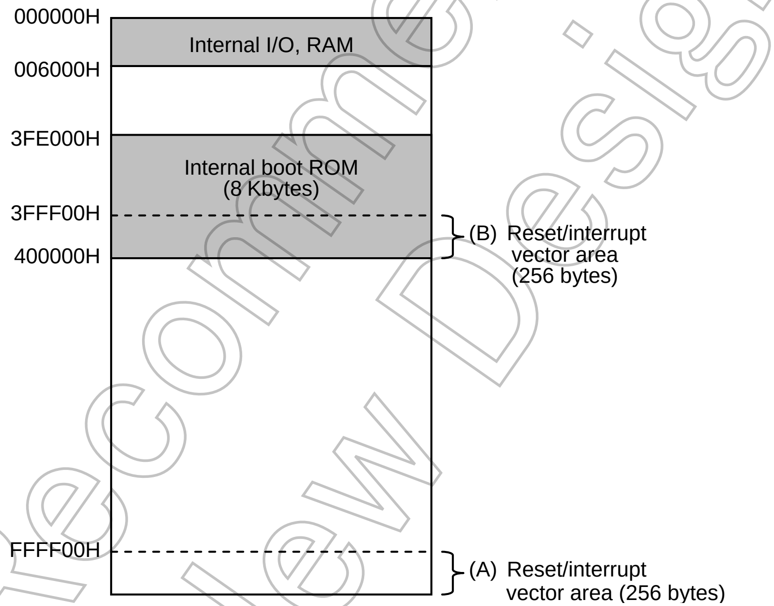
(1) BOOT mode

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 consists of an 8-Kbyte masked ROM and is assigned to address 3FE000H to 3FFFFFH.


000000H
Internal I/O, RAM
006000H
3FE000H
Internal boot ROM (8 Kbytes)
3FFF00H
400000H
(B) Reset/interrupt vector area (256 bytes)
FFFF00H
(A) Reset/interrupt vector area (256 bytes)

(3) Reset/interrupt address conversion circuit

The Reset/interrupt vector area is assigned to FFFF00H to FFFFEFH ((A) area) in the TLCS-900/H1.

Since the boot ROM is assigned to another area, a reset/interrupt vector address conversion circuit is provided.

In BOOT mode, the reset/interrupt vector area is assigned to 3FFF00H to 3FFFEFH ((B) area). Following boot sequence, the area can be changed to (A) area by setting BROMCR<VACE> to “0”. Therefore, (A) area can be used only for the application system program.

This <VACE> is initialized to “1” in BOOT mode. In any other starting mode, this register has no effect.

Note: As the last 16-byte area (FFFFF0H to FFFFFFH) is reserved for an emulator, this area is not changed by the <VACE> register.

(4) Bypassing 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 to address 3FE000H to 3FFFFFH cannot be accessed because the boot ROM is assigned.

To solve this, the internal boot ROM can be bypassed by setting BROMCR<ROMLESS> to "1".

This <ROMLESS> is initialized to "0" in BOOT mode. In any other starting mode, this register is initialized to "1".

If this register has been set to "1", writing "0" is prohibited.

		7	6	5	4	3	2	1	0
BROMCR (0167H)	Bit symbol							ROMLESS	VACE
	Read/Write							R/W	
	Reset State							0/1	1/0
	Function							Boot ROM 0: Use 1: Bypass	Vector address conversion 0: Disable 1: Enable

Note: Reset states differ depending on start modes.

3.6.6 Cautions

(1) Note on timing between $\overline{\text{CS}}$ and $\overline{\text{RD}}$

If the parasitic capacitance of the $\overline{\text{RD}}$ (Read signal) is greater than that of the $\overline{\text{CS}}$ (Chip select signal), it is possible that an unintended read cycle occurs due to a delay in the read signal. Such an unintended read cycle may cause a problem, as in the case of (a) in Figure 3.6.3.

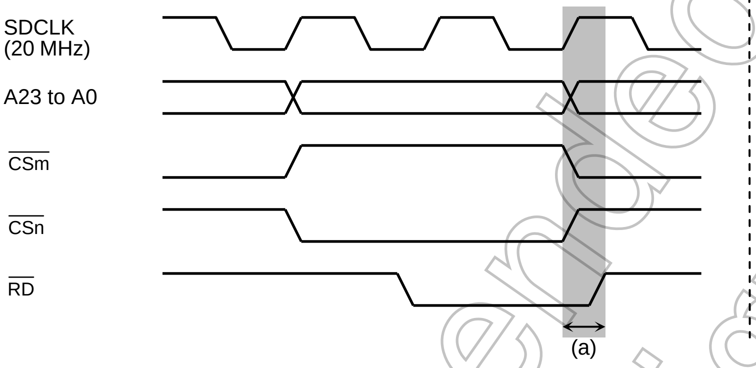


Figure 3.6.3 Read Signal Delay Read Cycle

Example: When using an externally connected NOR flash which uses 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 3.6.4, an unintended read cycle like the one shown in (b) may occur.

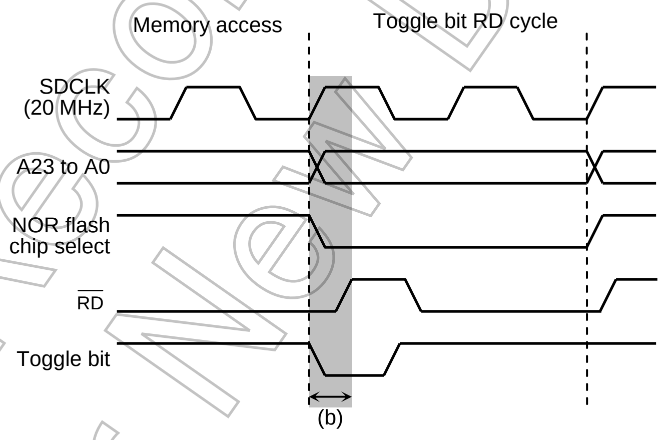


Figure 3.6.4 NOR Flash Toggle Bit Read Cycle

When the toggle bit is reversed by this unexpected read cycle, the CPU cannot read the toggle bit correctly since it always reads same value for the toggle bit. To avoid this phenomenon, data polling function control is recommended.

(2) Note on NAND flash area setting

Figure 3.6.5 shows a memory map for a NAND flash and RAM built-in LCD driver.

Ssince it is recommended that CS3 area be assigned to the address 000000H to 3FFFFFH, the following explanation is given.

In this case, the NAND flash and RAM built-in LCD driver overlap with CS3 area.

However, each access control circuit in the TMP92CH21 operates independently.

So, if a program on CS3 area accesses NAND flash, both $\overline{\text{CS3}}$ and NAND flash will be accessed at the same time and a problem such as data conflict will occur.

To avoid this phenomenon, it is recommended that CS0 area be assigned to the 32 Kbytes of address 000000H to 007FFFH as the $\overline{\text{CS0}}$ pin will not be needed.

Since CS0 has priority over CS3, only NAND flash will be accessed correctly by this setting.

Note: In this case, the 32 Kbytes of address 000000H to 007FFFH in CS3's memory cannot be used.

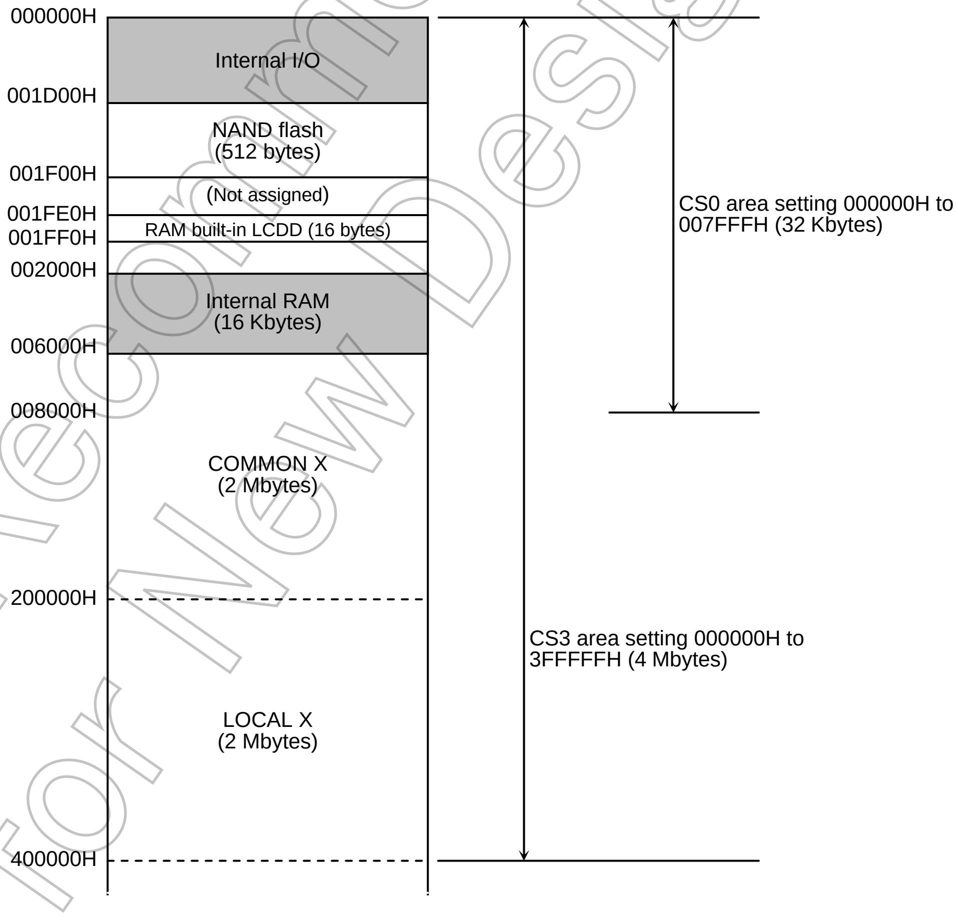


Figure 3.6.5 Recommended CS3 and CS0 Setting

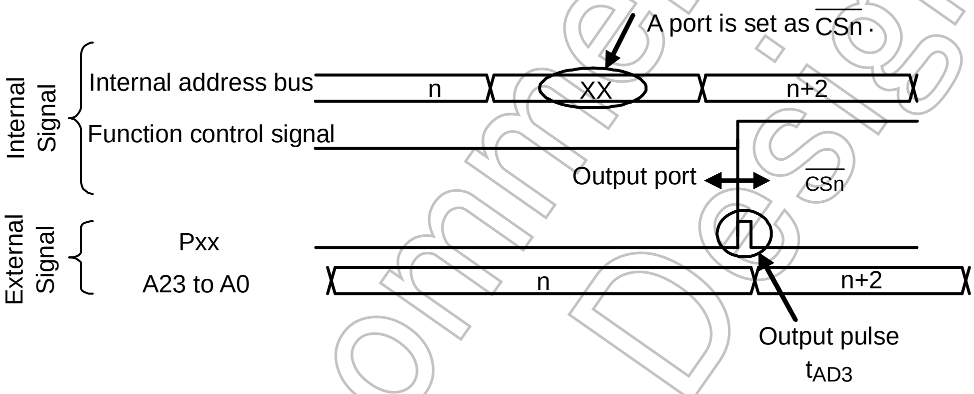
(3) The cautions at the time of the functional change of a $\overline{\text{CSn}}$.

A chip select signal output has the case of a combination terminal with a general-purpose port function. In this case, an output latch register and a function control register are initialized by the reset action, and an object terminal is initialized by the port output (“1” or “0”) by it.

Functional change

Although an object terminal is changed from a port to a chip select signal output by setting up a function control register (PnFC register), the short pulse for several ns may be outputted to the changing timing. Although it does not become especially a problem when using the usual memory, it may become a problem when using a special memory.

* XX is a function register address.(When an output port is initialized by “0”)

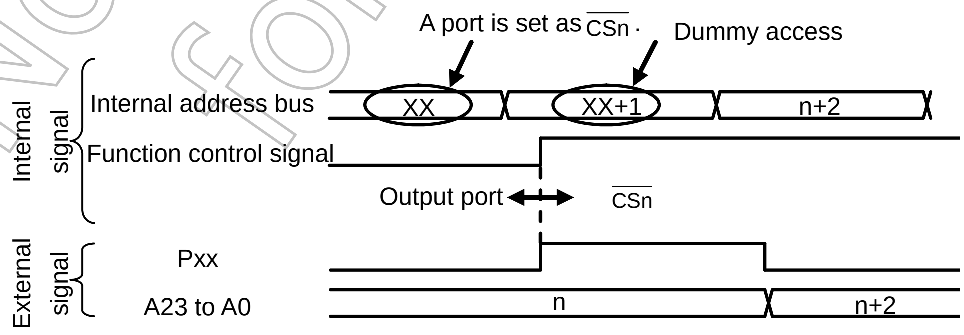


The measure by software

The countermeasures in S/W for avoiding this phenomenon are explained.

Since CS signal decodes the address of the access area and is generated, an unnecessary pulse is outputted by access to the object CS area immediately after setting it as a CSn function. Then, if internal area is accessed also immediately after setting a port as CS function, an unnecessary pulse will not output.

1. The ban on interruption under functional change (DI command)
2. A dummy command is added in order to carry out continuous internal access.
3. (Access to a functional change register is corresponded by 16-bit command. (LDW command))



3.7 8-Bit Timers (TMRA)

The TMP92CH21 features 4 built-in 8-bit timers (TMRA0-TMRA3).
These timers are paired into two 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 and 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 a five-byte SFR (special function register).

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

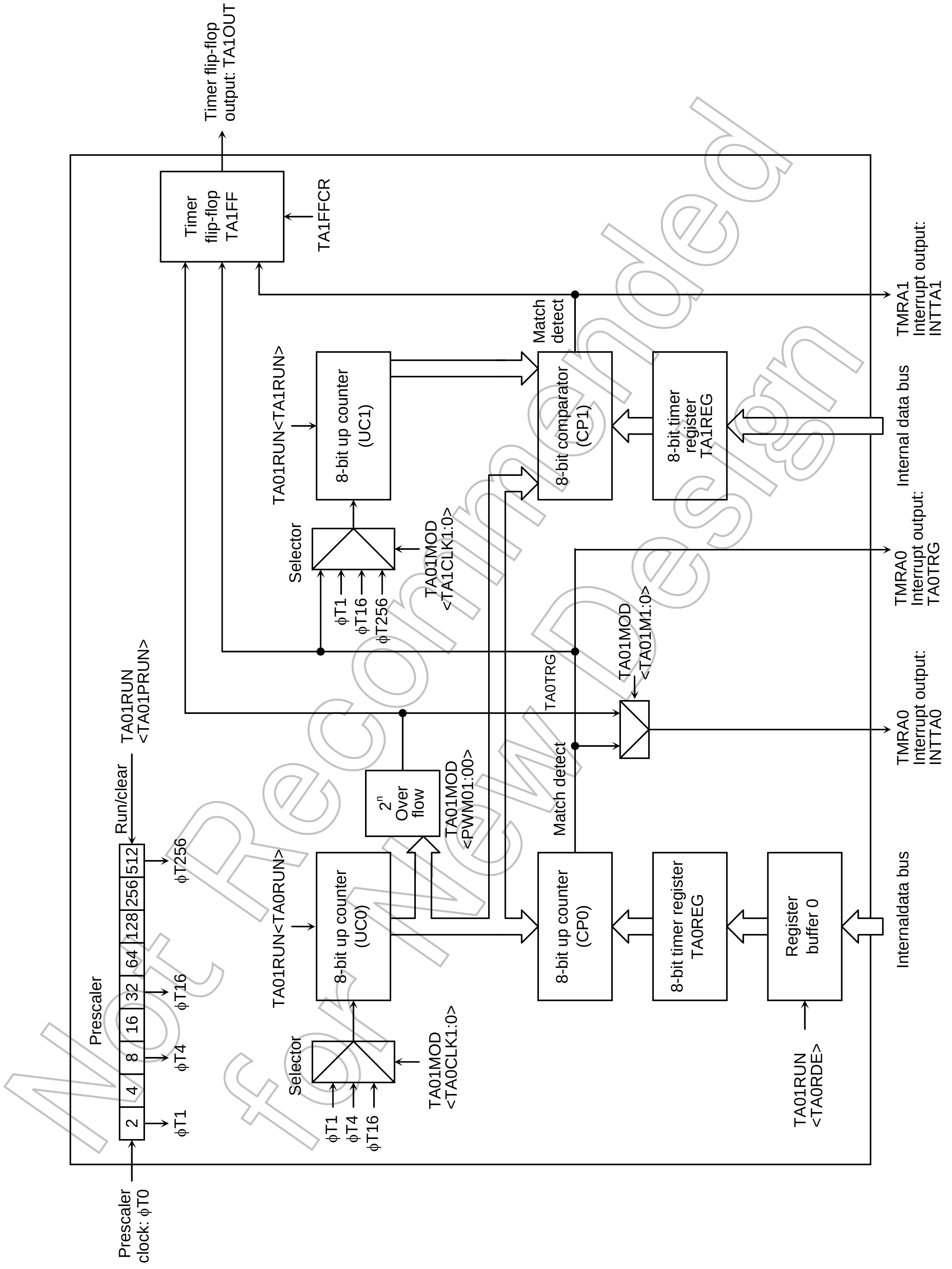


Figure 3.7.1 TMRA01 Block Diagram

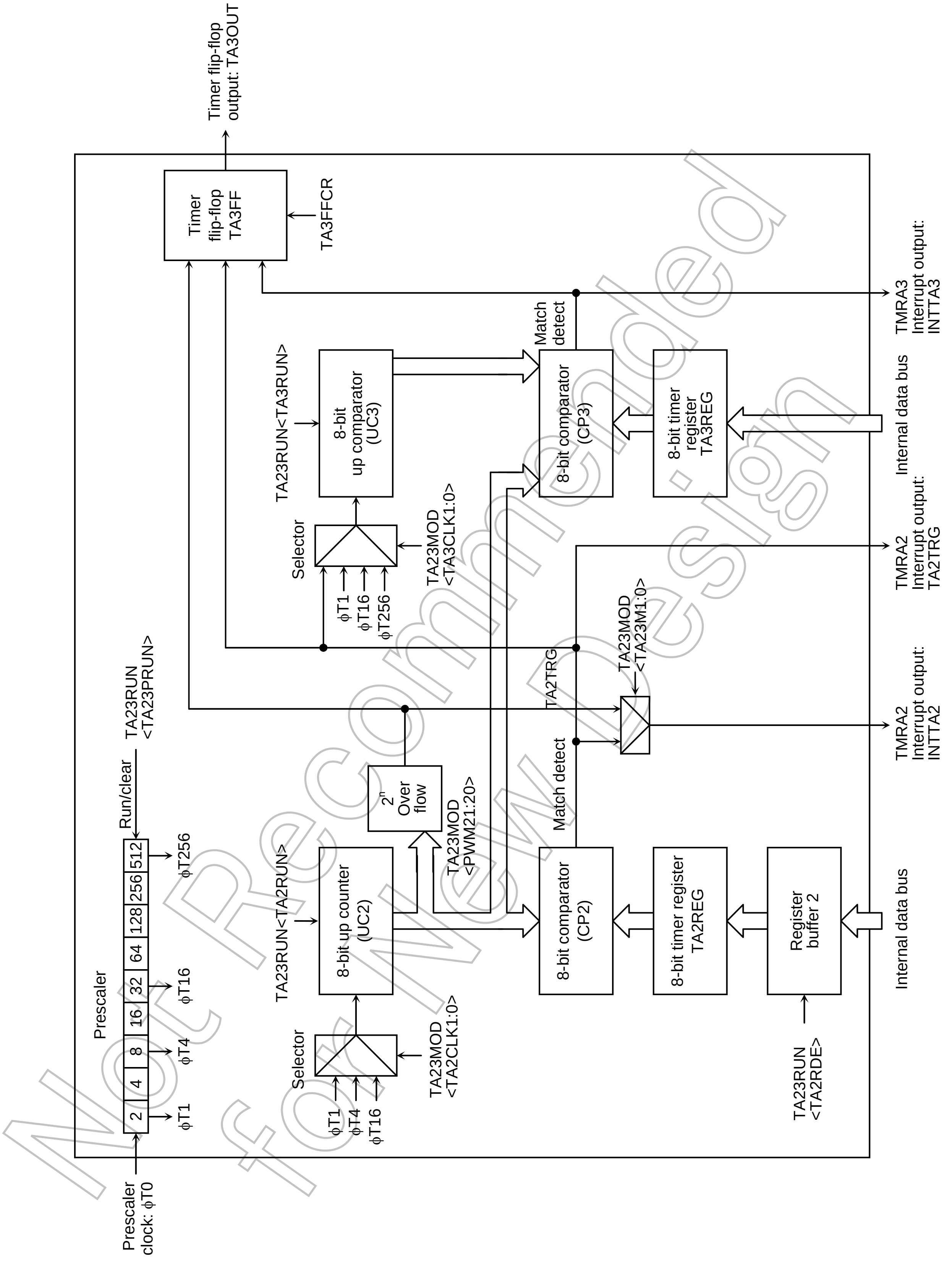


Figure 3.7.2 TMRA23 Block Diagram

3.7.2 Operation of Each Circuit

(1) Prescalers

A 9-bit prescaler generates the input clock to TMRA01.

The clock φT0 is divided into 8 by the CPU clock f_{SYS} and input to this prescaler.

The prescaler operation can be controlled using TA01RUN<TA01PRUN> in the timer control register. Setting <TA01PRUN> to “1” starts the count; setting <TA01PRUN> to “0” clears the prescaler to “0” and stops operation. Table 3.7.2 shows the various prescaler output clock resolutions.

Table 3.7.2 Prescaler Output Clock Resolution

System clock selection SYSCR1 <SYSCK>	Clock gear selection SYSCR1 <GEAR2:0>	–	Timer counter input clock TMRA prescaler TAxMOD<TAxCLK1:0>			
			φT1(1/2)	φT4(1/8)	φT16(1/32)	φT256(1/512)
1(fs)	–	1/8	fs/16	fs/64	fs/256	fs/4096
0(fc)	000 (1/1)		fc/16	fc/64	fc/256	fc/4096
	001 (1/2)		fc/32	fc/128	fc/512	fc/8192
	010 (1/4)		fc/64	fc/256	fc/1024	fc/16384
	011 (1/8)		fc/128	fc/512	fc/2048	fc/32768
	100 (1/16)		fc/256	fc/1024	fc/4096	fc/65536

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<TA01CLK1:0>.

The input clock for UC1 depends on the operation mode. In 16-bit timer mode, the overflow output from UC0 is used as the input clock. In any mode other than 16-bit timer mode, the input clock is selectable and can either be one of the internal clocks φT1, φT16 or φT256, or the comparator output (the match detection signal) from TMRA0.

For each interval timer the timer operation control register bits TA01RUN<TA0RUN> and TA01RUN<TA1RUN> can be used to stop and clear the up counters and to control their count. A reset clears both up counters, stopping the timers.

(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.

TA0REG has a double buffer structure, making a pair with the register buffer.

The setting of the bit TA01RUN<TA0RDE> determines whether TA0REG's double buffer structure is enabled or disabled. It is disabled if <TA0RDE> = "0" and enabled if <TA0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer to the timer register when a 2^n overflow occurs in PWM mode, or at the start of the PPG cycle in PPG mode. Hence the double buffer cannot be used in timer mode.

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 3.7.3 show the configuration of TA0REG.

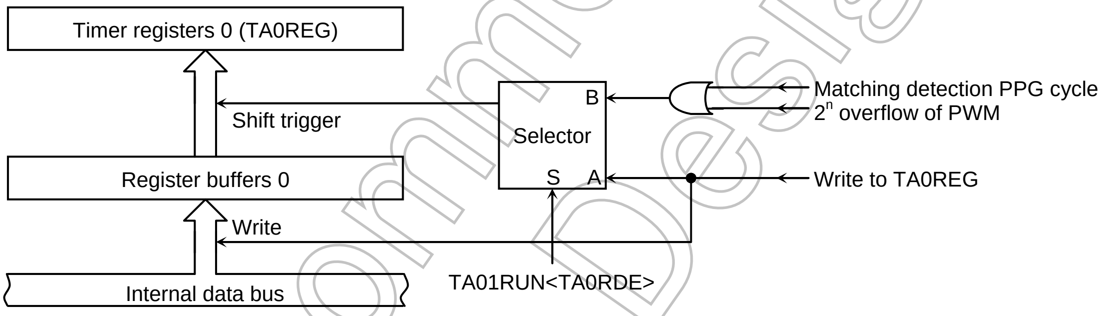


Figure 3.7.3 Configuration of TA0REG

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 “0” and an interrupt signal (INTTA0 or INTTA1) is generated. If timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detect signals (8-bit comparator output) of each interval timer.

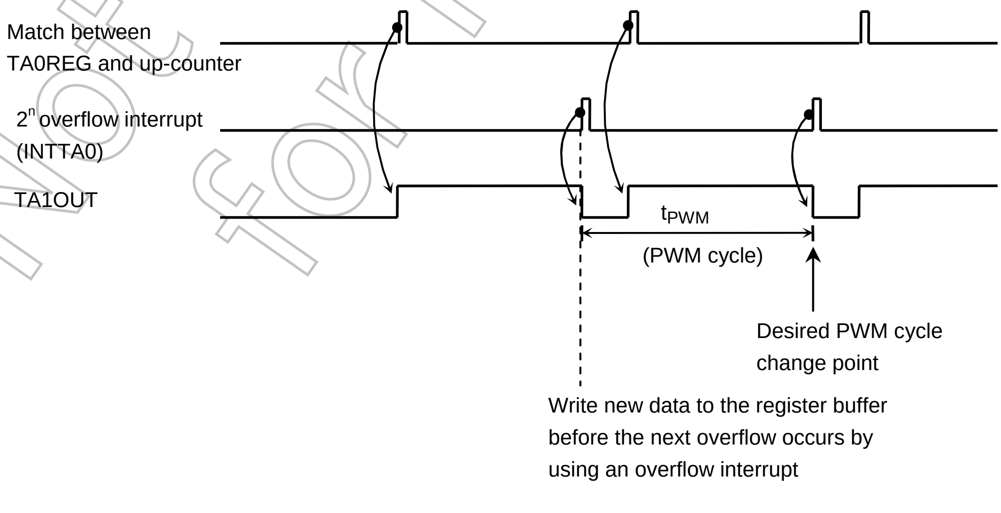
Whether inversion is enabled or disabled is determined by the setting of the bit TA1FFCR<TA1FFIE> in the timer flip-flops control register. A reset clears the value of TA1FF to “0”. Writing “01” or “10” to TA1FFCR<TA1FFC1:0> sets TA1FF to “0” or “1”. Writing “00” to these bits inverts the value of TA1FF (this is known as software inversion).

The TA1FF signal is output via the TA1OUT pin (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 and PCFC.

Note: When the double buffer is enabled for an 8-bit timer in PWM or PPG mode, caution is required as explained below.
If new data is written to the register buffer immediately before an overflow occurs by a match between the timer register value and the up-counter value, the timer flip-flop may output an unexpected value.
For this reason, make sure that in PWM mode new data is written to the register buffer by six cycles (f_{SYS} × 6) before the next overflow occurs by using an overflow interrupt.
When using PPG mode, make sure that new data is written to the register buffer by six cycles before the next cycle compare match occurs by using a cycle compare match interrupt.

Example when using PWM mode


Match between
TA0REG and up-counter
2^n overflow interrupt
(INTTA0)
TA1OUT
t_{PWM}
(PWM cycle)
Desired PWM cycle change point
Write new data to the register buffer before the next overflow occurs by using an overflow interrupt

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			
Reset State	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA01 prescaler	UP counter (UC1)	UP counter (UC0)
						0: Stop and clear 1: Run (Count up)		

TA0REG double buffer control

0	Disable
1	Enable

Timer run/stop control

0	Stop and clear
1	Run (Count up)

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			
Reset State	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA23 prescaler	UP counter (UC3)	UP counter (UC2)
						0: Stop and clear 1: Run (Count up)		

TA2REG double buffer control

0	Disable
1	Enable

Timer run/stop control

0	Stop and clear
1	Run (Count up)

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

Figure 3.7.4 TMRA01 Run Register and TMRA23 Run Register

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							
Reset State	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA0 00: 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 <TA01M1:0> ≠ 01	TA01MOD <TA01M1:0> = 01
00	Comparator output from TMRA0	Overflow output from TMRA0 (16-bit timer mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

PWM cycle selection

00	Reserved
01	$2^6 \times$ Source clock
10	$2^7 \times$ Source clock
11	$2^8 \times$ Source clock

TMRA01 operation mode selection

00	8-bit timers × 2ch
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA0) 8-bit timer (TMRA1)

Figure 3.7.5 TMRA Mode Register

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							
Reset State	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA3 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA2 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA2 source clock selection

00	(Reserved)
01	ϕT1 (Prescaler)
10	ϕT4 (Prescaler)
11	ϕT16 (Prescaler)

TMRA3 source clock selection

	TA23MOD <TA23M1:0> ≠ 01	TA23MOD <TA23M1:0> = 01
00	Comparator output from TMRA2	Overflow output from TMRA2 (16-bit timer mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

PWM cycle selection

00	Reserved
01	2^6 × Source clock
10	2^7 × Source clock
11	2^8 × Source clock

TMRA23 operation mode selection

00	8-bit timers × 2ch
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA2) 8-bit timer (TMRA3)

Figure 3.7.6 TMRA23 Mode Register

TMRA1 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA1FFCR (1105H)	Bit symbol					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
	Read/Write					R/W			
	Reset State					1	1	0	0
Read-modify-write instruction is 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 bits4 to 6 of TA1FFCR are undefined when read.

Figure 3.7.7 TMRA Flip-Flop Control Register

TMRA3 Flip-Flop Control Register

TA3FFCR
(110DH)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
Read/Write					R/W			
Reset State					1	1	0	0
Function					00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don’t care		TA3FF control for inversion 0: Disable 1: Enable	TA3FF inversion select 0: TMRA2 1: TMRA3

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 bits4 to 6 of TA3FFCR are undefined when read.

Figure 3.7.8 TMRA3 Flip-Flop Control Register

TMRA 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							

Note: Read-modify-write instruction is prohibited.

Figure 3.7.9 8-Bit Timers Register

3.7.4 Operation in Each Mode

(1) 8-bit timer mode

Both TMRA0 and TMRA1 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 TMRA1 (INTTA1), first stop TMRA1 then set the operation mode, input clock and a cycle to TA01MOD and TA1REG register, respectively. Then, enable the interrupt INTTA1 and start TMRA1 counting.

Example: To generate an INTTA1 interrupt every 40 μs at f_C = 40 MHz, set each register as follows:

		MSB							LSB	
		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 (=(16/fc)s at f_C = 40 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 TMRA1 counting.

X: Don't care, –: No change

Select the input clock using Table 3.7.3.

Table 3.7.3 Selecting Interrupt Interval and the Input Clock Using 8-Bit Timer

Input Clock	Interrupt Interval (at f_{SYS} = 20 MHz)	Resolution
ϕT1 (8/f_{SYS})	0.4 μs to 102.4 μs	0.4 μs
ϕT4 (32/f_{SYS})	1.6 μs to 409.6 μs	1.6 μs
ϕT16 (128/f_{SYS})	6.4 μs to 1.638 ms	6.4 μs
ϕT256 (2048/f_{SYS})	102.4 μs to 26.21 ms	102.4 μs

Note: The input clocks for TMRA0 and TMRA1 differ as follows:
TMRA0: Uses TMRA0 input (TA0IN) and can be selected from ϕT1, ϕT4 or ϕT16
TMRA1: Matches 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 f_C = 40 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 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 (=(16/fc)s at f_C = 40 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 TMRA1.
PCCR	←	–	–	–	–	–	–	–	1	Set PC0 to function as the TA1OUT pin.
PCFC	←	–	–	–	–	–	–	–	1	
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don't care, –: No change

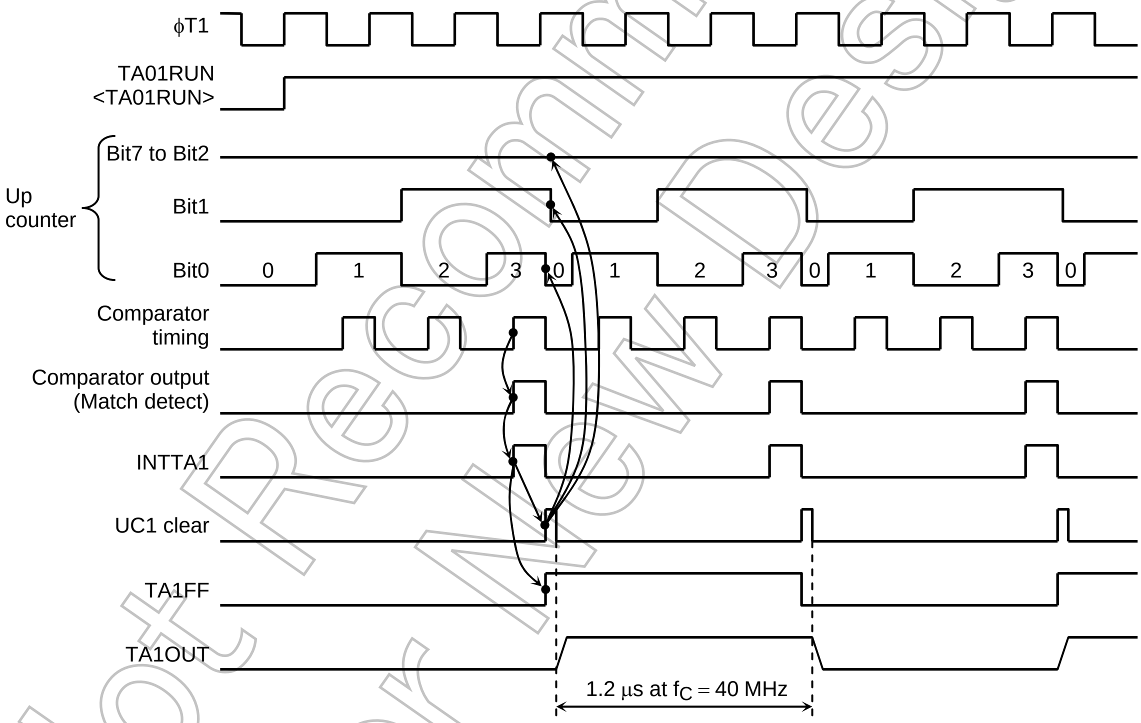


Figure 3.7.10 Square Wave Output Timing Chart (50 % Duty)

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.

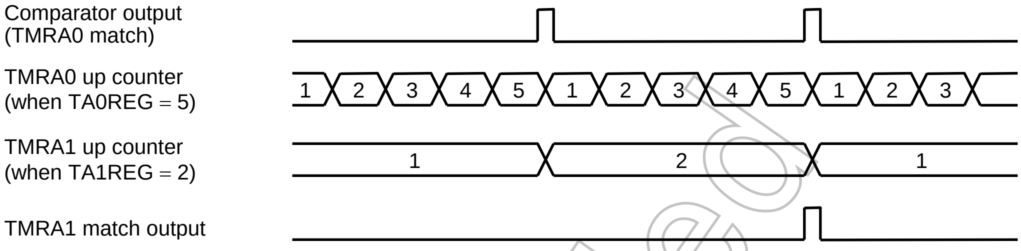


Figure 3.7.11 TMRA1 Count Up on Signal from TMRA0

(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<TA01M1:0> to “01”.

In 16-bit timer mode, the overflow output from TMRA0 is used as the input clock for TMRA1, regardless of the value set in TA01MOD<TA01CLK1:0>. Table 3.7.2 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 s at f_C = 40 MHz, set the timer registers TA0REG and TA1REG as follows:

If ϕT16 (=(256/fc)s at f_C = 40 MHz) is used as the input clock for counting, set the following value in the registers: 0.4 s ÷ =(256/fc)s = 62500 = F424H; e.g. 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 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

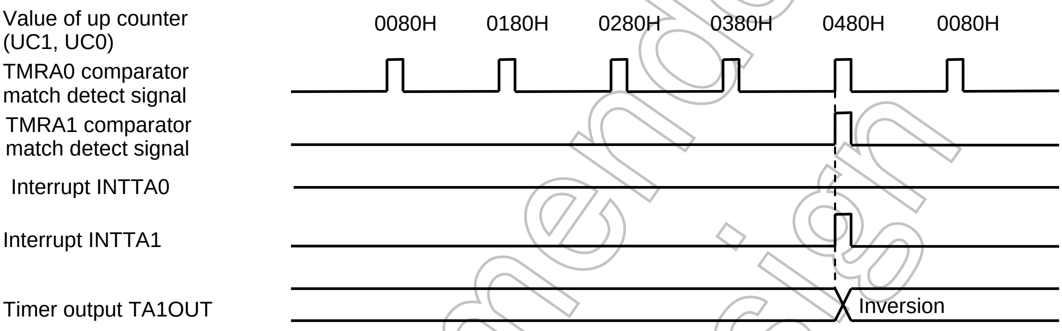

Value of up counter (UC1, UC0)
0080H
0180H
0280H
0380H
0480H
0080H
TMRA0 comparator match detect signal
TMRA1 comparator match detect signal
Interrupt INTTA0
Interrupt INTTA1
Timer output TA1OUT
Inversion

Figure 3.7.12 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).

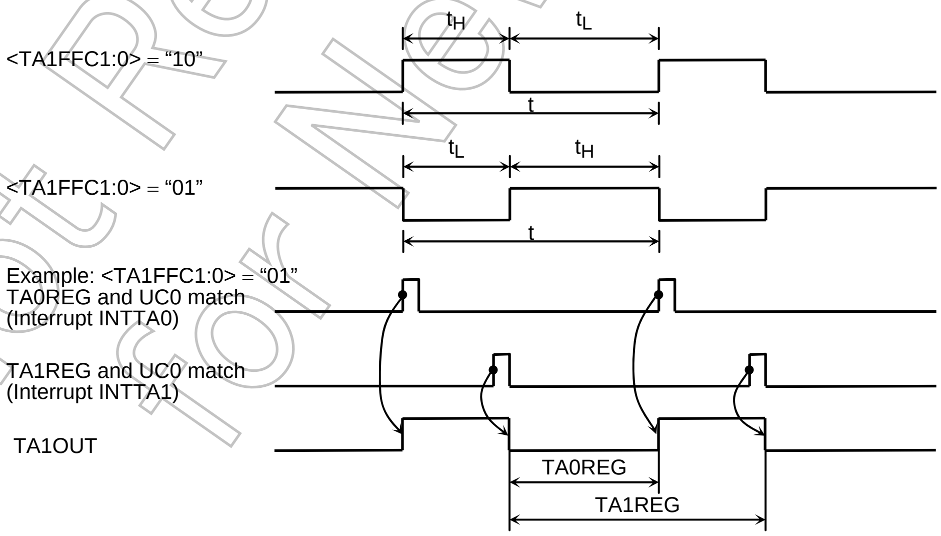


Figure 3.7.13 8-Bit PPG Output Waveforms

In this mode a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC0) matches the value in one of the timer registers TA0REG or TA1REG.

The value set in TA0REG must be smaller than the value set in TA1REG.

Although the up counter for TMRA1 (UC1) is not used in this mode, TA01RUN<TA1RUN> should be set to "1" so that UC1 is set for counting.

Figure 3.7.14 shows a block diagram representing this mode.

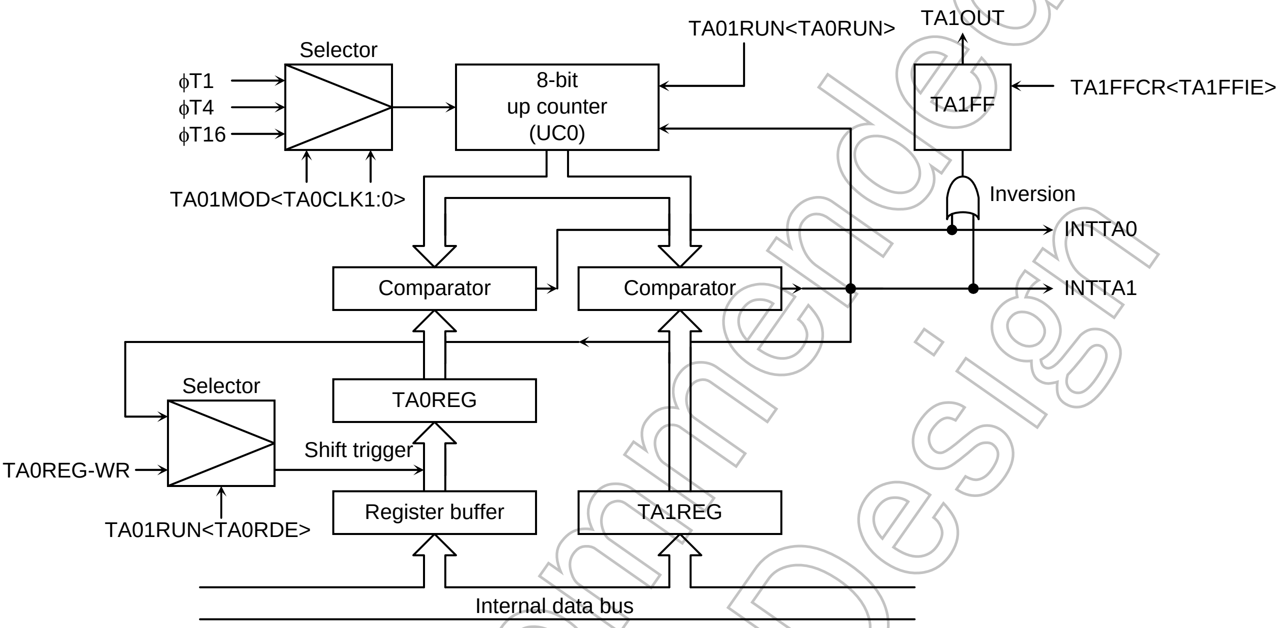


Figure 3.7.14 Block Diagram of 8-Bit PPG Output Mode

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).

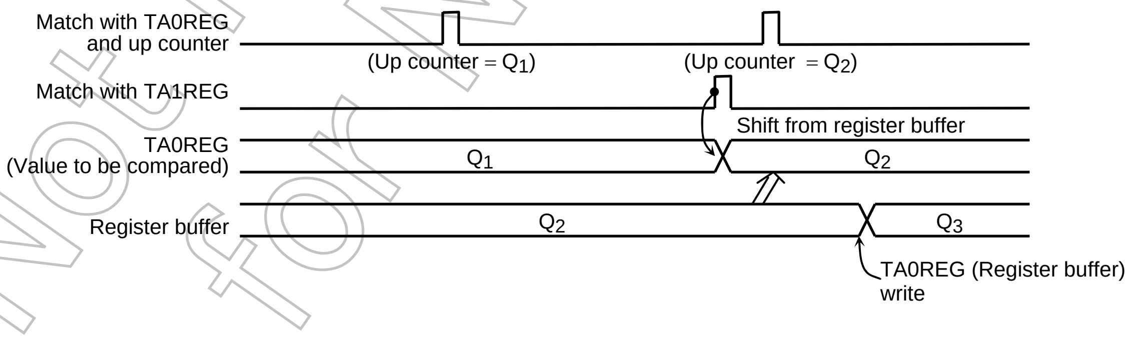
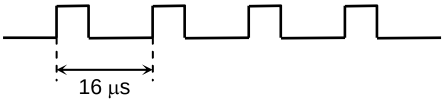

Match with TA0REG and up counter
(Up counter = Q1)
(Up counter = Q2)
Match with TA1REG
Shift from register buffer
TA0REG (Value to be compared)
Q1
Q2
Register buffer
Q2
Q3
TA0REG (Register buffer) write

Figure 3.7.15 Operation of Register Buffer

Example: To generate 1/4 duty 62.5 kHz pulses (at f_C = 40 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 (=(16/fc)s (at f_C = 40 MHz);

16 μs ÷ (16/fc)s = 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 ÷ (16/fc)s = 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.

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 counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01:00>). The up counter UC0 is cleared when 2^n counter overflow occurs. The following conditions must be satisfied before this PWM mode can be used.

Value set in TA0REG < value set for 2^n counter overflow

Value set in TA0REG ≠ 0

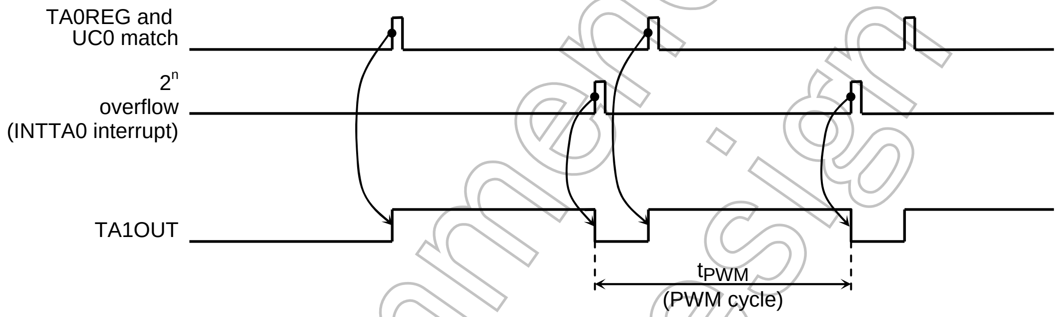


Figure 3.7.16 8-Bit PWM Waveforms

Figure 3.7.17 shows a block diagram representing this mode.

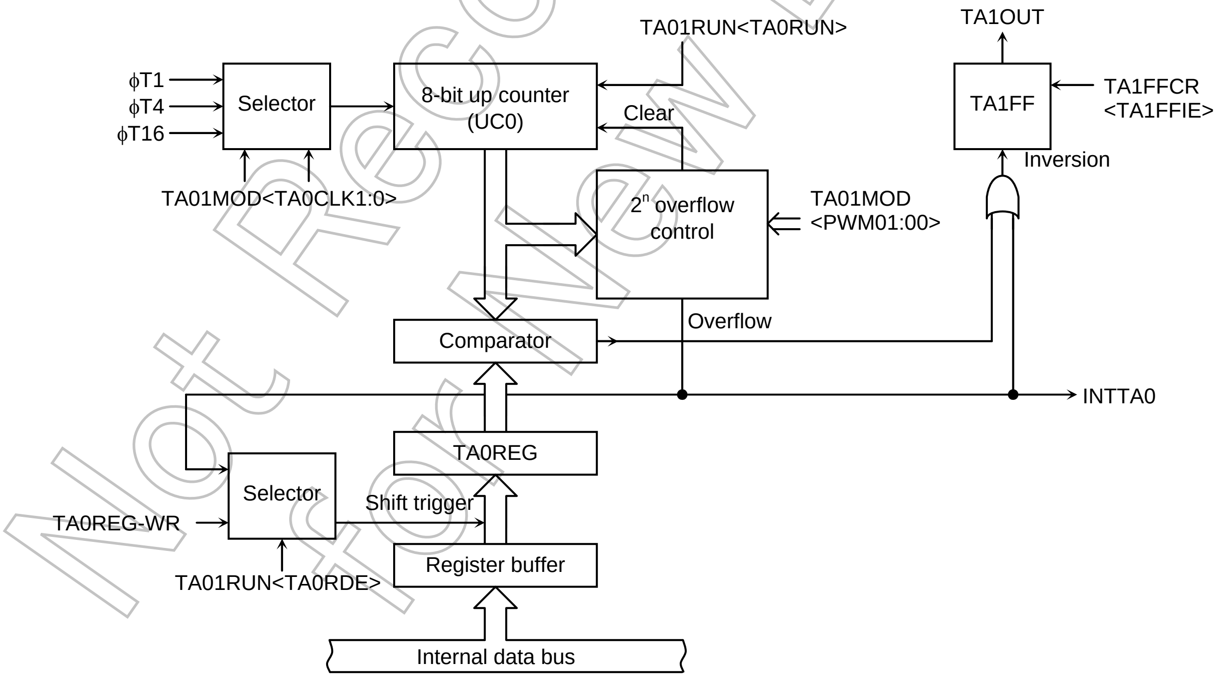


Figure 3.7.17 Block Diagram of 8-Bit PWM Mode

In this mode the value of the register buffer will be shifted into TA0REG if 2^n overflow is detected when the TA0REG double buffer is enabled.

Use of the double buffer facilitates the handling of low duty ratio waves.

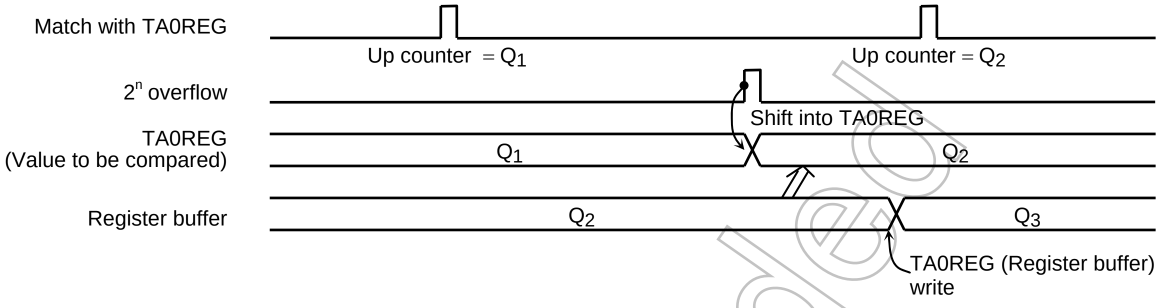
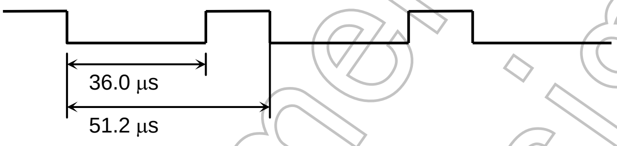


Figure 3.7.18 Register Buffer Operation

Example: To output the following PWM waves on the TA1OUT pin (at f_C = 40 MHz).



To achieve a 51.2-μs PWM cycle by setting φT1 (= (16/fc)s (@f_C = 40 MHz):
51.2 μs ÷ (16/fc)s = 128
2^n= 128
Therefore n should be set to 7.
Since the low level period is 36.0 μs when φT1 = (16/fc)s,
set the following value for TREG0:
36.0 μs ÷ (16/fc)s = 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) 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.

X: Don't care, –: No change

Table 3.7.4 PWM Cycle

System clock SYSCR0 <SYSCK>	Clock gear SYSCR1 <GEAR2:0>	–	PWM cycle TAxxMOD<PWMx1:0>								
			2^6 (x64)			2^7(x128)			2^8(x256)		
			TAxxMOD<TAxCLK1:0>			TAxxMOD<TAxCLK1:0>			TAxxMOD<TAxCLK1:0>		
			φT1(x2)	φT4(x8)	φT16(x32)	φT1(x2)	φT4(x8)	φT16(x32)	φT1(x2)	φT4(x8)	φT16(x32)
1(fs)	–	×8	1024/fs	4096/fs	16384/fs	2048/fs	8192/fs	32768/fs	4096/fs	16384/fs	65536/fs
0(fc)	000(x1)		1024/fc	4096/fc	16384/fc	2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc
	001(x2)		2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc
	010(x4)		4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc
	011(x8)		8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc
	100(x16)		16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc	65536/fc	262144/fc	1048576/fc

(5) Settings for each mode

Table 3.7.5 shows the SFR settings for each mode.

Table 3.7.5 Timer Mode Setting Registers

Register name	TA01MOD				TA1FFCR
<Bit Symbol>	<TA01M1: 0>	<PWM01: 00>	<TA1CLK1: 0>	<TA0CLK1: 0>	<TA1FFIS>
Function	Timer Mode	PWM Cycle	Upper Timer Input Clock	Lower Timer Input Clock	Timer F/F Invert Signal Select
8-bit timer × 2 channels	00	–	Lower timer match, φT1, φT16, φT256 (00, 01, 10, 11)	External clock, φT1, φT4, φT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
16-bit timer mode	01	–	–	External clock, φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit PPG × 1 channel	10	–	–	External clock, φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit PWM × 1 channel	11	2^6, 2^7, 2^8 (01, 10, 11)	–	External clock, φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit timer × 1 channel	11	–	φT1, φT16, φT256 (01, 10, 11)	–	Output disabled

–: Don't care

3.8 External Memory Extension Function (MMU)

By providing 3 local areas, the MMU function allows for the expansion of the program/data area up to 512 Mbytes.

The recommended address memory map is shown in Figure 3.8.1 and Figure 3.8.2.

However, when the memory used is less than 16 Mbytes, it is not necessary to set the MMU register. In this case, please refer to the Memory Controller section.

An area which can be set as a bank is called a local area. Since the address for local areas is fixed, it cannot be changed.

It is not possible for a program to branch between different banks of the same local area.

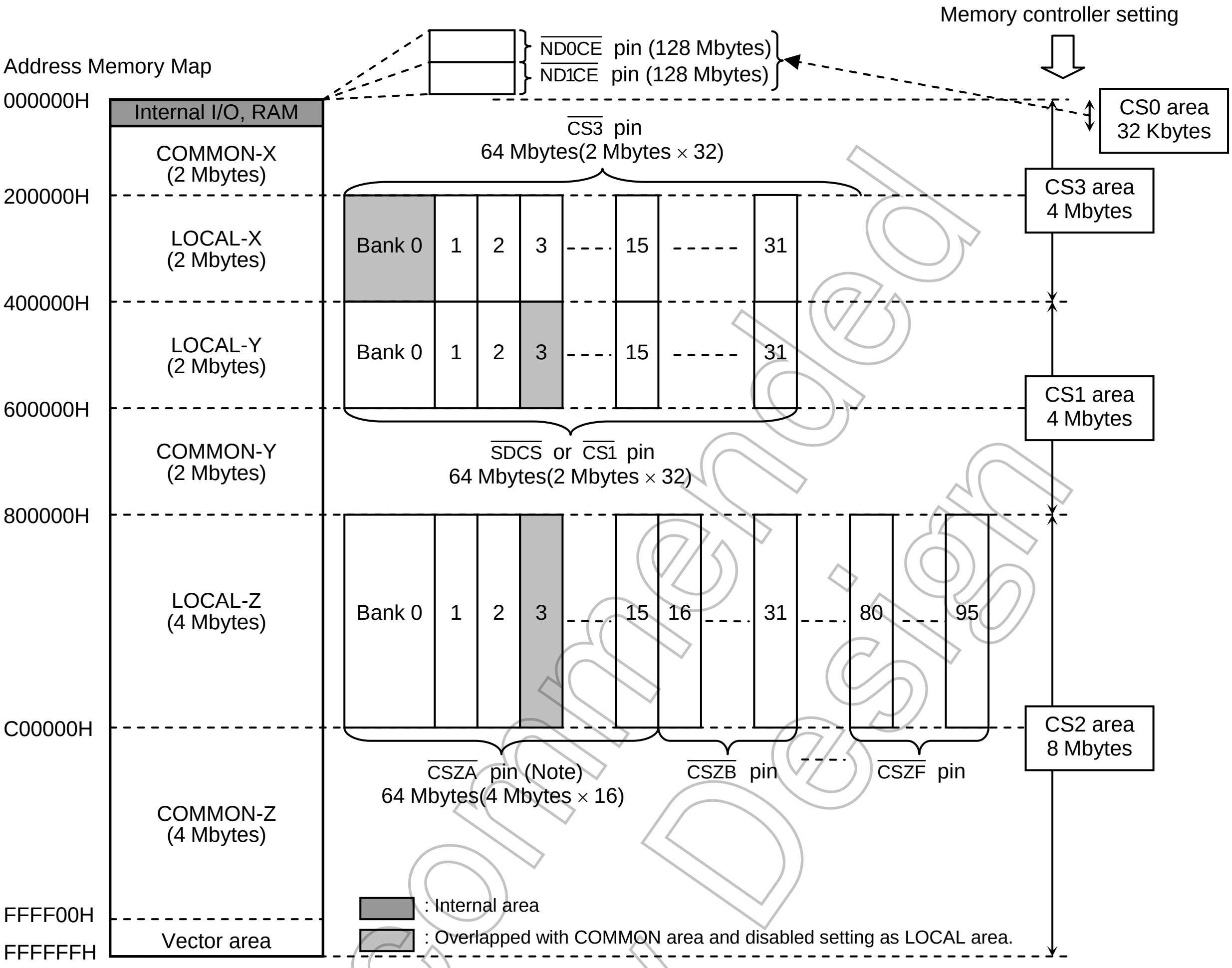
The TMP92CH21 has the following external pins for memory LSI connection.

Address bus: EA25, EA24 and A23 to A0
Chip select: $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, $\overline{\text{CSZA}}$ to $\overline{\text{CSZF}}$, $\overline{\text{SDCS}}$ $\overline{\text{ND0CE}}$ and $\overline{\text{ND1CE}}$
Data bus: D31 to D0

3.8.1 Recommended Memory Map

Figure 3.8.1 shows one recommended address memory map. This is for maximum expanded memory size and for a system in which an internal boot ROM with NAND flash is not needed.

Figure 3.8.3 also shows a recommended address memory map for a simple memory system such as an internal boot ROM with NAND flash and SDRAM.



Note: CSZA is a chip select for not only bank 0 to 15 of LOCAL-Z but also COMMON-Z.

Figure 3.8.1 Recommended Memory Map for Maximum Specification (Logical address)

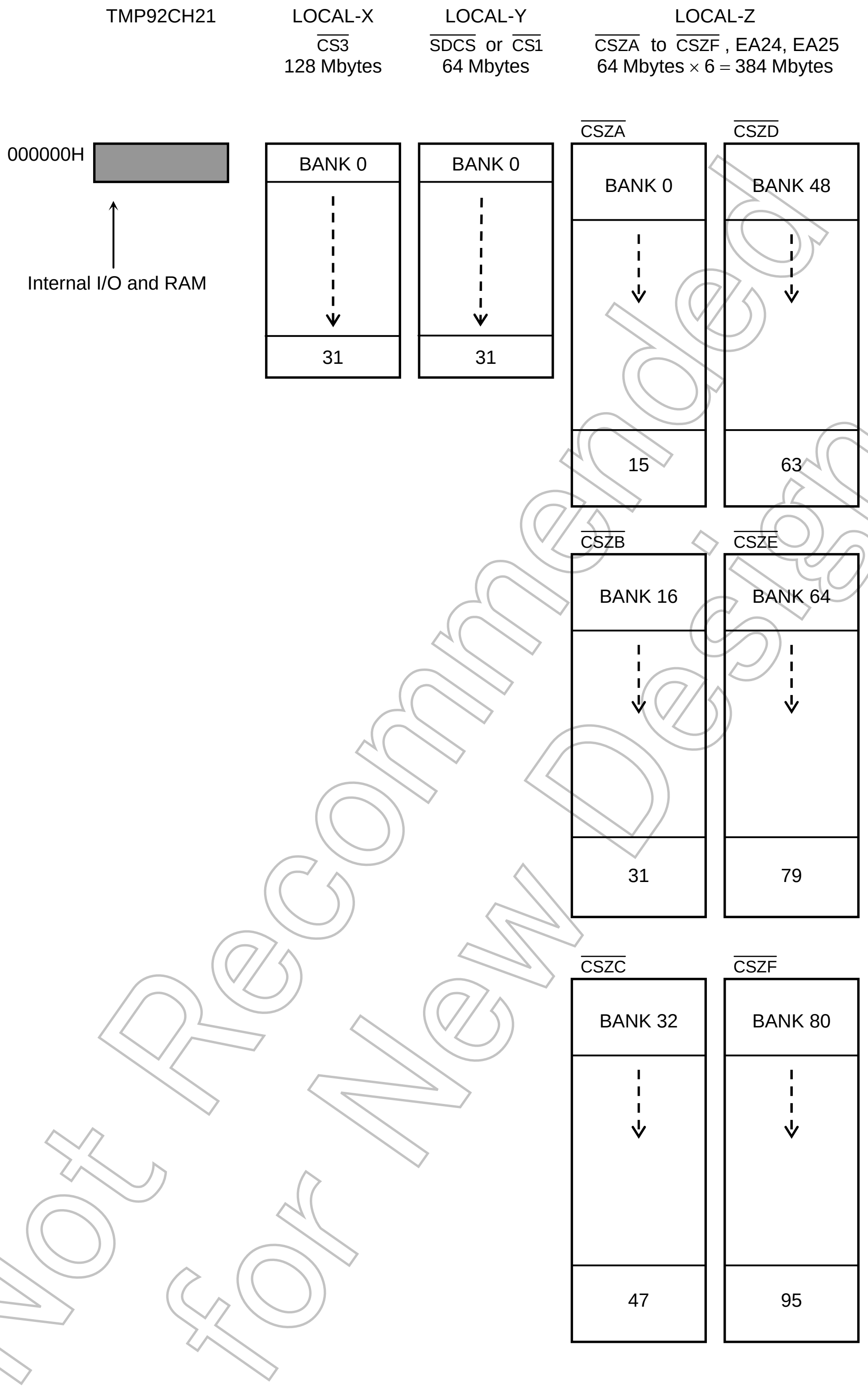


Figure 3.8.2 Recommended Memory Map for Maximum Specification (Physical address)

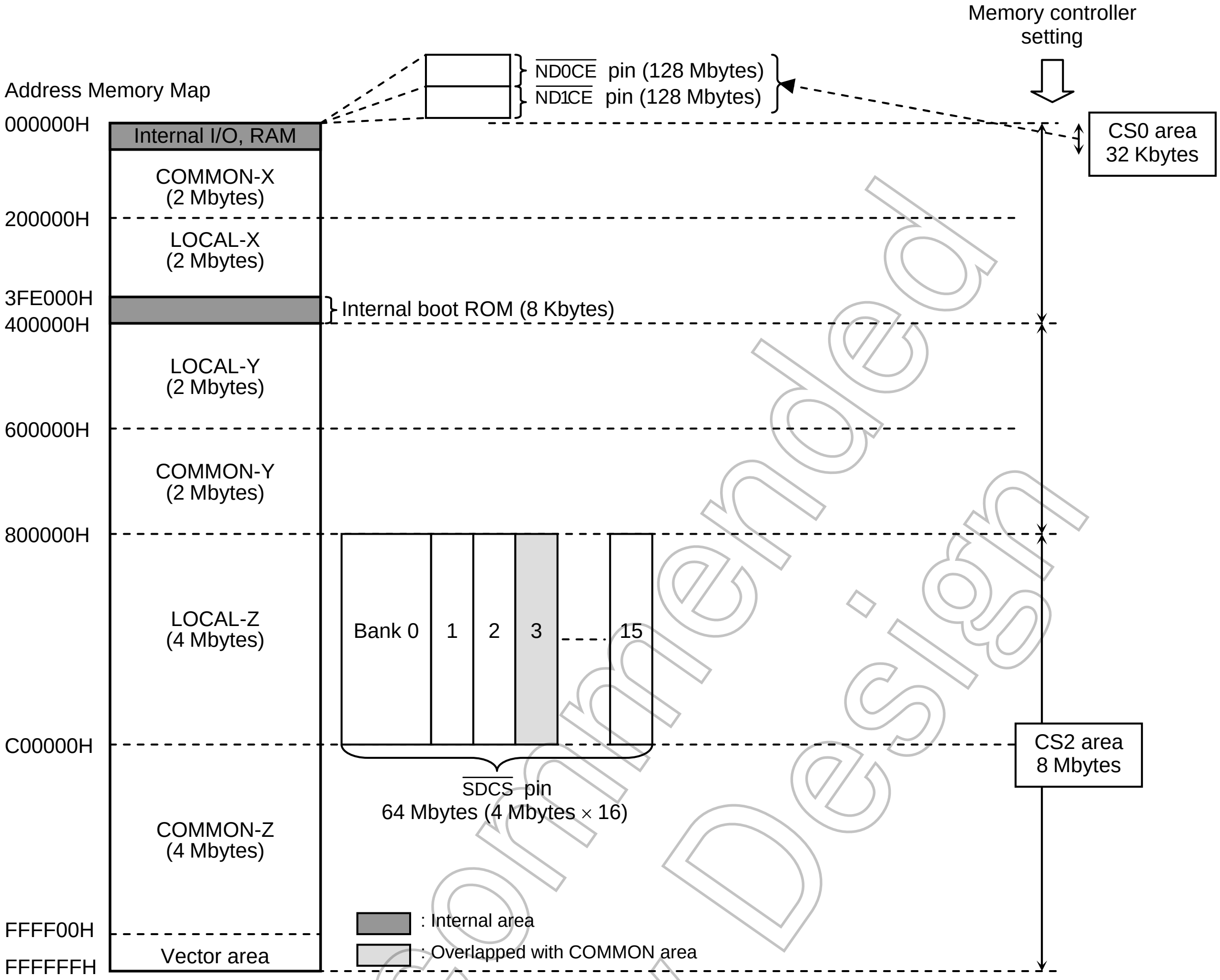


Figure 3.8.3 Recommended Memory Map for Simple System (Logical address)

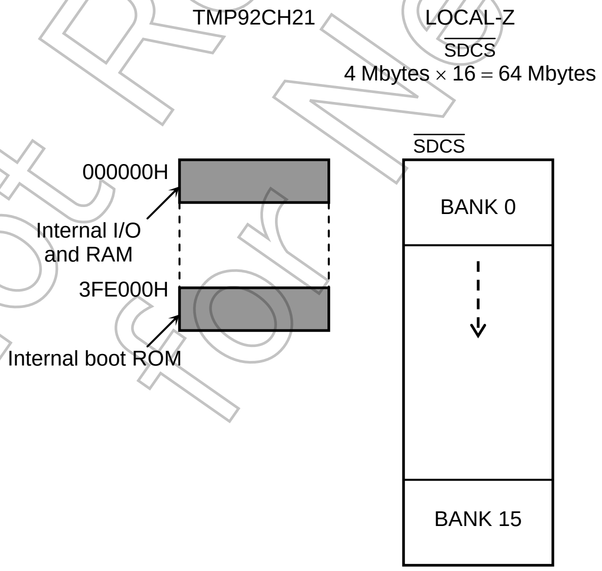


Figure 3.8.4 Recommended Memory Map for Simple System (Physical address)

3.8.2 Control Registers

There are 12 MMU registers, covering 4 functions (program, data read, data write and LCDC display data), in each of 3 local areas (Local-X, Y and Z), providing easy data access.

(Instructions for use)

First, set the enable register and bank number for each LOCAL register.

The relevant pin and memory settings should then be set to the ports and memory controller.

When the CPU or LCDC outputs a local area logical address, the MMU converts and outputs this to the physical address according to the bank number. The physical address bus is output to the external address bus pin, thereby enabling access to external memory.

Note 1: Since the common area cannot be used as local area, do not set a bank number to LOCAL register which overlaps with the common area.

Note 2: Changing program BANK number (LOCALPX, Y or Z) is disabled in the LOCAL area. The program bank setting for each local area must be changed in the common area. (But bank setting of read data, write data and data for LCD display can be changed in the local area.)

Note 3: After data bank number register (LOCALRn, LOCALWn or LOCALLn; where "n" means X, Y or Z) is set by an instruction, do not access its memory by the following instruction because several clocks are required for effective MMU setting. For this reason, insert between them a dummy instruction which accesses SFR or another memory, as in the 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 BANK 1 of LOCAL-X area.
```

Note 4: When LOCAL-Z area is used, chip select signal $\overline{\text{CSZA}}$ should be assigned to P82 pin.

In this case, $\overline{\text{CSZA}}$ works as chip select signal for not only BANK 0 to 15 but also COMMON-Z.

The following setting after reset is required before setting Port82.

```
ld      (localpz), 80H          ;  LOCAL-Z bank enable for program
ld      (localrz), 80H          ;  LOCAL-Z bank enable for data read
ld      (localwz), 80H          ;  LOCAL-Z bank enable for data write          (*1)
ld      (locallz), 80H          ;  LOCAL-Z bank enable for LCD display memory  (*2)
ld      (p8fc),  – – – – – 0 – – B  ;  Set P82 pin to CSZA output
ld      (p8fc2), – – – – – 1 – – B  ;
```

(*1) If COMMON-Z area is not used as data write memory, this setting is not required.

(*2) If COMMON-Z area is not used as LCD display memory, this setting is not required.

(1) Program bank register

The bank number used as program memory is set to these registers. It is not possible to change program bank number in the same local area.

LOCAL-X Register for Program

LOCALPX
(01D0H)

	7	6	5	4	3	2	1	0
Bit symbol	LXE			X4	X3	X2	X1	X0
Read/Write	R/W			R/W				
Reset State	0			0	0	0	0	0
Function	Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				

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				
Reset State	0			0	0	0	0	0
Function	Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				

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						
Reset State	0	0	0	0	0	0	0	0
Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						

(2) LCD Display bank register

The bank number used as LCD display memory is set to these registers. Since the bank registers for CPU and LCDC are prepared independently, the bank number for CPU (Program, Read data or Write data) can be changed during LCD display.

LOCAL-X Register for LCDC Display Data

		7	6	5	4	3	2	1	0
LOCALLX (01D4H)	Bit symbol	LXE			X4	X3	X2	X1	X0
	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
	Function	Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				

LOCAL-Y Register for LCDC Display Data

		7	6	5	4	3	2	1	0
LOCALLY (01D5H)	Bit symbol	LYE			Y4	Y3	Y2	Y1	Y0
	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
	Function	Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				

LOCAL-Z Register for LCDC Display Data

		7	6	5	4	3	2	1	0
LOCALLZ (01D7H)	Bit symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W	R/W						
	Reset State	0	0	0	0	0	0	0	0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						

(3) Read data bank register

The bank register number used as read data memory is set to these registers. The following is an example where the read data bank register of LOCAL-X is set to "1". When "ld wa, (xix)" instruction is executed, the bank becomes effective only at the read cycle for xix address.

(Example)

```
        ld      xix, 200000h       ;
        ld      (localrx), 81h     ; Set Read data bank.
        ld      wa, (localrx)      ; <-- Insert dummy instruction which accesses SFR
        ld      wa, (xix)          ; Read bank1 of LOCAL-X area
```

LOCAL-X Register for Read Data

		7	6	5	4	3	2	1	0
LOCALRX (01D8H)	Bit symbol	LXE			X4	X3	X2	X1	X0
	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
	Function	Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				

LOCAL-Y Register for Read Data

		7	6	5	4	3	2	1	0
LOCALRY (01D9H)	Bit symbol	LYE			Y4	Y3	Y2	Y1	Y0
	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
	Function	Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				

LOCAL-Z Register for Read Data

		7	6	5	4	3	2	1	0
LOCALRZ (01DBH)	Bit symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W	R/W						
	Reset State	0	0	0	0	0	0	0	0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						

(4) Write data bank register

The bank number used as write data memory is set to these registers. The following is an example where the data bank register of LOCAL-X is set to "1". When "ld (xix), wa" instruction is executed, the bank becomes effective only at the write cycle for xix address.

(Example)

```
            ld        xix, 200000h        ;
            ld        (localx), 81h       ; Set write data bank.
            ld        wa, (localwx)       ; <--Insert dummy instruction which accesses
SFR
            ld        wa, (xix)           ; Write to bank 1 of LOCAL-X area
```

LOCAL-X Register for Write Data

		7	6	5	4	3	2	1	0
LOCALWX (01DCH)	Bit symbol	LXE			X4	X3	X2	X1	X0
	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
	Function	Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				

LOCAL-Y Register for Write Data

		7	6	5	4	3	2	1	0
LOCALWY (01DDH)	Bit symbol	LYE			Y4	Y3	Y2	Y1	Y0
	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
	Function	Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				

LOCAL-Z Register for Write Data

		7	6	5	4	3	2	1	0
LOCALWZ (01DFH)	Bit symbol	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W	R/W						
	Reset State	0	0	0	0	0	0	0	0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						

3.8.3 Setting Example

Below is a setting example.

No.	Used as	Memory	Setting	MMU Area	Logical Address	Physical Address
(a)	Main routine	NOR flash (16 Mbytes, 1 pcs)	$\overline{\text{CSZA}}$, 32 bits, 1 wait	COMMON-Z	C00000H to FFFFFFH	
(b)	Character ROM			Bank 0 in LOCAL-Z	800000H to BFFFFFH	000000H to 3FFFFFH
(c)	Sub routine	SRAM (16 Mbytes, 1 pcs)	$\overline{\text{CS1}}$, 16 bits, 0 waits	Bank 0 in LOCAL-Y	400000H to 5FFFFFH	000000H to 1FFFFFH
(d)	LCD display RAM			Bank 1 in LOCAL-Y		200000H to 3FFFFFH
(e)	Stack RAM	Internal RAM (16 Kbytes)	– (32 bits, 1 clock)	Bank 2 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/8 Mbytes
C000xxH	←	3	ldw	(b2csl), C222H	; CS2 32-bit ROM, 1 wait
		4	ldw	(mamr1), 40FFH	; CS1 400000-7FFFFF/4 Mbytes
		5	ldw	(b1csl), 8111H	; CS1 16-bit RAM, 0 waits
		5.1	ld	(localpz), 80H	; LOCAL-Z bank enable for program
		5.2	ld	(localrz), 80H	; LOCAL-Z bank enable for data read
		6	ld	(p8fc), 02H	; P81: $\overline{\text{CS1}}$
		7	ld	(p8fc2), 04H	; P82: $\overline{\text{CSZA}}$
		8	ld	(pjfc), 07H	; PJ2: $\overline{\text{SRWR}}$, PJ1: $\overline{\text{SRLUB}}$, PJ0: $\overline{\text{SRLLB}}$
		9	ld	xsp, 6000H	; Stack pointer = 6000H
		10	ld	(localpy), 80H	; BANK 0 in LOCAL-Y is set as program for sub routine
		11	:		;
C000yyH	←	12	call	400000H	; Call sub routine
		13	:		;
		14	:		;
		15	:		;

- Instructions from No.2 to No.8 are settings for 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 $\overline{\text{CS1}}$ for SRAM will be asserted because its logical address is in the CS1area at the same time. These instructions allow the CPU to branch to sub routine.

(Note) This example assumes a sub routine program is already written on SRAM.

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

Logical Address	Physical Address	No	Instruction		Comment
		16	org	400000H	;
400000H	000000H	17	ld	(localwy), 81H	; BANK 1 in LOCAL-Y is set as write data for LCD display RAM
4000xxH	0000xxH	18	ld	(locally), 81H	; BANK 1 in LOCAL-Y is set as LCD display data for LCD display RAM
		19	ld	(localrz), 80H	; BANK 0 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 settings 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 the 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 $\overline{\text{CSZA}}$ for NOR flash will be asserted because its logical address is in the CS2 area at the same time.
  These instructions allow the CPU to read data from character ROM.
- No.23 is an instruction which changes the 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 $\overline{\text{CS1}}$ for SRAM will be asserted because its logical address is in the CS1area at the same time.
  These instructions allow the CPU to write data to SRAM.
- No.28 and No.29 are settings 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 $\overline{\text{CS1}}$ for SRAM will be asserted because its logical address is in the CS1 area at the same time.
  These instructions allow the LCDC to read data from SRAM.
- No.30 is an instruction to start LCD display operation.

3.9 Serial Channels

The TMP92CH21 includes 2 serial I/O channels. For each channel, 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 wakeup function for making the master controller start slave controllers via a serial link (a multi controller system).

Figure 3.9.2, Figure 3.9.3 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) $\overline{\text{CTS0}}$, SCLK0 (P92 or PF2)	TXD1 (PF0) RXD1 (PF1) $\overline{\text{CTS1}}$, SCLK1 (PF2)
IrDA mode	Yes	No

This chapter contains the following sections:

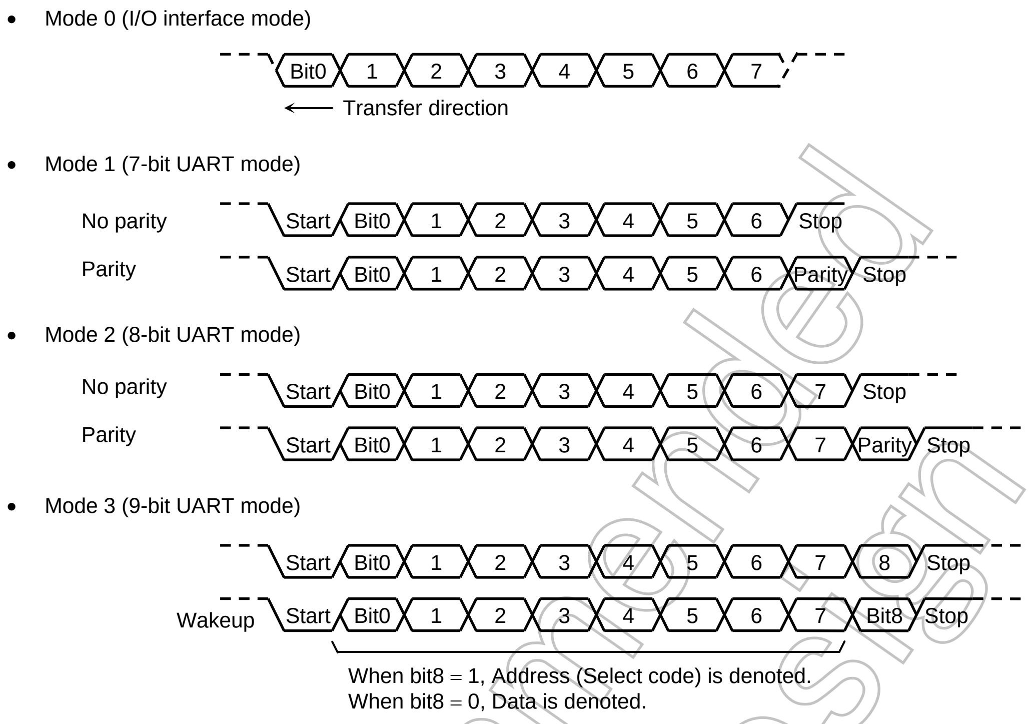

Mode 0 (I/O interface mode)
Bit0 1 2 3 4 5 6 7
Transfer direction
Mode 1 (7-bit UART mode)
No parity
Start Bit0 1 2 3 4 5 6 Stop
Parity
Start Bit0 1 2 3 4 5 6 Parity Stop
Mode 2 (8-bit UART mode)
No parity
Start Bit0 1 2 3 4 5 6 7 Stop
Parity
Start Bit0 1 2 3 4 5 6 7 Parity Stop
Mode 3 (9-bit UART mode)
Start Bit0 1 2 3 4 5 6 7 8 Stop
Wakeup
Start Bit0 1 2 3 4 5 6 7 Bit8 Stop
When bit8 = 1, Address (Select code) is denoted.
When bit8 = 0, Data is denoted.

Figure 3.9.1 Data Formats

3.9.1 Block Diagrams

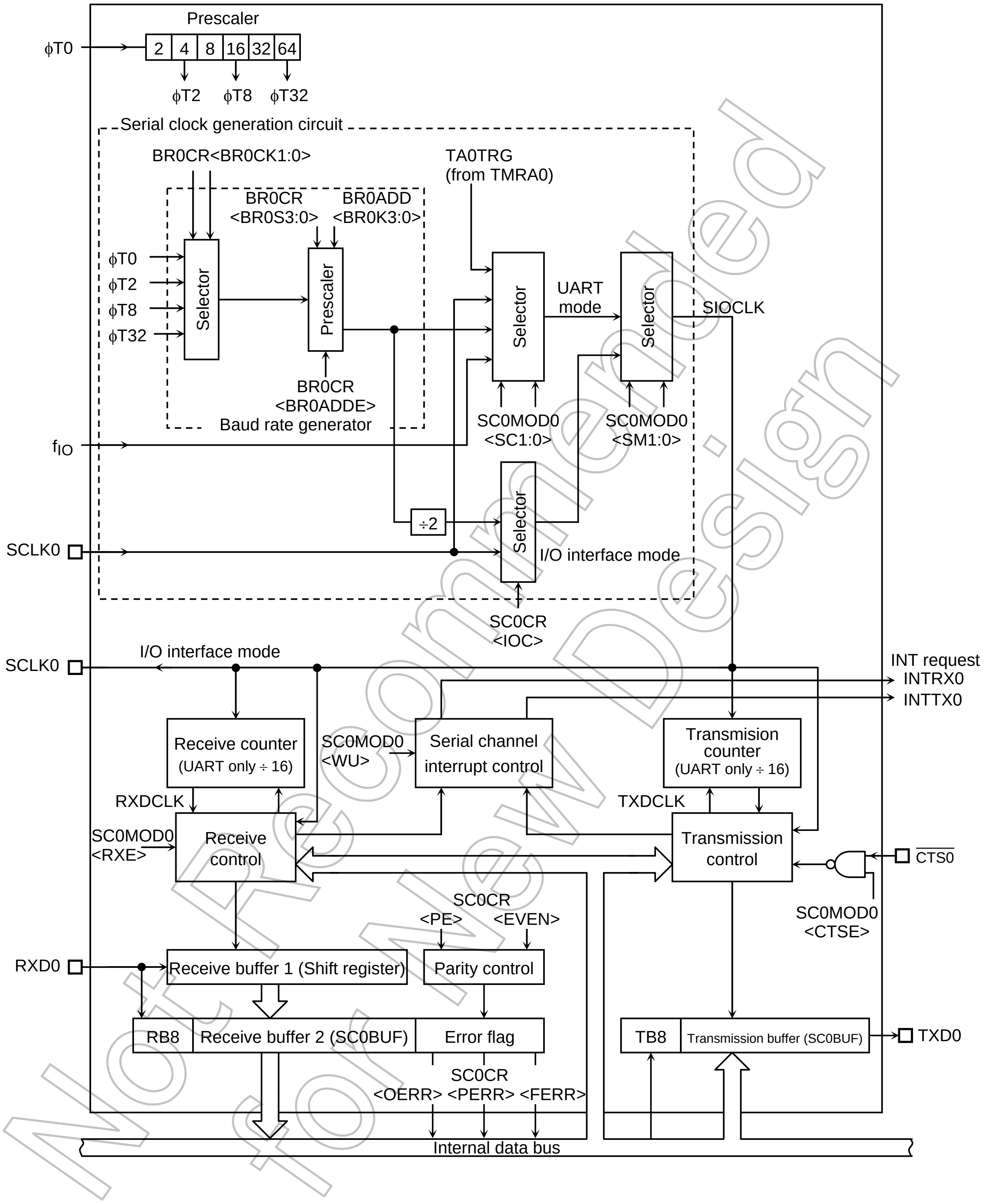


Figure 3.9.2 Block Diagram of Serial Channel 0

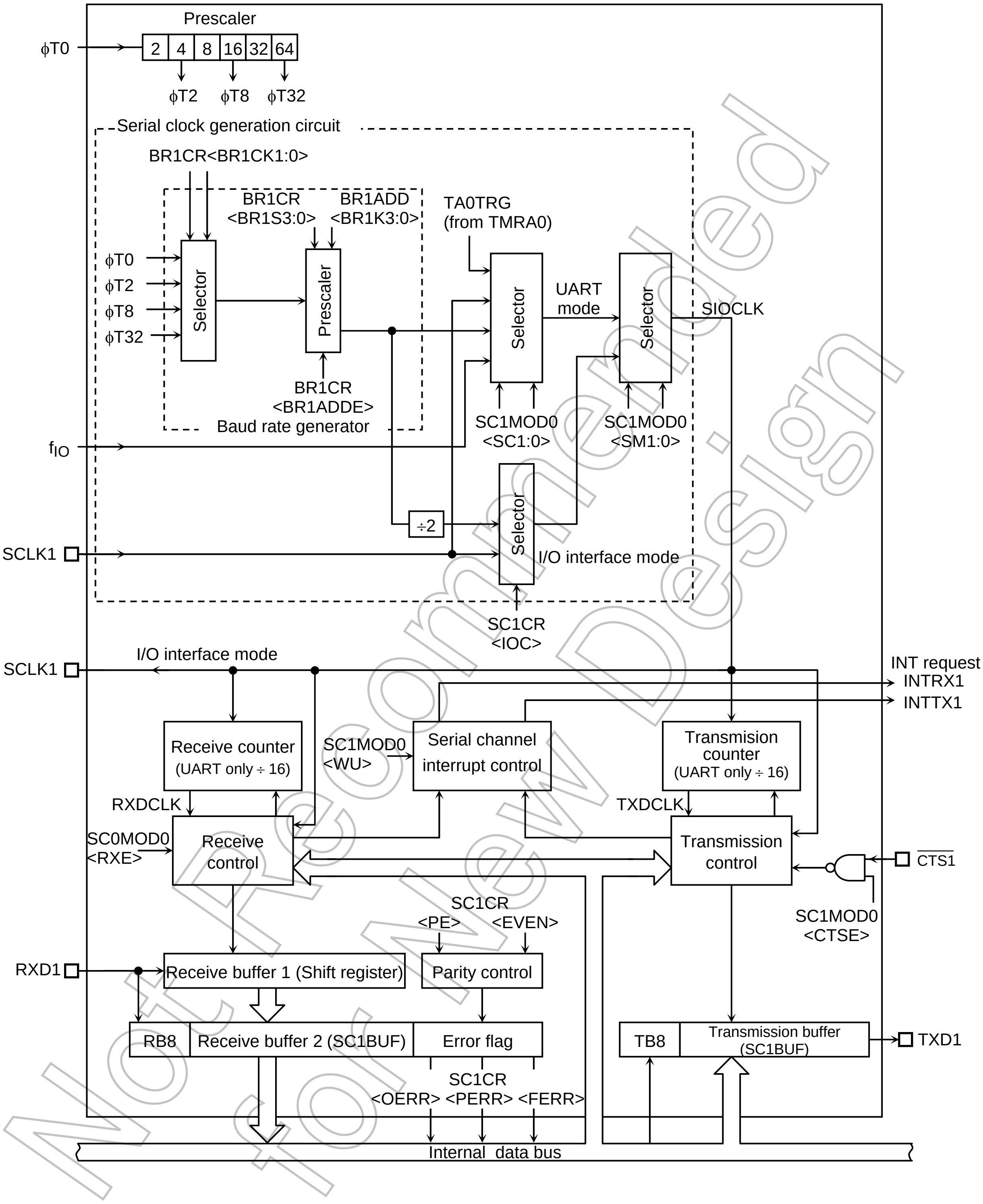


Figure 3.9.3 Block Diagram of Serial Channel 1

3.9.2 Operation for Each Circuit

(1) SIO Prescaler and 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.

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 selection SYSCR1 <SYSCK>	Clock gear selection SYSCR1 <GEAR2:0>	–	Baud rate generator input clock SIO prescaler BR0CR<BR0CK1:0>			
			φT0	φT2(1/4)	φT8(1/16)	φT32(1/64)
1 (fs)	–	1/8	fs/8	fs/32	fs/128	fs/512
0 (fc)	000(1/1)		fc/8	fc/32	fc/128	fc/512
	001(1/2)		fc/16	fc/64	fc/256	fc/1024
	010(1/4)		fc/32	fc/128	fc/512	fc/2048
	011(1/8)		fc/64	fc/256	fc/1024	fc/4096
	100(1/16)		fc/128	fc/512	fc/2048	fc/8192

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 SIO prescaler, which is shared by the timers. One of these input clocks is selected using the BR0CR<BR0CK1:0> field in the baud rate generator control register.

The baud rate generator includes a frequency divider, which divides the frequency by 1 or N + (16 − K)/16 or 16 values, thereby determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3:0> and BR0ADD<BR0K3:0>.

- In UART mode

(1) When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3:0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CK<BR0S3:0>. (N = 1, 2, 3 …16)

(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<BR0S3:0> (N = 2, 3…15) and the value of K set in BR0ADD<BR0K3:0> (K = 1, 2, 3…15)

Note: If N = 1 or N = 16, the N + (16 − K)/16 division function is disabled. 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 (f_C) is 39.3216 MHz, the input clock is ϕT2 (f_C/32), the frequency divider N (BR0CR<BR0S3:0>) = 8, and BR0CR<BR0ADDE> = 0, the baud rate in UART mode is as follows:

* Clock condition { Clock gear : 1/1

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

$$= \frac{f_C/32}{8} \div 16$$

$$= 39.3216 \times 10^6 \div 16 \div 8 \div 16 = 9600 \text{ (bps)}$$

Note: The N + (16 − K)/16 division function is disabled and setting BR0ADD<BR0K3:0> is invalid.

- N + (16 − K)/16 divider (UART mode only)

Accordingly, when the source clock frequency (f_C) = 31.9488 MHz, the input clock is ϕT2 (f_C/32), the frequency divider N (BR0CR<BR0S3:0>) = 6, K (BR0ADD<BR0K3:0>) = 8, and BR0CR<BR0ADDE> = 1, the baud rate in UART mode is as follows:

* Clock condition { Clock gear : 1/1

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

$$= \frac{f_C/32}{6 + \frac{(16-8)}{16}} \div 16$$

$$= 31.9488 \times 10^6 \div 16 \div (6 + \frac{8}{16}) \div 16 = 9600 \text{ (bps)}$$

Table 3.9.3 show examples of UART mode transfer rates.

Additionally, the external clock input is available in the serial clock. (Serial channels 0 and 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) ≥ 4/f_{SYS}

- In I/O interface mode

  Baud rate = external clock input frequency

  It is necessary to satisfy (External clock input cycle) ≥ 16/f_{SYS}

Table 3.9.3 Selection of Transfer Rate (1)
(when baud rate generator is used and BR0CR<BR0ADDE> = 0)

Unit (Kbps)

f_{SYS} [MHz]	Input Clock Frequency Divider	φT0 (f_{SYS}/4)	φT2 (f_{SYS}/16)	φT8 (f_{SYS}/64)	φT32 (f_{SYS}/256)
9.8304	2	76.800	19.200	4.800	1.200
↑	4	38.400	9.600	2.400	0.600
↑	8	19.200	4.800	1.200	0.300
↑	10	9.600	2.400	0.600	0.150
12.2880	5	38.400	9.600	2.400	0.600
↑	A	19.200	4.800	1.200	0.300
14.7456	2	115.200	28.800	7.200	1.800
↑	3	76.800	19.200	4.800	1.200
↑	6	38.400	9.600	2.400	0.600
↑	C	19.200	4.800	1.200	0.300
19.6608	1	307.200	76.800	19.200	4.800
↑	2	153.600	38.400	9.600	2.400
↑	4	76.800	19.200	4.800	1.200
↑	8	38.400	9.600	2.400	0.600
↑	10	19.200	4.800	1.200	0.300
22.1184	3	115.200	28.800	7.200	1.800
24.5760	1	384.000	96.000	24.000	6.000
↑	2	192.000	48.000	12.000	3.000
↑	4	96.000	24.000	6.000	1.500
↑	5	76.800	19.200	4.800	1.200
↑	8	48.000	12.000	3.000	0.750
↑	A	38.400	9.600	2.400	0.600
↑	10	24.000	6.000	1.500	0.375

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

In UART mode, TMRA match detect signal (TA0TRG) can be used for serial transfer clock.

Method for calculating the timer output frequency which is needed when outputting trigger of timer

TA0TRG frequency = Baud rate × 16

Note: The TMRA0 match detect signal cannot be used as the transfer clock in I/O Interface mode.

(3) Serial clock generation circuit

This circuit generates the basic clock for transmitting and receiving data.

- In I/O interface mode

  In SCLK output mode with the setting SC0CR<IOC> = 0, the basic clock is generated by dividing the output of the baud rate generator by 2, as described previously.

  In SCLK input mode with the setting SC0CR<IOC> = 1, the rising edge or falling edge will be detected according to the setting of the SC0CR<SCLKS> register to generate the basic clock.

- In UART mode

  The SC0MOD0<SC1:0> setting determines whether the baud rate generator clock, the internal clock f_{IO}, the match detect signal from 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 or falling of the shift clock, which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.

  In SCLK input mode with the setting SC0CR<IOC> = 1, the 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 wakeup 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 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.

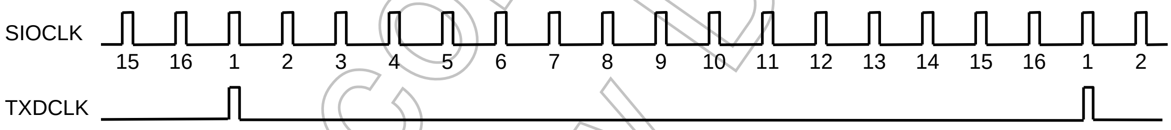


Figure 3.9.4 Generation of the Transmission Clock

(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 or falling edge of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.

  In SCLK input mode with the setting SC0CR<IOC> = 1, the 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

Use of $\overline{\text{CTS0}}$ pin allows data to be sent in units of one frame; thus, overrun errors can be avoided. The handshake function is enabled or disabled by the SC0MOD<CTSE> setting.

When the $\overline{\text{CTS0}}$ pin goes high on completion of the current data send, data transmission is halted until the $\overline{\text{CTS0}}$ pin goes low again. However, the INTTX0 interrupt is generated, and it requests the next data send from the CPU. The next data is written in the transmission buffer and data sending is halted.

Though there is no $\overline{\text{RTS}}$ pin, a handshake function can be easily configured by setting any port assigned to be the $\overline{\text{RTS}}$ function. The $\overline{\text{RTS}}$ should be output "high" to request send data halt after data receive is completed by software in the RXD interrupt routine.

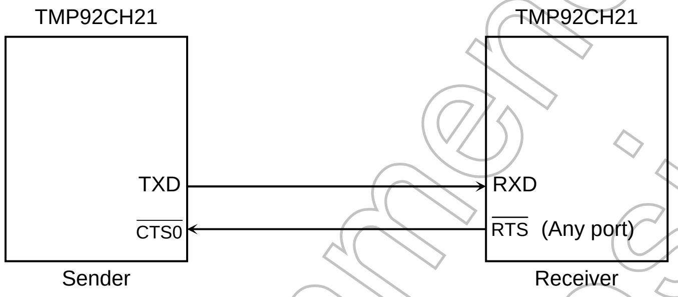
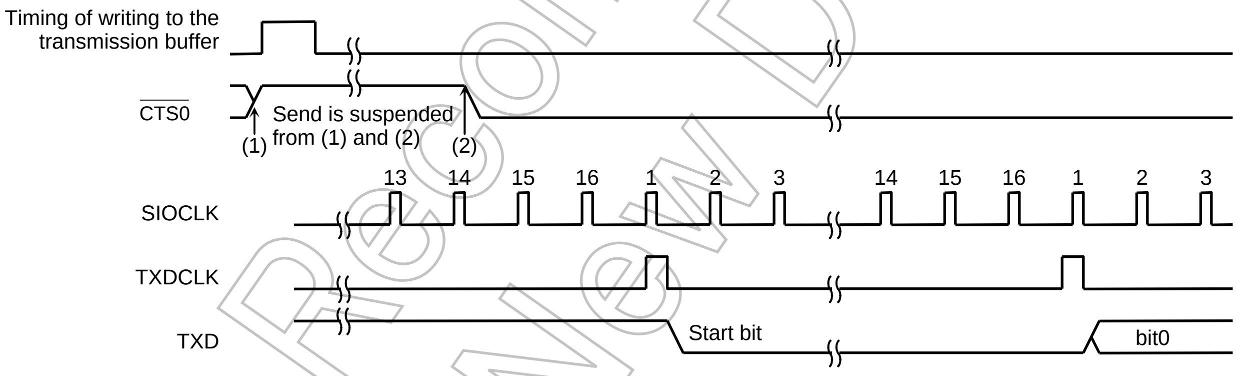


Figure 3.9.5 Handshake Function


Timing of writing to the transmission buffer
CTS0
Send is suspended from (1) and (2)
(1)
(2)
SIOCLK
13
14
15
16
1
2
3
14
15
16
1
2
3
TXDCLK
TXD
Start bit
bit0

Note 1: If the $\overline{\text{CTS0}}$ 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 $\overline{\text{CTS0}}$ signal has fallen.

Figure 3.9.6 $\overline{\text{CTS0}}$ (Clear to send) Timing

(9) Transmission buffer

The transmission buffer (SC0BUF) shifts out and sends the transmission data written from the CPU in order from the least significant bit (LSB). When all the bits are shifted out, the transmission buffer becomes empty and generates an INTTX0 interrupt.

(10) Parity control circuit

When SC0CR<PE> in the serial channel control register is set to “1”, it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART mode or 8-bit UART mode. The SC0CR<EVEN> field in the serial channel control register allows either even or odd parity to be selected.

In the case of transmission, parity is automatically generated when data is written to the transmission buffer SC0BUF. The data is transmitted after the parity bit has been stored in SC0BUF<TB7> in 7-bit UART mode or in SC0MOD0<TB8> in 8-bit UART mode. SC0CR<PE> and SC0CR<EVEN> must be set before the transmission data is written to the transmission buffer.

In the case of receiving, data is shifted into receiving buffer 1, and the parity is added after the data has been transferred to receiving buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> in 7-bit UART mode or with SC0CR<RB8> in 8-bit UART mode. If they are not equal, a parity error is generated and the SC0CR<PERR> flag is set.

(11) Error flags

Three error flags are provided to increase the reliability of data reception.

1. Overrun error <OERR>

If all the bits of the next data item have been received in receiving buffer 1 while valid data still remains stored in receiving buffer 2 (SC0BUF), an overrun error is generated.

The below is a recommended flow when the overrun-error is generated.

(INTRX interrupt routine)

```
1)  Read receiving buffer
2)  Read error flag
3)  If <OERR> = 1
    then
      a)  Set to disable receiving (Write “0” to SC0MOD0<RXE>)
      b)  Wait to terminate current frame
      c)  Read receiving buffer
      d)  Read error flag
      e)  Set to enable receiving (Write “1” to SC0MOD0<RXE>)
      f)  Request to transmit again
4)  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 Bits (Note)	8 Bits + Parity (Note)	8 Bits, 7 Bits + Parity, 7 Bits
Interrupt Timing	Center of last bit (bit8)	Center of last bit (parity bit)	Center of stop bit
Framing Error Timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity Error Timing	−	Center of last bit (parity bit)	Center of stop bit
Overrun Error Timing	Center of last bit (bit8)	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 Bits	8 Bits + Parity	8 Bits, 7 Bits + Parity, 7 Bits
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 last bit data. (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) (e.g. immediately after last SCLK). (See Figure 3.9.21.)
	SCLK input mode	Timing used to transfer received data to receive buffer 2 (SC0BUF) (e.g. immediately after last SCLK). (See Figure 3.9.22.)

3.9.3 SFR

		7	6	5	4	3	2	1	0
SC0MOD0 (1202H)	Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer data bit8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wakeup 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 clock (SCLK0 input)	

Serial transmission clock source (UART)

00	TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{IO}
11	External clock (SCLK0 input)

Note: The clock selection for the I/O interface mode is controlled by the serial control register (SC0CR).

Serial transmission mode

00	I/O interface mode	
01	UART mode	7-bit mode
10		8-bit mode
11		9-bit mode

Wakeup function

	9-bit UART	Other modes
0	Interrupt generated when data is received	Don't care
1	Interrupt generated only when SC0CR<RB8> = 1	

Receiving function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{\text{CTS}}$ pin)

0	Disabled (always transferable)
1	Enabled

Transmission data bit8

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							
Reset State	0	0	0	0	0	0	0	0
Function	Transfer data bit8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wakeup 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 clock (SCLK1 input)	

Serial transmission clock source (for UART)

00	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

Wakeup function

	9-bit UART	Other modes
0	Interrupt generated when data is received	Don't care
1	Interrupt generated only when SC1CR<RB8> = 1	

Receiving function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{\text{CTS}}$ pin)

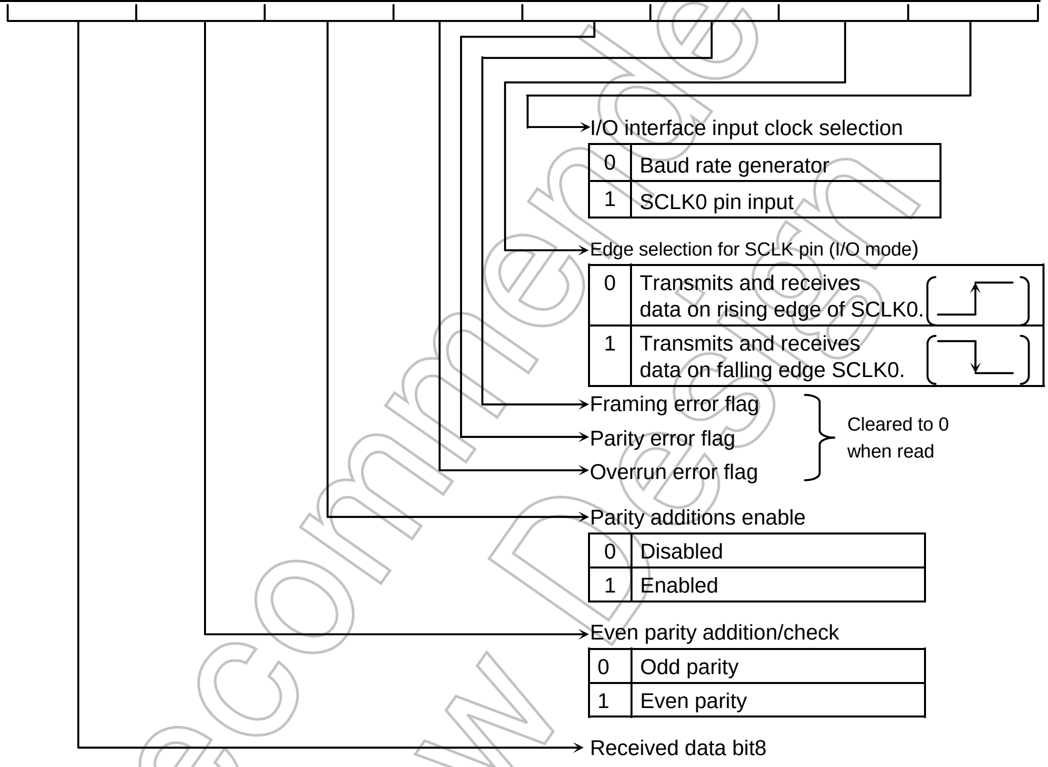
0	Disabled (always transferable)
1	Enabled

Transmission data bit8

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	
Reset State	Undefined	0	0	0	0	0	0	0
Function	Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK0 ↑ 1: SCLK0 ↓	0: Baud rate generator 1: SCLK0 pin input



I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Edge selection for SCLK pin (I/O mode)

0	Transmits and receives data on rising edge of SCLK0.	↑
1	Transmits and receives data on falling edge SCLK0.	↓

Framing error flag
Parity error flag
Overrun error flag
} Cleared to 0 when read

Parity additions enable

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

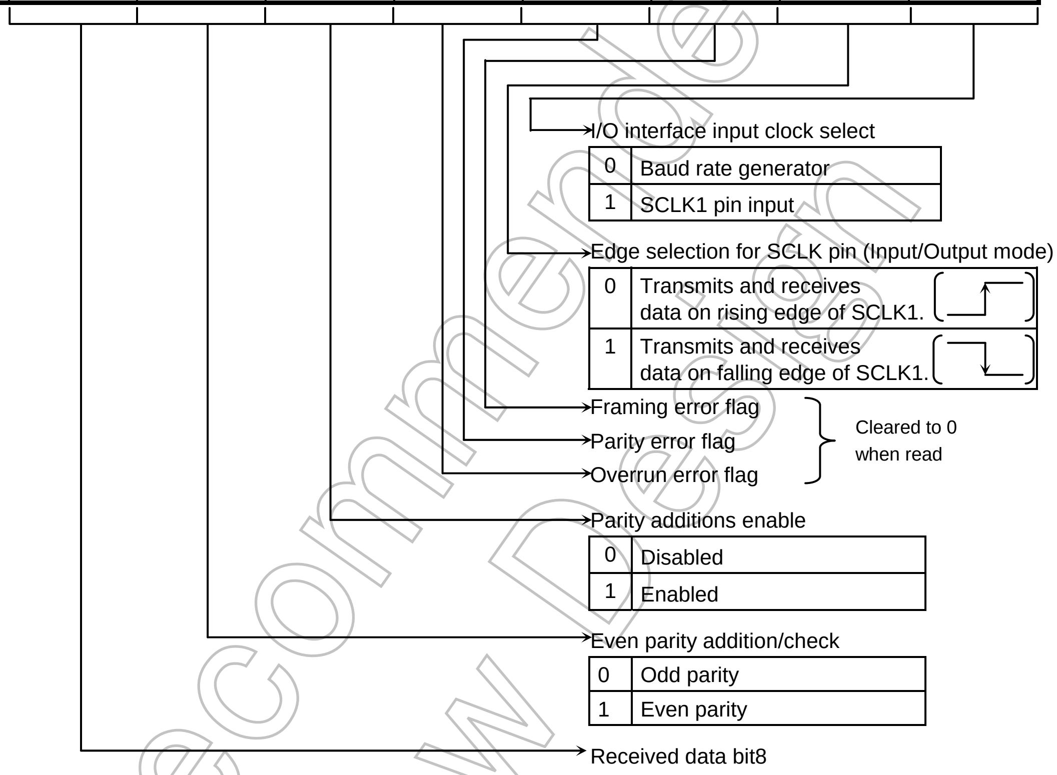
Received data bit8

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	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when read)			R/W	
Reset State	Undefined	0	0	0	0	0	0	0
Function	Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK1 1: SCLK1	0: Baud rate generator 1: SCLK1 pin input



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 0 when read

Parity additions enable

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data bit8

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							
	Reset State	0	0	0	0	0	0	0	0
	Function	Always write “0”.	+(16 − K)/16 division 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Divided frequency setting			

+(16 − K)/16 division enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

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

BR0ADD (1204H)		7	6	5	4	3	2	1	0
	Bit symbol					BR0K3	BR0K2	BR0K1	BR0K0
	Read/Write					R/W			
	Reset State					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) (Only UART) to 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	Divided by N
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by N + (16 − K)/16	

Note1:Availability of +(16-K)/16 division function

N	UART mode	I/O mode
2 to 15	○	×
1 , 16	×	×

The baud rate generator can be set to “1” in UART mode only when the +(16-K)/16 division function is not used. Do not use in I/O interface mode.

Note2:Set BR0CR <BR0ADDE> to 1 after setting K (K = 1 to 15) to BR0ADD<BR0K3:0> when +(16-K)/16 division function is used. Writes to unused bits in the BR0ADD register do not affect operation, and undefined data is read from these unused bits.

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							
Reset State	0	0	0	0	0	0	0	0
Function	Always write “0”.	+ (16 − K)/16 division 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Divided frequency setting			

+ (16 − K)/16 division enable

0	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			
Reset State					0	0	0	0
Function					Set frequency divisor K (divided by N + (16 − K)/16).			

Baud rate generator frequency divisor setting

	BR1CR<BR1ADDE> = 1		BR1CR<BR1ADDE> = 0
BR1CR <BR1S3:0> BR1ADD <BR1K3:0>	0000 (N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (Only UART) to 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	Divided by N
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by N + (16 − K)/16	

Note1:Availability of +(16-K)/16 division function

N	UART mode	I/O mode
2 to 15	○	×
1 , 16	×	×

The baud rate generator can be set “1” in UART mode only when the +(16-K)/16 division function is not used. Do not use in I/O interface mode.

Note2:Set BR1CR <BR1ADDE> to 1 after setting K (K = 1 to 15) to BR1ADD<BR1K3:0> when the +(16-K)/16 division function is used. Writes to unused bits in the BR1ADD register do not affect operation, and undefined data is read from these unused bits.

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)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

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						
Reset State	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)

7	6	5	4	3	2	1	0	
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	I2S1	FDPX1						
Read/Write	R/W	R/W						
Reset State	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.

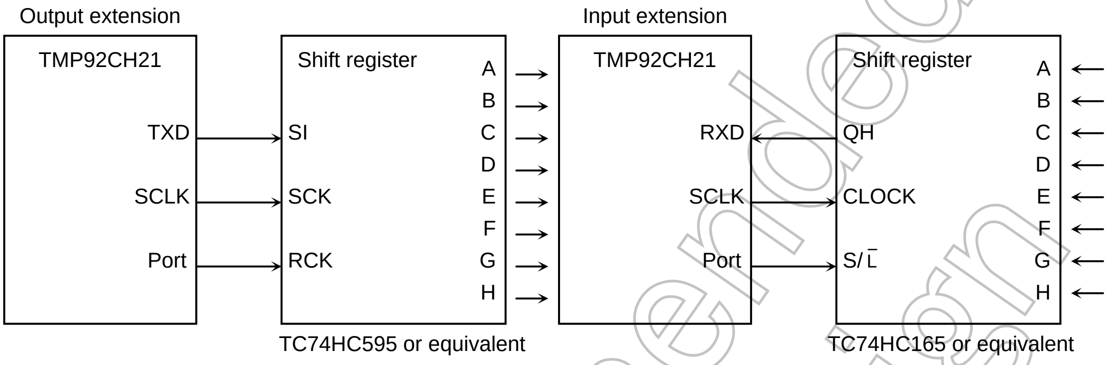


Figure 3.9.17 SCLK Output Mode Connection Example

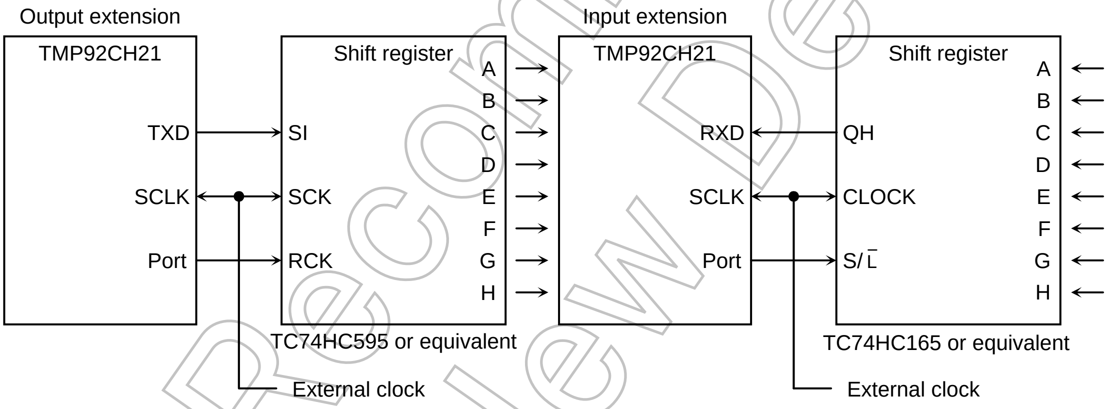


Figure 3.9.18 Example of SCLK Input Mode Connection

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 data to the transmission buffer. When all data is output, INTES0<ITX0C> will be set to generate the INTTX0 interrupt.

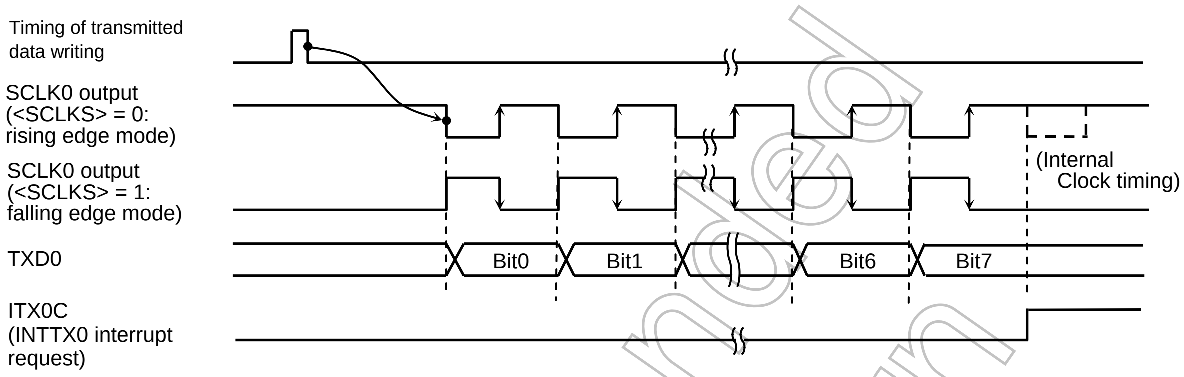

Timing of transmitted data writing
SCLK0 output (<SCLKS> = 0: rising edge mode)
SCLK0 output (<SCLKS> = 1: falling edge mode)
TXD0
Bit0
Bit1
Bit6
Bit7
(Internal Clock timing)
ITX0C (INTTX0 interrupt request)

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 an INTTX0 interrupt.

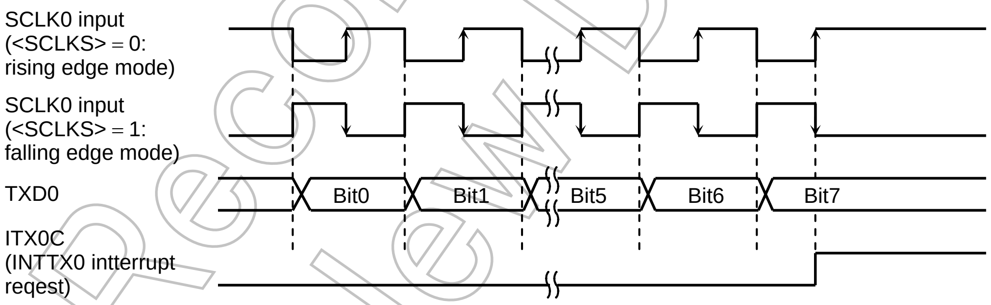

SCLK0 input (<SCLKS> = 0: rising edge mode)
SCLK0 input (<SCLKS> = 1: falling edge mode)
TXD0
Bit0
Bit1
Bit5
Bit6
Bit7
ITX0C (INTTX0 intterrupt reqest)

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.

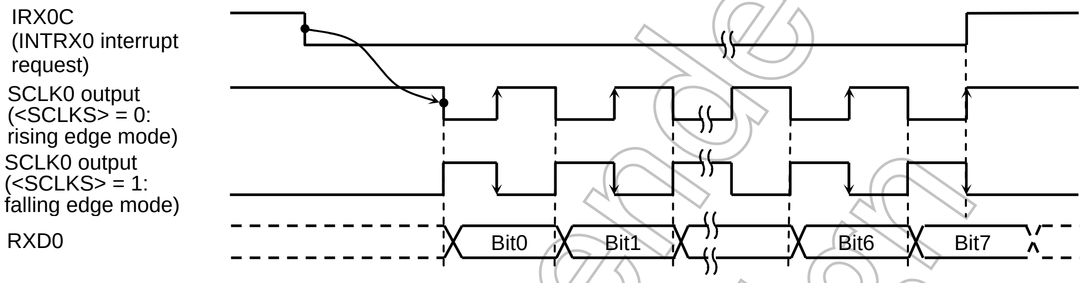

IRX0C
(INTRX0 interrupt request)
SCLK0 output
(<SCLKS> = 0: rising edge mode)
SCLK0 output
(<SCLKS> = 1: falling edge mode)
RXD0
Bit0
Bit1
Bit6
Bit7

Figure 3.9.21 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.

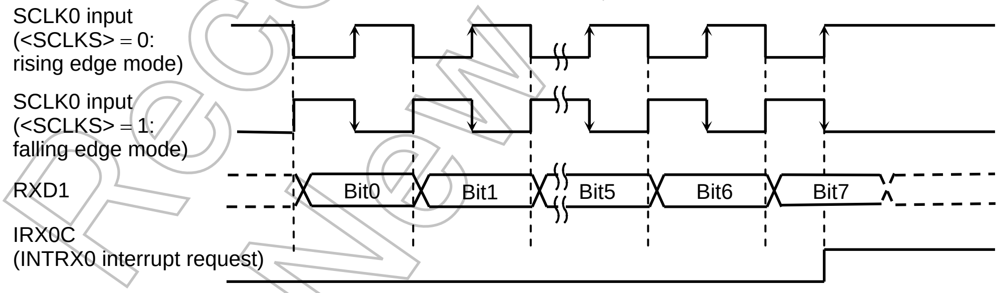

SCLK0 input
(<SCLKS> = 0: rising edge mode)
SCLK0 input
(<SCLKS> = 1: falling edge mode)
RXD1
Bit0
Bit1
Bit5
Bit6
Bit7
IRX0C
(INTRX0 interrupt request)

Figure 3.9.22 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 only set the interrupt level (from 1 to 6) of the 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 = 4.9152 MHz
*Clock condition: Clock gear 1/1(fc)

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	X	X	X	X	X	1	0	1	Set PF0, PF1 and PF2 to function as the TXD0, RXD0 and SCLK0 pins respectively.
PFFC	–	X	X	X	X	1	0	1	
SC0MOD0	–	–	–	–	0	0	–	–	Select I/O interface mode.
SC0MOD1	–	1	X	X	X	X	X	X	Select full duplex mode.
SC0CR	–	–	–	–	–	–	0	0	SCLK output, transmit on negative edge, receive on positive edge.
BR0CR	0	0	0	1	1	0	0	0	Baud rate = 9600 bps.
SC0MOD0	–	–	1	–	–	–	–	–	Enable receiving.
SC0BUF	*	*	*	*	*	*	*	*	Set the transmit data and start.

INTTX0 interrupt routine

A_{CC} ←	SC0BUF								Read the receiving buffer.
SC0BUF	*	*	*	*	*	*	*	*	Set the next transmit data.

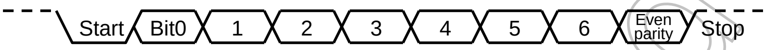
X: Don't care, –: No change

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

7-bit UART mode is selected by setting the serial channel mode register SC0MOD0<SM1:0> field to 01.

In this mode a parity bit can be added. Use of a parity bit is enabled or disabled by the setting of the serial channel control register SC0CR<PE> bit; whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (enabled).

Setting example: When transmitting data of the following format, the control registers should be set as described below.



← Transmission direction (Transmission rate: 2400 bps at f_C = 39.3216 MHz)

*Clock condition: Clock gear 1/1(fc)

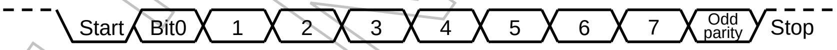
		7	6	5	4	3	2	1	0	
PFCR	←	X	X	X	X	X	–	–	1	Set PF0 to function as the TXD0 pin.
PFFC	←	–	X	X	X	X	–	–	1	
SC0MOD0	←	–	0	–	–	0	1	0	1	Select 7-bit UART mode.
SC0CR	←	–	1	1	–	–	–	0	0	Add even parity.
BR0CR	←	0	0	1	0	1	0	0	0	Set the transfer rate to 2400 bps.
INTES0	←	X	1	0	0	–	–	–	–	Enable the INTTX0 interrupt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

X: Don't care, –: No change

(3) Mode 2 (8-bit UART mode)

8-bit UART mode is selected by setting SC0MOD0<SM1:0> to 10. In this mode a parity bit can be added (use of a parity bit is enabled or disabled by the setting of SC0CR<PE>); whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (enabled).

Setting example: When receiving data of the following format, the control registers should be set as described below.



← Transmission direction (Transmission rate: 9600 bps at f_C = 39.3216 MHz)

Main settings

		7	6	5	4	3	2	1	0	
PFCR	←	X	X	X	X	X	–	0	–	Set PF1 to function as the RXD0 pin.
PFFC	←	–	X	X	X	X	–	0	–	
SC0MOD0	←	–	0	1	–	1	0	0	1	Enable receiving in 8-bit UART mode.
SC0CR	←	–	0	1	–	–	–	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

A_{CC}	← SC0CR AND 00011100	} Check for errors
if $A_{CC} \neq 0$ then ERROR		
A_{CC}	← SC0BUF	Read the received data

X: Don't care, –: No change

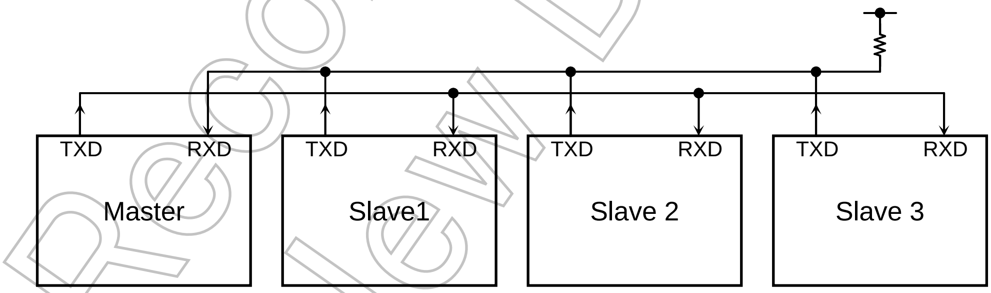
(4) Mode 3 (9-bit UART mode)

9-bit UART mode is selected by setting SC0MOD0<SM1:0> to 11. In this mode a 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 or read, <TB8>or <RB8> is read or written first, before the rest of the SC0BUF data.

Wakeup function

In 9-bit UART mode, the wakeup function for slave controllers is enabled by setting SC0MOD0<WU> to 1. The interrupt INTRX0 can only be generated when<RB8> = 1.

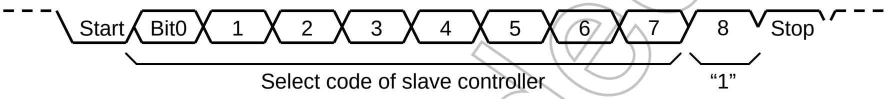


Note: The TXD pin of each slave controller must be in open-drain output mode.

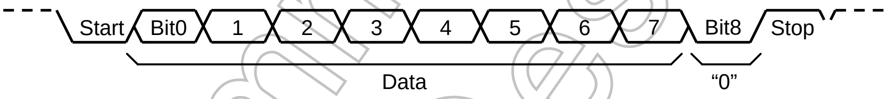
Figure 3.9.23 Serial Link Using Wakeup Function

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 (bit8) of the data (<TB8>) is set to 1.

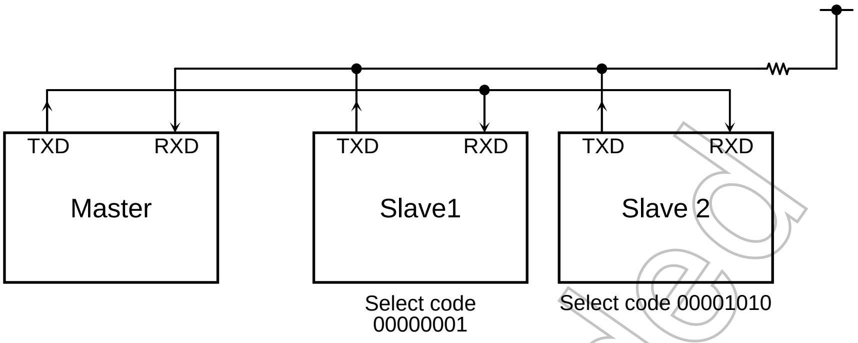


4. Each slave controller receives the above frame. Each controller checks the above select code against its own select code. The controller whose code matches clears its <WU> bit to 0.
5. The master controller transmits data to the specified slave controller (the controller whose SC0MOD0<WU> bit has been cleared to 0). The MSB (bit8) 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 (bit8 or <RB8>) are set to 0, disabling INTRX0 interrupts.
   The slave controller whose <WU> bit = 0 can also transmit to the master controller. In this way it can signal the master controller that the data transmission from the master controller has been completed.

Setting example: To link two slave controllers serially with the master controller using the internal clock f_{IO} as the transfer clock.



- Setting the master controller

```
Main

PFCR               ←  X  X  X  X  X  –  0  1   } Set PF0 and PF1 to function as the TXD0 and RXD0 pins
PFFC               ←  –  X  X  X  X  –  0  1   } respectively.
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               ←  X  X  X  X  X  –  0  1   } Select PF1 and PF0 to function as the RXD0 and TXD0 pins
PFFC               ←  –  X  X  X  X  –  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            ←  1  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 3.9.24 shows the block diagram.

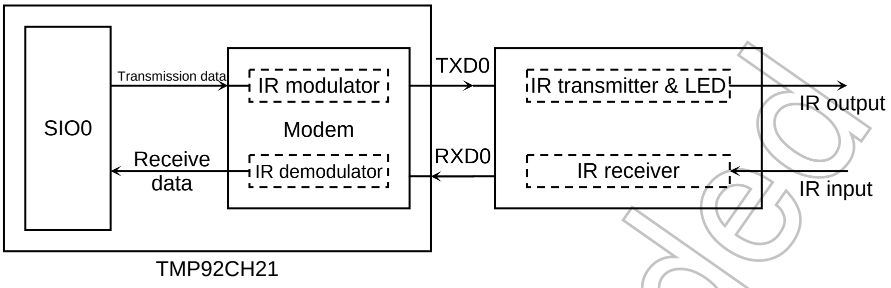


Figure 3.9.24 Block Diagram

(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.

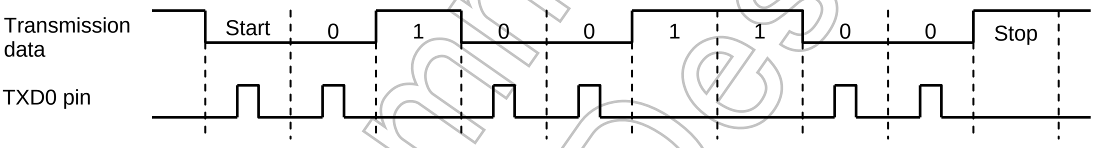


Figure 3.9.25 Transmission Example

(2) Modulation of the receive data

When the receive data has an effective pulse width of "1", the modem outputs "0" to SIO0. Otherwise the modem outputs "1" to SIO0. The effective pulse width is selected by SIRCR<SIRWD3:0>.

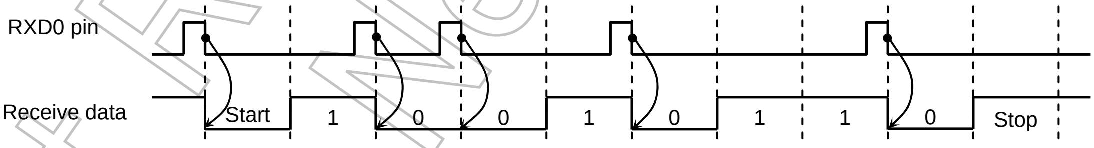


Figure 3.9.26 Receiving Example

(3) Data format

The data format is fixed as follows:

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

(4) SFR

Figure 3.9.27 shows the control register SIRCR. Set SIRCR data while SIO0 is stopped. The following example describes how to set this register:

```
1)  SIO setting                          ;   Set the SIO to UART mode.
    ↓
2)  LD      (SIRCR), 07H                 ;   Set the receive data pulse width to 16×.
3)  LD      (SIRCR), 37H                 ;   TXEN, RXEN Enable the transmission and receiving.
    ↓
4)  Start transmission                   ;   The modem operates as follows:
    and receiving for SIO0                   • SIO0 starts transmitting.
                                             • IR receiver starts receiving.
```

(5) Notes

1. Baud rate for IrDA

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

   Settings other than the above (TA0TRG, f_{IO} and SCLK0 input) cannot be used.

2. The pulse width for transmission

   The IrDA 1.0 specification is defined in Table 3.9.4.

Table 3.9.4 Baud Rate and Pulse Width Specifications

Baud Rate	Modulation	Rate Tolerance (% of rate)	Pulse Width (min)	Pulse Width (typ.)	Pulse Width (max)
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 as 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 a function which can select the pulse width of transmission as either 3/16 or 1/16. However, 1/16 pulse width can only be selected when the baud rate is equal to or less than 38.4 Kbps.

For the same reason, when using IrDA 115.2 Kbps with USB, the + (16 − K)/16 division function in the baud rate generator of SIO0 cannot be used to generate a 115.2 Kbps baud rate, except under special conditions as explained in (6) below.

The + (16 − K)/16 division function cannot be used also when the baud rate is 38.4 Kbps and the pulse width is 1/16.

Table 3.9.5 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	−	−	×	○	○	○

○: (16 − K)/16 division function can be used.
×: (16 − K)/16 division function cannot be used.
−: Cannot be set to 1/16 pulse width.
Note: (16 − K)/16 division function can be used under special conditions.

(6) Using IrDA 115.2 Kbps with USB

When the system uses USB , set f_{OSCH} to 9.0 MHz. In this case, the IrDA cannot be 115.2 Kbps without using the (16 − K)/16 division function.

Therefore, only in this case, the following conditions can be used.

(Setting condition)

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

(Calculation result)

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

SIRCR
(1207H)

	7	6	5	4	3	2	1	0
Bit symbol	PLSEL	RXSEL	TXEN	RXEN	SIRWD3	SIRWD2	SIRWD1	SIRWD0
Read/Write	R/W							
Reset State	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 to equal to or more than 2x × (value + 1) + 100 ns Can be set: 1 to 14 Cannot be set: 0, 15			

Select receive pulse width

Formula: Effective pulse width ≥ 2x × (value + 1) + 100 ns

$x = 1/f_{SYS}$

0000	Cannot be set
0001	Equal to or more than 4x + 100 ns
to	
1110	Equal to or more than 30x + 100 ns
1111	Cannot be set

Receive operation

0	Disable (Received input is ignored)
1	Enable

Transmit operation

0	Disable (Input from SIO is ignored)
1	Enable

Select transmit pulse width

0	3/16
1	1/16

Note: If a pulse width complying with IrDA1.0 standard (1.6 μs min.) can be guaranteed with a low baud rate, setting this bit to "1" will result in reduced power dissipation.

Figure 3.9.27 IrDA Control Register

3.10 USB Controller

3.10.1 Outline

This USB controller (UDC) is designed to support a variety of serial links in the construction of a USB system.

The outline is as follows:

(1) Compliant with USB rev1.1

(2) Full-speed: 12 Mbps (low-speed (1.5 Mbps) not supported)

(3) Auto bus enumeration with 384-byte descriptor RAM

(4) Supports 3 kinds of transfer type: Control, interrupt and bulk

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

(5) Built-in DPLL which generates sampling clock for receive data

(6) Detecting and generating SOP, EOP, RESUME, RESET and TIMEOUT

(7) Encoding and decoding NRZI data

(8) Inserting and discarding stuffed bit

(9) Detecting and checking CRC

(10) Generating and decoding packet ID

(11) Built-in power management function

(12) dual packet mode supported

Note1:The TMP92CH21 does not include the pull-up resistor necessary for D+pin. An external pull-up resistor plus software support is required.

Note2:There are some differences between our specifications and USB 1.1. Refer to “3.10.11 Notice and Restrictions”.

3.10.1.1 System Configuration

The USB controller (UDC) consists of the following 3 blocks.

1. 900/H1 CPU I/F (details given in Section 3.10.2, below).
2. UDC core block (DPLL, SIE, IFM and PWM), request controller, descriptor RAM and 4 endpoint FIFO (details given in Section 3.10.3, below).
3. USB transceiver

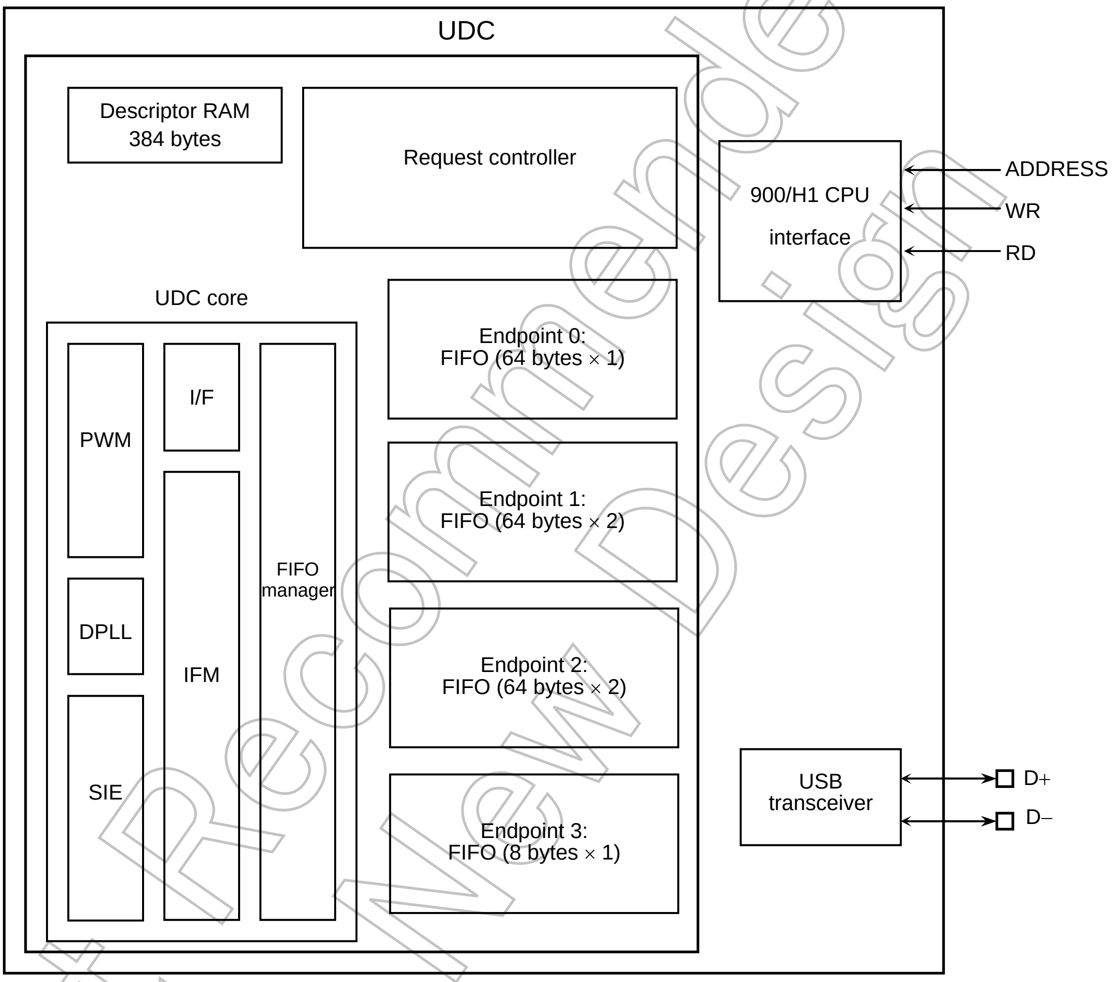
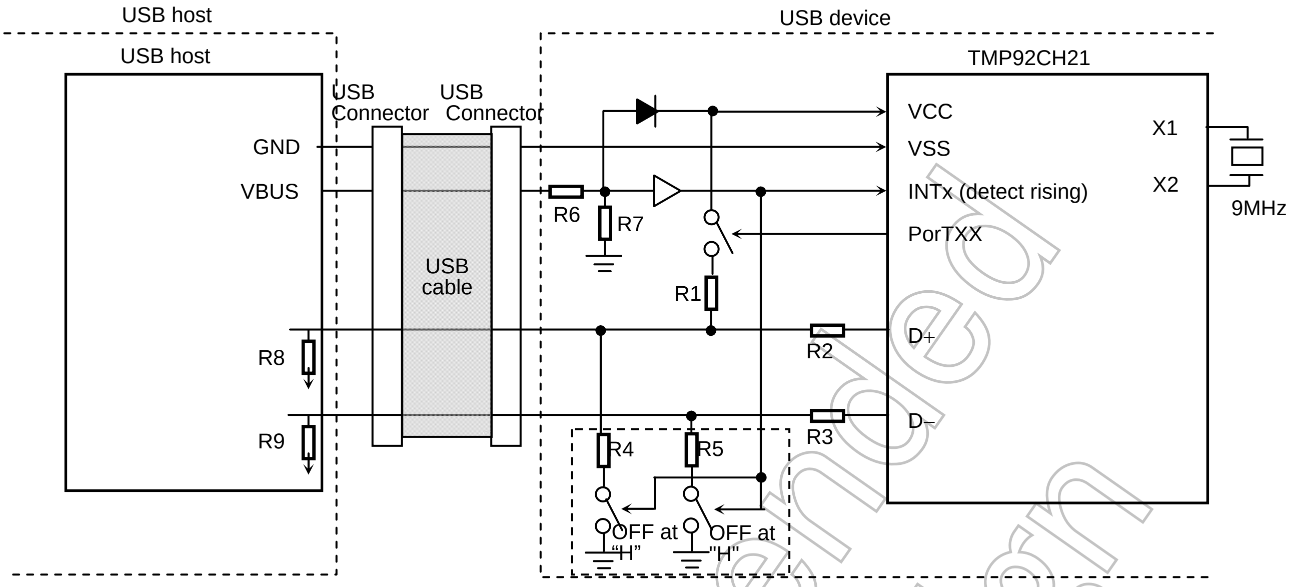


Figure 3.10.1 UDC Block Diagram

3.10.1.2 Example



The above setting is required when using the TMP92CH21's USB controller.

1) Pull-up of D^{+} pin

- In the USB standard, in Full Speed connection, the D^{+} pin must be set to pull-up. The ON/OFF control of this pull-up must be by S/W.

  Recommended value: R1=1.5kΩ

2) Add cascade resistor of D^{+}, D^{-}signal

- In the USB standard, for a D+ or D^{-} signal, a cascade resistor must be added to each signal. Recommended value : R2=27Ω, R3=27Ω

3) Flow current provision of the Connector connection and D^{+} pin, D^{-} pin

- For the D^{+} and D^{-} pin of the TMP92CH21, the level must be fixed for flow current provision when not in use (when not connected to host). In this case, the connector detection signal is used to control the pull-down resistor which determines the level..

  Recommended value: R4=10kΩ, R5=10kΩ

- The example shows use of the connector detection method using VBUS (5V voltage).

Note: Where waveform rise is solw, buffering of waveform is recommended .

Recommended value: R6=60kΩ, R7=100kΩ

(VBUS current consumption when suspended is <500μA)

4) Connection of 9MHz oscillator to X1, X2.

- When using USB with a combination of 9MHz external oscillator and internal PLL, the number of external hub stages which can be used is restricted by the accuracy of the internal PLL (Max 3 stages).

5) HOST side pull-down resistor

- In the USB standard, set pull-down D^{+} pin and D^{-} signal at USB_HOST side.

  Recommended value: R8=15kΩ, R9=15kΩ

Note: The above connections and resistor values, etc, are given as examples only. Operation is not guaranteed.
Please confirm the latest USB standard specifications and operations on your system.

3.10.2 900/H1 CPU I/F

The 900/H1 CPU I/F is a bridge between the 900/H1 CPU and the UDC. Its main functions are as follows:.

- INTUSB (interrupt from UDC) generation
- A bridge for SFR
- USB clock control (48 MHz)

3.10.2.1 SFRs

The 900/H1 CPU I/F incorporates the following SFRs to control the UDC and USB transceiver.

- USB control

USBCR1	(USB control register 1)

- USB interrupt control

USBINTFR1	(USB interrupt flag register 1)
USBINTFR2	(USB interrupt flag register 2)
USBINTFR3	(USB interrupt flag register 3)
USBINTFR4	(USB interrupt flag register 4)
USBINTMR1	(USB interrupt mask register 1)
USBINTMR2	(USB interrupt mask register 2)
USBINTMR3	(USB interrupt mask register 3)
USBINTMR4	(USB interrupt mask register 4)

Table 3.10.1 900/H1 CPU I/F SFR

Address	Read/Write	SFR Symbol
07F0H	R/W	USBINTFR1
07F1H	R/W	USBINTFR2
07F2H	R/W	USBINTFR3
07F3H	R/W	USBINTFR4
07F4H	R/W	USBINTMR1
07F5H	R/W	USBINTMR2
07F6H	R/W	USBINTMR3
07F7H	R/W	USBINTMR4
07F8H	R/W	USBCR1

3.10.2.2 USBCR1 Register

This register is used to set USB clock enables, transceiver enable etc.

USBCR1
(07F8H)

	7	6	5	4	3	2	1	0
bit Symbol	TRNS_USE	WAKEUP				−	SPEED	USBCLKE
Read/Write	R/W	R/W				R/W	R/W	R/W
Reset State	0	0				0	1	0
Function						Always write "0"		

- TRNS_USE (Bit7)

  0: Disable USB transceiver

  1: Enable USB transceiver

  Always set to "1" on the application using USB.

- WAKEUP (Bit6)

  0: −

  1: Start remote-wakeup function

  When the remote-wakeup function is needed, first check Current_Config<REMOTE WAKEUP>.

  If <REMOTE WAKEUP> = "1" (meaning SUSPEND-status), write "1", and "0" to <WAKEUP>. This will initiate the remote-wakeup function.

  If the <REMOTE WAKEUP> = "0" or EP0, 1, 2, 3_STATUS<SUSPEND> = "0", do not write "1" to <WAKEUP>.

- SPEED (Bit1)

  1: Full speed (12 MHz)

  0: Reserved

  This bit selects USB speed.

  Always set to "1".

- USBCLKE (Bit0)

  0: Disable USB clock

  1: Enable USB clock

  This bit controls supply of USB clock.

  The USB clock ("f_{USB}": 48MHz) is generated by an internal PLL. When the USB is started, write "1" to <USBCLKE> after confirming PLL lock up is terminated.

  Also, write "0" to <USBCLKE> before stopping the PLL.

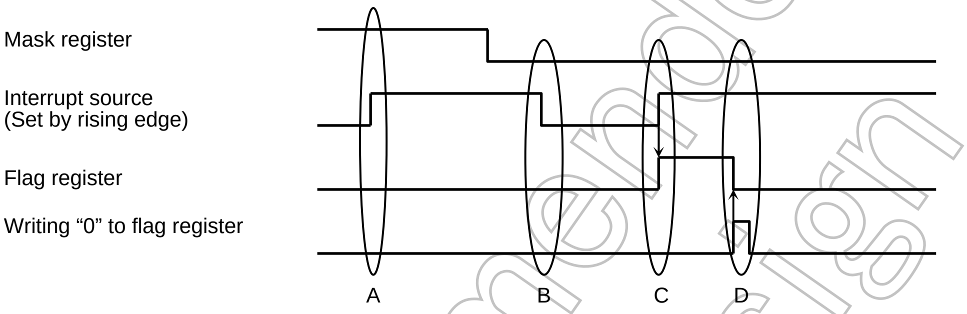
3.10.2.3 USBINTFRn, MRn Register

These SFRs control the INTUSB (only one interrupt to CPU) using the 23 interrupt sources output by the UDC.

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 common specification for all MASK and FLAG registers is shown below.

(Common specifications for all mask and flag registers.)



A: The flag register is not set because mask register = “1”.

B: The flag register is not set because interrupt souce changes “1” → “0”.

C: The flag register is set because mask register = “0” and interrupt souce changes “0” → “1”.

D: The flag register is reset to “0” by writing “0” to flag register.

Note 1: The “INTUSB generated number” and “bit number which is set to flag register” are not always equal. In the INTUSB interrupt routine, clear FLAG register (USBINTFRn) after checking it. The interrupt request flag, which occurs between the INTUSB interrupt routine and flag register (USBINTFRn) read, is kept in the interrupt controller.

Therefore, after returning from the interrupt routine, the CPU jumps to INTUSB interrupt routine again. Software support is required to avoid ending in an error routine when none of the bits in the flag register (USBINTFRn) is set to “1”.

Note 2: Disable INTUSB (write 00H to INTEUSB register) before writing to USBINTMRn or USBINTFRn .

		7	6	5	4	3	2	1	0
USBINTFR1 (07F0H)	bit Symbol	INT_URST_STR	INT_URST_END	INT_SUS	INT_RESUME	INT_CLKSTOP	INT_CLKON		
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		
	Reset State	0	0	0	0	0	0		
	Function	When read 0: Not generate interrupt When write 0: Clear flag 1: Generate interrupt 1: –							

Note: The above interrupts can release Halt state from IDLE2 and IDLE1 mode. (STOP mode cannot be released)

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

Shift to IDLE1 mode :

Execute Halt instruction when the INT_SUS or INT_CLKSTOP flag is "1" (SUSPEND state)

Release from IDLE1 mode :

Release Halt state by INT_RESUME or INT_CLKON request (request of release SUSPEND)

Release Halt state by INT_URST_STR or INT_URST_END request (request of RESET)

- INT_URST_STR (Bit7)

  This is the flag register for INT_URST_STR ("USB reset" start - interrupt).

  This is set to "1" when the UDC starts to receive a "USB reset" signal from a USB-host.

  An application program has to initialize the whole UDC with this interrupt.

- INT_URST_END (Bit6)

  This is the flag register for INT_URST_END ("USB reset" end - interrupt).

  This is set to "1" when the UDC receives a "USB reset end" signal from a USB-host.

- INT_SUS (Bit5)

  This is the flag register for INT_SUS (suspend - interrupt).

  This is set to "1" when the USB changes to "suspend status".

- INT_RESUME (Bit4)

  This is the flag register for INT_RESUME (resume - interrupt).

  This is set to "1" when the USB changes to "resume status".

- INT_CLKSTOP (Bit3)

  This is the flag register for INT_CLKSTOP (enables stopping of the clock supply - interrupt).

  This is set to "1" after the USB changes to "suspend status". Set USBCR1<USBCLKE> to "0" to stop the clock after detecting this interrupt if needed.

- INT_CLKON (Bit2)

  This is the flag register for INT_CLKON (enable starting of the clock supply - interrupt).

  This is set to "1" after changing to "resume status" or when the UDC started to receive a "USB reset" signal from a USB-host. In case the clock has be stopped, set USBCR1<USBCLKE> to "1" to start the clock after detecting this interrupt if needed.

USBINTFR2
(07F1H)

	7	6	5	4	3	2	1	0
bit Symbol	EP1_FULL_A	EP1_Empty_A	EP1_FULL_B	EP1_Empty_B	EP2_FULL_A	EP2_Empty_A	EP2_FULL_B	EP2_Empty_B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	0	0	0	0	0	0	0	0
Function	When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

Note: The above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode cannot be released.)

USBINTFR3
(07F2H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_FULL_A	EP3_Empty_A						
Read/Write	R/W	R/W						
Reset State	0	0						
Function	When read 0:Not generate interrupt 1:Generate interrupt When write 0: Clear flag 1: –							

Note: The above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode cannot be released.)

- EPx_FULL_A/B:

  (When transmitting)
  This is set to "1" when CPU full writes data to FIFO_A/B.
  (When receiving)
  This is set to "1" when UDC full receives data to FIFO_A/B.

- EPx_Empty_A/B:

  (When transmitting)
  This is set to "1" when FIFO becomes empty after transmission.
  (When receiving)
  This is set to "1" when FIFO becomes empty after CPU reads all data from FIFO.

Note: The EPx_FULL_A/B and EPx_Empty_A/B flags are not status flags. Therefore, check DATASET register to determine if FIFO-status is needed.

		7	6	5	4	3	2	1	0
USBINTFR4 (07F3H)	bit Symbol	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	EP2_Empty_B
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	0	0	0	0	0	0	0	0
	Function	When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

Note: The above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode cannot be released.)

- INT_SETUP (Bit7)

  This is the flag register for INT_SETUP (setup - interrupt).

  This is set to "1" when the UDC receives a request that S/W (software) control is needed from USB host.

  Using S/W (INT_SETUP routine), first read 8-byte device requests from the UDC and execute operation according to each request.

- INT_EP0 (Bit6)

  This is the flag register for INT_EP0 (received data of the data phase for Control transfer type - interrupt).

  This is set to "1" when the UDC receives data of the data phase for Control transfer type. If this interrupt occurs during Control write transfer, data reading from FIFO is needed. If this interrupt occurs during Control read transfer, transmission data writing to FIFO is needed.

  In some cases, the host may not assert "ACK" of the last packet in the data stage. In this case, this interrupt cannot be generated. Therefore, ignore this interrupt if it occurs after the last packet data has been written in the data stage because the transmission data number is specified by the host, or it depends on the capacity of the device.

- INT_STAS (Bit5)

  This is the flag register for INT_STAS (status stage end - interrupt).

  This is set to "1" when the status stage ends.

  If this interrupt is generated, it means that request ended normally.

  If this interrupt is not generated and INT_SETUP is generated, EP0_STATUS <STAGE_ERR> is set to "1", and it means that request did not end normally.

- INT_STASN (Bit4)

  This is the flag register for INT_STASN (change host status stage - interrupt).

  This is set to “1” when the USB host changes to status stage at the Control read transfer. This interrupt is needed if data length is less than wLength (specified by the host).

- INT_EPxN (Bit3, 2, 1)

  This is the flag register for INT_EPxN (NAK acknowledge to the USB host - interrupt).

  This is set to “1” when the Endpoint1, 2 and 3 transmit NAK.

USBINTMR1
(07F4H)

	7	6	5	4	3	2	1	0
bit Symbol	MSK_URST_STR	MSK_URST_END	MSK_SUS	MSK_RESUME	MSK_CLKSTOP	MSK_CLKON		
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		
Reset State	1	1	1	1	1	1		
Function	When read 0: not masked 1: masked When write 0: Clear flag 1: –							

- MSK_URST_STR (Bit7)

  This is the mask register for USBINTFR1<INT_URST_STR>.

- MSK_URST_END (Bit6)

  This is the mask register for USBINTFR1<INT_URST_END>.

- MSK_SUS (Bit5)

  This is the mask register for USBINTFR1<INT_SUS>.

- MSK_RESUME (Bit4)

  This is the mask register for USBINTFR1<INT_RESUME>.

- MSK_CLKSTOP (Bit3)

  This is the mask register for USBINTFR1<INT_CLKSTOP>.

- MSK_CLKON (Bit2)

  This is the mask register for USBINTFR1<INT_CLKON>.

USBINTMR2
(07F5H)

	7	6	5	4	3	2	1	0
bit Symbol	EP1_MSK_FA	EP1_MSK_EA	EP1_MSK_FB	EP1_MSK_EB	EP2_MSK_FA	EP2_MSK_EA	EP2_MSK_FB	EP2_MSK_EB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	1	1	1	1	1	1	1	1
Function	When read 0: not masked 1: masked When write 0: Clear flag 1: –							

- EP1/2_MSK_FA/FB/EA/EB

  This is the mask register for USBINTFR2<EPx_FULL_A/B> or <EPx_Empty_A/B>.

USBINTMR3
(07F6H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_MSK_FA	EP3_MSK_EA						
Read/Write	R/W	R/W						
Reset State	1	1						
Function	When read 0: not masked 1: masked When write 0: Clear flag 1: –							

- EP3_MSK_FA/FB/EA/EB:

  This is the mask register for USBINTFR3<EP3_FULL_A> or <EP3_Empty_A>.

USBINTMR4
(07F7H)

	7	6	5	4	3	2	1	0
bit Symbol	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset State	1	1	1	1	1	1	1	
Function	When read 0: not masked When write 0: Clear flag 1: masked 1: –							

- MSK_SETUP (Bit7)

  This is the mask register for USBINTFR4<INT_SETUP>.
- MSK_EP0 (Bit6)

  This is the mask register for USBINTFR4<INT_EP0>.
- MSK_STAS (Bit5)

  This is the mask register for USBINTFR4<INT_STAS>.
- MSK_STASN (Bit4)

  This is the mask register for USBINTFR4<INT_STASN>.
- MSK_EP1N (Bit3)

  This is the mask register for USBINTFR4<INT_EP1N>.
- MSK_EP2N (Bit2)

  This is the mask register for USBINTFR4<INT_EP2N>.
- MSK_EP3N (Bit1)

  This is the mask register for USBINTFR4<INT_EP3N>.

3.10.3 UDC CORE

3.10.3.1 SFRs

The UDC CORE has the following SFRs to control the UDC and USB transceiver.

a) FIFO

Endpoint 0 to 3 FIFO register

b) Device request

bmRequestType	register	bRequest	register
wValue_L	register	wValue_H	register
wIndex_L	register	wIndex_H	register
wLength_L	register	wLength_H	register

c) Status

Current_Config	register	USB_STATE	register
StandardRequest	register	Request	register
EPx_STATUS	register		

d) Setup

EPx_BCS	register	EPx_SINGLE	register
Standard Request Mode	register	Request Mode	register
Descriptor RAM	register	PortStatus	register

e) Control

EPx_MODE	register	EOP	register
COMMAND	register	INT_ Control	register
Setup Received	register	USBREADY	register

f) Others

ADDRESS	register	DATASET	register
EPx_SIZE_L_A	register	EPx_SIZE_H_A	register
EPx_SIZE_L_B	register	EPx_SIZE_H_B	register
FRAME_L	register	FRAME_H	register
USBBUFF TEST	register		

Table 3.10.2 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
*0794H	R	EP4_STATUS
*0795H	R	EP5_STATUS
*0796H	R	EP6_STATUS
*0797H	R	EP7_STATUS
0798H	R	EP0_SIZE_L_A
0799H	R	EP1_SIZE_L_A
079AH	R	EP2_SIZE_L_A
079BH	R	EP3_SIZE_L_A
*079CH	R	EP4_SIZE_L_A
*079DH	R	EP5_SIZE_L_A
*079EH	R	EP6_SIZE_L_A
*079FH	R	EP7_SIZE_L_A
07A1H	R	EP1_SIZE_L_B
07A2H	R	EP2_SIZE_L_B
07A3H	R	EP3_SIZE_L_B
*07A4H	R	EP4_SIZE_L_B
*07A5H	R	EP5_SIZE_L_B
*07A6H	R	EP6_SIZE_L_B
*07A7H	R	EP7_SIZE_L_B
07A9H	R	EP1_SIZE_H_A
07AAH	R	EP2_SIZE_H_A
07ABH	R	EP3_SIZE_H_A
*07ACH	R	EP4_SIZE_H_A
*07ADH	R	EP5_SIZE_H_A
*07AEH	R	EP6_SIZE_H_A
*07AFH	R	EP7_SIZE_H_A
07B1H	R	EP1_SIZE_H_B
07B2H	R	EP2_SIZE_H_B
07B3H	R	EP3_SIZE_H_B
*07B4H	R	EP4_SIZE_H_B
*07B5H	R	EP5_SIZE_H_B
*07B6H	R	EP6_SIZE_H_B
*07B7H	R	EP7_SIZE_H_B

Table 3.10.3 UDC CORE SFRs (2/2)

Address	Read/Write	SFR Symbol
07C0H	R	bmRequestType
07C1H	R	bRequest
07C2H	R	wValue_L
07C3H	R	wValue_H
07C4H	R	wIndex_L
07C5H	R	wIndex_H
07C6H	R	wLength_L
07C7H	R	wLength_H
07C8H	W	Setup Received
07C9H	R	Current_Config
07CAH	R	Standard Request
07CBH	R	Request
07CCH	R	DATASET1
07CDH	R	DATASET2
07CEH	R	USB_STATE
07CFH	W	EOP
07D0H	W	COMMAND
07D1H	R/W	EPx_SINGLE1
07D3H	R/W	EPx_BCS1
07D6H	R/W	INT_Control
07D8H	R/W	Standard Request Mode
07D9H	R/W	Request Mode
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

Note: “*” is not used in the TMP92CH21.

3.10.3.2 EPx_FIFO Register (x: 0 to 3)

This register is prepared for each endpoint independently.

This is the window register from or to FIFO RAM.

In the auto bus enumeration, the request controller in UDC sets the mode, which is defined by the endpoint descriptor, for each endpoint automatically. By this means, each endpoint is automatically set to each direction.

Endpoint0 (0780H)

	7	6	5	4	3	2	1	0
bit Symbol	EP0_DATA7	EP0_DATA6	EP0_DATA5	EP0_DATA4	EP0_DATA3	EP0_DATA2	EP0_DATA1	EP0_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Endpoint1 (0781H)

	7	6	5	4	3	2	1	0
bit Symbol	EP1_DATA7	EP1_DATA6	EP1_DATA5	EP1_DATA4	EP1_DATA3	EP1_DATA2	EP1_DATA1	EP1_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Endpoint2 (0782H)

	7	6	5	4	3	2	1	0
bit Symbol	EP2_DATA7	EP2_DATA6	EP2_DATA5	EP2_DATA4	EP2_DATA3	EP2_DATA2	EP2_DATA1	EP2_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Endpoint3 (0783H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_DATA7	EP3_DATA6	EP3_DATA5	EP3_DATA4	EP3_DATA3	EP3_DATA2	EP3_DATA1	EP3_DATA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Note1: Read or write to these window registers using 1-byte load instructions only, since each register has only a 1-byte address. Do not use load instructions of 2 bytes or 4 bytes.

Note2: When it is IN-token(except isochronous transfer) and the UDC transmits 1-byte data to the host, if the CPU writes "eop" to the endpoint on a certain timing, a NULL data(0-byte data) may be transmitted.Therefore, prevent the tramsfer of 1-byte by for example introducing dummy data.

The device request that is received from the USB host is stored in the following 8-byte registers:

bmRequestType, bRequest, wValue_L, wValue_H, wIndex_L, wIndex_H, wLength_L and wLength_H. These are updated whenever a new SETUP token is received from the host.

When the UDC receives without error, INT_SETUP interrupt is asserted, meaning the new device request has been received.

There is also a request which is operated automatically by the UDC, depending on the request received.

In that case, the UDC does not assert the INT_SETUP interrupt. Any request which the UDC is currently operating can be checked by reading STANDARD_REQUEST_FLAG and REQUEST_FLAG.

3.10.3.3 bmRequestType Register

This register shows the bmRequestType field of the device request.

bmRequestType
(07C0H)

	7	6	5	4	3	2	1	0
bit Symbol	DIRECTION	REQ_TYPE1	REQ_TYPE0	RECIPIENT4	RECIPIENT3	RECIPIENT2	RECIPIENT1	RECIPIENT0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

DIRECTION (Bit7)

0: from host to device
1: from device to host

REQ_TYPE [1:0] (Bit6 to bit5)

00: Standard
01: Class
10: Vendor
11: (Reserved)

RECIPIENT [4:0] (Bit4 to bit0)

00000: Device
00001: Interface
00010: Endpoint
00011: etc.
Others: (Reserved)

3.10.3.4 bRequest Register

This register shows the bRequest field of the device request.

bRequest
(07C1H)

	7	6	5	4	3	2	1	0
bit Symbol	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

(Standard)
00000000: GET_STATUS
00000001: CLEAR_FEATURE
00000010: Reserved
00000011: SET_FEATURE
00000100: Reserved
00000101: SET_ADDRESS
00000110: GET_DESCRIPTOR
00000111: SET_DESCRIPTOR
00001000: GET_CONFIGURATION
00001001: SET_CONFIGURATION
00001010: GET_INTERFACE
00001011: SET_INTERFACE
00001100: SYNCH_FRAME

(Printer class)
00000000: GET_DEVICE_ID
00000001: GET_PORT_STATUS
00000010: SOFT_RESET

3.10.3.5 wValue Register

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

wValue_L
(07C2H)

	7	6	5	4	3	2	1	0
bit Symbol	VALUE_L7	VALUE_L6	VALUE_L5	VALUE_L4	VALUE_L3	VALUE_L2	VALUE_L1	VALUE_L0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

wValue_H
(07C3H)

	7	6	5	4	3	2	1	0
bit Symbol	VALUE_H7	VALUE_H6	VALUE_H5	VALUE_H4	VALUE_H3	VALUE_H2	VALUE_H1	VALUE_H0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

3.10.3.6 wIndex Register

There are 2 registers, the wIndex_L register and wIndex_H register. The wIndex_L register shows the lower byte of the wIndex field of the device request, and wIndex_H register shows the upper byte.

These are usually used to transfer index or offset.

wIndex_L
(07C4H)

	7	6	5	4	3	2	1	0
bit Symbol	INDEX_L7	INDEX_L6	INDEX_L5	INDEX_L4	INDEX_L3	INDEX_L2	INDEX_L1	INDEX_L0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

wIndex_H
(07C5H)

	7	6	5	4	3	2	1	0
bit Symbol	INDEX_H7	INDEX_H6	INDEX_H5	INDEX_H4	INDEX_H3	INDEX_H2	INDEX_H1	INDEX_H0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

3.10.3.7 WLength Register

There are 2 registers, the wLength_L register and wLength_H register. The wLength_L register shows the lower-byte of the wLength field of the device request, and wLength_H register shows the upper byte.

In the case of data phase, these registers show the byte number to transfer.

wLength_L
(07C6H)

	7	6	5	4	3	2	1	0
bit Symbol	LENGTH_L7	LENGTH_L6	LENGTH_L5	LENGTH_L4	LENGTH_L3	LENGTH_L2	LENGTH_L1	LENGTH_L0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

wLength_H
(07C7H)

	7	6	5	4	3	2	1	0
bit Symbol	LENGTH_H7	LENGTH_H6	LENGTH_H5	LENGTH_H4	LENGTH_H3	LENGTH_H2	LENGTH_H1	LENGTH_H0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

3.10.3.8 Setup Received Register

This register informs the UDC that an application program has recognized the INT_SETUP interrupt.

SetupReceived
(07C8H)

	7	6	5	4	3	2	1	0
bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
Read/Write	W	W	W	W	W	W	W	W
Reset State	0	0	0	0	0	0	0	0

If this register is accessed by an application program, the UDC disables access to the EP0's FIFO RAM, because the UDC recognizes the device request has been received.

This is to protect data stored in the EP0 in the time between the completion of the previous device request and the recognition by the application program of the INT_SETUP interrupt relating to a new request.

Therefore, write "00H" to this register when the device request in INT_SETUP routine is recognized.

Note : A recovery time of 2 clocks at 12MHz is needed after writing to this register in order to access EP0_FIFO.

3.10.3.9 Current_Config Register

This register shows the present value that is set by SET_CONFIGURATION and SET_INTERFACE.

Current_Config
(07C9H)

	7	6	5	4	3	2	1	0
bit Symbol	REMOTEWAKEUP		ALTERNATE[1]	ALTERNATE[0]	INTERFACE[1]	INTERFACE[0]	CONFIG[1]	CONFIG[0]
Read/Write	R		R	R	R	R	R	R
Reset State	0		0	0	0	0	0	0

CONFIG[1:0] (Bit1 to bit0)

00: UNCONFIGURED	Set to UNCONFIGURED by the host.
01: CONFIGURED1	Set to CONFIGURED 1 by the host.
10: CONFIGURED2	Set to CONFIGURED 2 by the host.

INTERFACE[1:0] (Bit3 to bit2)

00: INTERFACE0	Set to INTERFACE 0 by the host.
01: INTERFACE1	Set to INTERFACE 1 by the host.
10: INTERFACE2	Set to INTERFACE 2 by the host.

ALTERNATE[1:0] (Bit5 to bit4)

00: ALTERNATE0	Set to ALTERNATE 0 by the host.
01: ALTERNATE1	Set to ALTERNATE 1 by the host.
10: ALTERNATE2	Set to ALTERNATE 2 by the host.

REMOTE WAKEUP (Bit7)

0: Disable	Disabled remote wakeup by the host.
1: Enable	Enabled remote wakeup by the host.

Note1: CONFIG, INTERFACE and ALTERNATE each support 3 kinds (0,1 and 2).

Note2: If each request is controlled by S/W, this register is not set.

3.10.3.10 Standard Request Register

This register shows the standard request currently being executed.
Any bit which is set to “1” shows a request currently being executed.

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
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

S_INTERFACE (Bit 7) : SET_INTERFACE
G_INTERFACE (Bit 6) : GET_INTERFACE
S_CONFIG (Bit 5) : SET_CONFIGRATION
G_CONFIG (Bit 4) : GET_CONFIGRATION
G_DESCRIPT (Bit 3) : GET_DESCRIPTOR
S_FEATURE (Bit 2) : SET_FEATURE
C_FEATURE (Bit 1) : CLEAR_FEATURE
G_STATUS (Bit 0) : GET_STATUS

3.10.3.11 Request Register

This register shows the device request currently being executed.
Any bit which is set to “1” shows a request currently being executed.

Request (07CBH)

	7	6	5	4	3	2	1	0
bit Symbol		SOFT_RESET	G_PORT_STS	G_DEVICE_ID	VENDOR	CLASS	ExSTANDARD	STANDARD
Read/Write		R	R	R	R	R	R	R
Reset State		0	0	0	0	0	0	0

SOFT_RESET (Bit 6) : SOFT_RESET
G_PORT_STS (Bit 5) : GET_PORT_STATUS
G_DEVICE_ID (Bit 4) : GET_DEVICE_ID
VENDOR (Bit 3) : Vendor class request
CLASS (Bit 2) : Class request
ExSTANDARD (Bit 1) : Auto Bus Enumeration not supported
(SET_DESCRIPTOR, SYNCH_FRAME)
STANDARD (Bit 0) : Standard request

3.10.3.12 DATASET Register

This register shows whether FIFO contains data or not.

The application program can access this register to check whether FIFO contains data or not.

In receive status, when a valid data transfer from the USB host has finished, the bit which corresponds to the applicable endpoint is set to “1” and an interrupt generated. And, when the application reads the 1-packet data, this bit is cleared to “0”. In transmit status, when it has completed the 1-packet data transfer to FIFO, this bit is set to “1”. And when valid data is transferred to the USB host, this bit is cleared to “0” and an interrupt generated.

		7	6	5	4	3	2	1	0
DATASET1 (07CCH)	bit Symbol	EP3_DSET_B	EP3_DSET_A	EP2_DSET_B	EP2_DSET_A	EP1_DSET_B	EP1_DSET_A		EP0_DSET_A
	Read/Write	R	R	R	R	R	R		R
	Reset State	0	0	0	0	0	0		0

		7	6	5	4	3	2	1	0
DATASET2 (07CDH)	bit Symbol	EP7_DSET_B	EP7_DSET_A	EP6_DSET_B	EP6_DSET_A	EP5_DSET_B	EP5_DSET_A	EP4_DSET_B	EP4_DSET_A
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	0	0	0	0

Note: DATASET1<EP3_DSET_B>, DATASET2 registers are not used in the TMP92CH21.

- Single packet mode
  (DATASET1: Bit0, bit2, bit4 and bit6 DATASET2: Bit0, bit2, bit4 and bit6)

  These bits show whether FIFO of the corresponding endpoint has data or not.

  In receive mode endpoint, if the corresponding endpoint bit is “1”, FIFO contains data to be read. Access EPx_SIZE register, determine the size of the data that should be read, and read data of this size. When this bit is “0”, there is no data to be read.

  In transmit mode endpoint, if the corresponding endpoint bit is “0”, the CPU can transfer data under the FIFO payload. If this bit is “1”, because FIFO has transfer data waiting, transfer data to FIFO from UDC after the corresponding bit has been cleared to “0”. When a short-packet is transferred, access EOP register after writing transmission data to the corresponding endpoint.
- Dual packet mode
  (DATASET1: Bit3, bit5 and bit7 DATASET2: Bit1, bit3 bit5 and bit7)

  These bits become effective in the dual packet mode. FIFO has 2-packets in this mode.

  Each packet (packet-A and packet-B) has its own DATASET-bit.

  Unlike as in the case above, in isochronous transfer, this shows the packet that can access the current frame. In this case, whether bit A or B is set to “1”, it is renewed according to the shifting frame.

Note1: In receive mode, if the endpoint bits corresponding to packet-A or packet-Bare “1”, read the required packet-number data after checking EPx_SIZE<PKT_ACTIVE>.

Note2: In transmit mode, if both A and B bits are not “1”, this means there is space in FIFO. So, write data of payload or less to FIFO. If the transmission is short-packet, write “0” to EOP<EPn_EOPB> after writing data to the FIFO. The maximum size that can be written to A or B packet is the same as the maximum payload size. If both A and B bits are “0”, continuous writing of double maximum payload size is available.

Note3: In dual packet transmit mode, if both A and B packet are empty and EOP<EPn_EOPB> is written “0”, the NULL-data is set to FIFO. In single mode, the NULL-data is also set to FIFO if the above operation is executed when packet-A contains no data.

Note4: No data is set in this register when NULL-packet (0Length-packet) is received.

3.10.3.13 EPx_STATUS Register (x: 0 to 7)

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

		7	6	5	4	3	2	1	0
EP0_STATUS (0790H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP1_STATUS (0791H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP2_STATUS (0792H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP3_STATUS (0793H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP4_STATUS (0794H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP5_STATUS (0795H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP6_STATUS (0796H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

		7	6	5	4	3	2	1	0
EP7_STATUS (0797H)	bit Symbol		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	1	1	1	0	0

Note: EP4, 5, 6 and 7_STATUS registers are not used in the TMP92CH21.

TOGGLE Bit (Bit6) This bit shows status of toggle sequence bit.

0: TOGGLE	Bit0
1: TOGGLE	Bit1

SUSPEND (Bit5) This bit shows status of UDC power management.
In the SUSPEND status, access to UDC is limited.
For details, refer to 3.10.9.

0: RESUME
1: SUSPEND

STATUS [2:0]
(Bit4 to bit2)

These bits show status of UDC endpoint.

The status shows whether transfer is possible or not, and the results of the transfer. . These depend on transfer type.

(For the Isochronous transfer type, refer to 3.10.6.)

000: READY	Receiving:	Device can be received. In endpoints 1 to 7, this register is initialized to “READY” by setting transfer type at SET_CONFIGURATION. In endpoint 0, this register is initialized to “READY” by detecting USB reset from the host. This is initialized to “READY” by terminating the status stage without error.
	Transmitting:	Basically, the same as “Receiving”. But in transmitting, when data for transmission is set to FIFO and answer to token from host and transfer data to host collect and received ACK, status register does not change, and it remains “READY”. In this case, EPx_Empty_A or EPx_Empty_B interrupt terminate the transfer correctly.
001: DATAIN		UDC set to DATAIN and generates EPx_FULL_A or EPx_FULL_B interrupt when data is received from the host without error.
010: FULL		Refer to 3.10.8 (2) Details for the STATUS register.
011: TX_ERR		After transfer of data to IN token from host, UDC sets TX-ER to status register when “ACK” is not received from host. In this case, an interrupt is not generated. The hosts re-try IN token transfer.
100: RX_ERR		UDC sets RX_ERR to status register without transmitting “ACK” to host when an error (such as a CRC-error) is detected in data of received token. In this case, an interrupt is not generated. The hosts re-try IN token transfer.
101: BUSY		This status is used only for the control transfer type and it is set when a status-stage token is received from the host after a terminated data-stage. When status-stage can be finished, terminates correctly and returns to READY. This is not used in the Bulk and interrupts transfer type.
110: STALL		This status shows that the corresponding endpoint is in STALL status. In this status, STALL-handshake returns, except for SETUP-token. The control endpoint returns to READY from stall condition when SETUP-token is received. Other endpoints return to READY when initialization command of FIFO is received. (Note) With Automatic Set_Interface request answer, requests to interface 4 to 6 may not become request errors. If this is a problem, in Set_Interface request answer, set Standard Request Mode <S_INTERFACE> to “1” and use software.
111: INVALID		This status shows that the corresponding endpoint is in UNCONFIGURED status. In this status, the UDC has no effect when a token is received from the host. On reset, all endpoints are set to INVALID status. Only endpoint 0 returns to READY on receiving USB-reset. Corresponding endpoints return to READY according to configuration.

FIFO_DISABLE (Bit1)

0: FIFO enabled
1: FIFO disabled

This bit symbol shows FIFO status, except for EP0.

If the FIFO is set to disabled, the UDC transmits NAK handshake for all transfers. Disabled or enabled status is set by the COMMAND register. This bit is cleared to "0" when transfer type is changed.

STAGE_ERROR (Bit0)

0: SUCCESS
1: ERROR

This bit symbol shows that the status stage has not been terminated correctly. ERROR is set when a status stage is not terminated correctly and a new SETUP token is received.

When this bit is "1", this bit is cleared to "0" by read EP0_STATUS register. This bit is not cleared even if normal control transfer or other transfer is executed after it. To clear, read this bit. When software transaction is finished and UDC writes EOP register, UDC shifts to status register and waits for termination of status stage. In this case, if software is needed to confirm that the status stage has been terminated correctly, when a new request flag is received, it is possible to confirm whether or not the last request was terminated correctly. It can also be confirmed, when a new request flag is asserted, whether or nor the last request was cancelled before completion.

3.10.3.14 EPx_SIZE Register (x: 0 to 7)

These registers have the following functions.

a) In receive mode, showing the 1-packet data number which was received correctly.

b) In transmit mode, showing payload size. Showing length value when short packet is transferred.
It is not necessary to read this register when it is transmitting.

c) Showing dual packet mode and currently effective packet.

Each endpoint has an H (High)-register that shows upper bit 9 to bit7 of data size, and an L (Low) register which shows lower bit 6 to bit0 and control bit of FIFO.

Each H/L register also has 2-set for dual-packet mode.

		7	6	5	4	3	2	1	0
EP0_SIZE_L_A (0798H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP1_SIZE_L_A (0799H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_L_A (079AH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_L_A (079BH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_L_A (079CH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_L_A (079DH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_L_A (079EH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_L_A (079FH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0

Note EP4,5,6,7_SIZE_L_A registers are not used in the TMP92CH21.

		7	6	5	4	3	2	1	0
EP1_SIZE_L_B (07A1H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP2_SIZE_L_B (07A2H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP3_SIZE_L_B (07A3H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP4_SIZE_L_B (07A4H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP5_SIZE_L_B (07A5H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP6_SIZE_L_B (07A6H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP7_SIZE_L_B (07A7H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

Note: EP3,4,5,6,7_SIZE_L_B registers are not used in the TMP92CH21.

		7	6	5	4	3	2	1	0
EP1_SIZE_H_A (07A9H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP2_SIZE_H_A (07AAH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP3_SIZE_H_A (07ABH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP4_SIZE_H_A (07ACH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP5_SIZE_H_A (07ADH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP6_SIZE_H_A (07AEH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP7_SIZE_H_A (07AFH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

Note: EP4,5,6,7_SIZE_H_A registers are not used in the TMP92CH21.

		7	6	5	4	3	2	1	0
EP1_SIZE_H_B (07B1H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP2_SIZE_H_B (07B2H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP3_SIZE_H_B (07B3H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP4_SIZE_H_B (07B4H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP5_SIZE_H_B (07B5H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP6_SIZE_H_B (07B6H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

		7	6	5	4	3	2	1	0
EP7_SIZE_H_B (07B7H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

Note: EP3,4,5,6,7_SIZE_H_B registers are not used in the TMP92CH21.

DATASIZE[9:7] (H register: Bit2 to bit0)

DATASIZE[6:0] (L register: Bit6 to bit0)

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

By setting EPx_MODE register, these bits are initialized to MAX pay load size in bulk/interrupt transfer, and "0" in isochronous transfer.

PKT_ACTIVE (L register: Bit7)

1: OUT_ENABLE

0: OUT_DISABLE

When dual-packet mode is selected, this bit shows the packet that can be accessed. In this case, the UDC accesses packets that divide FIFO (Packet A and Packet B) mutually. When FIFO in UDC is accessed by CPU, refer to this bit. If receiving endpoint, start reading from that packet that this bit is "1". In single-packet mode, this bit has no effect because packet-A is always used.

3.10.3.15 FRAME Register

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

Each HIGH and LOW register shows upper and lower bits.

FRAME_L (07E1H)

	7	6	5	4	3	2	1	0
bit Symbol	–	T[6]	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

FRAME_H (07E2H)

	7	6	5	4	3	2	1	0
bit Symbol	T[10]	T[9]	T[8]	T[7]		CREATE	FRAME_STS1	FRAME_STS0
Read/Write	R	R	R	R		R	R	R
Reset State	0	0	0	0		0	1	0

T[10:7] (H register: Bit7 to bit4)

T[6:0] (L register: Bit6 to bit0) These bits are renewed when SOF-token is received. They also show the frame-number.

CREATE (H register: Bit2)

0: DISABLE

1: ENABLE

These bits show whether the function that generates SOF automatically from the UDC is enabled or not. This is used in case of error in receiving SOF token.

This function is set by accessing COMMAND register.

On reset, this bit is initialized to “0”.

FRAME STS[1:0]

(H register: Bit1 and bit0)

0: BEFORE

1: VALID

2: LOST

These bits show the status whether a frame number that is shown in the FRAME register is correct or not. At the LOST status, a correct frame number is undefined.

If this register is “VALID”, the number that is shown to the FRAME register is correct.

If this register is “BEFORE”, during SOF auto generation, BEFORE condition shows it from USB host controller inside that from SOF generation time to reception of SOF token. Correct frame-number value is the value that is selected from FRAME register value.

3.10.3.16 ADDRESS Register

This register shows the device address which is specified by the host in bus enumeration.

By reading this register, the present address can be confirmed.

ADDRESS (07E3H)

	7	6	5	4	3	2	1	0
bit Symbol		A6	A5	A4	A3	A2	A1	A0
Read/Write		R	R	R	R	R	R	R
Reset State		0	0	0	0	0	0	0

ADDRESS [6:0] (Bit6 to bit0) The UDC compares this registers and address in all packet ID, and UDC judges whether it is an effective transaction or not.

This is initialized to “00H” by USB reset.

3.10.3.17 EOP Register

This register is used when a control transfer type dataphase terminates or when a short packet is transmitting bulk-IN or interrupt-IN.

EOP
(07CFH)

	7	6	5	4	3	2	1	0
bit Symbol	EP7_EOPB	EP6_EOPB	EP5_EOPB	EP4_EOPB	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
Read/Write	W	W	W	W	W	W	W	W
Reset State	1	1	1	1	1	1	1	1

Note1: EOP<EP7_EOPB, EP6_EOPB, EP5_EOPB, EP4_EOPB> registers are not used in the TMP92CH21.

Note2: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

In a control transfer type dataphase, write “0” to <EP0_EOPB> when all transmission data is written to the FIFO, or read all receiving data from the FIFO. The UDC terminates its status stage on this signal.

When a short packet is transmitted by using bulk-IN or interrupt-IN endpoint, use this to terminate writing of transmission data. In this case, write “0” to <EP0_EOPB> of writing endpoint. Write “1” to other bits.

3.10.3.18 Port Status Register

This register is used when a printer class request is received.

In the case of a GET_PORT_STATUS request, the UDC operates automatically using this data.

Port Status (07E0H)		7	6	5	4	3	2	1	0
	bit Symbol	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
	Read/Write	W	W	W	W	W	W	W	W
	Reset State	0	0	0	1	1	0	0	0

Note: The TMP92CH21 does not use this register since it does not support printer-class.

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

3.10.3.19 Standard Request Mode Register

This register sets the answer for Standard Request, either answering automatically in hardware, or by control through software. Each bit represents a kind of request.

When the relevant bit in this register is set to “0”, the answer is executed automatically by hardware. When the relevant bit in this register is set to “1”, the answer is controlled by software. If a request is received during hardware control, the interrupt signal (INT_SETUP, INT_ENDPOINT0, INT_STATUS, INT_STATUSNAK) is set to disable. If a request is received during software control, the interrupt signal is asserted, and it is controlled by software.

Note: With Automatic Set_Interface request answer, requests to interface 4 to 6 may not become request errors. If this is a problem, in Set_Interface request answer, set Standard Request Mode <S_INTERFACE> to “1” and use software.

Standard Request Mode (07D8H)		7	6	5	4	3	2	1	0
	bit Symbol	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	0	0	0	0	0	0	0	0

S_Intetface	(Bit 7) : SET_INTERFACE
G_Interface	(Bit 6) : GET_INTERFACE
S_Config	(Bit 5) : SET_CONFIGRATION
G_Config	(Bit 4) : GET_CONFIGRATION
G_Descript	(Bit 3) : GET_DESCRIPTOR
S_Feature	(Bit 2) : SET_FEATURE
C_Feature	(Bit 1) : CLEAR_FEATURE
G_Status	(Bit 0) : GET_STATUS

3.10.3.20 Request Mode Register

This register sets the answer for Class Request either automatically in hardware or by control through software. Each bit represents a kind of request.

When relevant bit in this register is set to “0”, the answer is executed automatically by hardware. When relevant bit in this register is set to “1”, the answer is controlled by software. If a request is received during hardware control, the interrupt signal (INT_SETUP, INT_ENDPOINT0, INT_STATUS, INT_STATUSNAK) is set to disable. If a request is received during software control, the interrupt signal is asserted, and it is controlled by software.

		7	6	5	4	3	2	1	0
Request Mode (07D9H)	bit Symbol		Soft_Reset	G_Port_Sts	G_DeviceId				
	Read/Write		R/W	R/W	R/W				
	Reset State		0	0	0				

Note: The TMP92CH21 does not use this register since it does not support printer-class.

–	(Bit 7)	: Reserved
Soft_Reset	(Bit 6)	: SOFT_RESET
G_Port_Sts	(Bit 5)	: GET_PORT_STATUS
G_Config	(Bit 4)	: GET_DEVICE_ID
G_Descript	(Bit 3 to 0)	: Reserved

Note1: SET_ADDRESS request is supported only by auto-answer .

Note2: SET_DESCRIPTOR and SYNCH_FRAME are controlled only by software .

Note3: Vendor Request and Class Request (Printer Class and so on) are controlled only by software.

Note4: INT_SETUP, ENDPOINT0, STATUS and STATUSNAK interrupts assert only when it is software-control.

3.10.3.21 COMMAND Register

This register sets COMMAND at each endpoint. This register can be set to select endpoint in bit6 to bit4 and kind of COMMAND in bit3 to bit0.

COMMAND for endpoint that is supported is ignored.

COMMAND (07D0H)

	7	6	5	4	3	2	1	0
bit Symbol		EP[2]	EP[1]	EP[0]	Command[3]	Command[2]	Command[1]	Command[0]
Read/Write		W	W	W	W	W	W	W
Reset State		0	0	0	0	0	0	0

Note: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

EP [2:0] (Bit6 to bit4)

000: Select endpoint 0

001: Select endpoint 1

010: Select endpoint 2

011: Select endpoint 3

COMMAND [3:0] (Bit3 to bit0)

0000: Reserved

0001: Reserved

0010: SET_DATA0 This COMMAND clear toggle sequence bit of corresponding endpoint (EP0 to EP3).

If this COMMAND is input, it sets toggle sequence bit of the corresponding endpoint to "0". Data toggle for transfer is renewed automatically by UDC. However, this COMMAND execution is required if setting toggle sequence bit of endpoint to "0",. If control transfer type and Isochronous transfer type, execution of this COMMAND is not required because of hardware control.

0011: RESET This COMMAND resets the corresponding endpoint (EP0 to EP3).

If this COMMAND is input, the corresponding endpoint is initialized. CLEAR_FEATURE request stalls endpoint. When this stall is cleared, execute this COMMAND. (This command does not affect transfer mode.)

This command initializes the following:.

・Clear toggle sequence bit of corresponding endpoint.

・Clear STALL of corresponding endpoint.

・Set to FIFO_ENABLE condition.

・Clear the data in FIFO

0100: STALL This COMMAND sets corresponding endpoint to STALL (EP0 to EP3).

If STALL handshake must be return as answer for device request, execute this command.

0101: INVALID This COMMAND sets condition to prohibition of use of corresponding endpoint (EP1 to EP3).

If UDC detects USB_RESET signal from USB host, it sets all endpoints (except endpoint 0) to prohibition using it automatically. If Config and Interface are changed by device request, set endpoint that is not used to prohibit use.

Note: If setting endpoint that is set to Isochronous transfer mode to "no use", after change to Isochronous mode, set to "no use" by COMMAND register.

0110: CREATE_SOF This COMMAND sets quasi-SOF generation function to enable (EP0).

Default is set to disable, it must be used for Isochronous transfer.

0111: FIFO_DISABLE This COMMAND sets FIFO of corresponding endpoint to disable (EP1 to EP3).

If this command is set externally, all transfers for corresponding endpoint return NAK. When it is set externally while receiving packet, this becomes valid from next token. This command does not affect the packet that is transferring.

1000: FIFO_ENABLE This COMMAND sets FIFO of corresponding endpoint to enable (EP1 to EP3).

If FIFO is set to disable by FIFO_DISABLE COMMAND, this command is used for release of disable condition. If set while receiving packet, this becomes valid from next token. If USB_RESET is detected from host and RESET COMMAND execute and transfer mode is set by using SET_CONFIG and SET_INTERFACE request, the corresponding endpoint enters FIFO_ENABLE condition.

1001: INIT_DESCRIPTOR This COMMAND is used if descriptor RAM is rewritten during system operation (EP0).

If UDC detects USB_RESET from host controller, it reads content of descriptor RAM automatically, and it performs relevant settings.

If descriptor RAM is changed during system operation, it must read setting again. Therefore, execute this command. When connected to USB host, this function starts reading automatically. Therefore, in this case, it is not necessary to execute this command.

1010: FIFO_CLEAR This COMMAND initializes FIFO of corresponding endpoint (EP1 to EP3).

However, EPx_STATUS<TOGGLE> is not initialized.

If resetting by software, execute this COMMAND.

This command iInitializes the following:

・Clear STALL of relevant endpoint.

・Set to FIFO_ENABLE condition.

・Clear the data in FIFO

1011: STAL_CLEAR This COMMAND clear STALL of corresponding endpoint (EP1 to EP3).

If clearing only STALL of endpoint, execute this COMMAND.

3.10.3.22 INT_Control Register

INT_STATUS_NAK interrupt is disabled and enabled by the value that is written to this register.

This is initialized to disable by external reset. When setup packet is received, it becomes disabled.

INT_Control (07D6H)

	7	6	5	4	3	2	1	0
bit Symbol								Status_nak
Read/Write								R/W
Reset State								0

In control read transfer, if the host terminates a dataphase with small data length (smaller than the data length that is specified by the host as wLength), the device side and stage management cannot be synchronized. Therefore, INT_STATUSNAK interrupt signals this shift to status stage. If needed, set to "1" after receiving setup packet.

STATUS_NAK (Bit0)

0: INT_STATUS_NAK interrupt disable
1: INT_STATUS_NAK interrupt enable

3.10.3.23 USB STATE Register

This register shows the current device state for connection with USB host.

USB STATE (07CEH)

	7	6	5	4	3	2	1	0
bit Symbol						Configured	Addressed	Default
Read/Write						R/W	R	R
Reset State						0	0	1

Note: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

Inside the UDC, the answer for each Device Request is managed by referring to these bits (Configured, Addressed and Default). If transaction for SET_CONFIG request is executed using software, write the present state to this register. If host appointconfig is 0, this becomes Unconfigured, and it is necessary to return to Addressed state. Therefore, if host appoint config is 0, write "0" to bit2.

When Configured bit (Bit2) is written "0", Addressed bit (bit 1) is set automatically by hardware. When host appoint config value that supported by device, the device must execute mode setting for each endpoint by using the value that is appointed by endpoint-descriptor in the config-descriptor. After finish mode setting, set Configured bit (Bit2) to "1" before accessing EOP register. When this bit is set to "1", Addressed bit (Bit1) is set to "0" automatically.

Bit2 to bit0

000: Default
010: Addressed
100: Configured

3.10.3.24 EPx_MODE Register (x: 1 to 3)

This register sets transfer mode of endpoint (EP1 to EP3).

If SET_CONFIG and SET_INTERFACE processing is set to software control, this control must use appointed config or interface. Access this register to set mode.

		7	6	5	4	3	2	1	0
EP1_MODE (0789H)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0
		7	6	5	4	3	2	1	0
EP2_MODE (078AH)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0
		7	6	5	4	3	2	1	0
EP3_MODE (078BH)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0

There is a limitation to the timing that can be written.

If SET_CONFIG and SET_INTERFACE processing is set to software control, after INT_SETUP interrupt is received, finish writing before accessing EOP register. This register prohibits writing when it is timing, and it is ignored.

Note1: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

Note2: When writing to this register, endpoint is initialized same as RESET of COMMAND register.

DIRECTION (Bit0)

0: OUT	Direction from host to device
1: IN	Direction from device to host

MODE [1:0] (Bit1 and bit2)

00: Control transfer type
01: Isochronous transfer type
10: Bulk transfer type or interrupt transfer type
11: Interrupt (No toggle)

Note: If setting endpoint that is set to Isochronous transfer mode to “no use”, after changing to Isochronous mode, set to “no use” by COMMAND register.

PAYLOAD [2:0] (Bit3, bit4 and bit5)

000:	8 bytes
001:	16 bytes
010:	32 bytes
011:	64 bytes
0100:	128 bytes
0101:	256 bytes
0110:	512 bytes
0111:	1023 bytes (Note1, 2)

Note3: Max packet size of Isochronous transfer type is 1023 bytes.

Note4: IfwMaxPacketSize of descriptor was set to other than 8, 16, ..., 1023, Payload more than descriptor value is set by auto-answer of Set_Configration and Set_Interface.

Others (Bit6 and bit7) Reserved

3.10.3.25 EPx_SINGLE Register

This register sets mode of FIFO in each endpoint (SINGLE/DUAL).

EPx_SINGLE1
(07D1H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	
Read/Write	R/W	R/W	R/W		R/W	R/W	R/W	
Reset State	0	0	0		0	0	0	

Note: Endpoint 3 supports only SINGLE mode in the TMP92CH21.

Bit number
0: No use
1: EP1_SINGLE
2: EP2_SINGLE
3: EP3_SINGLE
4: No use
5: EP1_SELECT
6: EP2_SELECT
7: EP3_SELECT

When EPx_SELECT bit is “1”, EPx_SINGLE bit becomes valid in the following content.
0: DUAL mode 1: SINGLE mode
If setting content of EPx_SINGLE bit to valid, set EPx_SELECT bit to “1”.
0: Invalid 1: Valid

3.10.3.26 EPx_BCS Register

This register sets mode of access to FIFO in each endpoint.

EPx_BCS1
(07D3H)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_BCS	EP2_BCS	EP1_BCS	
Read/Write	R/W	R/W	R/W		R/W	R/W	R/W	
Reset State	0	0	0		0	0	0	

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

Always write “1” to EPx_BCS bit regardless of whether endpoint is used or not.
0: Reserved 1: CPU access
If setting content of EPx_BCS bit to valid, set EPx_SELECT bit to “1”.
0: Invalid 1: Valid

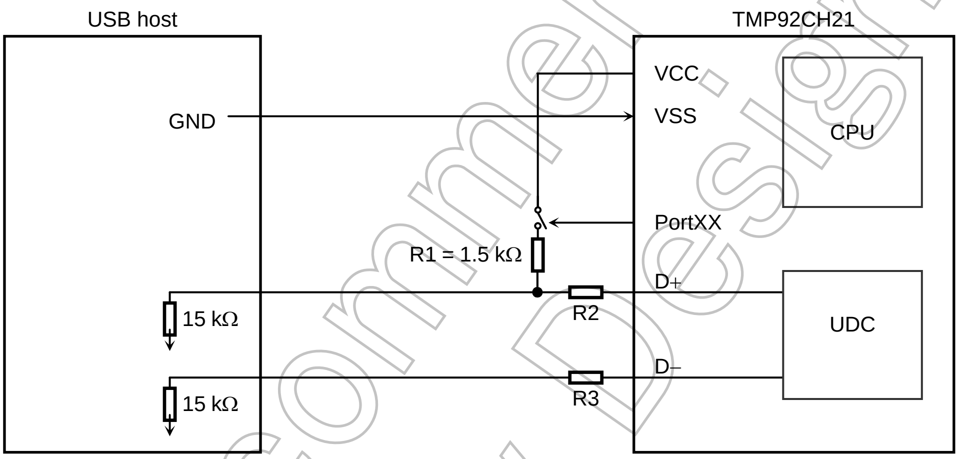
3.10.3.27 USBREADY Register

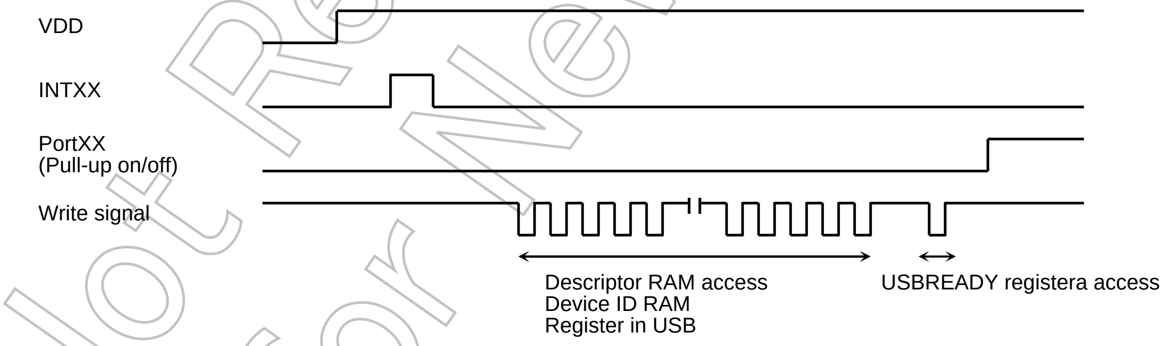
This register informs finishing writing data to descriptor RAM on UDC.
After assigned data to descriptor RAM, write “0” to bit0.

USBREADY
(07E6H)

	7	6	5	4	3	2	1	0
bit Symbol								USBREADY
Read/Write								R/W
Reset State								0

USBREADY (Bit0)
0: Writing to descriptor RAM has finished.
1: Writing to descriptor RAM is enabled.
(However, writing to descriptor RAM is prohibited when connected to host.)





Detect level of VDD signal from USB cable, and execute initialize sequence. In this case, UDC disable detecting USB_RESET signal until USBREADY register is written “0” after release of USB_RESET.

If the pull-up resistor on D+ signal is controlled by control signal, when pull-up resistor is connected to host in OFF condition, this condition is equivalent condition with USB_RESET signal by pull-down resistor on the host side. Therefore UDC is not detected in USB_RESET until “0” is written to USBREADY register

Note1: External pull-up resistor and control switch are needed with the TMP92CH21.

Note2: The above setting is an example for communication. A specific circuit is required to prevent current flow at connector detection , no-use, and no connection.

3.10.3.28 Set Descriptor STALL Register

This register sets whether returns STALL automatically in data stage or status stage for Set Descriptor Request.

Set Descriptor STALL
(07E8H)

	7	6	5	4	3	2	1	0
bit Symbol								S_D_STALL
Read/Write								W
Reset State								0

Bit0: S_D_STALL

0: Software control (Default)

1: Automatically STALL

3.10.3.29 Descriptor RAM

This register is used for store descriptor to RAM. The size of the descriptor is 384 bytes. However, when storing descriptor, write according to descriptor RAM structure sample.

Descriptor RAM
(0500H)
~
(067FH)

	7	6	5	4	3	2	1	0
bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Read/Write timing is only possible before detection of USB_RESET or during processing of SET_DESCRIPTOR request.

SET_DESCRIPTOR request processes from INT_SETUP assert until access of EOP register.

If there is rewriting request of descriptor in SET_DESCRIPTOR, process the request in the following sequence.

1) Read every packet of the descriptor that is transferred by SET_DESCRIPTOR .
2) When reading descriptor number of last packet finished, write all descriptors to RAM for descriptor.
3) When writing is completed, execute INIT_DESCRIPTOR of COMMAND register.
4) When all the process is completed, access EOP register, and finish status stage.
5) When INT_STATUS is received, it shows normal finish of status stage.

If USB_RESET is detected, it starts reading automatically. Therefore, when it connects to the host, executing INIT_DESCRIPTOR command is not necessary.

3.10.4 Descriptor RAM

This area stores the descriptor that is defined in USB. Device, Config, Interface, Endpoint and String descriptor must set to RAM using the following format.

Device descriptor	18 bytes
Config 1 descriptor (Interfaces, endpoints)	255 bytes or less
Config 2 descriptor (Interfaces, ENDPOINT)	255 bytes or less
String0 length	1 byte
String1 length	1 byte
String2 length	1 byte
String3 length	1 byte
String0 descriptor	63 bytes or less
String1 descriptor	63 bytes or less
String2 descriptor	63 bytes or less
String3 descriptor	63 bytes or less

Note 1: If String Descriptor is supported, set StringxLength area to size0. No support String Dedcriptor is returned STALL.

Note 2: Config Descriptior refers to descriptor sample.

Note 3: Sequencer in UDC determines Config number, Interface number and Endpoint number. Therefore, if supporting Endpoint number is small, assign address according to priority.

Note 4: This function become effective only in case of store descriptor as RAM.

Note 5: RAM size is total 384 bytes.

Note 6: Possible timing in RD/WR of descriptor RAM is only before detection of USB_RESET and processing of SET_DESCRIPTOR request. (Prohibit access other than this timing.)
Writing must finish before connection to USB host and processing of SET_DESCRIPTOR request.
SET_DESCRIPTOR request processes from INT_SETUP assert until access of EOP register.

Note 7: The class descriptor, and the vender descriptors, etc except a standard descriptor cannnot be supported by an auto bus enumeration.

Descriptor RAM setting example:

Address	Data	Description	Description
Device Descriptor			
500H	12H	bLength	
501H	01H	bDescriptorType	Device Descriptor
502H	00H	bcdUSB (L)	USB Spec 1.00
503H	01H	bcdUSB (H)	IFC's specify own
504H	00H	bDeviceClass	
505H	00H	bDeviceSubClass	
506H	00H	bDeviceProtocol	
507H	08H	bMaxPacketSize0	
508H	6CH	bVendor (L)	Toshiba
509H	04H	bVendor (H)	
50AH	01H	IdProduct (L)	
50BH	10H	IdProduct (H)	
50CH	00H	bcdDevice (L)	Release 1.00
50DH	01H	bcdDevice (H)	
50EH	00H	bManufacture	
50FH	00H	IProduct	
510H	00H	bSerialNumber	
511H	01H	bNumConfiguration	
Config1 Descriptor			
512H	09H	BLength	
513H	02H	bDescriptorType	Config Descriptor
514H	4EH	wtotalLength (L)	78 bytes
515H	00H	wtotalLength (H)	
516H	01H	bNumInterfaces	
517H	01H	bConfigurationValue	
518H	00H	iConfiguration	
519H	A0H	bmAttributes	Bus-powered remote wakeup
51AH	31H	MaxPower	98 mA
Interface0 Descriptor AlternateSetting0			
51BH	09H	bLength	
51CH	04H	bDescriptorType	Interface Descriptor
51DH	00H	bInterfaceNumber	
51EH	00H	bAlternateSetting	AlternateSetting0
51FH	01H	bNumEndpoint	
520H	07H	bInterfaceClass	
521H	01H	bInterfaceSubClass	
522H	01H	bInterfaceProtocol	
523H	00H	interface	
Endpoint1 Descriptor			
524H	07H	bLength	
525H	05H	bDescriptorType	Endpoint Descriptor
526H	01H	bEndpointAddress	OUT
527H	02H	bmAttributes	BULK
528H	40H	wMaxPacketSize (L)	64 bytes
529H	00H	wMaxPacketSize (H)	
52AH	00H	bInterval	

Address	Data	Description	Description
Interface0 Descriptor AlternateSetting1			
52BH	09H	bLength	
52CH	04H	bDescriptorType	Interface Descriptor
52DH	00H	bInterfaceNumber	
52EH	01H	bAlternateSetting	AlternateSetting1
52FH	02H	bNumEndpoints	
530H	07H	bInterfaceClass	
531H	01H	bInterfaceSubClass	
532H	02H	bInterfaceProtocol	
533H	00H	iInterface	
Endoint1 Descriptor			
534H	07H	bLength	
535H	05H	bDescriptorType	Endpoint Descriptor
536H	01H	bEndpointAddress	OUT
537H	02H	bmAttributes	BULK
538H	40H	wMaxPacketSize (L)	64 bytes
539H	00H	wMaxPacketSize (H)	
53AH	00H	bInterval	
Endpoint2 Descriptor			
53BH	07H	bLength	
53CH	05H	bDescriptorType	Endpoint Descriptor
53DH	82H	bEndpointAddress	IN
53EH	02H	bmAttributes	BULK
53FH	40H	wMaxPacketSize (L)	64 bytes
540H	00H	wMaxPacketSize (H)	
541H	00H	bInterval	
Interface0 Descriptor AlternateSetting2			
542H	09H	bLength	
543H	04H	bDescriptorType	Interface Descriptor
544H	00H	bInterfaceNumber	
545H	02H	bAlternateSetting	AlternateSetting2
546H	03H	bNumEndpoints	
547H	FFH	bInterfaceClass	
548H	00H	bInterfaceSubClass	
549H	FFH	bInterfaceProtocol	
54AH	00H	iInterface	
Endpoint1 Descriptor			
54BH	07H	bLength	
54CH	05H	bDescriptorType	Endpoint Descriptor
54DH	01H	bEndpointAddress	OUT
54EH	02H	bmAttributes	BULK
54FH	40H	wMaxPacketSize (L)	64 bytes
550H	00H	wMaxPacketSize (H)	
551H	00H	bInterval	
Endpoint2 Descriptor			
552H	07H	bLength	
553H	05H	bDescriptorType	Endpoint Descriptor
554H	82H	bEndpointAddress	IN
555H	02H	bmAttributes	BULK
556H	40H	wMaxPacketSize (L)	64 bytes
557H	00H	wMaxPacketSize (H)	
558H	00H	bInterval	

Address	DATA	Description	Description
Endpoint3 Descriptor			
559H	07H	bLength	
55AH	05H	bDescriptorType	Endpoint Descriptor
55BH	83H	bEndpointAddress	IN
55CH	03H	bmAttributes	Interrupt
55DH	08H	wMaxPacketSize (L)	8 bytes
55EH	00H	wMaxPacketSize (H)	
55FH	01H	bInterval	1 ms
String Descriptor Length Setup Area			
560H	04H	bLength	Length of String Descriptor0
561H	10H	bLength	Length of String Descriptor1
562H	00H	bLength	Length of String Descriptor2
563H	00H	bLength	Length of String Descriptor3
String Descriptor0			
564H	04H	bLength	
565H	03H	bDescriptorType	String Descriptor
566H	09H	bString	Language ID 0x0409
567H	04H	bString	
String Descriptor1			
568H	10H	bLength	
569H	03H	bDescriptorType	String Descriptor
56AH	00H	bString	(Toshiba)
56BH	54H	bString	T
56CH	00H	bString	
56DH	6FH	bString	o
56EH	00H	bString	
56FH	73H	bStrIng	s
570H	00H	bString	
571H	68H	bString	h
572H	00H	bString	
573H	69H	bString	i
574H	00H	bString	
575H	62H	bStrIng	b
576H	00H	bString	
577H	61H	bString	a
String Descriptor2			
String Descriptor3			

3.10.5 Device Request

3.10.5.1 Standard request

UDC support automatically answers in standard request.

(1) GET_STATUS Request

This request automatically returns to status that is determined by receive side.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B 10000001B 10000010B	GET_STATUS	0	0 Interface endpoint	2	Device, interface or endpoint status

Request to device returns according to priority of little endian as follows.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote wakeup	Self power
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- Remote wakeup — Reinstates current remote wakeup setting. This bit is set or reset by SET_FEATURE or CLEAR_FEATURE request. Default is “0”.

- Self power — Reinstates current power supply setting. This bit returns Self or BusPower according to value that is set to bmAttributes field in Config descriptor.

Request to interface returns 00H of 2 bytes.

Request to endpoint returns according to priority of little endian as follows.

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 — Returns to halt status of selected endpoint.

(2) CLEAR_FEATURE request

This request clears or disables the relevant function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	CLEAR_ FEATURE	Feature selector	0 Interface endpoint	0	None

- Reception side device

  Feature selector: 1 Present remote wakeup setting is disabled.

  Feature selector: except 1 STALL state
- Reception side interface

  STALL state
- Reception side end point

  Feature selector: 0 Halt of relevant endpoint is cleared.

  Note: When cleared HALT state, following is set.

  ･Initialize FIFO

  ･Clear the toggle sequence bit

  ･Clear STALL state

  Feature selector: except 0 STALL state

Note: Stalls if request is to non-existent endpoint.

(3) SET_FEATURE request

This request sets or enables the relevant function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	SET_ FEATURE	Feature selector	0 Interface endpoint	0	None

- Reception side device

  Feature selector: 1 Present remote wakeup setting is disabled.

  Feature selector: except 1 STALL state
- Reception side interface

  STALL state
- Reception side end point

  Feature selector: 0 Halt of relevant endpoint

  Feature selector: except 0 STALL state

Note: Stalls if request is to non-existent endpoint.

(4) SET_ADDRESS request

This request sets the device address. Answer subsequent requests using this device address.

Answer requests using the current device address until the status stage of this request is terminated normally.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ADDRESS	Device Address	0	0	None

(5) GET_DESCRIPTOR request

This request transmits appointed descriptor.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ DESCRIPTOR	Descriptor type and Descriptor index	0 or Language ID	Descriptor length	Descriptor

- Device Device transmits device descriptor that is stored in descriptor RAM.

  There is an IdProductm bcdDevice field as register in this device. When descriptor ROM is used, if the descriptor data is determined as ROM, this area only can be rewritten. Access this register before connecting to USB host.

- Config Config transmits config descriptor that is stored in descriptor RAM.

  At this point, it transmits not only config descriptor but also interface and endpoint descriptor.

- String String transmits string descriptor of index that is specified by lower byte of wValue field.

Note: Descriptor of short data length in wLength and descriptor length is automatically transmitted by answer of Get_Descriptor.

(6) SET_DESCRIPTOR request

This request sets or enables the relevant function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ Descriptor	Descriptor type and Descriptor index	0 or Language ID	Descriptor length	Descriptor

Automatic answer of this request is not supported.

According to INT_SETUP interrupt, if the request received was identified as a SET_DESCRIPTOR request, take back data after confirming EP0_DSET_A bit of DATASET register is "1". When completed, access EOP register, and write "0" to EP0_EOPB bit, so status stage is finished. The process is the same for a vendor request.

Please refer to vendor request section.

(7) GET_CONFIGURATION request

This request returns configuration value of present device.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ CONFIG	0	0	1	Configuration value

If it is not configured, it returns "0". Otherwise, it returns the configuration value.

(8) SET_CONFIGURATION request

This request sets device configuration.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ CONFIG	Configuration value	0	0	None

If is the configuration value is that specified using lower byte of wValue field.

When this value is "0", it is not configured.

(9) GET_INTERFACE request

This request returns AlternateSetting value that is set by specified interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000001B	GET_ INTERFACE	0	Interface	1	Alternate setting

If there is no specified interface, it enters STALL state.

Note: If the descriptor is configured no endpoint in interface, use the software answer since automatic answer of GET_INTERFACE request by hardware is not supported.

(10) SET_INTERFACE request

This request selects AlternateSetting in specified interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000001B	SET_ INTERFACE	Alternate setting	Interface	0	None

If there is no specified interface, it enters STALL state.

Note: If the descriptor is configured no endpoint in interface, use the software answer since automatic answer of SET_INTERFACE request by hardware is not supported.

(11) SYNCH_FRAME request

This request transmits synchronous frame of endpoint.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000010B	SYNCH_FRAME	0	Endpoint	2	Frame No.

Automatic answer of this request is not supported.

According to INT_SETUP interrupt, if request received was identified as a SYNCH_FRAME request, write 2byte data in Frame No after confirming EP0_DSET_A bit of DATASET register is "0". When completed, access EOP register, and write "0" to EP0_EOPB bit, so status stage is finished. This can be used only where the endpoint supports isochronous transfer type and supports this request. The process is the same for a vendor request.

Please refer to vendor request section.

3.10.5.2 Printer Class Request

UDC does not support "Automatic answer" of printer class request.

Processing of Class requests is the same as for vendor requests when answering INT_SETUP interrupts.

(1) GET_PORT_STATUS request

This request transmits Port Status to host.

bmRequestType	bRequest	wValue	WIndex	wLength	Data
10100001B	GET_ PORT_STATUS	0	Interface	1	Port status

UDC has an internal Port_STATUS register. Therefore, write port information to this register. When this request is received, data of Port_Status register is transmitted. Set port information to Port_Status register using application before setting register. Port information of only 1 type can be transmitted, so wIndex value is ignored.

D7	D6	D5	D4	D3	D2	D1	D0
Reserved	Reserved	Paper error	Select	Not error	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 soft reset is received, SOFT_RESET flag is set. When status stage is finished, SOFT_RESET flag is reset. This request receive flag is of only 1 type, so wIndex field value is ignored.

(3) Vendor request (Class request)

UDC does not support “Automatic answer” of Vendor requests.

According to INT_SETUP interrupt, access the register in which the device request is stored, and identify the request. If this request is a Vendor request, control the UDC externally, and process the Vendor request.

Below is an explanation for the case where data phase is transmitting (Control read), and for the case where data phase is receiving (Control write).

(a) Control request

bmRequestType	bRequest	wValue	wIndex	wLength	Data
110000xxB	Vendor specific	Vendor specific	Vendor specific	Vendor specific (Expire 0)	Vendor data

When INT_SETUP is received, identify contents of request by bmRequestType, bRequest, wValue, wIndex and wLength registers and process each request. According to application, access Setup_Received register after request has been identified. UDC must also be informed that INT_SETUP interrupt has been recognized.

After transmitting data prepared in application, access DATASET register, and confirm EP0_DSET_A bit is “0”. After confirming, write data FIFO of endpoint 0. If transmitting data is more than payload, write data after it confirming whether EP0_DSET_A bit in DATASET register is “0”. (INT_ENDPOINT0 interrupt can be used.) If writing all data is finished, write “0” to EP0 bit of EOP register. When UDC receives this, the status stage finishes automatically.

INT_STATUS interrupt is asserted when UDC finishes status stage normally. If finishing status stage normally is recognized by external application, manage this stage by using this interrupt signal. If status stage cannot be finished normally and during status stage, a new SETUP token may be received. In this case, when INT_SETUP interrupt signal is asserted, “1” is set to STAGE_ERROR bit of EP0_STATUS register informing externally that the status stage cannot be finished normally.

The dataphase may have finished on a data number that is shorter than the value showed to wLength by protocol of control read transfer type in USB. If the application program is configured using only the wLength value, processing cannot be carried out when the host shifts status stage without arriving at the expected data number. At this point, shifting to status stage can be confirmed by using INT_STATUSNAK interrupt signal. (However, releasing mask of STATUS_NAK bit by using interrupt control register is needed.) In Vendor Request, this problem will not occur because the receiving buffer size is set to host controller by driver. (In every host, data (data that is transmitted from device by payload of 8 bytes) may be taken to be short packet until confirmation of payload size on device side. Therefore, exercise care if controlling standard requests by software.)

(b) Control write/request

There is no dataphase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor specific	Vendor specific	Vendor specific	0	None

When INT_SETUP is received, identify contents of request by bmRequestType, bRequest, wValue, wIndex, wLength registers,and process each request. According to application, access Setup_Received register after request has been identified. UDC must also be informed that the INT_SETUP interrupt has been recognized. If application processing is finished, write "0" to EP0 bit of EOP register. When UDC receives this, the status stage finishes automatically.

There is dataphase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor specific	Vendor specific	Vendor specific	Vendor specific (Except for 0)	Vendor data

When INT_SETUP is received, identify contents of device request by bmRequestType, bRequest, wValue, wIndex, wLength registers, and process each request. According to application, access Setup_Received register after request has been identified. UDC must also be informed that the INT_SETUP interrupt has been recognized.

After receiving data prepared in application, access DATASET register, and confirm EP0_DSET is "1". After confirming, read data FIFO of endpoint 0. If data is more than payload, write data after it confirming whether the EP0_DSET_A bit in DATASET register is "1". (INT_ENDPOINT0 interrupt can be used.) If reading all data is finished, write "0" to EP0 bit of EOP register. When UDC receives this, the status stage finishes automatically.

INT_STATUS interrupt is asserted when UDC finishes status stage normally. If finishing status stage normally is recognized by external application, manage this stage by using this interrupt signal. If status stage cannot be finished normally and during status stage, a new SETUP token may be received. In this case, when INT_SETUP interrupt signal is asserted, "1" is set to STAGE_ERROR bit of EP0_STATUS registerinforming externally that the status stage cannot be finished normally.

Below is control flow in UDC as seen from application.

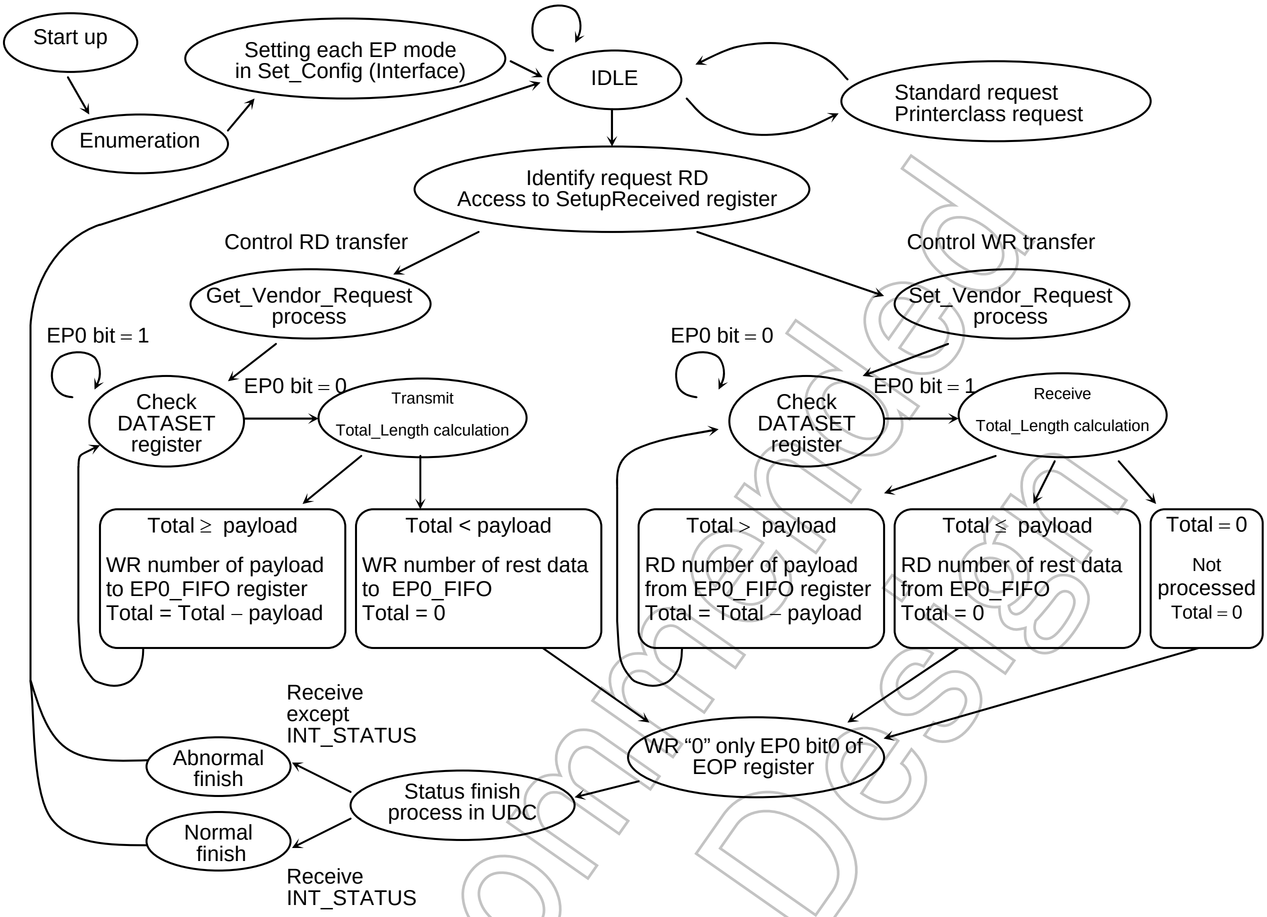


Figure 3.10.2 Control Flow in UDC as seen from Application

Note : This chart does not cover special cases such as overlap receive SETUP packet.
Please refer to chapter of 3.10.6 (2) (c)Control transfer type.

3.10.6 Transfer mode and Protocol Transaction

The UDC performs the following automatically in hardware;

- Receive packet
- Determine address endpoint transfer mode
- Error process
- Confirm toggle bit CRC of data receiving packet
- Generate toggle bit CRC of data transmitting packet, etc
- Handshake answer

(1) Protocol outline

Format of USB packet is shown below. This is processed during transmission and receiving by hardware into the UDC.

- SYNC field

  This field always comes first in each packet, and input data and internal CLK is synchronized in the UDC.

- Packet identification field (PID)

  This field follows SYNC field in every USB packet. The UDC distinguishes the PID type and determines the transfer type by decoding this code.

- Address field

  The UDC uses this field to confirm whether or not this function was specified by the host. The UDC compares the address with that set to the ADDRESS register. If the address accords with it, the UDC continues the process. If the address does not accord, the UDC ignores this token.

- Endpoint field

  If sub-channels of more than two are needed in fields of 4 bits, it decides the function. The UDC can support a maximum of seven endpoints, excluding the control endpoint. Tokens for endpoints that are not permitted are ignored.

- Frame number field

  A field of 11 bits is added by the host at each frame. This field follows the SOF token that is transmitted first in each frame, and the frame number is specified. The UDC reads the content of this field when the SOF token is received, and sets the frame number to the FRAME register.

- Data field

  This field is data of unit byte in bytes 0 to 1023. When receiving it, the UDC transfers only part of this data to FIFO, and after CRC is confirmed, an interrupt signal is asserted and the UDC informs FIFO that data transfer is completed. When transmitting, following IN token, FIFO data is transferred. Finally, data CRC field is attached.

- CRC function

  5 bits CRC is attached to the token, and 15 bits CRC to the data. The UDC automatically compares the CRC of the received data with the attached CRC. When transmitting, CRC is generated automatically and is transmitted. This function may be compared by various transfer modes.

(2) Transfer mode

UDC supports FULL speed transfer mode.

- FULL speed device

  Control transfer type

  Interrupt transfer type

  Bulk transfer type

  Isochronous transfer type

The following is an explanation of UDC operation in each transfer mode.

The explanation is of data flow up until FIFO.

(a) Bulk transfer type

Bulk transfer type warrants transferring no error between host and function by using detect error and retry. Basically, 3 phases are used - token, data and handshake. However, with flow control and a STALL condition, data phase is changed to hand shake phase, and it become to 2 phases. The UDC holds status of each endpoint, and flow control is controlled in hardware. Each endpoint condition can be confirmed using EPx_STATUS register.

(a-1) Transmission bulk mode

Below is the transaction format for bulk transfer during transmitting.

- Token: IN
- Data: DATA0/DATA1, NAK, STALL
- Handshake: ACK

Control flow

Below is the control-flow when the UDC receives an IN token.

1. The token packet is received and the address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the IN token. If it does not correspond, the state returns to IDLE.
2. Condition of EPx_STATUS register is confirmed.
   - INVALID condition: State returns to IDLE.
   - STALL condition: Stall handshake is returned and state returns to IDLE.

   FIFO condition is confirmed, if data number of 1 packet is not prepared, NAK handshake is returned, and state returns to IDLE.

   If data number of 1 packet is prepared to FIFO, it shifts to 3.
3. Data packet is generated.

   Data packet generated by using toggle bit register in UDC.

   Next, data is transferred from FIFO of internal UDC to SIE, and data packet is generated. At this point, the transferred data number is confirmed. And if there is more than the maximum payload size of each endpoint, bit stuff error is generated, transfer is finished, and STATUS becomes STALL.
4. CRC bit (counted transfer data of FIFO from first to last) is attached to last.
5. When ACK handshake from host is received,
   - Clear FIFO.
   - Clear DATASET register.
   - Renew toggle bit, and prepare for next.
   - Set STATUS to READY.

   UDC finishes normally. FIFO can receive the next data.

   If a time out occurs without receiving ACK from host,

   - Set STATUS to TX_ERR.
   - Return FIFO address pointer.

   Execute above setting. And wait next retry keeping FIFO data.

   This flow is shown in Figure 3.10.3.

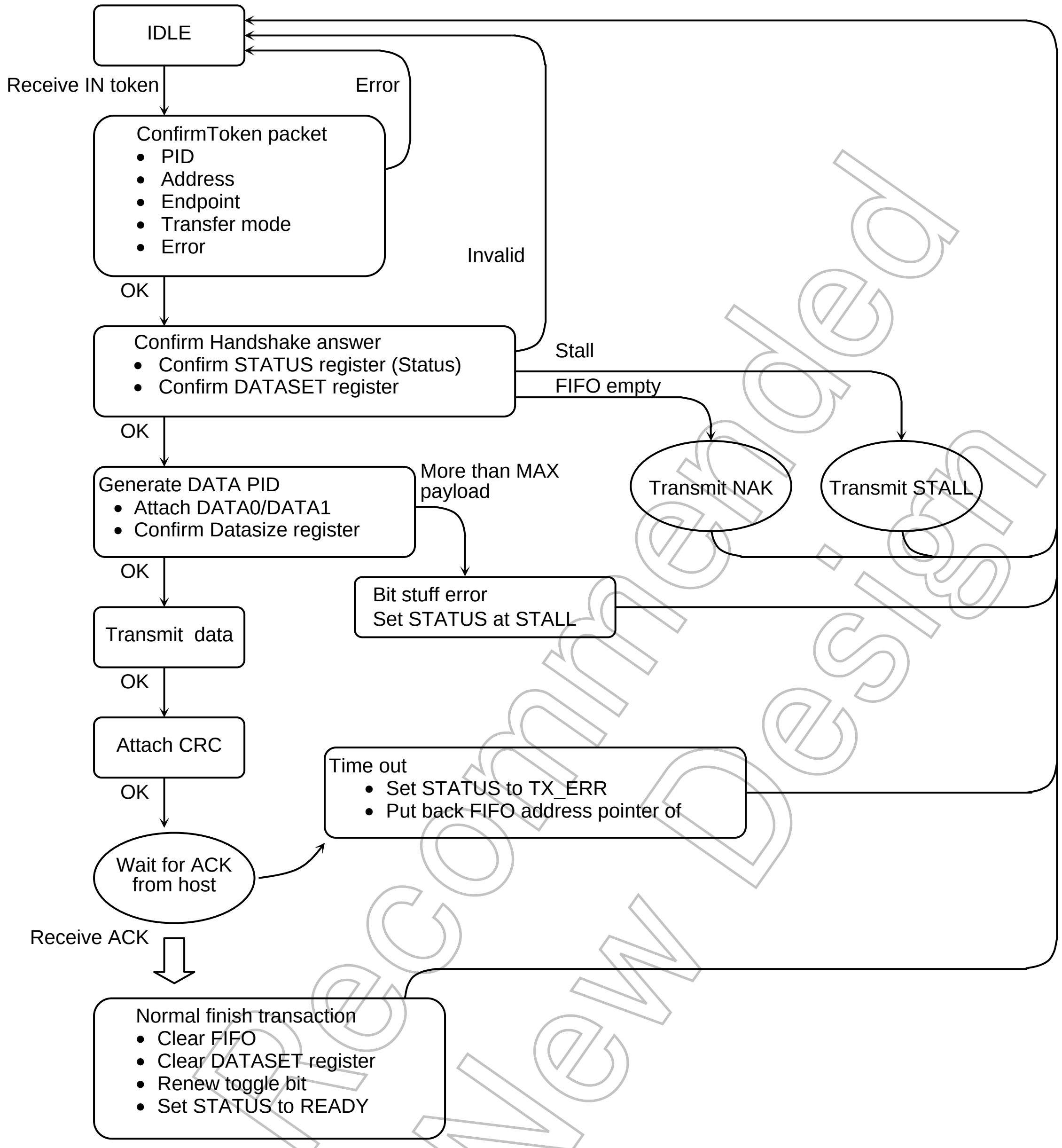


Figure 3.10.3 Control Flow in UDC (Bulk transfer type (transmission)/Interrupt transfer type (transmission))

(a-2) Receiving bulk mode

Below is the transaction format for receiving bulk transfer type.

- Token: OUT
- Data: DATA0/DATA1
- Handshake: ACK, NAK, STALL

Control flow

Below is the control-flow when the UDC receives an IN token.

1. The token packet is received and the address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the OUT token. If it does not correspond, the state returns to IDLE.
2. Condition of status register is confirmed.
    - INVALID condition: State returns to IDLE.
    - STALL condition: When dataphase finishes, stall handshake is returned, the state returns to IDLE, and data is canceled.

   FIFO condition is confirmed, if data number of 1 packet is not prepared, present transferred data is canceled, NAK handshake is returned after dataphase, and the state returns to IDLE.
3. Data packet is received.
   Data is transferred from SIE of internal UDC to FIFO. At this point, it confirms transferred data number, and if there is more than the maximum payload size of each endpoint, STATUS becomes STALL and the state returns to IDLE. ACK handshake does not return.
4. After last data is transferred, the counted CRC is compared with the transferred CRC. If they do not correspond, STATUS is set to RX_ERR and the state returns to IDLE. At this point ACK is not returned.
   After retry, when next data is received normally, STATUS changes to DATIN. If the data toggle does not correspond, it is judged not to have taken ACK in the last loading, the current loading is regarded as a retry of the last loading and data is canceled. Set STATUS as RX_ERR, return to host and return to IDLE. FIFO address pointer returns, and the next data can be received.
5. If CRC is compared with toggle and it finishes normally, ACK handshake is returned.
   Below is the process in the UDC.
    - Set transfer data number to DATASIZE register.
    - Set DATASET register.
    - Renew toggle bit, and prepare for next.
    - Set STATUS to READY.

UDC finishes normally.

This flow is shown in Figure 3.10.4.

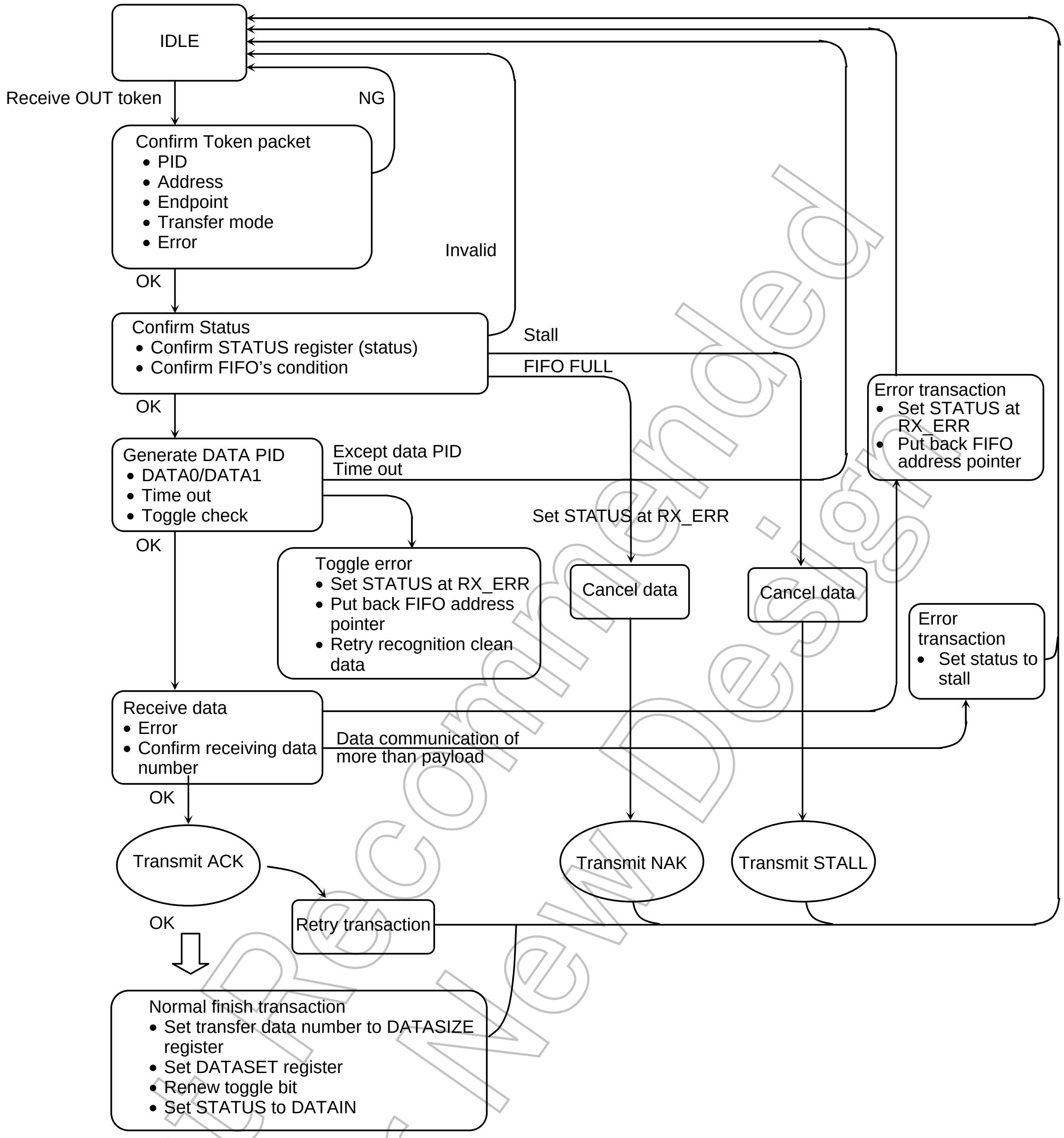


Figure 3.10.4 Control Flow in UDC (Bulk transfer type (Receiving))

(b) Interrupt transfer type

Interrupt transfer type uses the same transaction format as transmission bulk transfer.

For transmission using toggle bit, hardware setting and answer in the UDC are the same as for transmission bulk transfer. Interrupt transfer can be transferred without using toggle bit. In this case, if ACK handshake from host is not received, toggle bit is renewed, and finish is normal. The UDC clears FIFO for next transfer.

(b-1) Interrupt transmitting mode (Toggle mode)

UDC operation is same as in bulk transmission mode. Please refer to section (a).

(b-2) Interrupt transmission mode (Not toggle mode)

This is basically the same as bulk transmission mode. However, if ACK handshake from host is not received, transaction is different.

When ACK handshake from host is received after transmission of data packet,

Clear FIFO.

- Clear DATASET register.
- Renew toggle bit and prepare for next.
- Set STATUS to READY.

UDC finishes normally by above transaction. FIFO can receive next data.

If a time out occurs without receiving ACK from host,

- Clear FIFO.
- Clear DATASET register.
- Renew toggle bit and prepare for next.
- Set STATUS to TX_ERR.

Execute above setting. This setting is the same except for STATUS changes.

(c) Control transfer type

Control transfer type is configured in the three stages below.

- Setup stage
- Data stage
- Status stage

Data stage is sometimes skipped. Each stage is configured in one or several transactions. The UDC executes each transaction while managing three stages in hardware. Control transfer has the 3 types given below depending on whether there is data stage or not, and on direction.

- Control read transfer type
- Control write transfer type
- Control write transfer type (No data stage)

The 3 transfer sequences are shown in Figure 3.10.6, Figure 3.10.7 and Figure 3.10.8.

The UDC automatically answers standard requests in hardware. Class request and vendor request must have an intervening CPU controlling the UDC.

Below is the control flow in the UDC and the control flow in the intervening CPU.

(c-1) Setup stage

Setup stage is the same as transmission bulk transaction except that token ID becomes SETUP.

However, control flow in the UDC is different.

- Token: SETUP
- Data: DATA 0
- Handshake: ACK

Control flow

Below is the control flow in the UDC when SETUP token is received.

1. SETUP token packet is received and address, endpoint number and error are confirmed. It also checks whether the relevant endpoint is in control transfer mode.
2. STATUS register state is confirmed.

State returns to IDLE only if it is INVALID state.

In bulk transfer mode, receiving data is enabled by STATUS registers value and FIFO condition. However, in SETUP stage, STATUS is returned to READY and accessing from the CPU to FIFO is always prohibited, and internal FIFO of endpoint 0 is cleared. It also prepares for following dataphase.

If the CPU accesses Setup Received registers in the UDC, it recognizes Device request as received, and accessing from the CPU to EP0 is enabled.

This function is for receiving a new request when the current device request has not finished normally.

3. Data packet is received.

   Device request of 8 bytes from SIE in UDC is transferred to the request register below.

   - bmRequestType register
   - bmRequest register
   - wValue register
   - wIndex register
   - wLength register

4. After last data is transferred, counted CRC is compared with transferred CRC. If they do not correspond, STATUS is set to RX_ERR and the state returns to IDLE. At this point it does not return ACK, and host retries.

5. If CRC corresponds with toggle and it finishes normally, ACK handshake is returned to host. The process in the UDC is shown below.

   - Receiving device request is judged whether software control or hardware control. If the request needs control in software, INT_SETUP interrupt is asserted. If hardware is used, INT_SETUP interrupt is not asserted.
   - According to stage control flow, prepare for next stage.
   - Set STATUS to DATAIN.
   - Set toggle bit to "1".

The Setup stage is completed by the above.

This flow is shown in Figure 3.10.5.

8-byte data that is transferred by this SETUP stage is device request.

The CPU must process corresponding to device request.

The UDC detects the following contents only from data of 8 bytes, and it manages stage in hardware.

- Whether there is data stage or not
- Data stage direction

These are used to determine control read transfer type, control write transfer type, and control write transfer type (no data phase).

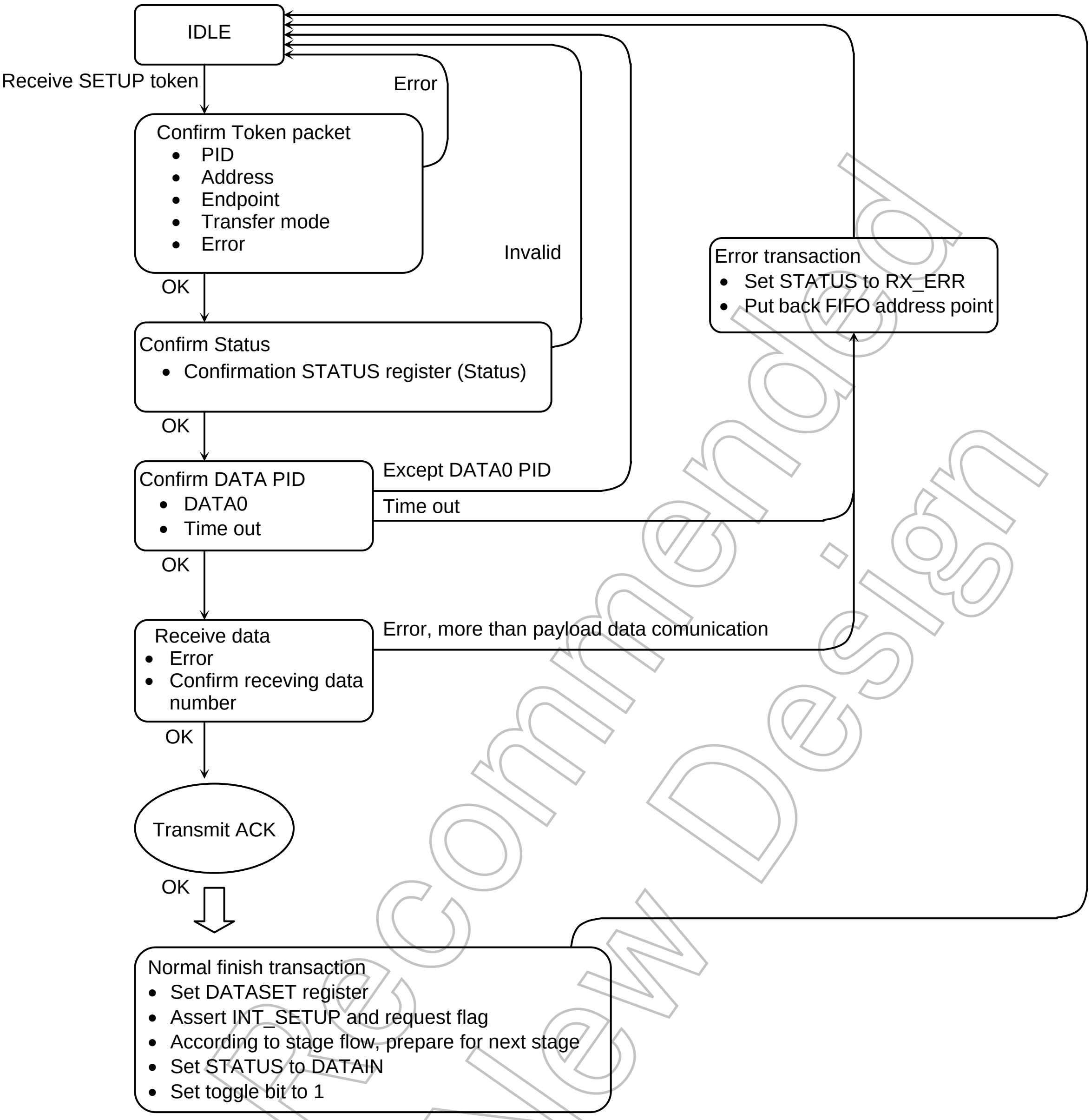


Figure 3.10.5 Control Flow in UDC (Setup stage)

(c-2) Data stage

Data stage is configured by one or several transactions based on toggle sequence.

The transaction is the same as for format transmission or receiving bulk transaction except for the following differences:

- Toggle bit starts from “1” by SETUP stage.
- It determines whether right or not by comparing IN and OUT token with direction bit of device request. If a token of the opposite direction is received, it is recognized as status stage.
- INT_ENDPOINT0 interrupt is asserted.

(c-3) Status stage

Status stage is configured 0-data-length packet with DATA1’s PID and handshake IN or OUT token. It uses a transaction in the opposite direction to the preceding stage.

The combination is given below.

- Control read transfer type: OUT
- Control write transfer type: IN
- Control write transfer type (not dataphase): IN

UDC processes status stage base of control flow in control transfer type. At this point, CPU must write “0” to EP0 bit of EOP register in last transaction for status stage to finish normally.

Details of status stage are given below.

(c-3-1) IN status stage

IN status stage transaction format is given below.

- Token: IN
- Data: DATA1 (0 data length), NAK, STALL
- Handshake: ACK

Control flow

The transaction flow of IN status stage in UDC is given below.

1. Token packet is received and address, endpoint number and error are confirmed. If it does not corrspond, the state returns to IDLE. If status stage is enabled based on stage control flow in the UDC, advance to next stage.
2. STATUS register state is confirmed.
    - INVALID condition: State returns to IDLE.
    - STALL condition: Stall handshake is returned and state returns to IDLE.

   Confirmation of whether EOP register is accessed or not is carried out externally. If it is not accessing, NAK handshake is returned to continue control transfer, and state returns to IDLE.
3. If EOP register access is confirmed, 0-data-length data packet and CRC are transmitted.

4. If ACK handshake from host is received,
    - Set STATU to READY.
    - Assert INT_STATUS interrupt.

It finishes normally by the above transaction.

If a time out occurs without receiving ACK from host,

- Set STATUS register to TX_ERR and state returns to IDLE, and wait for restring status stage.

At this point, if new SETUP stage is started without status stage finishing normally, the UDC sets error to STATUS register.

(c-3-2) OUT status stage

The transaction format for OUT status stage is given below.

- Token: OUT
- Data: DATA1 (0 data length)
- Handshake: ACK, NAK, STALL

Control flow

The transaction flow for OUT status stage in the UDC is given below.

1. Token packet is received and address, endpoint number and error are confirmed. If they do not correspond, the state returns to IDLE. If status stage is enabled based on stage control flow in the UDC, advance to next stage.
2. STATUS register state is confirmed.
    - INVALID condition: State returns to IDLE.
    - STALL condition: Data is cleared, stall handshake is returned, and state returns to IDLE.

    Whether EOP register is accessed or not is confirmed externally. If it is not accessed, NAK handshake is returned to continue control transfer, and state returns to IDLE.
3. If EOP register access is confirmed, 0-data-length data packet and CRC are received.
4. If there is no error in data, ACK handshake is transmitted to host.
    - Set STATUS to READY.
    - Assert INT_STATUS interrupt.

It finishes normally by the above transaction.

If there is an error in data, ACK handshake is not returned.

- Set RX_ERR to STATUS register and return to IDLE. It waits to retry status stage.

At this point, if new SETUP stage is started without status stage finishing normally, the UDC sets error to STATUS register. For sequence of this protocol, refer to section supplement.

(c-4) Stage management

The UDC manages each stage of control transfer by hardware.

Each stage is changed by receiving token from USB host, or CPU accesses register. Each stage in control transfer type has to process combination software. UDC detects the following contents from 8-byte data in SETUP stage. The stage is managed by determining control transfer type.

- Whether there is data stage or not
- Data stage direction

Based on these it is determined to be either control read transfer type, control write transfer type, or control write transfer type (No data stage).

Various conditions for changing stage in control transfer are given below.

If receiving token for next stage from host before switching to next stage from state of internal UDC, NAK handshake is returned and BUSY is informed to USB host. In all control transfer types, if SETUP token is received from host, present transaction is stopped, and it switches to SETUP stage in the UDC. The CPU receives new INT_SETUP even if it is processing previous control transfer.

Stage change condition of control read transfer type

1. Receive SETUP token from host
   - Start setup stage in UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt externally.
   - Change data stage in the UDC.
2. Receive IN token from host
   - The CPU receives a request from the request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register to inform the UDC that INT_SETUP interrupt has been recognized .
   - According to Device request, monitor EP0 bit of DATASET register, and write data to FIFO.
   - If the UDC is set data of payload to FIFO or CPU set short packet transfer in EOP register, EP0 bit of DATASET register is set.
   - The UDC transfers data that is set to FIFO to host by IN token interrupts.
   - When the CPU finishes transaction, it writes “0” to EP0 bit of EOP register.
   - Change status stage in the UDC.
3. Receive OUT token from host.
   - Return ACK to OUT token and change state to IDLE in the UDC.
   - Assert INT_STATUS interrupt externally.

These changing conditions are shown in Figure 3.10.6.

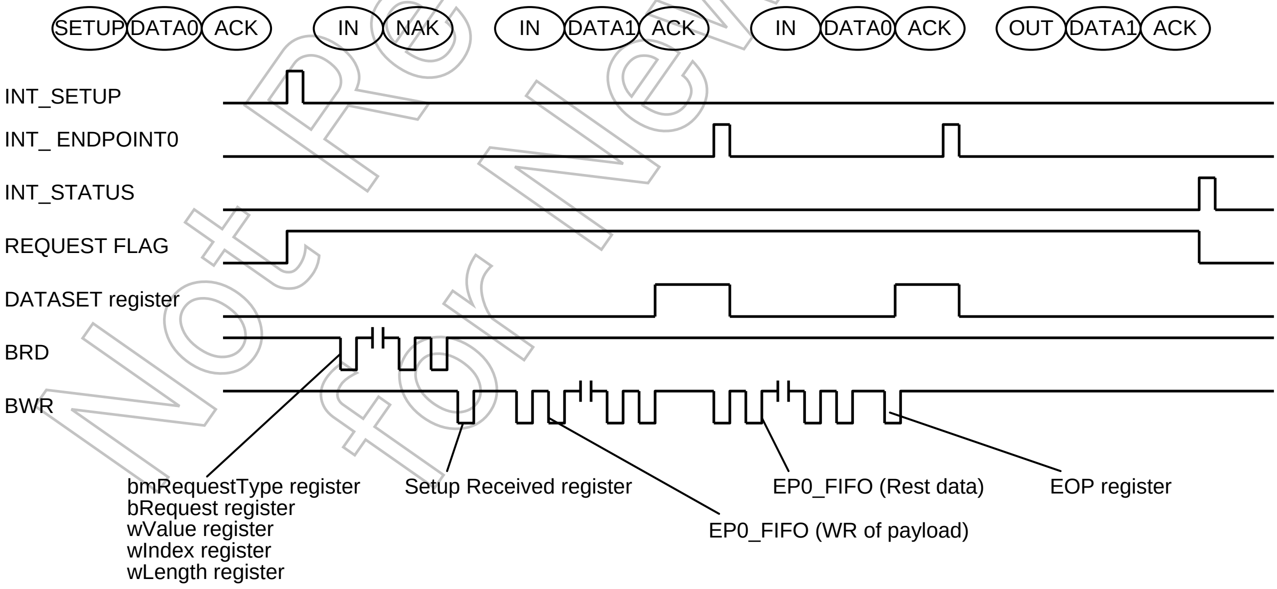


Figure 3.10.6 The Control Flow in UDC (Control Read Transfer Type)

Stage change condition of control writes transfer type

1. Receive SETUP token from host.
    - Start setup stage in the UDC.
    - Receive data in request normally and judge. And assert INT_SETUP interrupt externally.
    - Change data stage in the UDC.
2. Receive OUT token from host.
    - CPU receives a request from the request register every INT_SETUP interrupt.
    - Judge request and access Setup Received register for inform the UDC that INT_SETUP interrupt has been recognized.
    - Receive dataphase data normally, and set EP0 bit of DATASET register.
    - The CPU receives data in FIFO by setting DATASET.
    - The CPU processes receiving data by device request.
    - When the CPU finishes transaction, it writes "0" to EP0 bit of EOP register.
    - Change status stage in the UDC.
3. Receive IN token from host.
    - Return data packet of 0 data to IN token and change state to IDLE in the UDC.
    - Assert INT_STATUS interrupt externally when ACK for 0 data packet is received.

These changing conditions are shown in Figure 3.10.7.

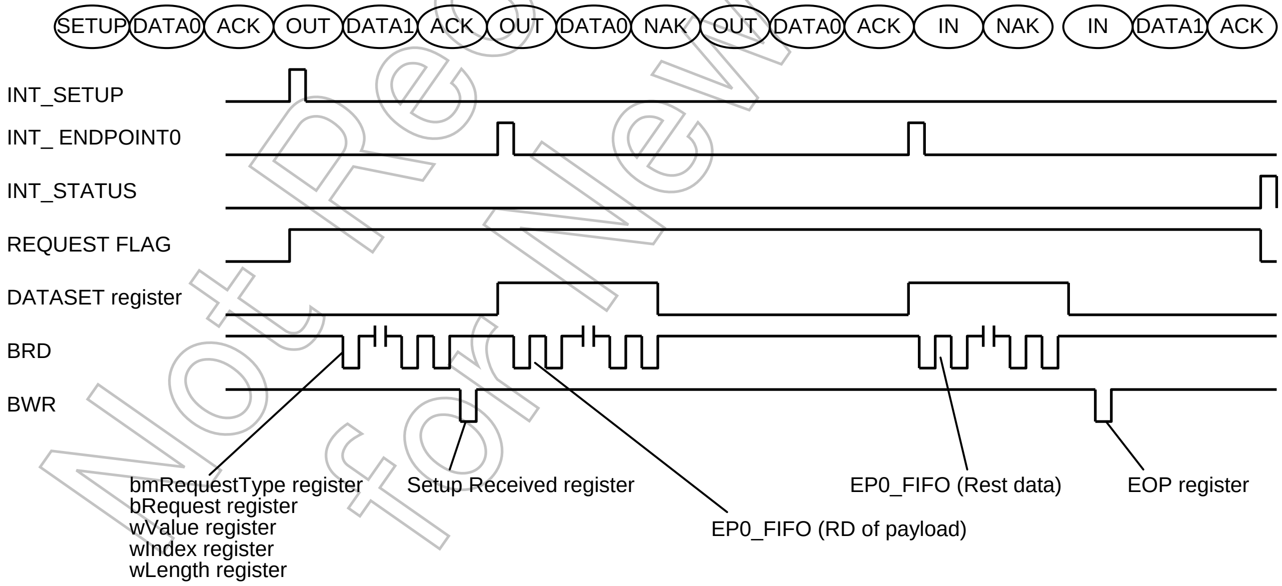


Figure 3.10.7 The Control Flow in UDC (Control Write Transfer Type)

In control read transfer type, transaction number of data stage does not always correspond with the data number specified by the device request. The CPU can therefore process using INT_STATUSNAK interrupt. However, when class and vendor request is used, wLength value corresponds to data transfer number in data phase. With this setting, using this interrupt is not need. Data stage data can be confirmed by accessing DATASIZE register.

Stage change condition of control write (no data stage) transfer type

1. Receive SETUP token from host
   - Start setup stage in UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt externally.
   - Change data stage in the UDC.
2. Receive IN token from host
   - CPU receives a request from the request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register to inform the UDC that INT_SETUP interrupt has been recognized.
   - The CPU processes receiving data by device request.
   - When the CPU finishes transaction, it writes “0” to EP0 bit of EOP register.
   - Change status stage in the UDC.
   - Return data packet of 0 data to IN token and change state to IDLE in the UDC.
   - Assert INT_STATUS interrupt externally when ACK for 0 data packet is received.

These change condition is Figure 3.10.8.

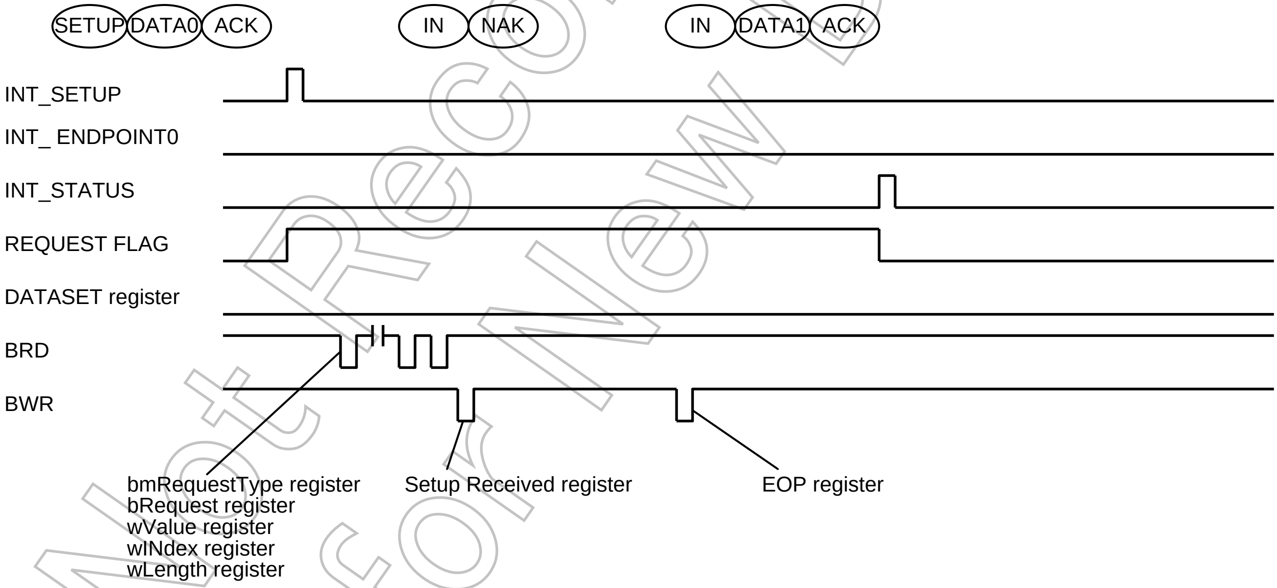


Figure 3.10.8 The Control Flow in UDC (Control Write Transfer Type not Dataphase)

(d) Isochronous transfer type

Isochronous transfer type is guaranteed transfer by data number that is limited to each frame.

However, this transfer does not retry when an error occurs. Therefore, Isochronous transfer type transfer only 2 phases (token, data) and it does not use handshake phase. And data PID for data phase is always DATA0 because this transaction does not support toggle sequence. Therefore, UDC does not confirm when data PID is in receiving mode.

Isochronous transfer type processes data every frame. Therefore, all transactions for completed transfers use receiving SOF token. The UDC uses FIFO that is divided into two in Isochronous transfer type.

(d-1) Isochronous transmission mode

The transaction format for Isochronous transfer type format in transmitting is given below.

- Token: IN
- Data: DATA0

Control flow

Isochronous transfer type is frame management. And data that is written to FIFO in endpoint is transmitted by IN token in the next frame.

Below are two conditions in FIFO of Isochronous transmission mode transferring.

X. FIFO for storing data that transmits to host in present frame
(DATASET register bit = 1)

Y. FIFO for storing data for transmitting host in next frame
(DATASET register bit = 0)

FIFO that is divided into two (packet A and packet B) conditions is whether X condition or Y condition. The flow below is explained as X Condition (packet A), Y Condition (packet B) in present frame.

X and Y conditions change one after the other by receiving SOF.

Control flow in the UDC when receiving IN token is shown below.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the IN token. If it does not correspond, the state returns to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State returns to IDLE.
3. Data packet is generated.

   Data packet is generated. At this point, data PID is always attached to DATA0. Next, data is transferred from FIFO (X condition) of packet A in UDC to SIE, and DATA packet is generated.
4. CRC bit (counted transfer data of FIFO from first to last) is attached to last.

5. Below is transaction when SOF token is received from host.
   - Change the packet A's FIFO from X Condition to Y Condition, and clear data.
   - Change the packet B from Y Condition to X Condition.
   - Set frame number to frame register.
   - Assert SOF and inform externally that frame is incremented.
   - DATASET register clears packet A bit and it sets packet B bit arrangement loading in present frame.
   - Set STATUS to READY.

The UDC finishes normally by above transaction.

Packet A's FIFO can be received with next data.

In renewed frame, Packet A's FIFO interchanges with packet B's FIFO, and transaction uses same flow.

If SOF token is not received by error and so on, this data is lost because frame is not renewed. There is no problem in receiving PID if frame data is received with CRC error, USB sets LOST to STATUS on FRAME register, and exact frame number is unknown. However, in this case, SOF is asserted and FIFO condition is renewed. If SOF token is received without transmit and transfer Isochronous in frame, UDC clears FIFO (X Condition) and sets STATUS to FULL.

Note1: In IN transfer, data ("LSB (bit0) of last byte in data" ="1" and "last 5 bits of result that CRC counted" = "1") is transmitted to USB host, that data is transmitted correctly.

However, CRC error is recognized. In transfer other than Isochronous IN transfer, the following handshake packet from USB host is recognized correctly because of data, is transmitted to USB host correctly, and transfer returns to normal.

However, in Isochronous transfer, token packet is transmitted to it. Therefore, it ignores token packet when CRC error is recognized.

Therefore, for isochronous transfer, execute the following.

- Transmit data that last bit of data field is "0".
- Attach data that last bit of data field is "0".
- After Isochronous IN transfer is executed, execute dummy transfer 1 time before receiving next SOF.

Note2: When using the Isochronous IN transfer, do not use other endpoints simultaneously.

Note: EPx_DATASETA,B change at 3 clocks of 12MHz after receiving SOF. Write data to FIFO after EPx_DATASETA,B are changing.

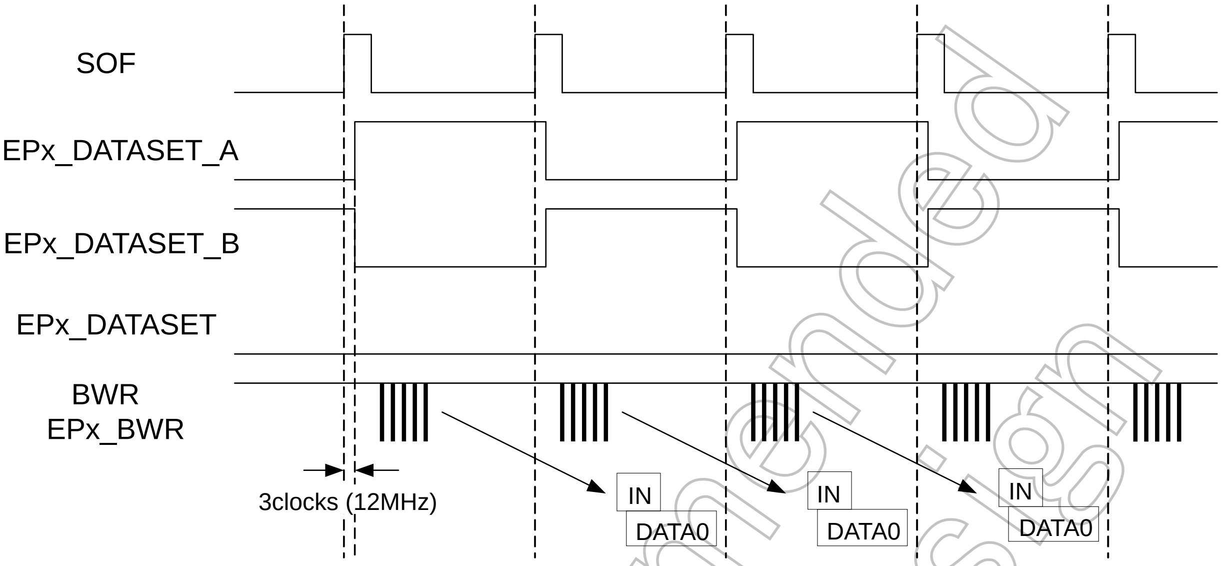


Figure 3.10.9 Isochronous transfer Mode

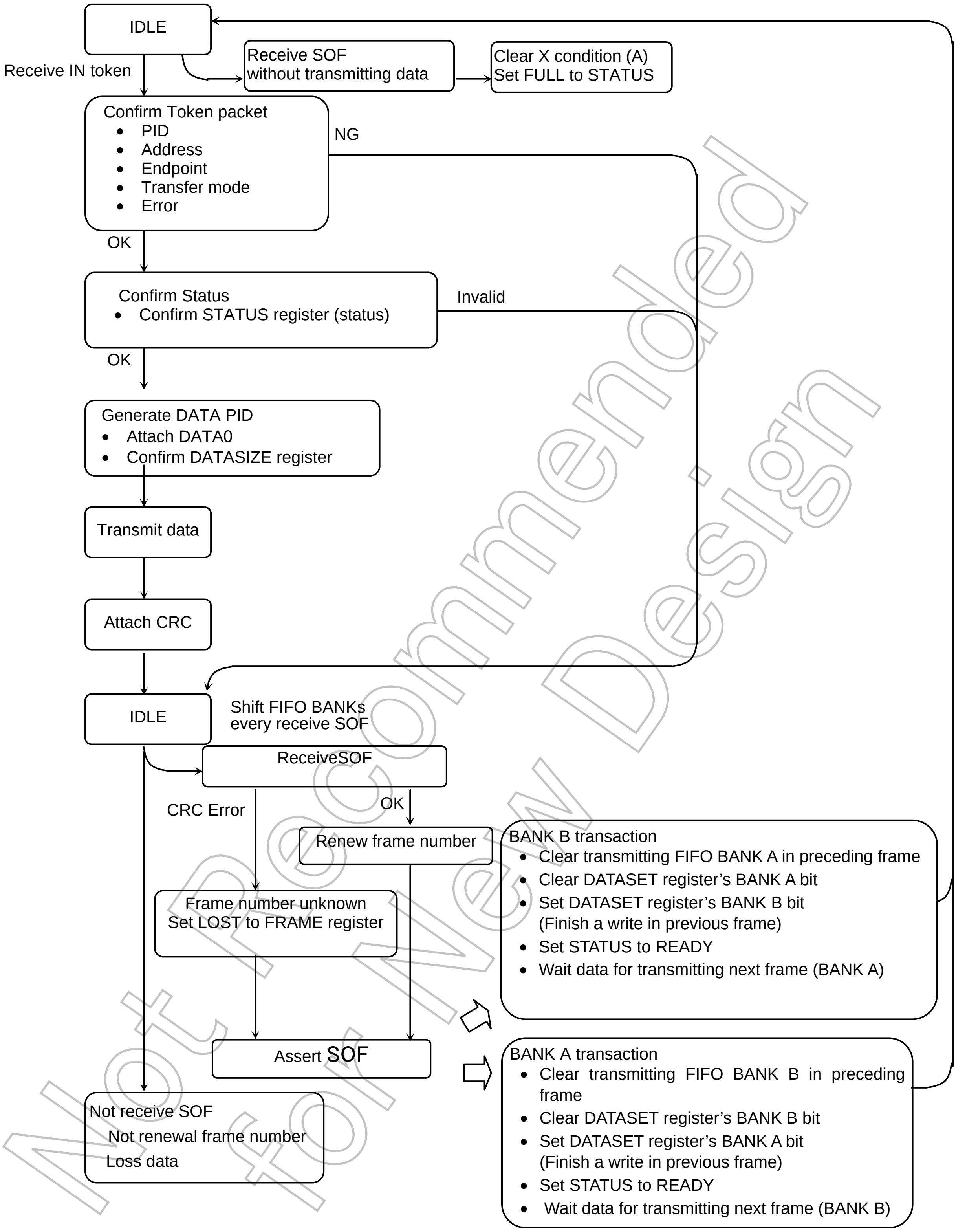

IDLE
Receive IN token
Receive SOF
without transmitting data
Clear X condition (A)
Set FULL to STATUS
Confirm Token packet
• PID
• Address
• Endpoint
• Transfer mode
• Error
NG
OK
Confirm Status
• Confirm STATUS register (status)
Invalid
OK
Generate DATA PID
• Attach DATA0
• Confirm DATASIZE register
Transmit data
Attach CRC
IDLE
Shift FIFO BANKs
every receive SOF
ReceiveSOF
CRC Error
OK
Renew frame number
Frame number unknown
Set LOST to FRAME register
Assert SOF
Not receive SOF
Not renewal frame number
Loss data
BANK B transaction
• Clear transmitting FIFO BANK A in preceding frame
• Clear DATASET register's BANK A bit
• Set DATASET register's BANK B bit
(Finish a write in previous frame)
• Set STATUS to READY
• Wait data for transmitting next frame (BANK A)
BANK A transaction
• Clear transmitting FIFO BANK B in preceding frame
• Clear DATASET register's BANK B bit
• Set DATASET register's BANK A bit
(Finish a write in previous frame)
• Set STATUS to READY
• Wait data for transmitting next frame (BANK B)

Figure 3.10.10 Control Flow in UDC (Isochronous transfer type (Transmission))

(d-2) Isochronous receiving mode

Transaction format for Isochronous transfer type in receiving is given below.

- Token: OUT
- Data: DATA0

Control flow

Isochronous transfer type is frame management. And data that is written to FIFO by OUT token is received to the CPU in the next frame.

Below are two conditions in FIFO of Isochronous receiving mode transferring

X. FIFO for storing data received from host in present frame (DATASET register bit = 0)

Y. FIFO for storing data for transmitting host in previous frame (DATASET register bit = 1)

FIFO that is divided into two (packet A and packet B) conditions is whether X condition or Y condition. The flow below explains X Condition (packet A) and Y Condition (packet B) in present frame.

X and Y conditions change one after the other by receiving SOF.

Below is control flow in the UDC when receiving OUT token.

The whole transaction is processed by hardware.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the OUT token. If it does not correspond, the state returns to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State returns to IDLE.
3. Data packet is received.
   Data is transferred from SIE into the UDC to packet A's FIFO (X Condition).
4. After last data was transferred, and counted CRC is compared with transferred CRC. When transfer is finished, the result is reflected to STATUS. However, data is stored FIFO, data number that packet A is received is set to DATASIZE register of packet A.
5. The transaction when SOF token from host is received is given below.
   - Change packet A's FIFO from X Condition to Y Condition.
   - Change packet B from Y Condition to X Condition, and clear data. Prepare for next transfer.
   - Set frame number to frame register.
   - Assert SOF and inform externally that frame is incremented.
   - DATASET register set packet A bit and clear packet B bit arrangement loading in present frame.
   - If CRC comparison result agrees, DATAIN is set to STATUS. If result does not agree, RX_ERR is set to STATUS.

The UDC finishes normally by the above transaction.

The CPU takes back packet A's data.

In renewed frame, Packet A's FIFO interchanges with packet B's FIFO, and the transaction uses the same flow.

If SOF token is not received by error and so on, this data is lost because the frame is not renewed. There is no problem in receiving PID and if frame data is received with CRC error, USB sets LOST to STATUS on FRAME register, and exact frame number is unknown. However, in this case, SOF is asserted and FIFO condition is renewed. If SOF token is received without transmit and transfer Isochronous in frame, UDC clears FIFO (X Condition) and sets STATUS to FULL.

These are shown in Figure 3.10.12.

Note: EPx_DATASET changes at 2 clocks of 12MHz after receiving SOF. Read data from FIFO after EPx_DATASET is rising.

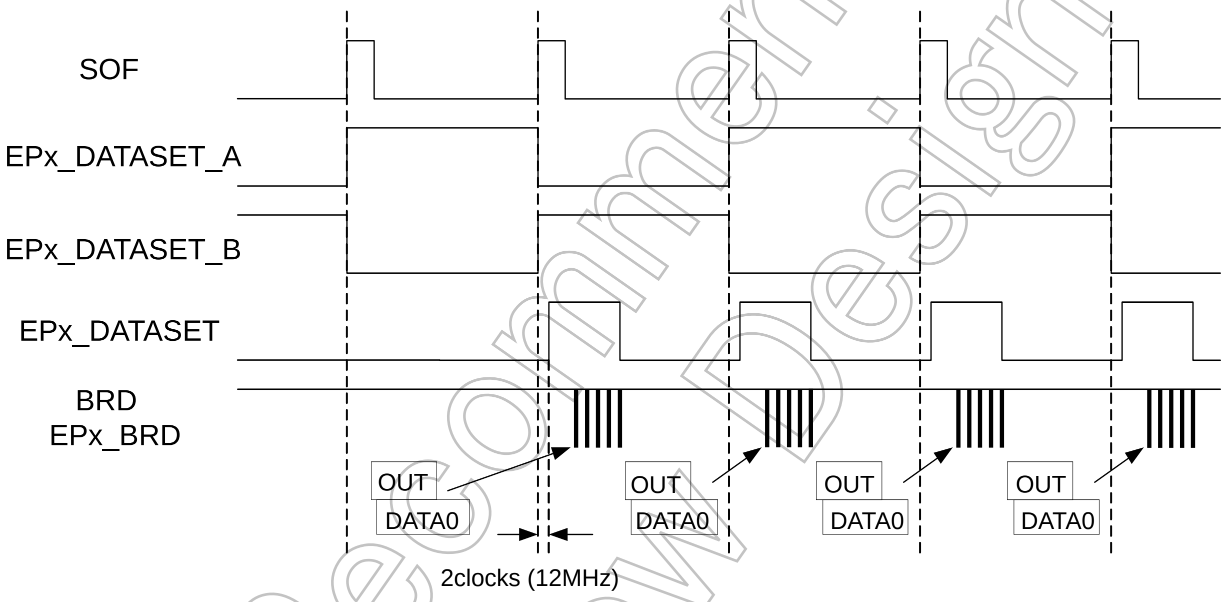


Figure 3.10.11 Isochronous Receiving mode

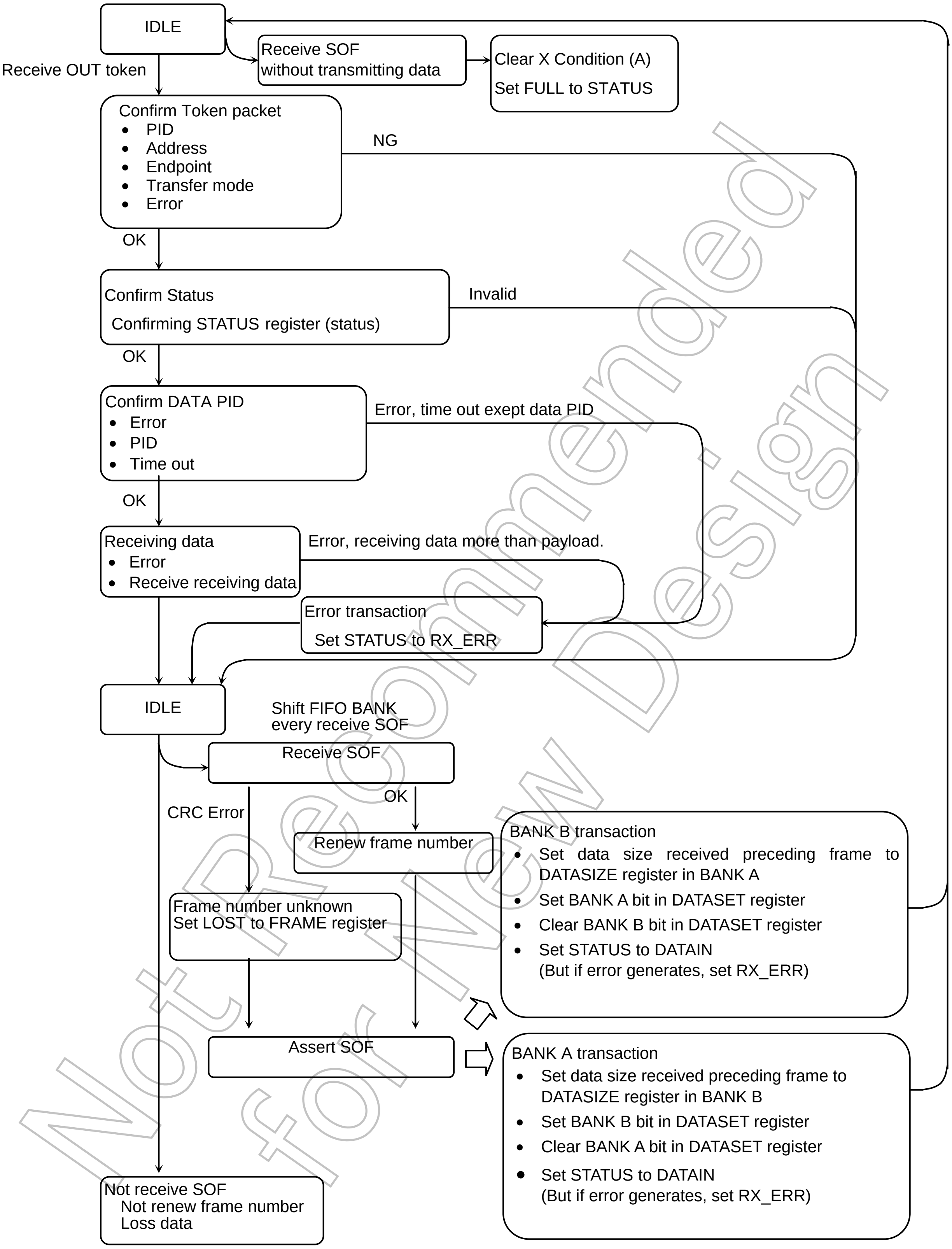

IDLE
Receive OUT token
Receive SOF
without transmitting data
Clear X Condition (A)
Set FULL to STATUS
Confirm Token packet
PID
Address
Endpoint
Transfer mode
Error
NG
OK
Confirm Status
Confirming STATUS register (status)
Invalid
OK
Confirm DATA PID
Error
PID
Time out
Error, time out exept data PID
OK
Receiving data
Error
Receive receiving data
Error, receiving data more than payload.
Error transaction
Set STATUS to RX_ERR
IDLE
Shift FIFO BANK
every receive SOF
Receive SOF
OK
CRC Error
Renew frame number
Frame number unknown
Set LOST to FRAME register
Assert SOF
BANK B transaction
Set data size received preceding frame to DATASIZE register in BANK A
Set BANK A bit in DATASET register
Clear BANK B bit in DATASET register
Set STATUS to DATAIN
(But if error generates, set RX_ERR)
BANK A transaction
Set data size received preceding frame to DATASIZE register in BANK B
Set BANK B bit in DATASET register
Clear BANK A bit in DATASET register
Set STATUS to DATAIN
(But if error generates, set RX_ERR)
Not receive SOF
Not renew frame number
Loss data

Figure 3.10.12 Control Flow in UDC (Isochronous transfer type (Receiving))

3.10.7 Bus Interface and Access to FIFO

(1) CPU bus interface

The UDC prepares two types of FIFO access, single packet and dual packet. In single packet mode, FIFO capacity that is implemented by hardware is used as large FIFO. In dual packet mode, FIFO capacity is divided into two and used as two FIFOs. It is also used as an independent FIFO. Even if the UDC is transmitting and receiving to USB host, it can be used as an efficient bus by possible load to FIFO.

But control transfer type receives only single packet mode.

Epx_SINGLE signal in dual packet mode must be fixed to “0”. If this signal is fixed to “0”, FIFO register runs in single mode.

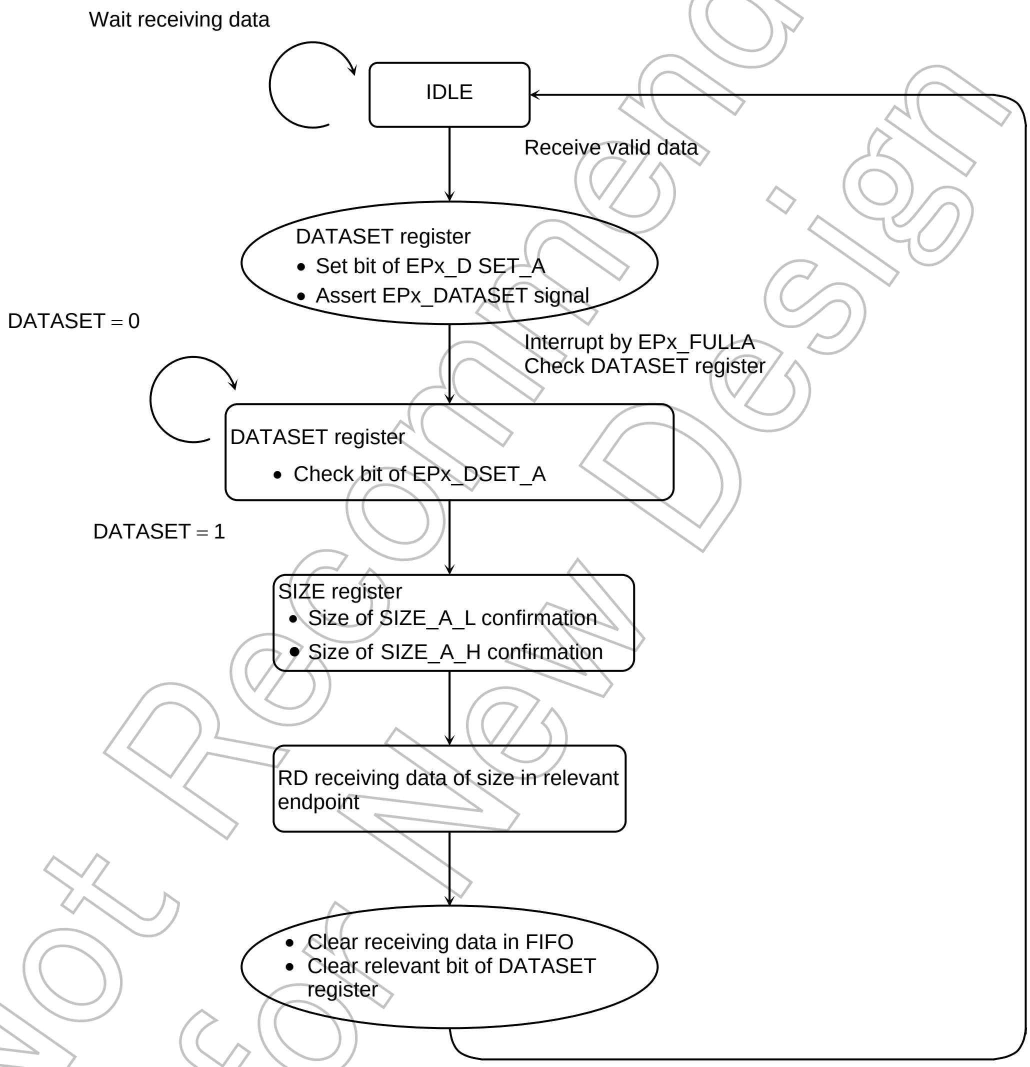
Sample: Where endpoint 1 is used to dual packet of payload 64 bytes.

EP1_FIFO size	:	Prepare 128 bytes
EP1_SINGLE signal	:	Hold 0
EP1 Descriptor setting		
Direction	:	Optional
Max payload size	:	64 bytes
Transfer mode	:	Optional

(a) Single packet mode

This is data sequence of single packet mode when CPU bus interface is used. Figure 3.10.13 is receiving sequence. Figure 3.10.14 is transmitting sequence. This chapter focuses on access to FIFO. For Data sequence with USB host refer to chapter 5.

Endpoint 0 cannot be changed to exclusive single packet mode. Endpoints 1 to 3 can be changed between single packet and dual packet by setting Epx_SINGLE register. Do not change packet when transferring.



Wait receiving data
IDLE
Receive valid data
DATASET register
• Set bit of EPx_D SET_A
• Assert EPx_DATASET signal
DATASET = 0
Interrupt by EPx_FULLA
Check DATASET register
DATASET register
• Check bit of EPx_DSET_A
DATASET = 1
SIZE register
• Size of SIZE_A_L confirmation
• Size of SIZE_A_H confirmation
RD receiving data of size in relevant endpoint
• Clear receiving data in FIFO
• Clear relevant bit of DATASET register

Figure 3.10.13 Receiving Sequence in Single Packet Mode

Below is the transmitting sequence in single packet mode.

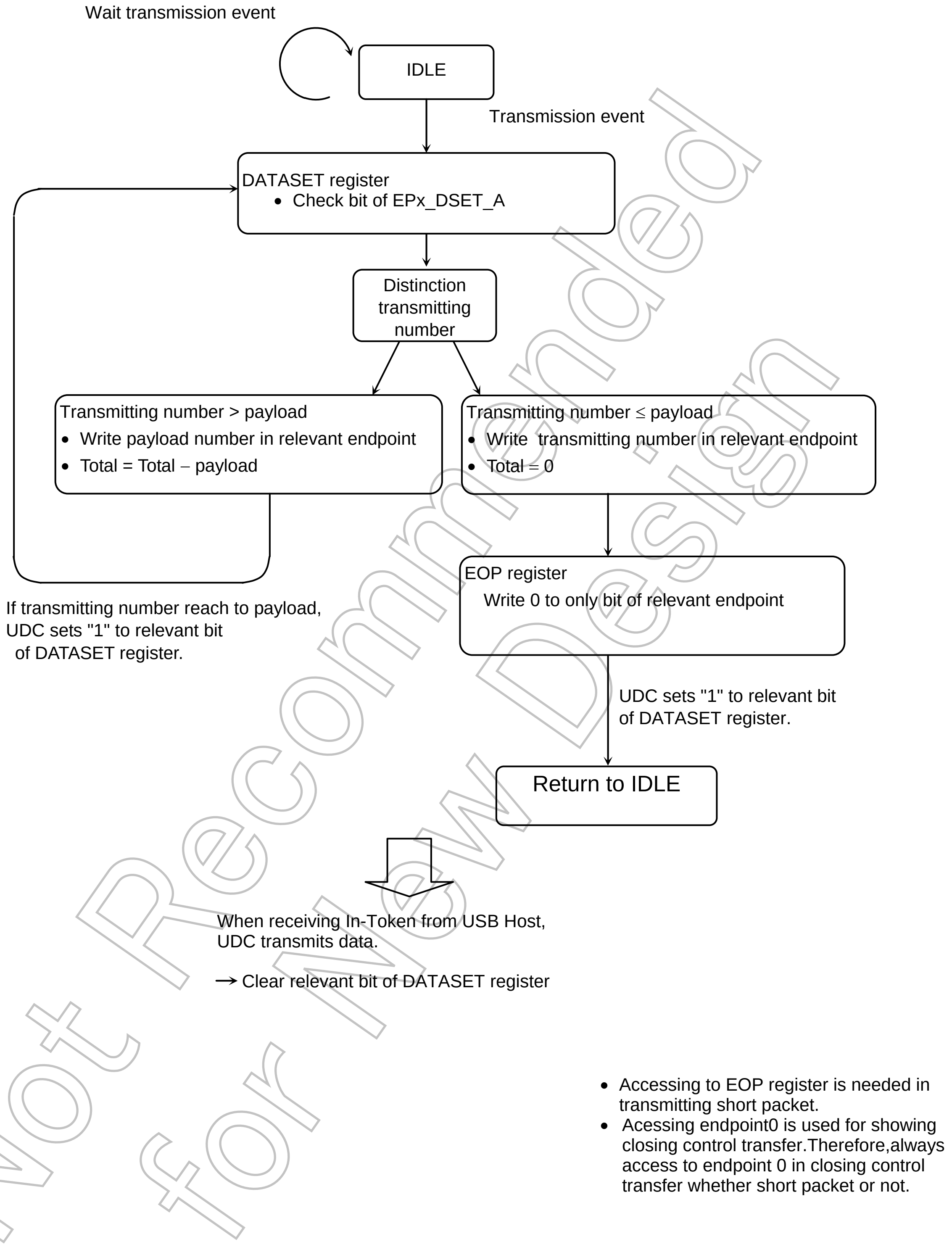

Wait transmission event
IDLE
Transmission event
DATASET register
• Check bit of EPx_DSET_A
Distinction transmitting number
Transmitting number > payload
• Write payload number in relevant endpoint
• Total = Total – payload
Transmitting number ≤ payload
• Write transmitting number in relevant endpoint
• Total = 0
If transmitting number reach to payload, UDC sets "1" to relevant bit of DATASET register.
EOP register
Write 0 to only bit of relevant endpoint
UDC sets "1" to relevant bit of DATASET register.
Return to IDLE
When receiving In-Token from USB Host, UDC transmits data.
→ Clear relevant bit of DATASET register
• Accessing to EOP register is needed in transmitting short packet.
• Acessing endpoint0 is used for showing closing control transfer.Therefore,always access to endpoint 0 in closing control transfer whether short packet or not.

Figure 3.10.14 Transmitting Sequence in Single Packet Mode

(b) Dual packet mode

In dual packet mode, FIFO is divided into A and B packet, and is controlled according to priority in hardware. It can be performed at once, transmitting and receiving data to USB host and exchanges to external of UDC. When it reads out data from FIFO for receiving, confirm condition of two packets, and consider the order of priority. If it has received data to two packets, the UDC outputs from first receiving data by FIFO that can be accessed are common in two packets. EPx_SIZE register is prepared for both packet A and packet B. First, the CPU must recognize the data number of first receiving packet by PKT_ACTIVE bit. If PKT_ACTIVE bit was set to 1, that packet is received first. Packet A and packet B set data turn about always.

This sequence is shown below.

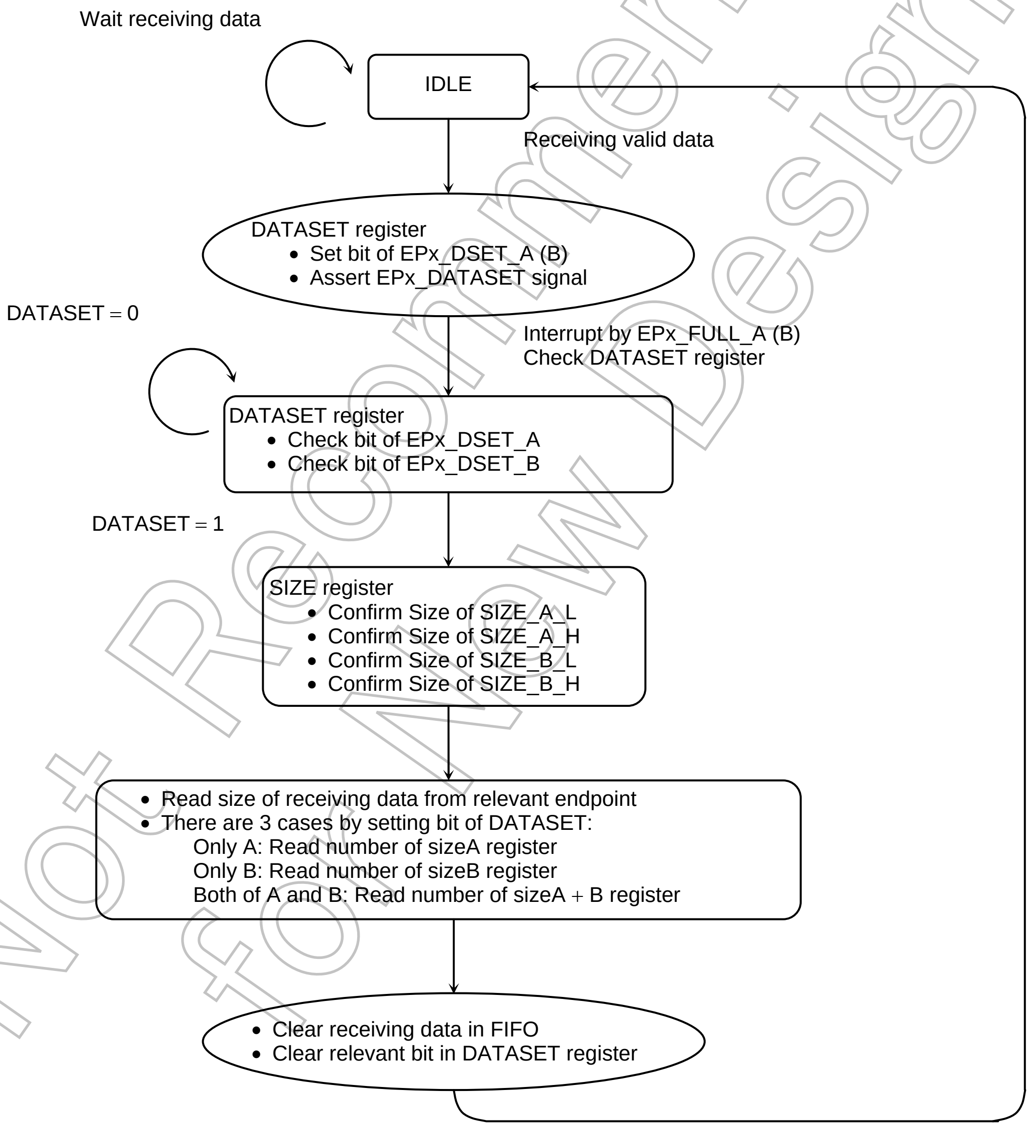

Wait receiving data
IDLE
Receiving valid data
DATASET register
• Set bit of EPx_DSET_A (B)
• Assert EPx_DATASET signal
DATASET = 0
Interrupt by EPx_FULL_A (B)
Check DATASET register
DATASET register
• Check bit of EPx_DSET_A
• Check bit of EPx_DSET_B
DATASET = 1
SIZE register
• Confirm Size of SIZE_A_L
• Confirm Size of SIZE_A_H
• Confirm Size of SIZE_B_L
• Confirm Size of SIZE_B_H
• Read size of receiving data from relevant endpoint
• There are 3 cases by setting bit of DATASET:
Only A: Read number of sizeA register
Only B: Read number of sizeB register
Both of A and B: Read number of sizeA + B register
• Clear receiving data in FIFO
• Clear relevant bit in DATASET register

Figure 3.10.15 Receiving Sequence in Dual Packet Mode

Data can be set to available FIFO when transmitting regardless of packet A or B.

Below is the Transmitting Sequence in Dual Packet Mode.

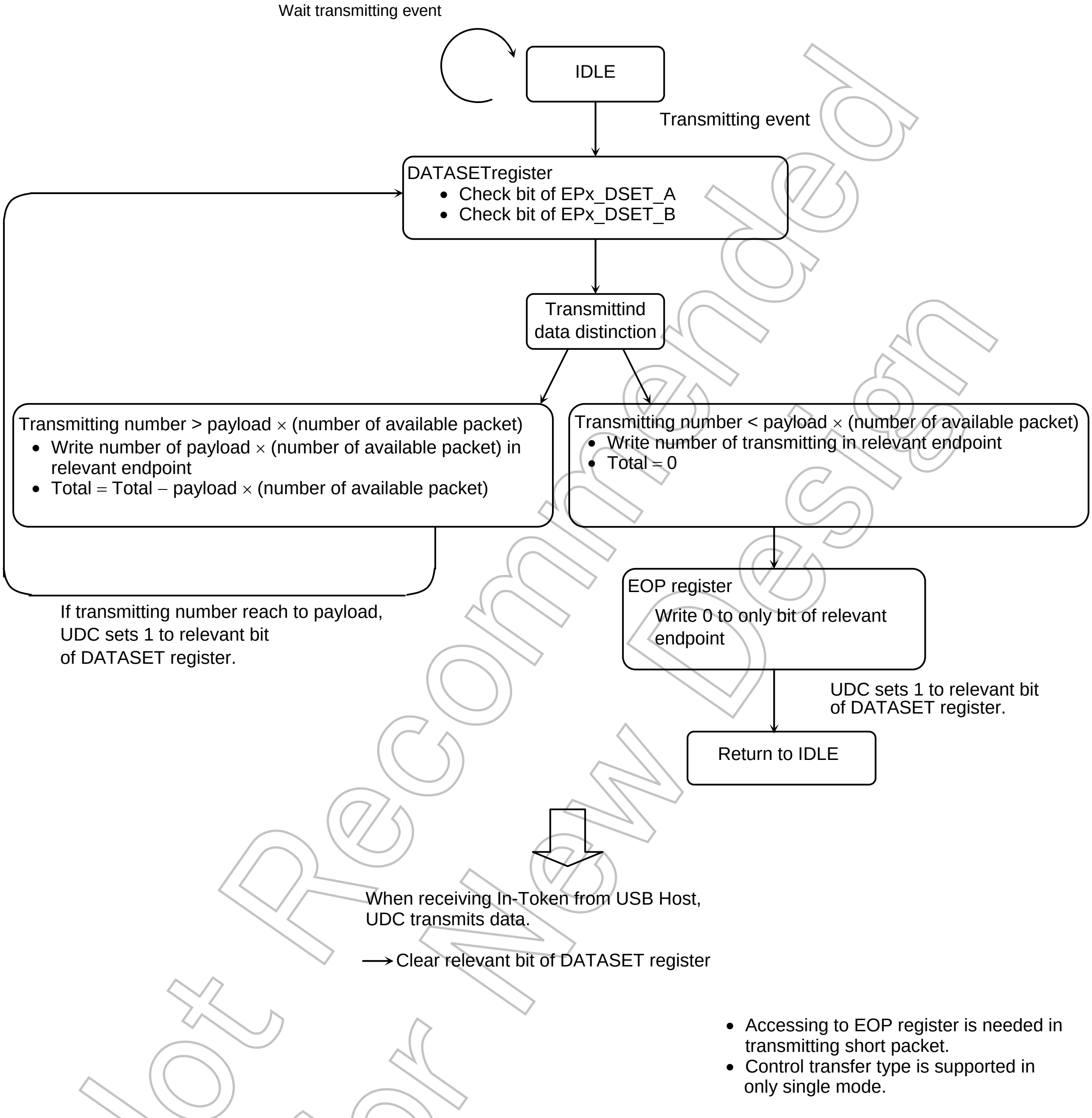

Wait transmitting event
IDLE
Transmitting event
DATASETregister
• Check bit of EPx_DSET_A
• Check bit of EPx_DSET_B
Transmittind data distinction
Transmitting number > payload × (number of available packet)
• Write number of payload × (number of available packet) in relevant endpoint
• Total = Total − payload × (number of available packet)
Transmitting number < payload × (number of available packet)
• Write number of transmitting in relevant endpoint
• Total = 0
If transmitting number reach to payload, UDC sets 1 to relevant bit of DATASET register.
EOP register
Write 0 to only bit of relevant endpoint
UDC sets 1 to relevant bit of DATASET register.
Return to IDLE
When receiving In-Token from USB Host, UDC transmits data.
Clear relevant bit of DATASET register
• Accessing to EOP register is needed in transmitting short packet.
• Control transfer type is supported in only single mode.

Figure 3.10.16 Transmitting Sequence in Dual Packet Mode

(c) Issuance of NULL packet

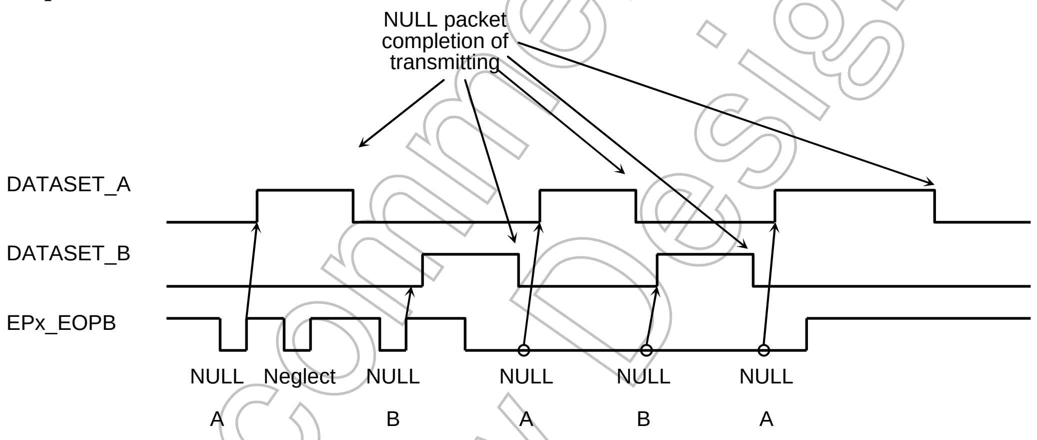
If transmitting NULL packet, by input L pulse from EPx_EOPB signal, data of 0 length is set to FIFO, and NULL packet can be transferred to IN token.

But if NULL data is set to FIFO, it is valid only in the case where SET signal is L level condition (where FIFO is empty). If it answers to receiving IN token by using NULL packet in a certain period, it is answered by keeping EPx_EOPB signal to L level.

However, if mode is dual packet mode, EPx_DATASET signal assert L level for showing space of data. Therefore, data condition (whether either has data or not) cannot be confirmed externally.

Note: NULL packet can also be set by accessing EOP register.

Example:



(2) Interrupt control

Interrupt signal is prepared. This function uses adept system.

For detail refer to 3.10.2 900/H1 CPU I/F.

3.10.8 USB Device answer

The USB controller (UDC) sets various registers and initialization in the UDC in detecting of hardware reset, detecting of USB bus reset, and enumeration answer.

Each condition is explained below.

(1) Bus reset detect condition.

When the UDC detects a bus reset on the USB signal line, it initializes internal register, and it prepares enumeration operation from USB host. After detecting a USB reset, the UDC sets ENDPOINT0 to control transfer type 8-byte payload and default address for using default pipe. Any endpoint other than this is prohibited.

Register name		Initial value
ENDPOINT STATUS	EP0	00H
	Except for EP0	1CH

(2) Detail of STATUS register

Status register that was prepared for each endpoint shows the condition of each endpoint in the UDC.

Each condition affects the various USB transfers. Refer to chapter 5 for the changing conditions for each transfer type.

EPx_STATUS register value is 0 to 3, and its conditions are shown below. 0 to 4 are the results of various transfers. It can be confirmed previous result that is transferred to endpoint by confirming from external of UDC.

0	READY
1	DATAIN
2	FULL
3	TX_ERR
4	RX_ERR

These conditions mean that the endpoint is operating normally. The meaning that is showed is different for each transfer mode. Therefore, please refer to each transfer mode column below.

ISO transfer mode

Below is the transfer condition for the previous frame. Receiving SOF renews this.

	OUT (RX)	IN (TX)
Initial	READY	READY
Not transfer	READY	FULL
Finish normally	DATAIN	READY
Detect anerror	RXERR	TXERR

Transfer modes other than ISO transfer

This is the result of the previous transfer. When transfer is finished, this is renewed.

	OUT, SETUP	IN
Initial	READY	READY
Transfer finish normally	DATAIN	READY
Status stage finish	READY	READY
Transfer error	RXERR	TXERR

"Initial" is that renew RESET, USB reset, Current_Config register. In detect error, it does not generate EPx_DATASET except in toggle transfer mode and Isochronous transfer mode of interrupt.

5 to 7 show the status register means that the endpoint is in special condition.

5 BUSY BUSY is generated only at endpoint of control transfer. If UDC transfer in control writes transfer, when CPU has not finished enumeration transaction, and if it receives ID of status stage from USB host, BUSY is set. STATUS is BUSY until CPU finishes enumeration transaction and EP0 bit of EOP register is written 0 in UDC. If CPU enumeration transaction finishes and EP0 bit of EOP register is written 0 and status stage from USB host finishes normally, it displays READY.

6 STALL STALL shows that endpoint is in STALL condition.

This condition is generated if it violates protocol or error in bus enumeration. To return endpoint to normal transfer condition, USB device request is needed. This request returns to normal condition. But control endpoint returns to normal condition by receiving SETUP token. And it becomes SETUP stage.

7 INVALID This condition shows condition that endpoint cannot be used. UDC sets condition that isn't designated in ENDPOINT to INVALID condition, and it ignores all tokens for this endpoint. In initializing, this condition is always generated. When UDC detects hardware reset, it sets all endpoints to INVALID condition. Next, if USB reset is received, endpoint 0 only is renewed to READY. Other endpoints that are defined on disruptor are renewed if SET_CONFIG request finishes normally.

3.10.9 Power Management

USB controller (UDC) can be switched from optional resume condition (turn on the power supply condition) to suspend (Suspension) condition, and it can be returned from suspend condition to turn on the power supply condition.

This function can be set to low electricity consumption by operating CLK supplying for UDC.

(1) Switch to suspend condition

The USB host can set the USB device to suspend condition by maintaining IDLE state. The UDC switches to suspend condition by the following process.

- UDC switches to suspend condition if it detects IDLE state of more than 3 ms (about 3.07ms) on USB signal. At this point, UDC sets SUSPEND bit of STATUS register to "1".
- UDC renews USBINTFR1<INT_SUS> and <INT_CLKSTOP> from "0" to "1" if it detects IDLE state of more than 5 ms (about 5.46ms) on USB signal. Afterward reset USBCR1<USBCLKE> to "0" to stop USB clock.
- In this condition, all register values in the UDC are kept. However, external access is not possible except for reading of STATUS register, Current_Config register, and USBINTFR1, USBINTFR2, USBINTMR1, USBINTMR2 and USBCR1.

(2) Return from suspend condition by host resume

When activity of bus on USB signal is restored by resume condition output from USB host, the UDC releases SUSPEND condition, and it resets SUSPEND bit of STATUS register to "0".The system is thereby resumed. The resume condition output from the host is maintained for at least 20 ms. Therefore effective protocol occurs on USB signal line after this time has elapsed.

(3) Return from suspend condition by remote wakeup

Remote wakeup is system for prompt resume from suspended USB device to USB host. Some applications do not support remote wakeup. Remote wakeup is also limited using from USB host by bus enumeration.

UDC remote wakeup function can be used when it is permitted.

Setting remote wakeup by bus can be confirmed by bit7 of Current_Config register. When this bit is "1", remote wakeup can be used. Remote wakeup is not disabled by this bit. Therefore, if this bit shows disabled, remote wakeup must not be set. If it fills the conditions, output resumes condition output to USB host by writing USBCR1<WAKEUP> from "1" to "0" of UDC in suspend condition. And it prompts resume from UDC to host. After UDC changes to suspend condition, WAKEUP input is ignored for 2 ms. Therefore, remote wakeup becomes effective when USBINTFR1<INT_SUS> is set to "1".

(4) Low power consumption by control of CLK input signal

When the UDC switches to suspend condition, it stops CLK and switches to low power consumption condition. But as system, this function enables low power consumption by stopping source of CLK. CLK that is supplied to the UDC can be controlled by using USBINTFR1<INT_SUS>, <INT_CLKSTOP> and USBCR1<USBCLKE>.

If UDC switches to suspend condition, USBINTFR1<INT_SUS> is set to "1", and <INT_CLKSTOP> is set to "1". After confirmation, stop CLK supply (USBCLK) by setting "0" to USBCR1<USBCLKE>. If SUSPEND condition is released by resuming from host, supply normal CLK to UDC within 3 ms.

When remote wakeup is used, it is necessary to supply a stable CLK to the UDC before use. When doubler circuit is used as generation source, the above control is needed.

- Return from suspend condition by USB reset (by INT_CLKON interrupt)

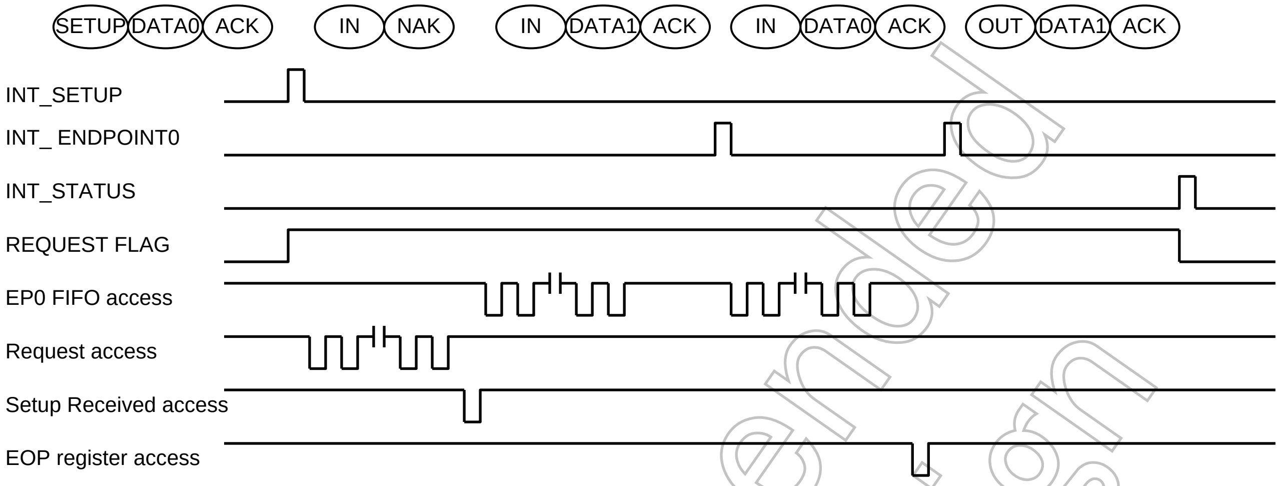
  When UDC stops CLK in suspend condition, UDC can not detect USB reset and control CLK in suspend condition as above mentioned.

  In case CLK is stopped in suspend condition, UDC can detect USB reset and return from suspend condition by supplying CLK (USBCR1<USBCLKE>=1) after detecting INT_CLKON interrupt.

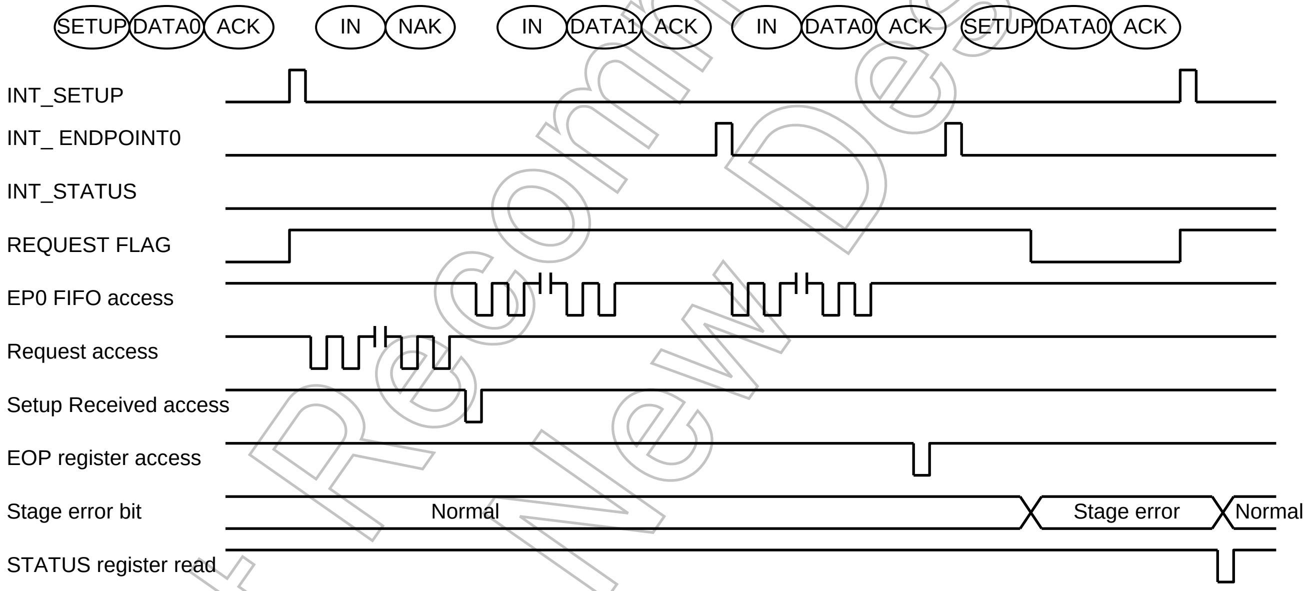
3.10.10 Supplement

(1) External access flow to USB communication

a) Normal movement



b) Stage error



(2) Register initial value

Register Name	Initial Value	Beginning Value
	OUTSIDE Reset	USB_RESET
bmRequestType	0x00	0x00
bRequest	0x00	0x00
wValue_L	0x00	0x00
wValue_H	0x00	0x00
wIndex_L	0x00	0x00
wIndex_H	0x00	0x00
wLength_L	0x00	0x00
wLength_H	0x00	0x00
Current_Config	0x00	0x00
Standard request	0x00	0x00
Request	0x00	0x00
DATASET	0x00	0x00
Port Status	0x18	Hold
Standard request mode	0x00	Hold
Request mode	0x00	Hold

Register Name	Initial Value	Initial Value
	OUTSIDE Reset	USB_RESET
INT control	0x00	0x00
USBBUFF_TEST	0x00	Hold
USB state	0x01	0x01
EPx_MODE	0x00	0x00
EPx_STATUS	0x1C	0x1C
EPx_SIZE_L_A	0x88	0x88
EPx_SIZE _L_B	0x08	0x08
EPx_SIZE_H_A	0x00	0x00
EPx_SIZE_H_B	0x00	0x00
FRAME_L	0x00	0x00
FRAME_H	0x02	0x02
ADRESS	0x00	0x00
EPx_SINGLE	0x00	Hold
EPx_BCS	0x00	Hold
ID_STATE	0x01	0x00

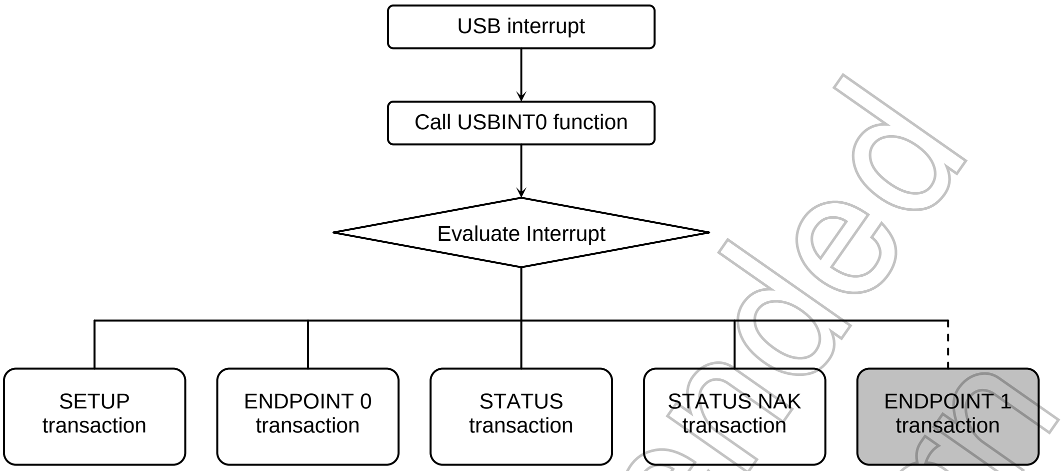
Note 1: The above initial value is the value that is initialized by external reset, USB_RESET. This value may differ from that displayed depending on conditions.
Please refer to register configure in chapter 2.

Note 2: EP0_STATUS register is initialized to 0x00 after USB_RESET is received.

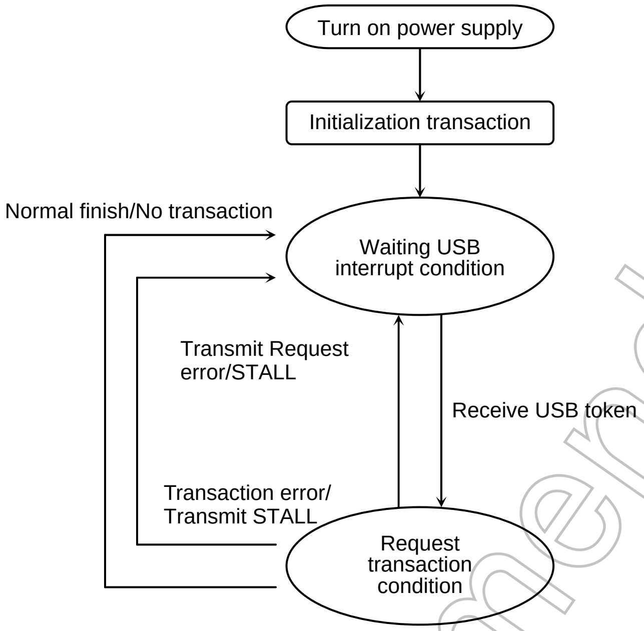
Note 3: Initial value of ID_STATE register is initialized by external reset, BRESET. When USB_RESET signal is received from host, it is initialized to 0x00.

(3) USB control flow chart

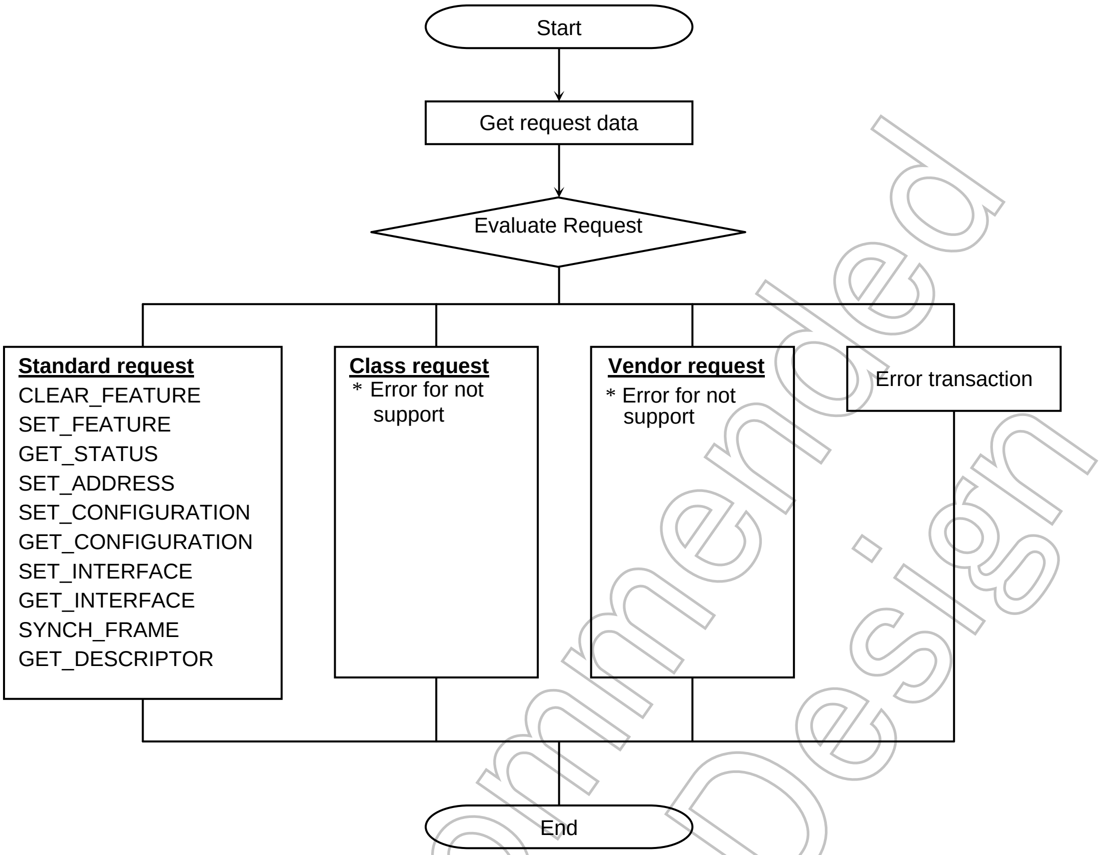
(a) Transaction for standard request (Outline flowchart (Example))



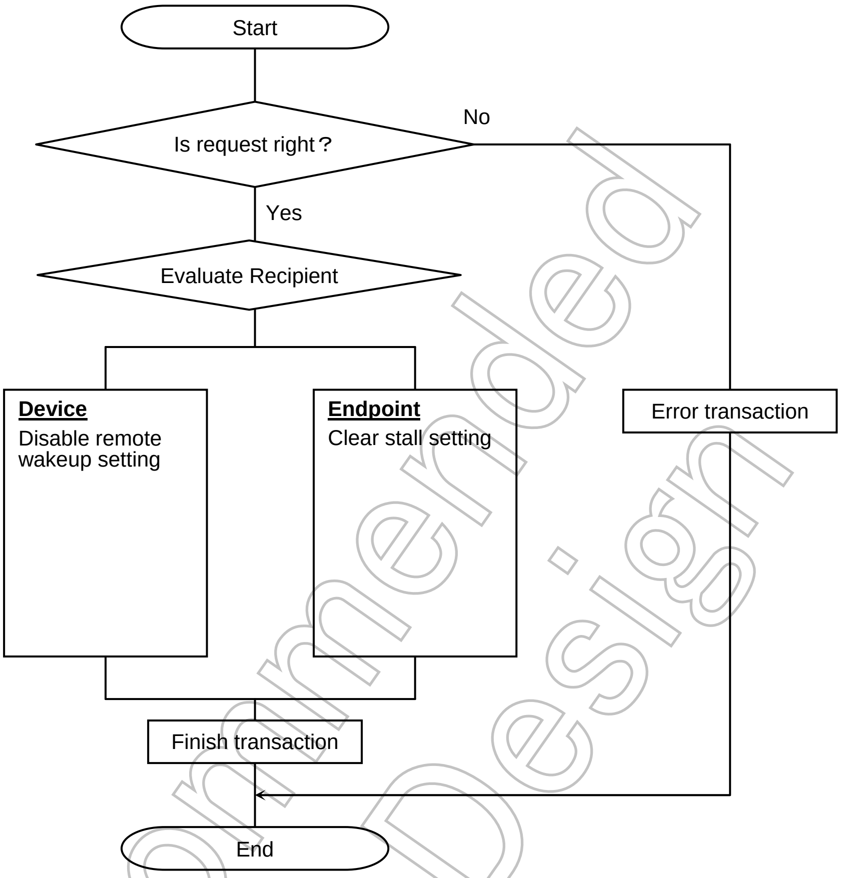
(b) Condition change


Turn on power supply
Initialization transaction
Normal finish/No transaction
Waiting USB interrupt condition
Transmit Request error/STALL
Receive USB token
Transaction error/ Transmit STALL
Request transaction condition

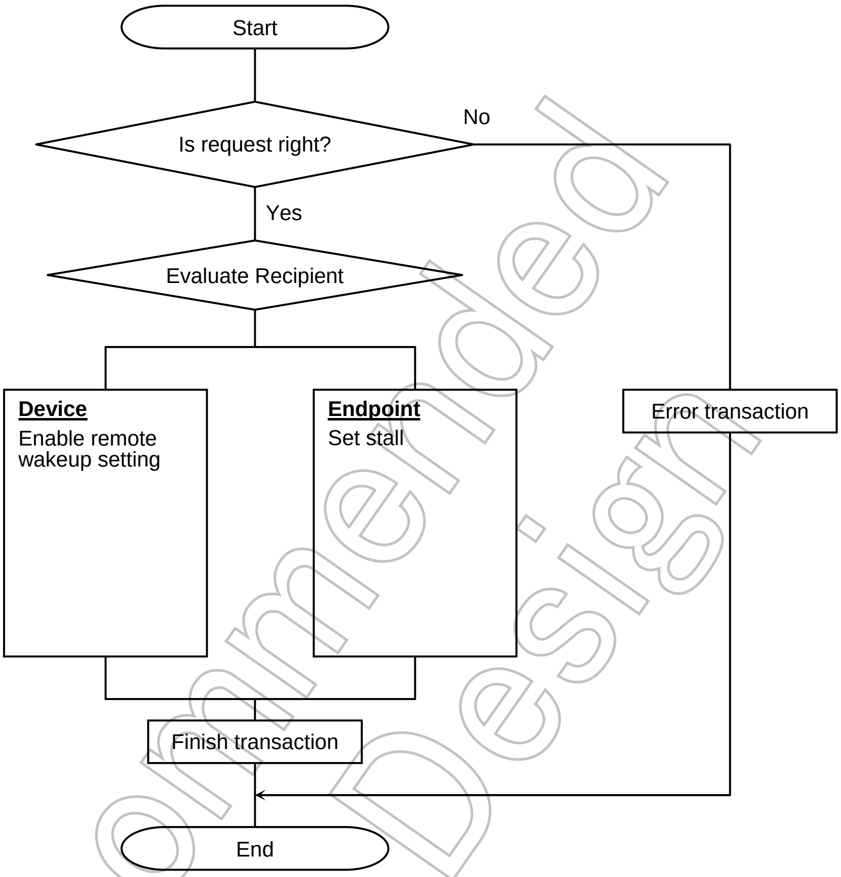
(c) Device request and evaluation of various requests



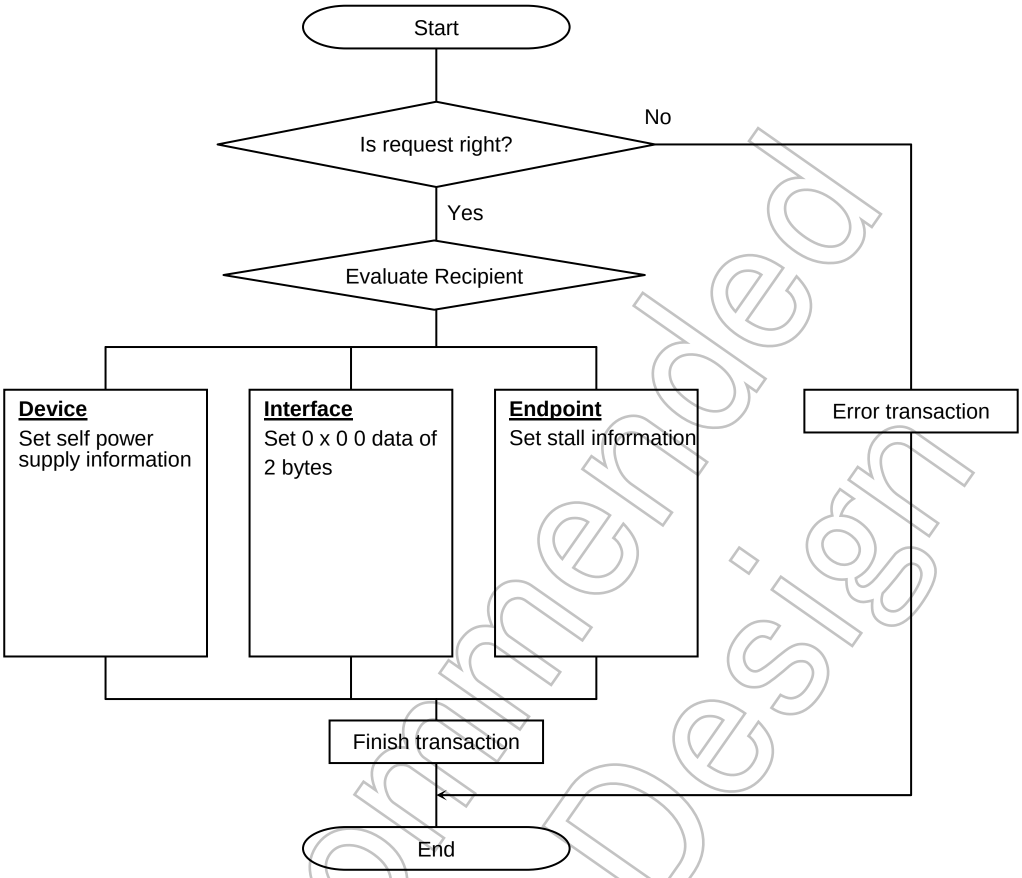
(c-1) CLEAR_FEATURE request transaction



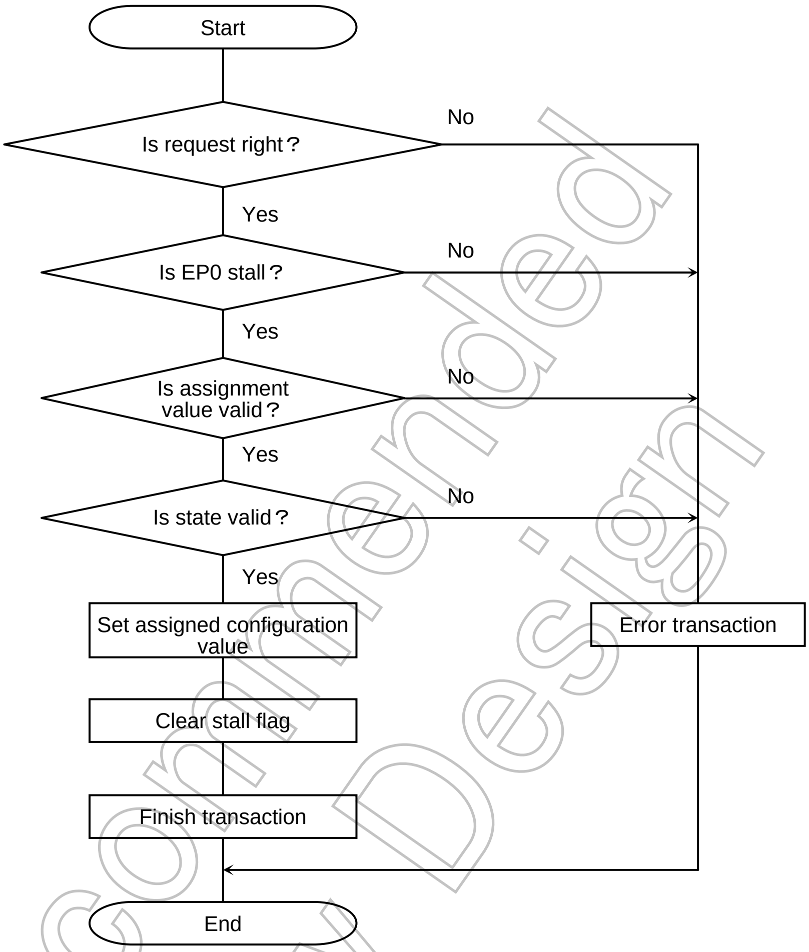
(c-2) SET_FEATURE request transaction



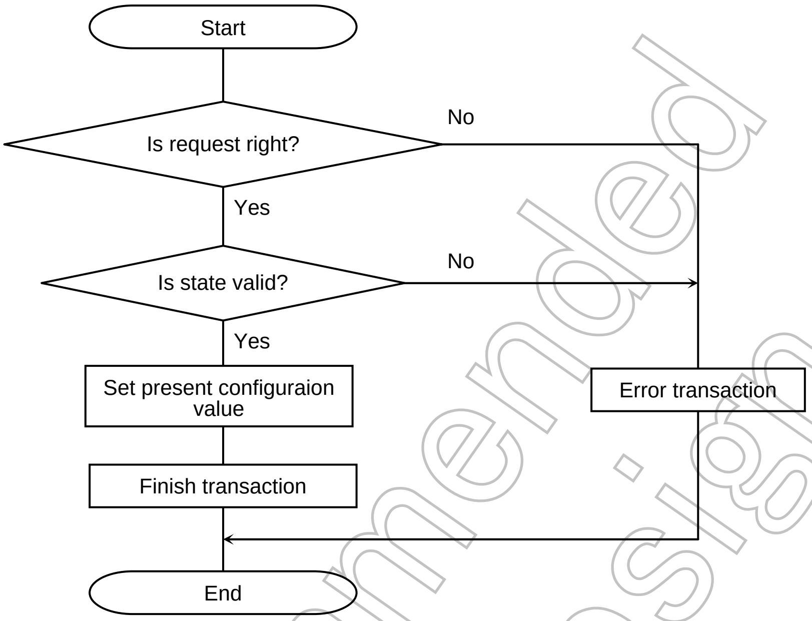
(c-3) GET_STATUS request transaction



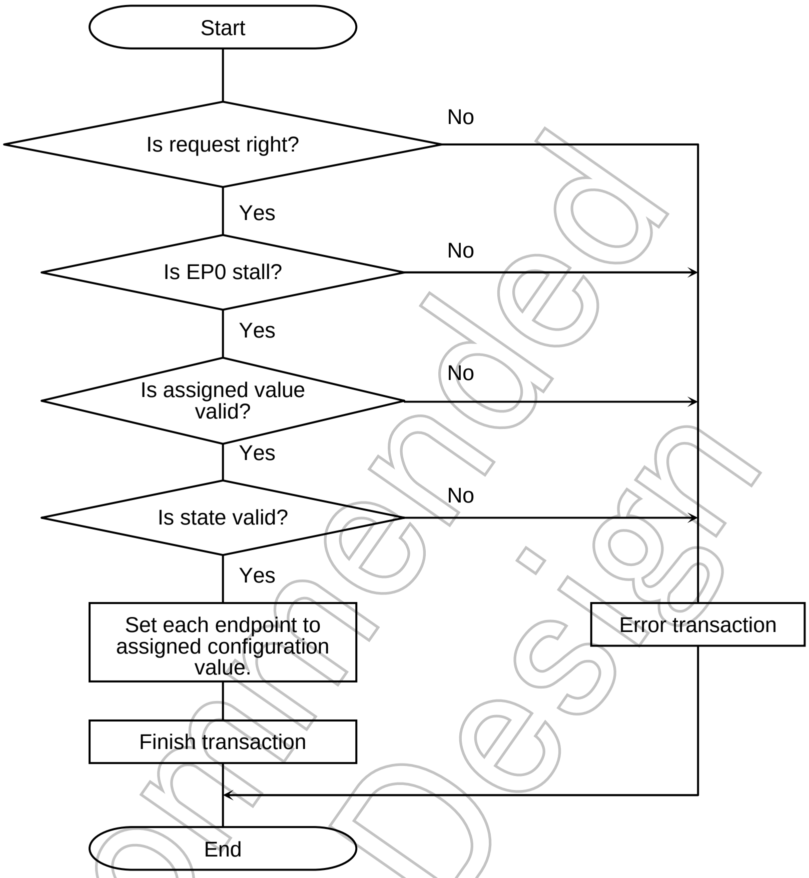
(c-4) SET_CONFIGRATION request transaction



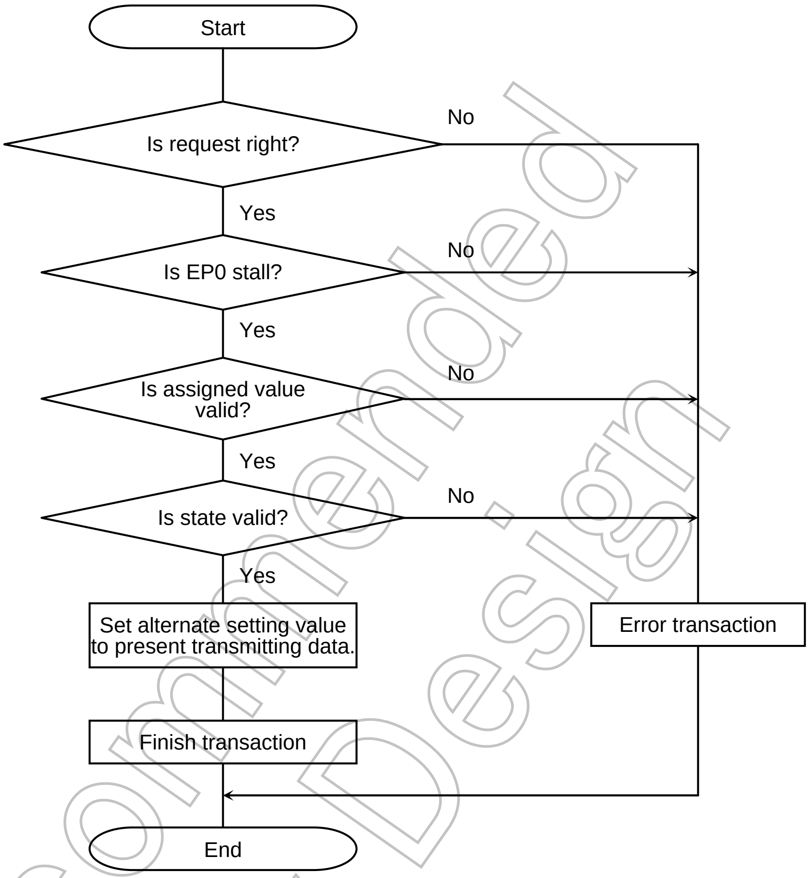
(c-5) GET_CONFIGRATION request transaction



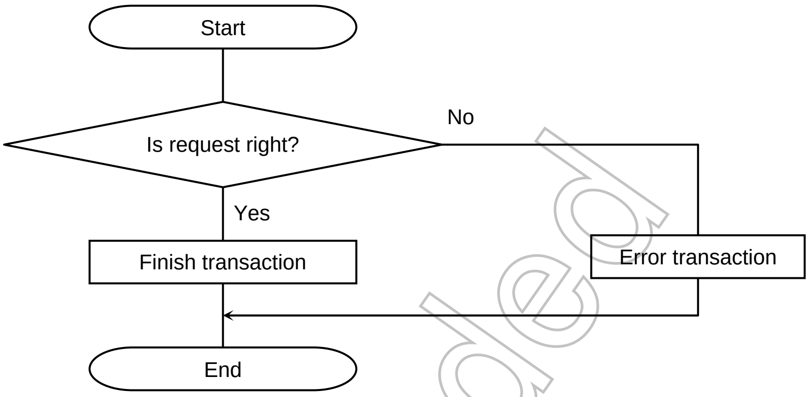
(c-6) SET_INTERFACE request transaction



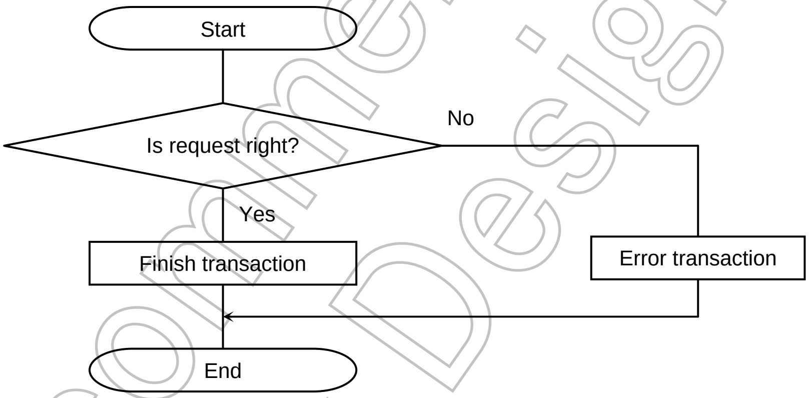
(c-7) SYNCH_FRAME request transaction



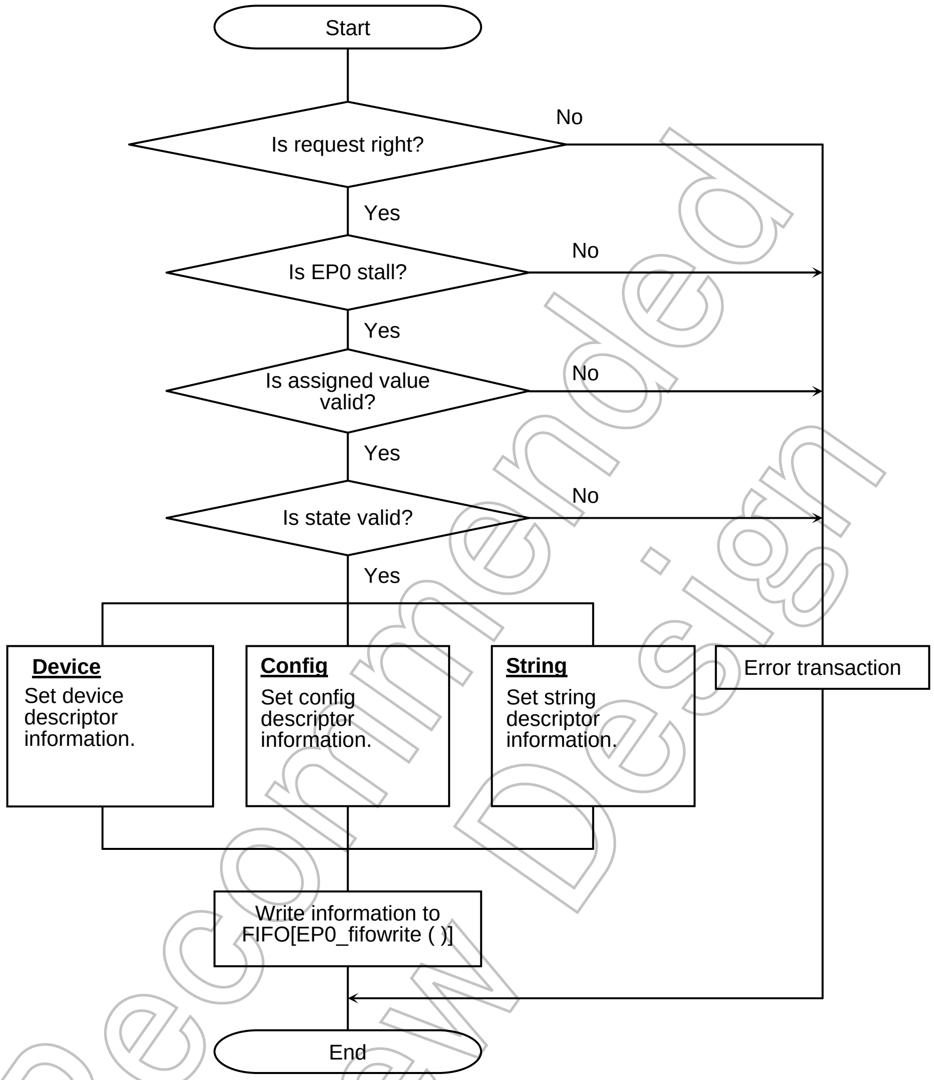
(c-8) SYNCH_FRAME request transaction



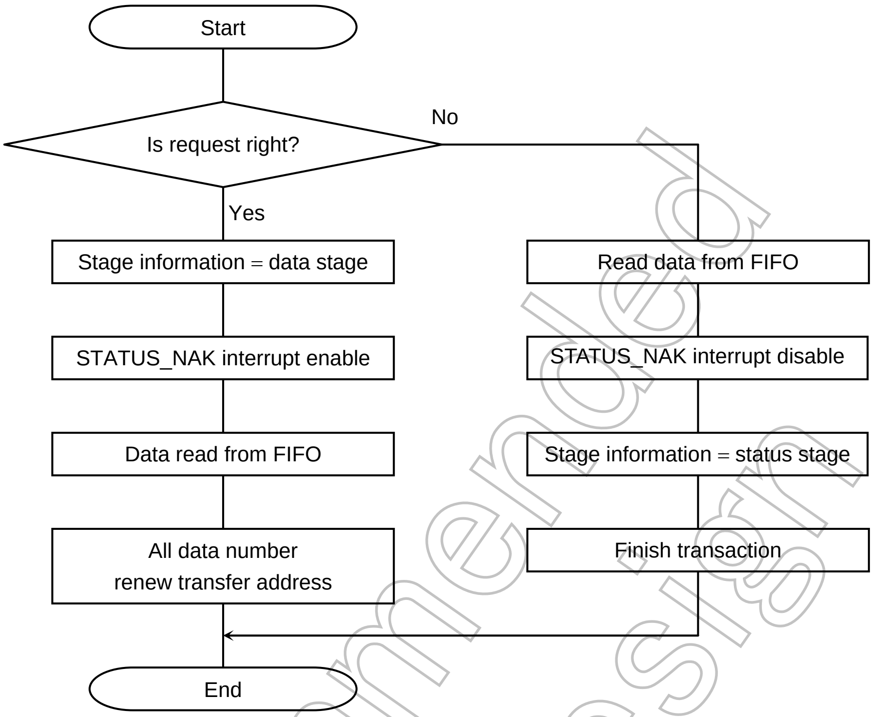
(c-9) SET_DESCRIPTOR request transaction



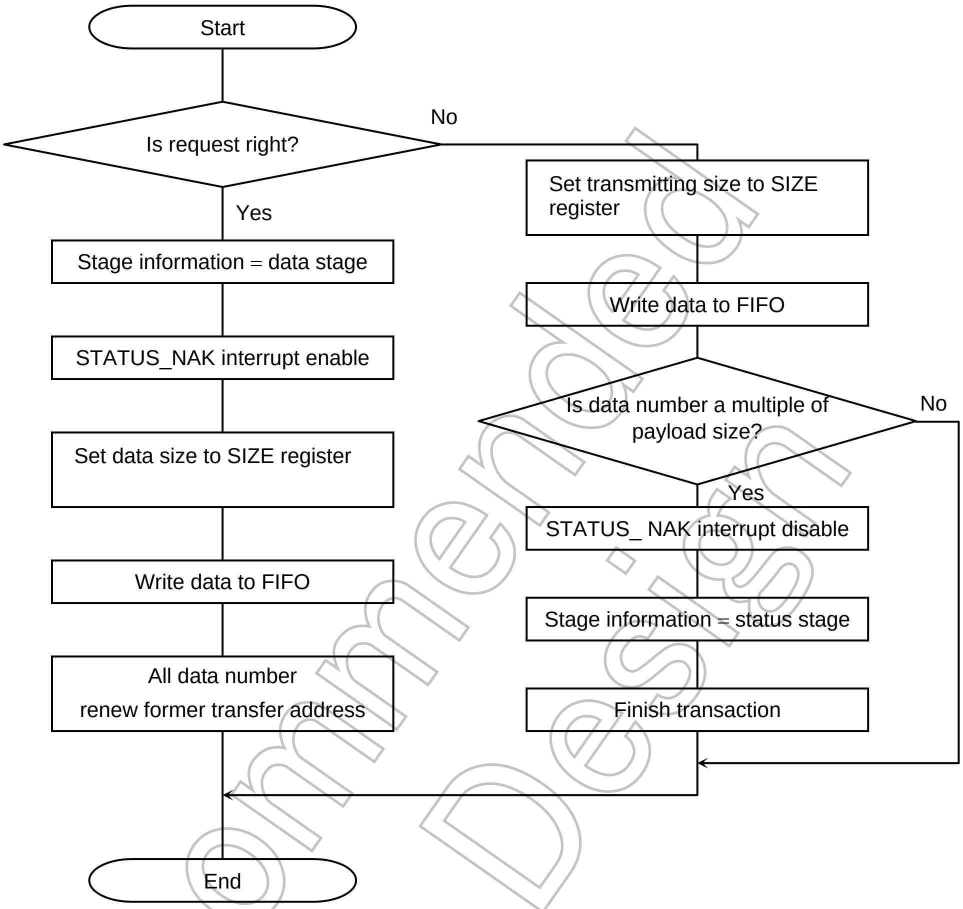
(c-10) GET_DESCRIPTOR request transaction


Start
Is request right?
No
Yes
Is EP0 stall?
No
Yes
Is assigned value valid?
No
Yes
Is state valid?
No
Yes
Device
Set device descriptor information.
Config
Set config descriptor information.
String
Set string descriptor information.
Error transaction
Write information to FIFO[EP0_fifowrite ( )]
End

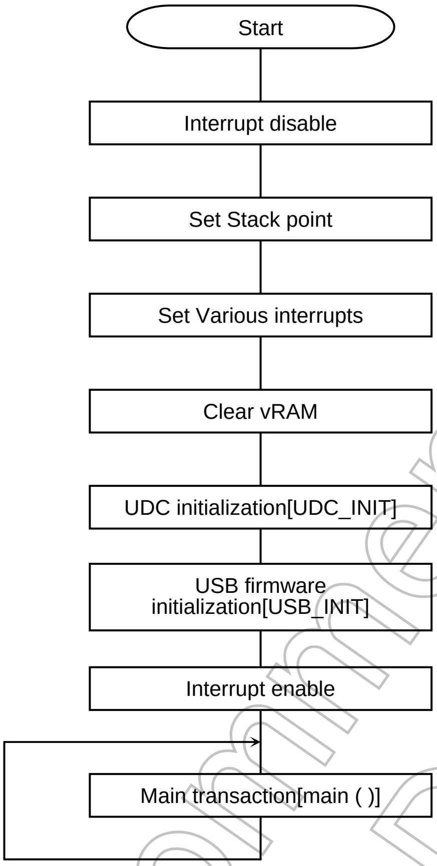
(c-11) Data read transaction to FIFO by EP0



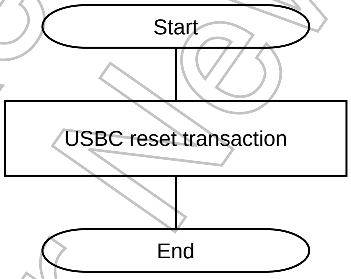
(c-12) Data write transaction to FIFO by EP0


Start
Is request right?
No
Yes
Stage information = data stage
STATUS_NAK interrupt enable
Set data size to SIZE register
Write data to FIFO
All data number
renew former transfer address
Set transmitting size to SIZE register
Write data to FIFO
Is data number a multiple of payload size?
No
Yes
STATUS_ NAK interrupt disable
Stage information = status stage
Finish transaction
End

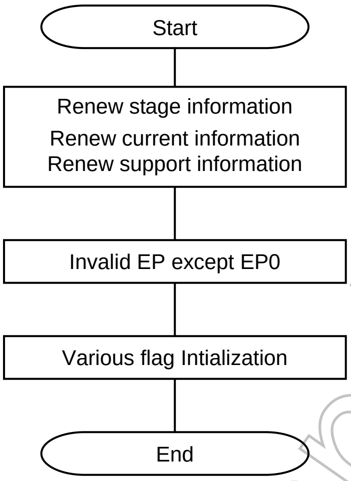
(c-13) Initial setting transaction of microcontroller


Start
Interrupt disable
Set Stack point
Set Various interrupts
Clear vRAM
UDC initialization[UDC_INIT]
USB firmware initialization[USB_INIT]
Interrupt enable
Main transaction[main ( )]

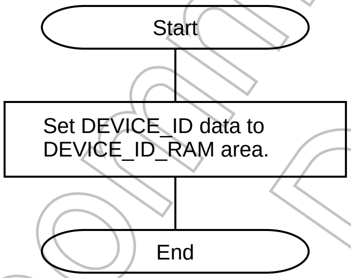
(c-14) Initial setting transaction of UDC



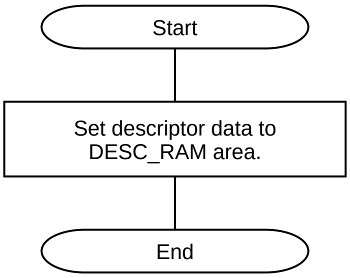
(c-15) Initial transaction of USB number changing firmware


Start
Renew stage information
Renew current information
Renew support information
Invalid EP except EP0
Various flag Intialization
End

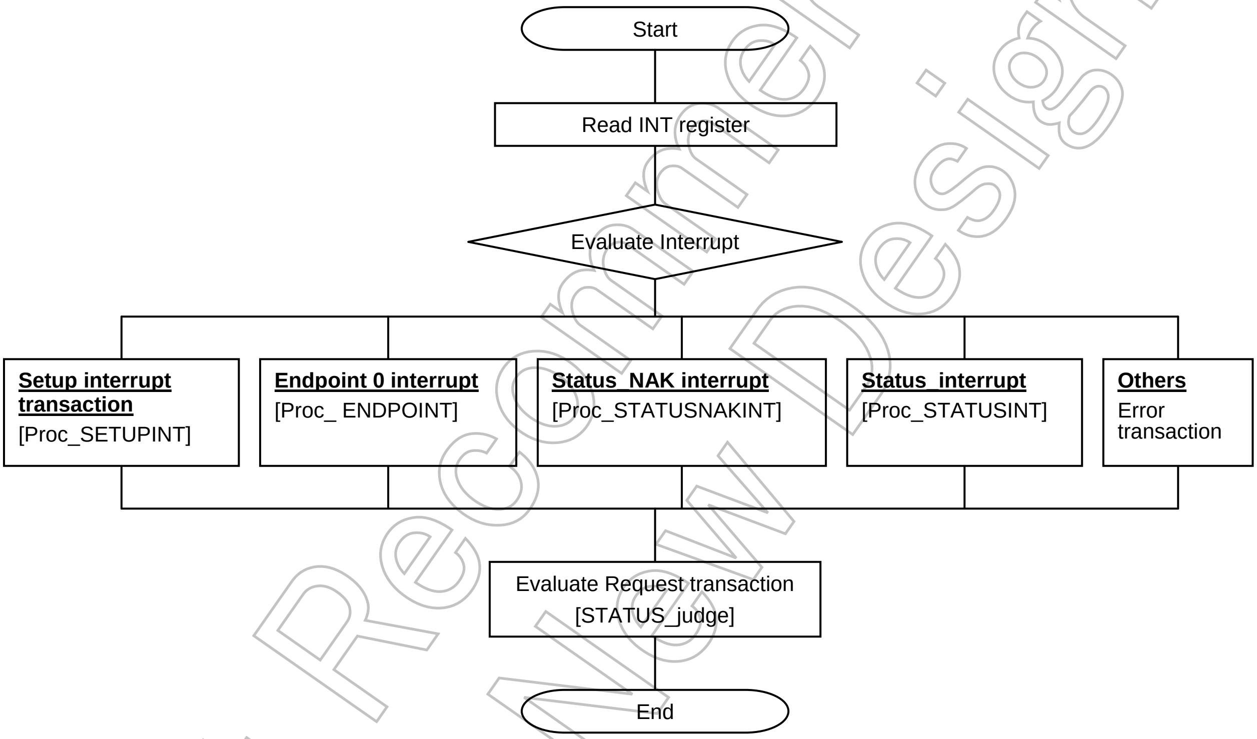
(c-16) Set DEVICE_ID data to DEVICE_ID of UDC



(c-17) Descriptor data set transaction



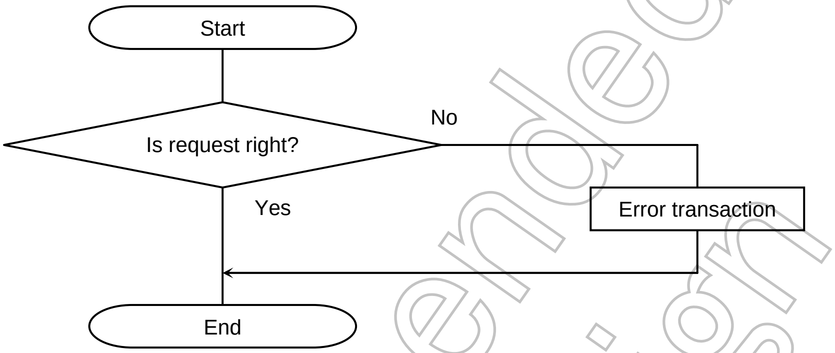
(c-18) USB interrupt transaction



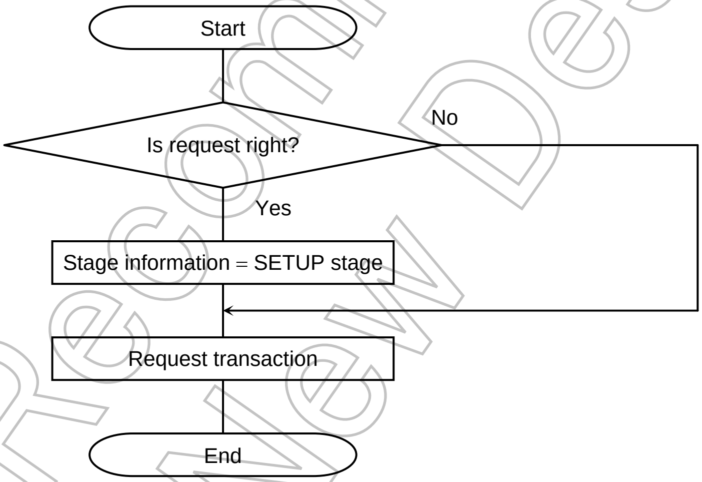
(c-19) Dummy function for not using maskable interrupts.

- Transaction performs nothing, therefore outline flow is skipped.

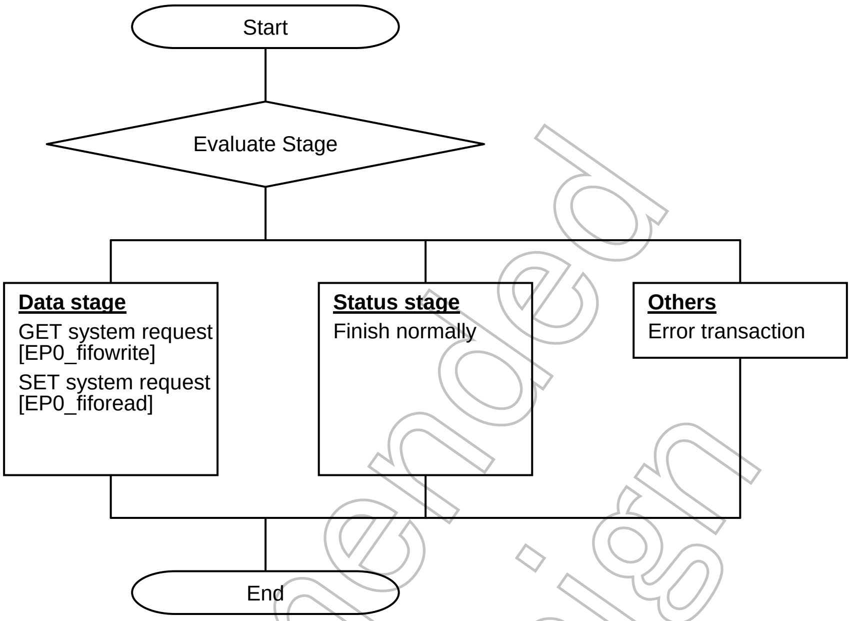
(c-20) Request evaluation transaction. If transaction result is error, it initiates STALL command.



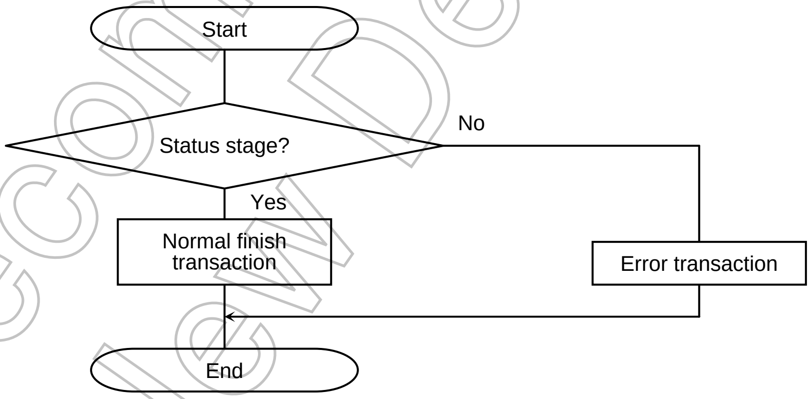
(c-21) SETUP stage transaction



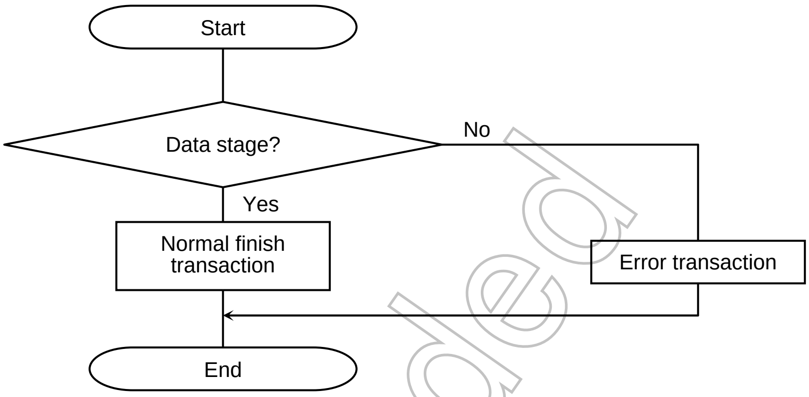
(c-22) Perform endpoint 0 transaction except in SETUP stage.



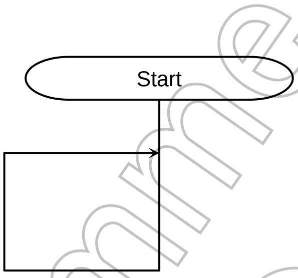
(c-23) Status stage interrupt transaction



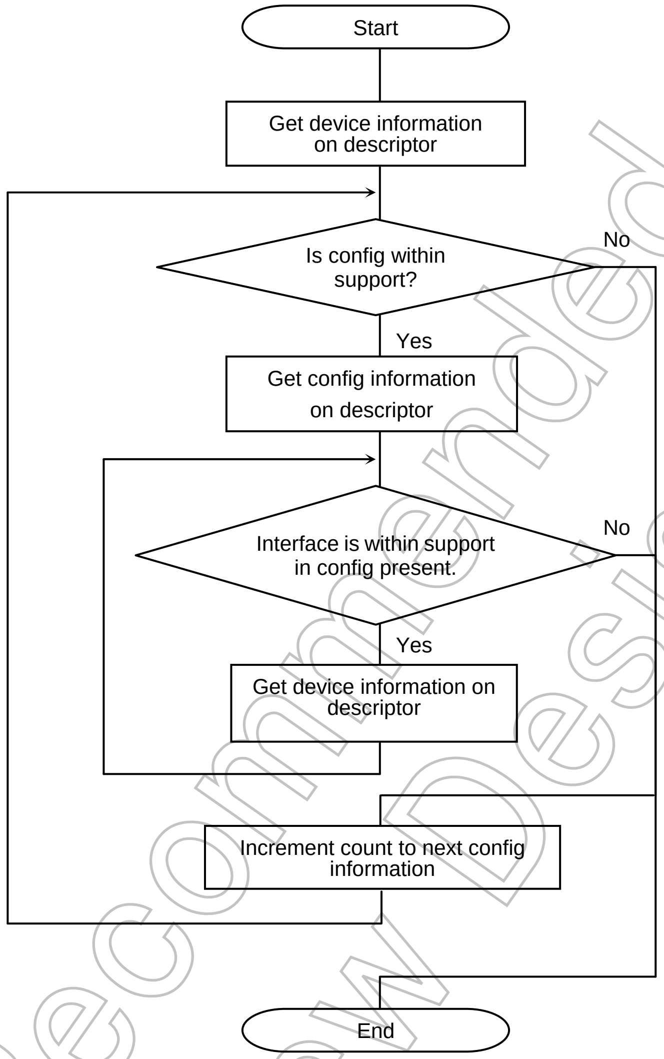
(c-24) STATUS_NAK interrupt transaction



(c-25) This transaction is a non-transaction for USB interrupts.



(c-26) Getting descriptor information (related to standard request)


Start
Get device information on descriptor
Is config within support?
No
Yes
Get config information on descriptor
Interface is within support in config present.
No
Yes
Get device information on descriptor
Increment count to next config information
End

3.10.11 Notice and Restrictions

1. Limitation of writing to COMMAND register in special timing

When “STALL” command is issued, ENDPOINT status might shift to “INVALID”. To avoid this problem, follow the routine below.

a. BULK (IN/OUT)

When issuing a STALL command to endpoint in BULK transfer, be sure to issue STALL command after stopping RD/WR access to endpoint; that is UDC returns NAK in response to token from host. INT_EPxNAK should be used to detect NAK transmit.

b. CONTROL OUT with data stage (software response)

If STALL needs to be set for endpoint 0 judging from request after receiving INT_SETUP interrupt, access SetupReceived register. After that, issue STALL command after detecting INT_ENDPOINT0 interrupt.

c. CONTROL OUT without data stage (software response)

If STALL needs to be set for endpoint 0 judging from request after receiving INT_SETUP interrupt, issue STALL command before access to eop register.

d. CONTROL IN(software response)

If STALL needs to be set for endpoint 0 judging from request after receiving INT_SETUP interrupt, issue STALL command before setting the first transmit data to host.

2. Limitation of EPx_STATUS<STATUS2:0> when executing USB_RESET command

EPx_STATUS<STATUS2:0> may indicate different condition, if aUSB_RESET command is executed to the endpoint processing the token. To avoid this phenomenon, do not RESET the endpoint while transferring. (It is available when processing a request that needs USB_RESET to that endpoint.)

3. When generating toggle error of device controller

   a. UDC operation

   If USB host fails to receive ACK transmitted from the UDC in OUT transfer, the USB host transmits the same data to the UDC again. When the FIFO is available to receive, the UDC detects toggle error because of detecting the same data(having the same toggle as the data which is received just before) and returns ACK. The UDC rejects it because the data have already been received normally. Meanwhile, if FIFO is not available, the UDC returns NAK and informs the USB host that it is unable to receive.

   b. USB1.1 Standard (from USB1.0 Standard description)

   The priority of each process in USB1.0 and USB1.1 standard is explained as follows in chapter “8.4.5.3 Function Response to an OUT Transaction”. It shows the priority of ACK response by toggle error (SequenceBitsMatch=No) is higher than that of NAK response.

Table 3.10.4 Function Responses to OUT Transactions in Order of Precedence

Data Packet Corrupted	Receiver Halt Feature	Sequence Bits Match	Function Can Accept Data	Handshake Returned by Function
Yes	N/A	N/A	N/A	None
No	Set	N/A	N/A	STALL
No	Not set	No	N/A	ACK
No	Not set	Yes	Yes	ACK
No	Not set	Yes	No	NAK

Since the UDC gives priority to detecting FIFO condition over toggle error, the UDC returns NAK in the response to USB host when FIFO is not available because it is full. This is shown in the flow chart “3.10.6(a-2) Receiving bulk mode”. Thus, the UDC operates differently from USB standard under conditions where FIFO is not available.

For that reason, the UDC may generate the retry process several times in case of toggle error, while USB standard finishes it after the first time.

That is, the UDC returns NAK if it receives the data including toggle error with FIFO full. However, after FIFO becomes available, the UDC returns ACK to the USB host and finishes the retry process.

4. When using the USB device controller in the TMP92CH21, a crystal oscillator is recommended (USB standard ≤ 9 MHz±2500ppm). In this case, a maximum of 3 stages of external hub can be used due to the precision of this USB device controller and the internal clock.

5. Limitation of using the Isochronous IN transfer

   When using the Isochronous IN transfer, do not use other endpoints simultaneously.

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 to AN3) are shared with the input only port G so they can be used as an input port.

Note: When IDLE2, IDLE1 or STOP mode is selected, in order to reduce power consumption, the system may enter a stand-by mode with some timings 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.

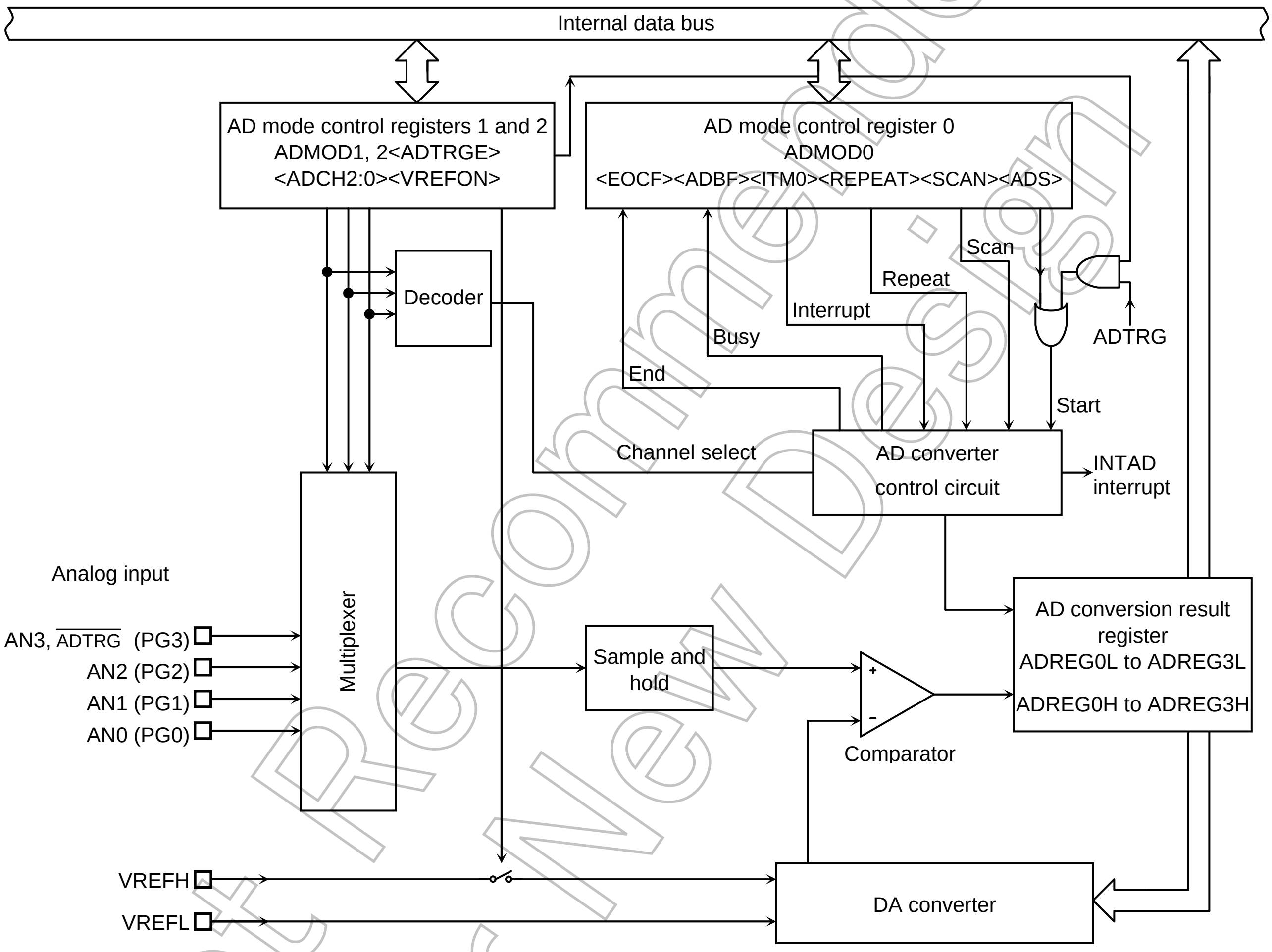


Figure 3.11.1 Block Diagram of AD Converter

3.11.1 Analog/Digital Converter Registers

The AD converter is controlled by the three AD mode control registers: ADMOD0, ADMOD1 and ADMOD2. The four 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					
Reset State	0	0	0	0	0	0	0	0
Function	AD conversion end flag 0: Conversion in progress 1: Conversion complete	AD conversion busy flag 0: Conversion stopped 1: Conversion in progress	Always write “0”	Always write “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

		7	6	5	4	3	2	1	0
ADMOD1 (12B9H)	Bit symbol	VREFON	I2AD	–	–	–	–	ADCH1	ADCH0
	Read/Write	R/W	R/W	R/W					
	Reset State	0	0	0	0	0	0	0	0
	Function	VREF application control 0: Off 1: On	IDLE2 0: Stop 1: Operate	Always write "0"	Always write "0"	Always write "0"	Always write "0"	Analog input channel selection	

Analog input channel selection

<SCAN> <ADCH1:0>	0 (Channel fixed)	1 (Channel scanned)
00	AN0	AN0
01	AN1	AN0→AN1
10	AN2	AN0→AN1→AN2
11 (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

		7	6	5	4	3	2	1	0
ADMOD2 (12BAH)	Bit symbol			–	–	–	–	–	ADTRGE
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Always write "0"	Always write "0"	Always write "0"	Always write "0"	Always write "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 functions as the $\overline{\text{ADTRG}}$ input pin, do not set < ADCH1:0 > = "11" when using $\overline{\text{ADTRG}}$ with < ADTRGE > set to "1".

Figure 3.11.3 AD Converter Related Register

AD Conversion Result Register 0 Low

		7	6	5	4	3	2	1	0
ADREG0L (12A0H)	Bit symbol	ADR01	ADR00						ADR0RF
	Read/Write	R							R
	Reset State	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

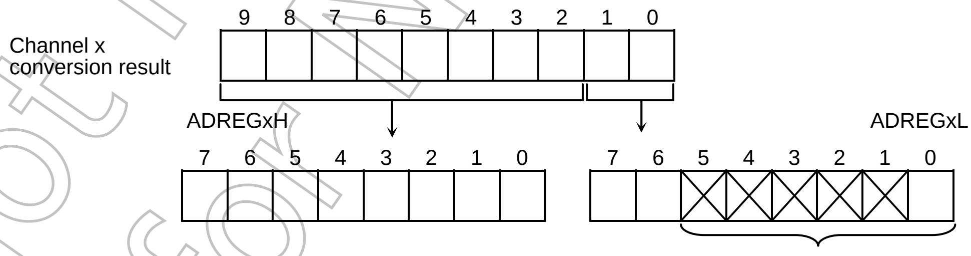
		7	6	5	4	3	2	1	0
ADREG0H (12A1H)	Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
	Read/Write	R							
	Reset State	Undefined							
	Function	Stores upper 8 bits of AD conversion result.							

AD Conversion Result Register 1 Low

		7	6	5	4	3	2	1	0
ADREG1L (12A2H)	Bit symbol	ADR11	ADR10						ADR1RF
	Read/Write	R							R
	Reset State	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

		7	6	5	4	3	2	1	0
ADREG1H (12A3H)	Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
	Read/Write	R							
	Reset State	Undefined							
	Function	Stores upper 8 bits of AD conversion result.							



- Bits 5 to 1 are always read as 1.
- Bit0 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

		7	6	5	4	3	2	1	0
ADREG2L (12A4H)	Bit symbol	ADR21	ADR20						ADR2RF
	Read/Write	R							R
	Reset State	Undefined							0
	Function	Stores lower 2 bits of AD conversion result.							AD conversion data storage flag 1: Conversion result stored

AD Conversion Result Register 2 High

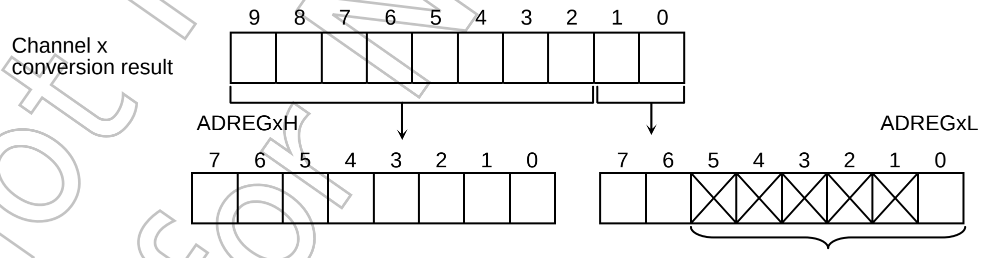
		7	6	5	4	3	2	1	0
ADREG2H (12A5H)	Bit symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R							
	Reset State	Undefined							
	Function	Stores upper 8 bits of AD conversion result.							

AD Conversion Result Register 3 Low

		7	6	5	4	3	2	1	0
ADREG3L (12A6H)	Bit symbol	ADR31	ADR30						ADR3RF
	Read/Write	R							R
	Reset State	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

		7	6	5	4	3	2	1	0
ADREG3H (12A7H)	Bit symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
	Read/Write	R							
	Reset State	Undefined							
	Function	Stores upper 8 bits of AD conversion result.							



- Bits 5 to 1 are always read as 1.
- Bit0 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 depending on the operation mode of the AD converter.

- In analog input channel fixed mode (ADMOD0<SCAN> = 0)
  Setting ADMOD1<ADCH1:0> selects one of the input pins AN0 to AN3 as the input channel.
- In analog input channel scan mode (ADMOD0<SCAN> = 1)
  Setting ADMOD1<ADCH1:0> selects one of the four 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<ADCH1:0> is initialized to 00. 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

<ADCH1:0>	Channel Fixed <SCAN> = "0"	Channel Scan <SCAN> = "1"
00	AN0	AN0
01	AN1	AN0 → AN1
10	AN2	AN0 → AN1 → AN2
11	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 AD 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 AD 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<REPEAT> 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<REPEAT> 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<REPEAT> 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<REPEAT> 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<REPEAT> 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 (e.g., in cases 3. and 4.), write a 0 to ADMOD0<REPEAT>. 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 (e.g., in cases 3. and 4.), when the halt is released, conversion restarts from the beginning. In single conversion modes (e.g., 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 fourth conversion	1		
Channel scan repeat conversion mode	After completion of every scan conversion	X	1	1

X: Don’t care

(5) AD conversion time

132 states (6.6 μs at 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 ADREG3H/L) store the results of AD conversion. (ADREG0H/L to ADREG3H/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 and 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 (ADMOD0<ITM0 = 1>)
AN0	ADREG0H/L	ADREG0H/L ←
		↓
AN1	ADREG1H/L	ADREG1H/L
		↓
AN2	ADREG2H/L	ADREG2H/L
		↓
AN3	ADREG3H/L	ADREG3H/L

<ADRxRF>, bit0 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-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2800H)	←	WA	Write contents of WA to memory address 2800H.

2. This example repeatedly converts the analog input voltages on the three pins AN0, AN1 and AN2, using channel scan repeat conversion mode.

INTE0AD	←	1	0	0	0	–	–	–	–	Disable INTAD.
ADMOD1	←	1	1	0	0	0	0	1	0	Set pins AN0 to AN2 to be the analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Start conversion in channel scan repeat conversion mode.

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.
(The level of external $\overline{\text{RESET}}$ pin is not changed.)

3.12.1 Configuration

Figure 3.12.1 is a block diagram of the watchdog timer (WDT).

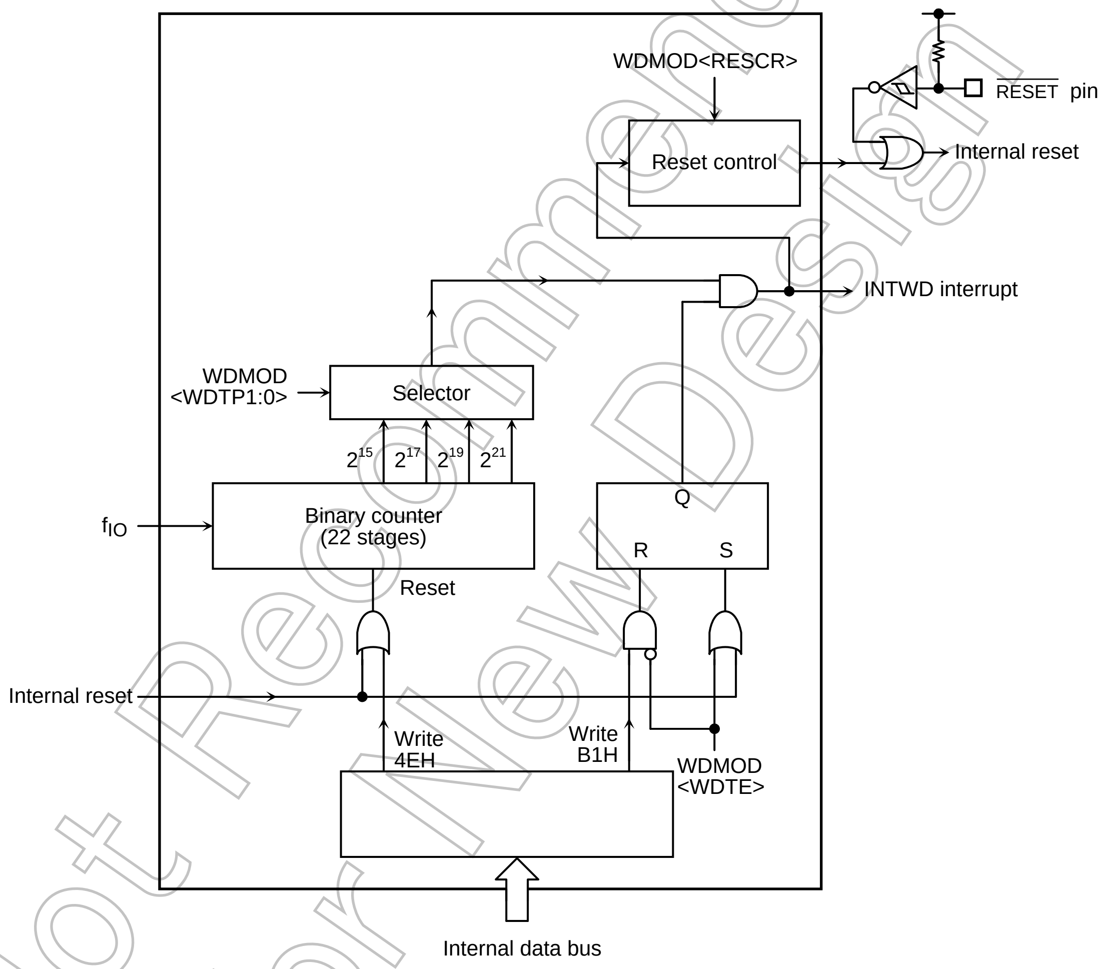


Figure 3.12.1 Block Diagram of Watchdog Timer

Note: Care must be exercised in the overall design of the apparatus since the watchdog timer may fail to function correctly due to external noise, etc.

3.12.2 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1:0> has elapsed. The watchdog timer must be cleared to zero in software before an INTWD interrupt will be generated. If the CPU malfunctions (e.g., if runaway occurs) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter will overflow and an INTWD interrupt will be generated. The CPU will detect malfunction (runaway) due to the INTWD interrupt, and in this case it is possible to return the CPU to normal operation by means of an anti-malfunction program.

The watchdog timer begins operating immediately on release of the watchdog timer reset.

The watchdog timer is reset and halted in IDLE1 or STOP mode. The watchdog timer counter continues counting during bus release (when $\overline{\text{BUSAK}}$ goes low).

When the device is in IDLE2 mode, the operation of the WDT depends on the WDMOD<I2WDT> setting. Ensure that WDMOD<I2WDT> is set before the device enters IDLE2 mode.

The watchdog timer consists of a 22-stage binary counter which uses the clock φ ($2/f_{IO}$) as the input clock. The binary counter can output $2^{15}/f_{IO}$, $2^{17}/f_{IO}$, $2^{19}/f_{IO}$ and $2^{21}/f_{IO}$.

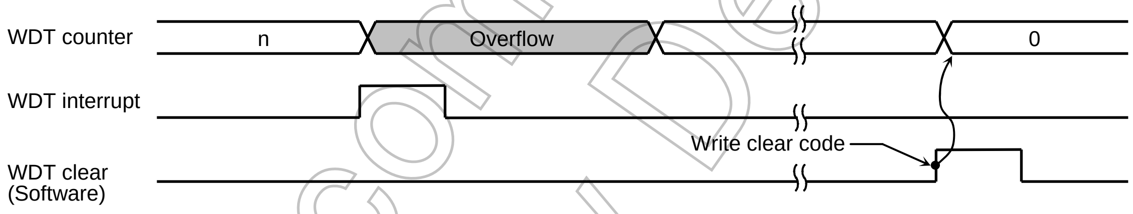


Figure 3.12.2 Normal Mode

The runaway detection result can also be connected to the reset pin internally.
In this case, the reset time will be between 22 and 29 system clocks (35.2 to 46.4 μs at f_{OSCH} = 40 MHz) as shown in Figure 3.12.3. After a reset, the f_{IO} clock is $f_{FPH}/4$, where f_{FPH} is generated by dividing the high-speed oscillator clock (f_{OSCH}) by sixteen through the clock gear function.

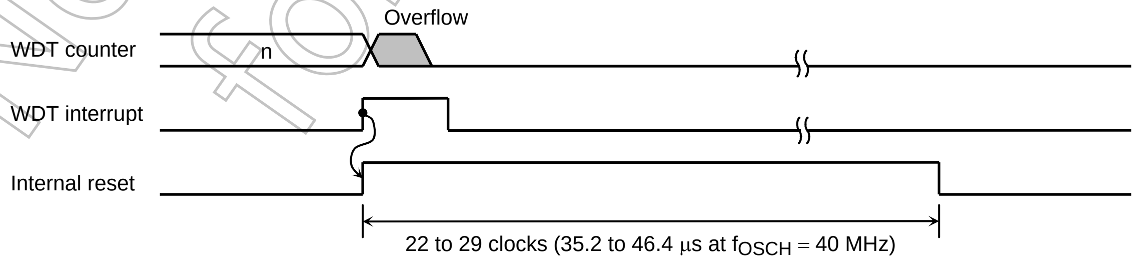


Figure 3.12.3 Reset Mode

3.12.3 Control Registers

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

(1) Watchdog timer mode register (WDMOD)

1. Setting the detection time for the watchdog timer in <WDTP1:0>

This 2-bit register is used for setting the watchdog timer interrupt time used when detecting runaway.

On a reset this register is initialized to WDMOD<WDTP1:0> = 00.

The detection time for WDT is $2^{15}/f_{IO}$ [s]. (The number of system clocks is approximately 65,536.)

2. Watchdog timer enable/disable control register <WDTE>

At reset, the WDMOD<WDTE> is initialized to 1, enabling the watchdog timer.

To disable the watchdog timer, it is necessary to 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.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).
WDMOD	←	0	–	–	X	0	–	–	0	Clear WDMOD <WDTE> to 0.
WDCR	←	1	0	1	1	0	0	0	1	Write the disable code (B1H).

- Enable control

Set WDMOD<WDTE> to 1.

- Watchdog timer clear control

To clear the binary counter and cause counting to resume, write the clear code (4EH) to the WDCR register.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).

Note1: If the disable control is used, set the disable code (B1H) to WDCR after writing the clear code (4EH) once. (Please refer to setting example.)

Note2: If the watchdog timer setting is changed, change setting after setting to disable condition once.

WDMOD
(1300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0		–	I2WDT	RESCR	–
Read/Write	R/W				R/W			
Reset State	1	0	0		0	0	0	0
Function	WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{IO}$ 01: $2^{17}/f_{IO}$ 10: $2^{19}/f_{IO}$ 11: $2^{21}/f_{IO}$			Always write “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	$2^{15}/f_{IO}$ (Approximately 3.28 ms at f_{OSCH} = 40 MHz)
01	$2^{17}/f_{IO}$ (Approximately 13.1 ms at f_{OSCH} = 40 MHz)
10	$2^{19}/f_{IO}$ (Approximately 52.4 ms at f_{OSCH} = 40 MHz)
11	$2^{21}/f_{IO}$ (Approximately 210 ms at f_{OSCH} = 40 MHz)

→ Watchdog timer enable/disable control

0	Disabled
1	Enabled

Figure 3.12.4 Watchdog Timer Mode Register

WDCR
(1302H)

Read
-modify
-write
instruction is
prohibited

	7	6	5	4	3	2	1	0
Bit symbol	–							
Read/Write	W							
Reset State	–							
Function	B1H: WDT disable code 4EH: WDT clear code							

→ WDT disable/clear control

B1H	Disable code
4EH	Clear code
Others	Don’t care

Figure 3.12.5 Watchdog Timer Control Register

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 s adjustment function (by software)
5) Alarm function (alarm output)
6) Alarm interrupt generate

3.13.2 Block Diagram

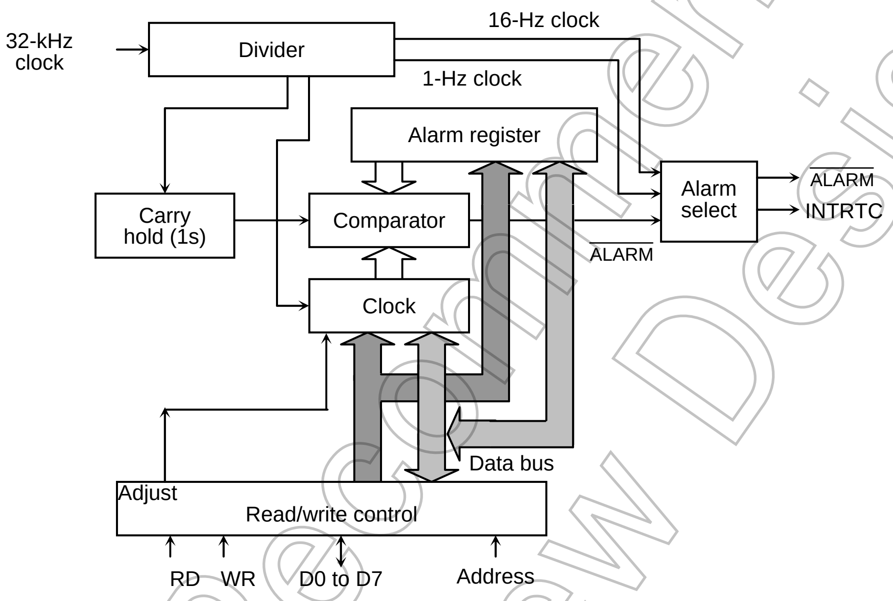


Figure 3.13.1 RTC Block Diagram

Note 1: Western calendar year column:

This product uses only the final two digits of the year. Therefore, the year following 99 is 00 years. In use, please take into account the first two digits when handling years in the western calendar.

Note 2: Leap year:

A leap year is divisible by 4, but the exception is any leap year which is divisible by 100; this is not considered a leap year. However, any year which is divisible by 400, is a leap year. This product does not take into account the above exceptions . Since this product accounts only for leap years divisible by 4, please adjust the system for any problems.

3.13.3 Control Registers

Table 3.13.1 PAGE 0 (Clock function) Registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	1320H		40 sec	20 sec	10 sec	8 sec	4 sec	2 sec	1 sec	Second column	R/W
MINR	1321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	1322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	1323H						W2	W1	W0	Day of the week column	R/W
DATER	1324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	1325H				Oct.	Aug.	Apr.	Feb.	Jan.	Month column	R/W
YEARR	1326H	Year 80	Year 40	Year 20	Year 10	Year 8	Year 4	Year 2	Year 1	Year column (Lower two columns)	R/W
PAGER	1327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	1328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write "0"				Reset register	W only

Note: When reading SECR, MINR, HOURR, DAYR, DATER,MONTHR and YEARR of PAGE0, the current state is read.

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	R/W
HOURR	1322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	1323H						W2	W1	W0	Day of the week column	R/W
DATER	1324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	1325H								24/12	24-hour clock mode	R/W
YEARR	1326H							LEAP1	LEAP0	Leap-year mode	R/W
PAGER	1327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	1328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write "0"				Reset register	W only

Note: When reading SECR, MINR, HOURR, DAYR, MONTHR, DATER,YEARR of PAGE1, the current state is read.

3.13.4 Detailed Explanation of Control Register

RTC is not initialized by system reset.

Therefore, all registers must be initialized at the beginning of the program.

(1) Second column register (for PAGE0 only)

		7	6	5	4	3	2	1	0
SECR (1320H)	Bit symbol		SE6	SE5	SE4	SE3	SE2	SE1	SE0
	Read/Write		R/W						
	Reset State		Undefined						
	Function	"0" is read.	40 sec. column	20 sec. column	10 sec. column	8 sec. column	4 sec. column	2 sec. column	1 sec. column

0	0	0	0	0	0	0	0 sec
0	0	0	0	0	0	1	1 sec
0	0	0	0	0	1	0	2 sec
0	0	0	0	0	1	1	3 sec
0	0	0	0	1	0	0	4 sec
0	0	0	0	1	0	1	5 sec
0	0	0	0	1	1	0	6 sec
0	0	0	0	1	1	1	7 sec
0	0	0	1	0	0	0	8 sec
0	0	0	1	0	0	1	9 sec
0	0	1	0	0	0	0	10 sec
			:				
0	0	1	1	0	0	1	19 sec
0	1	0	0	0	0	0	20 sec
			:				
0	1	0	1	0	0	1	29 sec
0	1	1	0	0	0	0	30 sec
			:				
0	1	1	1	0	0	1	39 sec
1	0	0	0	0	0	0	40 sec
			:				
1	0	0	1	0	0	1	49 sec
1	0	1	0	0	0	0	50 sec
			:				
1	0	1	1	0	0	1	59 sec

Note: Do not set data other than as shown above.

(2) Minute column register (for PAGE0/1)

MINR
(1321H)

	7	6	5	4	3	2	1	0
Bit symbol		MI6	MI5	MI4	MI3	MI2	MI1	MI0
Read/Write		R/W						
Reset State		Undefined						
Function	"0" is read.	40 min column	20 min column	10 min column	8 min column	4 min column	2 min column	1 min column

0	0	0	0	0	0	0	0 min
0	0	0	0	0	0	1	1 min
0	0	0	0	0	1	0	2 min
0	0	0	0	0	1	1	3 min
0	0	0	0	1	0	0	4 min
0	0	0	0	1	0	1	5 min
0	0	0	0	1	1	0	6 min
0	0	0	0	1	1	1	7 min
0	0	0	1	0	0	0	8 min
0	0	0	1	0	0	1	9 min
0	0	1	0	0	0	0	10 min

:

0	0	1	1	0	0	1	19 min
0	1	0	0	0	0	0	20 min

:

0	1	0	1	0	0	1	29 min
0	1	1	0	0	0	0	30 min

:

0	1	1	1	0	0	1	39 min
1	0	0	0	0	0	0	40 min

:

1	0	0	1	0	0	1	49 min
1	0	1	0	0	0	0	50 min

:

1	0	1	1	0	0	1	59 min

Note: Do not set data other than as shown above.

(3) Hour column register (for PAGE0/1)

1. In 24-hour clock mode (MONTHR<MO0> = "1")

		7	6	5	4	3	2	1	0
HOURR (1322H)	Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
	Read/Write			R/W					
	Reset State			Undefined					
	Function	"0" is read.		20 hours column	10 hours column	8 hours column	4 hours column	2 hours 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

Note: Do not set data other than as shown above.

2. In 12-hour clock mode (MONTHR<MO0> ="0")

		7	6	5	4	3	2	1	0
HOURR (1322H)	Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
	Read/Write			R/W					
	Reset State			Undefined					
	Function	"0" is read.		PM/AM	10 hours column	8 hours column	4 hours column	2 hours column	1 hour column

0	0	0	0	0	0	0 o'clock (AM)
0	0	0	0	0	1	1 o'clock
0	0	0	0	1	0	2 o'clock

:

0	0	1	0	0	1	9 o'clock
0	1	0	0	0	0	10 o'clock
0	1	0	0	0	1	11 o'clock
1	0	0	0	0	0	0 o'clock (PM)
1	0	0	0	0	1	1 o'clock

Note: Do not set data other than as shown above.

(4) Day of the week column register (for PAGE0/1)

DAYR
(1323H)

	7	6	5	4	3	2	1	0
Bit symbol						WE2	WE1	WE0
Read/Write						R/W		
Reset State						Undefined		
Function	"0" is read.					W2	W1	W0

0	0	0	Sunday
0	0	1	Monday
0	1	0	Tuesday
0	1	1	Wednesday
1	0	0	Thursday
1	0	1	Friday
1	1	0	Saturday

Note: Do not set data other than as shown above.

(5) Day column register (PAGE0/1)

DATER
(1324H)

	7	6	5	4	3	2	1	0
Bit symbol			DA5	DA4	DA3	DA2	DA1	DA0
Read/Write			R/W					
Reset State			Undefined					
Function	"0" is read.		Day 20	Day 10	Day 8	Day 4	Day 2	Day 1

0	0	0	0	0	1	1st day
0	0	0	0	1	0	2nd day
0	0	0	0	1	1	3rd day
0	0	0	1	0	0	4th day

:

0	0	1	0	0	1	9th day
0	1	0	0	0	0	10th day
0	1	0	0	0	1	11th day

:

0	1	1	0	0	1	19th day
1	0	0	0	0	0	20th day

:

1	0	1	0	0	1	29th day
1	1	0	0	0	0	30th day
1	1	0	0	0	1	31st day

Note1: Do not set data other than as shown above.

Note2: Do not set for non-existent days (e.g.: 30th Feb).

(6) Month column register (for PAGE0 only)

MONTHR
(1325H)

	7	6	5	4	3	2	1	0
Bit symbol				MO4	MO4	MO2	MO1	MO0
Read/Write				R/W				
Reset State				Undefined				
Function	“0” is read.			10 months	8 months	4 months	2 months	1 month

0	0	0	0	1	January
0	0	0	1	0	February
0	0	0	1	1	March
0	0	1	0	0	April
0	0	1	0	1	May
0	0	1	1	0	June
0	0	1	1	1	July
0	1	0	0	0	August
0	1	0	0	1	September
1	0	0	0	0	October
1	0	0	0	1	November
1	0	0	1	0	December

Note: Do not set data other than as shown above.

(7) Select 24-hour clock or 12-hour clock (for PAGE1 only)

MONTHR
(1325H)

	7	6	5	4	3	2	1	0
Bit symbol								MO0
Read/Write								R/W
Reset State								Undefined
Function	“0” is read.							1: 24-hour 0: 12-hour

(8) Year column register (for PAGE0 only)

		7	6	5	4	3	2	1	0
YEARR (1326H)	Bit symbol	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
	Read/Write	R/W							
	Reset State	Undefined							
	Function	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 year

0	0	0	0	0	0	0	0	00 years
0	0	0	0	0	0	0	1	01 years
0	0	0	0	0	0	1	0	02 years
0	0	0	0	0	0	1	1	03 years
0	0	0	0	0	1	0	0	04 years
0	0	0	0	0	1	0	1	05 years
				:				
1	0	0	1	1	0	0	1	99 years

Note: Do not set data other than as shown above.

(9) Leap year register (for PAGE1 only)

		7	6	5	4	3	2	1	0
YEARR (1326H)	Bit symbol							LEAP1	LEAP0
	Read/Write							R/W	
	Reset State							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 a leap year
0	1	Current year is the year following a leap year
1	0	Current year is two years after a leap year
1	1	Current year is three years after a leap year

(10) Setting PAGE register (for PAGE0/1)

PAGER
(1327H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	INTENA			ADJUST	ENATMR	ENAALM		PAGE
Read/Write	R/W			W	R/W			R/W
Reset State	0			Undefined	Undefined			Undefined
Function	INTRTC 0: Disable 1: Enable	"0" is read.		0: Don't care 1: Adjust	Clock 0: Disable 1: Enable	ALARM 0: Disable 1: Enable	"0" is read.	PAGE selection

Note: Please keep the setting order below of <ENATMR>, <ENAAML> and <INTENA>. Set different times for Clock/Alarm setting and interrupt setting.

(Example) Clock setting/Alarm setting

```
          ld      (pager), 0ch      :     Clock, Alarm enable

          ld      (pager), 8ch      :     Interrupt enable
```

PAGE	0	Select Page0
	1	Select Page1

ADJUST	0	Don't care
	1	Adjust sec. counter. When this bit is set to "1" the sec. counter becomes "0" when the value of the sec. counter is 0 – 29. When the value of the sec. counter is 30-59, the min. counter is carried and sec. counter becomes "0". Output Adjust signal during 1 cycle of f_{SYS}. After being adjusted once, Adjust is released automatically. (PAGE0 only)

(11) Setting reset register (for PAGE0/1)

RESTR
(1328H)

Read-modify write-instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	DIS1Hz	DIS16Hz	RSTTMR	RSTALM	–	–	–	–
Read/Write	W							
Reset State	Undefined							
Function	1Hz 0: Enable 1: Disable	16Hz 0: Enable 1: Disable	1:Clock reset	1: Alarm reset	Always write "0"			

RSTALM	0	Unused
	1	Reset alarm register

RSTTMR	0	Unused
	1	Reset counter

<DIS1HZ>	<DIS1HZ>	(PAGER) <ENAALM>	Source signal
1	1	1	Alarm
0	1	0	1Hz
1	0	0	16Hz
Others			Output "0"

3.13.5 Operational description

(1) Reading clock data

1. Using 1Hz interrupt

1Hz interrupt and the count up of internal data synchronize. Therefore, data can read correctly if reading data after 1Hz interrupt occurred.

2. Using two times reading

There is a possibility of incorrect clock data reading when the internal counter carries over. To ensure correct data reading, please read twice, as follows:

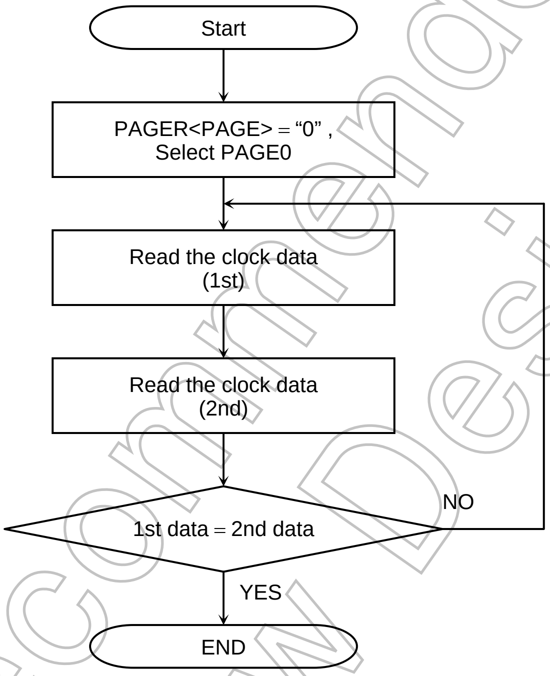


Figure 3.13.2 Flowchart of clock data read

(2) Writing clock data

When a carry over occurs during a write operation, the data cannot be written correctly. Please use the following method to ensure data is written correctly.

1. Using 1Hz interrupt

1Hz interrupt and the count up of internal data synchronize. Therefore, data can write correctly if writing data after 1Hz interrupt occurred.

2. Resetting a counter

There are 15-stage counter inside the RTC, which generate a 1Hz clock from 32,768 kHz. The data is written after reset this counter.

However, if clearing the counter, it is counted up only first writing at half of the setting time, first writing only. Therefore, if setting the clock counter correctly, after clearing the counter, set the 1Hz-interrupt to enable. And set the time after the first interrupt (occurs at 0.5s) is occurred.

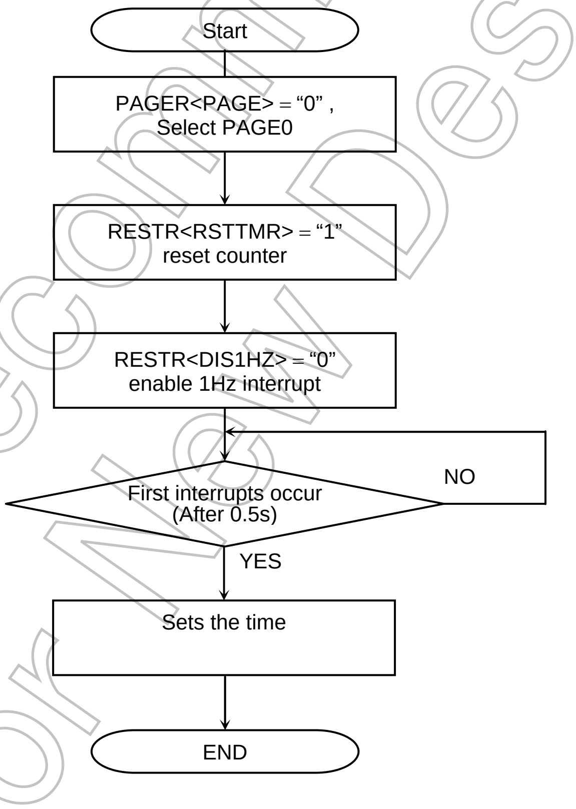


Figure 3.13.3 Flowchart of data write

2. Disabling the clock

A clock carry over is prohibited when “0” is written to PAGER<ENATMR> in order to prevent malfunction caused by the Carry hold circuit. While the clock is prohibited, the Carry hold circuit holds a one sec. carry signal from a divider. When the clock becomes enabled, the carry signal is output to the clock, the time is revised and operation continues. However, the clock is delayed when clock-disabled state continues for one second or more. Note that at this time system power is down while the clock is disabled. In this case the clock is stopped and clock is delayed.

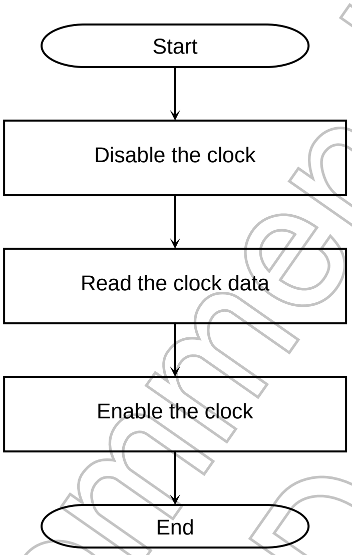


Figure 3.13.4 Flowchart of Clock disable

3.13.6 Explanation of the interrupt signal and alarm signal

The alarm function used by setting the PAGE1 register and outputting either of the following three signals from $\overline{\text{ALARM}}$ pin by writing “1” to PAGER<PAGE>. INTRTC outputs a 1-shot pulse when the falling edge is detected. RTC is not initialized by RESET. Therefore, when the clock or alarm function is used, clear interrupt request flag in INTC (interrupt controller).

(1) When the alarm register and the clock correspond, output “0”.

(2) 1Hz Output clock.

(3) 16Hz Output clock.

(1) When the alarm register and the clock correspond, output “0”

When PAGER<ENAALM>= “1”, and the value of PAGE0 clock corresponds with PAGE1 alarm register, output “0” to $\overline{\text{ALARM}}$ pin and generate INTRTC.

The methods for using the alarm are as follows:

Initialization of alarm is done by writing “1” to RESTR<RSTALM>. All alarm settings become Don’t care. In this case, the alarm always corresponds with value of the clock, and if PAGER<ENAALM> is “1”, INTRTC interrupt request is generated.

Setting alarm min., alarm hour, alarm date and alarm day is done by writing data to the relevant PAGE1 register.

When all setting contents correspond, RTC generates an INTRTC interrupt if PAGER<INTENA><ENAALM> is “1”. However, contents which have not been set up (don't care state) are always considered to correspond.

Contents which have already been set up, cannot be returned independently to the Don't care state. In this case, the alarm must be initialized and alarm register reset.

The following is an example program for outputting an alarm from $\overline{\text{ALARM}}$ -pin at noon (PM12:00) every day.

```
    LD      (PAGER), 09H        ;   Alarm disable, setting PAGE1
    LD      (RESTR), D0H        ;   Alarm initialize
    LD      (DAYR), 01H         ;   W0
    LD      (DATER),01H         ;   1 day
    LD      (HOURR), 12H        ;   Setting 12 o’clock
    LD      (MINR), 00H         ;   Setting 00 min
                                ;   Set up time 31 μs (Note)
    LD      (PAGER), 0CH        ;   Alarm enable
  ( LD      (PAGER), 8CH        ;   Interrupt enable )
```

When the CPU is operating at high frequency oscillation, it may take a maximum of one clock at 32 kHz (about 30us) for the time register setting to become valid. In the above example, it is necessary to set 31us of set up time between setting the time register and enabling the alarm register.

Note: This set up time is unnecessary when you use only internal interruption.

(2) With 1Hz output clock

RTC outputs a clock of 1Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “0”, <DIS16HZ>= “1”. RTC also generates an INTRTC interrupt on the falling edge of the clock.

(3) With 16Hz output clock

RTC outputs a clock of 16Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “1”, <DIS16HZ>= “0”. RTC also generates INTRTC an interrupt on the falling edge of the clock.

3.14 LCD Controller

This LSI incorporates two types of liquid crystal display driving circuit for controlling LCDs. One circuit supports an internal RAM LCD driver that can store display data in the LCD driver itself, and the other circuit supports a shift-register type LCD driver that must serially transfer the display data to the LCD driver for each display picture.

Software-programmable screen size supported.

This LCDC supports 2 bpp (bit per pixel: 4 grayscales), 3 bpp (8 grayscales), 4 bpp (16 grayscales) 8 bpp, (256 colors) and 12 bpp (4096 colors) for dot matrix panels. In passive matrix STN mode, it supports 8 bpp (256 colors) out of a palette of 4096 colors. And in active matrix TFT mode, it supports 8 bpp (256 colors) and 12 bpp (4096 colors).

Data bus width for 8- or 12-bit TFT panels is supported, and 8- and 4-bit wide LCD panel data bus for STN panels, plus 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 panels. Before setting start register, set the mode of operation, the start address of source data save memory and LCD size to control register.

After setting start register, the LCDC outputs a bus release request to the CPU and reads data from source memory.

The LCDC then transmits LCD size data to the external LCD driver through the special LCDC data bus (LD11 to LD0). At this time, the control signals connected to the LCD driver output the specified waveform which is synchronized with the data transmission. After display data reading from RAM is completed, the LCDC cancels the bus release request and the CPU will re-start.

In the TMP92CH21, SRAM and SDRAM burst mode can be used for the external display RAM.. 16-Kbytes of internal RAM are available for use as display RAM. As internal RAM access is very fast (32-bit bus width, 1 SYSCLK read/write), it is possible to reduce CPU load to a minimum, enabling LCDC DMA.

2) Color display mode for TFT panel

The data transmission process is as above in SR mode. LD11 to LD0; 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 control the TFT source driver.. And besides signals LCP1 and LBCD, OE can also control details of output timing for control of TFT gate driver.

3) Internal RAM LCD driver control mode (RAM mode)

Data transmission to the LCD driver is executed CPU command. After setting operation mode to control register, when CPU command is executed the LCDC outputs chip select signal to the LCD driver connected externally by control pin (LCP0 etc.). Therefore control of data transmission numbers corresponding to LCD size is controlled by CPU command.

3.14.1 LCDC features by Mode

The various features and pin operations of are as follows.

Table 3.14.1 LCDC features by Mode (example: T6C13B, T6B66A by Toshiba)

LCD driver		Shift Register Type LCD Driver Control Mode		RAM Built-in Type LCD Driver Control Mode
		TFT	STN	
Display color		256 colors, 4096 colors	Monochrome, 4-, 8- and16-level grayscale 256 colors, 4096 colors	Depends on LCD driver

The number of picture elements which can be handled		Row (Common):
		64, 128, 160, 200, 240, 320
		Column (Segment):
		64, 128, 160, 256, 320
			Monochrome, 4-, 8- and16-level grayscale
			Row (Common):
			64, 128, 160, 200, 240, 320, 480
			Column (Segment):
			64, 128, 160, 256, 320, 480, 640, 768, 960
			256 colors, 4096 colors
			Row (Common):
			64, 128, 160, 200, 240, 320
			Column (Segment):
			64, 128, 160, 256, 320
				Depends on LCD driver

		Internal SRAM: 256 to 128 × 128 max, 4096 to 128 × 64 max		
Data bus width (VRAM: RAM, SDRAM)		16 bits, 32 bits	16 bits, 32 bits	Depends on CS/WAIT controller (Same as normal memory access)
Data bus width (Destination: LCD driver)		8 bits, 12 bits	4 bits, 8 bits	
Maximum transmission rate (at f_{SYS} = 20 [MHz])		12.5 ns/byte at SDRAM/BURST 12.5 ns/bytes at internal RAM, 25 ns/byte at external SRAM		–
Pan function		Available to use		Depends on LCD driver
External pins	LCD data bus LD11 to LD0	Connect to data bus of LCD driver. • 4-bit LD3 to LD0 • 8-bit LD7 to LD0 • 12-bit LD11 to LD0 (Only use TFT panel)		Not used
	D7 to D0	Not used		Connect to data bus of LCD driver.
	Bus state R/W	Not used		Connect to $\overline{\text{WR}}$ pin of LCD driver.
	Address bus A0	Not used		Connect to D/I pin of LCD driver for distinction of data or instruction.
	LCP0	Horizontal shift clock for source driver of TFT panel	Shift clock 0 for column LCD driver Connect to CP pin of column LCD driver. LD bus data is latched at falling edge of this signal.	Chip enable signal for column LCD driver Connect to $\overline{\text{CE}}$ pin of 1st column LCD driver.
	LCP1	Vertical shift clock for gate driver of TFT panel	Shift clock 1 for column LCD driver Connect to CP pin of column LCD driver. LD bus data is latched at falling edge of this signal.	Not used
	LLP	Data load signal for source driver of TFT panel	Latch pulse output for column and row LCD driver Connect to LP pin of column and row LCD driver. Display data is renewed to output buffer at rising edge of this signal.	Chip enable signal for column LCD driver Connect to $\overline{\text{CE}}$ pin of 2nd column LCD driver.
	LGOE2 to LGOE0	Output enable signal for gate driver of TFT panel	Not used	Not used
	LFR	Alternating signal for LCD display control. Connect to FR pin of LCD driver.	Alternating signal for LCD display control. Connect to FR pin of LCD driver.	Chip enable signal for column LCD driver Connect to $\overline{\text{CE}}$ pin of 3rd column LCD driver.
	LBCD		Refresh rate signal	Chip enable signal for row LCD driver Connect to $\overline{\text{LE}}$ pin of row LCD driver.
	LDIV	Connect to LDIV pin for source driver of TFT panel. This signal shows output data inversion.	Not use	

3.14.2 SFRs

LCDMODE0 Register

		7	6	5	4	3	2	1	0
LCDMODE0 (0280H)	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
	Reset State	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 × f_{SYS} 10: 4 × f_{SYS} 11: 8 × f_{SYS}		Mode setting 0000: Built-in RAM type 0001: SR 1 bpp (monochrome) 0010: SR 2 bpp (4 grayscales) 0011: SR 3 bpp (8 grayscales) 0100: SR 4 bpp (16 grayscales) 0101: STN 8 bpp (256 colors) 0110: STN 12 bpp (4 K colors) 0111: Reserved 1000: TFT 8 bpp (256 colors) 1001: TFT 12 bpp (4 K colors) Others: Reserved			

Note: Only “burst 1clk access” SDRAM access is supported

LCDMODE1 Register

		7	6	5	4	3	2	1	0
LCDMODE1 (0281H)	Bit symbol			LLPMODE	LDINV	AUTOINV	INTMODE	LDO1	LDO0
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0
	Function			LLP mode 0: Mode 1 1: Mode 2	LD bus inversion (Note 1) 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: 4-bit width A type 01: 4-bit width B type 10: 8-bit width A type 11: 8-bit width B type	

When using TFT mode, LD bus width is fixed below setting. Set “10” to <LDO1:0> for 256-color. For 4096-color, setting <LDO1:0> is not needed, but bus-width becomes 12-bit automatically.

Note: When setting <LDINV> = 1, auto LD bus inversion function does not work.
<LDINV> = 0 must be set if you want to use auto LD bus inversion function.

LCD f_{FP} Register

		7	6	5	4	3	2	1	0
LCDFFP (0282H)	Bit symbol	FP7	FP6	FP5	FP4	FP3	FP2	FP1	FP0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Setting bit7 to bit0 for f_{FP}							

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							
	Reset State	0	0	0	0	0	0	0	0
	Function	Setting DVM bit7 to bit0							

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
Reset State	0	0	0	0	0	0	0	0
Function	Common setting 0000: Reserved 0001: 64 0010: 120 0011: 128 0100: 160 0101: 200 0110: 240 0111: 320 1000: 480 Others: Reserved				Segment setting 0000: Reserved 0001: 64 0010: 128 0011: 160 0100: 256 0101: 320 0110: 480 0111: 640 1000: 768 1001: 960 Others: Reserved			

Note 1: Maximum size in color mode (STN,TFT) is 320 × 320.

When internal SRAM is set as display RAM, the maximum size is as below.

1 bpp (Monochrome):	640 × 200
2 bpp (4 grayscales):	320 × 200
4 bpp (16 grayscales):	256 × 128
8 bpp (256 colors):	128 × 128
12 bpp (4096 colors):	128 × 64

Note 2: This LSI does not support 240-segment size, but if a cascade type segment driver is selected, it can used by setting for 256-segment size. In this case, a 256-segment display area must be prepared.

LCD Control-0 Register

LCDCTL0
(0285H)

	7	6	5	4	3	2	1	0
Bit symbol		ALL0	FRMON	–	FP9	MMULCD	FP8	START
Read/Write		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset State		0	0	0	0	0	0	0
Function		Column data setting 0: Normal 1: All display data "0"	Frame divide setting 0: Disable 1: Enable	Always write "0".	f_{FP} setting bit9	Built-in RAM LCD driver 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
Reset State	1	0	1				0	0
Function	LCP0 phase 0: Rise 1: Fall	LCP1 phase 0: Rise 1: Fall	LBCD phase 0: Low enable 1: High 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							
Reset State	0	0	0	0	0	0	0	0
Function	LCDC source clock counter bit7 to bit0							

LCD Clock Counter Register 0

LCDCCR0
(0288H)

	7	6	5	4	3	2	1	0
Bit symbol						PCPV2	PCPV1	PCPV0
Read/Write						R/W	R/W	R/W
Reset State						0	0	0
Function						Pre LCP1 CLK: LCP1 pulse number Dummy clock number until valid clock of gate driver LCP1		

LCD Clock Counter Register 1

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
Reset State				0	0	0	0	0
Function				LLP_Set-up time: f_{SYS} pulse × 8 Set up time for TFT source driver LLP signal (Offset of f_{SYS} 14~16 pulse)				

LCD Clock Counter Register 2

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
Reset State	0	0	0	0	0	0	0	0
Function	TFT source driver, LLP_Enable signal: f_{SYS} × 8 High width time for LLP signal							

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
	Reset State	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
	Reset State	0	1	1	0	0	1	0	0
	Function	256-color STN mode RED3 level setting				256-color STN mode RED2 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
	Reset State	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
	Reset State	1	1	1	0	1	1	0	0
	Function	256-color STN mode RED7 level setting				256-color STN mode RED6 level setting			

Note: The above palette settings cannot be changed in TFT mode.

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
	Reset State	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	2G3	2G2	2G1	2G0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	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
	Reset State	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
	Reset State	1	1	1	0	1	1	0	0
	Function	256-color STN mode GREEN7 level setting				256-color STN mode GREEN6 level 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
	Reset State	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
	Reset State	1	1	0	0	1	0	0	0
	Function	256-color STN mode BLUE3 level setting				256-color STN mode BLUE2 level setting			

Note: The above palette settings cannot be changed in TFT mode.

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
Reset State	0	0	0	0	0	0	0	0
Function	OE0 control of TFT panel gate driver							

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
Reset State	0	0	0	0	0	0	0	0
Function	OE0 control of TFT panel gate driver							

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
Reset State	0	0	0	0	0	0	0	0
Function	OE1 control of TFT panel gate driver							

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
Reset State	0	0	0	0	0	0	0	0
Function	OE1 control of TFT panel gate driver							

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
Reset State	0	0	0	0	0	0	0	0
Function	OE2 control of TFT panel gate driver							

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
Reset State	0	0	0	0	0	0	0	0
Function	OE2 control of TFT panel gate driver							

	Start Address Register			Row Number Setting Register		
	H (Bit23 to 16)	M (Bit15 to 8)	L (Bit7 to 1)	H (Bit8)	L (Bit7 to 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 registers can read-modify-write.

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	Depends on external LCD driver specification.							
Reset State	Depends on external LCD driver specification.							
Function	Depends on external LCD driver specification.							

Address	Function	Chip Enable Pin
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 and STN color)

3.14.3.1 Description of Operation

Set the mode of operation, start address of source data save memory, grayscale level and LCD size to control registers before setting start register.

After setting start register, the LCDC outputs a bus release request to the CPU and reads data from source memory. After data reading from source data is completed, the LCDC cancels the bus release request and the CPU will restart. The LCDC then transmits LCD size data to the external LCD driver through the LD bus (special data bus only for LCD driver). At this time, the control signals (LCP0 etc.) connected to the LCD driver output the specified waveform which is synchronized with the data transmission.

The LCD controller generates control signals (LFR, LBCD, LLP etc.) from base clock LCDSCC. LCDSCC is the base clock for the LCD controller, which is generated by system clock f_{SYS}.

This LSI has a special clock generator for the LCDC. Details of LCD frame refresh rate can be set using this special generator. This generator is made from an 8-bit counter and 1/16 speed clock from the system clock.

Note 1: During data read from source memory (during DMA operation), the CPU is stopped by the internal BUSREQ signal. When using SR mode LCDC, programmers must monitor CPU performance.

Note 2: This LSI has a 16-Kbyte SRAM, this internal RAM is available for use as display RAM. Internal RAM access is very fast (32-bit bus width, 1 SYSCLK read/write), it is possible to reduce CPU load to a minimum, enabling LCDC DMA.

This LCDC supports monochrome, 2 bpp (4 grayscales), 3 bpp (8 grayscales), 4 bpp (16 grayscales), 8 bpp (256 colors) and 12 bpp (4096 colors). Display RAM is supported by external SDRAM, SRAM and internal RAM (16 Kbytes).

It is automatically set to suitable condition data correction against interference between pixels in panels. Special adjustment is not required.

In passive matrix STN mode, 8 bpp (256 colors) is supported out of a palette of 4096 colors. Support is also given for 4096 colors out of a pallet of 4096 colors.

Data output width is selectable between 4 bits or 8 bits, and data output sequence selectable between 2 modes.

SR type LCD control setting is described below.

3.14.3.2 Memory Space (Common spec. SR mode and TFT mode)

The LCDC can display an LCD panel image which is divided horizontally into 3 parts; upper, middle and lower. Each area is called A area, B area and C area with the characteristics shown below.

The 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 in start address.

A and B areas can be displayed by program and set to enable or not in Start Address register and Row Number register. When the Row Number registers of A and B areas are set to 0, C area takes over all panel space.

When the size of A or B area is greater than the LCD panel, the area of the panel is all C area because the displaying priority is A > B > C. If the A area is set to enable while the panel area is defined as all C area (A and B areas are disabled), C area is shifted below the LCD panel and A area is inserted from the top of the LCD panel. Similarly if the B area is 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.

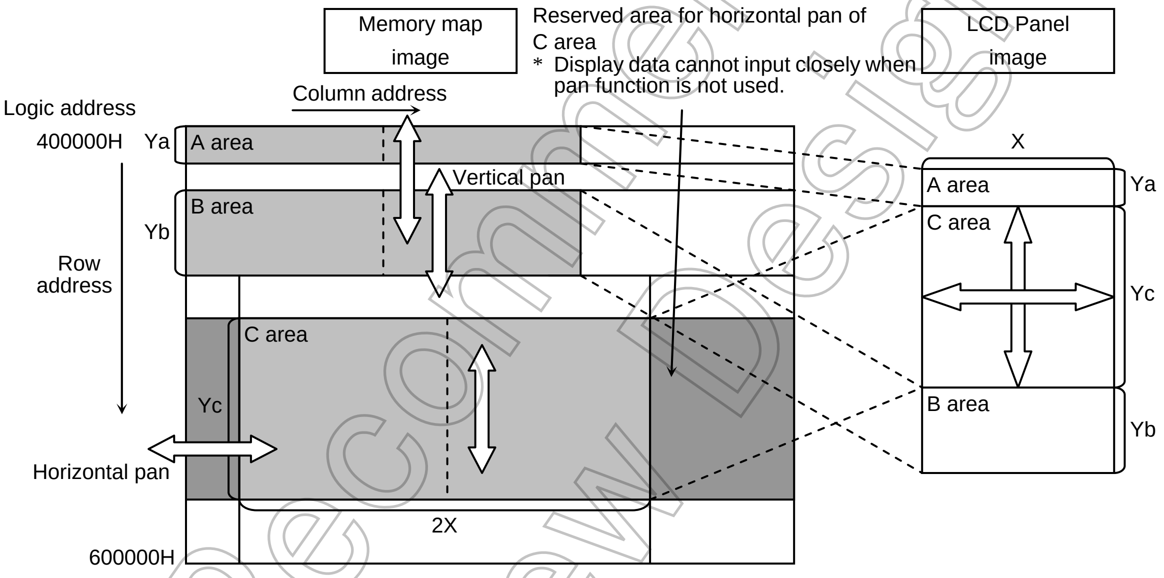


Figure 3.14.1 Memory Mapping from Physical Memory to LCD Panel

3.14.3.3 Display Memory Mapping and Panning Function (Common spec. SR mode and TFT mode)

The LCDC can only change the panel window if you change each start address of A, B and C areas. The display area can be panned vertically and horizontally by changing the row address and column address.

This LCDC can select many display modes: 1 bpp (monochrome), 2 bpp (4 grayscales), 3 bpp (8 grayscales), 4 bpp (16 grayscales), 8 bpp (256 colors) and 12 bpp (4096 colors) and 1-line (row). Data volume is different for each display mode. When using the panning function, care must be exercised in calculating the address for each display mode. For details, refer to Figure 3.14.2 to Figure 3.14.5, "Relation of memory map image and output data". This LCDC can also support external SDRAM, SRAM and internal SRAM for display RAM.

When using SDRAM for display RAM, data from one line to the next line cannot be input continuously in display RAM, even if the panning function is not used. One row address of display SDRAM corresponds to the first line of the display panel. Second line display data cannot now be set within the first row address of the display RAM even if the necessary data for the size you want to display does not fill the capacity of first row address of the display SDRAM. Adding one line to the display panel is equal to adding one address to the row address of the display SDRAM. In other words, when using SDRAM for display RAM, address calculation for panning is simple.

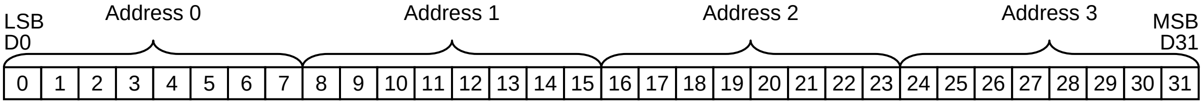
When using SRAM for display RAM, data from one line to the next line must be input continuously to the display RAM. However, address calculation for panning is complex and horizontal panning function is not supported.

3.14.3.4 Data Transmission

This LSI has an LD bus (LD7 to LD0): a special data bus for LCD driver. Bus width of 4-or 8-bits can be supported, and 2 formats selected for each bus width . The 2 formats of 8-bit bus width can support only STN color mode (256, 4096 colors). The 12-bit bus width supports only TFT mode.

LD bus data invert function is also supported. By setting LCDMODE2<LDINV> = 1, all LD bus data is inverted. There is <AUTOINV> bit in this LCDMODE2 register, but this automatic data invert function is only for TFT mode.

- Monochrome: 1 bpp (bit per pixel)
  Display memory image

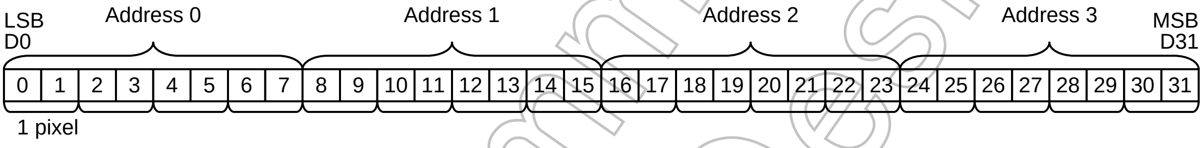


LD bus output sequence

4-bit width A type		4-bit width B type		8-bit width A type	
LD0	0 → 4 → 8 → 12 ...	LD0	4 → 0 → 12 → 8 ...	LD0	0 → 8 ...
LD1	1 → 5 → 9 → 13 ...	LD1	5 → 1 → 13 → 9 ...	LD1	1 → 9 ...
LD2	2 → 6 → 10 → 14 ...	LD2	6 → 2 → 14 → 10 ...	LD2	2 → 10 ...
LD3	3 → 7 → 11 → 15 ...	LD3	7 → 3 → 15 → 11 ...	LD3	3 → 11 ...
LD4	Not use	LD4	Not used	LD4	4 → 12 ...
LD5	Not use	LD5	Not used	LD5	5 → 13 ...
LD6	Not use	LD6	Not used	LD6	6 → 14 ...
LD7	Not use	LD7	Not used	LD7	7 → 15 ...

Note: This mode is not supported by 8 bit width B type.

- 4 grayscales (2 bpp)
  Display memory image



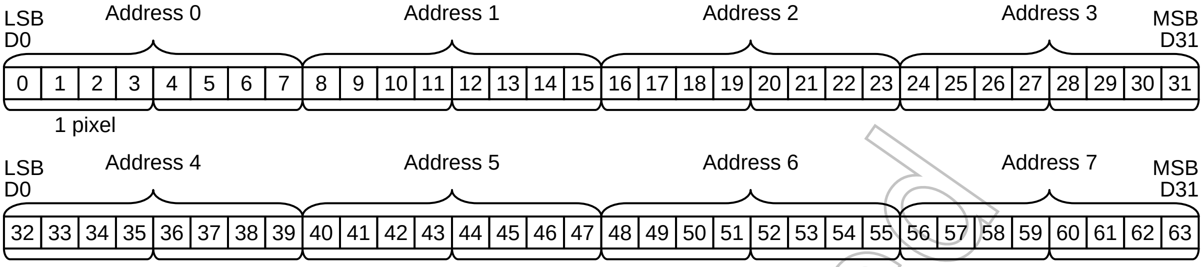
LD bus output sequence

4-bit width A type		4-bit width B type		8-bit width A type	
LD0	1-0 → 9-8 → 17-16 ...	LD0	9-8 → 1-0 → 25-24 ...	LD0	1-0 → 17-16 ...
LD1	3-2 → 11-10 → 19-18 ...	LD1	11-10 → 3-2 → 27-26 ...	LD1	3-2 → 19-18 ...
LD2	5-4 → 13-12 → 21-20 ...	LD2	13-12 → 5-4 → 29-28 ...	LD2	5-4 → 21-20 ...
LD3	7-6 → 15-14 → 23-22 ...	LD3	15-14 → 7-6 → 31-30 ...	LD3	7-6 → 23-22 ...
LD4	Not use	LD4	Not used	LD4	9-8 → 25-24 ...
LD5	Not use	LD5	Not used	LD5	11-10 → 27-26 ...
LD6	Not use	LD6	Not used	LD6	13-12 → 29-28 ...
LD7	Not use	LD7	Not used	LD7	15-14 → 31-30 ...

Note: This mode is not supported by 8 bit width B type.

Figure 3.14.2 Relation of Memory Map Image and Output Data (1)

- 8/16 grayscales (4 bpp: 8 grayscales case, valid data is 3 bits but data space needs 4 bits)
  Display memory image



LD bus output sequence

4-bit width A type

LD0	3-0	→	19-16	...
LD1	7-4	→	23-20	...
LD2	11-8	→	27-24	...
LD3	15-12	→	31-28	...
LD4	Not use			
LD5	Not use			
LD6	Not use			
LD7	Not use			

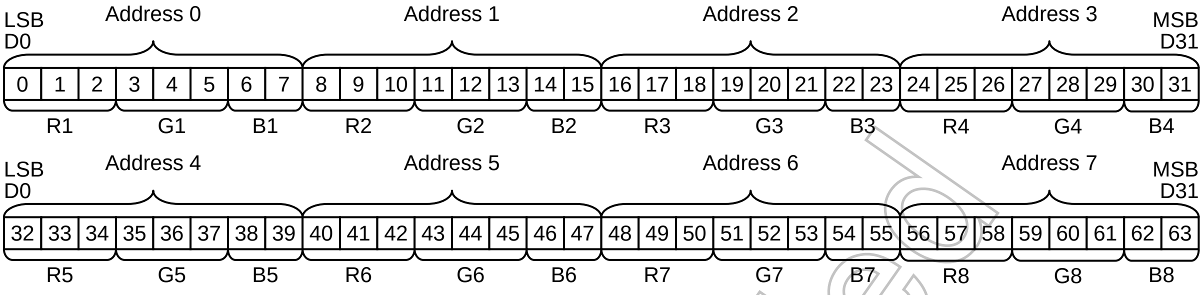
8-bit width A type

LD0	3-0	→	35-32	...
LD1	7-4	→	39-36	...
LD2	11-8	→	43-40	...
LD3	15-12	→	47-44	...
LD4	19-16	→	51-48	...
LD5	23-20	→	55-52	...
LD6	27-24	→	59-56	...
LD7	31-28	→	63-60	...

* 8 grayscales data format is the same as 16 grayscales, 1 pixel needs 4-bit space. LSB bit is invalid data. This mode is not supported by 4-bit width B type and 8-bit width B type.

Figure 3.14.3 Relation of Memory Map Image and Output Data (2)

- 256 colors (8 bpp; R: 3 bits, G: 3 bits, B: 2 bits)
  Display memory image



LD bus output sequence
4-bit width A type

LD0	2-0 (R1)	→	13-11 (G2)	...
LD1	5-3 (G1)	→	15-14 (B2)	...
LD2	7-6 (B1)	→	18-16 (R3)	...
LD3	10-8 (R2)	→	21-19 (G3)	...
LD4	Not used			
LD5	Not used			
LD6	Not used			
LD7	Not used			

LD bus output sequence
8-bit width A type

LD0	2-0 (R1)	→	23-22 (B3)	...
LD1	5-3 (G1)	→	26-24 (R4)	...
LD2	7-6 (B1)	→	29-27 (G4)	...
LD3	10-8 (R2)	→	31-30 (B4)	...
LD4	13-11 (G2)	→	34-32 (R5)	...
LD5	15-14 (B2)	→	37-35 (G5)	...
LD6	18-16 (R3)	→	39-38 (B5)	...
LD7	21-19 (G3)	→	42-40 (R6)	...

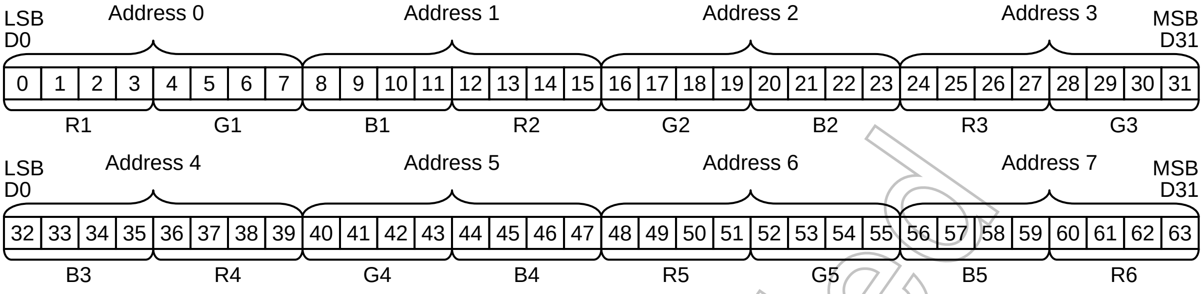
8-bit width B type

LD0	2-0 (R1)	→	7-6 (G1)	→	45-43 (G6)	→	47-46 (B6)	...
LD1	5-3 (B1)	→	10-8 (R2)	→	50-48 (R7)	→	53-51 (G7)	...
LD2	13-11 (G2)	→	15-14 (B2)	→	55-54 (B7)	→	58-56 (R8)	...
LD3	18-16 (R3)	→	21-19 (G3)	→	61-59 (G8)	→	63-62 (B8)	...
LD4	23-22 (B3)	→	26-24 (R4)	→	66-64 (R9)	→	69-67 (G9)	...
LD5	29-27 (G4)	→	31-30 (B4)	→	71-70 (B9)	→	74-72 (R10)	...
LD6	34-32 (R5)	→	37-35 (G5)	→	77-75 (G10)	→	79-78 (B10)	...
LD7	39-38 (B5)	→	42-40 (R6)	→	82-80 (R11)	→	85-83 (G11)	...

* This mode is not supported by 4-bit width B type.

Figure 3.14.4 Relation of Memory Map Image and Output Data (3)

- 4096 colors (12 bpp: R: 4 bits, G: 4 bits, B: 4 bits)
  Display memory image



LD bus output sequence

4-bit width A type

LD0	3-0 (R1)	→	19-16 (G2)	...
LD1	7-4 (G1)	→	23-20 (B2)	...
LD2	11-8 (B1)	→	27-24 (R3)	...
LD3	15-12 (R2)	→	31-28 (G3)	...
LD4	Not use			
LD5	Not use			
LD6	Not use			
LD7	Not use			

8-bit width A type

LD0	3-0 (R1)	→	35-32 (B3)	...
LD1	7-4 (G1)	→	39-36 (R4)	...
LD2	11-8 (B1)	→	43-40 (G4)	...
LD3	15-12 (R2)	→	47-44 (B4)	...
LD4	19-16 (G2)	→	51-48 (R5)	...
LD5	23-20 (B2)	→	55-52 (G5)	...
LD6	27-24 (R3)	→	59-56 (B5)	...
LD7	31-28 (G3)	→	63-60 (R6)	...

* 8 grayscales data format is the same as 16 grayscales, 1 pixel needs 4-bit space. LSB bit is invalid data. This mode is not supported by 4-bit width B type and 8-bit width B type.

Figure 3.14.5 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 (LCDSCC<SCC7:0>) and FP [9:0] (LCDCTL0<FP9, 8>, LCDFFP<FP7:0>). The LBCD terminal outputs one pulse every cycle and the LFR normally outputs an inverted signal every cycle. But when the DIVIDE FRAME function is used, the LFR signal changes to a special signal for high quality display.

(1) Basic clock setting

This LSI has a special clock generator for basic source clock used in the LCD controller. This generator can set details of the refresh rate for the LCDC.

This generator is made by dividing the system clock by 16 and an 8-bit counter.

The following shows the method 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: <SCC7:0> setting value of LSCC register

$$f_{BCD}\ [\text{Hz}] = f_{SYS}\ [\text{Hz}] / ((\text{SCC}+1) \times 16 \times \text{FP})$$

Example:

f_{SYS} [Hz] = 20MHz, 480COM (FP = 480), target refresh rate = 140Hz
140 [Hz] = 20000000 [Hz]/((SCC+1) × 16 × 480)
(SCC+1) = 20000000/(140 ×16 × 480) = 18.60

Value of setting to register is only integer, SCC = 17. The floating value is disregarded.

In this case, the 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							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCDC Source Clock Counter bit7 to bit0							

* Data should be written from 1-hex to FFFF-hex in the above register. It cancannot operate if set to “0”.

* If the refresh rate is set too fast, it may not be in time with the display data. t_{LP} time is determined by SCC.

$$t_{LP}\ [\text{s}] = (1/f_{SYS}\ [\text{Hz}]) \times 16 \times (\text{SCC} + 1)$$

t_{LP} is shown in 1-line (ROW) display time. 1-line data transmission must be completed during t_{LP} cycle time. AboutRefer to “Data transmission and bus occupation” for details of data transmission time.

(2) Refresh rate adjust function (Correct function)

In this function, the LBCD frequency: refresh rate is generated by setting LCDSCC<SCC7:0> and FP [9:0] register. The FFP value is normally set at the same value as the ROW number, but this value can be used for correction of BCD frequency: refresh rate.

This function always uses a value greater than the ROW number, set to slower frequency. The LCDC cannot operate correctly if a value smaller than the ROW number is set.

The following is an example of settings:.

Example:

f_{SYS} [Hz] = 20 MHz, 480COM (FP = 480), Target refresh rate = 140 Hz

140 [Hz] = 20000000 [Hz]/((SCC+1) × 16 × 480)

(SCC+1) = 20000000/(140 × 16 × 480) = 18.60

Value of setting to register is only integer, SCC = 17. The floating value is disregarded.

In this case, refresh rate comes to 144.6 [Hz]

f_{BCD} [Hz] = f_{SYS} [Hz]/((SCC+1) × 16 × FP)

FP value is adjusted to set SCC=17 in above equation again.

140 [Hz] = 20000000/(18 × 16 × FP)

FP = 496.03

Value of setting to register is only integer, FP = 496.

In this case, refresh rate comes to 140.0 [Hz]

LCD f_{FP} 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							
Reset State	0	0	0	0	0	0	0	0
Function	Setting bit7 to bit0 for f_{FP}							

Reference) We recommend refresh rate values in the region of:…
Monochrome: 70 [Hz]
4, 8, 16 grayscales and color: 140 to 200 [Hz]

(3) Divide frame adjust function

The DIVIDE FRAME function allows for adjustments to reduce uneven display in large LCD panels.

When this function is enabled by setting <FRMON> = 1, the LFR signal alternates between high and low level with each LLP cycle for the LCDDVM register values given below.

When this function is disabled by setting <FRMON> = 0, the LFR signal alternates between high and low level with each LBCD cycle. This function is not affected by the LBCD timing.

Note: Availability of this function depends on the actual LCD driver or LCD panel used. We recommend checking that register's value when used in the proposed environment.

Divide Frame 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							
	Reset State	0	0	0	0	0	0	0	0
	Function	Setting DVM bit7 to bit0							

(Reference) In general, prime numbers (3, 5, 7, 11, 13 ...) are best for the value of the LCDDVM register.

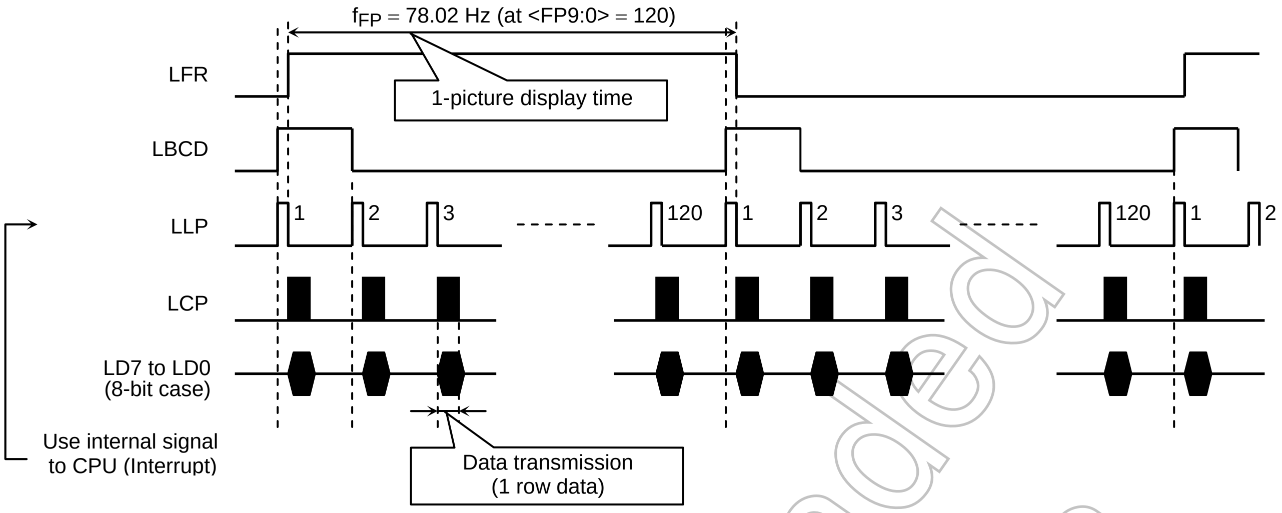

fFP = 78.02 Hz (at <FP9:0> = 120)
LFR
1-picture display time
LBCD
LLP
1
2
3
120
LCP
LD7 to LD0 (8-bit case)
Use internal signal to CPU (Interrupt)
Data transmission (1 row data)

Figure 3.14.6 Whole Timing Diagram of SR Mode

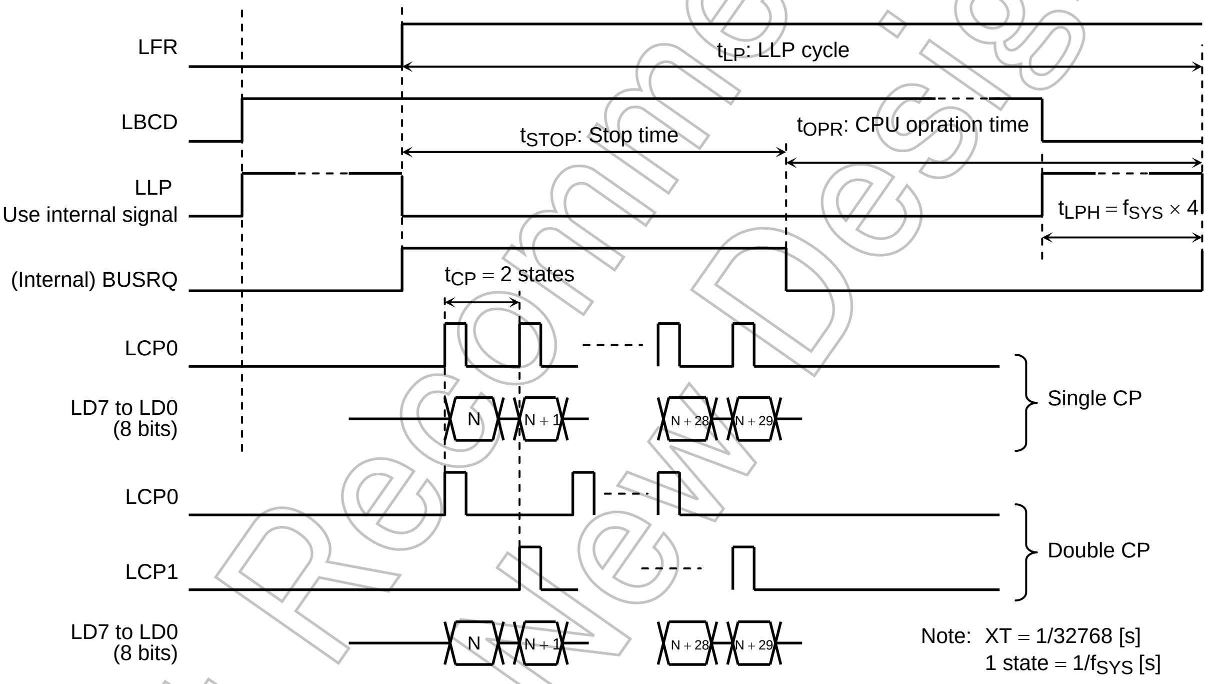


Figure 3.14.7 Detailed Timing Diagram of SR Mode

Condition: FP [9:0] setting = 240 (COM) + 63, LCDDVM<FMN7:0> = 0BH

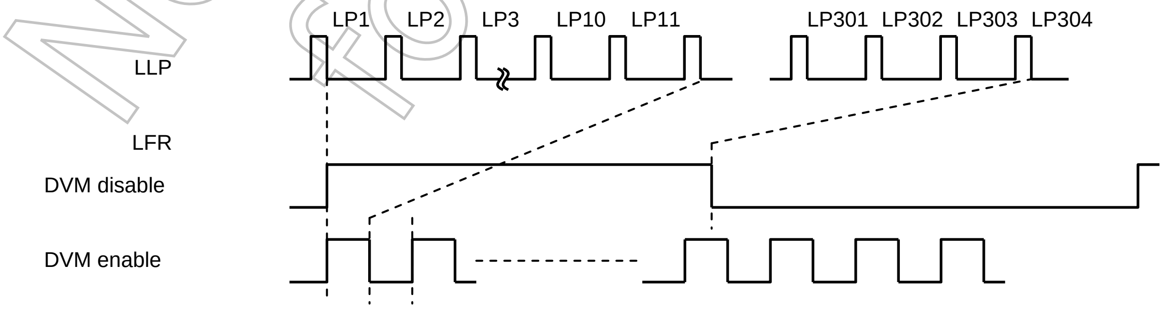


Figure 3.14.8 Waveform of LLP, LFR

3.14.3.6 LCD Data Transmission Speed and Data Bus Occupation Rate

After setting start register, the LCDC outputs a bus release request to the CPU and reads data from source memory. The LCDC then transmits LCD size data to the external LCD driver through the special LCDC data bus (LD11 to LD0). At this time, the control signals connected to the LCD driver output the specified waveform which is synchronized with the data transmission. After data reading from RAM for display is completed, the LCDC cancels the bus release request and the CPU will restart.

During data read from source memory (during DMA operation), the CPU is stopped by the internal BUSREQ signal. When using SR mode LCDC, programmers must monitor CPU performance. The occupation rate of the data bus depends on data size, transmission speed (CPU clock speed) and display RAM type used.

Display RAM	Bus Width	Valid Data Reading Time (f_{SYS} Clock/Byte)	Valid Data Reading Time t_{LRD} (ns/Byte) at f_{SYS} = 20 MHz
External SRAM	16 bits	2/2	50
	32 bits	2/4	25
Internal RAM	32 bits	1/4	12.5
External SDRAM	16 bits	*1/2	*25
	32 bits	*1/4	*12.5

Note: When using SDRAM for display RAM, overhead time (+ 8 clocks) is required for every 1 row data reading.

t_{STOP} refers to the CPU stoppage time during transmission of 1 row data. t_{STOP} is calculated by the equation below for each display mode.

t_{STOP} = (SegNum × K/8) × t_{LRD} ; Except SDRAM use

SegNum : Number of segment
K : bit number per pixel (bpp)

Monochrome	K = 1
4 Grayscales	K = 2
8/16 Grayscales	K = 4
256 colors	K = 8
4096 colors	K = 12

When SDRAM is used, more overhead time is required.

t_{STOP} = (SegNum × K/8) × t_{LRD} + ((1/f_{SYS}) × 8) ; SDRAM use

Data bus occupation rate equals the percentage of t_{STOP} time in t_{LP} time.

Data bus occupation rate = t_{STOP}/t_{LP}

Note: For t_{LP} time, refer to “refresh rate setting”.

3.14.3.7 Timing Diagram of LD Bus

The TMP92CH21 can select to display RAM for external SRAM: Available to set WAIT, internal SRAM and external SDRAM: 16, 32, 64, 128, 256 and 512 Mbits.

As a 480-byte FIFO buffer is built into this LCDC, the LD bus speed can be controlled.

The speed can be selected from 3 kinds of cycle: (f_{SYS}/2, f_{SYS}/4, and fSYS/8)

LD bus data: LD11 to LD0 is out at rising edge of LCP0, LCD driver receives at falling edge of LCP0.

Note: If the LCP cycle is too slow it may not transfer correctly.

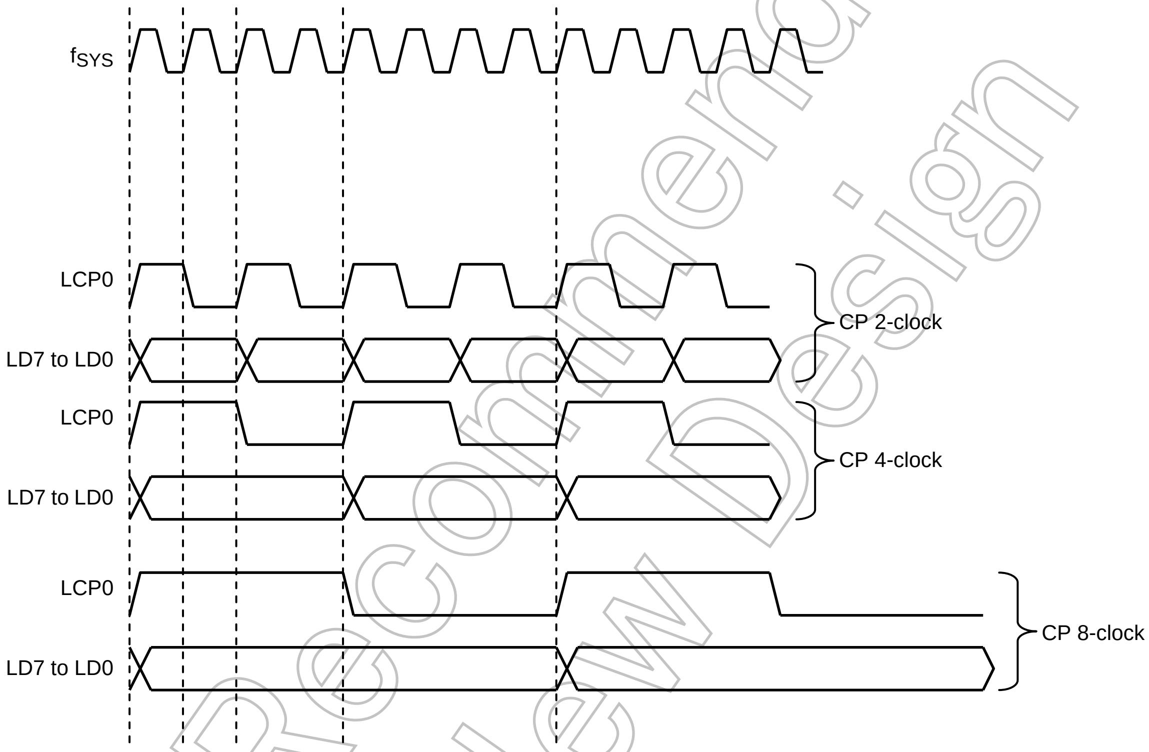


Figure 3.14.9 Selection of LCP Cycle

If LCP cycle is not set at a suitable speed with respect to the refresh rate, LD bus data will not transfer correctly. t_{LP} time is shown in the equation below.

$$t_{LP}\ [s] = (1/f_{SYS}\ [Hz]) \times 16 \times (SCC+1)$$

Data transmission must finish in t_{LP} time. Set SCC clock and LCP0 speed to be less than t_{LP} time. For setting of SCC, refer to “basic clock setting” of “refresh rate setting”.

The kind of display memory and display mode determine LCP speed. In other words, when the setting is too fast , there will be not enough transmission data in FIFO, and LCD data will not transfer correctly.

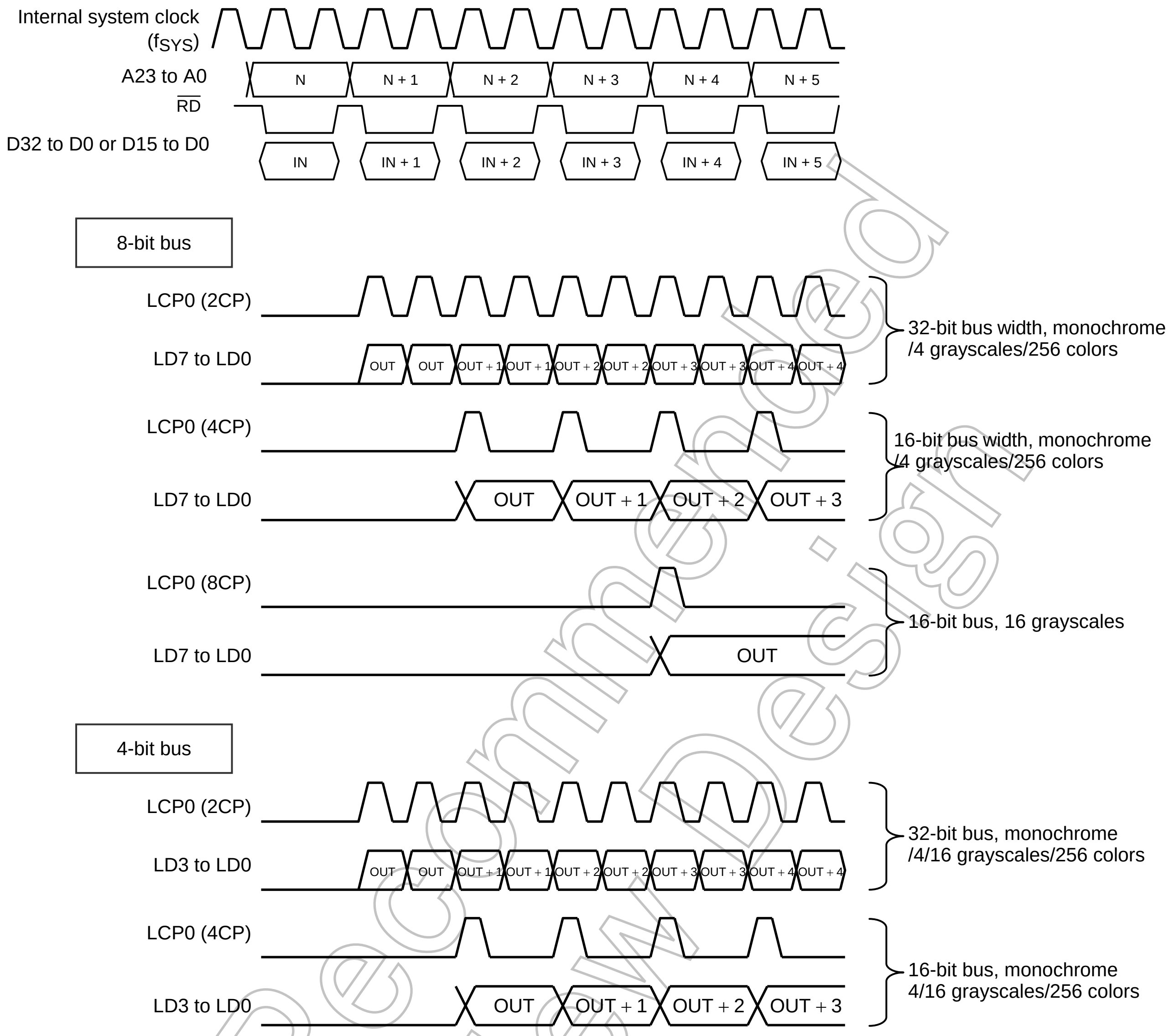
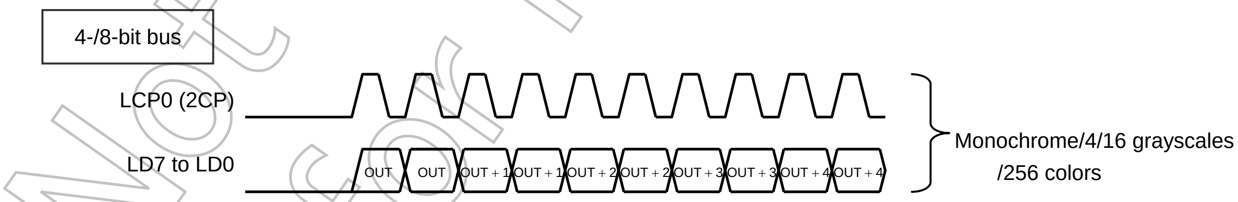


Figure 3.14.10 Fastest Timing Diagram for External SRAM, 0 waits



∗ When using internal SRAM, always select 32-bit bus width and 0 waits, 1 clocks access.

Figure 3.14.11 Timing Diagram for Internal SRAM

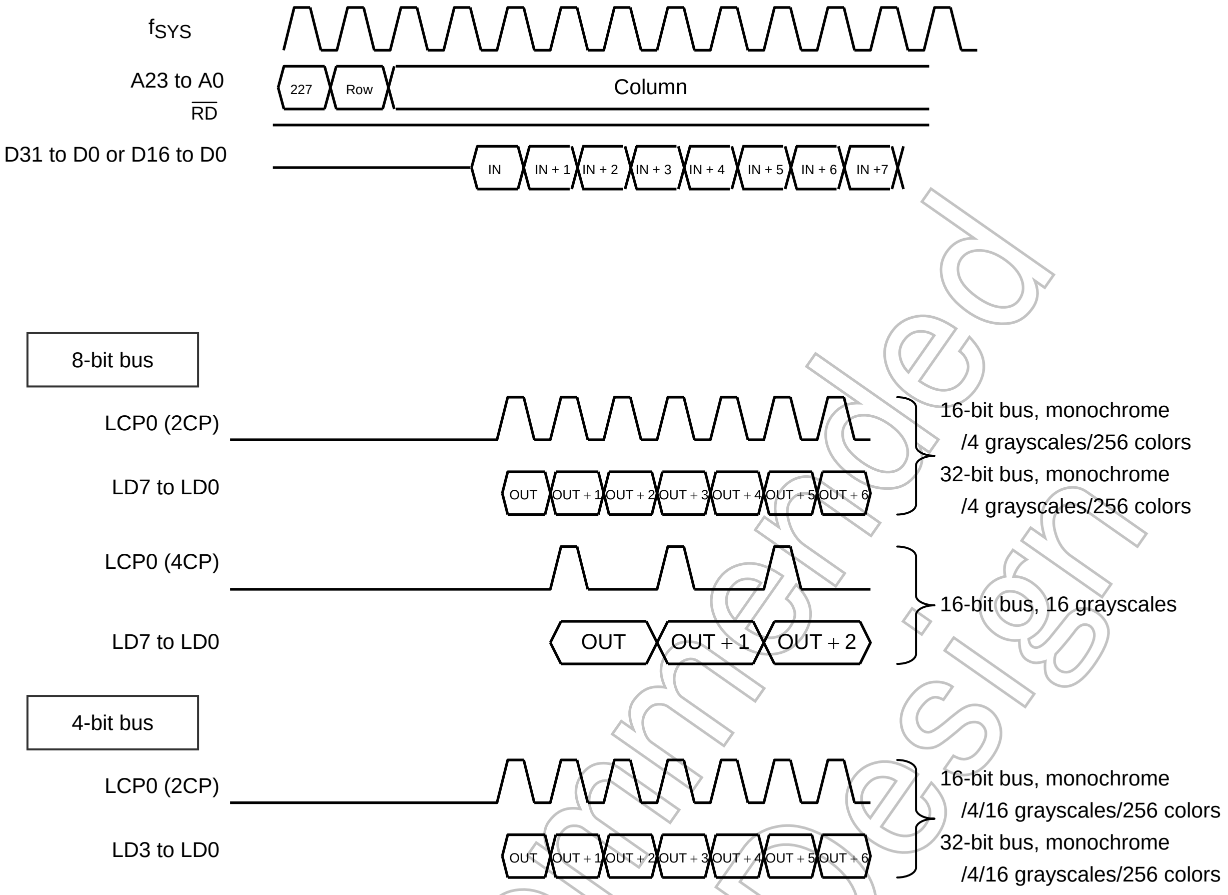


Figure 3.14.12 Timing Diagram of SDRAM Burst Run

3.14.3.8 Setting of Color Palette

This LSI can support monochrome, 4-, 8-, 16-level grayscales and color STN panels, and color TFT panels. The following shows the settings for each mode.

- Monochrome

  No need for special setting, simply select monochrome mode by LCDMODE1<MODE3:0> register.

- 4, 8, 16 grayscales

  No need for special setting, as with monochrome mode, simply select monochrome mode by LCDMODE1<MODE3:0> register.

  For 8- and 16-level grayscale modes, both settings need 4-bit data per 1 pixel. Even if set to 8 grayscales mode, the LSB bit of the display data is invalid.

- 256 colors STN

  Firstly, select STN256 color mode by LCDMODE1<MODE3:0> and next set the detail contrast adjustment. In 256STN color mode, 8-bit display data is divided into 3 bits (red), 3 bits (green) and 2 bits (blue). Red and green are 8-level contrast and blue is 4 level contrast. Each level can be adjusted from 16-level contrast.

  Red contrast level can be selected by LCDRP10, LCDRP32, LCDRP54 and LCDRP76registers, green by LCDGP10, LCDGP32, LCDGP54 and LCDGP76, and blue by LCDBP10, and LCDBP32.

  As a result, support is given for for 8 bpp: 256 colors out of a palette of 4096 colors.

- Color palette setting (Red)

Selectable 8-bit data	<<<							Contrast								>>>
	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 8-bit data	<<<							Contrast								>>>
	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 (Blue)

Selectable 8-bit data	<<<							Contrast								>>>
	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

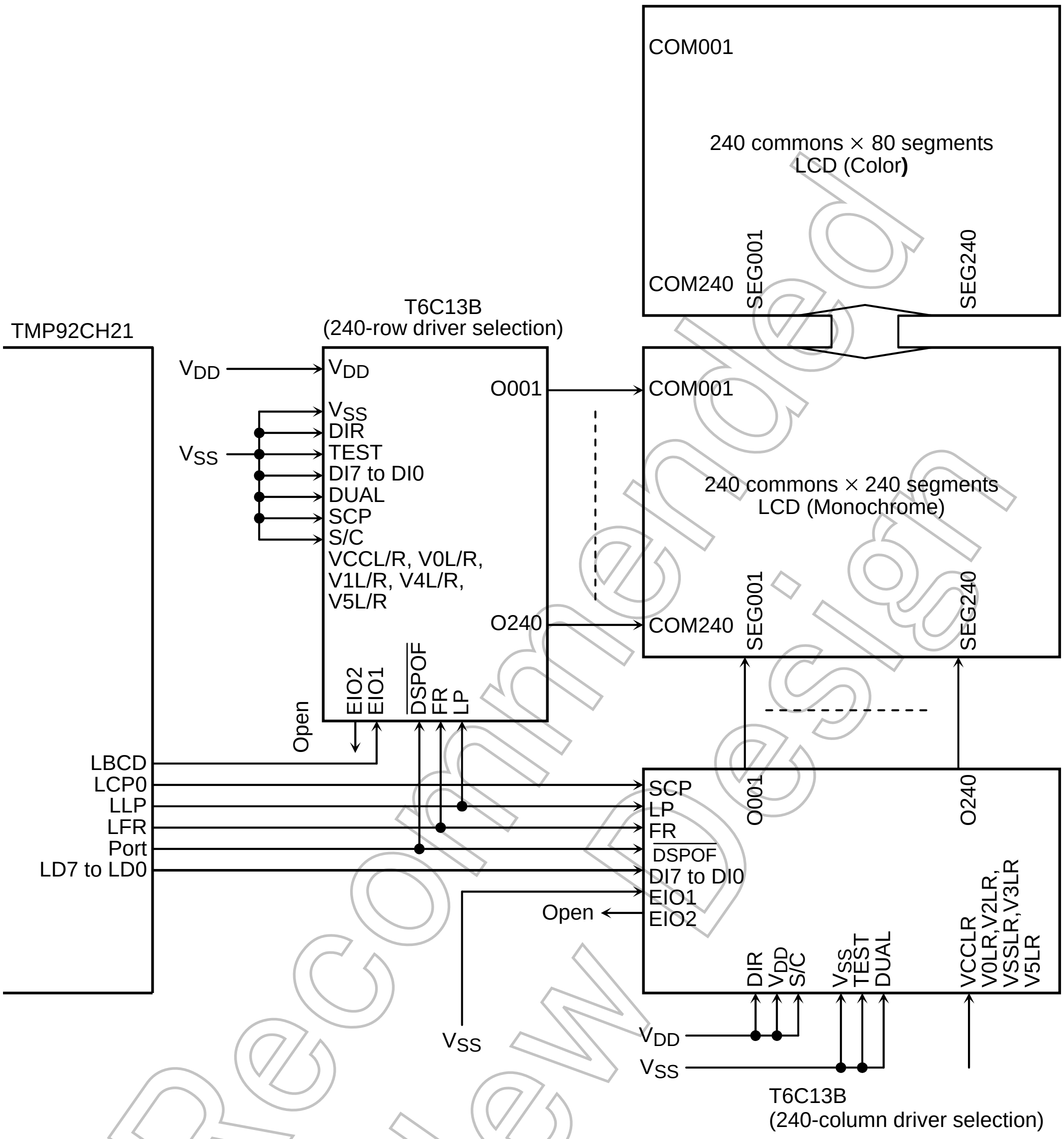
Figure 3.14.13 Palette Setting of Each Basic Color (RGB)

- 4096 colors STN

  STN4096 color mode is selected by LCDMODE1<MODE3:0>.

  This LCDC has a maximum 4096-color palette. If STN4096 color mode is selected, individual color contrast levels cannot be adjusted.

3.14.3.9 Example of SR Type LCD driver connection



Note: Other circuit is necessary for LCD drive power supply for LCD driver display.

Figure 3.14.14 Interface Example for Shift Register Type LCD Driver

3.14.3.10 Program Example (4 K colors STN)

```
; ********PORT settings *********
        ld      (PLFC), 0ffh            ;  LD7 to LD0 set
        ld      (PLCR), 0f0h            ;  Output mode
        ld      (PKFC), 0fh             ;  LBCD, LLP, LCP0

; ********LCD settings*********
        ld      (LCDSCC), 51            ;  Counter set (Refresh rate: 100 Hz at fc = 40 MHz)
        ld      (LCDCCR0), 01h          ;
        ld      (LCDCCR1), 01h          ;  SCP Negative edge
        ld      (LCDCCR2), 02h          ;

        ld      (LCDSIZE), 64h          ;  240 com × 256 seg
        ld      (LCDFFP), 240           ;  240 com
        ld      (LCDMODE0), 096h        ;  SDRAM, STN: 4 K
        ld      (LCDMODE1), 02h         ;  8-bit width A type

        ld      (LSARCL), 00h           ;  C area (enable)
        ld      (LSARCM), 00h           ;
        ld      (LSARCH), 40h           ;

        ld      (LSARAL), 00h           ;  A area (disable)
        ld      (LSARAM), 00h           ;
        ld      (LSARAH), 00h           ;

        ld      (LSARBL), 00h           ;  B area (disable)
        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: Negative edge, BCD:
        ld      (LCDDVM), 3
        ld      (LCDCTL0), 01h          ;  START (FP bit8 = 0)
```

3.14.4 TFT Color Display Mode

3.14.4.1 Description of Operation

This is basically the same setting as for 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 setting start register, the LCDC outputs a bus release request to the CPU and reads data from source memory. After data reading from source data is completed, the LCDC cancels the bus release request and the CPU will restart. The LCDC then transmits LCD size data to the external LCD driver through the LD bus (the special data bus only for LCD driver). At this time, the control signals (LCP0 etc.) connected to the LCD driver output the specified waveform which is synchronized with the data transmission.

In TFT mode LCDC, the CPU is stopped by the internal BUSREQ signal during data read from source memory (during DMA operation).

The LCD controller generates control signals (LFR, LBCD, LLP etc.) from base clock LCDSCC. LCDSCC is the base clock for the LCD controller, which is generated by system clock f_{SYS}.

For TFT source driver, the following signals are supported: 8-bit RGB or 4-bit × RGB special data bus and LCP0, LFR, LLP and LDIV.

And for TFT gate driver control, LCP1, LBCD and LGOE2 to LGOE0.

3.14.4.2 Memory Space

Memory space setting is the same as for SR mode. Refer to SR mode section.

3.14.4.3 Mapping of Display Memory and Panning Function

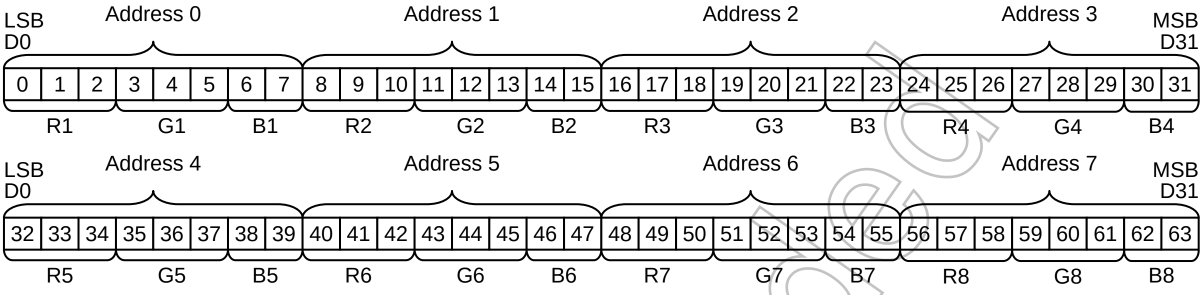
Panning function and display memory mapping are the same as for SR mode. Refer to SR mode section.

3.14.4.4 Data Transmission

This LSI outputs display data form special bus for LCD driver. The LCD driver input width can be selected. 8-bit and 12-bit widths are supported.

Relation of memory map image and output data

- 256 colors (8 bpp; R: 3 bits, G: 3 bits, B: 2 bits)
  Display memory image



LD bus output sequence

8 bits (TFT)

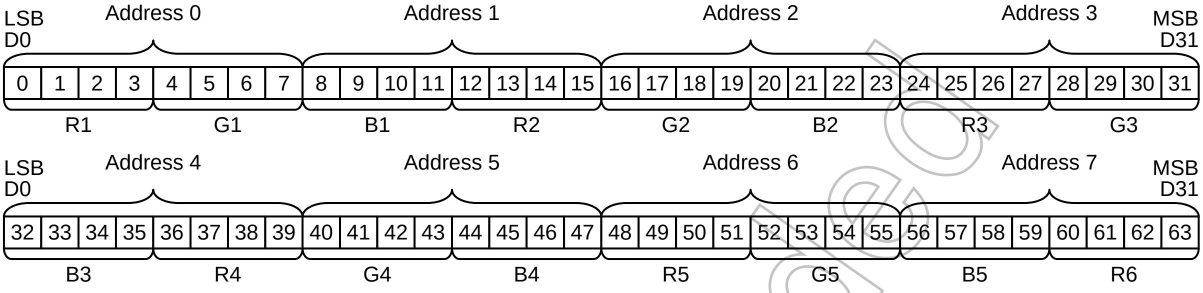
LD0	0 (R1)	→	8 (R2)	...
LD1	1 (R1)	→	9 (R2)	...
LD2	2 (R1)	→	10 (R2)	...
LD3	3 (G1)	→	11 (G2)	...
LD4	4 (G1)	→	12 (G2)	...
LD5	5 (G1)	→	13 (G2)	...
LD6	6 (B1)	→	14 (B2)	...
LD7	7 (B1)	→	15 (B2)	...

* When using 256-color TFT mode, 8-bit LD bus width must be used.
LD8, LD9, LD10 and LD11 terminals are available for use as general ports.

Figure 3.14.15 Relation of Memory Map Image and Output Data (5)

Relation of memory map image and output data

- 4096 colors (12 bpp; R: 4 bits, G: 4 bits, B: 4 bits)
  Display memory image



LD bus output sequence

12 bits (TFT)

LD0	0 (R1)	→	12 (R2)	…
LD1	1 (R1)	→	13 (R2)	…
LD2	2 (R1)	→	14 (R2)	…
LD3	3 (R1)	→	15 (R2)	…
LD4	4 (G1)	→	16 (G2)	…
LD5	5 (G1)	→	17 (G2)	…
LD6	6 (G1)	→	18 (G2)	…
LD7	7 (G1)	→	19 (G2)	…
LD8	8 (B1)	→	20 (B2)	…
LD9	9 (B1)	→	21 (B2)	…
LD10	10 (B1)	→	22 (B2)	…
LD11	11 (B1)	→	23 (B2)	…

Figure 3.14.16 Relation of Memory Map Image and Output Data (6)

3.14.4.5 Setting Each Control Signals

The TFT source driver is controlled by base clock (LCP0), data start clock (LFR) and load pulse (LLP). Special data bus LD11 to LD0 uses 8 bits or 12 bits for suitable LCD driver.

The timing of each signal can be finely adjusted using the relevant control register.

When using the TFT driver a large amount of data is required. So, when using wide bus and high speed transmission, some noise may be generated. This LSI has an LDIV function. This function automatically sets the minimum data change method, from inverting data and LDIV signal, to comparing current data with the previous data. If the TFT LCD driver supports the data inverting function, it is possible to decrease the noise.

The following shows basic timings.

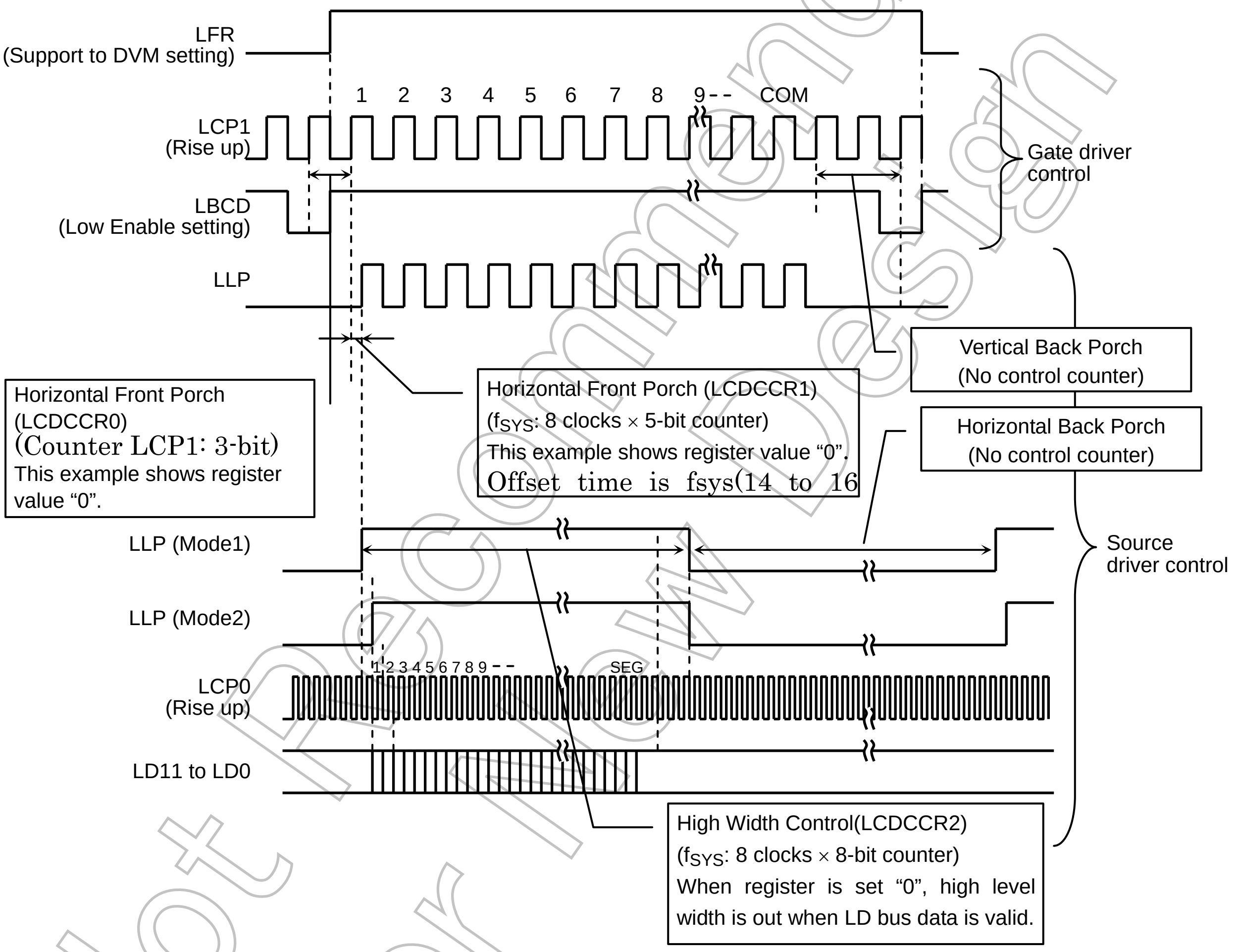

LFR
(Support to DVM setting)
1 2 3 4 5 6 7 8 9 - - COM
LCP1
(Rise up)
LBCD
(Low Enable setting)
LLP
Gate driver control
Vertical Back Porch
(No control counter)
Horizontal Front Porch
(LCDCCR0)
(Counter LCP1: 3-bit)
This example shows register value "0".
Horizontal Front Porch (LCDCCR1)
(f_{SYS}: 8 clocks × 5-bit counter)
This example shows register value "0".
Offset time is fsys(14 to 16
Horizontal Back Porch
(No control counter)
LLP (Mode1)
LLP (Mode2)
Source driver control
1 2 3 4 5 6 7 8 9 - -
SEG
LCP0
(Rise up)
LD11 to LD0
High Width Control(LCDCCR2)
(f_{SYS}: 8 clocks × 8-bit counter)
When register is set "0", high level width is out when LD bus data is valid.

Figure 3.14.17 Timing Diagram of TFT Driver Control

LCD Clock Counter Register 0

		7	6	5	4	3	2	1	0
LCDCCR0 (0288H)	Bit symbol						PCPV2	PCPV1	PCPV0
	Read/Write						R/W	R/W	R/W
	Reset State						0	0	0
	Function						Pre LCP1 CLK: LCP1 pulse number Dummy clock number until valid clock of gate driver LCP1		

Delay control 1 is set by LCDCCR0<PCPV2:0>. Delay of Dummy clock is controlled by pulse number derived by subtracting common number and <PCPV2:0> from LCDFFP<FP9:0> (refer to SR mode section).

LCD Clock Counter Register 1

		7	6	5	4	3	2	1	0
LCDCCR1 (0289H)	Bit symbol				TLDE4	TLDE3	TLDE2	TLDE1	TLDE0
	Read/Write				R/W	R/W	R/W	R/W	R/W
	Reset State				0	0	0	0	0
	Function				LLP_Set-up time: f_{SYS} pulse × 8 Set up time for TFT source driver LLP signal (Offset of f_{SYS} 14~16 pulse)				

Set up time of LLP (horizontal front porch) is set in LCDCCR1<TLDE4:0>. This is called “Delay control 2”. 1 pulse of this set up time in LCDCCR1 register is equal to 8 times of fsys regardless of LCP0 and LCP1. The set up time has offset time; fsys*14 to 16(f_{SYS} × 14.5 or more). If “0” is written in LCDCCR1 register, fsys*14 to 16 of time is delayed. This offset time changes according to the setting conditions. The cycle of LCP1 is determined by (the value of LCDSCC register +1) * fsys * 16, thus horizontal back porch is the time when offset time and set up time are subtracted from the cycle of LCP1.

LCD Clock Counter Register 2

		7	6	5	4	3	2	1	0
LCDCCR2 (028AH)	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
	Reset State	0	0	0	0	0	0	0	0
	Function	TFT source driver, LLP_Enable signal: f_{SYS} × 8 High width time for LLP signal							

The pulse number of LCP0 in LCDCCR2 means enable time of LLP. This register determines “High width” time as mentioned above. 1 pulse of this time in LCDCCR2 register is equal to 8 times of fsys regardless of LCP0 and LCP1. If “0” is written in LCDCCR2 register, High level is output during the period that the valid data is output from LD bus.

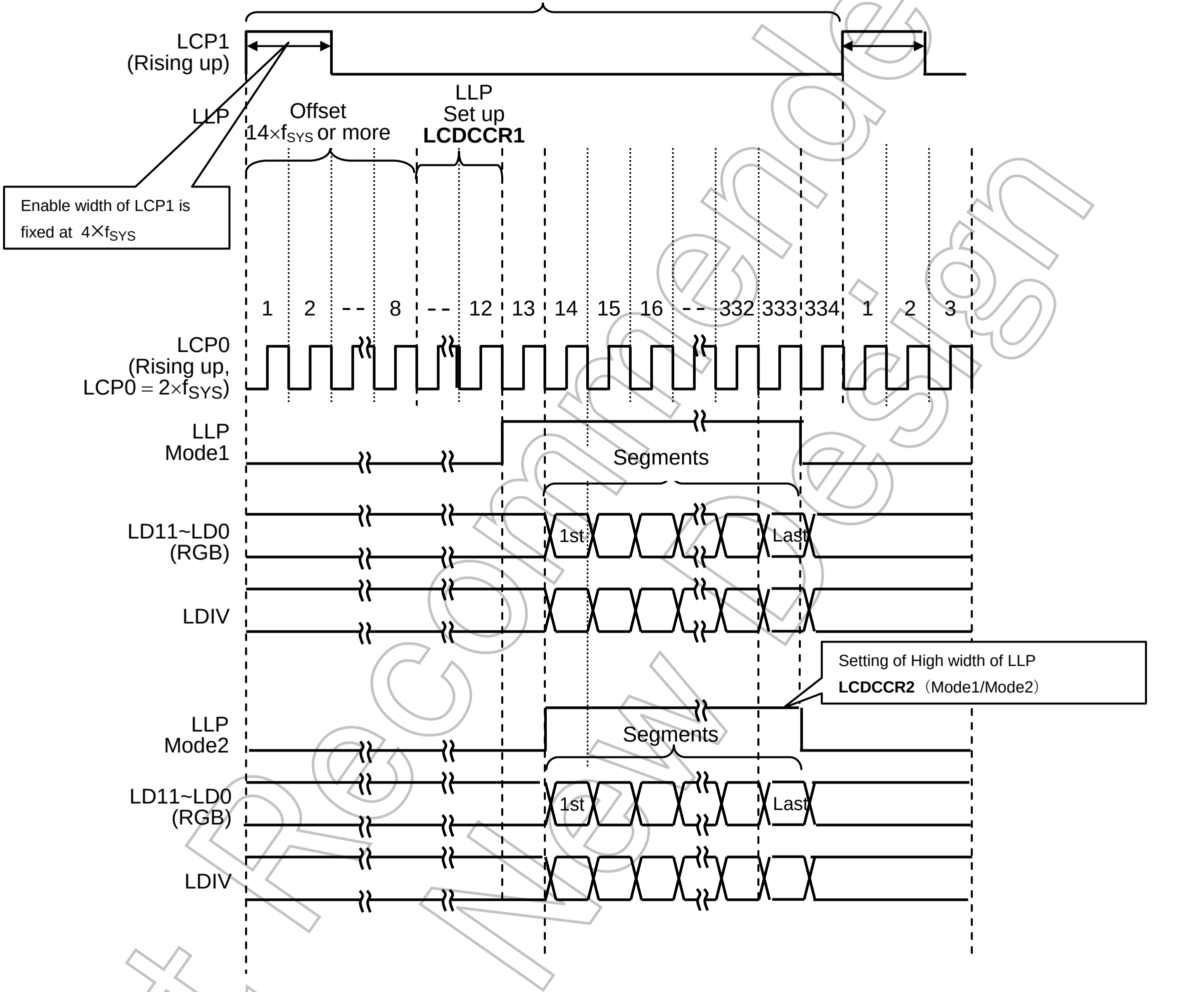
(In Mode1, high level is kept during one more LCP0 than valid data.)

3.14.5 Source Driver Control

Data shift clock; LCP0, Data; LD11 to LD0 and LLP signals become valid after the time (offset time + set up time set in (LCDCCR1)) from LCP1.

LLP signal has 2 modes; Mode1 (LLP rises 1 LCP0 clock before valid data) and Mode2 (LLP rises at the same time as valid data)

1. The cycle of LCP1 is determined by (LCDSCC+1) × f_{SYS} × 16
2. The number of LCP0 in the cycle of LCP1 is determined by the cycle of LCP0 clock.



* Condition: Segment = 320, LCDCCR1 = 1, LCP0 = 2×f_{SYS} , LCDSCC = 41

Figure 3.14.18 Timing Example for TFT Driver

Note: The above figure and explanation are under the following condition.
<LCDCTL2> CPHP = 1, CPVP = 0

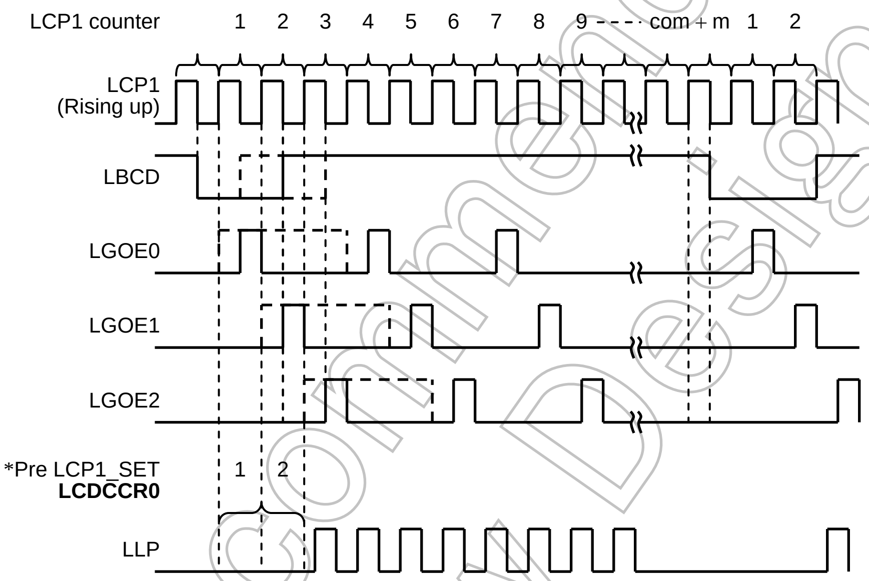
When <LCDCTL2> CPHP = 0, CPVP = 1 , the alignment of LCP0/LCP1 inverts.

3.14.5.1 Gate Driver Control

The TFT gate driver is controlled bybase clock LCP1and vertical shift data signal LBCD. This LSI has 3-bit output enable signals LGOE2 to LGOE0which can be controlled individually. The TFT gate driver's output can be controlled by this timing and is available for blanking adjustment and zoom function.

The LBCD signal begins output from rising of LCP1 and the TFT gate driver recognizes the start point of the vertical direction. LGOE0 then outputs from rising of LCP1 and repeats. LGOE1 outputs one LCP1 clock delayed. LGOE2 delay one LCP1 clock from LGOE1. By LCDCTL1<LBCDW1:0> setting, the width of LBCD can be selected from 1, 2, or 3 clocks of LCP1.

1. The cycle of LBCD is determined by the setting of (LCDFFP) register.
2. Enable width of LBCD is selectable from (1×LCP1) to (3×LCP1).



Note 1: LCP1 counter (LCDFFP) : 1024 clocks maximum

Note 2: Pre_LCP1_SET (LCDCCR0) : 8 clocks (3bits) of LCP1 maximum

Figure 3.14.19 Example of TFT Gate Driver Timing Control

(Notes on settings)

1. LCP0 cycle: LCP0=f_{SYS} × n (n=2, 4, 8: transmission speed of LD bus)
   LCP0 cycle is generated by system clock and value of LCDMODE0<SCPW1:0>
2. LCP1 cycle: LCP1= f_{SYS} × 16 × (SCC + 1)
   LCP1 cycle is generated by value of LCDSCC register.
   High level width of LCP1 is fixed to f_{SYS} × 4 times. (Positive edge)

As indicated above, the cycles of LCP0 and LCP1 are able to set each other.
There are some limitations to settings of LD bus speed and LCDSCC.

Segment Size	Transmission Speed of LD bus	Minimum LCDSCC value
64	2	9
	4	17
	8	33
128	2	17
	4	33
	8	65
160	2	21
	4	41
	8	81
256	2	33
	4	65
	8	129
320	2	41
	4	81
	8	161

High level width of LLP is adjusted every f_{SYS} × 8 cycle. However, LLP is adjusted every 2-clock of LLP when transmission speed of the LD bus is set to 2-Clock. LLP is also adjusted every 4-clock of LLP when transmission speed of the LD bus is set to 4-clock of LCP0.

Setting method is the same as in the STN case, following the calculation below.

f_{BCD} [Hz]	: Frame frequency (Refresh rate: LBCD cycle)
FP	: FP [9:0] FFP register setting value
SCC	: SCC [7:0] LSCC register setting value

$$f_{BCD}\ [\mathrm{Hz}] = f_{SYS}\ [\mathrm{Hz}] / ((SCC+1) \times 16 \times FP)$$

Frame correction function is the same as in the STN case.

3. LCP1 Setting: Vertical front porch is determined by LCDCCR0.

Vertical front porch is determined by the above 3bits; LCDCCR0<PCPV2:0>. Vertical back porch is controlled by the pulse number which is <PCPV2:0> subtracted from LCDFFP<FP9:0> as explained in the SR mode section.

4. LLP Setting: Set up time is determined by LCDCCR1.

Set up time of LLP (horizontal front porch) is set in LCDCCR1 register. This is called “Delay control 2”. 1 pulse of this set up time in LCDCCR1 register is equal to 8 times of fsys regardless of LCP0 and LCP1. The set up time has offset time; fsys×14 or more. If “0” is written in LCDCCR1 register, fsys×14.5 or more of time is delayed. This offset time changes according to the setting conditions. The cycle of LCP1 is determined by (the value of LCDSCC register +1) × fsys × 16, thus horizontal back porch is the time where offset time and set up time are subtracted from the cycle of LCP1.

5. LLP High width: High width of LLP is determined by LCDCCR2.

The pulse number of LCP0 in LCDCCR2 means enable time of LLP. This register determines “High width” time as mentioned above. 1 pulse of this time in LCDCCR2 register is equal to 8 times of fsys regardless of LCP0 and LCP1. If “0” is written in LCDCCR2 register, High level is output during the period that the valid data is output from the LD bus.

(In Mode1, high level is kept during one more LCP0 than valid data.)

6. LDIV: Enable/disable of Auto Invert function is determined by LCDMODE1<AUTOINV>.

If “1” is written in LCDMODE1<AUTOINV> bit, the LCD controller monitors the status of the LD bus. The LCDC compares the value of previous data with the data supposed to be sent. If more than a majority of all the LD bus change, LDIV outputs “1” and the LCDC inverts the LD bus value that was supposed to be sent.

For example, if 4096 color(12bit : LD11 to LD0) and the data changes from 000000000000→111111111111, the data remains 000000000000→000000000000 and only LDIV changes 0→1.

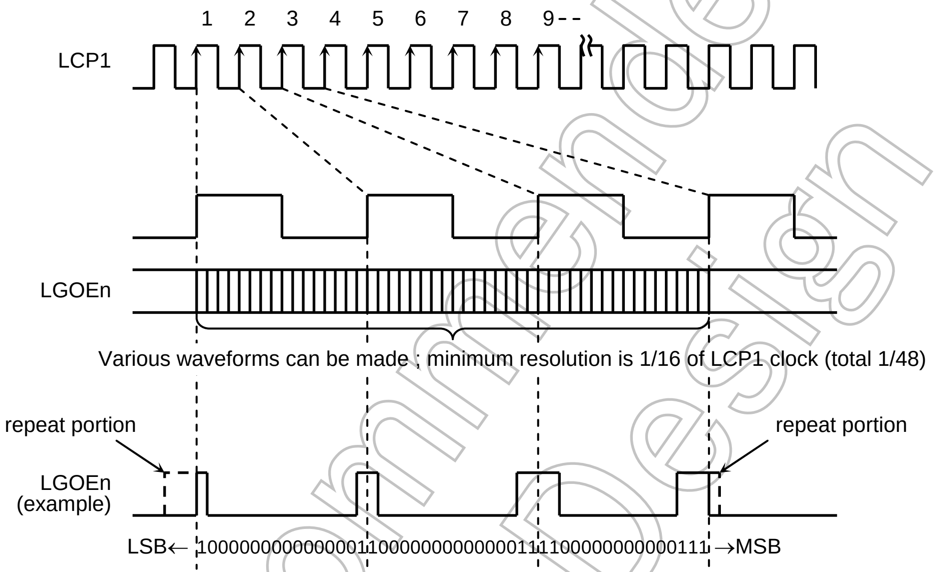
This is effective in reducing noise or power consumption where the LCD driver has an LDIV pin.

7. LGOE0 to 2: Programmable waveform

LGOE0 is output on the rising edge of the first LCP1 and repeats every 3 pulses of LCP1. LGOE1 is the 2nd LCP1 and LGOE2 the 3rd, and they also repeat every 3 pulses of LCP1.

LGOE0 to LGOE2 can be generated by 1/16 clocks of LCP1 cycle set in control registers (48bits × 3) respectively.

These signals can finely adjust the gate output signal and this will enables fine adjustment of gate bias (the setting of blanking) or zoom function without modification of data etc.


1 2 3 4 5 6 7 8 9- -
LCP1
LGOEn
Various waveforms can be made ; minimum resolution is 1/16 of LCP1 clock (total 1/48)
repeat portion
repeat portion
LGOEn
(example)
LSB← 1000000000000001 1000000000000011 1100000000000111 →MSB

* Various waveforms can be generated by writing LCDOEn5 to LCDOEn0 registers (48bits for each signal). (When “1” is written, output is High level. When “0” is written, output is Low level. The direction of data is from LSB to MSB.)

Figure 3.14.20 Details of waveform of LGOEn for gate driver

Note1) The above explanation and figures are given for the setting below.
Condition: <LCDCTL2>CPHP = 1, CPVP = 0

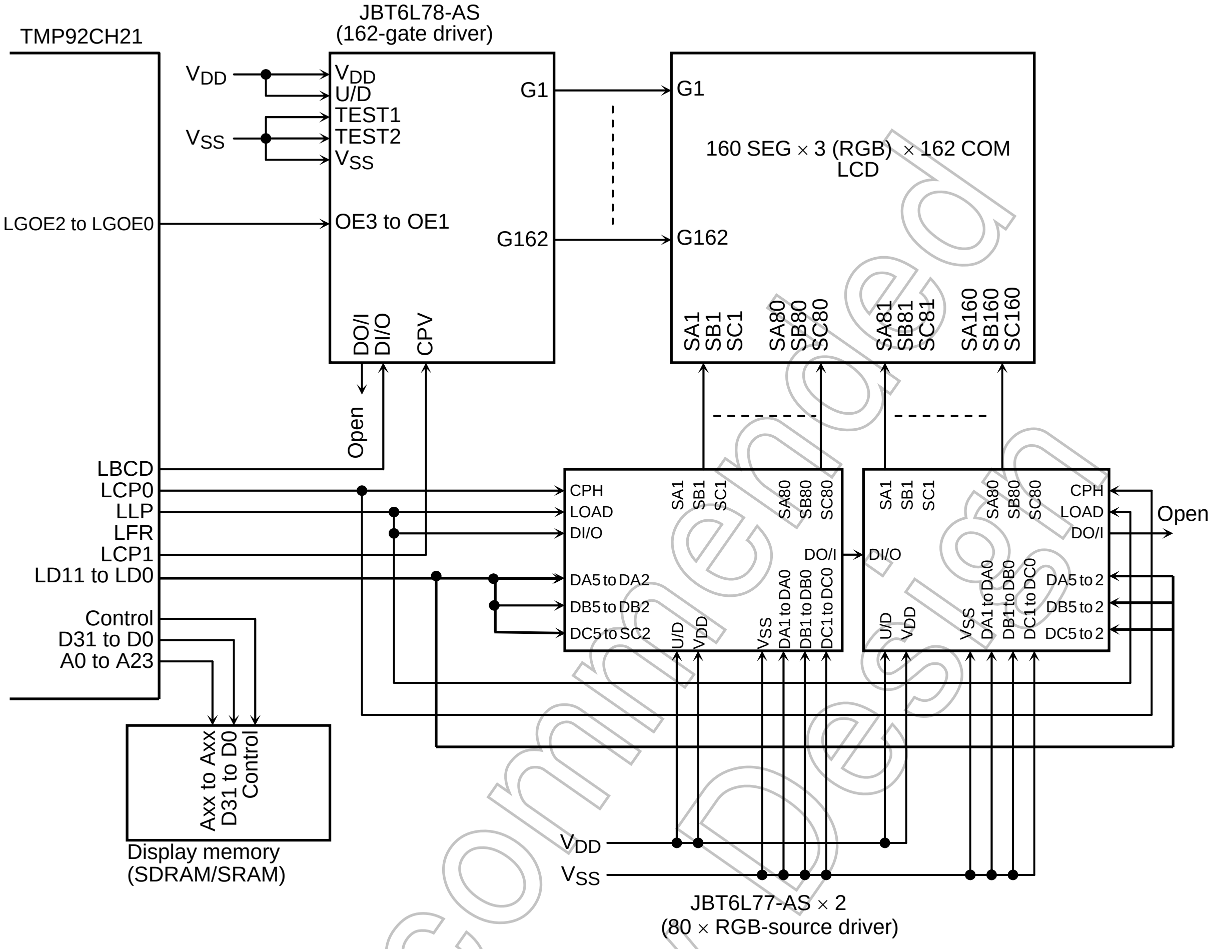
For CPHP = 0, CPVP = 1 setting, LCP0 and LCP1 phases are inverted.

Note2) The minimum resolution of LGOEn is 1/16 of LCP1 cycle.

LCP1= (LCDSCC+1) × f_{SYS} ×16

Thus, the minimum resolution of LGOEn is (LCDSCC +1) × f_{SYS}.

3.14.5.2 Example of TFT LCD driver connection



Note: Other circuit is required for power supply for driving LCD driver.

Figure 3.14.21 Example of TFT Type LCD Driver Interface

3.14.5.3 Program sample (4K color TFT)

```
; ********PORT settings *********
        ld          (PACR), 78h          ;  LD11-LD8 set
        ld          (PLFC), 0ffh         ;  LD7-LD0 set
        ld          (PLCR), 0f0h         ;  Output mode
        ld          (PKFC), 0bh          ;  LBCD, LLP, LCP0
        ld          (PCCR), 0c0h         ;  PC6: LDIV (for TFT) PC7: LCP1
        ld          (PCFC), 0c0h         ;  PC6: LDIV

; ********LCD settings*********
        ld          (LCDSCC), 100        ;  Counter set (refresh rate:50Hz at fc = 40MHz)
        ld          (LCDCCR0), 00h       ;
        ld          (LCDCCR1), 00h       ;
        ld          (LCDCCR2), 00h       ;

        ld          (LCDSIZE), 74h       ;  320 com × 256 seg
        ld          (LCDFFP), 49h        ;  320 com
        ld          (LCDMODE0), 059h     ;  SRAM, TFT 4096 color
        ld          (LCDMODE1), 01h      ;  Invalid 8bit A type

        ld          (LCDCTL0), 02h       ;  (FP bit8=1)
        ld          (LCDCTL1), 00h       ;  SCP0,SCP1:negedge, BCD: ↓

        ld          (LSARCL), 00h        ;  C area (enable)
        ld          (LSARCM), 00h        ;
        ld          (LSARCH), 40h        ;

        ld          (LSARAL), 00h        ;  A area (disable)
        ld          (LSARAM), 00h        ;
        ld          (LSARAH), 00h        ;

        ld          (LSARBL), 00h        ;  B area (disable)
        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          (LCDCTL0), 03h       ;  START (FP bit8 = 1)
```

3.14.6 Built-in RAM Type LCD driver Mode

3.14.6.1 Description of Operation

Data transmission to the LCD driver is executed by a transmit instruction from the CPU.

After setting operation mode of to the control register, when a CPU transmit instruction is executed the LCDC outputs a chip select signal to the LCD driver connected externally by the control pin (LCP0...). Therefore control of data transmission numbers corresponding to LCD size is controlled by CPU instruction. There are 2 kinds of LCD driver address in this case, which are selected by the LCDCTL<MMULCD> register.

3.14.6.2 Random Access Type

This corresponds to address direct writing type LCD driver when <MMULCD> = "1".

The transmission address can also assign the memory area 3C0000H – 3FFFFF, the four areas each being 64 Kbytes.

Interface and access timing are the same as for normal memory. Refer to the memory access timing section.

Table 3.14.2 Racdom Access Type Built-in RAM Type LCD driver

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

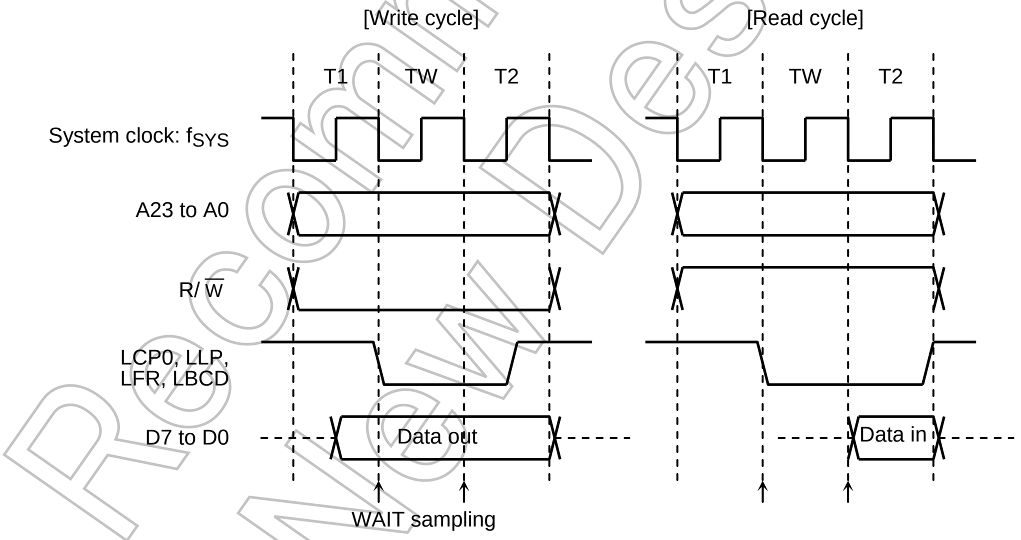
Data transmission to the LCD driver is executed by a transmit instruction from the CPU.

After setting operation mode to the control register, when a CPU transmit instruction is executed the LCDC outputs a chip select signal to the LCD driver connected externally by the control pin (LCP0...). Therefore control of data transmission numbers corresponding to LCD size is controlled by CPU instruction. There are 2 kinds of LCD driver address in this case, which are selected by the LCDCTL<MMULCD> register.

This corresponds to a LCD driver which has each 1 byte of instruction register and display data register in LCD driver when <MMULCD> = "0". Please select the transmission address at this time from 1FE0H to 1FE7H.

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	Depends on external LCDD specification							
Reset State	Depends on external LCDD specification							
Function	Depends on external LCDD specification							

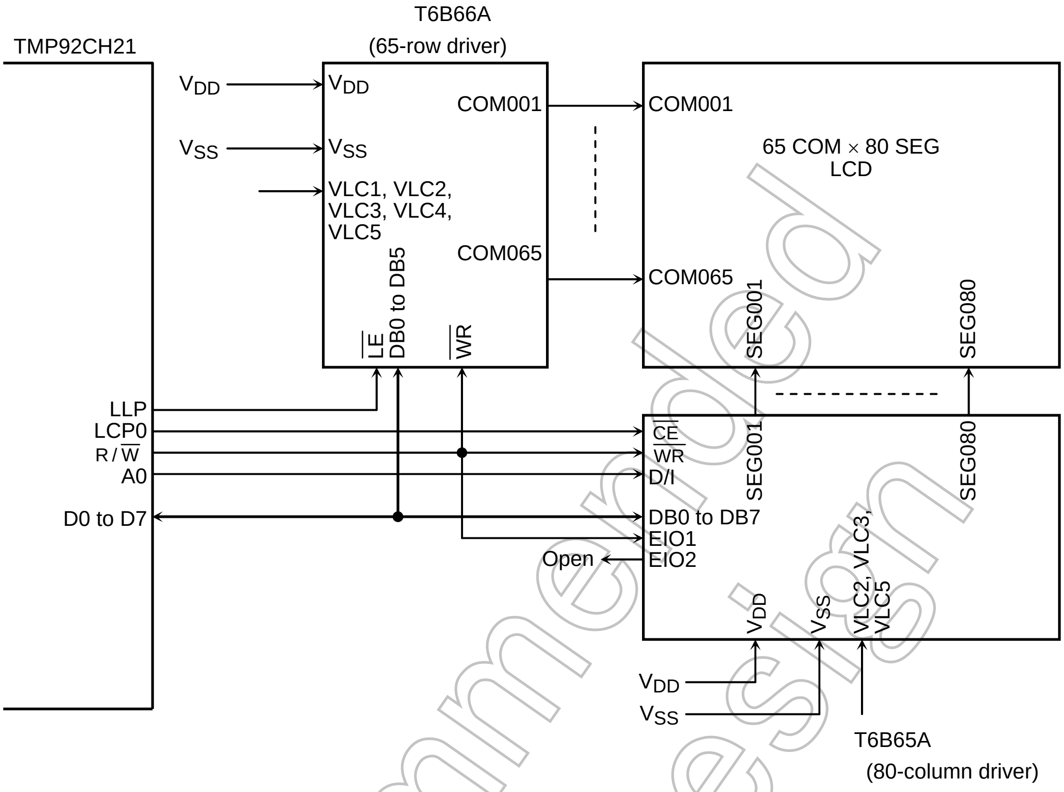


Note 1: This waveform is in the case of 3-state access.

Note 2: Rising timing of chip enable signal (e.g LCP0) is different.

Figure 3.14.22 Example of Access Timing for Built-in RAM Type LCD Driver (Wait = 0)

3.14.6.4 Example of Built-in RAM LCD driver connection



Note: Other circuit is required for power supply for LCD driver display.

Figure 3.14.23 Interface Example for Built-in RAM and Sequential Access Type LCD Driver

3.14.6.5 Program Example

- Setting example: when using 80 segments × 65 commons LCD driver.

Assign external column driver to LCDC1 and row driver to LCDC4.

This example uses LD instruction in setting of instruction and micro DMA burst function for soft start in setting of display data.

When storing 650-byte transfer data to LCD driver.

```
; ********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, 002000h          ;  Source address = 002000H
        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)

The TMP92CH21 contains a melody function and alarm function, both of which are output from the MLDALM pin. Five kinds of fixed cycle interrupt are generated using a 15-bit counter for use as the alarm generator.

The features are as follows.

1) Melody generator

The Melody function generates signals of any frequency (4 Hz to 5461 Hz) based on a low-speed clock (32.768 kHz), and outputs the signals from the MLDALM pin.

The melody tone can easily be heard by connecting an external loudspeaker.

2) Alarm generator

The alarm function generates eight kinds of alarm waveform having a modulation frequency (4096 Hz) determined by the low-speed clock (32.768 kHz). This waveform can be inverted by setting a value to a register.

The alarm tone can easily be heard by connecting an external loudspeaker.

Five kinds of fixed cycle interrupts are generated (1 Hz, 2 Hz, 64 Hz, 512 Hz, and 8192 Hz) by using a counter which is used for the alarm generator.

This section is constituted as follows.

3.15.1 Block Diagram

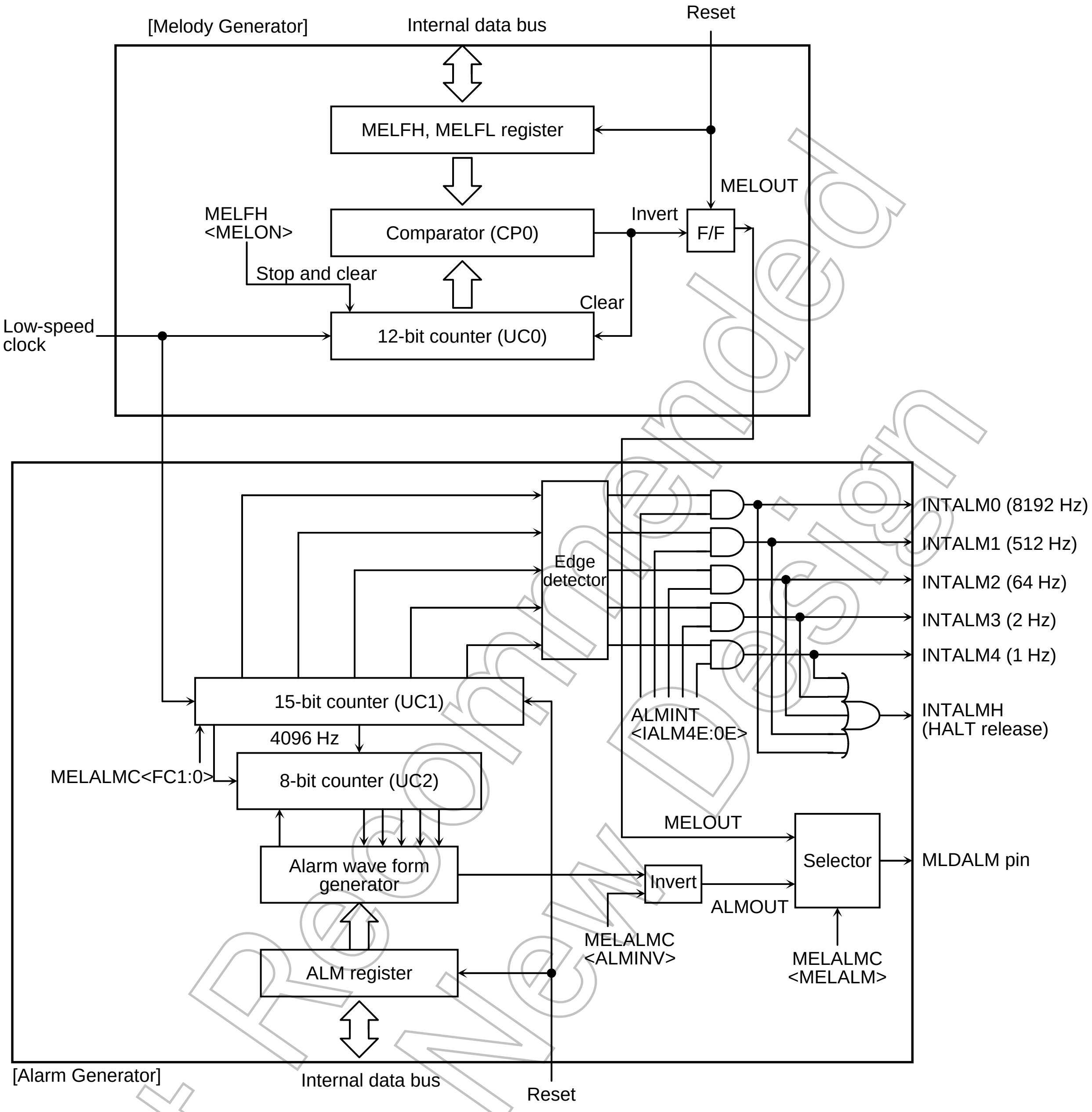


Figure 3.15.1 MLD Block Diagram

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							
Reset State	0	0	0	0	0	0	0	0
Function	Setting alarm pattern							

MELALMC Register

MELALMC
(1331H)

	7	6	5	4	3	2	1	0
Bit symbol	FC1	FC0	ALMINV	–	–	–	–	MELALM
Read/Write	R/W		R/W	R/W				R/W
Reset State	0	0	0	0	0	0	0	0
Function	Free-run counter control 00: Hold 01: Restart 10: Clear and stop 11: Clear and start		Alarm waveform invert 1: Invert	Always write “0”				Output waveform select 0: Alarm 1: Melody

Note 1: MELALMC<FC1> is always read “0”.

Note 2: When setting MELALMC register except <FC1:0> while 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							
Reset State	0	0	0	0	0	0	0	0
Function	Setting melody frequency (Lower 8 bits)							

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			
Reset State	0				0	0	0	0
Function	Control melody counter 0: Stop and clear 1: Start				Setting melody frequency (Upper 4 bits)			

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				
Reset State			0	0	0	0	0	0
Function			Always write “0”	1: Interrupt enable for INTALM4 to INTALM0				

3.15.3 Operational description

3.15.3.1 Melody Generator

The Melody function generates signals of any frequency (4 Hz to 5461 Hz) based on a low-speed clock (32.768 kHz) and outputs the signals from the MLDALM pin.

The melody tone can easily be heard by connecting an external loud speaker.

(Operation)

MELALMC<MELALM> must first be set as 1 in order to select the melody waveform to be output from MLDALM. The melody output frequency must then be set to 12-bit registers MELFH and MELFL.

The following are examples of settings and calculations of melody output frequency.

(Formula for calculating melody waveform frequency)

at fs = 32.768 [kHz]

Melody output waveform f_{MLD} [Hz] $= 32768/(2 \times N + 4)$

Setting value for melody $N = (16384/ f_{MLD}) - 2$

(Note: N = 1 to 4095 (001H to FFFH), 0 is not acceptable.)

(Example program)

When outputting an “A” musical note (440 Hz)

```
LD      (MELALMC), − − X X X X X 1 B   ;  Select melody waveform
LD      (MELFL), 23H                   ;  N = 16384/440 − 2 = 35.2 = 023H
LD      (MELFH), 80H                   ;  Start to generate waveform
```

Reference) 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 of 4096 Hz determined by the low-speed clock (32.768 kHz). This waveform is reversible by setting a value to a register.

The alarm tone can easily be heard by connecting an external loud speaker.

Five kinds of fixed cycle (interrupts can be generated 1 Hz, 2 Hz, 64 Hz, 512 Hz, 8 192 Hz) by using a counter which is used for the alarm generator.

(Operation)

MELALMC<MELALM> must first be set as 0 in order to select the alarm waveform to be output from MLDALMC. The "10" must be set on the MELALMC <FC1:0> register, and clear internal counter. Alarm pattern must then be set on the 8-bit register of ALM. If it is inverted output-data, set <ALMINV> as invert.

Then set the MELAMC<FC1:0> to "11" to start the free-run counter.

To stop the alarm output, write "00H" to the ALM register.

The following are examples of program, setting value of alarm pattern and waveform of each setting value.

(Setting value of alarm pattern)

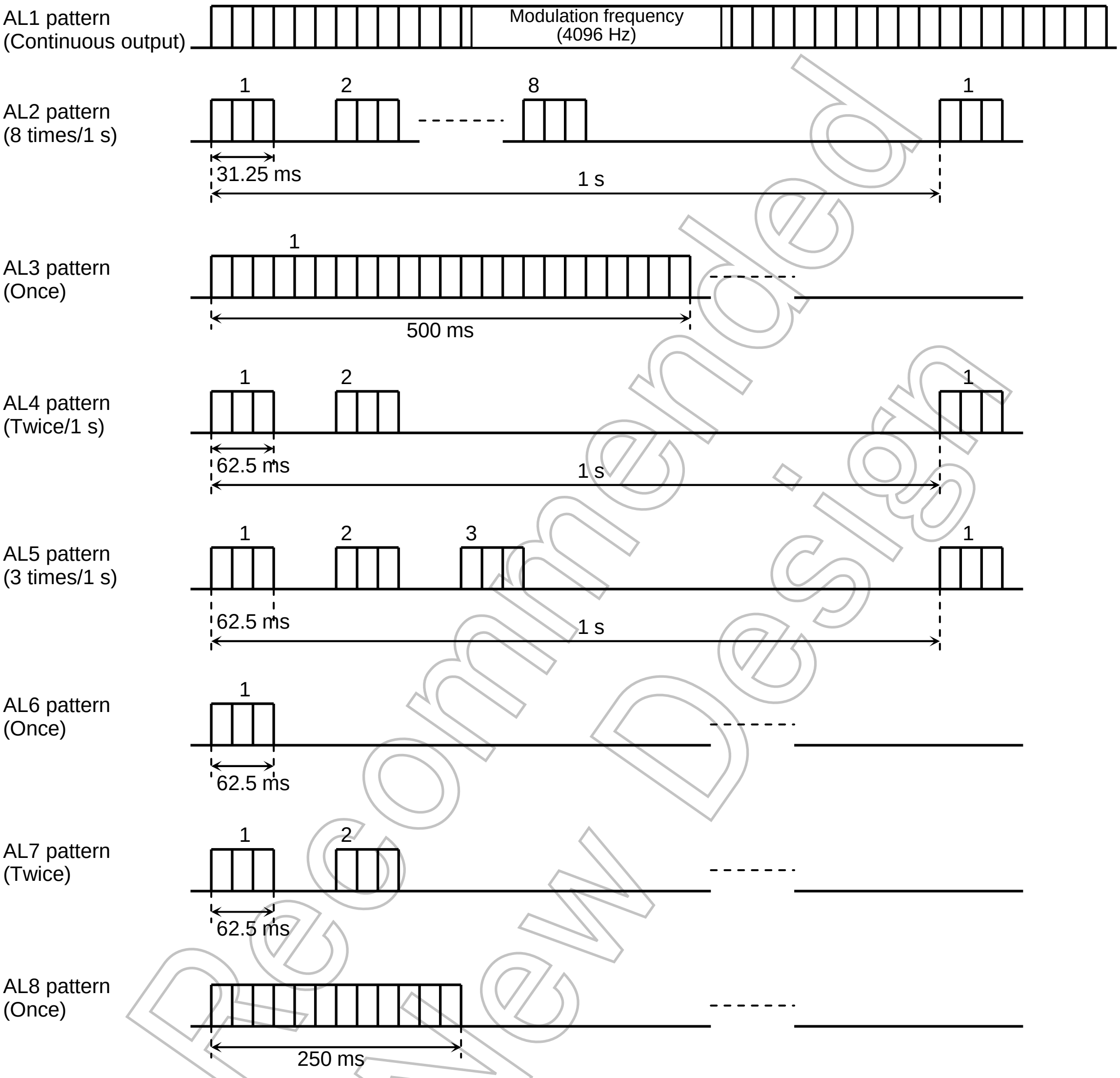
Setting Value for ALM Register	Alarm Waveform
00H	Write "0"
01H	AL1 pattern
02H	AL2 pattern
04H	AL3 pattern
08H	AL4 pattern
10H	AL5 pattern
20H	AL6 pattern
40H	AL7 pattern
80H	AL8 pattern
Others	Undefined (Do not set)

(Example program)

When outputting AL2 pattern (31.25 ms/8 times/1 s)

```
        LD      (MELALMC), 80H          ;  Clear counter, set output alarm waveform
        LD      (ALM), 02H              ;  Set AL2 pattern
        LD      (MELALMC), C0H          ;  Free-run counter start
```

Example: Waveform of alarm pattern for each setting value (Not inverted)



3.16 SDRAM Controller (SDRAMC)

The TMP92CH21 includes an 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:	16/64/128/256/512 Mbits
Number of banks:	2/4 banks
Width of data bus:	16/32
Read burst length:	1 word/full page
Write mode:	Single/burst

(2) Initialize function

All banks precharge command
8 times auto refresh command
Set the mode register command

(3) Access mode

	CPU Access	LCDC Access
Read burst length	1 word/full page selectable	Full page
Addressing mode	Sequential	Sequential
CAS latency (clock)	2	2
Write mode	Single/burst selectable	−

(4) Access cycle

CPU Access (Read/write)

Read cycle:	1 word– 4 states/full page − 1 state
Write cycle:	Single − 3 states/burst − 1 state
Access data width:	8 bits/16 bits/32 bits

LCDC Burst Access (Read only)

Read cycle:	1 word (50 ns at f_{SYS} = 20 MHz)
Over head:	4 states (200 ns at f_{SYS} = 20 MHz)
Access data width:	16 bits/32 bits

(5) Refresh cycle auto generate

Auto-refresh is generated while another area is being accessed.

Refresh interval is programmable.

Self-refresh is supported

Note 1: Display data for LCDC must be set from the head of each page.

Note 2: Condition of SDRAM's area set by CS1 or CS2 setting of memory controller.

3.16.1 Control Registers

Figure 3.16.1 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	–	–	SMRD	SWRC	SBST	SBL1	SBL0	SMAC
Read/Write	R/W							
Reset State	0	0	0	0	0	1	0	0
Function	Always write "0"	Always write "0"	Mode register set delay time 0: 1 clock 1: 2 clocks	Write recover time 0: 1 clock 1: 2 clocks	Burst stop command 0: Precharge all 1: Burst stop	Selecting burst length (Note 1) 00: Reserved 01: Full-page read, burst write 10: 1-word read, single write 11: Full-page read, single write		SDRAM controller 0: Disable 1: Enable

Note 1: Issue mode register set command after changing <SBL1:0>. Exercise care in settings when changing from "full-page read" to "1-word read". Please refer to "3.16.3 Limitations arising when using SDRAM".

SDRAM Access Control Register 2

SDACR2 (0251H)

	7	6	5	4	3	2	1	0
Bit symbol				SBS	SDRS1	SDRS0	SMUXW1	SMUXW0
Read/Write				R/W				
Reset State				0	0	0	0	0
Function				Number of banks 0: 2 banks 1: 4 banks	Selecting ROW address size 00: 2048 rows (11 bits) 01: 4096 rows (12 bits) 10: 8192 rows (13 bits) 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			
Reset State					0	0	0	0
Function					Refresh interval 000: 47 states 001: 78 states 010: 97 states 011: 124 states 100: 156 states 101: 195 states 110: 249 states 111: 312 states			Auto refresh 0: Disable 1: Enable

SDRAM Command Register

		7	6	5	4	3	2	1	0
SDCMM (0253H)	Bit symbol						SCMM2	SCMM1	SCMM0
	Read/Write						R/W		
	Reset State						0	0	0
	Function						Command issue (Note 1) (Note 2) 000: Not execute 001: Initialization sequence a. Precharge All command b. Eight Auto Refresh commands c. Mode Register Set command 100: Mode Register Set command 101: Self Refresh Entry command 110: Self Refresh Exit command Others: Reserved		

Note 1: <SCMM2:0> is automatically cleared to "000" after the specified command is issued. Before writing the next command, make sure that <SCMM2:0> is "000". In the case of the Self Refresh Entry command, however, <SCMM2:0> is not cleared to "000" by execution of this command. Thus, this register can be used as a flag for checking whether or not Self Refresh is being performed.

Note 2: The Self Refresh Exit command can only be specified while Self Refresh is being performed.

Figure 3.16.1 SDRAM Control Registers

3.16.2 Operation Description

(1) Memory access control

SDRAM controller is enabled when SDACR1<SMAC> = 1. And then SDRAM control signals ($\overline{\text{SDCS}}$, $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, $\overline{\text{SDWE}}$, SDLLDQM, SDLUDQM, SDULDQM, SDUUDQM, SDCLK and SDCKE) are operating during the time CPU or LCDC accesses CS1 or CS2 area.

1. Address multiplex function

In the access cycle, outputs row/column address through A0 to A15 pin. And multiplex width is decided by setting SDACR2<SMUXW0:1> of use memory size. The relation between multiplex width and Row/Column address is shown in Table 3.16.3.

Table 3.16.1 Address Multiplex

TMP92CH21 Pin Name	Address of SDRAM Accessing Cycle				
	Row Address			Column Address	
	TypeA <SMUXW> “00”	TypeB <SMUXW> “01”	TypeC <SMUXW> “10”	16-Bit Data Bus Width B1CSH<BnBUS> = “01”	32-Bit 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	EA25	EA26		

* AP: Auto Precharge

2. Burst length

When the CPU accesses the SDRAM, the burst length is fixed to 1-word read/single write. When the LCDC accesses the SDRAM, the burst length is fixed to full page.

SDRAM access cycle is shown in Figure 3.16.2 and Figure 3.16.3.

SDRAM access cycle number does not depend on the settings of B1CSL and B2CSL registers. In the full page burst read cycle, a mode register set cycle and a precharge cycle are automatically inserted at the beginning and end of a cycle.

(2) Instruction executing on SDRAM

The CPU can execute instructions on SDRAM. However, the following functions do not operate.

a) Executing HALT instruction

b) Execute instructions that write to SDCMM register

These operations must be executed by another memory such as the built-in RAM.

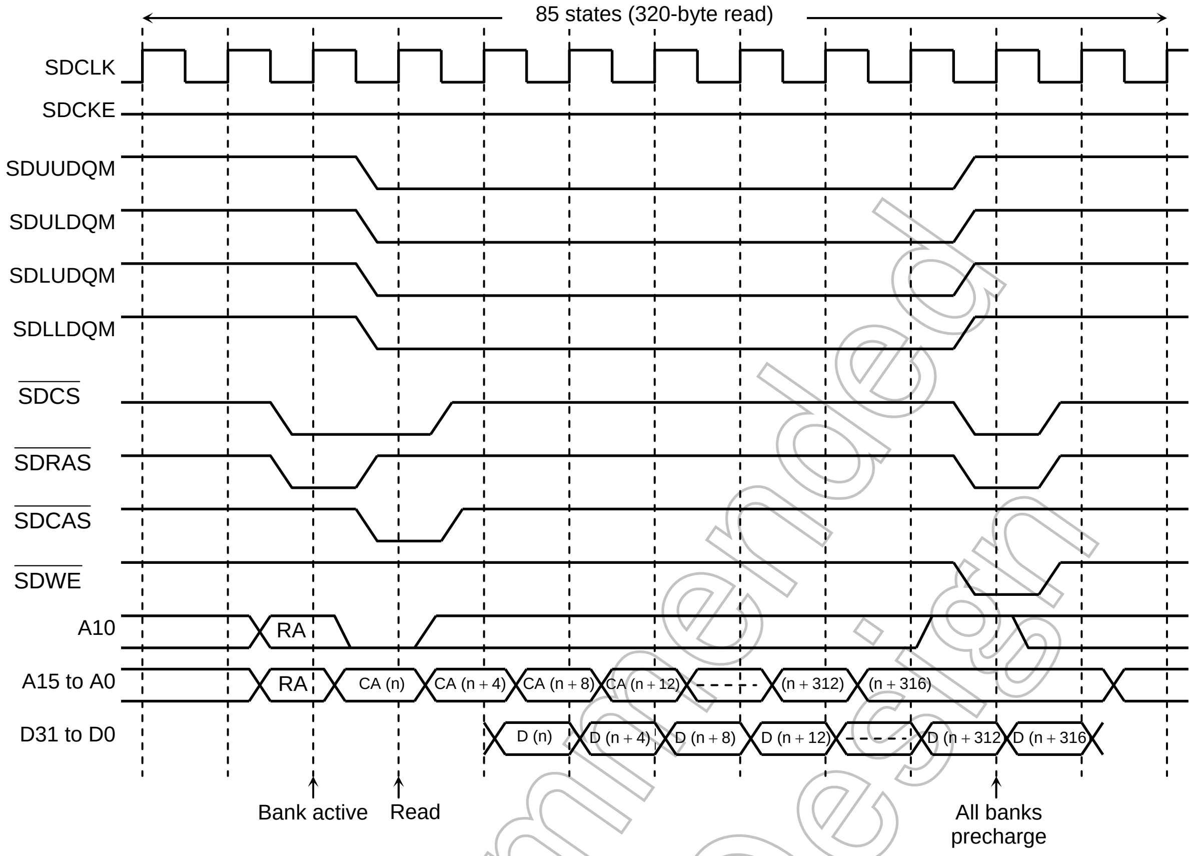


Figure 3.16.2 Timing of Burst Read Cycle

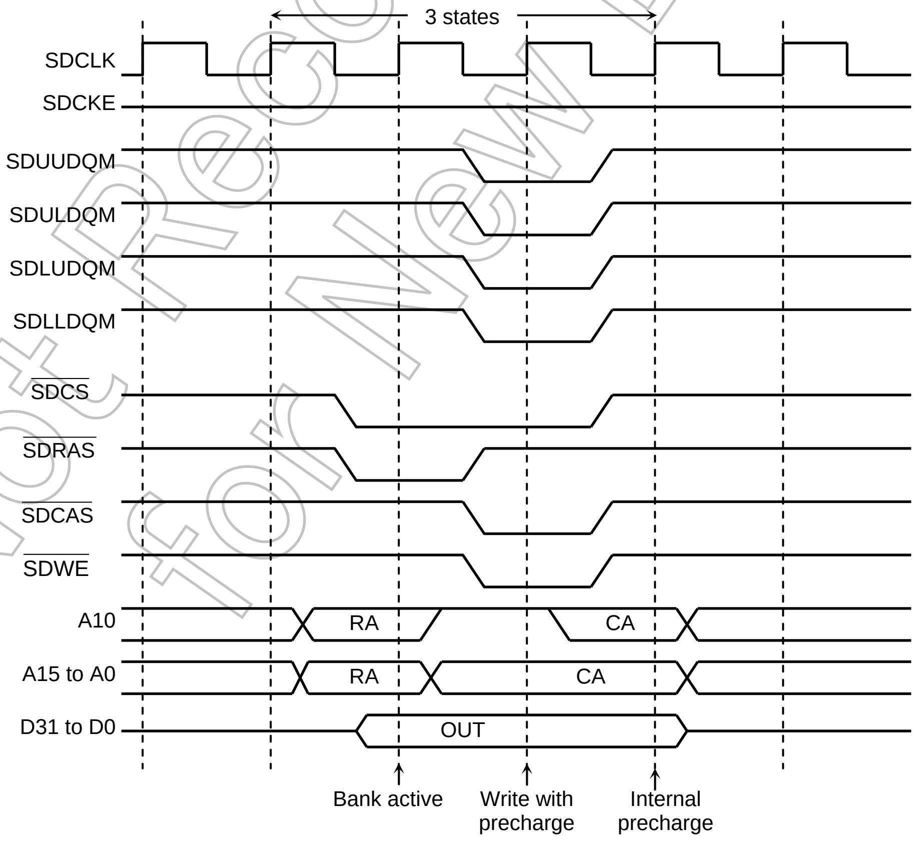


Figure 3.16.3 Timing of CPU Write Cycle
(Structure of Data Bus: 32 bits × 1, operand Size: 4 bytes, address: 4 n + 0)

(3) Refresh control

This LSI supports two refresh commands: auto-refresh and self-refresh.

(a) Auto-refresh

The auto-refresh command is automatically generated at intervals set by SDRCR<SRS2:0> by setting SDRCR<SRC> to "1". The generation interval can be set from between 47 to 312 states (2.4 μs to 15.6 μs at f_{SYS} = 20 MHz).

CPU operation (instruction fetch and execution) stops while performing the auto-refresh command. The auto-refresh cycle is shown in Figure 3.16.4 and the auto-refresh generation interval is shown in Table 3.16.2. The Auto-Refresh function cannot be used in IDLE1 and STOP modes. In these modes, use the Self-Refresh function to be explained next.

Note: A system reset disables the Auto-Refresh function.

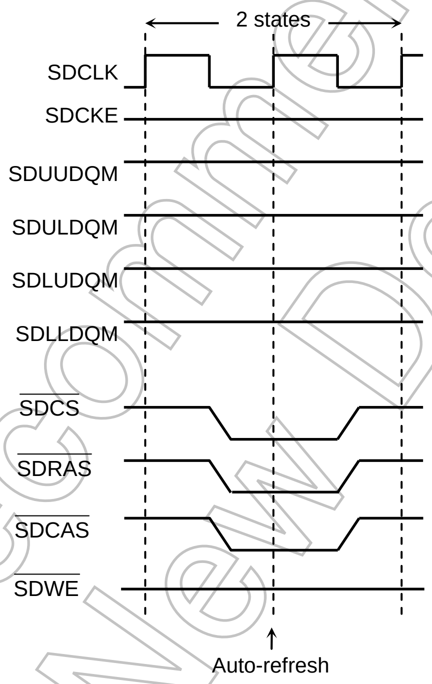


Figure 3.16.4 Timing of Auto-Refresh Cycle

Table 3.16.2 Refresh Cycle Insertion Interval (Unit: μs)

SDRCR<SRS2:0>			Insertion Interval (State)	f_{SYS} Frequency (System clock)					
SRS2	SRS1	SRS0		6 MHz	10 MHz	12.5 MHz	15 MHz	17.5 MHz	20 MHz
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	249	41.5	24.9	19.9	16.6	14.2	12.4
1	1	1	312	52.0	31.2	25.0	20.8	17.8	15.6

(b) Self-refresh

The self-refresh ENTRY command is generated by setting SDCMM<SCMM2:0> to "101". The self-refresh cycle is shown in Figure 3.16.5. During self-refresh Entry, refresh is performed within the SDRAM (an auto-refresh command is not needed).

The auto-refresh command is automatically executed once when self-refresh is released, following which, refresh is executed according to the setting of the auto-refresh command.

Note 1: When standby mode is released by a system reset, the I/O registers are initialized and the Self Refresh state is exited. Note that the Auto Refresh function is also disabled at this time.

Note 2: The SDRAM cannot be accessed while it is in the Self Refresh state.

Note 3: To execute the HALT instruction after the Self Refresh Entry command, insert at least 10 bytes of NOP or other instructions between the instruction to set SDCMM<SCMM2:0> to "101" and the HALT instruction.

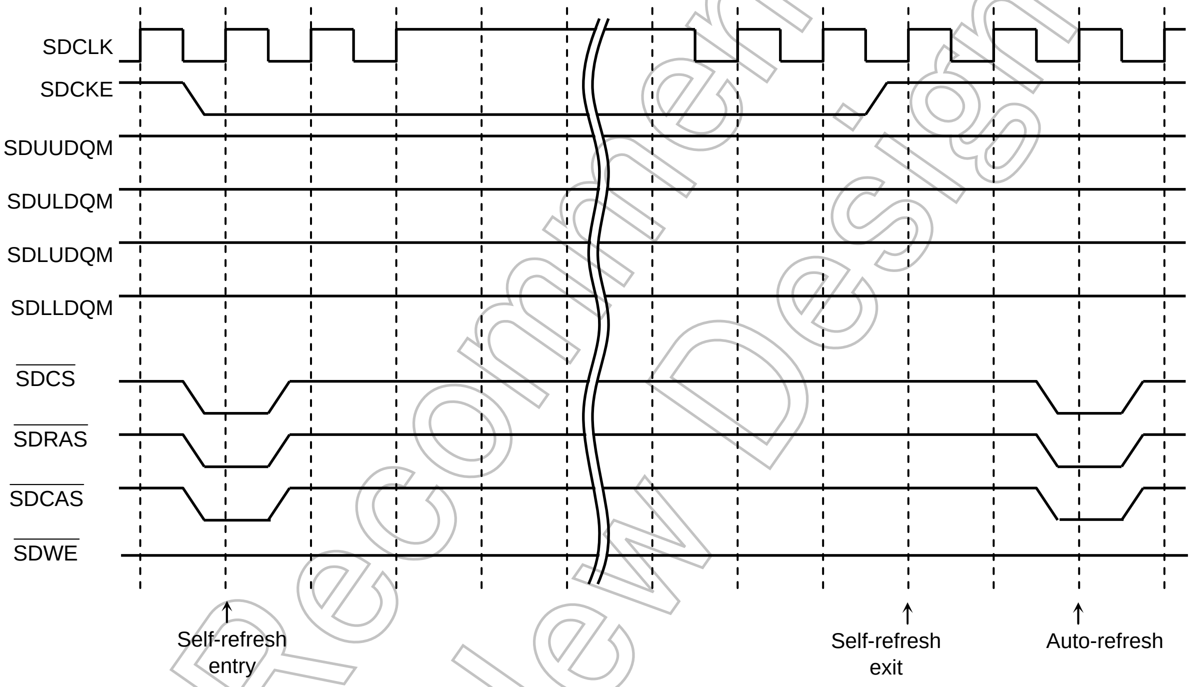


Figure 3.16.5 Timing of Self-Refresh Cycle

(4) SDRAM initialize

This LSI can generate the following SDRAM initialize routine after introduction of power supply to SDRAM. The command is shown in Figure 3.16.6.

1. Precharge all commnad
2. Eight Auto Refresh commands
3. Mode Register set command

The above commands are issued by setting SDCMM<SCMM2:0> to "001".

While these commands are issued, the CPU operation (an instruction fetch, command execution) is halted.

Before executing the initialization sequence, appropriate port settings must be made to enable the SDRAM control signals and address signals (A0 to A15).

After the initialization sequence is completed, SDCMM<SCMM2:0> is automatically cleared to "000".

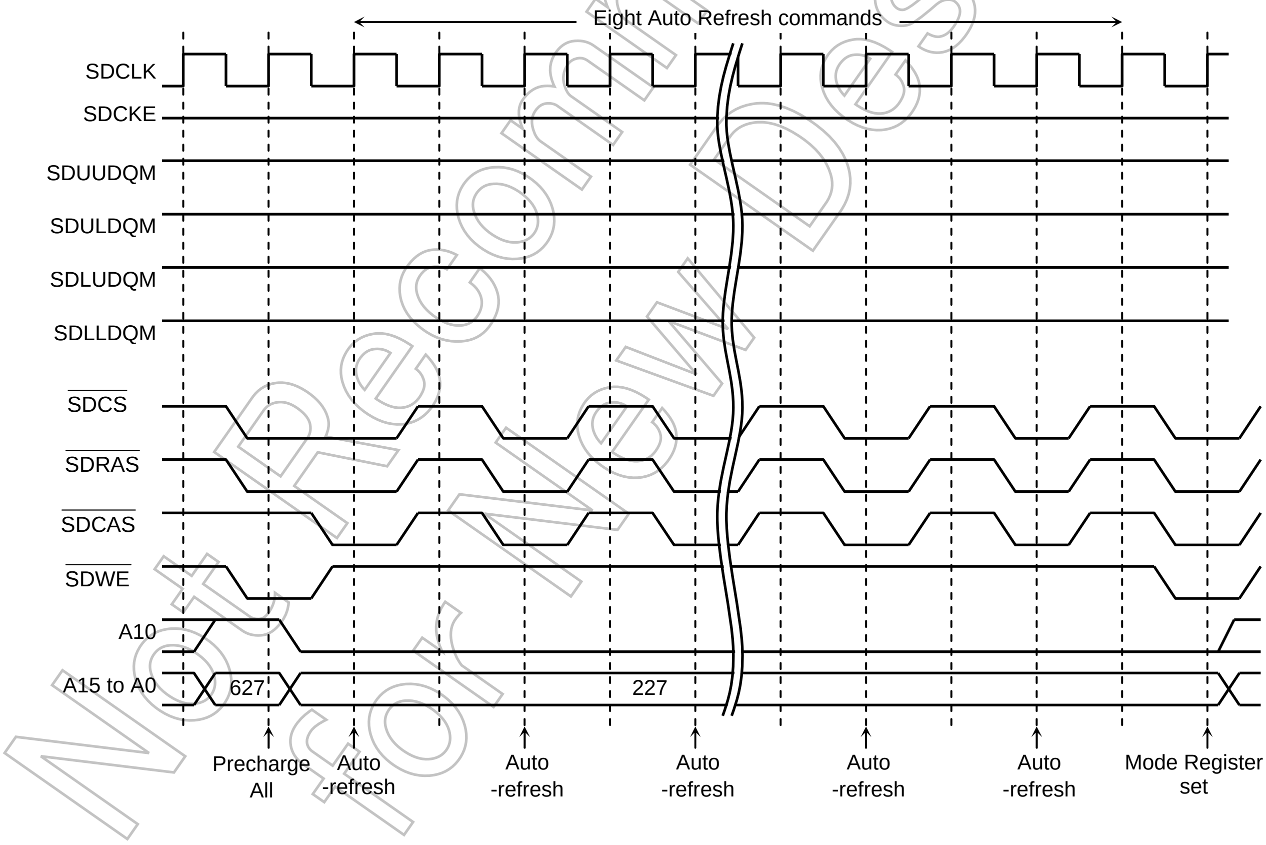


Figure 3.16.6 Timing of Initialization command

(5) Connection example

Figure 3.14.7-Figure 3.14.10 shows an example of connections between the TMP92CH21 and SDRAM

Table 3.16.3 Connection with SDRAM

TMP92CH21 Pin Name	SDRAM Pin Name												
	Data Bus Width: 16 Bits					Data Bus Width 32 Bits							
	16 M	64 M	128 M	256 M	512 M	16 M × 16 bits × 2		64 M × 16 bits × 2		128 M × 16 bits × 2		64 M × 32 bits	128 M × 32 bits
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	–	–	–	–	–	–	–	–	–	–	–	–	–
$\overline{\text{SDCS}}$	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
$\overline{\text{SDRAS}}$	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS	RAS
$\overline{\text{SDCAS}}$	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS	CAS
$\overline{\text{SDWE}}$	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

▨ : Command address pin of SDRAM

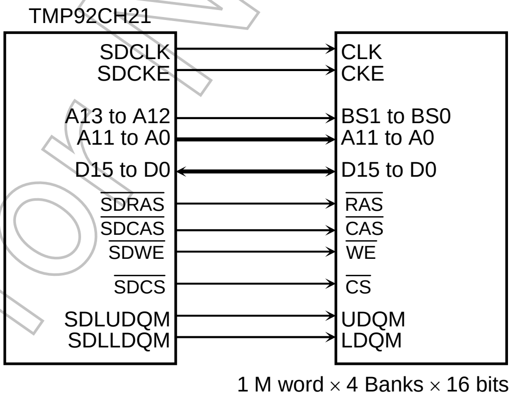


Figure 3.16.7 Connection with SDRAM (4 M word × 16 bits)

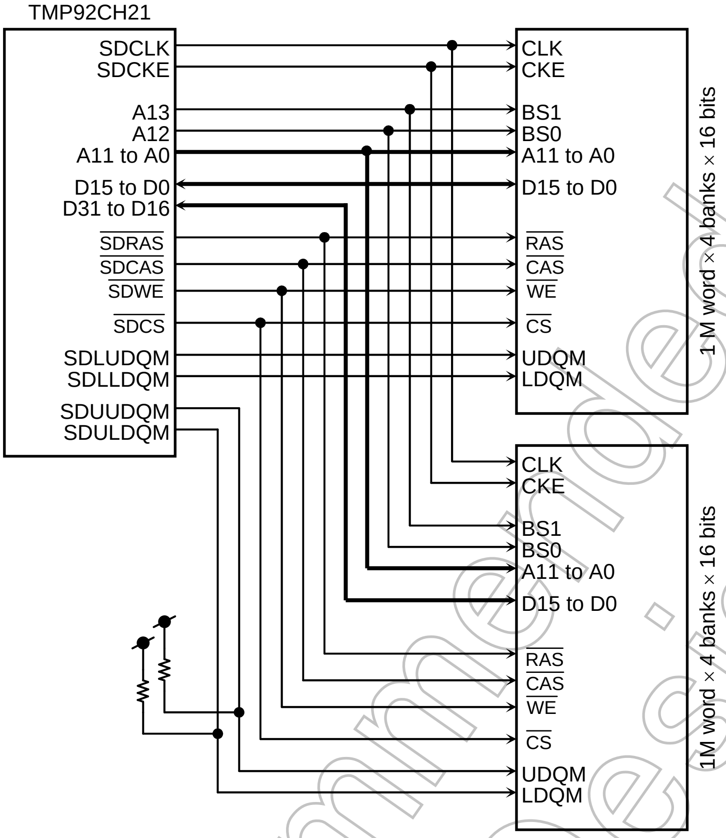


Figure 3.16.8 Connection with SDRAM (1 M word × 16 bits × 2)

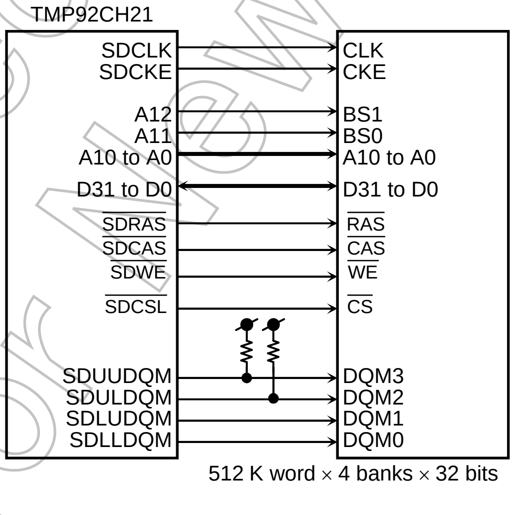


Figure 3.16.9 Connection with SDRAM (512 K word × 32 bits)

3.16.3 Limitations arising when using SDRAM

Take care to note the following points when using SDRAMC.

1. WAIT access

When using SDRAM, some limitation is added when accessing memory other than SDRAM. In WAIT-pin input setting of the Memory Controller, if the setting time is inserted as an external WAIT, set a time less than the Auto-Refresh cycle × 14 (Auto-Refresh function controlled by SDRAM controller).

2. Execution of SDRAM command before HALT instruction (SR (Self refresh)-Entry, Initialize, Mode-set)

When a SDRAM controller command (SR-Entry, Initialize and Mode-set) is issued, several states are required for execution time after the SDCMM register is set.

Therefore, when a HALT instruction is executed after the SDRAM command, please insert a NOP of more than 10 bytes or 10 other instructions before executing the HALT instruction.

3. AR (Auto-Refresh) interval time

When using SDRAM, set the system clock frequency to satisfy the minimum operation frequency for the SDRAM and minimum refresh cycle.

In a system in which SDRAM is used and the clock is geared up and down,exercise care in AR cycle for SDRAM.

When AR cycle is changed, set to disable by writing "0" to SDRCR<SRC>.

The AR cycle may also not correspond to the SDRAM A.C specification when stopping Auto-Refresh. Therefore, set Auto-Refresh cycle after adding 10 states to distibuted Auto-Refresh cycle.

(Example of calculation)
Condition:
f_{SYS}=12MHz,
SDRAM specification of distributed Auto-Refresh interval time =4096times/64ms

64ms/ 4096times = 15.625us/1time = 187.5state/1time
187.5 – 10 = 177.5state/less than 1 time is needed → 156 state is needed

4. Self-Refresh ENTRY method

In order to prevent a conflict between a Self-Refresh ENTRY command and an Auto-Refresh, please stop Auto-Refresh once.

A) Disable Auto Refresh before writing Self Refresh ENTRY command.
B) Enable Auto Refresh after writing Self Refresh ENTRY command.

Because the above instruction should be executed continuously, a 16-bit instruction must be used as below.

(Example of recommended settings)

```
*DI
LDW        (SDRCR),0000010100000010B        ;    Disable AR → SR-ENTRY
LD          (SDRCR),0000---1B               ;    Enable AR
```

Note : * When using SDRAM as a stack pointer, it is necessary to disable SDRAM access by, for example, a "DI" instruction.

5. Note when changing access mode

If changing access mode from “full page read” to “1 word read”, execute the following program. This program must not be executed on the SDRAM.

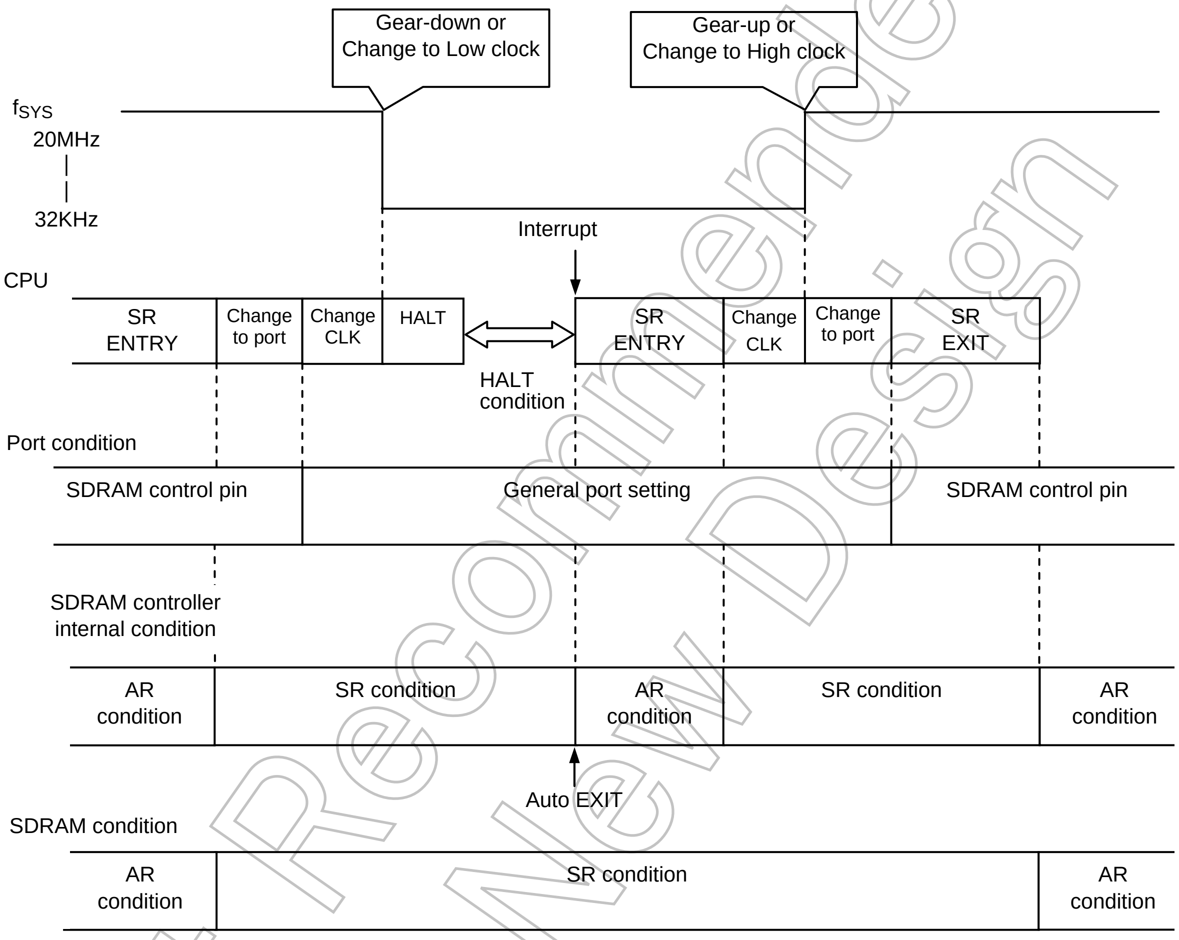
```
di                                                ; Interrupt Disable (Added)
ld        a,(optional external memory             ; Dummy read instruction (Added)
          address)
ld        (sdacr1),00001101b                      ; Change to “1-word read”
ld        (sdcmm),0x04                            ; Execute MRS (mode register setting)
ei                                                ; Interrupt enable (Added)
```

6. “Auto Exit” problem when exiting from SDRAM Self-Refresh Mode

The SDRAM specification may not be satisfied when using the Self-Refresh function together with CPU stand-by function or changing clock,. because when the CPU releases HALT mode, the Self-Refresh Auto Exit function automatically operates.

The following figure shows an example of how to avoid this problem using S/W.

(Control Outline)



*The target ports to change are SDCKE pin and $\overline{\text{SDCS}}$ pin.

*The method of Self-refresh Entry includes the condition 4).

* SR : Self-refresh , AR : Auto-refresh

```
; ********Sample program *********
LOOP1:
        LDB     A, (SDCMM)                      ;   Check the command register clear
        ANDB    A, 00000111B                    ;
        J       NZ, LOOP1                       ;


        LD      (SDRCR), 0000010100000010B      ;   AR stop → SR-ENTRY
        LD      (SDRCR), 0000---1B              ;   AR operation

        NOP×10                                  ;   Wait for execution of self-refresh entry
                                                ;
        RES     7, (PJ)                         ;   PJ7 (SDCKE)=Low
        LD      (PJFC), 0-------B               ;   PJ7=PORT
        SET     1, (P8)                         ;   P81 ( SDCS )
        LD      (P8FC), ------0-B               ;
        LD      (P8FC2), ------0-B              ;   P81 = PORT
        LD      (SYSCR1), 00001---B             ;   fs
        HALT
        NOP                                     ;   Self-refresh Exit (Internal signal only)

LOOP2:
        LDB     A, (SDCMM)                      ;   Check the command  register clear
        ANDB    A, 00000111B                    ;
        J       NZ, LOOP2                       ;

       LDW     (SDRCR), 0000010100000010B       ;   AR stop → SR-ENTRY Enable auto-refresh
        LD      (SDRCR), 0000----1B
                                                ;

        NOP × 10                                ;   Wait for execution of self-refresh entry

        LD      (SYSCR1), 00000---B             ;   fc
        LD      (PJFC), 1-------B               ;   PJ7 = SDCKE
        LD      (P8FC2), ------1-B              ;   P81 = SDCS
        LD      (SDCMM), 00000110B              ;   Self-refresh Exit (command)
```

3.17 NAND-Flash Controller

3.17.1 Characteristics

The NAND-Flash controller (NDFC) is provided with dedicated pins for connecting with NAND-Flash memory. The NDFC also has an ECC calculation function for error correction.

Although the NDFC has two channels (channel 0, channel 1), all pins except for Chip Enable are shared between the two channels. Only the operation of channel 0 is explained here.

The NDFC has the following features:

1) Controlled NAND-Flash interface by setting registers.

2) ECC calculating circuits. (for SCL-type)

Note 1: The $\overline{\text{WP}}$ (Write Protect) pin of NAND Flash is not supported. If this function is needed, prepare it on an external circuit.

Note 2: The two channels cannot be accessed simultaneously. It is necessary to switch between the two channels.

3.17.2 Block Diagram

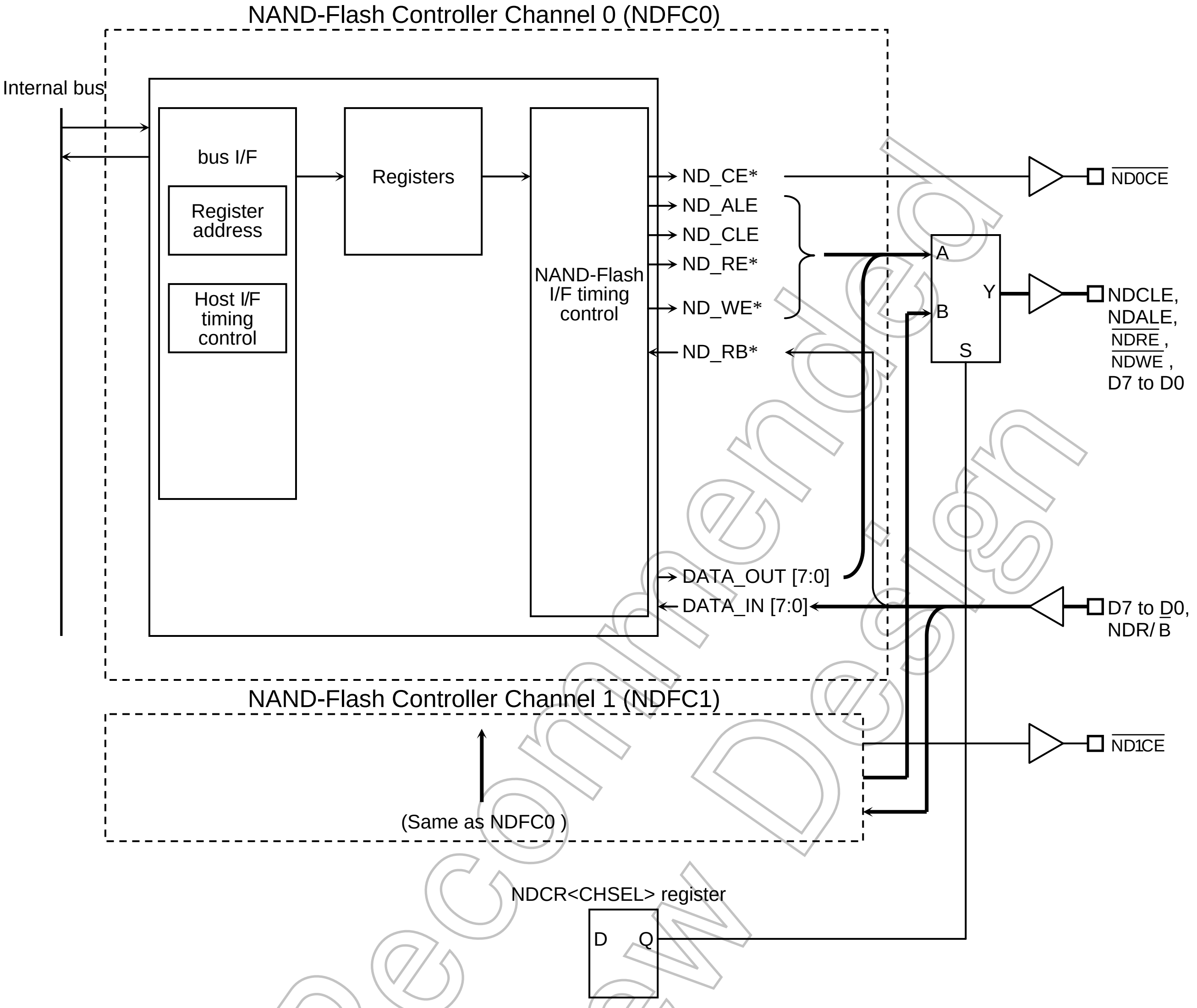


Figure 3.17.1 NAND-Flash Controller Block Diagram

3.17.3 Operation Description

3.17.3.1 Accessing NAND-Flash Memory

The NDFC accesses data on NAND Flash memory indirectly through its internal registers. It also contains the ECC calculating circuits. Please see 3.17.3.2 for details of the ECC. This section explains the operations for accessing the NAND Flash.

Basically, set the command in ND0FMCR and then read or write to ND0FDTR. The read cycle for ND0FDTR is completed after the external read cycle for the NAND-Flash is finished. Likewise, the write cycle for ND0FDTR is completed after the external write cycle for the NAND-Flash is finished.

1) Initialize

The initialize sequence is as follows.

(1) ND0FSPR: Set the low pulse width.

(2) ND0FIMR: Set 0x81 if interrupt is required.
(Release interrupt mask)

2) Write

The write sequence is as follows.

(1) ND0FMCR: Set 0x7C for ECC data reset.

(2) Write 512 bytes

ND0FMCR: Set 0x9D for NDCLE signal enable and command mode.

ND0FDTR: Set 0x80 for the serial data input command.

ND0FMCR: Set 0x9E for NDALE signal enable and address mode.

ND0FDTR: Write address. Set A [7:0], A [16:9], and A [24:17]. If it is required, set A [25].

ND0FMCR: Set 0xBC for the data mode.

ND0FDTR: Write 512 bytes data.

(3) Read ECC data

ND0FMCR: Set 0xDC for 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) Write 16-byte redundant data

ND0FMCR: Set 0x9C for the data mode without ECC calculation.

ND0FDTR: Write 16-byte 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

(5) Run page program

ND0FMCR: Set 0x9D for NDCLE signal enable and command mode.

ND0FDTR: Set 0x10 for the page program command.

ND0FMCR: Set 0x1C for NDALE signal disable.

Wait several states (e.g., "NOP" × 10)

ND0FSR: Check BUSY flag. If it is 0, go to the next.
If it is 1, wait until it becomes 0.

(6) Read status

ND0FMCR: Set 0x1D for NDCLE signal and command mode.

ND0FDTR: Set 0x70 for Status read command.

ND0FMCR: Set 0x1C for NDCLE signal disable.

ND0FDTR: Read the Status data from the NAND-Flash.

(7) Repeat 1 to 6 for all other pages if required.

3) Read

The read sequence is as follows.

(1) ND0FMCR: Set 0x7C for ECC data reset.

(2) Read 512 bytes

ND0FMCR: Set 0x1D for NDCLE signal enable and command mode.

ND0FDTR: Set 0x00 for the read command.

ND0FMCR: Set 0x1E for NDALE signal enable and address mode.

ND0FDTR: Set A [7:0], A [16:9], and A [24:17]. If it is required, set A [25].

ND0FMCR: Set 0x1C for NDALE signal disable.

Wait several states (e.g., "NOP" × 10)

ND0FSR: Check BUSY flag. If it is 0, go to the next.
If it is 1, wait until it becomes 0.

ND0FMCR: Set 0x3C for the data mode with ECC calculation.

ND0FDTR: Read 512-byte data.

ND0FMCR: Set 0x1C for the data mode without ECC calculation.

ND0FDTR: Read 16-byte redundant data.

(3) Read ECC data

ND0FMCR: Set 0x5C for the ECC data read mode.

NDECCRD: Read 6-byte 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) Software routine:

Compare ECC data and redundant data, run the error routine if error is generated.

(5) Read other pages

ND0FMCR: Set 0x1C.

ND0FSR: Check BUSY flag. If it is 0, go to the next.
If it is 1, wait until it becomes 0.

4) ID read

The ID read sequence is as follows.

(1)	ND0FMCR:	Set 0x1D for NDCLE signal enable and command mode.
(2)	ND0FDTR:	Set 0x90 for the ID Read command.
(3)	ND0FMCR:	Set 0x1E for NDALE signal enable and the address mode.
(4)	ND0FDTR:	Set 0x00.
(5)	ND0FMCR:	Set 0x1C for the data mode without ECC calculation.
(6)	ND0FDTR:	Read Maker code.
(7)	ND0FDTR:	Read Device code.

3.17.3.2 ECC Control

The NDFC contains the ECC calculating circuits. The circuits are controlled by ND0FMCR. This circuit executes ECC data calculation. However, ECC comparison and error correction is not executed. This must be executed using software.

The calculated ECC data can be read from the NDECCRD register when ND0FMCR is 0xD0 (write mode) or 0x50 (read mode). This is 6-byte data, and six NDECCRD read operations are required. The order of the data is as follows.

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 3.17.1 NAND-Flash Control Registers for Channel 0

Address	Register	Register Name
1D00H (1D00H to 1EFFH)	ND0FDTR	NAND-Flash data transfer register
1CB0H (1CB0H to 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 3.17.2 NAND-Flash Control Registers for Channel 1

Address	Register	Register Name
1D00H (1D00H to 1EFFH)	ND1FDTR	NAND-Flash data transfer register
1CB0H (1CB0H to 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 3.17.3 NAND-Flash Control Registers

Address	Register	Register Name
01C0H	NDCR	NAND-Flash control register

3.17.4.1 NAND-Flash Data Transfer Register (ND0FDTR and ND1FDTR)

7	0	
DATA		
R/W		: Type
–		: Default

Bit (s)	Mnemonic	Field Name	Description
7:0	DATA	DATA	NAND-Flash data. Read: Read the data that was read from the NAND-Flash. Write: Write data to the NAND-Flash.

Note 1: This register has a 512-address window from 1D00H to 1EFFH since a NAND-Flash page size is either 256 or 512 bytes.

When the CPU reads from or writes to the NAND-Flash , and if the block transfer instruction ("LDIR" instruction) is used, the following restriction applies to the 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)

Example 1) In case of 512-byte read

```
ld      bc, 512                ;  512 bytes
ld      xix, 2000H             ;  dst = 2000H
ld      xiy, 1D00H             ;  src = (1F00H − 512) = 1D00H
ldir    (xix + ), (xiy + )     ;  Block transfer instruction
```

Example 2) In case of 16-byte read

```
ld      bc, 16                 ;  16 bytes
ld      xix, 2000H             ;  dst = 2000H
ld      xiy, 1EF0H             ;  src = (1F00H − 16) = 1EF0H
ldir    (xix + ), (xiy + )     ;  Block transfer instruction
```

Note 2: Both ND0FDTR and ND1FDTR are assigned to the same address. The NDCR<CHSEL> register determines which channel is accessed.

Figure 3.17.2 NAND-Flash Data Transfer Register (ND0FDTR and ND1FDTR)

3.17.4.2 NAND-Flash ECC-code Read Register (ND0ECCRD and ND1ECCRD)

7							0	
ECC-code								
R								: Type
–								: Default

Bit (s)	Mnemonic	Field Name	Description
7:0	ECC-code	ECC-code	Read calculated ECC data.

Note 1: Both ND0ECCRD and ND1ECCRD are assigned to the same address. The NDCR<CHSEL> register determines which channel is accessed.

Figure 3.17.3 NAND-Flash ECC-code Read Register (ND0ECCRD and ND1ECCRD)

3.17.4.3 NAND-Flash Mode Control Register (ND0FMCR and ND1FMCR)

	7	6	5	4	3	5	1	0
	WE	ECC1	ECC0	CE	PCNT1	PCNT0	ALE	CLE
: Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
: Default	0	0	0	0	0	0	0	0

Bits	Mnemonic	Field Name	Description
7	WE	Write enable	Write enable (Default: 0) This bit enables the data write operation. When writing the data to the NAND-Flash, set this bit to “1”. When writing command or address, this bit need not be set to “1”. 0: Disable write operation 1: Enable write operation
6	ECC1	ECC control	ECC control (Default: 00) Control the ECC calculating circuits with <CE> (bit4) register. 11 (at<CE> = X): Reset ECC circuits 00 (at<CE> = 1): ECC circuits are disabled. 01 (at<CE> = 1): ECC circuits are enabled. 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. Set “1” to this bit when accessing the NAND-Flash. 0: Disable ($\overline{\text{NDCE}}$ is High.) 1: Enable ($\overline{\text{NDCE}}$ is Low.)
3	PCNT1	Power control	Power control (Default: 00) Always write “11”
2	PCNT0		
1	ALE	Address latch enable	Address latch enable (Default: 0) This bit specifies the value of the NDALE signal. 0: Low 1: High
0	CLE	Command latch enable	Command latch enable (Default: 0) This bit specifies the value of the NDCLE signal. 0: Low 1: High

Figure 3.17.4 NAND-Flash Mode Control Register (ND0FMCR and ND1FMCR)

3.17.4.4 NAND-Flash Status Register (ND0FSR and ND1FSR)

	7	6						0
	BUSY							
: Type	R							
: Default	–							

Bits	Mnemonic	Field Name	Description
7	BUSY	BUSY	BUSY (Default: Undefined) This bit shows the status of the NAND-Flash. 0: Ready 1: Busy
6:0	–	–	Reserved

Note: A noise-filter for some states is built into the NDFC, so when the NDR/$\overline{B}$ pin changes, a <BUSY> flag is not renewed at the same time. Therefore, insert several delays (e.g., "NOP" instruction × 10) using software before starting this flag check.

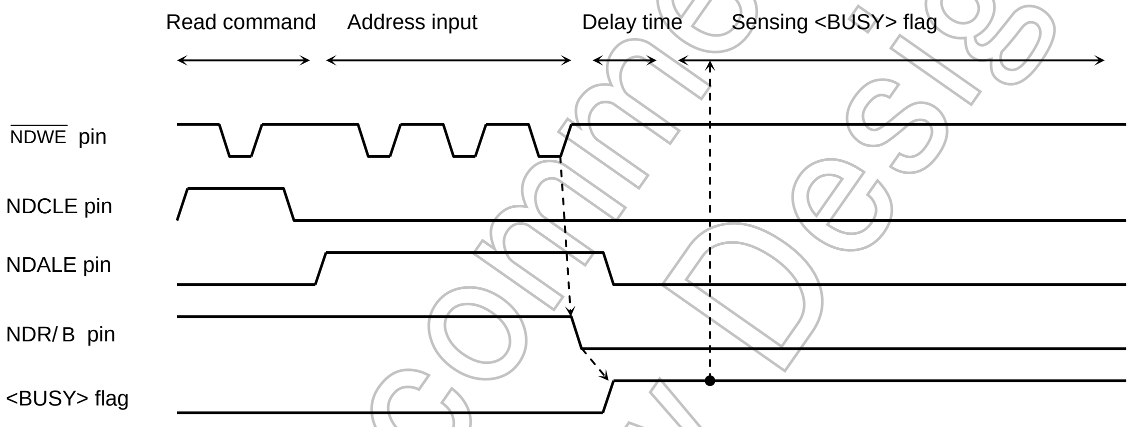


Figure 3.17.5 NAND-Flash Status Register (ND0FSR and ND1FSR)

3.17.4.5 NAND-Flash Interrupt Status Register (ND0FISR and ND1FISR)

	7	6	5	4	3	2	1	0
								RDY
: Type								
: Default								

Bits	Mnemonic	Field Name	Description
7:1	–	–	Reserved
0	RDY	Ready	Ready (Default: 0) When NDR/$\overline{B}$ signal changes from low (BUSY) to High (READY) and NDFIMR<MRDY> is “1”, this bit is set to “1”. By writing “1”, this bit is cleared to 0. Read: 0: None 1: Change NDR/$\overline{B}$ signal from BUSY to READY. Write: 0: No change 1: Clear to “0”

Figure 3.17.6 NAND-Flash Interrupt Status Register (ND0FISR and ND1FISR)

3.17.4.6 NAND-Flash Interrupt Mask Register (ND0FIMR and ND1FIMR)

	7	6		4	3	2	1	0
	INTEN	0						MRDY
: Type	R/W							R/W
: Default	0							0

Bits	Mnemonic	Field Name	Description
7	INTEN	Interrupt enable	Interrupt enable (Default: 0) When <INTEN> and <MRDY> are set “1” and NDFISR<RDY> becomes “1”, INTNDFC occurs. 0: Disable 1: Enable
6:1	–	–	Reserved
0	MRDY	Mask RDY interrupt	Mask RDY interrupt (Default: 0) This bit masks the NDFISR<RDY>. If <MRDY> is “1” and NDR/$\overline{B}$ signal changes from Low to High, NDFISR<RDY> is set to “1”. 0: Disable to set NDFISR<RDY> 1: Enable to set NDFISR<RDY>

Figure 3.17.7 NAND-Flash Interrupt Mask Register (ND0FIMR and ND1FIMR)

3.17.4.7 NAND-Flash Strobe Pulse Width Register (ND0FSPR and ND1FSPR)

7	6	5	4	3	2	1	0	
				SPW				
				R/W				: Type
				0000				: Default

Bits	Mnemonic	Field Name	Description
7:4	–	–	Reserved
3:0	SPW	Strobe pulse width	Strobe pulse width (Default: 0000) These bits set the Low pulse width of the $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ signals. The Low pulse width is ((value set to SPW) +1)× f_{SYS} clock

Figure 3.17.8 NAND-Flash Strobe Pulse Width Register (ND0FSPR and ND1FSPR)

3.17.4.8 NAND-Flash Reset Register (ND0FRSTR and ND1FRSTR)

7	6	5	4	3	2	1	0	
							RST	
							R/W	: Type
							0	: Default

Bits	Mnemonic	Field Name	Description
7:1	–	–	Reserved
0	RST	Reset	Reset (Default: 0) By setting this bit, reset the NDFC (except NDCR<CHSEL> register). By reset, this bit is automatically cleared to "0". 0: Don't care 1: Reset

Note: After writing <RST> register, several waits are required (about 10 states) before accessing the NDFC.

Figure 3.17.9 NAND-Flash Reset Register (ND0FRSTR and ND1FRSTR)

3.17.4.9 NAND-Flash Control Register (NDCR)

		7	6	5	4	3	2	1	0
NDCR (01C0H)	Bit symbol	CHSEL							
	Read/Write	R/W							
	Reset State	0							
	Function	0: Channel 0 1: Channel 1							

3.17.5 Timing Diagrams

3.17.5.1 Command and Address Cycle

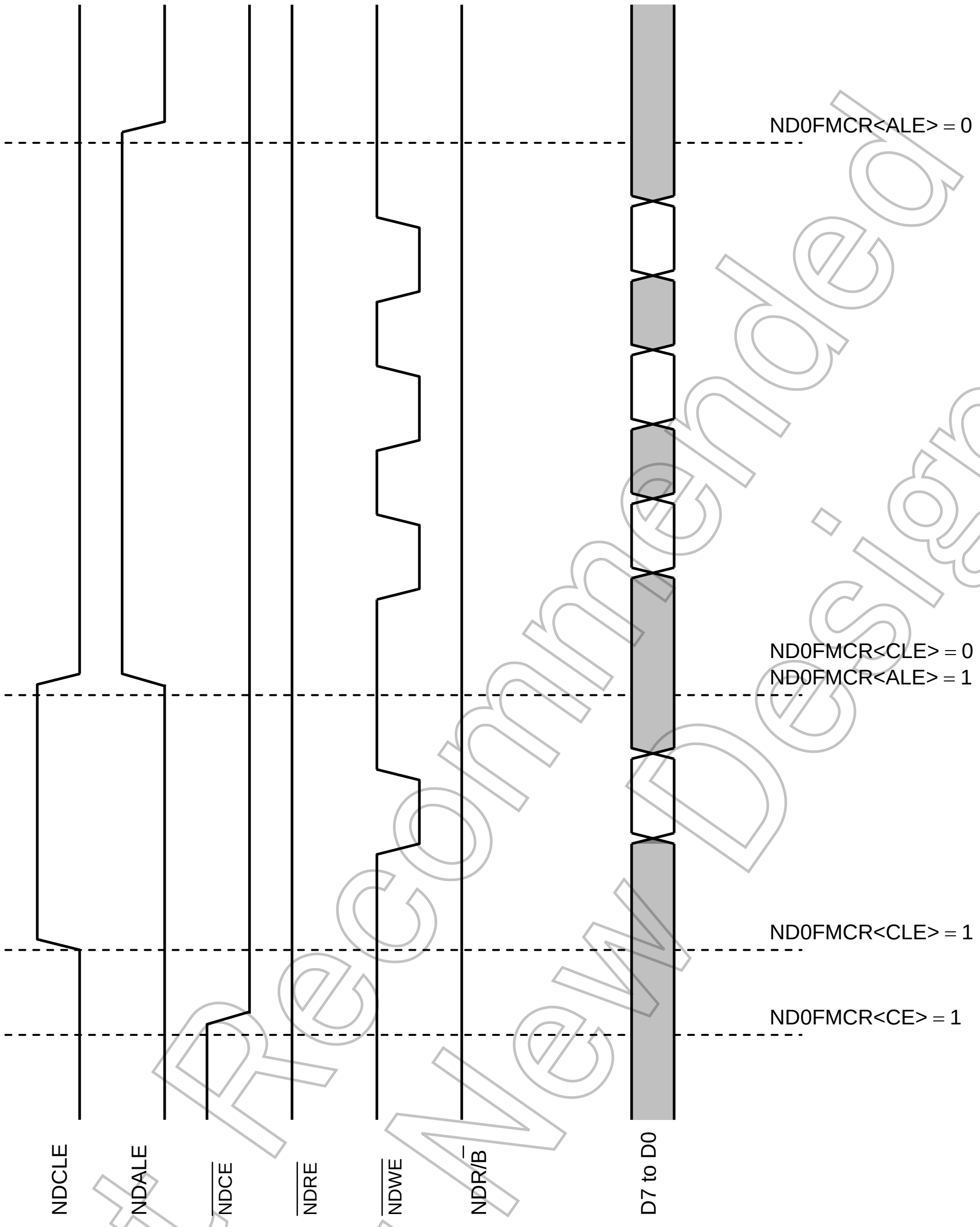


Figure 3.17.10 Command and Address Cycle

3.17.5.2 Data Read Cycle

Figure 3.17.11 shows a timing chart example for a Data Read cycle from the NAND-Flash at ND0FSPR = 02H.

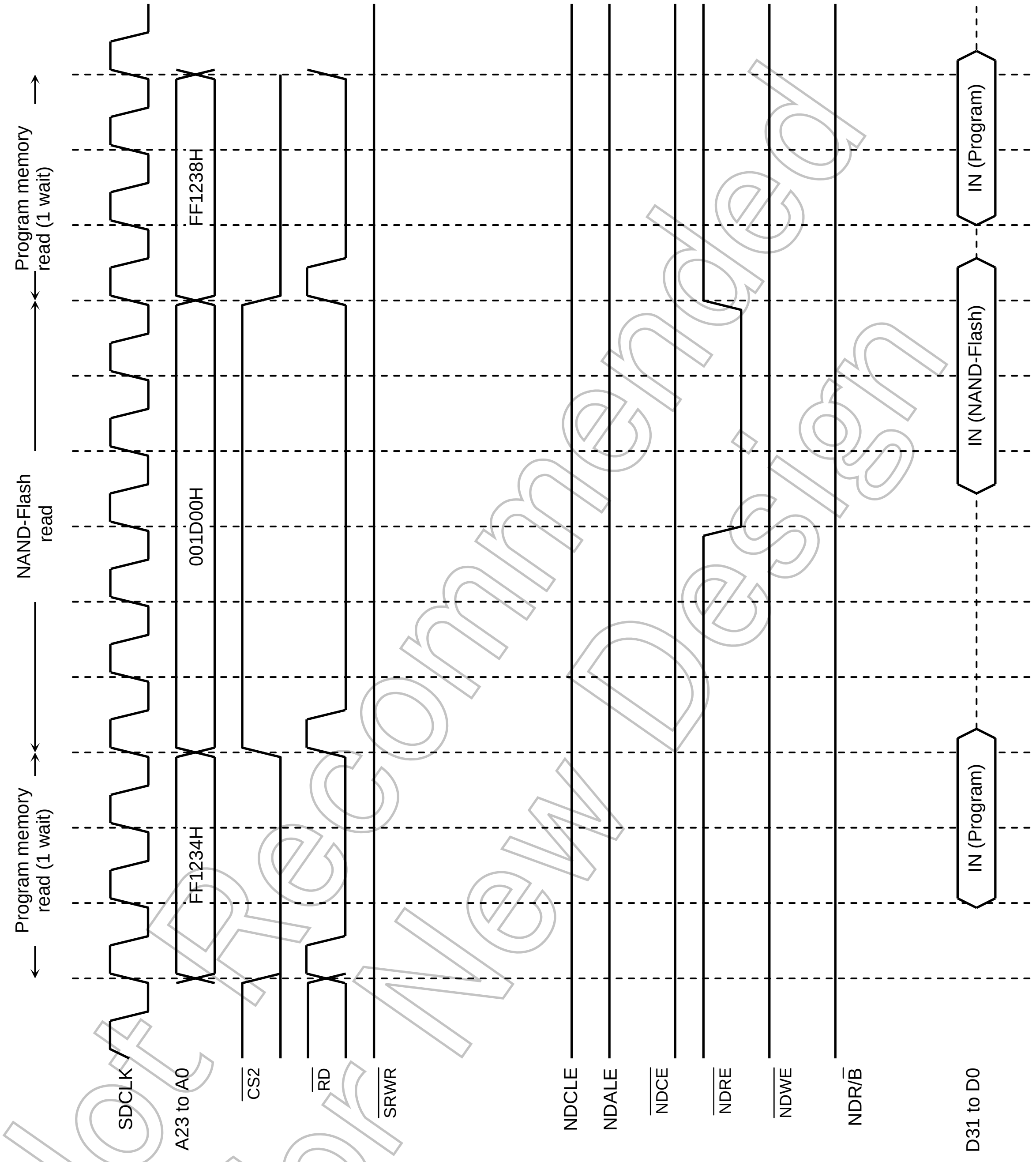


Figure 3.17.11 Data Read Cycle Example (ND0FSPR = 02H)

3.17.5.3 Data Write Cycle

Figure 3.17.12 shows a timing chart example for a Data Write cycle to the NAND-Flash at ND0FSPR = 02H.

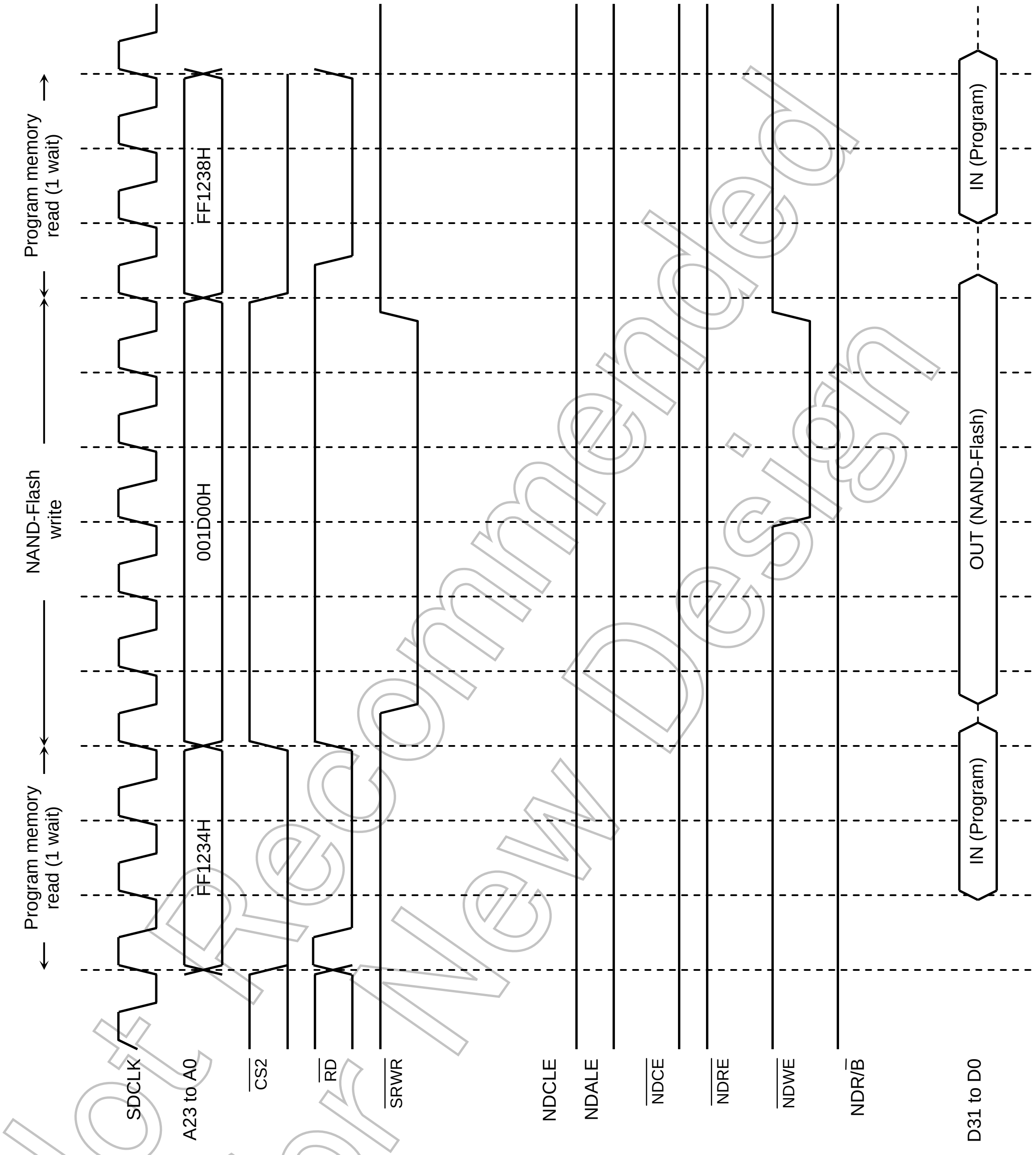
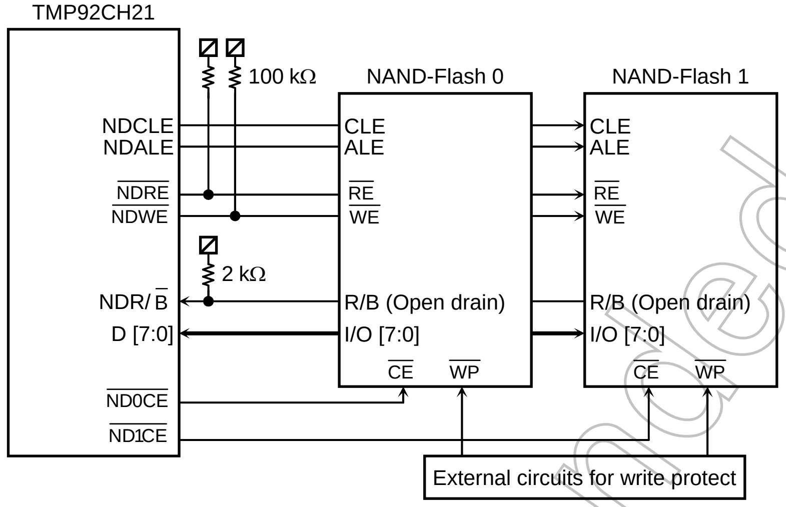


Figure 3.17.12 Data Write Cycle (ND0FSPR = 02H)

3.17.6 Example of NAND-Flash Use



Note 1: By reset, both NDRE and NDWE pins become input ports (Port 71 and Port 72) And so require pull-up resistors.

Note 2: Use the NAND-Flash memory and board capacitance to set the correct value for the NDR/B pin pull-up resistor . 2 kΩ is a typical value.

Note 3: The NAND-Flash WP (write protect) pin is not supported by the TMP92CH21. It must be provided by an external circuit if required.

Figure 3.17.13 Example of NAND-Flash Connection

3.18 16-Bit Timer/Event Counters (TMRB0)

The TMP92CH21 incorporates one multifunctional 16-bit timer/event counter (TMRB0) which has the following operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable pulse generation (PPG) mode

The 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 register, two comparators, a capture input controller, a timer flip-flop and a control circuit.

The timer/event counter is controlled by an 11-byte control SFR.

This chapter includes the following sections:

3.18.1 Block Diagrams

3.18.2 Operation of Each Block

3.18.3 SFRs

3.18.4 Operation in Each Mode
 (1) 16-bit interval timer mode
 (2) 16-bit programmable pulse generation (PPG) output mode

Table 3.18.1 Pins and SFR of TMRB0

Channel Spec.		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

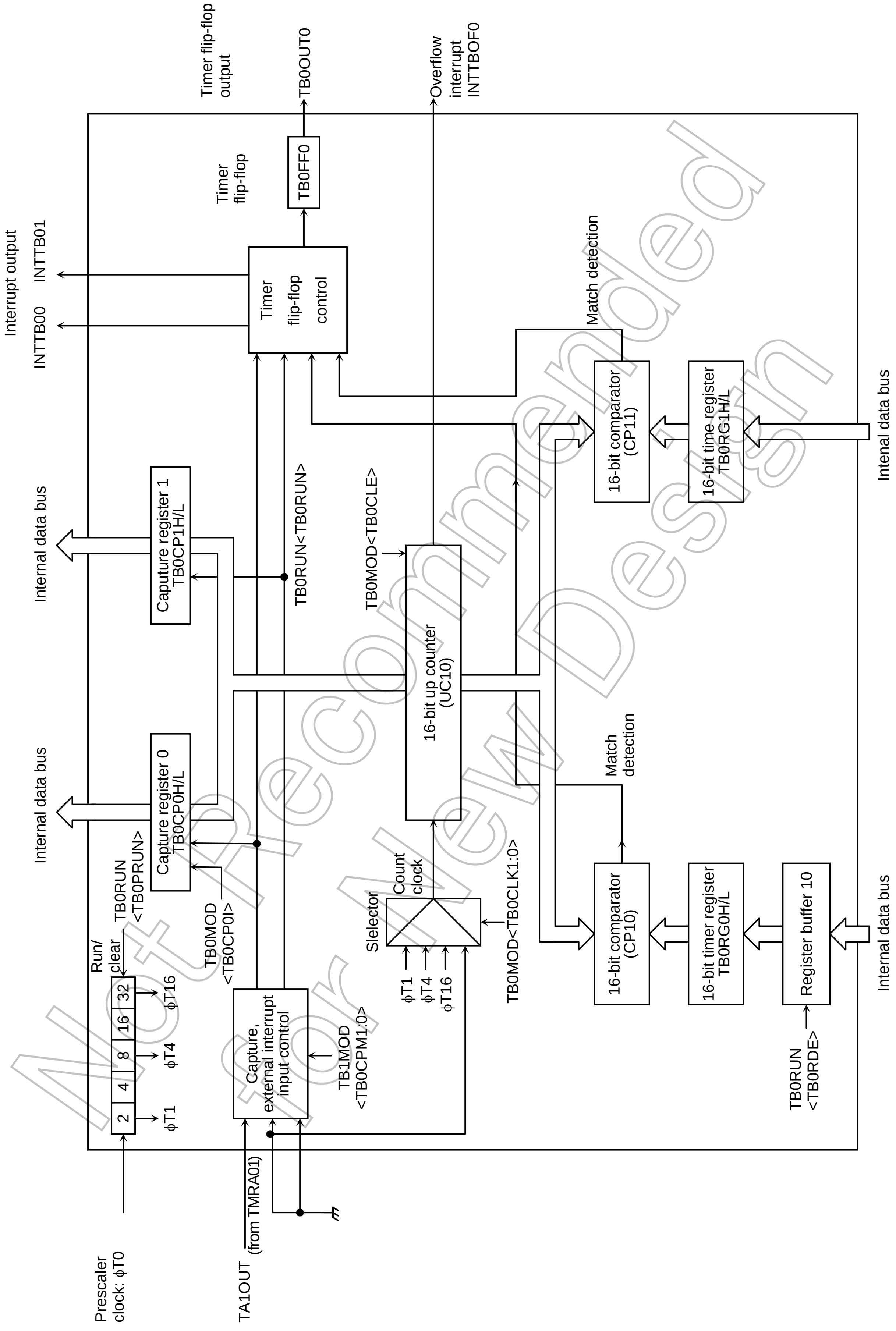


Figure 3.18.1 Block Diagram of TMRB0

3.18.2 Operation of Each Block

(1) Prescaler

The 5-bit prescaler generates the source clock for timer 0. The prescaler clock (φT0) is a divided clock (divided by 8) from the f_{FPH}.

This prescaler can be started or stopped using TB0RUN<TB0PRUN>. Counting starts when <TB0PRUN> is set to 1; the prescaler is cleared to 0 and stops operation when <TB0PRUN> is cleared to 0.

Table 3.18.2 Prescaler Clock Resolution

System clock selection SYSCR1 <SYSCK>	Clock gear selection SYSCR1 <GEAR2:0>	–	Timer counter input clock TMRB prescaler TB0MOD<TB0CLK1:0>		
			φT1(1/2)	φT4(1/8)	φT16(1/32)
1 (fs)	–	1/8	fs/16	fs/64	fs/256
0 (fc)	000 (1/1)		fc/16	fc/64	fc/256
	001 (1/2)		fc/32	fc/128	fc/512
	010 (1/4)		fc/64	fc/256	fc/1024
	011 (1/8)		fc/128	fc/512	fc/2048
	100 (1/16)		fc/256	fc/1024	fc/4096

(2) Up counter (UC10)

UC10 is a 16-bit binary counter which counts up pulses input from the clock specified by TB0MOD<TB0CLK1:0>.

Any one of the prescaler internal clocks φT1, φT4 and φT16 can be selected as the input clock. Counting or stopping and clearing of the counter is controlled by TB0RUN<TB0RUN>.

When clearing is enabled, the up counter UC10 will be cleared to 0 each time its value matches the value in the timer register TB0RG1H/L. If clearing is disabled, the counter operates as a free-running counter.

Clearing can be enabled or disabled using TB0MOD<TB0CLE>.

A timer overflow interrupt (INTTBOF0) is generated when UC10 overflow occurs.

(3) Timer registers (TB0RG0H/L and TB0RG1H/L)

These 16-bit registers are used to set the interval time. When the value in the up counter UC10 matches the value set in this timer register, the comparator match detect signal will go active.

Setting data for both Upper and Lower timer registers is always needed. For example, either using a 2-byte data transfer instruction or using a 1-byte data transfer instruction twice for the lower 8 bits and upper 8 bits in order.

The TB0RG0H/L timer register has a double-buffer structure, which is paired with a 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 (UC10) and the timer register TB0RG1H/L match.

After a reset, TB0RG0H/L and TB0RG1H/L are undefined. If the 16-bit timer is to be used after a reset, data should be written to it beforehand.

On a reset <TB0RDE> is initialized to 0, disabling the double buffer. To use the double buffer, write data to the timer register, set <TB0RDE> to 1, then write data to the register buffer as shown below.

TB0RG0H/L and the register buffer both have the same memory addresses (001188H and 001189H) 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:

TMRB0

TB0RG0H/L		TB0RG1H/L	
Upper 8 bits (TB0RG0H)	Lower 8 bits (TB0RG0L)	Upper 8 bits (TB0RG1H)	Lower 8 bits (TB0RG1L)
001189H	001188H	00118BH	00118AH

The timer registers are write-only registers and thus cannot be read.

(4) Capture registers (TB0CP0H/L and TB0CP1H/L)

These 16-bit registers are used to latch the values in the up counters.

All 16 bits of data in the capture registers should be read. For example, using a 2-byte data load instruction or two 1-byte data load instructions. The least significant byte is read first, followed by the most significant byte.

The addresses of the capture registers are as follows:

TMRB0

TB0CP0H/L		TB0CP1H/L	
Upper 8 bits (TB0CPH)	Lower 8 bits (TB0CP0L)	Upper 8 bits (TB0CP1H)	Lower 8 bits (TB0CP1L)
00118DH	00118CH	00118FH	00118EH

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 the up counter UC10 into TB0CP0H/L and TB0CP1H/L.

The value in the up counter can be loaded into a capture register by software. Whenever 0 is programmed to TB0MOD<TB0CP0I>, the current value in the up counter is loaded into capture register TB0CP0H/L. It is necessary to keep the prescaler in run mode (i.e., TB0RUN<TB0PRUN> must be held at a value of 1).

(6) Comparators (CP10 and CP11)

CP10 and CP11 are 16-bit comparators which compare the value in the up counter UC10 with the value set in TB0RG0H/L or TB0RG1H/L respectively, in order to detect a match. If a match is detected, the comparator generates an interrupt (INTTB00 or INTTB01 respectively).

(7) Timer flip-flops (TB0FF0)

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<TB0C1T1, TB0C0T1, TB0E1T1 and TB0E0T1>.

After a reset the value of TB0FF0 is undefined. If “00” is programmed to TB0FFCR <TB0FF0C1:0>, TB0FF0 will be inverted. If “01” is programmed to the capture registers, the value of TB0FF0 will be set to “1”. If “10” is programmed to the capture registers, the value of TB0FF0 will be cleared to “0”.

The values of TB0FF0 can be output via the timer output pin TB0OUT0 (which is shared with PC6). Timer output should be specified using the port B function register.

3.18.3 SFRs

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
Reset State	0	0			0	0		0
Function	Double buffer 0: Disable 1: Enable	Always write "0"			IDLE2 0: Stop 1: Operate	TMRB0 Prescaler		Up counter UC10
						0: Stop and clear 1: Run (Count up)		

Count operation

0	Stop and clear
1	Count

Note: 1, 4 and 5 of TB0RUN are read as undefined values.

Figure 3.18.2 The Registers for TMRB

TMRB0 Mode Register

TB0MOD
(1182H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	–	–	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
Read/Write	R/W		W*	R/W				
Reset State	0	0	1	0	0	0	0	0
Function	Always write “0”		Execute software capture 0: Software capture 1: Undefined	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	Reserved
01	ϕT1
10	ϕT4
11	ϕT16

Up counter clear control

0	Disable
1	Enable clearing on match with TB0RG1H/L

Capture/interrupt timing

00	Disable
01	Reserved
10	Reserved
11	Capture to TB0CP0H/L at rising edge of TA1OUT Capture to TB0CP1H/L at falling edge of TA1OUT

Software capture

0	The value in the up counter is captured to TB0CP0H/L.
1	Undefined

Figure 3.18.3 The Registers for TMRB0

TMRB0 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TB0FFCR (1183H)	Bit symbol	–	–	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
	Read/Write	W*		R/W				W*	
	Reset State	1	1	0	0	0	0	1	1
Read-modify -write instruction is prohibited.	Function	Always write “11”.		TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB0FF0 00: Invert 01: Set 10: Clear 11: Don’t care * Always read as 11.	
				Invert when the UC value is loaded into TB0CP1H/L.	Invert when the UC value is loaded into TB0CP0H/L.	Invert when the UC value matches the value in TB0RG1H/L.	Invert when the UC value matches the value in TB0RG0H/L.		

Timer flip-flop control (TB0FF0)

00	Invert
01	Set to 1
10	Clear to 0
11	Don’t care

Inverted when the UC10 value matches the value in TB0RG0H/L.

0	Disable trigger
1	Enable trigger

Inverted when the UC10 value matches the value in TB0RG1H/L.

0	Disable trigger
1	Enable trigger

Inverted when the UC10 value is loaded into TB0CP0H/L.

0	Disable trigger
1	Enable trigger

Inverted when the UC10 value is loaded into TB0CP1H/L.

0	Disable trigger
1	Enable trigger

Figure 3.18.4 The Registers for TMRB

TMRB0 register

		7	6	5	4	3	2	1	0
TB0RG0L (1188H)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0RG0H (1189H)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0RG1L (118AH)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0RG1H (118BH)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0CP0L (118CH)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0CP0H (118DH)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0CP1L (118EH)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							
TB0CP1H (118FH)	bit Symbol	–							
	Read/Write	W							
	Reset State	Undefined							

Note: All registers are prohibited to execute read-modify-write instruction.

Figure 3.18.5 The Registers for TMRB

3.18.4 Operation in Each Mode

(1) 16-bit interval timer mode

Generating interrupts at fixed intervals.

In this example, the interrupt INTTB01 is set to be generated at fixed intervals. The interval time is set in the timer register TB0RG1H/L.

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	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)			
TB0RG1H/L	←	*	*	*	*	*	*	*	*	Set the interval time (16 bits).
		*	*	*	*	*	*	*	*	
TB0RUN	←	0	0	X	X	–	1	X	1	Start TMRB0.

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 enabled by the match of the up counter UC10 with timer register TB0RG0H/L or TB0RG1H/L and is output to TB0OUT0. In this mode the following conditions must be satisfied.

(Value set in TB0RG0H/L) < (Value set in TB0RG1H/L)

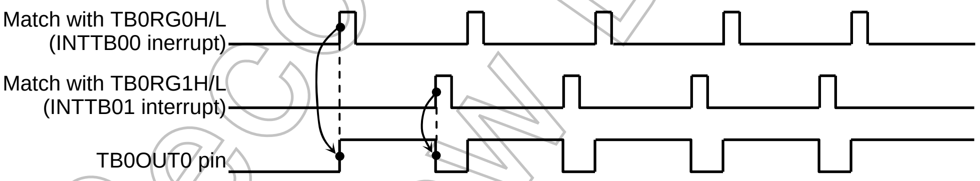


Figure 3.18.6 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0H/L double buffer is enabled in this mode, the value of register buffer 10 will be shifted into TB0RG0H/L at match with TB0RG1H/L. This feature facilitates the handling of low duty waves.

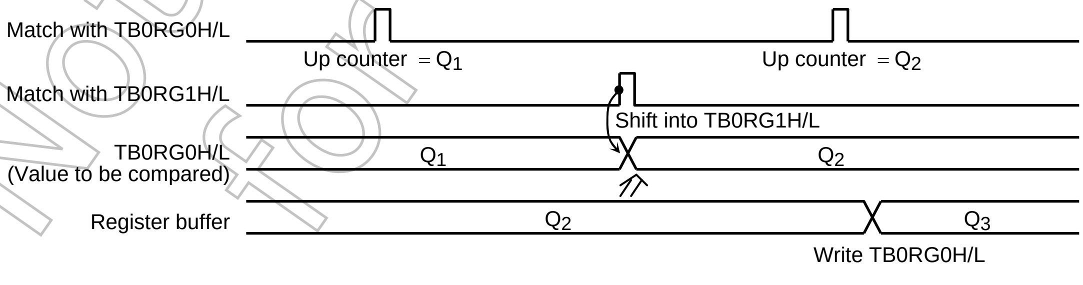


Figure 3.18.7 Operation of Register Buffer

The following block diagram illustrates this mode.

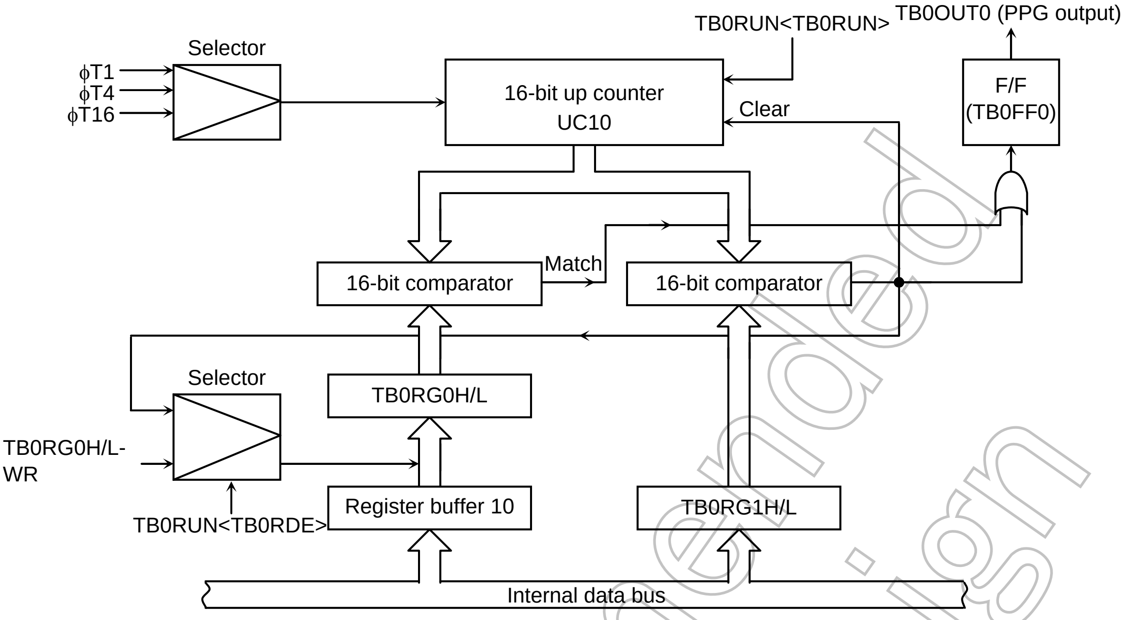


Figure 3.18.8 Block Diagram of 16-Bit Mode

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 TB0RG0H/L double buffer and stop TMRB0.
TB0RG0H/L	←	*	*	*	*	*	*	*	*	Set the duty ratio (16 bits).
		*	*	*	*	*	*	*	*	
TB0RG1H/L	←	*	*	*	*	*	*	*	*	Set the frequency (16 bits).
		*	*	*	*	*	*	*	*	
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0H/L double buffer. (The duty and frequency are changed on an INTTB01 interrupt.)
TB0FFCR	←	1	1	0	0	1	1	1	0	Set the mode to invert TB0FF0 at the match with TB0RG0H/L/TB0RG1H/L. Set TB0FF0 to 0.
TB0MOD	←	0	0	1	0	0	1	*	*	Select the Prescaler output clock as the input clock and disable the capture function.
						(** = 01, 10, 11)				
PCCR	←	–	1	X	X	–	–	–	–	Set PC6 to function as TB0OUT0.
PCFC	←	–	1	X	X	–	–	–	–	
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

3.19 Touch Screen Interface (TSI)

The TMP92CH21 has an interface for a 4-terminal resistor network touch screen.

This interface supports two procedures: an X/Y position measurement and touch detection.

Each procedure is executed by setting the TSI control register (TSICR0 and TSICR1) and using an internal AD converter.

3.19.1 Touch Screen Interface Module Internal/External Connection

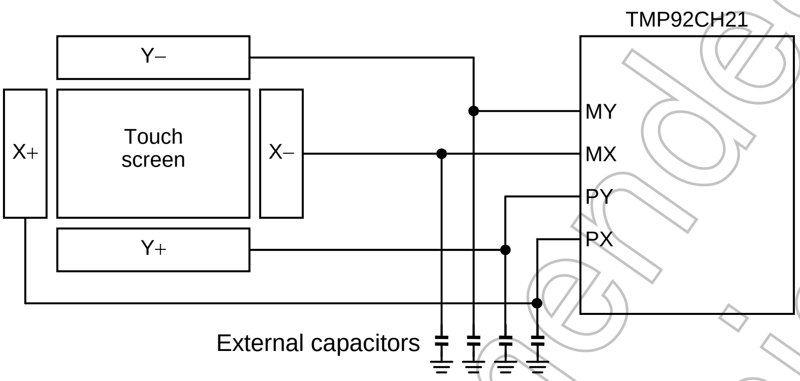


Figure 3.19.1 External Connection of TSI

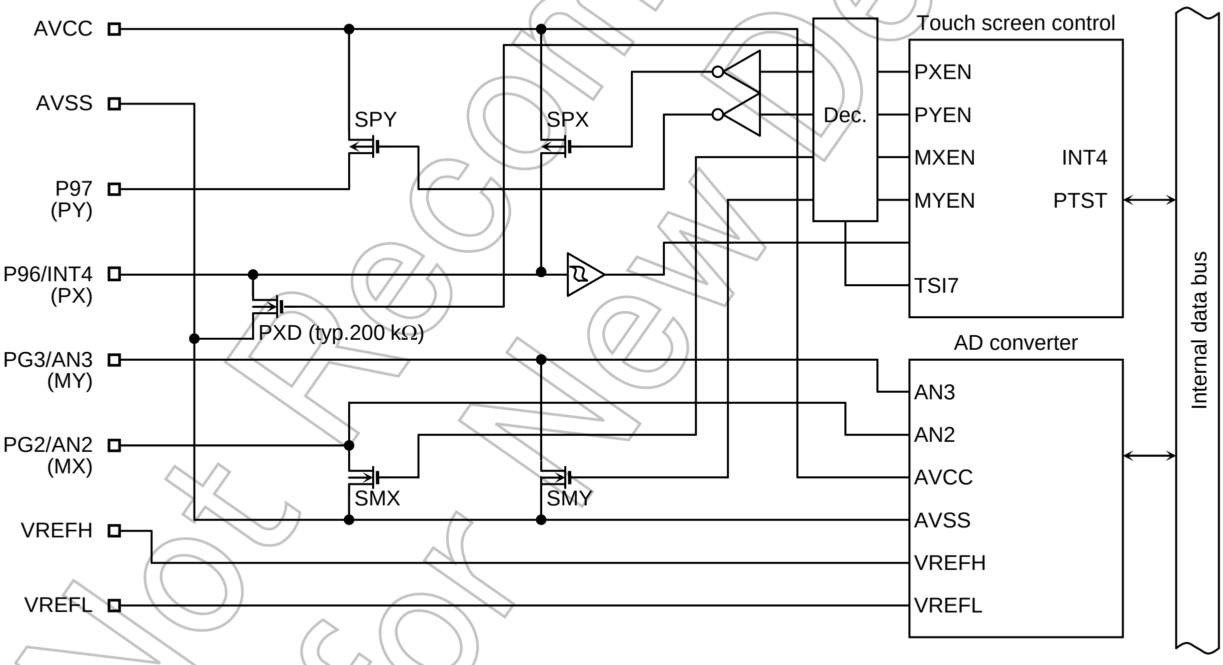


Figure 3.19.2 Internal Block Diagram of TSI

3.19.2 Touch Screen Interface (TSI) Control Register

TSI Control Register

		7	6	5	4	3	2	1	0
TSICR0 (01F0H)	Bit symbol	TSI7		PTST	TWIEN	PYEN	PXEN	MYEN	MXEN
	Read/Write	R/W		R	R/W	R/W	R/W	R/W	R/W
	Reset State	0		0	0	0	0	0	0
	Function	0: Disable 1: Enable		Detection condition 0: no touch 1: touch	INT4 interrupt control 0: Disable 1: Enable	SPY 0 : OFF 1 : ON	SPX 0 : OFF 1 : ON	SMY 0 : OFF 1 : ON	SMX 0 : OFF 1 : ON

PXD (Internal Pull-down resistance) ON/OFF setting

<PXEN> <TSI7>	0	1
0	OFF	OFF
1	ON	OFF

Debounce Time Setting Register

		7	6	5	4	3	2	1	0
TSICR1 (01F1H)	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
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Disable 1: Enable	1024	256	64	8	4	2	1
			Debounce time is set by the formula “$(N \times 64 - 16)/f_{SYS}$”. “N” is the number of bits between bit6 and bit0 which are set to “1”. Note2)						

Note1: Since an internal clock is used for the debounce circuit, when IDLE1, STOP mode, the de-bounce circuit don’t operate and also interrupt which through this circuit is not generated. When IDLE1, STOP mode, set this circuit to disable (Write “0” to TSICR1<DBC7>) before entering HALT state.

Note2: Ex:

TSICR1=95H → N = 64 + 4 + 1 = 69

3.19.3 Touch Detection Procedure

The Touch detection procedure shows procedure until a pen is touched by the screen and it is detected.

By touching, TSI generates interrupt (INT4) and this procedure terminates. After an X/Y position measuring procedure is terminated, return to this procedure and wait for the next touch.

When the waiting state, make ON only the SPY switch ON and OFF the other 3 switches (SMY, SPX and SMX).

The pull-down resistor that is connected to the P96/INT4/PX pin is ON when the SPX switch is OFF.

During this waiting state, P96/INT4/PX pin's level is L because the internal Pull-down resistors (PXD) between the X and Y directions in the touch screen are not connected and INT4 is not generated.

When the pen touches the screen, P96/INT4/PX pin's level is H because the internal resistors between the X and Y directions in the touch screen are connected and INT4 is generated.

In order to avoid the generation of several interrupts from one touch, a debounce circuit is used, as below.

This can ignore the pulse under the time which is set to TSICR1 register.

The circuit detects the rising of signal, counts-up the time of the counter which is set, after count, receive the signal internal. During counting, when the signal is set to Low, counter is cleared. And the state become to state of waiting a rising edge.

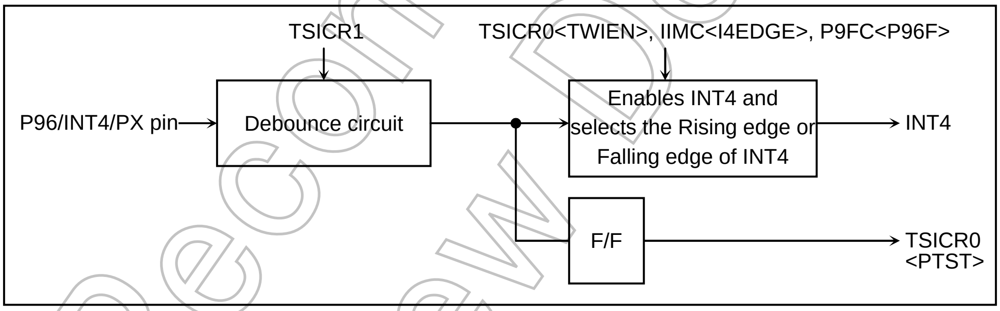

TSICR1
TSICR0<TWIEN>, IIMC<I4EDGE>, P9FC<P96F>
P96/INT4/PX pin
Debounce circuit
Enables INT4 and selects the Rising edge or Falling edge of INT4
INT4
F/F
TSICR0 <PTST>

Figure 3.19.3 Block Diagram of Debounce Circuit

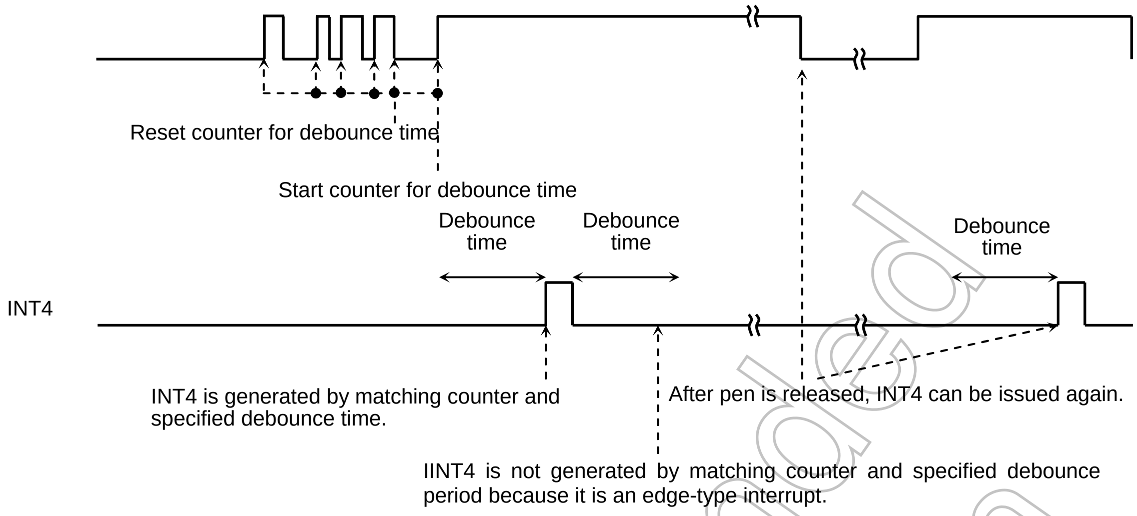

Reset counter for debounce time
Start counter for debounce time
Debounce time
Debounce time
Debounce time
INT4
INT4 is generated by matching counter and specified debounce time.
After pen is released, INT4 can be issued again.
IINT4 is not generated by matching counter and specified debounce period because it is an edge-type interrupt.

Figure 3.19.4 Timing Diagram of Debounce Circuit

3.19.4 X/Y Position Measuring Procedure

During the INT4 routine, execute an X/Y position measuring procedure as below.

<X position measurement>

Make both the SPX and SMX switches ON, and the SPY and SMY switches OFF.

With this setting, an analog voltage which shows the X position will be input to the PG3/MY/AN3 pin. The X position can be measured by converting this voltage to digital code using the AD converter.

<Y position measurement>

Next, make both the SPY and SMY switches ON and the SPX and SMX switches OFF.

With this setting, an analog voltage which shows the Y position will be input to the PG2/MX/AN2 pin. The Y position can be measured by converting this voltage to digital code using the AD converter.

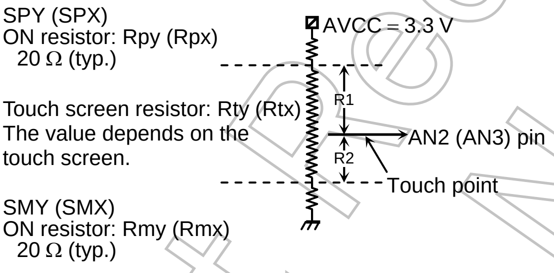
The above analog voltage which is inputted to AN3 or AN2 pin can be calculated as follows.

It is the ratio between the resistance value in the TMP92CH21F and the resistance value in the touch screen as shown in Figure 3.19.5.

Therefore, if the pen touches an area on the touch screen, the analog voltage will be neither 3.3 V nor 0.0 V.

Please remember to take into consideration the variation in the rate of resistance.

It is also recommended that an average taken from several AD conversions be adopted as the correct code.



[Formula to calculate analog voltage (E1) to AN2 or AN3 pin]

E1 = ((R2 + Rmy)/(Rpy + Rty + Rmy)) × AVCC [V]

Example: When AVCC = 3.3 V, Rpy = Rmy = 20 Ω,
R1 = 400 Ω and R2 = 100 Ω
E1 = ((100 + 20)/(20 + 400 + 100 + 20)) × 3.3
= 0.733 V

Note 1: An X position can be calculated in the same way though the above formula is for Y position.

Note 2: Rty = R1 + R2.

Figure 3.19.5 Calculation Analog Voltage

3.19.5 Flow Chart for TSI

(1) Touch detection procedure

(2) X/Y position measurement procedure

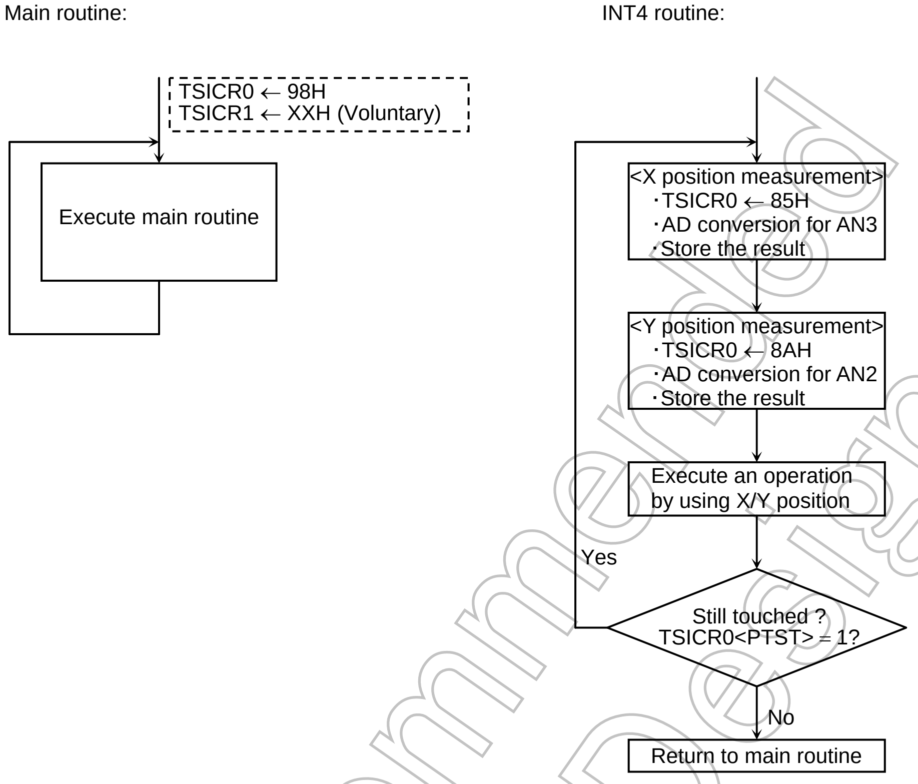


Figure 3.19.6 Flow Chart for TSI

3.20 I^{2}S (Inter-IC Sound)

An I^{2}S format compatible serial output circuit is built-in.

This product can be used in digital audio system applications by connecting LSI for sound generation (e.g., a DA converter).

This circuit has both I^{2}S mode and general SIO mode. But both modes have only clock output and data transmitting functions.

Figure 3.20.1 shows an outline for each mode.

Table 3.20.1 Outline for Each Mode

	I^{2}S mode	SIO mode
1) Format	I^{2}S-format compatible (Only master and transmitting)	General (Only master and 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 (at fc = 40 MHz)	Selectable either 20, 10, 5, or 2.5 Mbps	
5) Transmittion buffer	16 bytes × 2 channels (Right, left)	32 bytes
6) Direction of data	Selectable either MSB first or LSB first	
7) Data length	Selectable either 8 bits or 16 bits	
8) Edge of clock	Selectable either rising edge or falling edge	
9) Interrupt	INTI2S (FIFO empty interrupt)	

3.20.1 Block Diagram

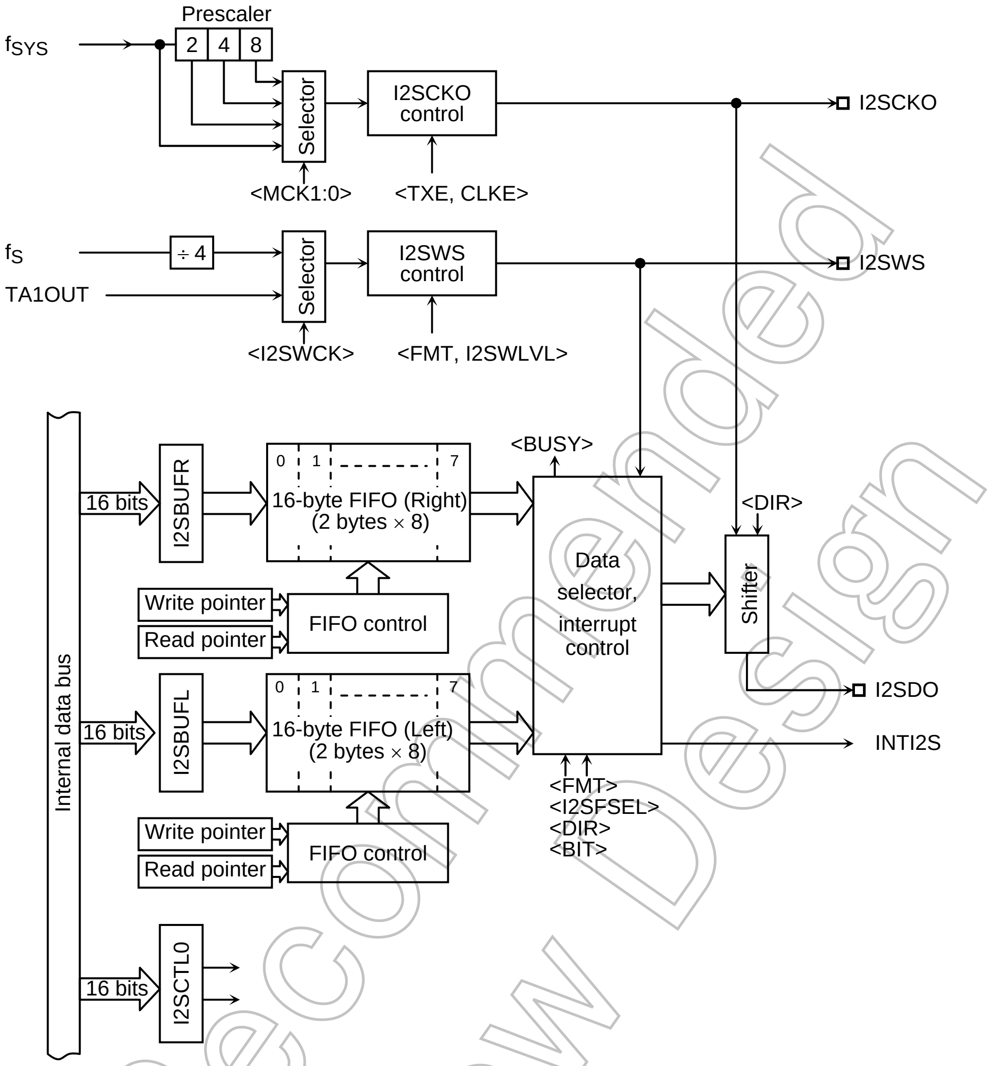


Figure 3.20.1 I^{2}S Block Diagram

3.20.2 SFR

The following tables show the SFR for I²S. This I²S is connected to the CPU by the 16-bit data bus. When the CPU accesses the SFR, use a 2-byte load instruction.

I2SCTL0 Register

		7	6	5	4	3	2	1	0
I2SCTL0 (080EH)	Bit symbol	TXE	FMT	BUSY	DIR	BIT	MCK1	MCK0	I2SWCK
	Read/Write	R/W		R	R/W				
	Reset State	0	0	0	0	0	0	0	0
	Function	Transmit 0: Stop 1: Start	Mode 0: I^{2}S 1: SIO	Status 0: Stop 1: Under transmitting	First bit 0: MSB 1: LSB	Bit number 0: 8 bits 1: 16 bits	Baud rate 00: f_{SYS} 10: f_{SYS}/4 01: f_{SYS}/2 11: f_{SYS}/8		WS clock 0: fs/4 1: TA1OUT

Note: <I2SWCK> is effective only for I^{2}S mode.

		15	14	13	12	11	10	9	8
(080FH)	Bit symbol	I2SWLVL	EDGE	I2SFSEL	I2SCLKE				SYSCKE
	Read/Write	R/W							R/W
	Reset State	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 channels) 1: Monaural (1 channel)	Clock enable (After transmit) 0: Operation 1: Stop				System clock 0: Disable 1: Enable

Note: <I2SWLVL>, <I2FSEL> and <I2SCLKE> are effective only in I^{2}S mode.

I2SBUFR Register

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2SBUFR (0800H) Read-modify-write instruction is prohibited	Bit symbol	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
	Read/Write	W															
	Reset State	Undefined															
	Function	Register for transmitting buffer (FIFO) (Right channel)															

I2SBUFL Register

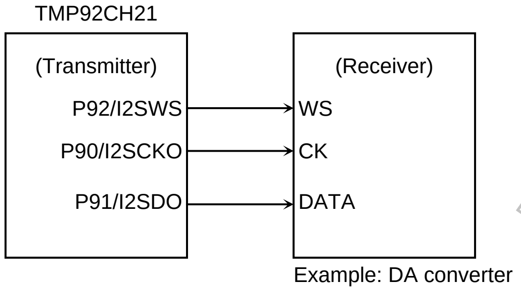
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2SBUFL (0808H) Read-modify-write instruction is prohibited	Bit symbol	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	L0
	Read/Write	W															
	Reset State	Undefined															
	Function	Register for transmitting buffer (FIFO) (Left channel)															

Figure 3.20.2 I^{2}S SFR

3.20.3 Explanation of I^2S Mode

(1) Connection example

Figure 3.20.3 shows an example with external LSI.



Note: After reset, P90 to P92 are placed in a high-impedance state. Connect each pin with a pull-up or pull-down resistor as necessary.

Figure 3.20.3 Example with External LSI

(2) Procedure

A 32-byte FIFO is built-in. If the FIFO's data becomes empty, an INTI2S interrupt is generated.

In the interrupt routine, write the next transmission data to the FIFO.

The following shows a setting example and timing diagram.

(Setting example) Transmitting by I^2S mode, I2SWS = 8.192 kHz, I2SCKO = 10 MHz, synchronous with rising edge (at f_{SYS} = 20 MHz)

(Main routine)

	7	6	5	4	3	2	1	0	
INTE5I2S	X	0	0	1	X	–	–	–	Set interrupts level.
P9CR	–	–	–	–	–	0	0	0	Set pins to P90 (I2SCKO), P91 (I2SDO), and P92 (I2SWS).
P9FC	–	–	–	–	–	1	1	1	
I2SCTL0	0	0	–	0	0	0	1	0	Set I^2S mode, MSB first, 8 bits, f_{SYS}/2 clocks.
	0	1	0	1	0	0	0	1	Set rising edge, clock stop.
I2SBUFR	**	**	**	**	**	**	**	**	Write 16-byte data to FIFO for right (8 times).
I2SBUFL	**	**	**	**	**	**	**	**	Write 16-byte data to FIFO for left (8 times).
I2SCTL0	1	0	–	0	0	0	1	0	Start transmitting.
	0	1	0	1	0	0	0	1	

(INTI2S interrupt routine)

I2SBUFR	**	**	**	**	**	**	**	**	Write 16-byte data to FIFO for right (8 times).
I2SBUFL	**	**	**	**	**	**	**	**	Write 16-byte data to FIFO for left (8 times).

X: Don't care, –: No change

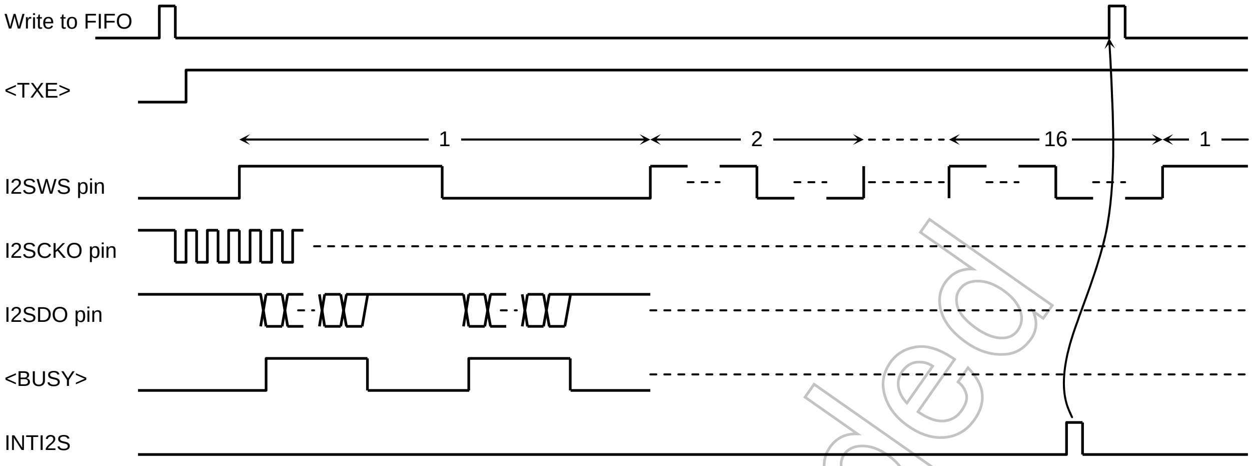


Figure 3.20.4 Whole Timing Diagram

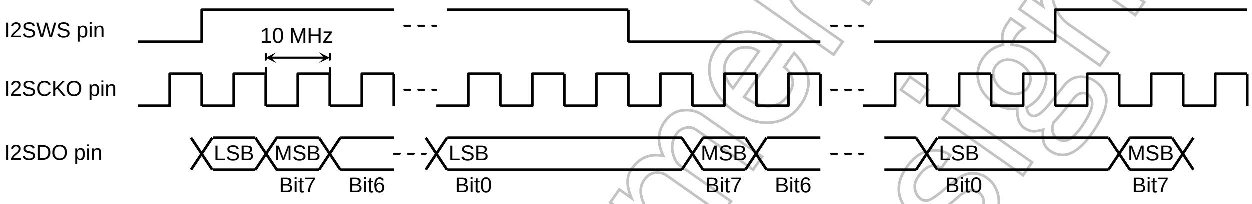


Figure 3.20.5 Detail Timing Diagram

(3) Notes

1) INTI2S timing

INTI2S is generated after the last data of FIFO is loaded to the internal shifter.

FIFO is now empty and it is possible to write the next data.

2) I2SCTL0<TXE>

A transmission is started by programming "1" to the <TXE> register and stopped by writing "0".

After<TXE> is programmed "1" once, the transmission is repeated automatically from right to left in order, alternately.

If a transmission should be stopped, program "0" to <TXE> after <BUSY> changes to "0" in the INTI2S interrupt routine.

When <TXE> is programmed "0" during transmitting, transmitting stops immediately.

3) FIFO size

A 16-byte FIFO is provided for both right and left channels. It is not necessary to use all data, but please use the even numbers (2, 4 ... 16).

4) I2SCTL0<I2SFSEL>

Write "1" to <I2SFSEL> and use the right channel FIFO for monaural.

It is not necessary to write data to the left channel FIFO. Channel transmission data is fixed at "0".

5) Address for I2SBUFR and I2SBUFL

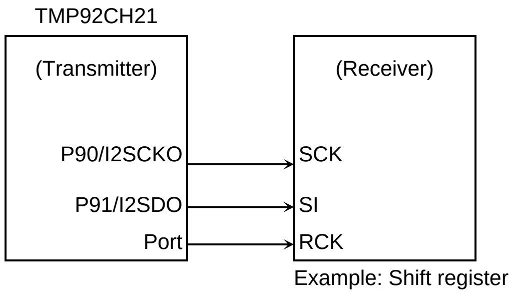
If writing data to I2SBUFR or I2SBUFL, use "word or long word data load instruction". A "byte data load instruction" cannot be used.

The address of I2SBUFR selectable from 0800H to 0803H, and I2SBUFL is selectable from 0808H to 080BH.

3.20.4 Explanation of SIO Mode

(1) Connection example

Figure 3.20.6 shows an example with external LSI.



Note: Since P90 to P91 become high impedance by reset, connect a pull-up or pull-down resistor if necessary.

Figure 3.20.6 Example with External LSI

(2) Procedure

A 32-byte FIFO is built-in. If the FIFO's data becomes empty, an INTI2S interrupt is generated.

In the interrupt routine, write the next transmission data to the FIFO.

The following shows a setting example and timing diagram.

(Setting example) Transmitting by SIO mode, I2SCKO = 10 MHz, synchronous with rising edge (at f_{SYS} = 20 MHz)

(Main routine)

	7	6	5	4	3	2	1	0	
INTE5I2S	X	0	0	1	X	–	–	–	Set interrupts level.
P9CR	–	–	–	–	–	–	0	0	Set pins to P90 (I2SCKO) and P91 (I2SDO).
P9FC	–	–	–	–	–	–	1	1	
I2SCTL0	0	1	–	1	0	0	1	–	Set SIO mode, LSB first, 8 bits, f_{SYS}/2 clocks.
	–	1	–	1	0	0	0	1	Set rising edge.
I2SBUFR	**	**	**	**	**	**	**	**	Write 32-byte data to FIFO (16 times).
I2SCTL0	1	1	–	1	0	0	1	–	Start transmitting.
	–	1	–	1	0	0	0	1	

(INTI2S interrupt routine)

I2SBUFR	**	**	**	**	**	**	**	**	Write 32-byte data to FIFO (16 times).
If <BUSY> = "1" then WAIT else NEXT									Confirm termination of the 32-byte data transfer.
I2SCTL0	1	1	–	1	0	0	1	–	Start transmitting.
	–	1	–	1	0	0	0	1	

X: Don't care, –: No change

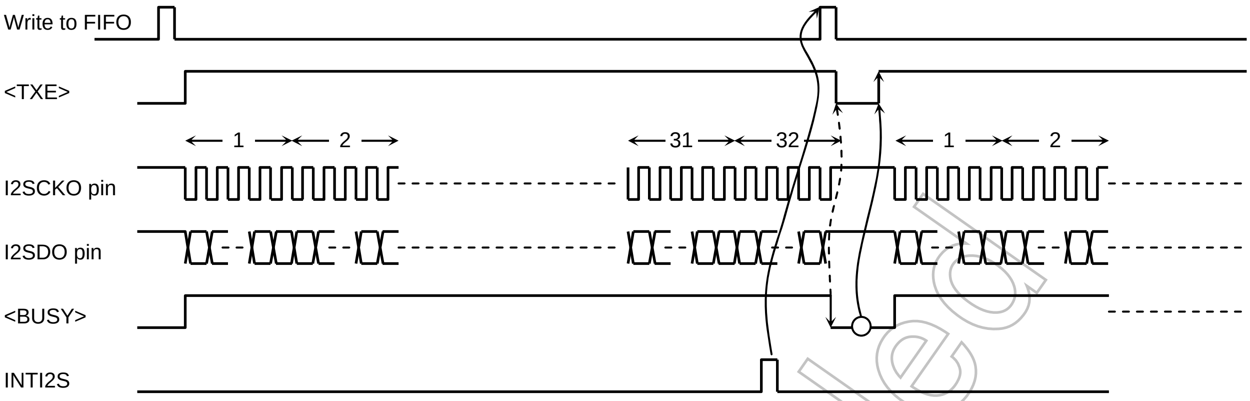


Figure 3.20.7 Whole Timing

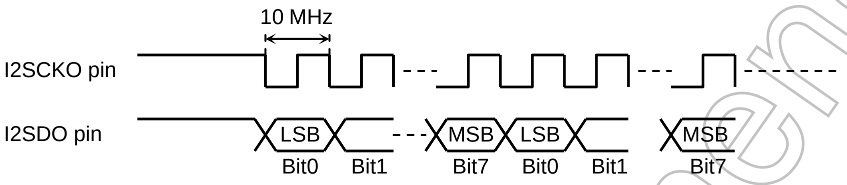


Figure 3.20.8 Detail Timing

(3) Notes

1) INTI2S timing

INTI2S is generated after the last data of FIFO is loaded to the internal shifter.

FIFO is now empty and it is possible to write the next data.

2) I2SCTL0 <TXE>

A transmission is started by programming "1" to the <TXE> register and stopped by programming "0".

<TXE> register is cleared to "0" when <BUSY> changes from "1" to "0".

When <TXE> is programmed "0" during transmitting, transmitting stops immediately.

3) FIFO size

A 32-byte FIFO is provided for SIO mode. It is not necessary to use all data but please use even numbers (2, 4 ... 32).

The <BUSY> will be changed to "0" and <TXE> will be cleared to "0" automatically after transmitting all programmed data to FIFO. In case of continuous transmitting, program "1" to <TXE> after programming data to FIFO.

The number of data programmed to FIFO is counted automatically and held by programming "1" to <TXE>.

4) Address for I2SBUFR and I2SBUFL

If writing data to I2SBUFR (I2SBUFL cannot be written), use "word or long word data load instruction". A "byte data load instruction" cannot be used.

The address of I2SBUFR is selectable from 0800H to 0803H.

3.21 Boot ROM

A boot ROM is built-in to download user's boot program.
Three downloading methods are supported.

3.21.1 Operation Mode

There are 2 operation modes: MULTI mode and BOOT mode. Each mode is set according to the status of the AM1 and AM0 pins when $\overline{\text{RESET}}$ is asserted.

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

(2) BOOT mode: After reset, the CPU fetches and executes instructions from the internal boot ROM. A user program which executes programming to on-board memory (e.g., NOR flash) is loaded from either NAND flash, UART or USB to internal RAM, and then branched to the internal RAM.
This operation will initiate a user program boot.
Table 3.21.2 shows an outline of boot operation.

Table 3.21.1 Operation Mode

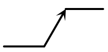
Mode Setting Pins			Operation Mode	
$\overline{\text{RESET}}$	AM1	AM0		
	0	1	MULTI	Start from external 16-bit bus memory
	1	0		Start from external 32-bit bus memory
	1	1	BOOT (Start from internal boot ROM)	
	0	0	TEST (Disabled to set)	

Table 3.21.2 Outline of Boot Operation

Name	Order of Setting	Loading			Operation after Loading
		Source	I/F	Destination	
(a)	1	NAND flash	Data bus	Internal RAM	Branch to internal RAM
(b)	2	PC	UART		
(c)	3	PC	USB		

3.21.2 Hardware Specification for Internal Boot ROM

(1) Memory map

Figure 3.21.1 shows a memory map of BOOT mode.

An 8-Kbyte ROM is built-in and it is mapped to address 3FE000H to 3FFFFFH.

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

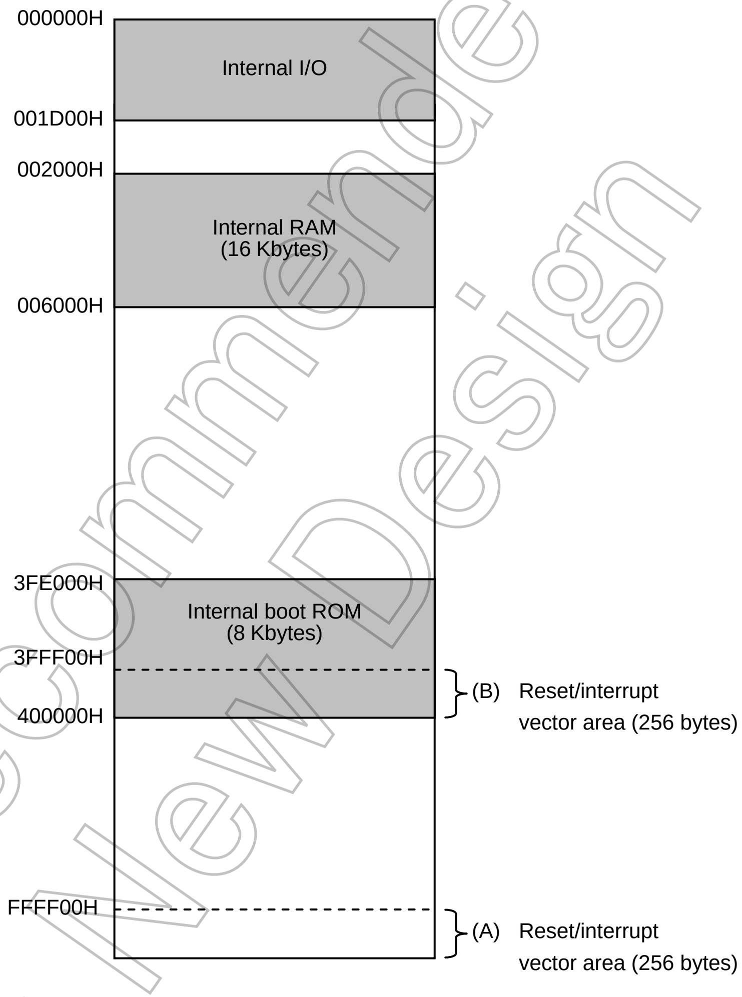


Figure 3.21.1 Memory Map of BOOT Mode

(2) Reset/interrupt address conversion circuit

A reset/interrupt vector address conversion circuit is included.

This function allows for individual reset/interrupt vector areas. For details, refer to section 3.6.5, Internal Boot ROM Control.

(3) Clearing boot ROM

After boot sequence in BOOT mode, the application system program may continue to run without reset asserting. In this case, any external memory which is mapped to address 3FE000H to 3FFFFFH cannot be accessed because the boot ROM is assigned here.

So, an internal boot ROM can be cleared by setting BROMCR<ROMLESS> to “1”.

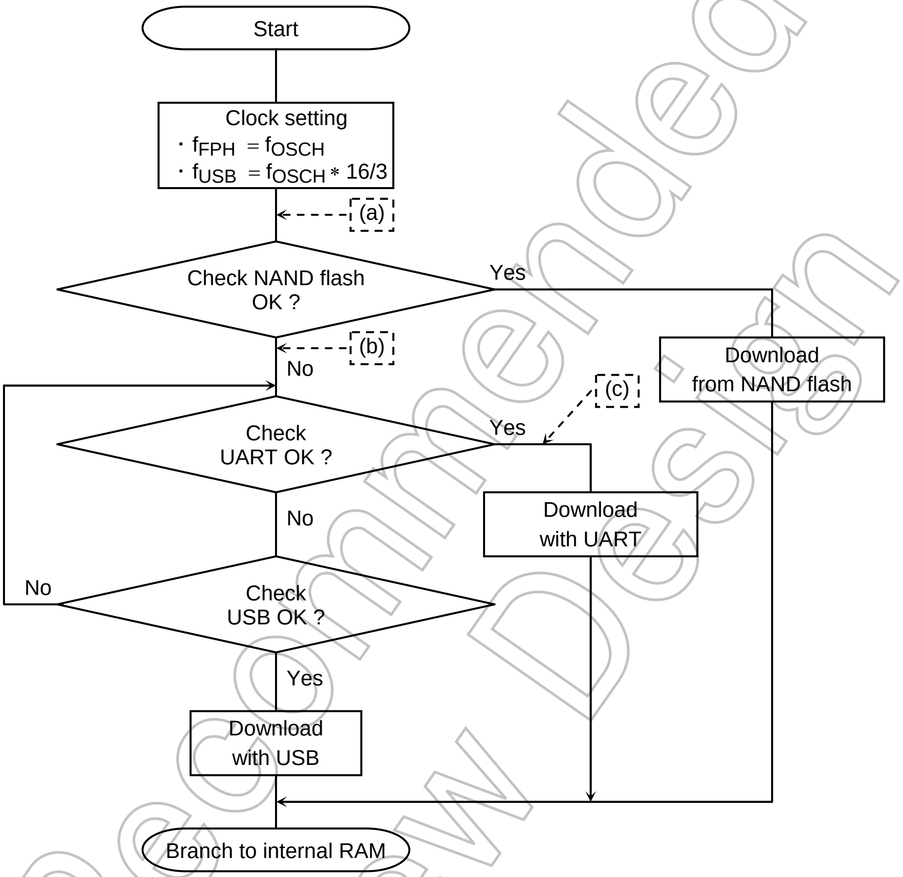
For the details, refer to section 3.6.5, Internal Boot ROM Control.

3.21.3 Outline of Boot Operation

There are 3 downloading methods: NAND flash, UART and USB.

After reset, a boot program in the boot ROM operates as shown in the Figure 3.21.2 flow chart.

Internal RAM use is the same regardless of downloading method, and is shown in Figure 3.21.3.



Note 1: When USB downloading is used, a special USB device driver and application software are needed on the PC.

Note 2: When UART downloading is used, special application software is needed on the PC.

Note 3: (a), (b) and (c) on the flow chart show the points at which external port pins are set. Refer to Table 3.21.3 for details.

Figure 3.21.2 Flow Chart Outline of Internal Boot ROM

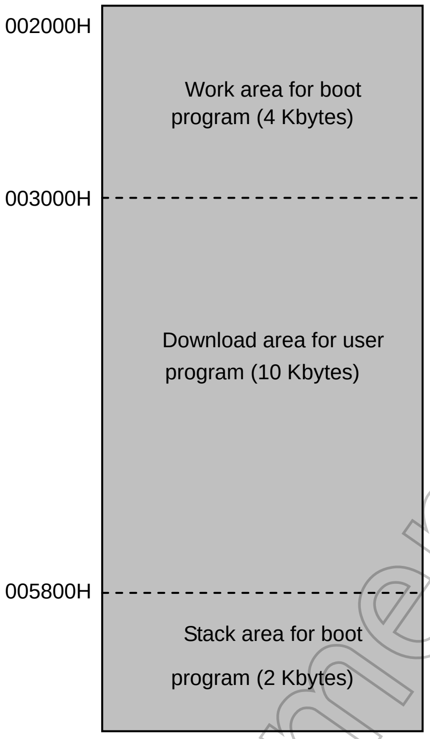

002000H
Work area for boot program (4 Kbytes)
003000H
Download area for user program (10 Kbytes)
005800H
Stack area for boot program (2 Kbytes)

Figure 3.21.3 Internal RAM Use

(1) Port setting

The boot program port settings are shown in Table 3.21.3, and Table 3.21.4 shows PCB design. These port settings must be carefully noted when designing an application system.

The remaining ports are not set, so they maintain their status after reset.

Table 3.21.3 Port Setting

Port		Function	I/O	Port Setting by Boot Program		
				(a)	(b)	(c)
NAND flash	P71	$\overline{\text{NDRE}}$	Output	Set to the function pin shown left	No change from (a)	No change from (a)
	P72	$\overline{\text{NDWE}}$	Output			
	P75	NDR/$\overline{\text{B}}$	Input			
	P84	$\overline{\text{ND0CE}}$	Output			
	PJ5	NDALE	Output			
	PJ6	NDCLE	Output			
	–	D7 to D0	I/O	No change		
UART	PF0	TXD1	Output	No change to input port status after reset	No change from (a)	Set to the TXD1 output pin
	PF1	RXD1	Input	Set to the RXD1 input pin		No change from (a)
USB	–	D +	I/O	No change		
	–	D –	I/O			
	PC6	PUCTL	Output	No change to input port status after reset	Set to the output port pin	No change from (b)

Table 3.21.4 How to Design PCB

Port		Function	I/O	Boot Method		
				NAND flash	UART	USB
NAND flash	P71	$\overline{\text{NDRE}}$	Output	Connect to NAND flash and pull-up by 100 kΩ resistor because this pin is changed to input port by reset.	Not affected by UART boot. If the NAND flash is not used in the system, ensure no conflict with the I/O direction shown left.	Not affected by USB boot. If the NAND flash is not used in the system, ensure no conflict with the I/O direction shown left.
	P72	$\overline{\text{NDWE}}$	Output			
	P75	NDR/$\overline{\text{B}}$	Input	Connect to NAND flash and pull-up by 2 kΩ resistor because R/B pin of NAND flash has open-drain output buffer.		
	P84	$\overline{\text{ND0CE}}$	Output	Connect to NAND flash.		
	PJ5	NDALE	Output			
	PJ6	NDCLE	Output			
	−	D7 to D0	I/O			
UART	PF0	TXD1	Output	Not affected by NAND flash boot.	Connect to level shifter.	Not affected by USB boot.
	PF1	RXD1	Input			Pull-up by 100 kΩ to avoid UART executing.
USB	−	D +	I/O	Not affected NAND flash boot.	Not affected by UART boot.	Connect to USB connector, add dumping resistor (27Ω) and 1.5 kΩ pull-up which can be switched ON/OFF.
	−	D −	I/O			Connect to USB connector and add dumping resistor (27Ω).
	PC6	PUCTL	Output			Used to control ON/OFF pull-up resistor of D + pin. The switch should be ON by "1". As this pin changes to input port by reset, add100 kΩ pull-down.

Note 1: When booting method is either NAND flash or UART and USB is used in the system, ensure the D + pin pull-up resistor is not on in the BOOT mode.

Note 2: When booting method is USB, do not start UART application software on the PC.

Note 3: When booting method is UART, do not connect the USB connector.

(2) I/O registers setting by boot program

Table 3.21.5 shows I/O register setting by boot program.

Take particular note of these set values when using an application system program which continues to run without asserting a reset after a boot sequence is executed .

Also take note of the status of the CPU registers and internal RAM following execution of a boot sequence.

Table 3.21.5 I/O Register Setting by Boot Program

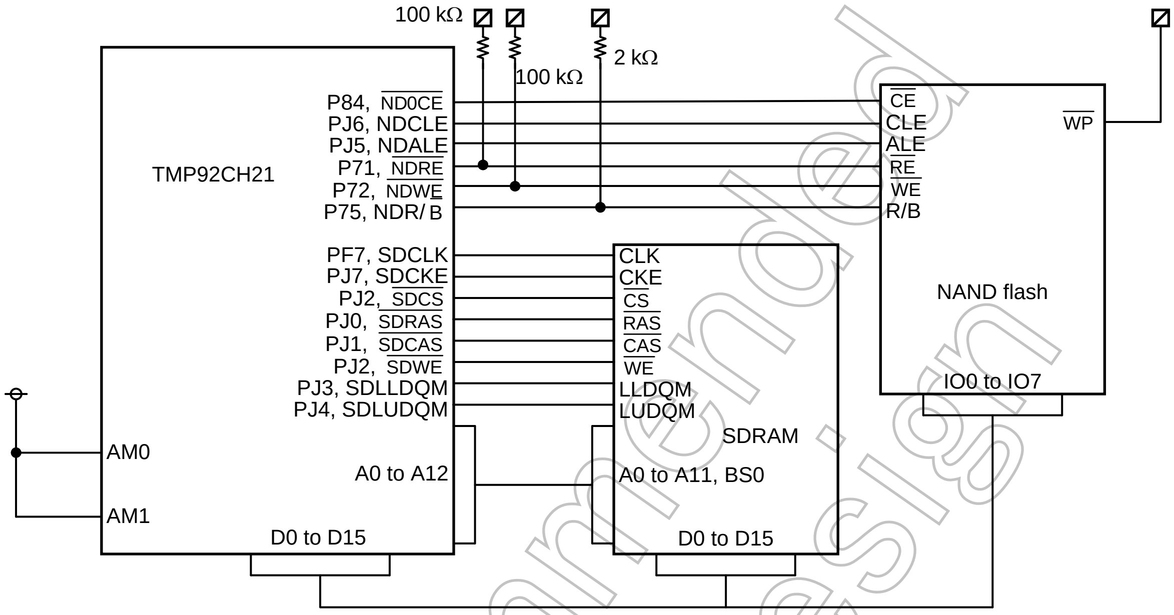
Symbol	Set Value	Set Content
WDMOD	00H	Stop watchdog timer.
WDCR	B1H	Disable watchdog timer.
SYSCR0	80H	Set system clock.
SYSCR1	00H	Set system clock.
SYSCR2	2CH	Set system clock.
PLLCR0	40H	Where USB is used for boot, set to use PLL output clock for f_{FPH}.
	00H	Where USB is not used for boot, set not to use PLL output clock for f_{FPH}.
PLLCR1	80H	Set to PLL ON. Not affected by boot method.
INTEUSB	04H	Set USB interrupt level.
INTETC01	44H	Set INTTC interrupt level.

Note: The setting values for NAND flash, UART and USB are not shown. Set each register where these functions are used in the system.

3.21.4 Download from NAND flash

(1) Connection example

Figure 3.21.4 shows an example of NAND flash. (A 16-bit SDRAM is used as program memory).



Note 1: The values of the pull-up resistors are recommended values.

Note 2: The WP (Write protect) pin of NAND flash is not supported by the TMP92CH21. If necessary, it must be prepared on an external circuit.

Figure 3.21.4 Example of NAND Flash Connection

(2) Supported NAND flash

The boot program is designed based on SmartMedia™ physical format specification Ver1.20. Table 3.21.6 shows supported memory devices and device codes.

Table 3.21.6 Supported Memory

Memory Size [Mbyte]	NAND Flash 3.3 V Model	Masked ROM 3.3 V Model
1	Not supported	
2		
4	OK (E3H)	OK (D5H)
8	OK (E6H)	OK (D6H)
16	OK (73H)	OK (57H)
32	OK (75H)	OK (58H)
64	OK (76H)	OK (D9H)
128	OK (79H)	OK (DAH)

(3) Data format

The download data consists of the boot identification code (4 bytes), user program size (2 bytes) and user program (max 10 Kbytes). These should be assigned (programmed) to NAND flash as shown in Figure 3.21.5. Also program the ECC code in the redundant area of the NAND flash, the block status area and thedata status area .

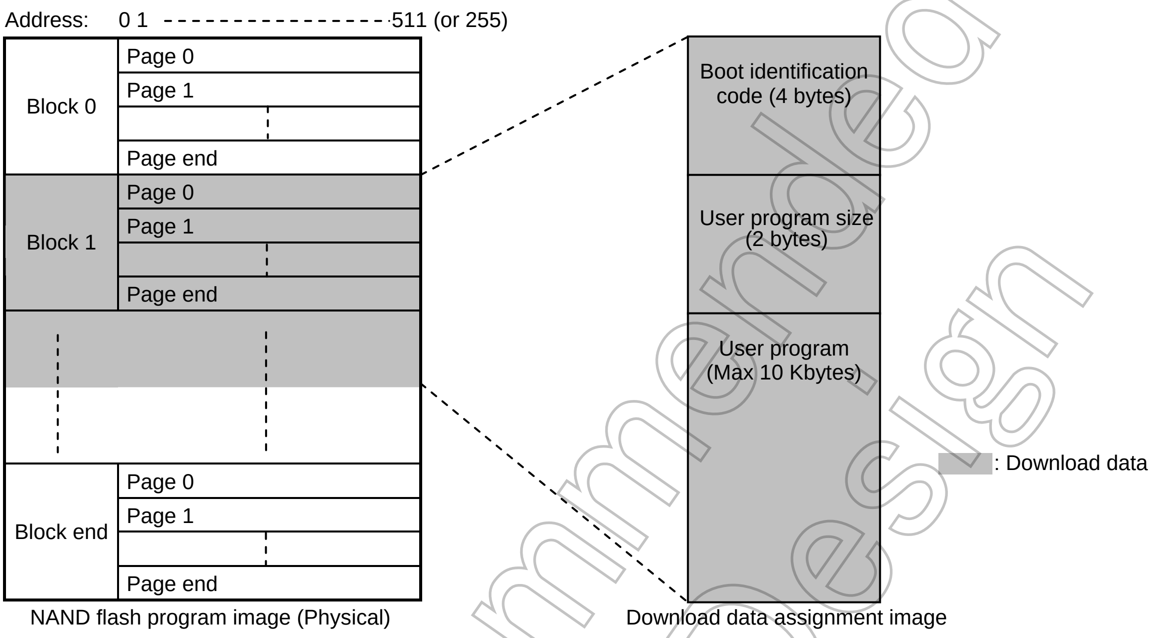

Address: 0 1 511 (or 255)
Block 0
Page 0
Page 1
Page end
Block 1
Page 0
Page 1
Page end
Block end
Page 0
Page 1
Page end
NAND flash program image (Physical)
Boot identification code (4 bytes)
User program size (2 bytes)
User program (Max 10 Kbytes)
▨ : Download data
Download data assignment image

Figure 3.21.5 Download Data Image

a) Boot identification code (4 bytes)

The boot program initially checks the boot identification code. If the boot characters in ASCII code are read from the first 4 bytes in page 0, block 1 of the NAND flash, the boot program will recognize the boot method as NAND flash.

42H (“B”)
4FH (“O”)
4FH (“O”)
54H (“T”)

Figure 3.21.6 Boot Identification Code

b) User program size (2 bytes)

The program size should be programmed to the next 2 bytes. The first byte is the lower 8 bits and the second is the upper 8 bits. This size indicates only the user program size; it does not include the boot identification code (4 bytes) and user program size (2 bytes).

This must be less than or equal to 10 Kbytes. So, the maximum number is 2800H.

Size (Lower 8 bits)
Size (Upper 8 bits)

Figure 3.21.7 User Program Size

c) User program (max 10 Kbytes)

This refers to a user program that is loaded to internal RAM.

When creating a user program, note the following points.

- Set start address to 3000H

  Beforehand, program (write) the user program to NAND flash in binary format.

  An example explaining how to make a binary format file is given below.

Example: How to convert from Intel Hex format file to binary format file

The following is an example of display in text editor when an Intel Hex format file is opened.

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

In fact, their data are as below because ASCII code is used for Intel Hex format files.

```
3A3130333030303030303630374631303030333030303046323031303330303030
423146313630313042370D0A3A303030303030303146460D0A
```

So, first convert the above data to binary format using the table below.

Before (ASCII)	After (Binary)
3A	3A (Only 3A should not be converted.)
30 to 39	0 to 9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete

Next, delete characters other than data
(Start mark, data number, address, record type and checksum).

The Intel Hex format and its meaning are given below.

```
Data record   3A  10  3000  00  0607F100030000F201030000B1F16010  B7
              |   |    |    ↓ Data                                Checksum
              |   |    ↓ Record type
              |   ↓ Address
              ↓ Data number
             : (Start mark)
End record    3A  00  0000  01  FF
              |   |    |    ↓ Data
              |   |    ↓ Record type
              |   ↓ Address
              ↓ Data number
             : (Start mark)
```

(4) Error check item

The items checked by the boot program are given below.

If an error occurs in any check, the boot program will cancel downloading from NAND flash and skip to the next operation (recognizing UART or USB).

a) Supported NAND flash

The boot program reads a device code from NAND flash and checks whether it is supported or not.

b) Boot identification code

c) User program size

The boot program checks whether it is less than or equal to 10 Kbytes.

d) Block status area

The boot program checks whether each block is normal or not. If the block status area on first page of any block has 2-bit or more “0” data, it is an error.

e) Data status area

The boot program checks whether each data status is correct or not. If the data status area has 4-bit or more “0” data, it is an error.

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) NAND flash R/B

The boot program checks whether NDR/B pin is normal or not in each action.

If the busy status is longer than 70 [μs] at f_{FPH} = 40 MHz, it is an error.

(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 as below.

1. Compares both calculated ECC code from NDFC and ECC code in NAND flash.
2. Evaluates and corrects according to the following cases.

Case (a): No data error	→ (OK) Next operation
Case (b): 1-bit data error	→ (OK) Error correction and next operation
Case (c): 2-bit or more data error	→ (Error) Termination
Case (d): ECC code 1-bit error	→ (OK) Next operation
Case (e): ECC code 2-bit or more error	→ (Error) Termination

For reference, details of calculation flow are given below.

1) Make XOR data by calculating exclusive OR after both ECC code from NDFC and NAND flash are placed to 4-byte data as below.
Lower 2 bytes: Line parity
Upper 2 bytes: Column parity
(Valid data of column parity is lower 6-bit in upper 2 bytes)

2) If XOR data equals "0", it will terminate normally because the ECC code is the same, but if not, they are checked as to whether they are correctable or not.

3) If XOR data does not have 2-bit or more "1" data, it will terminate normally because of the ECC code 1-bit error.

4) If the effective data (2-bit width from bit0 to bit21 in XOR data) equals either 01B or 10B, it corrects data because they are correctable.

If the effective data has either 00B or 11B, it terminates abnormally because they are not correctable.

Example 1: If the XOR data equals 0026A65AH, shown below in binary,
0000000000 10 01 10 10 10 01 10 01 01 10 10B
all effective data (2-bit width from bit0 to bit21) equals either 01B or 10B. So, this is evaluated as being correctable.

Example 2: If the XOR data equals 002EA65AH, shown below in binary,
0000000000 10 11 10 10 10 01 10 01 01 10 10B
bit18 and bit19 are 11B, so this is evaluated as being uncorrectable.

5) Data correcting takes error line information from line parity in XOR data and error bit information from column parity and inverts the bit.

Example: If the XOR data equals 0026A65AH, line parity is shown below in binary.
10 10 01 10 01 01 10 10B
If 10B is converted to 1B and 01B is converted to 0B,
they become 1 1 0 1 0 0 1 1B and meaning the 212th byte.
In the same manner, error bit information becomes bit5.
As a result, it inverts bit5 of 212th byte.

3.21.5 Download with UART

(1) Connection example

Figure 3.21.8 shows an example of UART. (A 16-bit NOR flash is used as program memory.)

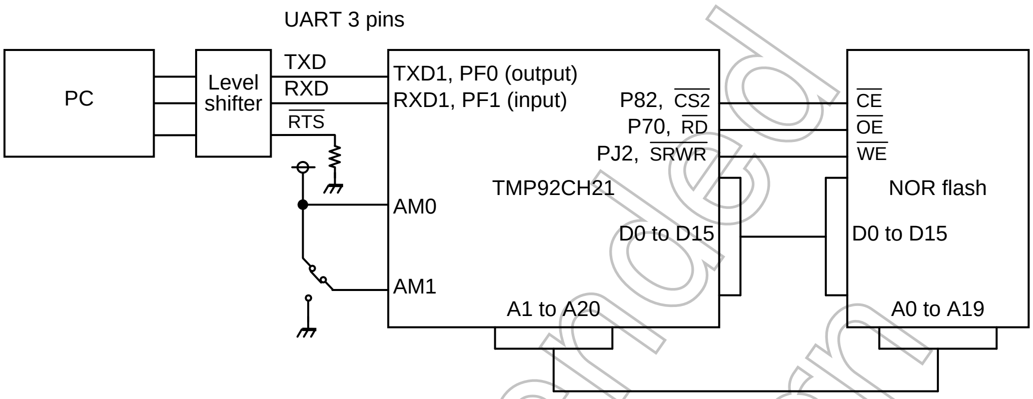


Figure 3.21.8 Example of UART

(2) UART interface specification

SIO channel 1 is used to download.

The following shows the UART communication format in BOOT mode.

Before booting, the PC side must also be setup in the same way.

The default baud rate is 9600 bps, but it can be changed to other values as shown in Table 3.21.9.

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

(3) UART data transfer format

Table 3.21.7 to Table 3.21.12 show the supported frequency, data transfer format, baud rate modification commands, operation commands, version management information, and frequency measurement result with data storing location, respectively.

Please also refer to the description of boot program operation in the following pages.

Table 3.21.7 Supported Frequency (f_{OSCH})

6.00 MHz	8.00 MHz	9.00 MHz	10.00 MHz	16.00 MHz	20.00 MHz	22.579 MHz	25.00 MHz	32.00 MHz	33.868 MHz	36.00 MHz	40.00 MHz

Note: Internal PLL (Clock multiplier) is not used.

Table 3.21.8 Transfer Format

	Byte Number to Transfer	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) Error: Nothing transmitted
	3rd byte to 6th byte	–		Version management information (Refer to Table 3.21.11)
	7th byte	–		Frequency information (Refer to Table 3.21.12)
	8th byte 9th byte	Baud rate modification command (Refer to Table 3.21.9) –		– OK: Echo back data Error: Error code × 3
	10th byte to n'th – 4 byte	User program Intel Hex format (binary)	New baud rate	Error: Stop operation by checksum error
	n'th – 3 byte	–		OK: SUM (High) (Refer (6) – c)
	n'th – 2 byte	–		OK: SUM (Low)
	n'th – 1 byte n'th byte	User program start command (C0H) (Refer to Table 3.21.10) –		– OK: Echo back data (C0H) Error: Error code × 3
RAM	–	Branch to user program start address		

"Error code × 3" means sending error code 3 times. For example, when error code is 62H, TMP92CH21 sends 62H 3 times. (For error code, refer to (4)-b.)

Table 3.21.9 Baud Rate Modification Command

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

Note 1:If f_{OSCH} is either 16.0, 20.0, 20.58 or 25.0 MHz, 115200 bps is not supported.

Note 2: If f_{OSCH} is 10.0 MHz, both 57600 and 115200 bps are not supported.

Note 3: If f_{OSCH} is 6.00, 8.00 or 9.00 MHz, then 38400, 57600 and 115200 bps are not supported.

Table 3.21.10 Operation Command

Operation Command	Operation
C0H	Start user program

Table 3.21.11 Version Management Information

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

Table 3.21.12 Frequency Measurement Result Data

f_{OSCH} [MHz]	6.000	8.000	9.000	10.000	16.000	20.000
2000H (RAM storing address)	09H	0AH	08H	0BH	00H	01H
	22.579	25.000	32.000	33.868	36.000	40.000
	02H	03H	04H	05H	06H	07H

(4) Description of UART boot program operation

The boot program receives data that is sent from the PC by UART, and loads it to internal RAM.

If the transferring terminates normally, it calculates SUM and sends the result to the PC before staring to execute the user program. The starting address to execute is the address received first . This boot program enables user's own on-board programming.

a) Operation procedure

1. Connect the serial cable . Make sure to perform connection before resetting the micro controller.
2. Set both AM1 and AM0 pins to "1" and reset the micro controller.
3. The receive data in the first byte is the matching data. When the boot program starts, it goes to a state in which it waits for the matching data to be received. Upon receiving the matching data, it automatically adjusts the serial channels' initial baud rate to 9600 bps. The matching data is 5AH.
4. The second byte is used to echo back 5AH to the 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 third through sixth bytes are used to send the boot program's version management information in ASCII code. The PC should check that the correct version of the boot program is used.

6. The seventh 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 eighth byte is the baud rate modification data. The five kinds of baud rate modification data shown in Table 3.21.9 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 ninth byte is used to echo back the received data to the PC when the data received in the eighth 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 bytes is received as binary data in Intel Hex format. No received data is echoed back to the PC.
   The boot program processing routine ignores the received data until it receives the start mark (3AH for " : ") in Intel Hex format. Nor does it send error code to the PC. 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 Intel Hex format occurs, the device goes to an IDLE state without returning error code to the PC.
   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 bytes and the n'th – 2 bytes are the SUM value that is sent to the PC in order of upper byte and lower byte. For details on how to calculate the SUM, refer to "notes on SUM" in the latter pages of this manual. The SUM calculation is performed only when no write error, receive error, or 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 PC. The PC 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 PC 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 PC. After sending the echo back to the PC, the stack pointer is set to 5FFFH and the boot program jumps to the 1st address that is received as data in 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 3 bytes of error code to the PC.

b) Error code

The boot program sends the processing status to the PC using various codes. The error codes are listed in the table below.

Table 3.21.13 Error Codes

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

Note 1: When a receive error occurs when receiving the user program, the device does not send the error code to the PC.

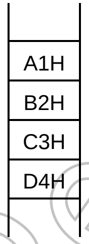
Note 2: After sending the error code, the device goes to an IDLE state.

c) Notes on SUM

1. Calculation method

SUM consists of byte + byte... + byte, the sum of which is returned in words as the result. Namely, data is read out in bytes, the sum of which is calculated, with the result returned in words.

Example:



If the data to be calculated consists of the 4 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 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 PC program has finished sending the checksum of the end record, it does not send anything and waits for 2 bytes 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 2 bytes to the PC.
3. Writing to areas outside the device's internal RAM causes incorrect operation. 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. The user program is assigned to the address from 3000H to 57FFH and it should be within 10 Kbytes.
6. Transmit a user program not by the ASCII code but by binary. An example explaining how to make binary format file is given below.

Example: How to convert from Intel Hex format file to binary format file.

The following is an example of display in text editor where an Intel Hex format file is opened.

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

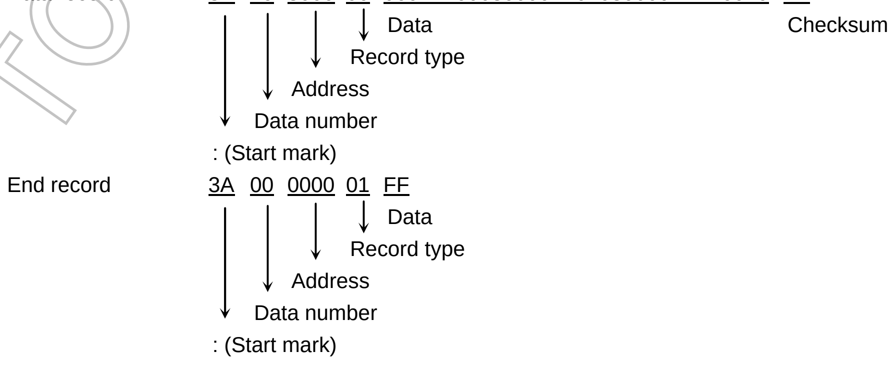
In fact, their data are as below because ASCII code is used for Intel Hex format files.

```
3A3130333030303030303036303746313030303330303030463230313033303030
4231463136303130423770D0A3A30303030303030314646 0D0A
```

So, first convert the above data to binary format using the table below.

Before (ASCII)	After (Binary)
3A	3A (Only 3A should not be converted.)
30 to 39	0 to 9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete it

The Intel Hex format and its meaning are given below.



e) Error when receiving user program

If the following errors occur in Intel Hex format when receiving the user program, the device goes to an idle state.

When the record type is not 00H, 01H, and 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 the 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 3.21.14. 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 3.21.14 and frequency error must be under 3%.

Table 3.21.14 Setting Error of Each Baud Rate (%)

	9600 bps	19200 bps	38400 bps	57600 bps	115200 bps
6.000 MHz	0.2	0.2	–	–	–
8.000 MHz	0.2	0.2	–	–	–
9.000 MHz	0.2	–0.7	–	–	–
10.000 MHz	0.2	0.2	–1.4	–	–
16.000 MHz	0.2	0.2	0.2	–0.8	–
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

–: Not supported

(5) Further notes

a) Handshake function

The TMP92CH21 has a $\overline{\text{CTS}}$ pin, but boot programs do not use it.

b) RS-232C connector

When the boot program is running, do not connect or disconnect an RS-232C connector.

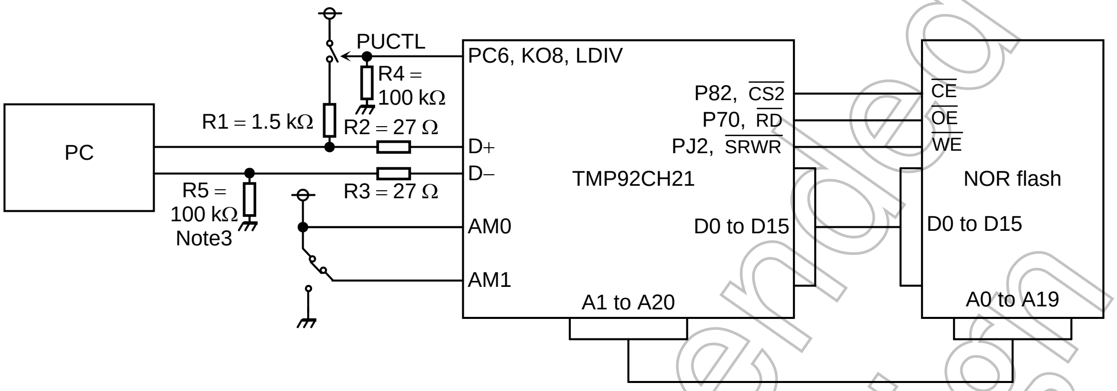
c) Software on PC

Special application software is needed on the PC.

3.21.6 Download with USB

(1) Connection example

Figure 3.21.9 shows an example of USB. (16-bit NOR flash is used as program memory.)



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

Note 2: The PC6 (KO8, LDIV) pin is assigned as PUCTL (Control to pull-up) for USB. So, note whether it is used as KO8 or LDIV.

Note 3: Pull-down resistor R2 is used only to fix the level for the flow current. If there is no ON/ OFF control by port for example, confirm operation by actual setting, and set the value to ensure the USB connection is not cut.

Figure 3.21.9 USB Connecting Example

(2) USB interface specification outline

For USB booting, make sure the oscillator is 9 MHz.

The baud rate is fixed at full speed (12 MHz).

The boot function is employed using the following 2 transfer types.

Table 3.21.15 Transfer Types Used by Boot Program

Transfer Type	Purpose
Control	Used as transmitting for standard request or vendor request
Bulk	Used as transmitting for vendor request or user program

An outline flowchart is given below.

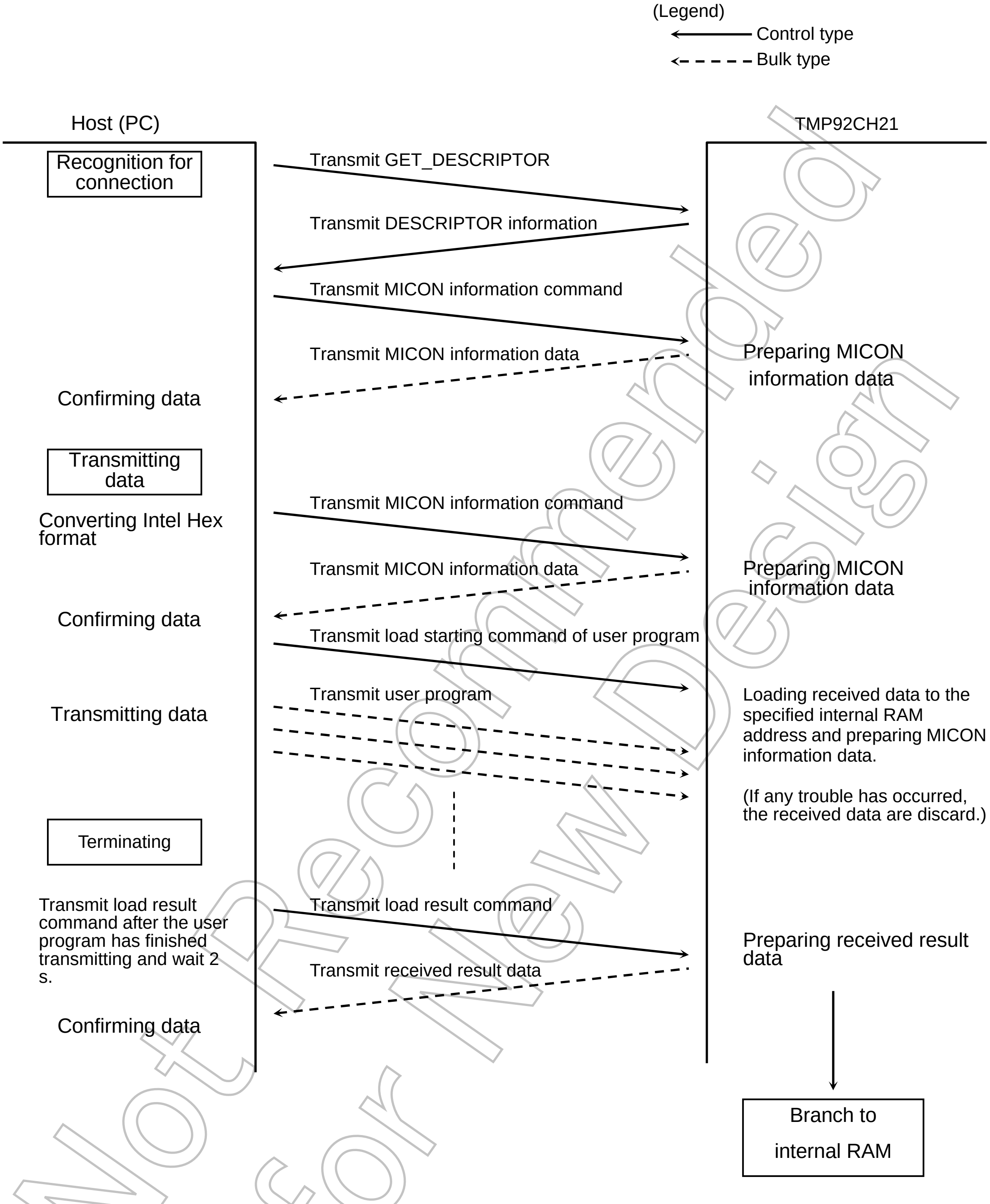

(Legend)
Control type
Bulk type
Host (PC)
TMP92CH21
Recognition for connection
Transmit GET_DESCRIPTOR
Transmit DESCRIPTOR information
Transmit MICON information command
Transmit MICON information data
Preparing MICON information data
Confirming data
Transmitting data
Converting Intel Hex format
Transmit MICON information command
Transmit MICON information data
Preparing MICON information data
Confirming data
Transmit load starting command of user program
Transmitting data
Transmit user program
Loading received data to the specified internal RAM address and preparing MICON information data.
(If any trouble has occurred, the received data are discard.)
Terminating
Transmit load result command after the user program has finished transmitting and wait 2 s.
Transmit load result command
Preparing received result data
Transmit received result data
Confirming data
Branch to internal RAM

Figure 3.21.10 Outline Flowchart

The vendor request command table is shown below.

Table 3.21.16 Vendor Request Command Table

Command Name	Value of Request	Outline	Notes
MICON (Microcomputer) information command	00H	Transmit microcomputer information	This is transmitted after a setup stage is terminated by bulk in transfer type.
Load starting command of user program	02H	Receive user program	Substitute size of user program to wIndex. The user program should be received after a setup stage is terminated by bulk out transfer type.
Transmit result command	04H	Transmit the result	This is transmitted after a setup stage is terminated by bulk in transfer type.

The data structure of setup command is shown below.

Table 3.21.17 Data Structure of Setup Command

Field Name	Value	Meaning
bmRequestType	40H	D7 0: Host to device D6-D5 2: Vender D4-D0 0: Device
bRequest	00H, 02H, 04H	00H: MICON information 02H: Start to transmit user program 04H: Result for user program received
wValue	00H to FFFFH	Own data number (Not used by boot program)
wIndex	00H to FFFFH	Size of user program (Used when a user program starts to be transmitted.)
wLength	0000H	Fixed

The standard request command table is shown below.

Table 3.21.18 The Standard Request Command Table

Standard Request	Response Medthod
GET_STATUS	By hardware, automatically
CLEAR_FEATURE	
SET_FEATURE	
SET_ADDRESS	
GET_DISCRIPTOR	
SET_DISCRIPTOR	Not supported
GET_CONFIGRATION	By hardware, automatically
SET_CONFIGRATION	
GET_INTERFACE	
SET_INTERFACE	
SYNCH_FRAME	Ignored

The information transmitted by GET_DISCRIPTOR is shown below.

Table 3.21.19 Information Transmitted by GET_DISCRIPTOR

Device Descriptor

Field Name	Value	Meaning
Blength	12H	18 bytes
BdescriptorType	01H	Device descriptor
BcdUSB	0110H	USB Version 1.1
BdeviceClass	00H	Device class is not used
BdeviceSubClass	00H	Sub command is not used
BdeviceProtocol	00H	Protocol is not used
BmaxPacketSize0	40H	EP0 max packet size 64 bytes
IdVendor	0930H	Vendor ID
IdProduct	6504H	Product ID (0)
BcdDevice	0001H	Device version (v 0.1)
Imanufacturer	00H	Index value of string descriptor in which producer is shown
Iproduct	00H	Index value of string descriptor in which product name is shown
IserialNumber	00H	Index value of string descriptor in which product number is shown
BnumConfigurations	01H	Configuration is 1

Configuration Descriptor

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

Interface Descriptor

Field Name	Value	Meaning
bLength	09H	9 bytes
bDescriptorType	04H	Interface descriptor
bInterfaceNumber	00H	Interface number 0
bAlternateSetting	00H	Alternate setting number 0
bNumEndpoints	02H	Endpoint is 2
bInterfaceClass	FFH	Specified device
bInterfaceSubClass	00H	
bInterfaceProtocol	50H	Bulk only protocol
iIinterface	00H	Index value of string descriptor in which this interface name is shown (Not used).

Endpoint Descriptor

Field Name	Value	Meaning
<Endpoint 1>		
blength	07H	7 bytes
bDescriptorType	05H	Endpoint descriptor
bEndpointAddress	01H	EP1 is OUT
bmAttributes	02H	Bulk transfer
wMaxPacketSize	0040H	Payload 64 bytes
bInterval	00H	(Ignored for bulk transfer)
<Endpoint 2>		
bLength	07H	7 bytes
bDescriptor	05H	Endpoint descriptor
bEndpointAddress	82H	EP2 is IN
bmAttributes	02H	Bulk transfer
wMaxPacketSize	0040H	Payload 64 bytes
bInterval	00H	(Ignored for bulk transfer)

The information transmitted by the MICON information command is shown below.

Table 3.21.20 Information Transmitted by MICON Information Command

Micon Information	ASCII Code
"TMP92CH21FG"	54H, 4DH, 50H, 39H, 32H, 43H, 48H, 32H, 31H, 46H, 47H, 20H, 20H, 20H, 20H

The information transmitted by the result information command is shown below.

Table 3.21.21 Information Transmitted by Result Information Command

Result	Value	Error Condition
No error	00H	
Not received user program error	02H	When a user program is received without receiving user program starting command.
Received except Intel Hex format error	04H	When the first data of the user program is not " : " (3AH).
Over user program size error	06H	When more than the value of wIndex is received.
Received incorrect address error	08H	When the user program address is incorrect. When the user program size is over 10 Kbytes
Protocol error or other error	0AH	When start or result of user program is received first. When check SUM is incorrect in Intel Hex file. When record type is incorrect in Intel Hex file. When address length is more than 2 in Intel Hex file. When end record length is not 0 in Intel Hex file.

(3) Description of USB boot program operation

The boot program provides the following RAM loader function.

The data, which is transmitted by the PC in Intel Hex format, is loaded to the internal RAM.

After loading normally, the user program will begin to execute. The first received address is set as the starting address.

By this function, this boot program enables the user's own on-board programming.

a. Operational procedure

1. Connect the USB cable.
2. Set both AM1 and AM0 pin to "1" and reset the micro controller.
3. On the PC side, recognize USB connection and confirm sub information by GET_DISCRIPTOR.
4. On the PC side, transmit MICON information command by vendor request and confirm MICON information data by Bulk IN after a setup stage is finished.
5. The boot program prepares MICON information in ASCII code after MICON information command is received.
6. On the PC side, convert user program into binary format.
7. On the PC side, transmit load-starting command by vendor request and transmit user program by Bulk OUT after a setup stage is finished.
8. On the PC side, wait 2 seconds and transmit load result command by vendor request. Confirm the result by bulk in after a setup stage is finished.
9. The boot program prepares the result after load result command is received.
10. If the result is not normal, the boot program cannot be returned normally. In this case, terminate device driver on the PC and retry from step 2.

b. Notes on user program format (Binary)

1. After receiving the checksum of a record, the device waits for the start mark (3AH for ": ") of the next record.The device therefore ignores all data received between records during that time unless the data is 3AH.
2. The first record type is not needed as an address record because the initial value of the address pointer is 00H.
3. The user program is assigned to the address from 3000H to 57FFH and it should be within 10 Kbytes.
4. In the user program, change the Intel Hex format file (usually ASCII code) to binary format and transfer it. The example below explains how to make a binary format file. (This is the same as with UART.)
   Make sure that the maximum data number of 1 record is FAH for the user program.

Example: Transfer data case of writing 16 bytes data from address 3000H by Intel Hex format file.

The following is an example of display in text editor where an Intel Hex format file is opened.

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

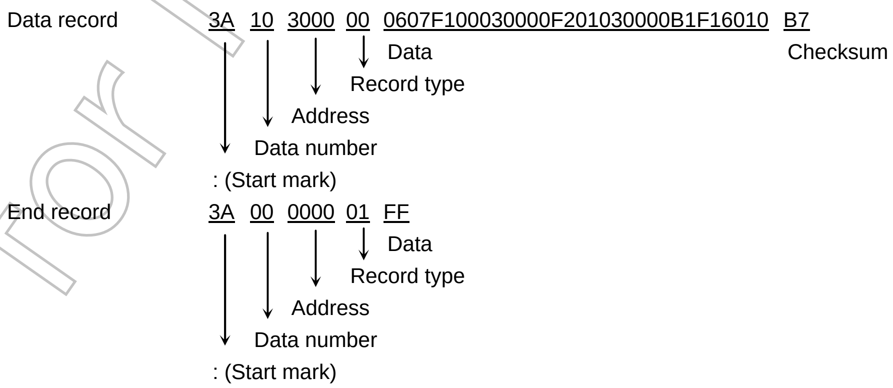
In fact, their data are as below because ASCII code is used for Intel Hex format files.

```
3A3130333030303030303036303746313030303330303030463230313033303030
423146313630313042370D0A3A30303030303030303146460D0A
```

So, first convert the above data to binary format using the table below.

Before (ASCII)	After (Binary)
3A	3A (Only 3A should not be converted.)
30 to 39	0 to 9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete it

The Intel Hex format and its meaning are given below.



(4) Further notes

a) USB connector

When the boot program is running, do not connect or disconnect the USB connector.

b) Software on PC

Special USB device driver and application software is needed on the PC.

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 to VCC + 0.5	V
Output Current	I_{OL}	2	mA
Output Current (MX, MY pin)	I_{OL}	15	mA
Output Current	I_{OH}	−2	mA
Output Current (PX, PY pin)	I_{OH}	−15	mA
Output Current (Total)	ΣI_{OL}	80	mA
Output Current (Total)	ΣI_{OH}	−80	mA
Power Dissipation (Ta = 85°C)	P_D	600	mW
Soldering Temperature (10 s)	T_{SOLDER}	260	°C
Storage Temperature	T_{STG}	−65 to 150	°C
Operation Temperature	T_{OPR}	−20 to 70	°C

Note: The absolute maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, the device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded.

Solderability

Test parameter	Test condition	Note
Solderability	(1) Use of Sn-37Pb solder Bath Solder bath temperature =230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	(2) Use of Sn-3.0Ag-0.5Cu solder bath Solder bath temperature =245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	

4.2 DC Electrical Characteristics (1/2)

V_{CC} = 3.3 ± 0.3V/X1 = 6 to 40 MHz/Ta = −20 to 70°C
V_{CC} = 2.7 − 3.6V/X1 = 6 to 27 MHz/Ta = −20 to 70°C

Parameter	Symbol	Min	Typ.	Max	Unit	Condition	
Power supply voltage (DVCC = AVCC) (DVSS = AVSS = 0 V)	V_{CC}	3.0		3.6	V	X1 = 6 to 40 MHz	XT1 = 30 to 34 kHz
		2.7				X1 = 6 to 27 MHz	
Input low voltage for D0 to D7 P10 to P17 (D8 to D15) P20 to P27 (D16 to D23) P30 to P37 (D24 to D31)	V_{IL0}	−0.3		0.6	V		
Input low voltage for P60 to P67, P71 to P72, P75 to P76, P90, P93 to P94, PC6 to PC7, PG0 to PG3, PJ5 to PJ6, PL4 to PL7	V_{IL1}			0.3 × V_{CC}			
Input low voltage for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PF0 to PF2, $\overline{\text{RESET}}$	V_{IL2}			0.25 × V_{CC}			
Input low voltage for AM0 to AM1	V_{IL3}			0.3			
Input low voltage for X1, XT1	V_{IL4}			0.2 × V_{CC}			
Input high voltage for D0 to D7 P10 to P17 (D8 to D15) P20 to P27 (D16 to D23) P30 to P37 (D24 to D31)	V_{IH0}	2.0		V_{CC} + 0.3	V		
Input high voltage for P60 to P67, P71 to P72, P75 to P76, P90, P93 to P94, PC6 to PC7, PG0 to PG3, PJ5 to PJ6, PL4 to PL7	V_{IH1}	0.7 × V_{CC}					
Input high voltage for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PF0 to PF2, $\overline{\text{RESET}}$	V_{IH2}	0.75 × V_{CC}					
Input high voltage for AM0 to AM1	V_{IH3}	V_{CC} − 0.3					
Input high voltage for X1, XT1	V_{IH4}	0.8 × V_{CC}					

DC Electrical Characteristics (2/2)

Parameter	Symbol	Min	Typ.	Max	Unit	Condition	
Output low voltage	V_{OL}			0.45	V	I_{OL} = 1.6 mA	
Output high voltage	V_{OH1}	2.4				I_{OH} = –400 μA	
	V_{OH2}	0.9 × V_{CC}				I_{OH} = –20 μA	
Internal resistor (ON) MX, MY pins	IMon			30	Ω	V_{OL} = 0.2V	V_{CC} = 3.0 to 3.6 V
Internal resistor (ON) PX, PY pins	IMon			30		V_{OH} = V_{CC} –0.2V	
Input leakage current	I_{LI}		0.02	±5	μA	0.0 ≤ V_{IN} ≤ V_{CC}	
Output leakage current	I_{LO}		0.05	±10	μA	0.2 ≤ V_{IN} ≤ V_{CC} – 0.2 V	
Power down voltage at STOP (for internal RAM backup)	V_{STOP}	1.8		3.6	V	V_{IL2} = 0.2 × V_{CC}, V_{IH2} = 0.8 × V_{CC}	
Pull-up resistor for $\overline{\text{RESET}}$, PA0 to PA7	R_{RST}	80		500	kΩ		
Programmable pull down resistor for P96	R_{KH}						
Pin capacitance	C_{IO}			10	pF	fc = 1 MHz	
Schmitt width for P91 to P92, P96 to P97, PA0 to PA7,PC0 to PC3, PF0 to PF2, $\overline{\text{RESET}}$	V_{TH}	0.4	1.0		V		
NORMAL (Note 2)	I_{CC}		33	65	mA	V_{CC} = 3.6 V, fc = 40 MHz	
IDLE2			16	26			
IDLE1			4.3	8.7			
SLOW (Note 2)			25.2	110	μA	Ta ≤ 70°C	V_{CC} = 3.6 V, fs = 32 kHz
				70		Ta ≤ 50°C	
IDLE2			15.1	80		Ta ≤ 70°C	
				30		Ta ≤ 50°C	
IDLE1			4.3	60		Ta ≤ 70°C	
				20		Ta ≤ 50°C	
STOP			0.2	50		Ta ≤ 70°C	V_{CC} = 3.6 V
				15		Ta ≤ 50°C	

Note 1: Typical values are for when Ta = 25°C and VCC = 3.3 V unless otherwise noted.

Note 2: ICC measurement conditions (NORMAL, SLOW):

All functions are operational; output pins are opened and input pins are fixed. C_L = 30 pF is loaded to data and address bus.

4.3 AC Characteristics

4.3.1 Basic Bus Cycle

Read cycle

No.	Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
			Min	Max				
1	OSC period (X1/X2)	t_{OSC}	25	166.7	25	27.7	37.0	ns
2	System clock period (= T)	t_{CYC}	50	333.3	50	55.5	74.0	
3	SDCLK low width	t_{CL}	0.5 T – 15		10	12.7	22	
4	SDCLK high width	t_{CH}	0.5 T – 15		10	12.7	22	
5-1	A0 to A23 valid → D0 to D31 Input at 0 waits	t_{AD} (3.0 V)		2.0 T – 30	70	81	–	
		t_{AD} (2.7 V)		2.0 T – 35	–	–	113	
5-2	A0 to A23 valid → D0 to D31 Input at 1 wait	t_{AD3} (3.0 V)		3.0 T – 30	120	136.5	–	
		t_{AD3} (2.7 V)		3.0 T – 35	–	–	187	
6-1	$\overline{RD}$ falling → D0 to D31 Input at 0 waits	t_{RD}		1.5 T – 30	45	53.3	81	
6-2	$\overline{RD}$ falling → D0 to D31 Input at 1 wait	t_{RD3}		2.5 T – 30	95	108.8	155	
7-1	$\overline{RD}$ low width at 0 waits	t_{RR}	1.5 T – 20		55	63.2	91	
7-2	$\overline{RD}$ low width at 1 wait	t_{RR3}	2.5 T – 20		105	118.8	165	
8	A0 to A23 valid → $\overline{RD}$ falling	t_{AR}	0.5 T – 20		5	7.7	17	
9	$\overline{RD}$ falling → SDCLK rising	t_{RK}	0.5 T – 20		5	7.7	17	
10	A0 to A23 valid → D0 to D31 hold	t_{HA}	0		0	0	0	
11	$\overline{RD}$ rising → D0 to D31 hold	t_{HR}	0		0	0	0	
12	$\overline{WAIT}$ setup time	t_{TK}	15		15	15	15	
13	$\overline{WAIT}$ hold time	t_{KT}	5		5	5	5	
14	Data byte control access time for SRAM	t_{SBA}		1.5 T – 30	45	53.3	81	
15	$\overline{RD}$ high width	t_{RRH}	0.5 T – 15		10	12.7	22	

Write cycle

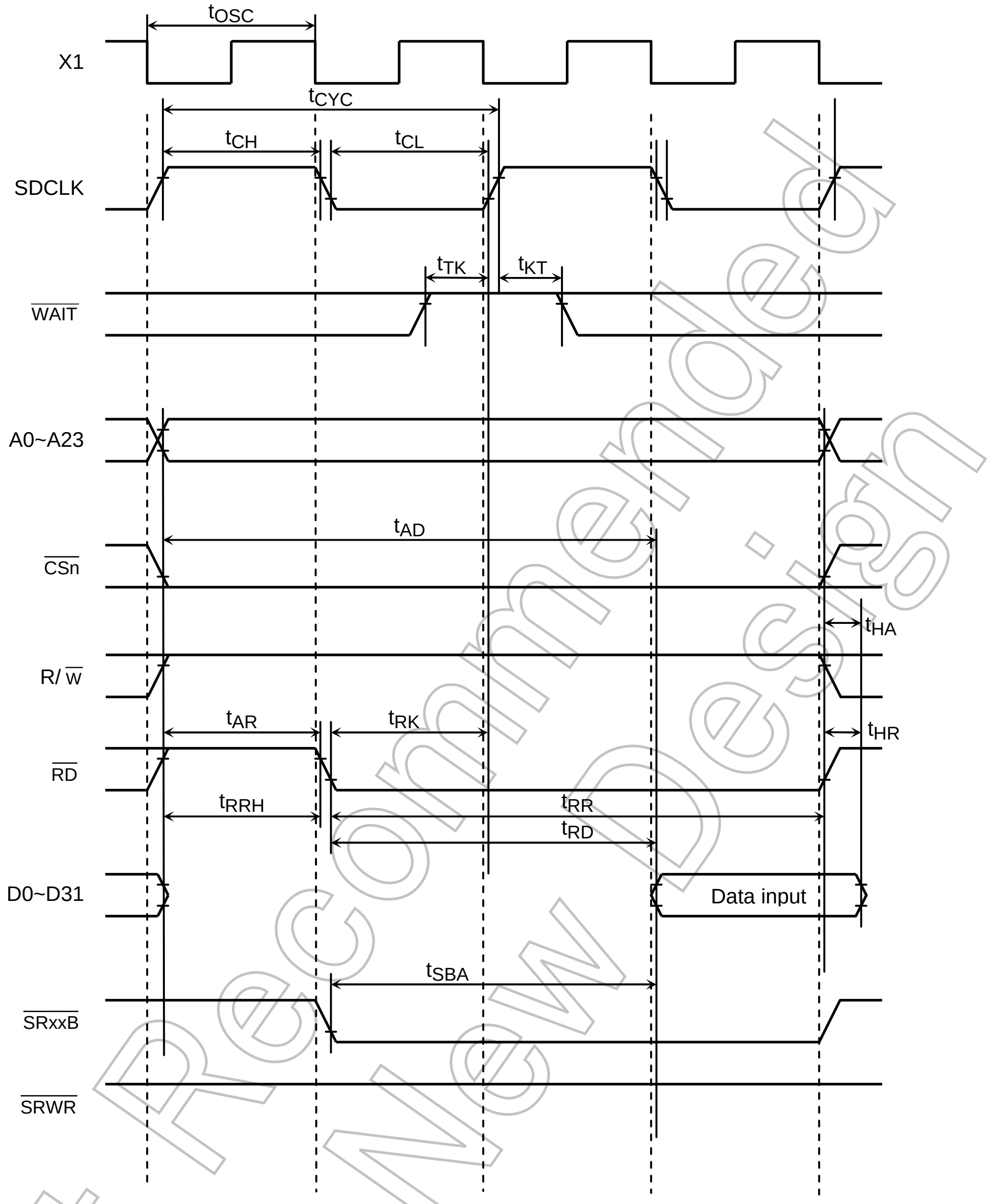
No.	Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
			Min	Max				
16-1	D0 to D31 valid → $\overline{WRxx}$ rising at 0 waits	t_{DW}	1.25T – 35		27.5	34.3	57.5	ns
16-2	D0 to D31 valid → $\overline{WRxx}$ rising at 1 wait	t_{DW3}	2.25T – 35		77.5	89.8	131.5	
17-1	$\overline{WRxx}$ low width at 0 waits	t_{WW}	1.25T – 30		32.5	34.3	62.5	
17-2	$\overline{WRxx}$ low width at 1 wait	t_{WW3}	2.25T – 30		82.5	89.8	136.5	
18	A0 to A23 valid → $\overline{WR}$ falling	t_{AW}	0.5T – 20		5	7.7	17	
19	$\overline{WRxx}$ falling → SDCLK rising	t_{WK}	0.5T – 20		5	7.7	17	
20	$\overline{WRxx}$ rising → A0 to A23 hold	t_{WA}	0.25T – 5		7.5	8.8	13.5	
21	$\overline{WRxx}$ rising → D0 to D31 hold	t_{WD}	0.25T – 5		7.5	8.8	13.5	
22	$\overline{RD}$ rising → D0 to D31 output	t_{RDO} (3.0 V)	0.5T – 5		20	22.7	–	
		t_{RDO} (2.7 V)	0.5T – 7		–	–	30	
23	Write pulse width for SRAM	t_{SWP}	1.25T – 30		32.5	39.3	62.5	
24	Data byte control to end of write for SRAM	t_{SBW}	1.25T – 30		32.5	39.3	62.5	
25	Address setup time for SRAM	t_{SAS}	0.5T – 20		5	7.7	17	
26	Write recovery time for SRAM	t_{SWR}	0.25T – 5		7.5	8.8	13.5	
27	Data setup time for SRAM	t_{SDS}	1.25T – 35		27.5	34.3	57.5	
28	Data hold time for SRAM	t_{SDH}	0.25T – 5		7.5	8.8	13.5	

AC measuring condition

- Output: High = 0.7 VCC, Low = 0.3 VCC, C_L = 50 pF
- Input: High = 0.9 VCC, Low = 0.1 VCC

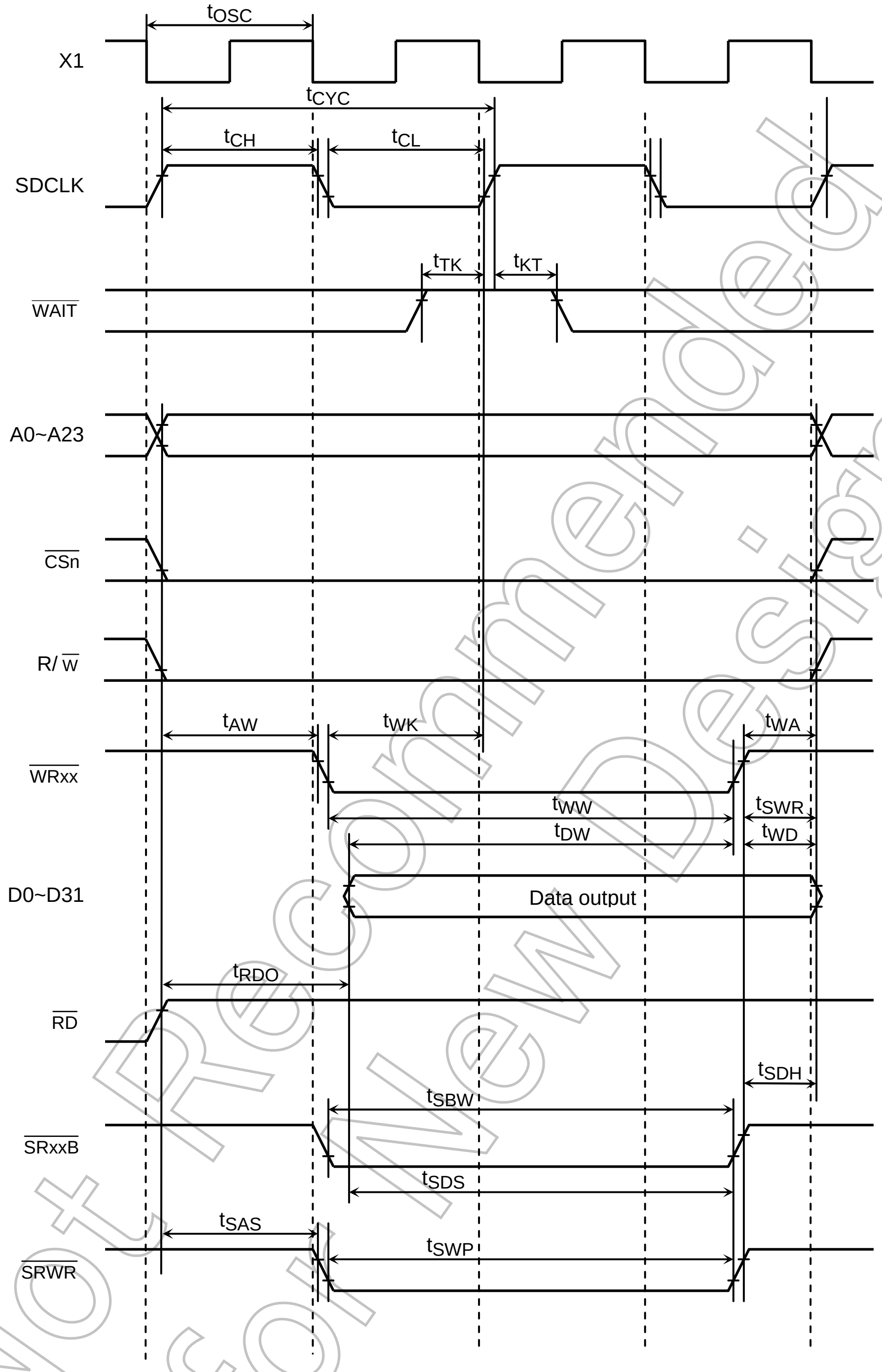
Note: The figures in the “Variable” column cover the whole VCC range (2.7 V to 3.6 V).
Exceptions are shown by the VCC (min), “(3.0 V)” or “(2.7 V)”, added to the “Symbol” column.

(1) Read cycle (0 waits)



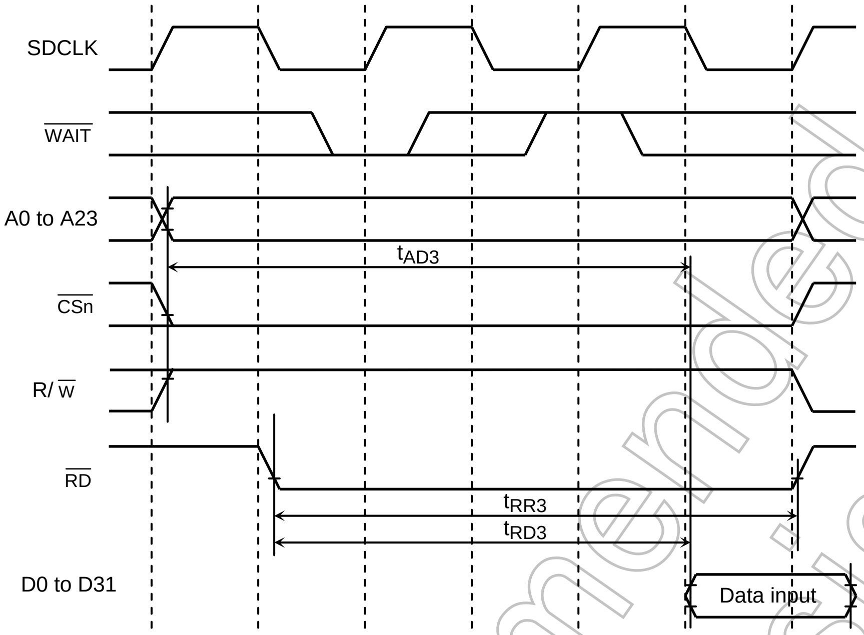
Note: The phase relation between X1 input signal and the other signals is undefined.
The above timing chart is an example.

(2) Write cycle (0 waits)

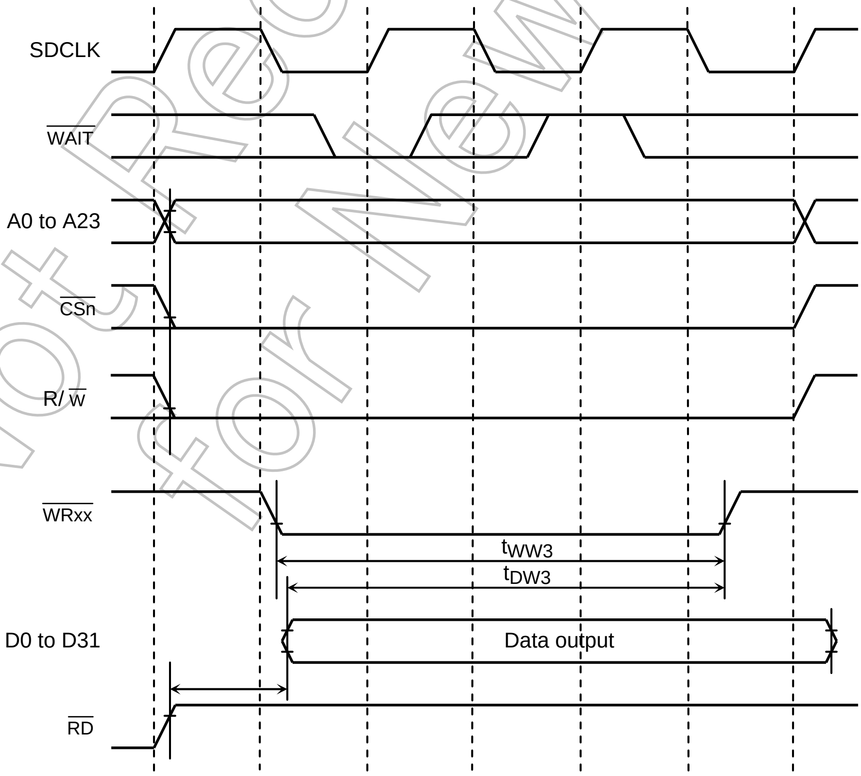


Note: The phase relation between X1 input signal and the other signals is undefined. The above timing chart is an example.

(3) Read cycle (1 wait)



(4) Write cycle (1 wait)



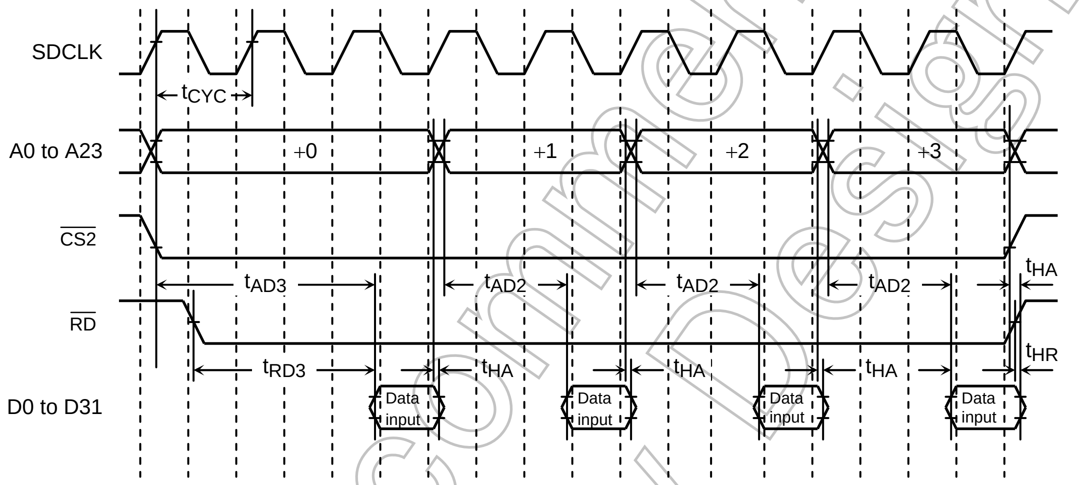
4.3.2 Page ROM Read Cycle

(1) 3-2-2-2 mode

No.	Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
			Min	Max				
1	System clock period (= T)	t_{CYC}	50	166.7	50	55.5	74	ns
2	A0, A1 → D0 to D31 input	t_{AD2}		2.0T − 50	50	61	98	
3	A2 to A23 → D0 to D31 input	t_{AD3}		3.0T − 50	100	116.5	172	
4	$\overline{\text{RD}}$ falling → D0 to D31 input	t_{RD3}		2.5T − 45	80	93.8	140	
5	A0 to A23 Invalid → D0 to D31 hold	t_{HA}	0		0	0	0	
6	$\overline{\text{RD}}$ rising → D0 to D31 hold	t_{HR}	0		0	0	0	

AC measuring condition

- Output: High = 0.7 VCC, Low = 0.3 VCC, C_L = 50 pF
- Input: High = 0.9 VCC, Low = 0.1 VCC



4.3.3 SDRAM Controller AC Characteristics

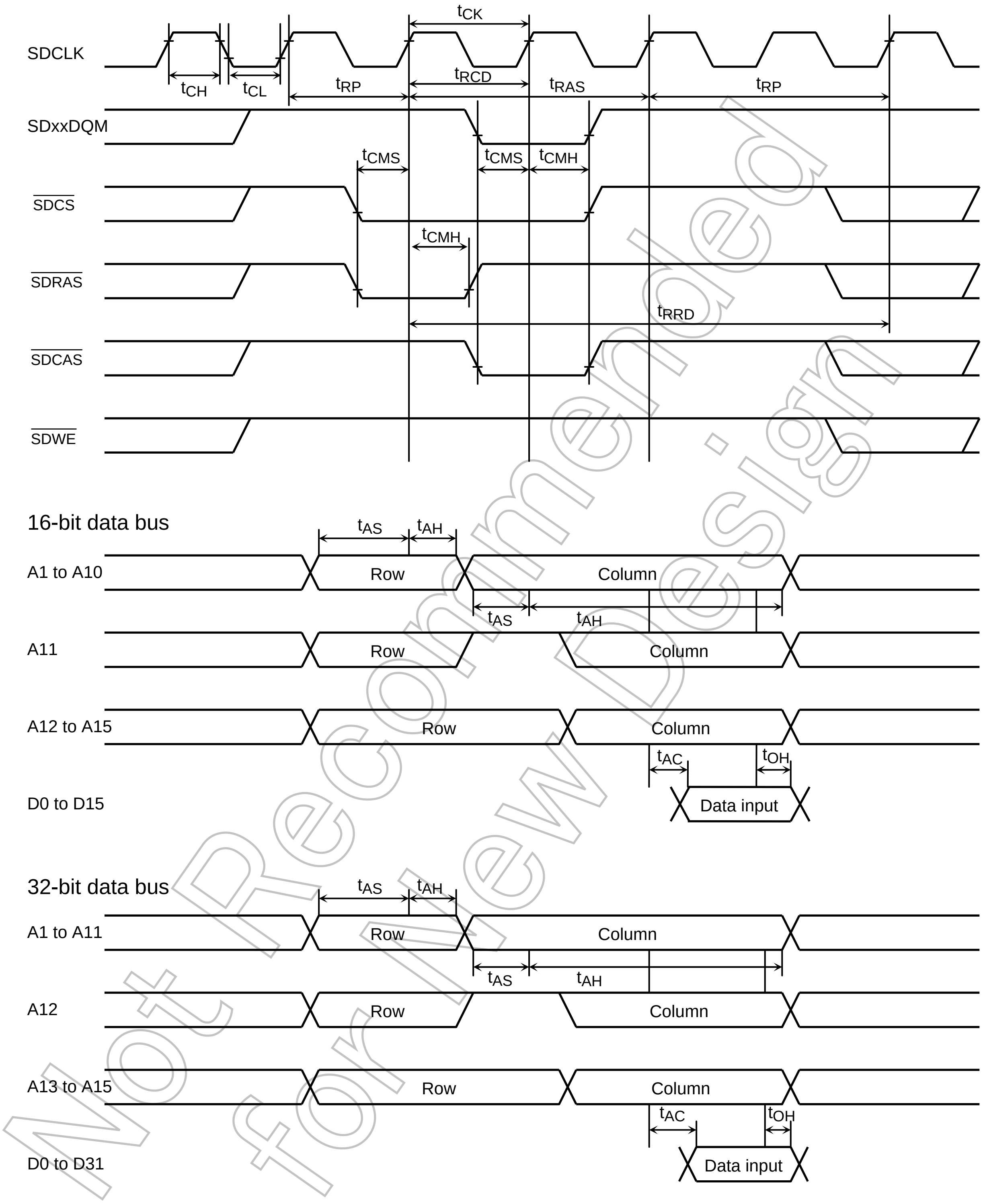
No.	Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
			Min	Max				
1	Ref/active to ref/active command period	t_{RC}	2T		100	111	148	ns
2	Active to precharge command period	t_{RAS}	2T	12210	100	111	148	
3	Active to read/write command delay time	t_{RCD}	T		50	55.5	74	
4	Precharge to active command period	t_{RP}	T		50	55.5	74	
5	Active to active command period	t_{RRD}	3T		150	166.5	222	
6	Write recovery time (CL* = 2)	t_{WR}	T		50	55.5	74	
7	Clock cycle time (CL* = 2)	t_{CK}	T		50	55.5	74	
8	Clock high level width	t_{CH}	0.5T – 15		10	12.7	22	
9	Clock low level width	t_{CL}	0.5T – 15		10	12.7	22	
10	Access time from clock (CL* =2)	t_{AC}		T – 30	20	25.5	44	
11	Output data hold time	t_{OH}	0		0	0	0	
12	Data in setup time	t_{DS}	T – 35		15	20.5	39	
13	Data in hold time	t_{DH}	T – 5		45	50.5	69	
14	Address setup time	t_{AS}	0.75T – 30		7.5	11.6	25.5	
15	Address hold time	t_{AH}	0.25T – 9		3.5	4.8	9.5	
16	CKE setup time	t_{CKS}	0.5T – 15		10	12.7	22	
17	Command setup time	t_{CMS}	0.5T – 15		10	12.7	22	
18	Command hold time	t_{CMH}	0.5T – 15		10	12.7	22	
19	Mode register set cycle time	t_{RSC}	T		50	55.5	74	

CL*: CAS latency.

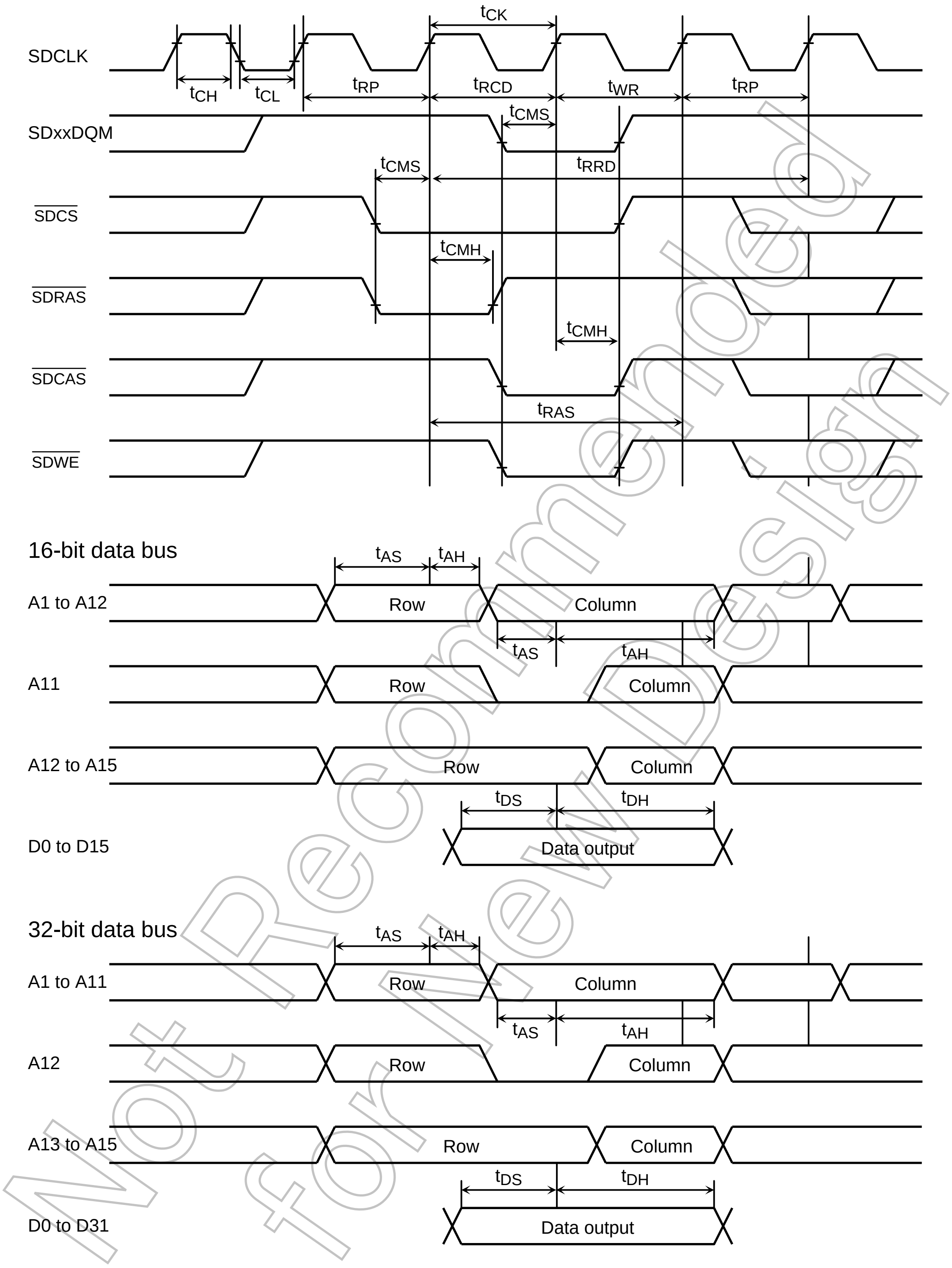
AC measuring conditions

- Output level: High = 0.7 VCC, Low = 0.3 VCC, C_L = 50 pF
- Input level: High = 0.9 VCC, Low = 0.1 VCC

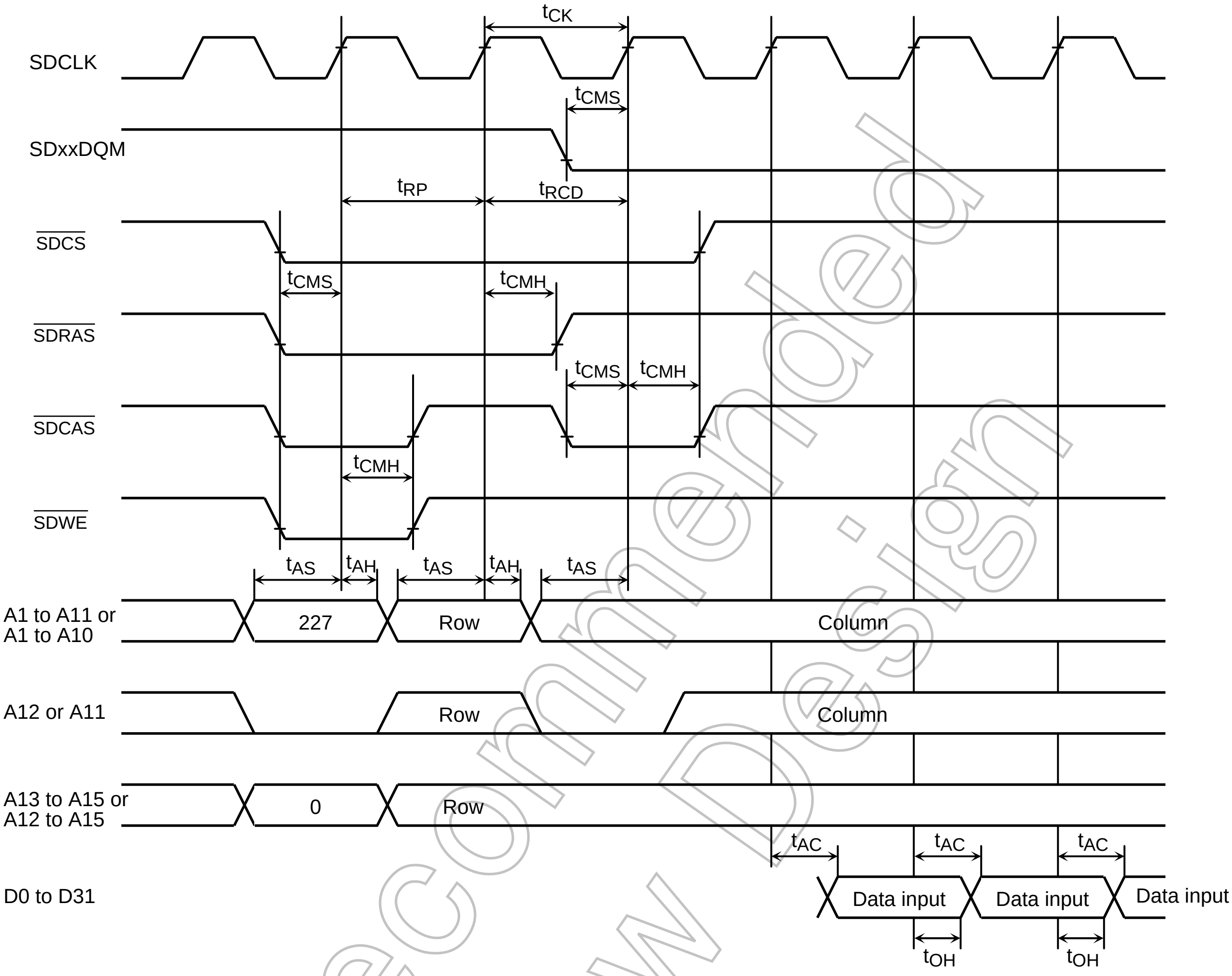
(1) SDRAM read timing (CPU access or LCDC normal access)



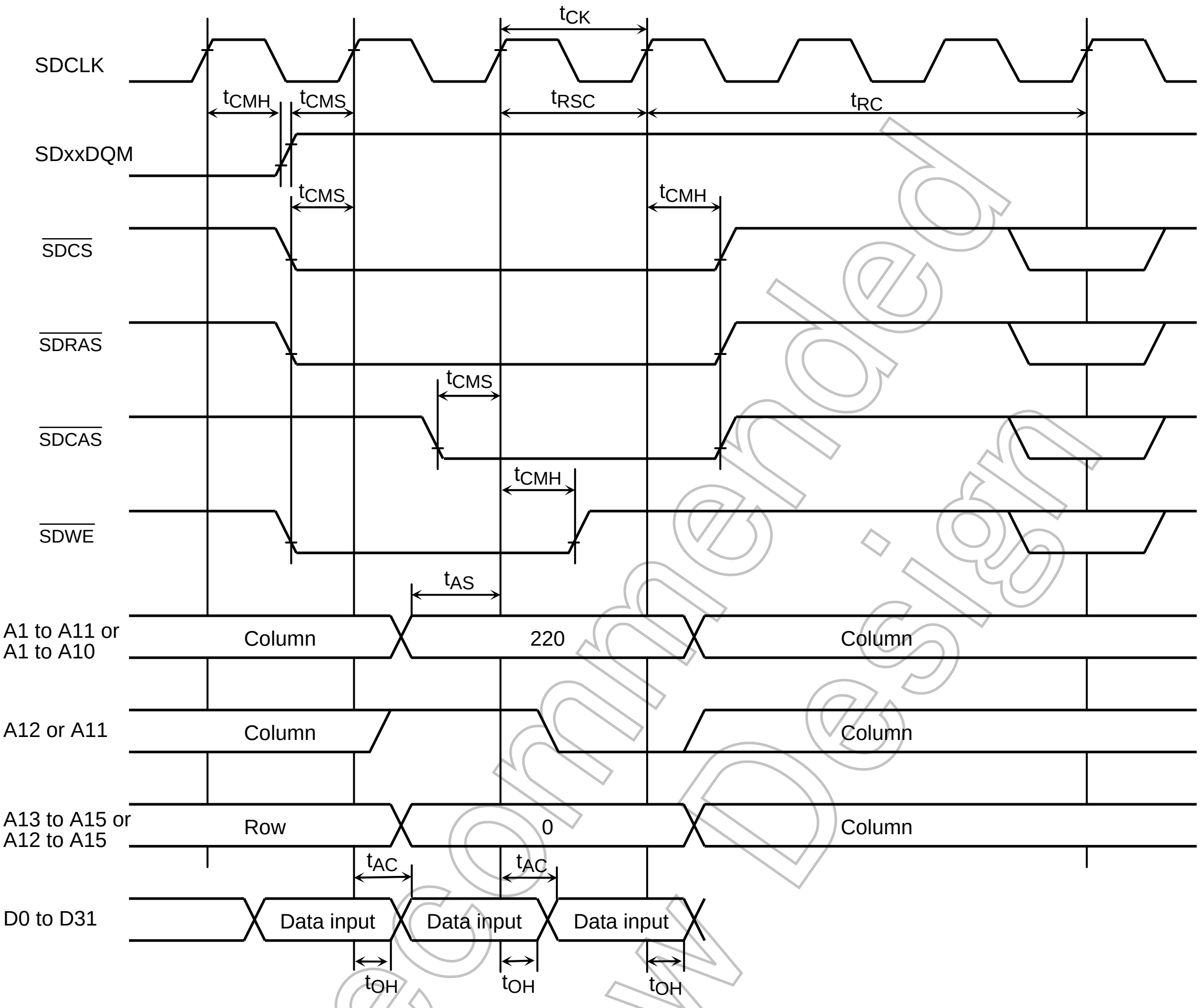
(2) SDRAM write timing (CPU access)



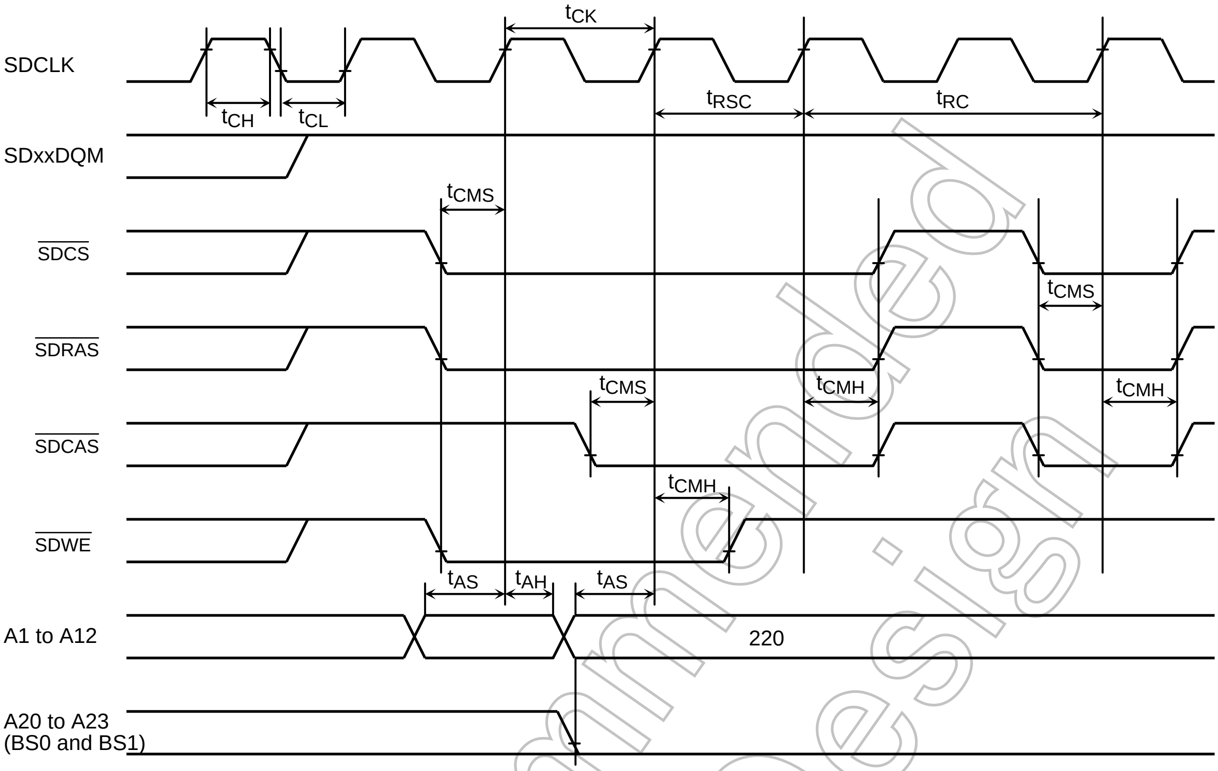
(3) SDRAM burst read timing (Start of burst cycle)



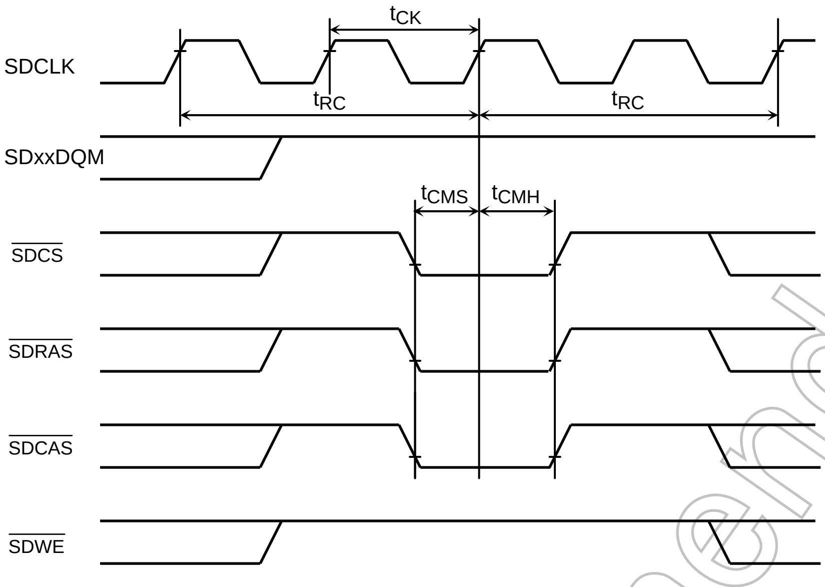
(4) SDRAM burst read timing (End of burst cycle)



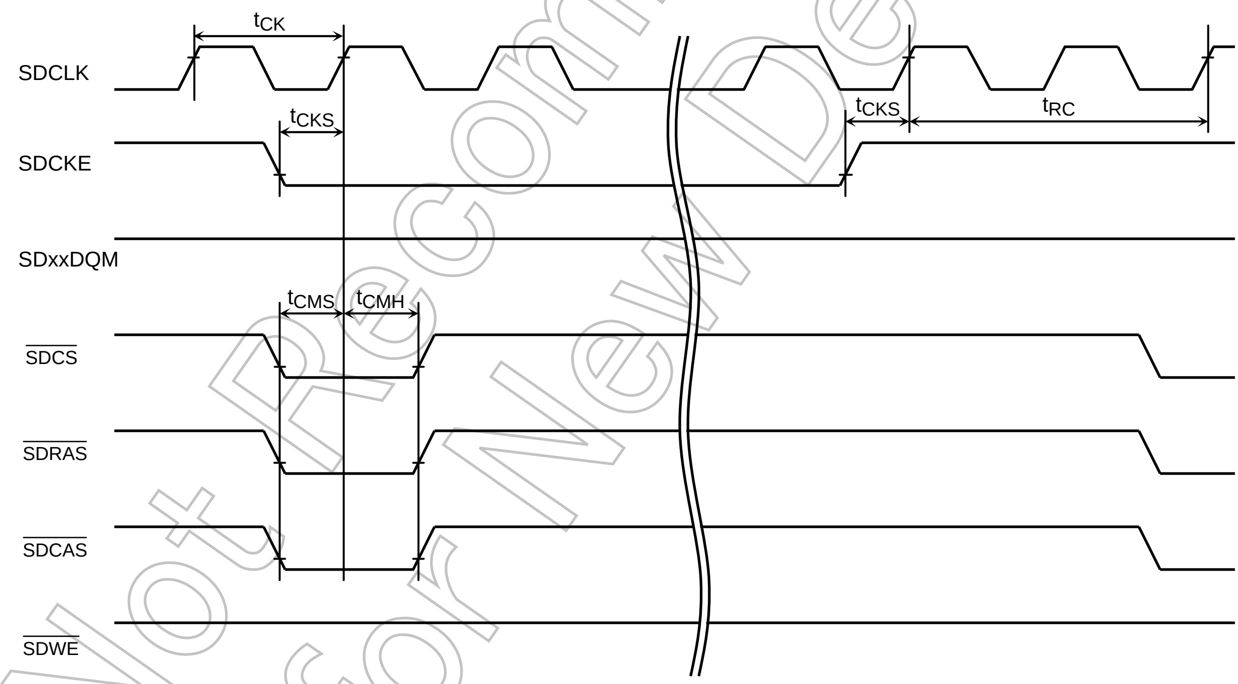
(5) SDRAM initialize timing



(6) SDRAM refresh timing



(7) SDRAM self refresh timing



4.3.4 NAND Flash Controller AC Characteristics

No.	Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
			Min	Max				
1	$\overline{\text{NDRE}}$ low width	t_{RP}	(1 + n) T – 12		38	43.5	62	ns
2	$\overline{\text{NDRE}}$ data access time	$t_{REA\ (3.0\ V)}$		(1 + n) T – 25	25	30.5	–	
		$t_{REA\ (2.7\ V)}$		(1 + n) T – 30	–	–	44	
3	Read data hold time	t_{OH}	0		0	0	0	
4	$\overline{\text{NDWE}}$ low width	t_{WP}	(0.75 + n) T – 20		17.5	21.6	35.5	
5	Write data setup time	t_{DS}	(3.25 + n) T – 30		132.5	150.3	210.5	
6	Write data hold time	t_{DH}	0.25 T – 2		10.5	11.8	16.5	

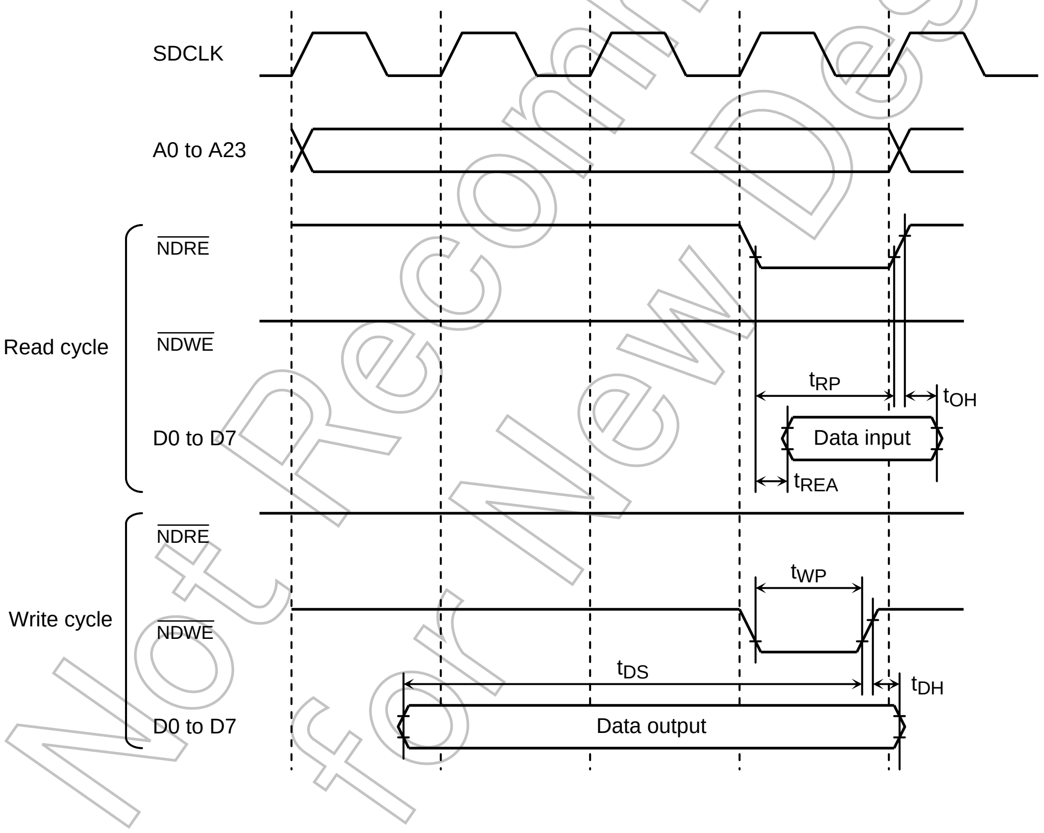
AC measuring conditions

- Output level: High = 0.7 VCC, Low = 0.3 VCC, C_L = 50 pF
- Input level: High = 0.9 VCC, Low = 0.1 VCC

Note 1: The "n" shown in "Variable" refers to the wait number which is set to NDnFSPR<SPW3:0> register.

Example: When NDnFSPR<SPW3:0> = "0001", t_{RP} = (1 + n) T – 12 = 2T – 12

Note 2: The figures in the "Variable" column cover the whole VCC range (2.7 to 3.6V).
Exceptions are shown by the VCC (min), "(3.0 V)" or " (2.7 V)", added to the "Symbol" column.



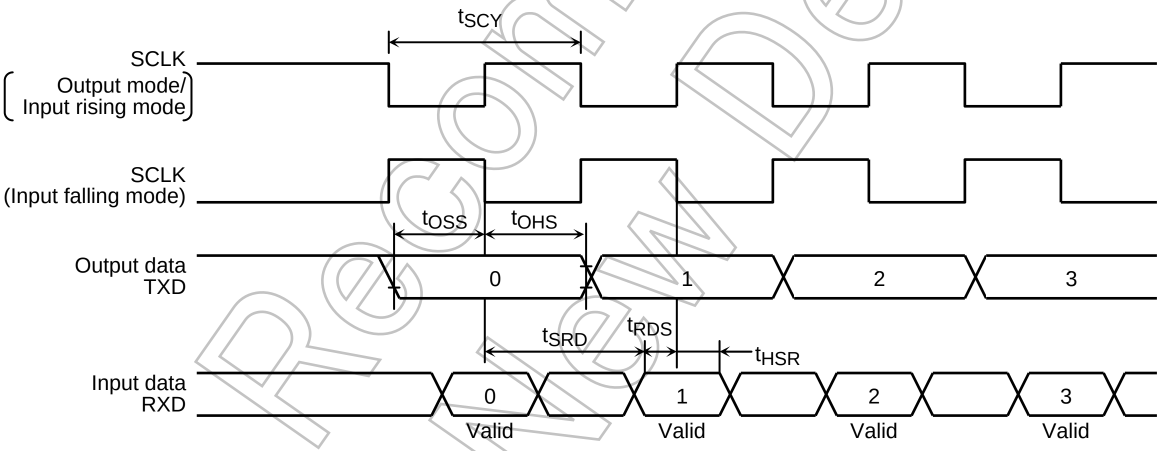
4.3.5 Serial Channel Timing

(1) SCLK input mode (I/O interface mode)

Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
		Min	Max				
SCLK cycle	t_{SCY}	16T		0.8	0.888	1.184	µs
Output data → SCLK rising/falling	t_{OSS}	$t_{SCY}/2 - 4T - 110$		90	114	186	ns
SCLK rising/falling → Output data hold	t_{OHS}	$t_{SCY}/2 + 2T + 0$		500	554	740	
SCLK rising/falling → Input data hold	t_{HSR}	3 T + 10		160	175	232	
SCLK rising/falling → Input data valid	t_{SRD}		$t_{SCY} - 0$	800	888	1184	
Input data valid → SCLK rising/falling	t_{RDS}	0		0	0	0	

(2) SCLK output mode (I/O Interface mode)

Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
		Min	Max				
SCLK cycle (Programmable)	t_{SCY}	16 T	8192T	0.8	0.888	1.184	µs
Output data → SCLK rising/falling	t_{OSS}	$t_{SCY}/2 - 40$		360	404	552	ns
SCLK rising/falling → Output data hold	t_{OHS}	$t_{SCY}/2 - 40$		360	404	552	
SCLK rising/falling → Input data hold	t_{HSR}	0		0	0	0	
SCLK rising/falling → Input data valid	t_{SRD}		$t_{SCY} - 1T - 180$	570	654	967	
Input data valid → SCLK rising/falling	t_{RDS}	1 T + 180		230	233	253	

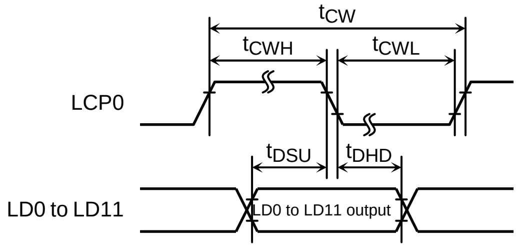


4.3.6 Interrupt Operation

Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
		Min	Max				
INT0 to INT5 low width	t_{INTAL}	4 T + 40		240	262	336	ns
INT0 to INT5 high width	t_{INTAH}	4 T + 40		240	262	336	

4.3.7 LCD Controller (SR mode)

Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
		Min	Max				
LCP0 clock period (= tm)	t_{CW}	2 T		100	111	148	ns
LCP0 high width	t_{CWH}	0.5 tm – 12		38	43.5	62	
LCP0 low width	t_{CWL}	0.5 tm – 12		38	43.5	62	
Data valid → LCP0 falling	t_{DSU}	0.5 tm – 20		30	35.5	54	
LCP0 falling → Data hold	t_{DHD}	0.5 tm – 5		45	50.5	69	

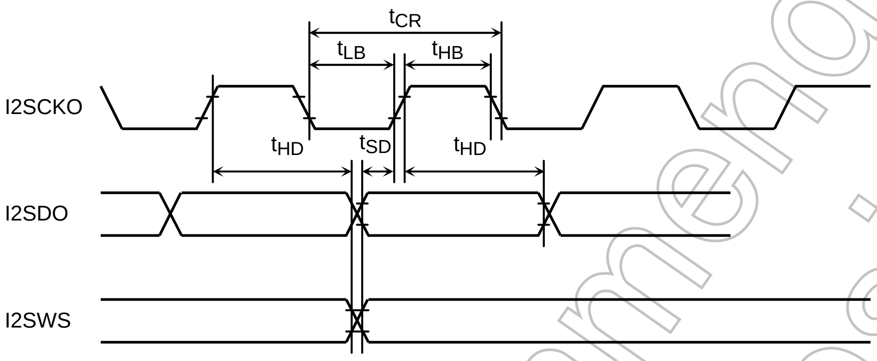


4.3.8 I^{2}S Timing (I^{2}S, SIO Mode)

Parameter	Symbol	Variable		40 MHz	36 MHz	27 MHz	Unit
		Min	Max				
I2SCKO clock period	t_{CR}	T		50	55	74	ns
I2SCKO high width	t_{HB}	0.5 t_{CR} − 15		10	12	22	
I2SCKO low width	t_{LB}	0.5 t_{CR} − 15		10	12	22	
I2SDO, I2SWS setup time	t_{SD}	0.5 t_{CR} − 15		10	12	22	
I2SDO, I2SWS hold time	t_{HD}	0.5 t_{CR} − 5		20	22	32	

AC measuring conditions

- Output level: High = 0.7 VCC, Low = 0.3 VCC, C_L = 10 pF

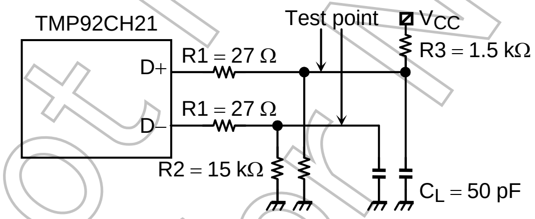


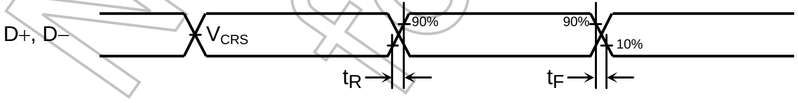
4.3.9 USB Timing (Full-speed)

V_{CC} = 3.3 ± 0.3 V/f_{USB} = 48 MHz/Ta = −20 to 70°C

Parameter	Symbol	Min	Max	Unit
Rising time for D+, D−	t_R	4	20	ns
Falling time for D+, D−	t_F	4	20	
Output signal crossover voltage	V_{CRS}	1.3	2.0	V

AC measuring conditions





4.4 AD Conversion Characteristics

Parameter	Symbol	Min	Typ.	Max	Unit
Analog reference voltage (+)	V_{REFH}	$V_{CC} - 0.2$	V_{CC}	V_{CC}	V
Analog reference voltage (−)	V_{REFL}	V_{SS}	V_{SS}	$V_{SS} + 0.2$	
AD converter power supply voltage	AV_{CC}	V_{CC}	V_{CC}	V_{CC}	
AD converter ground	AV_{SS}	V_{SS}	V_{SS}	V_{SS}	
Analog input voltage	AV_{IN}	V_{REFL}		V_{REFH}	
Analog current for analog reference voltage <VREFON> = 1	I_{REF}		0.8	1.35	mA
Analog current for analog reference voltage <VREFON> = 0			0.02	5.0	μA
Total error (Quantize error of ± 0.5 LSB is included.)	E_T		±1.0	±4.0	LSB

Note 1: 1LSB = (VREFH – VREFL) / 1024 [V]

Note 2: Minimum frequency for operation

AD converter operation is guaranteed only when using fc (high-frequency oscillator). fs is not guaranteed. However, operation is guaranteed if the clock frequency selected by the clock gear is over 4MHz.

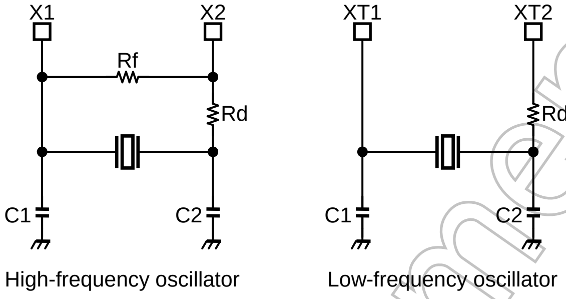
Note 3: The value for Icc includes the current which flows through the AV_{CC} pin.

4.5 Recommended Oscillation Circuit

The TMP92CH21 has been evaluated by the oscillator vender below. Use this information when selecting external parts.

Note: The total load value of the oscillator is the sum of external loads (C1 and C2) and the floating load of the actual assembled board. There is a possibility of operating error when using C1 and C2 values in the table below. When designing the board, design the minimum length pattern around the oscillator. We also recommend that oscillator evaluation be carried out using the actual board.

(1) Connection example



(2) TMP92CH21FG Recommended ceramic oscillator

The TMP92CH21FG recommend the high-frequency oscillators by Murata Manufacturing Co., Ltd and TDK Corporation.

Please refer to the following URL;

・Murata Manufacturing Co., Ltd

http://www.murata.com/

・TDK Corporation

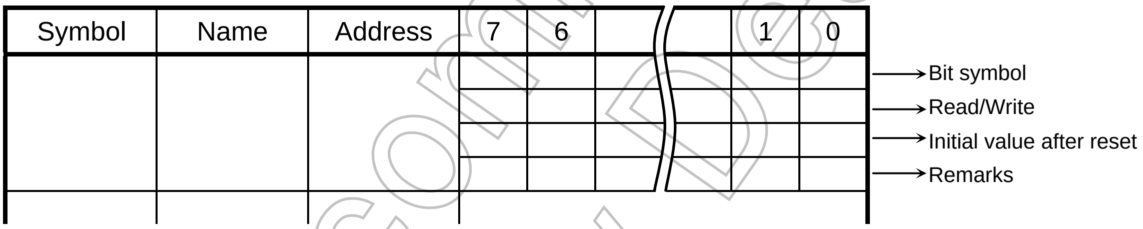
http://www.tdk.co.jp/tetop01/

5. Table of Special function registers (SFRs)

The SFRs include the I/O ports and peripheral control registers allocated to the 4-Kbyte 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) I²S

Table layout



Note: "Prohibit RMW" in the table means that you cannot use RMW instructions on these registers.

Example: When setting bit0 only of the register P0CR, the instruction "SET 0, (PxCR)" cannot be used. The LD (transfer) instruction must be used to write all eight bits.

Read/Write

R/W: Both read and write are possible.
R: Only read is possible.
W: Only write is possible.
W*: Both read and write are possible (when this bit is read as1)
Prohibit RMW: Read-modify-write instructions are prohibited. (The EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instructions are read-modify-write instructions.)
R/W*: Read-modify-write is prohibited when controlling the pull-up resistor.

Table 5.1 I/O Register Address Map

[1] Port

Address	Name	Address	Name	Address	Name	Address	Name
0000H		0010H	P4	0020H	P8	0030H	PC
1H		1H		1H	P8FC2	1H	
2H		2H		2H		2H	PCCR
3H		3H	P4FC	3H	P8FC	3H	PCFC
4H	P1	4H	P5	4H	P9	4H	
5H		5H		5H	P9FC2	5H	
6H	P1CR	6H		6H	P9CR	6H	
7H	P1FC	7H	P5FC	7H	P9FC	7H	
8H	P2	8H	P6	8H	PA	8H	
9H	P2FC2	9H		9H		9H	
AH	P2CR	AH	P6CR	AH	PACR	AH	
BH	P2FC	BH	P6FC	BH	PAFC	BH	
CH	P3	CH	P7	CH		CH	PF
DH		DH		DH		DH	PFFC2
EH	P3CR	EH	P7CR	EH		EH	PFCR
FH	P3FC	FH	P7FC	FH		FH	PFFC

Address	Name	Address	Name	Address	Name	Address	Name
0040H	PG	0050H	PK	0080H		0090H	PGDR
1H		1H		1H	P1DR	1H	
2H		2H		2H	P2DR	2H	
3H		3H	PKFC	3H	P3DR	3H	PJDR
4H		4H	PL	4H	P4DR	4H	PKDR
5H		5H		5H	P5DR	5H	PLDR
6H		6H	PLCR	6H	P6DR	6H	PMDR
7H		7H	PLFC	7H	P7DR	7H	
8H		8H	PM	8H	P8DR	8H	
9H		9H		9H	P9DR	9H	
AH		AH		AH	PADR	AH	
BH		BH	PMFC	BH		BH	
CH	PJ	CH		CH	PCDR	CH	
DH		DH		DH		DH	
EH	PJCR	EH		EH		EH	
FH	PJFC	FH		FH	PFDR	FH	

Note: Do not access un-named addresses.

[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 un-named addresses.

[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] LCDC1

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] LCDC2

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 un-named addresses.

[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] 8-bit 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] 16-bit timer

Address	Name
1180H	TB0RUN
1H	
2H	TB0MOD
3H	TB0FFCR
4H	
5H	
6H	
7H	
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

[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 un-named addresses.

[12] USB controller (1/2)

Address	Name
0500H	Descriptor-
to	RAM
067FH	(384 bytes)

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	bmRequest Type
1H	bRequest
2H	wValue_L
3H	wValue_H
4H	wIndex_L
5H	wIndex_H
6H	wLength_L
7H	wLength_H
8H	Setup Received
9H	Current_Config
AH	Standard Request
BH	Request
CH	DATASET1
DH	DATASET2
EH	USB_STATE
FH	EOP

Address	Name
07D0H	COMMAND
1H	EPx_SINGLE1
2H	
3H	EPx_BCS1
4H	
5H	
6H	INT_Control
7H	
8H	Standard Request Mode
9H	Request Mode
AH	
BH	
CH	
DH	
EH	ID_CONTROL
FH	ID_STATE

Note: Do not access un-named addresses.

[12] USB controller (2/2)

Address	Name
07E0H	Port_Status
1H	FRAME_L
2H	FRAME_H
3H	ADDRESS
4H	
5H	
6H	USBREADY
7H	
8H	Set Descriptor STALL
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
07F0H	USBINTFR1
1H	USBINTFR2
2H	USBINTFR3
3H	USBINTFR4
4H	USBINTMR1
5H	USBINTMR2
6H	USBINTMR3
7H	USBINTMR4
8H	USBCR1
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access un-named addresses.

[13] 10-bit 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 un-named addresses.

[17] NAND flash controller

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	ND0FDTR,
to	ND1FDTR
1EFFH	

Address	Name
1CB0H	ND0ECCRD
to	ND1ECCRD
1CB5H	

Address	Name
01C0H	NDCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access un-named addresses.

[18] I²S

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 un-named addresses.

(1) I/O ports (1/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1	Port 1	0004H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Data from external port (Output latch register is cleared to "0")							
P2	Port 2	0008H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			Data from external port (Output latch register is cleared to "0")							
P3	Port 3	000CH	P37	P36	P35	P34	P33	P32	P31	P30
			R/W							
			Data from external port (Output latch register is cleared to "0")							
P4	Port 4	0010H	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			0	0	0	0	0	0	0	0
P5	Port 5	0014H	P57	P56	P55	P54	P53	P52	P51	P50
			R/W							
			0	0	0	0	0	0	0	0
P6	Port 6	0018H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			Data from external port (Output latch register is cleared to "0")							
P7	Port 7	001CH		P76	P75	P74	P73	P72	P71	P70
				R/W						
				Data from external port (Output latch register is set to "1")		0	0	Data from external port (Output latch register is set to "1")		1
P8	Port 8	0020H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			1	1	1	1	1	0/1	1	1
P9	Port 9	0024H	P97	P96	P95	P94	P93	P92	P91	P90
			R		R/W					
			Data from external port		0	Data from external port (Output latch register is set to "1")				
PA	Port A	0028H	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R							
			Data from external port							
PC	Port C	0030H	PC7	PC6			PC3	PC2	PC1	PC0
			R/W				R/W			
			Data from external port (Output latch register is set to "1")				Data from external port (Output latch register is set to "1")			
PF	Port F	003CH	PF7					PF2	PF1	PF0
			R/W					R/W		
			1					Data from external port (Output latch register is set to "1")		
PG	Port G	0040H					PG3	PG2	PG1	PG0
							R			
							Data from external port			
PJ	Port J	004CH	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
			R/W							
			1	Data from external port (Output latch register is set to "1")		1	1	1	1	1
PK	Port K	0050H					PK3	PK2	PK1	PK0
							R/W			
							0	0	0	0
PL	Port L	0054H	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
			R/W							
			Data from external port (Output latch register is set to "0")				0	0	0	0
PM	Port M	0058H						PM2	PM1	
								R/W		
								1	1	

(1) I/O ports (2/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1CR	Port 1 control register	0006H (Prohibit RMW)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P1FC	Port 1 function register	0007H (Prohibit RMW)								P1F
										W
										0/1
										0:Port 1:Data bus (D8 to D15)
P2CR	Port 2 control register	000AH (Prohibit RMW)	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P2FC	Port 2 function register	000BH (Prohibit RMW)								P2F
										W
										0/1
										0:Port 1:Data bus (D16 to D23)
P2FC2	Port 2 function register2	0009H (Prohibit RMW)	P27F2	P26F2	P25F2	P24F2	P23F2	P22F2	P21F2	P20F2
			W							
			0	0	0	0	0	0	0	0
			0: CMOS output 1: Open-drain output							
P3CR	Port 3 control register	000EH (Prohibit RMW)	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P3FC	Port 3 function register	000FH (Prohibit RMW)					–	–	–	P3F
							W			
							0	0	0	0/1
							Always write “0”			0:Port 1:Data bus (D24 to D31)
P4FC	Port 4 function register	0013H (Prohibit RMW)	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			0: Port 1: Address bus (A0 to A7)							
P5FC	Port 5 function register	0017H (Prohibit RMW)	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			0: Port 1: Address bus (A8 to A15)							
P6CR	Port 6 control register	001AH (Prohibit RMW)	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P6FC	Port 6 function register	001BH (Prohibit RMW)	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			0: Port 1: Address bus (A16 to A23)							

(1) I/O ports (3/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P7CR	Port 7 control register	001EH (Prohibit RMW)		P76C	P75C			P72C	P71C	
				W	W			W	W	
				0	0			0	0	
				0: Input port, $\overline{WAIT}$ 1: Output port	0: Input port, NDR/$\overline{B}$ 1: Output port, R/$\overline{W}$			0: Input port, 1: Output port, $\overline{NDWE}$ @ <P72> = 0, $\overline{WRLH}$ @ <P72> = 1	0: Input port 1: Output port, $\overline{NDRE}$ @ <P71> = 0, $\overline{WRLL}$ @ <P71> = 0	
P7FC	Port 7 function register	001FH (Prohibit RMW)		P76F	P75F	P74F	P73F	P72F	P71F	P70F
				W						
				0	0	0	0	0	0	0/1
				0: Port 1: $\overline{WAIT}$	0: Port 1: NDR/$\overline{B}$, R/$\overline{W}$	0: Port 1: EA25	0: Port 1: EA24	0: Port 1: $\overline{NDWE}$, $\overline{WRLU}$	0: Port 1: $\overline{NDRE}$, $\overline{WRLL}$	0: Port 1: $\overline{RD}$
P8FC	Port 8 function register	0023H (Prohibit RMW)	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: $\overline{CSZE}$	0: Port 1: $\overline{CSZD}$	0: Port, $\overline{WRUU}$ 1: $\overline{CSZC}$, $\overline{ND1CE}$	0: Port, $\overline{WRUL}$ 1: $\overline{CSZB}$, $\overline{ND0CE}$	0: Port 1: $\overline{CS3}$	0: Port, $\overline{CSZA}$ 1: $\overline{CS2}$, $\overline{SDCS}$	0: Port 1: $\overline{CS1}$	0: Port 1: $\overline{CS0}$
P8FC2	Port 8 function register2	0021H (Prohibit RMW)	P87F2	P86F2	P85F2	P84F2	–	P82F2	P81F2	–
			W							
			0	0	0	0	0	0	0	0
			0: <P87F> 1: $\overline{SRUUB}$	0: <P86F> 1: $\overline{SRULB}$	0: Port, $\overline{CSZC}$ 1: $\overline{WRUU}$, $\overline{ND1CE}$	0: Port, $\overline{CSZB}$ 1: $\overline{WRUL}$, $\overline{ND0CE}$	Always write “0”	0: Port 1: $\overline{CSZA}$	0: <P81F> 1: $\overline{SDCS}$	Always write “0”
P9CR	Port 9 control register	0026H (Prohibit RMW)			P95C	P94C	P93C	P92C	P91C	P90C
					W					
					0	0	0	0	0	0
					0: Output port, LGOE2 1: CLK32KO	0: Input port, LGOE1 1: Output port	0:Input port, LGOE0 1: Output port	0:Input port, SCLK0, $\overline{CTS0}$ 1: I2SWS, SCLK0	0: Input port, RXD0, I2SDO 1: Output port	0: Input port, I2SCKO 1: Output port, TXD0
P9FC	Port 9 function register	0027H (Prohibit RMW)	P97F	P96F	P95F	P94F	P93F	P92F	P91F	P90F
			W							
			0	0	0	0	0	0	0	0
			0: Input port 1: INT5	0: Input port 1: INT4	0: Output port, CLK32KO 1: LGOE2	0: Port 1: LGOE1	0: Port 1: LGOE0	0: Port, SCLK0, $\overline{CTS0}$ 1: I2SWS, SCLK0	0: Port, RXD0 1: I2SDO	0: Port 1: I2SCKO, TXD0
P9FC2	Port 9 function register2	0025H (Prohibit RMW)								P90FC2
										W
										0
										0: CMOS 1: Open drain

(1) I/O ports (4/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PACR	Port A control register	002AH (Prohibit RMW)		PA6C	PA5C	PA4C	PA3C			
				W						
				0	0	0	0			
				0: Input port or key-in 1: LD11 to LD8 output						
PAFC	Port A function register	002BH (Prohibit RMW)	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
			W							
			0	0	0	0	0	0	0	0
			0: Key-in disable 1: Key-in enable							
PCCR	Port C control register	0032H (Prohibit RMW)	PC7C	PC6C			PC3C	PC2C	PC1C	PC0C
			W				W			
			0	0			0	0	0	0
			0: Input port, $\overline{\text{CSZF}}$ 1: Output port, LCP1	0: Input port, KO8 1: Output port, LDIV			0: Input port, INT3 1: Output port	0: Input port, INT2 1: Output port, TB0OUT0	0: Input port, INT1 1: Output port, TA3OUT	0: Input port, INT0 1: Output port, TA1OUT
PCFC	Port C function register	0033H (Prohibit RMW)	PC7F	PC6F			PC3F	PC2F	PC1F	PC0F
			W				W			
			0	0			0	0	0	0
			0: Port 1: $\overline{\text{CSZF}}$, LCP1	0: 3states 1: KO8, LDIV			0: Port 1: INT3	0: Port 1: INT2, TB0OUT0	0: Port 1: INT1, TA3OUT	0: Port 1: INT0, TA1OUT
PFCR	Port F control register	003EH (Prohibit RMW)						PF2C	PF1C	PF0C
								W		
								0	0	0
								0: Input port, SCLK1, $\overline{\text{CTS1}}$, SCLK0, $\overline{\text{CTS0}}$ 1: Output port, SCLK0	0: Input port, RXD0, RXD1, 1: Output port	0: Input port, TXD1 1: Output port, TXD0
PFFC	Port F function register	003FH (Prohibit RMW)	PF7F					PF2F	PF1F	PF0F
			W					W		
			1					0	0	0
			0: Port 1: SDCKE					0: Port, SCLK1, $\overline{\text{CTS1}}$, SCLK0, $\overline{\text{CTS0}}$ 1: SCLK0, SCLK1	Select RXD0 pin 0: Port F1 1: Port 91	0: Port 1: TXD1, TXD0
PFFC2	Port F function register2	003DH (Prohibit RMW)								PF0F2
										W
										0
										Output buffer 0: CMOS 1: Open-drain

(1) I/O ports (5/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PJCR	Port J control register	004EH (Prohibit RMW)		PJ6C	PJ5C					
				W						
				0	0					
				0:Input 1: Output						
PJFC	Port J function register	004FH (Prohibit RMW)	PJ7F	PJ6F	PJ5F	PJ4F	PJ3F	PJ2F	PJ1F	PJ0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: SDCKE	0: Port 1: NDCLE, SDUUDQM	0: Port 1: NDALE, SDULDQM	0: Port 1: SDLUDQM	0: Port 1: SDLLDQM	0: Port 1: $\overline{SDWE}$, $\overline{SDWR}$	0: Port 1: $\overline{SDCAS}$, $\overline{SRLUB}$	0: Port 1: $\overline{SRRAS}$, $\overline{SRLLB}$
PKFC	Port K function register	0053H (Prohibit RMW)					PK3F	PK2F	PK1F	PK0F
							W			
							0	0	0	0
							0: Port 1: LBCD	0: Port 1: LFR	0: Port 1: LLP	0: Port 1: LCP0
PLCR	Port L control register	0056H (Prohibit RMW)	PL7C	PL6C	PL5C	PL4C				
			W							
			0	0	0	0				
			0: Input 1: Output							
PLFC	Port L function register	0057H (Prohibit RMW)	PL7F	PL6F	PL5F	PL4F	PL3F	PL2F	PL1F	PL0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: Data bus for LCDC (LD7 to LD0)							
PMFC	Port M function register	005BH (Prohibit RMW)						PM2F	PM1F	
								W		
								0	0	
								0: Port 1: $\overline{ALARM}$ $\overline{MLDALM}$	0: Port 1: MLDALM output	

(1) I/O ports (6/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1DR	Port 1 drive register	0081H	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P2DR	Port 2 drive register	0082H	P27D	P26D	P25D	P24D	P23D	P22D	P21D	P20D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P3DR	Port 3 drive register	0083H	P37D	P36D	P35D	P34D	P33D	P32D	P31D	P30D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P4DR	Port 4 drive register	0084H	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P5DR	Port 5 drive register	0085H	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P6DR	Port 6 drive register	0086H	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P7DR	Port 7 drive register	0087H		P76D	P75D	P74D	P73D	P72D	P71D	P70D
				R/W						
				1	1	1	1	1	1	1
				Input/Output buffer drive register for standby mode						
P8DR	Port 8 drive register	0088H	P87D	P86D	P85D	P84D	P83D	P82D	P81D	P80D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
P9DR	Port 9 drive register	0089H	P97D	P96D	P95D	P94D	P93D	P92D	P91D	P90D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PADR	Port A drive register	008AH	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PCDR	Port C drive register	008CH	PC7D	PC6D			PC3D	PC2D	PC1D	PC0D
			R/W				R/W			
			1	1			1	1	1	1
			Input/Output buffer drive register for standby mode				Input/Output buffer drive register for standby mode			

(1) I/O ports (7/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PFDR	Port F drive register	008FH	PF7D			PF4D	PF3D	PF2D	PF1D	PF0D
			R/W			R/W				
			1			1	1	1	1	1
			Input/Output buffer drive register for standby mode			Input/Output buffer drive register for standby mode				
PGDR	Port G drive register	0090H					PG3D	PG2D		
							R/W			
							1	1		
							Input/Output buffer drive register for standby mode			
PJDR	Port J drive register	0093H	PJ7D	PJ6D	PJ5D	PJ4D	PJ3D	PJ2D	PJ1D	PJ0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PKDR	Port K drive register	0094H					PK3D	PK2D	PK1D	PK0D
							R/W			
							1	1	1	1
							Input/Output buffer drive register for standby mode			
PLDR	Port L drive register	0095H	PL7D	PL6D	PL5D	PL4D	PL3D	PL2D	PL1D	PL0D
			R/W							
			1	1	1	1	1	1	1	1
			Input/Output buffer drive register for standby mode							
PMDR	Port M drive register	0096H						PM2D	PM1D	
								R/W		
								1	1	
								Input/Output buffer drive register for standby mode		

(2) Interrupt control (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
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 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	00D5H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB01	INTTB0 & INTTB1 enable	00D8H	INTTB1 (TMRB1)				INTTB0 (TMRB0)			
			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		
			Always write “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		
			Always write “0”				0	0	0	0
INTEALM01	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
INTEALM23	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/W		
			Always write “0”				0	0	0	0
INTERTC	INTRTC enable	00E8H	–				INTRTC			
			–	–	–	–	IRC	IRM2	IRM1	IRM0
							R	R/W		
			Always write “0”				0	0	0	0
INTEKEY	INTKEY enable	00E9H	–				INTKEY			
			–	–	–	–	IKC	IKM2	IKM1	IKM0
							R	R/W		
			Always write “0”				0	0	0	0
INTELCD	INTLCD enable	00EAH	–				INTLCD			
			–	–	–	–	ILCD1C	ILCDM2	ILCDM1	ILCDM0
							R	R/W		
			Always write “0”				0	0	0	0
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	INTNDF0 & INTNDF1 enable	00ECH	INTNDF1				INTNDF0			
			IND1C	IND1M2	IND1M1	IND1M0	IND0C	IND0M2	IND0M1	IND0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 enable	00EEH	–				INTP0			
			–	–	–	–	IP0C	IP0M2	IP0M1	IP0M0
							R	R/W		
			Always write “0”				0	0	0	0

(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	00F5H (Prohibit RMW)	–						IR1LE	IR0LE
			W						W	W
			0						1	1
			Always write “0”.						0: INTRX1 edge mode 1: INTRX1 level mode	0: INTRX0 edge mode 1: INTRX0 level mode
IIMC	Interrupt input mode control	00F6H (Prohibit RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	–
			W						R/W	
			0	0	0	0	0	0	0	0
			INT5 edge 0: Rising 1: Falling	INT4 edge 0: Rising 1: Falling	INT3 edge 0: Rising 1: Falling	INT2 edge 0: Rising 1: Falling	INT1 edge 0: Rising 1: Falling	INT0 edge 0: Rising 1: Falling	0: INT0 edge mode 1:INT0 level mode	Always write “0”.
INTWDT	INTWD enable	00F7H	–				INTWD			
			–	–	–	–	ITCWD	–	–	–
							R			
			Always write “0”				0	–	–	–
INTCLR	Interrupt clear control	00F8H (Prohibit RMW)	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt vector							

(2) Interrupt control (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	0100H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 start vector	0101H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 start vector	0102H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 start vector	0103H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
DMA4V	DMA4 start vector	0104H			DMA4V5	DMA4V4	DMA4V3	DMA4V2	DMA4V1	DMA4V0
					R/W					
					0	0	0	0	0	0
					DMA4 start vector					
DMA5V	DMA5 start vector	0105H			DMA5V5	DMA5V4	DMA5V3	DMA5V2	DMA5V1	DMA5V0
					R/W					
					0	0	0	0	0	0
					DMA5 start vector					
DMA6V	DMA6 start vector	0106H			DMA6V5	DMA6V4	DMA6V3	DMA6V2	DMA6V1	DMA6V0
					R/W					
					0	0	0	0	0	0
					DMA6 start vector					
DMA7V	DMA7 start vector	0107H			DMA7V5	DMA7V4	DMA7V3	DMA7V2	DMA7V1	DMA7V0
					R/W					
					0	0	0	0	0	0
					DMA7 start vector					
DMAB	DMA burst	0108H	DBST7	DBST6	DBST5	DBST4	DBST3	DBST2	DBST1	DBST0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request on burst mode							
DMAR	DMA request	0109H (Prohibit RMW)	DREQ7	DREQ6	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1	DREQ0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request in software							

(3) Memory controller (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B0CSL	BLOCK0 CS/WAIT control register low	0140H (Prohibit RMW)		B0WW2	B0WW1	B0WW0		B0WR2	B0WR1	B0WR0
				W				W		
				0	1	0		0	1	0
				Write waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1+ N) waits 111: 4 waits Others: Reserved				Read waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1+ N) waits 111: 4 waits Others: Reserved		
B0CSH	BLOCK0 CS/WAIT control register high	0141H (Prohibit RMW)	B0E	–	–	B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
			W							
			0	0	0	0	0	0	0/1	0/1
			CS select 0: Disable 1: Enable	Always write “0”.	Always write “0”.	Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved	
B1CSL	BLOCK1 CS/WAIT control register low	0144H (Prohibit RMW)		B1WW2	B1WW1	B1WW0		B1WR2	B1WR1	B1WR0
				W				W		
				0	1	0		0	1	0
				Write waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1+ N) waits 111: 4 waits Others: Reserved				Read waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1+ N) waits 111: 4 waits Others: Reserved		
B1CSH	BLOCK1 CS/WAIT control register high	0145H (Prohibit RMW)	B1E	–	–	B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
			W							
			0	0	0	0	0	0	0/1	0/1
			CS select 0: Disable 1: Enable	Always write “0”.	Always write “0”.	Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: SDRAM		Data bus width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved	
B2CSL	BLOCK2 CS/WAIT control register low	0148H (Prohibit RMW)		B2WW2	B2WW1	B2WW0		B2WR2	B2WR1	B2WR0
				W				W		
				0	1	0		0	1	0
				Write waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1+ N) waits 111: 4 waits Others: Reserved				Read waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1+ N) waits 111: 4 waits Others: Reserved		
B2CSH	BLOCK2 CS/WAIT control register high	0149H (Prohibit RMW)	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: 16 MB 1: Sets area	Always write “0”.	Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: SDRAM		Data bus width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved	

(3) Memory controller (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B3CSL	BLOCK3 CS/WAIT control register low	014CH (Prohibit RMW)		B3WW2	B3WW1	B3WW0		B3WR2	B3WR1	B3WR0
				W				W		
				0	1	0		0	1	0
				Write waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1 + N) waits 111: 4 waits Others: Reserved				Read waits 001: 0 waits 010: 1 wait 101: 2 waits 110: 3 waits 011: (1 + N) waits 111: 4 waits Others: Reserved		
B3CSH	BLOCK3 CS/WAIT control register high	014DH (Prohibit RMW)	B3E	–	–	B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
			W							
			0	0	0	0	0	0	0/1	0/1
			CS select 0: Disable 1: Enable	Always write “0”.	Always write “0”.	Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved	
BEXCSL	BLOCK EX CS/WAIT control register low	0158H (Prohibit RMW)		BEXWW2	BEXWW1	BEXWW0		BEXWR2	BEXWR1	BEXWR0
				W				W		
				0	1	0		0	1	0
				Write waits 001: 2 waits 010: 1 wait 101: 2 waits 110: 2 waits 011: (1 + N) waits Others: Reserved				Read waits 001: 2 waits 010: 1 wait 101: 2 waits 110: 2 waits 011: (1 + N) waits Others: Reserved		
BEXCSH	BLOCK EX CS/WAIT control register high	0159H (Prohibit RMW)					BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
							W			
							0	0	0/1	0/1
							00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved	
PMEMCR	Page ROM control register	0166H				OPGE	OPWR1	OPWR0	PR1	PR0
						R/W				
						0	0	0	1	0
						ROM page access 0: Disable 1: Enable	Wait number on page 00: 1 CLK (n-1-1-1 mode) 01: 2 CLK (n-2-2-2 mode) 10: 3 CLK (n-3-3-3 mode) 11: Reserved		Byte number in page 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	

(3) Memory controller (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MAMR0	Memory address mask register 0	0142H	M0V20	M0V19	M0V18	M0V17	M0V16	M0V15	M0V14-9	M0V8
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR0	Memory start address register 0	0143H	M0S23	M0S22	M0S21	M0S20	M0S19	M0S18	M0S17	M0S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR1	Memory address mask register 1	0146H	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	MV15-9	M1V8
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR1	Memory start address register 1	0147H	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR2	Memory address mask register 2	014AH	M2V22	M2V21	M2V20	M2V19	M2V18	M2V17	M2V16	M2V15
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR2	Memory start address register 2	014BH	M2S23	M2S22	M2S21	M2S20	M2S19	M2S18	M2S17	M2S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR3	Memory address mask register 3	014EH	M3V22	M3V21	M3V20	M3V19	M3V18	M3V17	M3V16	M3V15
			R/W							
			1	1	1	1	1	1	1	1
			0:Compare enable 1:Compare disable							
MSAR3	Memory start address register 3	014FH	M3S23	M3S22	M3S21	M3S20	M3S19	M3S18	M3S17	M3S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
BROMCR	Boot ROM control register	0167H							ROMLESS	VACE
									R/W	
									0/1	1/0
									Boot ROM 0: Use 1: Bypass	Vector address 0: Disable 1: Enable

(4) MMU (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALPX	LOCALX register for program	01D0H	LXE			X4	X3	X2	X1	X0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				
LOCALPY	LOCALY register for program	01D1H	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				
LOCALPZ	LOCALZ register for program	01D3H	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0	0	0	0	0	0	0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						
LOCALLX	LOCALX register for LCDC	01D4H	LXE			X4	X3	X2	X1	X0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				
LOCALLY	LOCALY register for LCDC	01D5H	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				
LOCALLZ	LOCALZ register for LCDC	01D7H	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0	0	0	0	0	0	0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						
LOCALRX	LOCALX register for read	01D8H	LXE			X4	X3	X2	X1	X0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				
LOCALRY	LOCALY register for read	01D9H	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				

(4) MMU (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALRZ	LOCALZ register for read	01DBH	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0	0	0	0	0	0	0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						
LOCALWX	LOCALX register for write	01DCH	LXE			X4	X3	X2	X1	X0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-X 0: Disable 1: Enable			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)				
LOCALWY	LOCALY register for write	01DDH	LYE			Y4	Y3	Y2	Y1	Y0
			R/W			R/W				
			0			0	0	0	0	0
			Bank for LOCAL-Y 0: Disable 1: Enable			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)				
LOCALWZ	LOCALZ register for write	01DFH	LZE	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W	R/W						
			0	0	0	0	0	0	0	0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)						

(5) Clock gear, PLL

Symbol	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System clock control register 0	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 register 1	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/8 101: (Reserved) 110: (Reserved) 111: (Reserved) 100: fc/16		
SYSCR2	System clock control register 2	10E2H	–		WUPTM1	WUPTM0	HALTM1	HALTM0		
			R/W		R/W					
			0		1	0	1	1		
			Always write “0”		Warm-up timer 00: Reserved 01: 2^8/Inputted frequency 10: 2^{14}/Inputted frequency 11: 2^{16}/Inputted frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			
EMCCR0	EMC control register 0	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	High frequency oscillator driver ability 1: NORMAL 0: WEAK	Low frequency oscillator driver ability 1: NORMAL 0: WEAK
EMCCR1	EMC control register 1	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 register 2	10E5H								
PLLCR0	PLL control register 0	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 register 1	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
LCDMODE0	LCD mode 0 register	0280H	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
			R/W							
			0	0	1	0	0	0	0	0
			Display RAM 00: Internal SRAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transmission speed 00: Reserved 01: 2 × f_{SYS} 10: 4 × f_{SYS} 11: 8 × f_{SYS}		Mode setting 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 (4096) 0111: Reserved 1000: TFT 8bpp (256) 1001: TFT 12bpp (4096) Others: Reserved	
LCDMODE1	LCD mode 1 register	0281H			LLPMODE	LDINV	AUTOINV	INTMODE	LDO1	LDO0
					R/W					
					0	0	0	0	0	0
					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)	Select interrupt 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 Others: Reserved	
LCDFFP	LCD frame frequency register	0282H	FP7	FP6	FP5	FP4	FP3	FP2	FP1	FP0
			R/W							
			0	0	0	0	0	0	0	0
			Setting bit7 to bit0 f_{FP}							
LCDDVM	LCD divide FRM register	0283H	FMN7	FMN6	FMN5	FMN4	FMN3	FMN2	FMN1	FMN0
			R/W							
			0	0	0	0	0	0	0	0
			Setting DVM bit7 to bit0							
LCDSIZE	LCD size register	0284H	COM3	COM2	COM1	COM0	SEG3	SEG2	SEG1	SEG0
			R/W							
			0	0	0	0	0	0	0	0
			Common setting 0000: Reserved 0101: 200 0001: 64 0110: 240 0010: 120 0111: 320 0011: 128 1000: 480 0100: 160 Others: Reserved				Segment setting 0000: Reserved 0101: 320 0001: 64 0110: 480 0010: 128 0111: 640 0011: 160 1000: 768 0100: 256 1001: 960 Others: Reserved			
LCDCTL0	LCD control 0 register	0285H		ALL0	FRMON	–	FP9	MMULCD	FP8	START
				R/W						
				0	0	0	0	0	0	0
				Column Data setting 0: Normal 1: All display data "0"	FR divide setting 0: Disable 1: Enable	Always write "0"	f_{FP} setting bit 9	Built-in RAM LCDD setting 0: Sequential access 1: Random access	f_{FP} setting bit 8	LCDC start 0: STOP 1: START
LCDCTL1	LCD control 1 register	0286H	LCP0P	LCP1P	LBCDP				LBCDW1	LBCDW0
			R/W	R/W	R/W				R/W	R/W
			1	0	0				0	0
			LCP0 phase 0: Rising 1: Falling	LCP1 phase 0: Rising 1: Falling	LBCD phase 0: Low 1: High				LBCD width control 00: LCP1_1CLK 01: LCP1_2CLK 10: LCP1_3CLK 11: Reserved	

(6) LCD controller (2/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDSCC	LCD source clock counter register	0287H	SCC7	SCC6	SCC5	SCC4	SCC3	SCC2	SCC1	SCC0
			R/W							
			0	0	0	0	0	0	0	0
			LCDC source clock counter bit7 to bit0							
LCDCCR0	LCD clock counter register 0	0288H						PCPV2	PCPV1	PCPV0
								R/W		
								0	0	0
								Pre LCP1 CLK: LCP1 pulse number Dummy clock number until valid clock of gate driver LCP1		
LCDCCR1	LCD clock counter register 1	0289H				TLDE4	TLDE3	TLDE2	TLDE1	TLDE0
						R/W				
						0	0	0	0	0
						Set up time of LCP: SYSCLK pulse number × 8 Set up time of TFT source driver LLP signal (Offset time is 14 to 16 SYSCLK)				
LCDCCR2	LCD clock counter register 2	028AH	LLPSU7	LLPSU6	LLPSU5	LLPSU4	LLPSU3	LLPSU2	LLPSU1	LLPSU0
			R/W							
			0	0	0	0	0	0	0	0
			TFT source driver, LLP_Enable signal: f_{SYS} × 8 High width time for LLP signal							
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
			256 color STN mode RED1 level setting				256 color STN mode RED0 level setting			
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
			256 color STN mode RED3 level setting				256 color STN mode RED2 level setting			
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
			256 color STN mode RED5 level setting				256 color STN mode RED4 level setting			
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
			256 color STN mode RED7 level setting				256 color STN mode RED6 level setting			

(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
			256 color STN mode GREEN1 level setting				256 color STN mode GREEN0 level setting			
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
			256 color STN mode GREEN3 level setting				256 color STN mode GREEN2 level setting			
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
			256 color STN mode GREEN5 level setting				256 color STN mode GREEN4 level setting			
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
			256 color STN mode GREEN7 level setting				256 color STN mode GREEN6 level setting			
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
			256 color STN mode BLUE1 level setting				256 color STN mode BLUE0 level setting			
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
			256 color STN mode BLUE3 level setting				256 color STN mode BLUE2 level setting			

(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 to bit0)							
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 to bit8)							
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 to bit16)							
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 to bit0)							
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 to bit0)							
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 to bit8)							
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 to bit16)							
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 to bit0)							
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 to bit0)							
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 to bit8)							
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 to bit16)							

(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
			OE0 control gate driver of TFT panel							
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
			OE0 control gate driver of TFT panel							
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
			OE0 control gate driver of TFT panel							
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
			OE0 control gate driver of TFT panel							
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
			OE0 control gate driver of TFT panel							
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
			OE0 control gate driver of TFT panel							
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
			OE1 control gate driver of TFT panel							
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
			OE1 control gate driver of TFT panel							
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
			OE1 control gate driver of TFT panel							
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
			OE1 control gate driver of TFT panel							
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
			OE1 control gate driver of TFT panel							
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
			OE1 control gate driver of TFT panel							

(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
			OE2 control gate driver of TFT panel							
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
			OE2 control gate driver of TFT panel							
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
			OE2 control gate driver of TFT panel							
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
			OE2 control gate driver of TFT panel							
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
			OE2 control gate driver of TFT panel							
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
			OE2 control gate driver of TFT panel							

(7) Touch screen I/F

Symbol	Name	Address	7	6	5	4	3	2	1	0
TSICR0	Touch screen I/F control register 0	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		Detection condition 0: no touch 1: touch	INT4 interrupt control 0: Disable 1: Enable	SPY 0 : OFF 1 : ON	SPX 0 : OFF 1 : ON	SMY 0 : OFF 1 : ON	SMX 0 : OFF 1 : ON
TSICR1	Touch screen I/F control register 1	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	1024	256	64	8	4	2	1
				De-bounce time is set by “(N × 64 − 16)/f_{SYS}” − formula. “N” is sum of number which is set to “1” in bit6 to bit0.						

(8) SDRAM controller

Symbol	Name	Address	7	6	5	4	3	2	1	0
SDACR1	SDRAM access control register 1	0250H	–	–	SMRD	SWRC	SBST	SBL1	SBL0	SMAC
			R/W							
			0	0	0	0	0	1	0	0
			Always write “0”	Always write “0”	Mode register set delay time 0:1 clock 1:2 clocks	Write recovery time 0:1 clock 1:2 clocks	Burst stop command 0: recharge all 1:Burst stop	Select read burst length 00: Reserved 01: Full page read, Burst write 10: 1 word read, Single write 11: Full page read Single write		SDRAM controller 0: Disable 1: Enable
SDACR2	SDRAM access control register 2	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: 47 states 001: 78 states 010: 97 states 011: 124 states 100: 156 states 101: 295 states 110: 249 states 111: 312 states			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	TMRA01 prescaler	UP counter (UC1)	UP counter (UC0)
								0: Stop and clear 1: Run (Count up)		
TA0REG	8-bit timer register 0	1102H (Prohibit RMW)	–							
			W							
			Undefined							
TA1REG	8-bit timer register 1	1103H (Prohibit RMW)	–							
			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 10: 2^7 11: 2^8		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 control register	1105H (Prohibit RMW)					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don't care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF Inversion select 0: TMRA0 1: TMRA1
TA23RUN	TMRA23 RUN register	1108H	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	TMRA23 prescaler	UP counter (UC3)	UP counter (UC4)
								0: Stop and clear 1: Run (Count up)		
TA2REG	8-bit timer register 2	110AH (Prohibit RMW)	–							
			W							
			Undefined							
TA3REG	8-bit timer register 3	110BH (Prohibit RMW)	–							
			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 10: 2^7 11: 2^8		Source clock for TMRA3 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA2 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA3FFCR	TMRA3 flip-flop control register	110DH (Prohibit RMW)					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don't care		TA3FF control for inversion 0: Disable 1: Enable	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 write "0"			IDLE2 0: Stop 1: Operate	TMRB0 prescaler		UP counter (UC10)
								0: Stop and clear 1: Run (Count up)		
TB0MOD	TMRB0 mode register	1182H (Prohibit RMW)	–	–	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			Always write "0".		Execute software capture 0: capture 1: Undefined	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 (Prohibit RMW)	–	–	TB0CT1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*			R/W			W*	
			1	1	0	0	0	0	1	1
			Always write "11".		TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB0FF0 00: Invert 01: Set 10: Clear 11: Don't care * Always read as "11"	
					Invert when the UC value is loaded in to TB0CP1H/L.	Invert when the UC value is loaded in to TB0CP0H/L.	Invert when the UC value matches the value in TB0RG1H/L	Invert when the UC value matches the value in TB0RG0H/L		
TB0RG0L	16-bit timer register 0 low	1188H (Prohibit RMW)	–							
			W							
			Undefined							
TB0RG0H	16-bit timer register 0 high	1189H (Prohibit RMW)	–							
			W							
			Undefined							
TB0RG1L	16-bit timer register 1 low	118AH (Prohibit RMW)	–							
			W							
			Undefined							
TB0RG1H	16-bit timer register 1 high	118BH (Prohibit RMW)	–							
			W							
			Undefined							
TB0CP0L	Capture register 0 low	118CH	–							
			R							
			Undefined							
TB0CP0H	Capture register 0 high	118DH	–							
			R							
			Undefined							
TB0CP1L	Capture register 1 low	118EH	–							
			R							
			Undefined							
TB0CP1H	Capture register 1 high	118FH	–							
			R							
			Undefined							

(11) UART/serial channel (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial channel 0 buffer register	1200H (Prohibit RMW)	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R (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	
			Undefined	0	0	0	0	0	0	0
			Receive data bit8	Parity 0: Odd 1: Even	Parity 0: Disable 1: Enable	1: Error			0: SCLK0↑ 1: SCLK0↓	0: Baud rate generator 1: SCLK0 pin input
						Overrun	Parity	Framing		
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 bit8	0: CTS disable 1: CTS enable	0: Receive disable 1: Receive enable	Wake-up 0: Disable 1: Enable	00: I/O Interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		00: TA0TRG 01: Baud rate generator 10: Internal clock f_{IO} 11: External clock (SCLK0 input)	
BR0CR	Serial channel 0 baud rate control register	1203H	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
				R/W						
			0	0	0	0	0	0	0	0
			Always write “0”	(16-K)/16 divided 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency setting			
BR0ADD	Serial channel 0 K setting register	1204H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
							0	0	0	0
							Sets frequency divisor “K” (divided by N + (16 − K)/16).			
SC0MOD1	Serial channel 0 mode 1 register	1205H	I2S0	FDPX0						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Operate	Duplex 0: Half duplex 1: Full 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 × (value + 1) + 100ns Can be set: 1 to 14 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 (Prohibit RMW)	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	
			Undefined	0	0	0	0	0	0	0
			Receive data bit8	Parity 0: Odd 1: Even	Parity 0: Disable 1: Enable	1: Error			0: SCLK1↑ 1: SCLK1↓	0: Baud rate generator 1: SCLK1 pin input
						Overrun	Parity	Framing		
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: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		00: TA0TRG 01: Baud rate generator 10: Internal clock f_{IO} 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
			Always write "0"	(16 - K)/16 divided 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency setting			
BR1ADD	Serial channel 1 K setting register	120CH					BR1K3	BR1K2	BR1K1	BR1K0
							R/W			
							0	0	0	0
							Sets frequency divisor "K" (divided by N + (16 – K)/16).			
SC1MOD1	Serial channel 1 mode 1 register	120DH	I2S1	FDPX1						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Operate	Duplex 0: Half duplex 1: Full duplex						

(12) USB controller (1/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
Descriptor RAM0	Descriptor RAM 0 register	0500H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM1	Descriptor RAM 1 register	0501H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM2	Descriptor RAM 2 register	0502H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM3	Descriptor RAM 3 register	0503H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM381	Descriptor RAM 381 register	067DH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM382	Descriptor RAM 382 register	067EH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM383	Descriptor RAM 383 register	067FH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint0	Endpoint 0 register	0780H	EP0_DATA7	EP0_DATA6	EP0_DATA5	EP0_DATA4	EP0_DATA3	EP0_DATA2	EP0_DATA1	EP0_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint1	Endpoint 1 register	0781H	EP1_DATA7	EP1_DATA6	EP1_DATA5	EP1_DATA4	EP1_DATA3	EP1_DATA2	EP1_DATA1	EP1_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint2	Endpoint 2 register	0782H	EP2_DATA7	EP2_DATA6	EP2_DATA5	EP2_DATA4	EP2_DATA3	EP2_DATA2	EP2_DATA1	EP2_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint3	Endpoint 3 register	0783H	EP3_DATA7	EP3_DATA6	EP3_DATA5	EP3_DATA4	EP3_DATA3	EP3_DATA2	EP3_DATA1	EP3_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
EP1_MODE	Endpoint 1 mode register	0789H			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0
EP2_MODE	Endpoint 2 mode register	078AH			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0
EP3_MODE	Endpoint 3 mode register	078BH			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0

(12) USB controller (2/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
EP0_STATUS	Endpoint 0 status register	0790H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP1_STATUS	Endpoint 1 status register	0791H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP2_STATUS	Endpoint 2 status register	0792H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP3_STATUS	Endpoint 3 status register	0793H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP0_SIZE_L_A	Endpoint 0 size register Low A	0798H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP1_SIZE_L_A	Endpoint 0 size register Low A	0799H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP2_SIZE_L_A	Endpoint 2 size register Low A	079AH	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP3_SIZE_L_A	Endpoint 3 size register Low A	079BH	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP1_SIZE_L_B	Endpoint 1 size register Low B	07A1H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP2_SIZE_L_B	Endpoint 2 size register Low B	07A2H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP3_SIZE_L_B	Endpoint 3 size register Low B	07A3H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP1_SIZE_H_A	Endpoint 1 size register High A	07A9H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP2_SIZE_H_A	Endpoint 2 size register High A	07AAH						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP3_SIZE_H_A	Endpoint 3 size register High A	07ABH						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0

(12) USB controller (3/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
EP1_SIZE_H_B	Endpoint 1 size register High B	07B1H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP2_SIZE_H_B	Endpoint 2 size register High B	07B2H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP3_SIZE_H_B	Endpoint 0 size register High B	07B3H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
bmRequestType	bmRequest-Type register	07C0H	DIRECTION	REQ_TYPE1	REQ_TYPE0	RECIPIENT4	RECIPIENT3	RECIPIENT2	RECIPIENT1	RECIPIENT0
			R							
			0	0	0	0	0	0	0	0
bRequest	bRequest register	07C1H	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
			R							
			0	0	0	0	0	0	0	0
wValue_L	wValue register Low	07C2H	VALUE_L7	VALUE_L6	VALUE_L5	VALUE_L4	VALUE_L3	VALUE_L2	VALUE_L1	VALUE_L0
			R							
			0	0	0	0	0	0	0	0
wValue_H	wValue register High	07C3H	VALUE_H7	VALUE_H6	VALUE_H5	VALUE_H4	VALUE_H3	VALUE_H2	VALUE_H1	VALUE_H0
			R							
			0	0	0	0	0	0	0	0
wIndex_L	wIndex register Low	07C4H	INDEX_L7	INDEX_L6	INDEX_L5	INDEX_L4	INDEX_L3	INDEX_L2	INDEX_L1	INDEX_L0
			R							
			0	0	0	0	0	0	0	0
wIndex_H	wIndex register High	07C5H	INDEX_H7	INDEX_H6	INDEX_H5	INDEX_H4	INDEX_H3	INDEX_H2	INDEX_H1	INDEX_H0
			R							
			0	0	0	0	0	0	0	0
wLength_L	wLength register Low	07C6H	LENGTH_L7	LENGTH_L6	LENGTH_L5	LENGTH_L4	LENGTH_L3	LENGTH_L2	LENGTH_L1	LENGTH_L0
			R							
			0	0	0	0	0	0	0	0
wLength_H	wLength register High	07C7H	LENGTH_H7	LENGTH_H6	LENGTH_H5	LENGTH_H4	LENGTH_H3	LENGTH_H2	LENGTH_H1	LENGTH_H0
			R							
			0	0	0	0	0	0	0	0

(12) USB controller (4/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SetupReceived	SetupReceived register	07C8H	D7	D6	D5	D4	D3	D2	D1	D0
			W							
			0	0	0	0	0	0	0	0
Current_Config	Current_Config register	07C9H	REMOTEWAKEUP		ALTERNATE[1]	ALTERNATE[0]	INTERFACE[1]	INTERFACE[0]	CONFIG[1]	CONFIG[0]
			R		R					
			0		0	0	0	0	0	0
Standard Request	Standard-Request register	07CAH	S_INTERFACE	G_INTERFACE	S_CONFIG	G_CONFIG	G_DESCRIPT	S_FEATURE	C_FEATURE	G_STATUS
			R							
			0	0	0	0	0	0	0	0
Request	Request register	07CBH		SOFT_RESET	G_PORT_STS	G_DEVICE_ID	VENDOR	CLASS	ExSTANDARD	STANDARD
				R						
				0	0	0	0	0	0	0
DATASET1	DATASET 1 register	07CCH	EP3_DSET_B	EP3_DSET_A	EP2_DSET_B	EP2_DSET_A	EP1_DSET_B	EP1_DSET_A		EP0_DSET_A
			R							R
			0	0	0	0	0	0		0
DATASET2	DATASET 2 register	07CDH	EP7_DSET_B	EP7_DSET_A	EP6_DSET_B	EP6_DSET_A	EP5_DSET_B	EP5_DSET_A	EP4_DSET_B	EP4_DSET_A
			R							
			0	0	0	0	0	0	0	0
USB_STATE	USB state register	07CEH						Configured	Addressed	Default
								R/W	R	
								0	0	1
EOP	EOP register	07CFH	EP7_EOPB	EP6_EOPB	EP5_EOPB	EP4_EOPB	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
			W							
			1	1	1	1	1	1	1	1
COMMAND	Command register	07D0H		EP[2]	EP[1]	EP[0]	Command[3]	Command[2]	Command[1]	Command[0]
				W						
				0	0	0	0	0	0	0
EPx_SINGLE1	Endpoint 1 single register	07D1H	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	
			R/W				R/W			
			0	0	0		0	0	0	
EPx_BCS1	Endpoint 1 BCS register	07D3H	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_BCS	EP2_BCS	EP1_BCS	
			R/W				R/W			
			0	0	0		0	0	0	
INT_Control	Interrupt control register	07D6H								Status_nak
										R/W
										0
Standard Request Mode	Standard Request mode register	07D8H	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
			R/W							
			0	0	0	0	0	0	0	0
Request Mode	Request mode register	07D9H		Soft_Reset	G_Port_Sts	G_DeviceId				
				R/W						
				0	0	0				

(12) USB controller (5/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
Port Status	Port status register	07E0H	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
			W							
			0	0	0	1	1	0	0	0
FRAME_L	Frame register Low	07E1H	–	T[6]	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
			R							
			0	0	0	0	0	0	0	0
FRAME_H	Frame register H	07E2H	T[10]	T[9]	T[8]	T[7]		CREATE	FRAME_STS1	FRAME_STS0
			R					R		
			0	0	0	0		0	1	0
ADDRESS	Address register	07E3H		A6	A5	A4	A3	A2	A1	A0
				R						
				0	0	0	0	0	0	0
USBREADY	USB ready register	07E6H								USBREADY
										R/W
										0
Set Descriptor STALL	Set-Descriptor stall register	07E8H								S_D_STALL
										W
										0
USBINTFR1	USB interrupt flag register 1	07F0H	INT_URST_STR	INT_URST_END	INT_SUS	INT_RESUME	INT_CLKSTOP	INT_CLKON		
			R/W							
			0	0	0	0	0	0		
			When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							
USBINTFR2	USB interrupt flag register 2	07F1H	EP1_FULL_A	EP1_Empty_A	EP1_FULL_B	EP1_Empty_B	EP2_FULL_A	EP2_Empty_A	EP2_FULL_B	EP2_Empty_B
			R/W							
			0	0	0	0	0	0	0	0
			When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							
USBINTFR3	USB interrupt flag register 3	07F2H	EP3_FULL_A	EP3_Empty_A						
			R/W							
			0	0						
			When read 0:Not generate interrupt 1:Generate interrupt When write 0: Clear flag 1: –							
USBINTFR4	USB interrupt flag register 4	07F3H	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	EP2_Empty_B
			R/W							
			0	0	0	0	0	0	0	0
			When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

(12) USB controller (6/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
USBINTMR1	USB interrupt mask register 1	07F4H	MSK_URST_STR	MSK_URST_END	MSK_SUS	MSK_RESUME	MSK_CLKSTOP	MSK_CLKON		
			R/W							
			1	1	1	1	1	1		
			When read 0: Be not masked When write 0: Clear flag 1: Be masked 1: –							
USBINTMR2	USB interrupt mask register 2	07F5H	EP1_MSK_FA	EP1_MSK_EA	EP1_MSK_FB	EP1_MSK_EB	EP2_MSK_FA	EP2_MSK_EA	EP2_MSK_FB	EP2_MSK_EB
			R/W							
			1	1	1	1	1	1	1	1
			When read 0: Be not masked When write 0: Clear flag 1: Be masked 1: –							
USBINTMR3	USB interrupt mask register 3	07F6H	EP3_MSK_FA	EP3_MSK_EA						
			R/W							
			1	1						
			When read 0: Be not masked 1: Be masked When write 0: Clear flag 1: –							
USBINTMR4	USB interrupt mask register 4	07F7H	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	
			R/W							
			1	1	1	1	1	1	1	
			When read 0: Be not masked When write 0: Clear flag 1: Be masked 1: –							
USBCR1	USB control register 1	07F8H	TRNS_USE	WAKEUP				–	SPEED	USBCLKE
			R/W					R/W		
			0	0				0	1	0
								Always write “0”		

(13) AD converter (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD0	AD mode control register 0	12B8H	EOCF	ADBF	–	–	ITM0	REPEAT	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	Always write "0"	Always write "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"
ADMOD1	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	Always write "0"	Always write "0"	Always write "0"	Always write "0"	Input channel 000: AN0 001: AN1 010: AN2 011: AN3	
ADMOD2	AD mode control register 1	12BAH			–	–	–	–	–	ADTRGE
										R/W
					0	0	0	0	0	0
					Always write "0"	Always write "0"	Always write "0"	Always write "0"	Always write "0"	AD external trigger start control 0: Disable 1: Enable
ADREG0L	AD result register 0 low	12A0H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
ADREG0H	AD result register 0 high	12A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
ADREG1L	AD result register 1 low	12A2H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
ADREG1H	AD result register 1 high	12A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
ADREG2L	AD result register 2 low	12A4H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
ADREG2H	AD result register 2 high	12A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
ADREG3L	AD result register 3 low	12A6H	ADR31	ADR30						ADR3RF
			R							R
			Undefined							0
ADREG3H	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: $2^{15}/f_{IO}$ 01: $2^{17}/f_{IO}$ 10: $2^{19}/f_{IO}$ 11: $2^{21}/f_{IO}$			Always write "0"	IDLE2 0: Stop 1: Operate	1: Internally connects WDT out to the reset pin	Always write "0"
WDCR	WDT control register	1301H (Prohibit RMW)	–							
			W							
			–							
			B1H: WDT disable code 4E: WDT clear code							

(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” is read	40 sec.	20 sec.	10 sec.	8 sec.	4 sec.	2 sec.	1 sec.
MINR	Minute register	1321H		MI6	MI5	MI4	MI3	MI2	MI1	MI0
				R/W						
				Undefined						
			“0” is read	40 min.	20 min.	10 min.	8 min.	4 min.	2 min.	1 min.
HOURR	Hour register	1322H			HO5	HO4	HO3	HO2	HO1	HO0
					R/W					
					Undefined					
			“0” is read		20 hours (PM/AM)	10 hours	8 hours	4 hours	2 hours	1 hour
DAYR	Day register	1323H						WE2	WE1	WE0
								R/W		
								Undefined		
			“0” is read					W2	W1	W0
DATER	Date register	1324H			DA5	DA4	DA3	DA2	DA1	DA0
					R/W					
					Undefined					
			“0” is read		20 days	10 days	8 days	4 days	2 days	1 day
MONTHR	Month register	1325H				MO4	MO3	MO2	MO1	MO0
						R/W				
						Undefined				
		PAGE0	“0” is read			10 month	8 month	4 month	2 month	1 month
		PAGE1	“0” is read							0: Indicator for 12 hours 1: Indicator for 24 hours
YEARR	Year register	1326H	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
			R/W							
			Undefined							
		PAGE0	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 year
		PAGE1	“0” is read						Leap year setting 00: Leap year 01: One year after 10: Two years after 11: Three years after	
PAGER	Page register	1327H (Prohibit RMW)	INTENA			ADJUST	ENATMR	ENAALM		PAGE
			R/W			W	R/W			R/W
			0			Undefined	Undefined			Undefined
			INTRTC 0:disable 1:enable	“0” is read		0:Don’t care 1:Adjust	Clock 0:disable 1:enable	Alarm 0:disable 1:enable	“0” is read	PAGE setting
RESTR	Reset register	1328H (Prohibit RMW)	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	–	–	–	–
			W							
			Undefined							
			1 Hz 0:disable 1:enable	16 Hz 0:disable 1:enable	1: Reset clock	1: Reset alarm	Always write “0”			

(16) Melody/alarm generator

Symbol	Name	Address	7	6	5	4	3	2	1	0
ALM	Alarm pattern register	1330H	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
			R/W							
			0	0	0	0	0	0	0	0
			Alarm pattern set							
MELALMC	Melody/ alarm control register	1331H	FC1	FC0	ALMINV	–	–	–	–	MELALM
			R/W							
			0	0	0	0	0	0	0	0
			Free run counter control 00: Hold 01: Restart 10: Clear and stop 11: Clear and start		Alarm frequency invert 1: Invert	Always write “0”				Output frequency 0: Alarm 1: Melody
MELFL	Melody frequency L-register	1332H	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
			R/W							
			0	0	0	0	0	0	0	0
			Melody frequency set (Low 8bit)							
MELFH	Melody frequency H-register	1333H	MELON				ML11	ML10	ML9	ML8
			R/W				R/W			
			0				0	0	0	0
			Melody counter control 0: Stop and clear 1: Start				Melody frequency set (Upper 4 bits)			
ALMINT	Alarm interrupt enable register	1334H			–	IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
					R/W					
					0	0	0	0	0	0
					Always write “0”	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
			0: Disable write operation 1: Enable write operation	ECC circuit 11 (at <CE>=X): Reset 00 (at <CE>=1): Disable 01 (at <CE>=1): Enable 10 (at <CE>=1): Read ECC data calculated by NDFC 10 (at <CE>=0): Read ID data		Chip enable 0: Disable ($\overline{\text{NDCE}}$ is high) 1: Enable ($\overline{\text{NDCE}}$ is low)	Power Control Always write "11"		Address Latch Enable 0: Low 1: High	Command Latch Enable 0: Low 1: High
ND0FSR	NAND flash status register	1CC8H	BUSY							
			R							
			Undefined							
			0: Ready 1: Busy							
ND0FISR	NAND flash interrupt status register	1CCCH								RDY
										R/W
										0
										Read: 1: Change NDR/$\overline{\text{B}}$ Write: 1: Clear to "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	0	0	0
							Pulse width for $\overline{\text{NDRE}}$, $\overline{\text{NDWE}}$ = f_{SYS} × (This register'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: Channel 0 1: Channel 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
			0: Disable write operation 1: Enable write operation	ECC circuit 11 (at <CE>=X): Reset 00 (at <CE>=1): Disable 01 (at <CE>=1): Enable 10 (at <CE>=1): Read ECC data calculated by NDFC 10 (at <CE>=0): Read ID data		Chip enable 0: Disable ($\overline{\text{NDCE}}$ is high) 1: Enable ($\overline{\text{NDCE}}$ is low)	Power Control Always write "11"		Address Latch Enable 0: Low 1: High	Command Latch Enable 0: Low 1: High
ND1FSR	NAND flash status register	1CE8H	BUSY							
			R							
			Undefined							
			0: Ready 1: Busy							
ND1FISR	NAND flash interrupt status register	1CECH								RDY
										R/W
										0
										Read: 1: Change NDR/$\overline{\text{B}}$ Write: 1: Clear to "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	0	0	0
							Pulse width for $\overline{\text{NDRE}}$, $\overline{\text{NDWE}}$ = f_{SYS} × (This register'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^{2}S

Symbol	Name	Address	7	6	5	4	3	2	1	0
I2SBUFR	I^{2}S FIFO buffer (R)	0800H (Prohibit RMW)	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	I^{2}S FIFO buffer (L)	0808H (Prohibit RMW)	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	I^{2}S control register 0	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^{2}S 1: SIO	Status 0: Stop 1: Under transmitting	First bit 0: MSB 1: LSB	Bit number 0: 8 bits 1: 16 bits	Baud rate 00: f_{SYS} 01: f_{SYS}/2	10: f_{SYS}/4 11: f_{SYS}/8	WS clock 0: fs/4 1: TA1OUT
		080FH	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. Points of Note and Restrictions

6.1 Notation

(1) The notation for built-in I/O registers is as follows: Register symbol <Bit symbol>

Example: TA01RUN<TA0RUN> denotes bit TA0RUN of register TA01RUN.

(2) Read-modify-write instructions (RMW)

An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

Example 1: SET 3, (TA01RUN); Set bit3 of TA01RUN.

Example 2: INC 1, (100H); Increment the data at 100H.

- Examples of read-modify-write instructions on the TLCS-900:

Exchange instruction

```
EX    (mem), R
```

Arithmetic operations

```
ADD   (mem), R/#       ADC   (mem), R/#
SUB   (mem), R/#       SBC   (mem), R/#
INC   #3, (mem)        DEC   #3, (mem)
```

Logic operations

```
AND   (mem), R/#       OR    (mem), R/#
XOR   (mem), R/#
```

Bit manipulation operations

```
STCF#3/A, (mem)        RES   #3, (mem)
SET   #3, (mem)        CHG   #3, (mem)
TSET#3, (mem)
```

Rotate and shift operations

```
RLC   (mem)            RRC   (mem)
RL    (mem)            RR    (mem)
SLA   (mem)            SRA   (mem)
SLL   (mem)            SRL   (mem)
RLD   (mem)            RRD   (mem)
```

(3) f_{OSCH}, fc, f_{FPH}, f_{SYS}, f_{IO} and one state

The clock frequency input on pins X1 and 2 is referred to as f_{OSCH}. The clock selected by PLLCR0<FCSEL> is referred to as fc.

The clock selected by SYSCR1<SYSCK> is referred to as f_{FPH}. The clock frequency give by f_{FPH} divided by 2 is referred to as system clock f_{SYS}. The clock frequency given by f_{SYS} divided by 2 is referred to as f_{IO}.

One cycle of f_{SYS} is referred to as one state.

6.2 Notes

(1) AM0 and AM1 pins

These pins are connected to the V_{CC} (Power supply level) or the V_{SS} (Grand level) pin. Do not alter the level when the pin is active.

(2) Reserved address areas

The 16 bytes area (FFFFF0H ~ FFFFFFH) cannot be used since it is reserved for use as internal area. If using an emulator, an optional 64 Kbytes of the 16M bytes area is used for emulator control. Therefore, if using an emulator, this area cannot be used.

(3) Standby mode (IDLE1)

When the HALT instruction is executed in IDLE1 mode (in which only the oscillator operates), RTC (Real-time-clock) and MLD (Melody-alarm-generator) operate. When necessary, stop the circuit before the HALT instruction is executed.

(4) Warm-up counter

The warm-up counter operates when STOP mode is released, even if the system is using an external oscillator. As a result, a time equivalent to the warm-up time elapses between input of the release request and output of the system clock.

(5) Watchdog timer

The watchdog timer starts operation immediately after a reset is released. Disable the watchdog timer when it is not to be used.

(6) AD converter

The string resistor between the VREFH and VREFL pins can be cut by program so as to reduce power consumption. When STOP mode is used, disable the resistor using the program before the HALT instruction is executed.

(7) CPU (Micro DMA)

Only the “LDC cr, r” and “LDC r, cr” instructions can be used to access the control registers in the CPU (e.g., the transfer source address register (DMASn).).

(8) Undefined SFR

The value of an undefined bit in an SFR is undefined when read.

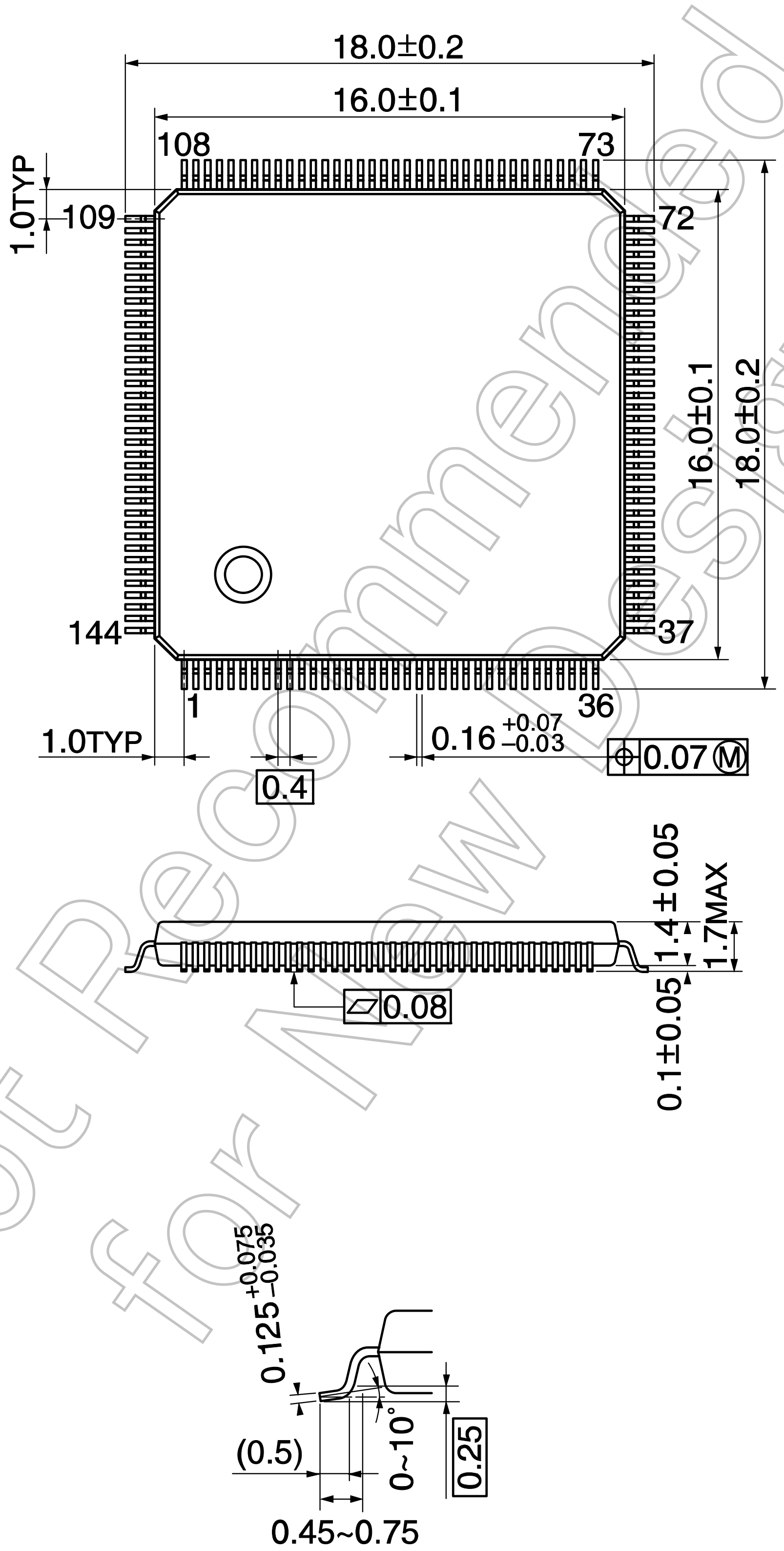
(9) POP SR instruction

Please execute the POP SR instruction during DI condition.

7. Package Dimensions

Package Name: LQFP144-P-1616-0.40C

Unit: mm



Note: Palladium plating

RESTRICTIONS ON PRODUCT USE

- Toshiba Corporation, and its subsidiaries and affiliates (collectively “TOSHIBA”), reserve the right to make changes to the information in this document, and related hardware, software and systems (collectively “Product”) without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA’s written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product’s quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before creating and producing designs and using, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the “TOSHIBA Semiconductor Reliability Handbook” and (b) the instructions for the application that Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS’ PRODUCT DESIGN OR APPLICATIONS.**
- Product is intended for use in general electronics applications (e.g., computers, personal equipment, office equipment, measuring equipment, industrial robots and home electronics appliances) or for specific applications as expressly stated in this document. Product is neither intended nor warranted for use in equipment or systems that require extraordinarily high levels of quality and/or reliability and/or a malfunction or failure of which may cause loss of human life, bodily injury, serious property damage or serious public impact (“Unintended Use”). Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, devices related to electric power, and equipment used in finance-related fields. Do not use Product for Unintended Use unless specifically permitted in this document.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. TOSHIBA assumes no liability for damages or losses occurring as a result of noncompliance with applicable laws and regulations.