

CMOS 8-Bit Microcontroller

**TMPA8700CHN/F, TMPA8700CKN/F, TMPA8700CMN/F,
TMPA8700CPN/F, TMPA8700CSN/F**

The A8700CH/CK/CM/CP/CS is the high speed and high performance 8 bit single chip microcomputer. This MCU contains CPU core, ROM, RAM, input / output ports, six multi-function timer / counter, serial interface, on-screen display, data slicer, jitter elimination circuit, PWM, 8 bit A/D converter and remote control signal processor on a chip.

The functions of the OSD circuit conform to the on-screen display functions of closed caption decoders based on FCC standards.

Part No.	ROM	RAM	Package	OTP MCU
TMPA8700CHN/F	16 Kbytes	1 Kbytes	SDIP42-P-600-1.78	TMPA8700PSN/F
TMPA8700CKN/F	24 Kbytes			
TMPA8700CMN/F	32 Kbytes			
TMPA8700CPN/F	48 Kbytes	2 Kbytes		
TMPA8700CSN/F	60 Kbytes			

Features

- ◆ 8 bit single chip microcomputer TLCS-870 Series
- ◆ Instruction execution time : 0.5 μ s (At 8 MHz)
- ◆ 412 basic instructions
 - Multiplication and Division (8 bits \times 8 bits, 16 bits \div 8 bits)
: Instruction execution time 3.5 μ s (At 8 MHz)
 - Bit manipulations (Set / Clear / Complement / Move / Test / Exclusive Or)
 - 16 bit data operations
 - 1 byte jump / subroutine-call (Short relative jump / Vector call)
- ◆ 14 interrupt sources (External : 5, Internal : 9)
 - All sources have independent latches each, and nested interrupt control is available.
 - Three edge-selectable external interrupts with noise reject
 - High-speed task switching by register bank changeover
- ◆ Input / Output ports (33 pins)
- ◆ Two 16 bit Timer / Counters
 - Timer, Event counter, Pulse width measurement, External trigger timer, Window modes
- ◆ Two 8 bit Timer / Counters
 - Timer, Event counter, Capture (Pulse width / duty measurement) modes

980910EBP1

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

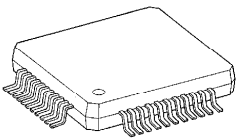


Purchase of TOSHIBA I²C components conveys a license under the Philips I²C Patent Rights to use these components in an I²C system, provided that the system conforms to the I²C Standard Specification as defined by Philips.

- ◆ Time Base Timer
 - Interrupt frequency : 1 Hz to 16384 Hz
- ◆ Watchdog Timer
- ◆ Serial bus Interface
 - I²C-bus (Single master) / 8 bits SIO timeshared 2ch
- ◆ On-screen display circuit
 - Character patterns : 251 characters
 - Character displayed : 32 columns 8 rows
 - Composition : 8 × 9 dots
 - Size of character : 3 kinds (Line by line)
 - Color of character : 7 kinds (Character by character)
 - Variable display position : Horizontal 128 steps, Vertical 256 steps
 - Fringing, Smoothing function
 - Conform to US CLOSED CAPTION DECODER REGULATION
- ◆ PWM outputs
 - 14 bit PWM output (1 channel)
 - 7 bit PWM outputs (9 channels)
- ◆ 8 bit successive approximate type A / D converter with sample and hold
 - 6 analog inputs
 - Conversion time : 23 μ s at 8 MHz
- ◆ High current outputs : LED direct drive capability (Typ. 20 mA × 4 bits).
- ◆ Remote control signal processor
- ◆ Data slicer circuit 1 ch
- ◆ Jitter elimination circuit
- ◆ Test video signal generator
- ◆ ROM corrective function
- ◆ Two Power saving operating modes
 - STOP mode : Oscillation stops. Battery / Capacitor back-up. Port output hold / high-impedance.
 - IDLE mode : CPU stops, and Peripherals operate using high-frequency clock. Release by interrupts.
- ◆ Emulation Pod : BMA8700CSN0A

Package

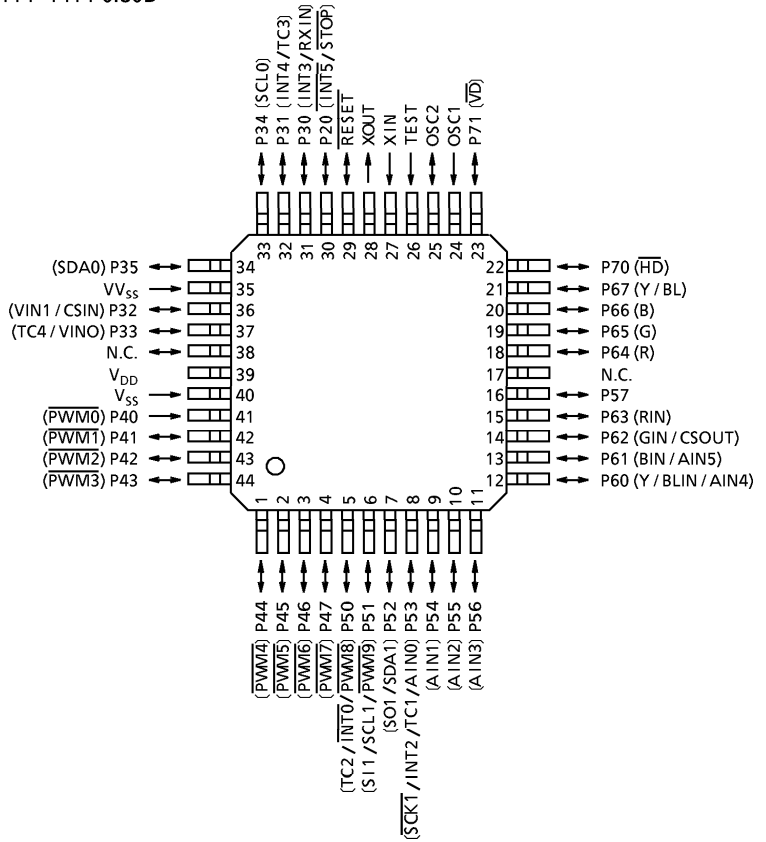
QFP44-P-1414-0.80D



TMPA8700CHF
 TMPA8700CKF
 TMPA8700CMF
 TMPA8700CPF
 TMPA8700CSF

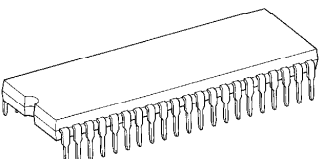
Pin Assignments (Top View)

QFP44-P-1414-0.80D



Package

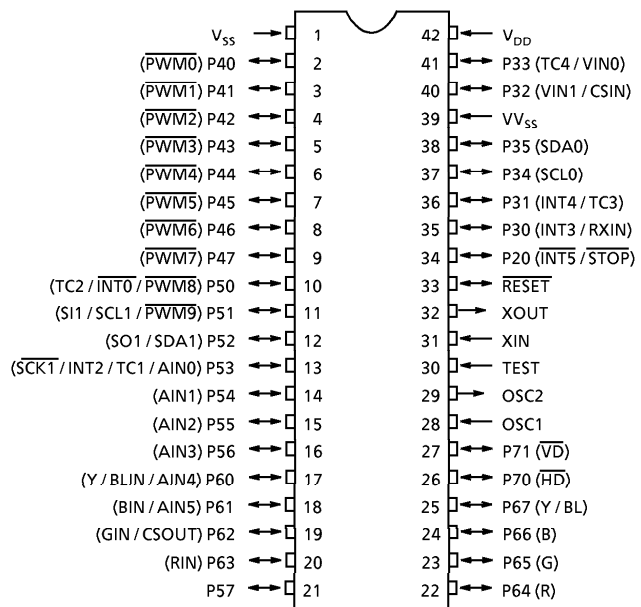
SDIP42-P-600-1.78



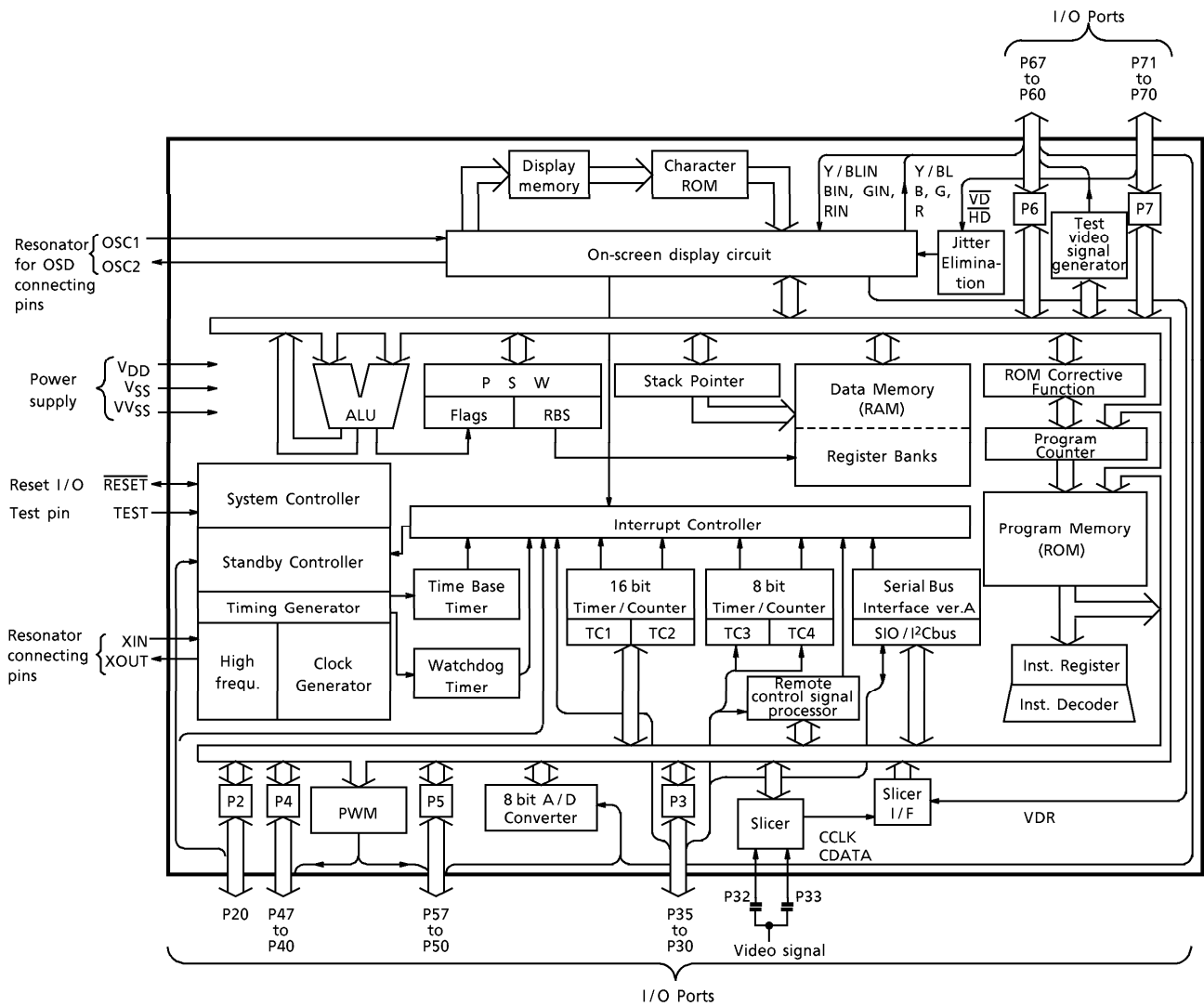
TMPA8700CHN
 TMPA8700CKN
 TMPA8700CMN
 TMPA8700CPN
 TMPA8700CSN

Pin Assignments (Top View)

SDIP42-P-600-1.78



Block Diagram



Pin Function

Pin No.	Pin Name	Input / Output	Function	
34	P20 ($\overline{\text{INT5}} / \overline{\text{STOP}}$)	I/O (Input, Input)	1 bit input / output port with latch. When used as an input port, an interrupt input or STOP mode release signal input, the latch must be set to "1".	External interrupt input 5 / STOP mode release signal input
38	P35 (SDA0)	I/O (I/O)	6 bit programmable input / output port. (tri-state) Each bit of this port can be individually as an input or an output under software control. During reset, all bits are configured as inputs.	I ² C bus 0 serial data input / output
37	P34 (SCL0)	I/O (I/O)		I ² C bus 0 serial clock input / output
41	P33 (TC4 / VIN0)	I/O (Input, Input)		Timer / counter 4 input / Video signal input 0
40	P32 (VIN1 / CSIN)	I/O (Input, Input)		Video signal input 1 / Composite sync input
36	P31 (INT4 / TC3)	I/O (Input, Input)		External interrupt input 4 / Timer / Counter 3 input
35	P30 (INT3 / RXIN)	I/O (Input, Input)	When used as a serial bus interface input / output, the pin must be set to the sink open drain mode, the latch must be set to "1", and the pin must be set to the output mode.	External interrupt input 3 / Remote control signal processor input
9 to 3	P47 ($\overline{\text{PWM7}}$) to P41 ($\overline{\text{PWM1}}$)	I/O (Output)	8 bit programmable input / output port (tri-state). Each bit of this port can be individually as an input or an output under software control. During reset, all bits are configured as inputs. When used as a PWM output, the latch must be set to "1" and the pin must be set to the output mode.	7 bit PWM outputs
2	P40 ($\overline{\text{PWM0}}$)			14 bit PWM output
21	P57	I/O	8 bit programmable input / output port. (tri-state)	A / D converter analog input A / D converter analog input / Timer / Counter 1 input / External interrupt input / SIO1 serial clock data input I ² C bus 1 serial data input / output SIO1 serial data output I ² C bus 1 serial clock input / output / 7 bit PWM outputs/SIO1 serial data input 7 bit PWM outputs / External interrupt input 0 / Timer / Counter 2 input
16 to 14	P56 (AIN3) to P54 (AIN1)	I/O (Input)	Each bit of this port can be individually as an input or an output under software control. During reset, all bits are configured as inputs.	
13	P53 (AIN0 / TC1 / INT2 / $\overline{\text{SCK1}}$)	I/O (Input, Input, Input, I/O)	When used as an input port, a serial bus interface input, the pin must be set to the input mode. When used as a PWM output, serial bus interface output, the latch must be set to "1", and the pin must be set to the output mode.	
12	P52 (SDA1 / SO1)	I/O (I/O, Output)	When used as a serial bus interface input / output, the pin must be set to the sink open drain mode, the latch must be set to "1", and the pin must be set to the output mode.	
11	P51 ($\overline{\text{PWM9}} / \text{SCL1} / \text{S11}$)	I/O (Output, I/O, Input)		
10	P50 ($\overline{\text{PWM8}} / \text{INT0} / \text{TC2}$)	I/O (Output, Input, Input)		
27	P71 ($\overline{\text{VD}}$)	I/O (Input)	2 bit input / output port with latch. When used as an input port, a vertical synchronous signal input, or a horizontal synchronous signal input, the latch must be set to "1".	
26	P70 ($\overline{\text{HD}}$)		Horizontal synchronous signal input	
25	P67 (Y / BL)	I/O (Output)	8 bit programmable input / output port (tri-state, P63 to P60 : High current output). Each bit of this port can be individually as an input or an output under software control. During reset, all bits are configured as inputs. When P67 to P64 ports are used as output port, bits 7 to 4 of address 0F91H must be set to "1".	R, G, B, Y / BL output G, B, Y / BL input Test video signal generator output A / D converter analog input
24	P66 (B)			
23	P65 (G)			
22	P64 (R)			
20	P63 (RIN)			
19	P62 (GIN / CSOUT)			
18	P61 (BIN / AIN5)			
17	P60 (Y / BLIN / AIN4)	I/O (Input / Input)	When P63 to P60 port used as RIN, GIN, BIN, Y / BLIN input, these ports must be set to the input mode.	
28, 29	OSC1, OSC2	Input, Output	Resonator connecting pin of on-screen display circuit	
31, 32	XIN, XOUT		Resonator connecting pin (High frequency). For external clock input, XIN is used and XOUT is opened.	
33	$\overline{\text{RESET}}$	I/O	Reset signal input or watchdog timer output / address-trap-reset output / system-clock-reset output.	
30	TEST	Input	Test pin for out-going test. Be tied to low.	
42 ; 1, 39	VDD, VSS, VVSS	Power Supply	+ 5 V, 0 V (GND)	

Operational Description

1. CPU Core Functions

The CPU core consists of a CPU, a system clock controller, an interrupt controller, a watchdog timer, and ROM corrective function.

This section provides a description of the CPU core, the program memory (ROM), the data memory (RAM), and the reset circuit.

1.1 Memory Address Map

The TLCS-870 Series is capable of addressing 64 Kbytes of memory. Figure 1-1. shows the memory address maps of the A8700CH / CK / CM / CP / CS. In the TLCS-870 Series, the memory is organized 4 address spaces (ROM, RAM, SFR, and DBR). It uses a memory mapped I / O system, and all I / O registers are mapped in the SFR / DBR address spaces. There are 16 banks of general-purpose registers. The register banks are also assigned to the first 128 bytes of the RAM address space.

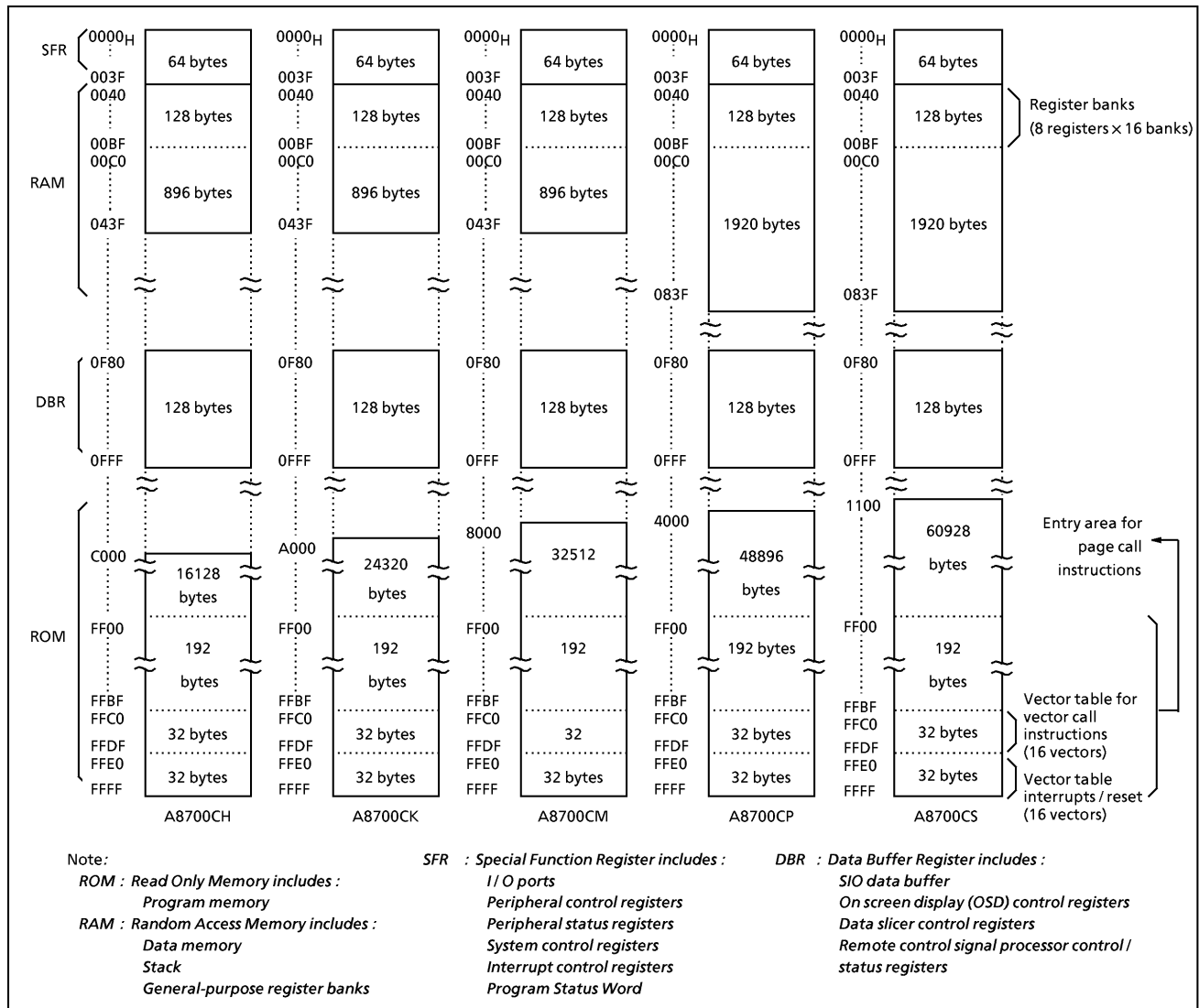


Figure 1-1. Memory Address Map

1.2 Program Memory (ROM)

The A8700CH / CK / CM / CP / CS have 16 K / 24 K / 32 K / 48 K / 60 Kbytes (addresses C000_H / A000_H / 8000_H / 4000_H / 1100_H to FFFF_H) of program memory (mask programmed ROM). Figure.1-2 shows the program memory map.

Addresses FF00_H to FFFF_H in the program memory can also be used for special purposes.

(1) Interrupt / reset vector table (addresses FFE0_H to FFFF_H)

This table stores of a reset vector and 15 interrupt vectors (2 bytes / vector). These vectors store a reset start address and 15 interrupt service routine entry addresses.

(2) Vector table for vector call instructions (addresses FFC0_H to FFDF_H)

This table stores call vectors (subroutine entry address, 2 bytes / vector) for the vector call instructions [CALLV n]. There are 16 vectors. The CALLV instruction is one-byte instruction, and increases memory efficiency when utilized for frequently used subroutine calls (called from 3 or more locations).

(3) Entry area for page call instructions (addresses FF00_H to FFFF_H)

This is the subroutine entry address area for the page call instructions [CALLP n]. Addresses FF00_H to FFBF_H are normally used because address FFC0_H to FFFF_H are used for the vector tables. This is two-byte instruction.

Programs and fixed data are stored in the program memory. The instruction to be executed next is read from the address indicated by the current contents of the program counter (PC). There are relative jump and absolute jump instructions. The concepts of page or bank boundaries are not used in the program memory concerning any jump instruction.

Example : The relationship between the jump instructions and the PC.

① 5bit PC-relative jump [JRS cc, \$ + 2 + d]

E8C4H : JRS T, \$ + 2 + 08H

When JF = 1, the jump is made to the address E8CE_H, which is 08_H added to the contents of the PC. (The PC contains the address of the instruction being executed + 2 ; therefore, in this case, the PC contents are E8C4_H + 2 = E8C6_H.)

② 8bit PC-relative jump [JR cc, \$ + 2 + d]

E8C4H : JR Z, \$ + 2 + 80H

When ZF = 1, the jump is made to the address E846_H, which is FF80_H (– 128) added to the current contents of the PC.

③ 16bit absolute jump [Ja]

E8C4H : JP 0C235H

An unconditional jump is made to the address C235_H. The absolute jump instruction can jump anywhere within the entire 64Kbyte space.

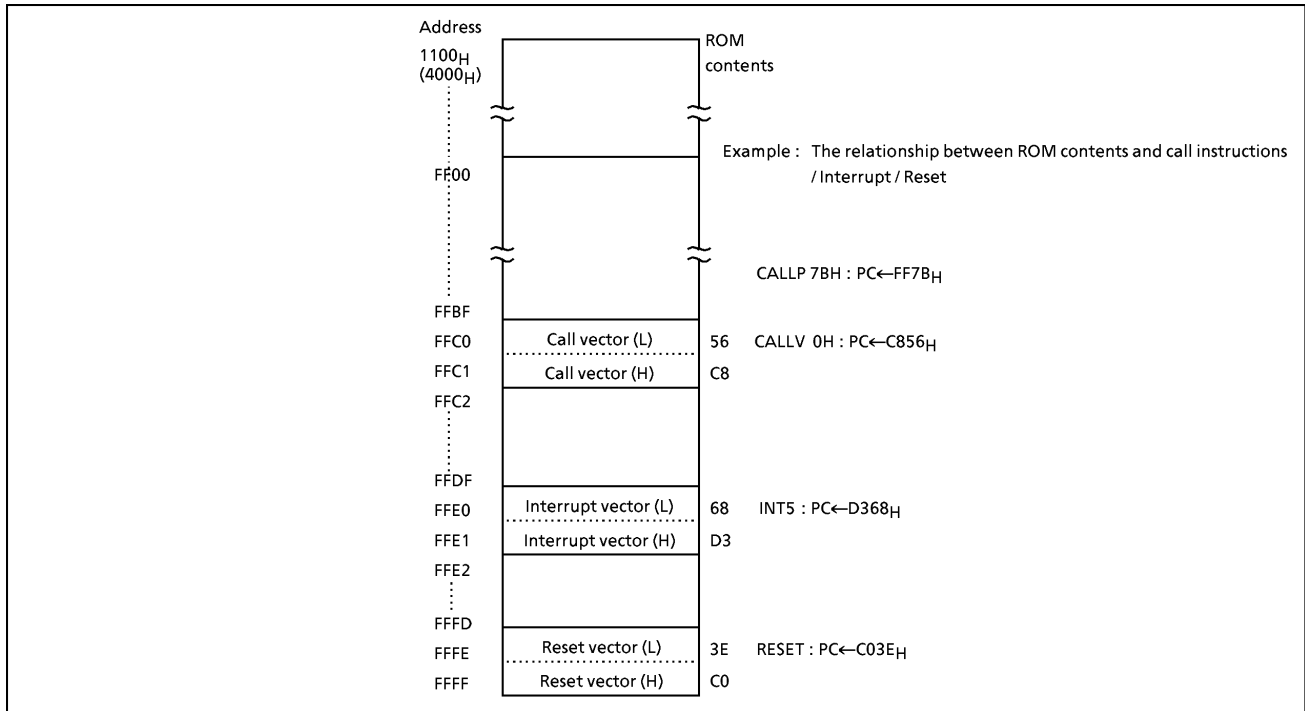


Figure 1-2. Program Memory Map

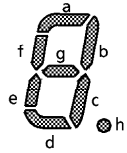
In the TLCS-870 Series, the same instruction used to access the data memory (e.g. [LD A, (HL)]) is also used to read out fixed data (ROM data) stored in the program memory. The register-offset PC-relative addressing (PC + A) instructions can also be used, and the code conversion, table look-up and n-way multiple jump processing can easily be programmed.

Example 1 : Loads the ROM contents at the address specified by the HL register pair contents into the accumulator (HL ≥ 1100_H for A8700CS) :

```
LD A, (HL) ; A ← ROM (HL)
```

Example 2 : Converts BCD to 7-segment code (common anode LED). When A = 05_H, 92_H is output to port P5 after executing the following program :

```
ADD A, TABLE - $ - 4 ; P5 ← ROM (TABLE + A)
LD (P5), (PC + A)
JRS T, SNEXT ; Jump to SNEXT
TABLE : DB 0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0D8H, 80H, 98H
SNEXT :
```



Note1: "\$" is a header address of ADD instruction.
 Note2: DB is a byte data definition instruction.

Example 3 : N-way multiple jump in accordance with the contents of accumulator (0 ≤ A ≤ 3) :

```
SHLC A ; if A = 00H then PC ← C234H
JP (PC + A) ; if A = 01H then PC ← C378H
; if A = 02H then PC ← DA37H
; if A = 03H then PC ← E1B0H
DW 0C234H, 0C378H, 0DA37H, 0E1B0H
```

Note: DW is a word data definition instruction.

SHLCA
JP (PC + A)
34
C2
78
C3
37
DA
B0
E1

1.3 Program Counter (PC)

The program counter (PC) is a 16 bit register which indicates the program memory address where the instruction to be executed next is stored. After reset, the user defined reset vector stored in the vector table (addresses FFFF_H and FFFE_H) is loaded into the PC ; therefore, program execution is possible to start from any desired address. For example, when C0_H and 3E_H are stored at addresses FFFF_H and FFFE_H, respectively, the execution starts from address C03E_H after reset.

The TLCS-870 Series utilizes pipelined processing (instruction pre-fetch) ; therefore, the PC always indicates 2 addresses in advance. For example, while a 1byte instruction stored at address C123_H is being executed, the PC contains C125_H.

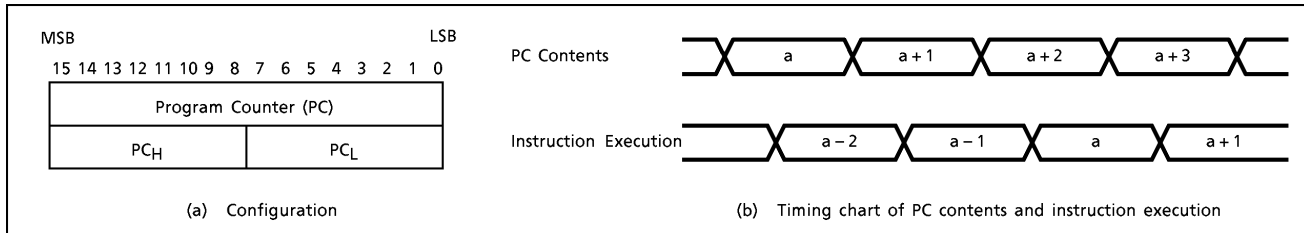


Figure 1-3. Program Counter

1.4 Data Memory (RAM)

The A8700CH / CK / CM have 1 Kbytes (addresses 0040_H to 043F_H) and the A8700CP / CS have 2 Kbytes (addresses 0040_H to 083F_H) of data memory (static RAM). Figure 1-4 shows the data memory map.

Addresses 0000_H to 00FF_H are used as a direct addressing area to enhance instructions which utilize this addressing mode ; therefore, addresses 0040_H to 00FF_H in the data memory can also be used for user flags or user counters.

Example 1 : If bit 2 at data memory address 00C0_H is "1", 00_H is written to data memory at address 00E3_H ; otherwise, FF_H is written to the data memory at address 00E3_H :

```

TEST (00C0H).2      ; if (00C0H) 2 = 0 then jump
JRS  T, SZERO
CLR  (00E3H)        ; (00E3H) ← 00H
JRS  T, SNEXT
SZERO : LD (00E3H), 0FFH ; (00E3H) ← FFH
SNEXT :
```

Example 2 : Increments the contents of data memory at address 00F5_H, and clears to 00_H when 10_H is exceeded :

```

INC  (00F5H)        ; (00F5H) ← (00F5H) + 1
AND  (00F5H), 0FH   ; (00F5H) ← (00F5H) ∧ 0FH
```

General-purpose register banks (8 registers x 16 banks) are also assigned to the 128bytes of addresses 0040_H to 00BF_H. Access as data memory is still possible even when being used for registers. For example, reading out of the contents of the data memory at address 0040_H means that the contents of the accumulator in the bank 0 are read out. The stack can be located anywhere within the data memory except the register bank area. The stack depth is limited only by the free data memory size. For more details on the stack, see section "1.7 Stack and Stack Pointer".

Note: The data memory contents become unstable when the power supply is turned on ; therefore, the data memory should be initialized by an initialization routine.

Example1 : Clears RAM to "00H" except the bank 0 (A8700CP / CS)

```
LD HL, 0048H      ; Sets start address to HL register pair
LD A, H          ; Sets initial data (00H) to A register
LD BC, 07F7H    ; Sets number of byte to BC register pair
SRAMCLR : LD (HL+), A
            DEC BC
            JRS F, SRAMCLR
```

Example2 : Clears RAM to "00H" except the bank 0 (A8700CH / CK / CM)

```
LD HL, 0048H      ; Sets start address to HL register pair
LD A, H          ; Sets initial data (00H) to A register
LD BC, 03F7H    ; Sets number of byte to BC register pair
SRAMCLR : LD (HL+), A
            DEC BC
            JRS F, SRAMCLR
```

Note: The general-purpose registers are mapped in the RAM ; therefore, do not clear RAM at the current bank addresses.

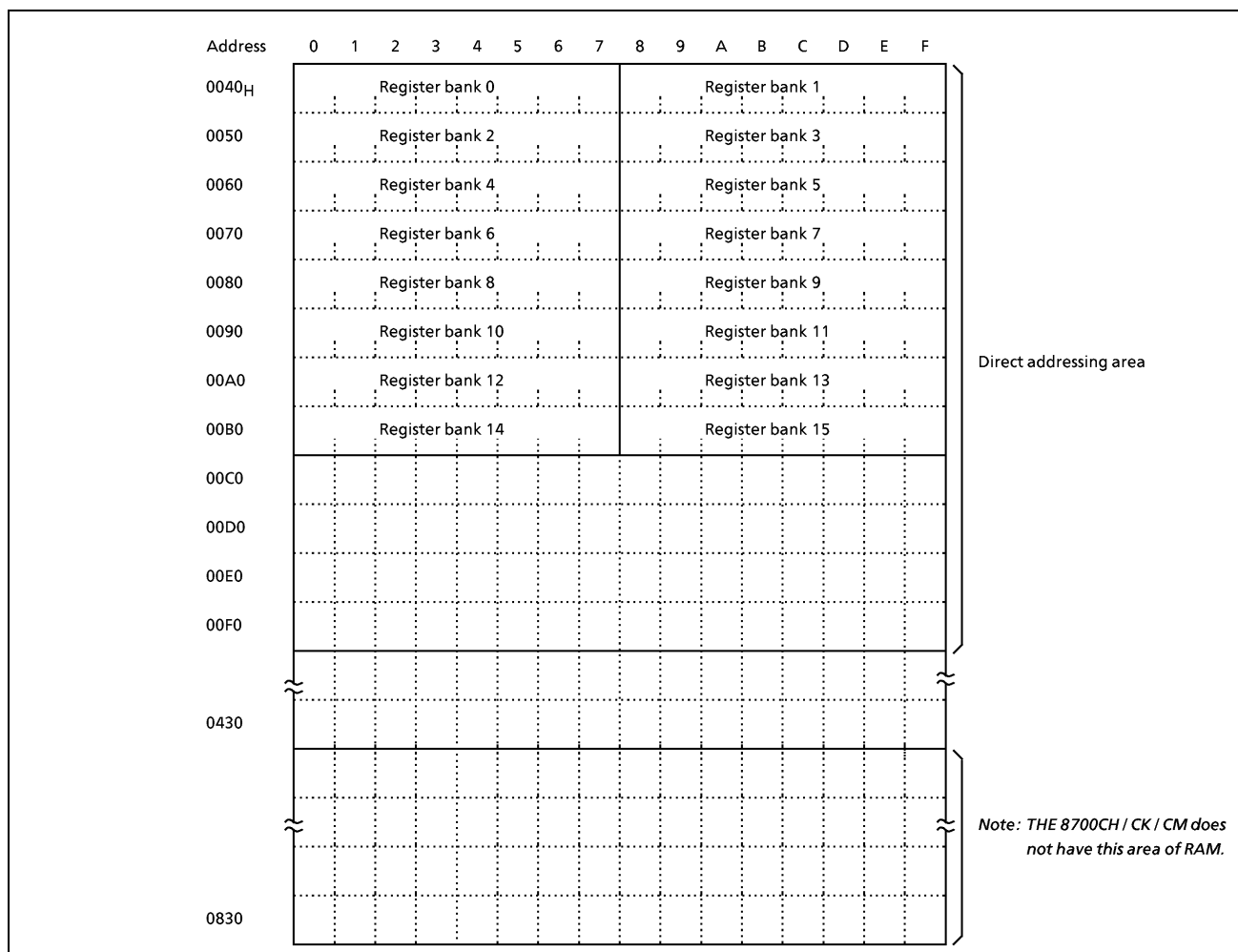


Figure 1-4. Data Memory Map

1.5 General-Purpose Register Banks

General-purpose registers are mapped into addresses 0040_H to 00BF_H in the data memory. There are 16 register banks, and each bank contains eight 8 bit registers W, A, B, C, D, E, H, and L. Figure 1-5 shows the general-purpose register bank configuration. The register bank which is not used can be used as the data memory.

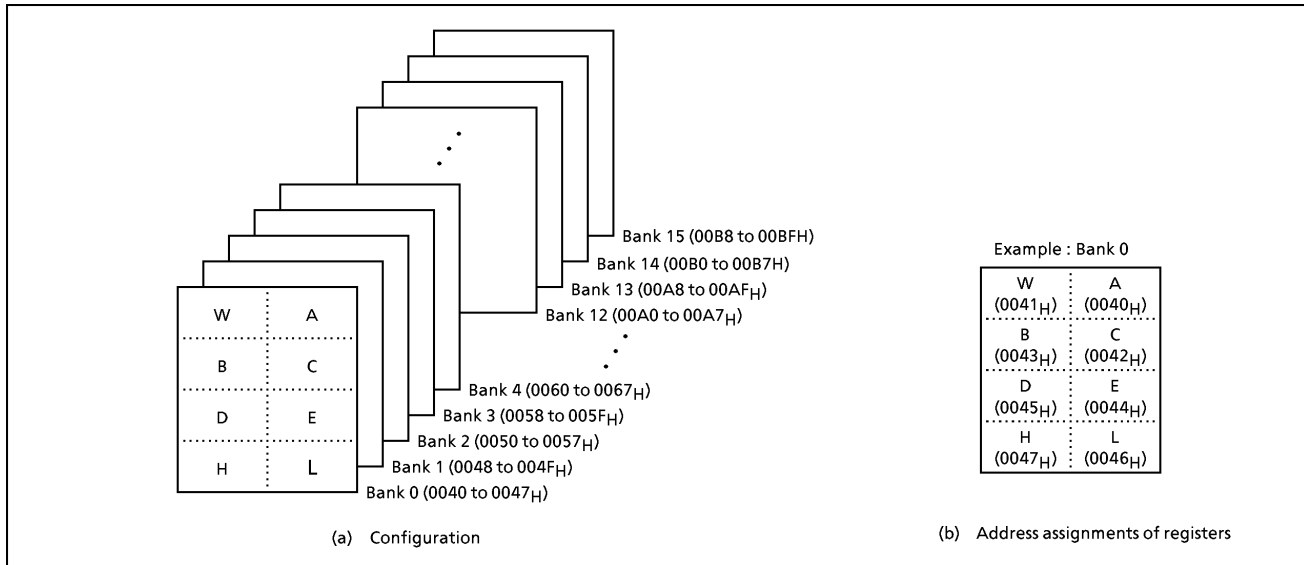


Figure 1-5. General-purpose Register Banks

In addition to access in 8 bit units, the registers can also be accessed in 16 bit units as the register pairs WA, BC, DE, and HL. Besides its function as a general-purpose register, the register also has the following functions :

(1) A, WA

The A register has a function as an 8 bit accumulator and the WA register pair has a function as a 16 bit accumulator (W is high byte and A is low byte). Other registers than A can also be used as accumulators for 8 bit operations.

- Examples:
- ① ADD A, B ; Adds B contents to A contents and stores the result into A.
 - ② SUB WA, 1234_H ; Subtracts 1234_H from WA contents and stores the result into WA.
 - ③ SUB E, A ; Subtracts A contents from E contents, and stores the result into E.

(2) HL, DE

The HL register pair has a function as data pointer (HL) / index register (HL + d) / base register (HL + C), and the DE register pair has a function as a data pointer (DE). The HL also has an auto-post-increment and an auto-pre-decrement function. This function simplifies a multiple digit data processing, software LIFO (last-in first-out) processing, etc.

Example 1:

① LD A, (HL)	;	Loads the memory contents at the address specified by HL into A.
② LD A, (HL + 52H)	;	Loads the memory contents at the address specified by the value obtained by adding 52 _H to HL contents into A.
③ LD A, (HL + C)	;	Loads the memory contents at the address specified by the value obtained by adding the register C contents to HL contents into A.
④ LD A, (HL +)	;	Loads the memory contents at the address specified by HL into A. Then increments HL.
⑤ LD A, (-HL)	;	Decrements HL. Then loads the memory contents at the address specified by new HL into A.

The TLCS-870 Series can directly transfer data from memory to memory, and directly operate between memory data and memory data. This facilitates the programming of block processing.

Example 2 : Block transfer

	LD B, m	;	Sets m (m = n - 1, n : number of bytes to transfer) to B
	LD HL, DSTA	;	Sets destination address to HL
	LD DE, SRCA	;	Sets source address to DE
SLOOP :	LD (HL), (DE)	;	(HL) ← (DE)
	INC HL	;	HL ← HL + 1
	INC DE	;	DE ← DE + 1
	DEC B	;	B ← B - 1
	JRS F, SLOOP	;	if B ≥ 0 then loop

(3) B, C, BC

Registers B and C can be used as 8 bit buffers or counters, and the BC register pair can be used as a 16 bit buffer or counter. The C register functions as an offset register for register-offset index addressing (refer to example 1 ③ above) and as a divisor register for the division instruction [DIV gg, C].

Example 1 : Repeat processing

	LD B, n	;	Sets n as the number of repetitions to B (n + 1 times processing)
SREPEAT :	Processing		
	DEC B		
	JRS F, SREPEAT		

Example 2 : Unsigned integer division (16 bit ÷ 8 bit)

	DIV WA, C	;	Divides the WA contents by the C contents, places the quotient in A and the remainder in W.
--	-----------	---	---

The general-purpose register banks are selected by the 4 bit register bank selector (RBS). During reset, the RBS is initialized to "0". The bank selected by the RBS is called the current bank.

The RBS is assigned to address 003F_H in the SFR as the program status word (PSW) with the flag. The PSW can be operated by the memory access instruction. And, there are 3 instructions [LD RBS, n], [PUSH PSW] and [POP PSW] to access the PSW.

Example 1 : Incrementing the RBS

```
INC (003FH) ; RBS ← RBS + 1
```

Example 2 : Reading the RBS

```
LD A, (003FH) ; A ← PSW (A3 to 0 ← RBS, A7 to 4 ← FLAG)
```

High efficient programming and high-speed task switching are possible by using bank changeover to save registers during interrupt and to transfer parameters during subroutine processing. During interrupt, the PSW is automatically saved onto the stack. The bank used before the interrupt was accepted is restored automatically by executing an interrupt return instruction [RETI] / [RETN] ; therefore, there is no need for the RBS save / restore software processing.

The TLCS-870 Series supports a maximum of 15 interrupt sources. One bank is assigned to the main program, and one bank can be assigned to each source. Also, to increase the efficiency of data memory usage, assign the same bank to interrupt sources which are not nested.

Example : Saving / restoring registers during interrupt task using bank changeover.

```
PINT1 : LD RBS, n ; RBS ← n (Bank changeover)
        Interrupt processing
        RETI ; Maskable interrupt return (Bank restoring)
```

1.6 Program Status Word (PSW)

The program status word (PSW) consists of a register bank selector (RBS) and four flags, and the PSW is assigned to address 003FH in the SFR.

The RBS can be read and written by using the memory access instruction however the flags can only be read. When writing to the PSW, the change specified by the instruction is made without writing data to the flags. For example, when the instruction [LD (003FH), 05H] is executed, "5" is written to the RBS and the JF is set to "1", but the other flags are not affected.

During interrupt, the PSW and the program counter are saved on to the stack. The PSW is restored from the stack by executing an interrupt return instruction [RETI], [RETN], and the state becomes the same as before the interrupt was accepted.

[PUSH PSW] and [POP PSW] are PSW access instructions.

1.6.1 Register Bank Selector (RBS)

The register bank selector (RBS) is a 4 bit register used to select general-purpose register banks. For example, when RBS = 2, bank 2 is currently selected.

During reset, the RBS is initialized to "0".

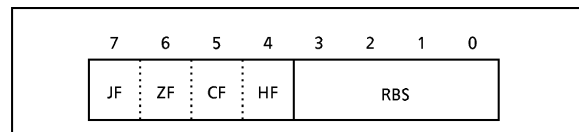


Figure 1-6. PSW (Flags, RBS) Configuration

1.6.2 Flags

The flags are configured with the upper 4bits : a zero flag, a carry flag, a half carry flag and a jump status flag. The flags are set or cleared under conditions specified by the instruction. These flags except the half carry flag are used as jump condition "cc" for conditional jump instructions [JR cc, \$ + 2 + d] / [JRS cc, \$ + 2 + d].

After reset, the jump status flag is initialized to "1", but the other flags are not affected.

(1) Zero flag (ZF)

The ZF is set to "1" if the operation result or the transfer data is 00H (for 8 bit operations and data transfers) / 0000H (for 16 bit operations) ; otherwise the ZF is cleared to "0". During the bit manipulation instructions [SET, CLR, and CPL], the ZF is set to "1" if the contents of the specified bit is "0" ; otherwise the ZF is cleared to "0". This flag is set to "1" when the upper 8 bits of the product are 00H during the multiplication instruction [MUL], and when 00H for the remainder during the division instruction [DIV] ; otherwise it is cleared to "0".

(2) Carry flag (CF)

The CF is set to "1" when a carry out of the MSB (most significant bit) of the result occurred during addition or when a borrow into the MSB of the result occurred during subtraction ; otherwise the CF is cleared to "0". During division, this flag is set to "1" when the divisor is 00H (divided by zero error), or when the quotient is 100H or higher (quotient overflow error) ; otherwise it is cleared. The CF is also affected during the shift / rotate instructions [SHLC, SHRC, ROLC, and RORC]. The data shifted out from a register is set to the CF.

This flag is also a 1bit register (a boolean accumulator) for the bit manipulation instructions. Set / clear / complement are possible with the CF manipulation instructions.

Example : Bit manipulation

```
LD   CF, (0007H). 5           ; (0001H) 2 ← (0007H) 5 ∨ (009AH) 0
XOR  CF, (009AH). 0
LD   (0001H). 2, CF
```

(3) Half carry flag (HF)

The HF is set to "1" when a carry occurred between bits 3 and 4 of the operation result during an 8 bit addition, or when a borrow occurred from bit 4 into bit 3 of the result during an 8 bit subtraction ; otherwise the HF is cleared to "0". This flag is useful in the decimal adjustment for BCD operations (adjustments using the [DAA r], or [DAS r] instructions).

Example : BCD operation

(The A becomes 47_H after executing the following program when A = 19_H, B = 28_H)

```

ADD    A, B          ; A ← 41H, HF ← 1, CF ← 0
DAA    A             ; A ← 41H + 06H = 47H (decimal-adjust)
    
```

(4) Jump status flag (JF)

Zero or carry information is set to the JF after operation (e.g. INC, ADD, CMP, TEST).

The JF provides the jump condition for conditional jump instructions [JRS T / F, \$ + 2 + d], [JR T / F, \$ + 2 + d] (T or F is a condition code). Jump is performed if the JF is "1" for a true condition (T), or the JF is "0" for a false condition (F).

The JF is set to "1" after executing the load / exchange / swap / nibble rotate / jump instruction, so that [JRS T, \$ + 2 + d] and [JR T, \$ + 2 + d] can be regarded as an unconditional jump instruction.

Example : Jump status flag and conditional jump instruction

```

INC    A
JRS    T, SLABLE1    ; Jump when a carry is caused by the immediately
:                                     preceding operation instruction.
LD     A, (HL)
JRS    T, SLABLE2    ; JF is set to "1" by the immediately preceding
:                                     instruction, making it an unconditional jump instruction.
    
```

Example : The accumulator and flags become as shown below after executing the following instructions when the WA register pair, the HL register pair, the data memory at address 00C5_H, the carry flag and the half carry flag contents being "219A_H", "00C5_H", "D7_H", "1" and "0", respectively.

Instruction	ACC. After Execution	Flag After Execution				Instruction	ACC. After Execution	Flag After Execution			
		JF	ZF	CF	HF			JF	ZF	CF	HF
ADDC A, (HL)	72	1	0	1	1	INC A	9B	0	0	1	0
SUBB A, (HL)	C2	1	0	1	0	ROL A	35	1	0	1	0
CMP A, (HL)	9A	0	0	1	0	ROR A	CD	0	0	0	0
AND A, (HL)	92	0	0	1	0	ADD WA, 0F508H	16A2	1	0	1	0
LD A, (HL)	D7	1	0	1	0	MUL W, A	13DA	0	0	1	0
ADD A, 66H	00	1	1	1	1	SET A.5	BA	1	1	1	0

1.7 Stack and Stack Pointer

1.7.1 Stack

The stack provides the area in which the return address or status, etc. are saved before a jump is performed to the processing routine during the execution of a subroutine call instruction or the acceptance of an interrupt. On a subroutine call instruction [CALL a] / [CALLP a] / [CALLV n], the contents of the PC (the return address) is saved ; on an interrupt acceptance, the contents of the PC and the PSW are saved (the PSW is pushed first, followed by PC_H and PC_L). Therefore, a subroutine call occupies two bytes on the stack ; an interrupt occupies three bytes.

When returning from the processing routine, executing a subroutine return instruction [RET] restores the contents to the PC from the stack ; executing an interrupt return instruction [RETI] / [RETN] restores the contents to the PC and the PSW (the PC_L is popped first, followed by PC_H and PSW).

The stack can be located anywhere within the data memory space except the register bank area.

1.7.2 Stack Pointer (SP)

The stack pointer (SP) is a 16 bit register containing the address of the next free locations on the stack.

The SP is post-decremented when a subroutine call or a push instruction is executed, or when an interrupt is accepted ; and the SP is pre-incremented when a return or a pop instruction is executed. Figure 1-8 shows the stacking order.

The SP is not initialized hardware-wise but requires initialization by an initialize routine (sets the highest stack address). [LD SP, mn], [LD SP, gg] and [LD gg, SP] are the SP access instructions (mn ; 16 bit immediate data, gg ; register pair).

Example 1 : To initialize the SP

LD SP, 043FH ; SP ← 083FH

Example 2 : To read the SP

LD HL, SP ; HL ← SP

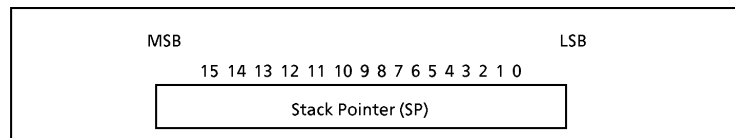


Figure 1-7. Stack Pointer

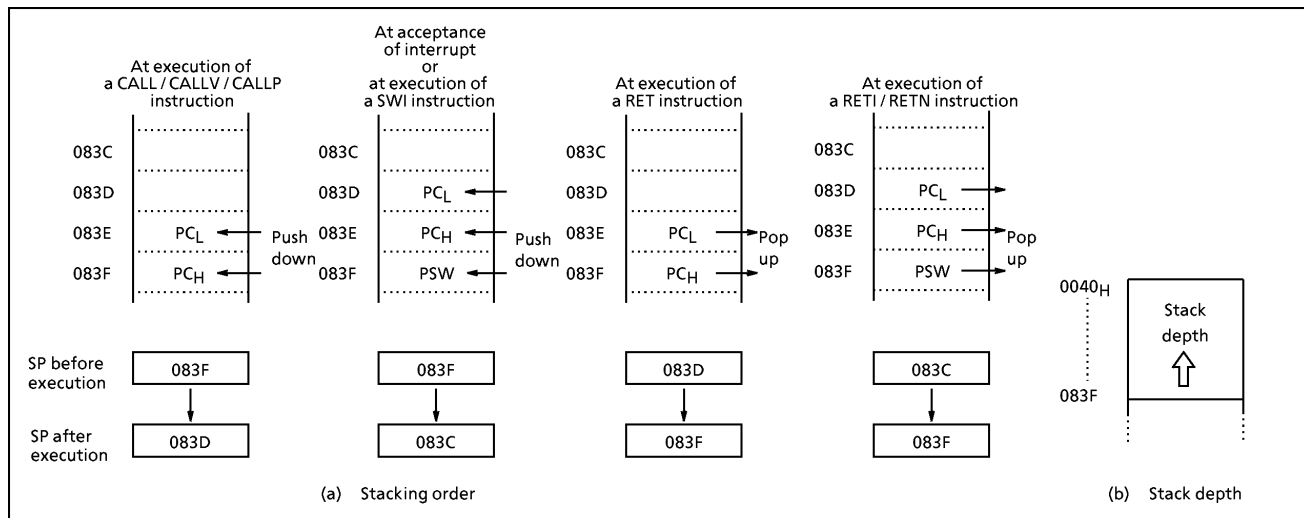


Figure 1-8. Stack

1.8 System Clock Controller

The system clock controller consists of a clock generator, a timing generator, and a stand-by controller.

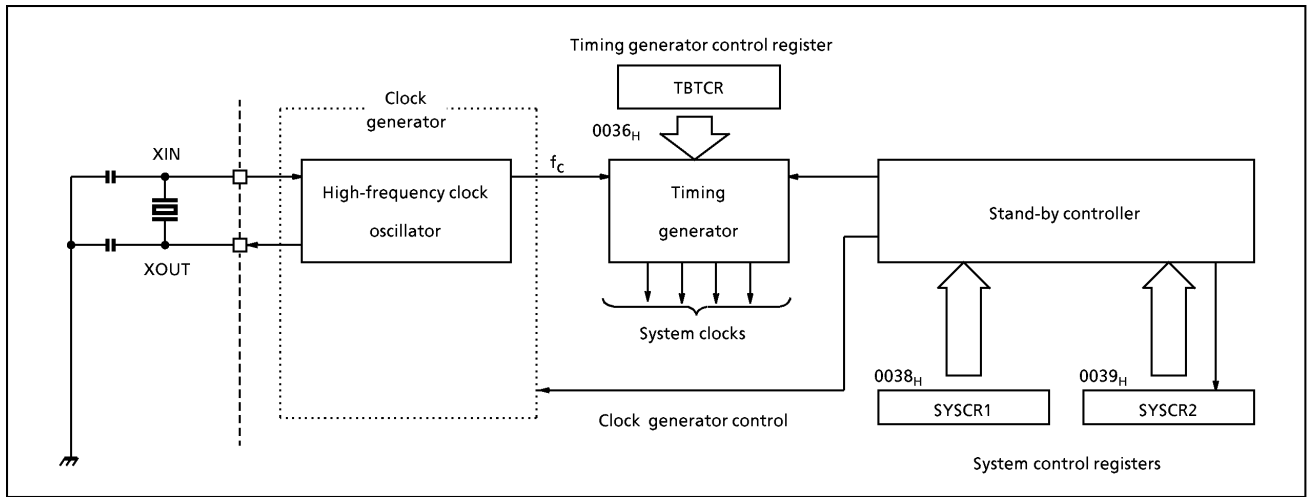


Figure 1-9. System Clock Controller

1.8.1 Clock Generator

The clock generator generates the basic clock which provides the system clocks supplied to the CPU core and peripheral hardware. It contains a oscillation circuit for the high-frequency clock.

The high-frequency (f_c) clock can be easily obtained by connecting a resonator between the XIN / XOUT pins, respectively. Clock input from an external oscillator is also possible. In this case, external clock is applied to the XIN pin with the XOUT pin not connected. The A8700CP / CS is not provided an RC oscillation.

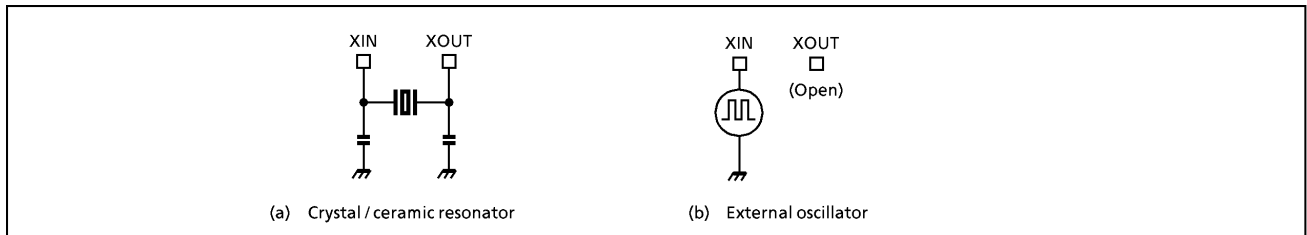


Figure 1-10. Examples of Resonator Connection

Note: *Accurate Adjustment of the Oscillation Frequency :*

Although no hardware to externally and directly monitor the basic clock pulse is not provided, the oscillation frequency can be adjusted by providing a program to output fixed frequency pulses to the port while disabling all interrupts and monitoring this pulse. With a system requiring adjustment of the oscillation frequency, the adjusting program must be created beforehand.

1.8.2 Timing Generator

The timing generator generates from the basic clock the various system clocks supplied to the CPU core and peripheral hardware. The timing generator provides the following functions :

- ① Generation of main system clock
- ② Generation of source clocks for time base timer
- ③ Generation of source clocks for watchdog timer
- ④ Generation of internal source clocks for time / counters TC1 to TC4
- ⑤ Generation of warm-up clocks for releasing STOP mode
- ⑥ Generation of a clock for releasing reset output

(1) Configuration of timing generator

The timing generator consists of a 21-stage divider with a divided-by-4 prescaler, a main system clock generator, and machine cycle counters, shown in Figure 1-11 as follows. During reset and upon releasing STOP mode, the divider is cleared to "0", however, the prescaler is not cleared.

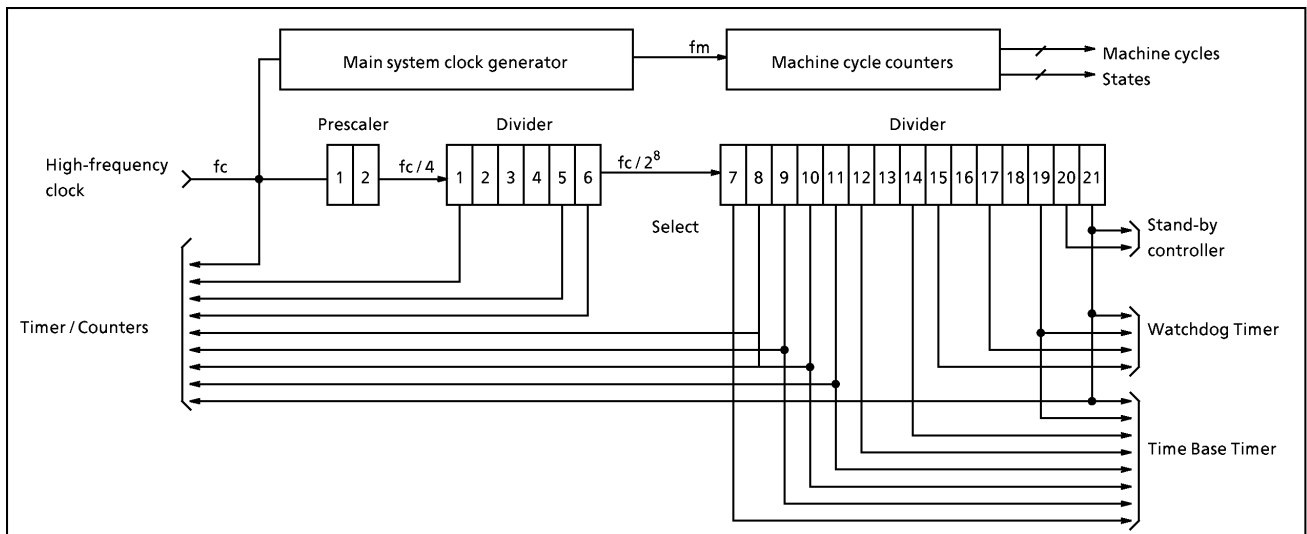


Figure 1-11. Configuration of Timing Generator

(2) Machine cycle

Instruction execution and peripherals operation are synchronized with the main system clock. The minimum instruction execution unit is called a "machine cycle". There are a total of 10 different types of instructions for the TLCS-870 Series : ranging from 1-cycle instructions which require one machine cycle for execution to 10-cycle instructions which require 10 machine cycles for execution. A machine cycle consists of 4 states (S_0 to S_3), and each state consists of one main system clock.

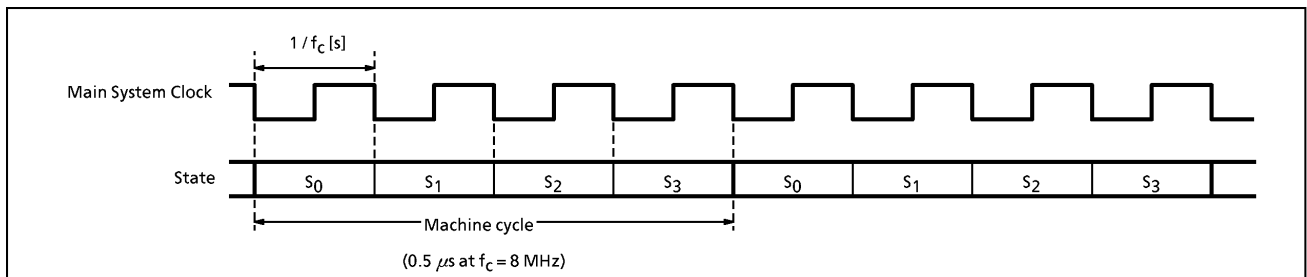


Figure 1-12. Machine Cycle

1.8.3 Stand-by Controller

The stand-by controller starts and stops the oscillation circuit for the high-frequency clock. Operating modes are controlled by the system control registers (SYSCR1, SYSCR2). Figure 1-13 shows the operating mode transition diagram and Figure 1-14 shows the system control registers.

(1) Operating mode

① NORMAL mode

In this mode, both the CPU core and on-chip peripherals operate.

② IDLE mode

In this mode, the internal oscillation circuit remains active. The CPU and the watchdog timer are halted ; however, on-chip peripherals remain active. IDLE mode is started by setting IDLE bit in the system control register 2 (SYSCR2), and IDLE mode is released to NORMAL mode by an interrupt request from on-chip peripherals or external interrupt inputs. When IMF (interrupt master enable flag) is "1" (interrupt enable), the execution will resume upon acceptance of the interrupt, and the operation will return to normal after the interrupt service is completed. When IMF is "0" (interrupt disable), the execution will resume with the next instruction which follows IDLE mode start instruction.

③ STOP mode

In this mode, the internal oscillation circuit is turned off, causing all system operations to be halted. The internal status immediately prior to the halt is held with the lowest power consumption during this mode. The output status of all output ports can be set to either output hold or high-impedance under software control.

STOP mode is started by setting STOP bit in the system control register 1 (SYSCR1), and STOP mode is released by an input (either level-sensitive or edge-sensitive can be programmably selected) to the $\overline{\text{STOP}}$ pin. After the warming-up period is completed, the execution resumes with the next instruction which follows the STOP mode start instruction.

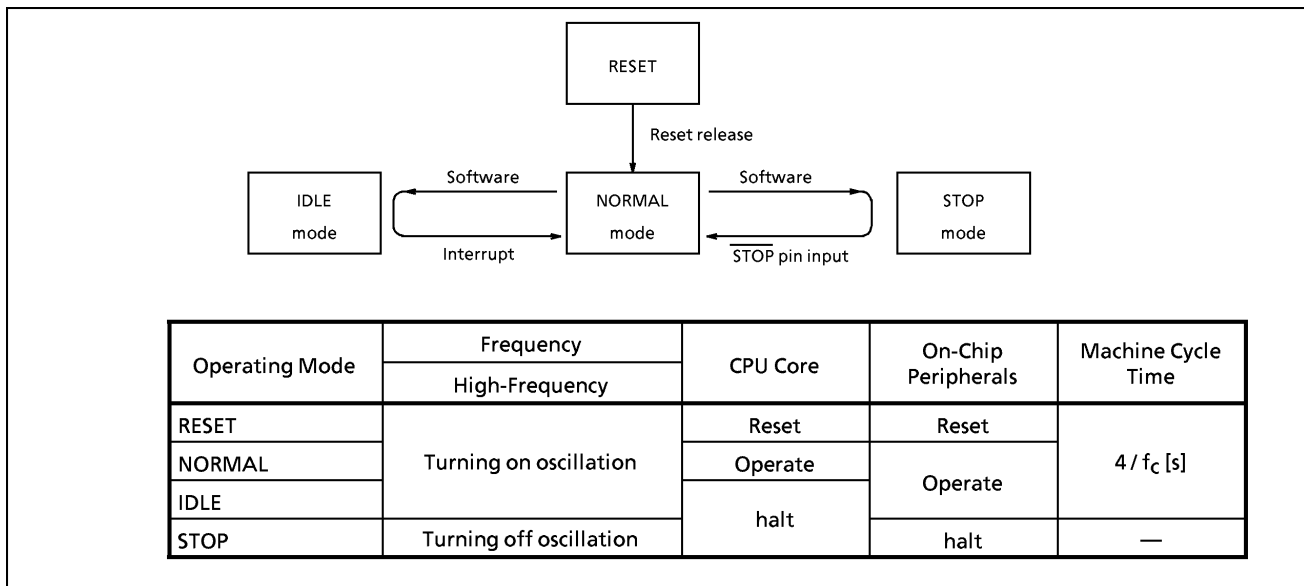


Figure 1-13. Operating Mode Transition Diagram

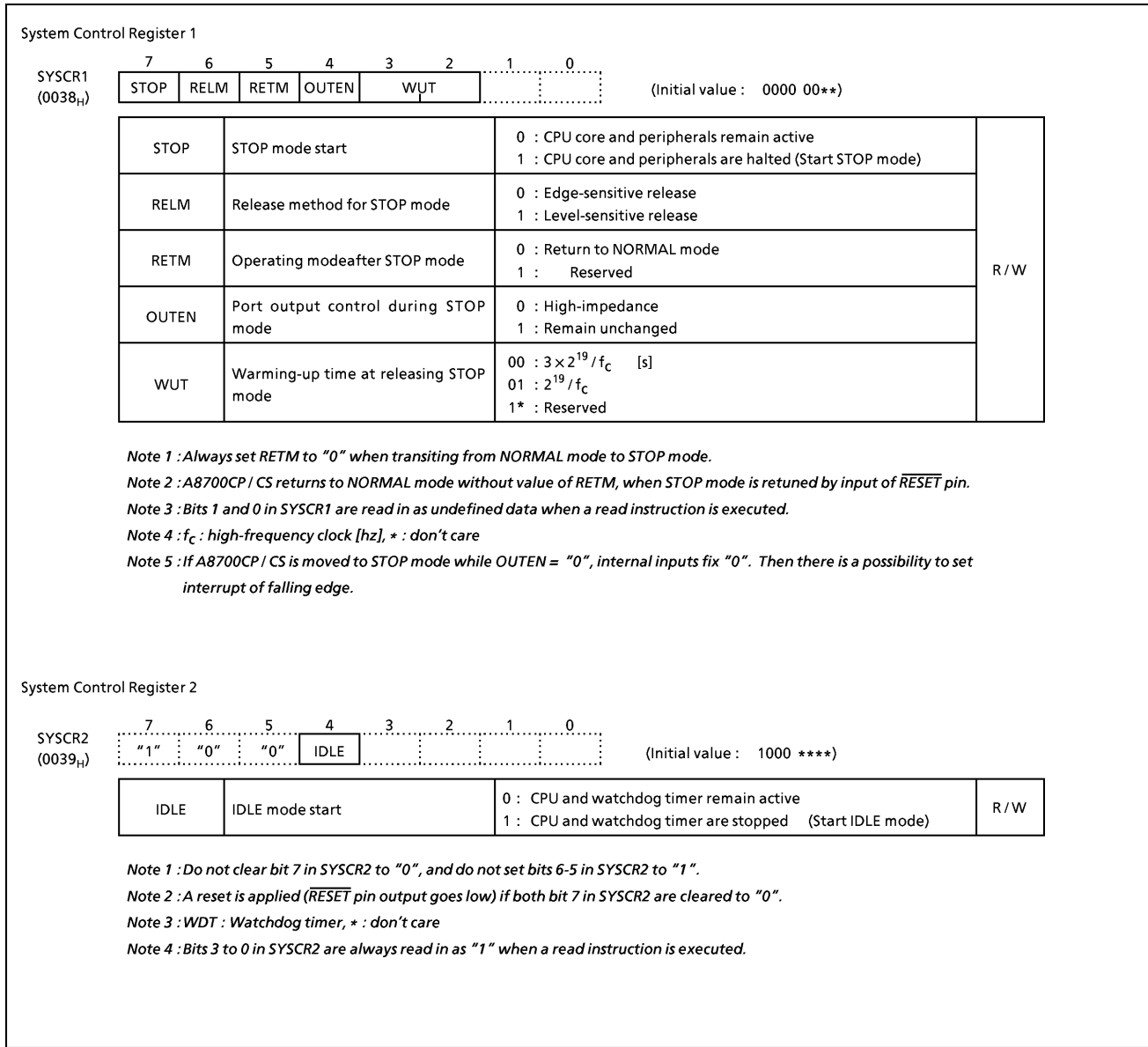


Figure 1-14. System Control Registers

1.8.4 Operating Mode Control

(1) STOP mode

STOP mode is controlled by the system control register 1 (SYSCR1) and the \overline{STOP} pin input. The \overline{STOP} pin is also used both as a port P20 and an $\overline{INT5}$ (external interrupt input 5) pin. STOP mode is started by setting STOP (bit 7 in SYSCR1) to "1". During STOP mode, the following status is maintained.

- ① Oscillation is turned off, and all internal operations are halted.
- ② The data memory, registers and port output latches are all held in the status in effect before STOP mode was entered. The port output can be select either output hold or high-impedance by setting OUTEN (bit 4 in SYSCR1).
- ③ The divider of the timing generator is cleared to "0".
- ④ The program counter holds the address of the instruction following the instruction which started STOP mode.

STOP mode includes a level-sensitive release mode and an edge-sensitive release mode, either of which can be selected with RELM (bit 6 in SYSCR1).

a. Level-sensitive release mode (RELM = 1)

In this mode, STOP mode is released by setting the \overline{STOP} pin high. This mode is used for capacitor back-up when the main power supply is cut off and for long term battery back-up.

When the \overline{STOP} pin input is high, executing an instruction which starts the STOP mode will not place in STOP mode but instead will immediately start the release sequence (warm-up). Thus, to start STOP mode in the level-sensitive release mode, it is necessary for the program to first confirm that the \overline{STOP} pin input is low. The following one method can be used for confirmation :

- Using an external interrupt input $\overline{INT5}$ ($\overline{INT5}$ is a falling edge-sensitive input).

Example : Starting STOP mode with an INT5 interrupt.

```

PINT5 : TEST (P2). 0           ; To reject noise, STOP mode does not start if
                                port P20 is at high
                                LD (SYSCR1), 01000000B       ; Sets up the level-sensitive release mode.
                                SET (SYSCR1). 7             ; Starts STOP mode
                                LDW (IL), 1110011101010111B ; IL12, 11, 7, 5, 3 ← 0 (Clears interrupt latches)
SINT5 : RETI

```

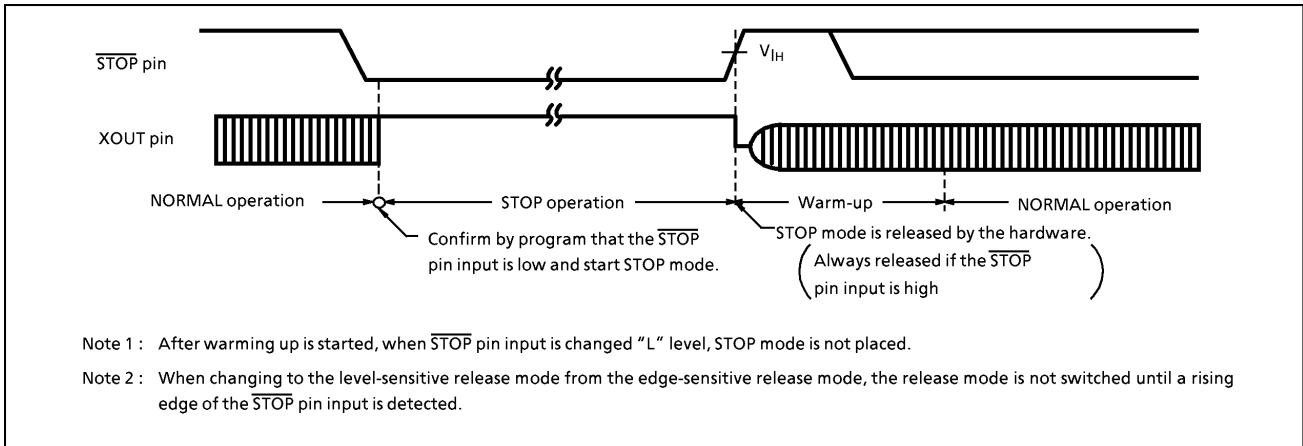


Figure 1-15. Level-Sensitive Release Mode

b. Edge-sensitive release mode (RELM = 0)

In this mode, STOP mode is released by a rising edge of the $\overline{\text{STOP}}$ pin input. This is used in applications where a relatively short program is executed repeatedly at periodic intervals. This periodic signal (for example, a clock from a low-power consumption oscillator) is input to the $\overline{\text{STOP}}$ pin. In the edge-sensitive release mode, STOP mode is started even when the $\overline{\text{STOP}}$ pin input is high.

Example : Starting STOP mode operation in the edge-sensitive release mode

```
LD (SYSCR1), 00000000B ; OUTEN ← 0 (Specifies high-impedance)
DI ; IMF ← 0 (Disables interrupt service)
SET (SYSCR1). STOP ; STOP ← 1 (Activates stop mode)
LDW (IL), 11100111010111B ; IL12, 11, 7, 5, 3 ← 0 (Clears interrupt latches)
EI ; IMF ← 1 (Enables interrupt service)
```

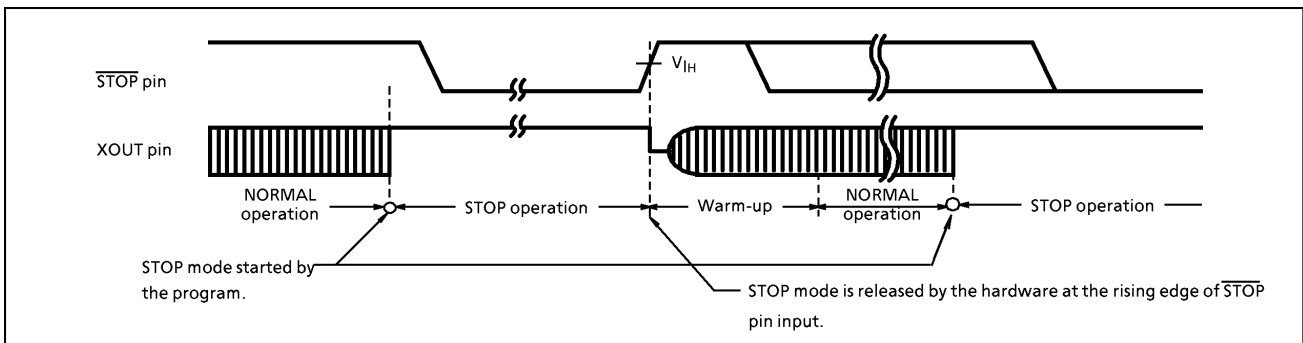


Figure 1-16. Edge-Sensitive Release Mode

STOP mode is released by the following sequence :

- ① The high-frequency clock oscillator is turned on.
- ② A warming-up period is inserted to allow oscillation time to stabilize. During warm-up, all internal operations remain halted. Two different warming-up times can be selected with WUT (bits 2 and 3 in SYSCR1) as determined by the resonator characteristics.
- ③ When the warming-up time has elapsed, normal operation resumes with the instruction following the STOP mode start instruction (e.g. [SET (SYSCR1). 7]). The start is made after the divider of the timing generator is cleared to "0".

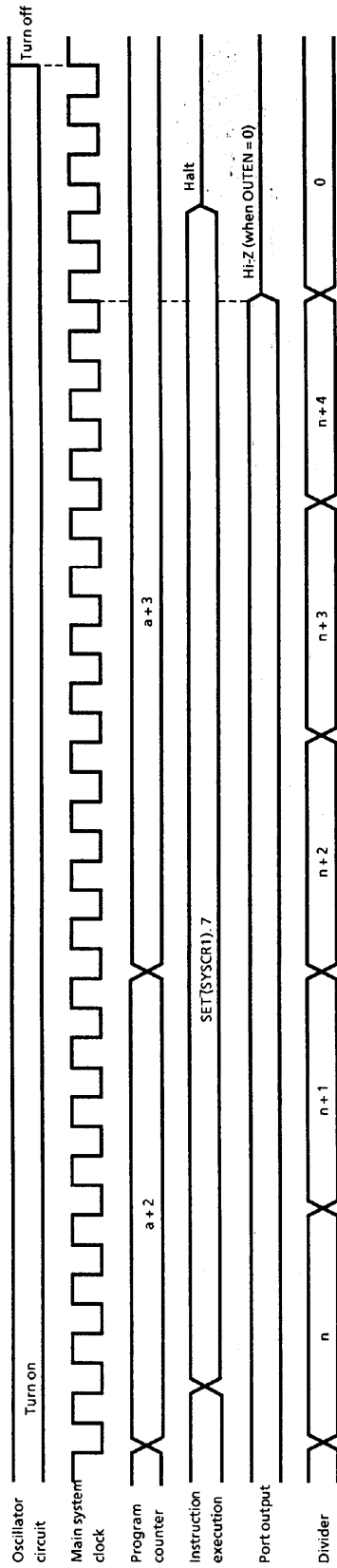
Table 1-1. Warming-Up Time Example

WUT	$f_c = 4.194304 \text{ MHz}$	$f_c = 8 \text{ MHz}$
$3 \times 2^{19} / f_c \text{ [s]}$	375 [ms]	196.6 [ms]
$2^{19} / f_c$	125	65.5

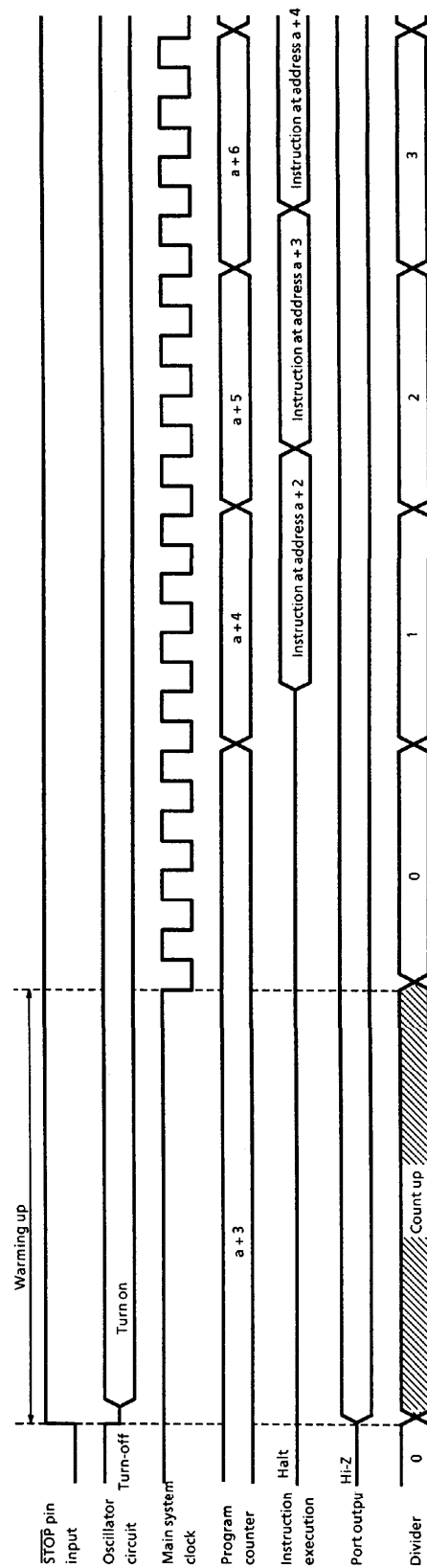
Note: The warming-up time is obtained by dividing the basic clock by the divider : therefore, the warming-up time may include a certain amount of error if there is any fluctuation of the oscillation frequency when STOP mode is released. Thus, the warming-up time must be considered an approximate value.

STOP mode can also be released by setting the $\overline{\text{RESET}}$ pin low, which immediately performs the normal reset operation.

Note: When STOP mode is released with a low hold voltage, the following cautions must be observed.
The power supply voltage must be at the operating voltage level before releasing the STOP mode. The $\overline{\text{RESET}}$ pin input must also be high, rising together with the power supply voltage. In this case, if an external time constant circuit has been connected, the $\overline{\text{RESET}}$ pin input voltage will increase at a slower rate than the power supply voltage. At this time, there is a danger that a reset may occur if input voltage level of the $\overline{\text{RESET}}$ pin drops below the non-inverting high-level input voltage (hysteresis input).



(a) STOP mode start (Example : start with SET (SYSCR1).7 instruction located at address a)



(b) STOP mode release

Figure 1-17. STOP Mode Start / Release

(2) IDLE mode

IDLE mode is controlled by the system control register 2 and maskable interrupts. The following status is maintained during IDLE mode.

- ① Operation of the CPU and watchdog timer is halted. On-chip peripherals continue to operate.
- ② The data memory, CPU registers and port output latches are all held in the status in effect before IDLE mode was entered.
- ③ The program counter holds the address of the instruction following the instruction which started IDLE mode.

Example : Starting IDLE mode.

```
SET (SYSCR2). 4 ; IDLE←1
```

IDLE mode includes a normal release mode and an interrupt release mode. Selection is made with the interrupt master enable flag (IMF). Releasing the IDLE mode returns to NORMAL mode.

a. Normal release mode (IMF = "0")

IDLE mode is released by any interrupt source enabled by the individual interrupt enable flag (EF). Execution resumes with the instruction following the IDLE mode start instruction (e.g. [SET (SYSCR2). 4]). Normally, IL (Interrupt Latch) of interrupt source to release IDEL mode must be cleared by load instructions.

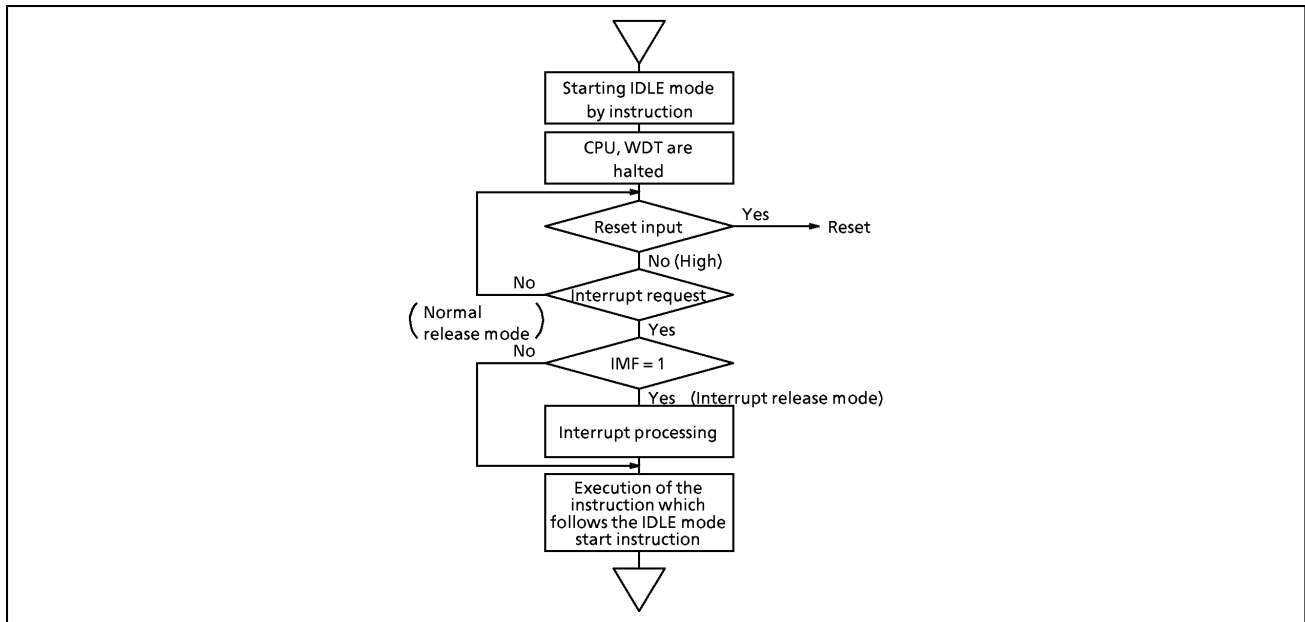


Figure 1-18. IDLE Mode

b. Interrupt release mode (IMF = "1")

IDLE mode is released and interrupt processing is started by any interrupt source enabled with the individual interrupt enable flag (EF). After the interrupt is processed, the execution resumes from the instruction following the instruction which started IDLE mode.

IDLE mode can also be released by setting the RESET pin low, which immediately performs the reset operation. After reset, the A8700CH / CK / CM / CP / CS are placed in NORMAL mode.

Note: When a watchdog timer interrupt is generated immediately before IDLE mode is started, the watchdog timer interrupt will be processed but IDLE mode will not be started.

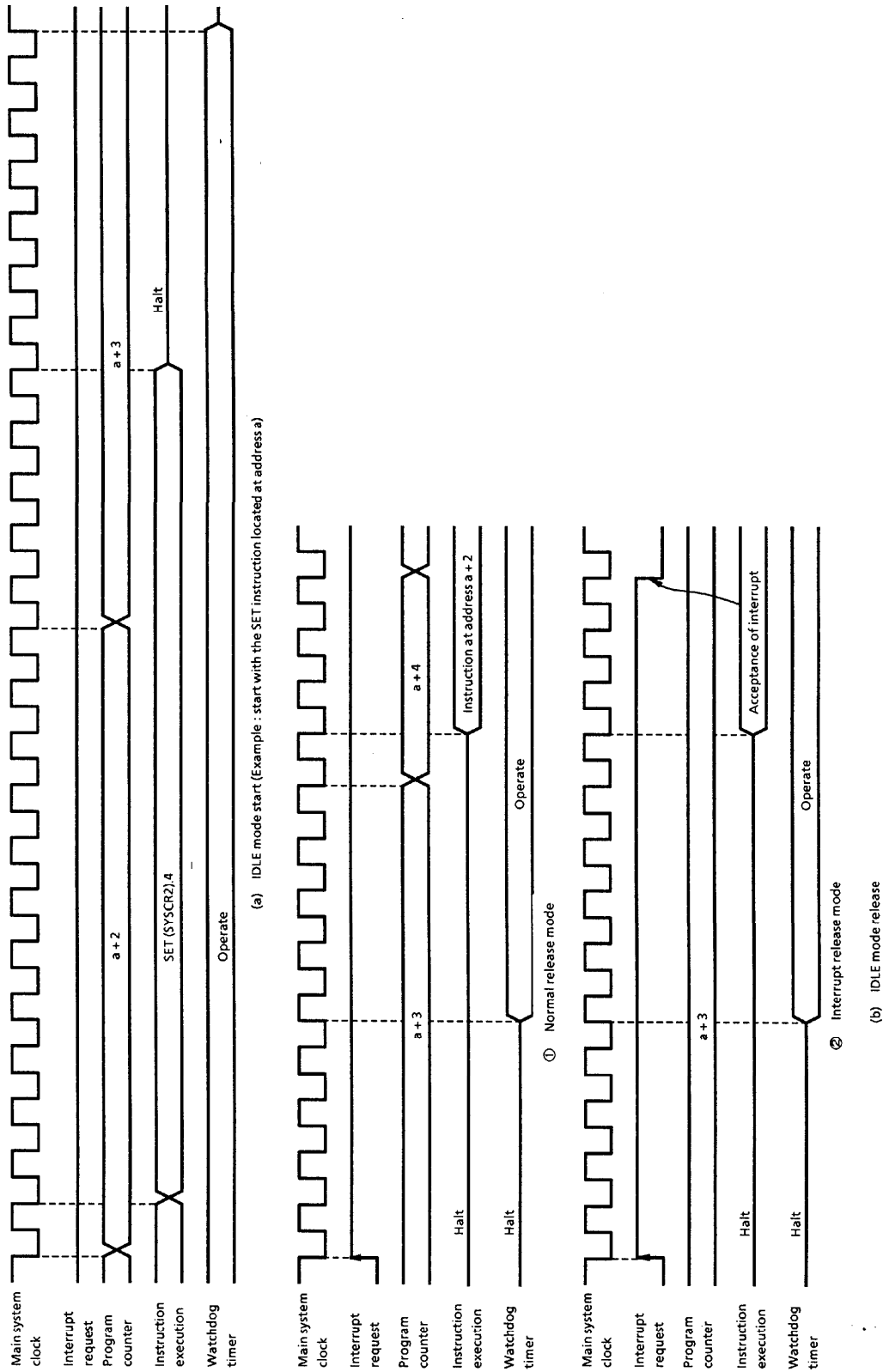


Figure 1-19. IDLE Mode Start / Release

1.9 Interrupt Controller

The A8700CH / CK / CM / CP / CS has a total of 14 interrupt sources : 5 externals and 9 internals. Nested interrupt control with priorities is also possible. Two of the internal sources are pseudo non-maskable interrupts ; the remainder are all maskable interrupts.

Interrupt latches (IL) that hold the interrupt requests are provided for interrupt sources. Each interrupt vector is independent. The interrupt latch is set to "1" when an interrupt request is generated and requests the CPU to accept the interrupt. The acceptance of maskable interrupts can be selectively enabled and disabled by the program using the interrupt master enable flag (IMF) and the individual interrupt enable flags (EF). When two or more interrupts are generated simultaneously, the interrupt is accepted in the highest priority order as determined by the hardware. Figure 1-20 shows the interrupt controller.

Table 1-2. Interrupt Sources

Interrupt Source		Enable Condition	Interrupt Latch	Vector Table Address	Priority
Internal / External	(Reset)	Non-Maskable	—	FFFE _H	High 0
Internal	INTSW (Software interrupt)	Pseudo non-maskable	—	FFFCH	1
Internal	INTWDT (Watchdog Timer interrupt)		IL ₂	FFFA _H	2
External	INT0 (External interrupt 0)	IMF = 1, INTOEN = 1	IL ₃	FFF8 _H	3
Internal	INTTC1 (16 bit TC1 interrupt)	IMF·EF ₄ = 1	IL ₄	FFF6 _H	4
Reserved		IMF·EF ₅ = 1	IL ₅	FFF4 _H	5
Internal	INTTBT (Time Base Timer interrupt)	IMF·EF ₆ = 1	IL ₆	FFF2 _H	6
External	INT2 (External interrupt 2)	IMF·EF ₇ = 1	IL ₇	FFF0 _H	7
Internal	INTTC3 (8 bit TC3 interrupt)	IMF·EF ₈ = 1	IL ₈	FFEE _H	8
Internal	INTSBI (Serial bus Interface interrupt)	IMF·EF ₉ = 1	IL ₉	FFEC _H	9
Internal	INTTC4 (8 bit TC4 interrupt)	IMF·EF ₁₀ = 1	IL ₁₀	FFEA _H	10
External	INT3 (External interrupt 3)	IMF·EF ₁₁ = 1	IL ₁₁	FFE8 _H	11
External	INT4 (External interrupt 4)	IMF·EF ₁₂ = 1	IL ₁₂	FFE6 _H	12
Internal	INTOSD (OSD interrupt / SLICER interrupt)	IMF·EF ₁₃ = 1	IL ₁₃	FFE4 _H	13
Internal	INTTC2 (16 bit TC2 interrupt)	IMF·EF ₁₄ = 1	IL ₁₄	FFE2 _H	14
External	INT5 (External interrupt 5)	IMF·EF ₁₅ = 1	IL ₁₅	FFE0 _H	Low 15

(1) Interrupt latches (IL₁₅ to 2)

Interrupt latches are provided for each source, except for a software interrupt. The latch is set to "1" when an interrupt request is generated, and requests the CPU to accept the interrupt. The latch is cleared to "0" just after the interrupt is accepted. All interrupt latches except INT3 are initialized to "0" during reset. The interrupt latch of INT3 is unstable during reset.

The interrupt latches are assigned to addresses 003C_H and 003D_H in the SFR. Each latch can be cleared to "0" individually by an instruction ; however, the read-modify-write instruction such as bit manipulation or operation instructions cannot be used (Do not clear the I_{L2} for a watch dog timer interrupt to "0"). Thus, interrupt requests can be cancelled and initialized by the program. Note that interrupt latches cannot be set to "1" by any instruction.

The contents of interrupt latches can be read out by an instruction. Therefore, testing interrupt requests by software is possible.

Example 1 : Clears interrupt latches

```
LDW    (IL), 1110101010111111B    ; IL12, IL10, IL8, IL6 ← 0
```

Example 2 : Reads interrupt latches

```
LD     WA, (IL)                    ; W ← ILH, A ← ILL
```

Example 3 : Tests an interrupt latch

```
TEST   (ILH). 4                    ; IL12 = 1 then jump
JR     F, SSET
```

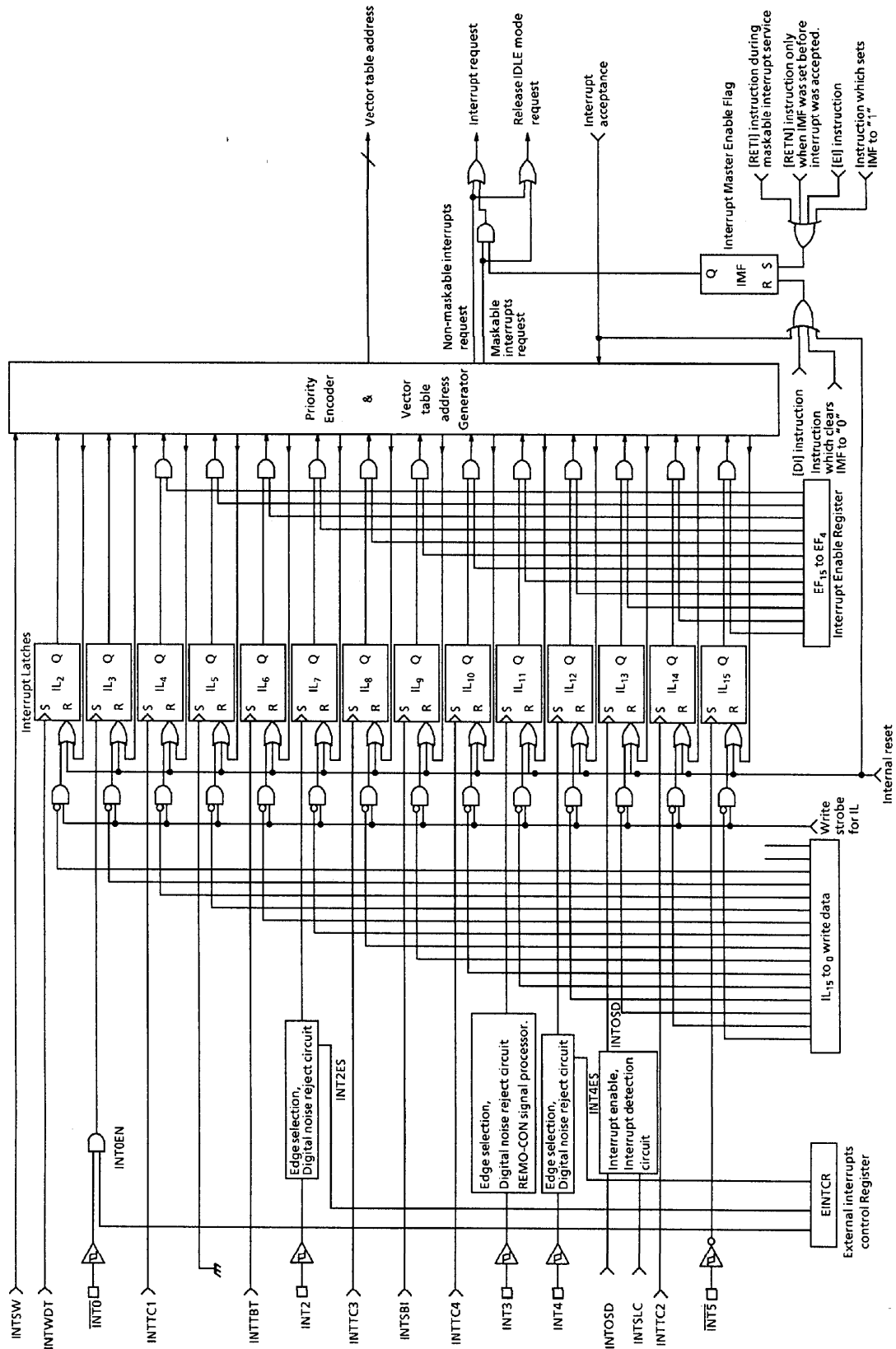


Figure 1-20. Interrupt Controller Block Diagram

(2) Interrupt enable register (EIR)

The interrupt enable registers (EIR) enable and disable the acceptance of interrupts, except for the pseudo non-maskable interrupts (software and watchdog timer interrupts). Pseudo non-maskable interrupts are accepted regardless of the contents of the EIR ; however, the pseudo non-maskable interrupts cannot be nested more than once at the same time. For example, the watchdog timer interrupt is not accepted during the software interrupt service.

The EIR consists of an interrupt master enable flag (IMF) and individual interrupt enable flags (EF). These registers are assigned to addresses 003A_H and 003B_H in the SFR, and can be read and written by an instruction (including read-modify-write instructions such as bit manipulation instructions).

① Interrupt master enable flag (IMF)

The interrupt master enable flag (IMF) enables and disables the acceptance of all interrupts, except for pseudo non-maskable interrupts. Clearing this flag to "0" disables the acceptance of all maskable interrupts. Setting to "1" enables the acceptance of interrupts.

When an interrupt is accepted, this flag is cleared to "0" to temporarily disable the acceptance of maskable interrupts. After execution of the interrupt service program, this flag is set to "1" by the maskable interrupt return instruction [RETI] to again enable the acceptance of interrupts. If an interrupt request has already been occurred, interrupt service starts immediately after execution of the [RETI] instruction.

Pseudo non-maskable interrupts are returned by the [RETN] instruction. In this case, the IMF is set to "1" only when pseudo non-maskable interrupt service is started with interrupt acceptance enabled (IMF = 1). Note that the IMF remains "0" when cleared by the interrupt service program.

The IMF is assigned to bit 0 at address 003A_H in the SFR, and can be read and written by an instruction. The IMF is normally set and cleared by the [EI] and [DI] instructions, and the IMF is initialized to "0" during reset.

② Individual interrupt enable flags (EF₁₅ to EF₄)

These flags enable and disable the acceptance of individual maskable interrupts. Setting the corresponding bit of an individual interrupt enable flag to "1" enables acceptance of an interrupt, setting the bit to "0" disables acceptance.

Example 1 : Sets EF for individual interrupt enable, and sets IMF to "1".

```
LDW (EIR), 111010000000001B ; EF15 to EF13, EF11, IMF ← 1
```

Example 2 : Sets an individual interrupt enable flag to "1".

```
SET (EIRH). 4 ; EF12 ← 1
```

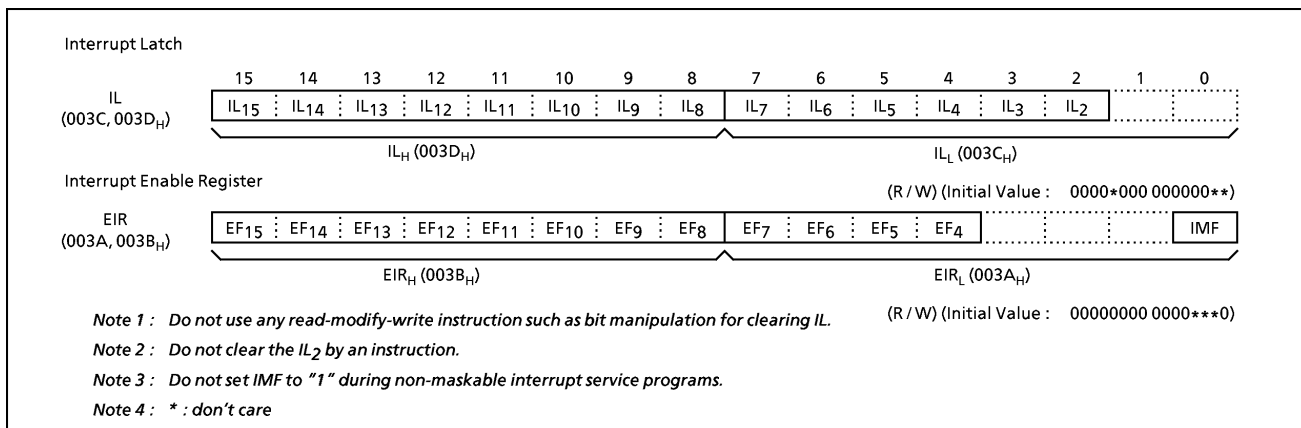


Figure 1-21. Interrupt Latch (IL) and Interrupt Enable Register (EIR)

1.9.1 Interrupt Sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to "0" by a reset or an instruction. Interrupt acceptance sequence requires 8 machine cycles (4 μs at f_c = 8 MHz in NORMAL mode) after the completion of the current instruction execution. The interrupt service task terminates upon execution of an interrupt return instruction [RETI] (for maskable interrupts) or [RETN] (for pseudo non-maskable interrupts).

(1) Interrupt acceptance processing is as follows :

- ① The interrupt master enable flag (IMF) is cleared to "0" to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.
- ② The interrupt latch (IL) for the interrupt source accepted is cleared to "0".
- ③ The contents of the program counter (return address) and the program status word are saved (pushed) onto the stack. The stack pointer is decremented 3 times.
- ④ The entry address of the interrupt service program is read from the vector table address, and the entry address is loaded to the program counter.
- ⑤ The instruction stored at the entry address of the interrupt service program is executed.

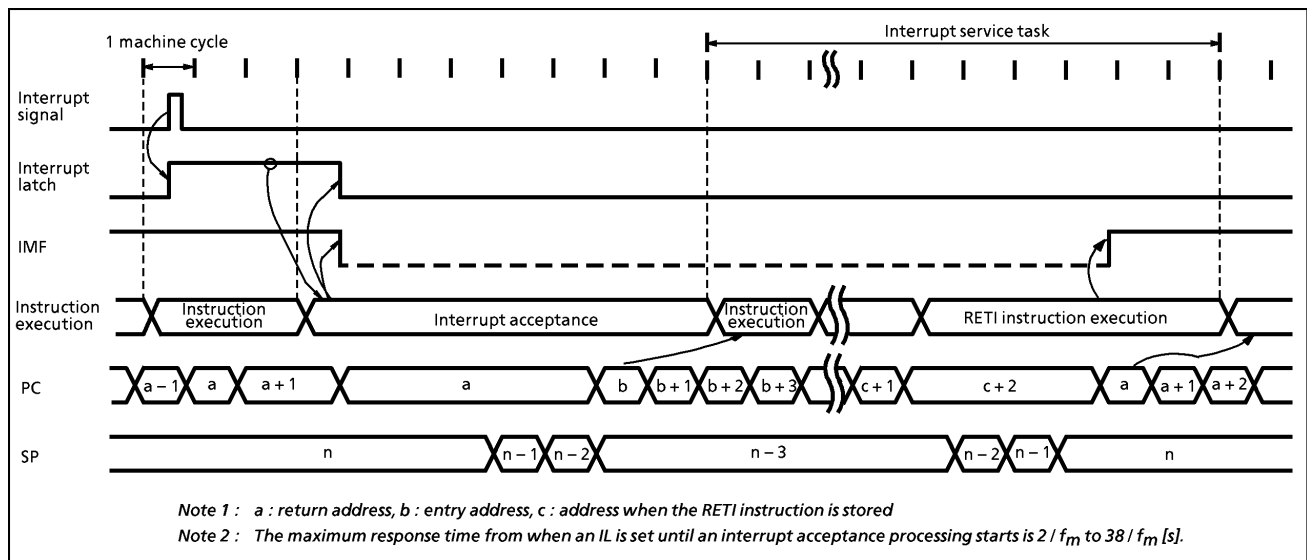
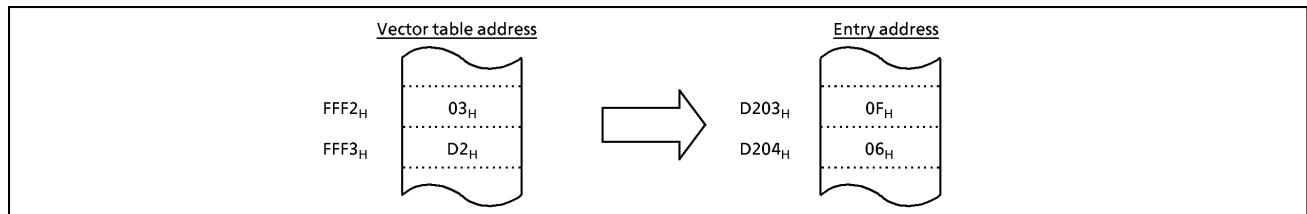


Figure 1-22. Timing Chart of Interrupt Acceptance and Interrupt Return Instruction

Example : Correspondence between vector table address for INTTBT and the entry address of the interrupt service program.



A maskable interrupt is not accepted until the IMF is set to "1" even if a maskable interrupt of higher priority than that of the current interrupt being serviced.

When nested interrupt service is necessary, the IMF is set to "1" in the interrupt service program. In this case, acceptable interrupt sources are selectively enabled by the individual interrupt enable flags.

(2) Saving / restoring general-purpose register

During interrupt acceptance processing, the program counter and the program status word are automatically saved on the stack, but not the accumulator and other registers. These registers are saved by the program if necessary. Also, when nesting multiple interrupt services, it is necessary to avoid using the same data memory area for saving registers.

The following method is used to save / restore the general-purpose registers :

① General-purpose register save / restore by register bank changeover :

General-purpose registers can be saved at high-speed by switching to a register bank that is not in use. Normally, bank 0 is used for the main task and banks 1 to 15 are assigned to interrupt service tasks. To increase the efficiency of data memory utilization, the same bank is assigned for interrupt sources which are not nested. The switched bank is automatically restored by executing an interrupt return instruction [RETI] or [RETN]. Therefore, it is not necessary for a program to save the RBS.

Example : Register Bank Changeover

```

PINTxx : LD      RBS, n      ; Switches to bank n (1 μs at 8 MHz)
          Interrupt processing
          RETI              ; Restores bank and Returns
    
```

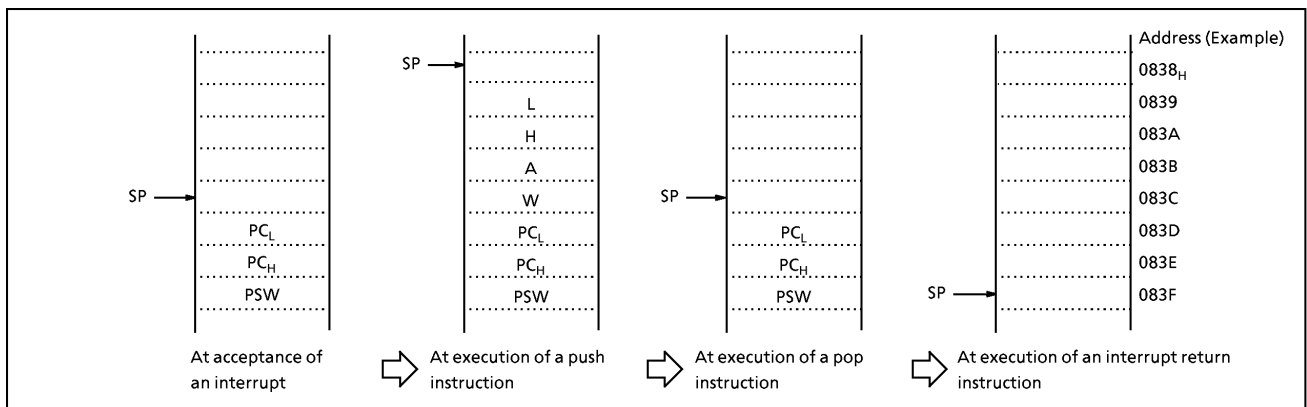
② General-purpose register save / restore using push and pop instructions :

To save only a specific register, and when the same interrupt source occurs more than once, the general-purpose registers can be saved / restored using push / pop instructions.

Example : Register save using push and pop instructions

```

PINTxx : PUSH   WA          ; Save WA register pair
          PUSH   HL          ; Save HL register pair
          Interrupt processing
          POP    HL          ; Restore HL register pair
          POP    WA          ; Restore WA register pair
          RETI              ; Return
    
```



- ③ General-purpose registers save / restore using data transfer instructions :
Data transfer instructions can be used to save only a specific general-purpose register during processing of a single interrupt.

Example : Saving / restoring a register using data transfer instructions

```

PINTxx : LD      (GSAVA), A    ; Save A register
          LD      A, (GSAVA)   ; Restore A register
          RETI                  ; Return
    
```

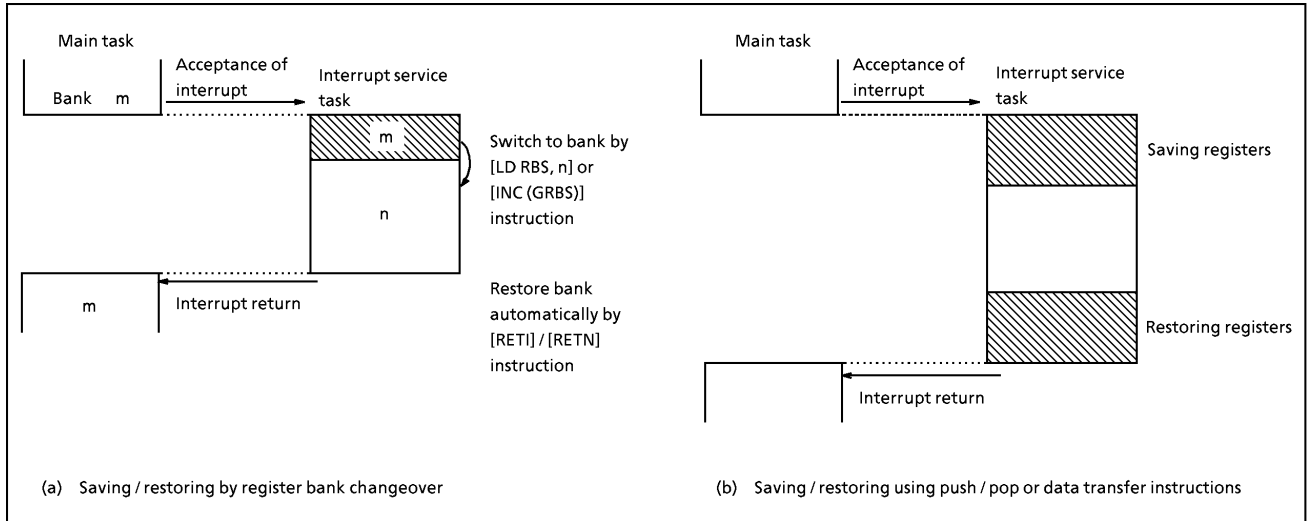


Figure 1-23. Saving / Restoring General-Purpose Registers

- (3) The interrupt return instructions [RETI] / [RETN] perform the following operations.

[RETI] Maskable Interrupt Return	[RETN] Non-Maskable Interrupt Return
① The contents of the program counter and the program status word are restored from the stack.	① The contents of the program counter and program status word are restored from the stack.
② The stack pointer is incremented 3 times.	② The stack pointer is incremented 3 times.
③ The interrupt master enable flag is set to "1".	③ The interrupt master enable flag is set to "1" only when a non-maskable interrupt is accepted in interrupt enable status. However, the interrupt master enable flag remains at "0" when so clear by an interrupt service program.

Interrupt requests are sampled during the final cycle of the instruction being executed. Thus, the next interrupt can be accepted immediately after the interrupt return instruction is executed.

Note: When the interrupt processing time is longer than the interrupt request generation time, the interrupt service task is performed but not the main task.

1.9.2 Software Interrupt (INTSW)

Executing the [SWI] instruction generates a software interrupt and immediately starts interrupt processing (INTSW is highest prioritized interrupt). However, if processing of a non-maskable interrupt is already underway, executing the SWI instruction will not generate a software interrupt but will result in the same operation as the [NOP] instruction. Thus, the [SWI] instruction behaves like the [NOP] instruction.

Note: At the development tool, if processing of a non-maskable interrupt is already underway, executing the SWI instruction will generate a software interrupt as a software brake.

Use the [SWI] instruction only for detection of the address error or for debugging.

① Address error detection

FF_H is read if for some cause such as noise the CPU attempts to fetch an instruction from a non-existent memory address. Code FF_H is the SWI instruction, so a software interrupt is generated and an address error is detected. The address error detection range can be further expanded by writing FF_H to unused areas of the program memory. The address trap reset is generated in case that an instruction fetch from a port of RAM area or SER area.

Note: The fetch data from addresses 7F80_H to 7FFF_H (test ROM area) is not "FF_H".

② Debugging

Debugging efficiency can be increased by placing the SWI instruction at the software break point setting address.

1.9.3 External Interrupt

The A8700CH / CK / CM / CP / CS have five external interrupt inputs ($\overline{INT0}$, INT2, INT3, INT4, and $\overline{INT5}$). Three of these are equipped with digital noise reject circuits (pulse inputs of less than a certain time are eliminated as noise). Edge selection is also possible with INT2, INT3 and INT4.

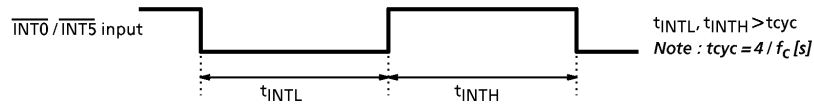
Edge selection and noise rejection control for INT3 pin input are performed by the Remote control signal processor control registers (refer to the selection of the Remote control signal processor.)

Edge selection for INT2 and INT4 are performed by the external interrupt control register.

Table 1-3. External Interrupts

Source	Pin	Enable Conditions	Enable Conditions	Edge	Digital Noise Reject
INT0	$\overline{INT0}$	P50 / $\overline{PWM8}$	IMF = 1, INT0EN = 1	Falling edge	— (Hysteresis input)
INT2	INT2	P53 / TC1	IMF·EF ₇ = 1	Falling edge or rising edge	Pulses of less than $7 / f_c$ [s] are eliminated as noise. Pulse of $24 / f_c$ [s] or more are considered to be signals.
INT3	INT3	P30 / RX1N	IMF·EF ₁₁ = 1	Falling edge, rising edge or falling / rising edge	Refer to the section of the Remote control signal preprocessor.
INT4	INT4	P31 / TC3	IMF·EF ₁₂ = 1	Falling edge or rising edge	Pulses of less than $7 / f_c$ [s] are eliminated as noise. Pulse of $24 / f_c$ [s] or more are considered to be signals.
INT5	$\overline{INT5}$	P20 / \overline{STOP}	IMF·EF ₁₅ = 1	Falling edge	— (Hysteresis input)

Note 1: The pulse width (both "H" and "L" level) for input to the $\overline{INT0}$ and $\overline{INT5}$ pins must be over 1 machine cycle.



Note 2: If a noiseless signal is input to the external interrupt pin in the NORMAL or IDLE mode, the maximum time from the edge of input signal until the IL is set is as follows :

INT2, INT3, INT4 pins $25 / f_c$ [s]

Note 3: The noise reject function is also attected for timer / counter input (TC1 and TC3 pins)

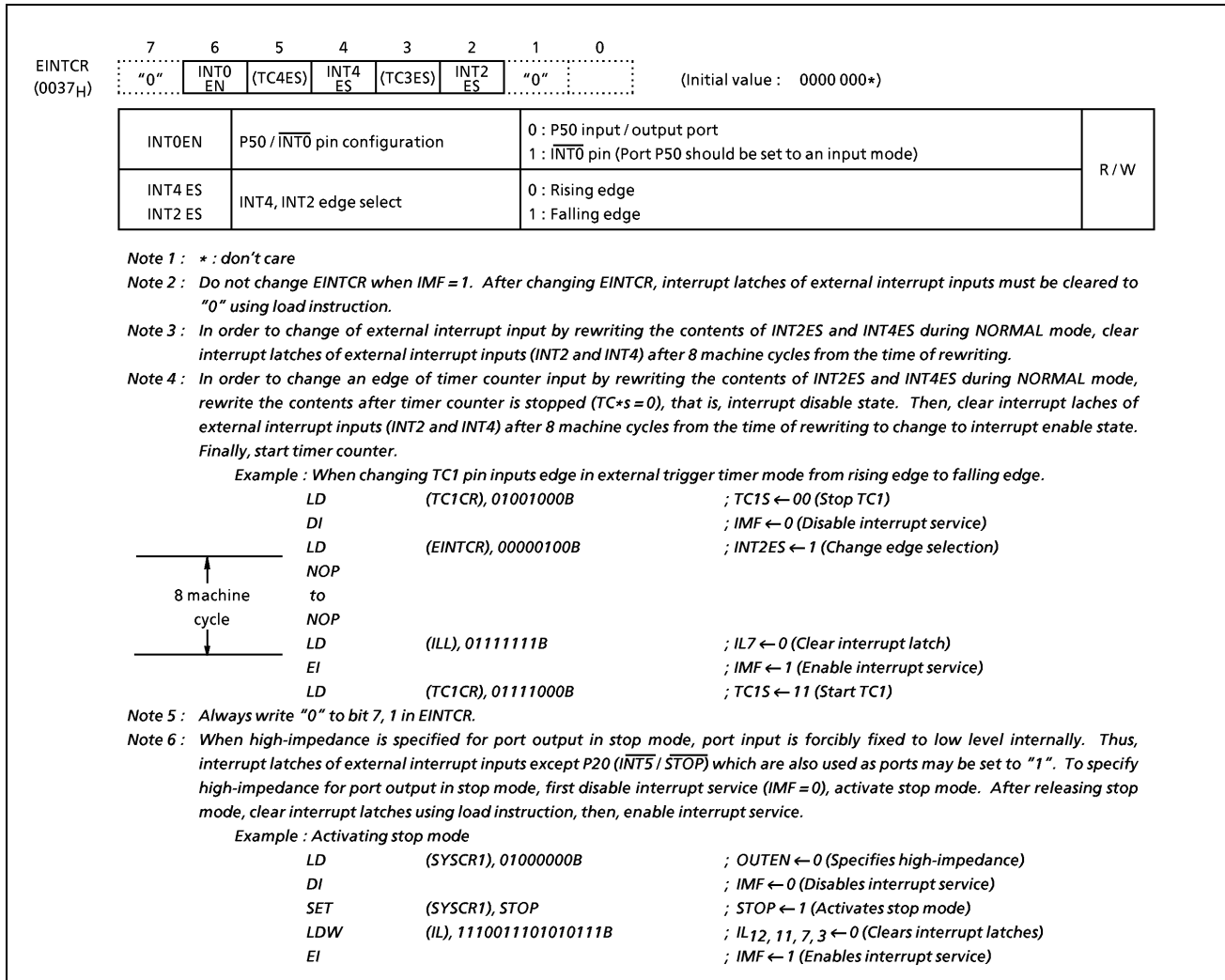


Figure 1-24. External Interrupt Control Register

1.10 Watchdog Timer (WDT)

The watchdog timer rapidly detects the CPU malfunction such as endless looping caused by noise or the like, and resumes the CPU to the normal state.

The watchdog timer signal for detecting malfunction can be selected either as a reset output or a non-maskable interrupt request. However, selection is possible only once after reset. At first, the reset output is selected.

When the watchdog timer is not being used for malfunction detection, it can be used as a timer to generate an interrupt at fixed intervals.

1.10.1 Watchdog Timer Configuration

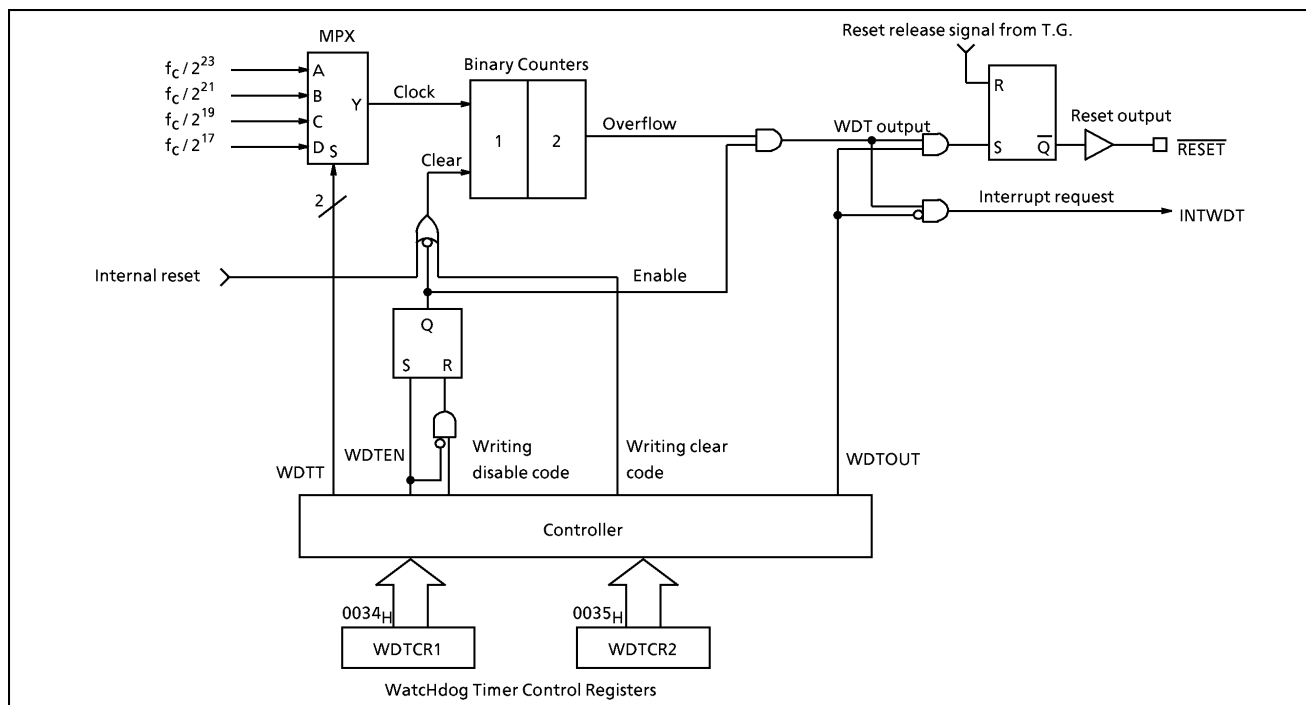


Figure 1-25. Watchdog Timer Configuration

1.10.2 Watchdog Timer Control

Figure 1-26 shows the watchdog timer control registers (WDTTCR1, WDTTCR2). The watchdog timer is automatically enabled after reset.

(1) Malfunction detection methods using the watchdog timer

The CPU malfunction is detected as follows :

- ① Setting the detection time, selecting output, and clearing the binary counter.
- ② Repeatedly clearing the binary counter within the setting detection time.

If a CPU malfunction occurs for any cause, the watchdog timer output will become active on the rise of an overflow from the binary counters unless the binary counters are cleared. At this time, when $WDTOUT = 1$ a reset is generated, which drives the \overline{RESET} pin low to reset the internal hardware and the external circuits. When $WDTOUT = 0$, a watchdog timer interrupt (INTWDT) is generated.

The watchdog timer temporarily stops counting in STOP mode (including warm-up) or IDLE mode, and automatically restarts (continues counting) when STOP / IDLE mode is released.

Example : Sets the watchdog timer detection time to $2^{21} / f_c$ [s] and resets the CPU malfunction.

```

LD (WDTCR2), 4Eh ; Clears the binary counters
LD (WDTCR1), 00001101B ; WDTT ← 10, WDTOUT ← 1
LD (WDTCR2), 4Eh ; Clears the binary counters
                    (Always clear immediately after changing
                    WDTT)
LD (WDTCR2), 4Eh ; Clears the binary counters
LD (WDTCR2), 4EH ; Clears the binary counters
    
```

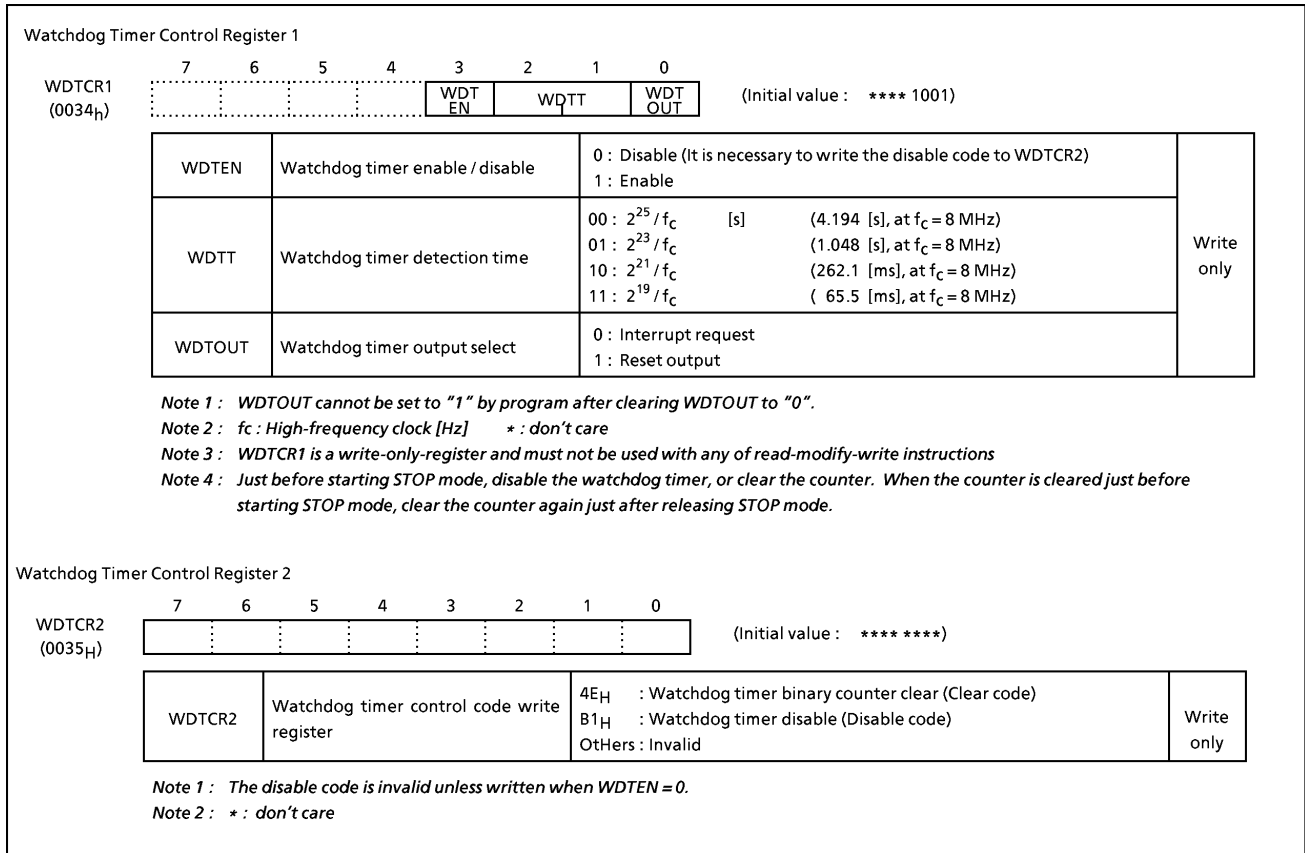


Figure 1-26. Watchdog Timer Control Registers

(2) Watchdog timer enable

The watchdog timer is enabled by setting WDTEN (bit 3 in WDTCR1) to "1". WDTEN is initialized to "1" during reset, so the watchdog timer operates immediately after reset is released.

Example : Enables watchdog timer

```
LD (WDTCR1), 00001000B ; WDTEN ← 1
```

(3) Watchdog timer disable

The watchdog timer is disabled by writing the disable code (B1_H) to WDTCR2 after clearing WDTEN (bit 3 in WDTCR1) to "0". The watchdog timer is not disabled if this procedure is reversed and the disable code is written to WDTCR2 before WDTEN is cleared to "0". The watchdog timer is halted temporarily in STOP mode (including warm-up) and IDLE mode, and restarts automatically after STOP or IDLE mode is released. During disabling the watchdog timer, the binary counters are cleared to "0".

Example : Disables watchdog timer

```
LDW (WDTCR1), 0B101H ; WDTEN ← 0, WDTCR2 ← disable code
```

1.10.3 Watchdog Timer Interrupt (INTWDT)

This is a pseudo non-maskable interrupt which can be accepted regardless of the contents of the EIR. If a watchdog timer interrupt or a software interrupt is already accepted, however, the new watchdog timer interrupt waits until the previous non-maskable interrupt processing is completed (the end of the [RETN] instruction execution).

The stack pointer (SP) should be initialized before using the watchdog timer output as an interrupt source with WDTOUT.

Example : Watchdog timer interrupt setting up.

```
LD SP, 043FH ; Sets the stack pointer
LD (WDTCR1), 00001000B ; WDTOUT ← 0
```

1.10.4 Watchdog Timer Reset

If the watchdog timer output becomes active, a reset is generated, which drives the $\overline{\text{RESET}}$ pin (sink open drain output) low to reset the internal hardware and the external circuits. The reset output time is $12 / f_c$ to $16 / f_c$ (1.5 to $2.0 \mu\text{s}$ at $f_c = 8 \text{ MHz}$)

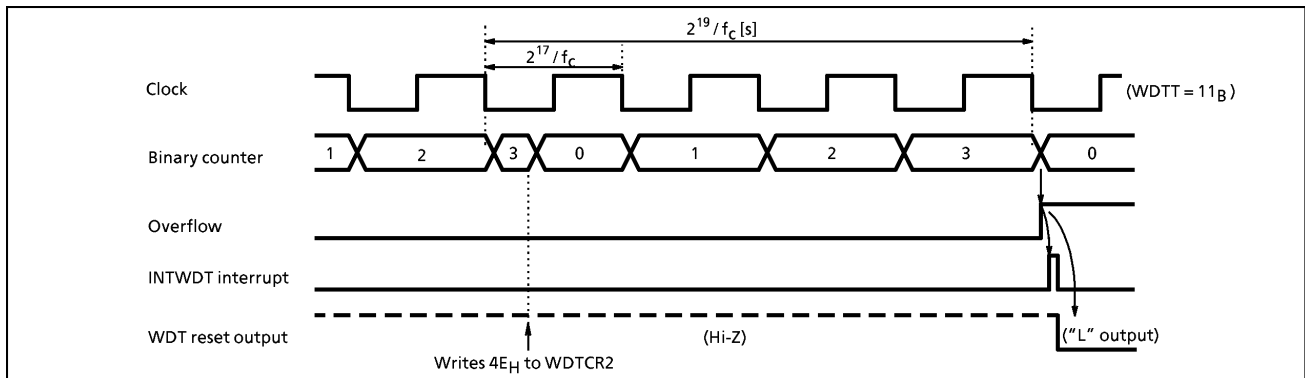


Figure 1-27. Watchdog Timer Interrupt / Reset

1.11 Reset Circuit

The TLCS-870 Series has four types of reset generation procedures : an external reset input, an address-trap-reset, a watchdog timer reset and a system-clock-reset. Table 1-4 shows on-chip hardware initialization by reset action. The internal source reset circuit (watchdog timer reset, address trap reset, and system clock reset) is not initialized when power is turned on. Thus, output from the $\overline{\text{RESET}}$ pin may go low $16 / f_c$ [s] max ($2 \mu\text{s}$ at 8 MHz) when power is turned on.

Table 1-4. Initializing Internal Status by Reset Action

On-Chip Hardware	Initial value	On-Chip Hardware	Initial value
Program counter (PC)	(FFFF _H)-(FFFE _H)	Divider of Timing generator	0
Register bank selector (RBS)	0	Watchdog timer	Enable
Jump status flag (JF)	1	Output latches of I/O ports	Refer to I/O port circuitry
Interrupt master enable flag (IMF)	0		
Interrupt individual enable flags (EF)	0	Control registers	Refer to each of control register
Interrupt latches (IL)	0		

1.11.1 External Reset Input

When the $\overline{\text{RESET}}$ pin is held at low for at least 3 machine cycles ($12 / f_c$ [s]) with the power supply voltage within the operating voltage range and oscillation stable, a reset is applied and the internal state is initialized.

When the $\overline{\text{RESET}}$ pin input goes high, the reset operation is released and the program execution starts at the vector address stored at addresses FFFE_H to FFFF_H. The $\overline{\text{RESET}}$ pin contains a Schmitt trigger (hysteresis) with an internal pull-up resistor. A simple power-on-reset can be applied by connecting an external capacitor and a diode.

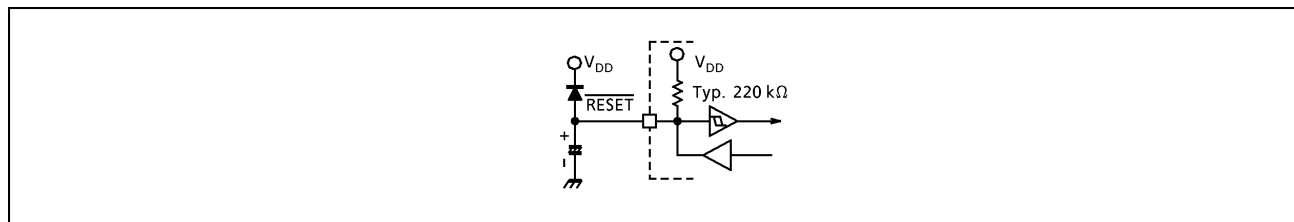


Figure 1-28. Simple Power-On-Reset Circuitry

1.11.2 Address-Trap-Reset

If a CPU malfunction occurs and an attempt is made to fetch an instruction from the RAM or the SFR area (addresses 0000_H to 043F_H of A8700CH / CK / CM and addresses 0000_H to 083F_H of A8700CP / CS), and address-trap-reset will be generated. Then, the $\overline{\text{RESET}}$ pin output will go low. However, when the ROM corrective function is enabled, the address-trap-reset is automatically disabled on the RAM area from 0240_H when the patched program is running. The reset time is $12 / f_c$ to $16 / f_c$ [s] (1.5 to $2.0 \mu\text{s}$ at $f_c = 8$ MHz).

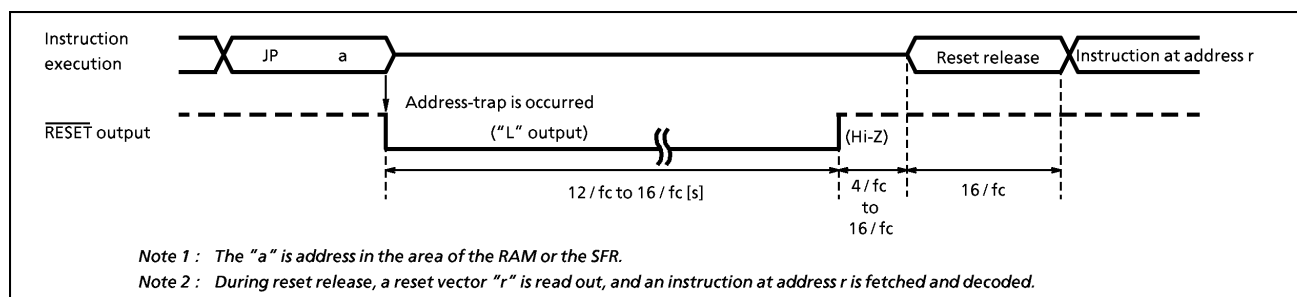


Figure 1-29. Address-Trap-Reset

1.11.3 Watchdog Timer Reset

Refer to Section "1.10 Watchdog Timer".

1.11.4 System-Clock-Reset

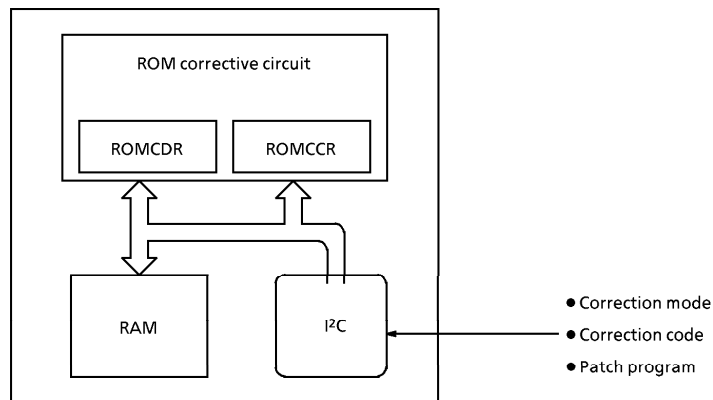
Clearing bit 7 in SYSCR2 to "0" stops high-frequency oscillation, and causes the MCU to deadlock. This can be prevented by automatically generating a reset signal whenever (bit7 in SYSCR2) = 0 is detected to continue the oscillation. Then, the $\overline{\text{RESET}}$ pin output goes low from high-impedance. The reset time is $12 / f_c$ to $16 / f_c$ [s] (1.5 to $2.0 \mu\text{s}$ at $f_c = 8$ MHz).

1.12 ROM Corrective Function

The ROM corrective function can patch the part (s) of on-chip ROM with some bugs. The patched data should be loaded from an external memory at the initialized routine beforehand. The Figure below shows one example of configurations loading the patched data via I²C-bus. The ROM corrective function have two modes. One is to replace the instruction on a certain address in the ROM with the jump instruction to branch into the RAM area where the patched codes and / or data are loaded (Program Jump Mode). The other is to replace a byte or a word (2 byte) length data in the ROM with the patched data (Data Replacement Mode). When the ROM corrective function is enabled, the address-trap-reset is automatically disabled on the RAM area from 0240_H where the patched program is running.

Note 1: For using this function, it is necessary to put the processing routine in the initialized routine beforehand on the premise that the ROM corrective function will be used.
Note 2: BMA8700CSN0A does not support the ROM corrective function.

Example :



1.12.1 Configuration

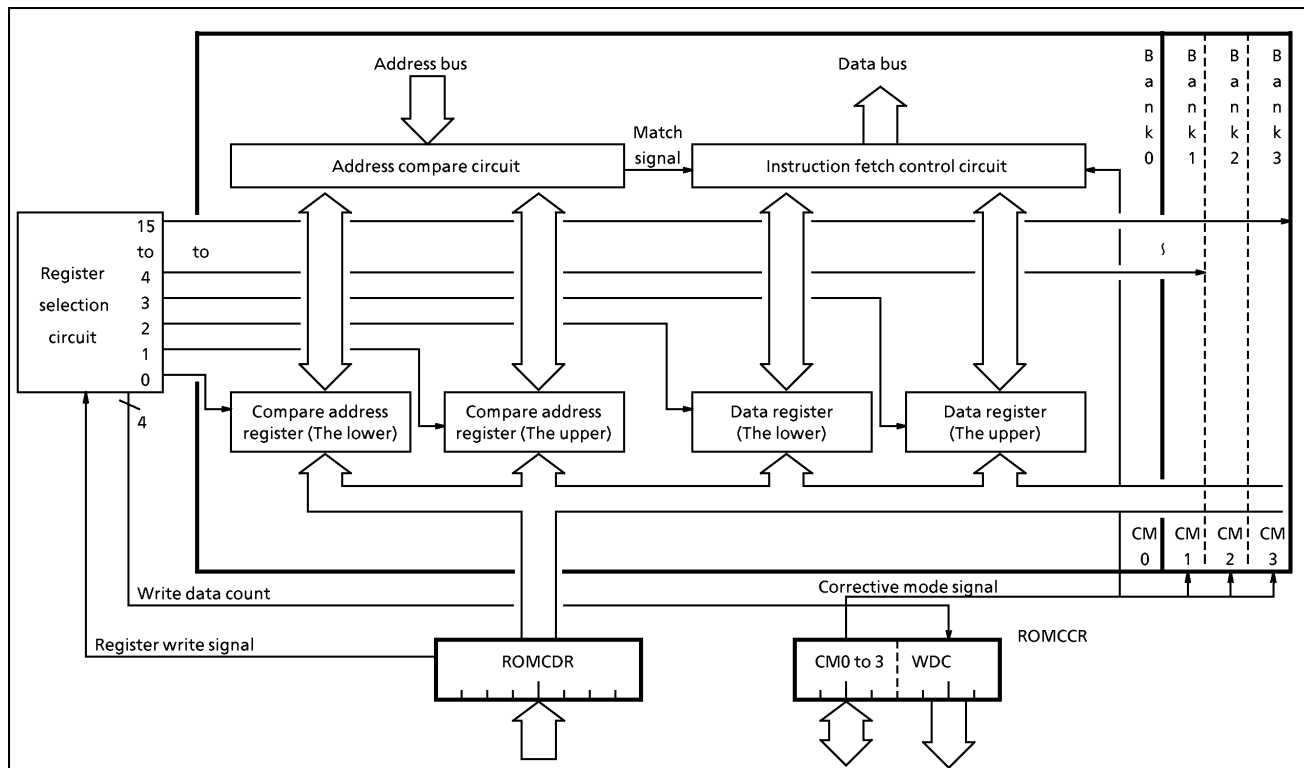


Figure 1-30. ROM Corrective Circuit

1.12.2 Control

The ROM corrective function is controlled by ROM corrective control register (ROMCCR) and ROM corrective data register (ROMCDR).

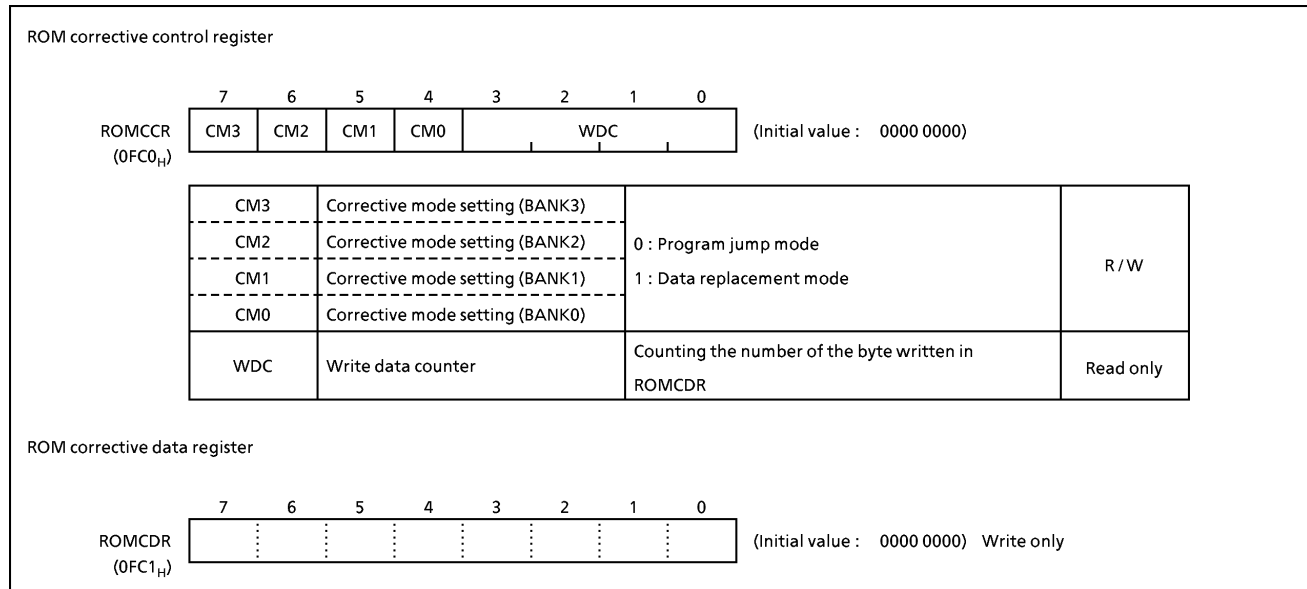


Figure 1-31. ROM Corrective Control Register and ROM Corrective Data Register

(1) The ROM corrective data register writing

The ROM corrective data register has four banks corresponding to four independent locations to patch. The write data counter (WDC) points each bank to set. (Figure 1-32)

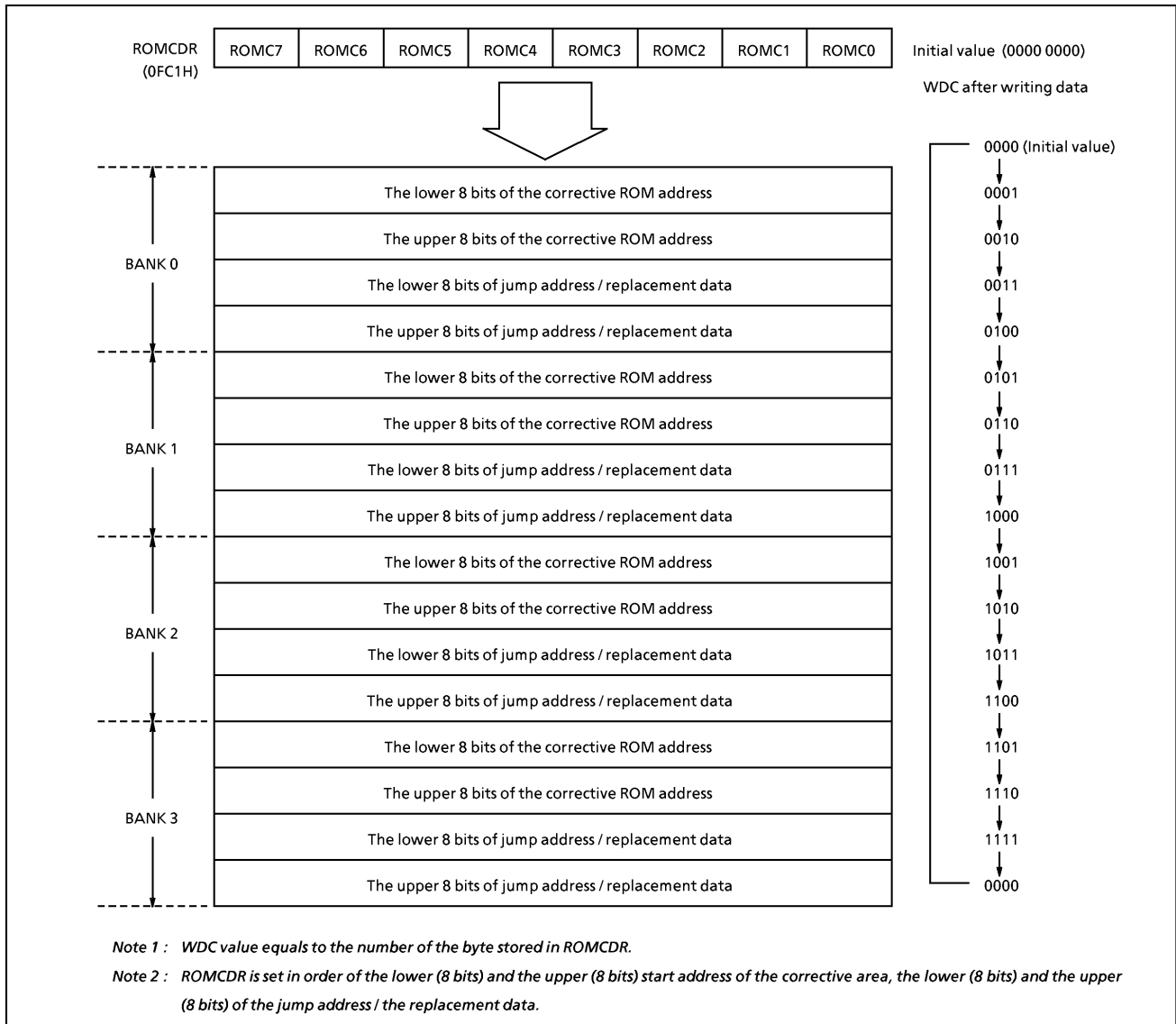


Figure 1-32. Banks and WDC Value of The Program Corrective Data Register

Whenever ROMCDR is written, WDC is incremented to indicate what data is written via ROMCDR. During reset, WDC is initialized to "0".

- ① The lower start address of the corrective area (8 bits)
- ② The upper start address of the corrective area (8 bits)
- ③ The lower jump address / replacement data (8 bits)
- ④ The upper jump address / replacement data (8 bits)
- ⑤ When the bank 1, bank 2, and bank are used, repeat writing ① to ④.

Note: Set the corrective ROM address at least 5 addresses apart from the set address of each bank.

1.12.3 Functions

The ROM corrective function can correct maximum four ROM areas with their corresponding four banks of ROM corrective registers. Either program jump mode or data replacement mode is selected for each bank by CM0 to CM3 respectively.

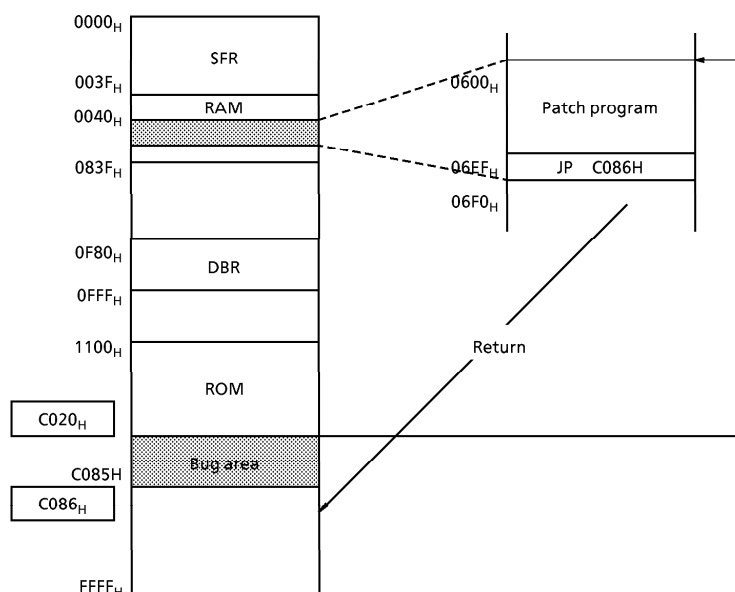
(1) Program jump mode

The program jump mode is to execute the program in the RAM area to correct the bug (s) in the ROM. The start address of ROM that should be patched and the jump vector pointing the RAM area are specified by ROMCDR. When the program is about to run on the code at this start address, the jump instruction is issued, the program branches into the RAM at the jump vector, and the subsequent program codes primarily loaded into this RAM area are executed. After this patch program execution, the program must be returned to the ROM area by any of the jump instructions at the end of this RAM area. By doing these, the correction of the bug is completed. The program jump mode can be selected at CMn = 0 (n = 0 to 3 for each bank).

Note: The start address must point the 1st byte of the instruction codes (Op-Code).

Example : There is bugs on the locations from C020_H to C085_H

The corrective address, the jump vector, the program patch codes and other information to patch the ROM with the bugs must be read out from any of memory storage that holds them during initial program routine. CMn = 0 specifies the program jump mode. Subsequently, the patch program codes are loaded into RAM (0600_H to 06EF_H). The start address (C020_H) of the ROM necessary to patch is written to the corrective ROM address registers, and the start address (0600_H) of the RAM area to patch is loaded onto the jump address registers. When the instruction at C020_H is fetched, the instruction to jump into 0600_H is unconditionally executed instead of the instruction at C020_H, and the subsequent patch program codes are executed. The jump instruction at the end of the patch program codes returns to the ROM at C086_H.



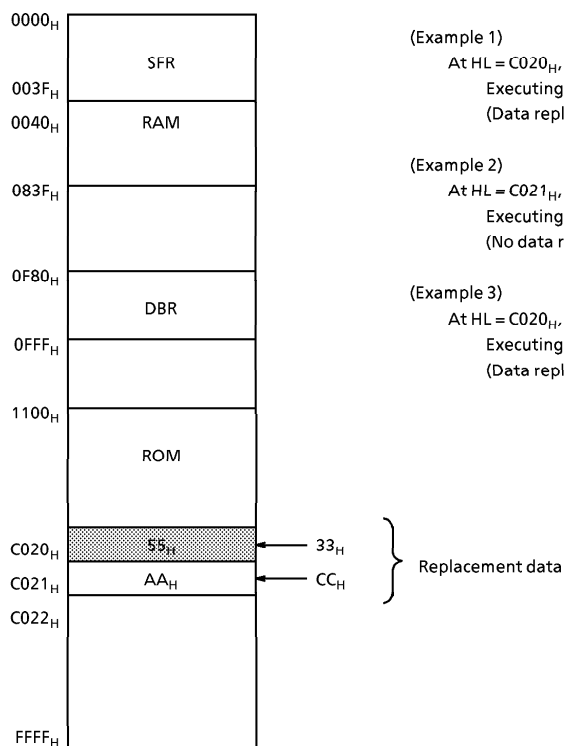
(2) Data replacement mode

The data replacement mode is to directly replace a single byte or word (2 byte) length data with the replacement data which are written via ROMCDR.

The program jump mode can work as the equivalent data replacement mode. However, when many instructions refer a certain data in the ROM which must be patched, the program jump mode consumes the same number of banks as that of the instructions referring this (these) data. ROM data replace mode reduces this kind of bank consumption. (Note : The instruction that gains access to an only byte is replaced to an only start byte.) By setting CMn to 1, the data replacement mode is selected. The start address of ROM data is set to the corrective ROM address, and two bytes replacement data is set to the patch data register via ROMCDR.

Note: The corrective address must point the constant data in the data replacement mode. It is impossible to replace opcode and operand in the data replacement mode.

Example : The start address is set to C020_H as the location of the replaced data. Two bytes of the patch data are set 33_H for C020_H, CC_H for C021_H.



- (Example 1)
At HL = C020_H,
Executing LD A, (HL) loads 33_H in A.
(Data replacement)
- (Example 2)
At HL = C021_H,
Executing LD A, (HL) loads AA_H in A.
(No data replacement)
- (Example 3)
At HL = C020_H,
Executing LD WA, (HL) loads CC33_H in WA.
(Data replacement)

Note: The instructions with memory prefix (HL +) or (-HL) can not be used in the data replacement mode.

2. On-Chip Peripherals Functions

2.1 Special Function Registers (SFR) and Data Buffer Registers (DBR)

The TLCS-870 Series uses the memory mapped I / O system and all peripherals control and data transfers are performed through the special function registers (SFR) and data buffer registers (DBR).

The SFR are mapped to addresses 0000_H to 003F_H, and the DBR to addresses 0F80_H to 0FFF_H.

Figure 2-1 shows the list of the A8700CH / CK / CM / CP / CS SFRs and DBRs.

Address	Read	Write	Address	Read	Write
0000 _H		Reserved	0020 _H	SBICR1 (ACK / CHS)	SBICR1 (SBI control 1)
01		Reserved	21		SBIDBR (SBI data buffer)
02		P2 Port	22		I2CAR (I ² C-bus address)
03		P3 Port	23	SBISR (SBI status)	SBICR2 (SBI control 2)
04		P4 Port	24		SBICR3 (SBI control 3)
05		P5 Port	25	PWMSR (PWM status)	PWMCR (PWM control)
06		P6 Port	26		PWMDBR (PWM data buffer)
07		P7 Port	27		
08	—	P3CR1 (P3 I / O control 1)	28		
09	—	P5CR1 (P5 I / O control 1)	29		
0A		Reserved	2A		
0B		Reserved	2B		OSD control registers
0C	—	P4CR (P4 I / O control)	2C		
0D	—	P6CR (P4 I / O control)	2D		
0E		ADCCR (A / D converter control)	2E		
0F		ADCDR (A / D conv. result)	2F		
10	—	TREG1A _L (Timer register 1A)	30		Reserved
11		TREG1A _H	31		Reserved
12	TREG1B _L (Timer register 1B)	—	32		Reserved
13	TREG1B _H	—	33		Reserved
14	—	TC1CR (TC1 control)	34	—	WDTCR1 (WDT control)
15	—	TC2CR (TC2 control)	35	—	WDTCR2
16	—	TREG2 _L (Timer register 2)	36		TBTCR (TBT control)
17	—	TREG2 _H	37		EINTCR (Exter. interrupt control)
18		TREG3A (Timer register 3A)	38		SYSCR1 (System control)
19	TREG3B (Timer register 3B)	—	39		SYSCR2 (System control)
1A	—	TC3CR (TC3 control)	3A		EIR _L (Interrupt enable register)
1B	—	TREG4 (Timer register 4)	3B		EIR _H
1C	—	TC4CR (TC4 control)	3C		IL _L (Interrupt latch)
1D	—		3D		IL _H
1E	OSD	OSD control registers	3E		Reserved
1F	OSD		3F	PSW (Program status word)	RBS (Register bank selector)

Note 1 : Do not access reserved areas by the program.
 Note 2 : — : Cannot be accessed.
 Note 3 : Write-only registers and interrupt latches cannot use the read-modify-write instructions. (Bit manipulation instructions such as SET, CLR, etc. and logical operation instructions such as AND, OR, etc.)
 Note 4 : When defining address 003F_H with assembler symbols, use GPSW and GRBS.
 Note 5 : SBI : Serial bus interface
 PWM : Pulse with modulation
 OSD : On screen display

(a) Special function registers

Figure 2-1-1. SFR & DBR

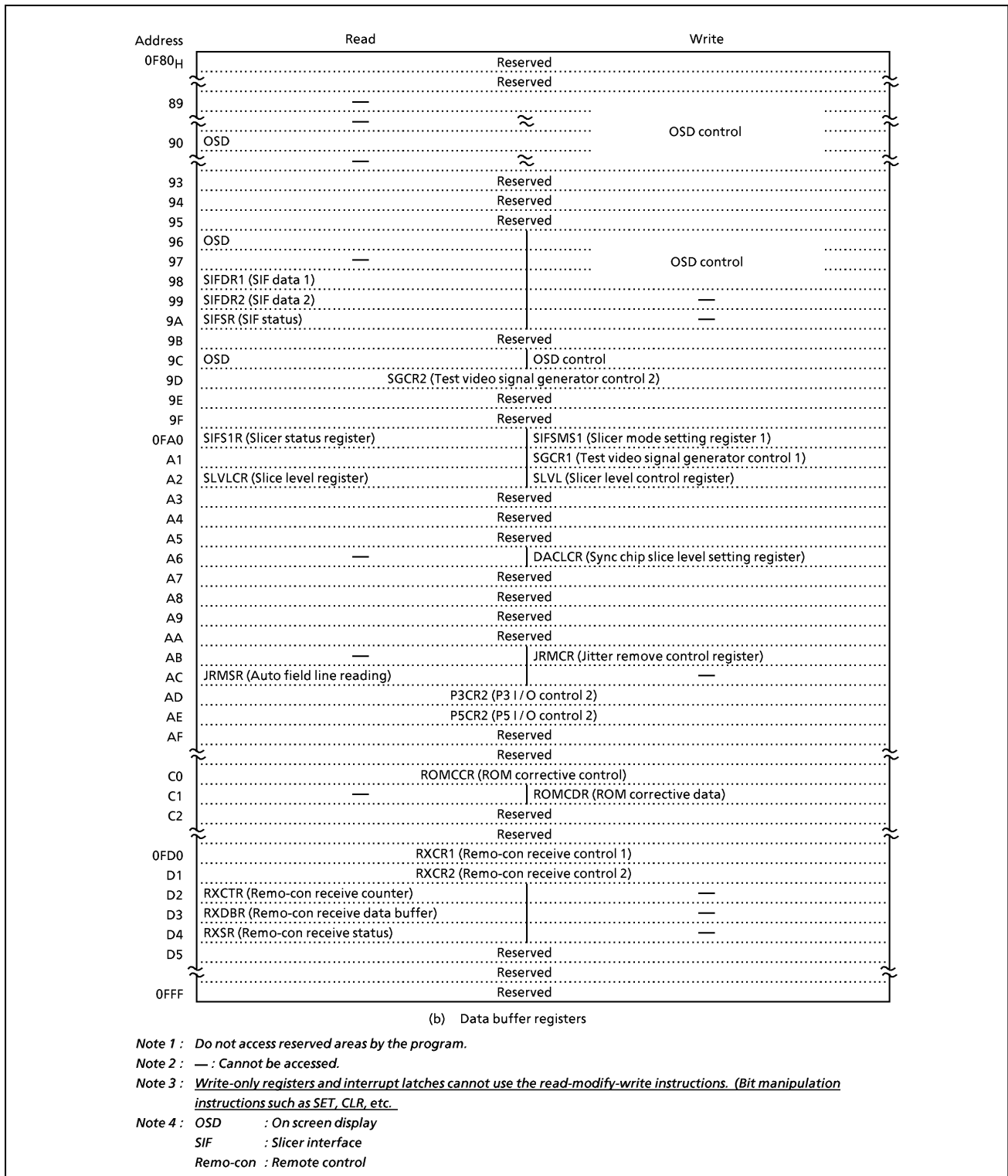


Figure 2-1-2. SFR & DBR

2.2 I/O Ports

The A8700CH / CK / CM / CP / CS has 6 parallel input / output ports (33 pins) as follows :

	Primary Function	Secondary Functions
Port P2	1 bit I/O port	External interrupt input, and STOP mode release signal input
Port P3	6 bit I/O port	External interrupt input, remote control signal input, timer / counter input, serial bus interface input / output and data slicer input
Port P4	8 bit I/O port	Pulse width modulation output
Port P5	8 bit I/O port	Pulse width modulation output external interrupt input, timer / counter input, serial ports, serial bus interface input / output, and analog input
Port P6	8 bit I/O port	R, G, B and Y / BL output from OSD circuitry, R.G.B and Y / BL input, analog input, and test video signal output
Port P7	2 bit I/O port	Horizontal synchronous pulse input and vertical synchronous pulse input to OSD circuitry

Each output port contains a latch, which holds the output data. All input ports do not have latches, so the external input data should either be held externally until read or reading should be performed several times before processing. Figure 2-2 shows input / output timing examples.

External data is read from an I/O port in the S1 state of the read cycle during execution of the read instruction. This timing can not be recognized from outside, so that transient input such as chattering must be processed by the program. Output data changes in the S2 state of the write cycle during execution of the instruction which writes to an I/O port.

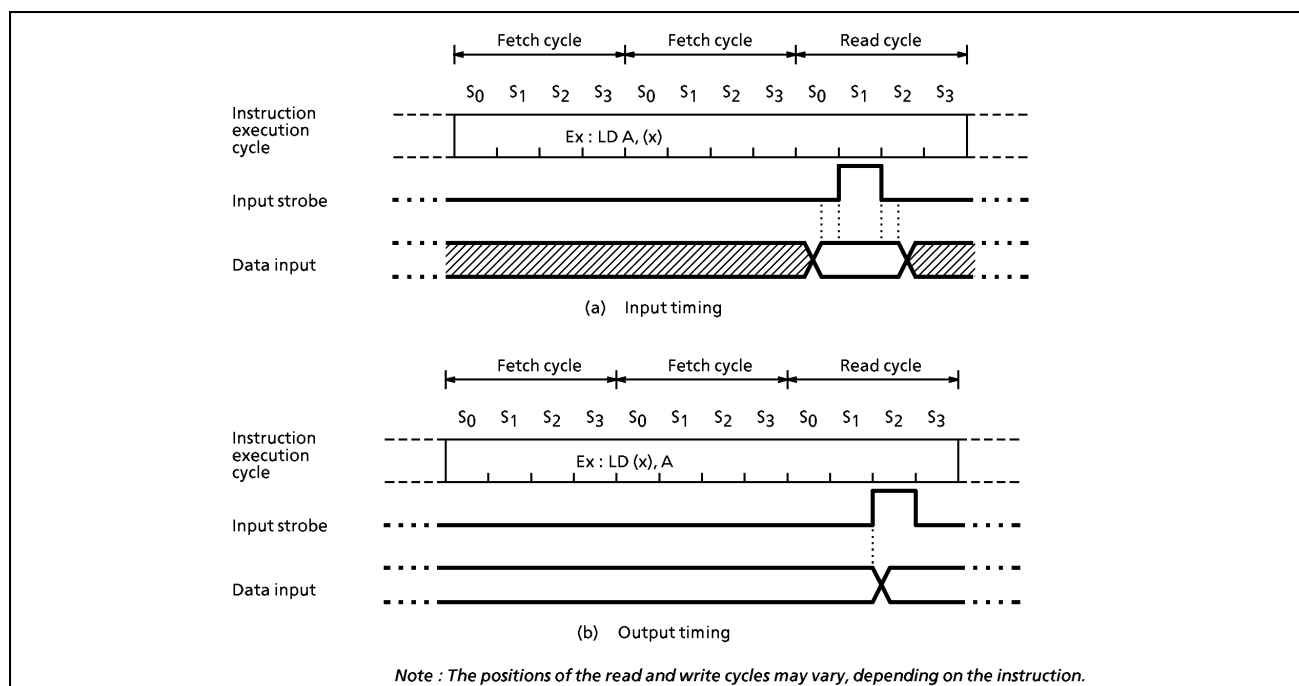


Figure 2-2. Input / Output Timing (Example)

When reading an I / O port except programmable I / O ports, whether the pin input data or the output latch contents are read depends on the instructions, as shown below :

(1) Instructions that read the output latch contents

- ① XCH r, (src)
- ② SET / CLR / CPL (src).b
- ③ SET / CLR / CPL (pp).g
- ④ LD (src).b, CF
- ⑤ LD (pp).b, CF
- ⑥ ADD / ADDC / SUB / SUBB / AND / OR / XOR (src), n
- ⑦ (src) side of ADD / ADDC / SUB / SUBB / AND / OR / XOR (src), (HL)

(2) Instructions that read the pin input data

- ① Instructions other than the above (1)
- ② (HL) side of ADD / ADDC / SUB / SUBB / AND / OR / XOR (src), (HL)

2.2.1 Port P2 (P20)

Port P2 is a 1bit input / output port. It is also used as an external interrupt input, and a STOP mode release signal input. When used as an input port, or a secondary function pin, the output latch should be set to "1". During reset, the output latch is initialized to "1".

It is recommended that pin P20 should be used as an external interrupt input, a STOP mode release signal input, or an input port. If used as an output port, the interrupt latch is set on the falling edge of the P20 output pulse.

When a read instruction for port P2 is executed, bits 7 to 1 in P2 are read in as undefined data.

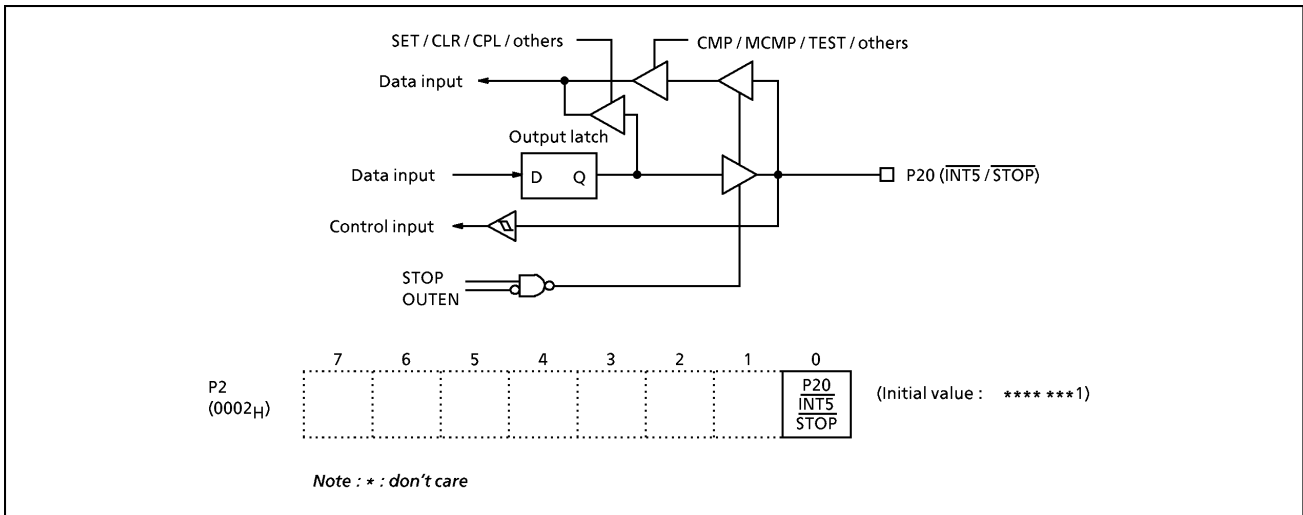


Figure 2-3. Port P2

2.2.2 Port P3 (P35 to P30)

Port P3 is an 6bit input / output port which can be configured as an input or an output in one-bit unit under software control. Input / output mode is specified by the corresponding bit in the port P3 input / output control register 1 (P3CR1). Port P3 is configured as an input if its corresponding P3CR1 bit is cleared to "0", and as an output if its corresponding P3CR1 bit is set to "1". During reset, P3CR1 is initialized to "0", which configures port P3 as an input. The P3 output latches are also initialized to "1". Data is written into the output latch regardless of the P3CR1 contents. Therefore initial output data should be written into the output latch before setting P3CR1.

Port P3 is also used as an external interrupt input, Remote-control signal input a timer / counter input, data slicer input and serial bus interface input / output. When used as a secondary function input pin except I²C bus interface input / output, the input pins should be set to the input mode. When used as a secondary function output pin except I²C bus interface input / output, the output pins should be set to the output mode and beforehand the output latch should be set to "1". When P34 and P35 are used as I²C bus interface input / output, P3CR2 bits should be set to the sink open drain mode, the output latches should be set to "1", and the output pins should be set to the output mode.

Note: Input mode port is read the state of input pin. When input / output mode is used mixed, the contents of output latch setting input mode may be changed by executing bit manipulation instructions.

Example 1 : Outputs an immediate data 5A_H to port P3.

```
LD (P3), 5AH ; P3 ← 5AH
```

Example 2 : Inverts the output of the lower 4 bits (P33 to P30) in port P3.

```
XOR (P3), 00001111B ; P33 to P30 ←  $\overline{P33}$  to  $\overline{P30}$ 
```

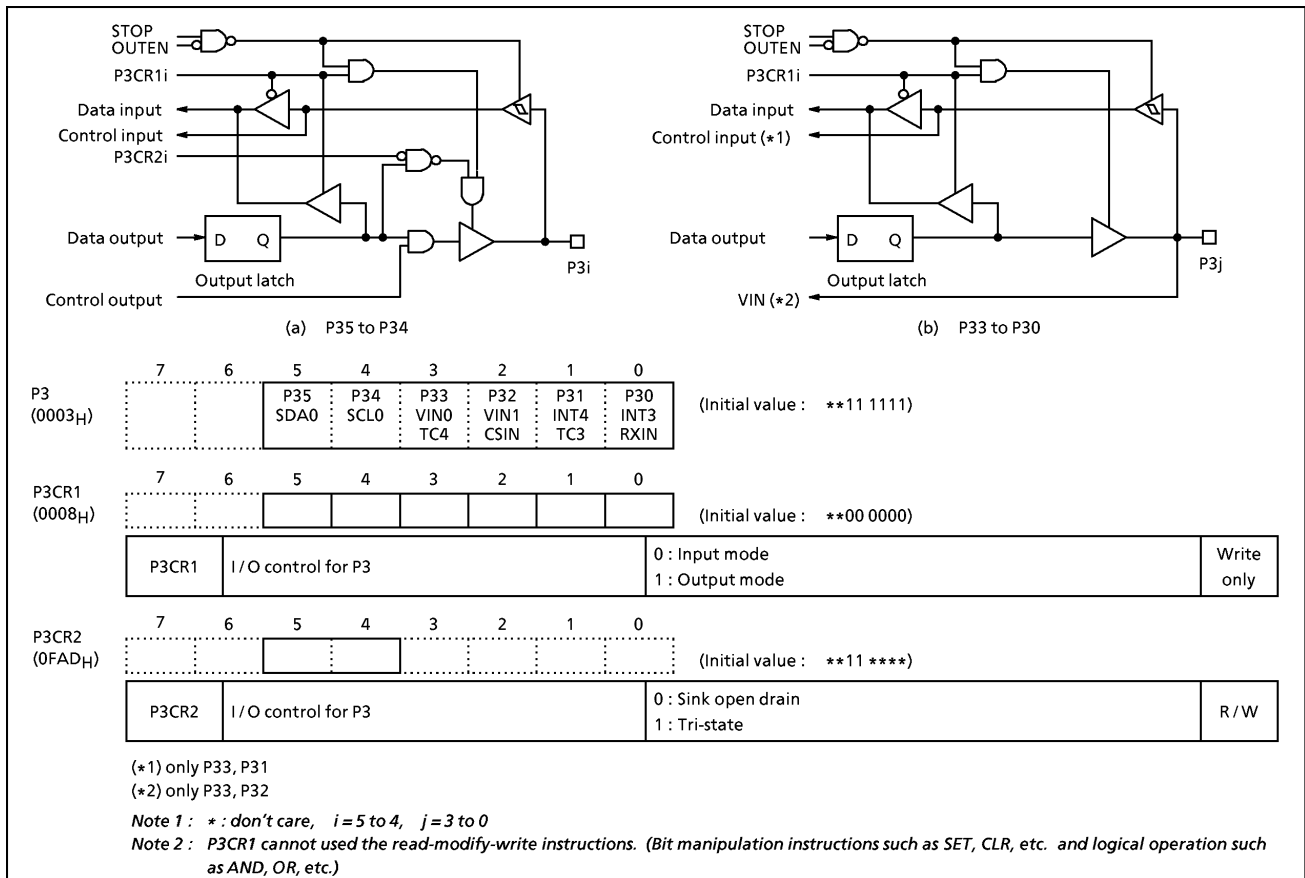


Figure 2-4. Port P3 and P3CR

2.2.3 Port P4 (P47 to P40)

Port P4 is an 8 bit input / output port which can be configured as an input or an output in one-bit unit under software control. Input / output mode is specified by the corresponding bit in the port P4 input / output control register (P4CR). Port P4 is configured as an input if its corresponding P4CR bit is cleared to "0", and as an output if its corresponding P4CR bit is set to "1". During reset, P4CR is initialized to "0", which configures port P4 as an input. The P4 output latches are also initialized to "1". Data is written into the output latch regardless of the P4CR contents. Therefore initial output data should be written into the output latch before setting P4CR.

Port P4 is also used as a pulse width modulation (PWM) output. When used as a PWM output pin, the output pins should be set to the output mode and beforehand the output latch should be set to "1".

Note: Input mode port is read the state of input pin. When input / output mode is used mixed, the contents of output latch setting input mode may be changed by executing bit manipulation instructions.

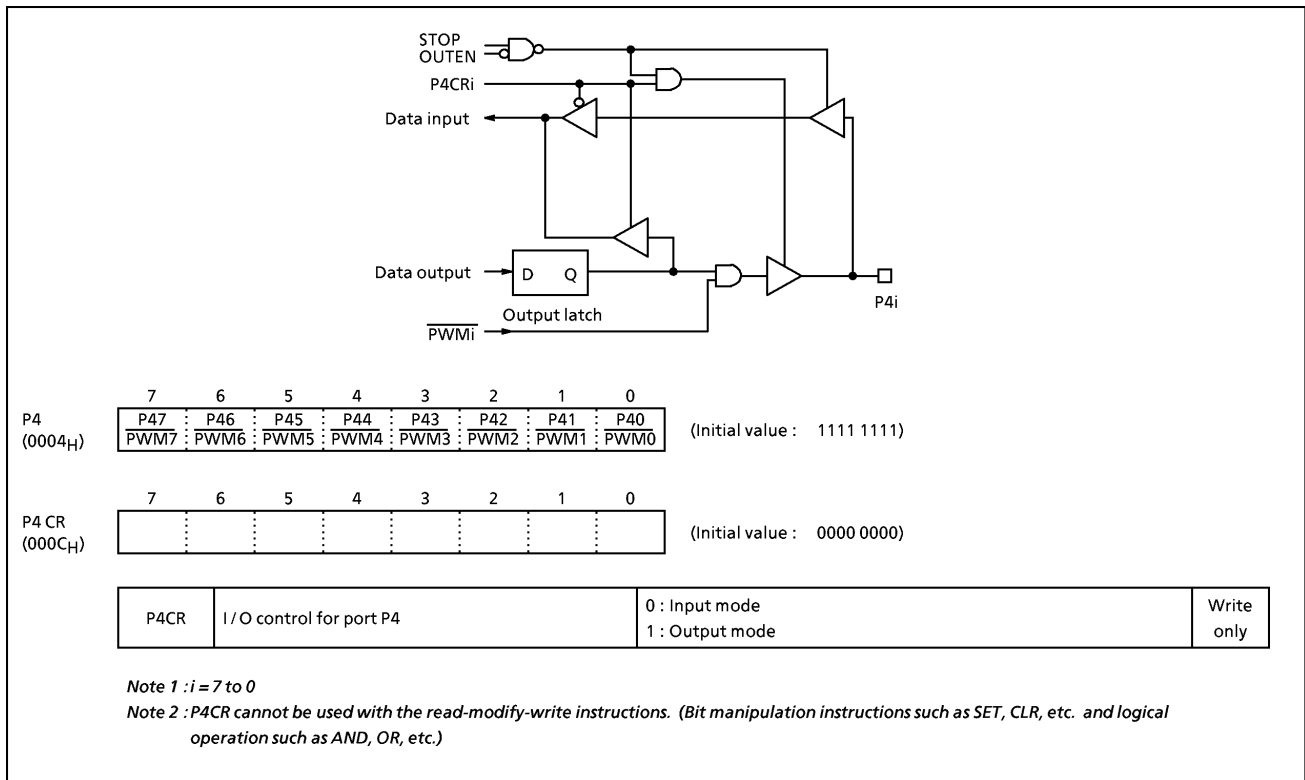


Figure 2-5. Ports P4 and P4CR

2.2.4 Port P5 (P57 to P50)

Port P5 is an 8 bit input / output port which can be configured as an input or an output in one-bit unit under software control. Input / output mode is specified by the corresponding bit in the port P5 input / output control register 1 (P5CR1). Port P5 is configured as an input if its corresponding P5CR1 bit is cleared to "0", and as an output if its corresponding P5CR1 bit is set to "1". During reset, P5CR1 is initialized to "0", which configures port P5 as an input. The P5 output latches are also initialized to "1". Data is written into the output latch regardless of the P5CR1 contents. Therefore initial output data should be written into the output latch before setting P5CR1.

Port P5 is also used as is also used as A / D converter analog input, a pulse width modulation (PWM) output external interrupt input, timer / counter input, and serial bus interface input / output. When used as a secondary function input pin except I²C bus interface input / output, the input pins should be set to the input mode. When used as a secondary function output pin except I²C bus interface input / output, the output pins should be set to the output mode and beforehand the output latch should be set to "1". When P52 and P51 are used as I²C bus interface input / output, P5CR2 bits should be set to the sink open drain mode, the output latches should be set to "1", and the output pins should be set to the output mode.

Note: Input mode port is read the state of input pin. When input / output mode is used mixed, the contents of output latch setting input mode may be changed by executing bit manipulation instructions.

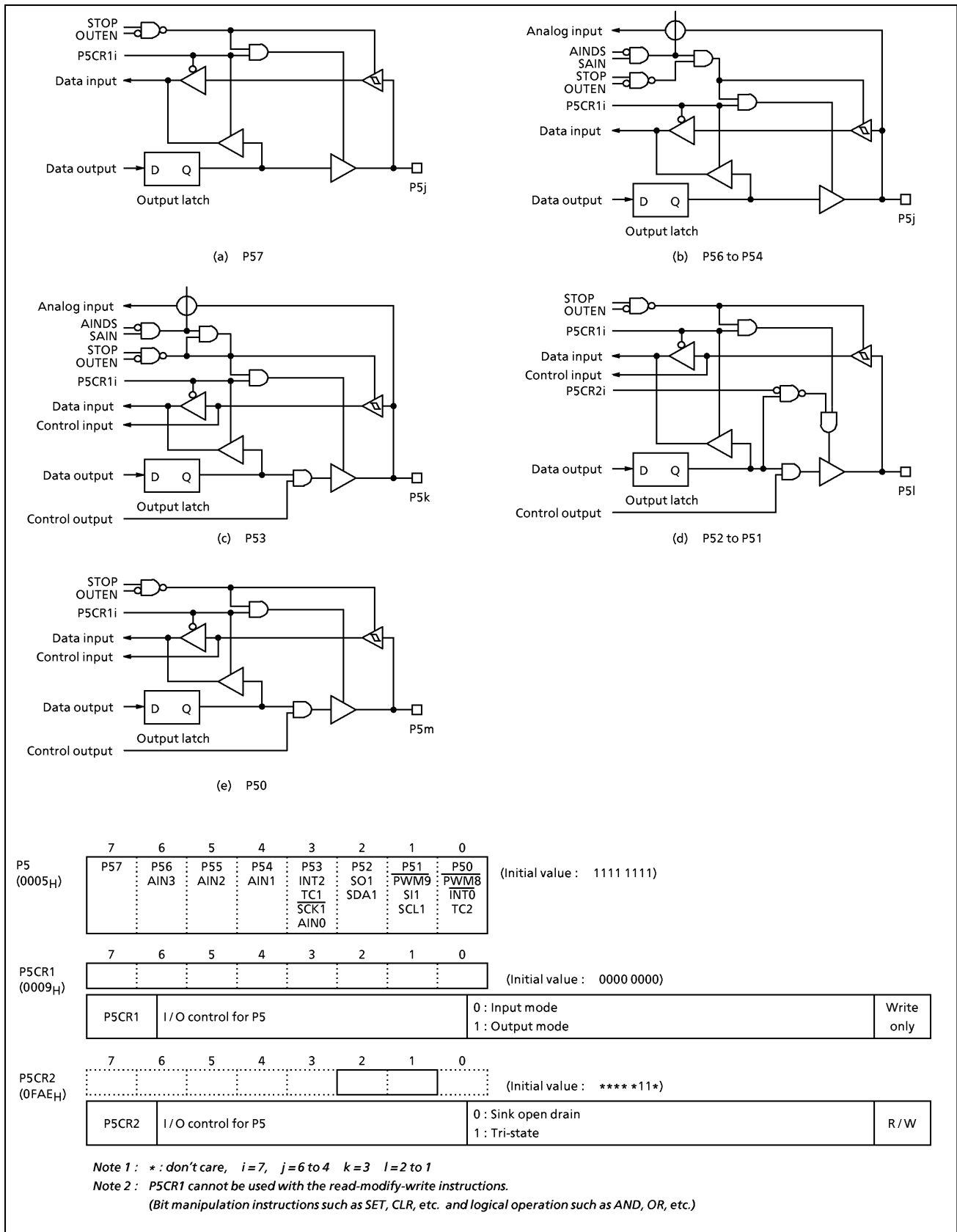


Figure 2-6. Ports P5

2.2.5 Port P6 (P67 to P60)

Port P6 is an 8 bit input / output port which can be configured as an input or an output in one-bit unit under software control. Input / output mode is selected by the corresponding bit in the port P6 input / output control register (P6CR). Port P6 is configured as an input if its corresponding P6CR bit is cleared to "0", and as an output if its corresponding P6CR bit is set to "1" and P6nS bit is set to "1". During reset, P6CR is initialized to "0", which configures port P6 as an input. The P6 output latches are also initialized to "1".

Data is written into the output latch regardless of the P6CR contents. Therefore initial output data should be written into the output latch before setting P6CR.

Port P6 is used as an on screen display (OSD) output (R, G, B, and Y / BL signal) / input (RIN, GIN, BIN, Y / BLIN signal), a test video signal output and A / D converter analog input. When used as a secondary function input, the input pins should be set to the input mode. When used as an OSD output pin, the output pins should be set to the output mode and beforehand the port P6 data selection register (P67S to P64S) should be clear to "0".

Note: Input mode port is read the state of input pin. When input / output mode is used mixed, the contents of output latch setting input mode may be changed by executing bit manipulation instructions.

Example : Sets the lower 4 bits (P63 to P60) in port P6 to the output mode, and the other bit to the input mode.

LD (P6CR), 0FH : P6CR ← 00001111_B

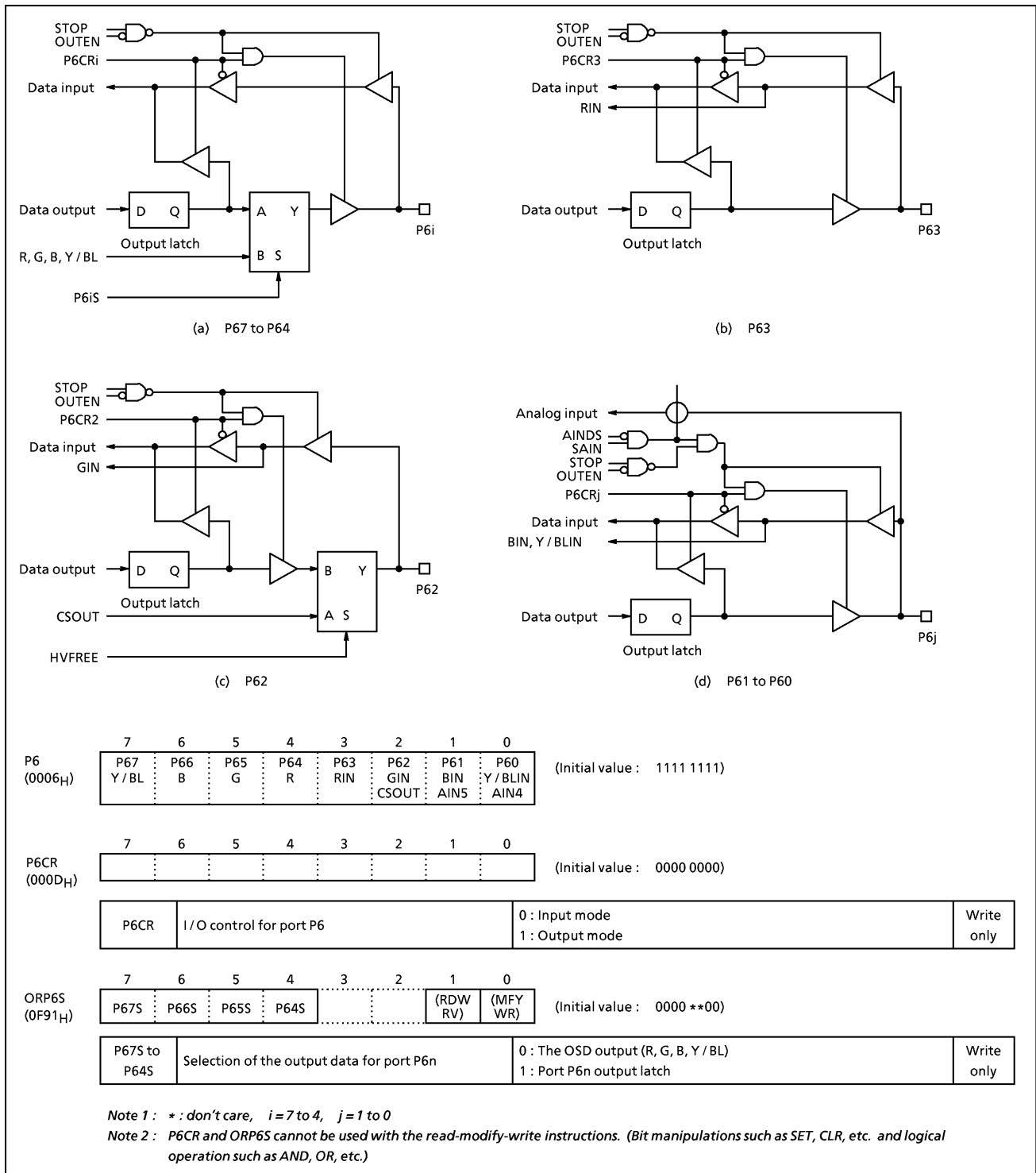


Figure 2-7. Ports P6, P6CR, and P67S to P64S

2.2.6 Port P7 (P73 to P70)

Port P7 is a 2bit input / output port, and is also used as a vertical synchronous signal (\overline{VD}) input and a horizontal synchronous signal (\overline{HD}) input for the on screen display (OSD) circuitry.

The output latches, are initialized to "1" during reset. When used as an input port or a secondary function pin, the output latch should be set to "1".

When a read instruction for port P7 is executed, bits 7 to 2 in P7 are read in as undefined data.

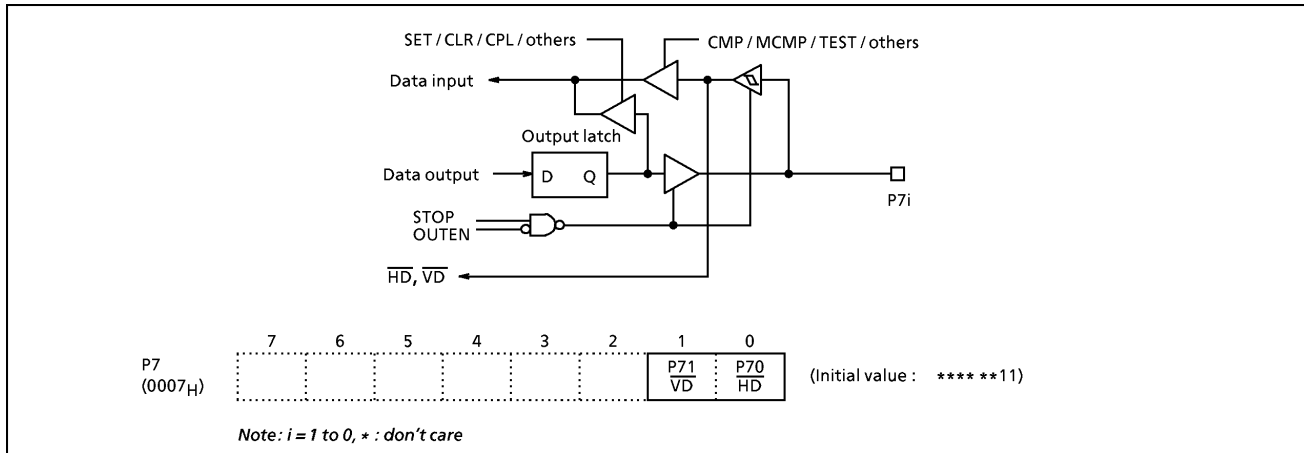


Figure 2-8. Ports P7

2.3 Time Base Timer (TBT)

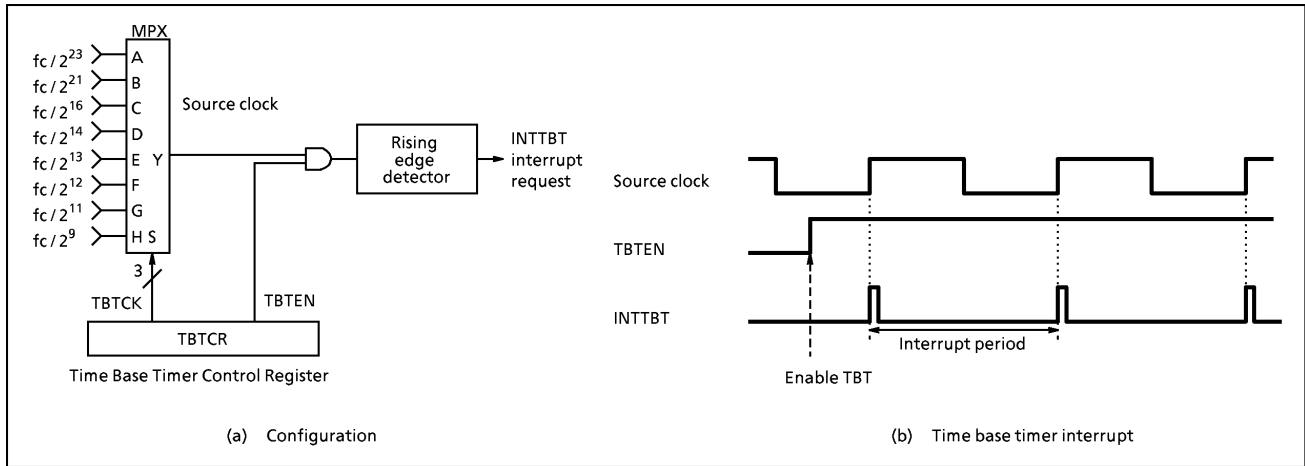


Figure 2-9. Time Base Timer

The time base timer generates time base for key scanning, dynamic displaying, etc. It also provides a time base timer interrupt (INTTBT). The time base timer is controlled by a control register (TBTCR) shown in Figure 2-10.

An INTTBT is generated on the first rising edge of source clock (the divider output of the timing generator) after the time base timer has been enabled. The divider is not cleared by the program ; therefore, only the first interrupt may be generated ahead of the set interrupt period. (Figure 2-9 (b)) The interrupt frequency (TBTCk) must be selected with the time base timer disabled (When the time base timer is changed from enabling to disabling, the interrupt frequency can't be changed.) both frequency selection and enabling can be performed simultaneously.

Example : Sets the time base timer frequency to $fc / 2^{16}$ [Hz] and enables an INTTBT interrupt.

```
LD (TBTCR), 00001010B
SET (EIRL). 6
```

TBTCR (0036H)	7	6	5	4	3	2	1	0	(Initial value 0**0 0**)
	"0"	"0"	"0"	"0"	TBTEN	TBTCk			
TBTEN	Time base timer enable / disable				0 : Disable 1 : Enable				R / W
TBTCk	Time base timer interrupt frequency select				000 : $fc / 2^{23}$ [Hz] (0.95 Hz, at $fc = 8$ MHz) 001 : $fc / 2^{21}$ (3.81 Hz, at $fc = 8$ MHz) 010 : $fc / 2^{16}$ (122.07 Hz, at $fc = 8$ MHz) 011 : $fc / 2^{14}$ (488.28 Hz, at $fc = 8$ MHz) 100 : $fc / 2^{13}$ (976.56 Hz, at $fc = 8$ MHz) 101 : $fc / 2^{12}$ (1953.12 Hz, at $fc = 8$ MHz) 110 : $fc / 2^{11}$ (3906.25 Hz, at $fc = 8$ MHz) 111 : $fc / 2^9$ (15625 Hz, at $fc = 8$ MHz)				
Note 1 : fc : High-frequency clock [Hz], * : don't care Note 2 : Always write "0" to bit 7 to 4 of TBTCR									

Figure 2-10. Time Base Timer Control Register

2.4 16 bit Timer 1 (TC1)

2.4.1 Configuration

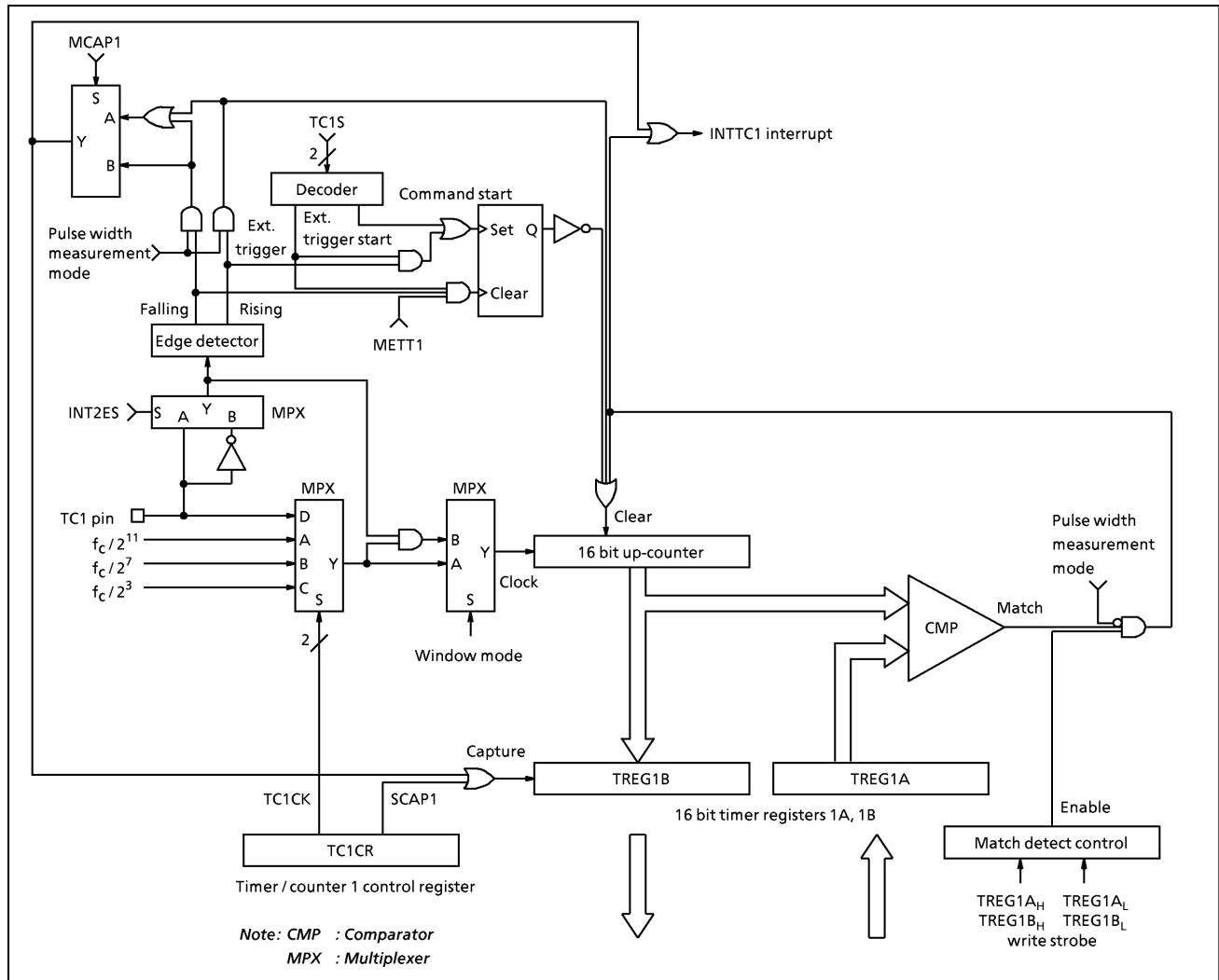


Figure 2-11. Timer / Counter 1 (TC1)

2.4.2 Control

The timer / counter 1 is controlled by a timer / counter 1 control register (TC1CR) and two 16 bit timer registers (TREG1A and TREG1B). Reset does not affect the TREG1A and TREG1B.

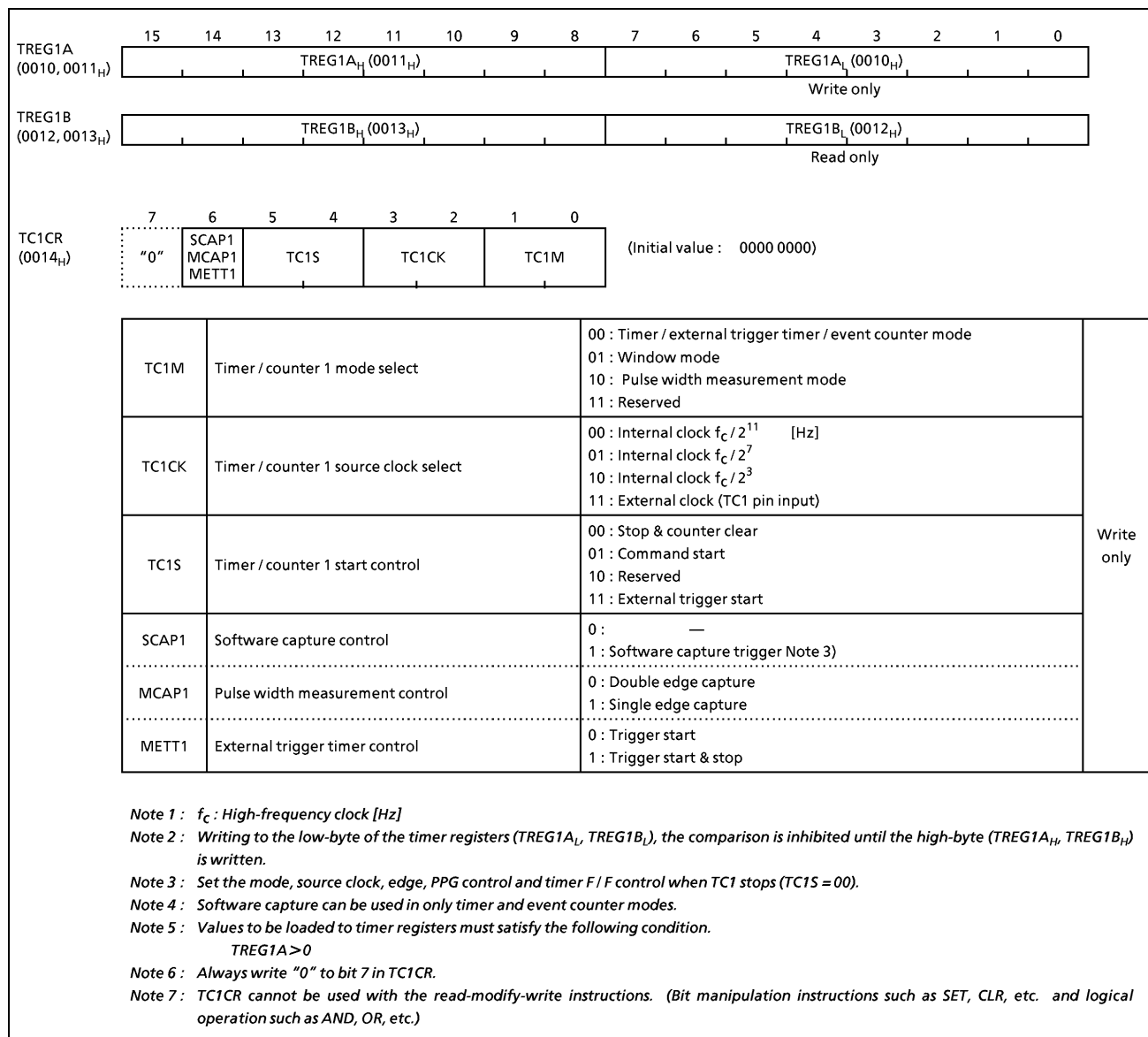


Figure 2-12. Timer Registers and TC1 Control Register

2.4.3 Function

Timer / counter 1 has five operating modes : timer, external trigger timer, event counter, window, pulse width measurement mode.

(1) Timer mode

In this mode, counting up is performed using the internal clock. The contents of the timer register 1A (TREG1A) are compared with the contents of the up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared to "0". Counting up resumes after the counter is cleared. The current contents of the up-counter can be transferred to the timer register 1B (TREG1B) by setting SCAP1 (bit 6 in TC1CR) to "1" (software capture function). SCAP1 is automatically cleared to "0" after capturing.

Table 2-1. Timer 1 Source Clock (Internal Clock)

Source Clock	Resolution (AT $f_c = 8 \text{ MHz}$)	Maximum Time Setting (AT $f_c = 8 \text{ MHz}$)
$f_c / 2^{11} [\text{Hz}]$	256 μs	16.77696 s
$f_c / 2^7$	16 μs	1.04856 s
$f_c / 2^3$	1 μs	65.535 ms

Example 1 : Sets the source clock to $f_c / 2^7$ [Hz] and generates an interrupt 1 [s] later (at $f_c = 8 \text{ MHz}$).

```
LD (TC1CR), 00000100B ; Sets the TC1 source clock
LDW (TREG1A), 0F424H ; Sets the timer register (1 s ÷ f_c / 2^7 = F424H)
SET (EIRL). EF4 ; Enables INTTC1 interrupt
EI
LD (TC1CR), 00010100B ; Starts TC1
```

Example 2 : Software capture

```
LD (TC1CR), 01010100B ; SCAP1 ← 1 (Captures)
LD WA, (TREG1B) ; Reads captured value
```

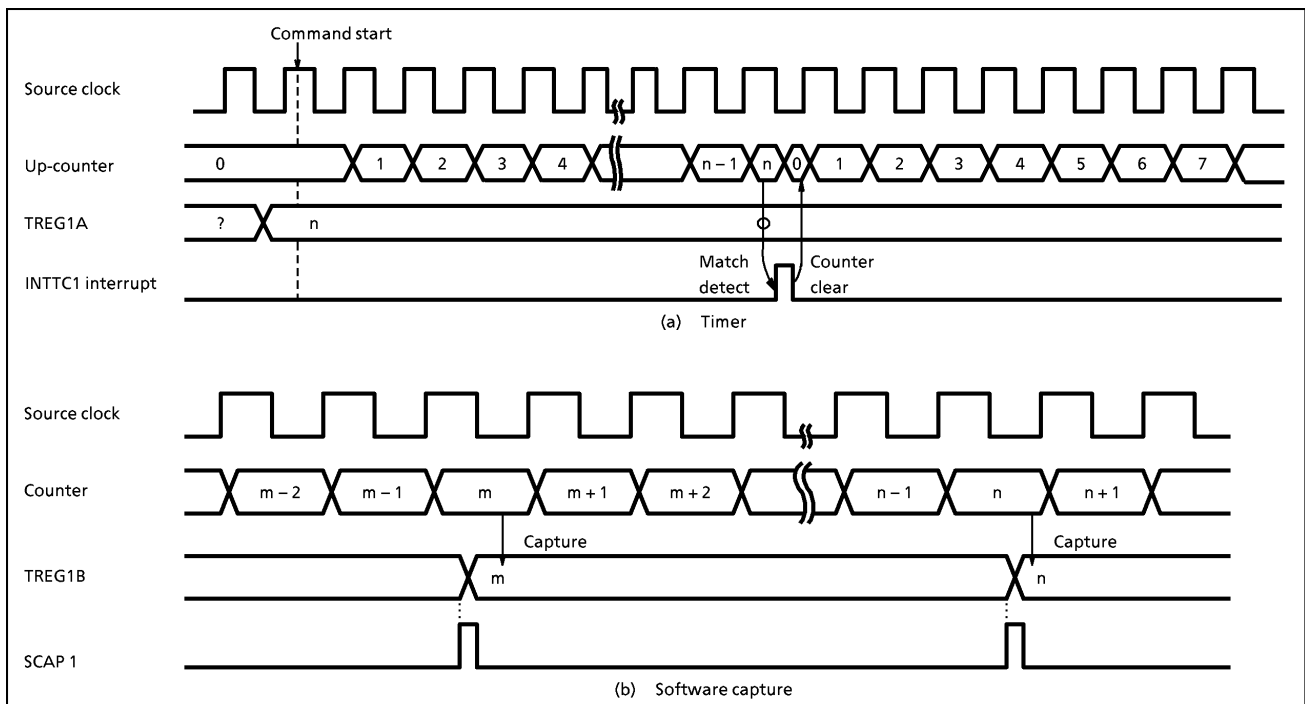


Figure 2-13. Timer Mode Timing Chart

(2) External trigger timer mode

This is the timer mode to start counting up by the external trigger. The trigger is the edge of the TC1 pin input. Either rising or falling edge can be selected with INT2ES. Edge selection is the same as for the external interrupt input INT2 pin. Source clock is used an internal clock selected with the TC1CK. The contents of the TREG1A is compared with the contents of the up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared to "0" and halted. The counter is restarted by the selected edge of the TC1 pin input.

The TC1 pin input has the same noise rejection as the INT2 pin ; therefore, pulses of $7 / f_c$ [s] or less are eliminated as noise in the NORMAL or IDLE mode. A pulse width of $24 / f_c$ [s] or more is required for edge detection.

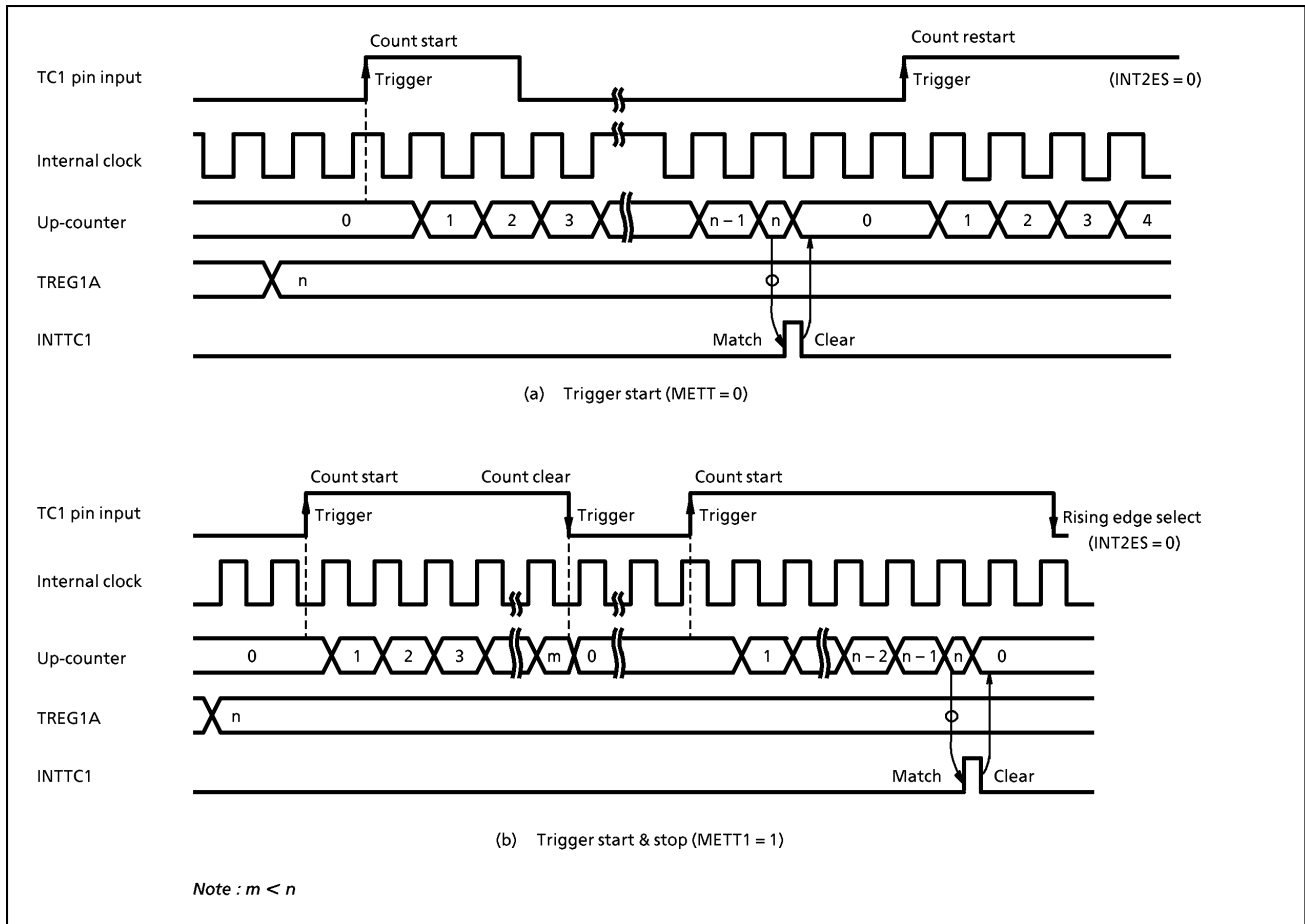


Figure 2-14. External Trigger Timer Mode Timing Chart

(3) Event counter mode

In this mode, events are counted at the edge of the TC1 pin input. Either rising or falling edge can be selected with INT2ES in EINTCR. The contents of the TREG1A are compared with the contents of the up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. The maximum applied frequency is $f_c / 2^4$ [Hz] in the NORMAL or IDLE mode.

Setting SCAP1 to "1" transfers the current contents of the up-counter to the TREG1B (software capture function).

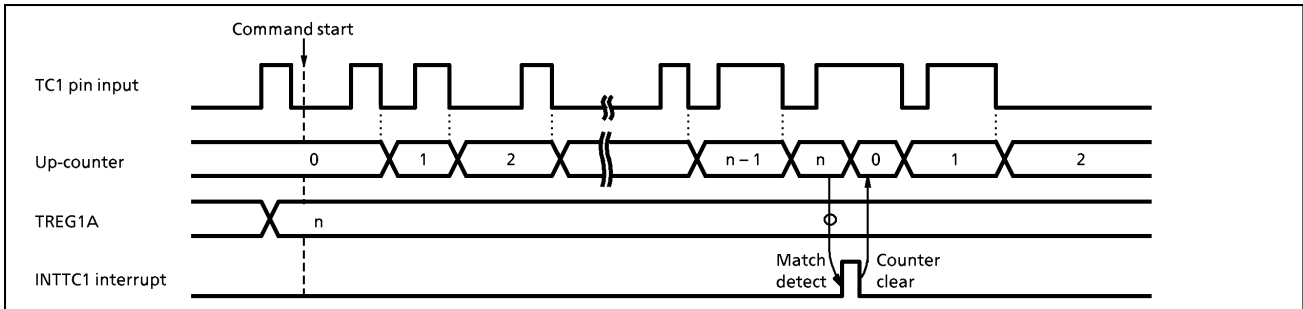


Figure 2-15. Event Counter Mode Timing Chart (INT2ES = 1)

(4) Window mode

Counting up is performed at the rising edge of the pulse that is the logical AND-ed product of the TC1 pin input (window pulse) and an internal clock. The contents of the TREG1A are compared with the contents of the up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. Positive or negative logic for the TC1 pin input can be selected with INT2ES. Setting SCAP1 to "1" transfers the current contents of the up-counter to the TREG1B. It is necessary that the maximum applied frequency (TC1 input) be such that the counter value can be analyzed by the program. That is, the frequency must be considerably slower than the selected internal clock.

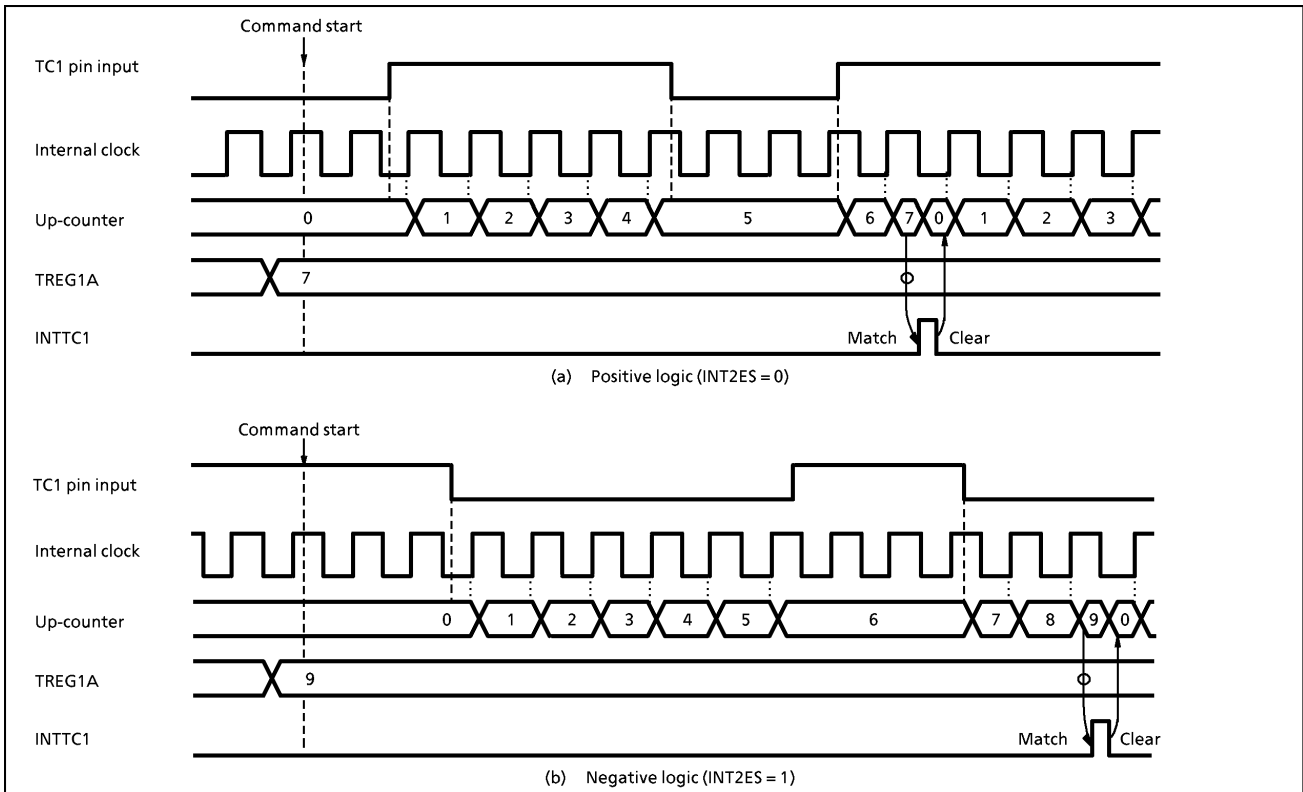


Figure 2-16. Window Mode Timing Chart

(5) Pulse width measurement mode

Counting is started by the external trigger (set to external trigger start by TC1S). The trigger is selected either rising or falling edge of the TC1 pin input. The source clock is used an internal clock. At the next falling (rising) edge, the counter contents are transferred to the TREG1B and an INTTC1 interrupt is generated. The counter is cleared when single edge capture mode is set. When double edge capture is set, the counter continues and, at the next rising (falling) edge, the counter contents are again transferred to the TREG1B. If a falling (rising) edge capture value is required, it is necessary to read out the TREG1B contents until a rising (falling) edge is detected. Falling or rising edge is selected with INT2ES, and single edge or double edge is selected with MCAP1 (bit 6 in TC1CR).

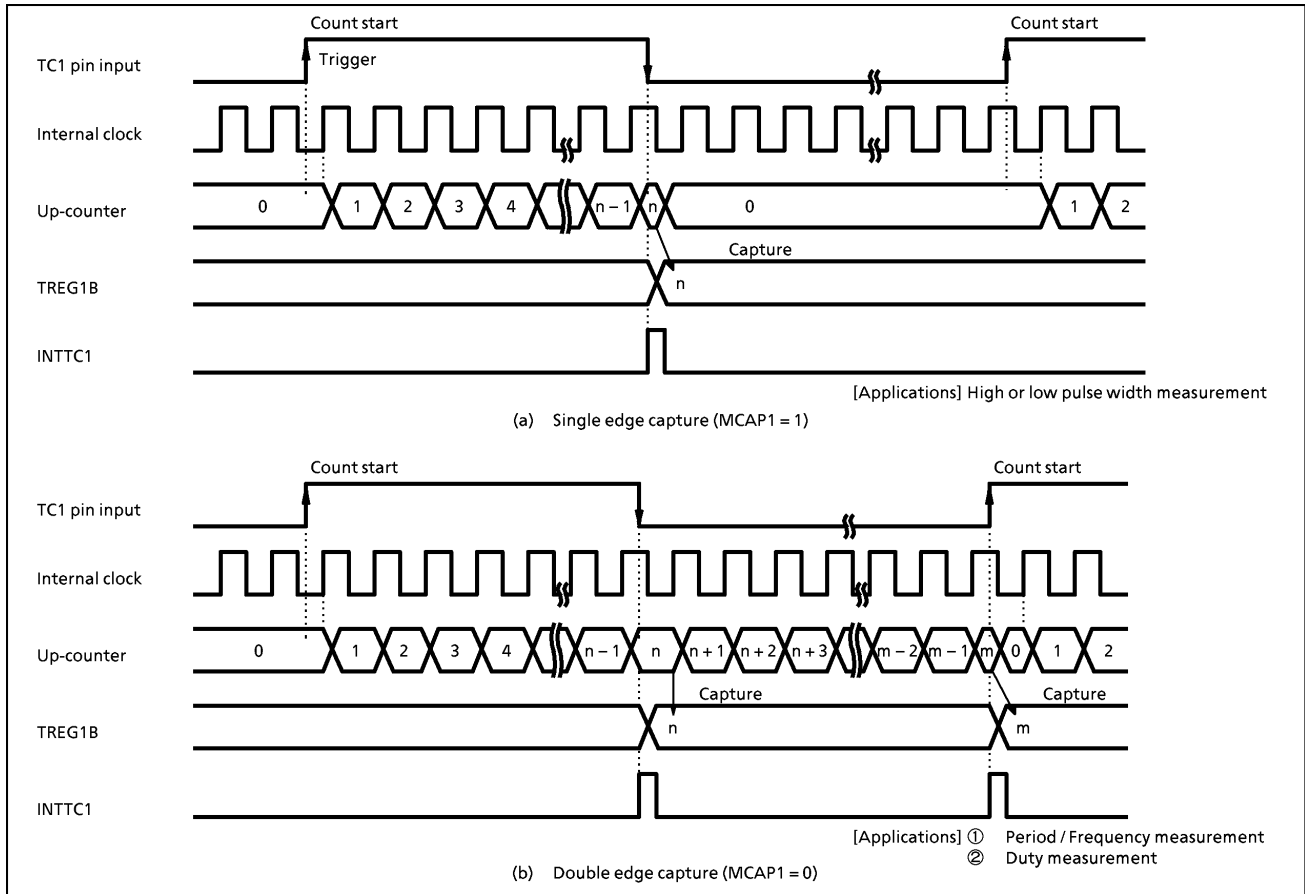


Figure 2-17. Pulse Width Measurement Mode Timing Chart

Example : Duty measurement (Resolution $f_c / 2^7$ [Hz])

```

CLR  (INTTC1C). 0           ; INTTC1 service switch initial setting
LD   (EINTCR), 00000000B   ; Sets the rise edge at the INT2 edge
LD   (TC1CR), 00000110B    ; Sets the TC1 mode and source clock
SET  (EIRL). 4             ; Enables INTTC1
LD   (TC1CR), 00110110B    ; Starts TC1 with an external trigger
:
PINTTC1 : CPL (INTTC1C). 0   ; Complements INTTC1 service switch
        : JRS F, SINTTC1
        : LD (HPULSE), (TREG1BL) ; Reads TREG1B
        : LD (HPULSE + 1), (TREG1BH)
        : RETI
SINTTC1 : LD (WIDTH), (TREG1BL) ; Reads TREG1B (Period)
        : LD (WIDTH + 1), (TREG1BH)
        :
    
```


2.5 16 bit timer / Counter 2 (TC2)

2.5.1 Configuration

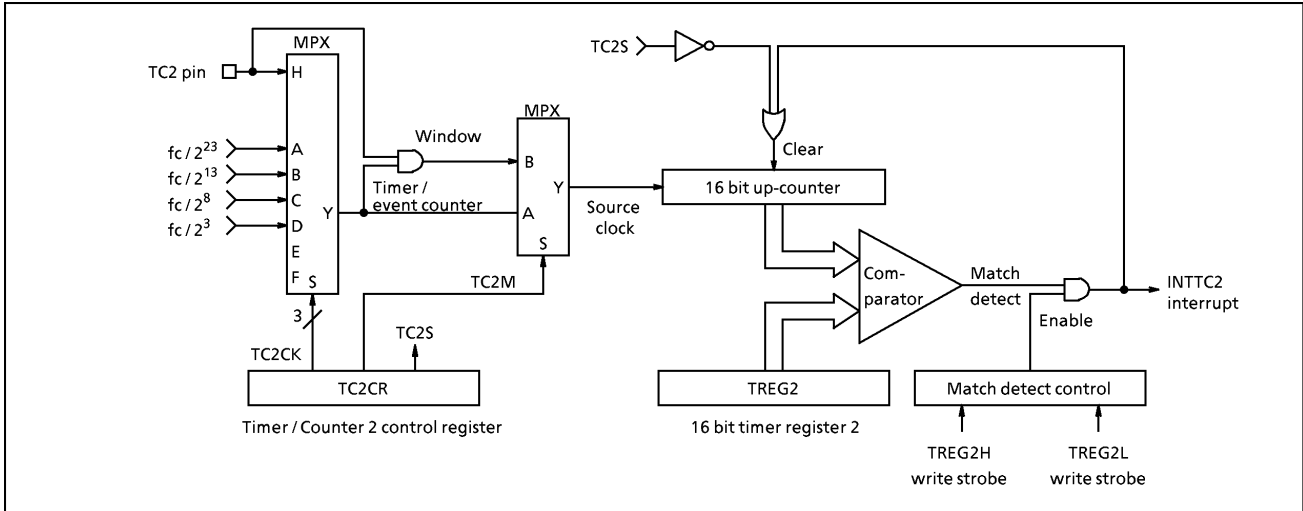


Figure 2-18. Timer / Counter 2 (TC2)

2.5.2 Control

The timer / counter 2 is controlled by a timer / counter 2 control register (TC2CR) and a 16 bit timer register 2 (TREG2). Reset does not affect the TREG2.

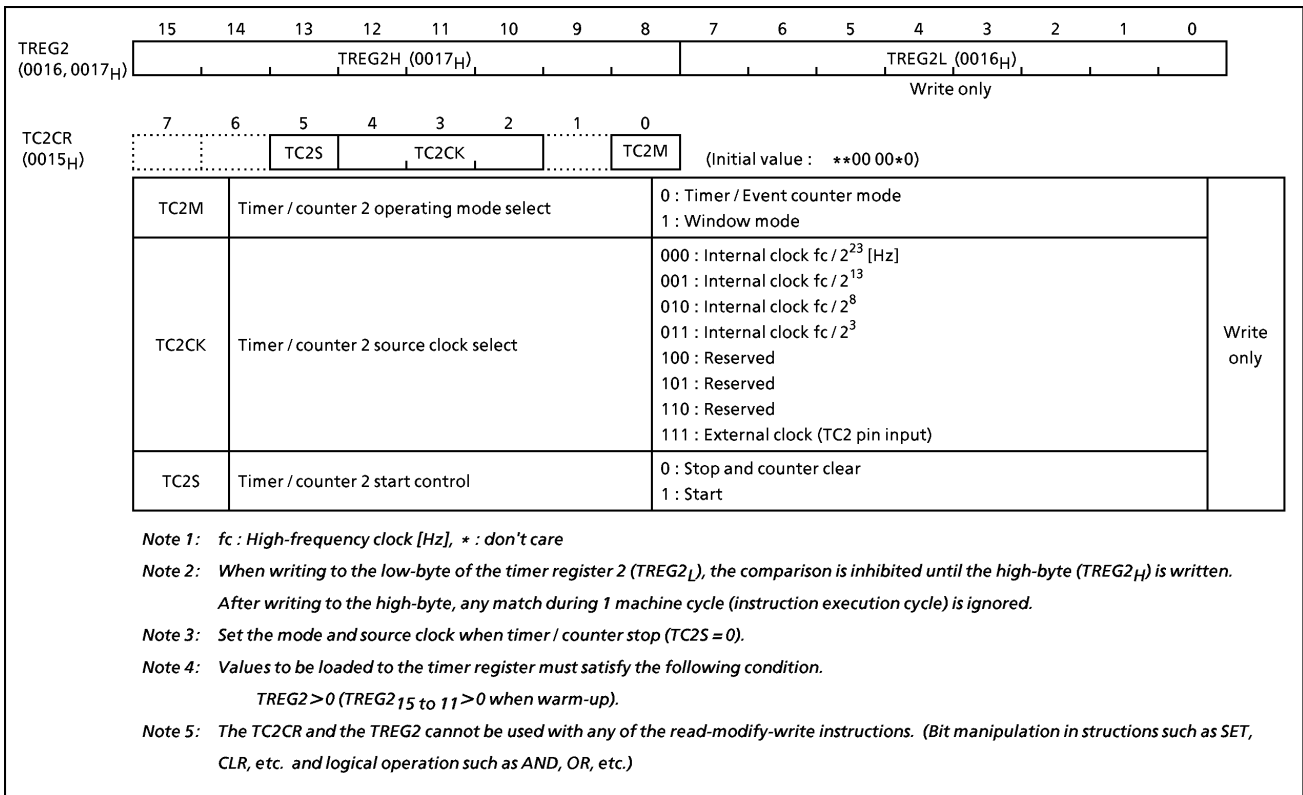


Figure 2-19. Timer Register 2 and TC2 Control Register

2.5.3 Function

The timer / counter 2 has three operating modes : timer, event counter and window modes.

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of the timer register 2 (TREG2) are compared with the contents of the up-counter. If a match is found, a timer / counter 2 interrupt (INTTC2) is generated, and the counter is cleared. Counting up is resumed after the counter is cleared.

Table 2-2. Source Clock (Internal Clock) for Timer 2

Source Clock	Resolution (AT $f_c = 8 \text{ MHz}$)	Maximum Time Setting (AT $f_c = 8 \text{ MHz}$)
$f_c / 2^{23} \text{ [Hz]}$	1.048576 s	19 h 5 min 18.4 s
$f_c / 2^{13}$	1.024 ms	1 min 7.1 s
$f_c / 2^8$	32 μs	2.09712 s
$f_c / 2^3$	1 μs	65.535 ms

Example : Sets the source clock $f_c / 2^3 \text{ [Hz]}$ and generates an interrupt every 25 ms (at $f_c = 8 \text{ MHz}$).

```
LD (TC2CR), 00001100B ; Sets the source clock
LDW (TREG2), 61A8H ; Sets TREG2 (25 ms ÷ 23 / fc = 61A8H)
SET (EIRH). EF14 ; Enables INTTC2 interrupt
EI
LD (TC2CR), 00101100B ; Starts TC2
```

(2) Event counter mode

In this mode, events are counted at the rising edge of the TC2 pin input. The contents of TREG2 are compared with the contents of the up-counter. If a match is found, an INTTC2 interrupt is generated, and the counter is cleared. The maximum frequency applied to the TC2 pin is $f_c / 2^4 \text{ [Hz]}$ in the NORMAL and IDLE mode.

Example : Sets the event counter mode and generates an INTT2 interrupt 640 counts later.

```
LD (TC2CR), 00011100B ; Sets the TC2 mode
LDW (TREG2), 0280H ; Sets TREG2
LD (TC2CR), 00111100B ; Starts TC2
```

(3) Window mode

In this mode, counting up is performed at rising edge of the pulse that is the logical AND-ed product of the TC2 pin input (window pulse) and an internal clock. The internal clock is selected with the TC2CK. The contents of the TREG2 are compared with the contents of the up-counter. If a match is found, an INTTC2 interrupt is generated, and the up-counter is cleared to "0". It is necessary that the maximum applied frequency (TC2 input) be such that the counter value can be analyzed by the program. That is, the frequency must be considerably slower than the selected internal clock.

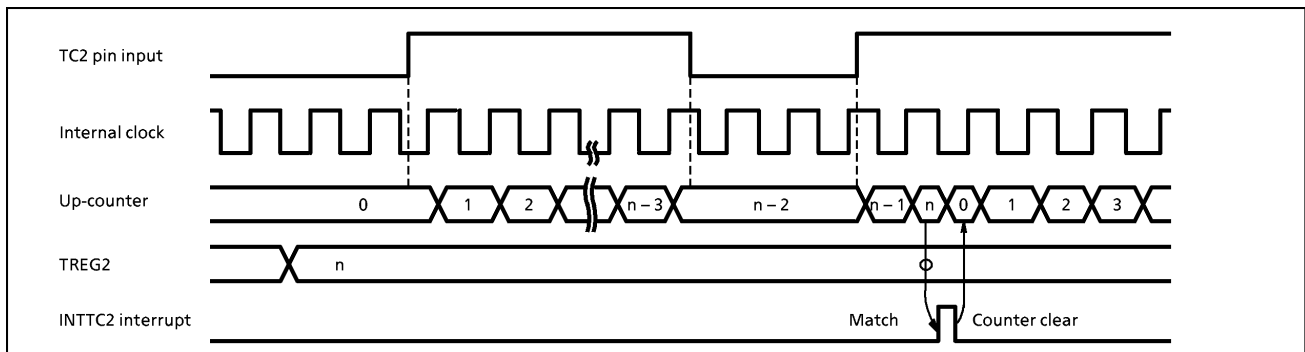


Figure 2-20. Window Mode Timing Chart

2.6 8 bit timer / Counter 3 (TC3)

2.6.1 Configuration

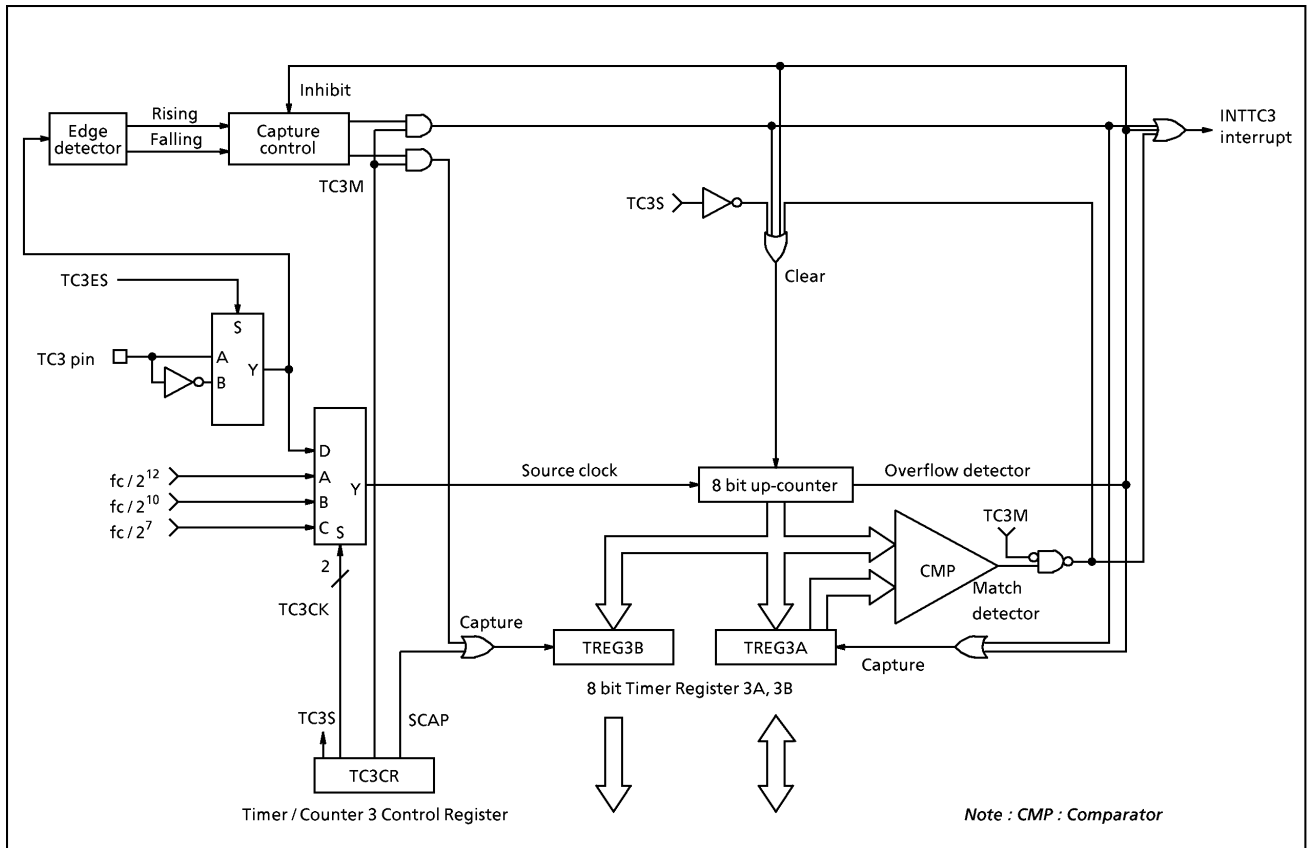


Figure 2-21. Timer / Counter (TC3)

2.6.2 Control

The timer / counter 3 is controlled by a timer / counter 3 control register (TC3CR) and two 8 bit timer registers (TREG3A and TREG3B). Reset does not affect these timer registers.

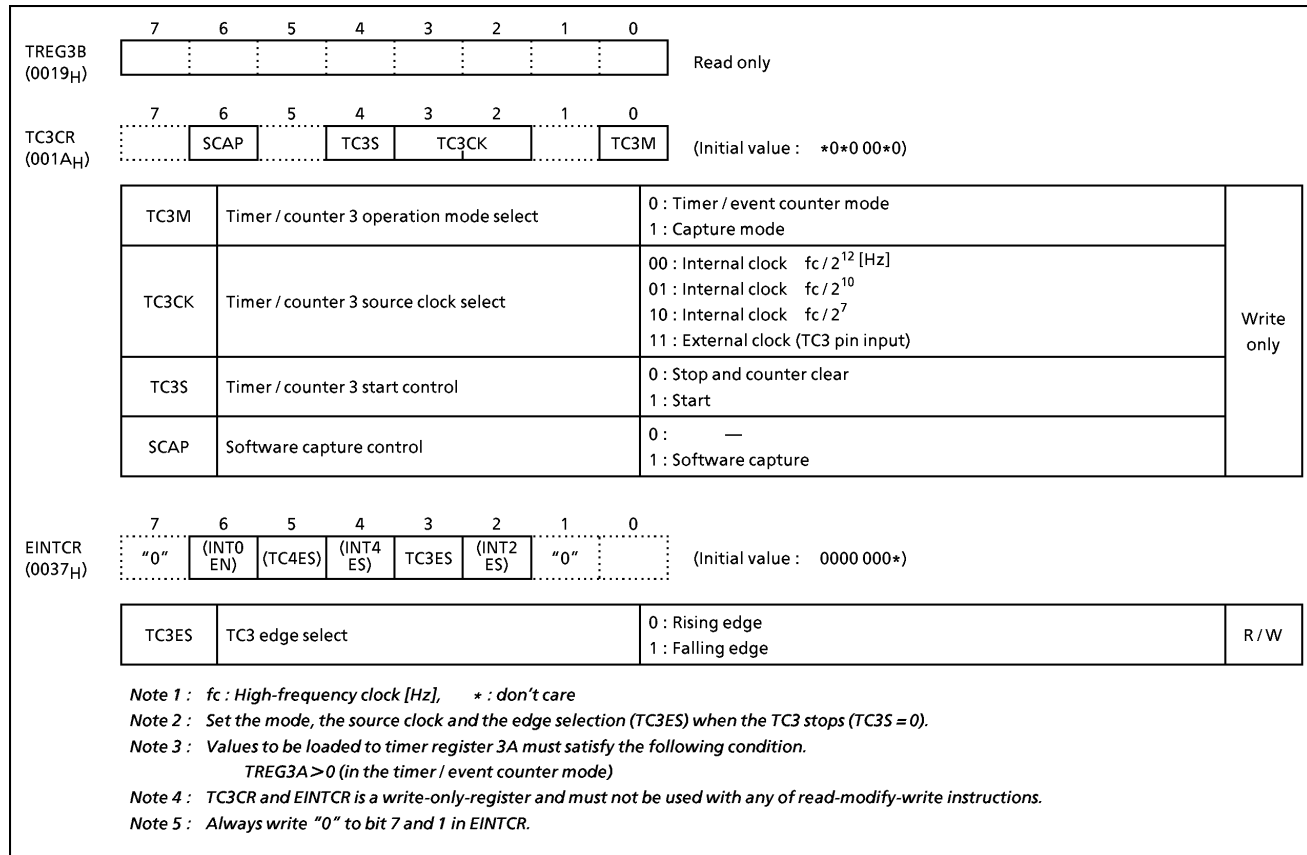


Figure 2-22. Timer Register 3 and TC3 Control Registers

2.6.3 Function

The timer / counter 3 has three operating modes : timer, event counter, and capture mode.

(1) Timer mode

In this mode, the internal clock shown in Table 2-3 is used for counting up. The contents of TREG3A are compared with the contents of the up-counter. If a match is found, a timer / counter 3 interrupt (INTTC3) is generated, and the up-counter is cleared. Counting up resumes after the up-counter is cleared. The current contents of up-counter are loaded into TREG3B by setting SCAP (bit 6 in TC3CR) to "1". SCAP is automatically cleared after capturing.

Table 2-3. Source Clock (Internal Clock) for Timer / Counter 3

Source Clock	Resolution (AT $f_c = 8 \text{ MHz}$)	Maximum Time Setting (AT $f_c = 8 \text{ MHz}$)
$f_c / 2^{12}$	512 μs	130.56 ms
$f_c / 2^{10}$	128 μs	32.64 ms
$f_c / 2^7$	16 μs	4.08 ms

(2) Event counter mode

In this mode, the TC3 pin input pulse are used for counting up. Either the rising or falling edge can be selected with TC3ES (bit 3 in EINTCR). The contents of TREG3A are compared with the contents of the up-counter. If a match is found, an INTTC3 interrupt is generated and the counter is cleared. The maximum applied frequency is $f_c / 2^4$ [Hz]. Two or more machine cycles are required for both the high and low levels of the pulse width.

The current contents of up-counter are loaded into TREG3B by setting SCAP (bit 6 in TC3CR) to "1". SCAP is automatically cleared after capturing.

Example : Generates an interrupt every 0.5 s, inputting 50 Hz pulses to the TC3 pin.

```
LD   (TC3CR), 00011100B   ; Sets TC3 mode and source clock
LD   (TREG3A), 19H        ; 0.5 s ÷ 1 / 50 = 25 = 19H
SET  (EIRH). EF8         ; Enables INTTC3 interrupt
EI
LD   (TC3CR), 0001 1100B   ; Starts TC3
```

(3) Capture mode

The pulse width, period and duty of the TC3 pin input are measured in this mode, which can be used in decoding the remote control signal, etc. The counter is running free by the internal clock. On the rising (falling) edge of the TC3 pin input, the current contents of counter is loaded into TREG3A, then the up-counter is cleared to "0" and an INTTC3 interrupt is generated. On the falling (rising) edge of the TC3 pin input, the current contents of the counter is loaded into TREG3B. In this case, counting continued. On the next rising (falling) edge of the TC3 pin input, the current contents of counter are loaded into TREG3A, then the counter is cleared again and an interrupt is generated. If the counter overflows before the edge is detected, FF_H is set to the TREG3A and an overflow interrupt (INTTC3) is generated. During interrupt processing, it can determine whether or not there is an overflow by checking whether or not the TREG3A value is FF_H. Also, after an interrupt (capture to TREG3A, or overflow detection) is generated, capture and overflow detection are halted until TREG3A has been read out ; however, the counter continues. After TREG3A has been read out, capture and overflow detection start again. Therefore, TREG3B must be read out earlier than TREG3A.

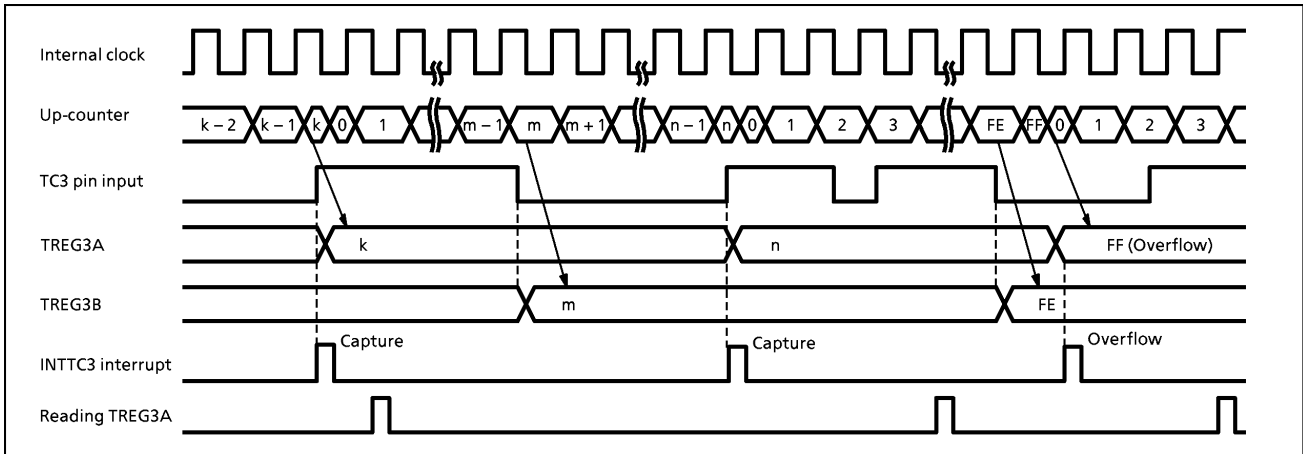


Figure 2-23. Timing Chart for Capture Mode (TC3ES = 0)

2.7 8 bit Timer / Counter (TC4)

2.7.1 Configuration

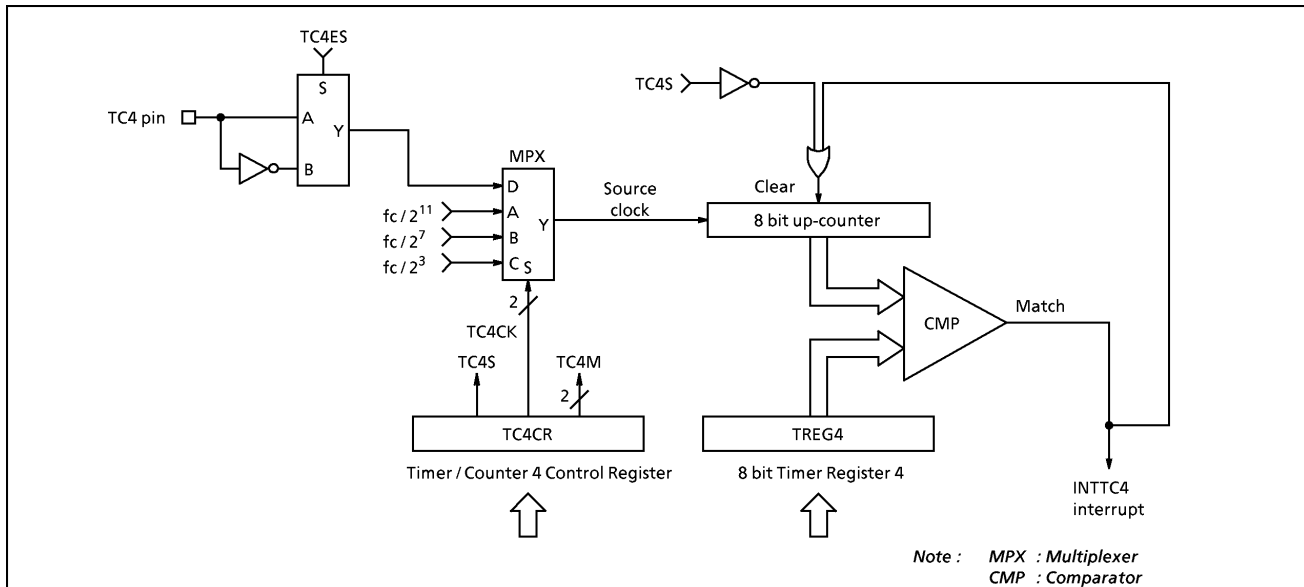


Figure 2-24. Timer / Counter 4

2.7.2 Control

The timer / counter 4 is controlled by a timer / counter 4 control register (TC4CR) and an 8 bit timer register 4 (TREG4). Reset does not affect the TREG4.

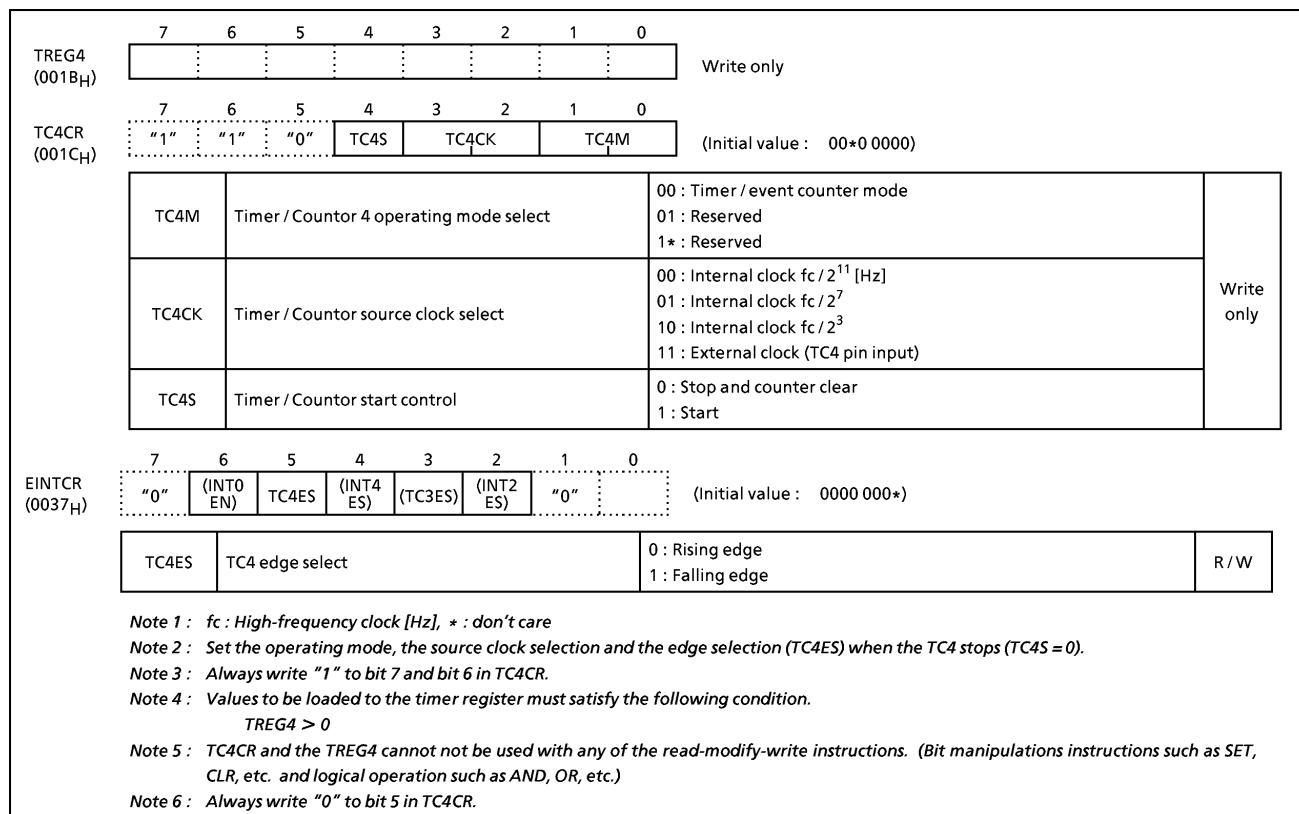


Figure 2-25. Timer Register 4 and TC4 Control Registers

2.7.3 Function

The timer / counter 4 has two operating modes : timer and event counter mode.

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of TREG4 are compared with the contents of the up-counter. If a match is found, a timer / counter 4 interrupt (INTTC4) is generated and the counter is cleared. Counting up resumes after the counter is cleared.

Table 2-4. Source Clock (Internal Clock) for Timer / Counter 4

Source Clock	Resolution (AT $fc = 8$ MHz)	Maximum Time Setting (AT $fc = 8$ MHz)
$fc / 2^{11}$ [Hz]	256 μ s	65.28 ms
$fc / 2^7$	16 μ s	4.08 ms
$fc / 2^3$	1 μ s	255 μ s

(2) Event counter mode

In this mode, the TC4 pin input (external clock) pulse is used for counting up. The contents of TREG4 are compared with the contents of the up-counter. If a match is found, an INTTC4 interrupt is generated and the counter is cleared. Counting up resumes after the counter is cleared. The maximum applied frequency is $fc / 2^4$ [Hz]. Two or more machine cycles are required for both the high and low levels of the pulse width.

2.8 Serial Bus Interface (SBI-ver.A)

The A8700CH / CK / CM / CP / CS has a 1-channel serial bus interface which employs a clocked-synchronous 8-bit serial bus interface and an I²C bus.

The serial bus interface is connected to an external device through P35 (SDA0) / P52 (SDA1) and P34 (SCL0) / P51 (SCL1) in the I²C bus mode; and through P53 ($\overline{SCK1}$), P52 (SO1), and P51 (SI1) in the clocked-synchronous 8-bit SIO mode.

The serial bus interface pins are also used as the P3 / P5 port. When used as serial bus interface pins, set the P3 / P5 output latches of these pins to "1". When not used as serial bus interface pins, the P3 / P5 port is used as a normal I/O port.

I²C bus has no an arbitration function which is necessary when two or more master devices scramble for the bus control. In master mode, other devices which are connected on the same bus need be slave devices. (single master)

Note : When a multi master I²C bus system operates in I²C bus mode of this serial bus interface circuit, there is a possibility that the following problems raise. I²C bus mode of this serial bus interface circuit should be used by a single master I²C bus system.

1. The SCL line is fixed to low level and transferring stops by the serial bus interface circuit. The other devices can not run on the SCL line. Thus the bus locks.
2. The SCL pin is pulled down to low level regardless of the state of the SCL line by the serial bus interface circuit. A period of high-level SCL clock pulse which other devices output is shortened. The minimum value of which the SCL clock holds high level is not satisfied, which is specified with the I2C bus standard.

2.8.1 Configuration

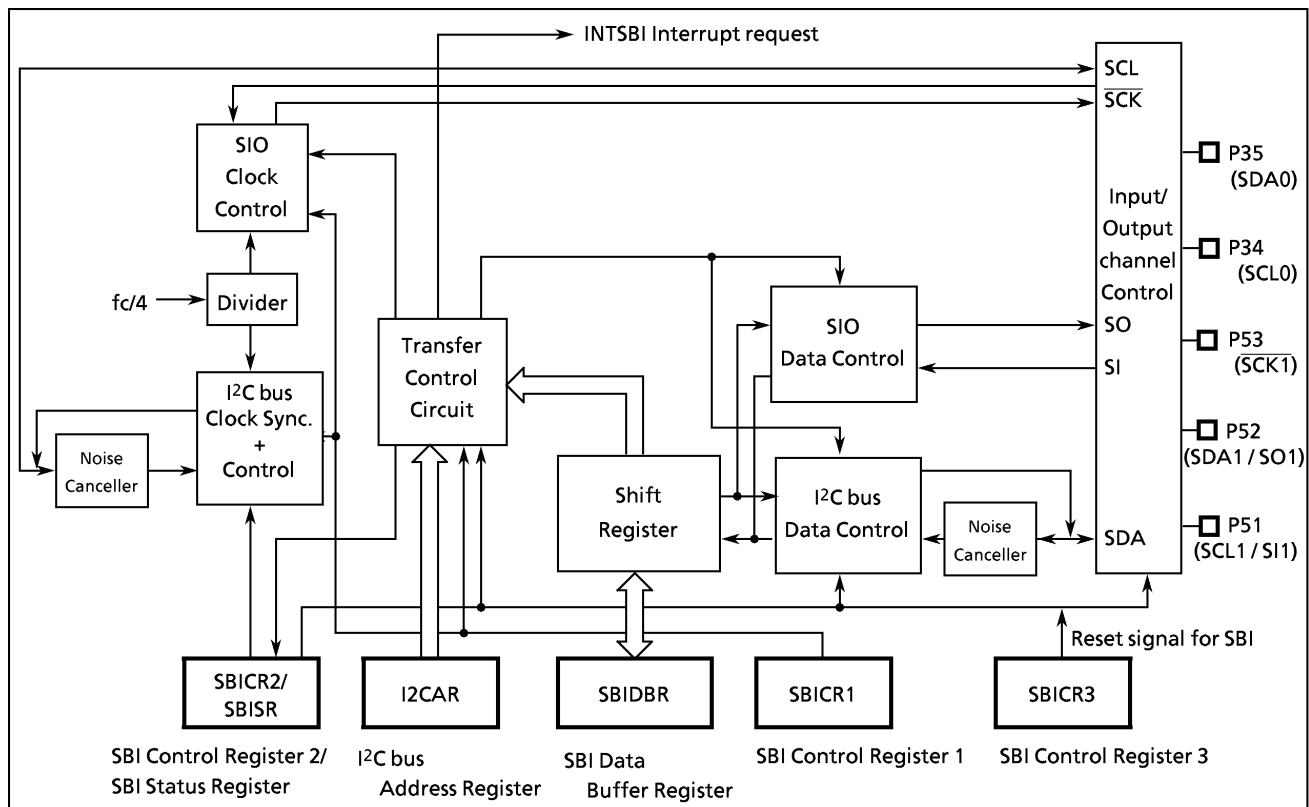


Figure 2-26. Serial Bus Interface (SBI-ver.A)

2.8.2 Serial Bus Interface (SBI-ver.A) Control

The following registers are used for control and operation status monitoring when using the serial bus interface (SBI-ver.A).

- Serial bus interface control register 1 (SBICR1)
- Serial bus interface control register 2 (SBICR2)
- Serial bus interface control register 3 (SBICR3)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register (SBISR)

The above registers differ depending on a mode to be used.

Refer to Section "2.8.4 I²C bus Mode Control" and "2.8.6 Clocked-synchronous 8-bit SIO Mode Control".

2.8.3 The Data Formats in the I²C Bus Mode

The data formats when using the serial bus interface circuit in the I²C bus mode are shown below.

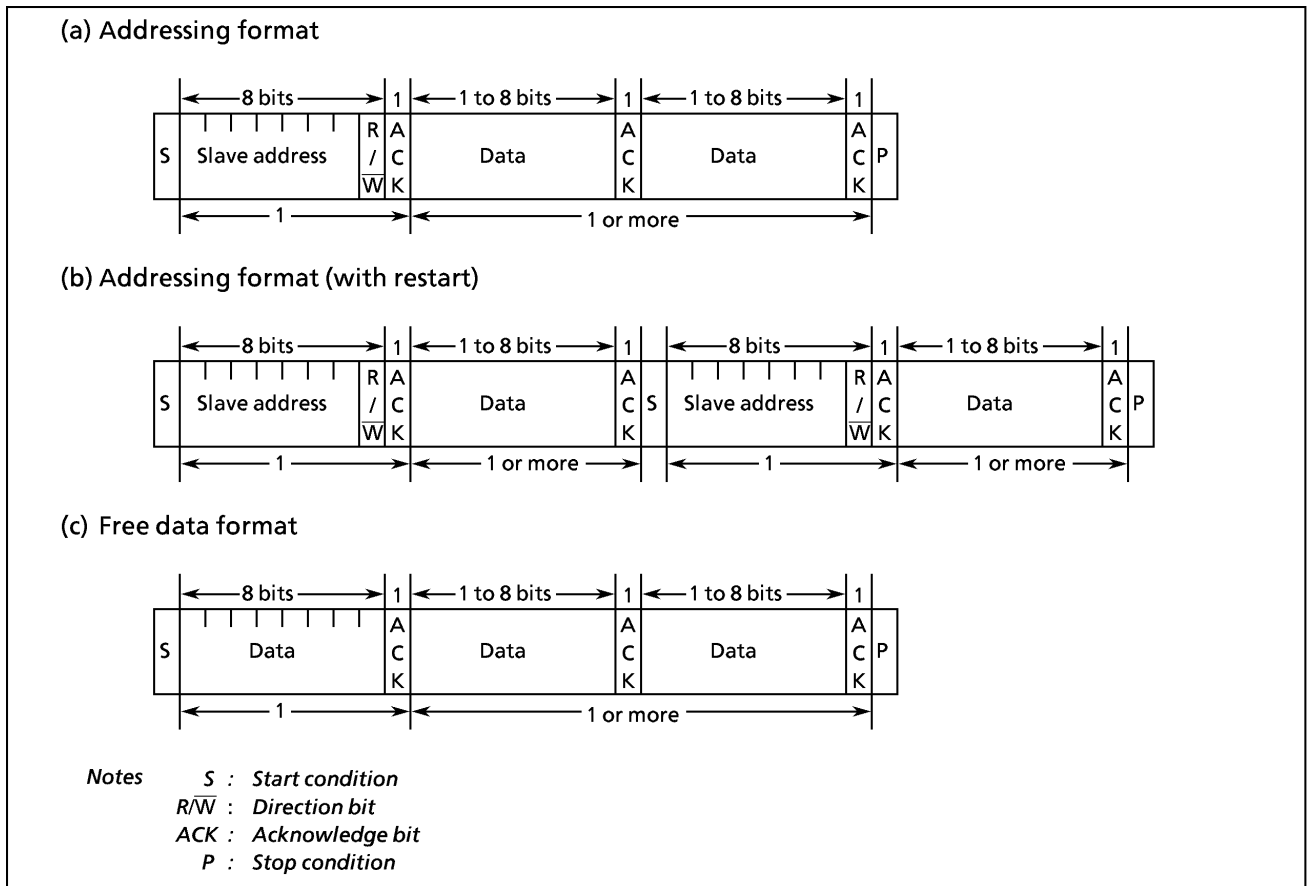


Figure 2-27. Data Format

2.8.4 I²C Bus Mode Control

The following registers are used for control and operation status monitoring when using the serial bus interface (SBI-ver.A) in the I²C bus mode.

Serial Bus Interface Control Register 1								
SBICR1 (0020 _H)	7	6	5	4	3	2 1 0		
	BC		ACK	CHS	SCK		(Initial value : 0000 0000)	
BC	Number of transferred bits	BC	ACK = 0		ACK = 1		Write only	
			Number of Clock	Bits	Number of Clock	Bits		
			000	8	8	9		8
			001	1	1	2		1
			010	2	2	3		2
			011	3	3	4		3
			100	4	4	5		4
			101	5	5	6		5
110	6	6	7	6				
111	7	7	8	7				
ACK	Acknowledge mode specification	0 : Does not generate clock pulse for acknowledgment. (master mode) / Does not count clock pulse for acknowledgment. (slave mode) 1 : Generates clock pulse for acknowledgment. (master mode) / Counts clock pulse for acknowledgment. (slave mode)				Read/Write		
CHS	Input / Output channel selection	0 : Channel0 (SCL0, SDA0) 1 : Channel1 (SCL1, SDA1)				Read/Write		
SCK	Serial clock selection	000 : 181.8 kHz 001 : 105.3 kHz 010 : 57.1 kHz 011 : 29.9 kHz 100 : 15.3 kHz 101 : 7.72 kHz 110 : 3.88 kHz 111 : reserved } at fc = 8 MHz (Output on SCL pin)				Write only		
<p>Note 1 : fc ; high-frequency clock [Hz], * ; don't care</p> <p>Note 2 : Set the BC to "000" before switching to a clock-synchronous 8-bit SIO mode.</p> <p>Note 3 : SBICR1 has write-only register bits, which cannot access any of in read-modify-write instructions such as bit operate, etc.</p> <p>Note 4 : Can not use SIO function when setting the CHS to "0".</p>								
Serial Bus Interface Data Buffer Register								
SBIDBR (0021 _H)	7	6	5	4	3	2 1 0		
							(Initial value : **** *) Read / Write	
<p>Note 1 : When writing transmitted data, start from the MSB.</p> <p>Note 2 : Cannot read the data which was written into SBIDBR, since a write data buffer and a read data buffer are independent in SBIDBR. Therefore, cannot access it any of in read-modify-write instructions such as bit operate, etc.</p> <p>Note3 : A value written to SBIDBR is cleared to "0" by INTSBI interrupt request signal.</p> <p>Note4 : * ; don't care</p>								
I ² C bus Address Register								
I2CAR (0022 _H)	7	6	5	4	3	2 1 0		
	Slave address						ALS	
SA6 SA5 SA4 SA3 SA2 SA1 SA0						(Initial value : 0000 0000)		
SA	A8700CH / CK / CM / CS slave address selection					Write only		
ALS	Address recognition mode specification							
						0 : Slave address recognition 1 : Non slave address recognition		
<p>Note : I2CAR is a write-only register, which cannot access any of in read-modify-write instructions such as bit operate, etc.</p>								

Figure 2-28. Serial Bus Interface Control Register 1 / Serial Bus Interface Data Buffer Register/ I²C Bus Address Register in the I²C Bus Mode

Serial Bus Interface Control Register 2										
SBICR2 (0023 _H)		7	6	5	4	3	2	1	0	
		MST	TRX	BB	PIN	SBIM		"0"	"0"	
(Initial value : 0001 00**)										
MST	Master / slave selection								0 : Slave 1 : Master	Write only
TRX	Transmitter / receiver selection								0 : Receiver 1 : Transmitter	
BB	Start / stop generation								0 : Generate the stop condition when the MST, TRX, and PIN are "1". 1 : Generate the start condition when the MST, TRX, and PIN are "1".	
PIN	Cancel interrupt service request								0 : - 1 : Cancel interrupt service request	
SBIM	Serial bus interface operating mode selection								00 : Port mode (serial bus interface output disable) 01 : SIO mode 10 : I ² C bus mode 11 : Reserved	
<p><i>Note 1 :</i> * ; don't care <i>Note 2 :</i> Switch a mode to port mode after confirming that the bus is free. <i>Note 3 :</i> Switch a mode to I²C bus mode after confirming that input signals via port are high level. <i>Note 4 :</i> SBICR2 has write-only register bits, which can not access any of in read-modify-write instructions such as bit operate, etc. <i>Note 5 :</i> Clear bits 1 and 0 in SBICR2 to "0".</p>										
Serial Bus Interface Status Register										
SBISR (0023 _H)		7	6	5	4	3	2	1	0	
		MST	TRX	BB	PIN	AL	AAS	AD0	LRB	
(Initial value : 0001 0000)										
MST	Master / Slave selection status monitor								0 : Slave 1 : Master	Read only
TRX	Transmitter / Receiver selection status monitor								0 : Receiver 1 : Transmitter	
BB	Bus status monitor								0 : Bus free 1 : Bus busy	
PIN	Interrupt service request status monitor								0 : Requesting interrupt service 1 : Releasing interrupt service request	
AL	Noise detection monitor								0 : Does not detect noise 1 : Detects noise	
AAS	Slave address match detection monitor								0 : Does not detect slave address match or "GENERAL CALL" 1 : Detects slave address match or "GENERAL CALL"	
AD0	"GENERAL CALL" detection monitor								0 : Does not detect "GENERAL CALL" 1 : Detects "GENERAL CALL"	
LRB	Last received bit monitor								0 : Last received bit is "0" 1 : Last received bit is "1"	
Serial Bus Interface Control Register 3										
SBICR3 (0024 _H)		7	6	5	4	3	2	1	0	
									SWRST	
(Initial value : **** **0)										
SWRST	Software Reset								0 : — (Initial state) 1 : Initialize SBI (After initializing SBI, the SWRST is automatically cleared to "0")	R / W

Figure 2-29. Serial Bus Interface Control Register 2 / Serial Bus Interface Status Register in the I²C Bus Mode

(1) Acknowledgment mode specification

Set the ACK (bit 4 in SBICR1) to "1" for operation in acknowledgment mode. When the serial bus interface circuit is the master mode, an additional clock pulse is generated for an acknowledge signal. In the transmitter mode during this additional clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during this additional clock pulse cycle, the SDA pin is set to low level generating the acknowledge signal.

Clear the ACK to "0" for operation in the non-acknowledgment mode. When the serial bus interface circuit is the master mode, a clock pulse for the acknowledge signal is not generated.

In the acknowledgment mode, when the serial bus interface circuit is the slave mode, clocks are counted for the acknowledge signal. During the clock for the acknowledge signal, when a received slave address matches to a slave address set to the I2CAR or a "GENERAL CALL" is received, the SDA pin is set to low level generating an acknowledge signal.

After a received slave address matches to a slave address set to the I2CAR and a "GENERAL CALL" is received, in the transmitter mode during the clock for the acknowledge signal, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode, the SDA pin is set to low level generating an acknowledge signal.

In the non-acknowledgment mode, when the serial bus interface circuit is the slave mode, clocks for the acknowledge signal are not counted.

(2) Number of transfer bits

The BC (bits 7 to 5 in the SBICR1) is used to select a number of bits for next transmitting and receiving data.

Since the BC is cleared to "000" by a start condition, a slave address and direction bit transmissions are executed in 8 bits. Other than these, the BC retains a specified value.

(3) Serial clock

a. Clock source

The SCK (bits 2 to 0 in the SBICR1) is used to select a maximum transfer frequency outputted on the SCL pin in the master mode.

Four or more machine cycles are required for both the high and low levels of the pulse width of a clock which is input externally in both the master and slave mode.

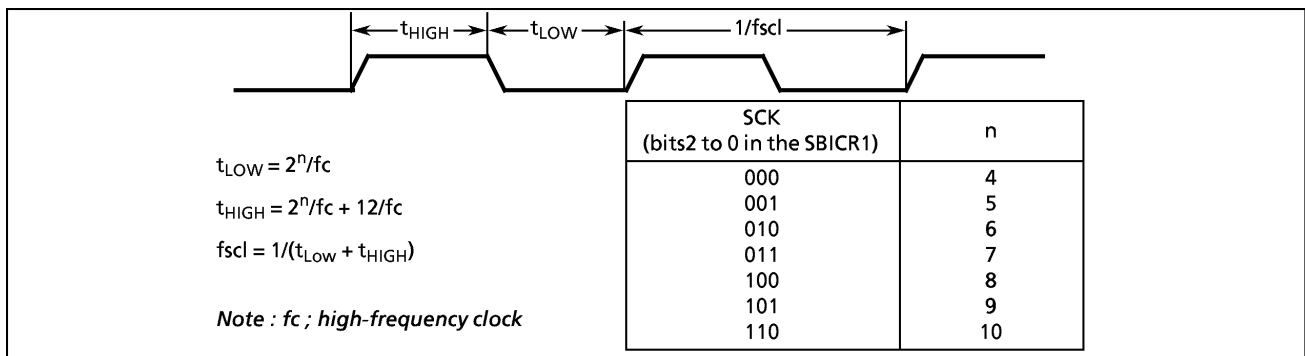


Figure 2-30. Clock Source

b. Clock synchronization

The I²C bus has a clock synchronization function to meet the transfer speed to a slow processing device when a transfer is performed between devices which have different process speed.

The clock synchronization functions when the SCL pin is high level and the SCL line of the bus is low level in the serial bus interface circuit. The serial bus interface circuit waits counting a clock pulse in high level until the SCL line of the bus is high level. When the SCL line of the bus is high level, the serial bus interface circuit starts counting during high level. The clock synchronization function holds clocks which are output from the serial interface circuit to be high level.

The slave device can stop the clock output of the master device on one word or one bit basis. Additionally, the transfer speed by the master device matches to the process speed of the slave device.

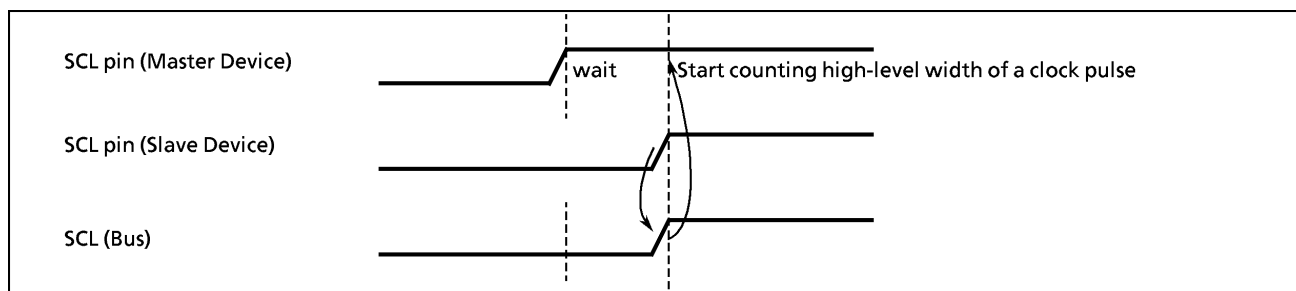


Figure 2-31. Clock Synchronization

(4) Slave address and address recognition mode specification

To operate the serial bus interface circuit in the addressing format which recognizes the slave address, clear the ALS (bit 0 in I2CAR) to "0" and set the slave address to the SA (bits 7 to 1 in I2CAR).

To operate the serial bus interface circuit in the free data format which does not recognize the slave address, set the ALS to "1". When the serial bus interface circuit is used in the free data format, the slave address and the direction bit are not recognized. They are handled as data just after generation of start conditions.

(5) Master/slave selection

Set the MST (bit 7 in the SBICR2) to "1" for operating the serial bus interface as a master device. Clear the MST to "0" for operation as a slave device. The MST is cleared to "0" by the hardware after a stop condition on a bus is detected or the noise is detected.

(6) Transmitter / receiver selection

Set the TRX (bit 6 in the SBICR2) to "1" for operating the serial bus interface circuit as a transmitter. Clear the TRX to "0" for operation as a receiver. When data with an addressing format is transferred in the slave mode, the TRX is set to "1" by the hardware if the direction bit (R/W) sent from the master device is "1", and is cleared to "0" by the hardware if the bit is "0". In the master mode, after an acknowledge signal is returned from the slave device, the TRX is cleared to "0" by the hardware if a transmitted direction bit is "1", and is set to "1" by the hardware if it is "0". When an acknowledge signal is not returned, the current condition is maintained.

The TRX is cleared to "0" by the hardware after a stop condition on the bus is detected or the noise is detected.

The following shows TRX change conditions in each mode and TRX after changing.

Mode	Direction bit	Change condition	TRX after changing
Slave mode	0	A received slave address is the same as a value set to I2CAR.	0
	1		1
Master mode	0	ACK signal is returned.	1
	1		0

When the serial bus interface circuit operates in the free data format, the slave address and the direction bit are not recognized. They are handled as data just after generating a start condition. The TRX was not changed by the hardware.

(7) Start/stop condition generation

When the BB (bit 5 in the SBICR2) is "0", the slave address and the direction bit which are set to the SBIDBR are output on a bus after generating a start condition by writing "1" to the MST, TRX, BB, and PIN. It is necessary to set transmitted data to the data buffer register (SBIDBR) and set "1" to ACK beforehand.

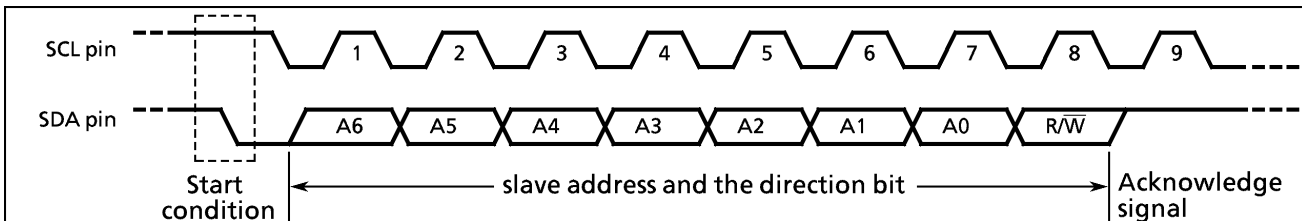


Figure 2-32. Start Condition Generation and Slave Address Generation

When the BB is "1", a sequence of generating a stop condition is started by writing "1" to the MST, TRX, and PIN, and "0" to the BB. Do not modify the contents of MST, TRX, BB and PIN until a stop condition is generated on a bus.

When a stop condition is generated and the SCL line on the bus is set to low level by another device, a stop condition is generated after releasing the SCL line.

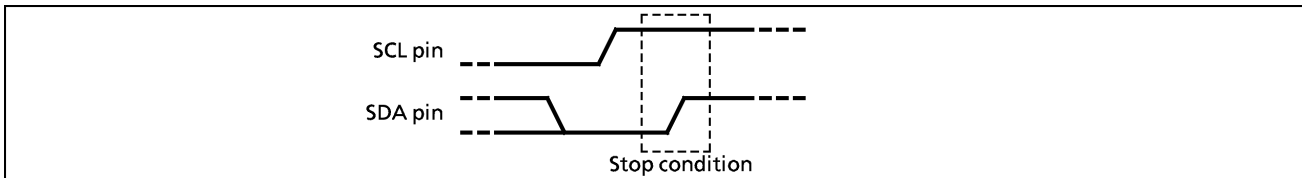


Figure 2-33. Stop Condition Generation

The bus condition can be indicated by reading the contents of the BB (bit 5 in the SBISR). The BB is set to "1" when a start condition on a bus is detected, and is cleared to "0" when a stop condition is detected on a bus.

(8) Interrupt service request and cancel

When the serial bus interface circuit is the master mode and transferring a number of clocks set by the BC and the ACK is complete, a serial bus interface interrupt request (INTSBI) is generated.

In the slave mode, the INTSBI is generated when the received slave address is the same as the value set to the I2CAR and an acknowledge signal is output, when a "GENERAL CALL" is received and an acknowledge signal is output, or when transferring / receiving data is complete after the received slave address is the same as the value set to the I2CAR and a "GENERAL CALL" is received.

When the serial bus interface interrupt request occurs, the PIN (bit 4 in the SBISR) is cleared to "0". During the time that the PIN is "0", the SCL pin is set to low level.

Either writing or reading data to or from the SBIDBR sets the PIN to "1".

The time from the PIN being set to "1" until the SCL pin is released takes t_{LOW} .

Although the PIN (bit 4 in the SBICR2) can be set to "1" by the program, the PIN is not cleared to "0" when it is written "0".

(9) Serial bus interface operating mode selection

The SBIM (bits 3 and 2 in the SBICR2) is used to specify the serial bus interface operation mode.

Set the SBIM to "10" when used in the I²C bus mode after confirming that the serial bus interface pin is high level. Switch a mode to port after confirming that the bus is free.

(10) Noise detection monitor

The I²C bus is easy to be affected by noise, because the bus is driven by the open drain and the pull-up resistor.

With the serial bus interface circuit, the SDA pin output and the SDA line level are compared at a rise of the SCL line on the bus, and whether data are output correctly on the bus is detected only in the master transmitter mode.

When the SDA pin output differs from the SDA line level, the AL (bit 3 in the SBISR) is set to "1". When the AL is set to "1", the SDA pin is released and the MST and the TRX are cleared to "0" by the hardware. The serial bus interface circuit changes to the slave receiver mode, and the serial bus interface circuit continues outputting clocks until transferring data when the AL was set to "1" is completed.

Either writing or reading data to or from the SBIDBR, or writing data to the SBICR2 clears to the AL to "0".

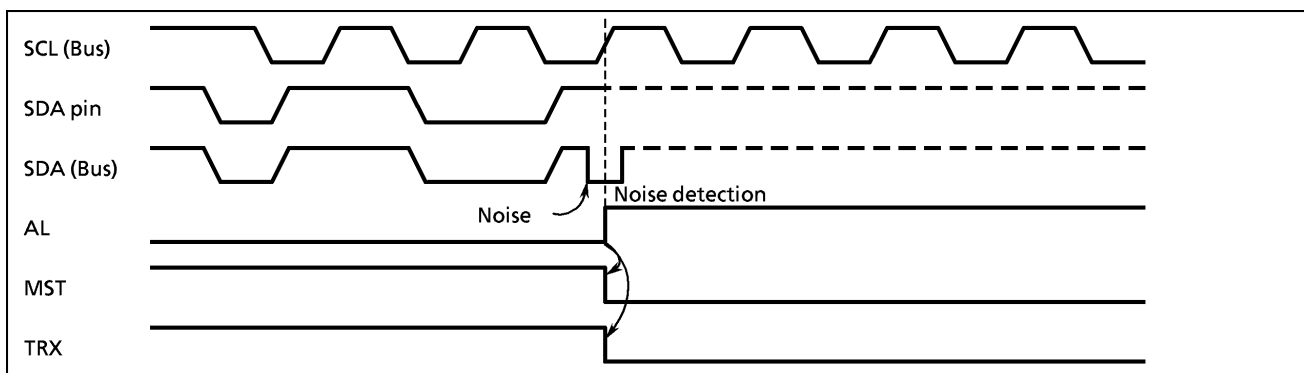


Figure 2-34. Noise Detection Monitor

(11) Slave address match detection monitor

The AAS (bit 2 in the SBISR) is set to "1" in the slave mode, in the address recognition mode (ALS = 0), when receiving "GENERAL CALL" or a slave address with the same value that is set to the I2CAR. When the ALS is "1", the AAS is set to "1" after receiving the first 1-word of data. The AAS is cleared to "0" by writing / reading data to / from a data buffer register.

(12) GENERAL CALL detection monitor

The AD0 (bit 1 in the SBISR) is set to "1" in the slave mode, when all 8-bit received data is "0", after a start condition (GENERAL CALL). The AD0 is cleared to "0" when a start or stop condition is detected on a bus.

(13) Last received bit monitor

The SDA value stored at the rising edge of the SCL is set to the LRB (bit 0 in the SBISR). In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the LSB.

(14) Software reset

Software reset function can be used for initializing just serial bus interface when Serial bus interface is rocked by external noise, etc.

The SWRST (bit 0 of SBICR3) is set to "1", internal reset signal pulse is generated and inputted into Serial bus interface circuit. All command registers and status register are initialized to the initial value. The SWRST is automatically cleared to "0" after initializing Serial bus interface circuit.

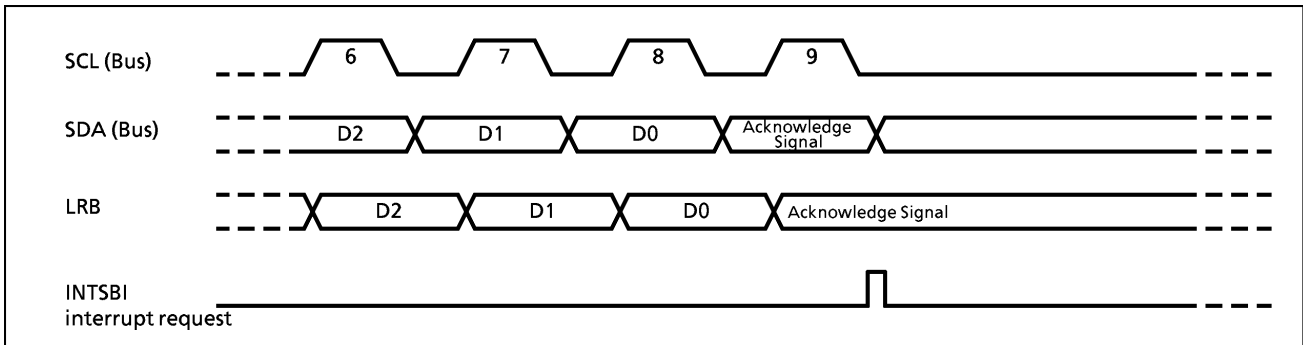


Figure 2-35. Last Received Bit Monitor

2.8.5 Data Transfer in I²C Bus Mode

(1) Device Initialization

Set the ACK in the SBICR1 to "1", and the BC to 000. Specify the data length to 8 bits to count clocks for acknowledge. Set a transfer frequency to the SCK and a serial bus interface pin to the CHS. Subsequently, set a slave address to the SA in the I2CAR and clear the ALS to "0" to set an addressing format.

After confirming that the serial bus interface pin is high-level, for specifying the default setting to a slave receiver mode, clear "0" to the MST, TRX, and BB in the SBICR2, set "1" to the PIN, "10" to the SBIM, and "0" to bits 1 and 0,

Note : The initialization of the serial bus interface circuit must be complete within the time from all devices which are connected to the bus have initialized to any device does not generate a start condition. If not, there is a possibility that another device starts transferring before an end of the initialization of the serial bus interface circuit. Data can not be received correctly.

(2) Start condition and slave address generation

Confirm a bus free status (when BB = 0).

Set the ACK to "1" and specify a slave address and a direction bit to be transmitted to the SBIDBR.

When the BB is "0", the start condition are generated and the slave address and the direction bit which are set to the SBIDBR are output on a bus by writing "1" to the MST, TRX, BB and PIN. An INTSBI interrupt request occurs at the 9th falling edge of the SCL clock cycle, and the PIN is cleared to "0". The SCL pin is pulled down to the low-level while the PIN is "0". When an interrupt request occurs, the TRX changes by the hardware according to the direction bit only when an acknowledge signal is returned from the slave device.

Note 1 : Do not write a slave address to be output to the SBIDBR while data are transferred. If data is written to the SBIDBR, data to been outputting may be destroyed.

Note 2 : Do not start transferring due to another master from writing a slave address to be output to the SBIDBR to writing a start condition generation command to the SBICR2. The serial bus interface circuit malfunctions because it has not an arbitration function.

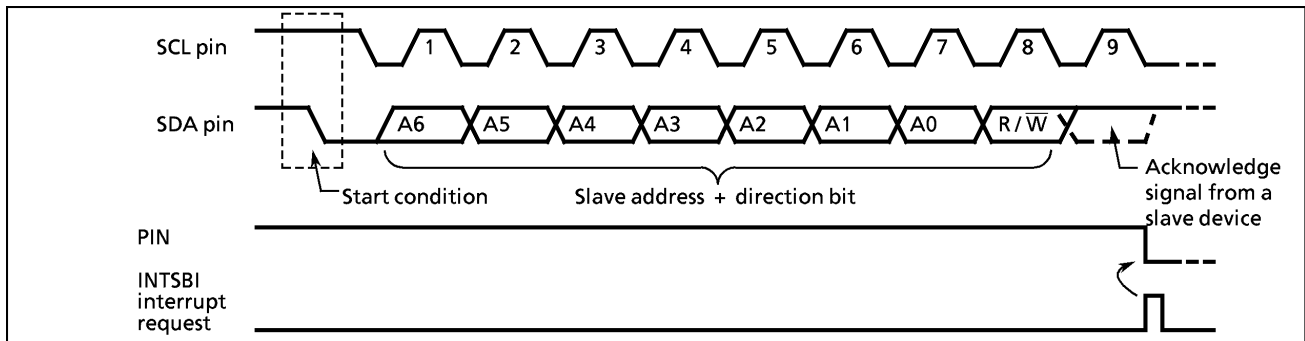


Figure 2-36. Start Condition Generation and Slave Address Transfer

(3) 1-word data transfer

Check the MST by the INTSBI interrupt process after an 1-word data transfer is completed, and determine whether the mode is a master or slave.

a. When the MST is "1" (Master mode)

Check the TRX and determine whether the mode is a transmitter or receiver.

① When the TRX is "1" (Master mode)

Test the LRB. When the LRB is "1", a receiver does not request data. Implement the process to generate a stop condition (described later) and terminate data transfer.

When the LRB is "0", the receiver requests new data. When the next transmitted data is other than 8 bits, set the BC, set the ACK to "1", and write the transmitted data to the SBIDBR. After writing the data, the PIN becomes "1", a serial clock pulse is generated for transferring a new 1-word of data from the SCL pin, and then the 1-word data is transmitted. After the data is transmitted, and an INTSBI interrupt request occurs. The PIN becomes "0" and the SCL pin is set to low level. If the data to be transferred is more than one word in length, repeat the procedure from the LRB test above.

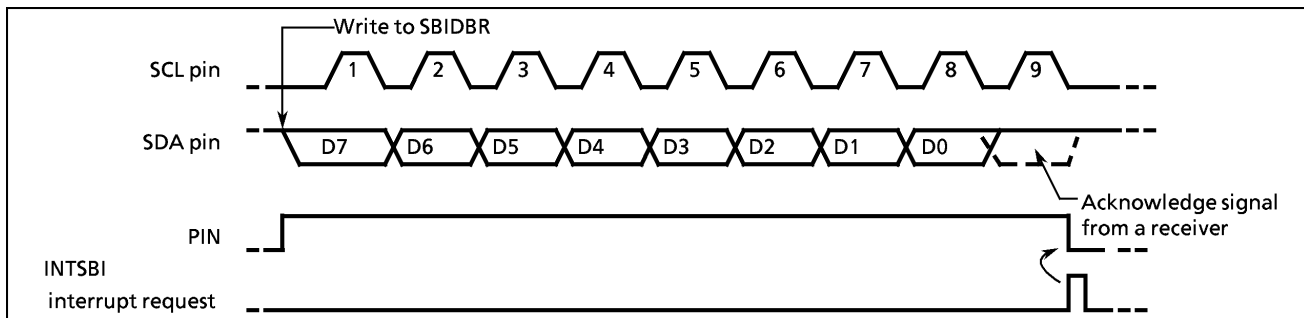


Figure 2-37. Example When BC = "000", ACK = "1" in Transmitter Mode

② When the TRX is "0" (Receiver mode)

When the next transmitted data is other than 8 bits, set the BC again. Set the ACK to "1" and read the received data from the SBIDBR (data which is read immediately after a slave address is sent is undefined). After the data is read, the PIN becomes "1". The serial bus interface circuit outputs a serial clock pulse to the SCL to transfer new 1-word of data and sets the SDA pin to "0" at the acknowledge signal timing.

An INTSBI interrupt request occurs and the PIN becomes "0". Then the serial bus interface circuit pulls down the SCL pin to the low level. The serial bus interface circuit outputs a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

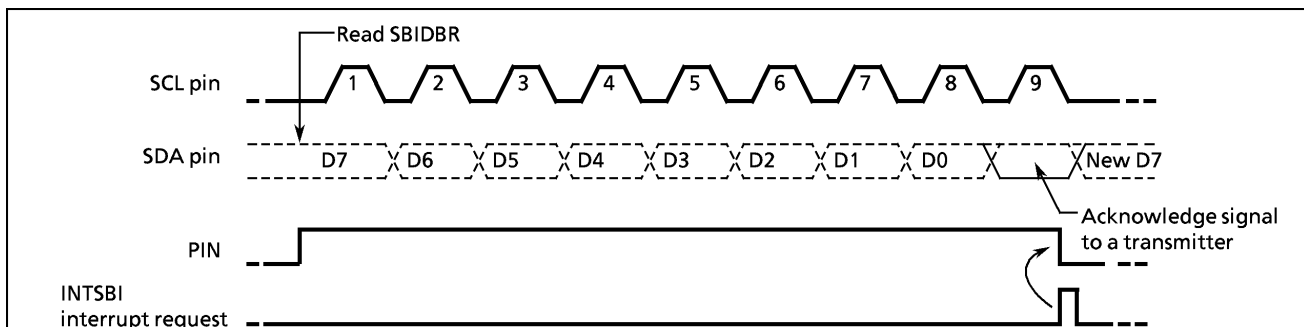


Figure 2-38. Example When BC = "000", ACK = "1" in Receiver Mode

In order to terminate transmitting data to a transmitter, clear the ACK to "0" before reading data which is 1 word before the last data to be received. The last data does not generate a clock pulse for the acknowledge signal. After the data is transmitted and an interrupt request has occurred, set the BC to "001" and read the data. The serial bus interface circuit generates a clock pulse for a 1-bit data transfer. Since the master device is a receiver, the SDA line on a bus keeps the high level. The transmitter receives the high-level signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete. After 1-bit data is received and an interrupt request has occurred, the serial bus interface circuit generates a stop condition (Refer to 2.8.5. (4)) and terminates data transfer.

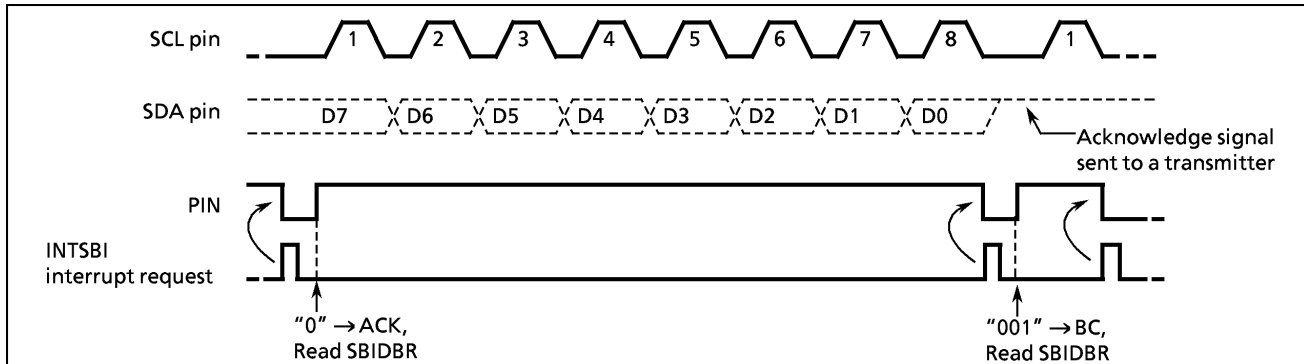


Figure 2-39. Termination of Data Transfer in Master Receiver Mode

b. When the MST is "0" (Slave mode)

In the slave mode, the serial bus interface circuit operates either in normal slave mode or in recovery process after a noise detection.

In the slave mode, an INTSBI interrupt request occurs when the serial bus interface circuit receives a slave address or a "GENERAL CALL" from the master device, or when a "GENERAL CALL" is received and data transfer is complete after matching a received slave address. In the master mode, the serial bus interface circuit operates in a slave mode if a noise is detected. An INTSBI interrupt request occurs when word data transfer terminates after a noise detection. When an INTSBI interrupt request occurs, the PIN (bit 4 in the SBICR2) is reset, and the SCL pin is set to low level. Either reading or writing from or to the SBIDBR or setting the PIN to "1" releases the SCL pin after taking t_{LOW} time. The serial bus interface circuit tests the AL (bit 3 in the SBISR), the TRX (bit 6 in the SBISR), the AAS (bit 2 in the SBISR), and the AD0 (bit 1 in the SBISR) and implements processes according to conditions listed in the next table.

Table 2-5. Operation in The Slave Mode

TRX	AL	AAS	AD0	Conditions	Process
1	0	1	0	In the slave receiver mode, the serial bus interface circuit receives a slave address of which the value of the direction bit sent from the master is "1".	Set the number of bits in 1-word to the BC and write transmitted data to the SBIDBR.
		0	0	In the slave transmitter mode, 1-word data is transmitted.	Check the LRB. If the LRB is set to "1", set the PIN to "1" since the receiver does not request next data. Then, clear the TRX to "0" release the bus. If the LRB is cleared to "0", set the number of bits in a word to the BC and write transmitted data to the SBIDBR since the receiver requests next data.
0	1	0	0	The serial bus interface circuit detects the noise when transmitting a slave address or data and terminates transferring word data.	There is a possibility that a serial bus interface circuit does not receive data normally. The recovery process such as a data re-transfer, etc. is needed.
				In the slave receiver mode, the serial bus interface circuit receives a slave address or GENERAL CALL of which the value of the direction bit sent from the master is "0".	Read the SBIDBR for setting the PIN to "1" (reading dummy data) or set the PIN to "1".
				In the slave receiver mode, the serial bus interface circuit terminates receiving of 1-word data.	Set the number of bits in a word to the BC and read received data from the SBIDBR.

(4) Stop condition generation

When the BB is "1", a sequence of generating a stop condition is started by setting "1" to the MST, TRX and PIN, and "0" to the BB. Do not modify the contents of the MST, TRX, BB, PIN until a stop condition is generated on a bus. When a SCL line of bus is pulled down by other devices, the serial bus interface circuit generates a stop condition after they release a SCL line.

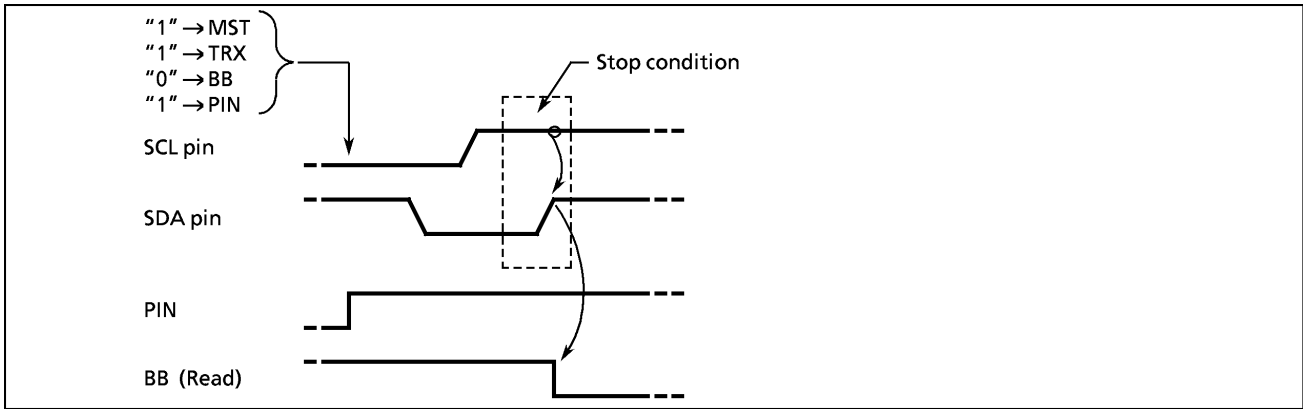


Figure 2-40. Stop Condition Generation

(5) Restart

Restart is used to change the direction of data transfer between a master device and a slave device during transferring data. The following explains how to restart the serial bus interface circuit.

Clear "0" to the MST, TRX, and BB and set "1" to the PIN. The SDA pin retains the high level and the SCL pin is released. Since a stop condition is not generated on the bus, the bus is assumed to be in a busy state from other devices. Test the BB until it becomes "0" to check that the SCL pin of the serial bus interface circuit is released. Test the LRB until it becomes "1" to check that the SCL line of the bus is not set to low level by other devices. After confirming that the bus stays in a free state, generate a start condition with procedure (2).

In order to meet setup time when restarting, take at least 4.7 μ s of waiting time by software from the time of restarting to confirm that the bus is free until the time to generate the start condition.

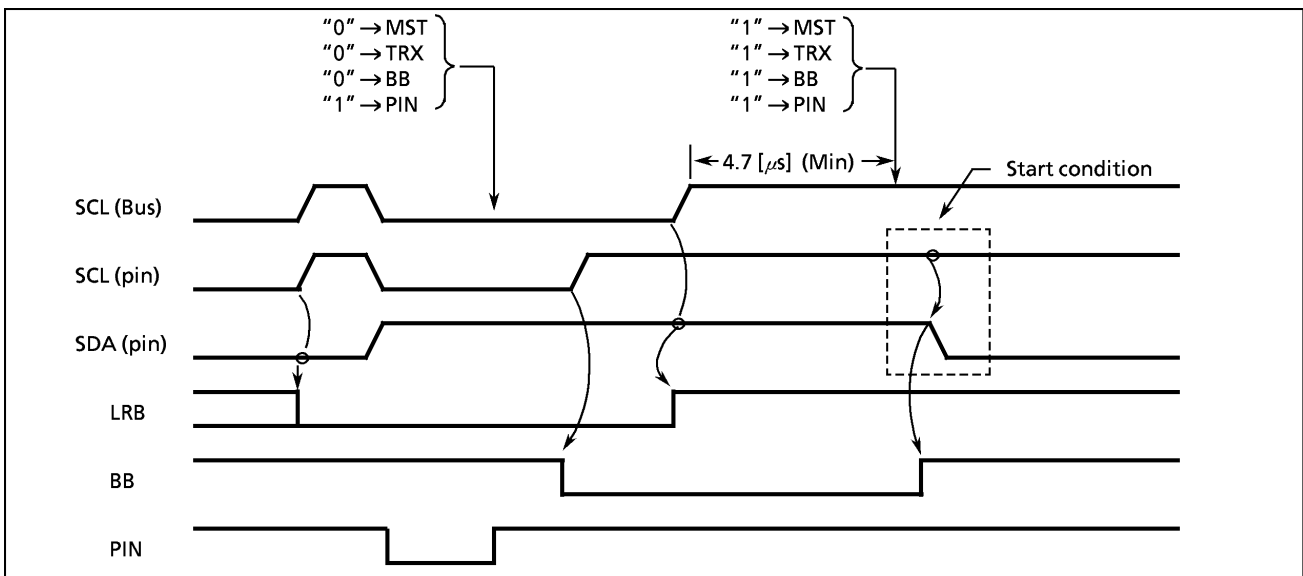


Figure 2-41. Timing Diagram When Restarting

2.8.6 Clocked-Synchronous 8-Bit SIO Mode Control

The following registers are used for control and operation status monitoring when using the serial bus interface (SBI-ver.A) in the clocked-synchronous 8-bit SIO mode.

Serial Bus Interface Control Register 1

SBICR1 (0020_H)

	7	6	5	4	3	2	1	0	
	SIOS	SIO INH	SIOM	"1"	SCK				(Initial value : 0000 0000)

SIOS	Indicate transfer start/stop	0 : Stop 1 : Start												
SIOINH	Continue/abort transfer	0 : Continue transfer 1 : Abort transfer (automatically cleared after abort)												
SIOM	Transfer mode select	00 : 8-bit transmit mode 01 : reserved 10 : 8-bit transmit/receive mode 11 : 8-bit receive mode	Write only											
SCK	Serial clock select	<table style="border: none;"> <tr> <td style="padding-right: 10px;">000 : $f_c/2^5$ (250 kHz)</td> <td rowspan="7" style="font-size: 2em; vertical-align: middle;">}</td> <td rowspan="7" style="vertical-align: middle;">at $f_c = 8$ MHz (Output on $\overline{\text{SCK}}$ pin)</td> </tr> <tr> <td>001 : $f_c/2^6$ (125 kHz)</td> </tr> <tr> <td>010 : $f_c/2^7$ (62.5 kHz)</td> </tr> <tr> <td>011 : $f_c/2^8$ (31.25 kHz)</td> </tr> <tr> <td>100 : $f_c/2^9$ (15.62 kHz)</td> </tr> <tr> <td>101 : $f_c/2^{10}$ (7.81 kHz)</td> </tr> <tr> <td>110 : $f_c/2^{11}$ (3.90 kHz)</td> </tr> <tr> <td colspan="2">111 : External clock (input from $\overline{\text{SCK}}$ pin)</td> <td></td> </tr> </table>		000 : $f_c/2^5$ (250 kHz)	}	at $f_c = 8$ MHz (Output on $\overline{\text{SCK}}$ pin)	001 : $f_c/2^6$ (125 kHz)	010 : $f_c/2^7$ (62.5 kHz)	011 : $f_c/2^8$ (31.25 kHz)	100 : $f_c/2^9$ (15.62 kHz)	101 : $f_c/2^{10}$ (7.81 kHz)	110 : $f_c/2^{11}$ (3.90 kHz)	111 : External clock (input from $\overline{\text{SCK}}$ pin)	
000 : $f_c/2^5$ (250 kHz)	}	at $f_c = 8$ MHz (Output on $\overline{\text{SCK}}$ pin)												
001 : $f_c/2^6$ (125 kHz)														
010 : $f_c/2^7$ (62.5 kHz)														
011 : $f_c/2^8$ (31.25 kHz)														
100 : $f_c/2^9$ (15.62 kHz)														
101 : $f_c/2^{10}$ (7.81 kHz)														
110 : $f_c/2^{11}$ (3.90 kHz)														
111 : External clock (input from $\overline{\text{SCK}}$ pin)														

Note 1 : f_c ; high-frequency clock [Hz]
Note 2 : Clear the SIOS to "0" and set the SIOINH to "1" when setting the transfer mode and serial clock.
Note 3 : SBICR1 is a write-only register, which cannot access any of in read-modify-write instructions such as bit operate, etc.
Note 4 : Set bit 3 to "1" in SBICR1 when SBI is used as SIO mode.

Serial Bus Interface Data Buffer Register

SBIDBR (0021_H)

	7	6	5	4	3	2	1	0	
	[8-bit data field]								(Initial value : **** ***)

Read / Write

Note 1 : Cannot read the data which was written into SBIDBR, since a write data buffer and a read data buffer are independent in SBIDBR. Therefore, cannot access it any of in read-modify-write instructions such as bit operate, etc.
Note 2 : don't care

Serial Bus Interface Control Register 2

SBICR2 (0023_H)

	7	6	5	4	3	2	1	0	
	"0"	"0"	"0"	"1"	SBIM	"0"	"0"	(Initial value : **** 00**)	

SBIM	Serial bus interface operation mode selection	00 : Port mode (serial bus interface output disable) 01 : SIO mode 10 : I ² C bus mode 11 : reserved	Write only
------	---	--	------------

*Note 1 : * ; don't care*
Note 2 : Switch a mode to port after data transfer is complete.
Note 3 : Switch a mode to SIO mode after confirming that input signals via port are high level.
Note 4 : SBICR2 is a write-only register, which cannot access any of in read-modify-write instructions such as bit operate, etc.
Note 5 : Clear bits 7 to 5 in the SBICR2 to "0", and set bit 4 to "1".

Figure 2-42-1. Serial Bus Interface Control Register 1 / Serial Bus Interface Data Buffer Register / Serial Bus Interface Control Register 2 in SIO Mode

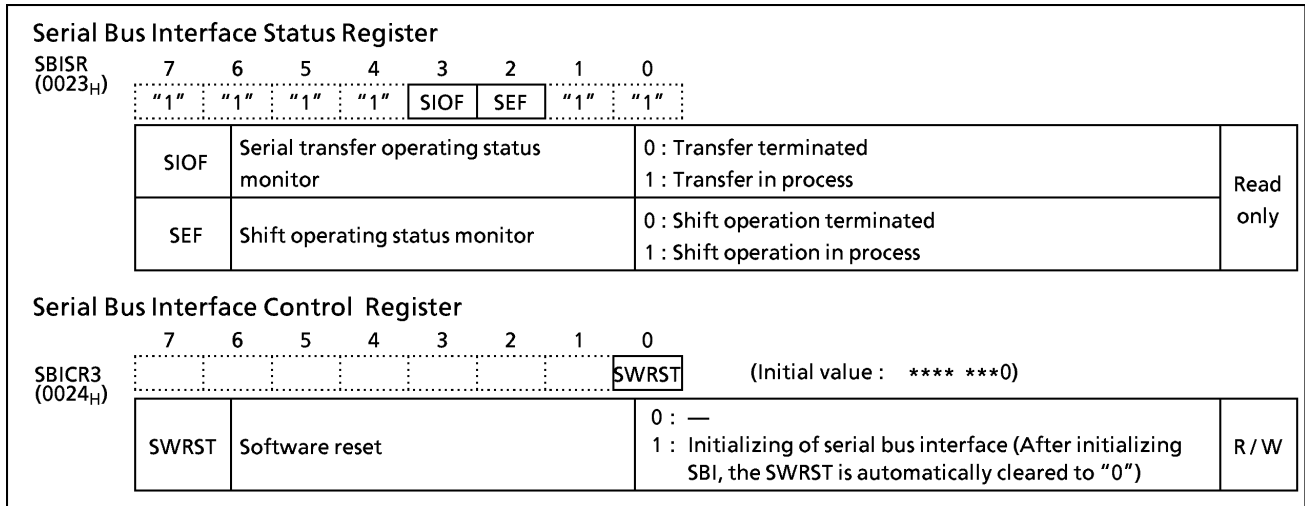


Figure 2-42-2. Serial Bus Interface Status Register in SIO Mode

(1) Serial clock

a. Clock source

The SCK (bit 2 to 0 in the SBICR1) is used to select the following functions.

① Internal clock

In an internal clock mode, any of seven frequencies can be selected. The serial clock is output to the outside on the \overline{SCK} pin. The \overline{SCK} pin becomes a high-level when data transfer starts. When writing (in the transmit mode) or reading (in the receive mode) data cannot follow the serial clock rate, an automatic-wait function is executed to stop the serial clock automatically and hold the next shift operation until reading or writing is complete.

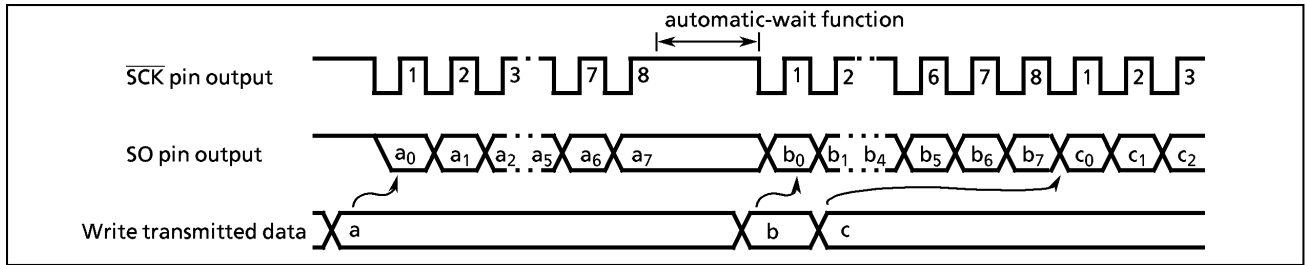


Figure 2-43. Automatic-wait Function

② External clock (SCK = "111")

An external clock supplied to the \overline{SCK} pin is used as the serial clock. In order to ensure shift operation, a pulse width of at least 4 machine cycles is required for both high-level and low-level in the serial clock. The maximum data transfer frequency is 250 kHz (when $f_c = 8$ MHz).

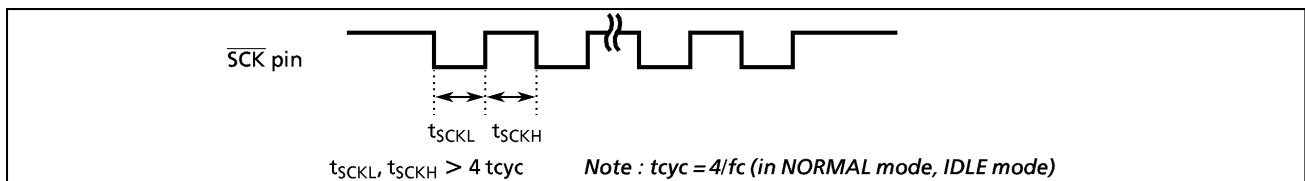


Figure 2-44. Maximum Data Transfer Frequency When External Clock Input

b. Shift edge

The leading edge is used to transmit data, and the trailing edge is used to receive data.

① Leading edge shift

Data is shifted on the leading edge of the serial clock (at a falling edge of the \overline{SCK} pin input/output).

② Trailing edge shift

Data is shifted on the trailing edge of the serial clock (at a rising edge of the \overline{SCK} pin input/output).

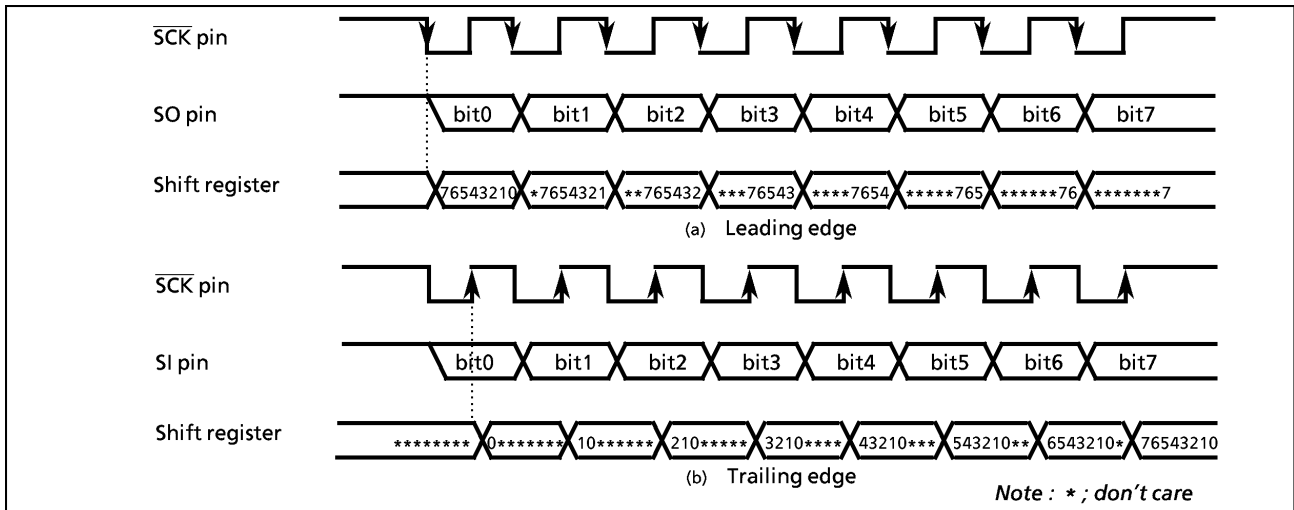


Figure 2-45. Shift Edge

(2) Transfer mode

The SIOM (bit 5 and 4 in the SBICR1) is used to select a transmit, receive, or transmit/receive mode.

a. 8-bit transmit mode

Set a control register to a transmit mode and write transmit data to the SBIDBR.

After the transmit data is written, set the SIOS to "1" to start data transfer. The transmitted data is transferred from the SBIDBR to the shift register and output to the SO pin in synchronous with the serial clock, starting from the least significant bit (LSB). When the transmit data is transferred to the shift register, the SBIDBR becomes empty. The INTSBI (buffer empty) interrupt request is generated to request new data.

When the internal clock is used, the serial clock will stop and automatic-wait function will be initiated if new data is not loaded to the data buffer register after the specified 8-bit data is transmitted. When new transmit data is written, automatic-wait function is canceled.

When the external clock is used, data should be written to the SBIDBR before new data is shifted. The transfer speed is determined by the maximum delay time between the time when an interrupt request is generated and the time when data is written to the SBIDBR by the interrupt service program.

When the transmit is started, after the SIOF goes "1" output from the SO pin holds final bit of the last data until falling edge of the \overline{SCK} .

Transmitting data is ended by clearing the SIOS to "0" by the buffer empty interrupt service program or setting the SIOINH to "1". When the SIOS is cleared, the transmitted mode ends when all data is output. In order to confirm if data is surely transmitted by the program, set the SIOF (bit 3 in the SBISR) to be sensed. The SIOF is cleared to "0" when transmitting is complete. When the SIOINH is set, transmitting data stops. The SIOF turns "0".

When the external clock is used, it is also necessary to clear the SIOS to "0" before new data is shifted; otherwise, dummy data is transmitted and operation ends.

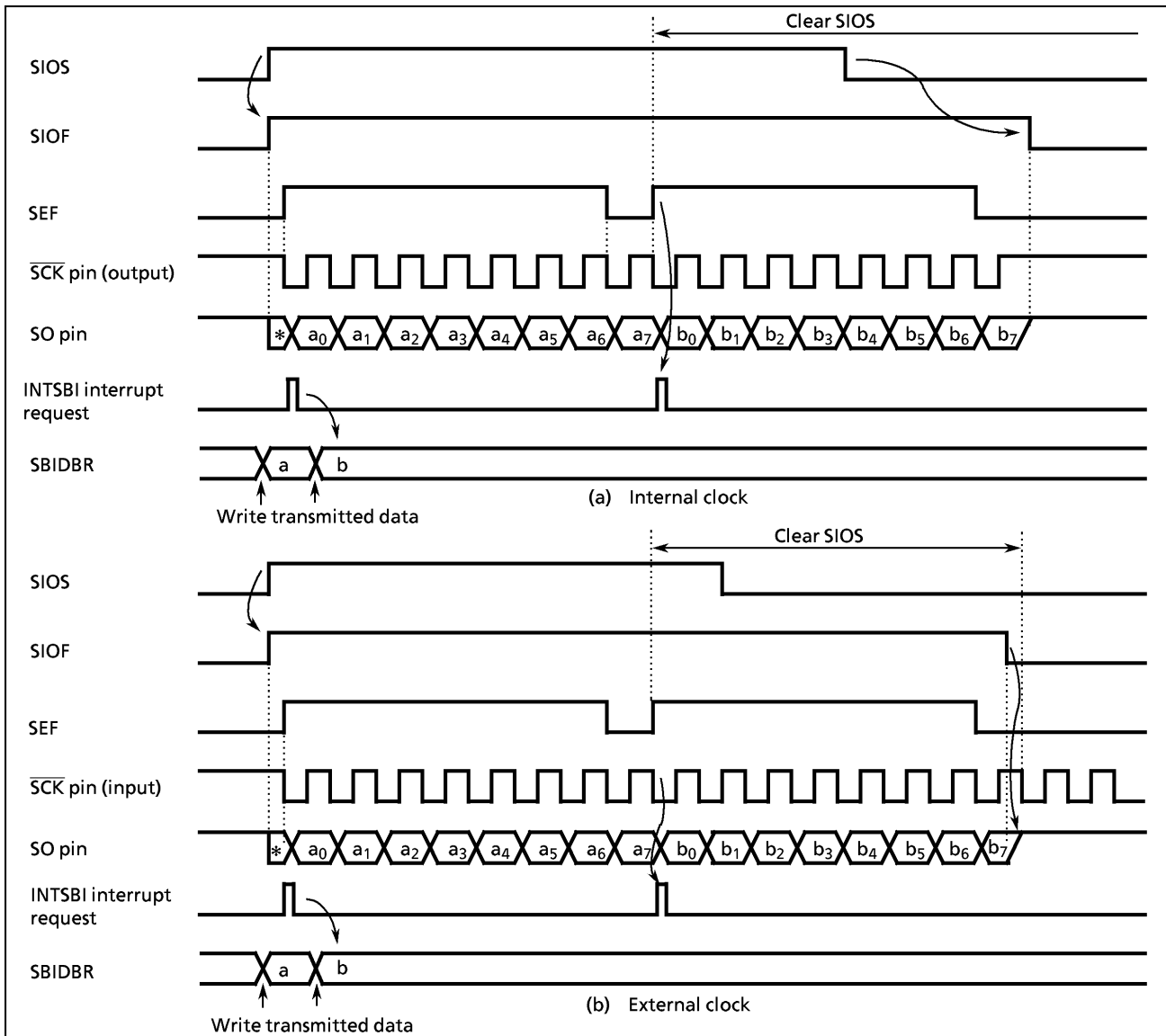


Figure 2-46. Transfer Mode

Example : Program to stop transmitting data (when external clock is used)

```

STEST1 : TEST    (SBISR). SEF          ; If SEF = 1 then loop
          JRS     F , STEST1
STEST2 : TEST    (P5). 3              ; If SCK = 0 then loop
          JRS     T , STEST2
          LD     (SBICR1), 00000111B ; SIOS ← 0
    
```

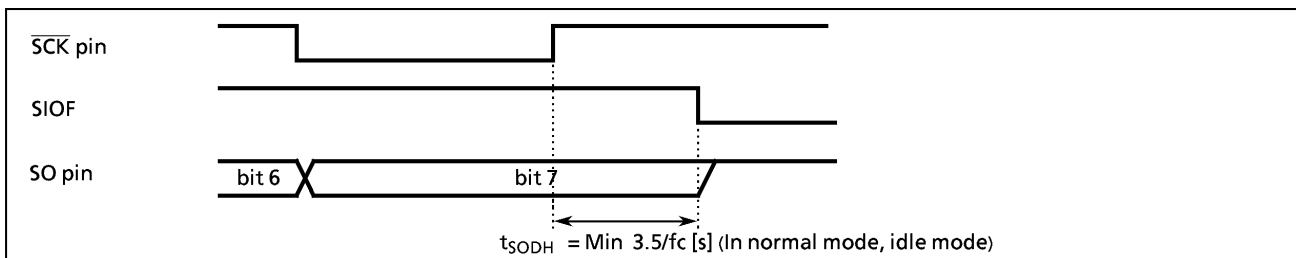


Figure 2-47. Transmitted Data Hold Time at End of Transmit

b.8-bit receive mode

Set the control register to receive mode and the SIOS to "1" for switching to receive mode. Data is received from the SI pin to the shift register in synchronous with the serial clock, starting from the least significant bit (LSB). When the 8-bit data is received, the data is transferred from the shift register to the SBIDBR. The INTSBI (buffer full) interrupt request is generated to request of reading the received data. The data is then read from the SBIDBR by the interrupt service program.

When the internal clock is used, the serial clock will stop and automatic-wait function will be initiated until the received data is read from the SBIDBR.

When the external clock is used, since shift operation is synchronized with the clock pulse provided externally, the received data should be read from the SBIDBR before next serial clock is input. If the received data is not read, further data to be received is canceled. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when received data is read.

Receiving data is ended by clearing the SIOS to "0" by the buffer full interrupt service program or setting the SIOINH to "1". When the SIOS is cleared, received data is transferred to the SBIDBR in complete blocks. The received mode ends when the transfer is complete. In order to confirm if data is surely received by the program, set the SIOF (bit 3 in the SBIDBR) to be sensed. The SIOF is cleared to "0" when receiving is complete. After confirming that receiving has ended, the last data is read. When the SIOINH is set, receiving data stops. The SIOF turns "0" (the received data becomes invalid, therefore no need to read it).

Note : When the transfer mode is switched, the SBIDBR contents are lost. In case that the mode needs to be switched, receiving data is concluded by clearing the SIOS to "0", read the last data, and then switch the mode.

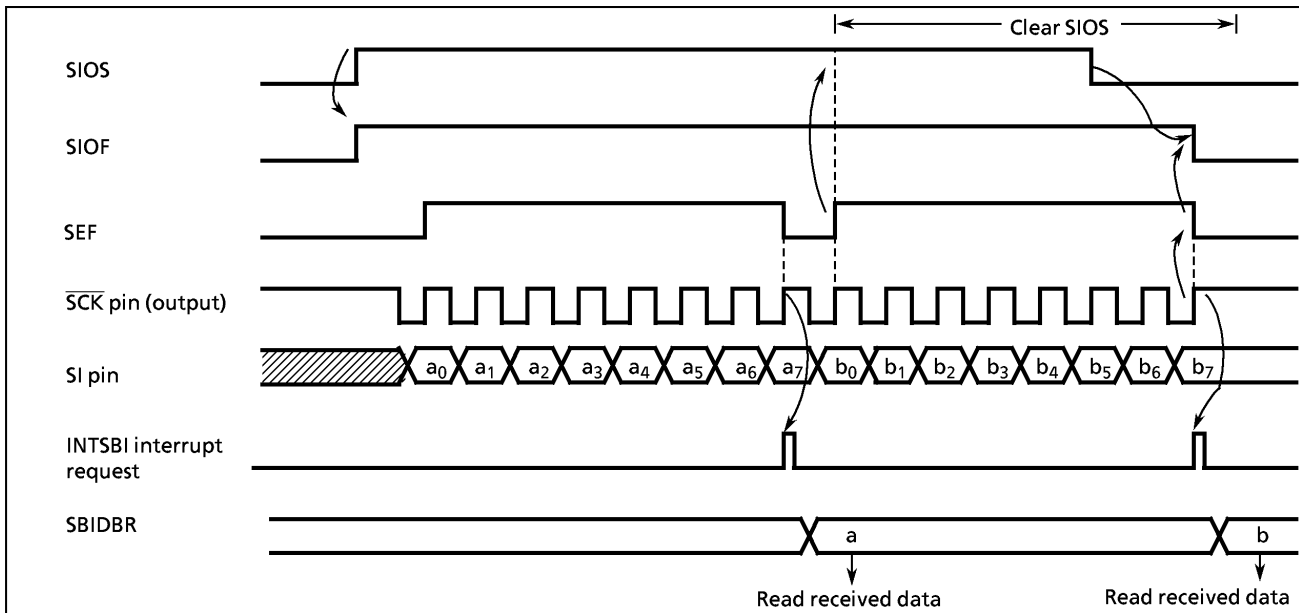


Figure 2-48. Receive Mode (Example : Internal Clock)

c. 8-bit transmit / receive mode

Set a control register to a transmit / receive mode and write data to the SBIDBR. After the data is written, set the SIOS to "1" to start transmitting / receiving. When transmitting, the data is output from the SO pin on the leading edges in synchronous with the serial clock, starting from the least significant bit (LSB). When receiving, the data is input to the SI pin on the trailing edges of the serial clock. 8-bit data is transferred from the shift register to the SBIDBR, and the INTSBI interrupt request occurs. The interrupt service program reads the received data from the data buffer register and writes data to be transmitted. The SBIDBR is used for both transmitting and receiving. Transmitted data should always be written after received data is read.

When the internal clock is used, automatic-wait function is initiated until received data is read and next data is written.

When the external clock is used, since the shift operation is synchronized with the external clock, received data is read and transmitted data is written before new shift operation is executed. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when received data is read and transmitted data is written.

When the transmit is started, after the SIOF goes "1" output from the SO pin holds final bit of the last data until falling edge of the \overline{SCK} .

Transmitting / receiving data is ended by clearing the SIOS to "0" by the INTSBI interrupt service program or setting the SIOINH to "1". When the SIOS is cleared, received data is transferred to the SBIDBR in complete blocks. The transmit / receive mode ends when the transfer is complete. In order to confirm if data is surely transmitted / received by the program, set the SIOF (bit3 in the SBISR) to be sensed. The SIOF becomes "0" after transmitting / receiving is complete. When the SIOINH is set, transmitting / receiving data stops. The SIOF turns "0".

Note : When the transfer mode is switched, the SBIDBR contents are lost. In case that the mode needs to be switched, conclude transmitting / receiving data by clearing the SIOS to "0", read the last data, and then switch the transfer mode.

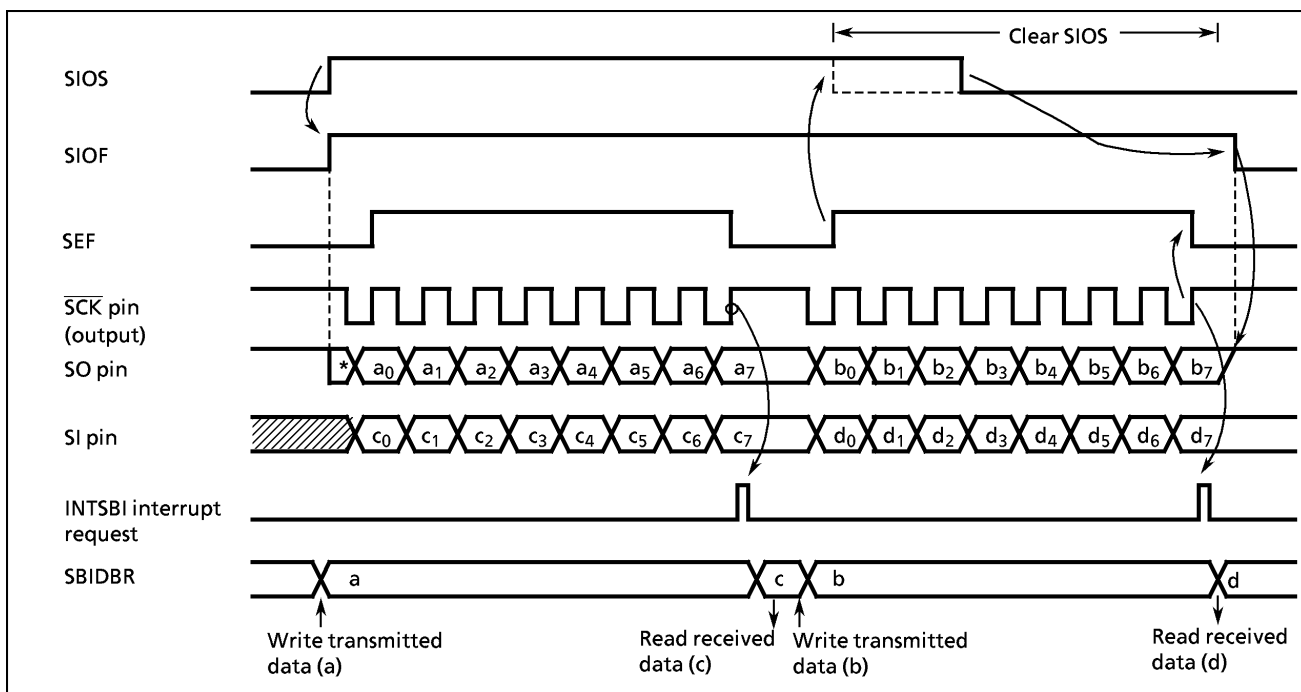


Figure 2-49. Transmit / Receive Mode (Example : Internal Clock)

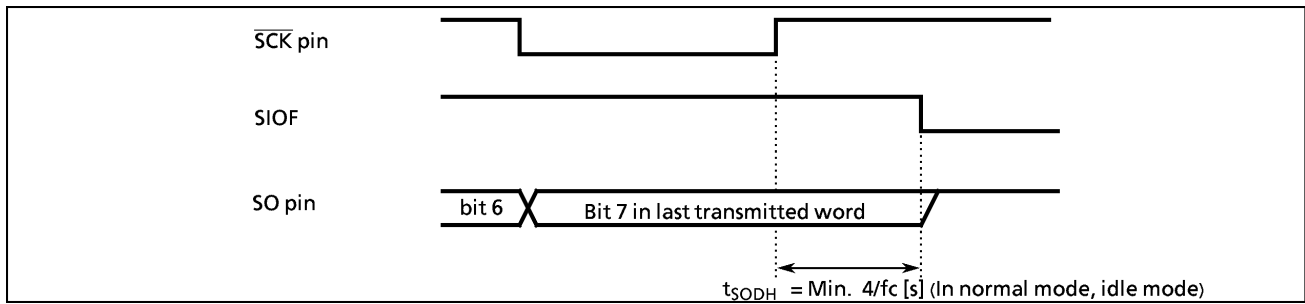


Figure 2-50. Transmitted Data Hold Time at End of Transmit / Receive

(3) Software reset (Bit 0 of SBICR3)

Software reset function can be used for initializing just serial bus interface when Serial bus interface is rocked by external noise, etc.

The SWRST (bit 0 of SBICR3) is set to "1", internal reset signal pulse is generated and inputted into Serial bus interface circuit. All command registers and status register are initialized to the initial value. The SWRST is automatically cleared to "0" after initializing Serial bus interface circuit.

2.9 Remote Control Signal Processor / External Interrupt 3 Input Pin

The remote control signal waveform can be determined by inputting the remote control signal waveform from which the carrier wave was eliminated by the receive circuit. When the remote control signal processor / external interrupt 3pin is also used as the P30 port, set the P30 port output latch to 1. When it is not used as the remote control signal processor / external interrupt 3 input pin, it can be used for normal I/Os.

2.9.1 Configuration

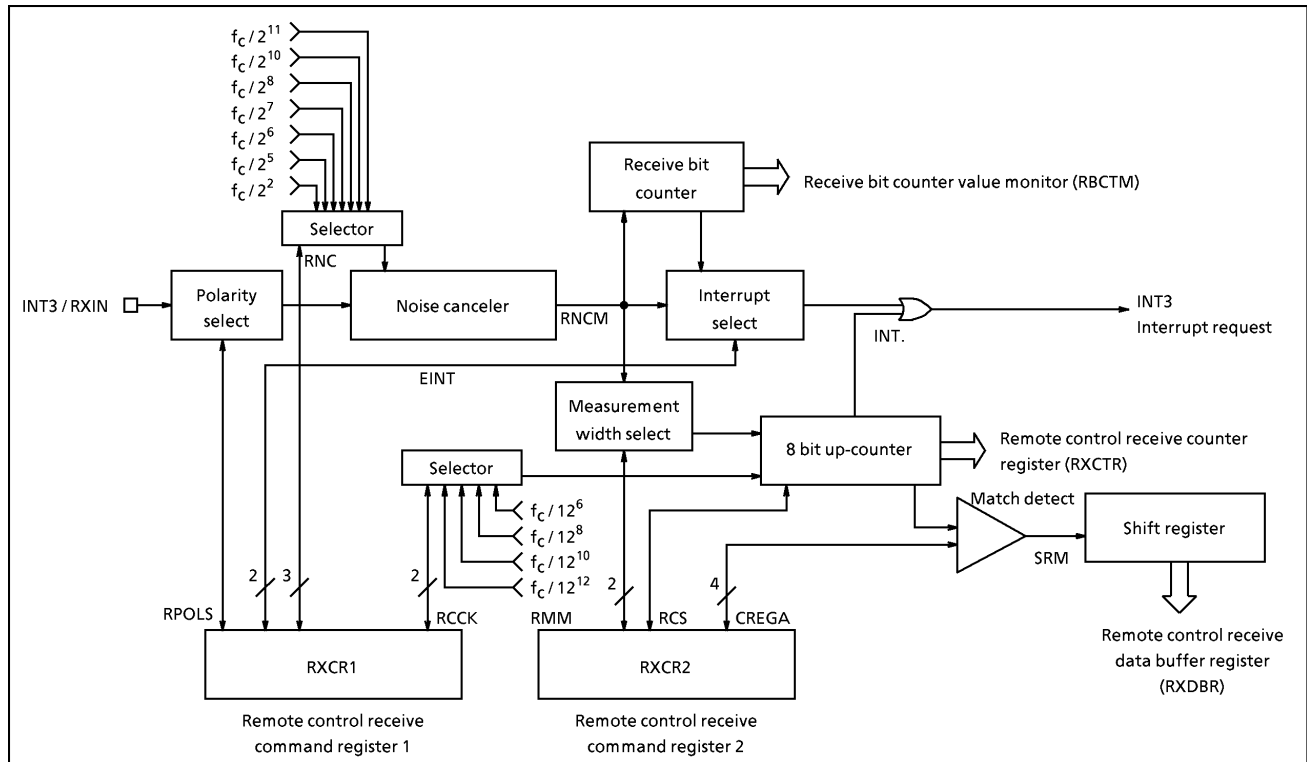


Figure 2-51. Remote Control Signal Processor

2.9.2 Remote Control Signal Processor Control

When the remote control signal processor is used, operating states are controlled and monitored by the following registers. Interrupt requests also use the remote control signal processor / external interrupt 3 input pin.

- Remote control receive control register 1 (RXCR1)
- Remote control receive control register 2 (RXCR2)
- Remote control receive counter register 1 (RXCTR)
- Remote control receive data buffer register 2 (RXDBR)
- Remote control receive status register (RXSR)

When this pin is used for the external interrupt 3 input, set EINT in RXCR1 to other than "11".

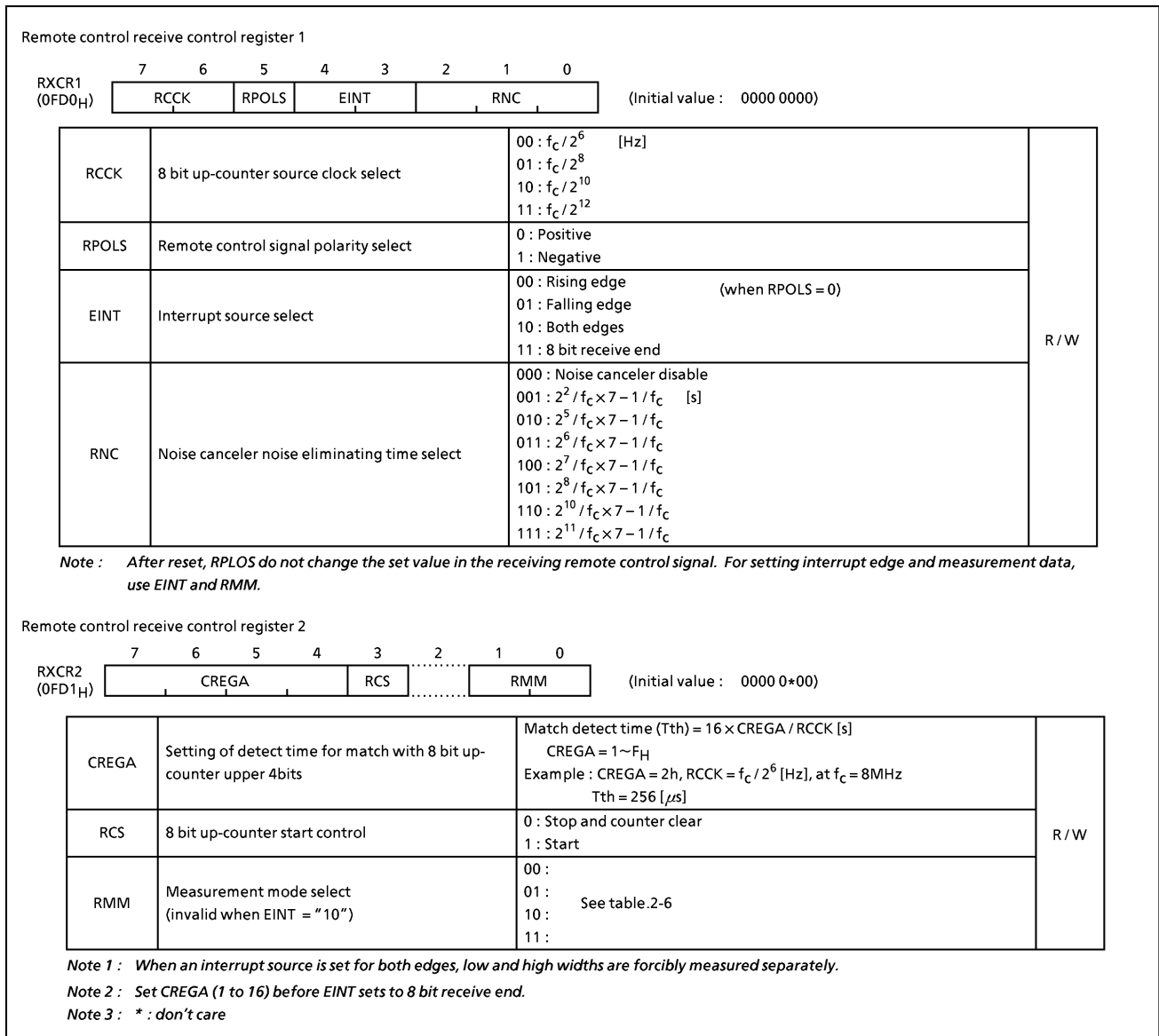


Figure 2-52. Remote Control Receive Control Register 1, 2

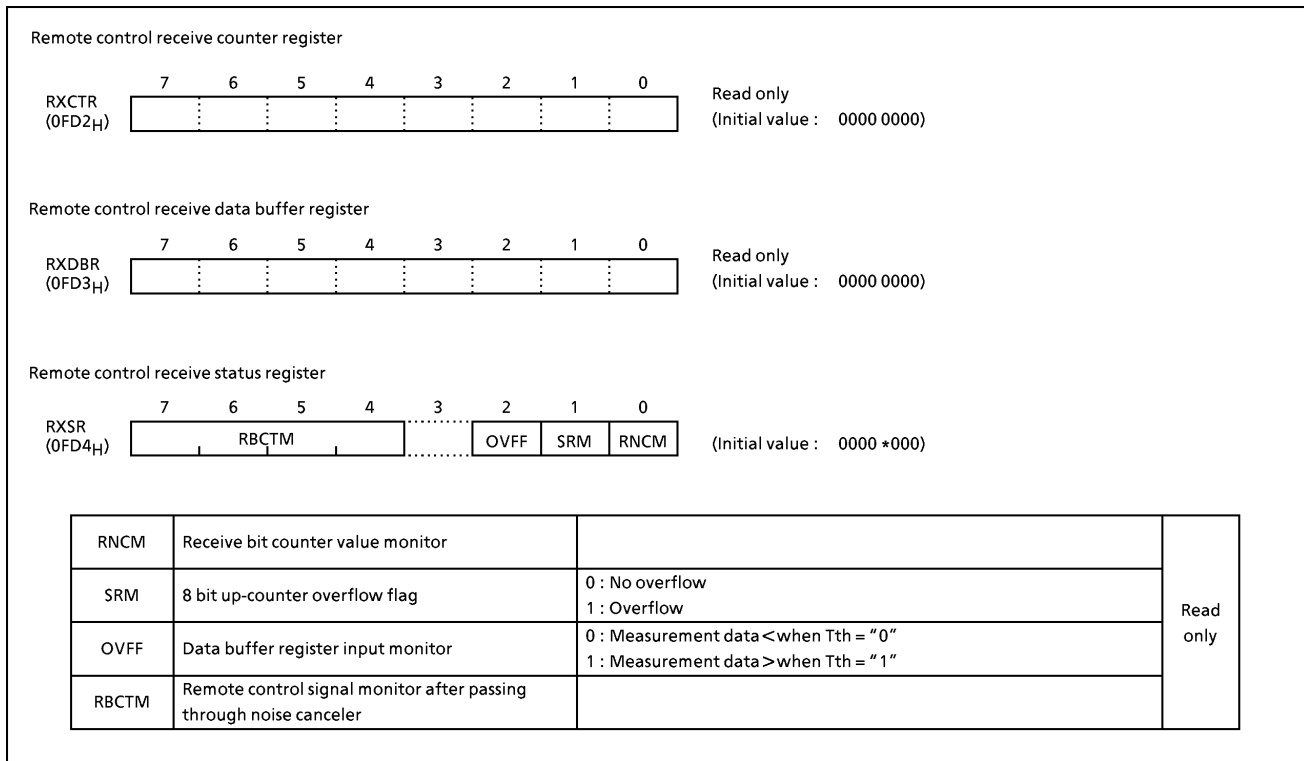


Figure 2-53. Remote Control Receive Counter Register, Data Buffer Register, Status Register

Table 2-6. Combination of Interrupt Source and Measurement Mode

RPOLS	EINT	RMM	Interrupt Source	Measurement Mode
0	00	00		
		10		
		11		
	01	01		
		10		
		11		
	10	—		
	11	00	10	Receive end
1	00	00		
		10		
		11		
	01	01		
		10		
		11		
	10	—		
	11	00	10	Receive end

2.9.3 Noise Elimination Time Setting

The remote control receive circuit has a noise canceler. By setting RNC in RXCR1, input signals shorter than the fixed time can be eliminated as noise.

Table 2-7. Noise Elimination Time Setting

RNC	Minimum Signal Pulse Width	AT fc = 8 MHz	Maximum Noise Width to be Eliminated	AT fc = 8 MHz
000	—	—	—	—
001	$(2^5 + 5) / fc$ [s]	4.63 [μs]	$(2^2 \times 7 - 1) / fc$ [s]	3.38 [μs]
010	$(2^8 + 5) / fc$	32.63	$(2^5 \times 7 - 1) / fc$	27.88
011	$(2^9 + 5) / fc$	64.63	$(2^6 \times 7 - 1) / fc$	55.88
100	$(2^{10} + 5) / fc$	128.63	$(2^7 \times 7 - 1) / fc$	111.88
101	$(2^{11} + 5) / fc$	256.63	$(2^8 \times 7 - 1) / fc$	223.88
110	$(2^{13} + 5) / fc$	1.025 [ms]	$(2^{10} \times 7 - 1) / fc$	895.88
111	$(2^{14} + 5) / fc$	2.049	$(2^{11} \times 7 - 1) / fc$	1.792 [ms]

2.9.4 Operation

- (1) interrupts at rising, falling, or both edges, and measurement modes

First set EINT and RMM. Next, write "1" in RCS ; the 8 bit up-counter is counted up by the internal clock. After measurement, the 8 bit up-counter value is saved in RXCTR. Then, the 8 bit up-counter is cleared, an INT3 request is generated, and the 8 bit up-counter resumes counting.

If the 8 bit up-counter overflows (FFH) before measurement is completed, an INT3 request is generated and the overflow flag (OVFF) is set to "1". Then, the 8 bit up-counter is cleared. An overflow can be detected by reading OVFF by the interrupt processing. To restart the 8 bit up-counter, set RCS to "1". As setting RCS to "1" the OVFF is cleared to "0".

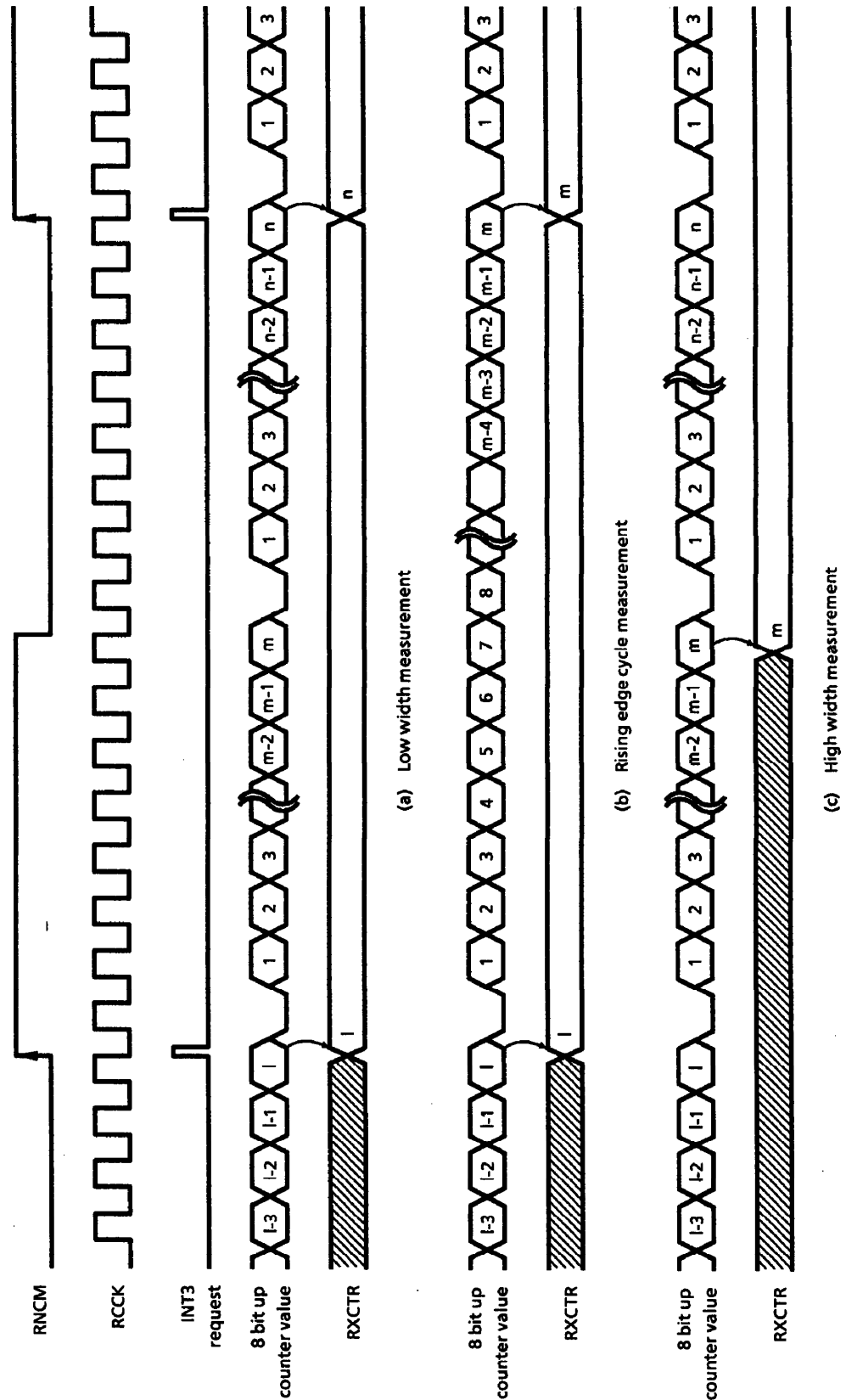


Figure 2-54. Rising Edge Interrupt Timing Chart (RPOLS = 0)

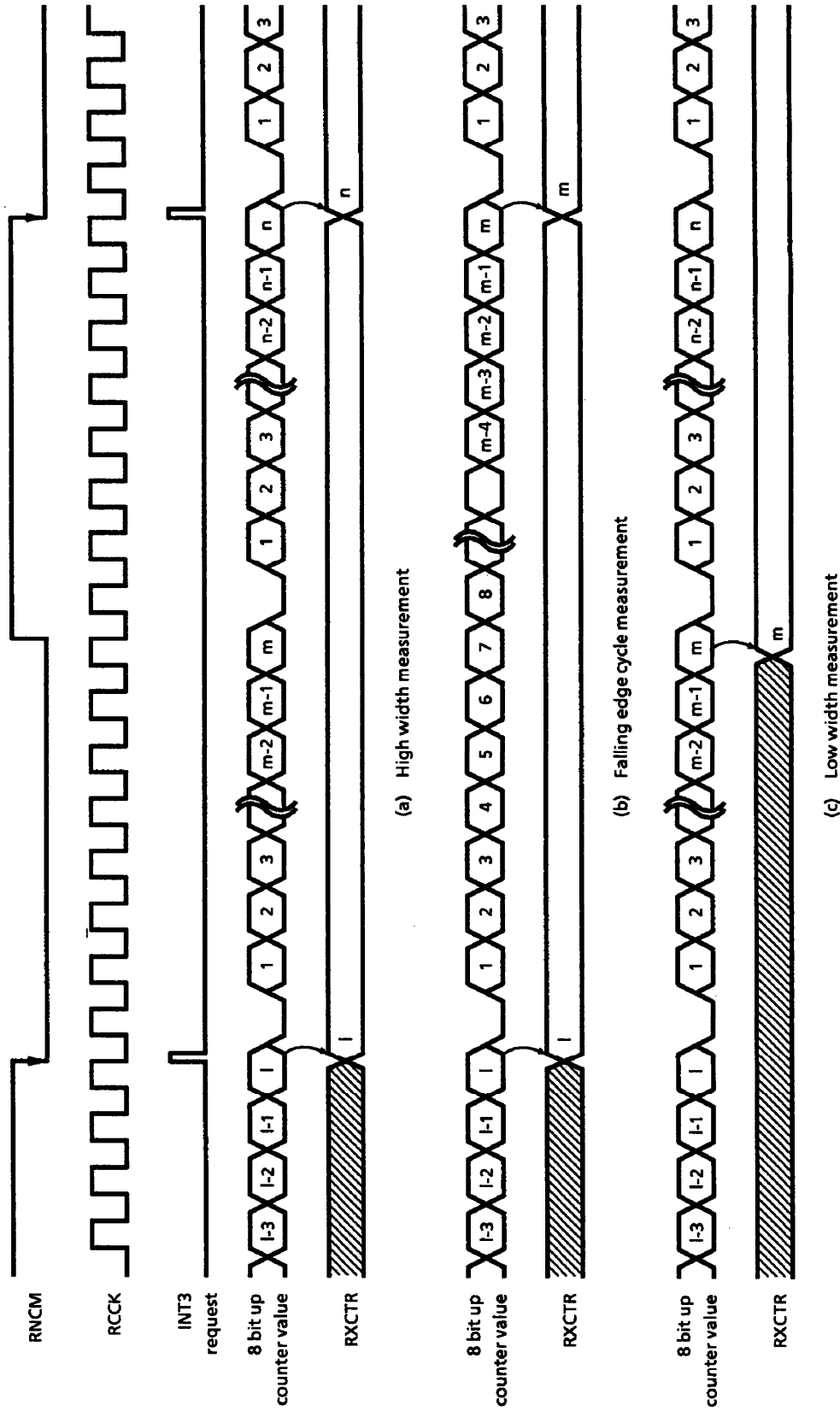
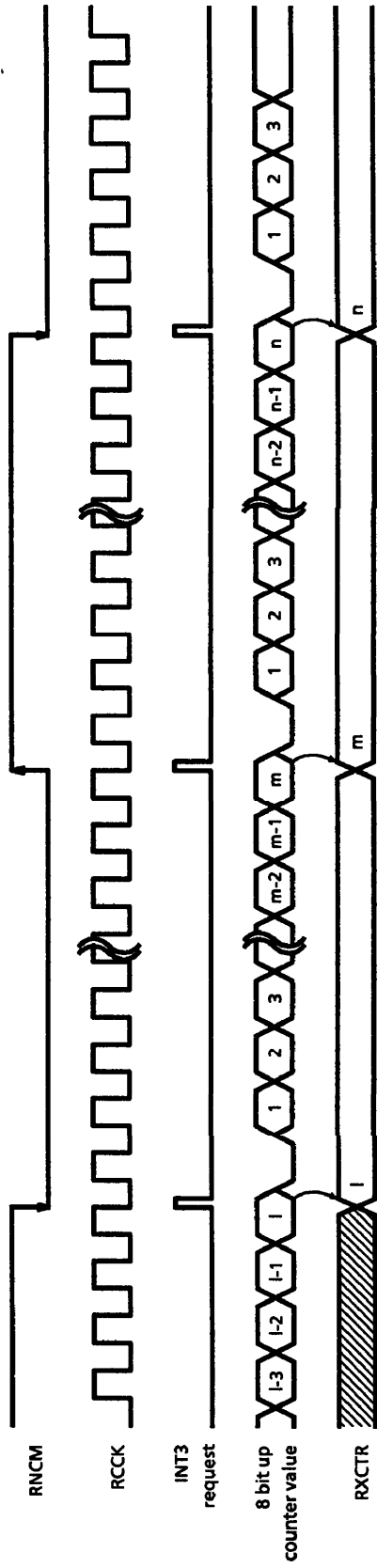


Figure 2-55. Falling Edge Interrupt Timing Chart (RPOLS = 0)



(a) High and low width measurement

Figure 2-56. Both Edge Interrupt Timing Chart

(2) 8 bit receive end interrupts and measurement modes

By determining one-cycle remote control signal as one-bit data set to "0" or one-pulse width remote control signal as one-bit data set to "1", an INT3 request is generated after 8 bit data is received. When "0" is determined, this means the upper four bits in the 8 bit up-counter have not reached the CREGA value. When "1" is determined, this means the upper four bits in the 8 bit up-counter have reached or exceeded the CREGA value. The 8 bit up-counter value is saved in RXCTR after one bit is determined. The determined data are saved, bit by bit, in RXDBR at the rising edge of the remote control signal (when RPLOS = 1, falling edge). The number of bits saved in RXDBR is counted by the receive bit counter and saved in RBCTM. RBCTM is set to "0001B" at the rising edge of the input (when RPOLS = 1, falling edge) after the INT3 request is generated.

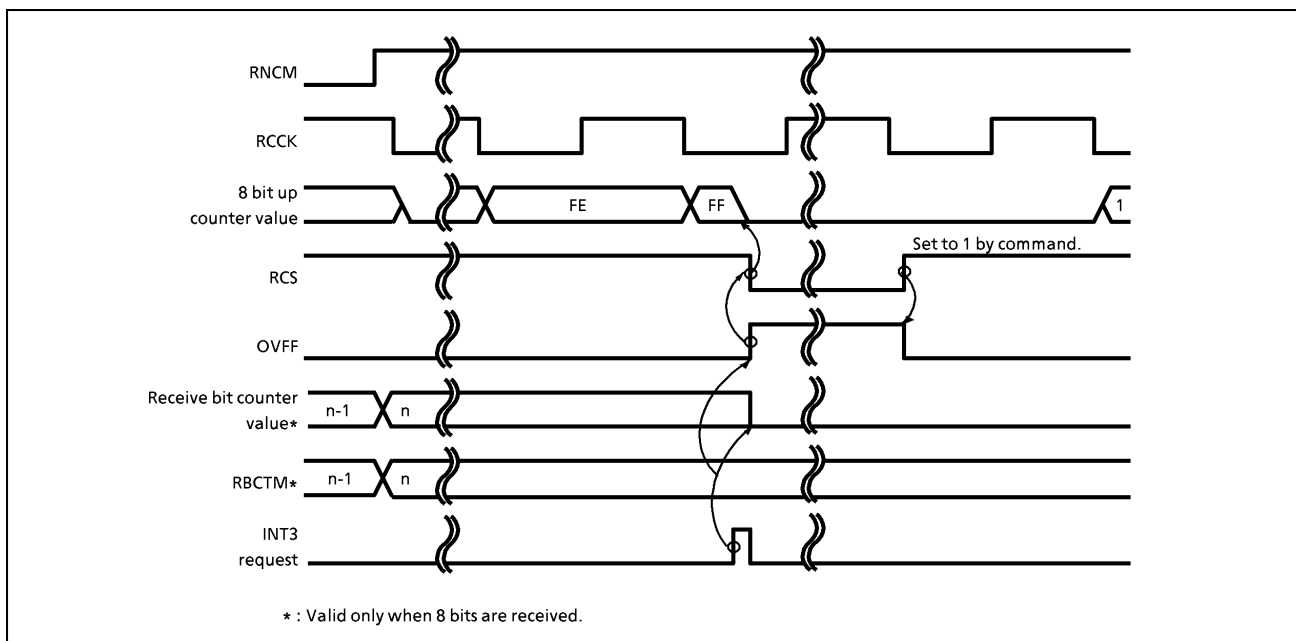
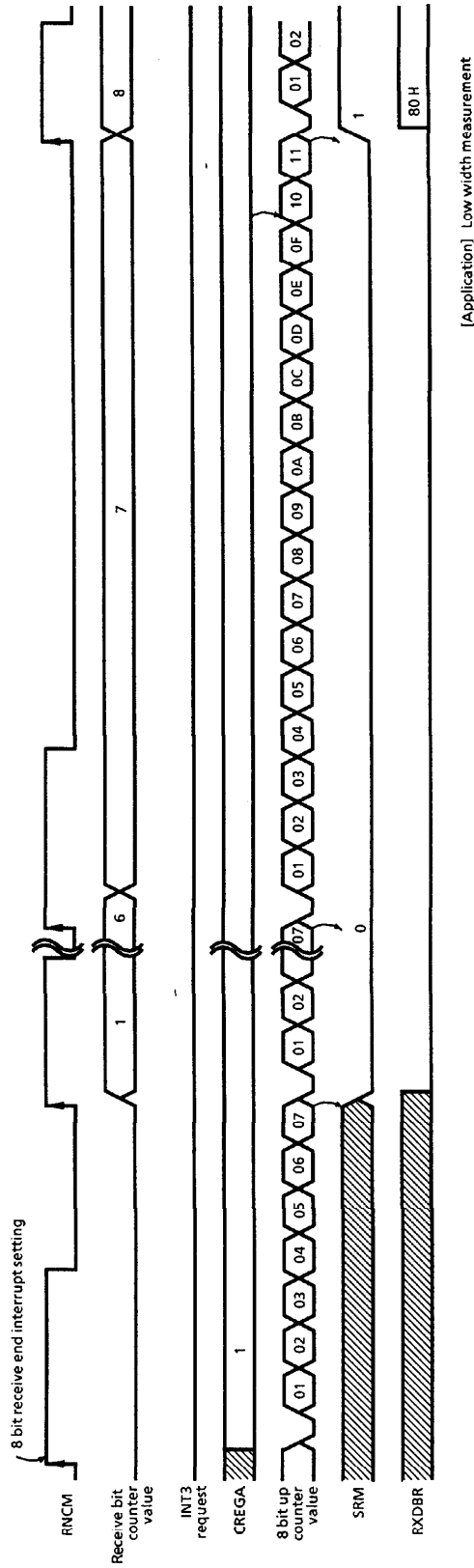


Figure 2-57. Overflow Interrupt Timing Chart



(a) Rising edge cycle measurement

Figure 2-58. 8 bit Receive End Interrupt Timing Chart (RPOLS = 0)

Table 2-8. Count Clock for Remote Control Decision Circuit

Count Clock (RCCK)	AT fc = 8 MHz	
	Resolution	Maximum Setting Time
$fc / 2^{12}$ [Hz]	512 [μ s]	131.072 [ms]
$fc / 2^{10}$	128 [μ s]	32.768 [ms]
$fc / 2^8$	32 [μ s]	8.192 [ms]
$fc / 2^6$	8 [μ s]	2.048 [ms]

2.10 8 bit A / D Converter (ADC)

The A8700CH / CK / CM / CP / CS each have an 8-channel multiplexed-input 8 bit successive approximate type A / D converter with sample and hold.

2.10.1 Configuration

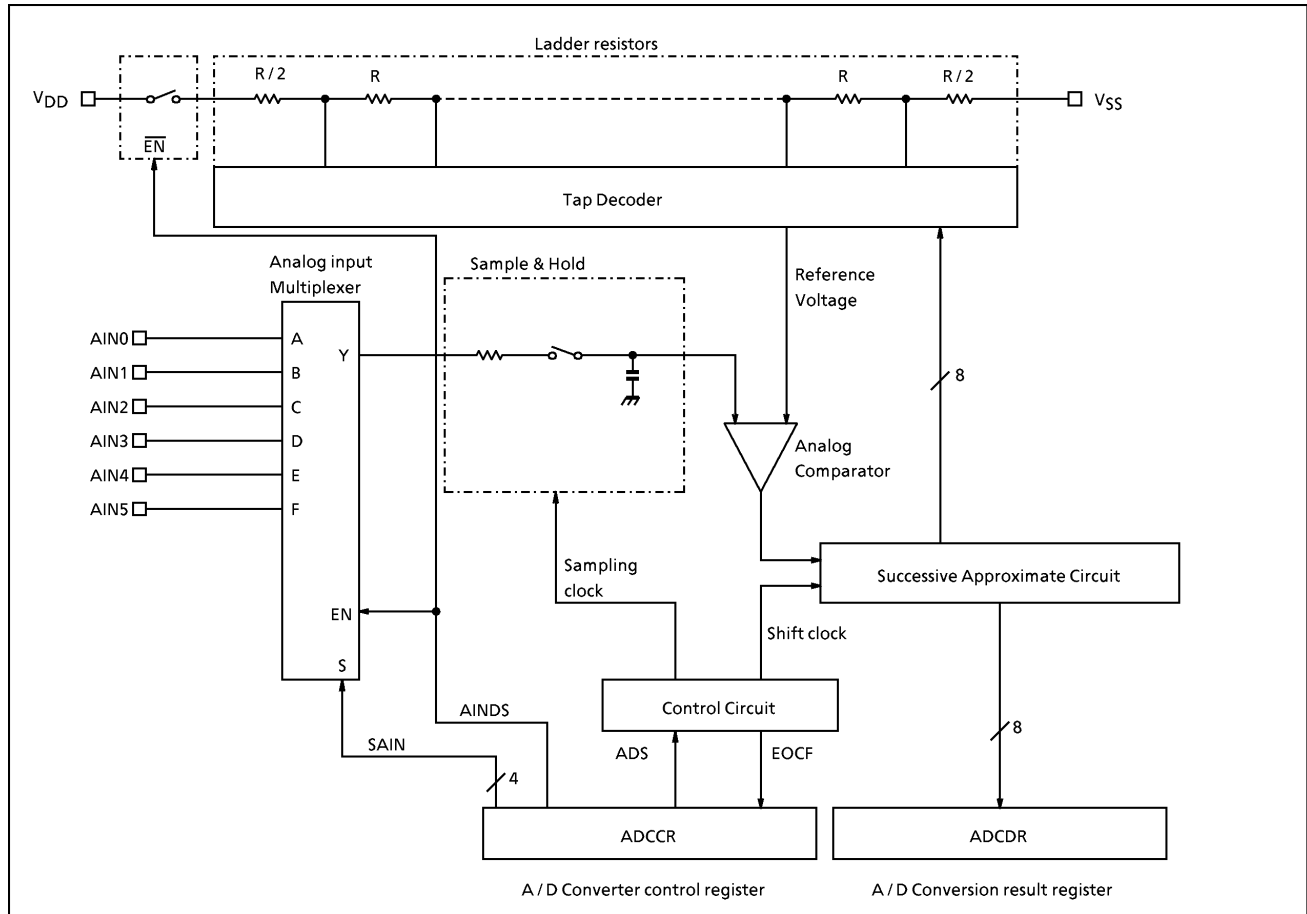


Figure 2-59. A / D Converter

2.10.2 Control

The A / D converter is controlled by the A / D converter control register (ADCCR) and the P5 / P6 port registers (P5CR1 / P6CR). The status of A / D converter operation is obtained by reading the EOCF of the ADCCR ; the result of conversion is obtained by reading the A / D conversion result register (ADCDR).

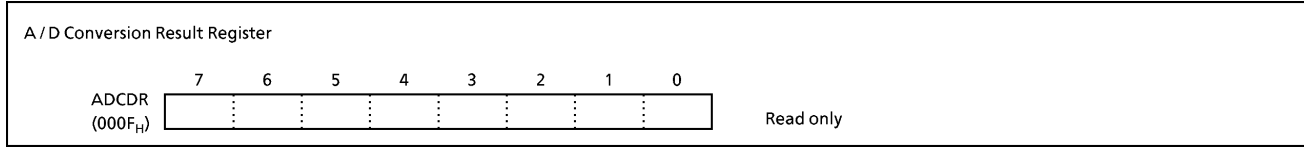


Figure 2-60. A / D Conversion Result Register

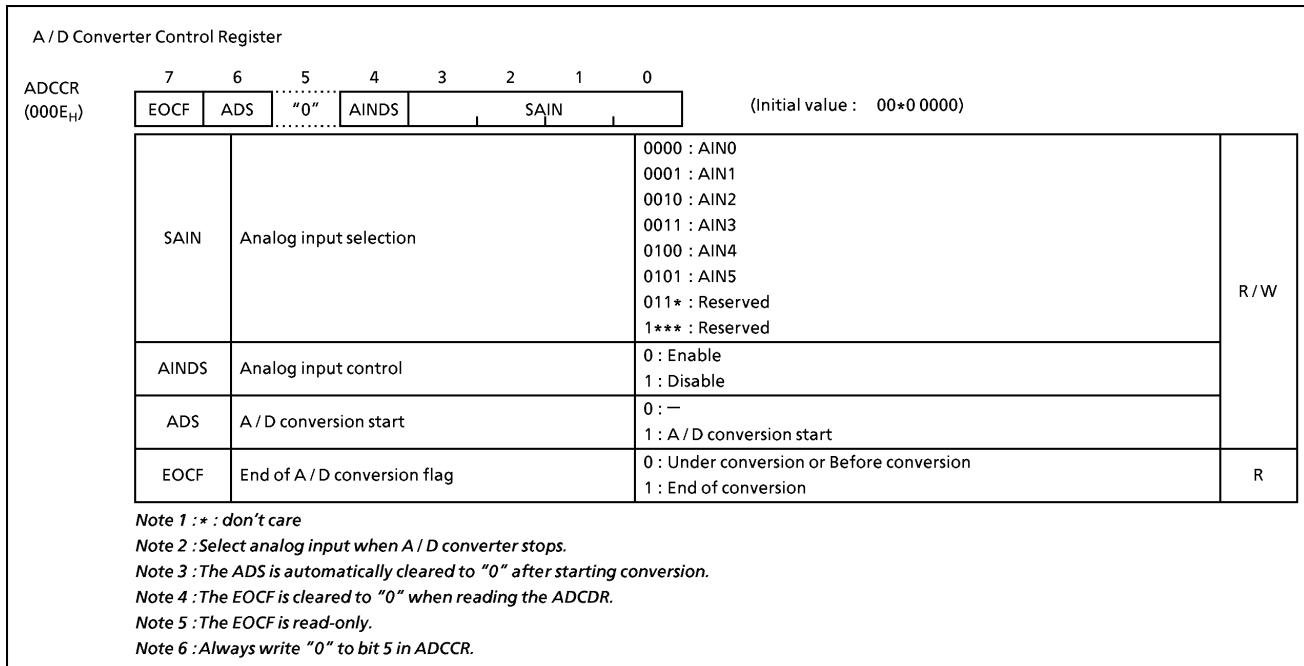


Figure 2-61. A / D Converter Control Register and A / D Conversion Result Register

2.10.3 A / D Converter Operation

The high analog reference voltage is applied to the V_{DD} pin and the low voltage to the V_{SS} pin. The reference voltages between V_{DD} and V_{SS} are split using a ladder resistor to correspond to bits. The analog input voltage is then converted to digital by comparing it against the reference voltages.

(1) A / D conversion

Prior to A / D conversion, one of the analog input channels (AIN5 to AIN0) is selected by SAIN (ADCCR bits 3 to 0). AINDS (ADCCR bit 4) is cleared to "0", and P6 input control (P6CR) clears to "0" the channel to be used for analog input.

A / D conversion starts when ADS (ADCCR bit 6) is set to "1". On completion of conversion, EOCF (ADCCR bit 7) is set to "1" to show the end of conversion. If ADS is set to "1" while A / D conversion is in progress, the converter is initialized and conversion 20 is started again from the beginning.

A / D conversion takes 46 machine cycles ($184 / f_c$ [s]), the analog input voltage being sampled 4 machine cycles after the instruction to start conversion.

Note 1: The pins not used for analog input can be used as normal I / O pins. However, to ensure accuracy, do not execute any output instructions for any of these pins while conversion is in progress.

Note 2: The sample-hold circuit has a built-in 12 pF (typ.) capacitor connected via a 5 kΩ (typ.) resistor. The capacitor must therefore be charged within the 4 machine cycles.

(2) Reading of A / D conversion result

After the end of conversion, read the conversion result from the ADCDR.

The EOCF is automatically cleared to "0" when reading the ADCDR.

During conversion indefinite data is read.

(3) A / D conversion in STOP mode

When the MCU places in the STOP mode during the A / D conversion, the conversion is terminated and the ADCDR contents become indefinite.

However, if the STOP mode is started after the end of conversion (EOCF = 1), the ADCDR contents are held.

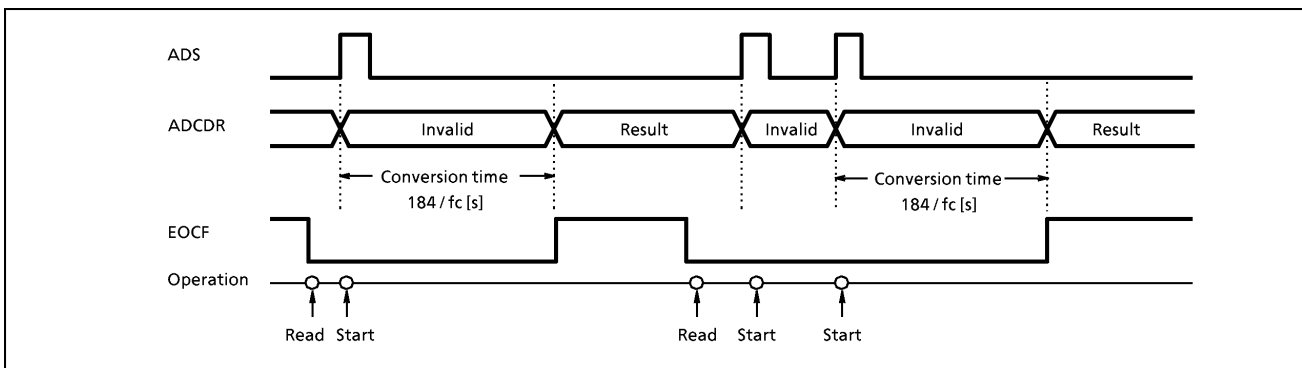


Figure 2-62. A / D Conversion Timing Chart

2.11 Pulse Width Modulation Circuit Output

A8700CH / CK / CM / CP / CS has 10 built-in pulse width modulation (PWM) channels. D / A converter output can easily be obtained by connecting an external low-pass filter.

PWM outputs are multiplexed with general purpose I / O ports as ; P40 (PWM0) to P47 (PWM7), P50 (PWM8), P51 (PWM9). When these ports are used PWM outputs, the corresponding bits of P4, P5 output latches should be set to "1", and be set to the output mode.

2.11.1 Configuration

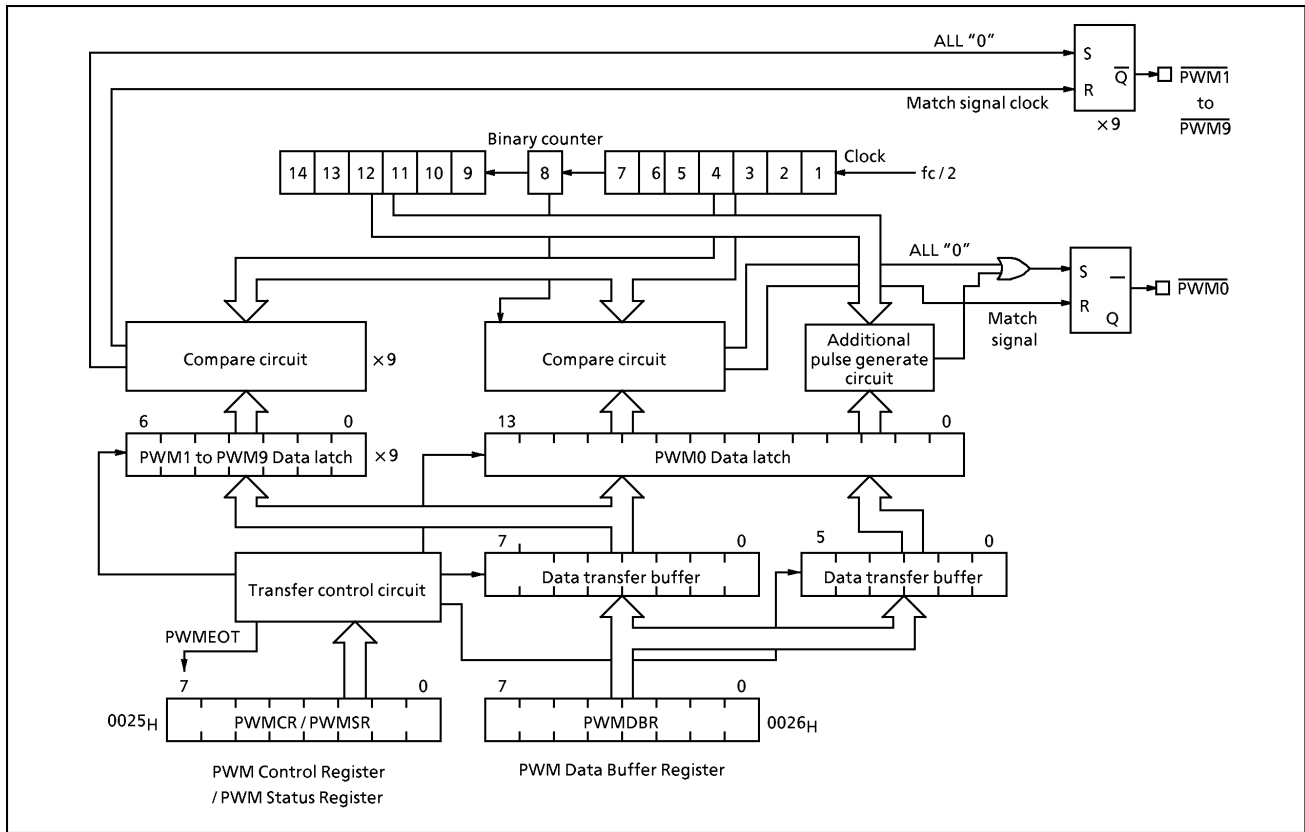


Figure 2-64. Pulse Width Modulation Circuit

2.11.2 PWM Output Wave Form

(1) $\overline{PWM0}$ output

This is 14 bit resolution PWM output and one period is $T_M = 2^{15} / f_c$ [s].

The 8 high-order bits of the PWM data latch control the pulse width of the pulse output with a period of T_S ($T_S = T_M / 64$), which is the sub-period of the $\overline{PWM0}$. When the 8 bit data are decimal n ($0 \leq n \leq 255$), this pulse width becomes $n \times t_0$, where $t_0 = 2 / f_c$.

The lower 6 bit of 14 bit data are used to control the generation of additional to wide pulse in each T_S period. When the 6 bit data are decimal m ($0 \leq m \leq 63$), the additional pulse is generated in each of m periods out of 64 periods contained in a T_M period. The relationship between the 6 bits data and the position of T_S period where the additional pulse is generated is shown in Table 2-9.

Table 2-9. Correspondence between 6 bits data and the additional pulse generated T_S period

Bit Position of 6 bits Data	Relative Position of T_S Where The Output Pulse is Generated. (No. 1 of T_S (i) is Listed)
Bit 0	32
Bit 1	16, 48
Bit 2	8, 24, 40, 56
Bit 3	4, 12, 20, 28, 36, 44, 52, 60
Bit 4	2, 6, 10, 14, 18, 22, 26, 30, ..., 58, 62
Bit 5	1, 3, 5, 7, 9, 11, 13, 15, 17, ..., 59, 61, 63

Note: When the corresponding bit is "1", it is output.

(2) $\overline{PWM1}$ to $\overline{PWM9}$ outputs

These are 7 bit resolution PWM outputs and one period is $T_N = 2^8 / f_c$ [s]. When the 7 bit data are decimal k ($0 \leq k \leq 127$), the pulse width becomes $k \times t_0$.

The wave form is illustrated in Figure 2-65.

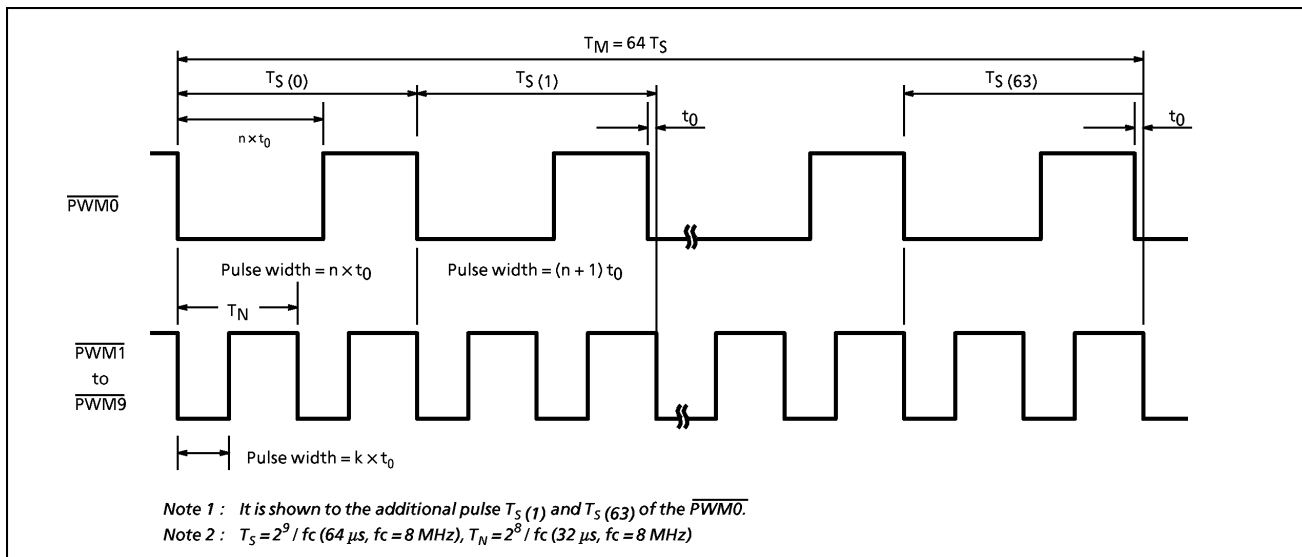


Figure 2-65. PWM Output Wave Form

2.11.3 Control

PWM output is controlled by PWM Control Register (PWMCr) and PWM Data Buffer Register (PWMDbR). The status of transfer PWM data from PWMDbR to PWM data latch is read by PWMEOT of PWM status register (PWMSr).

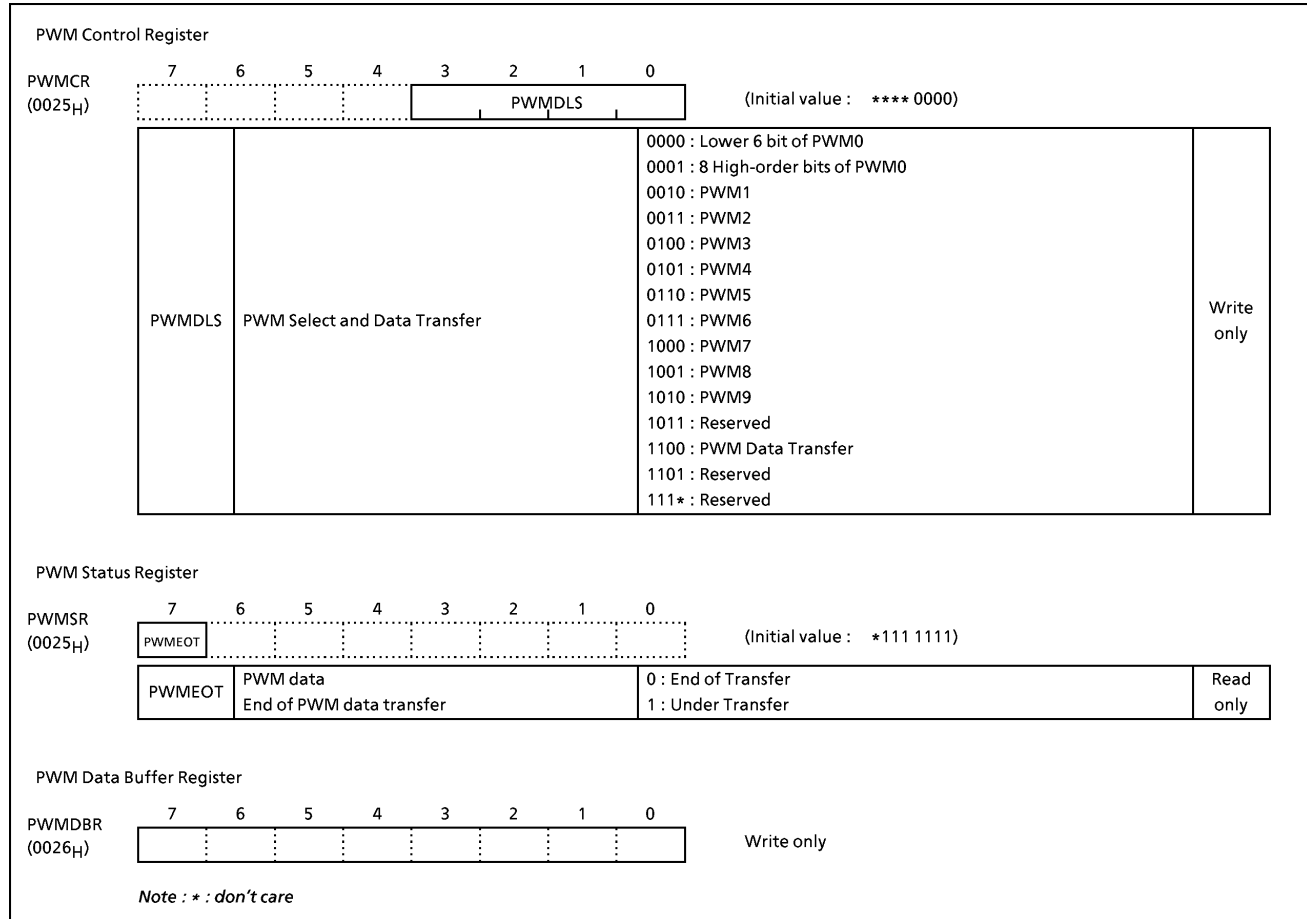


Figure 2-66. PWM Control Register / PWM Status Register / PWM Data Buffer Register

(1) Programming of PWM data

PWM output is controlled by PWM writing the output data to data latches. For the writing the output data are divided using the PWM Control Register (PWMCR).

1. Write the number of the data latch which the data are to be written to the PWMDLS.
2. Write PWM output data to the PWMDBR.
3. Write "0C_H" to the PWMCR.

*Note: When switching of the output data is completed, the end of PWM data transfer flag becomes "0", indicating that the next data can be written. Do not write PWM data when the end of PWM data is "1" because write errors can occur in this case.
When writing the output data to PWM0, write "0C_H" to the PWMDLS after writing of the 14 bits output data is completed.*

While the output data are being written to the data latch, the previously written data are being output. The maximum time from the point at which "0C_H" is written to the data latch until PWM output is switched is $2^{15} / f_c$ [s] (4.096 ms, at $f_c = 8$ MHz) for PWM0 output and $2^9 / f_c$ [s] (64 μ s, at $f_c = 8$ MHz) for PWM1 to PWM9 output.

Example : PWM0 pin outputs 32 μ s pulse width without the additional pulse.

PWM1 pin outputs 16 μ s pulse width.

PWM2 pin outputs 8 μ s pulse width.

Note : at $f_c = 8$ MHz

```

LD (PWMCR), 00H ; Select lower 6 bit of PWM0
LD (PWMDBR), 00H ; Without the additional pulse
LD (PWMCR), 01H ; Select 8 high-order bits of PWM0
LD (PWMDBR), 80H ; 32  $\mu$ s  $\div$  2 /  $f_c = 80_H$ 
LD (PWMCR), 0CH ; PWM Data Transfer
WAIT0 : TEST (PWMSR). 7 ; PWMEOT = 0?
JRS F, WAIT0
LD (PWMCR), 02H ; Slect PWM1
LD (PWMDBR), 40H ; 16  $\mu$ s  $\div$  2 /  $f_c = 40_H$ 
LD (PWMCR), 0CH ; PWM Data Transfer
WAIT1 : TEST (PWMSR). 7 ; PWMEOT = 0?
JRS F, WAIT1
LD (PWMCR), 03H ; Select PWM2
LD (PWMDBR), 20H ; 8  $\mu$ s  $\div$  2 /  $f_c = 20_H$ 
LD (PWMCR), 0CH ; PWM Data Transfer
WAIT2 : TEST (PWMSR). 7 ; PWMEOT = 0?
JRS F, WAIT2

```

2.12 On-Screen Display (OSD) Circuit

The A8700CH / CK / CM / CP / CS features a built-in on-screen display circuit used to display characters and symbols on the TV screen. There are 251 characters and any characters can be displayed in an area of 32 columns×8 lines. With an OSD interrupt, additional lines can be displayed. The functions of the OSD circuit meet the requirements of on-screen display functions of closed caption decoders based on FCC standards.

OSD circuit functions are as follows :

- ① Number of characters : 251
- ② Number of display characters : 256 (32 columns×8 lines). With OSD interrupt, nine or more lines can be displayed.
- ③ Character matrix : 8×9 dots (8×13 dots including space around character)
- ④ Character sizes : 3 (Specified by line)
- ⑤ Display colors : Character colors : 7 colors
(Specified character by character)
Fringe colors : 8 colors
Background colors : 8 colors
- ⑥ Fringing and smoothing functions (For large, middle, and small characters)
- ⑦ Display position : 128 horizontal steps and 256 vertical steps
- ⑧ Full-raster blanking function
- ⑨ Blinking function
- ⑩ Underline
- ⑪ Solid space
- ⑫ Slant function (Italics)
- ⑬ Window function

2.12.1 Configuration

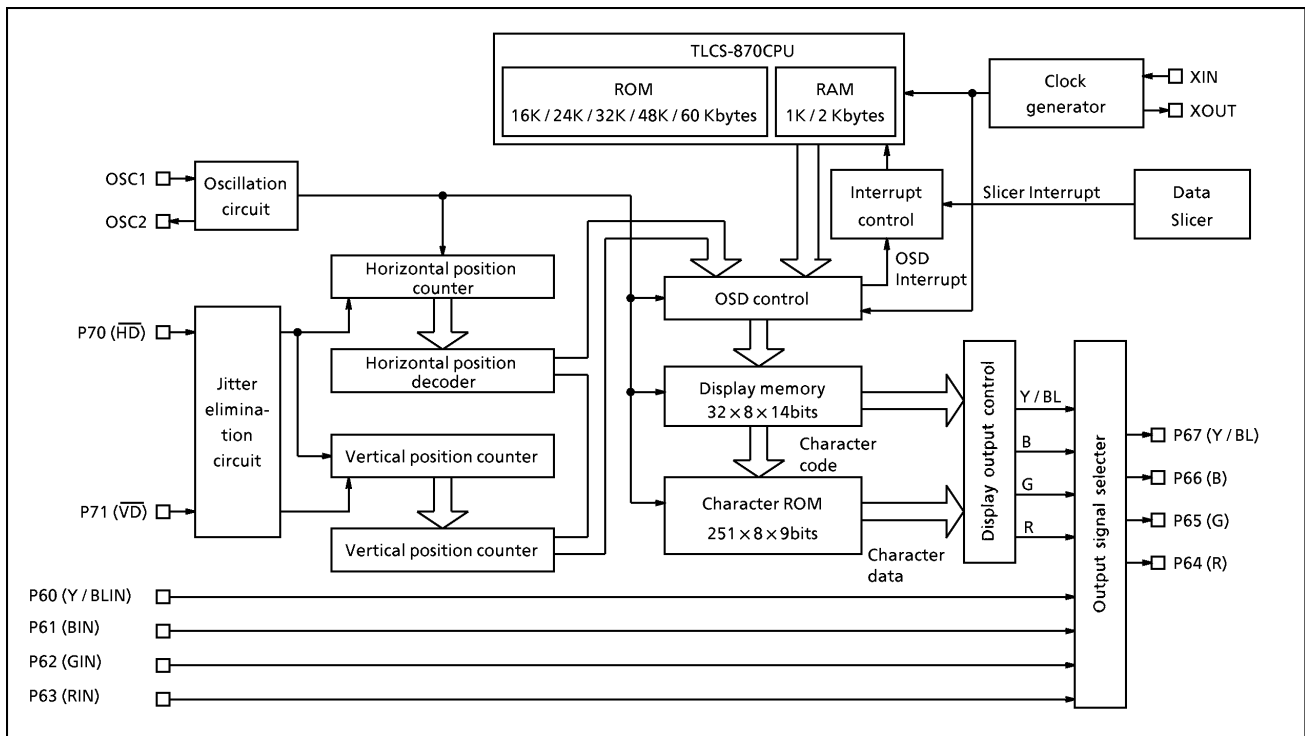


Figure 2-67. OSD Circuit

2.12.2 Memories for OSD

(1) Character ROM

The character ROM contains 251 character patterns. It is possible to design any patterns as below. The character ROM consists of 256 characters stored as 8 × 9 dots (character codes 00_H to FF_H except 01_H to 05_H). Each bit in the character ROM corresponds to one dot. When a bit in the character ROM is set to 1, the corresponding dot is displayed ; if set to 0, the dot is not displayed. The start address in the character ROM for each characters can be calculated as follows :

$$\text{Start address in character ROM} = \text{CRA}_H \times 10_H + 4000_H$$

Note: CRA : Character code (00_H to FF_H except 01_H to 05_H)

Character code 00_H is fixed as blank data and cannot be changed to represent a different character. Character codes 01_H to 05_H cannot be used by users.

Figure 2-68 shows an example of the character pattern configuration for an 8 × 9 bit character (character code 01_H), with the ROM addresses and data.

In case of using SLANT function, the character data should be fixed in 6 × 9 dots area on the left side as below.

Figure 2-69 shows the character ROM dump list for the above character pattern.

When ordering a mask, load the data to character ROM starting at address 4000_H going to 4FFF_H in the 1Mbit EPROM. Write "00" for all the data of address 4000_H to 4008_H, 4010_H to 4018_H, 4020_H to 4028_H, 4030_H to 4038_H, 4040_H to 4048_H and 4050_H to 4058_H in character ROM. Write "FF" for all the data which has ***9_H to ***F_H as an address in character ROM.

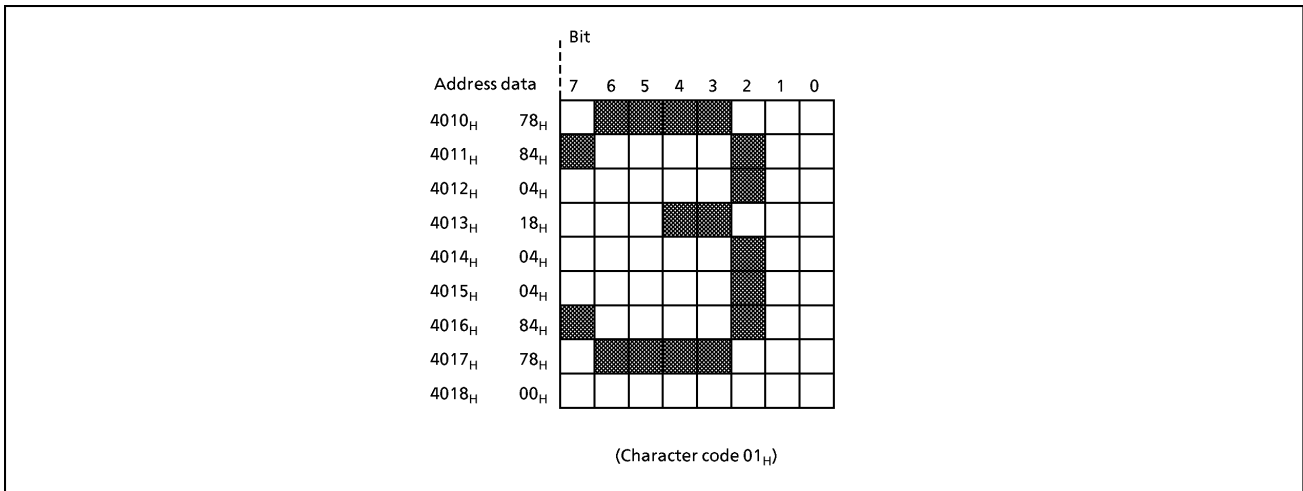


Figure 2-68. 8 × 9 Dot Pattern Configuration

4010/ 78 84 04 18 04 04 84 78 00 FF FF FF FF FF FF FF

Figure 2-69. Character ROM Dump List

(2) Display memory

Each character of the 256 characters displayed in 32 columns×8 lines consists of 14bits in the display memory. Five data items are written to the display memory : character code, color data, blinking specification, underline enable, and slant enable. The display memory is in an unknown state at reset. There are two modes for writing display data to the display memory. One mode is used for writing all display data (character code, color data, blinking specification, underline enable, and slant enable) simultaneously. The other mode is used for changing either character codes or the remaining data items (color data, blinking specification, underline enable, and slant enable). How to write display data to the display memory is described in section 2.12.3 (24).

Display memory configuration

- Character code specification register (8 bits) CRA7 to CRA0
- Color data specification register (3bits) RDT / GDT / BDT
- Blinking specification flag (1 bit) BLF
- Underline enable flag (1 bit) EUL
- Slant enable flag (1 bit) SLNT

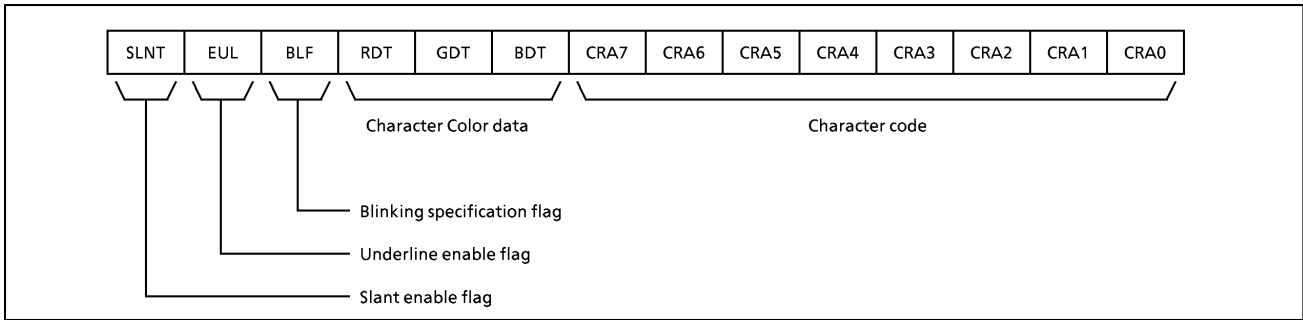


Figure 2-70. Display Memory Bit Configuration

Column \ Line	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	
1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	
2	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F	
3	40	41																														5F	
4	60																															7F	
5	80																															9F	
6	A0																															BF	
7	C0																															DF	
8	E0																															FE	FF

Note : Numerals in the table indicate (hexadecimal) addresses in the display memory.

Figure 2-71. Display Memory Address Configuration

2.12.3 OSD Circuit Control

The OSD circuit performs control functions using the OSD control registers which reside in addresses 001D_H to 001F_H and 0027_H to 002F_H in the special function registers (SFR), and in addresses 0F89_H to 0F92_H and 0F96_H to 0F98_H in the data buffer register (DBR). Section 2.12.3 (28) shows the OSD control registers. To write data to the OSD control registers, use the normal data buffer register access method. The OSD control registers are used to set display start position, display character designs (that is, fringing, smoothing, color data, character size, and etc.), display memory addresses, and character codes. Setting the display on-off control bit, DON, (bit 0 in ORDON) to 1 enables display (starts display). Setting DON to 0 disables display (halts display).

How to write to or read from the OSD control registers refer to section 2.12.3 (28).

(1) Display position

The horizontal display start position can be set to any value by the 128 steps. The vertical display start position can be set to any value by the 256 steps. The horizontal display start position is specified by OSD control registers HS16 to HS10 (ORHS1). The vertical display start position for the first line is specified using VS17 to VS10 (ORVS1). The vertical display start position for the second line to the eighth line are specified by setting VS27 to VS20 (ORVS2) ... VS87 to VS80 (ORVS8).

Horizontal display start position

Specified Page by Page.

Specification steps : 128

Vertical display start position

Specified Line by Line.

Specification steps : 256

Horizontal display start position register (7 bits)

Lines 1 to 8 HS16 to HS10 (ORHS1)

Vertical display start position registers (8 bits × 8)

Line 1 : VS17 to VS10 (ORVS1)

Line 2 : VS27 to VS20 (ORVS2)

⋮ ⋮

Line 8 : VS87 to VS80 (ORVS8)

Horizontal display start position

When FORS = 0, normal frequency mode

$$HS1 = \{(HS16 \text{ to } HS14) \times 16^1 + (HS13 \text{ to } HS10) \times 16^0\} \times 2T_{OSC} + 10T_{OSC}$$

When FORS = 1, double frequency mode

$$HS1 = \{(HS16 \text{ to } HS14) \times 16^1 + (HS13 \text{ to } HS10) \times 16^0\} \times T_{OSC} + 5T_{OSC}$$

Vertical display start position

When VDSMD = 0, normal mode

$$\text{Line } n : VS_n = \{(VS_{n7} \text{ to } VS_{n4}) \times 16^1 + (VS_{n3} \text{ to } VS_{n0}) \times 16^0\} \times 1T_{HD}$$

When VDSMD = 1, double scan mode

$$\text{Line } n : VS_n = \{(VS_{n7} \text{ to } VS_{n4}) \times 16^1 + (VS_{n3} \text{ to } VS_{n0}) \times 16^0\} \times 2T_{HD}$$

T_{OSC} : One cycle of OSC oscillation

T_{HD} : One cycle of HD signal

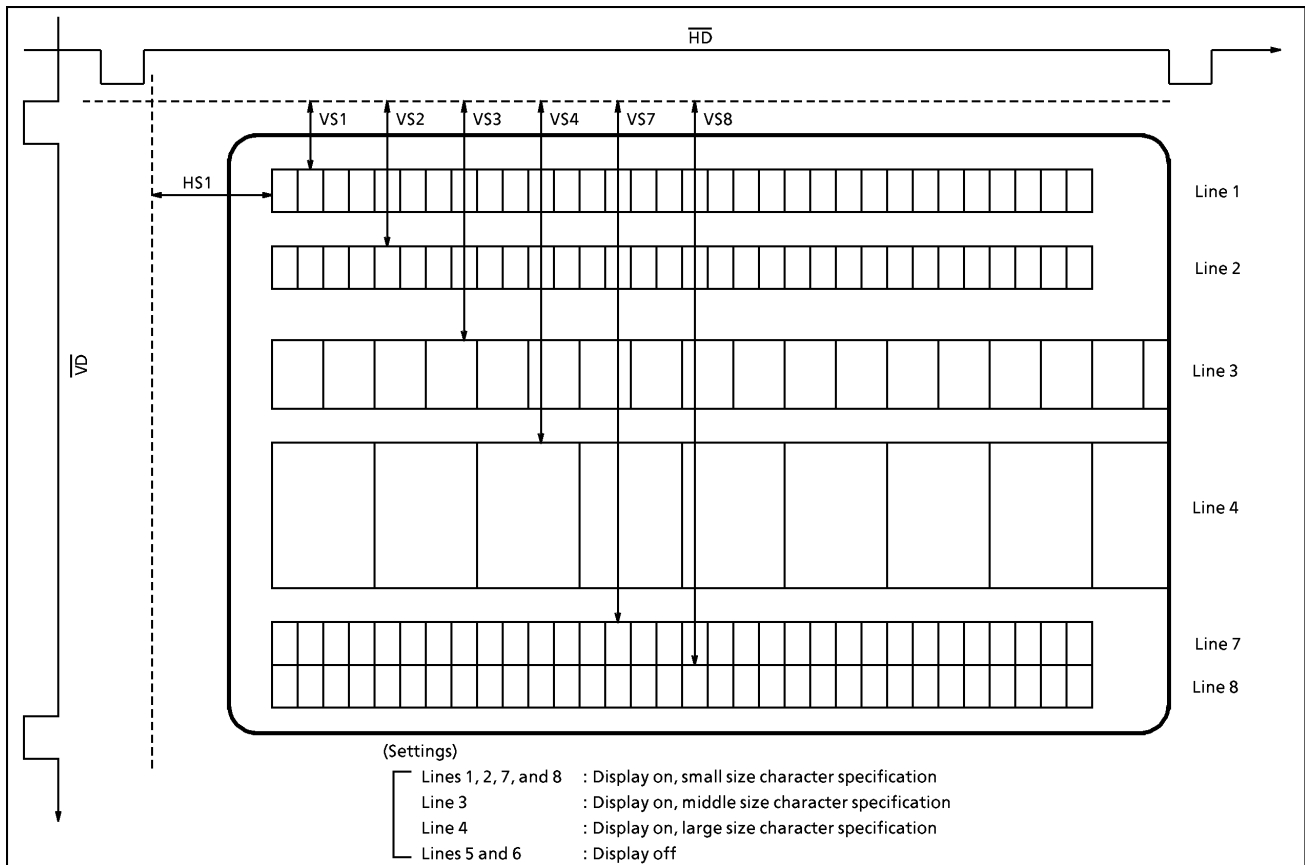


Figure 2-72. TV Screen Display Image

• Notices for setting horizontal display start position

Lines of OSD (VS1 to VS8) are fixed priority levels as follows :

VS1 > VS2 > VS3 > VS4 > VS5 > VS6 > VS7 > VS8

① When horizontal display start positions are set as follows :

$(ORVS1) \leq (ORVS2) \leq (ORVS3) \leq (ORVS4) \leq (ORVS5) \leq (ORVS6) \leq (ORVS7) \leq (ORVS8)$,

if higher priority level line overlaps lower one, lower one is not displayed.

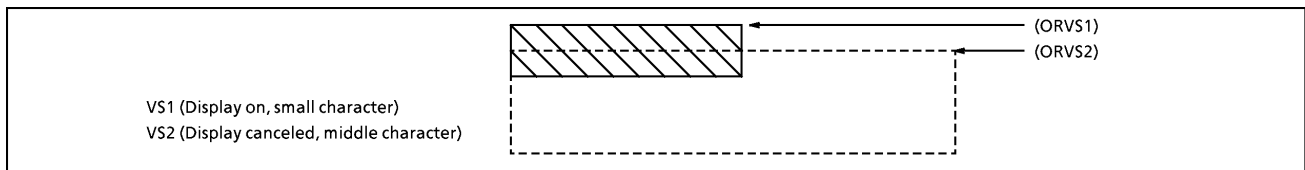


Figure 2-73. Occasion of Overlapping (VS1 ≤ VS2)

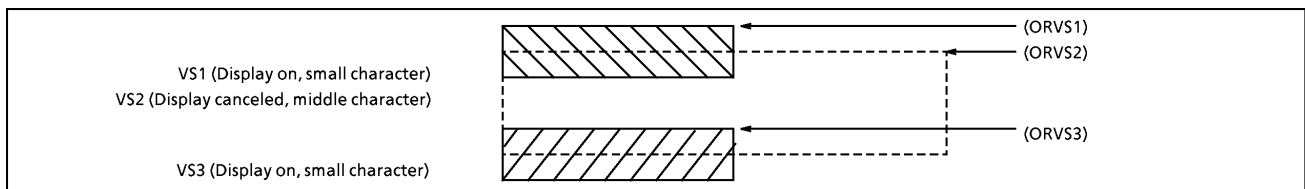


Figure 2-74. Occasion of Overlapping (VS1 ≤ VS2 ≤ VS3)

Then the display line counter (refer to section 2.12. (16)) does not count up canceled lines.

- ② When horizontal display start position is set as follows :
 $ORVS1 > ORVS2 > ORVS3 > ORVS4 > ORVS5 > ORVS6 > ORVS7 > ORVS8$,
 if higher priority level line overlaps lower one, lower one is changed to higher one in the middle of lower one.

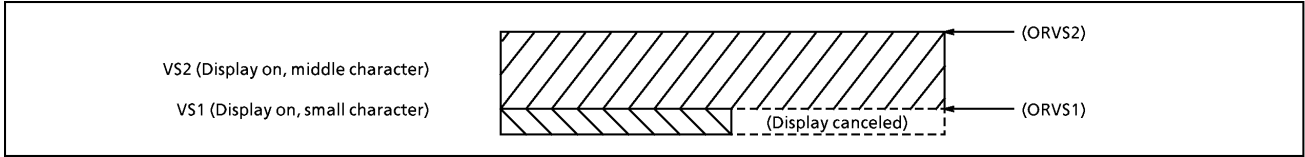


Figure 2-75. Occasion of Overlapping (VS1 > VS2)

Then the display line counter counts up only one.

If higher priority level line is filled with space character, under side of lower one is broken off.

Note 1: (ORVS_n) means the value of the vertical display start position register VS_n7 to VS_n0.
Note 2: If display lines are overlapped each other a display line is canceled as above. Set a space line for not overlapping display lines, or set vertical display start position to the out of display area.
Note 3: When the display line is also set to the display off by character size (CS_n1, 0 = 00), overlapped display line is canceled as above. The width of the display off line is the same as that of the small character line.

(2) Double scan mode

The OSD circuit has a double scan mode. This enables counting by double steps in the vertical direction to handle non-interlaced scanning TVs. This mode also enables vertical display position to be set for the whole screen. Setting the OSD control register VDSMD (bit 4 in ORETC) to "1" sets double scan mode. after releasing the reset mode, the initial mode is the normal mode.

Double scan mode select register (1 bit) VDSMD (Bit 4 in ORETC)
 "0" Normal mode
 "1" Double scan mode

(3) Character sizes and display on / off

There are three character sizes : large, middle and small. One character size can be specified for each line. Display on / off can also be specified for each line. Character size and display on / off are specified using OSD control registers CS11, CS10...CS81, CS80 (ORCS4 and ORCS8).

Character sizes : Large, middle, small

Character size and display on / off specification unit : Line

Character size select / display on / off register (2 bits × 8)

For line 1 : CS11 and CS10 (Bits 1 and 0 in ORCS4)

For line 2 : CS21 and CS20 (Bits 3 and 2 in ORCS4)

: :

For line 8 : CS81 and CS80 (Bits 7 and 6 in ORCS8)

Table 2-10. Character Size and Display On / Off Specifications (n : 1 to 8)

CSn1	CSn0	Character Size	Display On / Off
1	1	Small	On
1	0	Middle	On
0	1	Large	On
0	0	— (Note)	Off

Table 2-11. Dot and Character Sizes

		VDSMD = 0, (Normal Mode)		VDSMD = 1, (Double Scan Mode)	
		DOT Size	Character Size	DOT Size	Character Size
FORS = 0 (Normal mode)	Small	2 T _{OSC} × 1 T _{HD}	16 T _{OSC} × 9 T _{HD}	2 T _{OSC} × 2 T _{HD}	16 T _{OSC} × 18 T _{HD}
	Middle	4 T _{OSC} × 2 T _{HD}	32 T _{OSC} × 18 T _{HD}	4 T _{OSC} × 4 T _{HD}	32 T _{OSC} × 36 T _{HD}
	Large	8 T _{OSC} × 4 T _{HD}	64 T _{OSC} × 36 T _{HD}	8 T _{OSC} × 8 T _{HD}	64 T _{OSC} × 72 T _{HD}
FORS = 1 (Double frequency mode)	Small	1 T _{OSC} × 1 T _{HD}	8 T _{OSC} × 9 T _{HD}	1 T _{OSC} × 2 T _{HD}	8 T _{OSC} × 18 T _{HD}
	Middle	2 T _{OSC} × 2 T _{HD}	16 T _{OSC} × 18 T _{HD}	2 T _{OSC} × 4 T _{HD}	16 T _{OSC} × 36 T _{HD}
	Large	4 T _{OSC} × 4 T _{HD}	32 T _{OSC} × 36 T _{HD}	4 T _{OSC} × 8 T _{HD}	32 T _{OSC} × 72 T _{HD}

T_{OSC} : One cycle of OSC oscillation T_{HD} : One cycle of HD signal

Note: The display off line operates like the width of a small character size line though the character is not displayed.

(4) Character configuration

The area for a character consists of 8 × 13 dots : character display area, underline display area, and space area. A display character is specified by a character code ; underline display is enabled or disabled by the underline enable flag. Figure 2-76 shows a display character image.

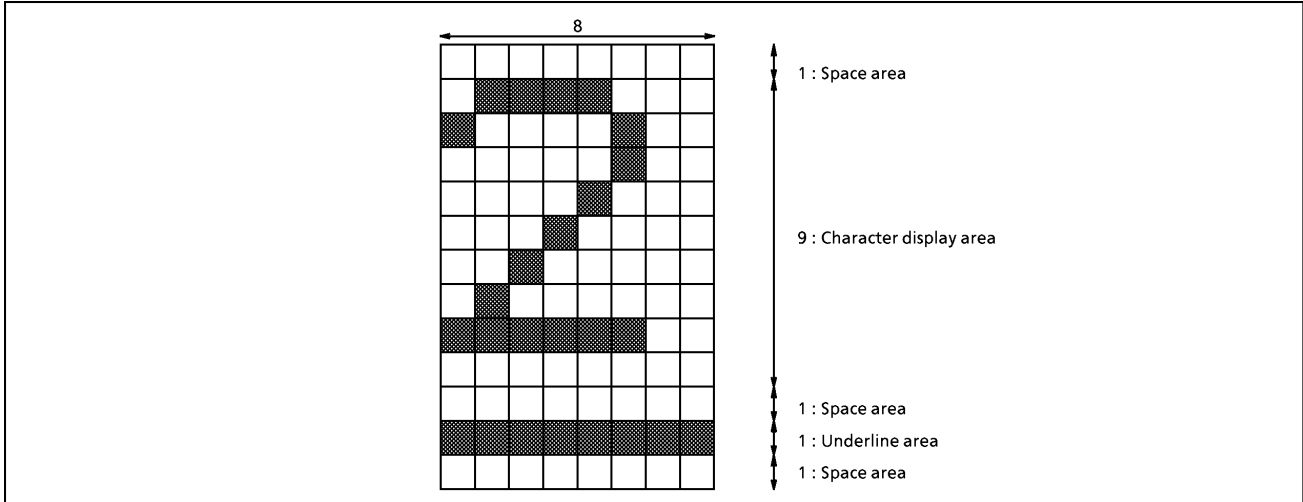


Figure 2-76. Display Character Image

(5) Smoothing function

The smoothing function removes the jagged edges from displayed characters. When smoothing is enabled, 1 / 4-size dots are displayed between any two dots connected corner to corner within a character, as shown in Figure 2-77. Smoothing is enabled by setting = 20 ESMZ (ORETC bit 5) in the OSD control register to "1".

Smoothing specification unit : Page

Smoothing specification register (1bit) ESMZ (ORETC bit 5)

"0" No smoothing

"1" Smoothing enabled

Note: When smoothing small characters, set the command register of the jitter elimination circuit as follows :

Jitter elimination circuit command register :

HJRM = 1 (Jitter elimination mode enabled)

AFLD = 1 (Auto field line decision mode)

(6) Fringing function

The fringing function is used to display a character with a fringe (width is 1 / 2 dot) which has a different color from that of the character. When a character has dots which are active on the edge of the 8x9 character area, the fringe exceeds the boundaries of the character area by 1 / 2 a dot. This occurs on the, left, and right of the character area is displayed with the maximum of 8 vertical dots and 9 horizontal dots, the fringe exceeds right and left, of the character display area. If there is an adjacent character whose outer dot is active, then this dot will overrule the fringe in the horizontal direction. Underlines are not fringed.

Fringing is enabled for each line by setting EFR1 to EFR8 (at address 0F8B_H) in the OSD control register to "1".

A fringe color, which is common to all lines, is specified using OSD control registers, RFDT, GFDT, and BFDT, (bits 2 to 0 in ORBK). Do not enable both fringing and solid space simultaneously.

Specified Line by line

Fringe color specification unit : Common to all lines
(1 color can be selected in 7 colors.)

Fringing enable register (1 bit x 8)

- line 1 EFR1 (Bit 0 in OREFR)
- line 2 EFR2 (Bit 1 in OREFR)
- :
- :
- line 8 EFR8 (Bit 7 in OREFR)

Fringing specification

- EFRn (n : 1 to 8)
- "0" No fringing
- "1" Fringing enable

Fringe color register (3 bits) : RFDT, GFDT, BFDT (Bits 2 to 0 in ORBK)

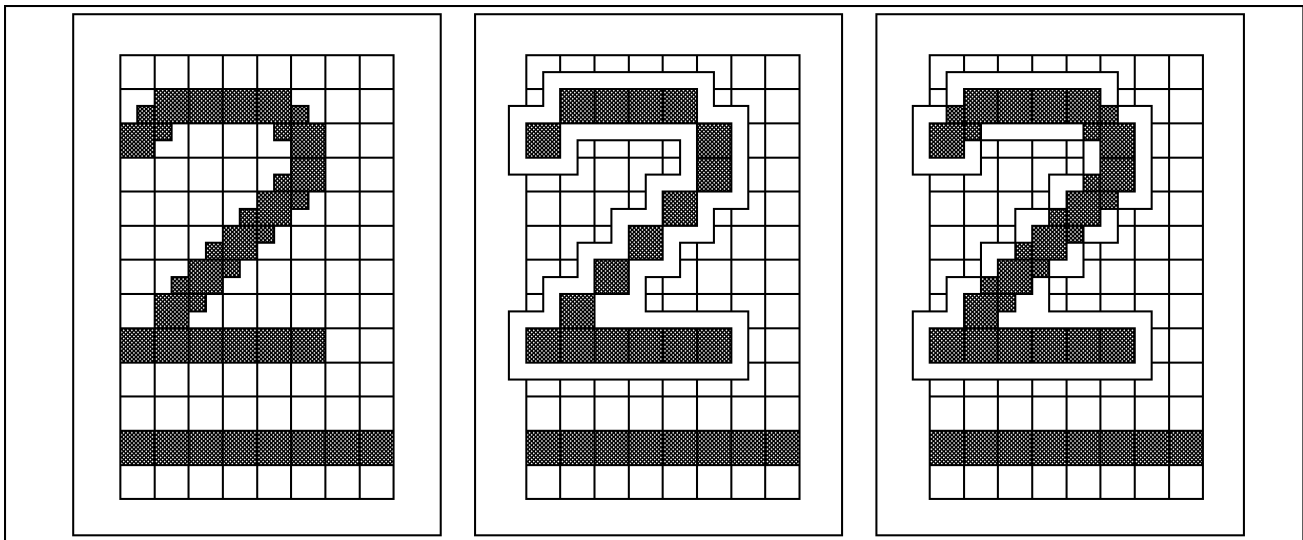


Figure 2-77. Smoot

Figure 2-78. Fringing

Figure 2-79. Priority of smoothing and fringing

(7) Background function

The background function is able to display with a background color in the color of 8 × 13 dots areas except the character. (Except for areas whose character code is blank data)

This function is specified for each display page by setting the OSD control register EBKGD (bit 7 in ORBK) to "1".

A background color is specified for each display page by setting the OSD control registers, RBDT, GBDT, and BBDT (bits 5 to 3 in ORBK) to "1".

Background specification unit : display page (1 color can be selected in 8 colors.)

Background enable register (1 bit) : EBKGD (Bit 7 in ORBK)

"0" No background function

"1" Background function enable

Background color specification registers (3 bits) :

..... RBDT, GBDT, BBDT (Bits 5 to 3 in ORBK)

Note: When the background function is used, the blank character (Code 00H) can not be used as the first character on the fringing line.

(8) Full-raster blanking function

The full-raster blanking function deletes the display of the video signal and is able to color on the whole screen with the background color. Display page video signal is used to first delete by BL signal. In case of using the full-raster blanking function, it is necessary to select BL signal, because it is impossible to remove display page video signal by Y signal.

This function is specified for each display page by setting OSD control register EXBL (bit 6 in ORBK) to "1".

Color specification : Same as that for background.

Full-raster blanking specified display page by display page.

Full-raster blanking enable register : EXBL (Bit 6 in ORBK)

"0" No full-raster blanking

"1" Full-raster blanking

Full-raster blanking color specification registers (3 bits) :

..... RBDT, GBDT, BBDT (same as background color)

(9) f_{OSC} frequency select

f_{OSC} frequency mode select : This function is to select f_{OSC} frequency mode. By setting FORS (bit 6 in ORIV) to "1", the OSD circuit is operated with clock (2 × f_{OSC}).

f_{OSC} frequency mode select register (1 bit) FORS (Bit 6 in ORIV)

"0" Normal frequency mode

"1" Double frequency mode

(10) Character

Characters : 251 (including blank data).

Character specification register (8 bits) CRA7 to CRA0 (Bits 7 to 0 in ORCRA)

"00" Blank data

"01" to "05" Can not be used by users

"06" to "FF" User programmable by character ROM

(11) Character color

Character colors : 7

Character color specification unit : character

Character color specification register (3 bits) RDT, GDT, BDT (In the display memory)

Table 2-12. Character Color

RDT	GDT	BDT	Character Color
0	0	1	Blue
0	1	0	Green
0	1	1	Cyan
1	0	0	Red
1	0	1	Magenta
1	1	0	Yellow
1	1	1	White

(12) Solid space control

Solid space control is used to display one column of solid space to the left and right of 32 columns.

Solid space control is used to delete the Video signal in the areas where solid spaces are located in the original display page, then add color to them.

Solid space specification unit : line

Solid space specification register (16 bits)

For line 1 SOL11 and SOL10 (Bits 1 and 0 in ORSOL4)

For line 2 SOL21 and SOL20 (Bits 3 and 2 in ORSOL4)

⋮ ⋮

For line 8 SOL81 and SOL80 (Bits 7 and 6 in ORSOL8)

Solid space specification

The solid space control functions as follows :

SOL*1 / SOL*0

"00" No solid space display

"01" Solid space display left for 32 columns

"10" Solid space display right for 32 columns

"11" Solid space display left and right for 32 columns

Solid space color specification registers (3 bits)

..... RBDT, GBDT, BBDT (Bits 5 to 3 in ORBK)

(Same color as that of background)

(13) Underline function

Underline function is used to add a line under a display character. The underline is same color as that of character.

Underline specification unit : Character

Underline enable register (1 bit) EUL (Bit 4 in ORDSN)
 (This resides in the display memory)

"0" No underline

"1" Underline

Underline color specification registers (3 bits)

..... RDT, GDT, BDT (Bit 2 to 0 in ORDSN)
 (This resides in the display memory, same color as that of character)

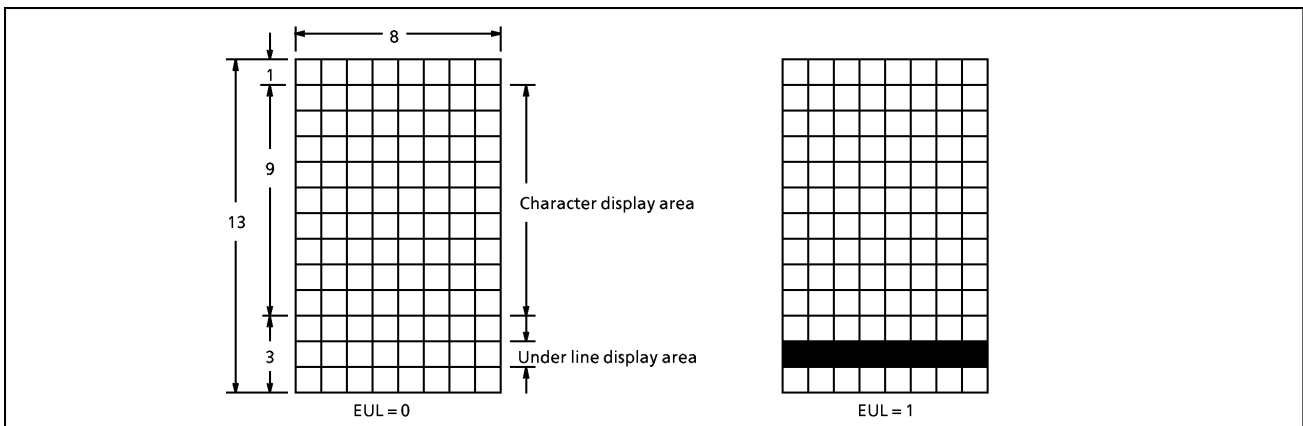


Figure 2-80. Underline

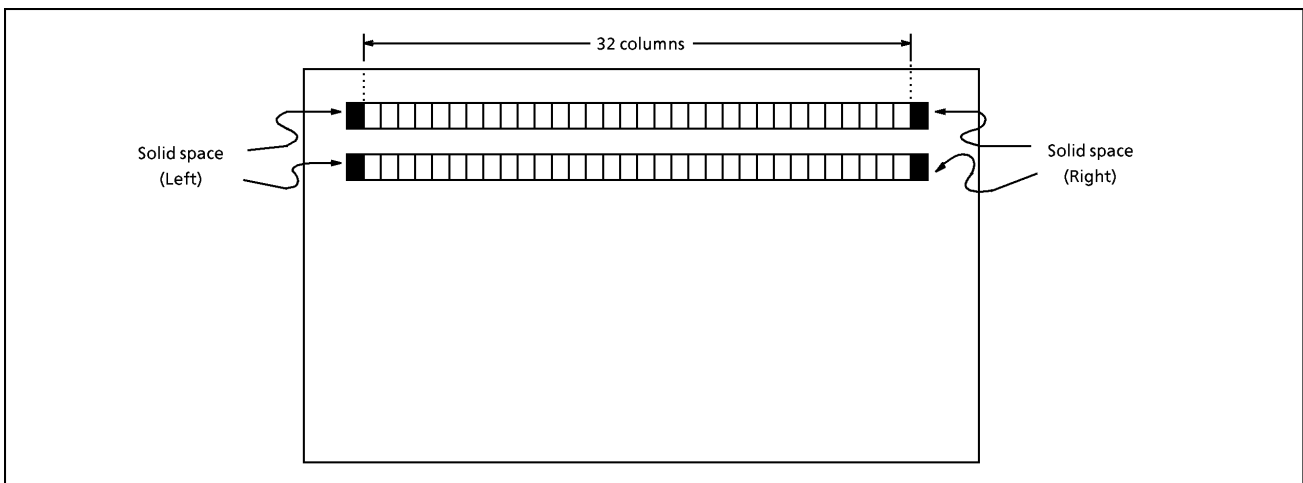


Figure 2-81. Solid Space

(14) Blinking function

Blinking function is used to blink any display character.

When BKMF = 1, characters specified to blink by BLF are not displayed. Blinking display is able to be set BKMF by software. (Space is displayed. That is, if the background color function is used, the background color is not disappeared.)

Blinking specification unit : Character

Blinking specification register (1 bit) BLF (Bit 3 in ORDSN) (In the display memory)

"0" No blinking

"1" Blinking

Blinking master flag (1 bit) BKMF (Bit 6 in ORETC)

"0" Blinking function disable

"1" Blinking function enable

(Characters whose BLF is set to "1" are not displayed.)

(15) Slant function

Slant function is used to slant characters for italics.

Slant specification unit : Character

Slant enable register (1 bits) SLNT (Bit 5 in ORDSN) (In the display memory)

"0" No slant

"1" Slant

Note: SLANT function is enabled each characters, and therefore, in case of using background function, this color of the Background is enable as slant. Regarding the extra dots of the left and / or right character by fringing function, it is not enabled as slant.

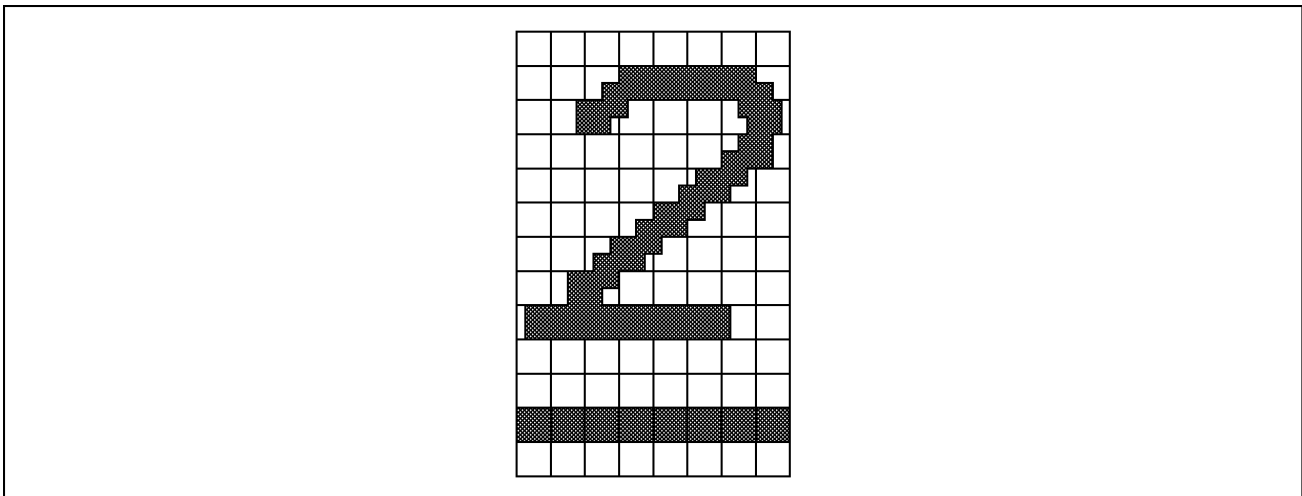


Figure 2-82. Slant

(16) Multiple line display by OSD interrupt

Nine or more lines can be displayed using OSD interrupts. By changing the display start position and display data after the display of each line has been finished, the additional line is able to appear on screen.

Interrupt source select register (1 bit) SVD (Bit 7 in ORIRC)

- "0" An interrupt request is generated when scanning of a line specified by the value set in ISDC is finished (Falling edge of \overline{HD} signal).
- "1" An interrupt request is generated at falling edge of \overline{VD} signal

Display line counter

4 bit counter used to indicate a line being displayed.

This counter is cleared to "0000" the falling edge of the \overline{VD} signal.

The counter is incremented after the scanning of one displayline. (falling edge of the \overline{HD} signal).

The counter is incremented even when a line with all blank data or a line with display on / off bit off occurs. (The display line specified as display off by character size register is incremented by the same width of small character.)

It is necessary to be read out display line counter several time, because it does not synchronize CPU clock. (There is a possibility which the display line counter's value changes during reading)

Display line counter register (4 bit s) DCTR (Bits 3 to 0 in ORIRC)

- "0000" No display line or end of display of line 16
- "0001" End of display of line 1
- "0010" End of display of line 2
- "0011" End of display of line 3
- "0100" End of display of line 4
- "0101" End of display of line 5
- "0110" End of display of line 6
- "0111" End of display of line 7
- "1000" End of display of line 8
- "1001" End of display of line 9
- "1010" End of display of line 10
- "1011" End of display of line 11
- "1100" End of display of line 12
- "1101" End of display of line 13
- "1110" End of display of line 14
- "1111" End of display of line 15

Interrupt generation line specification register (3 bits) ISDC (Bits 6 to 4 in ORIRC)

When the lower 3 bits in DCTR are set as follows :

- "000" Interrupt request generated when the display line counter is cleared or at end points of the last scanning line of the 16'th display line.
- "001" Interrupt request generated at end points of the last scanning line of the first or 9'th display line.
- "010" Interrupt request generated at end points of the last scanning line of the 2'nd or 10'th display line.
- to
- "111" Interrupt request generated at end points of the last scanning line of the 7'th or 15'th display line.

(17) P6 port output select

P67S to P64S are able to be selected as P67 to P64 port or R / G / B / Y / BL output port.

P6 port output select registers (4 bits) P67S to P64S (Bits 7 to 4 in ORP6S)

"0" R, G, B, Y / BL signal output

"1" P6 port input / output

(18) OSD pin output polarity control

Output polarity control

Output polarity control registers (3 bits)

For BL BLIV (Bit 4 in ORIV)

For Y YIV (Bit 3 in ORIV)

For R, G, and B .. RGBIV (Bit 2 in ORIV)

Output polarity control

**IV

"0" Active high

"1" Active low

(19) OSD pin input polarity control

Input polarity control

Input polarity control register of RIN / GIN / BIN / Y / BLIN (2 bits)

For Y / BLIN YBLII (Bit 1 in ORIV)

For RIN, GIN, and BIN RGBII (Bit 0 in ORIV)

Input polarity control

**II

"0" Active high

"1" Active low

Input polarity control register of \overline{HD} / \overline{VD} (2 bits)

For \overline{VD} VDPOL (Bit 1 in ORPOL)

For \overline{HD} HDPOL (Bit 0 in ORPOL)

Input polarity control

**POL

"0" Active low

"1" Active high

(20) Y / BL signal select

Selects of either Y or BL signal output from the Y / BL pin

Y signal Logical OR for R, G, B, character pattern, and fringing

BL signal EXBL (Bit 6 in ORBK)

When EXBL = 0 (no full-raster blanking) :

Output in all areas where characters can be displayed. (Except for character code 00_H : blank data)

When EXBL = 1 (full-raster blanking) :

Output in the whole screen

Y / BL signal select register (1 bit) YBLCS (Bit 7 in ORETC)

"0" Y signal output

"1" BL signal output

(21) R, G, B, Y / BL signal select

Selects either R, G, B, and Y / BL signals from the internal OSD circuit, or RIN, GIN, BIN, and Y / BLIN signals externally input.

- R, G, B, Y / BL signal select registers (2 bits) MPXS1 / MPXS0
(Bits 3 and 2 in ORETC)
- "00" Simultaneous output (Signal from the OSD circuit has higher priority.)
- "01" Output of signal from internal OSD circuit
- "10" Output of signal from externally input
- "11" Simultaneous output (Externally input signal has higher priority.)

(22) Display memory access

There are two types of access : write data to the display memory and read data from the display memory.

The display memory is accessed using the following registers : DMA7 to DMA0, CRA7 to CRA0, RDT, GDT, BDT, BLF, EUL, SLNT, MBK, MFYWR, RDWRV.

- Display memory read mode specification register (1 bit) MFYWR (Bit 0 in ORP6S)
 - "0" Normal mode
 - "1" Read-modify-write-mode
- Read / write mode select register at normal mode (MFYWR = 0) RDWRV (Bit 1 in ORP6S)
 - "0" Data write mode
 - "1" Data read mode
- Display memory bank switching register (1 bit) MBK (Bit 0 in ORETC)
 - "0" Access to either character code or character display options
 - "1" Access to both character code and character display options

Display memory auto increment depends on MBK setting.

Table 2-13. Address Increment

		RD		WR	
		Color Data	Character Data	Color Data	Character Data
MFYWR = 0	MBK = 0 MBK = 1	INC —	INC INC	INC —	INC INC
MFYWR = 1	MBK = 0 MBK = 1	— —	— —	INC —	INC INC

INC : Automatic address increment at read or write

— : No address change at data read or write

- Display memory address specification register (8 bits) DMA7 to DMA0 (Bits 7 to 0 in ORDMA)
- Display memory data access register
 - ① For character code access (8 bits) CRA 7 to CRA0 (Bits 7 to 0 in ORCRA)
 - ② For character design access (6 bits) SLNT, EUL, BLF, RDT, GDT, and BDT
(Bits 5 to 0 in ORDSN)

There are two types of display memory access : normal mode and read-modify-write mode.

Note 1: Don't use the operation [LDW (HL), mn] when accessing display memory.
Note 2: It is impossible to use the read-modify-write instructions (Bit manipulation instructions such as SET, CLR, etc. and logical operation such as AND, OR, etc.)

1. Normal mode

In normal mode, display memory addresses are automatically incremented for every read or write. Since addresses are automatically incremented, this mode is used for simultaneously reading data from multiple consecutive addresses and for simultaneously writing data to multiple consecutive addresses.

– Display memory read sequence –

- ① Set MFYWR to 0. (Set to normal mode.)
EX : LD (ORP6S), 00_H
- ② Set MBK to 0 or 1.
EX : LD (ORETC), 00_H or 01_H
- ③ Set RDWRV to 1. (Set to data read mode.)
EX : LD (ORP6S), 02_H
- ④ Set the display memory address to DMA7 to DMA0.
EX : LD (ORDMA), n
- ⑤ Read data from CRA7 to CRA0, SLNT, EUL, BLF, BDT, GDT, and RDT.
(DMA7 to DMA0 are automatically incremented.)
EX : LD A, (ORDSN) or (ORCRA)
- ⑥ For continuous read, repeat ④ and ⑤.
(For data read from continuous addresses, repeat ⑤.)

– Display memory write sequence –

- ① Same as above.
EX : LD (ORP6S), 00_H
- ② Same as above.
EX : LD (ORETC), 00_H or 01_H
- ③ Set RDWRV to 0. (Set to data write mode.)
EX : LD (ORP6S), 00_H
- ④ Same as above.
EX : LD (ORDMA), n
- ⑤ Write data to CRA7 to CRA0, SLNT, EUL, BLF, BDT, GDT, and RDT.
(DMA7 to DMA0 are automatically incremented.)
EX : LD A, (ORDSN) or (ORCRA), n
- ⑥ Same as above.

2. Read-modify-write mode

In read-modify-write mode, display memory addresses are automatically incremented during a write ; not incremented during a read. Thus, immediately after data is read from a display memory address, data can be written to the same address. After a write, the display memory address is automatically incremented.

– Read-modify-write sequence –

- ① Set MFYWR to 1.
EX : LD (ORP6S), 01_H
- ② Set MBK to 0 or 1.
EX : LD (ORETC), 00_H or 01_H
- ③ Set display memory address to DMA7 to DMA0.
EX : LD (ORDMA), n
- ④ Read data from CRA7 to CRA0, SLNT, EUL, BLF, BDT, GDT, and RDT. (DMA7 to DMA0 are not incremented.)
EX : LD A, (ORDEC) or (ORCRA)
- ⑤ Write data to CRA7 to CRA0, SLNT, EUL, BLF, BDT, GDT, and RDT. (DMA7 to DMA0 are automatically incremented.)
EX : LD (ORDEC) or (ORCRA), n
- ⑥ For continuous read-modify-write, repeat ③, ④, and ⑤.
(For read-modify-write at consecutive addresses, repeat ④ and ⑤.)

Note: In read-modify-write mode, reading only or writeing only can be executed.

(23) Display on / off

Function used to display a line specified for on / off display.

Display on / off specified page by page

Display on / off specification register (1 bit) ... DON (Bit 0 in ORDON)

"0"	Disable display
"1"	Enable display

Note: Don't start STOP mode during display enable. When starting STOP mode, it is necessary that DON is specified as display off. If starting STOP mode during display, the contents of the display memory might be broken.

(24) Window

Function used to set upper and lower limit of page. Window upper limit is specified by WVSH7 to WVSH0. Window lower limit is specified by WVSL7 to WVSL0. This function is enabled by setting EWDW (Bit 1 in ORDON) to "1".

- Window upper limit specification register (8bit) WVSH7 to WVSH0 (QRWVSH)
- Window lower limit specification register (8bit) WVSL7 to WVSL0 (QRWVSL)

Window upper and lower limit position

When VDSMD = 0 (normal mode) :
 $WVSH = (WVSH7 \text{ to } WVSH0) \times H \times 1 T_{HD}$
 $WVSL = (WVSL7 \text{ to } WVSL0) \times H \times 1 T_{HD}$

When VDSMD = 1 (double scan mode) :
 $WVSH = (WVSH7 \text{ to } WVSH0) \times H \times 2 T_{HD}$
 $WVSL = (WVSL7 \text{ to } WVSL0) \times H \times 2 T_{HD}$

T_{HD} : One cycle of \overline{HD} signal

Note: Modify value of window limit registers, at the timing from the end position of display all line to the start position of first display line, or the timing until the position of window upper limit.

- Window enable flag (1 bit) EWDW (Bit 1 in ORDON)
 "0" Disable window function
 "1" Enable window function

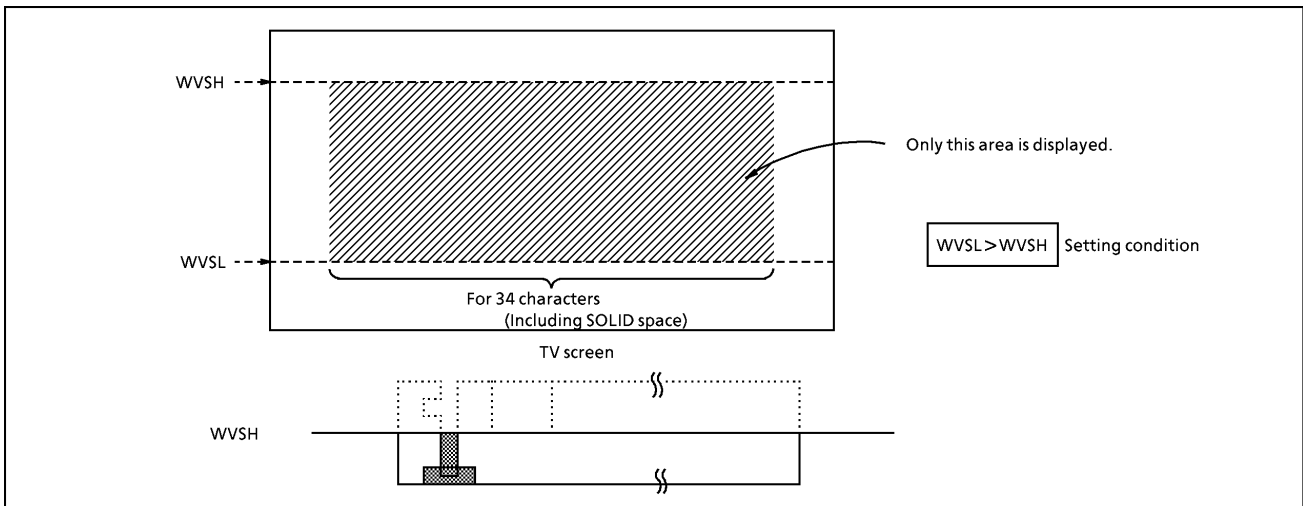


Figure 2-83. Window

<Usage example >

The following can be displayed by combining the window and full-raster blanking functions.

Table 2-14. Window / Full-Raster Blanking

EXBL	EWDW	Internal OSD BL Output
0	0	Vertical direction : Character area is only displayed. Horizontal direction : Character area is only displayed. Horizontal direction : Character area is only displayed.
0	1	Vertical direction : Window area is displayed. Horizontal direction : Character area is only displayed.
1	0	Vertical direction : Whole page is displayed. Horizontal direction : Whole page is displayed.
1	1	Vertical direction : Window area is displayed. Horizontal direction : Window area is only displayed.

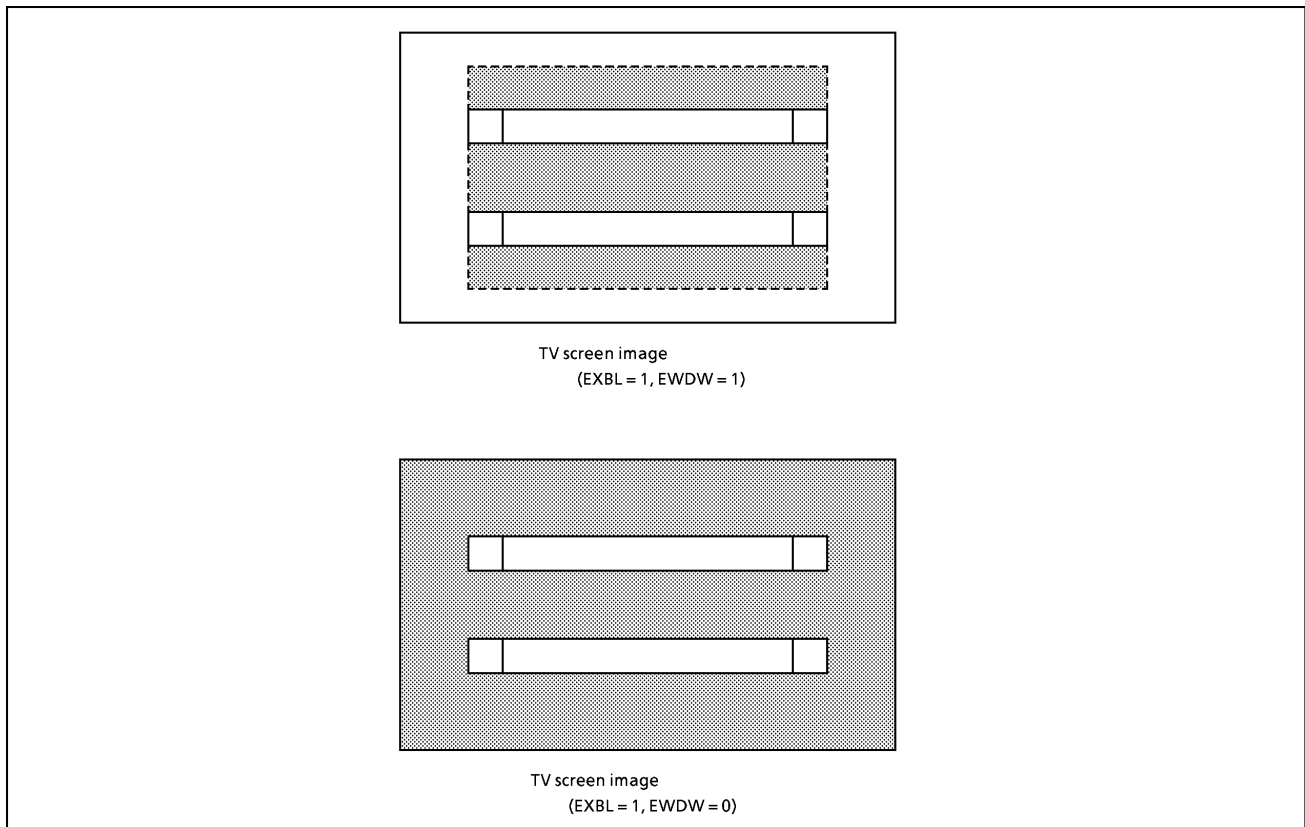


Figure 2-84. TV Screen Image Using Window Function

(25) OSD interrupt control

Interrupt latch of the both OSD interrupt and SLICER interrupt is the same. OSD and SLICER of ORIRC register are interrupt enable registers and interrupt source detection register. When OSD interrupt is enabled, set the OSD of the OSD interrupt enable register (bit 3 in ORIRC) to "1", and when SLICER interrupt is enabled, set the SLICR of the SLICER interrupt enable register (bit 2 in ORIRC) to "1". OSD and SLICER interrupts can be enabled at the same time.

OSD interrupt enable register	OSD (Bit 3 in ORIRC)
"0" OSD interrupt is disabled
"1" OSD interrupt is enabled
SLICER interrupt enable register	SLICR (Bit 2 in ORIRC)
"0" SLICER interrupt is disabled
"1" SLICER interrupt is enabled

When OSD and SLICER are enabled at the same time, the interrupt source can be detected by OSD (bit 5 in ORIRC) and SLICR (bit 4 in ORIRC) of the interrupt source monitor register whether it is OSD interrupt or it is SLICER interrupt.

These OSD and SLICR bits of the interrupt source monitor register are cleared by executing the read instruction for this register.

Note: OSD and SLICR bits of the interrupt enable register is different from those of the interrupt source monitor register.

(26) OSD control register write / read

The addresses of the OSD control registers are assigned to the DBR register.

For writing data to or reading data from the OSD control registers, access the DBR register in the normal way.

If RGWR register is set to "1" the written data is transferred to the OSD circuit and become valid.

However, while the display line is being scanned, the data written after the line is scanned is transferred to the OSD circuit and becomes valid.

The registers for writing data to display memory become valid, when its data is written. (DMA7 to DMA0, CRA7 to CRA0, RDT, GDT, BDT, BLF, EUL, SLNT, YBLCS, BKMF, ESMZ, VDSMD, MPX, MBK, P67S to P64S, RDWRV, MFYWR, VDPOL and HDPOL)

Written data transfer register (1 bit)	RGWR (Bit 2 in ORDON)
"0"	Initialized state
"1"	Transfers written data to OSD circuit. (After transfer, RGWR is reset to 0.)

Note: Don't write "0" to RGWR.

<RGWR timing>

1) RGWR system

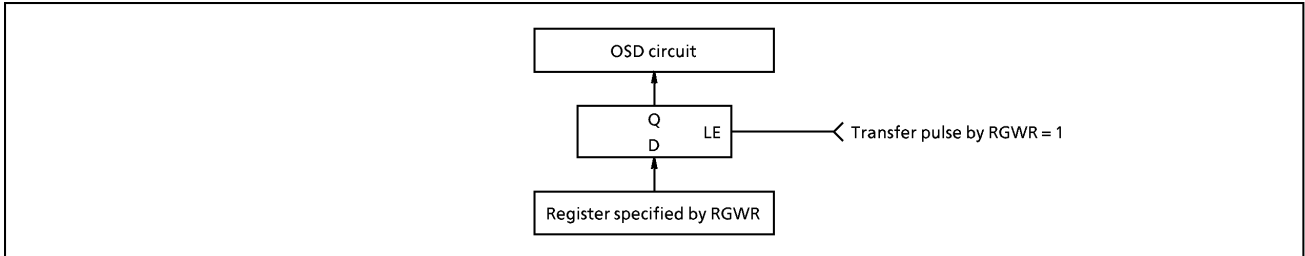


Figure 2-85. RGWR System

2) Transfer timing

- ① No display area (including any lines specified as display off by character size)
When having set RGWR to "1" during no display area, the timing OSD register can be transferred is at the falling edge of \overline{HD} signal.

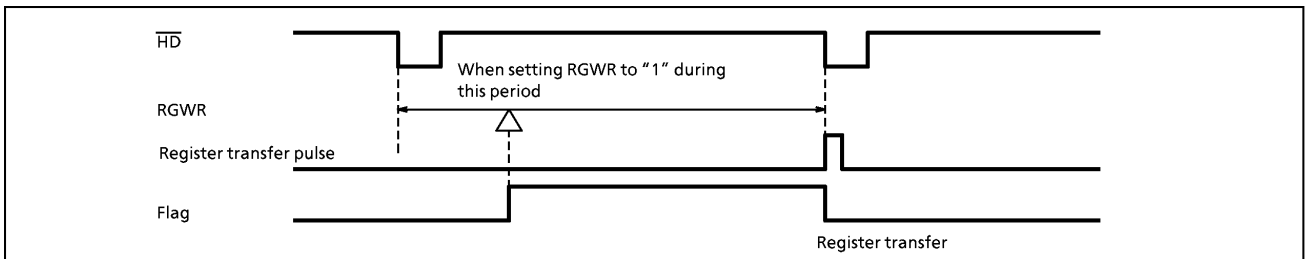


Figure 2-86. Data Transfer Timing in No Display Area

- ② Display area
When having set RGWR to "1" during display area, the timing OSD register can be transferred is at the falling edge of \overline{HD} signal when the display line has been finished.

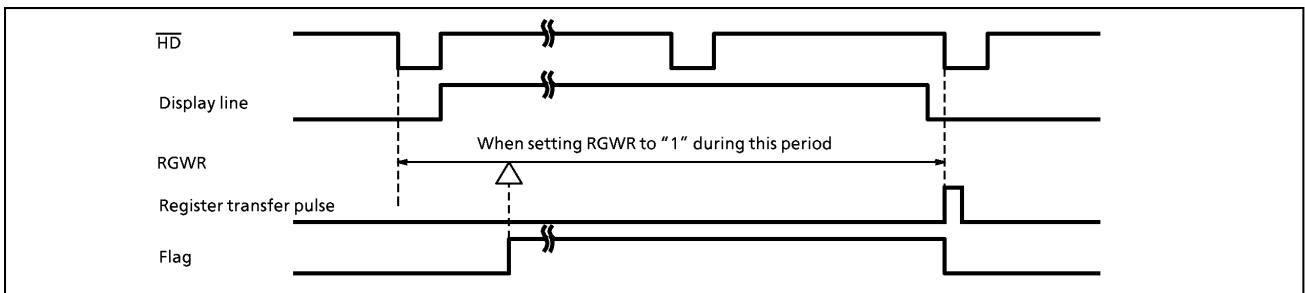


Figure 2-87. Data Transfer Timing in Display Area

3) Flag

RGWR flag is set to "1" during the period from the timing having sett RGWR to "1" to the timing register transfer pulse is generated.
When RGWR flag becomes "0", the data of OSD register can be available. After setting RGWR to "1", it is possible to write OSD registers even RGWR flag is "1".

(27) OSD control registers

ORHS1 (0027 _H)	7	6	5	4	3	2	1	0	(Initial value : *000 0000)	
	"0"	HS16	HS15	HS14	HS13	HS12	HS11	HS10		
	HS16 to 10	Horizontal display start position						Write only		
ORVS1 (0028 _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS17	VS16	VS15	VS14	VS13	VS12	VS11	VS10		
ORVS2 (0029 _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS27	VS26	VS25	VS24	VS23	VS22	VS21	VS20		
ORVS3 (002A _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS37	VS36	VS35	VS34	VS33	VS32	VS31	VS30		
ORVS4 (002B _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS47	VS46	VS45	VS44	VS43	VS42	VS41	VS40		
ORVS5 (002C _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS57	VS56	VS55	VS54	VS53	VS52	VS51	VS50		
ORVS6 (002D _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS67	VS66	VS65	VS64	VS63	VS62	VS61	VS60		
ORVS7 (002E _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS77	VS76	VS75	VS74	VS73	VS72	VS71	VS70		
ORVS8 (002F _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	VS87	VS86	VS85	VS84	VS83	VS82	VS81	VS80		
	VSn8 to 0	Vertical display start position for line n						Write only	(n = 1 to 8)	
ORCS4 (0F89 _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	CS4	CS3	CS2	CS1						
ORCS8 (0F8A _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	CS8	CS7	CS6	CS5						
	CSn	Character size and display on / off for line n					00 : Display off 01 : Large size 10 : Middle size 11 : Small size		Write only	(n = 1 to 8)
OREFR (0F8B _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	EFR8	EFR7	EFR6	EFR5	EFR4	EFR3	EFR2	EFR1		
	EFRn	Fringing enable for line n					0 : Disable fringing 1 : Enable fringing		Write only	(n = 1 to 8)
ORSOL4 (0F8C _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	SOL4	SOL3	SOL2	SOL1						
ORSOL8 (0F8D _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 0000)	
	SOL8	SOL7	SOL6	SOL5						
	SOLn	Solid space enable for line n					00 : No solid space display 01 : Solid space display left for 32 columns 10 : Solid space display right for 32 columns 11 : Solid space display left and right for 32 columns		Write only	(n = 1 to 8)

Figure 2-88-1. OSD Control Registers (I)

		7	6	5	4	3	2	1	0		
ORBK (0F8E _H)	EBKGD	EXBL	RBDT	GBDT	BBDT	RFDT	GFDT	BFDT	(Initial value : 0000 0000)		
	EBKGD	Background function enable						0 : No background function 1 : Background function enable		Write only	
	EXBL	Full-raster blanking enable						0 : No Full-raster blanking 1 : Full-raster blanking			
	RBDT / GBDT / BBDT	Background color select						000 : Black 001 : Blue 010 : Green 011 : Cyan 100 : Red 101 : Magenta 110 : Yellow 111 : White			
	RFDT / GFDT / BFDT	Fringing color select						000 : Black 001 : Blue 010 : Green 011 : Cyan 100 : Red 101 : Magenta 110 : Yellow 111 : White			
		7	6	5	4	3	2	1	0		
ORETC (0F8F _H)	YBLC	BKMF	ESMZ	VDSMD	MPXS	"0"	MBK	(Initial value : 0000 0000)			
	YBLC	Y / BL signal select						0 : Y signal output 1 : BL signal output		Write only	
	BKMF	Blinking master flag						0 : Disable blinking 1 : Enable blinking			
	ESMZ	Smoothing enable						0 : Disable smoothing 1 : Enable smoothing			
	VDSMD	Double scan mode select						0 : Normal mode 1 : Double scan mode			
	MPXS	R, G, B, Y / BL signal select						00 : Simultaneous output (Signal from the OSD circuit has higher priority.) 01 : Output of signal from internal OSD circuit 10 : Output of signal from externally input 11 : Simultaneous output (Externally input signal has higher priority.)			
	MBK	Display memory bank switching						0 : Access to either character code or character display options 1 : Access to both character code and character display options			

Figure 2-88-2. OSD Control Registers (II)

ORIRC (0F90 _H)	7	6	5	4	3	2	1	0	(Initial value : 0000 00**) Write only	
	SVD	ISDC		OSD	SLCR					
	SVD	Interrupt source select						0 : Interrupt request by the value of ISDC 1 : Interrupt request by the falling edge of \overline{VD} signal		
	ISDC	Interrupt generation line select						When the lower 3bit in DCTR are set as follows : 000 : Interrupt request generated at display end when SVD is "0" and DCTR is "*000". 001 : Interrupt request generated at display end when SVD is "0" and DCTR is "*001". 010 : Interrupt request generated at display end when SVD is "0" and DCTR is "*010". 011 : Interrupt request generated at display end when SVD is "0" and DCTR is "*011". 100 : Interrupt request generated at display end when SVD is "0" and DCTR is "*100". 101 : Interrupt request generated at display end when SVD is "0" and DCTR is "*101". 110 : Interrupt request generated at display end when SVD is "0" and DCTR is "*110". 111 : Interrupt request generated at display end when SVD is "0" and DCTR is "*111".		
	OSD	OSD interrupt enable select						0 : Disable OSD interrupt 1 : Enable OSD interrupt		
SLCR	SLICER interrupt enable select						0 : Disable SLICER interrupt 1 : Enable SLICER interrupt			
ORIRC (0F90 _H)	7	6	5	4	3	2	1	0	Read only	
			OSD	SLCR	DCTR					
	OSD	OSD interrupt monitor						0 : OSD interrupt no generated 1 : OSD interrupt generated		
	SLCR	SLICER interrupt monitor						0 : SLICER interrupt not generated 1 : SLICER interrupt generated		
DCTR	Display line counter						0000 : No display line or end of display of line 16 0001 : End of display of line 1 0010 : End of display of line 2 0011 : End of display of line 3 0100 : End of display of line 4 0101 : End of display of line 5 0110 : End of display of line 6 0111 : End of display of line 7 1000 : End of display of line 8 1001 : End of display of line 9 1010 : End of display of line 10 1011 : End of display of line 11 1100 : End of display of line 12 1101 : End of display of line 13 1110 : End of display of line 14 1111 : End of display of line 15			

Figure 2-88-3. OSD Control Registers (III)

ORP6S (0F91 _H)	<table border="1" style="width:100%; text-align:center;"> <tr> <td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>P67S</td><td>P66S</td><td>P65S</td><td>P64S</td><td></td><td></td><td>RDWRV</td><td>MFYWR</td> </tr> </table> (Initial value : 0000 **00)								7	6	5	4	3	2	1	0	P67S	P66S	P65S	P64S			RDWRV	MFYWR	Write only
	7	6	5	4	3	2	1	0																	
	P67S	P66S	P65S	P64S			RDWRV	MFYWR																	
	P67S to P64S		P6 port output select		0 : R, G, B, Y / BL signal output 1 : Port contents output																				
RDWRV		Read / write mode select at normal mode		0 : Data write mode 1 : Data read mode																					
MFYWR		Display memory read mode		0 : Normal mode 1 : Read-modify-write-mode																					
ORIV (0F92 _H)	<table border="1" style="width:100%; text-align:center;"> <tr> <td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>"0"</td><td>FORS</td><td>"1"</td><td>BLIV</td><td>YIV</td><td>RGBIV</td><td>YBLII</td><td>RGBII</td> </tr> </table> (Initial value : 0000 0000)								7	6	5	4	3	2	1	0	"0"	FORS	"1"	BLIV	YIV	RGBIV	YBLII	RGBII	Write only
	7	6	5	4	3	2	1	0																	
	"0"	FORS	"1"	BLIV	YIV	RGBIV	YBLII	RGBII																	
	FORS		f _{osc} frequency select		0 : Normal frequency mode 1 : Double frequency mode																				
	BLIV		BL output polarity select		0 : Active high 1 : Active low																				
	YIV		Y output polarity select		0 : Active high 1 : Active low																				
	RGBIV		R, G, B output polarity select		0 : Active high 1 : Active low																				
YBLII		Y / BLIN input polarity select		0 : Active high 1 : Active low																					
RGBII		RIN, GIN, BIN input polarity select		0 : Active high 1 : Active low																					
ORDMA (001D _H)	<table border="1" style="width:100%; text-align:center;"> <tr> <td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>DMA7</td><td>DMA6</td><td>DMA5</td><td>DMA4</td><td>DMA3</td><td>DMA2</td><td>DMA1</td><td>DMA0</td> </tr> </table> (Initial value : 0000 0000)								7	6	5	4	3	2	1	0	DMA7	DMA6	DMA5	DMA4	DMA3	DMA2	DMA1	DMA0	Write only
	7	6	5	4	3	2	1	0																	
DMA7	DMA6	DMA5	DMA4	DMA3	DMA2	DMA1	DMA0																		
DMA7 to 0		Display memory address																							
ORDEC (001E _H)	<table border="1" style="width:100%; text-align:center;"> <tr> <td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td></td><td></td><td>SLNT</td><td>EUL</td><td>BLF</td><td>RDT</td><td>GDT</td><td>BDT</td> </tr> </table> (Initial value : **** ***)								7	6	5	4	3	2	1	0			SLNT	EUL	BLF	RDT	GDT	BDT	R/W
	7	6	5	4	3	2	1	0																	
			SLNT	EUL	BLF	RDT	GDT	BDT																	
	SLNT		Slant enable		0 : Disable slant 1 : Enable slant																				
	EUL		Underline enable		0 : Disable underline 1 : Enable underline																				
BLF		Blinking enable		0 : Disable blinking 1 : Enable blinking																					
RDT / GDT / BDT		Character color select		001 : Blue 010 : Green 011 : Cyan 100 : Red 101 : Magenta 110 : Yellow 111 : White																					

Figure 2-88-4. OSD Control Registers (IV)

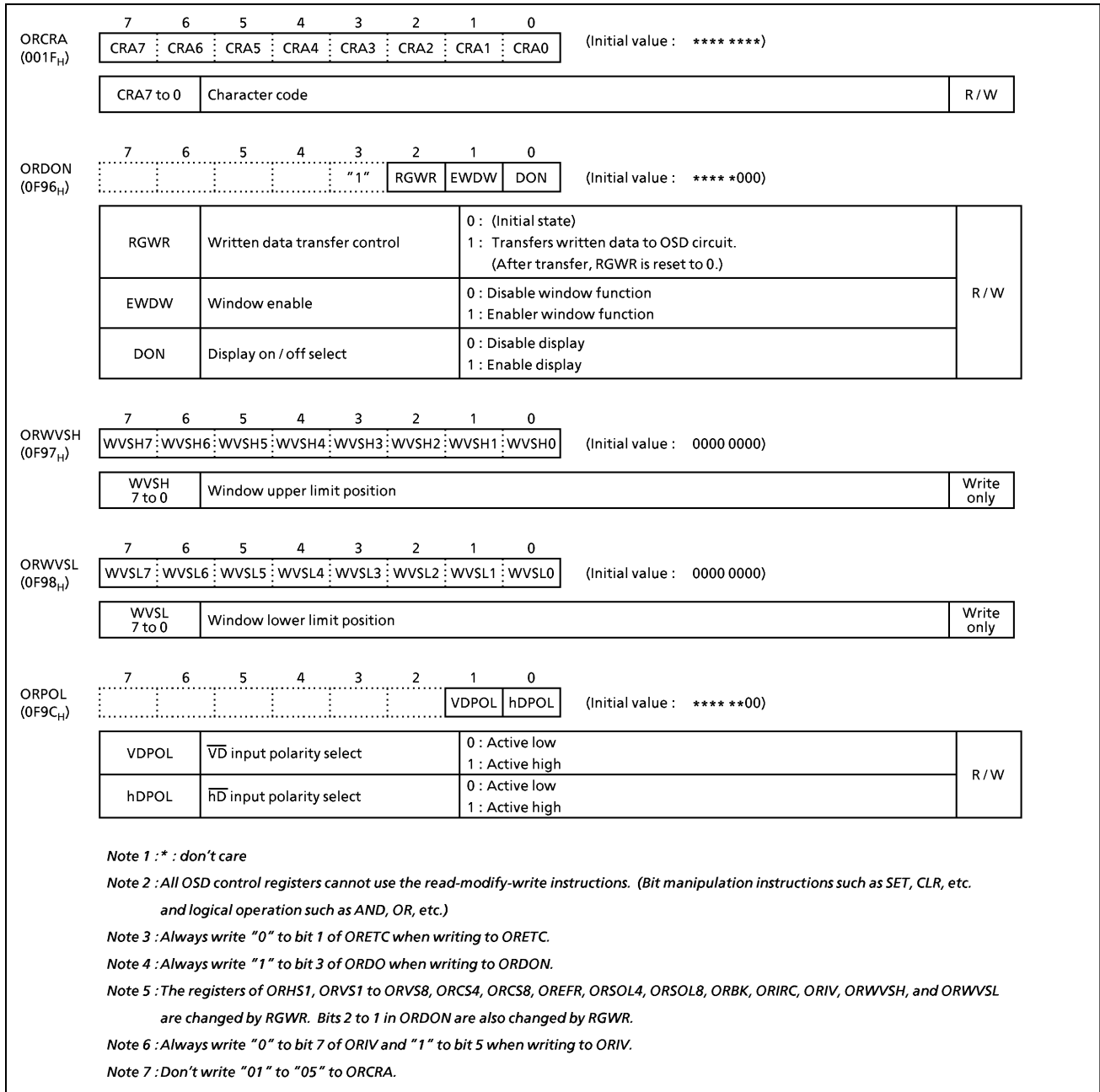


Figure 2-88-5. OSD Control Registers (V)

(28) OSD comand register list

address	7	6	5	4	3	2	1	0
	Horizontal display start position							
☆0027		HS16	HS15	HS14	HS13	HS12	HS11	HS10
	Vertical display start position for line 1							
☆0028	VS17	VS16	VS15	VS14	VS13	VS12	VS11	VS10
	Vertical display start position for line 2							
☆0029	VS27	VS26	VS25	VS24	VS23	VS22	VS21	VS20
	Vertical display start position for line 3							
☆002A	VS37	VS36	VS35	VS34	VS33	VS32	VS31	VS30
	Vertical display start position for line 4							
☆002B	VS47	VS46	VS45	VS44	VS43	VS42	VS41	VS40
	Vertical display start position for line 5							
☆002C	VS57	VS56	VS55	VS54	VS53	VS52	VS51	VS50
	Vertical display start position for line 6							
☆002D	VS67	VS66	VS65	VS64	VS63	VS62	VS61	VS60
	Vertical display start position for line 7							
☆002E	VS77	VS76	VS75	VS74	VS73	VS72	VS71	VS70
	Vertical display start position for line 8							
☆002F	VS87	VS86	VS85	VS84	VS83	VS82	VS81	VS80
	Character size and display on / off for line 1 / 2 / 3 / 4							
☆0F89	CS41	CS40	CS31	CS30	CS21	CS20	CS11	CS10
	Character size and display on / off for line 5 / 6 / 7 / 8							
☆0F8A	CS81	CS80	CS71	CS70	CS61	CS60	CS51	CS50
	Fringing enable							
☆0F8B	EFR8	EFR7	EFR6	EFR5	EFR4	EFR3	EFR2	EFR1
	Solid space enable for line 1 / 2 / 3 / 4							
☆0F8C	SOL41	SOL40	SOL31	SOL30	SOL21	SOL20	SOL11	SOL10
	Solid space enable for line 5 / 6 / 7 / 8							
☆0F8D	SOL81	SOL80	SOL71	SOL70	SOL61	SOL60	SOL51	SOL50
	Background function enable, full-raster blanking enable, background color select, fringing color select							
☆0F8E	EBKGD	EXBL	RBDT	GBDT	BBDT	RFDT	GFDT	BFDT
	Y / BL signal select, blinking master flag, smoothing enable, double scan mode select, R, G, B, Y / BL signal select, display memory bank switching							
0F8F	YBLCS	BKMF	ESMZ	VDSMD	MPXS1	MPXS0	"0"	MBK
	OSD interrupt control, display line counter							
☆0F90	SVD	ISDC2	ISDC1	ISDC0	OSD	SLCR		
		OSD	SLCR	DCTR3	DCTR2	DCTR1	DCTR0	
	P6 port output select, read / write mode select, display memory read mode							
0F91	P67S	P66S	P65S	P64S			RDWRV	MFYWR
	f _{OSC} frequency select, OSD output polarity select, OSD input polarity select							

☆	0F92	"0"	FORS	"1"	BLIV	YIV	RGBIV	YBLII	RGBII
		Display memory address							
	001D	DMA7	DMA6	DMA5	DMA4	DMA3	DMA2	DMA1	DMA0
		Display memory character modification data read / write							
	001E*		SLNT	EUL	BLF	RDT	GDT	BDT	
		Display memory character data read / write							
	001F*	CRA7	CRA6	CRA5	CRA4	CRA3	CRA2	CRA1	CRA0
		Written data transfer control, window enable, display on / off select							
☆☆	0F96				"1"	RGWR	EWDW	DON	
		Window upper limit position							
☆	0F97	WVSH7	WVSH6	WVSH5	WVSH4	WVSH3	WVSH2	WVSH1	WVSH0
		Window lower limit position							
☆	0F98	WVSL7	WVSL6	WVSL5	WVSL4	WVSL3	WVSL2	WVSL1	WVSL0
		OSD input polarity select							
	0F9C						VDPOL	HDPOL	
		<input type="checkbox"/> Read only, <input checked="" type="checkbox"/> R / W, the other ; write only							

Note: ☆ : These OSD registers are changed by RGWR.
 ☆☆ : Only lower 2bits are changed by RGWR. (The register in address 0F96H must not be used with any of the read-modify-write instructions such as SET, CLR, etc.)

2.13 Jitter Elimination Circuit

The A8700CH / CK / CM / CP / CS has a built-in jitter elimination circuit which maintains the vertical stability of the OSD even when input of the vertical sync signal fluctuates.

Using the hardware jitter elimination mode enables smoothing for lower-case characters on the OSD without field decision information from the slicer chip.

2.13.1 Configuration

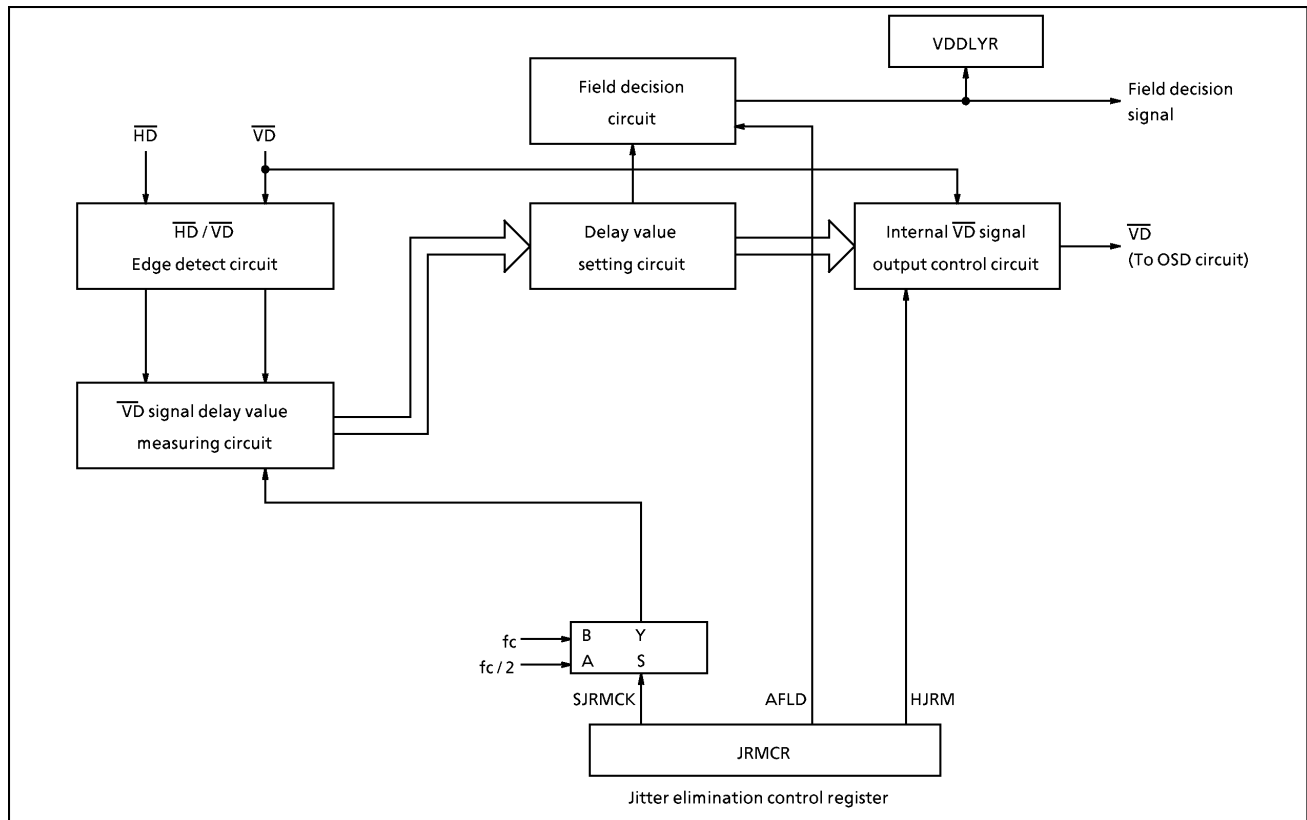


Figure 2-89. Jitter Elimination Circuit

2.13.2 Control

Jitter removal is controlled by the jitter removal control register (JRMCR).

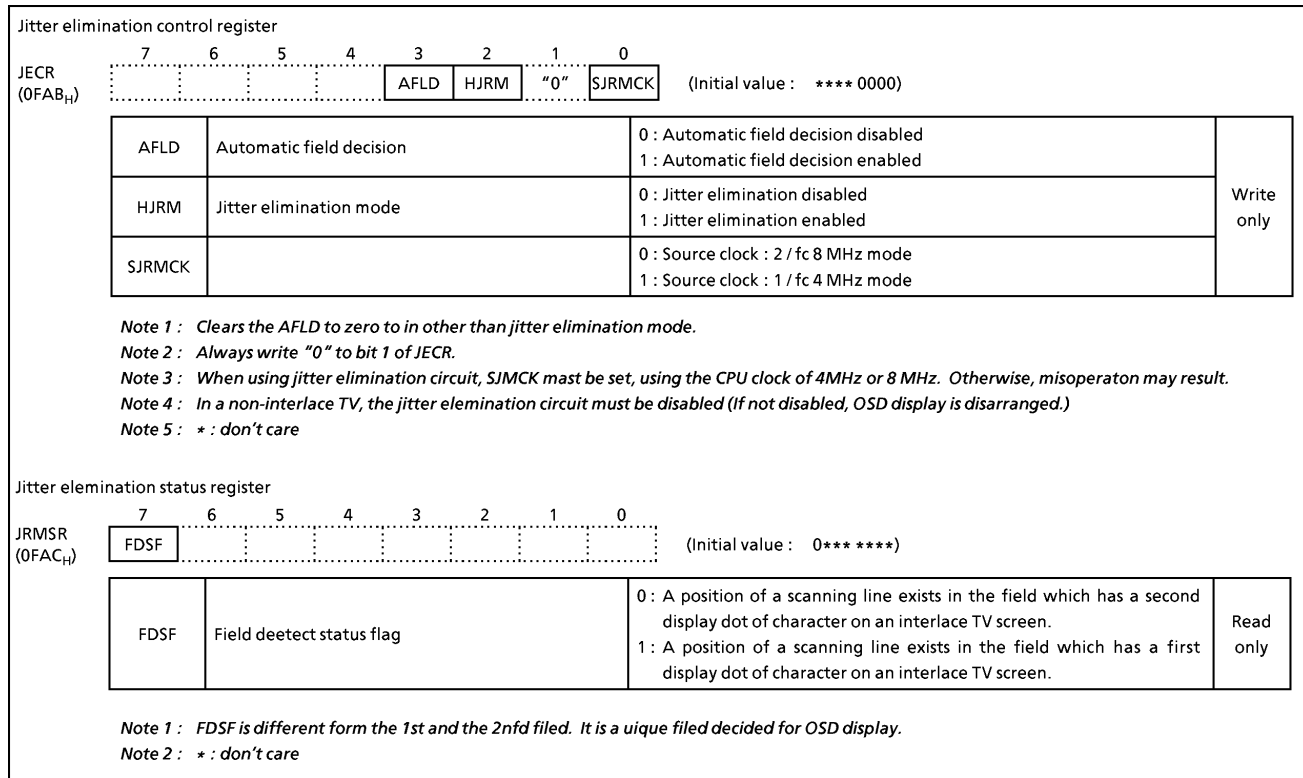


Figure 2-90 Jitter Elimination Control Register and Auto Field Line Reading Register

2.13.3 Jitter Elimination Mode

The hardware jitter elimination mode is to identify the phase of the falling edges of the external \overline{VD} signal and \overline{HD} signal. When \overline{VD} signal is falling within \overline{HD} signal falling $\pm 1/4HD$, the jitter is automatically eliminated and internal \overline{VD} signal is set to the stable location.

When the jitter elimination control register HJRM (2nd bit at 0FABH address) is set to 1, the mode is turned to the jitter elimination mode. When the jitter elimination mode is used, CPU clock has to be used at 8 MHz or 4 MHz. SJRMCK is set to 0 at 8 MHz and SJRMCK is set to 1 at 4 MHz.

2.13.4 Auto Field Line Decision

The internal vertical and horizontal sync signals corrected by the jitter elimination circuit generate the field line decision signals used in the OSD.

The smoothing of small characters of the OSD is achieved using the field line decision signals.

To enable smoothing of small characters, select jitter elimination mode and set AFLD (0FABH, bit 3) to "1".

By reading the field line decision signal, the roll-up function for closed captions can be effected more smoothly. The roll-up function is achieved by changing the display position each frame. In order to make the roll-up display smoother, the field line decision signal is monitored, and the display position changed at the end of the previous field or at the start of the next field.

2.14 Data Slicer

The A8700CH / CK / CM / CP / CS contains the data slicer to decode the caption data multiplied during vertical flyback time of the composite video signal.

The composite video signal is input to the data slicer circuit through P32 (V_{IN1}) and P33 (V_{IN0}). The caption data is decoded from the video signal. The sync signal inputs negative composite video signal to V_{IN0} and V_{IN1} pins. This can comply with the copy guard signal and special signals and receive accurately the caption data under the condition of a weak electrical field or a ghost.

2.14.1 Configuration

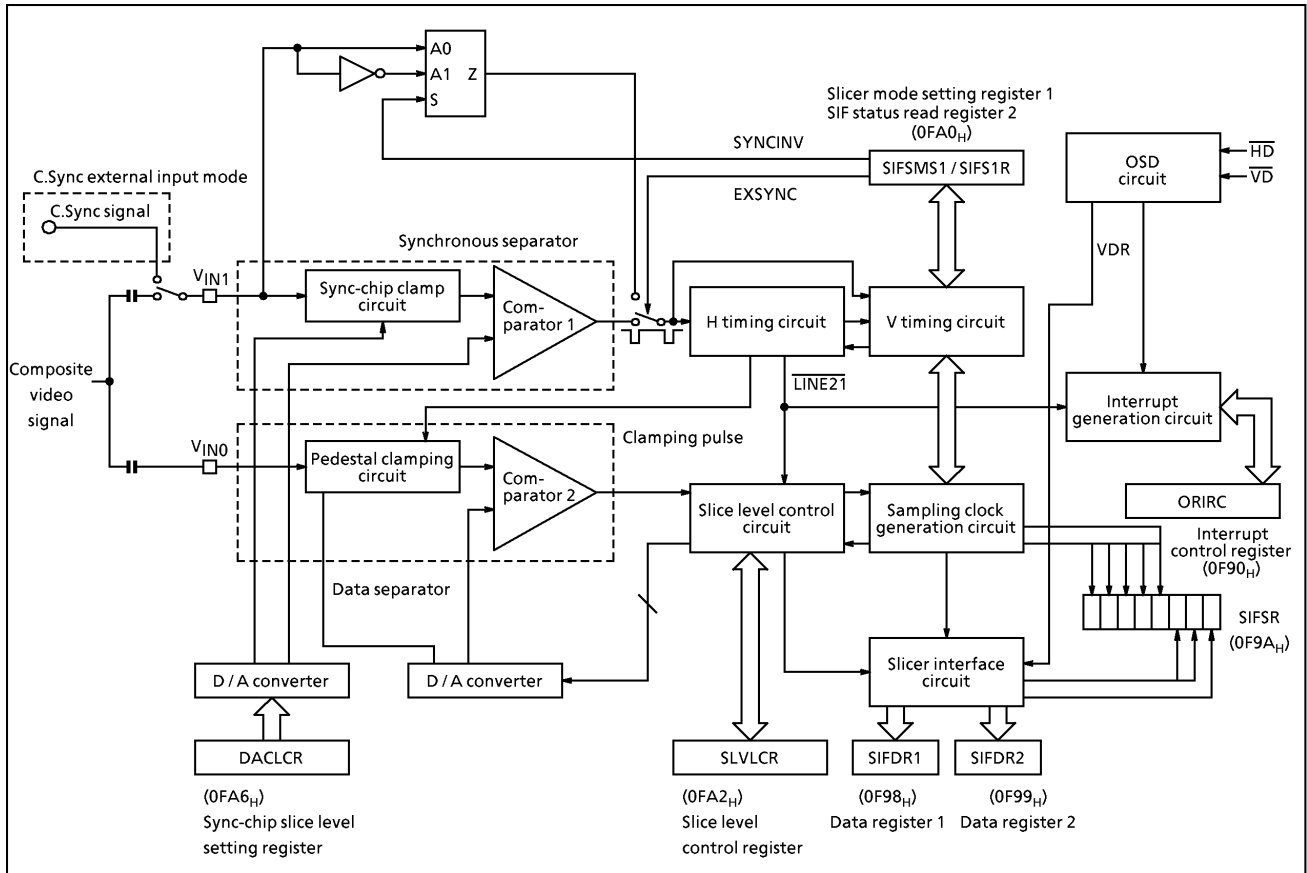


Figure 2-91. Data Slicer

2.14.2 Functions

(1) Video signal input

A low pass filter, a voltage amplifier and a condenser of about $0.1\mu\text{F}$ are connected between the video signal and the video signal input pin of V_{IN1} and V_{IN0} pins, that is shown as Figure 2-93 (a). The low pass filter functions to reduce noise and color burst from the video signal, passes the amplifier and inputs the video signal to both V_{IN1} and V_{IN0} pins.

(2) Synchronous separator

This circuit is to separate the synchronous signal from the video signal. When $\text{DA}7$ to 0 of $\text{DA}7$ to 0 are set for the synchronous separation, the sync slice level is capable of setting. $\text{DA}7$ to 4 set the slice level at the rising edge of the sync signal clamped data, and $\text{DA}3$ to 0 set the slice level at the falling edge of the sync-chip clamped data. (Refer to section 2.14.5)

(3) Data separator

The data separator replaces the caption data piled on the video signal with the digital signal. When $\text{SL}5$ to 0 of $\text{SL}5$ to 0 are set to get the digital signal, the Initial value : of the caption data slice level is capable of setting. (Refer to section 2.14.5)

(4) Sync-chip clamp circuit

The sync-chip level is clamped to the specified value.

(5) Pedestal clamp circuit

The video signal is set to the specified voltage with the clamp pulse generated from the H / V timing part, which is called as a pedestal clamp.

(6) D / A converter

This converter gets the D / A changed slice level of the clamp circuit to the comparator.

(7) Comparator

This comparator replaces the composite video signal with the digital value while inputting to the comparator.

(8) H timing circuit

This circuit detects the horizontal synchronous signal from C.Sync signal separated synchronously from the video signal, and generates the clamp pulse to clamp the video signal and provides it to the pedestal clamp circuit. In addition, the circuit detects the change of H frequency and provides the data to the sampling clock generation part.

(9) V timing circuit

This circuit detects the horizontal synchronous signal from C.Sync signal separated synchronously from the video signal, and provides line 21 detection signal to take out caption signal to the slice level control part.

(10) Slice level control circuit

This circuit detects CRI (clock run in) signal from VIDEO signal with line 21 detection signal generated at H / V timing part after slicing, and controls to the most suitable slice level and takes out the caption data.

(11) Sampling clock generation circuit

This circuit generates the sampling clock which is phase-locked to CRI signal with CRI signal detected at the slice level control part. In addition, the circuit revises the location where the sampling clock generates with H frequency variable data generated at H timing generation part.

(12) Slicer interface circuit

This is a 16 bit serial interface to receive the serial data.

(13) Interrupt generation circuit

Interrupts are generated by a rise in the caption line detection signal.

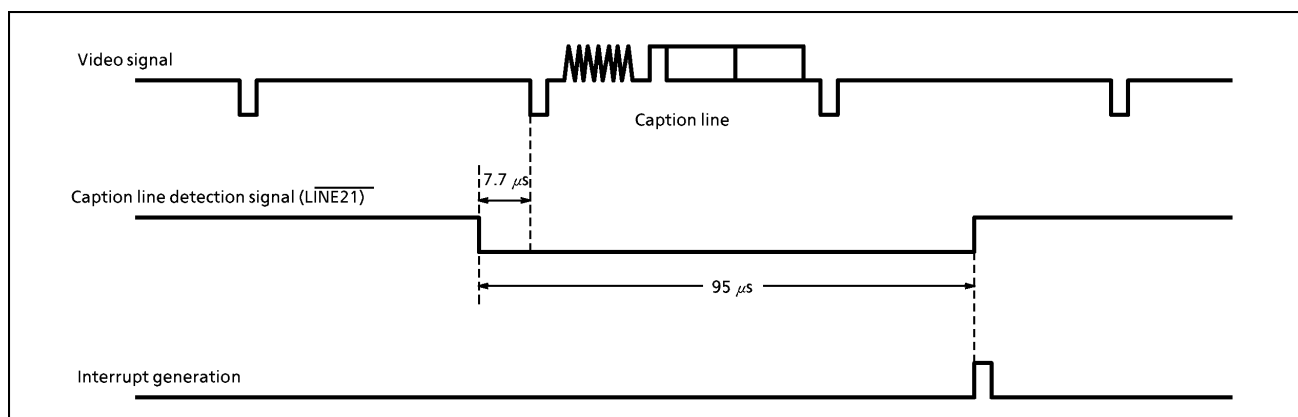


Figure 2-92. Interrupt Generation Timing

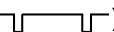

See the description of the on-screen display circuit interrupt vectors for details of interrupt vectors.

(14) C.Sync external input mode

The external C.Sync signal can be used internally by setting EXSYNC (SIFSMS1 bit 5) to "1".

As shown in Figure 2-93 (b), insert a low-pass filter ($f_T = 503$ kHz), voltage amplifier ($\times 2$ voltage amplification), and a capacitor of approximately $0.1 \mu\text{F}$ between the video signal and the video signal input pin V_{IN1} and input an external C.Sync signal to V_{IN0} .

The polarity of the C.Sync signal is selected by SYNCINV (SIFSMS1 bit 6). (Internally used as $\overline{\text{C.Sync}}$.)

CSIN (P32)	SYNCINV
$\overline{\text{C.Sync}}$ ()	"0"
C.Sync ()	"1"

2.14.3 Video Signal Connection

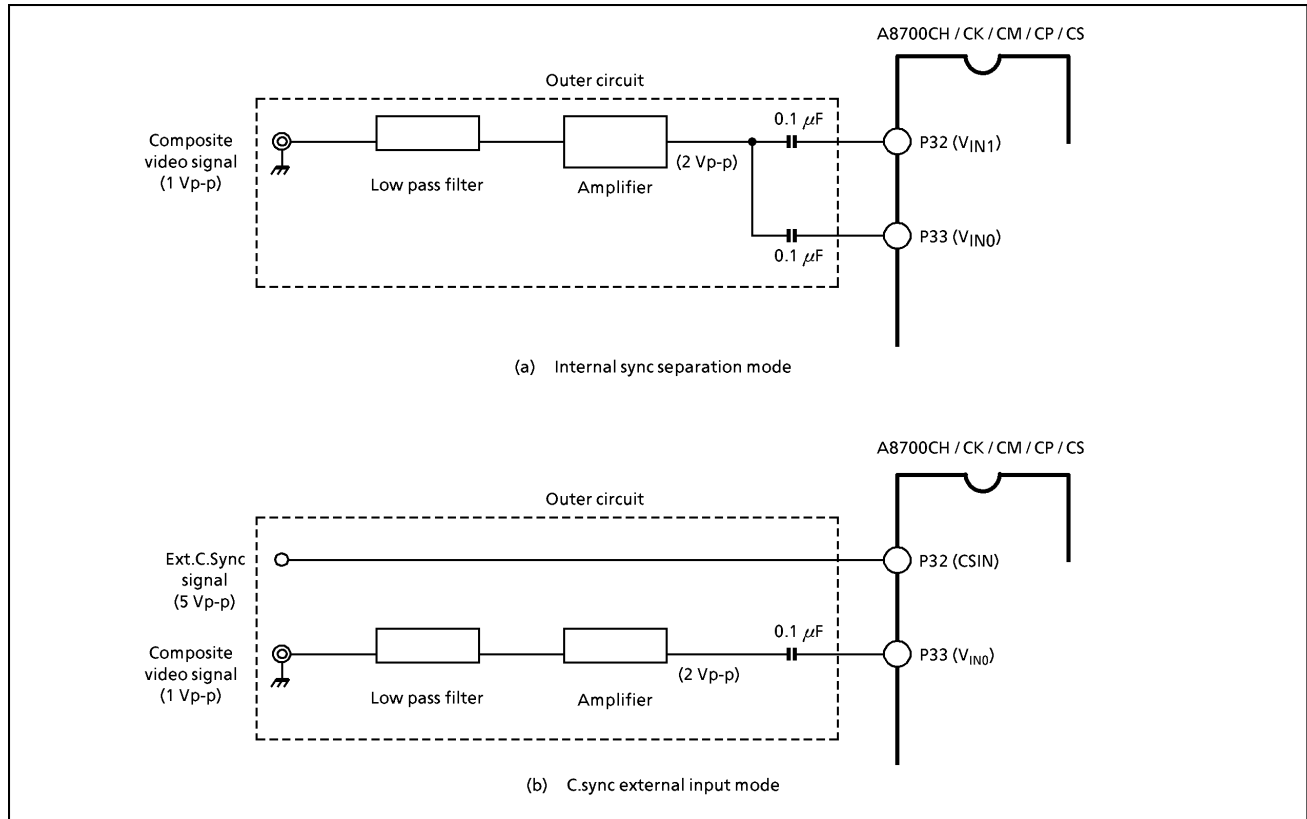


Figure 2-93. Video Signal Connection

2.14.4 Control

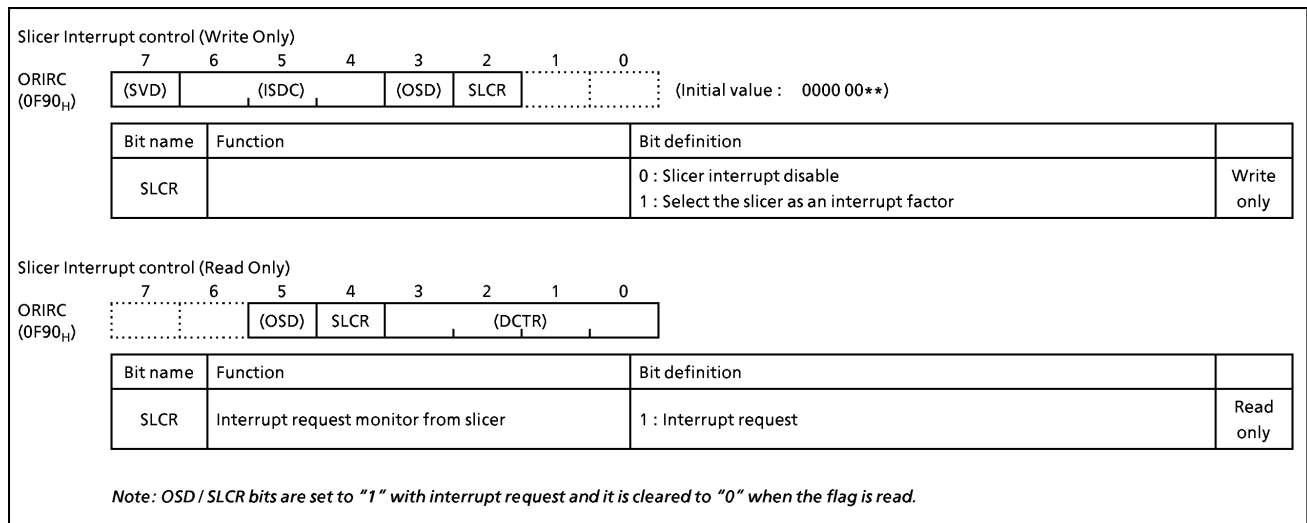


Figure 2-94-1. Data Slicer Control (I)

SIF data register 1 (Caption data 1st byte read register) (Read Only)

7	6	5	4	3	2	1	0
D1ST7	D1ST6	D1ST5	D1ST4	D1ST3	D1ST2	D1ST1	D1ST0

SIFDR1 (0F98_H)

Bit name	Function	Bit definition	
DIST7-0	Caption data 1st byte read register		Read only

SIF data register 2 (Caption data 2nd byte read register) (Read Only)

7	6	5	4	3	2	1	0
D2ND7	D2ND6	D2ND5	D2ND4	D2ND3	D2ND2	D2ND1	D2ND0

SIFDR2 (0F99_H)

Bit name	Function	Bit definition	
D2ND7-0	Caption data 2nd byte read register		Read only

SIF status register (Read Only)

7	6	5	4	3	2	1	0
STCRI	CRIN3	CRIN2	CRIN1	CRIN0	STFLD	STSB	STDE

SIFSR (0F9A_H)

Bit name	Function	Bit definition	
STCRI	Clock run in detection	1 : Clock run in detection 0 : No clock run in detection	Read only
CRIN	CRI number -1	Actual CRI number-1	
STFLD	Field identification	1 : 2nd field 0 : 1st field	
STSB	Start bit identification flag	1 : From detection of star bit until fall in \overline{VD} 0 : Other times	
STDE	16 bit data receive end identification flag	1 : From end of 16 bit data reception until fall in \overline{VD} 0 : Other times	

Figure 2-94-2. Data Slicer Control (II)

Slicer mode setting register 1 (Write Only)

7	6	5	4	3	2	1	0
"0"	SYNC INV	EXSYNC	"1"	CLINE3	CLINE2	CLINE1	CLINE0

SIFSMS1 (0FA0_H) (Initial value : 0001 1011)

Bit name	Function	Bit definition	
SYNCINV	Sync signal input inversion	0 : No inversion 1 : Inversion of C.Sync external input signal	Write only
EXSYNC	Sync signal selection	0 : Internal sync separation 1 : External C.Sync input	
CLINE	Setting lines piled on caption data	0000 : 10 line 0001 : 11 line 0010 : 12 line 0011 : 13 line 0100 : 14 line 0101 : 15 line 0110 : 16 line 0111 : 17 line 1000 : 18 line 1001 : 19 line 1010 : 20 line 1011 : 21 line 1100 : 22 line 1101 : 23 line 1110 : 24 line 1111 : 25 line	

Note: Always write "0" to bit 7 of SIFSMS1 and "1" to bit 4 when writing to SIFSMS1.

Figure 2-94-3. Data Slicer Control (III)

SIF status read register 2

SIFS1R (OFA0_H)

	7	6	5	4	3	2	1	0
			GOODV	FLINE4	FLINE3	FLINE2	FLINE1	FLINE0

Bit name	Function	Bit definition	
GOODV	Monitor signal of synchronization	0 : Out of synchronization (One or more) 1 : V timing synchronizing	
FLINE	Field scanning line (Standard 262.5 = - 1) Two's complement	00000 : 0 263.5 00001 : 1 264.5 00010 : 2 00011 : 3 00100 : 4 00101 : 5 00110 : 6 00111 : 7 01000 : 8 01001 : 9 01010 : 10 01011 : 11 01100 : 12 01101 : 13 01110 : 14 01111 : 15 278.5 10000 : V synchronizing adjustment 10001 : -15 248.5 10010 : -14 10011 : -13 10100 : -12 10101 : -11 10110 : -10 10111 : -9 11000 : -8 11001 : -7 11010 : -6 11011 : -5 11100 : -4 11101 : -3 11110 : -2 261.5 11111 : -1 262.5	Read only

Figure 2-94-4. Data Slicer Control (IV)

The explanation of the monitor signals (GOODV, FLINE) are as follows.

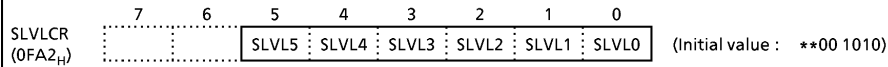
- ① GOODV 0: Data slicer can not synchronize video signal.
1: Data slicer can synchronize video signal.
- ② FLINE The number of filed signal scanning line which the data slicer is detecting or monitor flag of detecting state.

Example

FLINE = 1FH → NTSC Signal

FLINE = 10H → V synchronizing adjustment

Caption data slice level control register (Write / Read)

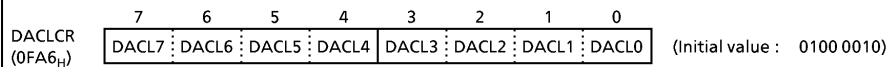


Bit name	Function	Bit definition	
SLVL	Slice level (Initial value :) setting Slice level setting	000000 : VPCLAMP + (1 / 256) V _{DD} 000001 : VPCLAMP + (2 / 256) V _{DD} 000010 : VPCLAMP + (3 / 256) V _{DD} 000011 : VPCLAMP + (4 / 256) V _{DD} 000100 : VPCLAMP + (5 / 256) V _{DD} ⋮ 111101 : VPCLAMP + (62 / 256) V _{DD} 111110 : VPCLAMP + (63 / 256) V _{DD} 111111 : VPCLAMP + (64 / 256) V _{DD}	Write
SLVL	Slice level (Final value)		Read

Note 1 : VPCLAMP (Pedestal clamp) = (1 / 2) V_{DD}

Note 2 : The SLVLCR has different write buffer and read buffer, and cannot be read write buffer data. The SBIDBR cannot be used with any read-modify-write instructions. (Bit manipulation instructions such as SET, CLR, etc. and logical operation such as AND, OR, etc.)

Sync-chip slice level setting register (Write Only)



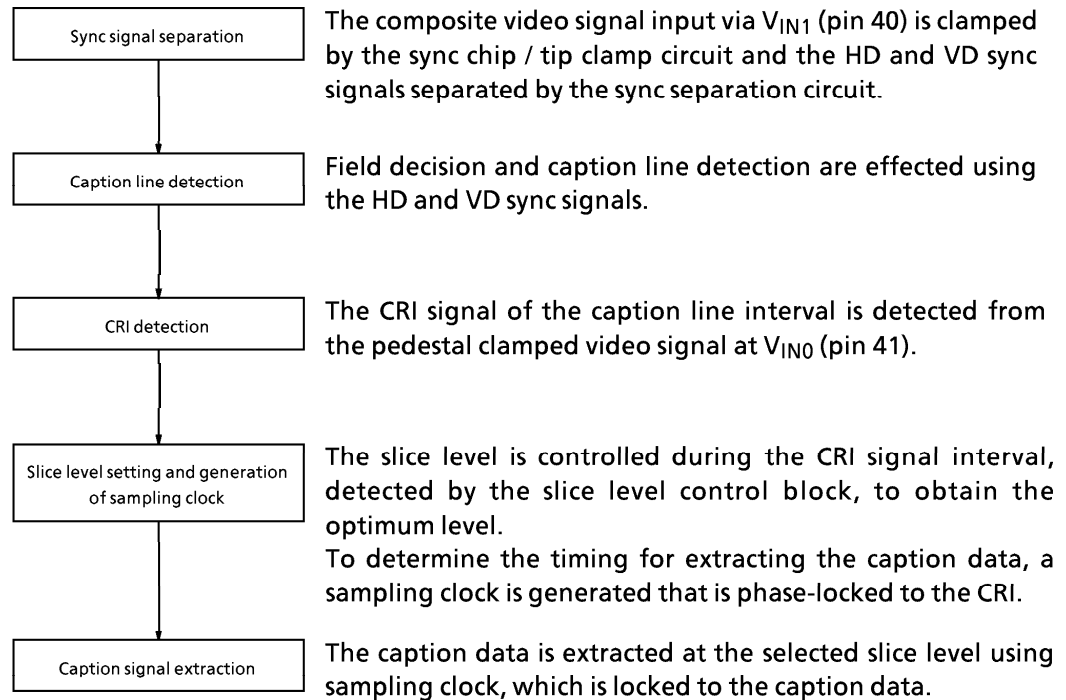
Bit name	Function	Bit definition	
DACL	DACL7 to 4: Slice level Upper limit setting DACL3 to 0: Slice level Lower limit setting	0000 : VSCLAMP + (3 / 512) V _{DD} 0001 : VSCLAMP + (6 / 512) V _{DD} 0010 : VSCLAMP + (9 / 512) V _{DD} 0011 : VSCLAMP + (12 / 512) V _{DD} ⋮ 1101 : VSCLAMP + (42 / 512) V _{DD} 1110 : VSCLAMP + (45 / 512) V _{DD} 1111 : VSCLAMP + (48 / 512) V _{DD}	Write only

Note : VSCLAMP (Sync-chip clamp) = (204 / 512) V_{DD}

Figure 2-94-5. Data Slicer Control (V)

2.14.5 Clamp and Data Slicer Operation

The slicer uses the following steps to obtain the caption signals :



The data slicer has two separation circuits :

- Sync signal (sync chip / tip clamp + sync signal slice) separation.
- Caption data (pedestal clamp + data slice) separation.

The two circuits are described briefly below.

a. Sync signal (sync tip clamp + sync signal slice)

a-1 Sync tip clamp (pin 40) The sync tip is clamped at $(204 / 512) V_{DD}$ [V] as shown in Figure 2-95.

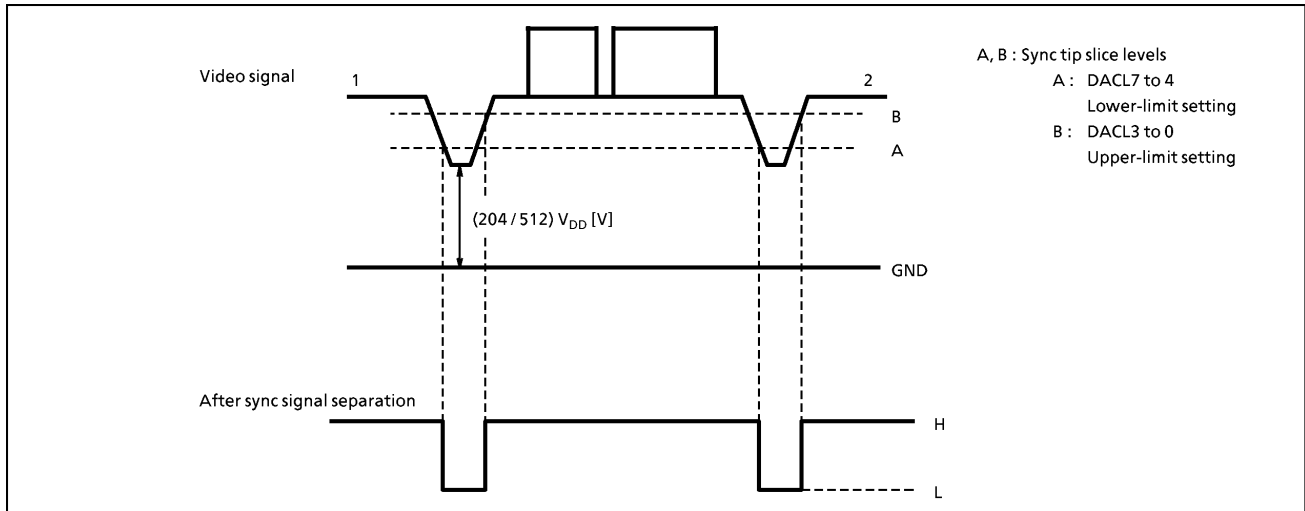


Figure 2-95. Sync Signal Slice

a-2 Method of sync signal slice

The sync signal is separated as shown in Figure 2-95.

Sync signal separation is accomplished by comparing the voltage of the sync tip-clamped video signal with the sync tip slice level. For a 1 → 2 video signal change, if the sync signal after separation is high, the slice level A is selected ; if low, the slice level B is selected.

(Sync tip slice level)

$$\text{Slice level} = \text{VSCLAMP} + \{(3 + 3X) / 512\} V_{DD}$$

V_{DD} : power supply voltage

VSCLAMP : sync tip clamp = $(204 / 512) V_{DD}$

X : setup data (4 bits)

b. Caption data (pedestal clamp + data slice)

b-1 Pedestal clamp (pin 41) Clamped at $(1/2) V_{DD}$ [V] as shown in Figure 2-96.

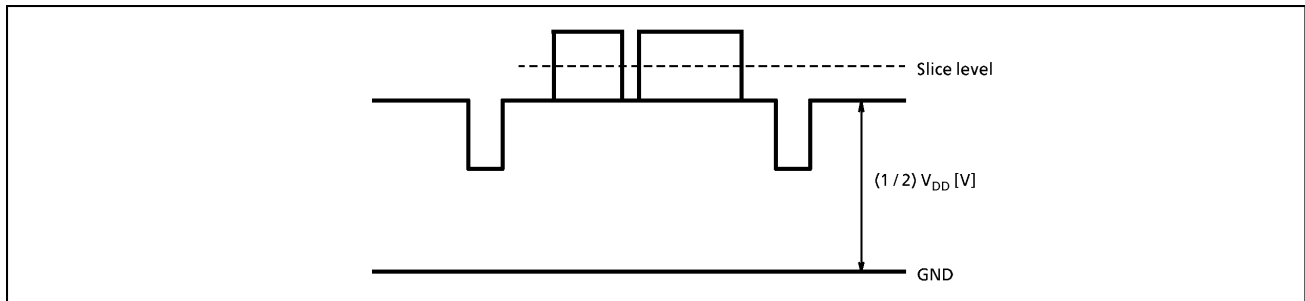


Figure 2-96. Pedestal Clamp

b-2 Method of data slice

The data slice level constitutes a level at which the CCD data is differentiated.

The slice level's setup value is indicated by the following :

$$\text{Slice level} = \text{VPCLAMP} + (X / 256) V_{DD} \text{ [V]}$$

V_{DD} : power supply voltage

VPCLAMP : pedestal clamp = $(1/2) V_{DD}$

X : setup data (6 bits)

b-3 Automatic slice level correction circuit

The slice level is corrected to the appropriate value during the CRI period.

Slice level correction always begins with the setup value of SLVL (bits 5 to 0 of SLVLCR).

If you want the last value to become the initial value of the next slice level, set it to SLVL (bits 5 to 0 of SLVLCR).

2.15 Test video Signal Generator

The A8700CH / CK / CM / CP / CS have a built-in test video signal generation circuit to output necessary signal for TV screen adjustment.

- Mode : NTSC
- Picture pattern : Total eight types, Monochromatic inversion possible
- Output format : Three states (H, L, Hi-Z) output
 - Comp.Sync duration time L output
 - Black level / Pedestal duration time Hi-Z output
 - White level duration time H output

2.15.1 Configuration

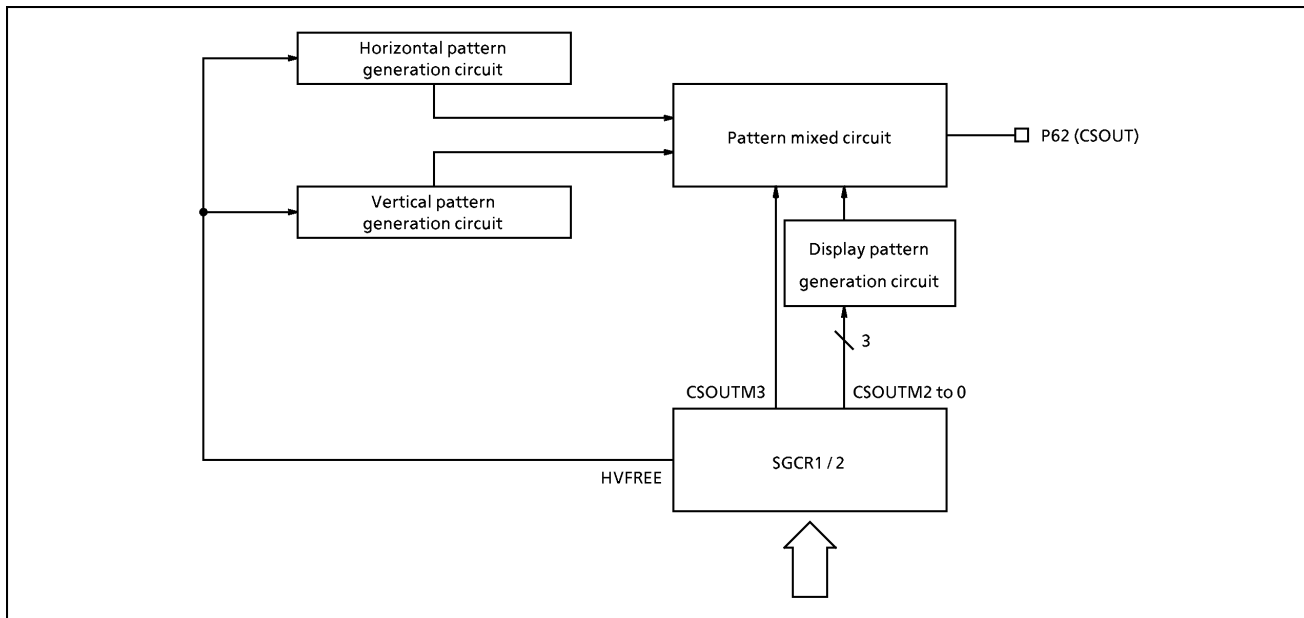


Figure 2-97. Test Video Signal Generation Circuit

2.15.2 Control

The video signal output circuit can be controlled with the signal control register.

Test video signal generator control register 1									
SGCR1 (0FA1 _H)	7	6	5	4	3	2	1	0	(Initial value : *00* *000)
		CSOUT M3	CSOUT M2			CSOUT M1	CSOUT M0	HVFREE	
Bit name	Function	Bit definition							
CSOUTM3	Pattern monochromatic inversion	0 : No inversion 1 : Inversion							
CSOUTM2 to 0	Display pattern	000 : White on the whole screen 001 : Black horizontal hatch 010 : Black vertical hatch 011 : Black cross hatch 100 : White on the upper side and black on the lower 101 : Black cross bar 110 : Black dot 111 : Black cross hatch and dot							Write only
HVFREE	Test video signal output	0 : Disable 1 : Enable							
Test video signal generator control register 2									
SGCR2 (0F9D _H)	7	6	5	4	3	2	1	0	(Initial value : *****0)
								SGCHS	
Bit name	Function	Bit definition							
SGCHS	Synchronous signal for OSD	0 : Port 1 : Test video signal generator							R/W

Figure 2-98. Test Video Signal Generator Control Register

2.15.3 Functions

Test video signal output is to generate monochromatic picture signal output to take easily the necessary tests such as TV screen white adjustment and screen distortion amplitude adjustment implemented on the final manufacturing process of a TV receiver set.

When CSOUT2 to 0 of the test video signal generator control register 1 (5th, 2nd, 1st bit at 0FA1_H address) is set, the display pattern is selected. When CSOUTM3 (6th bit at 0FA1_H address) is set to 1, the pattern is inverted. When HVFREE (0th bit at 0FA1_H address) is set to 1, the pattern is generated.

When the on screen display and the display pattern of test video signal are displayed at the same time, SGCHS of the test video signal generator control register 2 (0th bit at 0F9D_H address) is set to 1, and the synchronous signal generated in the test video signal generator is also used for the OSD circuit. SGCHS has to be clear to 0 when the test video signal generator is not enabled.


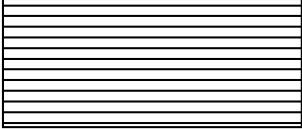

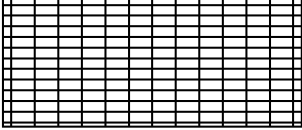
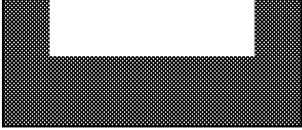
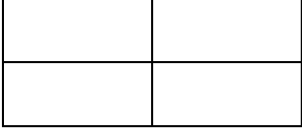
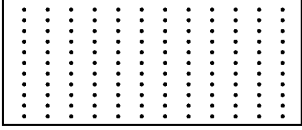
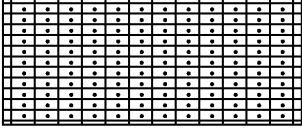
Display Pattern (Csoutm2 to 0)	TV Screen
000 (White on the whole screen)	
001 (Black horizontal hatch)	
010 (Black vertical hatch)	
011 (Black cross hatch)	
100 (White on the upper side and black on the lower side)	
101 (Black cross bar)	
110 (Black dot)	
111 (Black cross hatch and dot)	

Figure 2-99. Display Pattern and TV Screen

There are three states of the output to generate picture signal with the external circuit of the resistance divided voltage.

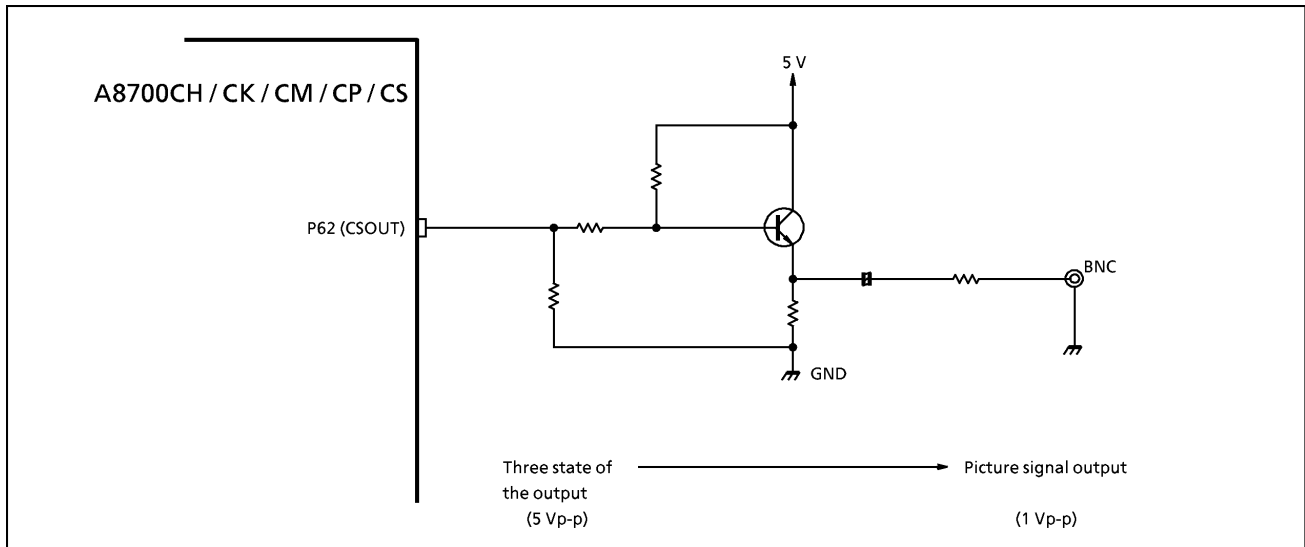


Figure 2-100. Example of Picture Output Generation

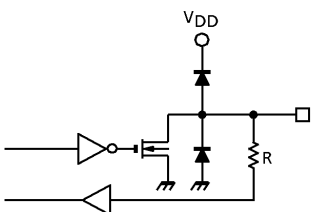
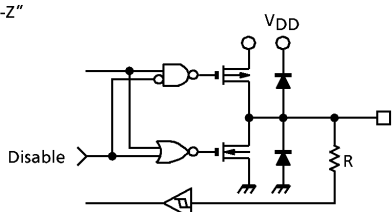
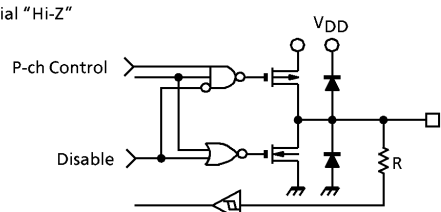
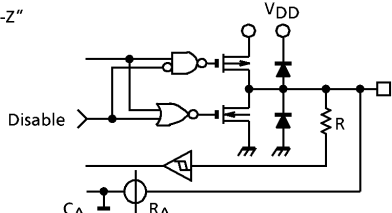
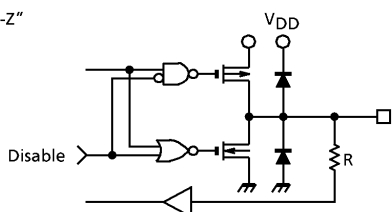
Input / Output Circuitry

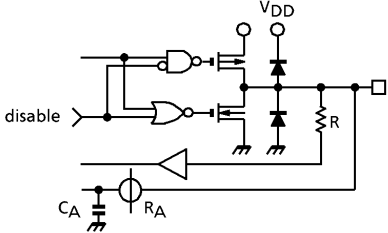
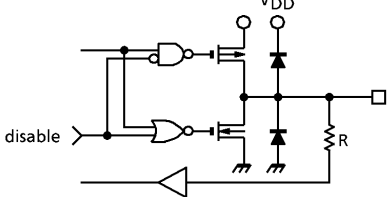
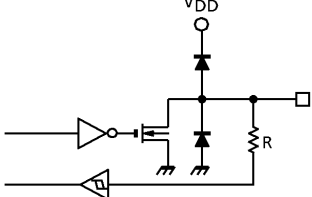
(1) Control pins

The input / output circuitries of the A8700CH / CK / CM / CP / CS control pins are shown below.

Control Pin	I / O	Input / Output Circuitry	Remarks
XIN XOUT	Input Output		Resonator connecting pins (High-frequency) $R_f = 1.2 \text{ M}\Omega$ (Typ.) $R_O = 1.5 \text{ k}\Omega$ (Typ.) $R = 1 \text{ k}\Omega$ (Typ.)
$\overline{\text{RESET}}$	I / O		Sink open drain output Hysteresis input Pull-up resistor $R_{IN} = 220 \text{ k}\Omega$ (Typ.)
$\overline{\text{STOP}} / \overline{\text{INT5}}$ (P20)	Input		Hysteresis input $R = 1 \text{ k}\Omega$ (Typ.)
TEST	Input		Pull-down resistor $R_{IN} = 70 \text{ k}\Omega$ (Typ.) $R = 1 \text{ k}\Omega$ (Typ.)
OSC1 OSC2	Input Output		Osc. connecting pin for on- screen display $R_f = 1.2 \text{ M}\Omega$ (Typ.) $R_O = 1.5 \text{ k}\Omega$ (Typ.) $R = 1 \text{ k}\Omega$ (Typ.)

(2) Input/output ports

Port	I/O	Input / Output Circuitry	Remarks
P20	I/O	Initial "Hi-Z" 	Sink open drain output R = 1 kΩ (Typ.)
P30 to P33 P50 P57	I/O	Initial "Hi-Z" 	Tri-state I/O Hysteresis input R = 1 kΩ (Typ.)
P34 P35 P51 P52	I/O	Initial "Hi-Z" 	Tri-state I/O Hysteresis input R = 1 kΩ (Typ.)
P53 to P56	I/O	Initial "Hi-Z" 	Tri-state I/O Hysteresis input R = 1 kΩ (Typ.) RA = 5 kΩ (Typ.) CA = 12 pF (Typ.)
P4 P64 to P67	I/O	Initial "Hi-Z" 	Tri-state I/O R = 1 kΩ (Typ.)

Port	I/O	Input / Output Circuitry	Remarks
P60 P61	I/O	<p>Initial "Hi-Z"</p> 	<p>Tri-state I/O High current output $I_{OL} = 20 \text{ mA (Typ.)}$</p> <p>$R = 1 \text{ k}\Omega \text{ (Typ.)}$ $R_A = 5 \text{ k}\Omega \text{ (Typ.)}$ $C_A = 12 \text{ pF (Typ.)}$</p>
P62 P63	I/O		<p>Tri-state I/O High current output $I_{OL} = 20 \text{ mA (Typ.)}$</p> <p>$R = 1 \text{ k}\Omega \text{ (Typ.)}$</p>
P70 P71	I/O		<p>Sink open drain output Hysteresis input</p> <p>$R = 1 \text{ k}\Omega$</p>

Electrical Characteristics

Absolute Maximum Ratings

 $(V_{SS} = 0\text{ V})$

Parameter	Symbol	Condition	Rating	Unit
Supply Voltage	V_{DD}	—	– 0.3 to 6.5	V
Input Voltage	V_{IN}	—	– 0.3 to $V_{DD} + 0.3$	V
Output Voltage	V_{OUT1}	—	– 0.3 to $V_{DD} + 0.3$	V
Output Current (Per 1pin)	I_{OUT1}	Ports P2, P3, P4, P5, P64 to P67, P7	3.2	mA
	I_{OUT2}	P60 to P63	30	
Output Current (Total)	ΣI_{OUT1}	Ports P2, P3, P4, P5, P64 to P67, P7	120	mA
	ΣI_{OUT2}	P60 to P63	120	
Power Dissipation [$T_{opr} = 70^{\circ}\text{C}$]	P_D	—	600	mW
Soldering Temperature (time)	T_{sld}	—	260 (10 s)	$^{\circ}\text{C}$
Storage Temperature	T_{stg}	—	– 55 to 125	$^{\circ}\text{C}$
Operating Temperature	T_{opr}	—	– 30 to 70	$^{\circ}\text{C}$

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, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded

Recommended Operating Conditions	($V_{SS} = 0\text{ V}$, $T_{opr} = -30\text{ to }70^{\circ}\text{C}$)
----------------------------------	--

Parameter	Symbol	Pins	Test Condition	Min	Max	Unit	
Supply Voltage	V_{DD}	—	$f_c = 8\text{ MHz}$	NORMAL mode	4.5	5.5	V
				IDLE mode			
				STOP mode	2.0		
Input High Voltage	V_{IH1}	Except hysteresis input	$V_{DD} \geq 4.5\text{ V}$	$V_{DD} \times 0.70$	V_{DD}	V	
	V_{IH2}	Hysteresis input		$V_{DD} \times 0.75$			
	V_{IH3}	—	$V_{DD} < 4.5\text{ V}$	$V_{DD} \times 0.90$			
Input Low Voltage	V_{IL1}	Except hysteresis input	$V_{DD} \geq 4.5\text{ V}$	0	$V_{DD} \times 0.30$	V	
	V_{IL2}	Hysteresis input			$V_{DD} \times 0.25$		
	V_{IL3}	—	$V_{DD} < 4.5\text{ V}$		$V_{DD} \times 0.10$		
Clock Frequency	f_c	XIN, XOUT	$V_{DD} = 4.5\text{ to }5.5\text{ V}$	—	8.0	MHz	
	f_{OSD}	OSC1, OSC2	$V_{DD} = 4.5\text{ to }5.5\text{ V}$	Double frequency mode (FORS = 1)	2		6
					Normal frequency mode (FORS = 0)	2	12

Note 1: The recommended operating conditions for a device are operating conditions under which it can be guaranteed that the device will operate as specified. If the device is used under operating conditions other than the recommended operating conditions (supply voltage, operating temperature range, specified AC/DC values etc.), malfunction may occur. Thus, when designing products which include this device, ensure that the recommended operating conditions for the device are always adhered to.

Note 2: f_c : The condition of power supply voltage is limited to NORMAL and IDLE mode.

Furthermore, since the CPU clock serves dual purposes as a clock for the CCD slier, always be sure to use an 8 MHz oscillator.

Electrical Characteristics

D.C. Characteristics

 ($V_{SS} = 0\text{ V}$, $T_{opr} = -30\text{ to }70^{\circ}\text{C}$)

Parameter	Symbol	Pins	Test Circuit	Test Condition	Min	Typ.	Max	Unit
Hysteresis Voltage	V_{HS}	Hysteresis inputs	—	—	—	0.9	—	V
Input Current	I_{IN1}	TEST	—	$V_{DD} = 5.5\text{ V}$, $V_{IN} = 5.5\text{ V} / 0\text{ V}$	—	—	± 2	μA
	I_{IN2}	Open drain ports	—	$V_{DD} = 5.5\text{ V}$, $V_{IN} = 5.5\text{ V} / 0\text{ V}$	—	—	± 2	
	I_{IN3}	Tri-state ports	—	$V_{DD} = 5.5\text{ V}$, $V_{IN} = 5.5\text{ V} / 0\text{ V}$	—	—	± 2	
	I_{IN4}	RESET, STOP						
Input Resistance	R_{IN2}	RESET	—	—	100	220	450	$\text{k}\Omega$
Output Leakage Current	I_{LO}	Open drain ports and tri-state ports	—	$V_{DD} = 5.5\text{ V}$, $V_{OUT} = 5.5\text{ V}$	—	—	2	μA
Output High Voltage	V_{OH2}	Tri-state ports	—	$V_{DD} = 4.5\text{ V}$, $I_{OH} = -0.7\text{ mA}$	4.1	—	—	V
Output Low Voltage	V_{OL}	Except XOUT, OSC2, P60 to 63	—	$V_{DD} = 4.5\text{ V}$, $I_{OL} = 1.6\text{ mA}$	—	—	0.4	V
Output Low Current	I_{OL3}	P60 to P63	—	$V_{DD} = 4.5\text{ V}$, $V_{OL} = 1.0\text{ V}$	—	20	—	mA
Supply Current in NORMAL Mode	I_{DD}	—	—	$V_{DD} = 5.5\text{ V}$ (Note 3) $f_c = 8\text{ MHz}$ $V_{IN} = 5.3\text{ V} / 0.2\text{ V}$	—	13	25	mA
Supply Current in IDLE Mode					—	8	18	mA
Supply Current in STOP Mode					—	0.5	10	μA

Note 1: Typical values show those at $T_{opr} = 25^{\circ}\text{C}$, $V_{DD} = 5\text{ V}$.

Note 2: Input Current : The current through pull-up or pull-down resistor is not included.

Note 3: Typical current consumption during A / D conversion is 1.2 mA.

A/D Conversion Characteristics

 ($V_{SS} = 0\text{ V}$, $V_{DD} = 4.5\text{ to }5.5\text{ V}$, $T_{opr} = -30\text{ to }70^{\circ}\text{C}$)

Parameter	Symbol	Test Circuit	Test Condition	Min	Typ.	Max	Unit
Analog Reference Voltage	V_{AREF}	—	—	—	V_{DD}	—	V
	V_{ASS}		—	—	0		
Analog Reference Voltage Range	ΔV_{AREF}	—	$= V_{DD} - V_{SS}$	—	V_{DD}	—	V
Analog Input Voltage	V_{AIN}	—	—	V_{SS}	—	V_{DD}	V
Nonlinearity Error	—	—	$V_{DD} = 4.5\text{ to }5.5\text{ V}$	—	—	± 1	LSB
Zero Point Error	—	—		—	—	± 2	LSB
Full Scale Error	—	—		—	—	± 2	LSB
Total Error	—	—		—	—	± 3	LSB

Note: Total error does not include quantization error.

A.C. Characteristics

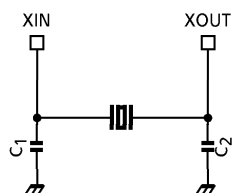
($V_{SS} = 0\text{ V}$, $V_{DD} = 4.5\text{ to }5.5\text{ V}$, $T_{opr} = -30\text{ to }70^\circ\text{C}$)

Parameter	Symbol	Test Circuit	Test Condition	Min	Typ.	Max	Unit
Machine Cycle Time	tcy	—	In NORMAL mode	—	0.5	—	μs
			In IDLE mode				
High Level Clock Pulse Width	t _{WCH}	—	For external clock operation (XIN input), f _c = 8 MHz	62.5	—	—	ns
Low Level Clock Pulse Width	t _{WCL}						

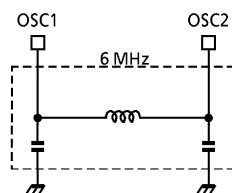
Recommended Oscillating

($V_{SS} = 0\text{ V}$, $V_{DD} = 4.5\text{ to }5.5\text{ V}$, $T_{opr} = -30\text{ to }70^\circ\text{C}$)

Parameter	Oscillator	Frequency	Recommended Oscillator		Recommended Conditions	
					C ₁	C ₂
High-frequency	Ceramic Resonator	8 MHz	KYOCERA	KBR8.0M	30 pF	30 pF
	Crystal Oscillator	8 MHz	TOYOCOM	210B 8.0000	20 pF	20 pF
OSD	LC Resonator	6 MHz	TOKO	A285HCIS-13319 (5 mm)	—	—
		12 MHz	TOKO	TA285HCIS-13306 (5 mm)		



(1) High-frequency



(2) LC Resonator for OSD

Note: An electrical shield by metal shield plate on the surface of the IC package should be recommendable in order to prevent the device from the high electric fieldstress applied from CRT (Cathode Ray Tube) for continuous reliable operation.