CMOS 8-Bit Microcontroller

# TMP88CS38NG/FG, TMP88CM38ANG/F, TMP88CP38ANG/F

The TMP88CS38/CM38A/CP38A is the high speed and high performance 8-bit single chip microcomputers. This MCU contain CPU core, ROM, RAM, input/output ports, four multi-function timer/counters, serial bus interface, on-screen display, PWM output, 8-bit AD converter, and remote control signal preprocessor on chip.

| Product No. | ROM | RAM | Package | OTP MCU |
|---|---|---|---|---|
| TMP88CS38NG/FG | 64 K × 8 bits | 2 K × 8 bits | P-SDIP42-600-1.78<br>P-QFP44-1414-0.80K | TMP88PS38NG/FG |
| TMP88CM38ANG/F | 32 K × 8 bits | 1.5 K × 8 bits | | |
| TMP88CP38ANG/F | 48 K × 8 bits | | | |

## Features

- 8-bit single chip microcomputer TLCS-870/X series
- Instruction execution time: 0.25 μs (at 16 MHz)
- 842 basic instructions
  - Multiplication and division (8 bits × 8 bits, 16 bits × 8 bits, 16 bits/8 bits)
  - Bit manipulations (Set/clear/complement/move/test/exclusive or)
  - 16-bit data and 20-bit data operations
  - 1-byte jump/subroutine call (Short relative jump/vector call)


**RESTRICTIONS ON PRODUCT USE**

20070701-EN

- The information contained herein is subject to change without notice.
- TOSHIBA is continually working to improve the quality and 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 comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products 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 TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the “Handling Guide for Semiconductor Devices,” or “TOSHIBA Semiconductor Reliability Handbook” etc.
- The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury (“Unintended Usage”). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc.. Unintended Usage of TOSHIBA products listed in his document shall be made at the customer’s own risk.
- The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations.
- The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patents or other rights of TOSHIBA or the third parties.
- Please contact your sales representative for product-by-product details in this document regarding RoHS compatibility. Please use these products in this document in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances. Toshiba assumes no liability for damage or losses occurring as a result of noncompliance with applicable laws and regulations.

- ♦ I/O ports: Maximum 33 (High current output: 4)
- ♦ 17 interrupt sources: External 6, internal 11
  - All sources have independent latches each, and nested interrupt control is available.
  - Edge-selectable external interrupts with noise reject
  - High-speed task switching by register bank changeover
- ♦ ROM corrective function
- ♦ Two 16-bit timer/counters: TC1, TC2
  - Timer, event counter, pulse width measurement, external trigger timer, window modes
- ♦ Two 8-bit timer/counters: TC3, TC4
  - Timer, event counter, capture (Pulse width/duty measurement) mode
- ♦ Time base timer (Interrupt frequency: 0.95 Hz to 31250 Hz)
- ♦ Watchdog timer
  - Interrupt sourse/reset output
- ♦ Serial bus interface
  - $I^2C$ bus, 8-bit SIO mode (Selectable two I/O channels)
- ♦ On-screen display circuit
  - Font ROM characters: 384 characters
  - Characters display: 32 columns × 12 lines
  - Composition: 16 × 18 dots
  - Size of character: 3 kinds (Line by line)
  - Color of character: 8 or 15 kinds (Character by character)
  - Variable display position: Horizontal 256 steps, vertical 512 steps
  - Fringing, smoothing, slant, underline, blinking function
- ♦ Jitter elimination
- ♦ Data slicer circuit 1 channel
- ♦ DA conversion (Pulse width modulation) outputs
  - 14- or 12-bit resolution (2 channels)
  - 12-bit resolution (2 channels)
  - 7-bit resolution (6 channels)
- ♦ 8-bit successive approximate type AD converter with sample and hold
- ♦ Remote control signal preprocessor
- ♦ Two power saving operating modes
  - STOP mode: Oscillation stops. Battery/capacitor backup. Port output hold/high impedance.
  - IDLE mode: CPU stops, and peripherals operate using high-frequency clock. Release by interrupts.

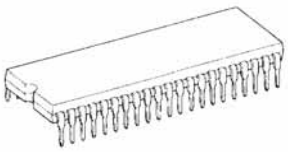
♦Operating voltage: 4.5 to 5.5 V at 16 MHz

♦Emulation POD: BM88CS38N0A-M15

# Pin Assignments

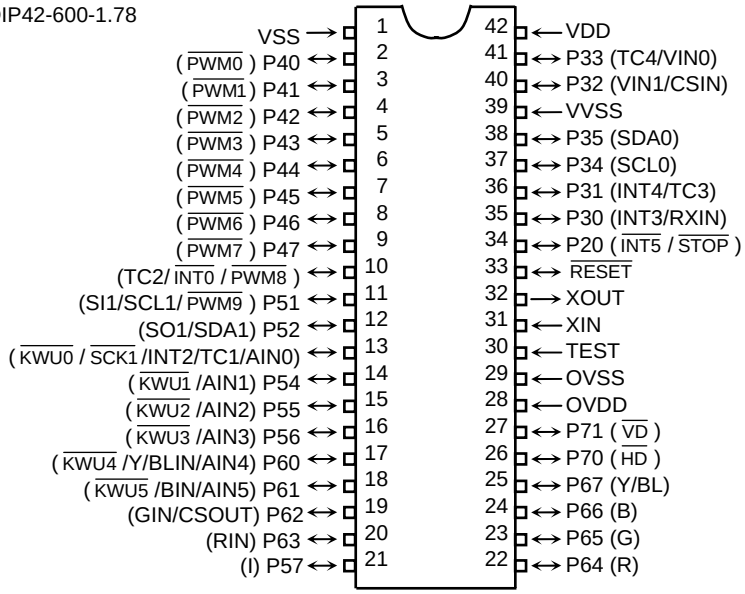
Package

P-SDIP42-600-1.78



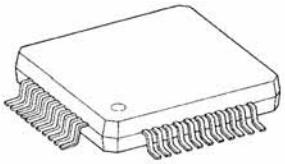
TMP88CS38NG
TMP88CP38ANG
TMP88CM38ANG
TMP88PS38NG

P-SDIP42-600-1.78



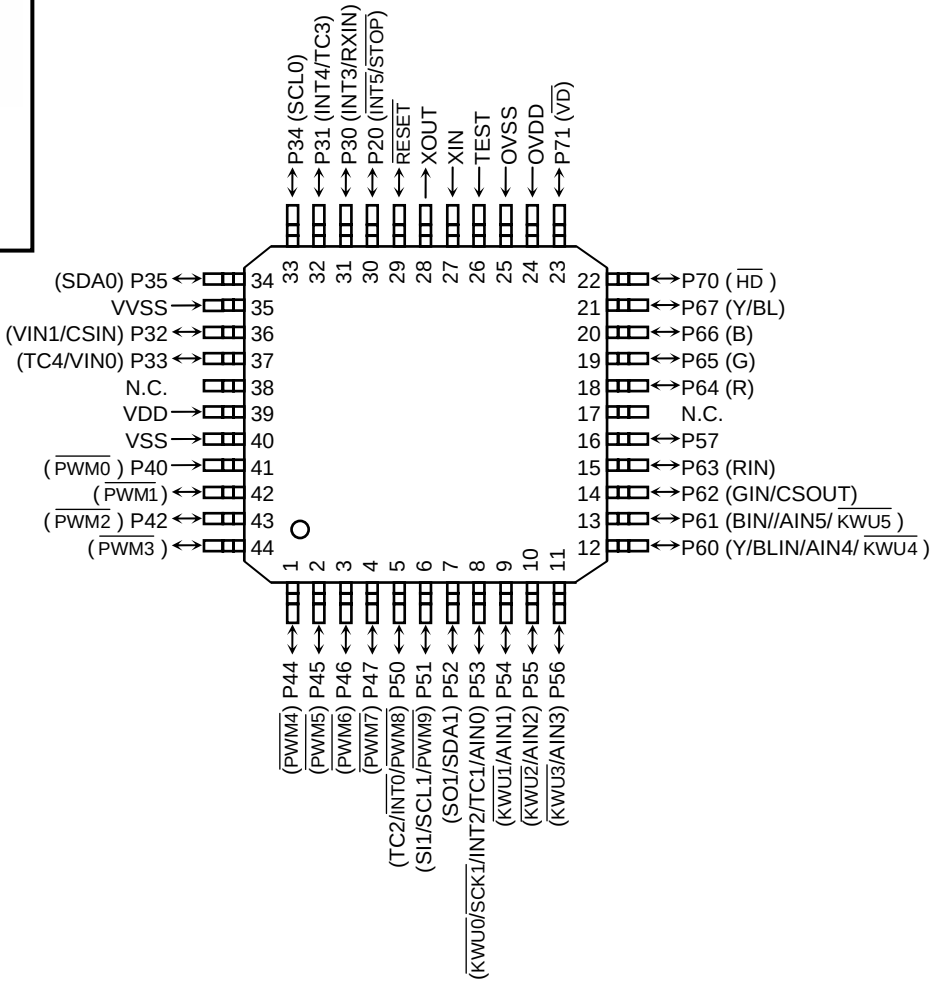

Package

P-QFP44-1414-0.80K



TMP88CS38FG
TMP88CP38AF
TMP88CM38AF
TMP88PS38FG

P-QFP44-1414-0.80K



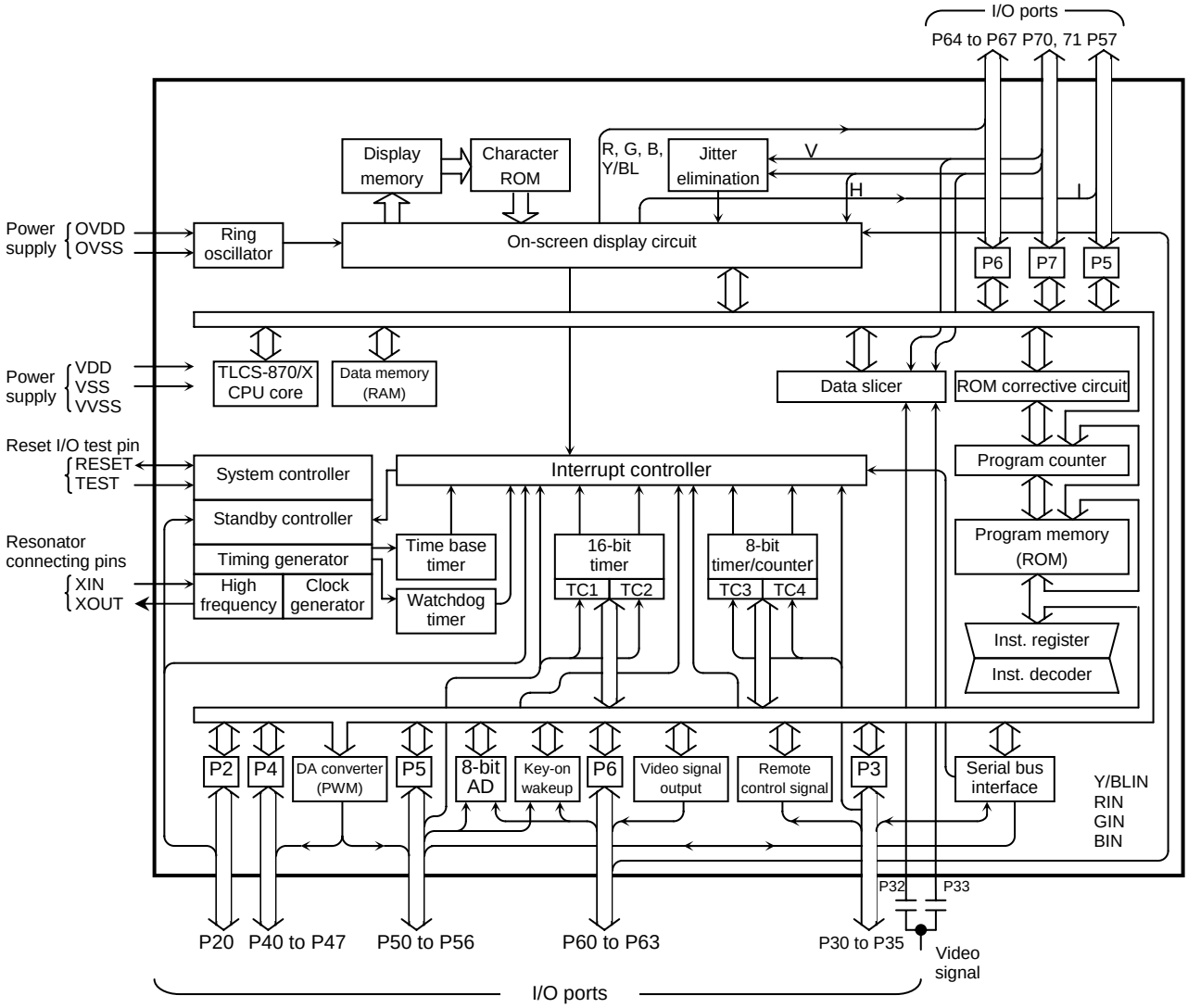
# Pin Functions (1/2)

| Pin Name | I/O | Function | |
|---|---|---|---|
| P20 ( $\overline{\text{INT5}}$ / $\overline{\text{STOP}}$ ) | I/O (Input) | 1-bit input/output port with latch. When used as an input port, the latch must be set to “1”. | External interrupt input 5 or STOP mode release signal input |
| P35 (SDA0) | I/O (Input/Output) | 6-bit programmable input/output port. Each bit of these ports can be individually configured as an input or an output under software control. During reset, all bits are configured as inputs. When used as a serial bus interface input/output, the latch must be set to “1”. | $I^2C$ bus serial data input/output 0 |
| P34 (SCL0) | I/O (Input/Output) | | $I^2C$ bus serial clock input/output 0 |
| P33 (TC4/VIN0) | I/O (Input) | | Timer counter input 4 or video signal Input 0 |
| P32 (VIN1/CSIN) | I/O (Input) | | Video signal input 1 or composite sync input |
| P31 (INT4/TC3) | I/O (Input) | | External interrupt input 4 or timer counter input 3 |
| P30 (INT3/RXIN) | I/O (Input) | | External interrupt input 3 or remote control signal preprocessor input |
| P47 ( $\overline{\text{PWM7}}$ ) | I/O (Output) | 8-bit programmable input/output port. Each bit of these ports can be individually configured 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”. | 7-bit DA conversion (PWM) outputs |
| P46 ( $\overline{\text{PWM6}}$ ) | I/O (Output) | | |
| P45 ( $\overline{\text{PWM5}}$ ) | I/O (Output) | | |
| P44 ( $\overline{\text{PWM4}}$ ) | I/O (Output) | | |
| P43 ( $\overline{\text{PWM3}}$ ) | I/O (Output) | | 12-bit DA conversion (PWM) outputs |
| P42 ( $\overline{\text{PWM2}}$ ) | I/O (Output) | | |
| P41 ( $\overline{\text{PWM1}}$ ) | I/O (Output) | | 14/12-bit DA conversion (PWM) outputs |
| P40 ( $\overline{\text{PWM0}}$ ) | I/O (Output) | | |
| P57 (I) | I/O (Output) | 8-bit programmable input/output port. Each bit of these ports can be individually configured as an input or an output under software control. During reset, all bits are configured as inputs. When used as a PWM output, a serial bus interface input/output, the latch must be set to “1”. | Translucent signal output |
| P56 ( $\overline{\text{KWU3}}$ /AIN3) | I/O (Input) | | Key-on wakeup inputs or AD converter analog inputs |
| P55 ( $\overline{\text{KWU2}}$ /AIN2) | I/O (Input) | | |
| P54 ( $\overline{\text{KWU1}}$ /AIN1) | I/O (Input) | | |
| P53 ( $\overline{\text{KWU0}}$ /AIN0/TC1 /INT2/ $\overline{\text{SCK1}}$ ) | I/O (Input /Input/Input /Input/Output) | | Key-on wakeup input or AD converter analog input or timer counter input 1 or external interrupt input 2 or SIO serial clock input/output 1 |
| P52 (SDA1/SO1) | I/O (Input/Output /Output) | | $I^2C$ bus serial data input/output 1 or SIO serial data output 1 |
| P51 ( $\overline{\text{PWM9}}$ /SCL1/SI1) | I/O (Output/Input/Output /Input) | | 7-bit DA conversion (PWM) output or $I^2C$ bus serial data input/output 1 or SIO serial data input 1 |
| P50 ( $\overline{\text{PWM8}}$ /TC2/ $\overline{\text{INT0}}$ ) | I/O (Output/Input /Input) | | 7-bit DA conversion (PWM) output or timer counter input 2 or external interrupt input 0 |
| P67 (Y/BL) | I/O (Output) | 8-bit programmable input/output port. (P67 to P64: Tri-State, P63 to P60: High current output) Each bit of these ports can be individually configured as an input or an output under software control. During reset, all bits are configured as inputs. When used P64 to P67 as port, each bit of the P6 port data selection register (Bit7 to 4 in ORP6S) must be set to “1”. | Y or BL output |
| P66 (B) | I/O (Output) | | R/G/B outputs |
| P65 (G) | I/O (Output) | | |
| P64 (R) | I/O (Output) | | |
| P63 (RIN) | I/O (Input) | | R input |
| P62 (GIN/CSOUT) | I/O (Input/Output) | | G input or TEST video signal output |
| P61 ( $\overline{\text{KWU5}}$ /BIN/AIN5) | I/O (Input) | | Key-on wakeup input 5 or B input or AD converter analog input 5 |
| P60 ( $\overline{\text{KWU4}}$ /YBLIN/AIN4) | I/O (Input) | | Key-on wakeup input 4 or Y/BL input or AD converter analog input 4 |

# Pin Functions (2/2)

| Pin Name | I/O | Function | |
|---|---|---|---|
| P71 ( $\overline{VD}$ ) | I/O (Input) | 2-bit programmable input/output port. Each bit of these ports can be individually configured as an input or an output under software control. During reset, all bits are configured as inputs. | Vertical synchronous signal input |
| P70 ( $\overline{HD}$ ) | I/O (Input) | | Horizontal synchronous signal input |
| XIN, XOUT | Input, Output | Resonator connecting pins. For inputting external clock, XIN is used and XOUT is opened. | |
| RESET | I/O | Reset signal input or watchdog timer output/address-trap-reset output/system-clock-reset output | |
| TEST | Input | Test pin for out-going test. Be tied to low. | |
| OVDD, OVSS | Power supply | +5 V, 0 V (GND) for OSD oscillator circuit. | |
| VDD, VSS, VVSS | Power supply | +5 V, 0 V (GND) | |

# Block Diagram



# Operational Description

## 1. CPU Core Functions

The CPU core consists of a CPU, a system clock controller, and an interrupt controller.

This section provides a description of the CPU core, the program memory, the data memory, the external memory interface, and the reset circuit.

### 1.1 Memory Address Map

The TMP88CS38/CM38A/CP38A memory consists of four blocks: ROM, RAM, SFR (Special function register), and DBR (Data buffer register). They are all mapped to a 1-Mbyte address space. Figure 1.1.1 shows the TMP88CS38/CM38A/CP38A memory address map. There are 16 banks of the general-purpose register. The register banks are also assigned to the RAM address space.

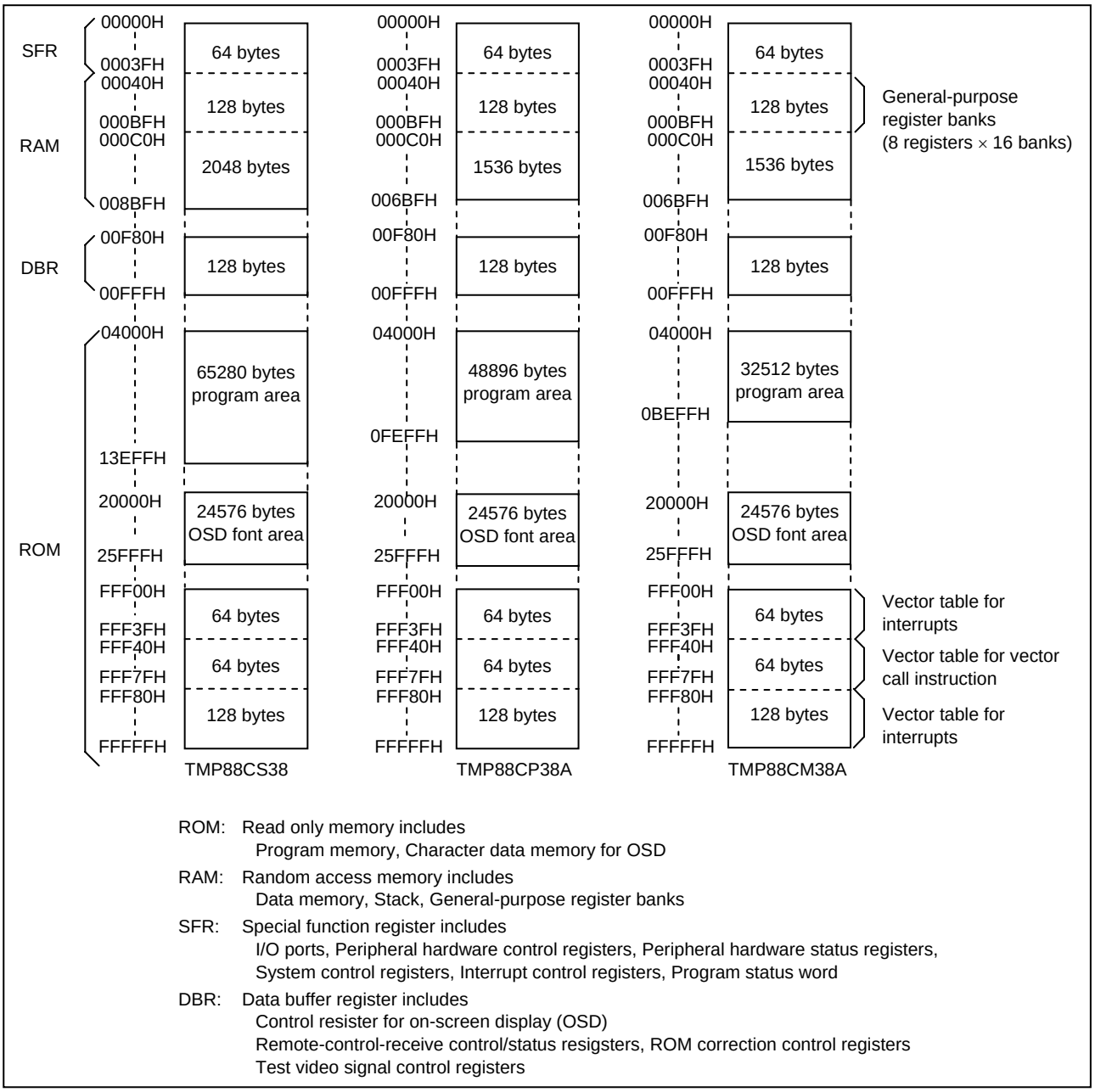



Figure 1.1.1 Memory Address Map

## 1.2 Program Memory (ROM)

The TMP88CS38 contains a 64-Kbyte program memory (Mask ROM) at addresses from 04000H to 13EFFH and FFF00H to FFFFFH.

The TMP88CM38A contains a 32-Kbyte program memory (Mask ROM) at address from 04000H to 0BEFFH and FFF00H to FFFFFH. The TMP88CP38A contains a 48-Kbyte program memory (Mask ROM) at address from 04000H to 0FEFFH and FFF00H to FFFFFH.

Addresses FFF00H through FFFFFH in the program memory are also used for a particular purpose.

## 1.3 Data Memory (RAM)

The TMP88CS38 has a 2-Kbyte data memory (Static RAM) address from 0040H to 08BFH.

The TMP88CM38A/P38A has a 1.5-Kbyte data memory (Static RAM) (Address from 0040H to 06BFH.

The first 128 bytes (Addresses 00040H through 000BFH) in the built-in RAM are also available as general-purpose register banks.

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

Example: Clears RAM to “00H” except the bank 0 (TMP88CS38/CM38A/CP38A):

```
            LD      HL, 0048H       ;   Sets start address to HL register pair
            LD      A, H            ;   Sets initial data (00H) to A register
            LD      BC, 0877H       ;   Sets number of byte to BC register pair
SRAMCLR:    LD      (HL+), A
            DEC     BC
            JRS     F, SRAMCLR
```

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

## 1.4 System Clock Controller

The system clock controller consists of a clock generator, a timing generator, and a standby controller.

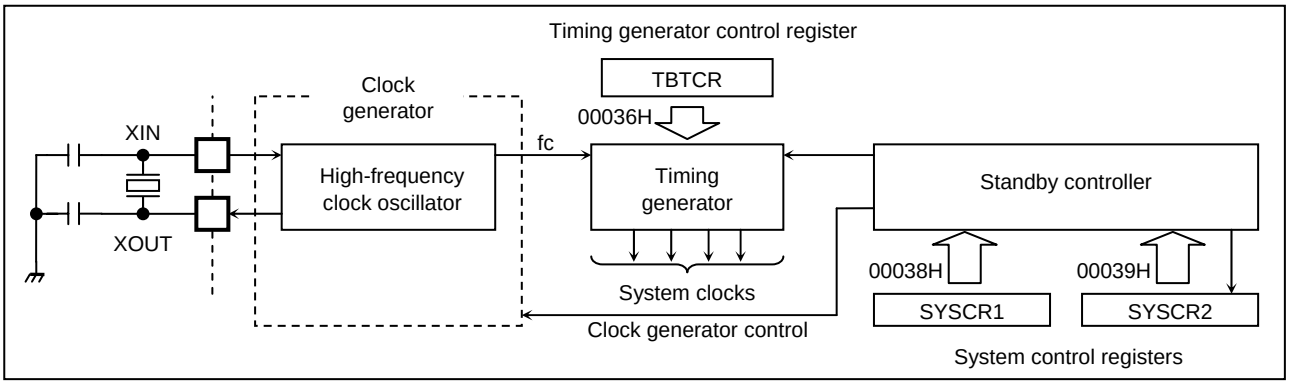



Figure 1.4.1 System Clock Controller

### 1.4.1 Clock Generator

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

The high-frequency (fc) clock can be easily obtained by connecting a resonator between the XIN/XOUT pin, respectively. Clock input from an external oscillator is also possible. In this case, external clock is applied to the XIN/XTIN pin not connected. The TMP88CS38/CM38A/CP38A is not provided an LC oscillation.

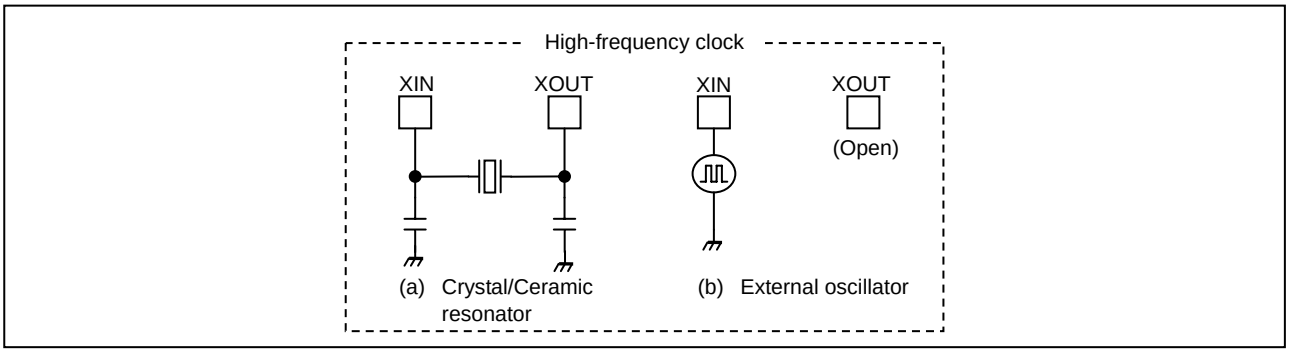



Figure 1.4.2 Examples of Resonator Connection

Note: Accurate adjustment of the oscillation frequency:
Although hardware to externally and directly monitor the basic clock pulse is not provided, the oscillation frequency can be adjusted by making the 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.

### 1.4.2 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 timer/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 3 prescaler, a main system clock generator, and machine cycle counters.

During reset and at releasing STOP mode, the prescaler and the divider are cleared to “0”, however, the prescaler is not cleared.

An input clock to the 7th stage of the divider depends on the operating mode.

A divided by 256 of high-frequency clock ($fc/2^8$) is input to the 7th stage of the divider.

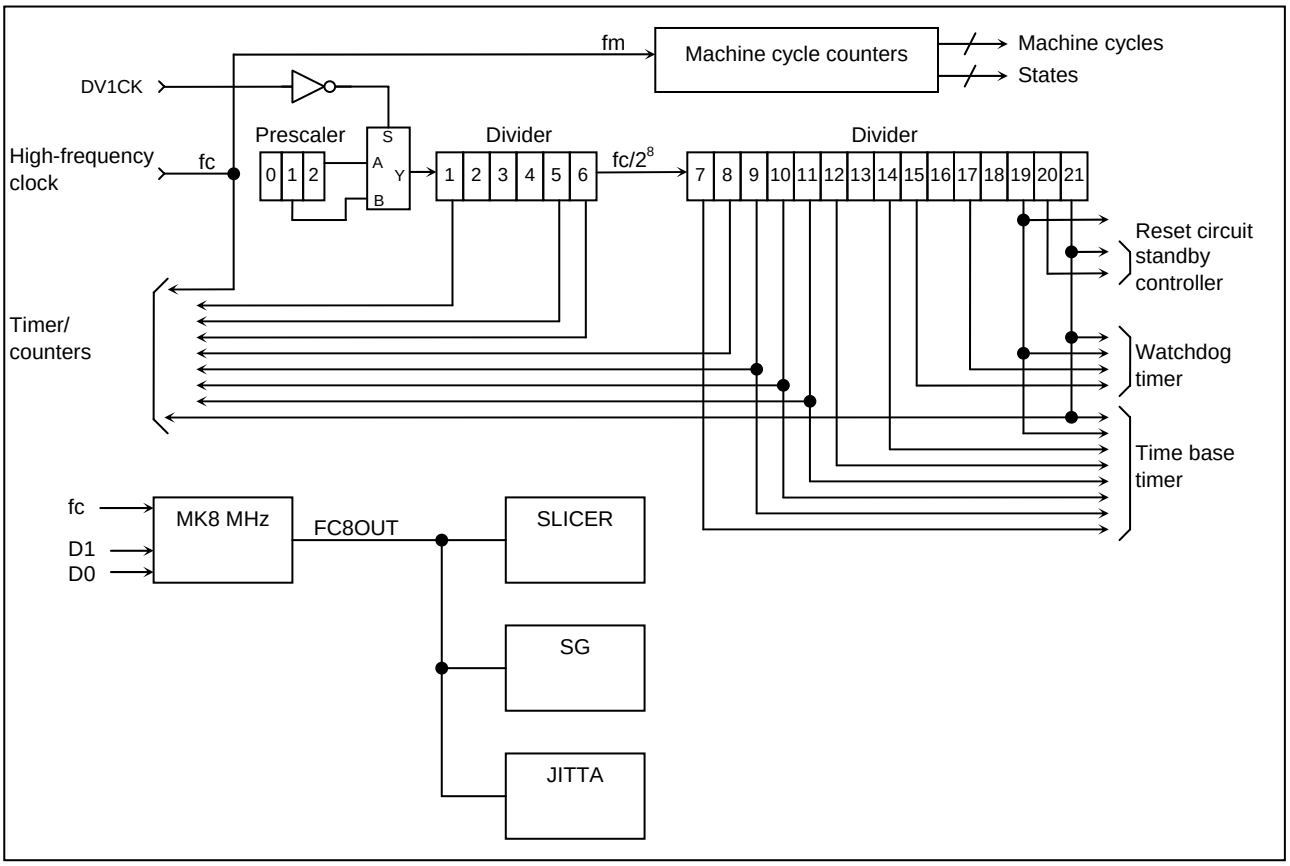



Figure 1.4.3 Configuration of Timing Generator

CGCR (00030H)

| "0" | "0" | DV1CK | "0" | "0" | "0" | "0" | "0" |
|---|---|---|---|---|---|---|---|

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| DV1CK | Selection of input clock to the 1st stage of the divider. | 0: fc/4<br>1: fc/8 | R/W |

Note 1: fc: High-frequency clock [Hz], *: Don't care

Note 2: The all bits except DV1CK are cleared to "0".

Figure 1.4.4 Divider Control Register

FC8CR (00FEEH)

| | | | | | | D1 | D0 |
|---|---|---|---|---|---|---|---|

Read/Write (Initial value: 0000 0010)

| D1 | D0 | FC8OUT |
|---|---|---|
| 1 | 0 | 1/2 fc |
| 0 | 0 | 1/1 fc |

Figure 1.4.5 FC8 Control Register

(2) Machine cycle

Instruction execution and peripheral hardware operation are synchronized with the main system clock. The minimum instruction execution unit is called a "machine cycle". There are a total of 15 different types of instructions for the TLCS-870/X series: Ranging from 1-cycle instructions which require one machine cycle for execution to 15-cycle instructions which require 15 machine cycles for execution.

A machine cycle consists of 4 states (S0 to S3), and each state consists of one main system clock.

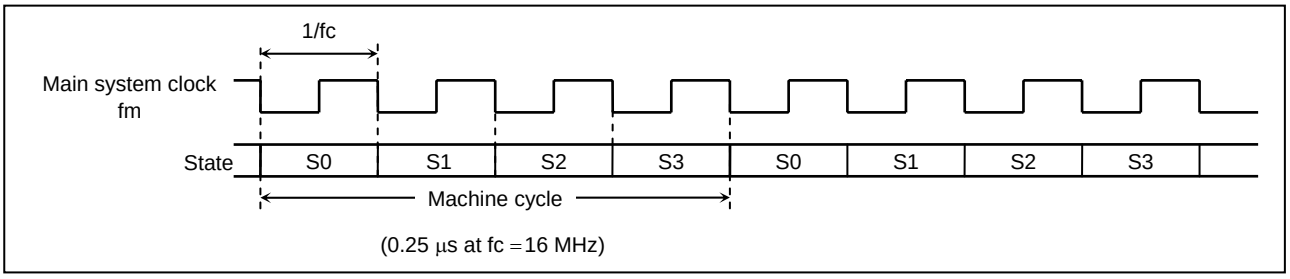



Figure 1.4.6 Machine Cycle

### 1.4.3 Standby Controller

The standby controller starts and stops the switches the main system clock. These modes are controlled by the system control registers (SYSCR1, SYSCR2).

Figure 1.4.7 shows the operating mode transition diagram and Figure 1.4.8 shows the system control registers.

(1) Single-clock mode

In the single-clock mode, the machine cycle time is 4/fc [s] (0.25 μs at fc = 16 MHz).

1. NORMAL mode

   In this mode, both the CPU core and on-chip peripherals operate using the high-frequency clock.

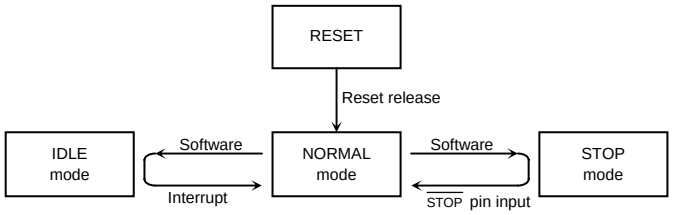
2. IDLE mode

   In this mode, the internal oscillation circuit remains active. The CPU and the watchdog timer are halted; however, on-chip peripherals remain active (Operate using the high-frequency clock). IDLE mode is started by setting IDLE bit in the system control register 2 (SYSCR2), and IDLE1 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 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.

   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 warm-up period is completed, the execution resumes with the next instruction which follows the STOP mode start instruction.




(a) Single-clock mode

Note: NORMAL mode are generically called NORMAL; STOP mode is called STOP; and IDLE mode is called IDLE.

| Operating Mode | | Frequency | CPU Core | On-chip Peripherals | Machine Cycle Time |
|---|---|---|---|---|---|
| | | High-frequency | | | |
| Single clock | RESET | Turning on oscillation | Reset | Reset | 4/fc [s] |
| | NORMAL | | Operate | Operate | |
| | IDLE | | Halt | | |
| | STOP | Turning off oscillation | | Halt | – |

Figure 1.4.7 Operating Mode Transition Diagram

System Control Register 1

| SYSCR1 (00038H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | STOP | RELM | “0” | “1” | WUT | | | |

(Initial value: 0000 00** )

| | | | | | |
|---|---|---|---|---|---|
| STOP | STOP mode start | 0: CPU core and peripherals remain active<br>1: CPU core and peripherals are halted (Start STOP mode) | | | R/W |
| RELM | Release method for STOP mode | 0: $\overline{\text{STOP}}$ edge-sensitive release (Rising edge)<br>1: $\overline{\text{STOP}}$ level-sensitive release (“H” level) | | | |
| WUT | Warm-up time at releasing STOP mode | | Return to NORMAL mode | | |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 00 | $3 \times 2^{16}$/fc | $3 \times 2^{17}$/fc | |
| | | 01 | $2^{16}$/fc | $2^{17}$/fc | |
| | | 10 | $2^{14}$/fc | $2^{15}$/fc | |
| | | 11 | Reserved | Reserved | |

Note 1: Always set bit5 in SYSCR1 to “0”.

Note 2: When STOP mode is released with $\overline{\text{RESET}}$ pin input, a return is made to NORMAL mode regardless of the RETM contents.

Note 3: fc: High-frequency clock [Hz], *: Don’t care

Note 4: Bits 1 and 0 in SYSCR1 are read in as undefined data when a read instruction is executed.

Note 5: Always set bit4 in SYSCR1 to “1” when STOP mode is started.

System Control Register 2

| SYSCR2 (00039H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | “1” | “0” | “0” | IDLE | | | | |

(Initial value: 1000 **** )

| | | | |
|---|---|---|---|
| IDLE | IDLE mode start | 0: CPU and watchdog timer remain active<br>1: CPU and watchdog timer are stopped (Start IDLE mode) | R/W |

Note 1: *: Don’t care

Note 2: Always set bit7, 6 and 5 in SYSCR2 to “100”.

Figure 1.4.8 System Control Registers

### 1.4.4 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 (Bit7 in SYSCR1 ) to "1". During STOP mode, the following status is maintained.

1. Oscillations are 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.
3. The prescaler and the divider of the timing generator are cleared to "0".
4. The program counter holds the address of the instruction following the instruction which started the STOP mode.

STOP mode includes a level-sensitive release mode and an edge-sensitive release mode, either of which can be selected with RELM (Bit6 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 backup when the main power supply is cut off and long term battery backup.

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 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, the STOP mode does not start if port
                                                P20 is at high
          JRS    F, SINT5
          LD     (SYSCR1), 01010000B       ;  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
```

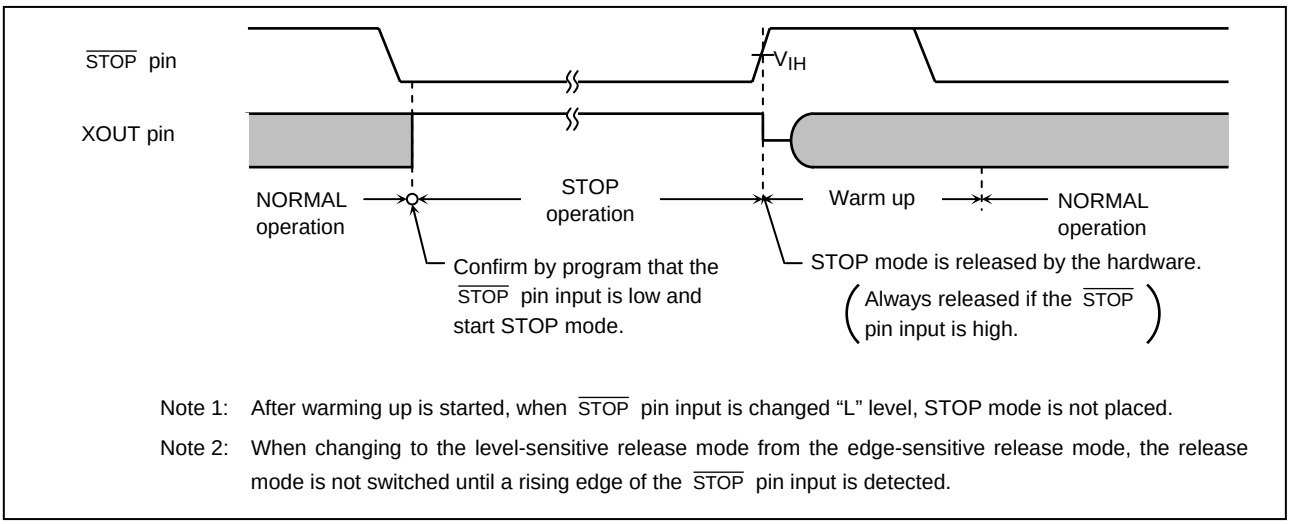



Figure 1.4.9 Level-sensitive Release Mode

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

In this mode, STOP mode is released by a rising edge of the $\overline{STOP}$ 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 $\overline{STOP}$ pin.

In the edge-sensitive release mode, STOP mode is started even when the $\overline{STOP}$ pin input is high.

Example: Starting STOP mode from NORMAL mode

```
LD      (SYSCR1), 10010000B          ;    Starts after specified to the edge-sensitive mode
```

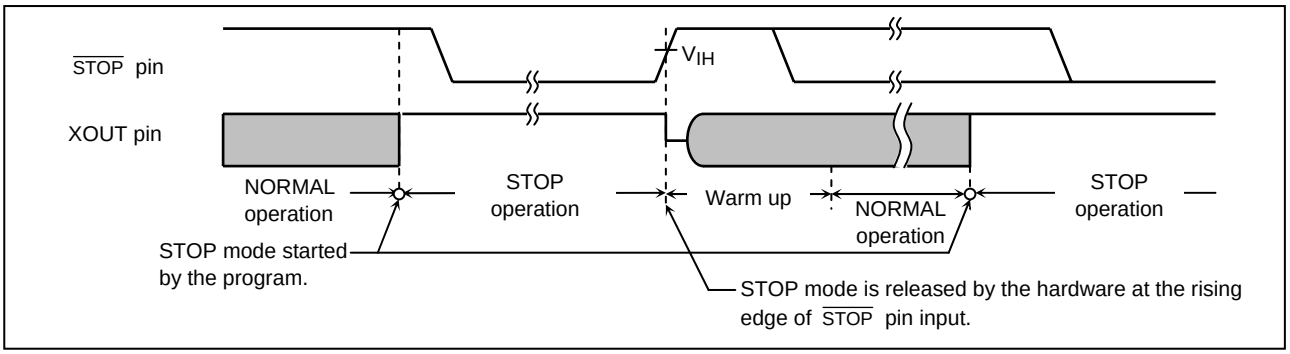



Figure 1.4.10 Edge-sensitive Release Mode

STOP mode is released by the following sequence:

1. When returning to NORMAL, clock oscillator is turned on.
2. A warm-up period is inserted to allow oscillation time to stabilize. During warm-up, all internal operations remain halted. Two different warm-up times can be selected with WUT (Bits2 and 3 in SYSCR1) as determined by the resonator characteristics.
3. When the warm-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 1.4.1 Warm-up Time Example

| WUT | Warm-up Time [s] | | | |
|---|---|---|---|---|
| | Return to NORMAL mode | | | |
| | DV1CK = 0 | | DV1CK = 1 | |
| 00 | $3 \times 2^{16}$/fc | (12.29 m) | $3 \times 2^{17}$/fc | (24.58 m) |
| 01 | $2^{16}$/fc | (4.10 m) | $2^{17}$/fc | (8.20 m) |
| 10 | $2^{14}$/fc | (1.02 m) | $2^{15}$/fc | (2.05 m) |
| 11 | Reserved | ( - ) | Reserved | ( - ) |

Note: The warm-up time is obtained by dividing the basic clock by the divider: therefore, the warm-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 warm-up time must be considered an approximate value.

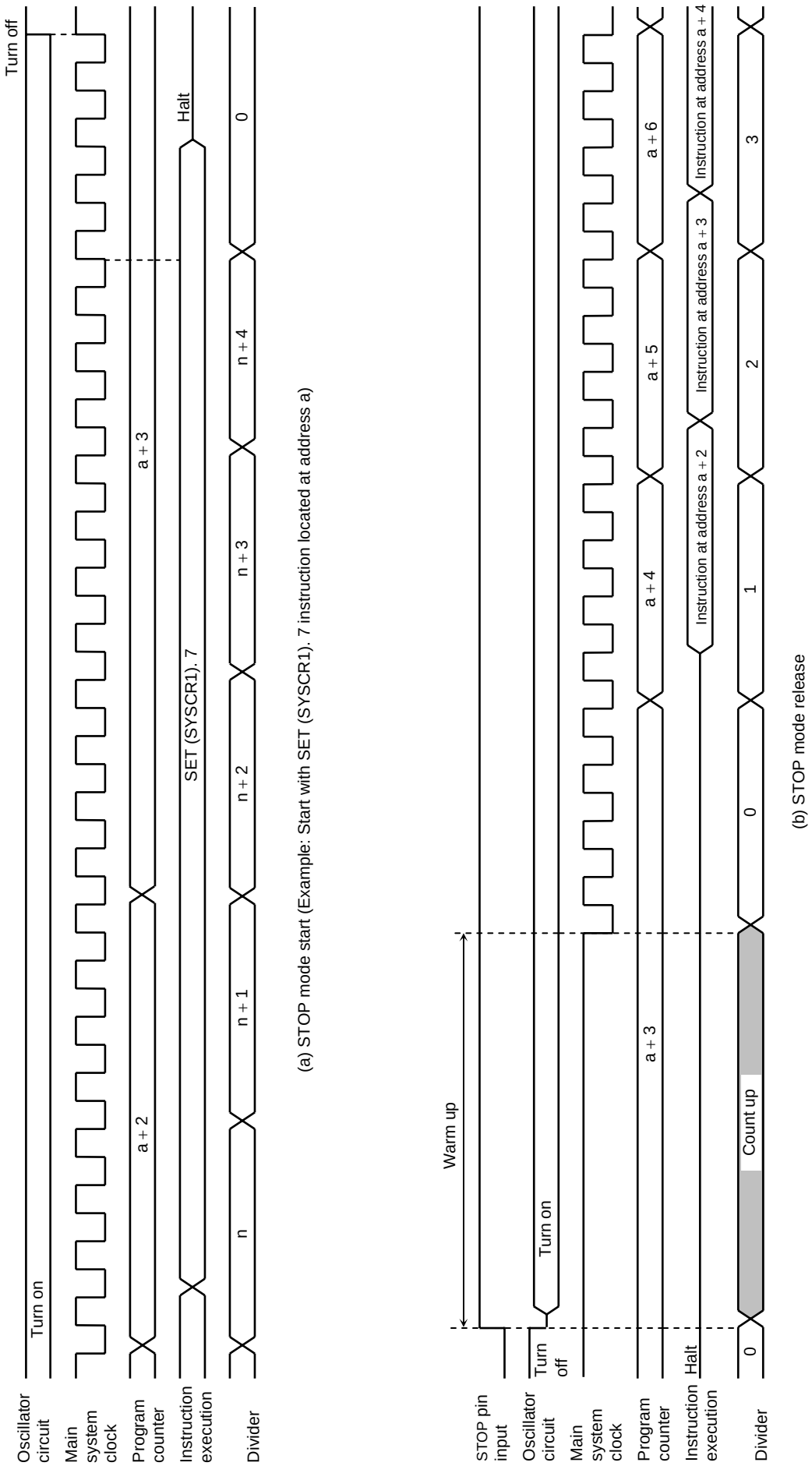



Figure 1.4.11 STOP Mode Start/Release

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 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).

(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
```

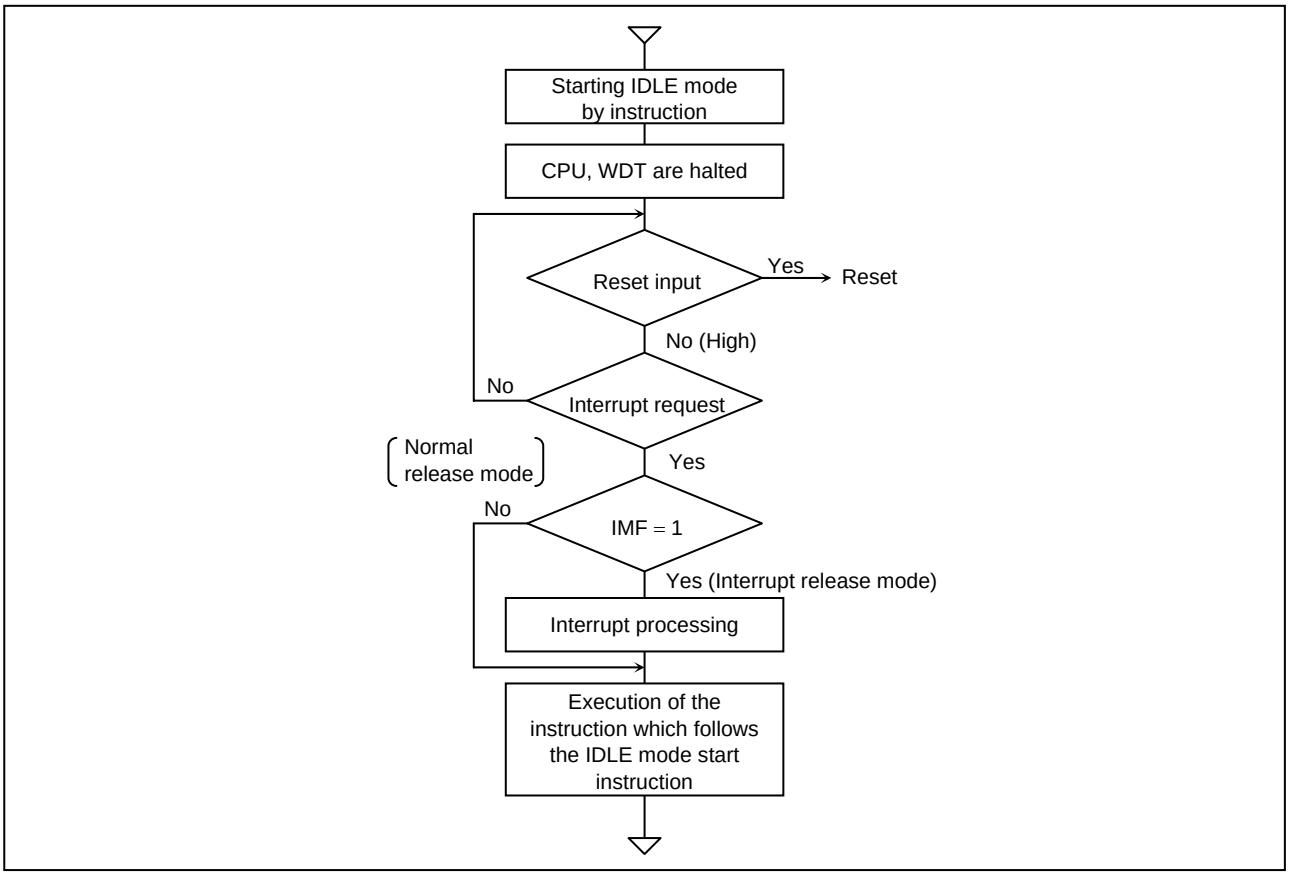



Figure 1.4.12 IDLE Mode

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 from IDLE to NORMAL.

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

IDLE mode is released by any interrupt source enabled by the individual interrupt enable flag (EF) or an external interrupt 0 ($\overline{\text{INT0}}$ pin) request. 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 IDLE mode must be cleared by load instructions.

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) or an external interrupt 0 ($\overline{\text{INT0}}$ pin) request. After the interrupt is processed, the execution resumes from the instruction following the instruction which started IDLE mode.

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

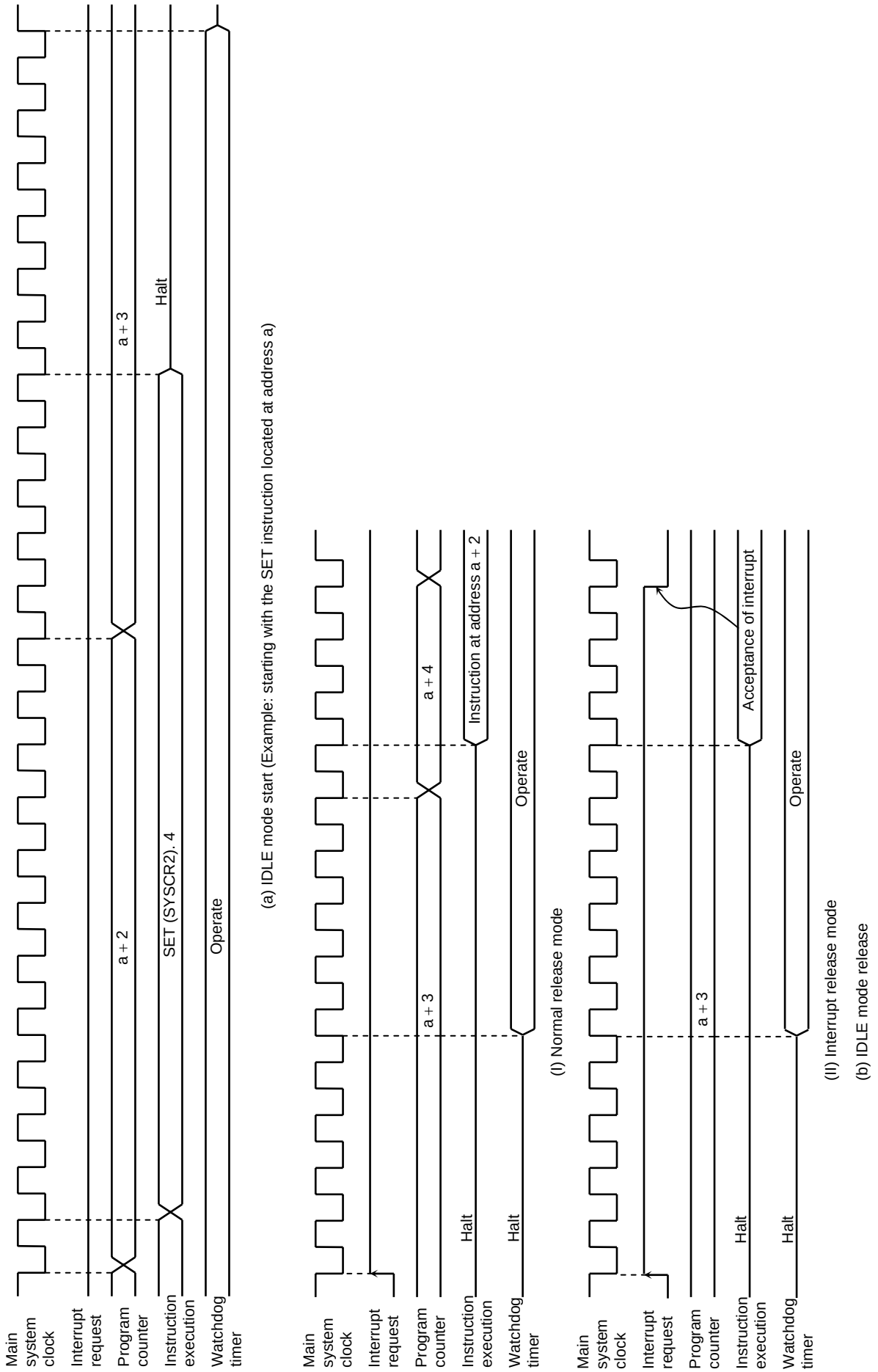



Figure 1.4.13 IDLE Mode Start/Release

IDLE mode can also be released by setting the $\overline{\text{RESET}}$ pin low, which immediately performs the reset operation. After reset, the TMP88CS38/CM38A/CP38A is placed in NORMAL mode.

## 1.5 Interrupt Controller

The TMP88CS38/CM38A/CP38A has a total of 17 interrupt sources; 6 externals and 11 internals. Multiple interrupts with priorities are also possible. Two of the internal sources are pseudo non-maskable interrupts; the remainder are all maskable interrupts.

Table 1.5.1 Interrupt Sources

| Interrupt Source | | | Enable Condition | Interrupt Latch | Vector Table Address | Priority |
|---|---|---|---|---|---|---|
| Internal/External | (Reset) | | Non maskable | – | FFFFCH | High 0 |
| Internal | INTSW | (Software interrupt) | Pseudo non maskable | – | FFFF8H | 1 |
| Internal | INTWDT | (Watchdog timer interrupt) | | $IL_2$ | FFFF4H | 2 |
| External | INT0 | (External interrupt 0) | IMF ·$EF_3$ = 1, INT0EN = 1 | $IL_3$ | FFFF0H | 3 |
| Internal | INTTC1 | (16-bit TC1 interrupt) | IMF ·$EF_4$ = 1 | $IL_4$ | FFFECH | 4 |
| External | INTKWU | (Key-on wakeup) | IMF ·$EF_5$ = 1 | $IL_5$ | FFFE8H | 5 |
| Internal | INTTBT | (Time base timer interrupt) | IMF ·$EF_6$ = 1 | $IL_6$ | FFFE4H | 6 |
| External | INT2 | (External interrupt 2) | IMF ·$EF_7$ = 1 | $IL_7$ | FFFE0H | 7 |
| Internal | INTTC3 | (8-bit TC3 interrupt) | IMF ·$EF_8$ = 1 | $IL_8$ | FFFDCH | 8 |
| Internal | INTTSBI | (SBI interrupt) | IMF ·$EF_9$ = 1 | $IL_9$ | FFFD8H | 9 |
| Internal | INTTC4 | (8-bit TC4 interrupt) | IMF ·$EF_{10}$ = 1 | $IL_{10}$ | FFFD4H | 10 |
| External | INT3 | (External interrupt 3) | IMF ·$EF_{11}$ = 1 | $IL_{11}$ | FFFD0H | 11 |
| External | INT4 | (External interrupt 4) | IMF ·$EF_{12}$ = 1 | $IL_{12}$ | FFFCCH | 12 |
| Internal | INTADC | (AD converter interrupt) | IMF ·$EF_{13}$ = 1 | $IL_{13}$ | FFFC8H | 13 |
| Internal | INTTC2 | (16-bit TC2 interrupt) | IMF ·$EF_{14}$ = 1 | $IL_{14}$ | FFFC4H | 14 |
| External | INT5 | (External interrupt 5) | IMF ·$EF_{15}$ = 1 | $IL_{15}$ | FFFC0H | 15 |
| Internal | INTOSD | (OSD interrupt) | IMF ·$EF_{16}$ = 1 | $IL_{16}$ | FFFBCH | 16 |
| Internal | INTSLI | (Slicer interrupt) | IMF ·$EF_{17}$ = 1 | $IL_{17}$ | FFFB8H | 17 |
| | | Reserved | IMF ·$EF_{18}$ = 1 | $IL_{18}$ | FFFB4H | 18 |
| | | Reserved | IMF ·$EF_{19}$ = 1 | $IL_{19}$ | FFFB0H | 19 |
| | | Reserved | IMF ·$EF_{20}$ = 1 | $IL_{20}$ | FFFACH | 20 |
| | | Reserved | IMF ·$EF_{21}$ = 1 | $IL_{21}$ | FFFA8H | 21 |
| | | Reserved | IMF ·$EF_{22}$ = 1 | $IL_{22}$ | FFFA4H | 22 |
| | | Reserved | IMF ·$EF_{23}$ = 1 | $IL_{23}$ | FFFA0H | 23 |
| | | Reserved | IMF ·$EF_{24}$ = 1 | $IL_{24}$ | FFF9CH | 24 |
| | | Reserved | IMF ·$EF_{25}$ = 1 | $IL_{25}$ | FFF98H | 25 |
| | | Reserved | IMF ·$EF_{26}$ = 1 | $IL_{26}$ | FFF94H | 26 |
| | | Reserved | IMF ·$EF_{27}$ = 1 | $IL_{27}$ | FFF90H | 27 |
| | | Reserved | IMF ·$EF_{28}$ = 1 | $IL_{28}$ | FFF8CH | 28 |
| | | Reserved | IMF ·$EF_{29}$ = 1 | $IL_{29}$ | FFF88H | 29 |
| | | Reserved | IMF ·$EF_{30}$ = 1 | $IL_{30}$ | FFF84H | 30 |
| | | Reserved | IMF ·$EF_{31}$ = 1 | $IL_{31}$ | FFF80H | Low 31 |

Note: Before you change each enable flag (EF) and/or each interrupt latch (IL), be sure to clear the interrupt master enable flag (IMF) to “0” (to disable interrupts).

a. After a DI instruction is executed.

b. When an interrupt is accepted, IMF is automatically cleared to “0”.
However to enable nested interrupts, change EF and/or IL before setting IMF to “1” (to enable interrupts).

If the individual enable flags (EF) and interrupt latches (IL) are set under conditions other than the above, the proper operation cannot be guaranteed.

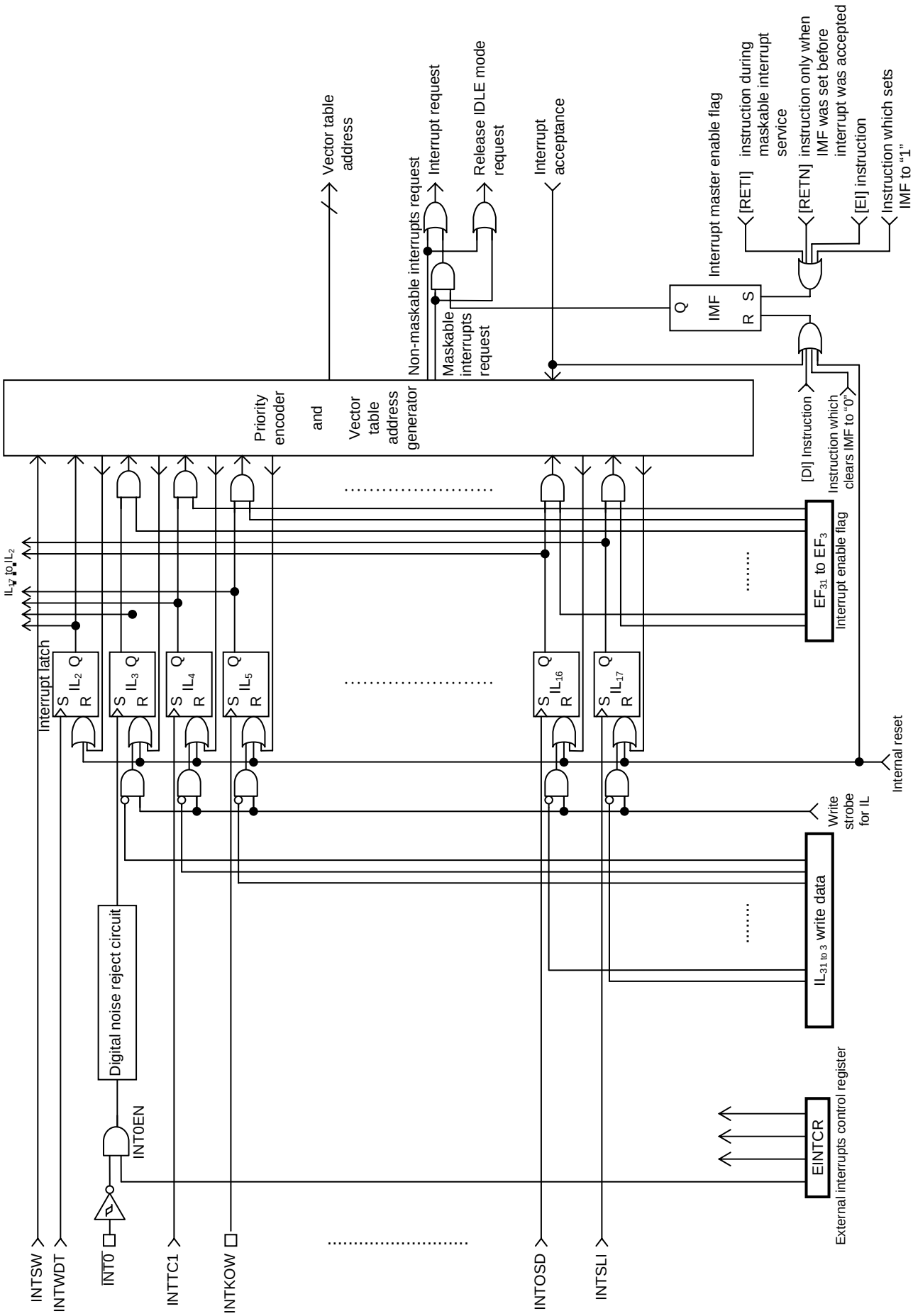



Figure 1.5.1 Interrupt Controller Block Diagram

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 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 1.5.1 shows the interrupt controller.

(1) Interrupt latches ($IL_{31}$ to $IL_2$)

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 are initialized to "0" during reset.

The interrupt latches are assigned to addresses 0003CH, 0003DH, 0002EH and 0002FH in the SFR. Except for IL2, 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. When interrupt occurred during order execution, the reason is because interrupt request is cleared. Thus, interrupt requests can be canceled and initialized by the program. Note that request the interrupt latches cannot be set to "1" by an instruction. For example, it may be that each latch is cleared even if an interrupt request is generated during instruction exection.

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

```
Example 1: Clears interrupt latches
        DI                                  ;   Disable interrupt
        LDW     (ILL), 1110100000111111B    ;   IL12, IL10 to IL6 ← 0

Example 2: Reads interrupt latches
        LD      WA, (ILL)                   ;   W ← ILH, A ← ILL

Example 3: Tests an interrupt latch
        TEST    (ILL). 7                    ;   if IL7 = 1 then jump
        JR      F, SSET
```

(2) Interrupt enable register (EIR)

The interrupt enable register (EIR) enables and disables 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 interrupt cannot be nested more than once at the same time.

The EIR consists of an interrupt master enable flag (IMF) and the individual interrupt enable flags (EF). These registers are assigned to addresses 0003AH, 0003BH, 0002CH and 0002DH in the SFR, and can be read and written by an instruction (including read-modify-write instruction such as bit manipulation instructions).

Note: Do not use the read-modify-write instruction for the EIRL (Address 0003AH) during pseudo non-maskable interrupt service task. If the read-modify-write instruction is used, the IMF is not set to "1" after RETN.

1. Interrupt master enable flag (IMF)

   The interrupt master enable flag (IMF) enables and disables the acceptance of all 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 other 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 bit0 at address $0003A_H$ 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 ($EF_{17}$ to $EF_3$)

   These flags enable and disable the acceptance of individual maskable interrupts, except for an external interrupt 0. 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".

```
DI                                  ;  Disable interrupt
LD      (EIRE), 00000001B           ;  EF16 ← 1
LDW     (EIRL), 1110100010100001B      EF15 to EF13, EF11, EF7, EF5, IMF ← 1
```

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

```
SET     (EIRH). 4                   ;  EF12 ← 1
```

Interrupt Latches (IL)

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IL (0002E, 0002FH) | $IL_{31}$ | $IL_{30}$ | $IL_{29}$ | $IL_{28}$ | $IL_{27}$ | $IL_{26}$ | $IL_{25}$ | $IL_{24}$ | $IL_{23}$ | $IL_{22}$ | $IL_{21}$ | $IL_{20}$ | $IL_{19}$ | $IL_{18}$ | $IL_{17}$ | $IL_{16}$ |
| | $IL_D$ (0002FH) | | | | | | | | $IL_E$ (0002EH) | | | | | | | |

(Initial value: 00000000 00000000)

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IL (0003C, 0003DH) | $IL_{15}$ | $IL_{14}$ | $IL_{13}$ | $IL_{12}$ | $IL_{11}$ | $IL_{10}$ | $IL_9$ | $IL_8$ | $IL_7$ | $IL_6$ | $IL_5$ | $IL_4$ | $IL_3$ | $IL_2$ | INF | |
| | $IL_H$ (0003DH) | | | | | | | | $IL_L$ (0003CH) | | | | | | | |

(Initial value: 00000000 00000000**)

Interrupt Enable Registers (EIR)

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EIR (0002C, 0002DH) | $EF_{31}$ | $EF_{30}$ | $EF_{29}$ | $EF_{28}$ | $EF_{27}$ | $EF_{26}$ | $EF_{25}$ | $EF_{24}$ | $EF_{23}$ | $EF_{22}$ | $EF_{21}$ | $EF_{20}$ | $EF_{19}$ | $EF_{18}$ | $EF_{17}$ | $EF_{16}$ |
| | $EIR_D$ (0002DH) | | | | | | | | $EIR_E$ (0002CH) | | | | | | | |

(Initial value: 00000000 00000000)

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EIR (0003A, 0003BH) | $EF_{15}$ | $EF_{14}$ | $EF_{13}$ | $EF_{12}$ | $EF_{11}$ | $EF_{10}$ | $EF_9$ | $EF_8$ | $EF_7$ | $EF_6$ | $EF_5$ | $EF_4$ | $EF_3$ | | | IMF |
| | $EIR_H$ (0003BH) | | | | | | | | $EIR_L$ (0003AH) | | | | | | | |

(Initial value: 00000000 0000000**0)

Note 1: Do not clear IL with read-modify-write instructions such as bit operations.

Note 2: Do not set IMF to “1” during non-maskable interrupt service program.

Note 3: Bits 1 and 0 in $IL_L$ are read in as undefined data when a read instruction is executed.

Note 4: *: Don’t care

Note 5: Do not clear $IL_2$ to “0” by an instruction.

Note 6: At TMP88CS38/CM38A/CP38A, $IL_{18}$ to $IL_{31}$ and $IF_{18}$ to $IF_{31}$ are not used.

Note 7: After IMF is cleared, modify EF and IL.

Figure 1.5.2 Interrupt Latches (IL) and Interrupt Enable Registers (EIR)

### 1.5.1 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 12 machine cycles (3 μs at fc = 16 MHz in the 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). Figure 1.5.3 shows the timing chart of interrupt acceptance processing.

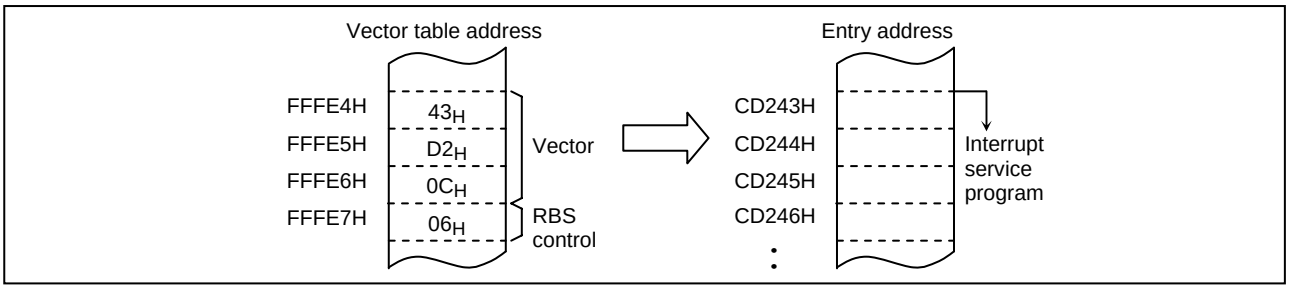
(1) Interrupt acceptance

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 (PC) and the program status word (PSW) are saved (Pushed) on the stack in sequence of $PSW_H$, $PSW_L$, $PC_E$, $PC_H$, $PC_L$. The stack pointer (SP) is decremented five times.
4. The entry address of the interrupt service program is read from the vector table, and set to the program counter.

5. The RBS control code is read from the vector table. The lower 4-bit of this code is added to the RBS.
6. The instruction stored at the entry address of the interrupt service program is executed.

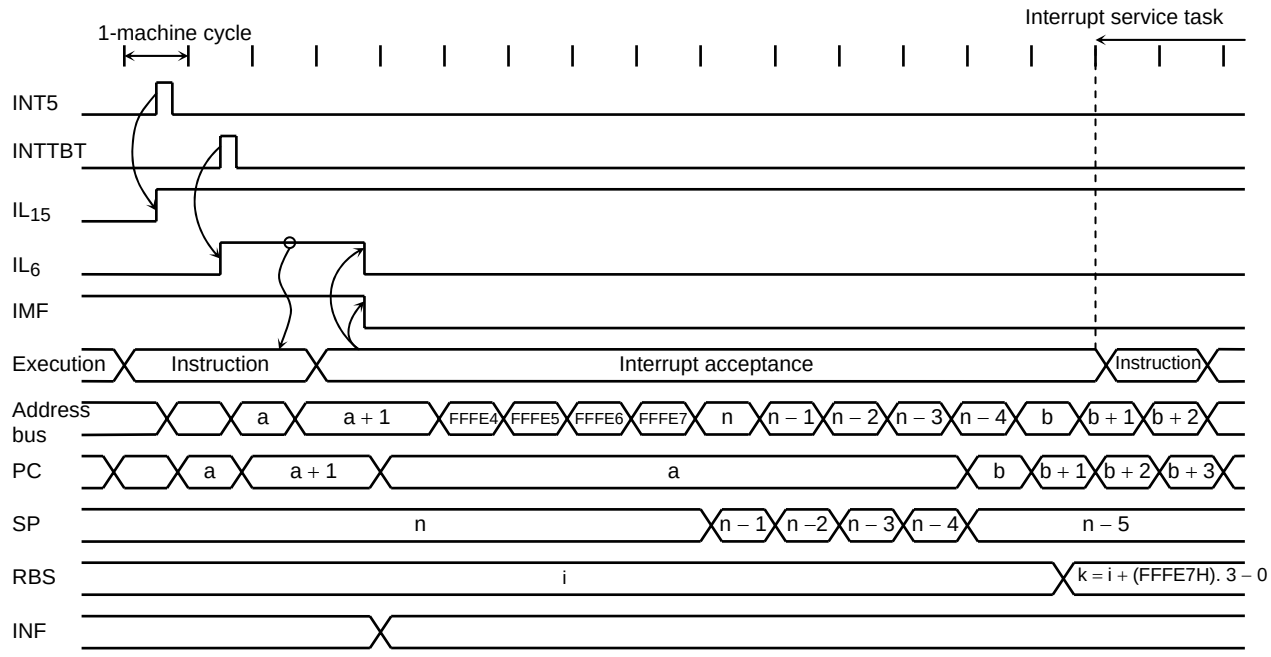
Example: Correspondence between vector table address for INTTBT and the entry address of the interrupt service program.



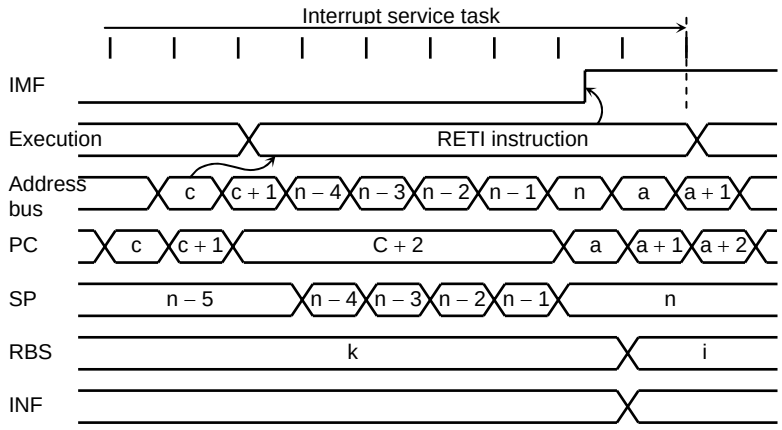

A maskable interrupt is not accepted until the IMF is set to "1" even if the maskable interrupt higher than the level of current servicing interrupt is occurred.

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.

Note: Do not use the read-modify-write instruction for the EIRL (Address 0003AH) during pseudo non-maskable interrupt service task.




(a) Interrupt acceptance




(b) Return from interrupt instruction

Note 1: a: Return address, b: Entry address, c: Address which the RETI instruction is stored

Note 2: The maximum response time from when an IL is set until an interrupt acceptance processing starts is 62/fc [s] with interrupt enabled.

Figure 1.5.3 Timing Chart of Interrupt Acceptance and Interrupt Return Instruction

(2) Saving/restoring general-purpose registers

During interrupt acceptance processing, the program counter (PC) and the program status word (PSW) 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 automatic register bank changeover

The general-purpose registers can be saved at high speed by switching to a register bank that is not in use. Normally, the bank0 is used for the main task and the 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:   Interrupt processing
          RETI
          ⋮
VINTxx:   DP      PINTxx
          DB      1          ;   RBS ← RBS + 1
```

2. General-purpose register save/restore by register bank changeover

The general-purpose registers can be saved at high speed by switching to a register bank that is not in use. Normally, the bank0 is used for the main tank and the banks 1 to 15 are assigned to interrupt service tasks.

Example: Register bank changeover

```
PINTxx:   LD RBS, n
          Interrupt processing
          RETI                   ;   Restores bank and returns
          ⋮
VINTxx:   DP      PINTxx         ;   Interrupt service routine entry address
          DB      0
```

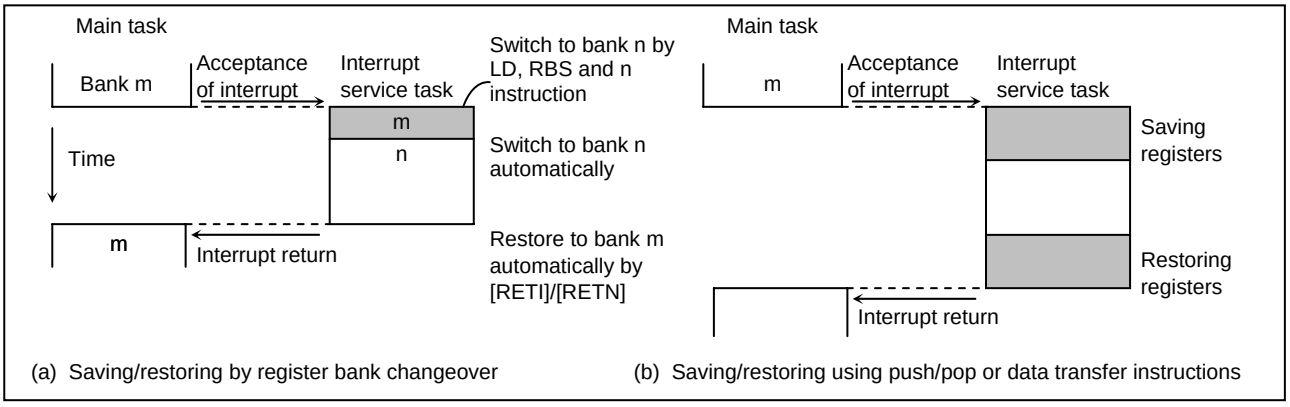


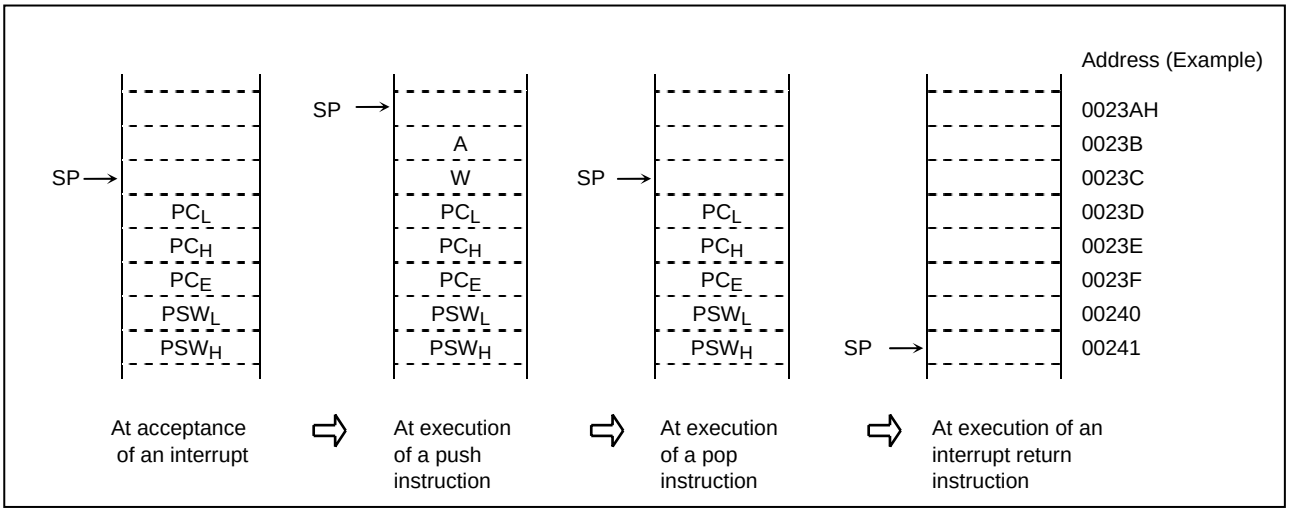

Figure 1.5.4 Saving/Restoring General-purpose Registers

3. General-purpose registers 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 the push/pop instructions.

Example: Register save/restore using push and pop instructions

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




4. General-purpose registers save/restore using data transfer instructions

Data transfer instruction can be used to save only a specific general-purpose register during processing of single interrupt.

Example: Saving/restoring a register using data transfer instructions

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

(3) Interrupt return

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 5 times. | 2. The stack pointer is incremented 5 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. |
| 4. The interrupt nesting counter is decremented, and the interrupt nesting flag is changed. | 4. The interrupt nesting counter is decremented, and the interrupt nesting flag is changed. |

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.

### 1.5.2 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.

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. Address-trap reset is generated in case that an instruction is fetched from RAM, SFR or DBR areas.

2. Debugging

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

### 1.5.3 External Interrupts

The TMP88CS38/CM38A/CP38A each have five external interrupt inputs ($\overline{INT0}$, INT2, INT3, INT4, and $\overline{INT5}$). Three of these are equipped with digital noise rejection circuits (Pulse inputs of less than a certain time are eliminated as noise). Edge selection is also possible with INT2, INT3 and INT4.

The $\overline{INT0}$/P50 pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.

Edge selection, noise rejection control except INT3 pin input and $\overline{INT0}$/P50 pin function selection are performed by the external interrupt control register (EINTCR). Edge selecting and noise rejection control for INT3 pin input are preformed by the remote control signal preprocessor control registers. (Refer to the section of the remote control signal preprocessor.) When INT0EN = 0, the IL3 will not be set even if the falling edge of $\overline{INT0}$ pin input is detected.

Table 1.5.2 External Interrupts

| Source | Pin | Secondary Function Pin | Enable Conditions | Edge | Digital Noise Rejection |
|---|---|---|---|---|---|
| INT0 | $\overline{INT0}$ | P50/TC2/ $\overline{PWM8}$ | IMF = 1, INT0EN = 1, $EF_3$ = 1 | Falling edge | Any pulse shorter than 2/fc [s] is regarded as noise and removed. Pulses not shorter than 7/fc [s] are definitely regarded as signals. |
| INT2 | INT2 | P53/TC1/ $\overline{SCK1}$ / AIN0/ $\overline{KWU0}$ | IMF·$EF_7$ = 1 | Falling edge or rising edge | Pulses of less than 7/fc [s] are eliminated as noise. Pulses equal to or more than 25/fc [s] are regarded as signals. |
| INT3 | INT3 | P30/RXIN | IMF·$EF_{11}$ = 1 | Falling edge, rising edge or falling/rising edge | Refer to the section of the remote control preprocessor |
| INT4 | INT4 | P31/TC3 | IMF·$EF_{12}$ = 1 | Falling edge or rising edge | Pulses of less than 7/fc [s] are eliminated as noise. Pulses of 25/fc [s] or more are considered to be signals. |
| INT5 | $\overline{INT5}$ | P20/ $\overline{STOP}$ | IMF·$EF_{15}$ = 1 | Falling edge | Any pulse shorter than 2/fc [s] is regarded as noise and removed. Pulses not shorter than 7/fc [s] are definitely regarded as signals. |

Note 1: The noise rejection function is also affected for timer counter input 1 (TC1 pin).

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:
(1) INT2, INT4 pin 31/fc [s]
(2) INT3 pin Refer to the section of the remote control preprocessor.

Note 3: If a dual-function pin is used as an output port, changing data or switching between input and output generates a pseudo interrupt request signal. To ignore this signal, it is necessary to reset the interrupt enable flag.

Note 4: If INT0EN = “0”, detecting the falling edge of the $\overline{INT0}$ pin input does not set the interrupt latch IL3.

| EINTCR (00037H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | "0" | INT0 EN | – | INT4 ES | – | INT2 ES | "0" | – | (Initial value: 00*0 *00*) |

| | | | |
|---|---|---|---|
| INT0EN | P50/$\overline{\text{INT0}}$ pin configuration | 0: P50 input/output port<br>1: $\overline{\text{INT0}}$ pin (Port P50 should be set to an input mode) | Write only |
| INT4ES<br>INT2ES | INT4 and INT2 edge select | 0: Rising edge<br>1: Falling edge | |

Note 1: fc: High-frequency clock [Hz], *: Don't care

Note 2: Edge detection during switching edge selection is invalid.

Note 3: Do not change EINTCR only when IMF = 1. After changing EINTCR, interrupt latches of external interrupt inputs must be cleared to "0" using load instruction.

Note 4: In order to change of external interrupt input by rewriting the contents of INT2ES and INT4ES during NORMAL mode, clear interrupt latches of external interrupt inputs (INT2 and INT4) after 8 machine cycles from the time of rewriting.

Note 5: In order to change an edge of timer counter input by rewritng the contents of INT2ES during NORMAL mode, rewrite the contents after timer counter is stopped (TC*s = 0) , that is, terrupt disable state.
Then, clear a interrupt latch of external interrupt input (INT2) after 8 machine cycles from the time of rewriting to change to interrupt enable state. Finally, start timer counter.

Example: When change TC1 pin inputs edge in external trigger timer mode from rising edge falling edge.

```
                    LD (TC1CR) , 01001000B     ;   TC1S ← 00 (Stops TC1)
                    DI                         ;   IMF ← 0 (Disables interrupt service)
                    LD (EINTCR) , 00000100B    ;   INT2ES ← 1 (Change edge selection)
                    NOP
  ↑ 8-machine       to
  ↓ cycles          NOP
                    LD (ILL) , 01111111B       ;   IL7 ← 0 (Clears interrupt latch)
                    EI                         ;   IMF ← 1 (Enable interrupt service)
                    LD (TC1CR) , 01111000B     ;   TC1S ← 11 (Starts TC1)
```

Figure 1.5.5 External Interrupt Control Register

## 1.6 Reset Circuit

The TMP88CS38/CM38A/CP38A has four types of reset generation procedures : an external reset input, an address trap reset output, a watchdog timer reset output and a system clock reset output. Table 1.6.1 shows on-chip hardware initialization by reset action.

The malfunction reset output circuit such as watchdog timer reset, address trap reset and system clock reset is not initialized when power is turned on. The $\overline{\text{RESET}}$ pin can output level "L" at the maximum 24/fc [s] (1.5 μs at 16 MHz) when power is turned on.

Table 1.6.1 Initializing Internal Status by Reset Action

| On-chip Hardware | Initial Value | On-chip Hardware | Initial Value |
|---|---|---|---|
| Program counter (PC) | (FFFFEH to FFFFCH) | Prescaler and divider of timing generator | 0 |
| Stack pointer (SP) | Not initialized | | |
| General-purpose registers (W, A, B, C, D, E, H, L) | Not initialized | | |
| Register bank selector (RBS) | 0 | Watchdog timer | Enable |
| Jump status flag (JF) | 1 | | |
| Zero flag (ZF) | Not initialized | Output latches of I/O ports | Refer to I/O port circuitry |
| Carry flag (CF) | Not initialized | | |
| Half carry flag (HF) | Not initialized | | |
| Sign flag (SF) | Not initialized | | |
| Overflow flag (VF) | Not initialized | | |
| Interrupt master enable flag (IMF) | 0 | | |
| Interrupt individual enable flags (EF) | 0 | Control registers | Refer to each of control register |
| Interrupt latches (IL) | 0 | | |
| – | – | RAM | Not initialized |

### 1.6.1 External Reset Input

The $\overline{\text{RESET}}$ pin contains a Schmitt trigger (Hysteresis) with an internal pull-up resistor.

When the $\overline{\text{RESET}}$ pin is held at "L" level for at least 3 machine cycles (12/fc [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 FFFFCH to FFFFEH.

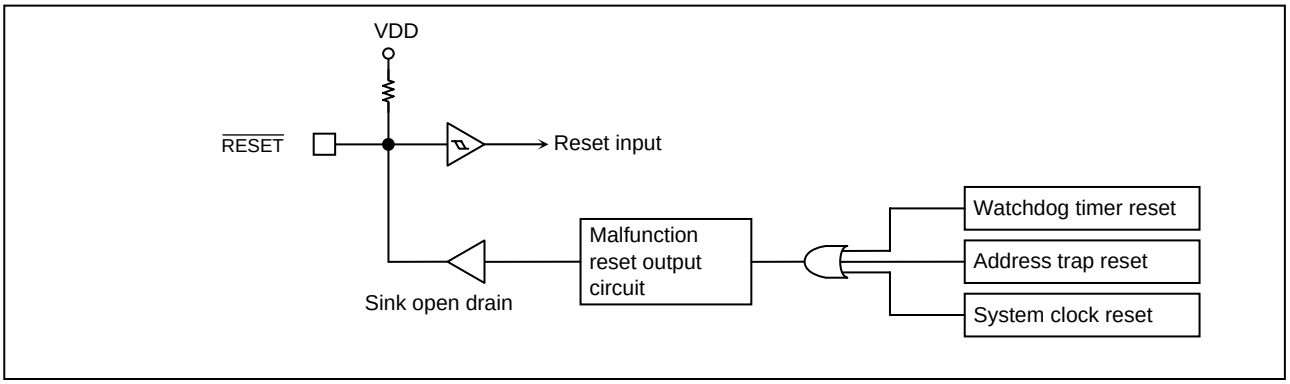
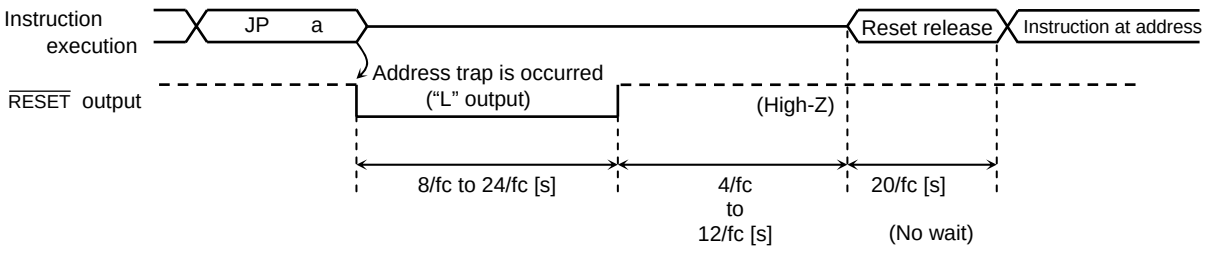



Figure 1.6.1 Reset Circuit

### 1.6.2 Address-trap-reset

If the CPU should start looping for some cause such as noise and an attempt be made to fetch an instruction from the on-chip RAM, DBR or the SFR area, address-trap-reset will be generated. Then, the $\overline{\text{RESET}}$ pin output will go low. The reset time is about 8/fc to 24/fc [s] (0.5 to 1.5 μs at 16 MHz).




Note 1: 0 ≤ a ≤ 008BFH (TMP88CS38), 0 ≤ a ≤ 006BFH (TMP88CM38A/CP38A)
0 ≤ a ≤ 002BFH (The ROM corrective function is enabled.)

Note 2: During reset release, reset vector "r" is read out, and an instruction at address "r" is fetched and decoded.

Figure 1.6.2 Address-trap-reset

### 1.6.3 Watchdog Timer Reset

Refer to section 2.4 "Watchdog Timer".

### 1.6.4 System-clock-reset

Clearing bits 7 in SYSCR2 to "0", system clock stops and causes the microcomputer to deadlock. This can be prevented by automatically generating a reset signal whenever bits 7, 6 and 5 in SYSCR2 = 000 is detected to continue the oscillation. The $\overline{\text{RESET}}$ pin output goes low from high-impedance. The reset time is about 8/fc to 24/fc [s] (0.5 to 1.5 μs at 16 MHz).

## 1.7 ROM Corrective Function

The ROM corrective function can patch the part (s) of on-chip ROM with some bugs.

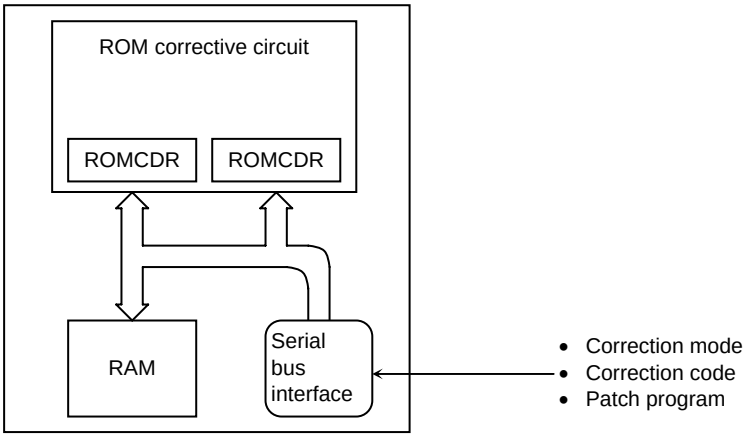
The ROM corrective function have two modes. One is to replaced the instruction on a certain address in the ROM with the jump instruction to branch into the RAM area where the patched codes (Program jump mode). The other is to replace a byte or a word (2 or 3 bytes) length data in the ROM with the patched data (Data replacement mode). Four independent location can be patched.

Note 1: When use ROM corrective circuit, it is necessary to contain a program which operates to load patched program and/or replacement data from external memory into an internal data RAM in an initial routine.

Note 2: The address of a instruction for IDLE mode can not be specificated as start address of corrective area.

Note 3: The BM88CS38N0A does not support the ROM corrective circuit. Use the TMP88PS38 to debug a program of this circuit. In this case, note the following.
In program jump mode, jump target addresses that can be specified with the TMP88CM38A/CP38A (002C0H to 006BFH) are different from those that can be specified with the TMP88PS38 (002C0H to 008BFH). Therefore, if a jump target address is within a range of 006C0H to 008BFH, it is necessary to change this addresse and also addresses for loading a patch program.

Example:



### 1.7.1 Configuration

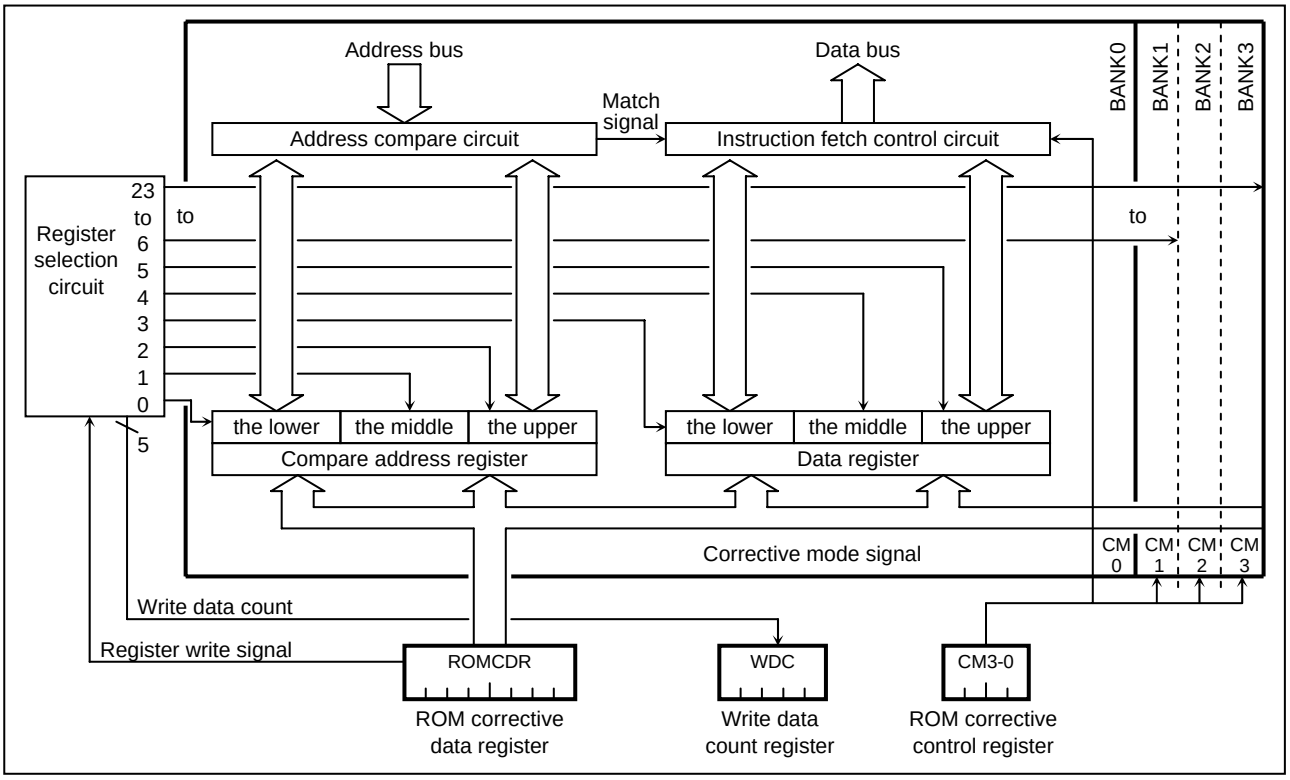



Figure 1.7.1 ROM Corrective Circuit

### 1.7.2 Control

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

ROM Corrective Control Register

| ROMCCR (00FE0H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | – | – | – | – | CM3 | CM2 | CM1 | CM0 | (Initial value: **** 0000) |

| | | | |
|---|---|---|---|
| CM3 | Corrective mode setting (BANK3) | 0: Proguram jump mode<br>1: Data replacement mode | R/W |
| CM2 | Corrective mode setting (BANK2) | | |
| CM1 | Corrective mode setting (BANK1) | | |
| CM0 | Corrective mode setting (BANK0) | | |

ROM Corrective Status Register

| ROMCSR (00FE1H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | – | – | – | WDC | | | | | (Initial value: ***0 0000) |

| | | | |
|---|---|---|---|
| WDC | Write data counter | Counting the number of the byte written in ROMCDR | Read only |

ROM Corrective Data Register

| ROMCDR (00FE2H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| ROMC | ROM Corrective data register | | Write only |

Figure 1.7.2 ROM Corrective Control Register, Status Register and ROM Corrective Data Register

(1) Enable and disable

The ROM corrective function is disabled after releasing reset. It is enabled after setting the data for one bank into ROMCDR. And the address-trap-reset is not generated when fetching an instruction from the RAM area except the address 02C0H to 08BFH.

After the ROM corrective function is enabled, it is necessary to reset the microcontroller in order to disable it.

(2) Data replacement mode

The ROM corrective function has the program jump mode and the data replacement mode.

By setting CMx (x: 0 to 3) in ROMCCR, the data replacement mode is selected.

(3) 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 set. (Figure 1.7.2)

ROM Corrective Data Register

ROMCDR (00FE2H)

| ROMC7 | ROMC6 | ROMC5 | ROMC4 | ROMC3 | ROMC2 | ROMC1 | ROMC0 |
|---|---|---|---|---|---|---|---|

(Initial value: 0000 0000)

| | | The value of WDC after writing a data to ROMCDR |
|---|---|---|
| | | 00000 (Initial value) |
| BANK0 | The lower start address of the corrective area (8 bits) | 00001 |
| | The middle start address of the corrective area (8 bits) | 00010 |
| | The upper start address of the corrective area (4 bits) | 00011 |
| | The lower 8 bits of the jump address/replacement data | 00100 |
| | The middle 8 bits of the jump address/replacement data | 00101 |
| | The upper 4 bits of the jump address/replacement data | 00110 |
| BANK1 | The lower start address of the corrective area (8 bits) | 00111 |
| | The middle start address of the corrective area (8 bits) | 01000 |
| | The upper start address of the corrective area (4 bits) | 01001 |
| | The lower 8 bits of the jump address/replacement data | 01010 |
| | The middle 8 bits of the jump address/replacement data | 01011 |
| | The upper 4 bits of the jump address/replacement data | 01100 |
| BANK2 | The lower start address of the corrective area (8 bits) | 01101 |
| | The middle start address of the corrective area (8 bits) | 01110 |
| | The upper start address of the corrective area (4 bits) | 01111 |
| | The lower 8 bits of the jump address/replacement data | 10000 |
| | The middle 8 bits of the jump address/replacement data | 10001 |
| | The upper 4 bits of the jump address/replacement data | 10010 |
| BANK3 | The lower start address of the corrective area (8 bits) | 10011 |
| | The middle start address of the corrective area (8 bits) | 10100 |
| | The upper start address of the corrective area (4 bits) | 10101 |
| | The lower 8 bits of the jump address/replacement data | 10110 |
| | The middle 8 bits of the jump address/replacement data | 10111 |
| | The upper 4 bits of the jump address/replacement data | 00000 |

Note 1: WDC value equals to the number of the byte stored in ROMCDR.

Note 2: ROMCDR is set in order of the lower (8 bits), the middle (8 bits) and the upper (4 bits) start address of the corrective area, the lower (8 bits), the middle (8 bits) and the upper (4 bits) of the jump address/the replacement data.

Figure 1.7.3 Banks and WDC Value of the Program Corrective Data Register

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

(1) The lower start address of the corrective area (8 bits)

(2) The middle start address of the corrective area (8 bits)

(3) The upper start address of the corrective area (4 bits)

(4) The lower jump address/replacement data (8 bits)

(5) The middle jump address/replacement data (8 bits)

(6) The upper jump address (4 bits)/replacement data

Note 1: Corrective addresses must have over five addresses each other.

Note 2: The address of a instruction for IDLE mode can not be specificated as start address of corrective area.

### 1.7.3 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

In the program jump mode, the system executes a jump instruction when the program execution reaches the instruction at the corrective ROM address, skips from the instruction which would have been executed, and executes an instruction at a preset jump address.

Clearing ROMCCR CMx (x: 0 to 3) to "0" puts the system in the program jump mode. Use ROMCDR to set the corrective ROM address and jump address.

When the start address of an erroneous program is a corrective ROM address, and that of the patch program is a jump address, the bug in the erroneous program can be fixed. Note that the patch program should end with a jump instruction, which causes a return to the built-in ROM.

Note: For program jump mode, the address to be corrected must be the start address of the instruction.

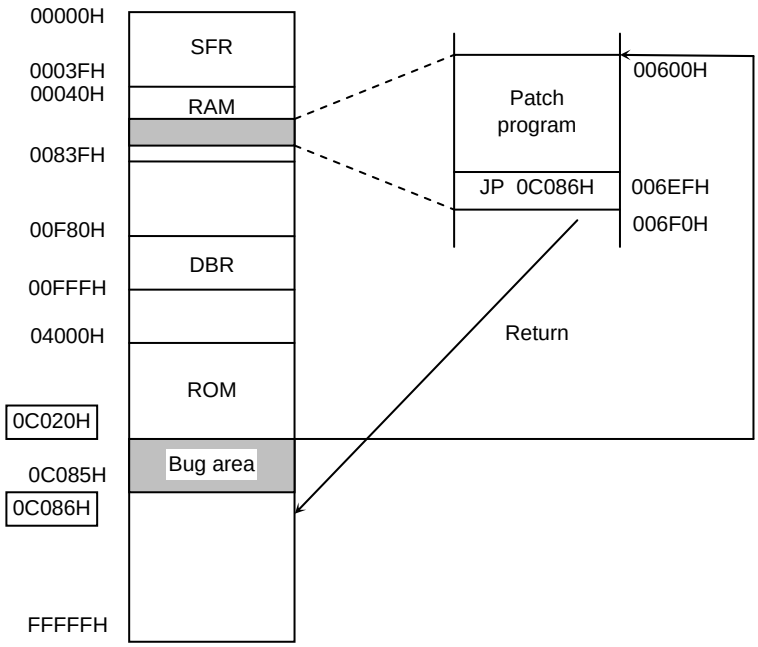
Example 1: Setting the program correction circuit with the initial routine

Using the initial routine program, which is executed right after reset, set the program correction circuit's register and stores the patch program into the built-in RAM as follows.

1. Read the flag, which indicates whether to use the program correction circuit, from the external memory.
2. If that circuit is not used, perform normal initial processing.
3. If it is used, clear CMx to 0 to establish the program jump mode.
4. Read the corrective ROM address and jump address from the external memory.
5. Set the corrective ROM address and jump address, which were read in step "4.", in ROMCDR.
6. Read the number of bytes for the patch program from the external memory.
7. Read the program with a number of bytes, equal to the byte count read in step "6.", from the external memory, and store that program into the built-in RAM.
8. Repeat steps "4." through "7." as many times as there are required banks.

Example 2: There is bugs on the locations from 0C020H to 0C085H

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 (00600H to 006EFH). The start address (0C020H) of the ROM necessary to patch is written to the corrective ROM address registers, and the start address (00600H) of the RAM area to patch is loaded onto the jump address registers. When the instruction at 0C020H is fetched, the instruction to jump into 00600H is unconditionally executed instead of the instruction at 0C020H, and the subsequent patch program codes are executed. The jump instruction at the end of the patch program codes returns to the ROM at 0C086H.




Note: Corrective address must be assigned to 1st byte of instruction codes on the program jump mode.

(2) Data replacement mode

In the data replacement mode, the system replaces reference data stored in the ROM area with the new instead of correcting the data reference instruction when that reference data is changed.

The program jump mode reduces the complexity of correcting the processing routine. However, when this mode is used, if there is a need to replace only the fixed data in ROM, the instruction to reference this ROM data should be corrected. Thus, a large amount of ROM is required for the patch program. To avoid this, the system has the data replacement mode. With this mode, three consecutive bytes of data can be replaced for each bank. (For an instruction which accesses only one byte, only the first byte can be replaced. For an instruction which accesses only two bytes, the two consecutive bytes can be replaced.) Setting ROMCCR CMx (x: 0 to 3) to "1" puts the system in the data replacement mode. Specify the start address of ROM data to be replaced as the corrective ROM address. Then, specify the new three-byte data as the patch data.

Note: For data replacement mode, the corrective address should be the address of fixed data (including a vector). (The operation code and operand cannot be changed.)

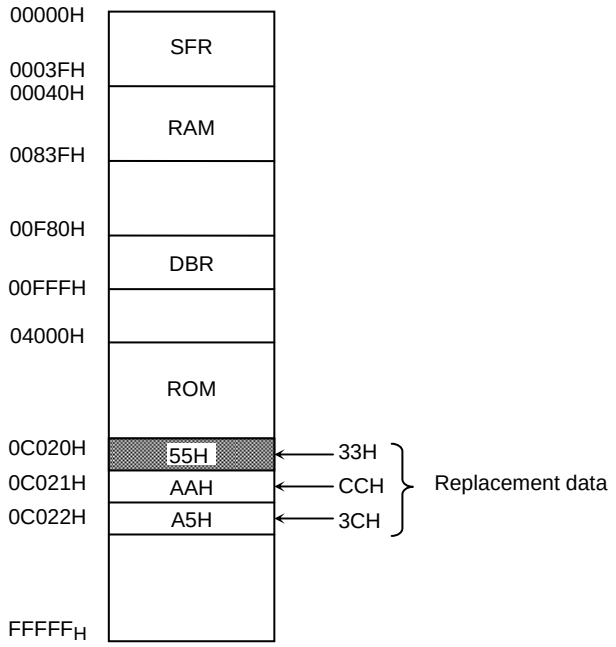
Example 1: Setting the program correction circuit with the initial routine

Using the initial routine program, which is executed right after reset, set the program correction circuit's register as follows.

1. Read the flag, which indicates whether to use the program correction circuit, from the external memory.
2. If that circuit is not used, perform normal initial processing.
3. If it is used, set CMx to "1" to establish the data replacement mode.
4. Read the address of the data to be replaced and the patch data from the external memory.
5. Set the address and patch data, which were read in step "4.", in ROMCDR.
6. Repeat steps "4." and "5." as many times as there are required banks.

Example 2: Replacing data 55H at 0C020H with 33H

Using the initial routine program, which is executed right after reset, read the start address of the data to be replaced and the patch data from the external memory. Set CMx (x: 0 to 3) to "1" to change the correction mode to the data replacement mode. Specify the start address (0C020H) of the data to be replaced as the corrective ROM address. Then, specify the new three-byte data (33H for 0C020H, CCH for 0C021H, and C3H for 0C022H) as the patch data.




1. At HL = 0C020H, Executing LD A, (HL) loads 33H in A. (Data replacement)
2. At HL = 0C021H, Executing LD A, (HL) loads AAH in A. (No data replacement)
3. At HL = 0C020H, Executing LD WA, (HL) loads CC33H in WA. (Data replacement)
4. At HL = 0C020H, Executing LD IX, (HL) loads CCC33H in IX. (Data replacement)

Note 1: Corrective address must be assigned to constant data area on the data replacement mode. (Ope-code and ope-rand can't be replaced by ROM correction circuit.)

Note 2: Instructions which includes "(HL+)" or "(−HL) " operation can't be replaced by ROM corrective circuit on the data replacement mode.

# 2. On-chip Peripheral Functions

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

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

The SFR are mapped to addresses 00000H to 0003FH, and DBR are mapped to address 00F80H to 00FFFH.

Figure 2.1.1 shows the list of the TMP88CS38/CM38A/CP38A SFRs and-DBRs.

| Address | Read | Write |
|---|---|---|
| 00000H | Reserved | |
| 00001 | Reserved | |
| 00002 | P2 port | |
| 00003 | P3 port | |
| 00004 | P4 port | |
| 00005 | P5 port | |
| 00006 | P6 port | |
| 00007 | P7 port | |
| 00008 | – | P5CR1 (P5 port I/O control1) |
| 00009 | – | P7CR (P7 port I/O control) |
| 0000A | Reserved | |
| 0000B | Reserved | |
| 0000C | – | P4CR (P4 port I/O control) |
| 0000D | – | P6CR (P6 port I/O control) |
| 0000E | ADCCRA (AD converter control A) | |
| 0000F | ADCCRB (AD converter control B) | |
| 00010 | $TC1DRA_L$ (Timer register 1A) | |
| 00011 | $TC1DRA_H$ | |
| 00012 | $TC1DRB_L$ (Timer register 1B) | – |
| 00013 | $TC1DRB_H$ | – |
| 00014 | TC1CR (TC1 control) | |
| 00015 | – | TC2CR (TC2 control) |
| 00016 | – | $TC2DR_L$ (Timer register 2) |
| 00017 | – | $TC2DR_H$ |
| 00018 | TC3DRA (Timer register 3A) | |
| 00019 | TC3DRB (Timer register 3B) | – |
| 0001A | – | TC3CR (TC3 control) |
| 0001B | – | TC4DR (Timer register 4) |
| 0001C | – | TC4CR (TC4 control) |
| 0001D | ORDSN (OSD control) | |
| 0001E | $ORCRA_L$ (OSD control) | |
| 0001F | $ORCRA_H$ (OSD control) | |

| Address | Read | Write |
|---|---|---|
| 00020H | SBISRA (SBI statusA) | SBICRA (SBI control register A) |
| 00021 | SBIDBR (SBI data buffer) | |
| 00022 | – | $I^2CAR$ ($I^2C$ bus address) |
| 00023 | SBISRB (SBI statusB) | SBICRB (SBI control register B) |
| 00024 | – | $ORDMA_L$ (OSD control) |
| 00025 | – | $ORDMA_H$ (OSD control) |
| 00026 | RCSR (TC3 status) | RCCR (TC3 control) |
| 00027 | PMPXCR (Port control) | |
| 00028 | – | PWMCR1A (PWM control 1A) |
| 00029 | – | PWMCR1B (PWM control 1B) |
| 0002A | – | PWMDBR1 (PWMDBR1) |
| 0002B | – | P3CR1 (P3 I/O control) |
| 0002C | $EIR_E$ (Interrupt enable register) | |
| 0002D | $EIR_D$ | |
| 0002E | $IL_E$ (Interrupt latch) | |
| 0002F | $IL_D$ | |
| 00030 | CGCR (Divider control) | |
| 00031 | ADCDR1 (AD conversion result) | |
| 00032 | ADCDR2 (AD conversion result) | |
| 00033 | Reserved | |
| 00034 | – | WDTCR1 (Watchdog timer control) |
| 00035 | – | WDTCR2 |
| 00036 | TBTCR (TBT/TG control) | |
| 00037 | – | EINTCR (External interrupt control) |
| 00038 | $SYSCR1$ (System control) | |
| 00039 | $SYSCR2$ | |
| 0003A | $EIR_L$ (Interrupt enable register) | |
| 0003B | $EIR_H$ | |
| 0003C | $IL_L$ (Interrupt latch) | |
| 0003D | $IL_H$ | |
| 0003E | $PSW_L$ (Program status word) | |
| 0003F | $PSW_H$ | |

(a) Special function registers

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 0003FH with assembler symbols, use GRBS.
Address 0003EH must be GPSW/GFLAG.

Figure 2.1.1 (a) SFR

| Address | Read | Write |
|---|---|---|
| 00F80H | ORDON (OSD control) | |
| 81 | – | OSD control register |
| ≈ | – | OSD control register |
| A1 | ORCLKC (OSD clock status) | ORCLKF (OSD clock control) |
| A2 | – | OSD control register |
| ≈ | – | OSD control register |
| B9 | ORIRC (OSD display counter) | ORIRC (OSD interrupt control) |
| BA | – | OSD control register |
| ≈ | – | OSD control register |
| C0 | Reserved | |
| ≈ | Reserved | |
| D0 | IDLEINV (Key-on wakeup status) | IDLECR (Key-on wakeup control) |
| D1 | Reserved | |
| ≈ | Reserved | |
| D8 | SINTCR (Data slicer interrupt control) | |
| D9 | – | DACLCR (Sync. tip slice level setting) |
| DA | SLVLCR (Slice level control) | |
| DB | SIFDR1 (Caption data 1st byte) | – |
| DC | SIFDR2 (Caption data 2nd byte) | – |
| DD | SIFSR (Data slicer status) | – |
| DE | – | – |
| DF | SIFS1R (Data slicer status 2) | SIFSMS1 (Data slicer mode setting) |
| E0 | ROMCCR (ROM corrective control) | |
| E1 | ROMCSR (ROM corrective status) | – |
| E2 | – | ROMCDR (ROM corrective data) |
| E3 | Reserved | |
| E4 | JECR (Jitter elimination control) | |
| E5 | JESR (Jitter elimination status) | – |
| E6 | – | TVSCR (Test video signal output) |
| E7 | Reserved | |
| E8 | RXCR1 (Remote control receive control 2) | |
| E9 | RXCR2 (Remote control receive control 1) | |
| EA | RXCTR (Remote control receive counter) | – |
| EB | RXDBR (Remote control receive data buffer) | – |
| EC | RXSR (Remote control status) | – |
| ED | Reserved | |
| EE | FC8CR (FC8 control) | |
| EF | Reserved | |
| F0 | Reserved | |
| F1 | SCCRB (Serial clock source control) | SCSR (Serial clock source status) |
| F2 | Reserved | |
| F3 | Reserved | |
| F4 | Reserved | |
| F5 | – | PWMCR2A (PWM control 2A) |
| F6 | – | PWMCR2B (PWM control 2B) |
| F7 | – | PWMDBR2 (PWM data buffer) |
| F8 | Reserved | |
| | Reserved | |
| FE | – | PSELCR (P3, P5 control 2) |
| FF | Reserved | |

(b) Data buffer registers

Note 1: Do not access reserved areas by the program.

Note 2: –: Cannot be accessed.

Note 3: Write-only registers cannot use the read-modify-write instructions (Bit manipulation instructions such as SET, CLR, etc. and logical operation instructions such as AND, OR, etc.).

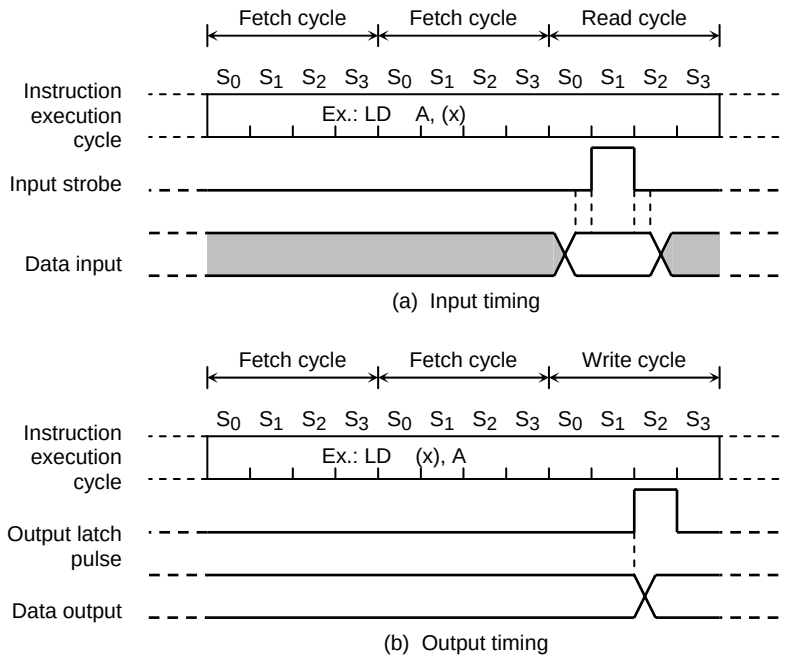
Figure 2.1.2 (b) DBR

## 2.2 I/O Ports

The TMP88CS38/CM38A/CP38A 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, data slicer analog 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, key-on wakeup input, serial bus interface input/output, analog input and I output from OSD circuitry. |
| 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, test video signal output and key-on wakeup input |
| 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 2.2.1 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.




Note: The positions of the read and write cycles may vary, dispending on the instruction.

Figure 2.2.1 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.

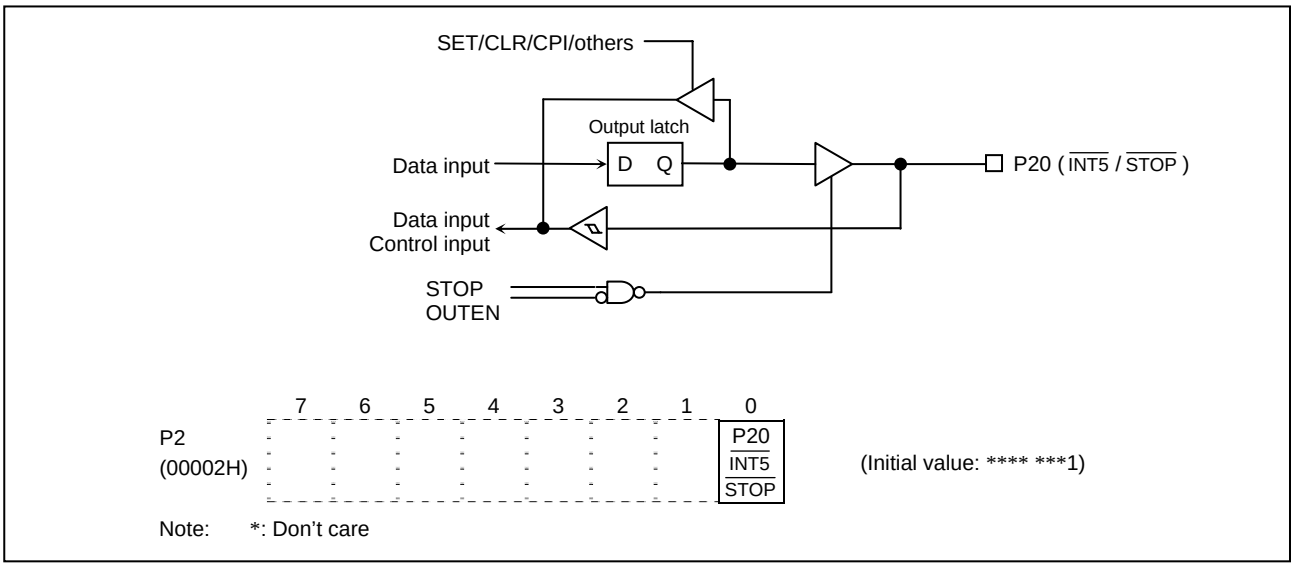



Figure 2.2.2 Port P2

### 2.2.2 Port P3 (P35 to P30)

Port P3 is an 6-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 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 $I^2C$ bus interface input/output, the input pins should be set to the input mode. When used as a secondary function output pin except $I^2C$ 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 $I^2C$ 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.

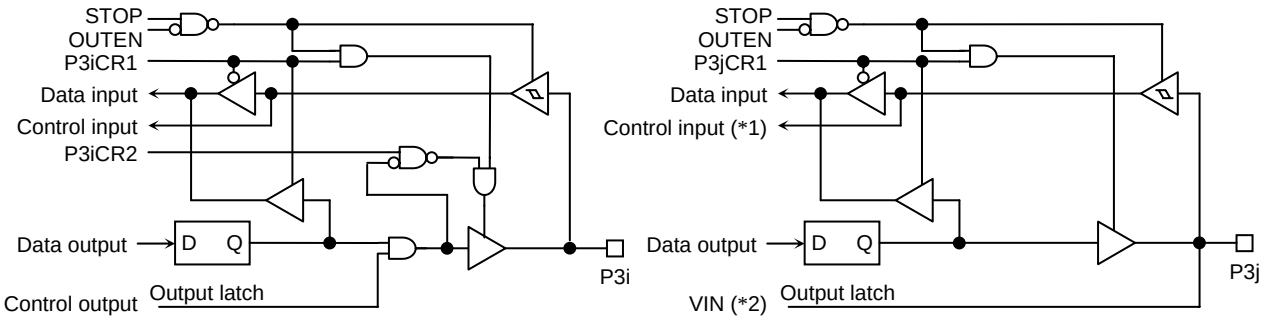
Note: Input mode port is read the state of input pin. When input/output mode is used mixed, the contents of output latch setting input mode may be changed by executing bit manipulation instructions.

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 ← P33 to P30
```




(a) P35 to P34 (b) P33 to P30

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| P3 (00003H) | - | - | P35 SDA0 | P34 SCL0 | P33 VIN0 TC4 | P32 VIN1 CSIN | P31 INT4 TC3 | P30 INT3 RXIN | (Initial value: **11 1111) |
| P3CR1 (0002BH) | | | P35CR1 | P34CR1 | P33CR1 | P32CR1 | P31CR1 | P30CR1 | (Initial value: **00 0000) |
| PSELCR (0FFEH) | "0" | "0" | P35CR2 | P34CR2 | "0" | P52CR2 | P51CR2 | "0" | (Initial value: 0*00 *00*) |

| | | | |
|---|---|---|---|
| P3CR1 | I/O control for P3 | 0: Input mode<br>1: Output mode | Write only |
| P3CR2 | I/O control for P3 | 0: Sink open drain<br>1: Tri-state | Write only |

(*1) only P33, P31, P30

(*2) only P33, P32

Note 1: *: Don't care, i = 5 to 4, j = 3 to 0

Note 2: P3CR1 cannot used the read-modify-write instructions.
(Bit manipulation instructions such as SET, CLR, etc. and logical operation such as AND, OR, etc.)

Note 3: Clear bit7, 6, 3 and 0 to "0" in PSELCR.

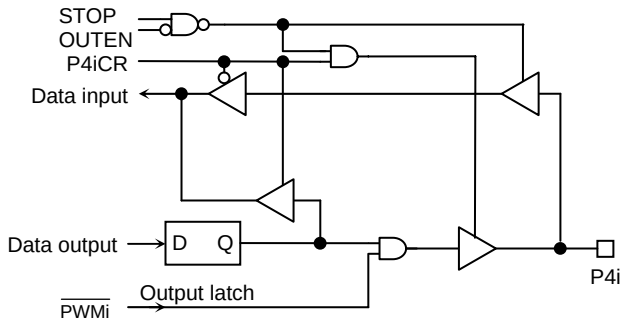
Figure 2.2.3 Port P3 and P3CR

### 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.




| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| P4 (00004H) | P47 / PWM7 | P46 / PWM6 | P45 / PWM5 | P44 / PWM4 | P43 / PWM3 | P42 / PWM2 | P41 / PWM1 | P40 / PWM0 | (Initial value: 1111 1111) |
| P4CR (0000CH) | P47CR | P46CR | P45CR | P44CR | P43CR | P42CR | P41CR | P40CR | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| P4CR | I/O control for port P4 | 0: Input mode<br>1: Output mode | Write only |

Note 1: i = 7 to 0.

Note 2: P4CR cannot used the read-modify-write instructions.
(Bit manipulation instructions such as SET, CLR, etc. and logical operation such as AND, OR, etc.)

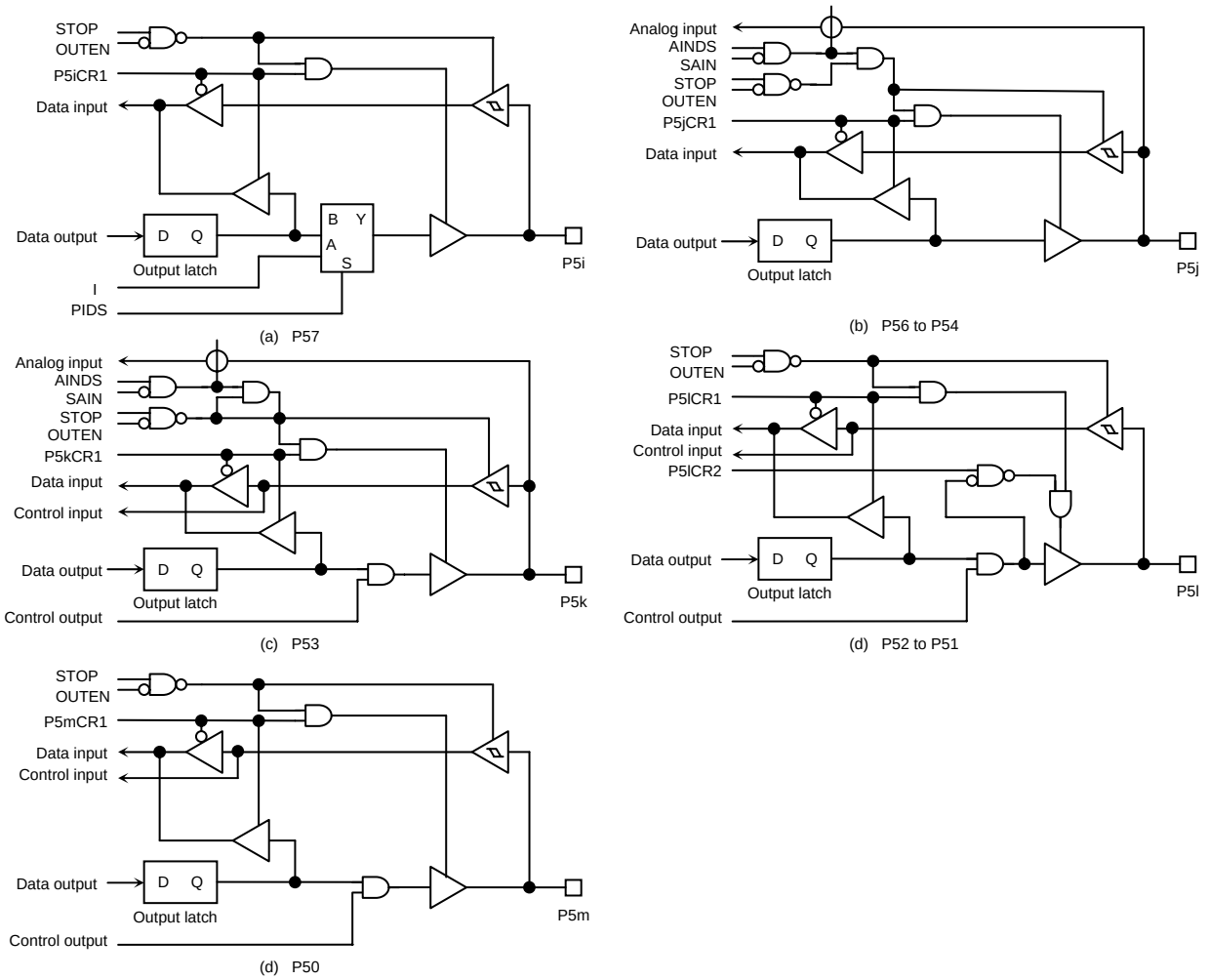
Figure 2.2.4 Port P4 and P4CR

### 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 AD converter analog input, a pulse width modulation (PWM) output external interrupt input, timer/counter input, serial bus interface input/output, and an on screen display (OSD) output (I signal). When used as a secondary function input pin except $I^2C$ bus interface input/output, the input pins should be set to the input mode. When used as a secondary function output pin except $I^2C$ 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 $I^2C$ 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. When P57 is used as an OSD output pin, the output pin should be set to the output mode and beforehand the port 6 data selection register (PIDS) should be clear to "0". When used as port P5, the port 6 data selection register (PIDS) 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.




| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P5 (00005H) | P57 I | P56 AIN3 | P55 AIN2 | P54 AIN1 | P53 INT2 TC1 $\overline{\text{SCK1}}$ AIN0 | P52 SO1 SDA1 | P51 $\overline{\text{PWM9}}$ SL1 SLC1 | P50 $\overline{\text{PWM8}}$ INT0 TC2 |

(Initial value: 1111 1111)

| P5CR1 (00008H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P57CR1 | P56CR1 | P55CR1 | P54CR1 | P53CR1 | P52CR1 | P51CR1 | P50CR1 |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P5CR1 | I/O control for P5 | 0: Input mode<br>1: Output mode | Write only |

| PSELCR (00FFEH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | "0" | "0" | P35CR2 | P34CR2 | "0" | P52CR2 | P51CR2 | |

(Initial value: 0*00 *00*)

| | | | |
|---|---|---|---|
| P5CR2 | I/O control for P5 | 0: Skin open drain<br>1: Tri-state | Write only |

| ORP6S (00FBAH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P67S | P66S | P65S | P64S | PIDS | YBLCS | MPXS | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| PIDS | Selection of the output data for port P57 | 0: The OSD output (I)<br>1: Port P57 output latch | Write only |

Note 1: *: Don't care, i = 7, j = 6 to 4, k = 3, l = 2 to 1, m = 0.

Note 2: P5CR1 cannot be used the read-modify-write instructions.
(Bit manipulation instructions such as SET, CLR, etc. and logical operation such as ADN, OR, etc.)

Note 3: Clear bit7, 6 and 3 to "0" in PSELCR.

Figure 2.2.5 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". P63 to P60 are sink open-drain ports. During reset, P6CR is initialized to "0", which configures port P6 as an input. The P6 output latches are also initialized to "1".

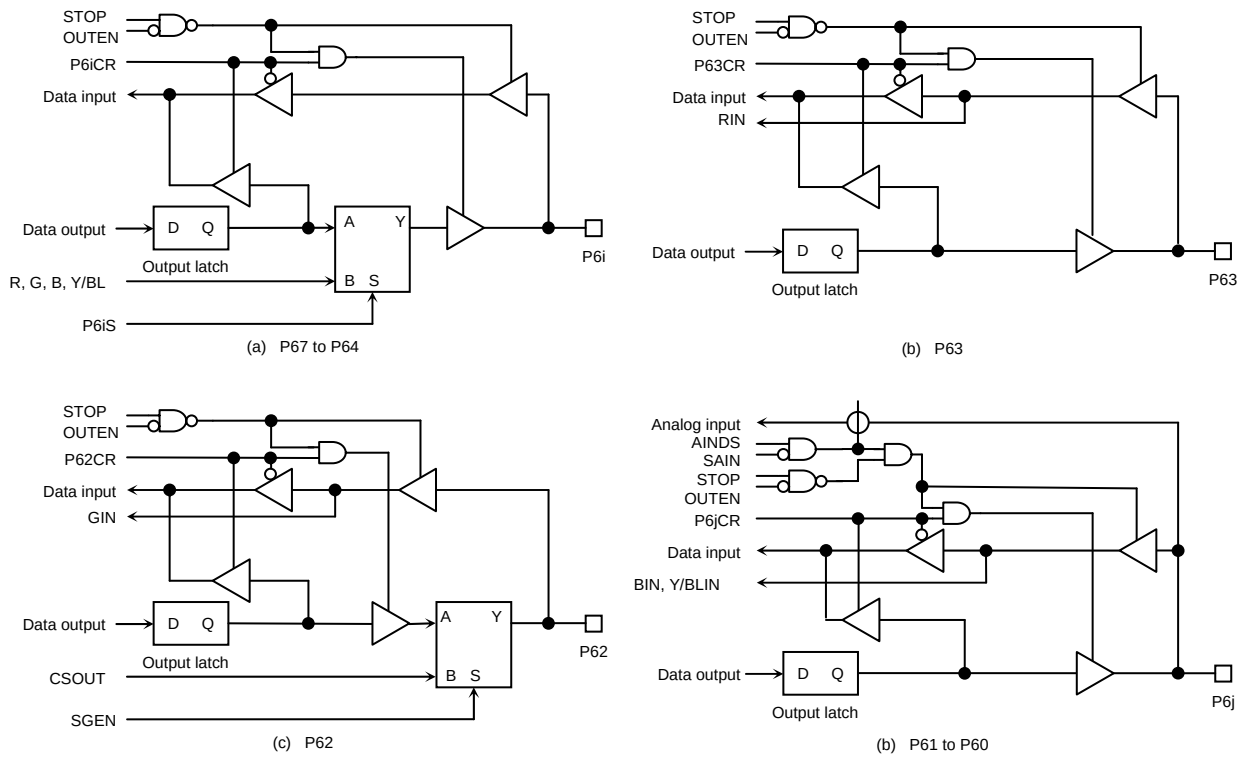
Data is written into the output latch regardless of the P6CR contents. Therefore initial output data should be written into the output latch before setting P6CR.

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 AD converter analog input. When used as a test video signal output pin, the output pins should be set to the output mode and beforehand the signal control register (SGEN) should be set to "1". 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". When used as port P6, the signal control register (P67 to P64) 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.

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 ← 00001111B
```




| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| P6 (00006H) | P67 Y/BL | P66 B | P65 G | P64 R | P63 RIN | P62 GIN CSOUT | P61 BIN AIN5 | P60 Y/BLIN AIN4 | (Initial value: 1111 1111) |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| P6CR (0000DH) | P67CR | P66CR | P65CR | P64CR | P63CR | P62CR | P61CR | P60CR | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| P6CR | I/O control for port P6 | 0: Input mode<br>1: Output mode | Write only |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| SGCR (00FE6H) | SGEN | SGVBLK | SGPAL | SGIV | SGCHS | | SGPAT | | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| SGEN | Function selection | 0: Disable<br>1: Enable | Write only |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| ORP6S (00FBAH) | P67S | P66S | P65S | P64S | PIDS | YBLCS | MPXS | | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| P67S to P64S | Selection of the output data for port P6i | 0: The OSD output (R, G, B, Y/BL)<br>1: Port P6i output latch | Write only |

Note 1: ∗: Don't care, i = 7 to 4, j = 1 to 0.

Note 2: P6CR and ORP6S cannot used with the read-modify-write instructions. (Bit manipulations such as SET, CLR, etc. and logical operation such as ADN, OR, etc.)

Note 3: Clear bit2 and 0 to "0" in TVSCR

Figure 2.2.6 Ports P6, P6CR, and P67S to P64S

### 2.2.6 Port P7 (P71 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.

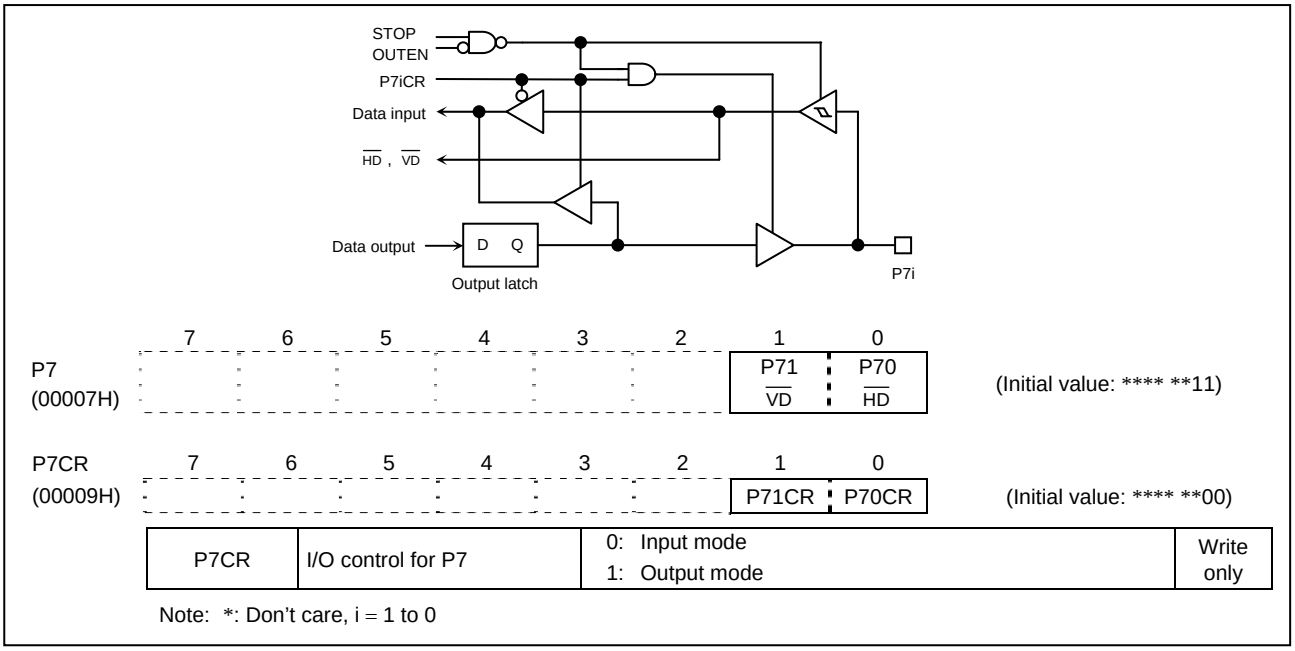



Figure 2.2.7 Ports P7

## 2.3 Time Base Timer (TBT)

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 2.3.1.

An INTTBT is generated on the first falling 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.

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), 00000010B      ;   TBTCK = "010"
LD      (TBTCR), 00001010B      ;   TBTEN = "1"
SET     (EIRL). 6
```

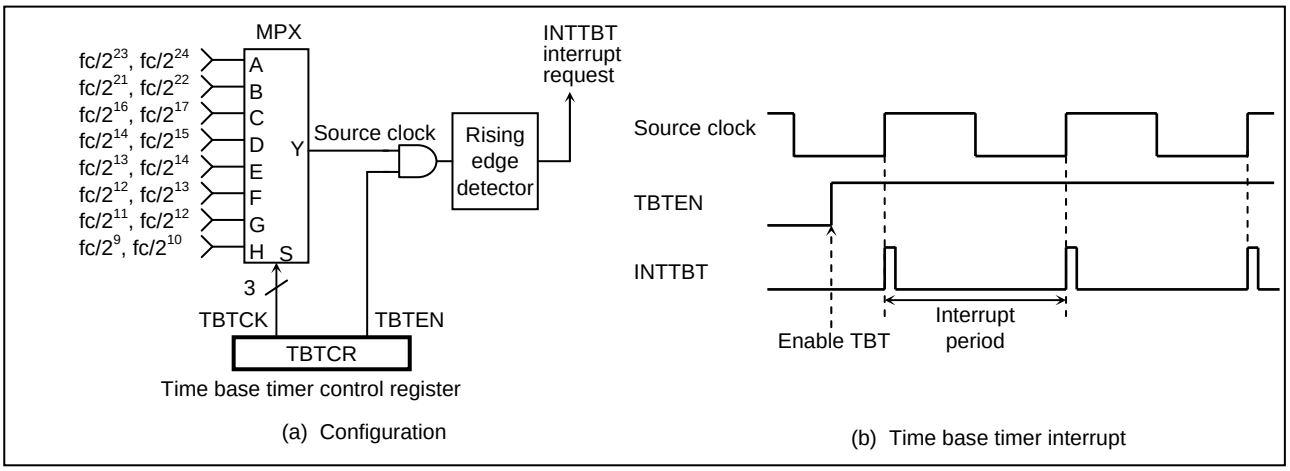



Figure 2.3.1 Time Base Timer

| TBTCR (00036H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | "0" | – | – | "0" | TBTEN | TBTCK | | |

(Initial value: 0**0 0***)

| | | | | | |
|---|---|---|---|---|---|
| TBTEN | Time base timer enable/disable | 0: Disable<br>1: Enable | | | Write only |
| TBTCK | Time base timer interrupt frequency select | | NORMAL, IDLE mode | | |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 000 | $fc/2^{23}$ [Hz] | $fc/2^{24}$ [Hz] | |
| | | 001 | $fc/2^{21}$ | $fc/2^{22}$ | |
| | | 010 | $fc/2^{16}$ | $fc/2^{17}$ | |
| | | 011 | $fc/2^{14}$ | $fc/2^{15}$ | |
| | | 100 | $fc/2^{13}$ | $fc/2^{14}$ | |
| | | 101 | $fc/2^{12}$ | $fc/2^{13}$ | |
| | | 110 | $fc/2^{11}$ | $fc/2^{12}$ | |
| | | 111 | $fc/2^{9}$ | $fc/2^{10}$ | |

Note 1: fc: High-frequency clock [Hz], *: Don't care

Note 2: TBTCR is a write-only register and must not be used with any of read-modify-write instruction.

Note 3: Set bit7 and 4 in TBTCR to "0".

Figure 2.3.2 Time Base Timer and Divider Output Control Register

Table 2.3.1 Time Base Timer Interrupt Frequency (Example: at fc = 16 MHz)

| TBTCK | Time Base Timer Interrupt Frequency [Hz] | |
|---|---|---|
| | NORMAL, IDLE mode | |
| | DV1CK = 0 | DV1CK = 1 |
| 000 | 1.90 | 0.95 |
| 001 | 7.62 | 3.81 |
| 010 | 244.14 | 122.07 |
| 011 | 976.56 | 488.28 |
| 100 | 1953.12 | 976.56 |
| 101 | 3906.25 | 1953.12 |
| 110 | 7812.50 | 3906.25 |
| 111 | 31250 | 15625 |

## 2.4 Watchdog Timer (WDT)

The watchdog timer is a fail-safe system to rapidly detect the CPU malfunctions such as endless looping caused by noise or the like, or deadlock and resume the CPU to the normal state.

The watchdog timer signal for detecting malfunction can be selected either a reset output or a pseudo 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.

Note: Care must be given in system design so as to protect the watchdog timer from disturbing noise. Otherwise the watchdog timer may not fully exhibit its functionality.

### 2.4.1 Watchdog Timer Configuration

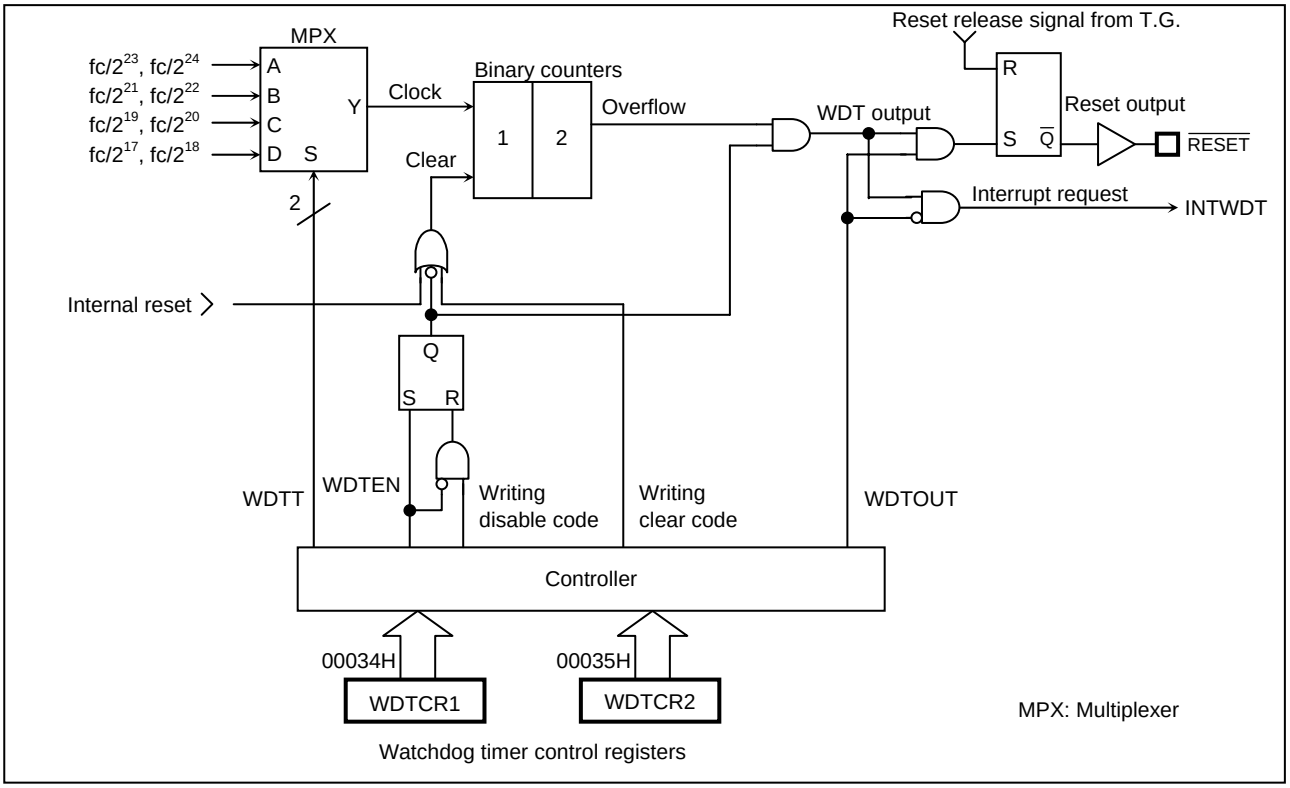



Figure 2.4.1 Watchdog Timer Configuration

### 2.4.2 Watchdog Timer Control

Figure 2.4.2 shows the watchdog timer control registers (WDTCR1, WDTCR2). The watchdog timer is automatically enabled after reset.

(1) Malfunction detection methods using the watchdog timer

The CPU malfunction is detected at follows.

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

Note: The watchdog timer consists of an internal divider and two-stage binary counter. Writing the clear code ($4E_H$) clears the binary counter, but not the internal divider. The minimum overflow time for the binary counter might be three quarters of the WDTCR1 (WDTT) time setting depending on when the clear code ($4E_H$) is written into the WDTCR2 register. So, write the clear code on a cycle which is shorter than that minimum overflow time.

If the CPU malfunctions such as endless looping or deadlock occur for any cause, the watchdog timer output will become active at the rising of an overflow from the binary counters unless the binary counters are cleared. At this time, when WDTOUT = 1 a reset is generated, which drivers the $\overline{\text{RESET}}$ pin low to reset the internal hardware and the external circuit. 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 the STOP/IDLE mode is released.

Example: Sets the watchdog timer detection time to $2^{21}$/fc [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
Within 3/4 of WDT                      (always clear immediately before and after changing WDTT)
detection time
        LD    (WDTCR2), 4EH          ; Clears the binary counters
Within 3/4 of WDT
detection time
        LD    (WDTCR2), 4EH          ; Clears the binary counters
```

Watchdog Timer Register 1

WDTCR1 (00034H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | WDTEN | WDTT | | WDTOUT |

(Initial value: **** 1001)

| | | | | | |
|---|---|---|---|---|---|
| WDTEN | Watchdog timer enable/disable | 0: Disable (It is necessary to write the disable code to WDTCR2)<br>1: Enable | | | Write only |
| WDTT | Watchdog timer detection time [s] | | NORMAL mode | | |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 00 | $2^{25}$ /fc | $2^{26}$/fc | |
| | | 01 | $2^{23}$/fc | $2^{24}$/fc | |
| | | 10 | $2^{21}$/fc | $2^{22}$/fc | |
| | | 11 | $2^{19}$/fc | $2^{20}$/fc | |
| WDTOUT | Watchdog timer output select | 0: Interrupt request<br>1: Reset output | | | |

Note 1: WDTOUT cannot be set to “1” by program after clearing WDTOUT to “0”.

Note 2: fc: High-frequency clock [Hz], *: Don’t care

Note 3: WDTCR1 is a write-only register and must not be used with any of read-modify-write instructions.

Note 4: The watchdog timer must be disabled or the counter must be cleared immediately before entering to the STOP mode. When the counter is cleared, the counter must be cleared again immediately after releasing the STOP mode.

Note 5: Just right before disabling the watchdog timer, disable the acceptance of interrupts (DI) and clear the watchdog timer.
If the watchdog timer is disabled under conditions other than the above, the proper operation cannot be guaranteed.

Watchdog Timer Register 2

WDTCR2 (00035H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: **** ****)

| | | | |
|---|---|---|---|
| WDTCR2 | Watchdog timer control code write register | $4E_H$: Watchdog timer binary counter clear (Clear code)<br>$B1_H$: Watchdog timer disable (Disable code)<br>Others: Invalid | Write only |

Note 1: The disable code is invalid unless written when WDTEN = 0.

Note 2: *: Don’t care

Note 3: The binary counter of the watchdog timer must not be cleared by the interrupt task.

Note 4: Clears the binary counter does not clear the source clock.
It is recommended that the time to clear is set to 3/4 of the detecting time.

Note 5: The watchdog timer counter must be disabled by writing the disable code ($B1_H$) to WDRCR2 after writing WDTCR2 to. “$4E_H$”.

Figure 2.4.2 Watchdog Timer Control Registers

(2) Watchdog timer enable

The watchdog timer is enabled by setting WDTEN (Bit3 in WDTCR1) to "1". WDTEN is initialized to "1" during reset, so the watchdog timer operates immediately after reset is released.

Example: Disables watchdog timer

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

(3) Watchdog timer disable

To disable the watchdog timer, clear the interrupt mask enable flag (IMF) to "0" and write the clear code ($4E_H$) into WDTCR2. Then, clear WDTEN (Bit3 in WDTCR1) to "0". When WDTEN is "0", the watchdog timer is disabled by writing the disable code ($B1_H$) into WDTCR2. If WDTEN is cleared to "0" after the disable code has been written into WDTCR2, the watchdog timer is not disabled. While it is disabled, its binary counter is cleared.

Example:

```
DI                                  ;    Disables interrupt acceptance.
LD      (WDTCR2), 4EH               ;    Clears the watchdog timer.
LDW     (WDTCR1), B101H             ;    Disables the watchdog timer.
EI                                  ;    Enables interrupt acceptance.
```

Table 2.4.1 Watchdog Timer Detection Time (Example: fc = 16 MHz)

| WDTT | Watchdog timer detection time [s] | |
|---|---|---|
| | NORMAL mode | |
| | DV1CK = 0 | DV1CK = 1 |
| 00 | 2.097 | 4.194 |
| 01 | 524.288 m | 1.048 |
| 10 | 131.072 m | 262.1 m |
| 11 | 32.768 m | 65.5 m |

### 2.4.3 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 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, 023FH                   ;    Sets the stack pointer
LD      (WDTCR1), 00001000B         ;    WDTOUT ← 0
```

### 2.4.4 Watchdog Timer Reset

If the watchdog timer output becomes active, a reset is generated, which drivers the $\overline{\text{RESET}}$ pin (Sink open drain input/output with pull-up) low to reset the internal hardware. The reset output time is about 8/fc to 24/fc [s] (0.5 to 1.5 μs at fc = 16.0 MHz).

Note: If there is any fluctuation in the oscillation frequency at the start of clock oscillation, the reset time includes error. Thus, regard the reset time as an approximate value.

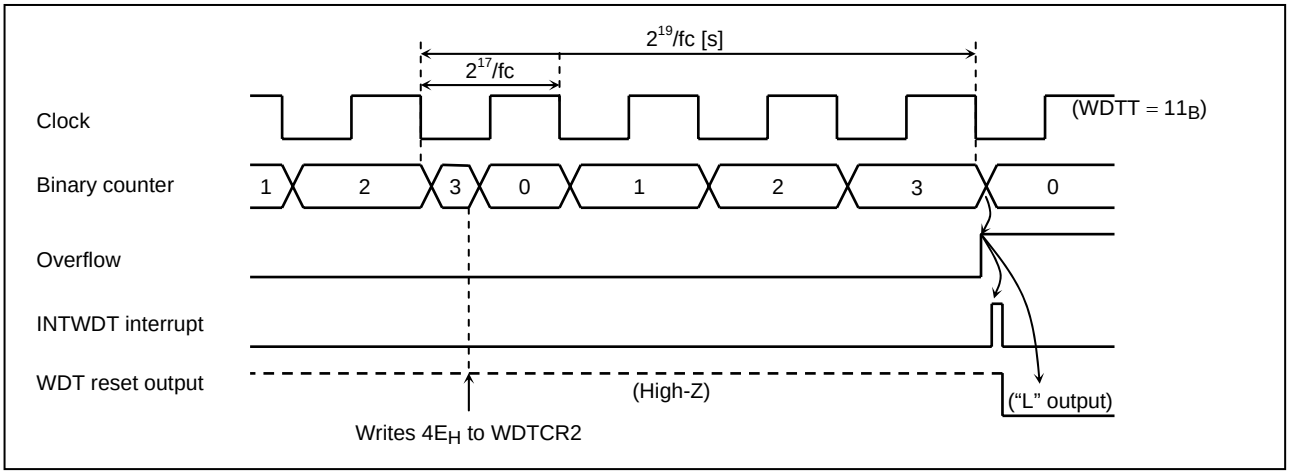



Figure 2.4.3 Watchdog Timer Interrupt/Reset

## 2.5 16-Bit Timer/Counter 1 (TC1A)

### 2.5.1 Configuration

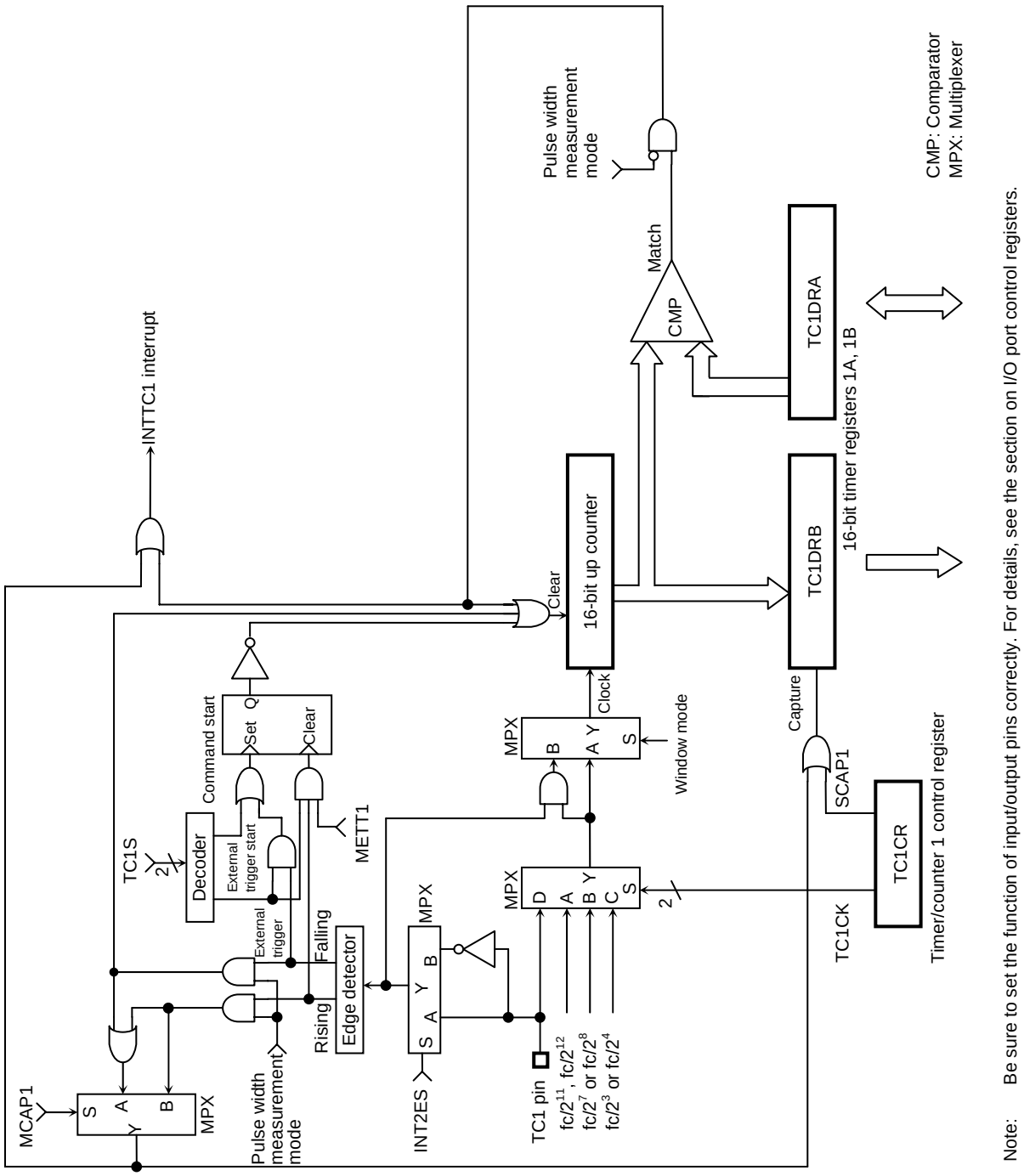



Figure 2.5.1 Timer/Counter 1

### 2.5.2 Control

The timer/counter 1 is controlled by a timer/counter 1 control register (TC1CR) and two 16-bit timer registers (TC1DRA and TC1DRB).

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TC1DRA (00010, 00011H) | $TC1DRA_H$ (00011H) | | | | | | | | $TC1DRA_L$ (00010H) | | | | | | | |
| | Read/Write | | | | | | | | | | | | | | | |
| TC1DRB (00012, 00013H) | $TC1DRB_H$ (00013H) | | | | | | | | $TC1DRB_L$ (00012H) | | | | | | | |
| | Read only | | | | | | | | | | | | | | | |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TC1CR (00014H) | "0" | ACPAP1 MCAP1 METT1 MPPG1 | TC1S | | TC1CK | | TC1M | |

Read/Write
(Initial value: 0000 0000)

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| TC1M | TC1 operating mode select | 00: Timer/external trigger timer/event counter mode<br>01: Window mode<br>10: Pulse width measurement mode<br>11: Reserved | | | | | | | | | R/W |
| TC1CK | TC1 source clock select [Hz] | | NORMAL, IDLE mode | | | | | | | | |
| | | | DV7CK = 0, DVCK = 00 | | | | | | | | |
| | | | DV1CK = 0 | DV1CK = 1 | | | | | | | |
| | | 00 | $fc/2^{11}$ | $fs/2^{12}$ | | | | | | | |
| | | 01 | $fc/2^{7}$ | $fc/2^{8}$ | | | | | | | |
| | | 10 | $fc/2^{3}$ | $fc/2^{4}$ | | | | | | | |
| | | 11 | External clock (TC1 pin input) | | | | | | | | |
| TC1S | TC1 start control | | | | Timer | Extend | Event | Window | Pulse | PPG | |
| | | 00: Stop and counter clear | | | ○ | ○ | ○ | ○ | ○ | ○ | |
| | | 01: Command start | | | ○ | × | × | × | × | × | |
| | | 10: External trigger start at the rising edge | | | × | ○ | ○ | ○ | ○ | ○ | |
| | | 11: External trigger start at the falling edge | | | × | ○ | ○ | ○ | ○ | ○ | |
| ACAP1 | Auto capture control | 0: Auto-capture disable | 1: Auto-capture enable | | | | | | | | |
| MCAP1 | Pulse width measurement mode control | 0: Double edge capture | 1: Single edge capture | | | | | | | | |
| METT1 | External trigger timer mode control | 0: Trigger start | 1: Trigger start and stop | | | | | | | | |

Note 1: fc: High-frequency clock [Hz]

Note 2: The timer register consists of two shift registers. A value set in the timer register is put in effect at the rising edge of the first source clock pulse that occurs after the upper data (TC1DRAH) are written. Therefore, the lower byte must be written before the upper byte (It is recommended that a 16-bit access instruction be used in writing). Writing only the lower data (TC1DRAL) does not put the setting of the timer register in effect.

Note 3: Set the mode, source clock PPG control and timer F/F control when TC1 stops (TC1S = 00).

Note 4: Auto capture can be used in only timer, event counter, and window modes.

Note 5: Values to be loaded to timer registers must satisfy the following condition.
TC1DRA > TC1DRB, TC1DRA > 1

Note 6: Always write "0" to TFF1 except PPG output mode.

Note 7: On entering STOP mode, the TC1 start control (TC1S) is cleared to "00" automatically. So, the timer stops. Once the STOP mode has been released, to start using the timer counter, set TC1S again.

Note 8: In the Auto-capture function, when the capture value is read after stop and clear counter or Auto-capture disable is executed by the TC1 start control (TC1S), the correct capture value might not be able to be read.When using Auto-capture function, set capture to enable.

Note 9: Since the up-counter value is captured into TC1DRB by the source clock of up-counter after setting TC1CR<ACAP1> to "1". Therefore, to read the captured value, wait at least one cycle of the internal source clock before reading TC1DRB for the first time.

Figure 2.5.2 Timer Registers and TC1 Control Register

### 2.5.3 Function

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

(1) Timer mode

In this mode, counting up is performed using the internal clock. The contents of TC1DRA are compared with the contents of 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 up counter can be transferred to TC1DRB by setting ACAP1 (Bit6 in TC1CR) to "1" (Software capture function). (Auto-capture function)

Table 2.5.1 Source Clock (Internal clock) for Timer/Counter 1 (Example: at fc = 16.0 MHz)

| TC1CK | NORMAL, IDLE mode | | | |
|---|---|---|---|---|
| | DV1CK = 0 | | DV1CK = 1 | |
| | Resolution [μs] | Maximum time setting [s] | Resolution [μs] | Maximum time setting [s] |
| 00 | 128.0 | 8.39 | 256.0 | 16.78 |
| 01 | 8.0 | 0.524 | 16.0 | 1.049 |
| 10 | 0.5 | 32.77 m | 1.0 | 65.54 m |

Example 1: Sets the timer mode with source clock $fc/2^{11}$ [Hz] and generates an interrupt 1 later (at fc = 16 MHz)

```
LDW     (TC1DRA), 1E84H          ;   Sets the timer register
                                     (1 s ÷ 2^11/fc = 1E84H)
DI
SET     (EIRL). 4                ;   Enable INTTC1
EI
LD      (TC1CR), 00000000B       ;   Selects the source clock and mode
LD      (TC1CR), 00010000B       ;   Starts TC1
```

Example 2: Auto capture

```
LD      (TC1CR), 01010000B       ;   ACAP1 ← 1 (Capture)
:       :                        ;   Wait at least one cycle of the internal source clock
LD      WA, (TC1DRB)             ;   Reads the capture value
```

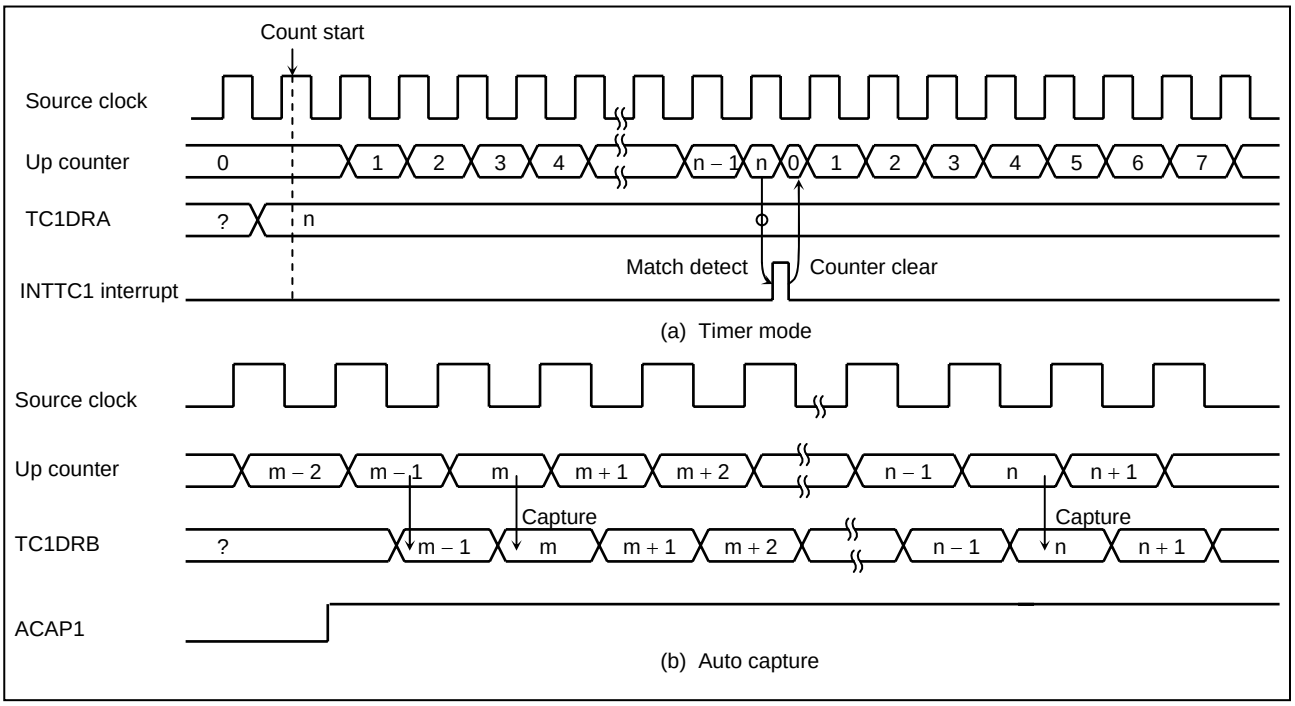



Figure 2.5.3 Timer Mode Timing Chart

(2) External trigger timer mode

In this mode, counting up is started by an external trigger. This trigger is the edge of the TC1 pin input. Either the rising or falling edge can be selected with TC1S. Source clock is an internal clock. The contents of TC1DRA is compared with the contents of 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.

When METT1 (Bit6 in TC1CR) is "1", inputting the edge to the reverse direction of the trigger edge to start counting clears the counter, and the counter is stopped. Inputting a constant pulse width can generate interrupts. When METT1 is "0", the reverse directive edge input is ignored. The TC1 pin input edge before a match detection is also ignored.

The TC1 pin input has the noise rejection; therefore, pulses of 7/fc [s] or less are rejected as noise. A pulse width of 13/fc [s] or more is required for edge detection in NORMAL or IDLE mode.

Example 1: Detects rising edge in TC1 pin input and generates an interrupt 100 μs later. (at fc = 16.0 MHz, DV1CK = 1)

```
LDW   (TC1DRA), 0064H        ;   100 μs ÷ 2^4/fc = 64H
DI
SET   (EIRL). 4              ;   INTTC1 interrupt enable
EI
LD    (TC1CR), 00001000B     ;   Selects the source clock and mode
LD    (TC1CR), 00101000B     ;   TC1 external trigger start, METT1 = 0
```

Example 2: Generates an interrupt, inputting "L" level pulse (Pulse width: 4 ms or more) to the TC1 pin. (at fc = 16.0 MHz, DV1CK = 1)

```
LDW   (TC1DRA), 00FAH        ;   4 ms ÷ 2^8/fc = FAH
DI
SET   (EIRL). 4              ;   INTTC1 interrupt enable
EI
LD    (TC1CR), 00000100B     ;   Selects the source clock and mode
LD    (TC1CR), 01110100B     ;   TC1 external trigger start, METT1 = 1
```

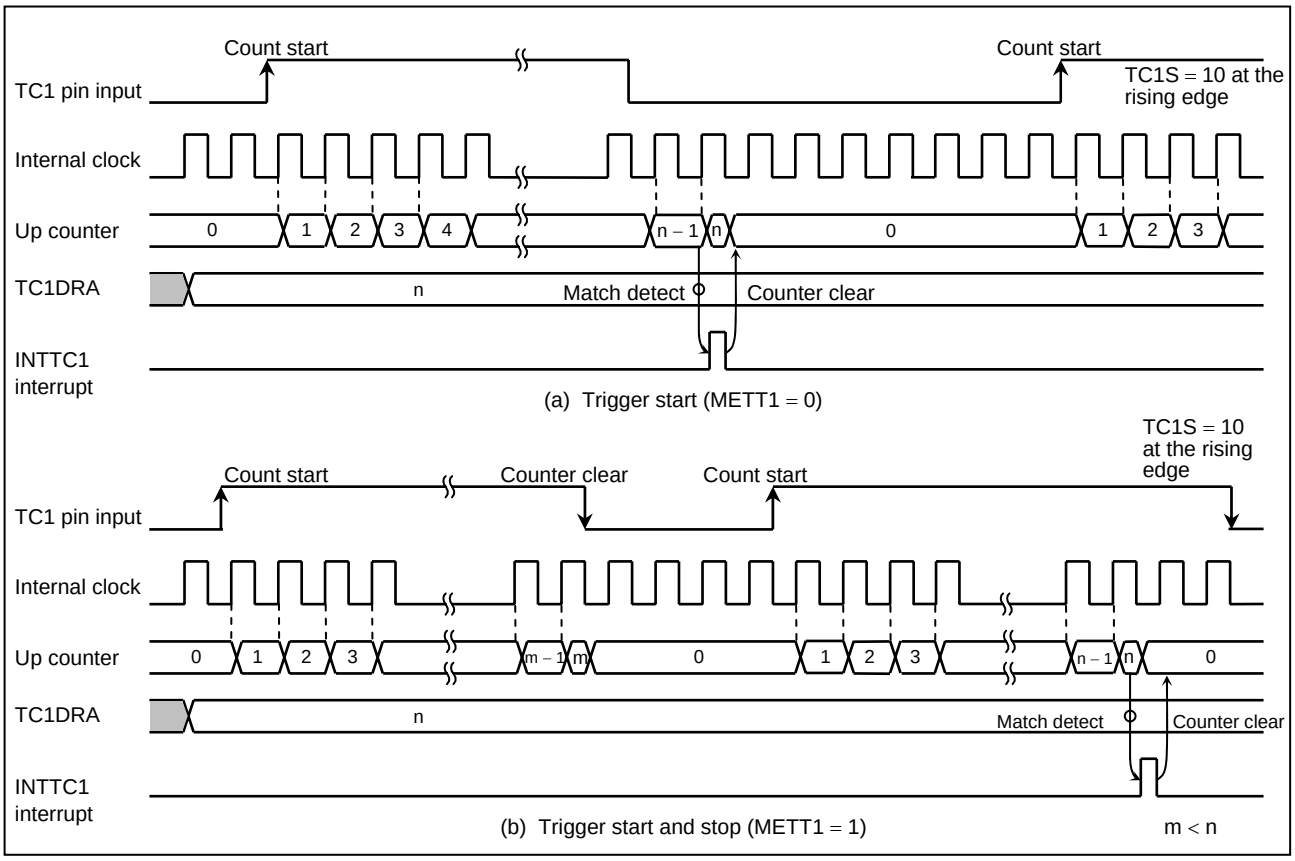



Figure 2.5.4 External Trigger Timer Mode Timing Chart

(3) Event counter mode

In this mode, events are counted at the edge of the TC1 pin input (Either the rising or falling edge can be selected with the external trigger TC1CR<TC1S>). The contents of TC1DRA are compared with the contents of up counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared.

Match detect is executed on other edge of count-up. A match can not be detected and INTTC1 is not generated when the pulse is still in same state.

Setting ACAP1 to "1" transfers the current contents of up counter to TC1DRB (Auto-capture function).

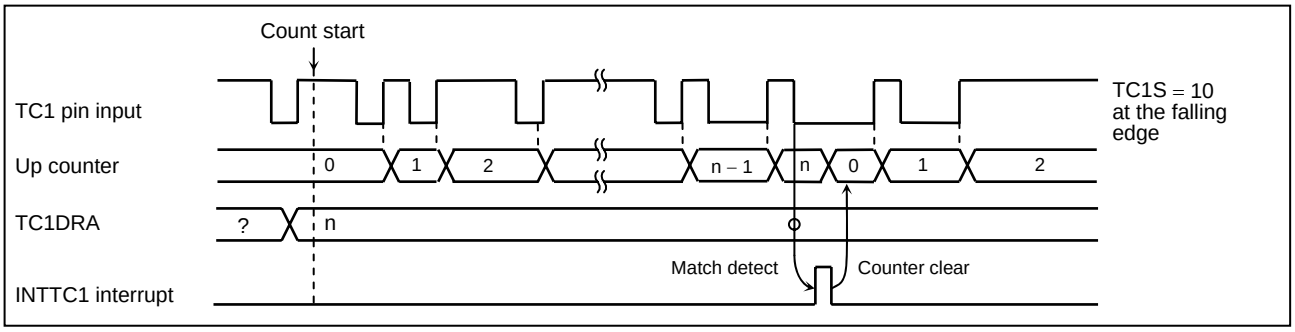



Figure 2.5.5 Event Counter Mode Timing Chart

Table 2.5.2 Input Pulse Width for Timer/Counter 1

| | Minimum Pulse Width [s] |
|---|---|
| | NORMAL/IDLE |
| "H" Width | $2^3$/fc |
| "L" Width | $2^3$/fc |

(4) Window mode

Counting up is performed on 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 TC1DRA are compared with the contents of 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 bit4 or 5 in TC1CR.

It is necessary that the maximum applied frequency 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.

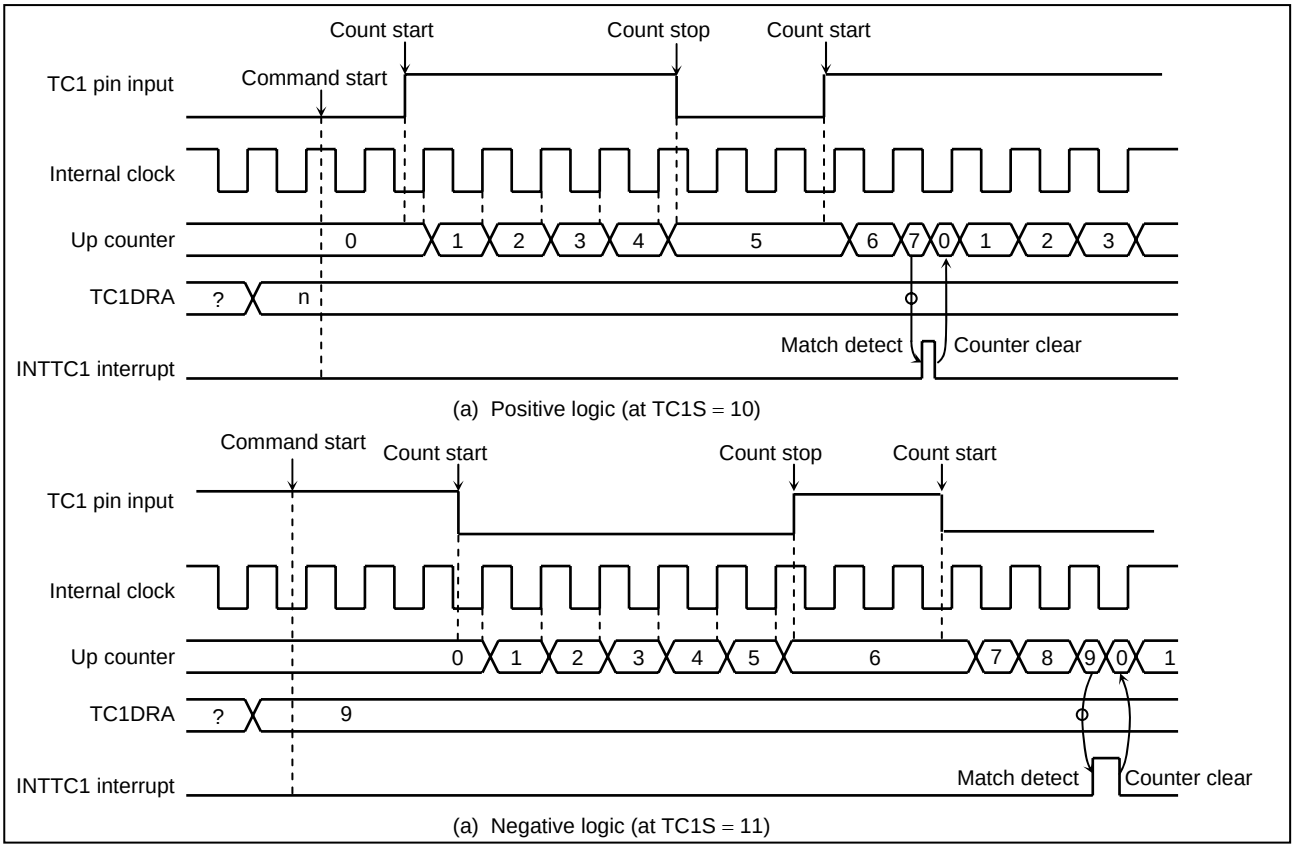



Figure 2.5.6 Window Mode Timing Chart

(5) Pulse width measurement mode

In this mode, counting is started by the external trigger (Set to external trigger start by TC1CR). The trigger can be selected either the rising or falling edge of the TC1 pin input. The source clock is used an internal clock. On the next falling (rising) edge, the counter contents are transferred to TC1DRB and an INTTC1 interrupt is generated. The counter is cleared when the 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 TC1DRB. If a falling (rising) edge capture value is required, it is necessary to read out TC1DRB contents until a rising (falling) edge is detected. Falling or rising edge is selected with the external trigger TC1S (Bit4 or 5 in TC1CR), and single edge or double edge is selected with MCAP1 (Bit6 in TC1CR).

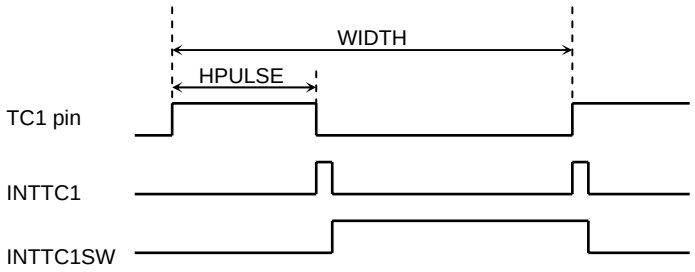
Note 1: Be sure to read the captured value from TC1DRB before the next trigger edge is detected. If fail to read it, it becomes undefined. It is recommended that a 16-bit access instruction be used to read from TC1DRB.

Note 2: If either the falling or rising edge is used in capturing values, the counter stops at “1” after a value has been captured until the next edge is detected. So, the value captured next will become “1” larger than the value captured right after capturing starts.

Note 3: In the Pulse width measurement mode, the capture value of the first time after the timer starts might not be a correct value. Thus, execute the dummy read once.

Example: Duty measurement (Resolution $fc/2^7$ [Hz] DV1CK = 0)

```
            CLR     (INTTC1SW). 0          ;  INTTC1 service switch initial setting:
                                              Clears bit0 of INTTC1SW. This bit is inverted by
                                              CPL instruction before INTTC1 is generated.
            LD      (TC1CR), 00000110B     ;  Sets the TC1 mode and source clock
            DI
            SET     (EIRL). 4              ;  Enables INTTC1
            EI
            LD      (TC1CR), 00100110B     ;  Starts TC1 with an external trigger at MCAP1 = 0
            :
PINTTC1:    CPL     (INTTC1SW). 0          ;  Complements INTTC1 service switch
            JRS     F, SINTTC1
            LD      WA, (TC1DRBL)          ;  Reads TC1DRB (“H” level pulse width)
                                              Lower address in TC1DRBL: TC1DRB
            LD      (HPULSE), WA
            RETI
SINTTC1:    LD      WA, (TC1DRBL)          ;  Reads TC1DRB (Period)
            LD      (WIDTH), WA
            :
            RETI                           ;  Duty calculation
            :
VINTTC1:    DW      PINTTC1                ;  Sets INTTC1
```



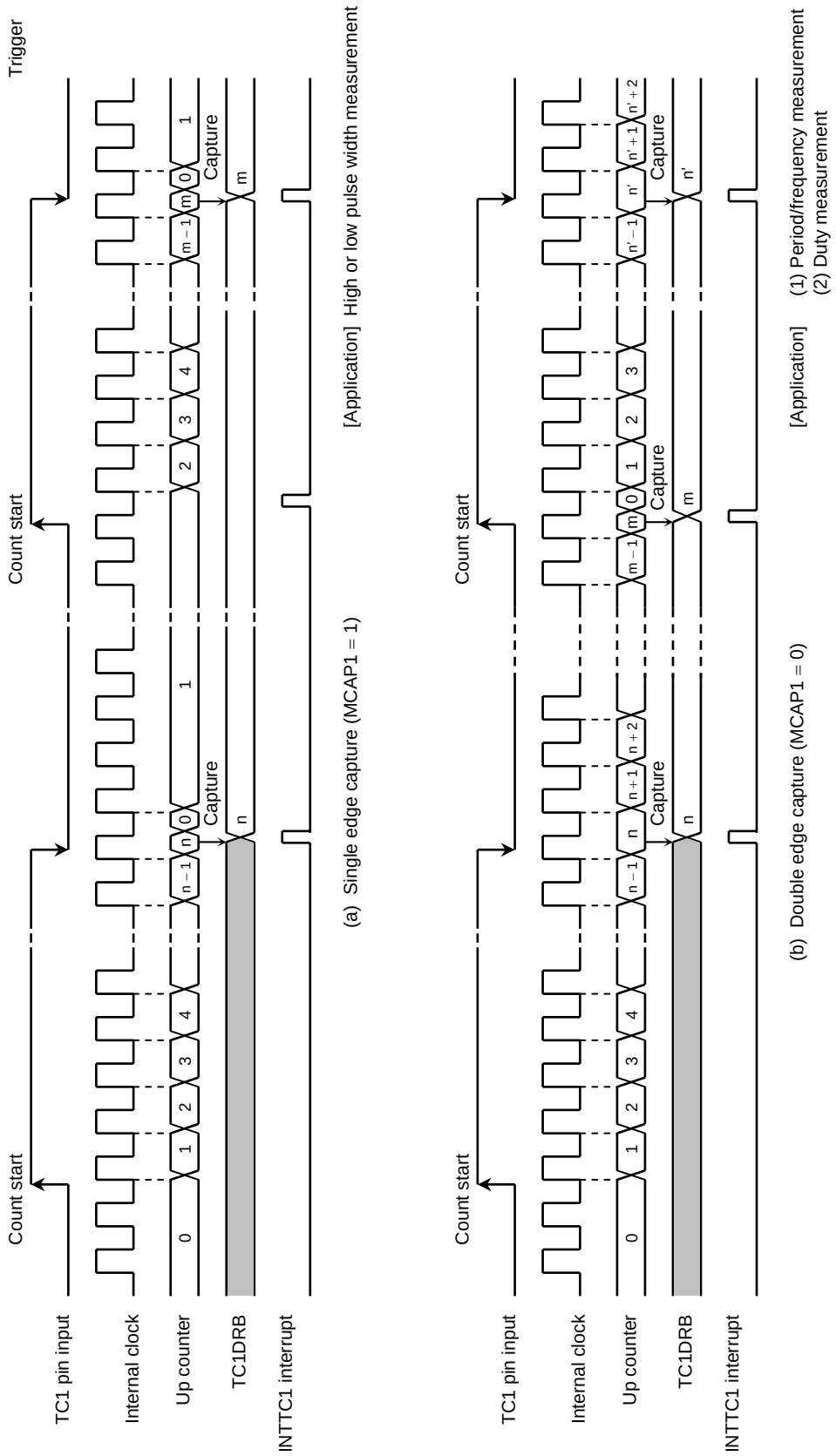
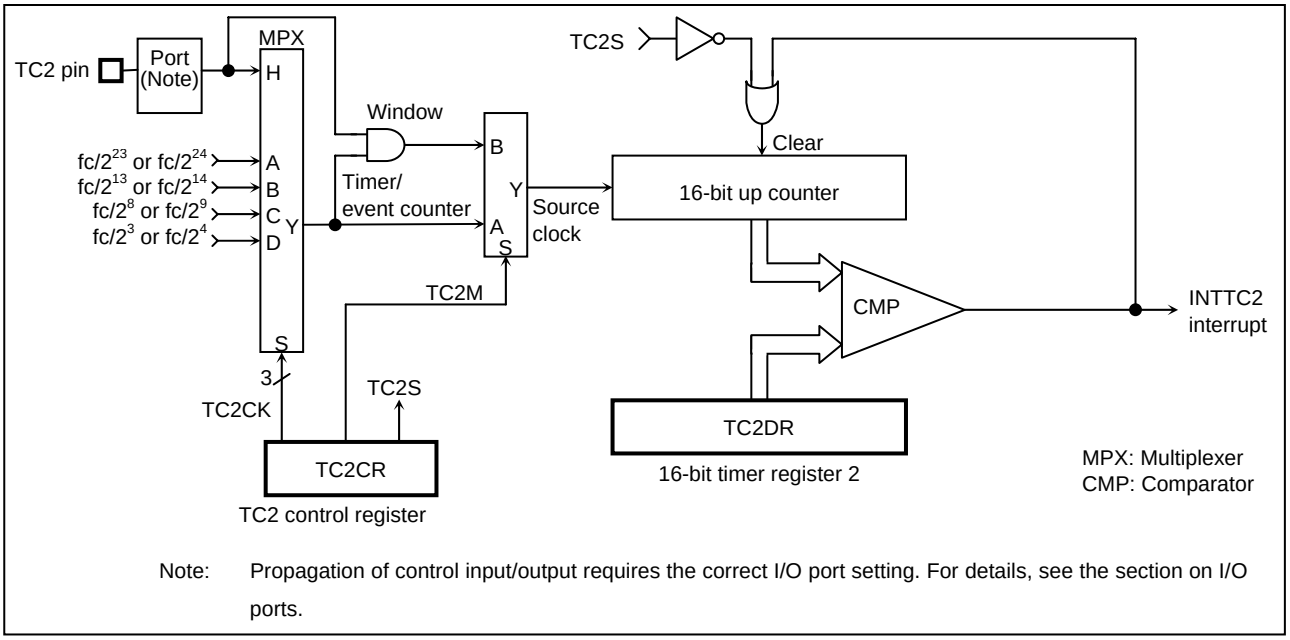



Figure 2.5.7 Pulse Width Measurement Mode Timing Chart

## 2.6 16-Bit Timer/Counter 2 (TC2A)

### 2.6.1 Configuration




Note: Propagation of control input/output requires the correct I/O port setting. For details, see the section on I/O ports.

Figure 2.6.1 Timer/Counter 2 (TC2)

### 2.6.2 Control

The timer/counter 2 is controlled by a timer/counter 2 control register (TC2CR) and a 16-bit timer register 2 (TC2DR). Reset does not affect TC2DR.

| TC2DR (00016H, 00017H) | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TC2DRH (00017H) | | | | | | | | TC2DRL (00016H) | | | | | | | |
| | | | | | | | | | Read/Write | | | | | | | |

| TC2CR (00015H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TC2S | TC2CK | | | | TC2M |

(Initial value: **00 00*0)

| | | | | | |
|---|---|---|---|---|---|
| TC2M | TC2 operating mode select | 0: Timer/event counter mode<br>1: Window mode | | | Write only |
| TC2CK | TC2 source clock select [Hz] | | NORMAL1/2, IDLE1/2 mode | | |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 000 | $fc/2^{23}$ | $fc/2^{24}$ | |
| | | 001 | $fc/2^{13}$ | $fc/2^{14}$ | |
| | | 010 | $fc/2^{8}$ | $fc/2^{9}$ | |
| | | 011 | $fc/2^{3}$ | $fc/2^{4}$ | |
| | | 100 | Reserved | Reserved | |
| | | 101 | Reserved | Reserved | |
| | | 110 | Reserved | | |
| | | 111 | External clock (TC2 pin input) | | |
| TC2S | TC2 start control | 0: Stop and counter clear<br>1: Start | | | |

Note 1: fc: High-frequency clock [Hz], *: Don't care.

Note 2: Writing to the lower byte of timer register 2 (TC2DRL), the comparison is inhibited until the upper byte (TC2DRH) is written. After writing to the upper byte, any match during 1 machine cycle (Instruction execution cycle) is ignored.

Note 3: Set the mode and source clock when the TC2 stops (TC2S = 0).

Note 4: Values to be loaded to timer registers must satisfy the following condition.

TC2DR > 1

Note 5: TC2CR are write-only registers and must not be used with any of the read-modify-write instructions.

Note 6: When STOP mode is started, timer counter is stopped and cleared. Set TC2S to "1" after STOP mode is released for restarting timer counter.

Figure 2.6.2 Timer Registers 2 and TC2 Control Register

### 2.6.3 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 TC2DR are compared with the contents of 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 2.6.1 Source Clock (Internal clock) for Timer/Counter 2 (at fc = 16.0 MHz)

| TC2CK | NORMAL, IDLE Mode | | | |
|---|---|---|---|---|
| | DV1CK = 0 | | DV1CK = 1 | |
| | Resolution | Maximum Time Setting | Resolution | Maximum Time Setting |
| 000 | 524.3 [ms] | 9.54 [h] | 1.05 [s] | 19.1 [h] |
| 001 | 512.0 [μs] | 33.6 [s] | 1.02 [ms] | 1.12 [min] |
| 010 | 16.0 [μs] | 1.05 [s] | 32.0 [μs] | 2.09 [s] |
| 011 | 0.5 [μs] | 32.8 [ms] | 1.0 [μs] | 65.5 [ms] |
| 100 | Reserved | Reserved | Reserved | Reserved |
| 101 | Reserved | Reserved | Reserved | Reserved |

Example: Sets the source clock $fc/2^4$ [Hz] and generates an interrupt event 25 ms (at fc = 16 MHz, DV1CK = 1)

```
LDW   (TC2DR), 61A8H        ;  Sets TC2DR (25 ms ÷ 2^4/fc = 61A8H)
DI
SET   (EIRH). 6             ;  Enable INTTC2 interrupt
EI
LD    (TC2CR), 00001100B    ;  Selects TC2 source clock
LD    (TC2CR), 00101100B    ;  Starts TC2
```

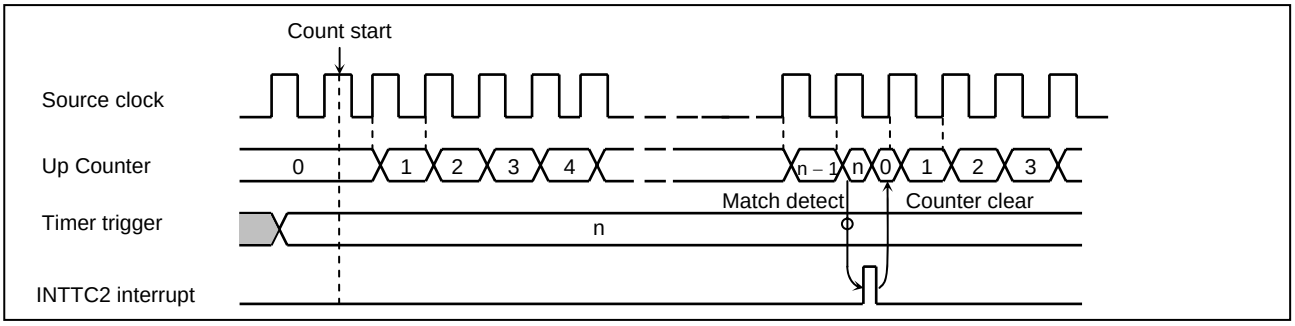



Figure 2.6.3 Timer Mode Timing Chart

(2) Event counter mode

In this mode, events are counted on the rising edge of the TC2 pin input. The contents of TC2DR 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 minimum pulse width to the TC2 pin is shown in Table 2.6.2. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width. Match detect is executed on the falling edge of the TC2 pin. A match can not be detected and INTTC2 is not generated when the pulse is still in a falling state.

Example: Sets the event counter mode and generates an INTTC2 interrupt 640 counts later.

```
LDW   (TC2DR), 640          ;  Sets TC2DR
DI
SET   (EIRH). 6             ;  Enables INTTC2 interrupt
EI
LD    (TC2CR), 00011100B    ;  Selects TC2 source clock
LD    (TC2CR), 00111100B    ;  Starts TC2
```

Table 2.6.2 Timer/Counter 2 External Clock Source

| | Minimum Pulse Width [S] |
|---|---|
| | NORMAL, IDLE Mode |
| "H" Width | $2^3$/fc |
| "L" Width | $2^3$/fc |

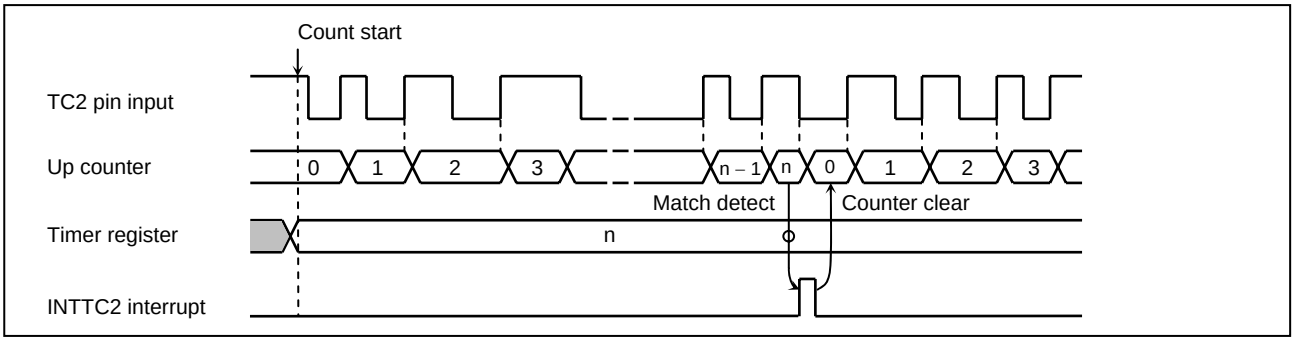



Figure 2.6.4 Event Counter Mode Timing Chart

(3) Window mode

In this mode, counting up performed on the rising edge of an internal clock during TC2 external pin input (window pulse) is "H" level. The contents of TC2DR are compared with the contents of up counter. If a match found, an INTTC2 interrupt is generated, and the up counter is cleared.

The maximum applied frequency (TC2 input) must be considerably slower than the selected internal clock.

Example: Generates an interrupt, inputting "H" level pulse width of 120 ms or more. (at fc = 16.0 MHz, DV1CK = 1)

```
LDW   (TC2DR), 0075H          ;  Sets TC2DR (120 ms ÷ 2^14/fc = 0075H)
DI
SET   (EIRH). 6               ;  Enables INTTC2 interrupt
EI
LD    (TC2CR), 00000101B      ;  Selects TC2 source clock
LD    (TC2CR), 00100101B      ;  Starts TC2
```

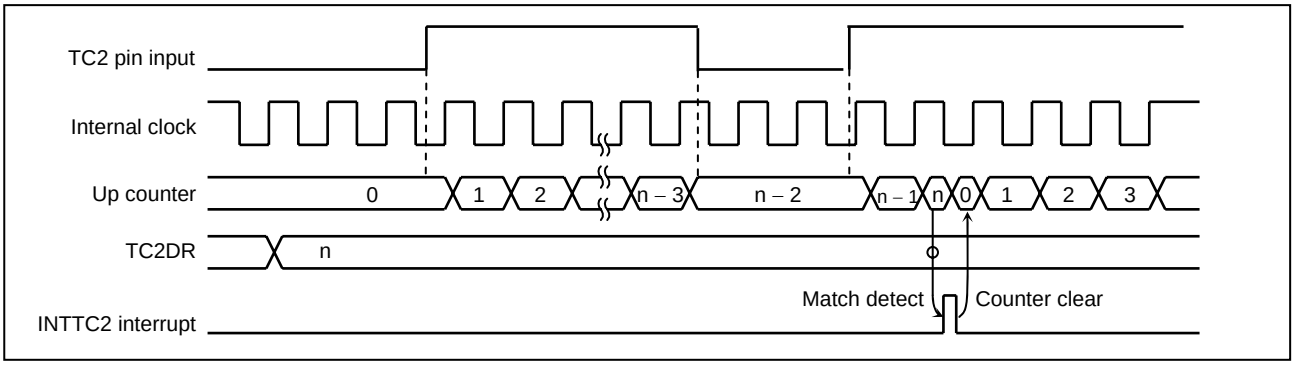
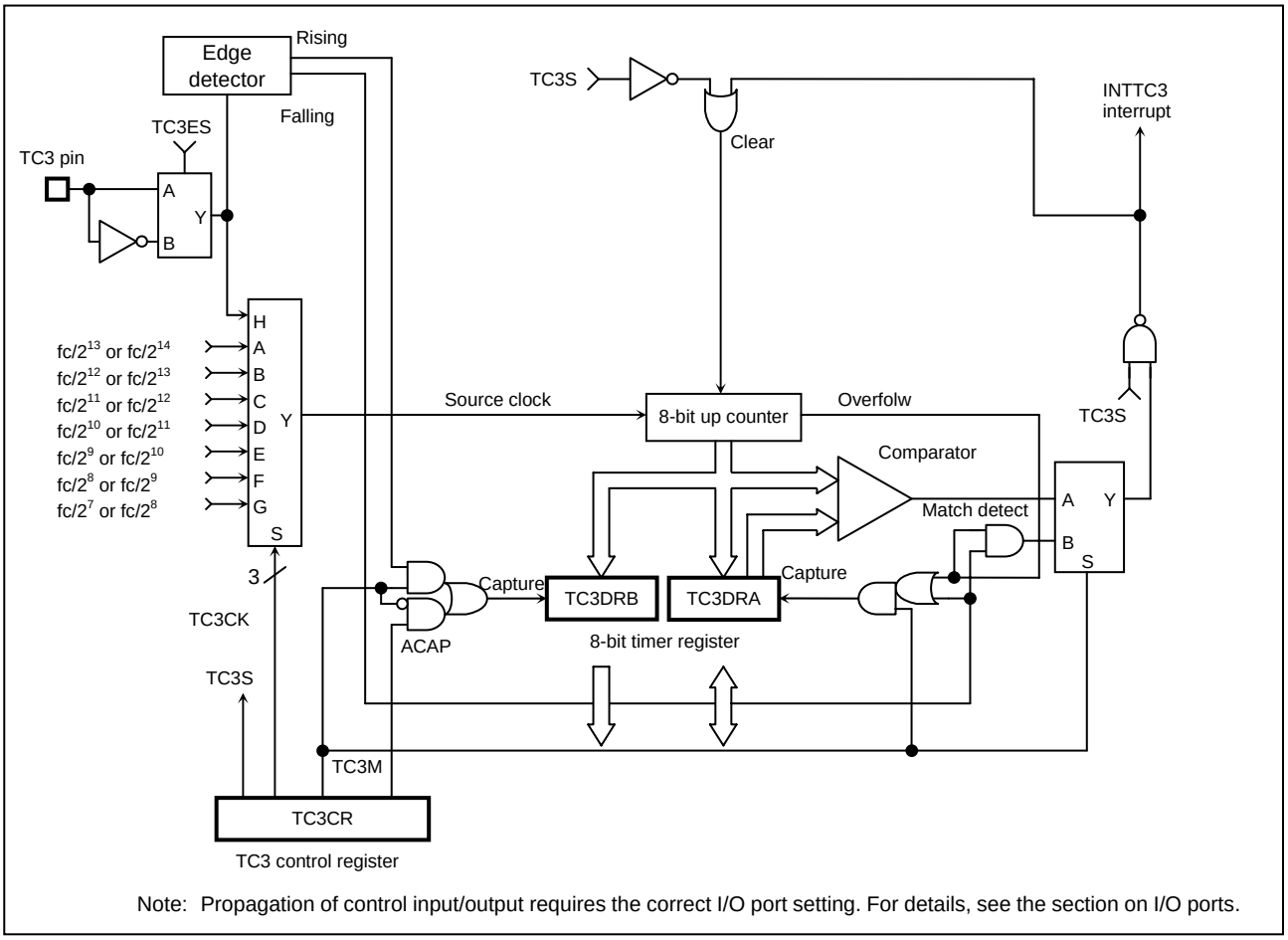



Figure 2.6.5 Window Mode Timing Chart

## 2.7 8-Bit Timer/Counter 3 (TC3B)

### 2.7.1 Configuration




Note: Propagation of control input/output requires the correct I/O port setting. For details, see the section on I/O ports.

Figure 2.7.1 Timer/Counter 3 (TC3)

### 2.7.2 Control

The timer/counter 3 is controlled by a timer/counter 3 control register (TC3CR) and two 8-bit timer registers (TC3DRA and TC3DRB) and port multiplex control register (PMPXCR).

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TC3DRA (0018H) | | | | | | | | | Read/Write (Initial value: 1111 1111) |
| TC3DRB (0019H) | | | | | | | | | Read only (Initial value: 1111 1111) |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TC3CR (001AH) | | ACAP | | TC3S | TC3K | | | TC3M | (Initial value: *0*0 0000) |

| | | | | | |
|---|---|---|---|---|---|
| TC3M | TC3 operating mode select | 0: Timer/event counter<br>1: Capture | | | Write only |
| TC3CK | TC3 source clock select [Hz] | | NORMAL, IDLE mode | | |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 000 | $fc/2^{13}$ | $fc/2^{14}$ | |
| | | 001 | $fc/2^{12}$ | $fc/2^{13}$ | |
| | | 010 | $fc/2^{11}$ | $fc/2^{12}$ | |
| | | 011 | $fc/2^{10}$ | $fc/2^{11}$ | |
| | | 100 | $fc/2^{9}$ | $fc/2^{10}$ | |
| | | 101 | $fc/2^{8}$ | $fc/2^{9}$ | |
| | | 110 | $fc/2^{7}$ | $fc/2^{8}$ | |
| | | 111 | External clock (TC3 pin input) | | |
| TC3S | TC3 start control | 0: Stop and clear<br>1: Start | | | |
| ACAP | Auto-capture control | 0: –<br>1: Auto-capture enable | | | |

Note 1: fc: High-frequency clock [Hz], *: Don't care.

Note 2: Set the mode and source clock when the TC3 stops (TC3S = 0).

Note 3: Values to be loaded to timer register 3A must satisfy the following condition TC3DRA > 0 (in the timer and event counter mode).

Note 4: Auto-capture can be used only int the timer and event counter mode.

Note 5: Before setting TC3DRA or switching the operation mode, stop the TC3 (TC3S = 0).

Note 6: When STOP mode is started, timer counter is stopped and TC3 start control (TC3S) is cleared to "0" automatically. Set TC3S to "1" after STOP mode is released for restarting timer counter.

Note 7: TC3CR, TCESCR is a write-only register and must not be used with any of the read-modify-write instructions.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| PMPXCR (0027H) | "0" | CHS | | | | | TC4ES | TC3ES | (Initial value: 00** **00) |

| | | | |
|---|---|---|---|
| TC3ES | TC3 input control | 0: Normal<br>1: Invert | Write only |

Note 8 Always write "0" to bit7 in PMPXCR.

Figure 2.7.2 Timer Registers 3 and TC3 Control Register

### 2.7.3 Function

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

When it is used in the capture mode, the noise rejection time of TC3 pin input can be set by remote control receive control register.

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of TC3DRA are compared with the contents of up counter. If a match is found, a timer/counter 3 interrupt (INTTC3) is generated, and the up counter is cleared. The current contents of up counter are loaded into TC3DRB by setting ACAP (Bit6 in TC3CR) to "1" (Auto-capture function).

The contents of up counter can be easily confirmed by executing the read instruction (RD instruction) of TC3DRB. Loading the contents of up counter is not synchronized with counting up. The contents of over flow (FFH) and 00H can not be loaded correctly. It is necessary to consider the count cycle.

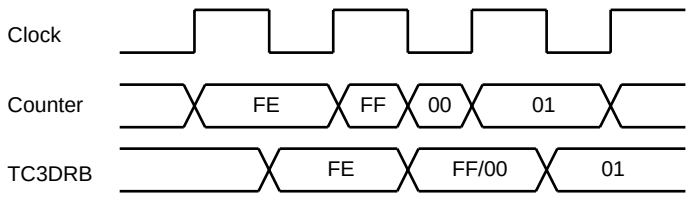



Table 2.7.1 Source Clock (Internal clock) for Timer/Counter 3 (Example: at fc = 16.0 MHz)

| TC3CK | NORMAL, IDLE Mode | | | |
|---|---|---|---|---|
| | DV1CK = 0 | | DV1CK = 1 | |
| | Resolution [μs] | Maximum Setting Time [ms] | Resolution [μs] | Maximum Setting Time [ms] |
| 000 | 512 | 130.6 | 1024 | 261.1 |
| 001 | 256 | 65.3 | 512 | 130.6 |
| 010 | 128 | 32.6 | 256 | 65.3 |
| 011 | 64 | 16.3 | 128 | 32.6 |
| 100 | 32 | 8.2 | 64 | 16.3 |
| 101 | 16 | 4.1 | 32 | 8.2 |
| 110 | 8 | 2.0 | 16 | 4.1 |

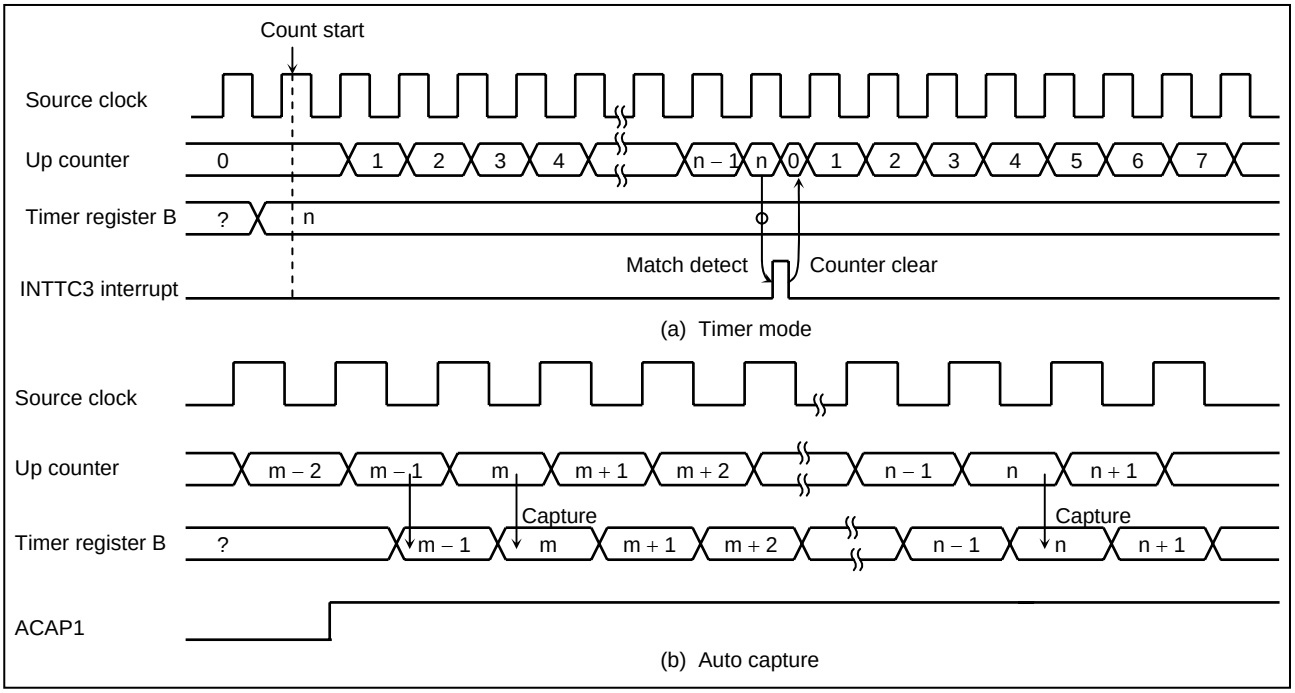



Figure 2.7.3 Timer Mode Timing Chart

(2) Event counter mode

In this mode, the TC3 pin input pulses are used for counting up Either the rising on falling edge can be selected with TC3ES (Bit0 in PMPXCR). The contents of TC3DRA are compared with the contents of the up counter. If a match is found, an INTTC3 interrupt is generated and the counter is cleared. Match detect is executed on the falling edge of the TC3 pin. A match can not be detected, and INTTC3 is not generated when the pulse is still in a falling state.

The maximum applied frequency is shown in Table 2.7.2. 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 TC3DRB by setting ACAP (Bit6 in TC3CR) to "1" (Auto-capture function).

The contents of up counter can be easily confirmed by executing the read instruction (RD instruction) of TC3DRB. Loading the contents of up counter is not synchronized with counting up. The contents of over flow (FFH) and 00H can not be loaded correctly. It is necessary to consider the count cycle.

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

```
LD    (TC3CR), 00001110B    ;  Sets TC3 mode and source clock
LD    (TC3DRA), 19H         ;  0.5 s ÷ 1/50 = 25 = 19H
LD    (TC3CR), 00011100B    ;  Starts TC3
```

Table 2.7.2 Source Clock (External clock) for Timer/Counter

| | Minimum Applied Frequency [Hz] |
|---|---|
| | NORMAL, IDLE Mode |
| "H" Width | $2^2$/fc |
| "L" Width | $2^2$/fc |

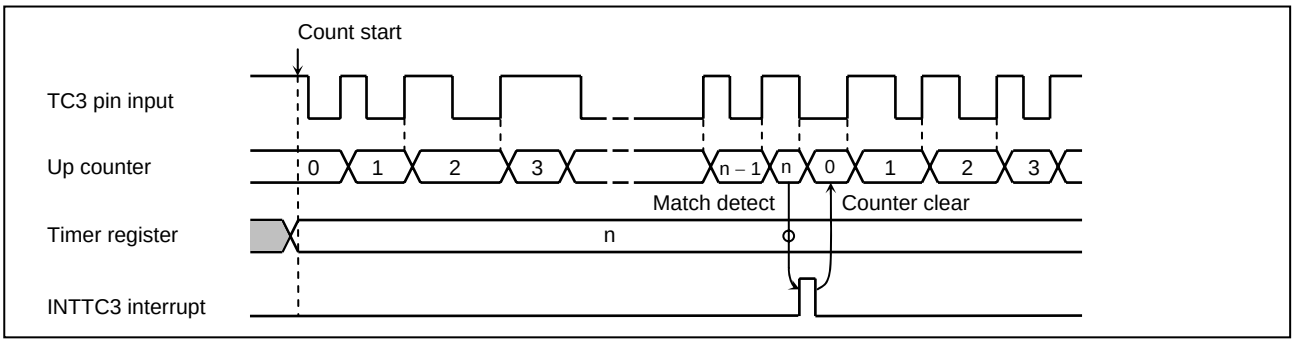



Figure 2.7.4 Event Counter Mode Timing Chart

(3) Capture mode

In this 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 signals or distinguishing AC 50/60 Hz, etc. The TC3 pin input can have its polarity changed between normal and inverse by using the TC3ES Register.

a. If TC3ES = "0" (Non-inverting input)

Once command operation has started, the counter free-runs on an internal source clock.

When the falling edge of the TC3 pin input is detected, the counter value is loaded into TC3DRB. When the rising edge is detected, the counter value is loaded into TC3DRA, and the counter is cleared, generating an INTTC3 interrupt.

If the rising edge is detected right after command operation has started, no capture to TC3DRB and an INTTC3 interrupt occurs only on capture to TC3DRA. If a read instruction is executed for TC3DRB, the value that exists at the end of the previous capture (immediately after a reset, "FF") is read.

b. If TC3ES = "1" (Inverse input)

Once command operation has started, the counter free-runs on an internal clock.

When the rising edge of the TC3 pin input is detected, the counter value is loaded into TC3DRB. When the falling edge is detected, the counter value is loaded into TC3DRA, and the counter is cleared, generating an INTTC3 interrupt.

If the falling edge is detected right after command operation has started, the counter value is not captured into TC3DRB and an INTTC3 interrupt occurs only on capture to TC3DRA. If a read instruction is executed for TC3DRB, the value that exists at end of the previous capture (immediately after a reset, "FF") is read.

The minimum acceptable input pulse width is equal to the length of one source clock period selected by TC3CR <TC3CK>.

Table 2.7.3 TC3INV-based Capture Input Edges

| TC3ES | Capture into TC3DRB | Capture into TC3DRA | INTTC3 Interrupt |
|---|---|---|---|
| "0" (Non-inverting input) | Falling edge | Rising edge | |
| "1" (Inverting input) | Rising edge | Falling edge | |

Note: Capture of the TC3 pin input requires at least 1 cycle of the selected source clock.

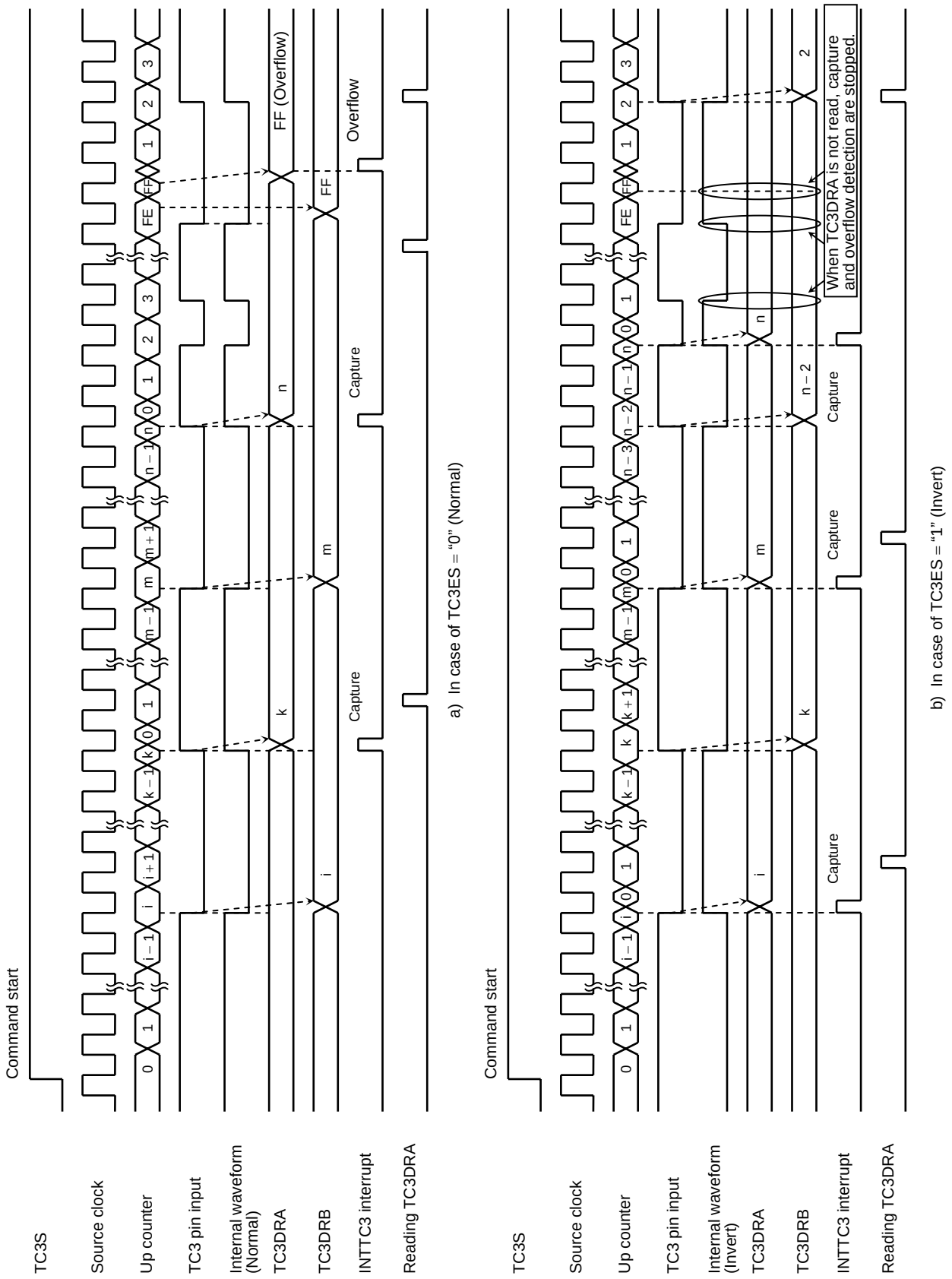



Figure 2.7.5 Capture Mode Timing Chart

The edge of TC3 pin input is detected in the remote control receive circuit with noise rejection. The remote control receive circuit is controlled by the remote control receive control register (RCCR). The remote control receive status register (RCSR) can monitor the polarity selection and noise rejection circuit.

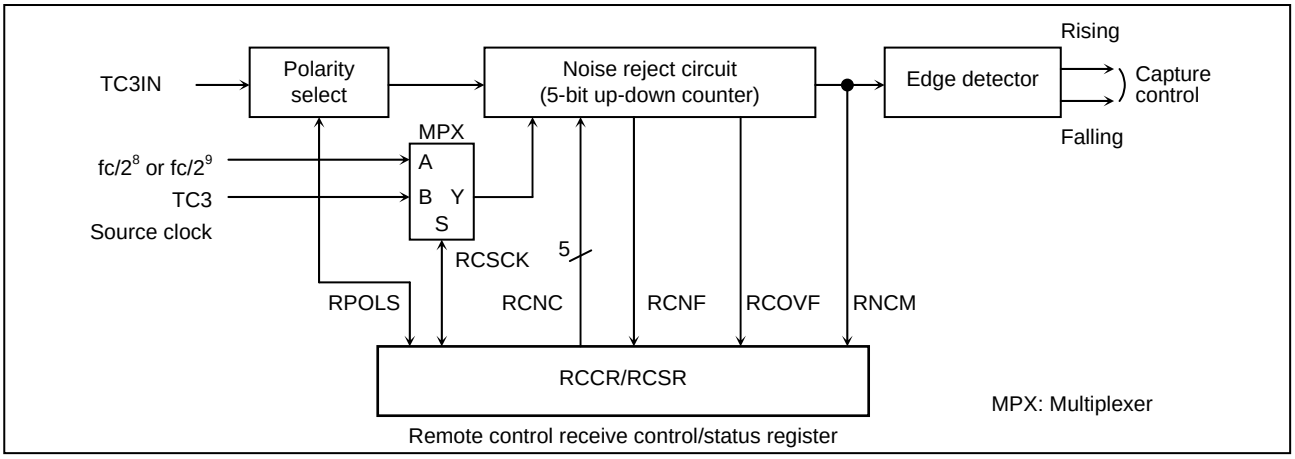



Figure 2.7.6 Remote Control Receiving Circuit

RCCR
(00026H)

| RCEN | RPOLS | RCSCK | RCNC |
|---|---|---|---|

(Initial value: 0001 1111)

| | | | | | |
|---|---|---|---|---|---|
| RCNC | Noise reject time select<br>02H ≤ RCNC ≤ 1FH | (Source clock) × (RCNC − 1) [s] | | | Write only |
| RCSCK | Noise reject cricuit<br>Source clock select | | NORMAL, IDLE mode | | R/W |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 0 | $2^8$/fc | $2^9$/fc | |
| | | 1 | TC3CK Note 2 | | |
| RPOLS | Remote control signal polarity select | 0: Positive<br>1: Negative | | | Write only |
| RCEN | Remote control receive circuit operation control | 0: Disable<br>1: Enable | | | |

Note 1: Set RPOLS and RCSCK when the timer/counter stops (TC3S = 0).

Note 2: Source clock of timer/counter 3.

Note 3: fc: High-frequency clock [Hz], *: Don't care.

Note 4: RCCR includes a write-only register and must not be used with any of read-modify-write instructions.

Note5: Values to be loaded to RCNC must satisfy the following condition (02 ≤ RCNC ≤ 1F).

RCSR
(00026H)

| RCNF | RPOLS | RCSCK | RCOVF | RNCM | - - - |
|---|---|---|---|---|---|

(Initial value: 0000 0***)

| | | | | | |
|---|---|---|---|---|---|
| RNCM | Remote control signal monitor after noise rejector | 0: Low level<br>1: High level | | | Read only |
| RCOVF | Noise reject circuit overflow flag | 0: Signal and definition by overwriting the noise reject time RCNC<br>1: Overflow | | | |
| RCSCK | Noise reject circuit<br>Source clock Select | | NORMAL, IDLE mode | | R/W |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 0 | $2^8$/fc | $2^9$/fc | |
| | | 1 | TC3CK (Note 2) | | |
| RPOLS | Remote control signal polarity select | 0: Positive<br>1: Negative | | | |
| RCNF | Remote control signal monitor after noise rejctor | 0: Without noise<br>1: With noise | | | Read only |

Note 1: Reading out the register RCSR resets RCNF and RCOVF.

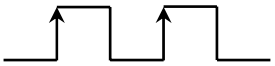
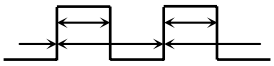
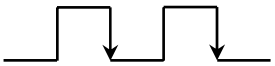
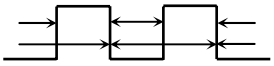
Note 2: Source clock of timer/counter 3

Note 3: When a 5-bit up-down counter counts down to "0" after counting up, the RCNF defines to be noise.

Note 4: fc: High-frequency clock [Hz], *: Don't care.

Figure 2.7.7 Remote Control Rceive Control Register and Remote Control Receive Status Register

Table 2.7.4 Combination between The Polarity and The Edge Selection

| RPOLS | TC3 Pin Input Pulse<br>(Interrupt occurrence is shown as allow.) | Measurement |
|---|---|---|
| 0 |  |  |
| 1 |  |  |

Note: When TC3CK is used in RCSCK, do not select an external clock to the TC3CK.

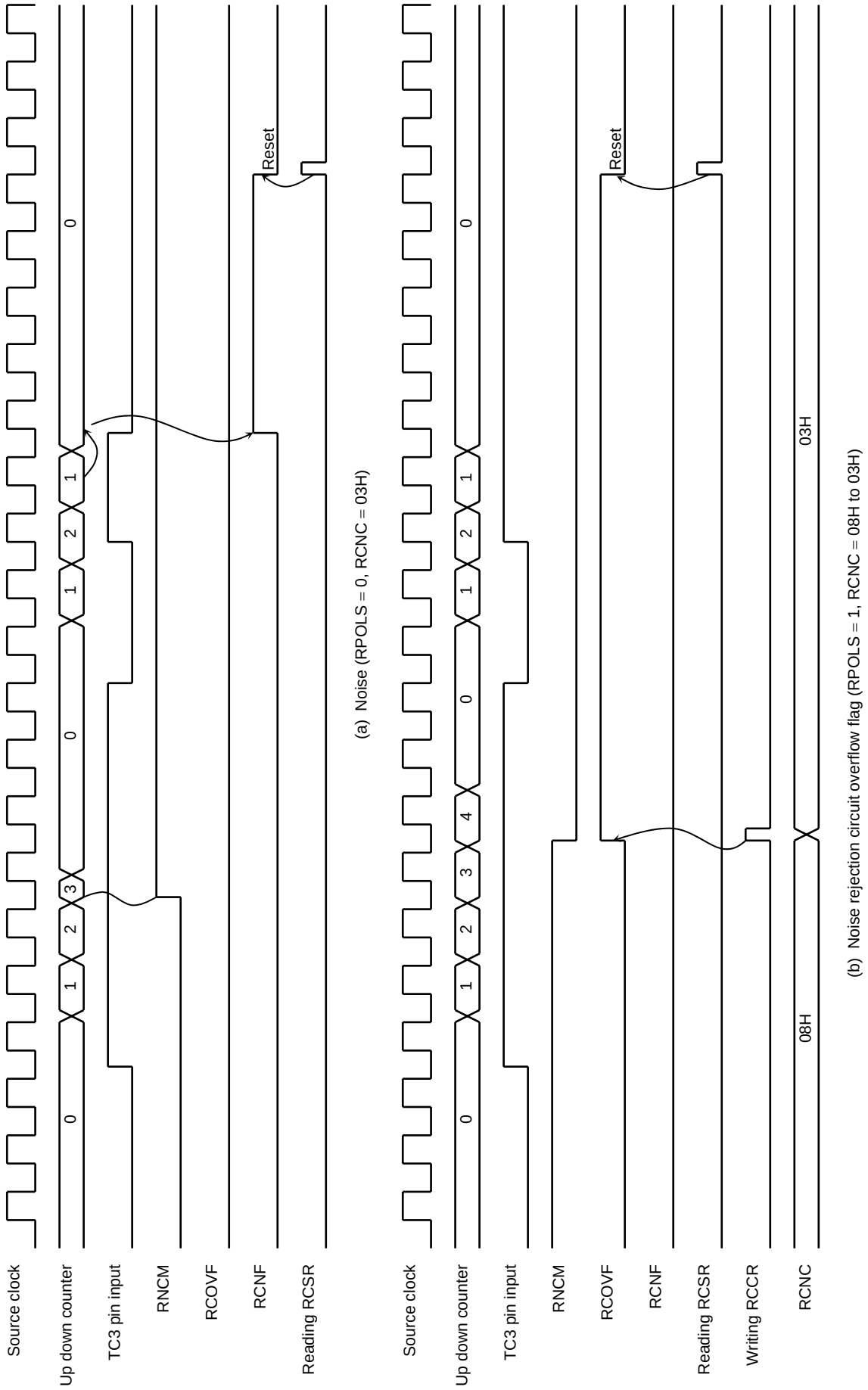
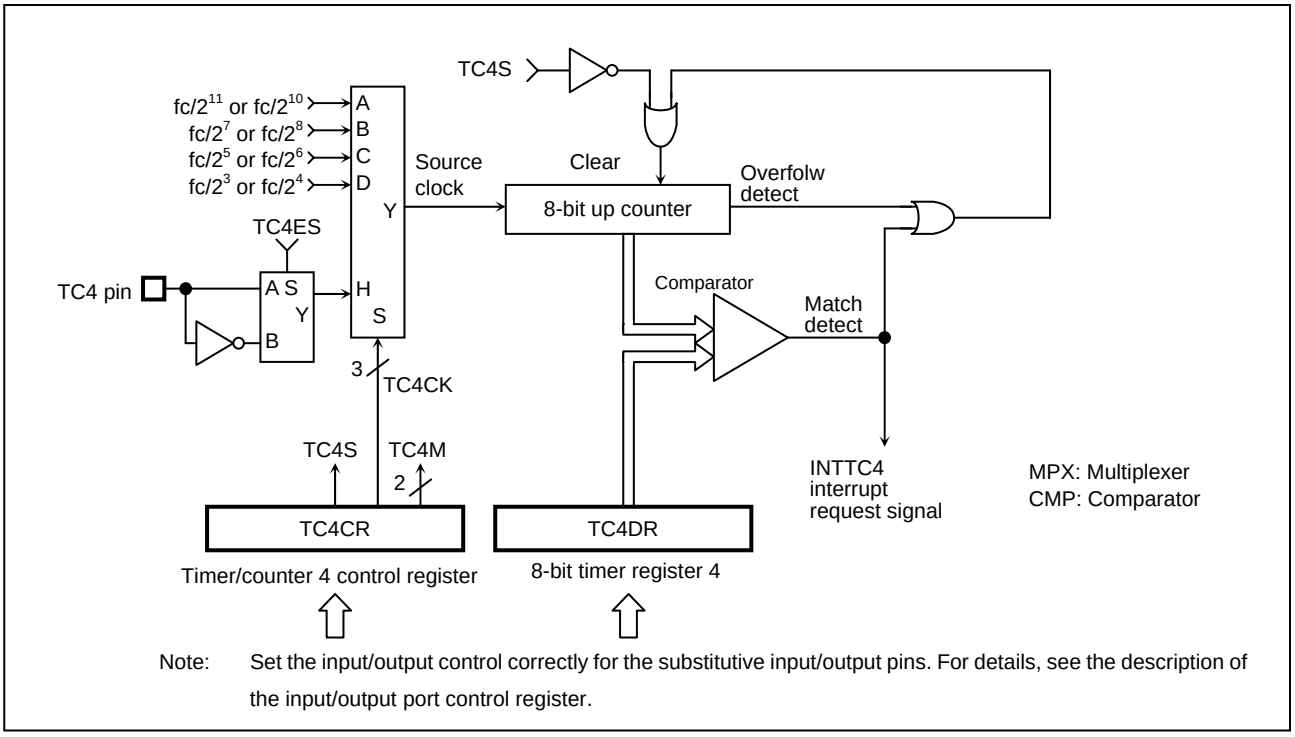



Figure 2.7.8 Remote Control Receive Circuit Timing Chart

## 2.8 8-Bit Timer/Counter 4 (TC4)

### 2.8.1 Configuration




Note: Set the input/output control correctly for the substitutive input/output pins. For details, see the description of the input/output port control register.

Figure 2.8.1 Timer/Counter 4 (TC4)

### 2.8.2 Control

The timer/counter 4 is controlled by a timer/counter 4 control register (TC4CR) and an 8-bit timer register 4 (TC4DR). Reset does not affect TC4DR.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TC4DR (0001BH) | | | | | | | | | Write only (Initial value: 1111 1111) |
| TC4CR (0001CH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| | : | : | TC4S | TC4CK | | | TC4M | | Write only (Initial value: **00 0000) |

| | | | | | |
|---|---|---|---|---|---|
| TC4S | TC4 start control | 00: Timer/event counter mode<br>01: Reserved<br>10: Reserved<br>11: Reserved | | | R/W |
| TC4CK | TC4 source clock select [Hz] (Note 4) | | NORMAL, IDLE mode | | |
| | | | DV1CK = 0 | DV1CK = 1 | |
| | | 000 | $fc/2^{11}$ | $fs/2^{12}$ | |
| | | 001 | $fc/2^{7}$ | $fs/2^{8}$ | |
| | | 010 | $fc/2^{5}$ | $fs/2^{6}$ | |
| | | 011 | $fc/2^{3}$ | $fs/2^{4}$ | |
| | | 100 | Reserved | Reserved | |
| | | 101 | Reserved | Reserved | |
| | | 110 | Reserved | Reserved | |
| | | 111 | External clock (TC4 pin input) | | |
| TC4M | TC4 operating mode select | 00: Timer/event counter mode<br>01: Reserved<br>10: Reserved<br>11: Reserved | | | |

Note 1: fc: High-frequency clock [Hz], *; Don't care

Note 2: Values to be loaded to the timer register must satisfy the following condition (1 ≤ TC4DR ≤ 255).

Note 3: When the TC4 is started (TC4S = 0 → 1) or disabled (TC4S = 1 → 0) or while the TC4 is operating (TC4S = 1 → 1), do not write to TC4M and TC4CK in TC4CR. If these registers are selected/changed during these operations, counting up is not performed properly.

Note 4: When STOP mode is started, timer counter is stopped and cleared. Set TC4S to "1" after STOP mode is released for restarting timer counter.

Note 5: Undefined values are read from bits 6 and 7 of TC4CR.

Note 6: Do not change TC4DR while the TC4 is operating.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| PMPXCR (00027H) | "0" | CHS | | | | | TC4ES | (TC3ES) | (Initial value: 00** **00) |

| | | | |
|---|---|---|---|
| TC4ES | TC4 edge select | 0: Rising edge<br>1: Falling edge | Write only |

Note 1: TC4CR, TC4DR and PMPXCR are write only register and must not be used with any of the read-modify-write instructions such as SET, CLR, etc.

Figure 2.8.2 Timer Register 4 and TC4 Control Register

### 2.8.3 Function

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

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of TC4DR are compared with the contents of up counter. If a match is found, an INTTC4 interrupt is generated and the up counter is cleared to "0". Counting up resumes after the up counter is cleared.

Table 2.8.1 Source Clock (Internal clock) for Timer/Counter 4 (Example: at fc = 16.0 MHz)

| TC4CK | NORMAL, IDLE Mode | | | |
|---|---|---|---|---|
| | DV1CK = 0 | | DV1CK = 1 | |
| | Resolution [μs] | Maximum Setting Time [ms] | Resolution [μs] | Maximum Setting Time [ms] |
| 000 | 128.0 | 32.6 | 256.0 | 65.3 |
| 001 | 8.0 | 2.0 | 16.0 | 4.1 |
| 010 | 2.0 | 0.510 | 4.0 | 1.0 |
| 100 | 0.5 | 0.128 | 1.0 | 0.255 |

(2) Event counter mode

In this mode, the TC4 pin input (External clock) pulse is used for counting up. Either the rising or falling edge can be selected with TC4ES (Bit1 PMPXCR). The contents of TC4DR are compared with the contents of the up counter. If a match is found, an INTTC4 interrupt is generated and the counter is cleared. The maximum applied frequency is shown Table 2.8.2. Two or more machine cycles are required for both the high and low level of the pulse width.

Note: The event counter mode can only be used in NORMAL or IDLE mode.

Table 2.8.2 Timer/Counter 4 External Clock Source

| | Minimum Input Pulse Width [s] |
|---|---|
| | NORMAL1, IDLE1 Mode |
| "H" Width | $2^3$/fc |
| "L" Width | $2^3$/fc |

## 2.9 Serial Bus Interface (SBI-ver. D)

The TMP88CS38 and TMP88CM38ACP38A has a 1-channel serial bus interface which employs a clocked-synchronous 8-bit serial bus interface and an I²C bus (a bus system by Philips). The serial bus interface pins are selectively used as either channel 0 or channel 1.

The serial interface is connected to external devices 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 for the P3/P5 port. When used for 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.

Note 1: When P3 and P5 is used as serial bus interface pins, P35, P34, P51 and P50 should be set as a sink open-drain output by clearing PSELCR to "0".

Note 2: The I²C of TMP88CS38 and TMP88CM38A/CP38A can be used only in the standard mode of I²C. The fast mode and the high speed mode can not be used.

### 2.9.1 Configuration

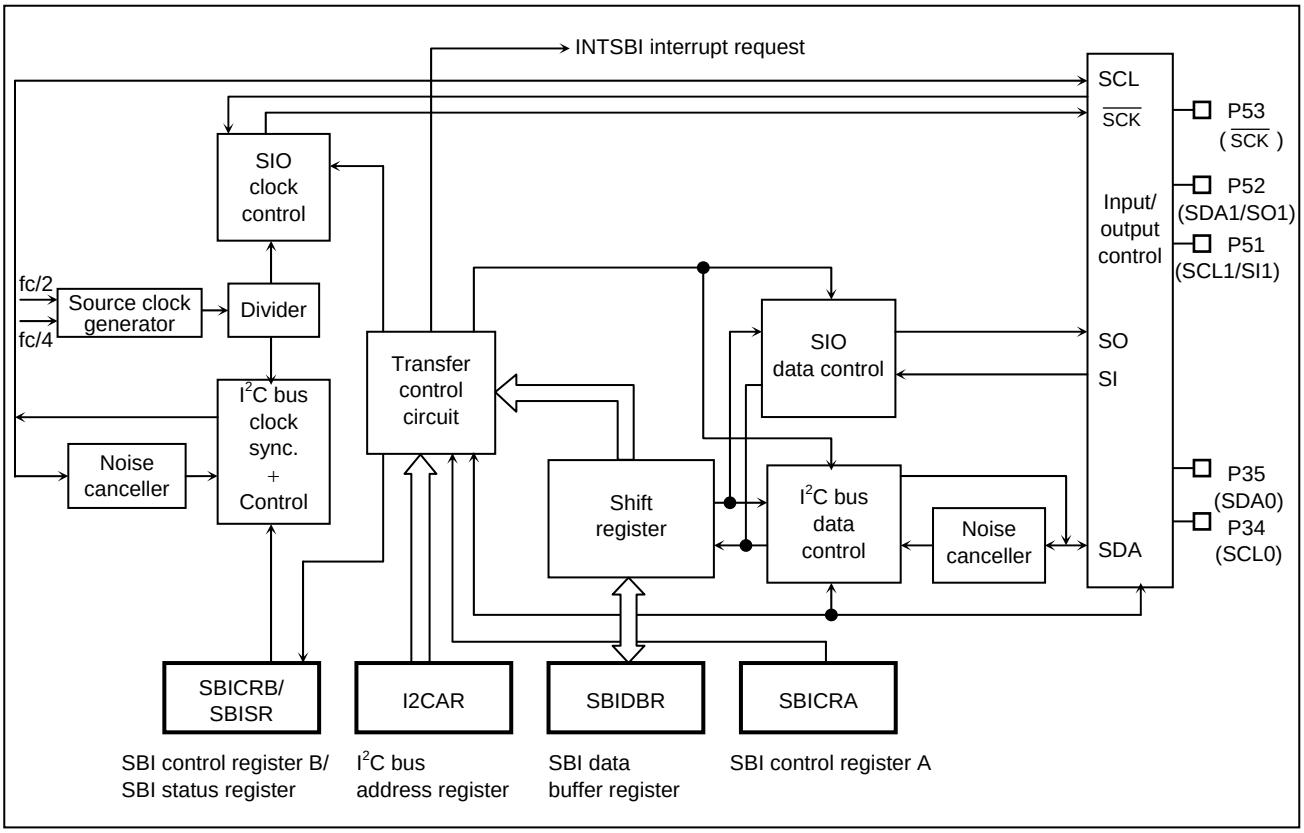



Figure 2.9.1 Serial Bus Interface (SBI)

### 2.9.2 Control

The following registers are used for control the serial bus interface and monitor the operation status.

- Serial bus interface control register A (SBICRA)
- Serial bus interface control register B (SBICRB)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register A (SBISRA)
- Serial bus interface status register B (SBISRB)
- Serial clock source control register (SCCRB)
- Serial clock control status register (SCSR)

The above registers differ depending on a mode to be used. Refer to section 2.9.7 "I²C Bus Mode Control" and 2.9.9 "Clocked-synchronous 8-Bit SIO Mode Control".

### 2.9.3 Serial Clock Source Control

A serial bus interface circuit can reduce the power consumption by stopping a serial clock generater.

Serial Clock Source Control Register

SCCRB (00FF1H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SCEN | - | - | - | - | - | - | - |

(Initial value: 0*** ****)

| | | | |
|---|---|---|---|
| SCEN | Serial clock source control | 0: Do not generate source clock<br>1: Generate source clock | Write only |

Note: When SCRQ and SCEN are "1", SCEN cannot be cleared to "0". When SCRQ is "0", SCEN is cleared to "0".

Serial Clock Control Status Register

SCSR (00FF1H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SCRQ | - | - | - | - | - | - | - |

(Initial value: 0*** ****)

| | | | |
|---|---|---|---|
| SCRQ | Serial clock source request | 0: No source clock request from serial bus interface<br>1: Source clock request from serial bus interface | Read only |

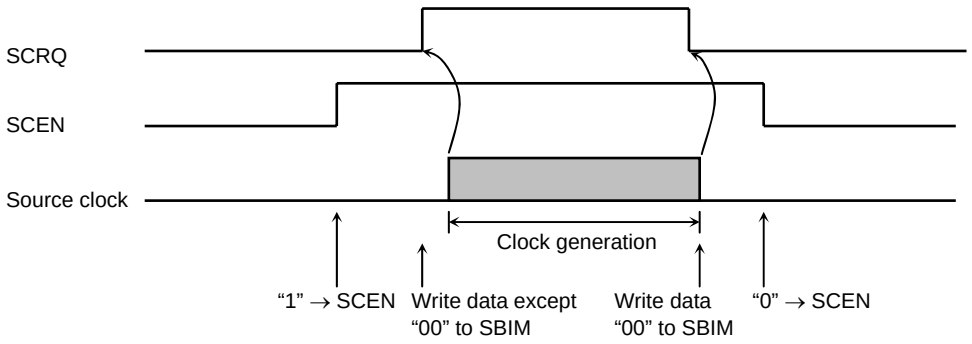



Figure 2.9.2 Serial Clock Source

### 2.9.4 Channel Select

A serial bus interface circuit can select I/O pin when a serial bus interface is used for I²C bus mode.

Port Switching Register

| PMPXCR (00027H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | "0" | CHS | : | : | : | : | (TC4ES) | (TC3ES) | (Initial value: 00** **00) |

| | | | |
|---|---|---|---|
| CHS | I²C bus Channel Select | 0: Channel 0<br>1: Channel 1 | R/W |

Note 1: When SIO mode, don't use channel 0. Therefore, set to "1" in PMPXCR at SIO mode.

Note 2: Always write "0" to bit7 in PMPXCR.

Note 3: *: Don't care

Figure 2.9.3 Channel Select

### 2.9.5 Software Reset

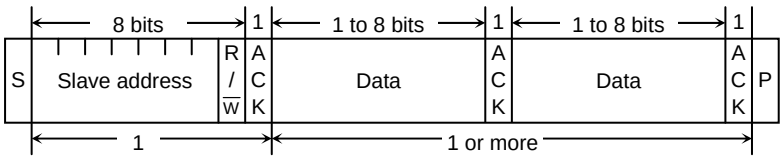
A serial bus interface circuit has a software reset function, when a serial bus interface circuit is locked by an external noise, etc.

To occur software reset, write "01", "10" into the SWRST (Bit1, 0 in SBICRB). During software reset, the SWRMON (Bit0 in SBISRA) is clear to "0".

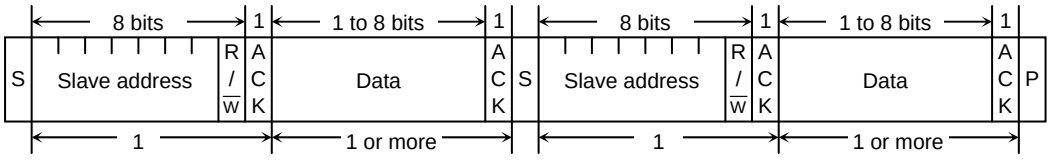
### 2.9.6 The Data Format in The I²C bus Mode

The data format when using the TMP88CS38 AND TMP88CM38A/CP38A in the I²C bus mode are shown in as below.

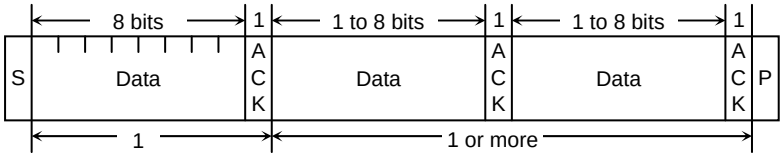
(a) Addressing format




(b) Addressing format (with restart)




(c) Free data format




Notes: S: Start condition
R/$\overline{W}$ : Direction bit
ACK: Acknowledge bit
P: Stop condition

Figure 2.9.4 Data Format in I²C Bus Mode

### 2.9.7 I²C Bus Mode Control

The following registers are used to control the serial bus interface (SBI) and monitor the operation status in the I²C bus mode.

Serial Bus Interface Control Register A

| SBICRA (00020H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | BC | | ACK | | | SCK | |

(Initial value: 0000 *000)

| | | BC | ACK = 0 Number of clock | ACK = 0 Bits | ACK = 1 Number of clock | ACK = 1 Bits | |
|---|---|---|---|---|---|---|---|
| BC | Number of transferred bits | 000 | 8 | 8 | 9 | 8 | Write only |
| | | 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 | Master mode | Slave mode | |
|---|---|---|---|---|---|
| ACK | Acknowledgement mode specification | 0 | Not generate a clock pulse for an acknowledgement. | Not count a clock pulse for an acknowledgement. | R/W |
| | | 1 | Generate a clock pulse for an acknowledgement. | Count a clock pulse for an acknowledgement. | |

| | | DV1CK = 0 | DV1CK = 1 | |
|---|---|---|---|---|
| SCK | Serial clock selection (At fc = 16 MHz, output on SCL pin) | 000: Reserved (Note 3)<br>001: Reserved (Note 3)<br>010: Reserved (Note 3)<br>011: 60.6 kHz<br>100: 30.7 kHz<br>101: 15.5 kHz<br>110: 7.8 kHz<br>111: Reserved | 000: Reserved(Note 3)<br>001: Reserved(Note 3)<br>010: 58.8 kHz<br>011: 30.3 kHz<br>100: 15.4 kHz<br>101: 7.7 kHz<br>110: 3.9 kHz<br>111: Reserved | Write only |

Note 1: Set the BC to "000" before switching to 8-bit SIO bus mode.
Note 2: SBICRA cannot be used with any of read-modify-write instructions such as bit manipulation, etc.
Note 3: This I2C bus circuit does not support the Fast mode. It supports the Standard mode only. Although the I2C bus circuit itself allows the setting of a baud rate over 100 kbps, the compliance with the I2C specification is not guaranteed in that case.

Serial Bus Interface Data Buffer Register

| SBIDBR (00021H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

(Initial value: **** ****) R/W

Note 1: For writing transmitted data, start from the MSB (Bit7).
Note 2: The data which was written into SBIDBR cannot be read, since a write data buffer and a read buffer are independent in SBIDBR. Therefore, SBIDBR cannot be used with any of read-modify-write instructions such as bit manipulation, etc.

I²C bus Address Register

| I2CAR (00022H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Slave address | | | | | | | ALS |
| | SA6 | SA5 | SA4 | SA3 | SA2 | SA1 | SA0 | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| SA | Slave address selection | | Write only |
| ALS | Address recognition mode specification | 0: Slave address recognition<br>1: Non slave address recognition | |

Note 1: I2CAR is write-only register and cannot be used with any of read-modify-write instruction such as bit manipulation, etc.
Note 2: Do not set I2CAR to "00H" to avoid the incorrect response of acknowledgment in slave mode. If "00H" is set to I2CAR as the Slave Address and received "01H" in slave mode, the device might transmit the acknowledgement incorrectly.

Figure 2.9.5 Serial Bus Interface Control Register A, Serial Bus Interface Data Buffer Register and I²C Bus Address Register In The I²C Bus Mode

Serial Bus Interface Control Register B

SBICRB (00023H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MST | TRX | BB | PIN | SBIM | | SWRST1 | SWRST0 |

(Initial value: 0001 0000)

| | | | |
|---|---|---|---|
| MST | Master/slave selection | 0: Slave<br>1: Master | Write only |
| TRX | Transmitter/receiver selection | 0: Receiver<br>1: Transmitter | |
| BB | Start/stop generation | 0: Generate a stop condition when MST, TRX and PIN are “1”.<br>1: Generate a start condition when MST, TRX and PIN are “1”. | |
| PIN | Cancel interrupt service request | 0: –<br>1: Cancel interrupt service request | |
| SBIM | Serial bus interface operating mode selection | 00: Port mode (Serial bus interface output disable)<br>01: Clocked synchronous 8-bit SIO mode<br>10: $I^2C$ bus mode<br>11: Reserved | |
| SWRST1<br>SWRST0 | Software reset start bit | Software reset starts by first writing “10” and next writing “01”. | |

Note 1: Switch a mode to port after confirming that the bus is free.

Note 2: Switch a mode to $I^2C$ bus mode or clock synchronous 8-bit SIO mode after confirming that the port is high level.

Note 3: SBICRB has write-only register and must not be used with any of read-modify-write instructions such as bit manipulation, etc.

Note 4: When the SWRST (Bit1, 0 in SBICRB) is written to “01”, “10”, software reset (Four machine cycles) is occurred.
This time, control the serial bus interface and monitor the operation status registers except the SBIM (Bit3, 2 in SBICRB) and the CHS (Bit6 in PMPXCR) are reseted.
Control the serial bus interface and monitor the operation status registers are SBICRA, SBICRB, SBIDBR, I2CAR, SBISRA, SBISRB, SCCRA and SCSR.

Serial Bus Interface Status Register A

SBISRA (00020H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | ACK | - | - | - | SWR MON |

(Initial value: **** ***1)

| | | | |
|---|---|---|---|
| SWRMON | Software reset monitor | 0: During software reset<br>1: – (Initial) | Read only |

*: Don’t care

Serial Bus Interface Status Register B

SBISRB (00023H)

| 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<br>1: Master | Read only |
| TRX | Transmitter/receiver selection status monitor | 0: Receiver<br>1: Transmitter | |
| BB | Bus status monitor | 0: Bus free<br>1: Bus busy | |
| PIN | Interrupt service requests status monitor | 0: Requesting interrupt service<br>1: Releasing interrupt service request | |
| AL | Arbitration lost detection monitor | 0: –<br>1: Arbitration lost detected | |
| AAS | Slave address match detection monitor | 0: Not detect slave address match or “GENERAL CALL”<br>1: Detect slave address match or “GENERAL CALL” | |
| AD0 | “GENERAL CALL” detection monitor | 0: Not detect “GENERAL CALL”<br>1: Detect “GENERAL CALL” | |
| LRB | Last Received bit monitor | 0: Last receive bit is “0”<br>1: Last receive bit is “1” | |

Figure 2.9.6 Serial Bus Interface Control Register B and Serial Bus Interface Status Register A/B in the $I^2C$ Bus Mode

(1) Acknowledgement mode specification

a. Acknowledgement mode (ACK = "1")

To set the device as an acknowledgement mode, the ACK (Bit4 in SBICRA) should be set to "1". When a serial bus interface circuit is a master mode, an additional clock pulse is generated for an acknowledge signal. In a slave mode, a clock is counted for the acknowledge signal.

In the master transmitter mode, the SDA pin is released in order to receive an acknowledge signal from the receiver during additional clock pulse cycle. In the master receiver mode, the SDA pin is set to low level generation an acknowledge signal during additional clock pulse cycle.

In a slave mode, when a received slave address matches to a slave address which is set to the I2CAR or when a "GENERAL CALL" is received, the SDA pin is set to low level generating an acknowledge signal. After the matching of slave address or the detection of "GENERAL CALL", in the transmitter the SDA pin is released in order to receive an acknowledge signal from the receiver during additional clock pulse cycle. In a receiver, the SDA pin is set to low level generation an acknowledge signal during additional clock pulse cycle after the matching of slave address or the detection of "GENERAL CALL".

The Table 2.9.1 shows the SCL and SDA pins status in acknowledgement mode.

Table 2.9.1 SCL and SDA Pins Status in Acknowledgement Mode

| Mode | Pin | | Transmitter | Receiver |
|---|---|---|---|---|
| Master | SCL | | An additional clock pulse is generated. | |
| | SDA | | Released in order to receive and acknowledge signal. | Set to low level generating an acknowledge signal. |
| Slave | SCL | | A clock is counted for the acknowledge signal. | |
| | SDA | When slave address matches or a general call is detected | – | Set to low level generating an acknowledge signal. |
| | | After matching of slave address or general call | Released in order to receive an acknowledge signal. | Set to low level generating an acknowledge signal. |

b. Non-acknowledgement mode (ACK = "0")

To set the device as a non-acknowledgement mode, the ACK should be cleared to "0". In the master mode, a clock pulse for an acknowledge signal is not generated. In the slave mode, a clock for a acknowledge signal is not counted.

(2) Number of transfer bits

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

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

(3) Serial clock

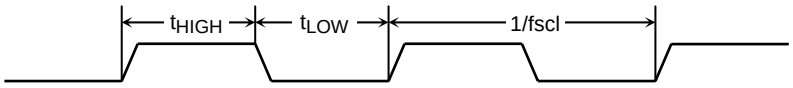
a. Clock source

The SCK (Bits 2 to 0 in SBICRA) is used to select a maximum transfer frequency output from the SCL pin in the master mode. Set a communication baud rate that meets the I2C bus specification, such as the shortest pulse width of tLOW, based on the equations shown below.

Four or more machine cycles are required for both high and low levels of pulse width in the external clock which is input from SCL pin.

Note: Since the $I^2C$ of TMP88CS38 AND TMP88CM38A/CP38A can not be used as the Fast mode and the High Speed mode, do not set SCK as the frequency that is over 100 kHz.

This $I^2C$ bus circuit does not support high-speed mode, it supports standard mode only. Set the baud rates, which have been calculated according to the formula below, to meet the specifications of the I2C bus, such as the smallest pulse width of $t_{LOW,}$




$t_{LOW} = 2^n/fc$

$t_{HIGH} = 2^n/fc + 8/fc$

$fscl = 1/(t_{Low} + t_{HIGH})$

| SCK (Bits 2 to 0 in the SBICRA) | n | |
|---|---|---|
| | DV1CK = 0 | DV1CK = 1 |
| 000 | 4 | 5 |
| 001 | 5 | 6 |
| 010 | 6 | 7 |
| 011 | 7 | 8 |
| 100 | 8 | 9 |
| 101 | 9 | 10 |
| 110 | 10 | 11 |

Note: fc: High-frequency clock

Figure 2.9.7 Clock Source

b. Clock synchronization

In the $I^2C$ bus mode, in order to drive a bus with a wired AND, a master device which pulls down a clock pulse to low will, in the first place, invalidate a clock pulse of another master device which generates a high-level clock pulse.

The serial bus interface circuit has a clock synchronization function. This function ensures normal transfer even if there are two or more masters on the same bus.

The example explains clock synchronization procedures when two masters simultaneously exist on a bus.

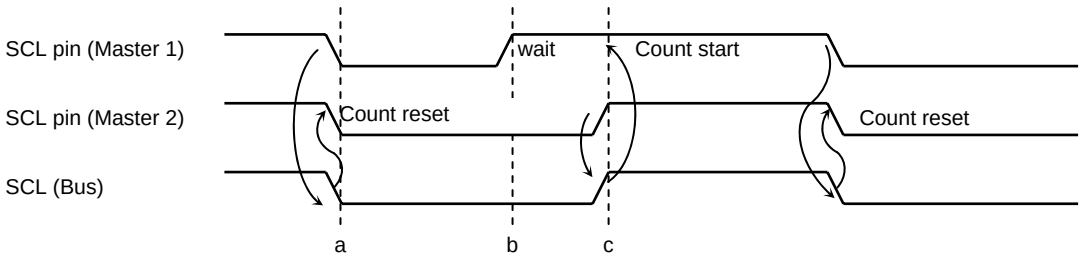



Figure 2.9.8 Clock Synchronization

As master 1 pulls down the SCL pin to the low level at point "a", the SCL line of the bus becomes the low level. After detecting this situation, master 2 resets counting a clock pulse in the high level and sets the SCL pin to the low level.

Master 1 finishes counting a clock pulse in the low level at point "b" and sets the SCL pin to the high level. Since master 2 holds the SCL line of the bus at the low level, master 1 waits for counting a clock pulse in the high level. After master 2 sets a clock pulse to the high level at point "c" and detects the SCL line of the bus at the high level, master 1 starts counting a clock pulse in the high level. Then, the master, which has finished the counting a clock pulse in the high level, pulls down the SCL pin to the low level.

The clock pulse on the bus is deteminded by the master device with the shortest high-level period and the master device with the longest low-level period from among those master devices connected to the bus.

(4) Slave address and address recognition mode specification

When the serial bus interface circuit is used with an addressing format to recognize the slave address, clear the ALS (Bit0 in I2CAR) to "0", and set the SA (Bits 7 to 1 in I2CAR) to the slave address.

When the serial bus interfac circuit is used with a free data format not to recognize the slave address, set the ALS to "1". With a free data format, the slave address and the direction bit are not recognized, and they are processed as data from immediately after start condition.

(5) Master/slave selection

To set a master device, the MST (Bit7 in SBICRB) should be set to "1". To set a slave device, the MST should be cleared to "0".

When a stop condition on the bus or an arbitration lost is detected, the MST is cleared to "0" by the hardware.

(6) Transmitter/receiver selection

To set the device as a transmitter, the TRX (Bit6 in SBICRB) should be set to "1". To set the device as a receiver, the TRX should be cleared to "0". When data with an addressing format is transferred in the slave mode, the TRX is set to "1" by a hardware if the direction bit ($R/\overline{W}$) sent from the master device is "1", and is cleared to "0" by a 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 a hardware if a transmitted direction bit is "1", and is set to "1" by a hardware if it is "0". When an acknowledge signal is not returned, the current condition is maintained.

When a stop condition on the bus or an arbitration lost is detected, the TRX is cleared to "0" by the hardware. The following table show TRX changing conditions in each mode and TRX value after changing.

| Mode | Direction Bit | Conditions | TRX after Changing |
|---|---|---|---|
| Slave mode | "0" | A received slave address is the same value set to I2CAR | "0" |
| | "1" | | "1" |
| Master mode | "0" | ACK signal is returned | "1" |
| | "1" | | "0" |

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

(7) Start/stop condition generation

When the BB (Bit5 in SBICRB) is "0", a slave address and a 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 SBIDBR and set "1" to ACK beforehand.

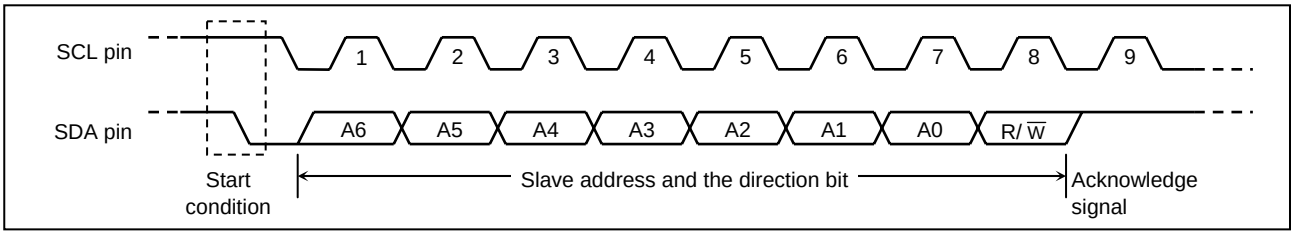



Figure 2.9.9 Start Condition Generation and Slave Address Generation

When the BB is "1", sequence of generating a stop condition is started by writeng "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.

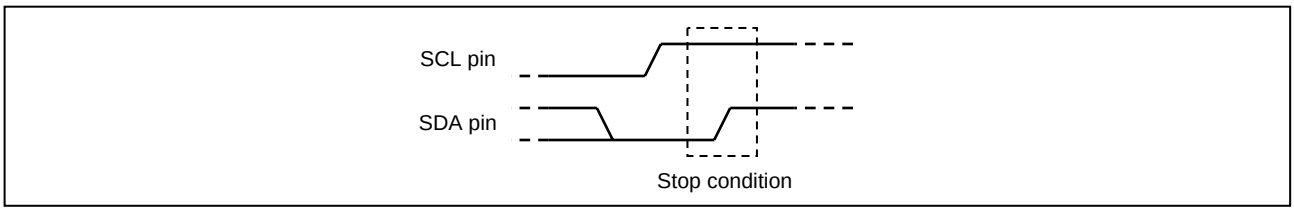



Figure 2.9.10 Stop Condition Generation

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

The bus condition can be indicated by reading the contents of the BB (Bit5 in SBISRB). 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.

(8) Interrupt service request and cancel

When a serial bus interface circuit is in 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 conditions of generating INTSBI are follows:

- At the end of acknowledge signal when the received slave address matches to the value set by the I2CAR
- At the end of acknowledge signal when a "GENERAL CALL" is received
- At the end of transferring or receiving after matching of slave address or receiving of "GENRAL CALL"

When a serial bus interface interrupt request occurs, the PIN (Bit4 in SBISR) is cleared to "0". During the time that the PIN is "0", the SCL pin is pulled down to low level.

Either writing data to SBIDBR or reading data 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 (Bit4 in SBICRB) can be set to "1" by the program, the PIN can not be cleared to "0" by the program.

Note: If the arbitration lost occurs, when the slave address does not match, the PIN is not cleared to "0" even thought INTSBI is generated.

(9) Serial bus interface operating mode selection

The SBIM (Bit3 and 2 in SBICRB) is used to specify a serial bus interface operation mode.

Set the SBIM to "10" in order to change a operation mode to I²C bus mode. Before changing operation mode, confirm serial bus interface pins in a high level. And switch a mode to port after confirming that a bus is free.

(10) Arbitration lost detection monitor

Since more than one master device can exist simultaneously on a bus in the I²C bus mode, a bus arbitration procedure is implemented in order to guarantee the contents of transferred data.

Data on the SDA line is used for bus arbitration of the I²C bus.

The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on a bus. Master 1 and master 2 output the same data until point "a". After master 1 outputs "1" and master 2, "0", the SDA line of a bus is wired AND and the SDA line is pulled down to the low level by master 2. When the SCL line of a bus is pulled up at point "b", the slave device reads data on the SDA line, that is data in master 2.

Data transmitted from master 1 becomes invalid. The state in master 1 is called "arbitration lost". A master device which loses arbitration releases the SDA pin and the SCL pin in order not to effect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.

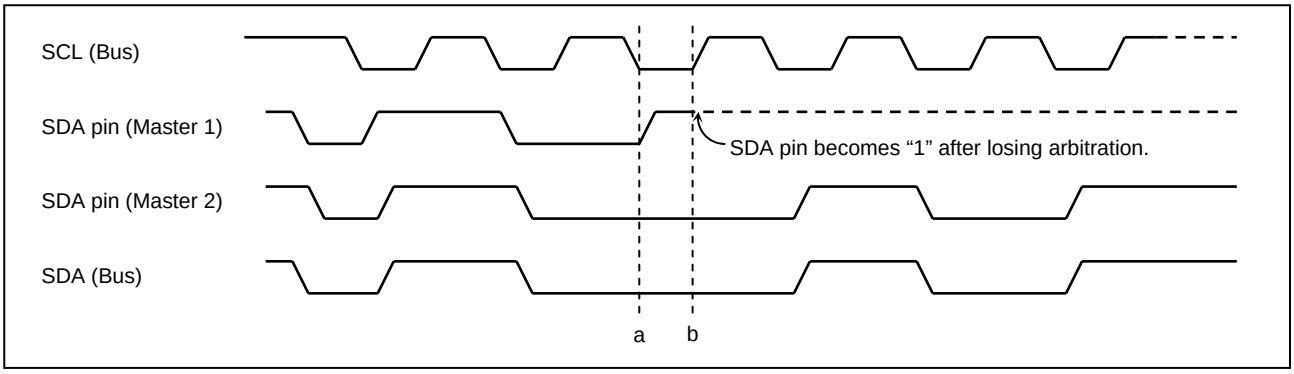



Figure 2.9.11 Arbitration Lost

The serial bus interface circuit compares levels of a SDA line of a bus with its those SDA pin at the rising edge of the SCL line. If the levels are unmatched, arbitration is lost and the AL (Bit3 in SBISRB) is set to “1”.

When the AL is set to “1”, the MST and TRX are cleared to “0” and the mode is switched to a slave receiver mode.

The AL is cleared to “0” by writing or reading data to or from the SBIDBR or writing data to the SBICRB.

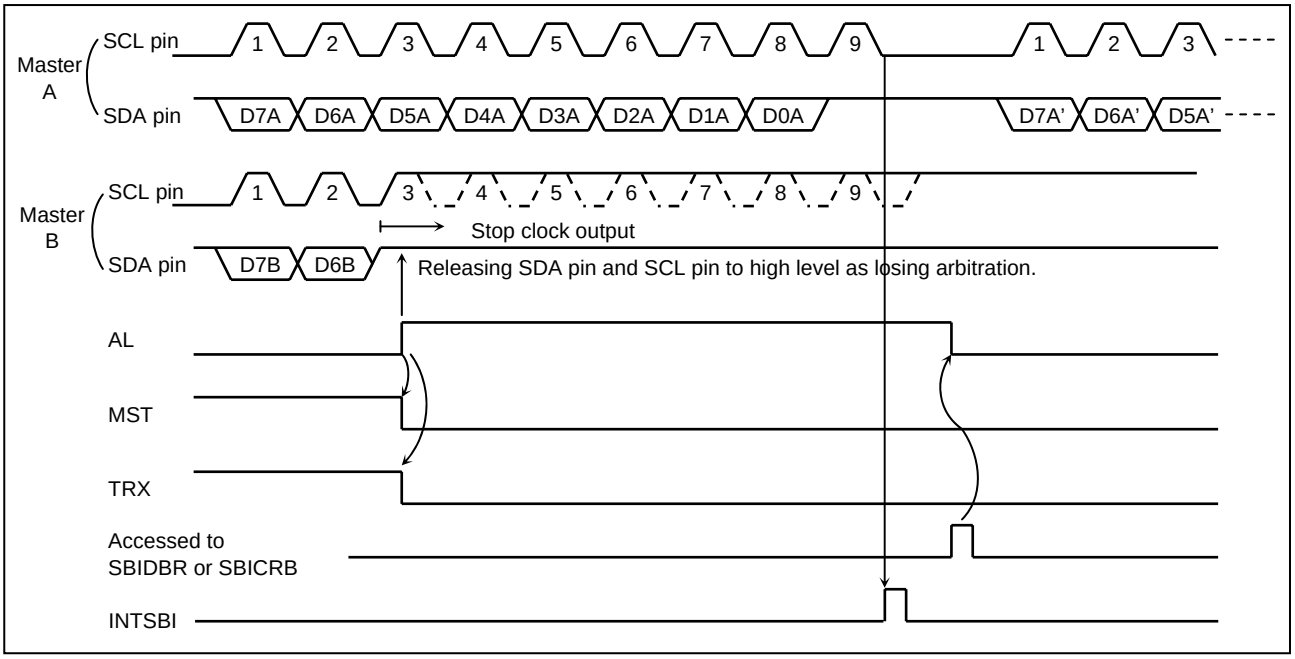



Figure 2.9.12 Example of when a Serial Bus Interface Circuit is a Master B

(11) Slave address match detection monitor

In the slave mode, the AAS (Bit2 in SBISR) is set to “1” when the received data is “GENERAL CALL” or the received data matches the slave address setting by I2CAR with an address recognition mode (ALS = 0).

When a serial bus interface circuit operates in the free data format (ALS = 1), the AAS is set to “1” after receiving the first 1-word of data.

The AAS is cleared to “0” by writing data to the SBIDBR or reading data from the SBIDBR.

(12) GENERAL CALL detection monitor

The AD0 (Bit1 in SBISR) is set to “1” when all 8-bit received data is “0” immediately after a start condition in a slave mode. 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 (Bit0 in SBISRB). In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the LRB.

### 2.9.8 Data Transfer of I$^2$C Bus

(1) Device initialization

For initialization of device, set the ACK in SBICRA to "1" and the BC to "000". Specify the data length to 8 bits to count clocks for an acknowledge signal. Set a transfer frequency to the SCK in SBICRA.

Next, set the slave address to the SA in 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 SBICRB, set "1" to the PIN, "10" to the SBIM, and "00" to bits SWRST1 and SWRST0.

Note: The initialization of a serial bus interface circuit must be complete within the time from all devices which are connected to a bus have initialized to and device does not generate a start condition. If not, the data can not be received correctly because the other device starts transferring before an end of the initialization of a serial bus interface circuit.

(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.

By writing "1" to the MST, TRX, BB and PIN, the start condition is generated on a bus and then, the slave address and the direction bit which are set to the SBIDBR are output. An INTSBI interrupt request occurs at the 9th falling edge of a 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 bits 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 is transferred. If data is written to the SBIDBR, data to been outputting may be destroyed.

Note 2: The bus free must be confirmed by software within 98.0 μs (the shortest transmitting time according to the I$^2$C bus standard) after setting of the slave address to be output. Only when the bus free is confirmed, set "1" to the MST, TRX, BB, and PIN doesn't finish within 98.0 μs, the other masters may start the transferring and the slave address data written in SBIDBR may be broken.

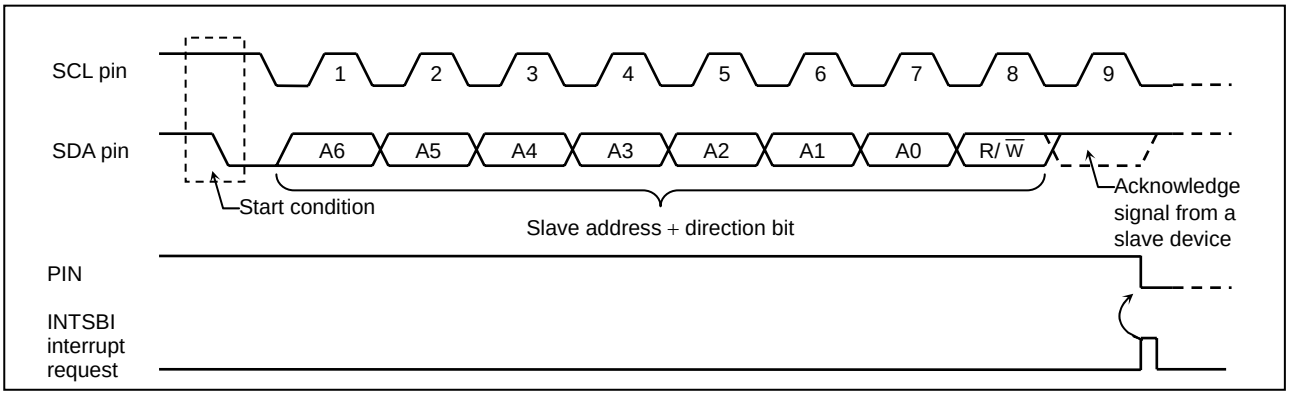



Figure 2.9.13 Start Condition Generation and Slave Address Transfer

(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” (Transmitter 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 next 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 next 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 become “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.

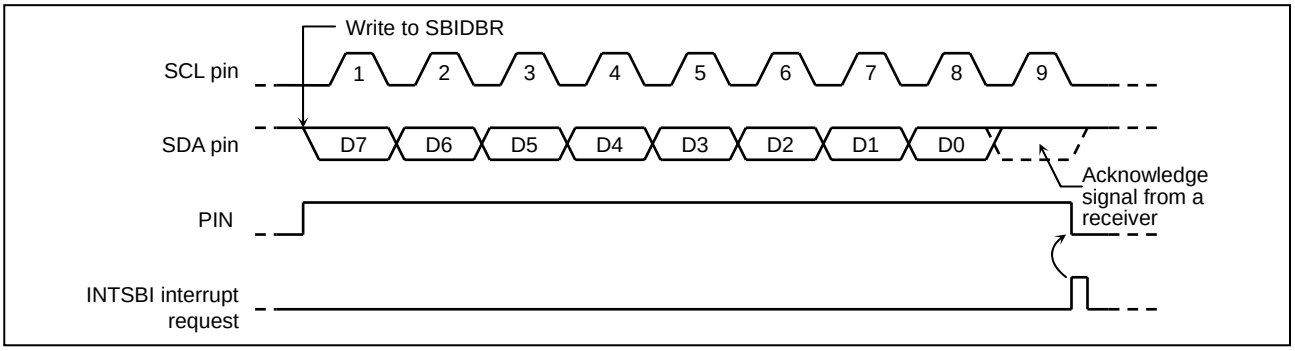



Figure 2.9.14 Example of when BC = “000”, ACK = “1”

2. When the TRX is “0” (Receiver mode)

When the next transmitted data is other than of 8 bits, set the BC again. Set the ACK to “1” and read the received data from the SBIDBR (Reading data is undefined immediately after a slave address is sent). After the data is read, the PIN becomes “1”. A serial bus interface circuit outputs a serial clock pulse to the SCL to transfer next 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 a 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.

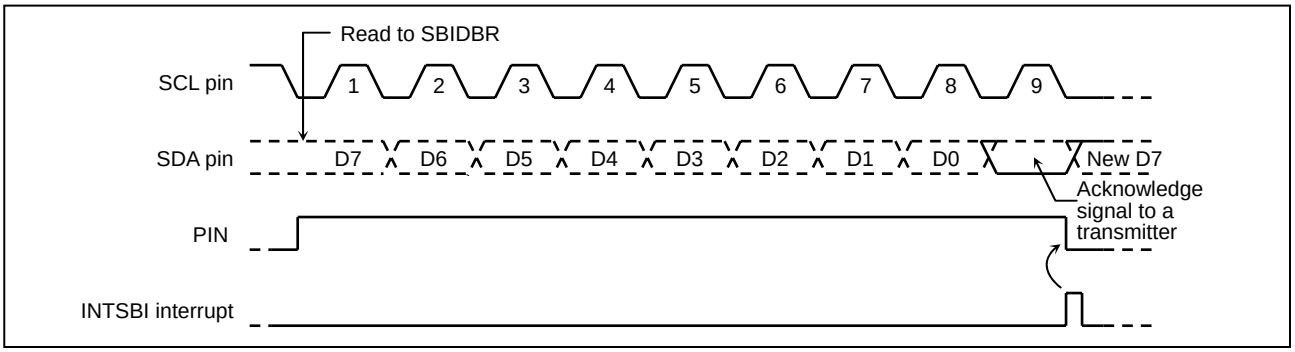



Figure 2.9.15 Example of when BC = “000”, ACK = “1”

To make the transmitter terminate transmit, clear the ACK to "0" before reading data which is 1 word before the last data to be received. A serial bus interface circuit does not generate a clock pulse for the acknowledge signal by clearing ACK. In the interrupt routine of end of transmission, when the BC is set to "001" and read the data, PIN is set to "1" and generates a clock pulse for a 1-bit data transfer. In this case, 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, generates the stop condition to terminate data transter.

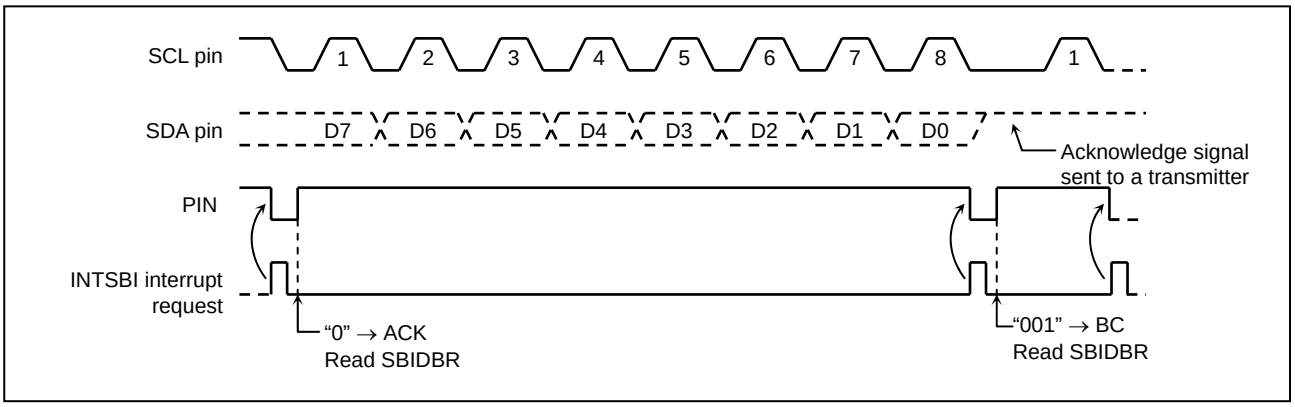



Figure 2.9.16 Termination of Data Transfer in Master Receiver Mode

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

In the slave mode, a serial bus interface circuit operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, the conditions of generating INTSBI are follows:

- When the received slave address matches to the value set by the I2CAR
- When a "GENERAL CALL" is received
- At the end of transferring or receiving after matching of slave address or receiving of "GENERAL CALL"

A serial bus interface circuit changes to a slave mode if arbitration is lost in the master mode. And an INTSBI interrupt request occurs when word data transfer terminates after losing arbitration. The behavior of INTSBI and PIN after losing arbitration are shown in Table 2.9.2.

Table 2.9.2 The Behavior of INTSBI and PIN after Losing Arbitration

| | When the arbitration occurs during transmission of slave address as a master | When the arbitration occurs during transmission of data as a master transmit mode |
|---|---|---|
| INTSBI | INTSIB is generated at the terminatin of word data. | |
| PIN | When the slave address matches the value set by I2CAR, the PIN is cleared to "0" by generating of INTSBI. When the slave address doesn't match the value set by I2CAR, the PIN keeps "1". | PIN keeps "1". |

Check the AL (Bit3 in the SBISR), the TRX (Bit6 in the SBISR), the AAS (Bit2 in the SBISR), and the AD0 (Bit1 in the SBISR) and implements processes according to conditions listed in Table 2.9.3.

Table 2.9.3 Operation in the Slave Mode

| TRX | AL | AAS | AD0 | Conditions | Process |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | A serial bus interface circuit loses arbitration when transmitting a slave address. And receives a slave address of which the value of the direction bit sent from another master is “1”. | Set the number of bits in 1 word to the BC and write transmitted data to the SBIDBR. |
| | 0 | 1 | 0 | In the slave receiver mode, a serial bus interface circuit receives a slave address of which the value of the direction bit sent from the master is “1”. | |
| | | 0 | 0 | In the slave transmitter mode, 1-word data is transmitted. | Test 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 set to “0”, set the number of bits in 1-word to the BC and write transmitted data to the SBIDBR since the receiver requests next data. |
| 0 | 1 | 1 | 1/0 | A serial bus interface circuit loses arbitration when transmitting a slave address. And receives a slave address of which the value of the direction bit sent from another master is “0” or receives a “GENERAL CALL”. | Read the SBIDBR for setting the PIN to “1” (Reading dummy data) or write “1” to the PIN. |
| | | 0 | 0 | A serial bus interface circuit loses arbitration when transmitting a slave address or data. And terminates transferring word data. | A serial bus interface circuit is changed to slave mode. To clear AL to “0”, read the SBIDBR or write the data to SBIDBR. |
| | 0 | 1 | 1/0 | In the slave receiver mode, a serial bus interface circuit receives a slave address of which the value of the direction bit sent from the master is “0” or receives “GENERAL CALL”. | Read the SBIDBR for setting the PIN to “1” (Reading dummy data) or write “1” to the PIN. |
| | | 0 | 1/0 | In the slave receiver mode, a serial bus interface circuit terminates receiving of 1-word data. | Set the number of bits in 1 word to the BC and read received data from the SBIDBR. |

Note: In the slave mode, if the slave address set in I2CAR is “00000000B”, the TRX changes to “1” by receiving the start byte data “00000001B”.

(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 clear “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 on a bus is pulled down by other devices, a serial bus interface circuit generates a stop condition after they release a SCL line.

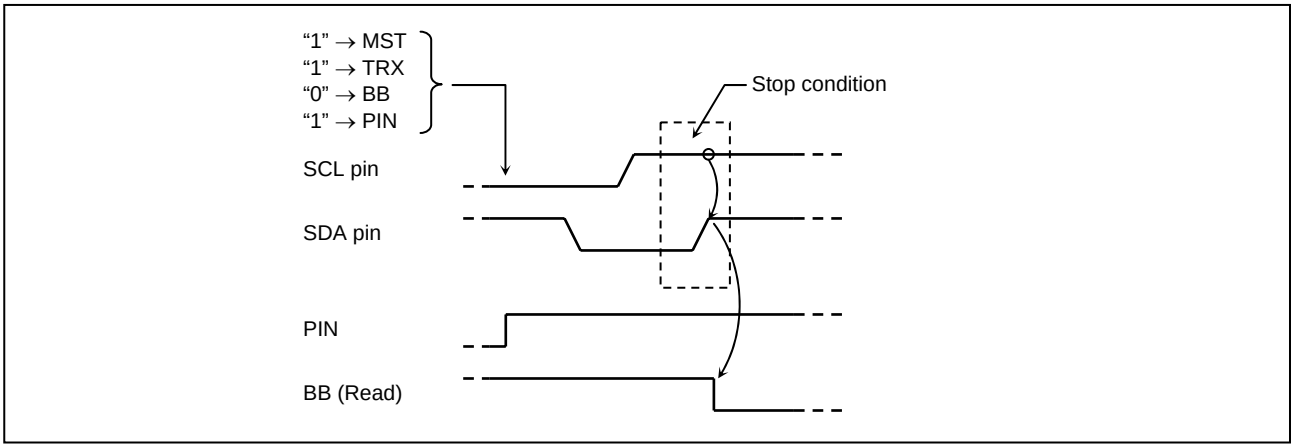



Figure 2.9.17 Stop Condition Generation

(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 a 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 a bus, a 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 a serial bus interface circuit is released. Test the LRB until it becomes "1" to check that the SCL line on a bus is not pulled down to the low level by other devices. After confirming that a 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 μs of waiting time by software from the time of restarting to confirm that a bus is free until the time to generate a start condition.

Note: When restarting after receiving in master receiver mode, because the divice doesn't send an acknowledgement as a last data, the level of SCL line can not be conrirmied by reading LRB. Therefore, confirm the status of SCL line by reading P5PRD register.

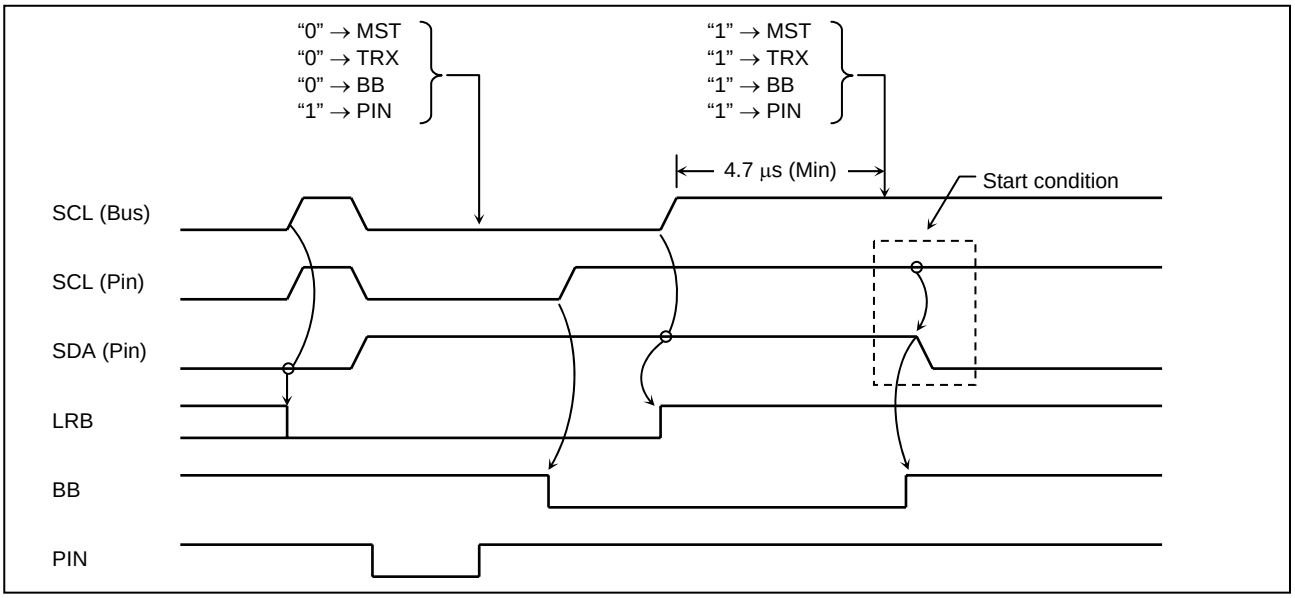



Figure 2.9.18 Timing Diagram when Restarting

### 2.9.9 Clocked-synchronous 8-Bit SIO Mode Control

The following registers are used to control the serial bus interface (SBI) and monitor the operation in the clocked-synchronous 8-bit SIO mode.

Serial Bus Interface Control Register A

| SBICRA (00020H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SIOS | SIOINH | SIOM | | "0" | SCK | | |

(Initial value: 0000 *000)

| | | | | |
|---|---|---|---|---|
| SIOS | Indicate transfer start/stop | 0: Stop<br>1: Start | | Write only |
| SIOINH | Continue/abort transfer | 0: Continue transfer<br>1: Abort transfer (Automatically cleared after abort) | | |
| SIOM | Transfer mode select | 00: 8-bit transmit mode<br>01: Reserved<br>10: 8-bit transmit/receive mode<br>11: 8-bit receive mode | | |
| SCK | Serial clock selection (at fc = 16 MHz, Output on $\overline{SCK}$ pin) | DV1CK = 0 | DV1CK = 1 | |
| | | 000: 1000.0 kHz<br>001: 500.0 kHz<br>010: 250.0 kHz<br>011: 125.0 kHz<br>100: 62.5 kHz<br>101: 31.2 kHz<br>110: 15.6 kHz<br>111: External clock (Input from $\overline{SCK}$ pin) | 000: 500.0 kHz<br>001: 250.0 kHz<br>010: 125.0 kHz<br>011: 62.5 kHz<br>100: 31.2 kHz<br>101: 15.6 kHz<br>110: 7.8 kHz<br>111: External clock (Input from $\overline{SCK}$ pin) | |

Note 1: fc: High-frequency clock [Hz], *: Don't care
Note 2: Clear the SIOS to "0" and set the SIOINH to "1" when setting the transfer mode and serial clock.
Note 3: SBICRA is write-only register and cannot be used with any of read-modify-write instructions such as bit manipulation, etc.

Serial Bus Interface Data Register

| SBIDBR (00021H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

(Initial value: **** ****) R/W

Note1 : The data which was written into SBIDBR cannot be read, since a write buffer and a read buffer are independent in SBIDBR. Therefore, SBIDBR cannot be used with any of read-modify-write instructions such as bit manipulation, etc.
Note 2: *: Don't care

Serial Bus Interface Control Register B

| SBICRB (00023H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | "0" | "0" | "0" | "1" | SBIM | | SWRST1 | SWRST0 |

(Initial value: **** 0000)

| | | | |
|---|---|---|---|
| SBIM | Serial bus interface operation mode selection | 00: Port mode (Serial bus interface output disable)<br>01: SIO mode<br>10: $I^2C$ bus mode<br>11: Reserved | Write only |
| SWRST1<br>SWRST0 | Software reset start bit | Software reset starts by first writing "10" and next writing "01" | |

Note 1: *: Don't care
Note 2: Switch a mode to port after data transfer is complete.
Note 3: Switch a mode to $I^2C$ bus mode or clock synchronous 8-bit SIO mode after confirming that the port is high level.
Note 4: SBICRB is a write-only register and cannot be used with any of read-modify-write instructions such as bit manipulation, etc.
Note 5: Clear bit7 to 5 in SBICRB to "0", and set bit4 to "1".
Note 6: When the SWRST (Bit1, 0 in SBICRB) is written to "01", "10", software reset is occurred.
This time, control the serial bus interface and monitor the operation status registers except the SBIM (Bit3, 2 in SBICRB) and the CHS (Bit6 in PMPXCR) are reseted.
Control the serial bus interface and monitor the operation status registers are SBICRA, SBICRB, SBIDBR, I2CAR, SBISRA, SBISRB, SCCRA, SCCRB and SCSR.

Figure 2.9.19 Control Register/Data Buffer Register/Status Register in SIO Mode (1)

Serial Bus Interface Status Register A

| SBISRA (00020H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | - | - | - | - | - | - | - | SWR MON |

(Initial value: **** ***1)

| | | | |
|---|---|---|---|
| SWRMON | Software reset monitor | 0: During software reset<br>1: – (Initial) | Read only |

Serial Bus Interface Status Register B

| SBISRB (00023H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | "1" | "1" | "1" | "1" | SIOF | SEF | "1" | "1" |

| | | | |
|---|---|---|---|
| SIOF | Serial transfer operating status monitor | 0: Transfer terminated<br>1: Transfer in process | Read only |
| SEF | Shift operating status monitor | 0: Shift operation terminated<br>1: Shift operation in process | |

Note: Set bit7 to 4, bit1 and bit0 in SBISRB to "1".

Figure 2.9.20 Control Register/Data Buffer Register/Status Register in SIO Mode (2)

(1) Serial clock

a. Clock source

The SCK (Bits 2 to 0 in SBICRA) 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.

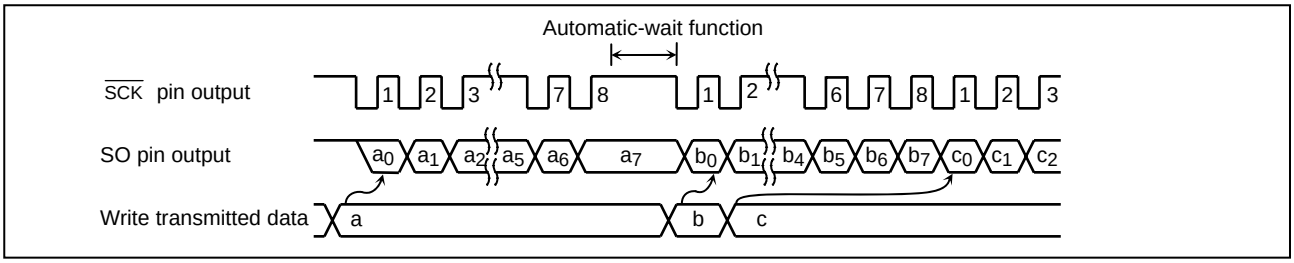



Figure 2.9.21 Automatic Wait Function

2. External (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 2-machine cycles is required for both high and low levels in the serial clock. The maximum data transfer frequency is 1MHz (fc = 16.0 MHz).

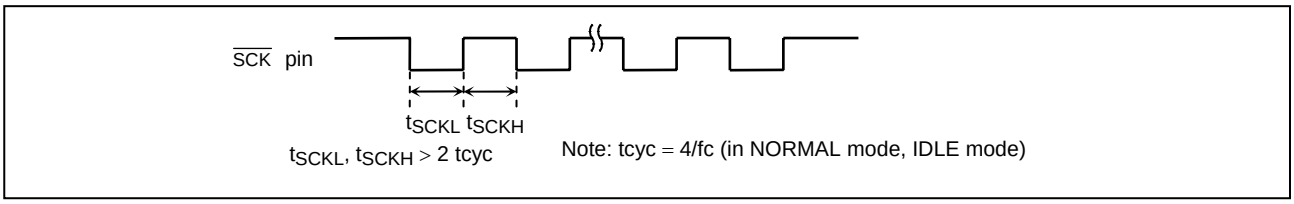



Figure 2.9.22 The Maximum Data Transfer Frequency in The External Clock Input

b. Shift edge

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

1. Leading edge

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

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

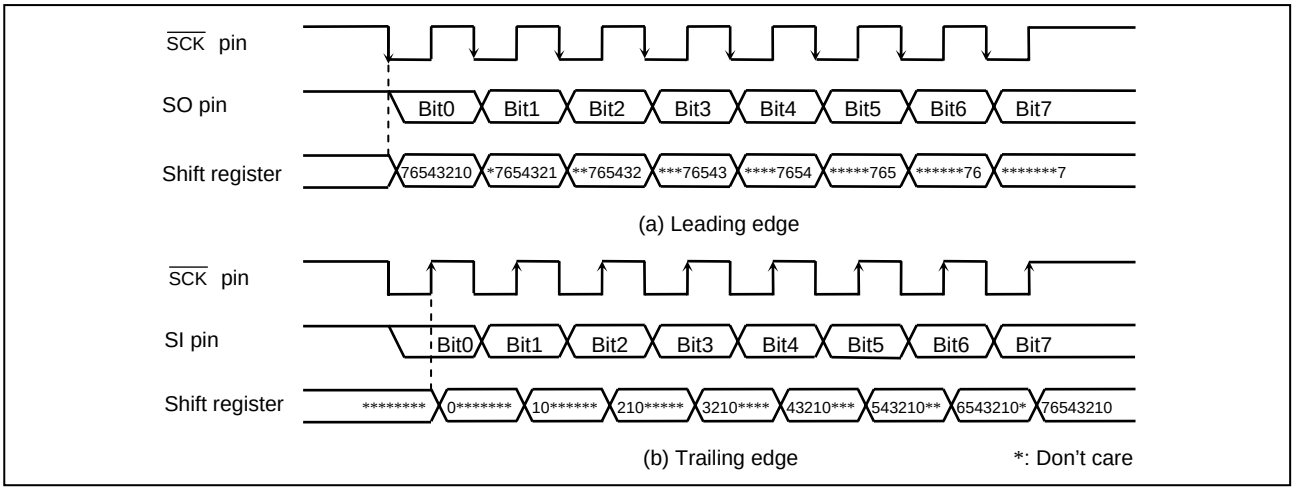



Figure 2.9.23 Shift Edge

(2) Transfer mode

The SIOM (Bits 5 and 4 in SBICRA) 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 transmit new 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 SO pin is "1" from the time transmission starts until the first data bit is sent. When SIOF becomes "0", the shift register is cleared. So, output of an undefined value is not prevented at the start of the next transmission.

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.

Transmitting data is ended by cleaning 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 (Bit3 in the SBISRB) 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.

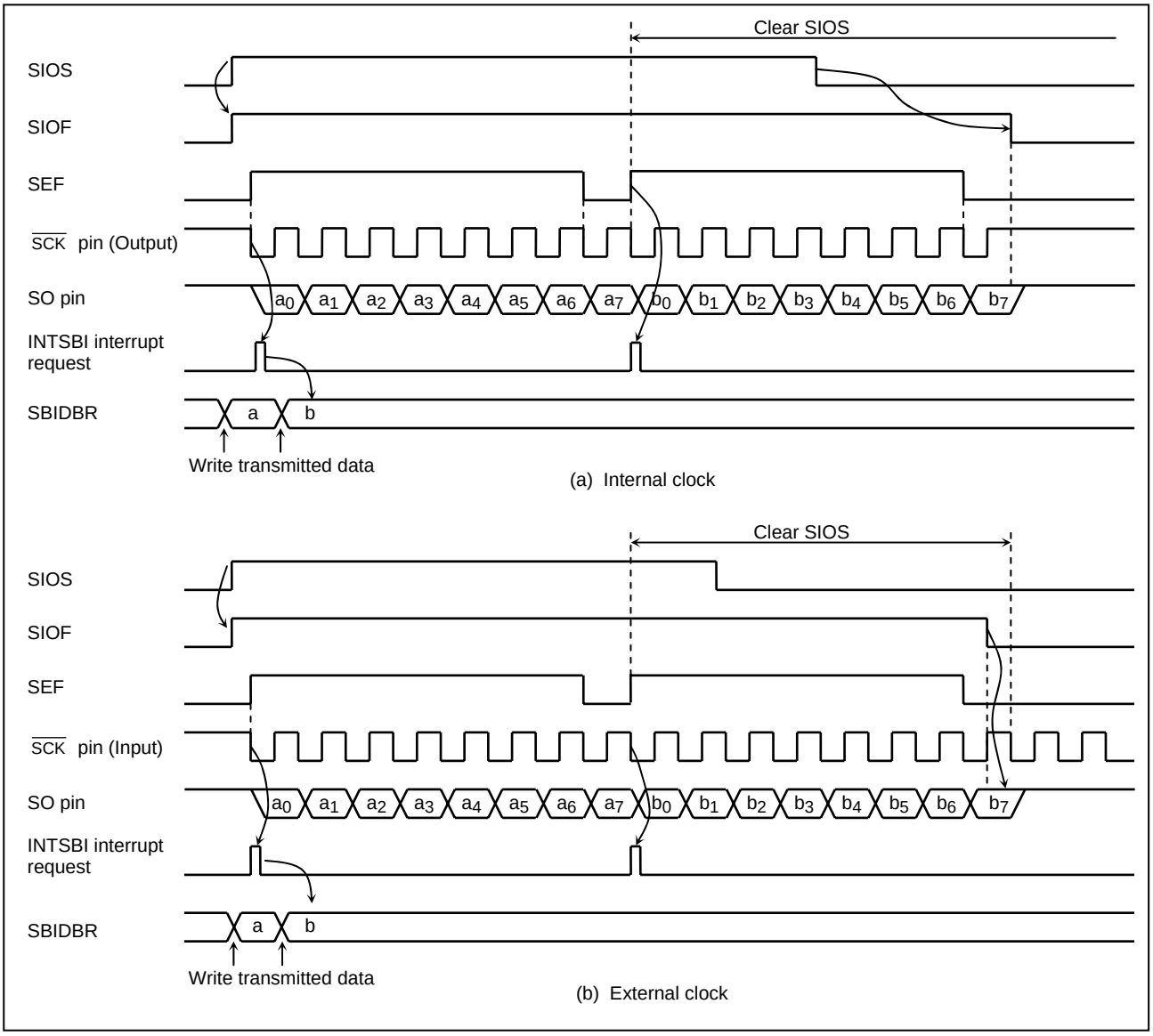



Figure 2.9.24 Transfer Mode

```
Example:  Program to stop transmitting data. (When external clock is used.)
STEST1:     TEST    (SBISRB). SEF          ;   If SEF = 1 then loop
            JRS     F, STEST1
STEST2:     TEST    (P5). 3                ;   If SCK = 0 then loop
            JRS     T, STEST2
            LD      (SBICRA), 00000111B    ;   SIOS ← 0
```

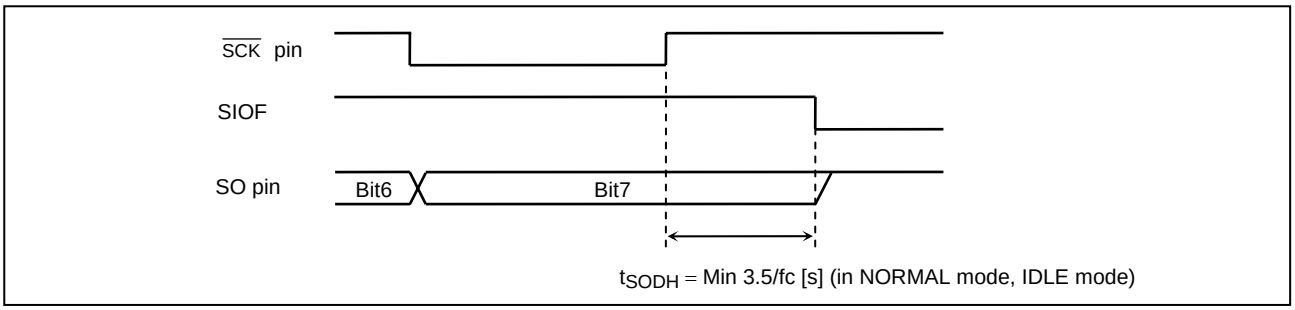



Figure 2.9.25 Transmitted Data Hold Time at End of Transmit

b. 8-bit receive mode

Set a control register to a receive mode and the SIOS to "1" for switching to a 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 read from the SBIDBR by the interrupt service program.

When the external clock is used, since shift operation is synchronized with the clock pulse provided externally, the received data should be read from SBIDBR before next serial clock is input. If the received data is not read, further data to be received is canceled.

When the internal clock is used, the automatic wait function is executed until received data is read from SBIDBR.

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.

Received data disappears if this data is not completely read before reception of the next data terminates. In this case, the next data received 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 (Bit3 in 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.

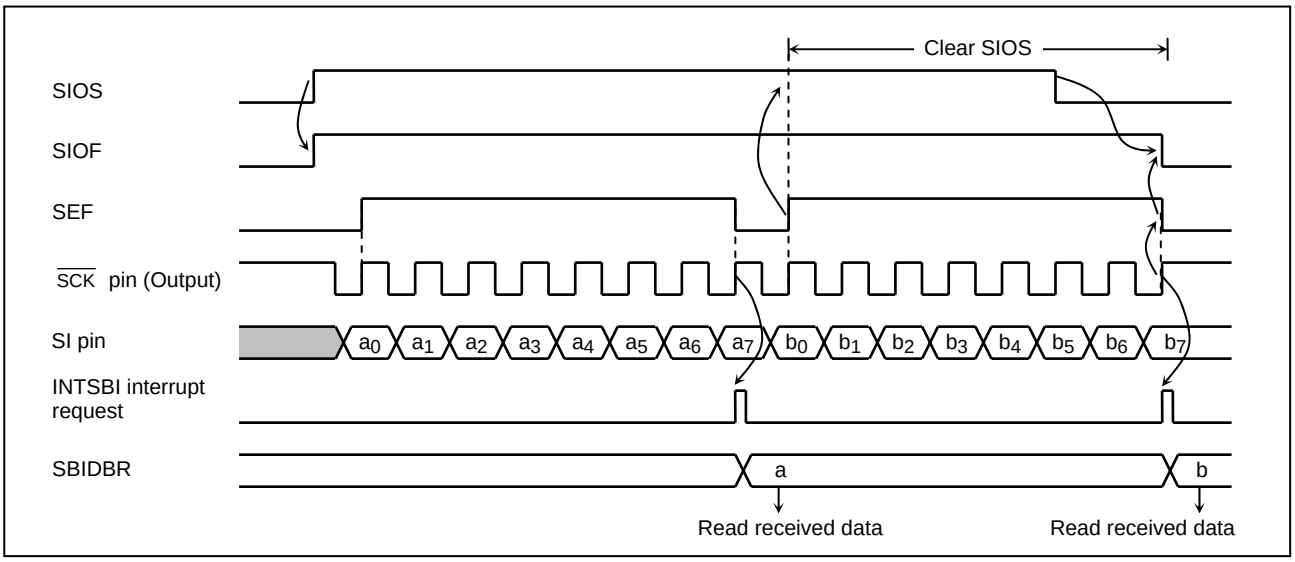



Figure 2.9.26 Receive Mode (Example: Internal clock)

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 transmission starts, a value which is the same as the last bit of previously transmitted data is output from the time SIOF is set to "1" until the falling edge of $\overline{\text{SCK}}$ occurs.

Transmitting/receiving data is ended by cleaning the SIOS to "0" by the INTSBI interrupt service program or setting the SIONH 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 SBISRB) to be sensed. The SIOF becomes "0" after transmitting/receiving is complete. When the SIONH 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.

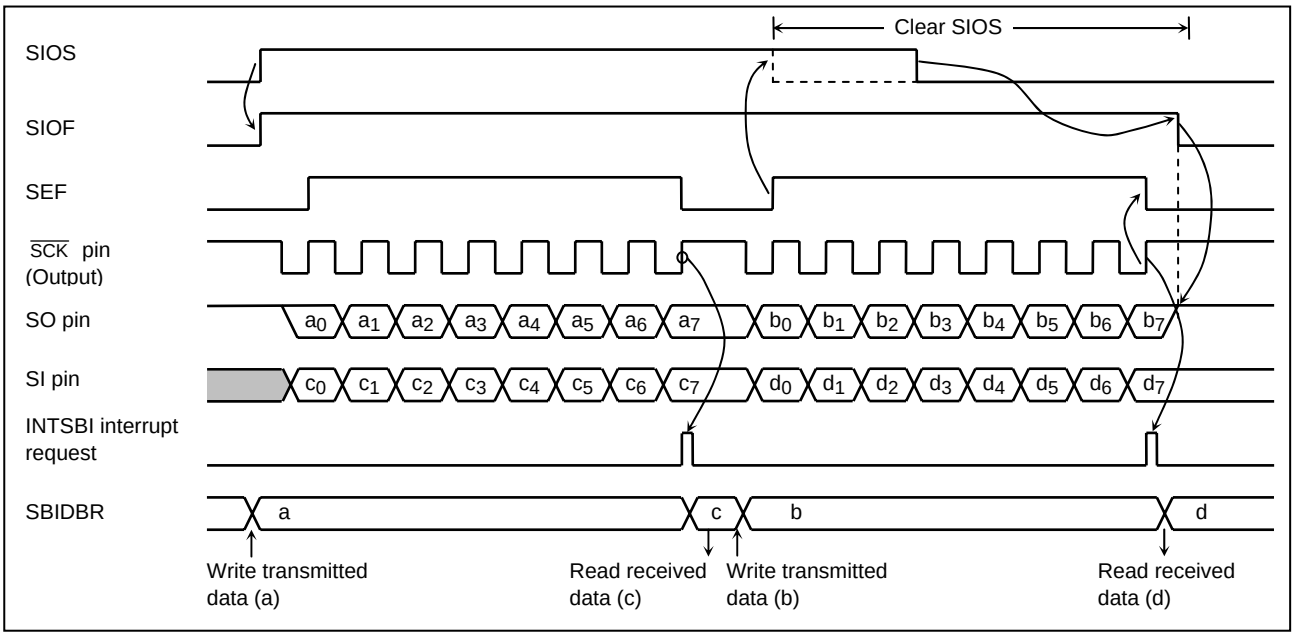



Figure 2.9.27 Transmit/Receive Mode (Example: Internal clock)

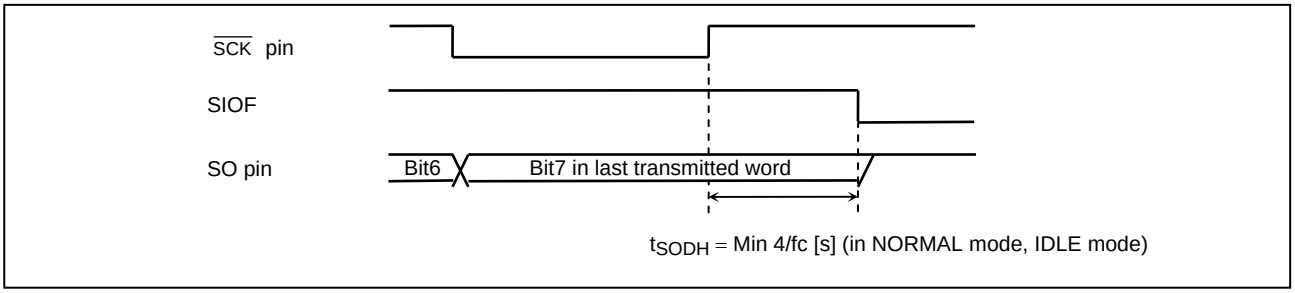



Figure 2.9.28 Transmitted Data Hold Time at End of Transmit/Receive

## 2.10 Remote Control Signal Preprocessor/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 to P30 (INT3/RXIN) pin. When the remote control signal preprocessor/external interrupt 3 pin 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 preprocessor/external interrupt 3 input pin, it can be used for normal port.

### 2.10.1 Configuration

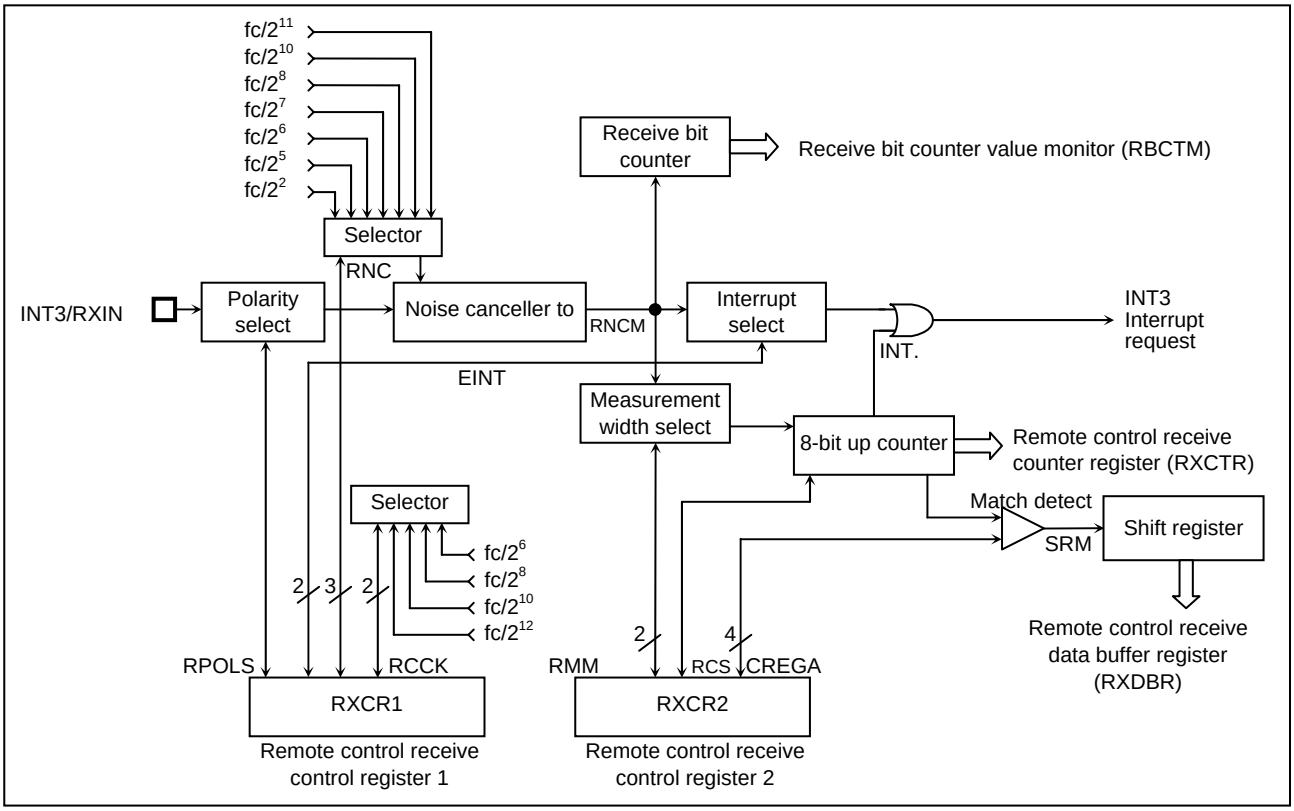



Figure 2.10.1 Remote Control Signal Preprocessor

### 2.10.2 Remote Control Signal Preprocessor Control

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

- Remote control receive control register 1 (RXCR1)
- Remote control receive control register 2 (RXCR2)
- Remote control receive counter register (RXCTR)
- Remote control receive data buffer register (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".

Remote Control Receive Control Register 1

RXCR1 (00FE8H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RCCK | | RPOLS | EINT | | RNC | | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| RCCK | 8-bit up counter source clock select | 00: $fc/2^6$ (Hz)<br>01: $fc/2^8$<br>10: $fc/2^{10}$<br>11: $fc/2^{12}$ | R/W |
| RPOLS | Remote control signal polarity select | 0: Positive<br>1: Negative | |
| EINT | Interrupt source select | 00: Rising edge<br>01: Falling edge (at RPOLS = 0)<br>10: Rising/falling edge<br>11: 8-bit receive end | |
| RNC | Noise canceler noise eiminating time select | 001: $2^2/fc \times 7 - 1/fc$ (s)<br>010: $2^5/fc \times 7 - 1/fc$<br>011: $2^6/fc \times 7 - 1/fc$<br>100: $2^7/fc \times 7 - 1/fc$<br>101: $2^8/fc \times 7 - 1/fc$<br>110: $2^{10}/fc \times 7 - 1/fc$<br>111: $2^{12}/fc \times 7 - 1/fc$<br>000: Noise canceler disable | |

Note 1: fc: High-frequency clock [Hz]

Note 2: After reset, RPOLS do not change the set value in the receiving remote control signal. For setting interrupt edge and measurement data, use EINT and RMM.

Remote Control Receive Control Register 2

RXCR2 (00FE9H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CREGA | | | | RCS | RMCEN | RMM | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| CREGA | Setting of detect time for match with 8-bit up counter upper 4 bits | Match detect time (Tth) = 16 × CREGA/RCCK [s]<br>CREGA = 0H to FH<br>Example: CREGA = 2H, RCCK = $fc/2^6$ [Hz], at fc = 16 MHz, DV1CK = 1<br>Tth = 128 [μs] | R/W |
| RCS | 8-bit up counter start control | 0: Stop and counter clear<br>1: Start | |
| RMCEN | Remote control signal preprocesser enable/disable | 0: Disable<br>1: Enable | |
| RMM | Measurement mode select (invalid when EINT = "10") | 00:<br>01: Refer to Talbe 2.10.1<br>10:<br>11: | |

Note 1: fc: High-frequency clock [Hz]

Note 2: When an interrupt source is set for rising/falling edge, low and high widths are forcibly measured separately.

Note 3: Set CREGA (0H to FH) before EINT sets to 8-bit receive end.

Figure 2.10.2 Remote Control Receive Control Register 1, 2

Remote Control Receive Counter Register

| RXCTR (00FEAH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Read Only (Initial value: 0000 0000) |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |

Remote Control Receive Data Buffer Register

| RXDBR (00FEBH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Read Only (Initial value: 0000 0000) |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |

Remote Control Receive Status Register

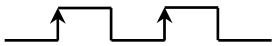
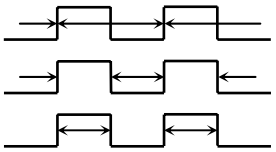
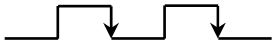
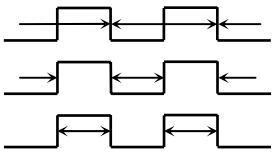
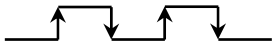
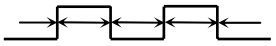
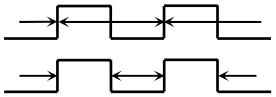
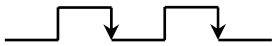
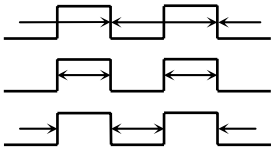
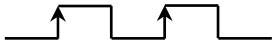
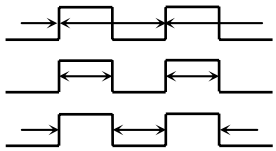
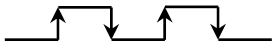
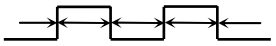
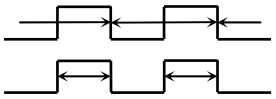
| RXSR (00FECH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Read Only (Initial value: 0000 *000) |
|---|---|---|---|---|---|---|---|---|---|
| | RBCTM | | | | | OVFF | SRM | RNCM | |

| | | | |
|---|---|---|---|
| RBCTM | Receive bit counter value monitor | | Read only |
| OVFF | 8-bit up counter overflow flag | 0: No overflow<br>1: Overflow | |
| SRM | Data buffer register input monitor | 0: Upper 4 bits of 8-bit up counter < CREGA<br>1: Upper 4 bits of 8-bit up counter ≥ CREGA | |
| RNCM | Remote control signal monitor after passing through noise canceler | | |

*: Don't care

Figure 2.10.3 Remote Control Receive Counter Register, Data Buffer Register, Status Register

Table 2.10.1 Combination of Interrupt Source and Measurement Mode

| RPOLS | EINT | RMM | Interrupt Source | Measurement Mode |
|---|---|---|---|---|
| 0 | 00 | 00<br>10<br>11 |  |  |
| | 01 | 01<br>10<br>11 |  |  |
| | 10 | – |  |  |
| | 11 | 00<br>10 | Receive end |  |
| 1 | 00 | 00<br>10<br>11 |  |  |
| | 01 | 01<br>10<br>11 |  |  |
| | 10 | – |  |  |
| | 11 | 00<br>10 | Receive end |  |

### 2.10.3 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 2.10.2 Noise Elimination Time Setting (fc = 16 MHz)

| RNC | Minimum Signal Pulse Width | | Maximum Noise Width to be Eliminated | |
|---|---|---|---|---|
| 000 | – | | – | |
| 001 | $(2^5 + 5)/fc$ | (2.31 μs) | $(2^2 \times 7 - 1)/fc$ | (1.69 μs) |
| 010 | $(2^8 + 5)/fc$ | (16.31 μs) | $(2^5 \times 7 - 1)/fc$ | (13.88 μs) |
| 011 | $(2^9 + 5)/fc$ | (32.31 μs) | $(2^6 \times 7 - 1)/fc$ | (27.88 μs) |
| 100 | $(2^{10} + 5)/fc$ | (64.31 μs) | $(2^7 \times 7 - 1)/fc$ | (55.88 μs) |
| 101 | $(2^{11} + 5)/fc$ | (128.3 μs) | $(2^8 \times 7 - 1)/fc$ | (111.9 μs) |
| 110 | $(2^{13} + 5)/fc$ | (512.3 μs) | $(2^{10} \times 7 - 1)/fc$ | (447.9 μs) |
| 111 | $(2^{14} + 5)/fc$ | (1.024 ms) | $(2^{11} \times 7 - 1)/fc$ | (895.9 μs) |

### 2.10.4 Operation

(1) Interrupts at rising, falling, or rising/falling edge, and measurement modes

First set EINT and RMM. Next, set RCS to “1”; 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”.

Setting RCS to “1” zero clears OVFF.

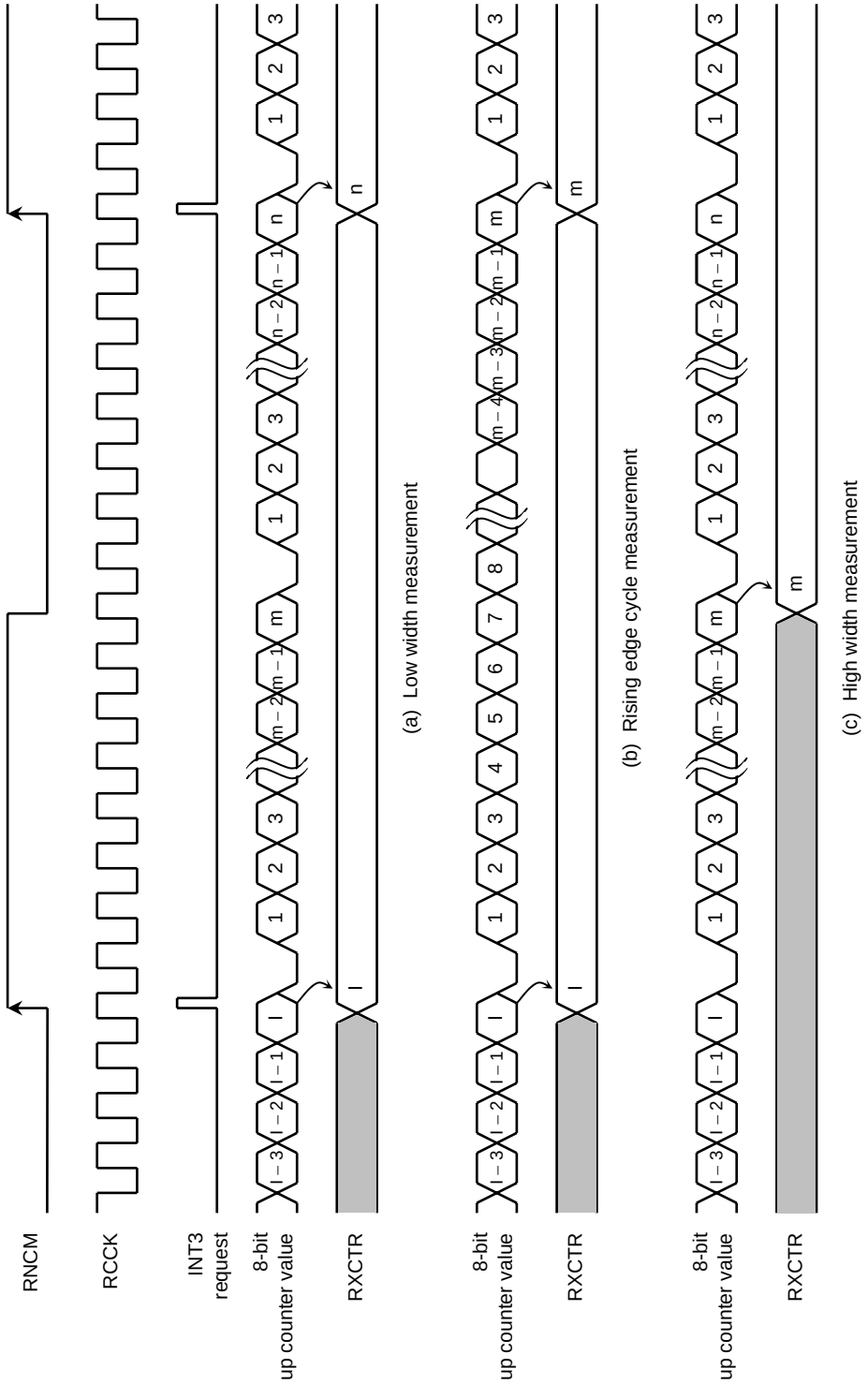



Figure 2.10.4 Rising Edge Interrupt Timing Chart (RPOLS = 0)

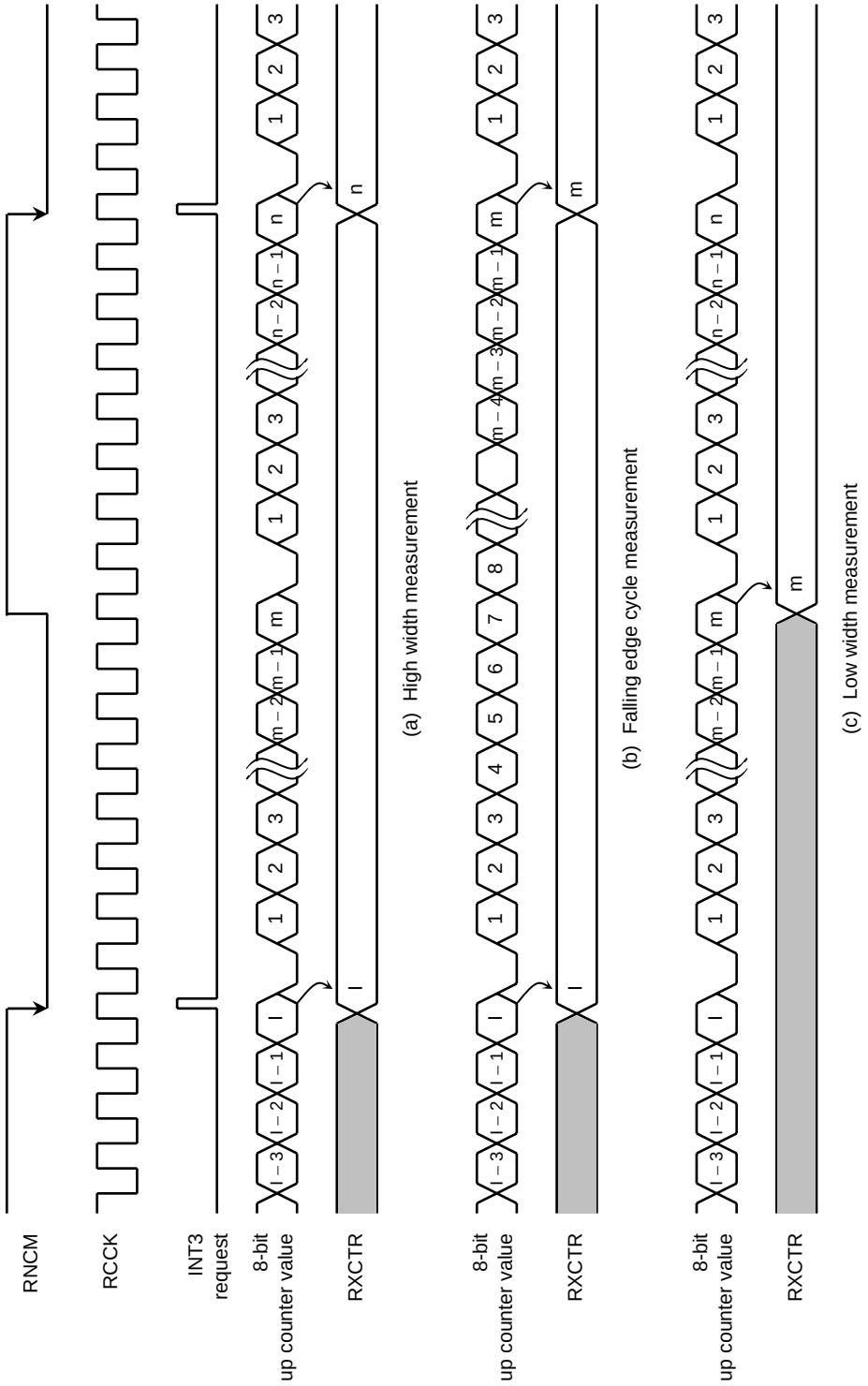



Figure 2.10.5 Falling Edge Interrupt Timing Chart (RPOLS = 0)

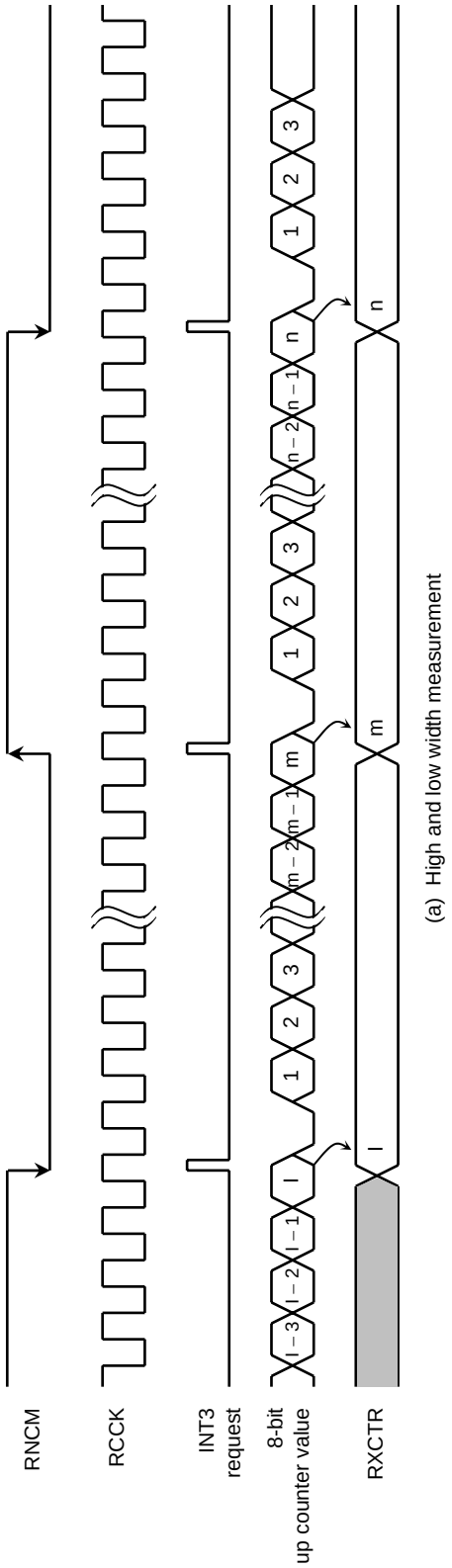



Figure 2.10.6 Rising/Falling Edge Interrupt Timing Chart

(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 is saved, bit by bit, in RXDBR at the rising edge of the remote control signal (when RPOLS = 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.

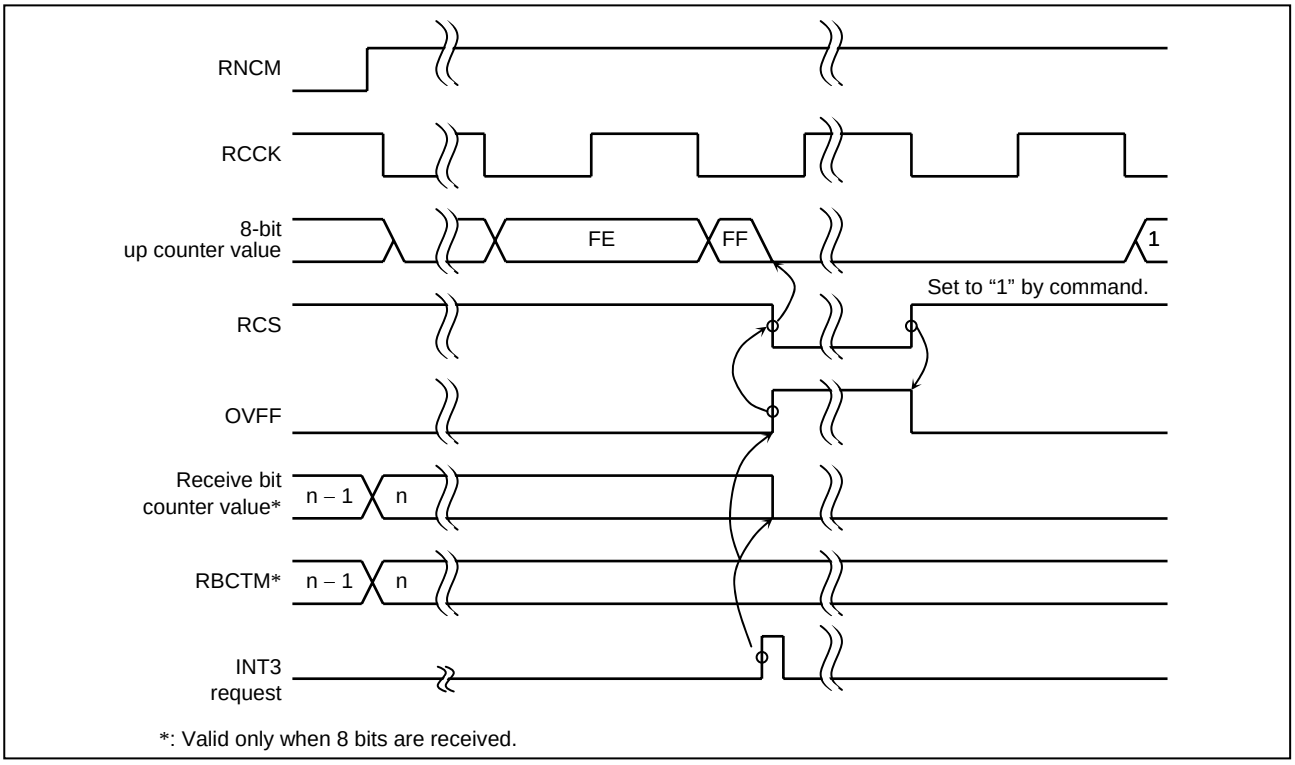



Figure 2.10.7 Overflow Interrupt Timing Chart

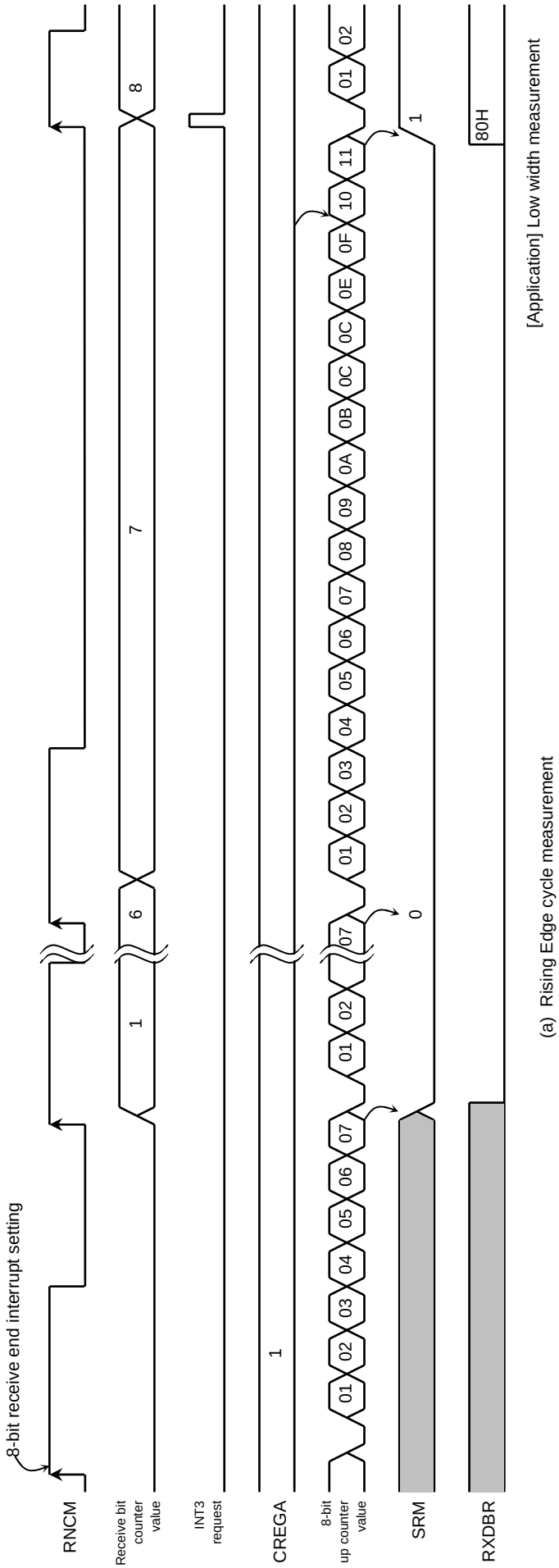



Figure 2.10.8 8-Bit Receive End Interrupt Timing Chart (PROLS = 0)

Table 2.10.3 Count Clock for Remote Control Preprocessor Circuit (at fc = 16 MHz)

| Count Clock (RCCK) | Resolution [μs] | Maximum Setting Time [ms] |
|---|---|---|
| 00 | 4 | 1.024 |
| 01 | 16 | 4.096 |
| 10 | 64 | 16.38 |
| 11 | 256 | 65.53 |

## 2.11 8-Bit AD Converter (ADC)

The TMP88CS38/CM38A/CP38A has a 8-bit successive approximation type AD converter.

### 2.11.1 Configuration

Figure 2.11.1 shows the circuit configuration of the AD converter.

The AD converter includes control registers ADCCRA and ADCCRB, conversion result registers ADCDR1 and ADCDR2, a DA converter, a sample hold circuit, a comparator, and sequential transducer circuit.

To use P5 and P6 as analog inputs, clear the output latch for P5 and P6 to "0". Also, clear the input/output control registers (P5CR1 and P6CR) to "0".

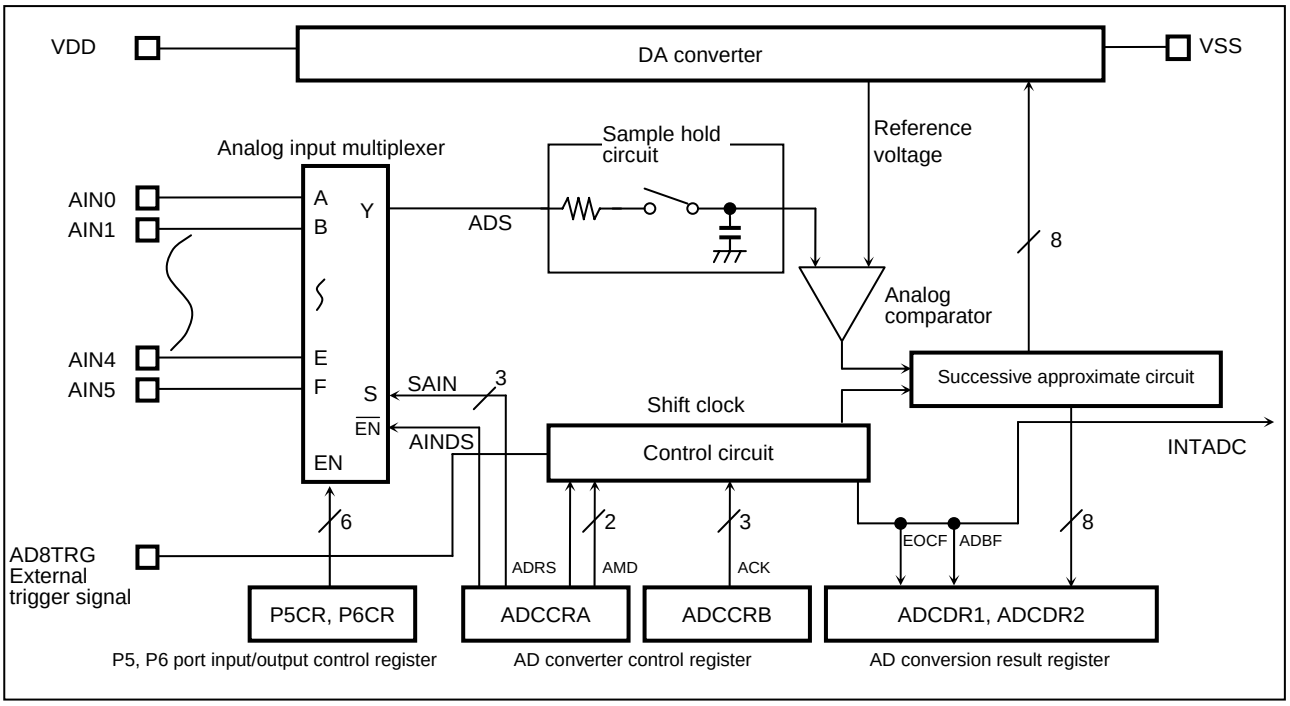



Figure 2.11.1 AD Converter (ADC)

### 2.11.2 Control Register

The following register are used foe AD converter.

- AD converter control register 1 (ADCCRA)
- AD converter control register 2 (ADCCRB)
- AD conversion result register (ADCDR1/ADCDR2)

(1) AD converter control register 1 (ADCCRA)

ADCCRA control AD conversion start, AD operation mode select, analog input control and analog input channel select.

(2) AD converter control register 2 (ADCCRB)

ADCCRB control AD conversion time select.

(3) AD conversion result register (ADCDR1)

AD conversion result is stored after end of conversion.

(4) AD conversion result register (ADCDR2)

For monitoring status of conversion.

Figure 2.11.2 and Figure 2.11.3 show AD converter control register.

AD Converter Control Register 1

| ADCCRA (0000EH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADRS | AMD | | AINDS | "0" | SAIN | | |

(Initial value: 0001 0000)

| | | | |
|---|---|---|---|
| ADRS | AD conversion start | The ADRS bit is automatically cleared after starting AD conversion. During AD conversion, setting ADRS to "1" initializes the ADRS bit and resets conversion.<br>0: –<br>1: AD conversion restart | R/W |
| AMD | AD Operation mode select | 00: STOP mode<br>01: Software start mode<br>00: Trigger start mode<br>11: reserved | |
| AINDS | Analog input control | 0: Analog input enable<br>1: Analog input disable | |
| SAIN | Analog input channel selection | 000: select AIN0<br>001: select AIN1<br>010: select AIN2<br>011: select AIN3<br>100: select AIN4<br>101: select AIN5<br>110: –<br>111: – | |

Note 1: Select analog input when AD converter stops.

Note 2: When the analog input is all use disabling, the AINDS should be set to "1".

Note 3: During conversion, do not perform output instruction to maintain a precision for all of the pins. And port near to analog input, do not input intense signaling of change.

Note 4: The ADRS is automatically cleared to "0" after starting conversion.

Note 5: Always set bit 3 in ADCCRA to "0".

Note 6: Do not set ADRS (Bit7 in ADCCRA) to "1" during AD conversion. Re-set it after confirming with EOCF (Bit5 in ADCDR2) that the conversion is completed or after generation an interrupt signal (INTADC) (by the interrupt processing routine or the like).

Note 7 In the trigger mode, the system does not accept the second and subsequent triggers after accepting the first trigger for starting AD conversion. To restart AD conversion by a trigger, set AMD (Bits 6 and 5 in ADCCRA) to "00" and then put the system in trigger start mode again (with AMD = "10").

Note 8: When the system enters STOP mode, AD converter control register 1 (ADCCRA) is initialized. Re-set this register after the system reenters NORMAL mode.

AD Converter Control Register 2

| ADCCRB (0000FH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | - | - | - | "0" | "1" | ACK | | "0" |

(Initial value: **0* 000*)

| | | ACK | Conversion time | DV1CK = 0 | | DV1CK = 1 | | |
|---|---|---|---|---|---|---|---|---|
| | | | | fc = 16 MHz | fc = 8 MHz | fc = 16 MHz | fc = 8 MHz | |
| ACK | AD conversion time select | 000 | Reserved | | | | | R/W |
| | | 001 | | | | | | |
| | | 010 | | | | | | |
| | | 011 | 156/fc [s] | – | 19.5 | – | 39 | |
| | | 100 | 312/fc [s] | 19.5 | 39.0 | 39 | 78 | |
| | | 101 | 624/fc [s] | 39.0 | 78.0 | 78 | 156 | |
| | | 110 | 1248/fc [s] | 78.0 | – | 156 | – | |
| | | 111 | Reserved | | | | | |

Note 1: Do not use setting except the above liset.

Note 2: Set conversion time by analog reference voltage ($V_{DD}$) as follows.

$V_{DD}$ = 4.5 to 5.5 V (15.6 μ or more)

Note 3: Always set bit0 and bit5 in ADCCRB to "0" and set bit4 in ADCCRB to "1".

Note 4: When a read instruction for ADCCRB, bit6 to 7 in ADCCRB read in as undefined data.

Note 5: fc: High-frequency clock [Hz]

Note 6: When the system enters STOP mode, AD converter control register 2 (ADCCRA) is initialized. Re-set this register after the system reenters NORMAL mode.

Figure 2.11.2 AD Converter Control Register

AD Conversion Result Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| ADCDR1 (00031H) | AD07 | AD06 | AD05 | AD04 | AD03 | AD02 | AD01 | AD00 | (Initial value: 0000 0000) |
| ADCDR2 (00032H) | – | – | EOCF | ADBF | – | – | – | – | (Initial value: **00 ****) |

| | | | |
|---|---|---|---|
| EOCF | AD conversion end flag | 0: Under conversion or before conversion<br>1: End of conversion | Read only |
| ADBF | AD conveersion busy flag | 0: During stop of AD conversion<br>1: During AD conversion | |

Note 1: The EOCF is cleared to “0” when reading the ADCDR1.
Therefore, the AD conversion result should be read to ADCDR1 more first than ADCDR2.

Note 2: ADBR is set to “1” by starting AD conversion and cleared to “0” by end of AD conversion. Additionally, ADBF is cleared to “0” by setting AMD = “00” in ADCCR2 or entering to the STOP mode.

Figure 2.11.3 AD Converter Result Register

### 2.11.3 AD Converter Operation

The high side of an analog reference voltage is applied to VDD, and the low side is applied to VSS pin. Dividing a reference voltage between VDD and VSS to the voltage corresponding to a bit by a rudder resistance and comparing it with the analog input voltage converts the AD.

Table 2.11.1 AD Converter Operation Mode

| Mode | Function |
|---|---|
| AD converter disable mode | AD converter stop mode. This mode is always used to change modes. |
| Software start mode | Single AD conversion of 1 channel which specifies input. |
| Trigger start mode | Single AD conversion of 1 channel which specifies input (AD8TRG) from Key-on wakeup circuit as a trigger. |

### 2.11.4 Interrupt

Interrupt request signal occur at the timing when the EOCF bit is set to “1”.

### 2.11.5 AD Converter Operation Modes

When the MCU places in the STOP mode during the AD conversion, the conversion is stopped and the ADCDR2 content becomes indefinite. After returning from the STOP mode, the EOCF and INTADC does not occur. Therefore, the AD conversion must be restarted after returning from the STOP mode.

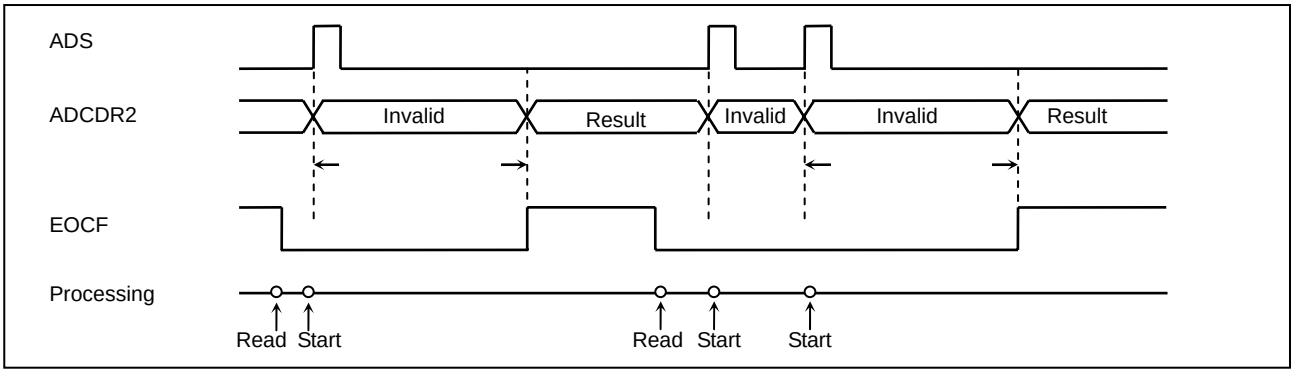



Figure 2.11.4 AD Conversion Timing chart

(1) AD conversion in STOP mode

When the AD converter stop mode is specified during AD conversion, the AD conversion is stopped immediately. The AD conversion is not implemented, so the undefined value is not written to the AD conversion result register. The AD conversion start commands which occur is the AD converter stop mode are ignored.

This mode is automatically selected by reset.

This mode is used to change the AD converter operation mode.

(2) Single mode

When the AMD (Bit6, 5 to in ADCCRA) set to “01”, the AD conversion signal mode

This mode does AD conversion of single channel, and conversion result is stored in ADCDR1. The EOCF (Bit5 in ADCDR2) is set to “1” at end of one conversion, and an intcrrupt request signal occurs. The EOCF is cleared to “0” by reading the AD conversion registers.

But when the AD conversion is restarted before the ADCDR is read, the EOCF is cleared to “0” and the last AD conversion result is maintained till next conversion end.

Do not set ADRS (Bit7 in ADCCRA) during AD conversion. Again set it after confirming with EOCF (Bit5 in ADCDR2) that the conversion is completed or after generating an interrupt signal (INTADC) (by the interrupt processing routine or the like).

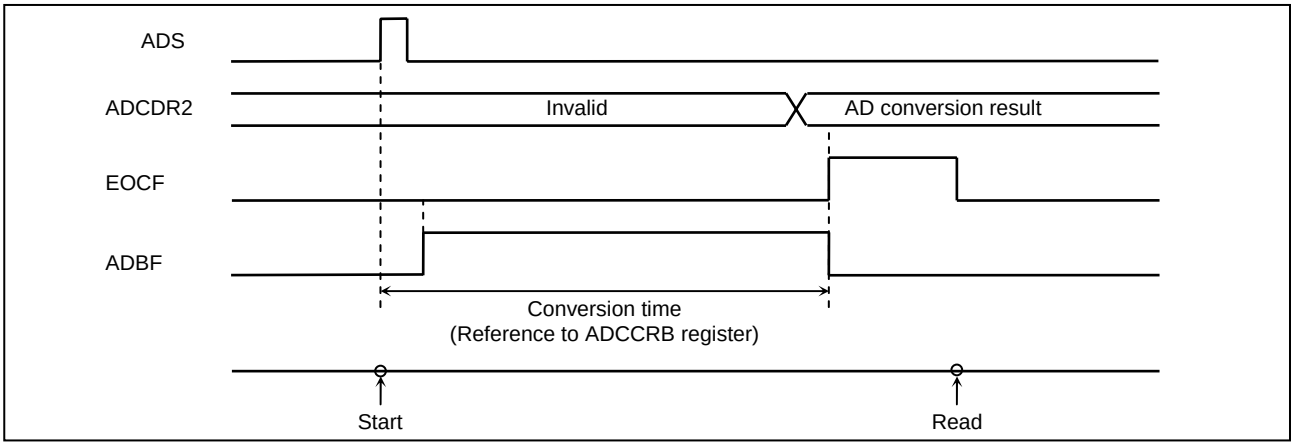



Figure 2.11.5 Single Mode

Example: The AD conversion starts after 19.5 μs (at fc = 16 MHz) and AIN4 pin are selected as the conversion time and the analog input channel. Confirming the EOCF, the converted value is read out, and the 8 bits data is stored to address 009EH in RAM. The operation mode is a signal mode.

```
        ;AIN SELECT
        LD      (P5), 00000000B
        LD      (P5CR1), 00000000B
        LD      (P6), 00000000B
        LD      (P6CR), 00000000B
        LD      (ADCCRA), 00100100B     ;   Selects AIN4, selects the software start mode
        LD      (ADCCRB), 00011000B     ;   Selects the conversion time and the operation mode.

        ; AD CONVERT START
        SET     (ADCCRA). 7             ;   ADRS = 1
SLOOP:  TEST    (ADCCR2). 5             ;   EOCF = 1 ?
        JRS     T, SLOOP
        ; RESULT DATA READ
        LD      (9EH), (ADCDR1)
```

(3) Trigger start mode

The AD conversion of a specified single channel is executed when input (AD8TRG) from Key-on wakeup circuit is set as trigger, the conversion result is stored in the ADCDR1.

The EOCF (Bit5 in ADCDR2) is set to "1" at end of one conversion, and an interrupt request signal occurs.

It needs to be set the STOP mode by bit5 to 6 in ADCCRA before the AD conversion is executed again.

### 2.11.6 Analog Input Voltage and AD Conversion Result

The analog input voltage is corresponded to the 8-bit digital value converted by the AD as shown in Figure 2.11.6.

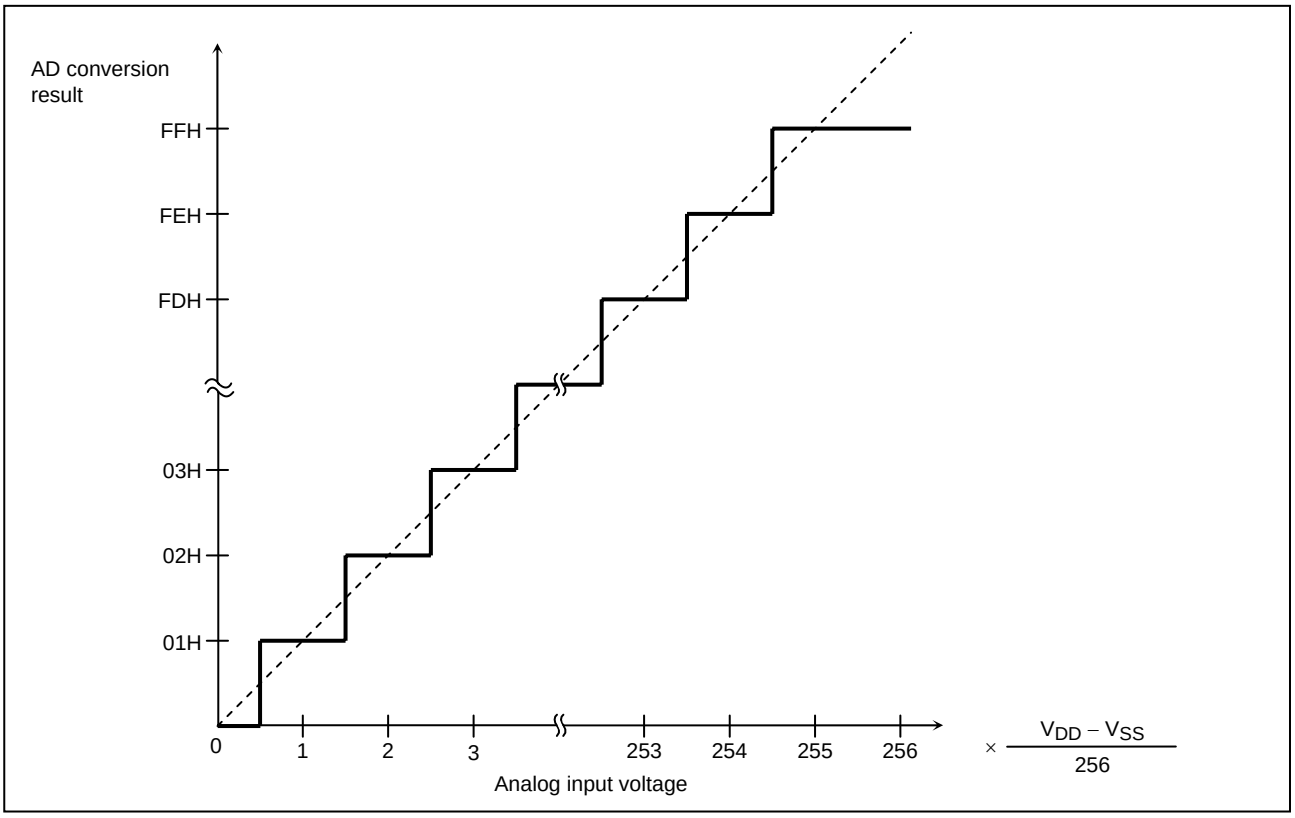



Figure 2.11.6 Analog Input Voltage and AD Conversion Result (typ.)

### 2.11.7 STOP Modes during AD Conversion

When standby mode (STOP mode) is entered forcibly during AD conversion, the AD convert operation is suspended and the AD converter is initialized. (ADCCRA and ADCCRB are initialized to initial value.) Also, the conversion result is indeterminate. (Conversion results up to the previous operation are cleared, so be sure to read the conversion results before entering standby mode.) When restored from standby mode, AD conversion is not automatically restarted, so it is necessary to restart AD conversion after setting ADCCRA and ADCCRB. Note that since the analog reference voltage is automatically disconnected, there is no possibility of current flowing into the analog reference voltage.

### 2.11.8 Notice of AD Converter

(1) Analog input voltage range

Voltage range of analog input (AIN0 to AIN5) must be forced from VSS to VDD. If input voltage of which out of range is forced to analog input pin, AD conversion result to unknown. Also, this cause other analog input pin unstable.

(2) I/O port with analog input

Analog input pins (AIN0 to AIN5) are also I/O port. During AD conversion using any analog input pin, don't operate other I/O port with analog input. Because, AD accuracy would be worse. Also, other electrically swinging port without analog input may cause noise to near analog input pin.

(3) Reduce to noise

Figure 2.11.7 is shown as internal equivalent circuit of analog input pin.

Increasing output impedance of analog input supply, cause noise or other non-good condition.

Therefore, output impedance of analog input supply must be less than 5 kΩ.

And we recommend to connect capacitance to analog input pin.

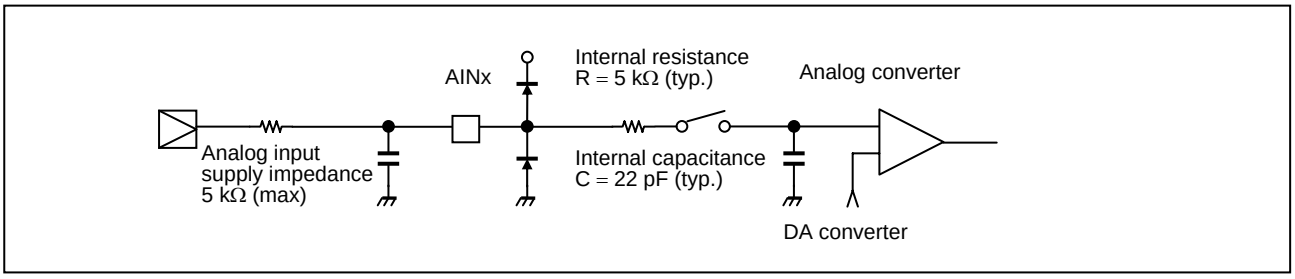



Figure 2.11.7 Analog Input Equivalent Circuit and Analog Input Pin

## 2.12 Key-on Wakeup

In this MCU the IDLE mode is also released by low active port inputs. The low input voltage is regulated higher than the other normal ports. Therefore the ports can be enabled by analog input level.

### 2.12.1 Configuration

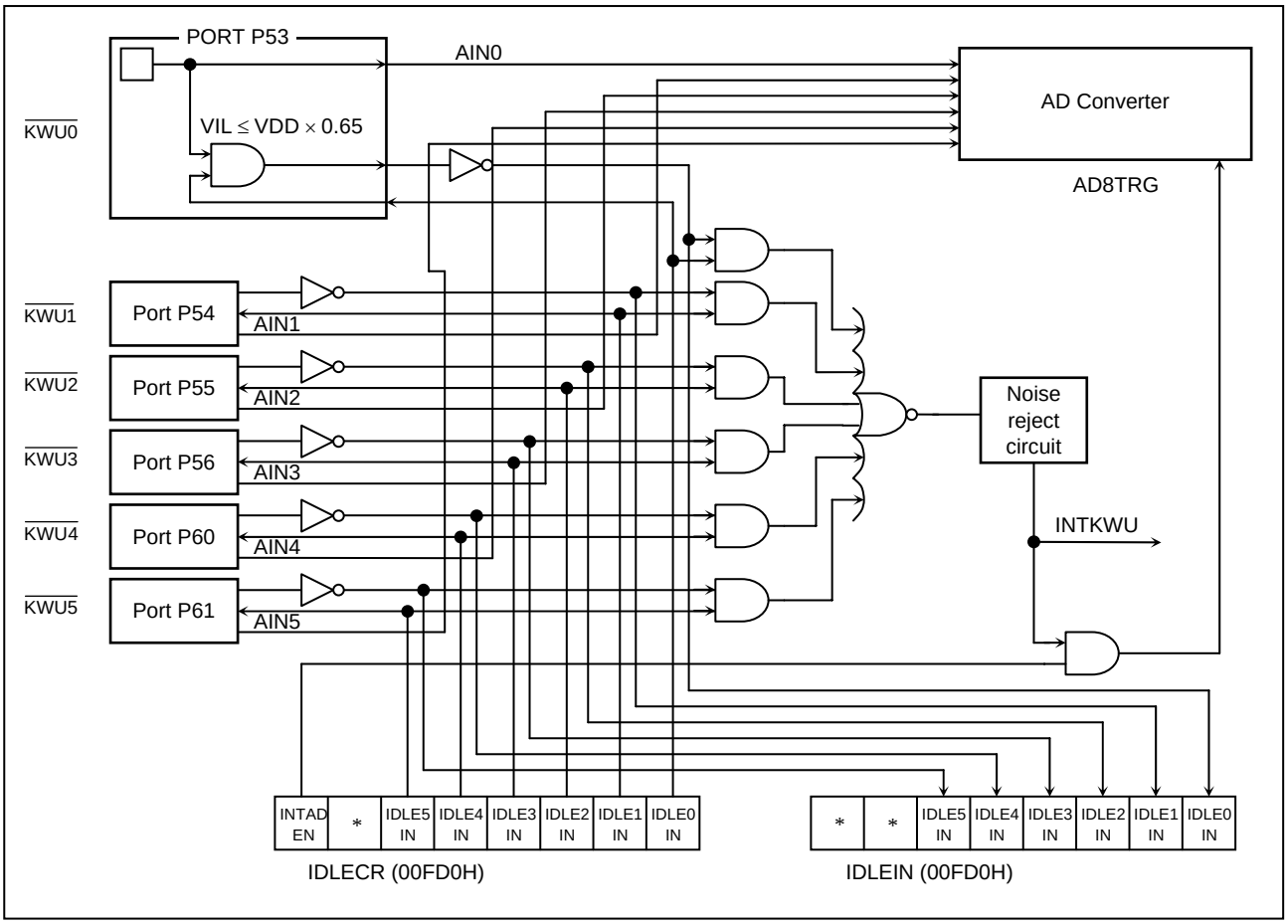



Figure 2.12.1 Key-on Wakeup Control Circuit

### 2.12.2 Control

P53 to P56 and P60, P61 ports can be controlled by IDLE control register (IDLECR).

It can be configured as enable/disable in one-bit unit. When those pins are used by IDLE mode release, those pins must be set input mode (P5CR1, P5, P6CR, P6, ADCCRA).

IDLE mode is controlled by system control register 2 (SYSCR2) and maskable interrupts. After the individual enable flag (EF5) is set to "1", the IDLE mode must starts. When enabled port input generates INTKWU interrupt, the IDLE mode is released. Low level input voltage in those ports is regulated to less than VDD × 0.65 (V).

IDLE port monitorring register (IDLEIN) can be used to check state of ports.

INTADEN can enable to generate AD8TRG, which is used as trigger of AD converter trigger start mode.

Noise reject circuit eliminate noise, which is less than 24 μs period.

IDLE Control Register

| IDLECR (00FD0H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INTAD EN | * | IDLE5 EN | IDLE4 EN | IDLE3 EN | IDLE2 EN | IDLE1 EN | IDLE0 EN |

(Initial value: 0*00 0000)

| | | | |
|---|---|---|---|
| INTADEN | Generation of $\overline{\text{AD8TRG}}$ | 0: Disable<br>1: Enable | Write only |
| IDLE5EN | Release IDLE mode by $\overline{\text{KWU5}}$ | 0: Disable<br>1: Enable | |
| IDLE4EN | Release IDLE mode by $\overline{\text{KWU4}}$ | 0: Disable<br>1: Enable | |
| IDLE3EN | Release IDLE mode by $\overline{\text{KWU3}}$ | 0: Disable<br>1: Enable | |
| IDLE2EN | Release IDLE mode by $\overline{\text{KWU2}}$ | 0: Disable<br>1: Enable | |
| IDLE1EN | Release IDLE mode by $\overline{\text{KWU1}}$ | 0: Disable<br>1: Enable | |
| IDLE0EN | Release IDLE mode by $\overline{\text{KWU0}}$ | 0: Disable<br>1: Enable | |

*: Don't care

IDLE Port Monitoring Register

| IDLEIN (00FD0H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | * | * | IDLE5 IN | IDLE4 IN | IDLE3 IN | IDLE2 IN | IDLE1 IN | IDLE0 IN |

(Initial value: **00 0000)

| | | | |
|---|---|---|---|
| IDLE5IN | Input level of $\overline{\text{KWU5}}$ | 0: "0" detect<br>1: "1" detect | Read only |
| IDLE4IN | Input level of $\overline{\text{KWU4}}$ | 0: "0" detect<br>1: "1" detect | |
| IDLE3IN | Input level of $\overline{\text{KWU3}}$ | 0: "0" detect<br>1: "1" detect | |
| IDLE2IN | Input level of $\overline{\text{KWU2}}$ | 0: "0" detect<br>1: "1" detect | |
| IDLE1IN | Input level of $\overline{\text{KWU1}}$ | 0: "0" detect<br>1: "1" detect | |
| IDLE0IN | Input level of $\overline{\text{KWU0}}$ | 0: "0" detect<br>1: "1" detect | |

*: Don't care

Figure 2.12.2 Key-on Wakeup Control Register

## 2.13 Pulse Width Modulation Circuit Output

The TMP88CS38/CM38A/CP38A has four 12-bit resolution PWM output channels including two 14-bit resolution selectable and six 7-bit resolution PWM output channels.

DA 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 ($\overline{PWM0}$) to P47 ($\overline{PWM7}$), P50 ($\overline{PWM8}$), P51 ($\overline{PWM9}$). PWM output is negative logic. When these ports are used PWM outputs, the corresponding bits of P4, P5 output latches and input/output control latches should be set to "1".

In STOP mode, PWM output pin keeps high-level. When operation mode is changed from STOP mode to NORMAL mode, PWM control register (PWMCR1A, PWMCR2A, PWMCR1B, PWMCR2B) are initialized.

### 2.13.1 Configuration

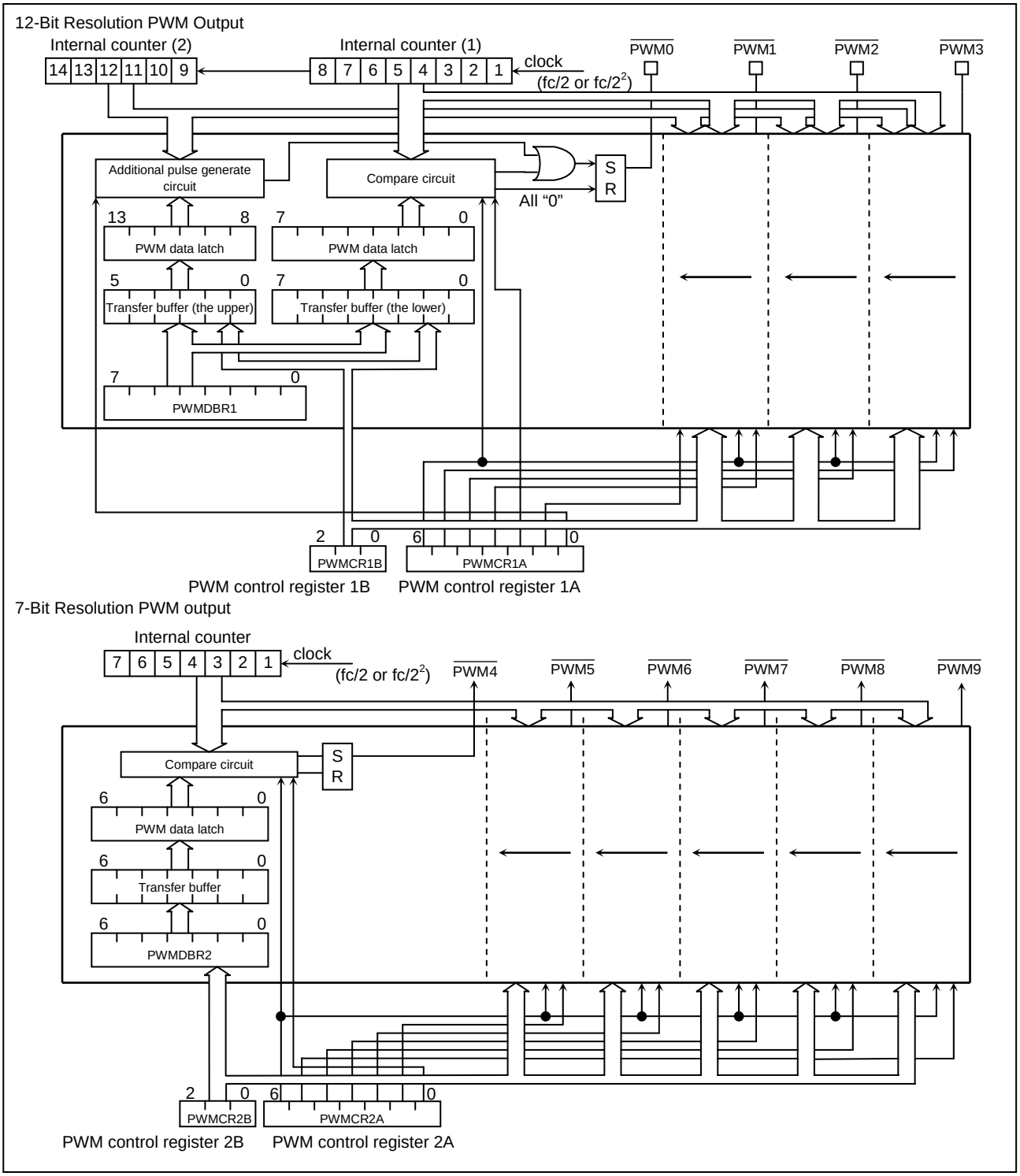



Figure 2.13.1 PWM Output Circuit

### 2.13.2 PWM Output Wave Form

(1) $\overline{PWM0}$ to $\overline{PWM1}$ Outputs

$\overline{PWM0}$ and $\overline{PWM1}$ output can be selected 12-bit or 14-bit resolution PWM outputs.

1. 12-bit resolution PWM output

When these are used as 12-bits PWM output, one period is $T_M = 2^{13}/fc$ [s] (When DV1CK = 0) and $T_M = 2^{14}/fc$ [s] (When DV1CK = 1) and sub period is $T_S = T_M/16$.

The lower 8 bits of the PWM data latch controls the low level pulse width with a cycle of $T_S$. The lower 8 bits of the PWM data latch is n (n = 1 to 255), the low level pulse width with a cycle becomes $n \times t_0$ [s] ($t_0 = 2/fc$ [s] when DV1CK = 0, $t_0 = 4/fc$ [s] when DV1CK = 1).

The upper 4 bits of the PWM data latch controls a position to output the additional pulses. When the upper 4 bits of the PWM data latch is m, the additional pulses are generated in each of m periods out of 16 periods contained in a $T_M$ period.

The relationship between the 4-bit data and the position of $T_S$ period where the additional pulses are generated is shown in Table 2.13.1.

Table 2.13.1 The Addition Pulse (12-bit mode)

| | Bit Position of the Lower 4 Bits of PWMDRxH | | | | Relative position of $T_S$ in $T_M$ period where the additional pulse is generated. (Number of $T_{S\ (I)}$ is listed) |
|---|---|---|---|---|---|
| | Bit11 | Bit10 | Bit9 | Bit8 | |
| a) | 0 | 0 | 0 | 0 | No additional pulse |
| b) | 0 | 0 | 0 | 1 | 8 |
| c) | 0 | 0 | 1 | 0 | 4, 12 |
| d) | 0 | 1 | 0 | 0 | 2, 6, 10, 14 |
| e) | 1 | 0 | 0 | 0 | 1, 3, 5, 7, 9, 11, 13, 15 |

Note 1: The bit positions of a) to e) can be combined.

Note 2: If the low order eight bits for the PWM data latch are set to “FFH”, be sure to set the high order four bits for this latch to “00H”.

2. 14-bit resolution PWM output

When these are used as 14-bit PWM output, one period is $T_M = 2^{15}/fc$ [s] (When DV1CK = 0) and $TM = 2^{16}/fc$ [s] (When DV1CK = 1) and sub period is $T_S = T_M/64$.

The lower 8 bits of the PWM data latch controls the low level pulse width with a cycle of $T_S$. The lower 8 bits of the PWM data latch is n (n = 1 to 255), the low level pulse width with a cycle becomes $n \times t_0$ [s] ($t_0 = 2/fc$ [s] when DV1CK = 0, $t_0 = 4/fc$ [s] when DV1CK = 1).

The upper 6 bits of the PWM data latch controls a position to output the additional pulses. When the upper 6 bits of the PWM data latch is m, the additional pulses are generated in each of m periods out of 64 periods contained in a $T_M$ period.

The relationship between the 6-bit data and the position of $T_S$ period where the additional pulses are generated is shown in Table 2.13.2.

Table 2.13.2 The Addition Pulse (14 bit mode)

| | Bit Position of the Lower 6 Bits of PWMDRxH | | | | | | Relative position of $T_S$ in $T_M$ period where the additional pulse is generated. (Number of $T_{S\ (I)}$ is listed) |
|---|---|---|---|---|---|---|---|
| | Bit13 | Bit12 | Bit11 | Bit10 | Bit9 | Bit8 | |
| a) | 0 | 0 | 0 | 0 | 0 | 0 | No additional pulse |
| b) | 0 | 0 | 0 | 0 | 0 | 1 | 32 |
| c) | 0 | 0 | 0 | 0 | 1 | 0 | 16, 48 |
| d) | 0 | 0 | 0 | 1 | 0 | 0 | 8, 24, 40, 56 |
| e) | 0 | 0 | 1 | 0 | 0 | 0 | 4, 12, 20, 28, 36, 44, 52, 60 |
| f) | 0 | 1 | 0 | 0 | 0 | 0 | 2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62 |
| g) | 1 | 0 | 0 | 0 | 0 | 0 | 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63 |

Note 1: The bit positions of a) to g) can be combined.

Note 2: If the low order eight bits for the PWM data latch are set to "FFH", be sure to set the high order 6 bits for this latch to "00H".

(2) $\overline{PWM2}$ to $\overline{PWM3}$ Outputs

$\overline{PWM2}$ and $\overline{PWM3}$ output are 12-bit resolution PWM outputs.

One period is $T_M = 2^{13}/fc$ [s] (When DV1CK = 0) and $TM = 2^{14}/fc$ [s] (When DV1CK = 1) and sub period is $T_S = T_M/16$.

The lower 8 bits of the PWM data latch controls the low level pulse width with a cycle of $T_S$. The lower 8 bits of the PWM data latch is n (n = 1 to 255), the low level pulse width with a cycle becomes $n \times t_0$ [s] ($t_0 = 2/fc$ [s] when DV1CK = 0, $t_0 = 4/fc$ [s] when DV1CK = 1).

The upper 4 bits of the PWM data latch controls a position to output the additional pulses. When the upper 4 bits of the PWM data latch is m, the additional pulses are generated in each of m periods out of 16 periods contained in a $T_M$ period.

The relationship between the 4-bit data and the position of $T_S$ period where the additional pulses are generated is shown in Table 2.13.1.

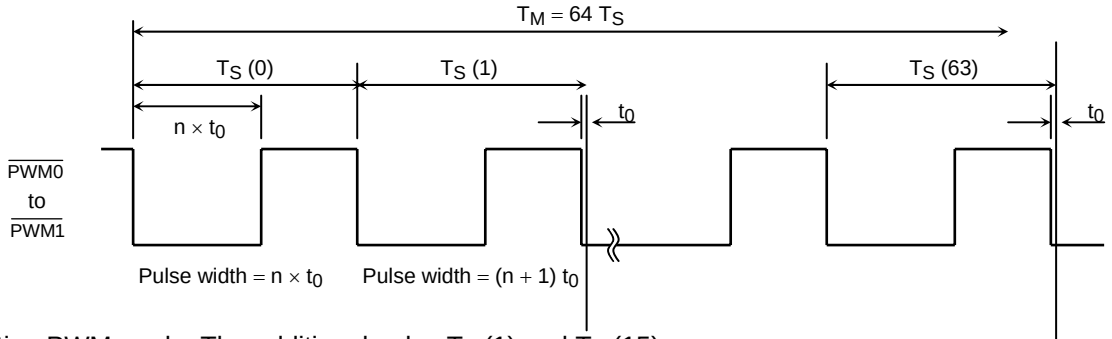
(3) $\overline{PWM4}$ to $\overline{PMW9}$ Outputs

These are 7-bit resolution PWM outputs.

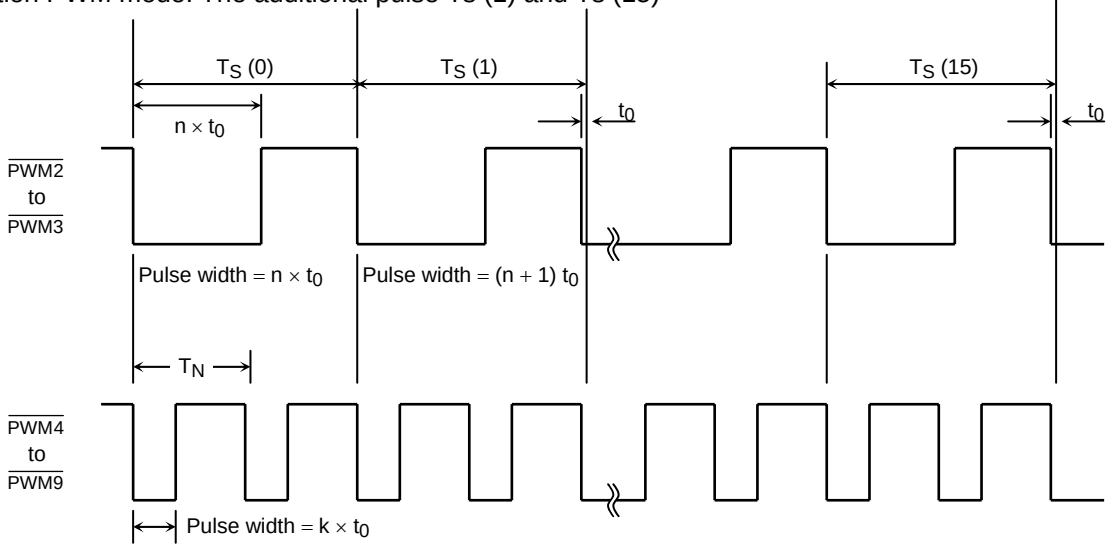
One period is $T_N = 2^8/fc$ [s] (When DV1CK = 0) and $T_N = 2^9/fc$ [s] (When DV1CK = 1).

The 7 bits of the PWM data latch controls the low level pulse width with a cycle of $T_N$. The lower 7 bits of the PWM data latch is k (k = 1 to 127), the low level pulse width with a cycle becomes $k \times t_0$ [s] ($t_0 = 2/fc$ [s] when DV1CK = 0, $t_0 = 4/fc$ [s] when DV1CK = 1).

14-bit resolution PWM mode: The additional pulse Ts (1) and Ts (63)




12-bit resolution PWM mode: The additional pulse Ts (1) and Ts (15)




Note 1: If the pulse width is set to “00H”, PWM will note operate. Its output will remain high.

Note 2: If the pulse width is set to “FFH”, settings for additional pulses cannot be made. Be sure to set the pulse width to “00H”.

Figure 2.13.2 PWM Output Waveform

### 2.13.3 Control

PWM output is controlled by PWM control register (PWMCR1A, PWMCR1B, PWMCR2A, PWMCR2B) and PWM data buffer register (PWMDBR1, PWMDBR2).

PWM Control Register 1A

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PWMCR1A (00028H) | – | ABORT1 | START3 | START2 | START1 | START0 | RESOLUTION 1 | RESOLUTION 0 |

(Initial value: *000 0000)

| | | | |
|---|---|---|---|
| ABORT1 | Abort PWM operation of channel 3 to 0 | 0: Operation<br>1: PWM abort (PWM outputs are fixed to a high level.) | Write only |
| START3 | Start channel 3 | 0: Stop $\overline{PWM3}$<br>1: Start $\overline{PWM3}$ | |
| START2 | Start channel 2 | 0: Stop $\overline{PWM2}$<br>1: Start $\overline{PWM2}$ | |
| START1 | Start channel 1 | 0: Stop $\overline{PWM1}$<br>1: Start $\overline{PWM1}$ | |
| START0 | Start channel 0 | 0: Stop $\overline{PWM0}$<br>1: Start $\overline{PWM0}$ | |
| RESOLUTION1 | Select channel 1 resolution | 0: 14-bit resolution<br>1: 12-bit resolution | |
| RESOLUTION2 | Select channel 0 resolution | 0: 14-bit resolution<br>1: 12-bit resolution | |

Note 1: *: Don't care

Note 2 After set the ABORT1 to "1", the ABORT1 is cleared to "0" automatically.

Note 3: PWMCR1A is write-only register and cannot be used with any of the read-modify-write instructions such as SET, CLR, etc.

PWM Control Register 1B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PWMCR1B (00029H) | | | | | | PWMCHS1 | | PWMHL |

(Initial value: **** *000)

| | | | |
|---|---|---|---|
| PWMCHS1 | Select the PWM data latch of 12-bit PWM channel | 00: Channel 0<br>01: Channel 1<br>10: Channel 2<br>11: Channel 3 | Write only |
| PWMHL | Select upper or lower data transfer buffer (PWMDBR1) | 0: Lower 8 bits<br>1: Upper 4 bits or 6 bits | |

Note 1: *: Don't care

Note 2: PWMCR1B is write-only register and cannot be used with any of the read-modify-write instructions such as SET, CLR, etc.

PWM Data Buffer Register 1

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PWMDBR1 (0002AH) | | | | | | | | |

Write only
(Initial value: 0000 0000)

Note 1: PWMDBR1 is write-only register and cannot be used with any of the read-modify-write instructions such as SET, CLR, etc.

Note 2: When operation mode is changed from STOP mode to NORMAL mode, PWMCR1A, PWMCR1B are initialized.

Figure 2.13.3 PWM Control Register 1A/1B and PWM Data Buffer Register 1

PWM Control Register 2A

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| PWMCR2A (00FF5H) | – | ABORT2 | START9 | START8 | START7 | START6 | START5 | START4 | (Initial value: *000 0000) |

| | | | |
|---|---|---|---|
| ABORT2 | Abort PWM operation of channel 9 to 4 | 0: Operation<br>1: PWM abort | Write only |
| START9 | Start channel 9 | 0: Stop $\overline{PWM9}$<br>1: Start $\overline{PWM9}$ | |
| START8 | Start channel 8 | 0: Stop $\overline{PWM8}$<br>1: Start $\overline{PWM8}$ | |
| START7 | Start channel 7 | 0: Stop $\overline{PWM7}$<br>1: Start $\overline{PWM7}$ | |
| START6 | Start channel 6 | 0: Stop $\overline{PWM6}$<br>1: Start $\overline{PWM6}$ | |
| START5 | Start channel 5 | 0: Stop $\overline{PWM5}$<br>1: Start $\overline{PWM5}$ | |
| START4 | Start channel 4 | 0: Stop $\overline{PWM4}$<br>1: Start $\overline{PWM4}$ | |

Note 1: *: Don't care

Note 2 After set the ABORT2 to "1", the ABORT2 is cleared to "0" automatically.

Note 3: PWMCR2A is write-only register and cannot be used with any of the read-modify-write instructions such as SET, CLR, etc.

PWM Control Register 2B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| PWMCR2B (00FF6H) | | | | | | PWMCHS2 | | | (Initial value: **** *000) |

| | | | |
|---|---|---|---|
| PWMCHS2 | Select the PWM data latch of 7-bit PWM channel | 000: Channel 4<br>001: Channel 5<br>010: Channel 6<br>011: Channel 7<br>100: Channel 8<br>101: Channel 9<br>110: Reserved<br>111: Reserved | Write only |

Note 1: *: Don't care

Note 2: PWMCR2B is write-only register and cannot be used with any of the read-modify-write instructions such as SET, CLR, etc.

PWM Data Buffer Register 2

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| PWMDBR2 (00FF7H) | | | | | | | | | Write only<br>(Initial value: *000 0000) |

Note 1: *: Don't care

Note 2: PWMDBR2 is write-only register and cannot be used with any of the read-modify-write instructions such as SET, CLR, etc.

Note 3: When operation mode is changed from STOP mode to NORMAL mode, PWMCR2A, PWMCR2B are initialized.

Figure 2.13.4 PWM Control Register 2A/2B and PWM Data Buffer Register 2

Binary Counter Control Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| CGCR (00030H) | "0" | "0" | DV1CK | "0" | "0" | "0" | "0" | "0" | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| DV1CK | Select of input clock to 1st divider | 0: fc/4<br>1: fc/8 | R/W |

Note 1: *: Don't care

Note 2: The all bits except DV1CK are cleared to "0".

Figure 2.13.5 DIVIDER Control Register

(1) Internal counter

The internal counter of PWM outputs is a free running counter. The all bits of counter are set to "1" and are not counted up at one of the following conditions.

1. During reset
2. The operation mode is changed to STOP mode.
3. Setting ABORTx (x: 1, 2) to "1".
4. The START3 to 0 are "0" in 12-bit PWM outputs. The START9 to 4 are "0" in 7-bit PWM outputs.
5. The lower 8-bit of PWM data latch in 12-bit PWM outputs is "00H". The PWM data latch in 7-bit PWM outputs is "00H".

(2) Outputs control and programming of PWM data

The PWM outputs are fixed to a high-level immediately when the ABORTx (x: 1, 2) is set to "1". The PWM outputs starts the operation when the STARTx (x: 0 to 9) is set to "1".

The data from the transfer buffer to a PWM data latch is transferred when the all bits of internal counter are set to "1". Therefore, the data is transferred to a PWM data latch immediately when the internal counter is initialized. And the data is transferred to a PWM data latch at the beginning of the next cycle when all bits of the internal counter are not set to "1".

The sequence of writing the output data to PWM data latches is shown as follows;

1. $\overline{\text{PWM0}}$ to $\overline{\text{PWM1}}$
   a) Write the channel number of PWM data latch to PWMCHS1 (Bit2 and 1 in PWMCR1B) and clear PWMHL (Bit0 in PWMCR1B) to "0".
   b) Write the lower 8-bit PWM output data to PWMDBR1.
   c) Write the channel number of PWM data latch to PWMCHS1 and set PWMHL to "1".
   d) Write the upper 4-bit or 6-bit PWM output data to PWMDBR1.
   e) Select the resolution of PWM output to RESOLUTIONx (x: 0, 1) (Bit0 and 1 in PWMCR1A) and set STARTx (x: 0, 1) (Bit2 and 3 in PWMCR1B) to "1".

Note: PWM output data must be write to PWMDBR1 in the order of the lower 8-bit PWM output data, the upper 4-bit (or 6-bit) PWM output data. If the upper 4-bit (or 6-bit) PWM output data is write to PWMDBR1, the lower 8-bit PWM output data is not changed (except when lower 8-bit PWM output data is "00H").

2. $\overline{PWM2}$ to $\overline{PWM3}$

a) Write the channel number of PWM data latch to PWMCHS1 and clear PWMHL to "0".

b) Write the lower 8-bit PWM output data to PWMDBR1.

c) Write the channel number of PWM data latch to PWMCHS1 and set PWMHL to "1".

d) Write the upper 4-bit PWM output data to PWMDBR1.

e) Set STARTx (x: 2, 3) to "1".

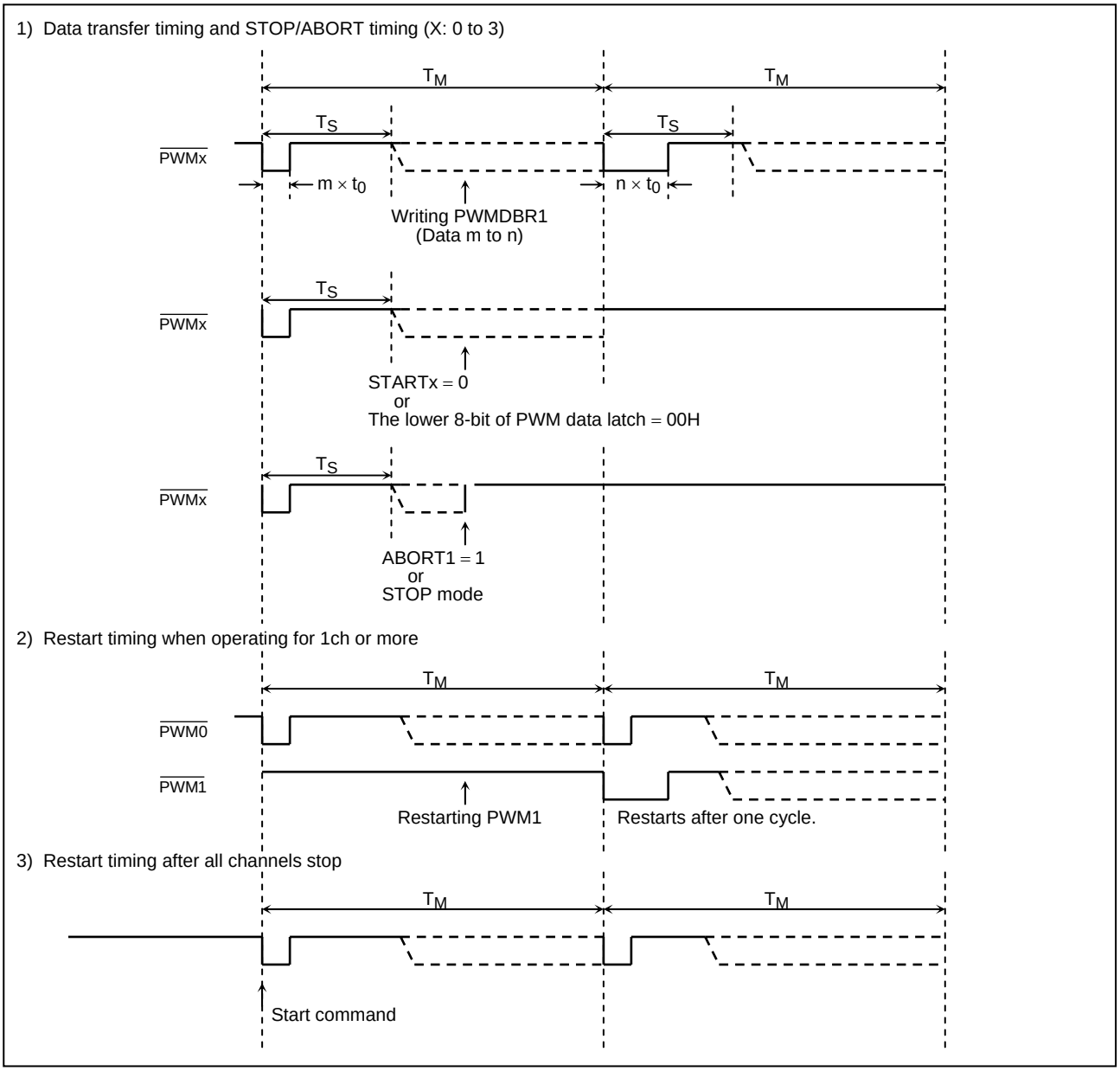



Figure 2.13.6 Waveform of $\overline{PWM0}$ to $\overline{PWM3}$

Note: PWM output data must be write to PWMDBR1 in the order of the lower 8-bit PWM output data, the upper 4-bit (or 6-bit) PWM output data. If the upper 4-bit (or 6-bit) PWM output data is write to PWMDBR1, the lower 8-bit PWM output data is not changed (except when lower 8-bit PWM output data is "00H").

3. $\overline{PWM4}$ to $\overline{PWM9}$
   a) Write the channel number of PWM data latch to PWMCHS2.
   b) Write the lower 7-bit PWM output data to PWMDBR2.
   c) Set STARTx (x: 4 to 9) to "1".

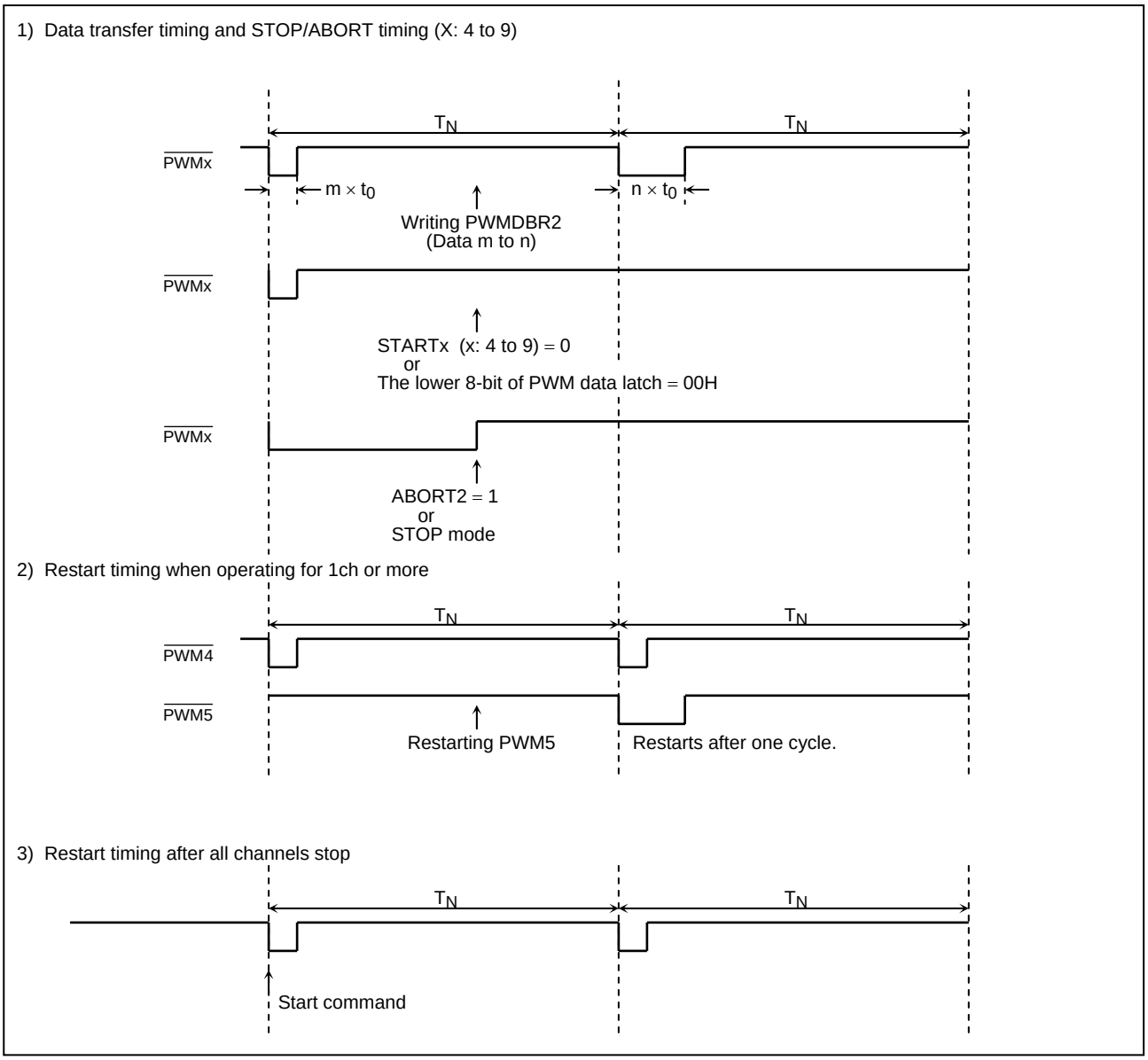



Figure 2.13.7 Waveform of $\overline{PWM4}$ to $\overline{PWM9}$

Example: at fc = 16 MHz, DV1CK = 0

$\overline{PWM0}$ pin outputs a 14-bit resolution PWM wave form with a low level of 32 μs width and no additional pulse.
$\overline{PWM1}$ pin outputs a 12-bit resolution PWM wave form with a low level of 16 μs width and no additional pulse.
$\overline{PWM4}$ pin outputs a PWM wave form with a low level of 8 μs width.

```
LD   (CGCR), 00H          ;  DV1CK = 0

LD   (PWMCR1B), 00H       ;  Select the lower 8 bits of PWM0 output data latch
LD   (PWMDBR1), 80H       ;  32 μs ÷ 4/fc = 80H
LD   (PWMCR1B), 01H       ;  Select the upper 6 bits of PWM0 output data latch
LD   (PWMDBR1), 00H       ;  No additional pulse = 00H
LD   (PWMCR1B), 02H       ;  Select the lower 8 bits of PWM0 output data latch
LD   (PWMDBR1), 40H       ;  16 μs ÷ 4/fc = 40H
LD   (PWMCR1B), 03H       ;  Select the upper 4 bits of PWM0 output data latch
LD   (PWMDBR1), 01H       ;  Additional pulse (Ts(8)) = 01H
LD   (PWMCR1A), 0DH       ;  Start PWM0 and PWM1,
                             PWM0 : 14-bit resolution, PWM1 : 12-bit resolution

LD   (PWMCR2B), 00H       ;  Select PWM4 output data latch
LD   (PWMDBR2), 20H       ;  8 μs ÷ 2/fc = 20H
LD   (PWMCR2A), 01H       ;  Start PWM4
```

## 2.14 Test Video Signal Output for Adjusting TV Screen

The TMP88CS38/CM38A/CP38A has a built-in video signal output circuit to output necessary signal for TV screen adjustment.

| | | | |
|---|---|---|---|
| Picture pattern | : | Total eight types, monochromatic inversion possible | |
| Output format | : | Three states (H, L, High-Z) output | |
| | | Comp.sync duration time | L output |
| | | Black level/pedestal duration time | High-Z output |
| | | White level duration time | H output |

### 2.14.1 Configuration

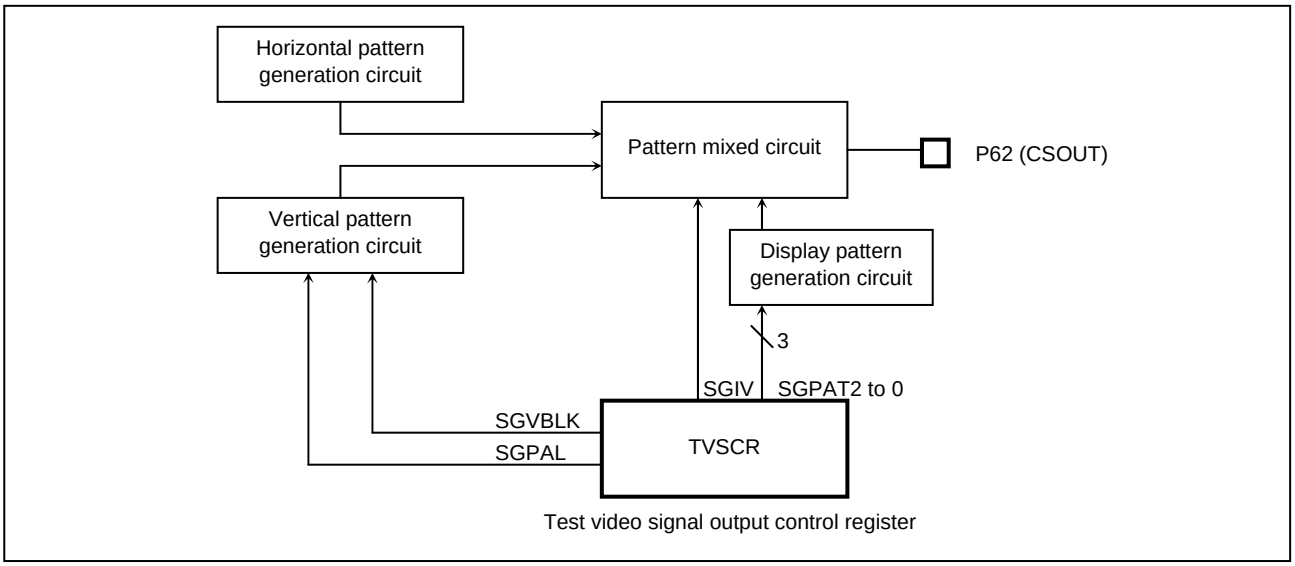



Figure 2.14.1 Test Video Signal Output Circuit

### 2.14.2 Control

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

| TVSCR (00FE6H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SGEN | SGVBLK | SGPAL | SGIV | SGCHS | “0” | SGPAT | “0” |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| SGEN | SG function selection | 0: Disable<br>1: Enable | Write only |
| SGVBLK | Picuture signal for VBLK duration time | 0: Output<br>1: No output | |
| SGPAL | PAL/NTSC selection | 0: NTSC<br>1: PAL | |
| SGIV | Pattern monochromatic inversion | 0: No inversion<br>1: Inversion | |
| SGCHS | OSD synchronous signal selection | 0: Port<br>1: Pseudo signal circuit | |
| SGPAT | Display pattern | 000: Black on the whole screen<br>001: White on the whole screen<br>010: Cross hatch<br>011: Cross dot pattern<br>100: Cross bar<br>101: White on the upper side/black on the lower side<br>110: H signal pattern<br>111: H resolution pattern | |

Note 1: Test video signal output function does work correctly when fc is not 16 MHz.

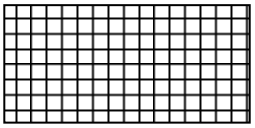
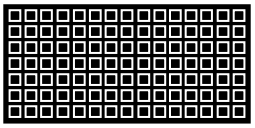
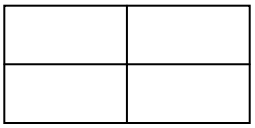
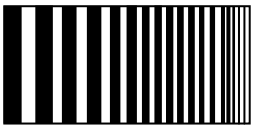
Note 2: Clear the bit2 and bit0 of TVSCR to “0”.

Figure 2.14.2 Test Video Signal Control Register

### 2.14.3 Functions

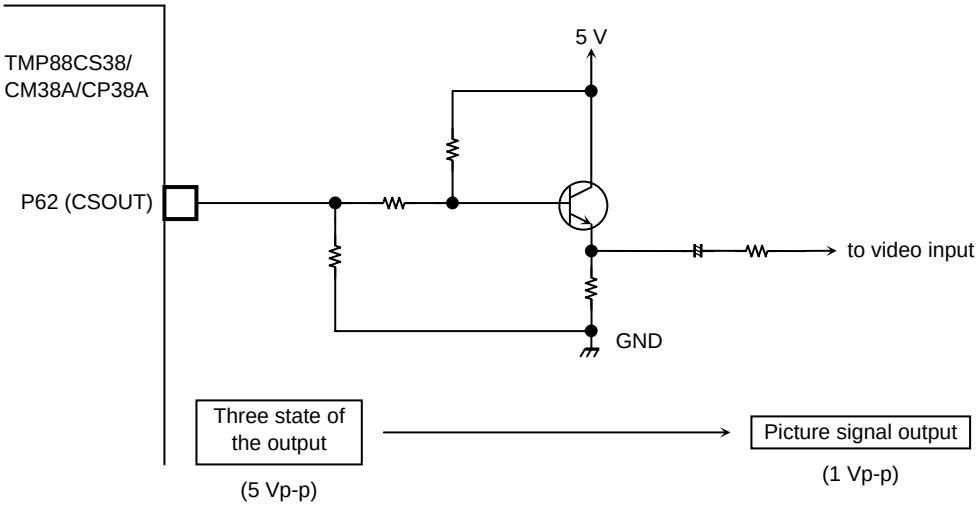
Video signal output is to generate monochromatic picture signal output to take easily the necessary tests such as TV screen white adjustment and screen distortion amplitude adjustment implemented on the final manufacturing process of a TV receiver set.

Table 2.14.1 Display Pattern and TV Screen

| Display pattern | TV screen |
|---|---|
| 000<br>(Black on the whole surface) | |
| 001<br>(White on the whole surface) | |
| 010<br>(Cross hatch) |  |
| 011<br>(Cross dot) |  |
| 100<br>(Cross bar) |  |
| 101<br>(White on the upper side/<br>black on the lower side) | |
| 110<br>(H signal pattern) | |
| 111<br>(H resolution pattern) |  |

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

Example of picture output generation)



## 2.15 On-screen Display (OSD) Circuit

The TMP88CS38/CM38A/CP38A features a built-in on-screen display circuit used to display characters and symbols on the TV screen. There are 384 characters and any characters can be displayed in an area of 32 columns × 12 lines (Include 2 columns for solid space). 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 character fonts: 384

(2) Number of display characters: 384 (32 columns × 12 lines)

(3) Composition of character: Horizontal 16 × vertical 18 dots

(4) Character sizes: 3 kinds for large, middle and small characters (Selectable line by line)

(5) Character ornamentation function
Fringing function
Smoothing function
Slant function (Italics)
Blinking function
Underline

(6) Solid space

(7) Area plane function: 2 planes

(8) Full-raster blanking function

(9) Display colors
Character colors: 8 or 15 colors (Selectable character by character)
Fringe color: 8 or 15 colors (Selectable page by page)
Background color: 8 or 15 colors (Selectable page by page)
Area plane color: 8 or 15 colors (Selectable each of 2 planes)
Raster color: 8 or 15 colors (Selectable page by page)

(10) Display position: 256 horizontal steps and 512 vertical steps for code plane
: 512 horizontal steps and 512 vertical steps for Area plane

(11) Window function: 512 vertical steps

(12) Half transparency output function

The TMP88CS38/CM38A/CP38A outputs OSD through 3 planes; code, area, and raster. 3 planes function independently. In addition, they are displayed simultaneously. There is the priority among these 3 planes, so they are displayed on a screen according to the priority.

These 3 planes have the priority such as
Code > Area > Raster.

1. Code plane

   OSD character is displayed on the code plane.

   The code plane consists of 32 characters × 1 row and a total of 12 planes. The 12 planes have the priority such as code 1 > code 2 > ... > code 11 > code 12.

   On the code plane, characters of 16 × 18 dots is displayed. These fonts are called characters, and read from character ROM and display memory through the character code on the display memory.

2. Area plane

   The area on a screen is displayed on the area plane.

   The area plane can display 2 square areas of any size by specifying coordinates. The 2 planes have the priority such as area plane 1 > area plane 2.

### 2.15.1 Configuration

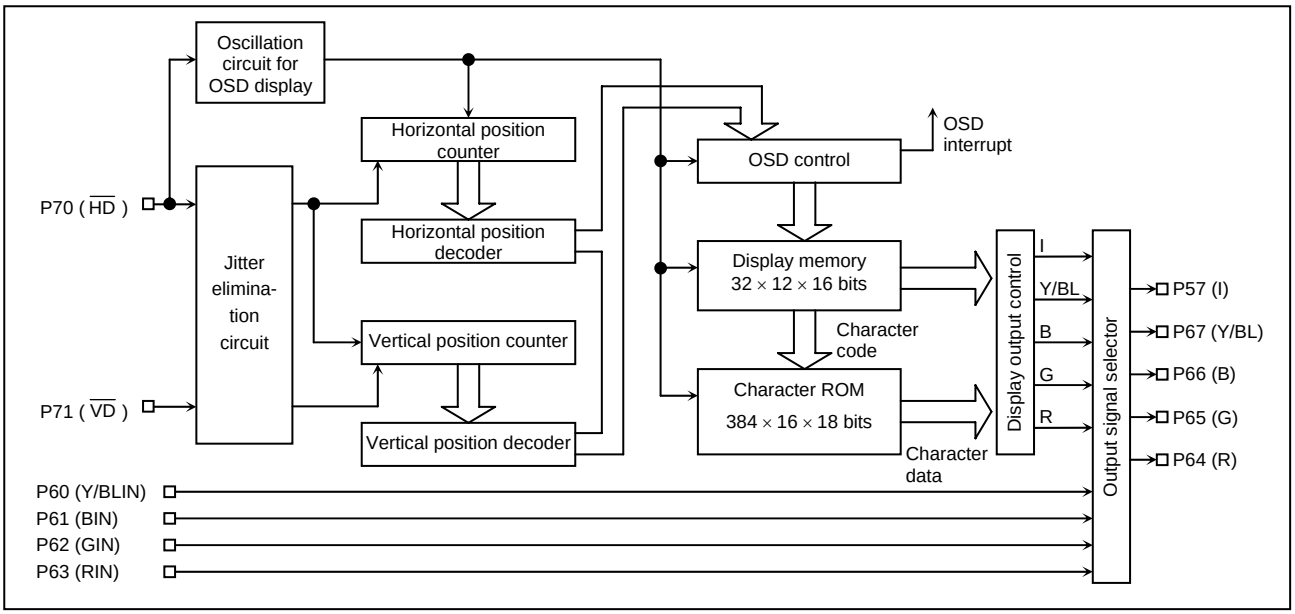



Figure 2.15-1 OSD Circuit

### 2.15.2 Character ROM and Display Memory

(1) Character ROM

The character ROM contains 384 character fonts. The user can set fonts as desired. The character ROM consists of 384 characters in 16 × 18 dots (Character codes 000H to 17FH). Each dot corresponds to one bit in the character ROM. 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 corresponding to a character code is determined by the following expression:

Start address in character ROM = CRA × 40H + 20000H

Since character code 000H is used as blank character, the character font for this character code cannot be changed. Write "0" in the data of character code 000H.

Write the data "FFH" to all unused address (5th bit of an address is "1" and also the lower 4 bits of an address are 2H to FH) in character ROM.

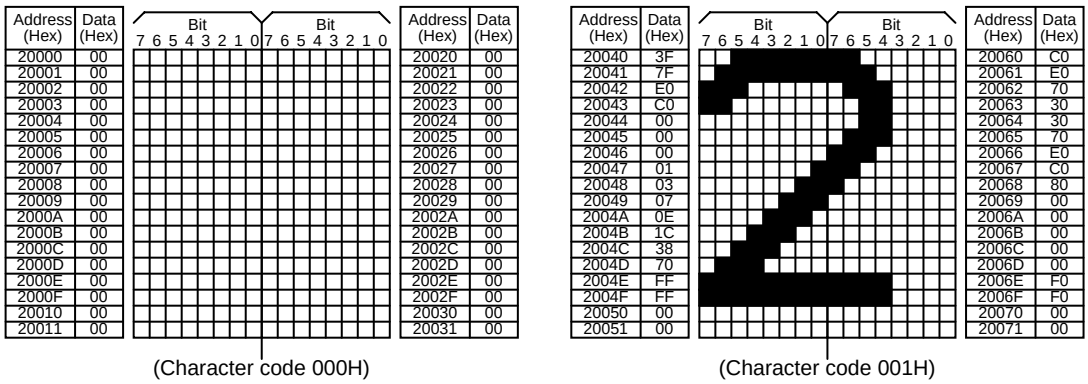
Figure 2.15-2 (a) shows an example of the character font configuration for the character code 000H and 001H, together with the ROM addresses and data.

Figure 2.15.2 (b) shows the character ROM dump list for these 2 character fonts.

Note 1: CRA: Character code (000H to 17FH).

Note 2: A data can not be read from character ROM by software.

Note 3: When ordering a mask, load the data to character ROM at addresses 20000H to 25FFFH.
And the data in unused are of character ROM are must be specified to FFH.




(a) Character font configuration

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 20000/ | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 20010/ | 00 | 00 | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF |
| 20020/ | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 20030/ | 00 | 00 | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF |
| 20040/ | 3F | 7F | E0 | C0 | 00 | 00 | 00 | 01 | 03 | 07 | 0E | 1C | 38 | 70 | FF | FF |
| 20050/ | 00 | 00 | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF |
| 20060/ | C0 | E0 | 70 | 30 | 30 | 70 | E0 | C0 | 80 | 00 | 00 | 00 | 00 | 00 | F0 | F0 |
| 20070/ | 00 | 00 | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF |

(b) ROM dump list

Note: Shared portions indicate unused data.

Figure 2.15-2 Character Font Configuration and ROM Dump List

(2) Display memory

Each character of the 384 characters displayed in 32 columns × 12 lines consists of 16 bits in the display memory. Five data items are written to the display memory: character code, color data, blinking specification, underline enable, and slant enable.

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 2.15.5.7 (1).

Note: The display memory is in an unknown state at reset.

Display memory configuration

- Character code specification register (9 bits)..... CRA8 to CRA0
- Color data specification register (4 bits) ..... IDT/RDT/GDT/BDT
- Blinking specification register (1 bit) ..... BLF
- Underline enable register (1 bit)..... EUL
- Slant enable register (1 bit) ..... SLNT

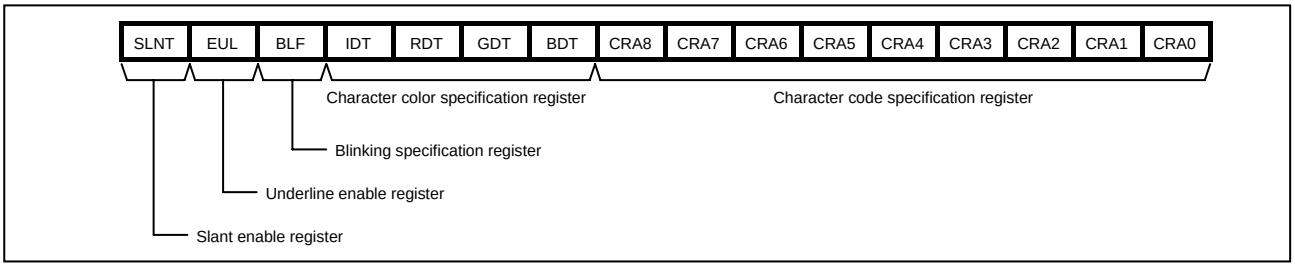



Figure 2.15-3 Display Memory Bit Configuration

| Line \ Column | 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 | 000 | 001 | 002 | 003 | 004 | 005 | 006 | 007 | 008 | 009 | 00A | 00B | 00C | 00D | 00E | 00F | 010 | 011 | 012 | 013 | 014 | 015 | 016 | 017 | 018 | 019 | 01A | 01B | 01C | 01D | 01E | 01F |
| 2 | 020 | 021 | 022 | 023 | 024 | 025 | 026 | 027 | 028 | 029 | 02A | 02B | 02C | 02D | 02E | 02F | 030 | 031 | 032 | 033 | 034 | 035 | 036 | 037 | 038 | 039 | 03A | 03B | 03C | 03D | 03E | 03F |
| 3 | 040 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 060 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 080 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 0A0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 0C0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 0E0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 100 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 120 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | 140 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | 160 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 17F |

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

Figure 2.15-4 Display Memory Address Configuration

### 2.15.3 OSD Circuit Control

The OSD circuit performs control functions using the OSD control registers which reside in addresses 0001DH to 0001FH and 00024H to 00025H in the special function registers (SFR), and in addresses 0F80H to 0FBFH in the data buffer register (DBR). Section 2.15.5.9 shows the OSD control registers. 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, (Bit0 in ORDON) to "1" enables display (Starts display). Setting DON to "0" disables display (Halts display).

Note: The contents of OSD control registers except PIDS, P67S to P64S, ORCLKF, CRCLKC are initialized in STOP mode. Then, OSD display clock does not stop in STOP mode. Therefore, clear ORCLKC to "00H" when stop the OSD display clock.

### 2.15.4 OSD Control Register Write

There are lists of the OSD control registers on Figure 2.15-30 and Figure 2.15-31.

When data is written into a shaded register, the data is transferred to the OSD circuit, and then the data becomes valid. After data is written into an unshaded register, the data is transferred to the OSD circuit, and then the data becomes valid.

To transfer the contents of a control register to the OSD circuit, use data transfer request register RGWR (Bit2 in ORDON).

Setting "1" in the RGWR register outputs the transfer request signal to the OSD circuit. Three instruction cycles later, transfer of the written data to the OSD circuit starts. While the data is being transferred, data transfer status monitoring flag RGWR (Bit2 in ORDON) is "1". When this transfer is completed, the flag is cleared to "0".

Written data transfer register (1 bit) ..... RGWR (Bit2 in ORDON)

| | | |
|---|---|---|
| "0" | ..... | Initialized state |
| "1" | ..... | Transfers written data to OSD circuit.<br>(after transfer, RGWR is reset to 0.) |

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

(1) RGWR system

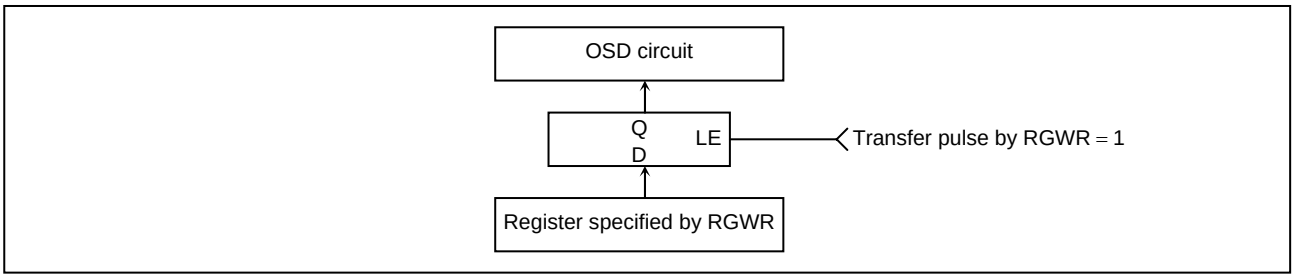



Figure 2.15-5 RGWR System

(2) Transfer timing

1. No display area

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.

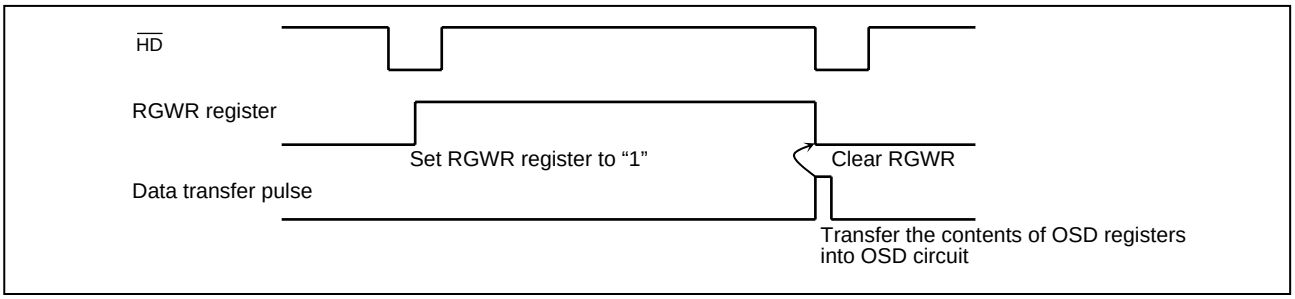



Figure 2.15-6 Data Transfer Timing in No Display Area

2. Display area (Including any lines specified as display off by character size)

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.

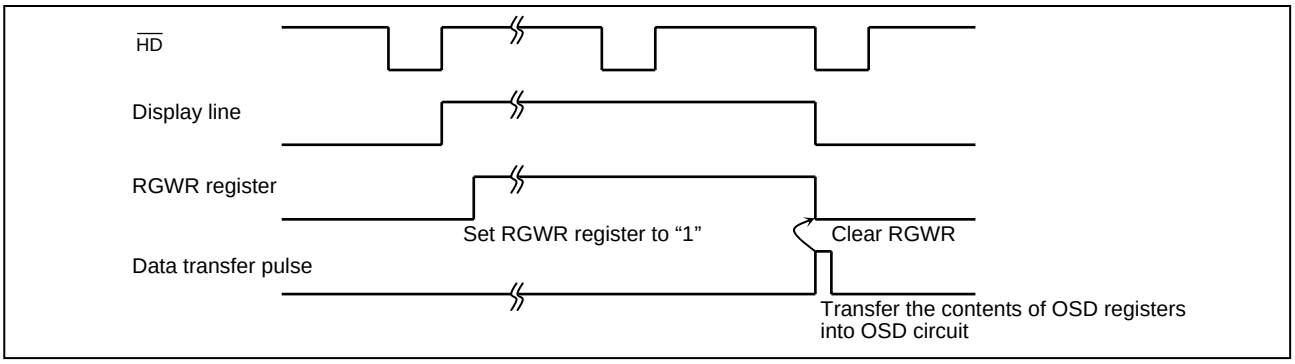



Figure 2.15-7 Data Transfer Timing in Display Area

## 2.15.5 OSD Function

### 2.15.5.1 Signal Control (Port I/O)

(1) P6 port output select function

This function is used to select whether the contents of port P57, P67 to P64 will be output or I, R, G, B, Y/BL signals of the OSD circuit will be output on pins P57, P67 to P64.

P57 port output select registers (1 bit): PIDS (Bit3 in ORP6S)

| | PIDS = 0 | PIDS = 1 |
|---|---|---|
| P57 | I | Port |

P67 to P64 port output select registers (4 bits): P67S, P66S, P65S, P64S, (Bit7 to 4 in ORP6S)

| | P6nS = 0 | P6nS = 1 |
|---|---|---|
| P64 | R | Port |
| P65 | G | |
| P66 | B | |
| P67 | Y/BL | |

(2) OSD pin output polarity control function

This function is used to select the polarity of the OSD outputs for RGB, I and Y/BL.

Output polarity control register (4 bits) ..... BLIV, YIV, RGBIV, IIV (Bit3 to 0 in ORIV)

"0" .... Active high
"1" .... Active low

(3) OSD pin input polarity control

Input polarity control

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

For Y/BLIN ..... YBLII (Bit5 in ORIV)

For RIN, GIN, and BIN ..... RGBII (Bit4 in ORIV)

Input polarity control

YBLII,
RGBII
"0" ... Active high
"1" ... Active low

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

For $\overline{VD}$ ..... VDPOL (Bit7 in ORIV)

For $\overline{HD}$ ..... HDPOL (Bit6 in ORIV)

Input polarity control

VDPOL, HDPOL

"0" ..... Not invert input signal
"1" ..... Invert input signal

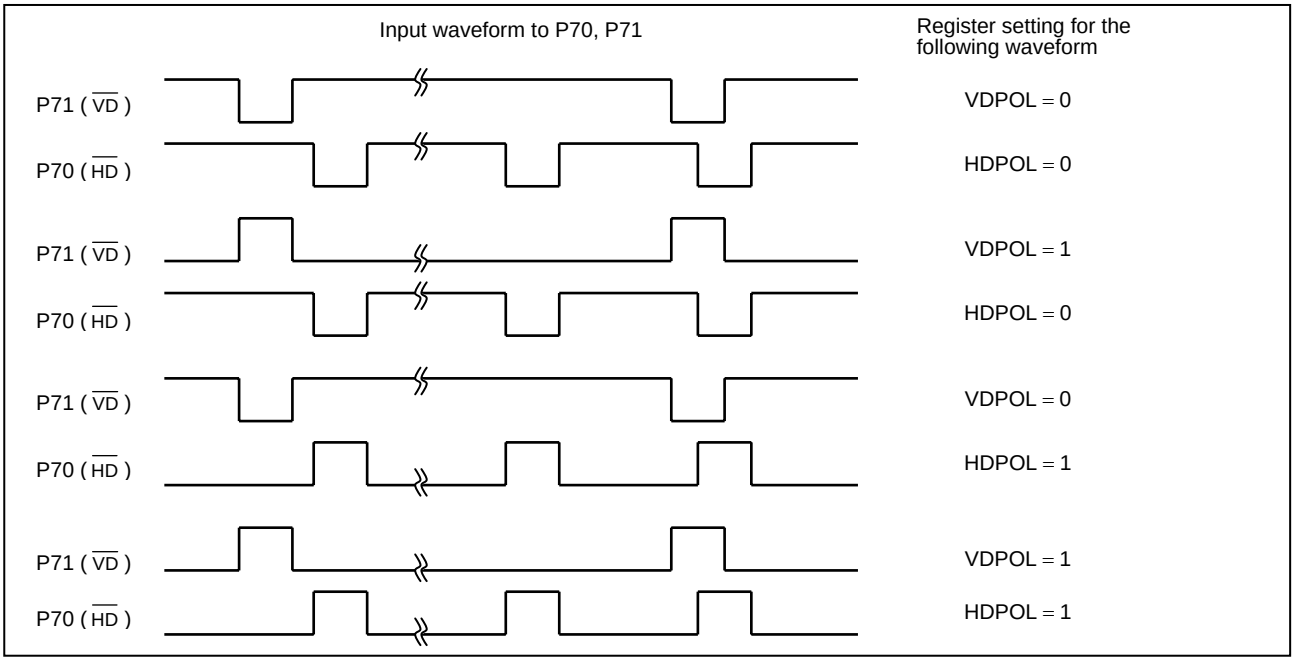



Figure 2.15-8 $\overline{VD}$ / $\overline{HD}$ input and VDPOL/HDPOL

(4) Y/BL signal select function

This function is used to select either Y or BL signal output from the Y/BL pin.

Y/BL signal select register (1 bit).....BLCS (Bit7 in ORP6S)

"0" ..... Y signal output
"1" ..... BL signal output

Y signal..... Output in all OSD areas (Logical OR for R, G, B data as character data, fringing data, area data, etc.)

BL signal ..... When EXBL is "0":

Output in all display character areas

(except for character code 000H: Blank character)

When EXBL is "1":

Output in the whole page

(5) I signal function select

When PISEL (Bit6 in ORETC) is set to "1" and PIDS (Bit3 in ORP6S) is set to "0", Port 57 (I pin) can be used as half transparency/half tone through an extra circuit.

At half transparency/half tone function, contents of IDT (Bit3 in ORDSN) is make no sense. Therefore character color are limited to 8 colors.

Similarly background color, fringing color, raster plane color and area plane color are limited to 8 colors.

When PISEL (Bit6 in ORETC) sets to "0" and, PIDS (Bit3 in ORP6S) set to "0", 15 colors to be selectable.

(6) R, G, B, Y/BL Internal/external signal select.

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

R, G, B, Y/BL signal select registers (2 bits).....MPXS1/MPXS0

(Bits 1 and 0 in ORP6S)

| | | |
|---|---|---|
| "00" | ..... | Simultaneous output (Signal from the OSD circuit has higher priority.) |
| "01" | ..... | Output of signal from internal OSD circuit |
| "10" | ..... | Output of signal from external input |
| "11" | ..... | Simultaneous output (External input signal has higher priority.) |

### 2.15.5.2 OSD Data Output Format Control

(1) Scan mode

The double scan mode is used to handle non-interlaced scanning TV. When double scan mode is enabled, the vertical display counter increases every 2 scan lines and a vertical size of a dot is double. This function is enabled by setting VDSMD (Bit7 in ORETC) in the OSD control register to "1".

Scan mode select register (1 bit)..... VDSMD (Bit7 in ORETC)

"0" ..... Normal mode
"1" ..... Double scan mode

Note 1: The data written to those control register is transferred to the OSD circuit and become valid when the data is written.

Note 2: When OSD circuit is used on an interlace scanning TV, a jitter elimination circuit must be enabled and set AFLD to "1" in JECR.

Table 2.15.1 The Difference of 2 Types of Scan Mode

| | Normal Mode | Double Scan Mode |
|---|---|---|
| Specification unit of vertical display start position | One scanning line | Two scanning lines |
| 1 dot height | – | Normal mode height × 2 |

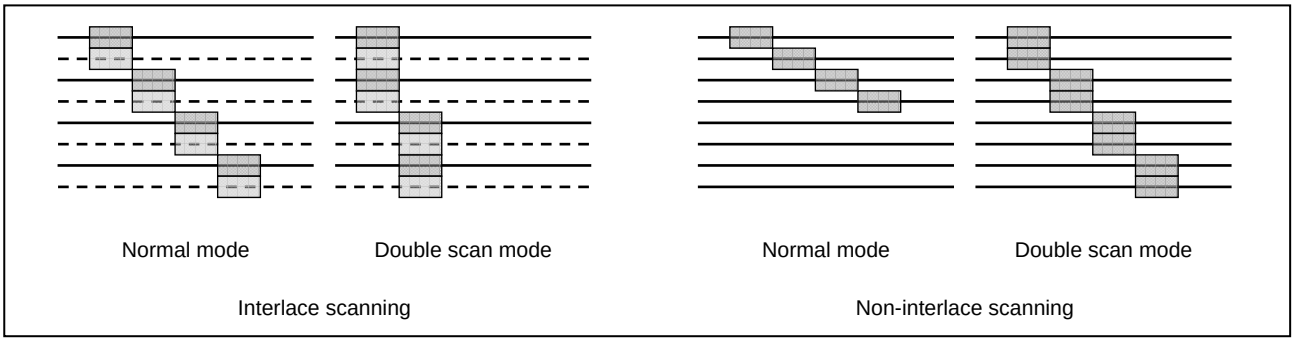



Figure 2.15-9 Scan Mode

#### 2.15.5.3 Display Position Control

(1) Code display position setting

1. Horizontal display start position

The horizontal display start position can be set in 256 steps by writing to OSD control registers HS17 to HS10 (Bit7 to 0 in ORHS1). The value is in common with all lines.

Specification unit: 2 $T_{OSC}$

Specification steps: 256

Specification horizontal display start position: Line 1 to 12: HS17 to HS10 (ORHS1)

$$HS1 = (HS17 \text{ to } HS10)_H \times 2T_{OSC} + 20T_{OSC} \text{ (Line1 to 12)}$$

Note 1: $T_{OSC}$: One cycle of OSD oscillation.

Note 2: The data written to these control registers is transmitted to OSD circuit by setting RGWR (Bit2 in ORDON) to "1".

2. Vertical display start position

The vertical display start position can be specified for each display line using 512 steps by writing to VSn8 to VSn0 (in ORVSn (n:1 to 12)).

Specification unit: 1 scan line

Specification steps: 512

Specification vertical display start position:

Line1: VS18 to VS10 (ORVS 1)

Line2: VS28 to VS20 (ORVS 2)

⋮

Line12: VS128 to VS120 (ORVS 12)

Line n: $VSn = (VSn8 \text{ to } VSn0)_H \times 1T_{HD}$ (n: 1 to 12)

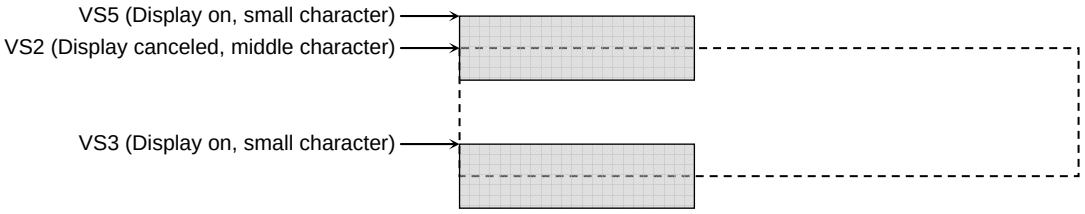
Note 1: $T_{HD}$: One cycle of $\overline{HD}$ signal.

Note 2: The data written to these control registers is transmitted to OSD circuit by setting RGWR
(Bit2 in ORDON) to “1”.

Note 3: If display lines are overlapped each other, previous display line is enabled and next line is disabled.
If vertical display start positions of two or more lines are set on same value, high priority line is enabled. Lines of OSD (VS1 to VS12) are fixed priority levels as follows:

VS1 > VS2 > VS3 > ..... > VS12

Set the vertical display start position not to overlap display lines.




Occasion of overlapping

Note 4: The line which is displayed off is managed as a small size character line.

Note 5: Transfer the contents of vertical display start position registers into OSD circuit before the
position of the scanning line coincides with their own vertical display start position.

(2) Area display position setting

The planes have the priority such as Code plane > Area plane 1 > Area plane 2 > Raster plane.

1. Horizontal display start and end position

The horizontal display start position can be set in 512 steps by writing to OSD control registers AHSn8 to AHSn0 (Bit8 to 0 in ORAHSn). And also display stop position is correspond to AHEn8 to AHEn0 (Bit8 to 0 in ORAHEn). (n: 1 to 2)

Horizontal display start position

$$AHSn = (AHSn8 \text{ to } AHSn0)_H \times 2T_{OSC}$$

Horizontal display end position

$$AHEn = (AHEn8 \text{ to } AHEn0)_H \times 2T_{OSC}$$

Note 1: $T_{OSC}$: One cycle of OSD oscillation.

Note 2: If the horizontal display start position for characters is the same as that for areas, the two positions are not displayed at the same time. The horizontal display start position for characters is displayed 16 $T_{OSC}$ (Corresponding to a register value of 8) later than that for areas.

2. Vertical display start and end position

The vertical display start position can be set in 512 steps by writing to OSD control registers AVSn8to AVSn0 (Bit8 to 0 ORAVSn). And also display stop position is correspond to AVEn8 to AVEn0 (Bit8 to 0 in ORAVEn). (n: 1 to 2)

Vertical display start position

$$AVSn = (AVSn8 \text{ to } AVSn0)_H \times T_{HD}$$

Vertical display end position

$$AVEn = (AVEn8 \text{ to } AVEn0)_H \times T_{HD}$$

Note: $T_{HD}$: One cycle of $\overline{HD}$ signal.

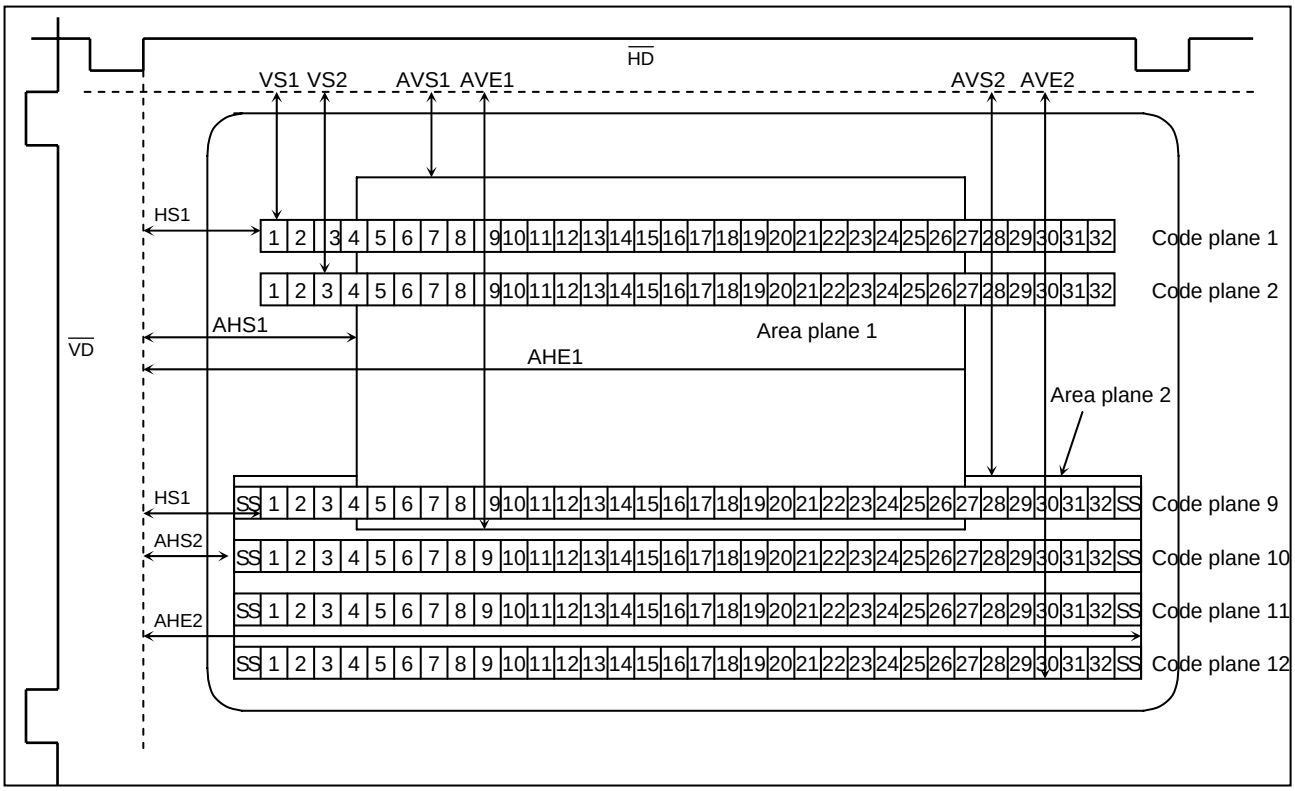



Figure 2.15-10 TV scan image

### 2.15.5.4 Character Ornamentation Control

(1) Character sizes

Character size can be selected line by line from 3 sizes. And display on/off also can be set line by line. Small, middle and large character size and display on/off can be set with OSD control registers CSn (n: 1 to 12, ORCS4, ORCS8, ORCS12) in the OSD control registers.

Character sizes: 3 sizes (Small, middle and large)

Character size and display on/off specification unit: Line

Character size select/display on/off register (2 bits × 12)

Line 1: CS1

Line 2: CS2

: :

Line 12: CS12

Table 2.15.2 Character Size and Display On/Off Specifications (n: 1 to 12)

| CSn (Upper Bit) | CSn (Lower Bit) | Character Size | Display On/Off |
|---|---|---|---|
| 1 | 1 | Small | On |
| 1 | 0 | Middle | On |
| 0 | 1 | Large | On |
| 0 | 0 | – | Off |

Note 1: The display off line operates like the width of small character size line thought the character is not displayed.

Note 2: The data written to these control registers is transmitted to OSD circuit by setting RGWR (Bit2 in ORDON) to “1”.

Note 3: When OSD circuit is used on an interlace scanning TV, a jitter elimination circuit must be enabled and set AFLD to “1” in JECR.

Note 4: When VDSMD and AFLD are “0”, only character of even display dot is displayed. (Refer to 2.16 a jitter elimination circuit.)

Table 2.15.3 Dot and Character Sizes

| | | VDSMD = 0 (Normal mode) | | | VDSMD = 1 (Double scan mode) | | |
|---|---|---|---|---|---|---|---|
| | | Dot Size | Character Size | | Dot Size | Character Size | |
| | | | EFRn = 0 (Fringe off) | EFRn = 1 (Fringe on) | | EFRn = 0 (Fringe off) | EFRn = 1 (Fringe on) |
| EULAn = 0 (Underline off) | Small | $1T_{OSC} \times 0.5T_{HD}$ | $16T_{OSC} \times 9T_{HD}$ | $16T_{OSC} \times 11T_{HD}$ | $1T_{OSC} \times 1T_{HD}$ | $16T_{OSC} \times 18T_{HD}$ | $16T_{OSC} \times 20T_{HD}$ |
| | Middle | $2T_{OSC} \times 1T_{HD}$ | $32T_{OSC} \times 18T_{HD}$ | $32T_{OSC} \times 20T_{HD}$ | $2T_{OSC} \times 2T_{HD}$ | $32T_{OSC} \times 36T_{HD}$ | $32T_{OSC} \times 40T_{HD}$ |
| | Large | $4T_{OSC} \times 2T_{HD}$ | $64T_{OSC} \times 36T_{HD}$ | $64T_{OSC} \times 40T_{HD}$ | $4T_{OSC} \times 4T_{HD}$ | $64T_{OSC} \times 72T_{HD}$ | $64T_{OSC} \times 80T_{HD}$ |
| EULAn = 1 (Underline on) | Small | $1T_{OSC} \times 0.5T_{HD}$ | $16T_{OSC} \times 12T_{HD}$ | $16T_{OSC} \times 13T_{HD}$ | $1T_{OSC} \times 1T_{HD}$ | $16T_{OSC} \times 24T_{HD}$ | $16T_{OSC} \times 25T_{HD}$ |
| | Middle | $2T_{OSC} \times 1T_{HD}$ | $32T_{OSC} \times 24T_{HD}$ | $32T_{OSC} \times 25T_{HD}$ | $2T_{OSC} \times 2T_{HD}$ | $32T_{OSC} \times 48T_{HD}$ | $32T_{OSC} \times 50T_{HD}$ |
| | Large | $4T_{OSC} \times 2T_{HD}$ | $64T_{OSC} \times 48T_{HD}$ | $64T_{OSC} \times 50T_{HD}$ | $4T_{OSC} \times 4T_{HD}$ | $64T_{OSC} \times 96T_{HD}$ | $64T_{OSC} \times 100T_{HD}$ |

$T_{OSC}$: One cycle of OSD oscillation, $T_{HD}$: One cycle of $\overline{HD}$ signal

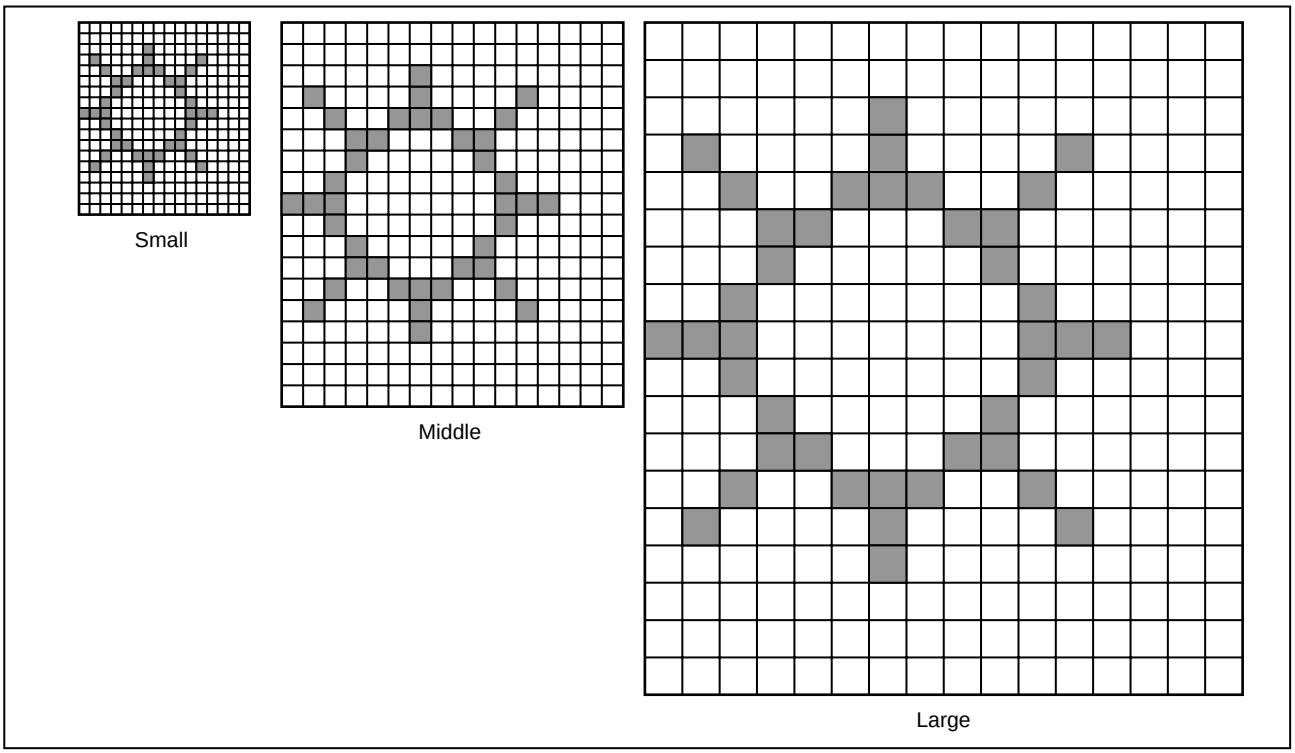



Figure 2.15-11 Character Size

(2) Smoothing function

The smoothing function is used to make characters look smooth. Enabling smoothing displays 1/4 dot between two dots connecting corner to corner within a character. Small size character can not be enabled smoothing. Smoothing is enabled by setting ESMZ (Bit4 in ORETC) in the OSD control register to "1".

Smoothing specification unit:Display page

Smoothing specification register (1 bit) ..... ESMZ (Bit4 in ORETC)

"0" ..... Disable smoothing
"1" ..... Enable smoothing

Note: Data of the register is transferred to the OSD circuit and become valid when the data is written.

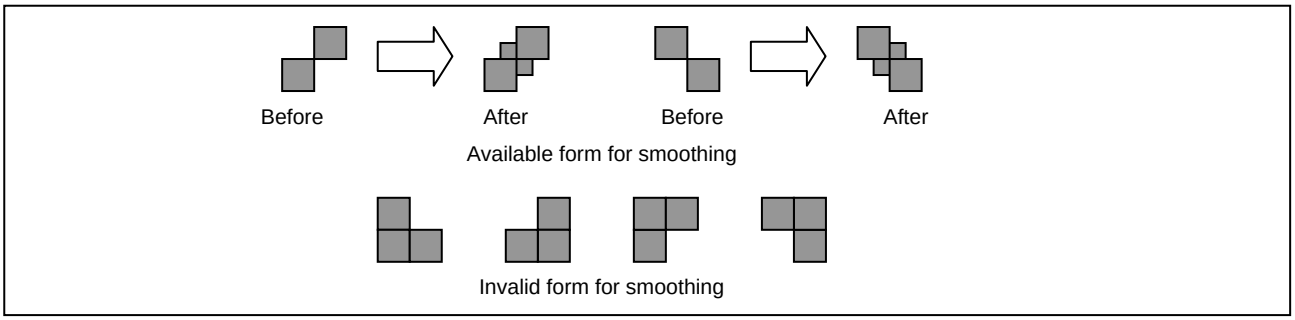



Figure 2.15-12 Available Form and Invalid Form for Smoothing

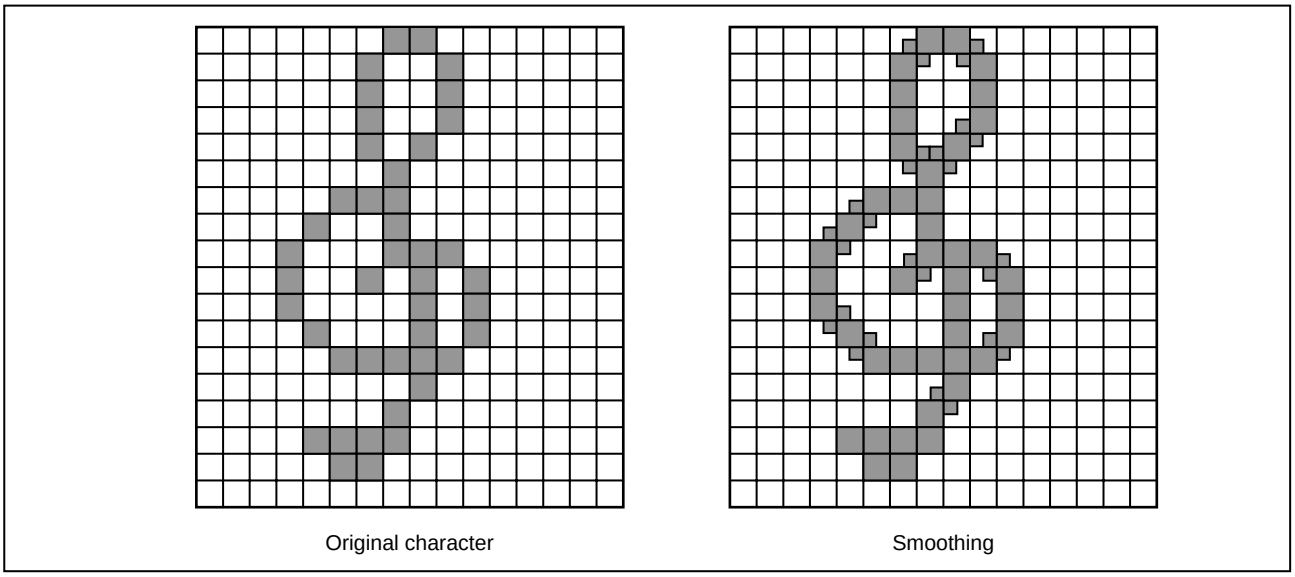



Figure 2.15-13 Smoothing Example

(3) Fringing function

The fringing function is used to display a character with a fringe width is 1 dot in a different color from that of the character. When a character is displayed with the maximum of 18 vertical dots and 16 horizontal dots, the fringe exceeds right and left, top, and bottom of the character display area. If there is an adjacent character that 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 (OREFR8) and EFR9 to EFR12 (OREFR12) in the OSD control register to "1".

A color for fringe is specified common to all lines using OSD control registers, IFDT, RFDT, GFDT, and BFDT (Bit3 to 0 in ORBK).

Fringing specification unit: Line

Fringing enable register (1 bit × 12) ... EFRn (n: 1 to 8) (OREFR8), EFRn (n: 9 to 12) (OREFR12)

"0" ..... Disable fringing
"1" ..... Enable fringing

Fringe colors: 8 or 15

Fringe color specification unit: Display page

Fringe color register (4 bits) .... IFDT, RFDT, GFDT, BFDT (Bit3 to 0 in ORBK)

I signal function select: PISEL (Bit6 in ORETC)

"0" ..... 15 colors specification
I pin can be used to make a half level of R, G, B signal (Dark color) through an extra circuit.

"1" ..... 8 colors specification
Contents of IDT register is disregarded.
I pin can be used as half transparency/half tone through an extra circuit.

Note: The fringe of 1st column character does not exceed left, and the fringe of 32th character does not exceed right.

Table 2.15.4 Fringe Color (15 colors)

| IFDT | RFDT | GFDT | BFDT | Figure Color |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Black |
| | 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 |
| 1 | 0 | 0 | 0 | Black |
| | 0 | 0 | 1 | Dark blue |
| | 0 | 1 | 0 | Dark green |
| | 0 | 1 | 1 | Dark cyan |
| | 1 | 0 | 0 | Dark red |
| | 1 | 0 | 1 | Dark magenta |
| | 1 | 1 | 0 | Dark yellow |
| | 1 | 1 | 1 | Gray |

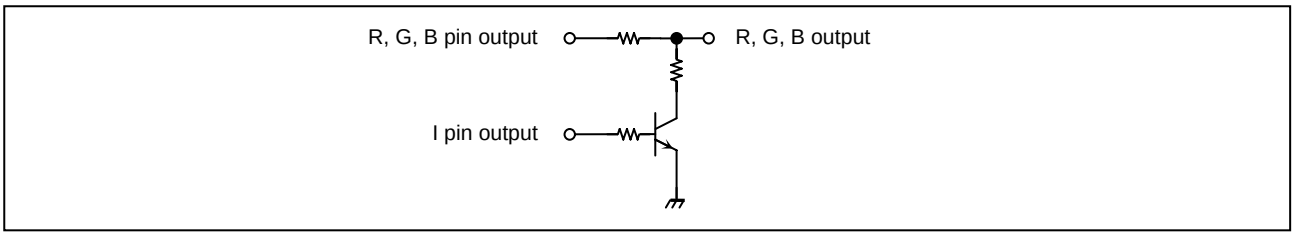



Figure 2.15-14 Example Circuit for 15 Colors by I Pin.

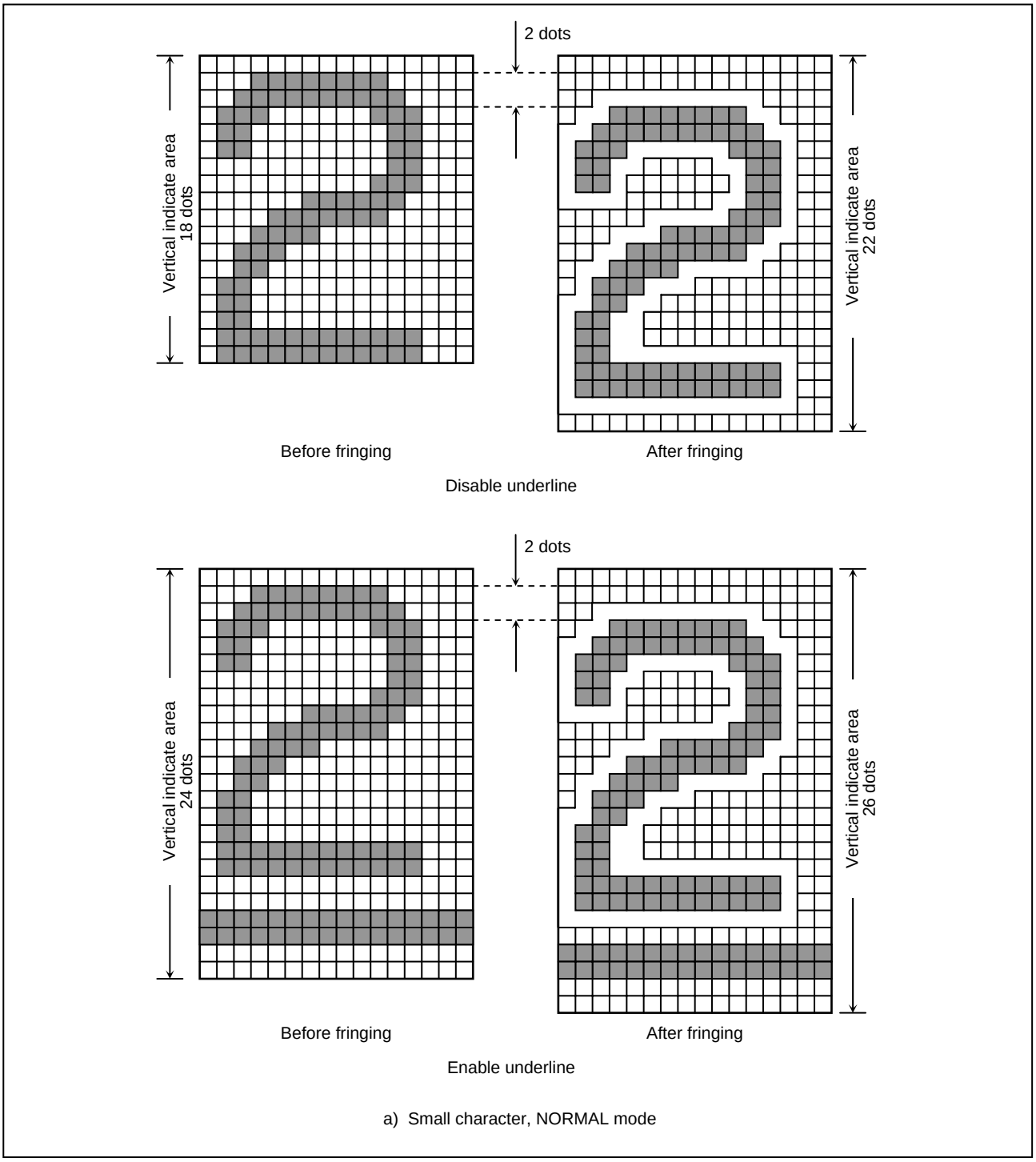
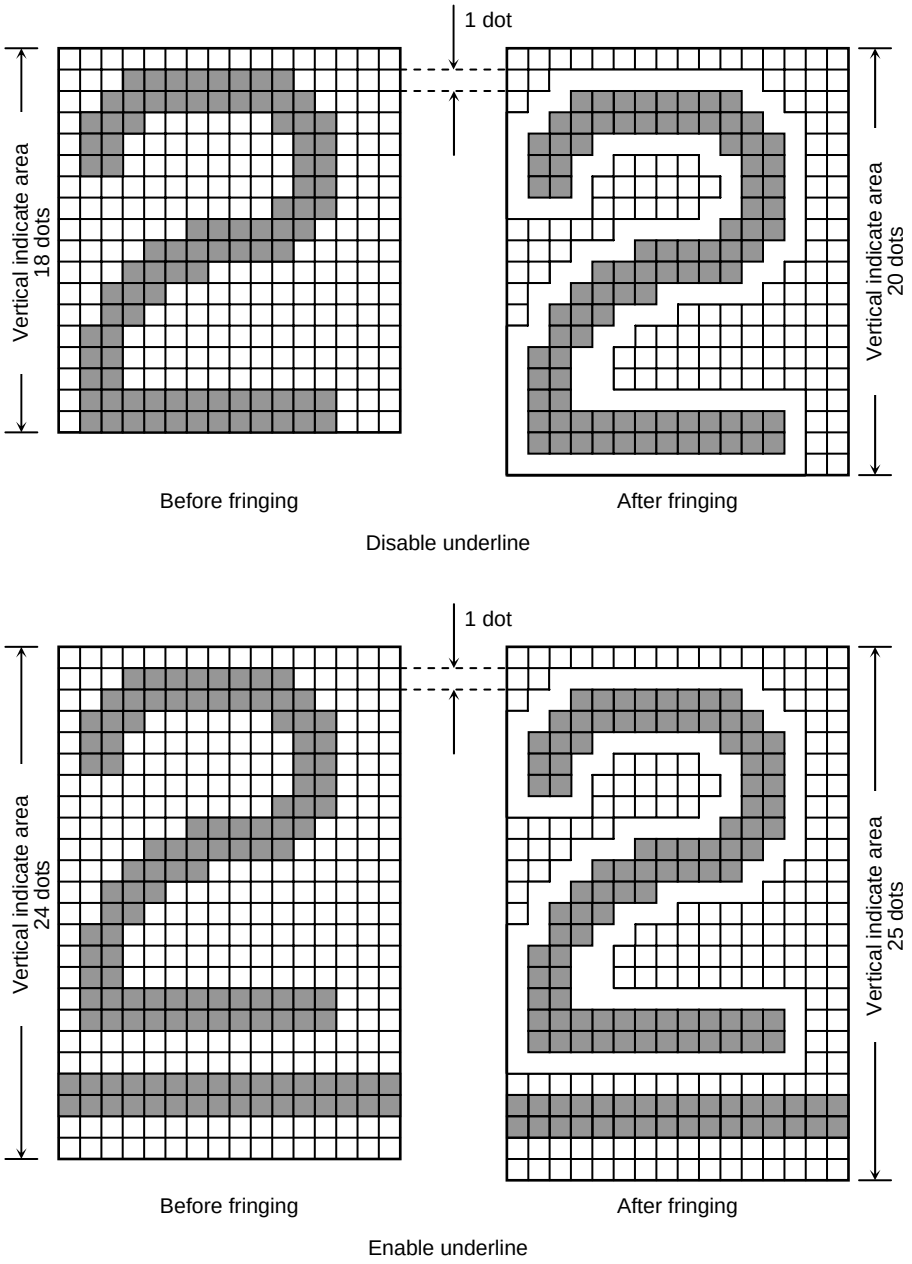
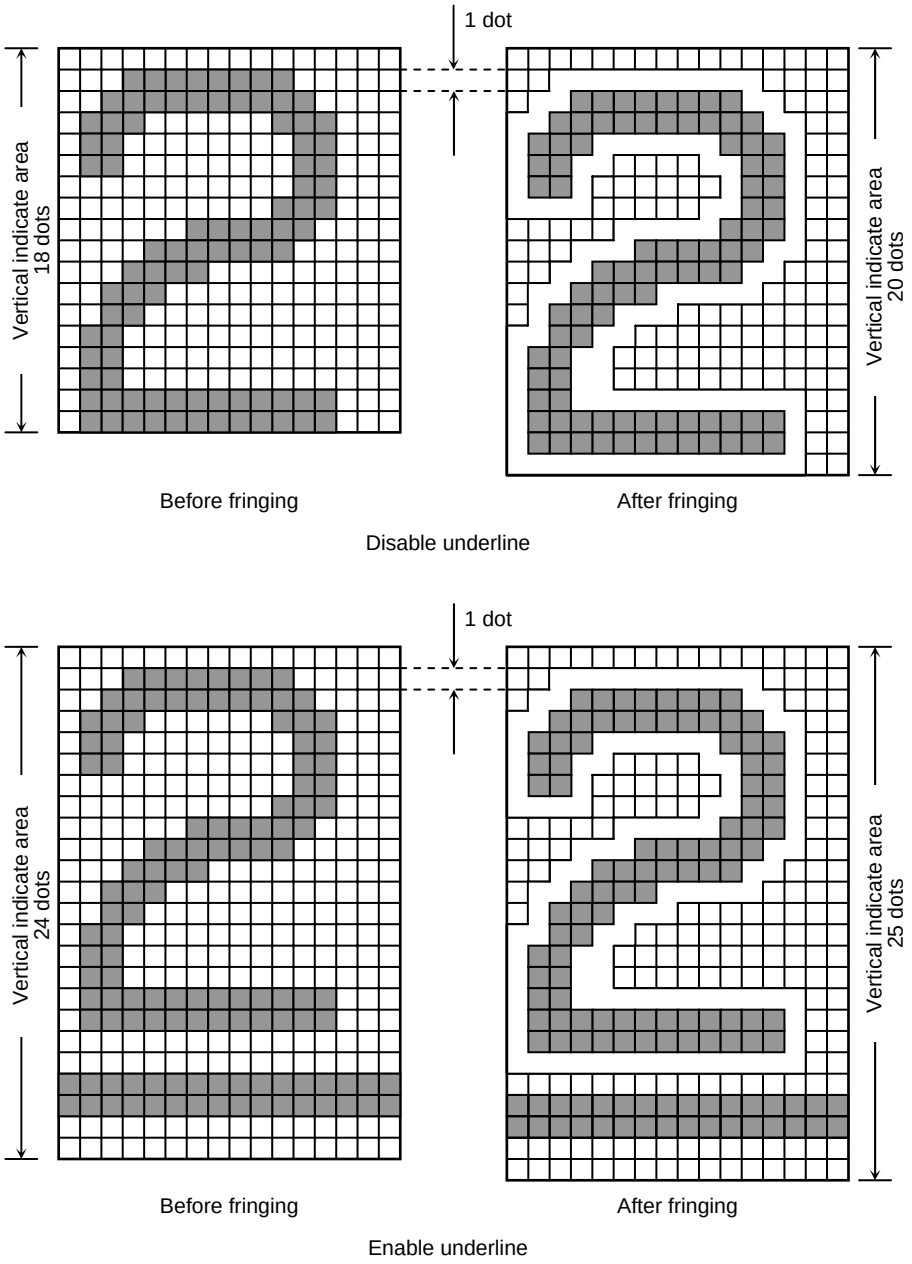



Figure 2.15-15 (a) Fringing Example



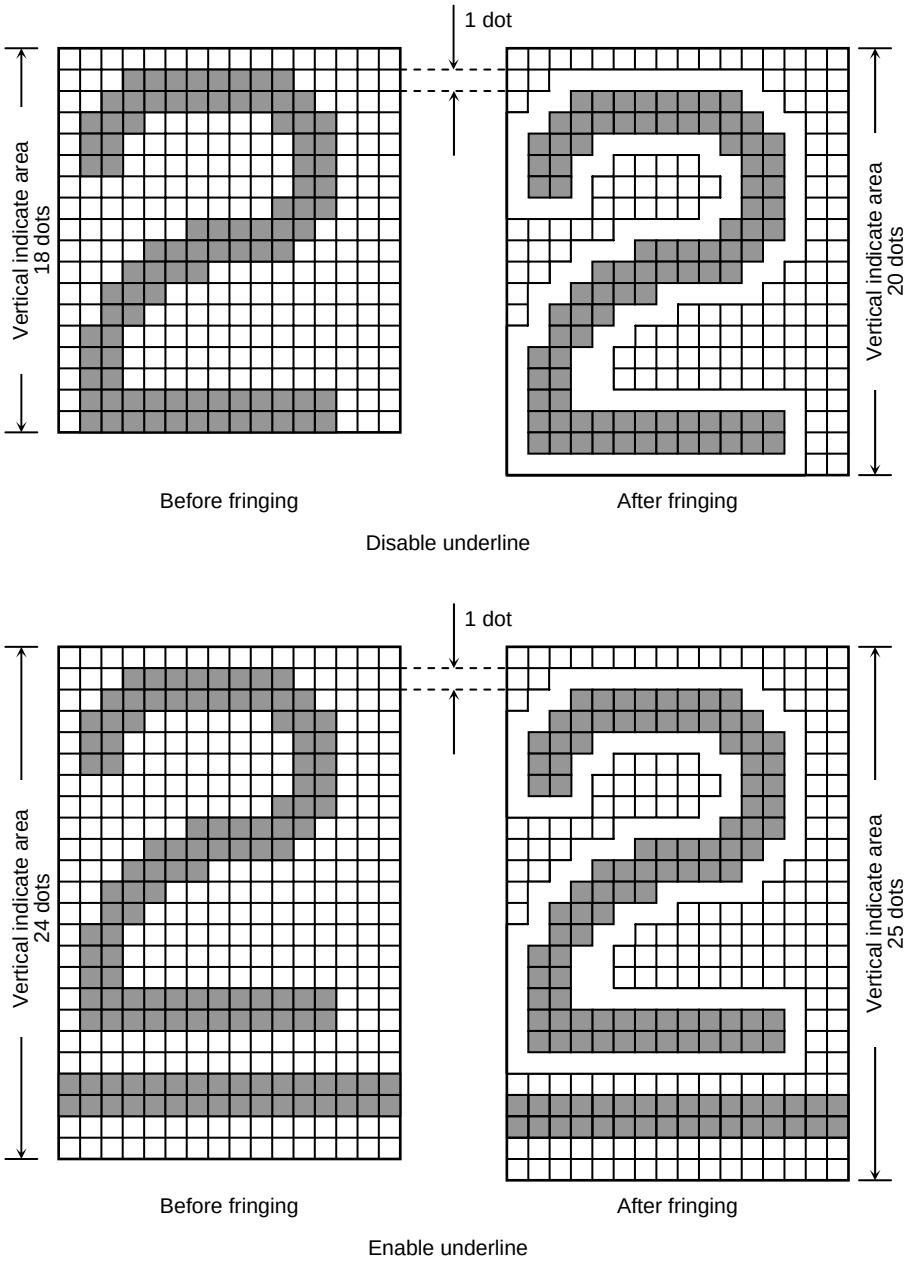

b) Small character, double scan mode

Figure 2.15-16 (b) Fringing Example




c) Middle/Large character, NORMAL mode

Figure 2.15-17 (c) Fringing Example




d) Middle/Large character, double scan mode

Figure 2.15-18 (d) Fringing Example

(4) Background function

Background color function is used to color the entire background for the character area (Refer to Table 2.15.4). Except the character area whose character code is 000H.

This function is specified for each display page by setting EBKGD (Bit7 in ORRCL) in the OSD control register to "1".

A background color is specified for each display page by setting IBDT, RBDT, GBDT, and BBDT (Bit7 to 4 in ORBK) in the OSD control registers.

Background specification unit: Display page

Background enable register (1 bit) ..... EBKGD (Bit7 in ORRCL)

"0" ..... Disable background
"1" ..... Enable background

Background color specification unit: Display page

Background color specification registers (4 bits) ... IBDT, RBDT, GBDT, BBDT (Bit7 to 4 in ORBK)

I signal function select: PISEL (Bit6 in ORETC)

"0" ..... 15 colors specification
I pin can be used to make a half level of R, G, B signal (Dark color) through an extra circuit.

"1" ..... 8 colors specification
Contents of IBDT register is disregarded.
I pin can be used as half transparency/half tone through an extra circuit.

Table 2.15.5 Background Color (15 colors)

| IBDT | RBDT | GBDT | BBDT | Background Color |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Black |
| | 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 |
| 1 | 0 | 0 | 0 | Black |
| | 0 | 0 | 1 | Dark blue |
| | 0 | 1 | 0 | Dark green |
| | 0 | 1 | 1 | Dark cyan |
| | 1 | 0 | 0 | Dark red |
| | 1 | 0 | 1 | Dark magenta |
| | 1 | 1 | 0 | Dark yellow |
| | 1 | 1 | 1 | Gray |

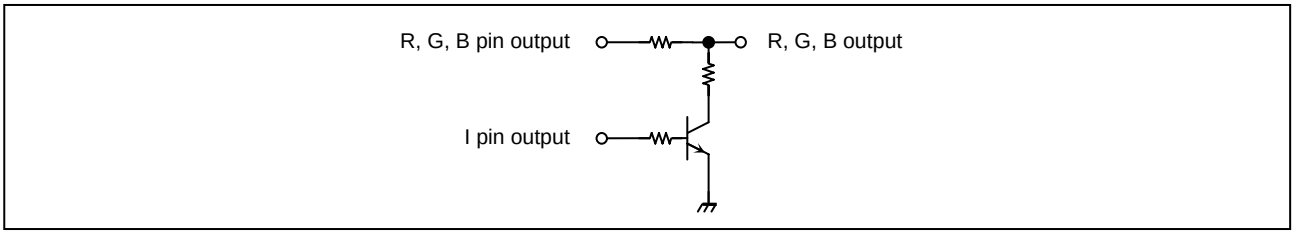



Figure 2.15-19 Example Circuit for 15 Colors by I Pin.

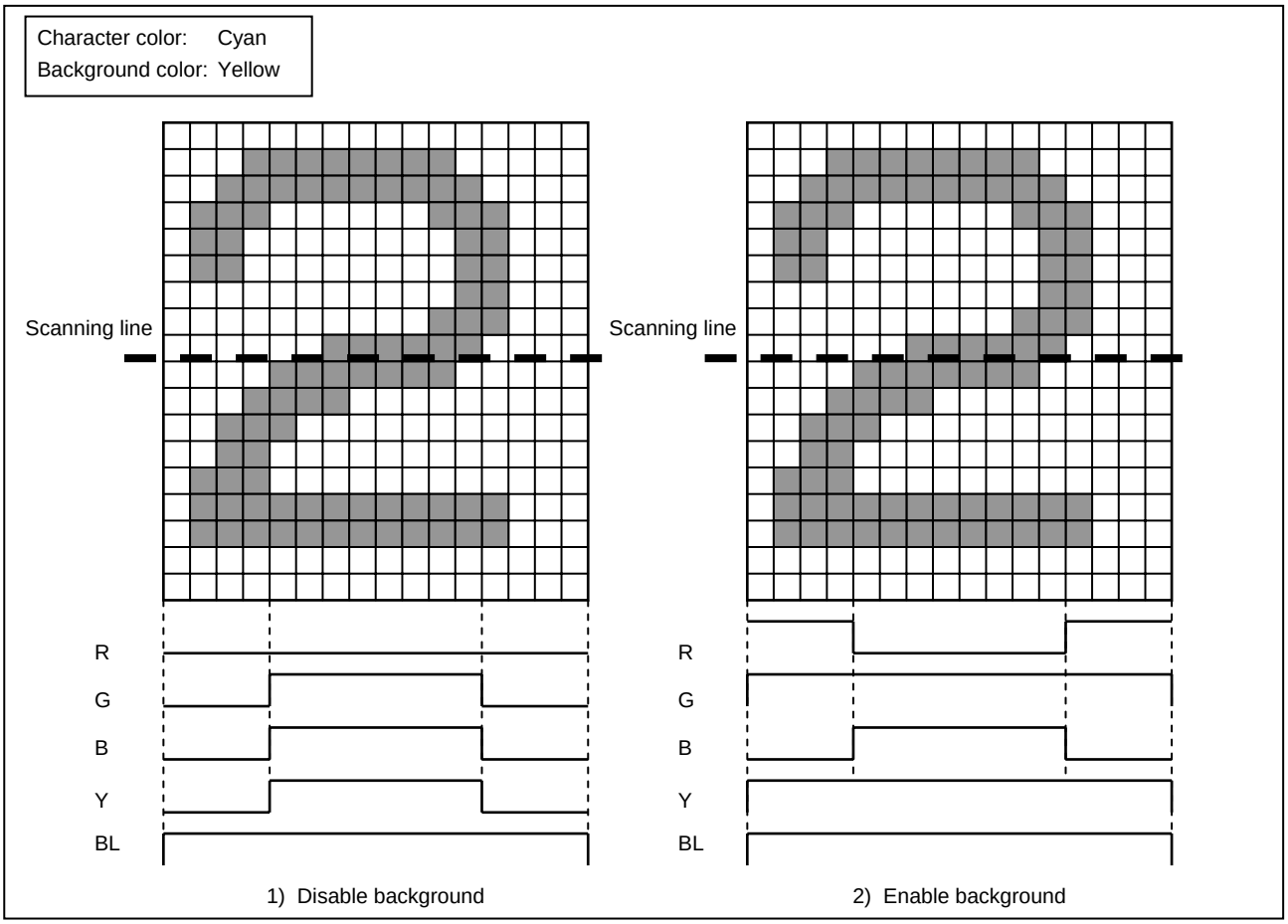



Figure 2.15-20 Background Function

Note: When the background function is enabled, the line enable the fringing function should not start with a blank character. If it starts with a blank character, a fringe is displayed to the left of the blank character.

2.15.5.5 OSD Display Screen Control

(1) Display on/off

This function is used to display characters specified for on/off display.

Display on/off specification unit: Display page

Display on/off specification register (1 bit) ..... DON (Bit0 in ORDON)

"0" ..... Disable display
"1" ..... Enable display

Note: Do not start STOP mode during display is enable.

(2) Window function

This function is used to set upper and lower limit of display page. Window upper limit is specified by WVSH (ORWVSH). Window lower limit is specified by WVSL (ORWVSL). This function is enabled by setting EWDW (Bit1 in ORDON ) in the OSD control register to 1.

Window specification unit: Display page

Window function enable specification register (1 bit) ..... EWDW (Bit1 in ORDON)

"0" ..... Disable window function
"1" ..... Enable window function

Window upper limit specification register (9 bits) .... WVSH8 to 0 (ORWVSH)

Window lower limit specification register (9 bits) ..... WVSL8 to 0 (ORWVSL)

Window upper and lower limit position .....

When VDSMD is "0" (Normal mode):

$$WVSH = (WVSH8 \text{ to } WVSH0)_H \times T_{HD}$$

$$WVSL = (WVSL8 \text{ to } WVSL0)_H \times T_{HD}$$

When VDSMD is "1" (Double scan mode):

$$WVSH = (WVSH8 \text{ to } WVSH0)_H \times 2T_{HD}$$

$$WVSL = (WVSL8 \text{ to } WVSL0)_H \times 2T_{HD}$$

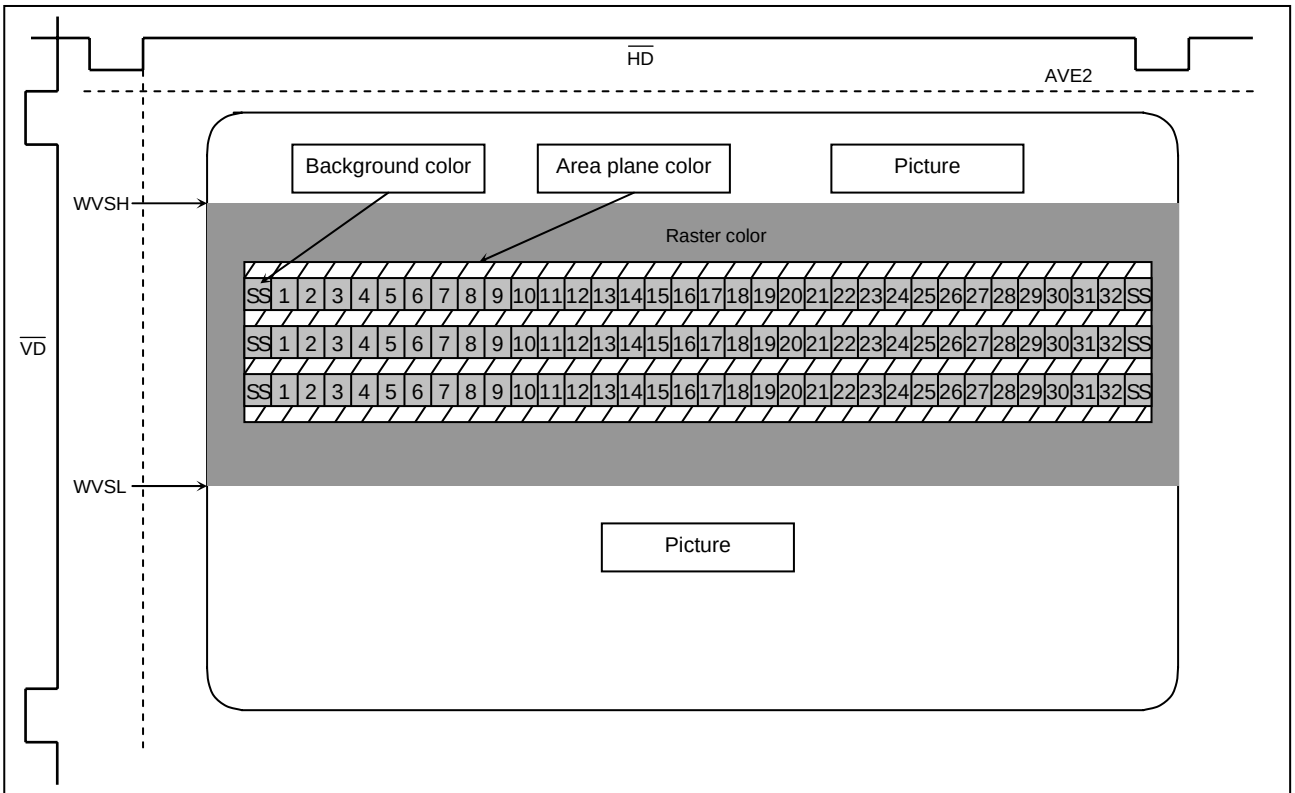
Note 1: THD: One cycle of $\overline{HD}$ signal

Note 2: WVSL > WVSH ≥ "1"

Note 3: Modify the value of window upper and lower limit register and the value of EWDW during $\overline{VD}$ signal is low.

Note 4: It is recommendable that the window function is always enabled (EWDW = "1") and set WVSH to "01H", WVSL to "1FEH".

Note 5: Characters and symbols at scanning line specified by WVSL are not displayed.




Window display: On, Area plane display: On, Background color display: On, Raster plane display: On

Correspond to Closed Caption

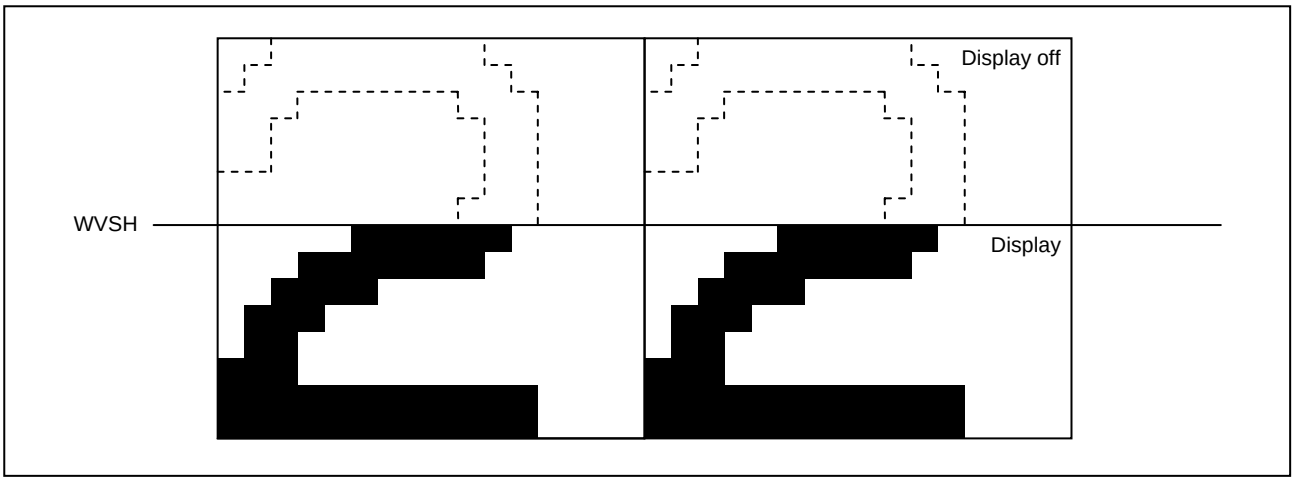



Figure 2.15-21 If WVSH is on a Code Plane

(3) Full-raster blanking function

Full-raster blanking function is used to color the entire background for the display area (TV screen). When using the full-raster blanking function, set YBLCS (Bit2 in ORP6S) to "1", output BL signal from Y/BL pin, because Y signal cannot delete whole display page from video signal.

This function is specified for each display page by setting EXBL (Bit6 in ORRCL) in the OSD register to "1".

Full-raster blanking specification unit: Display page

Full-raster blanking enable register (1 bit) ..... EXBL (Bit6 in ORRCL)

"0" ..... Disable full-raster blanking
"1" ..... Enable full-raster blanking

Full-raster blanking color specification registers (4 bits) ..... RCLI, RCLR, RCLG, RCLB (Bit3 to 0 in ORRCL)

I signal function select: PISEL (Bit6 in ORETC)

"0" ..... 15 colors specification
I pin can be used to make a half level of R, G, B signal (Dark color) through an extra circuit.

"1" ..... 8 colors specification
Contents of RCLI register is disregarded.
I pin can be used as half transparency/half tone through an extra circuit.

Table 2.15.5.6 Raster Plane Color (15 colors)

| RCLI | RCLR | RCLG | RCLB | Raster Plane Color |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Black |
| | 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 |
| 1 | 0 | 0 | 0 | Black |
| | 0 | 0 | 1 | Dark blue |
| | 0 | 1 | 0 | Dark green |
| | 0 | 1 | 1 | Dark cyan |
| | 1 | 0 | 0 | Dark red |
| | 1 | 0 | 1 | Dark magenta |
| | 1 | 1 | 0 | Dark yellow |
| | 1 | 1 | 1 | Gray |

(4) Area plane function

Area plane function is used to display square area to two points on a screen.

Two planes operate independently. They are displayed according to the priority (Area plane 1 > Area plane 2).

See area plane display position setting in section 2.15.5.3 (2) how to set display positions for each area.

Each area plane is set to ON or OFF by AON2 and AON1 (Bit5 and bit4 in ORRCL).

Area plane colors are set by ACLIx, ACLRx, ACLGx, ACLBx (Bit7 to bit0 in ORACL, x: 1, 2).

Area plane colors: 8 or 15

Area plane specification unit: plane

Area plane color specification register (8 bit)

Area plane 1: ACLI1/ACLR1/ACLG1/ACLB1 (Bit3 to 0 in ORACL)

Area plane 2: ACLI2/ACLR2/ACLG2/ACLB2 (Bit7 to 4 in ORACL)

I signal function select: PISEL (Bit6 in ORETC)

"0" ..... 15 colors specification
I pin can be used to make a half level of R, G, B signal (Dark color) through an extra circuit.

"1" ..... 8 colors specification
Contents of ACLI1 and ACLI2 register is disregarded.
I pin can be used as half transparency/half tone through an extra circuit.

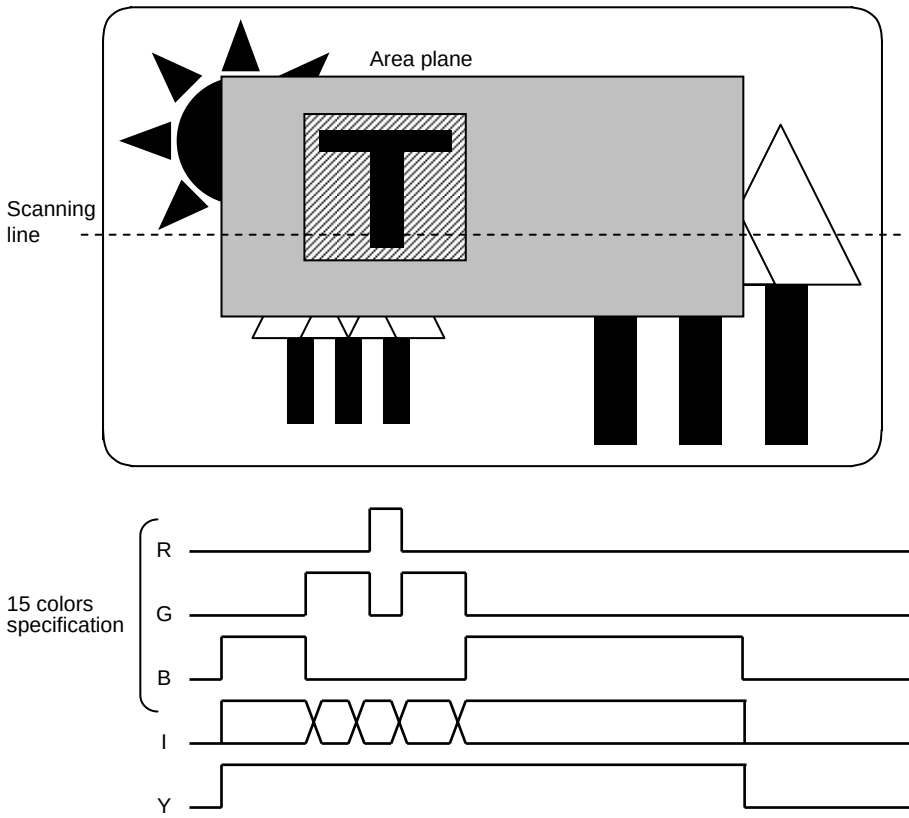
Table 2.15.5.7 Area Plane Color (15 colors)

| ACLIx | ACLRx | ACLGx | ACLBx | Area Plane Color |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Black |
| | 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 |
| 1 | 0 | 0 | 0 | Black |
| | 0 | 0 | 1 | Dark blue |
| | 0 | 1 | 0 | Dark green |
| | 0 | 1 | 1 | Dark cyan |
| | 1 | 0 | 0 | Dark red |
| | 1 | 0 | 1 | Dark magenta |
| | 1 | 1 | 0 | Dark yellow |
| | 1 | 1 | 1 | Gray |

x: 1, 2

I signal function select

1. Using for 15 colors (PISEL = 0)




Example color

| | I = 0 | I = 1 |
|---|---|---|
| Character color | Red | Dark red |
| Character background color | Green | Dark green |
| Area plane color | Blue | Dark blue |

Raster plane: Off
Character background: On
YBLCS: 0 (Y select)

Figure 2.15-22 TV Display and OSD Signals (PISEL = 0)

2. Using for half transparency/half tone (PISEL = 1)

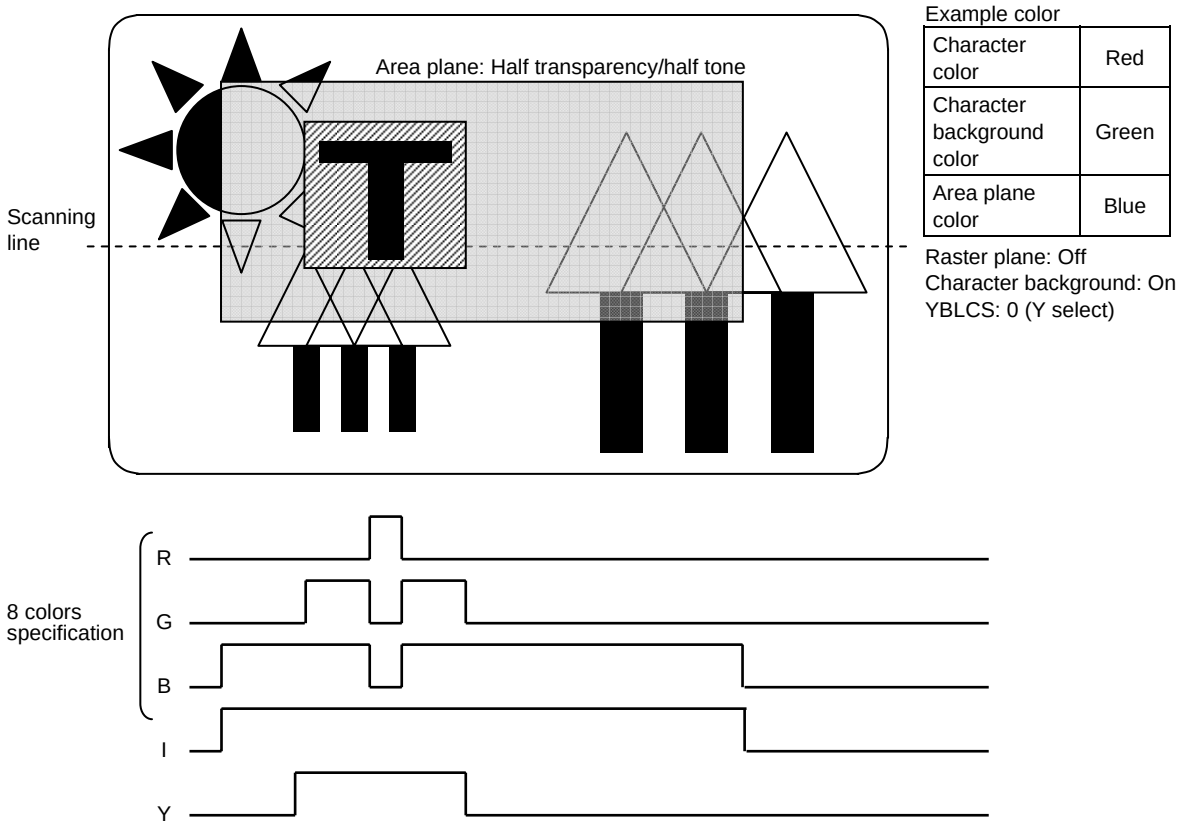



Figure 2.15-23 TV Display and OSD Signals (PISEL = 1)

2.15.5.6 Interrupt Control

(1) Display line counter

The display line counter indicates number of display line (s) by OSD circuit on the TV screen. The display line counter is a 4-bit counter which is initialized to "0" by the falling edge of the $\overline{VD}$ signal and which increments when last scanning of each display line is completed (Falling edge of the $\overline{HD}$ signal). It is necessary to be read out display line counter several times, because it does not synchronize CPU clock.

Display line counter register (4 bits) ... DCTR (Bit3 to 0 in ORIRC)

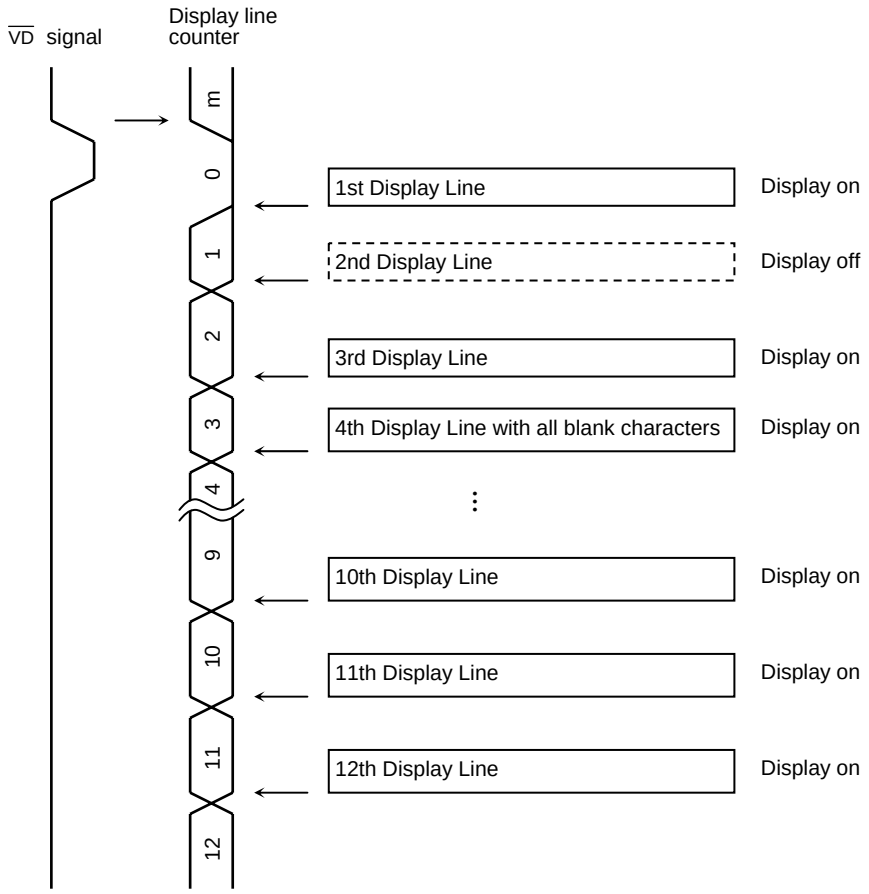
"0000" ..... No display line is completed.

"0001" ..... 1st display line is completed.

"0010" ..... 2nd display line is completed.

⋮ ⋮

"1111" .... 15th display line is completed.




Note 1: The display line counter also increments when a line with all blank characters or a line with display off is specified.

Note 2: When display lines are overlapped each other, previous display line is enabled and next line is canceled. At this time, the display line counter does not increment for canceled line.

Figure 2.15-24 Display Line Counter

(2) Interrupt generator circuit

An interrupt request is generated when a falling edge of $\overline{VD}$ signal or when line counter (DCTR) is counted to the certain value specified by ISDC.

Interrupt source select register (1 bit) ..... SVD (Bit4 in ORIRC)

"0" ..... Interrupt request generated when the display line counter (DCTR) is counted to the certain value which is specified by ISDC.

"1" ..... Interrupt request is generated when a falling edge of $\overline{VD}$ signal.

Interrupt generation line specification register (4 bits) ..... ISDC (Bit3 to 0 in ORIRC)

"0000" ..... Interrupt request generated when the display line counter is cleared.

"0001" ..... Interrupt request generated at end points of the last scanning line of the first display line

"0010" ..... Interrupt request generated at end points of the last scanning line of the 2'nd display line

⋮

"1111" ..... Interrupt request generated at end points of the last scanning line of the 15'th display line

2.15.5.7 Display Memory Access

(1) Display memory

The display memory is accessed for two purposes, one for writing data to the display memory, and one for reading data from the display memory.

Display memory address specification registers .... DMA8 to MDA0 (ORDMA) (9 bits)

Display memory data write registers

Character code write register (9 bits) .... CRA8 to CRA0 (ORCRA)

Character ornamentation data write registers (7 bits) .... SLNT, EUL, BLF, IDT, RDT, GDT, and BDT (ORDSN)

Display memory bank select register MBK (Bit1 in ORETC)

"0" ..... When writing either character code or character ornamentation data

"1" ..... When writing both character code and character ornamentation data

Note 1: These control registers have a characteristic that immediately when a value is written to the register, the content of the register is transferred as valid data to the OSD circuit/display memory.

Note 2: The data written to the display memory takes effect at the same time it is written. When character code or character ornamentation data is written to the display memory while it is displaying some character, the character may not be displayed correctly. When writing data to the display memory, make sure no character is being displayed in the memory location where you are going to write data.

Note 3: When writing data to or reading data from the display memory, do not use two-byte transfer instructions such as "LDW(HL),mn LD rr, (pp)". Otherwise, erroneous data may be written to the display memory or data may be written to an incorrect address.

Note 4: Allow for at least two instruction cycles between a display memory address write instruction and a data write or read instruction. Also, when continuous writing data to or reading data from the display memory, allow for at least two instruction cycles between one write or read instruction and the next. Otherwise, erroneous data may be written to the display memory or data may be written to an incorrect address.

Note 5: When setting display memory addresses, always be sure to write all of 9 address bits sequentially in order of DMA8 and DMA7 to DMA0.

1. Normal mode

In normal mode, the display memory addresses are automatically incremented each time data is read from or written to the memory. Because addresses are automatically incremented, this mode may be used for reading from or writing data to multiple continuous addresses simultaneously.

<Display memory write sequence in normal mode>

a. When writing either character code or character ornamentation data

(1) Set MFYWR, MBK, and RDWRV all to 0.

(2) Write the most significant address bit of the display memory to DMA8. Go on and write the 8 low-order address bits of the display memory to DMA7 to DMA0.

(3) Writing character code or character ornamentation data

- Writing character code

Write the most significant bit of character code to CRA8. Go on and write the 8 low-order bits of character code to CRA7 through CRA0. At this point in time, the 9 bits of character code written are transferred to the display memory, and DMA8 to DMA0 are automatically incremented.

- Writing character ornamentation data

Write character ornamentation data to SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. At this point in time, the character ornamentation data written are transferred to the display memory, and DMA8 to DMA0 are automatically incremented.

(4) To write data (character code or character ornamentation data) to continuous addresses, repeat step (3).

b. When writing character code and character ornamentation data at a time

(1) Set MFYWR to 0, MBK to 1, and RDWRV to 0.

(2) Write the most significant address bit of the display memory to DMA8. Go on and write the 8 low-order address bits of the display memory to DMA7 to DMA0.

(3) Write character ornamentation data to SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. At this point in time, the character ornamentation written are transferred to the display memory.

(4) Write the most significant bit of character code to CRA8. Go on and write the 8 low-order bits of character code to CRA7 to CRA0. At this point in time, the 9 bits of character code written and the character ornamentation data written in step (3) are transferred to the display memory, and DMA8 to DMA0 are automatically incremented.

(5) To write data to continuous addresses, repeat steps (3) and (4).

<Display memory read sequence in normal mode>

a. When reading either character code or character ornamentation data

(1) Set MFYWR to 0, MBK to 0, and RDWRV to 1.

(2) Write the most significant address bit of the display memory to DMA8. Go on and write the 8 low-order address bits of the display memory to DMA7 to DMA0.

(3) Reading character code or character ornamentation data

- Reading character code

  Read the most significant bit of character code to CRA8. Go on and read the 8 low-order bits of character code to CRA7 to CRA0. At this point in time, DMA8 to DMA0 are automatically incremented.

- Reading character ornamentation data

  Read character ornamentation data SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. At this point in time, DMA8 through DMA0 are automatically incremented.

(4) To read data (Character code or character ornamentation data) from continuous addresses, repeat step (3).

b. When reading character code and character ornamentation data at a time

(1) Set MFYWR to 0, MBK to 1, and RDWRV to 1.

(2) Write the most significant address bit of the display memory to DMA8. Go on and write the 8 low-order address bits of the display memory to DMA7 to DMA0.

(3) Read character ornamentation data SLNT, EUL, BLF, IDT, RDT, GDT, and BDT.

(4) Read the most significant bit of character code to CRA8. Read the 8 low-order bits of character code to CRA7 to CRA0. At this point in time, DMA8 to DMA0 are automatically incremented.

(5) To read data from continuous addresses, repeat steps (3) and (4).

2. Read-modify-write mode

When writing data in read-modify-write mode, the display memory addresses are automatically incremented as in normal mode, but when reading data in this mode, the memory addresses are not automatically incremented.

Therefore, immediately after executing a read from some display memory address, you can execute a write to the same display memory address. After executing a write, the display memory addresses are automatically incremented.

a. Reading/writing either character code or character ornamentation data in read-modify-write mode

(1) Set MFYWR to 1 and MBK to 0, and RDWRV to 1.

(2) Write the most significant address bit of the display memory to DMA8. Go on and write the 8 low-order address bits of the display memory to DMA7 to DMA0.

(3) Reading character code or character ornamentation data

- Reading character code

  Read the most significant bit of character code to CRA8. Read the 8 low-order bits of character code to CRA7 to CRA0. DMA8 to DMA0 are not incremented.

- Reading character ornamentation data

  Read character ornamentation data SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. DMA8 to DMA0 are not incremented.

(4) Writing character code or character ornamentation data

- Set RDWRV to "0".
- Writing character code

  Write the most significant bit of character code to CRA8. Go on and write the 8 low-order bits of character code to CRA7 to CRA0. At this point in time, the 9 bits of character code written are transferred to the display memory, and DMA8 to DMA0 are automatically incremented.
- Writing character ornamentation data

  Write character ornamentation data to SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. At this point in time, the character ornamentation data written are transferred to the display memory, and DMA8 to DMA0 are automatically incremented.

(5) To continue executing read-modify-write operations, repeat steps (1) to (4). To read/write data (Character code or character ornamentation data). To continue executing read-modify-write mode from continuous addresses, repeat steps (3) and (4).

b. Reading/writing both character code and character ornamentation data in read-modify-write mode

(1) Set MFYWR to 1, MBK to 1 and RDWRV to 1

(2) Write the most significant address bit of the display memory to DMA8. Go on and write the 8 low-order address bits of the display memory to DMA7 to DMA0.

(3) Read character ornamentation data SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. At this point in time, DMA8 to DMA0 are not incremented.

(4) Read the most significant bit of character code to CRA8. Read the 8 low-order bits of character code to CRA7 to CRA0. At this point in time, DMA8 to DMA0 are not incremented.

(5) Set RDWRV to "0".

(6) Write character ornamentation data to SLNT, EUL, BLF, IDT, RDT, GDT, and BDT. At this point in time, the character ornamentation data written is transferred to the display memory.

(7) Write the most significant bit of character code to CRA8. Go on and write the 8 low-order bits of character code to CRA7 to CRA0. At this point in time, the 9 bits of character code written and the character ornamentation data written in step (6) are transferred to the display memory, and DMA8 to DMA0 are automatically incremented.

(8) To continue executing read-modify-write operations, repeat steps (1) to (7). (To read/write data to and from continuous addresses in read-modify-write mode, repeat steps (3) to (7).)

Table 2.15.5.8 Address Increment

| | | RD (RDWRV = 1) | | WR (RDWRV = 0) | |
|---|---|---|---|---|---|
| | | Character Ornamentation | Character Code | Character Ornamentation | Character Code |
| MFYWR = 0 | MBK = 0 | INC | INC | INC | INC |
| | MBK = 1 | – | INC | – | INC |
| MFYWR = 1 | MBK = 0 | – | – | INC | INC |
| | MBK = 1 | – | – | – | INC |

INC: Automatic address increment at read or write.

–: No address change at data read or write.

Example: Setting a character code (020H) to the display memory (Address: 120H) and setting (001H) for a character ornamentation.

1. MBK = 0

```
; Set display memory address
        LD      (0x25),     0x01      ; ORDMA<DMA8>
        LD      (0x24),     0x20      ; ORDMA<DMA7:0>
; Set character code
        LD      (0x1F),     0x00      ; ORCRA<CRA8>
        LD      (0x1E),     0x20      ; ORCRA<CRA7:0>
; Set display memory address again
        LD      (0x25),     0x01
        LD      (0x24),     0x20
; Set character ornamentation
        LD      (0x1D),     0X01      ; ORDSN<SLNT, ......... BDT>
```

2. MBK = 1

```
; Set display memory address
        LD      (0x25),     0x01
        LD      (0x24),     0x20
; Set character ornamentation
        LD      (0x1D),     0X01
; Set character code
        LD      (0x1F),     0x00
        LD      (0x1E),     0x20
```

Note 1: To write character code into the display memory, first write into register CRA8 and then write into registers CRA7 to CRA0. When data is written into registers CRA7 to CRA0, DMA8 to DMA0 is incremented. It is impossible to write into the display memory for CRA7 to CRA0 alone. If no data is written into register CRA8 while data is written into registers CRA7 to CRA0, the value previously written into register CRA8 is written into the associated display memory.

Note 2: To read character code from the display memory, first read from register CRA8, and then read from registers CRA7 to CRA0. When data is read from registers CRA7 to CRA0, DMA8 to DMA0 is incremented.

Note 3: There should be a time interval of at least two machine cycles between a DMA set instruction and a data write/read instruction. There should be a time interval of at least two machine cycles between a data write instruction and a data read instruction.

(2) Character

Characters: 384 (including blank character)

Character specification register (9 bits) ..... CRA8 to CRA0 (Bit8 to 0 in RCRA)

Character code “000H” ..... Blank character

Character code “001H” to “017FH” ..... User programmable by character ROM

(3) Character color

Character colors: 8 or 15

Character color specification unit: Character

Character color specification register (4 bits): IDT/RDT/GDT/BDT (Bit3 to 0 in ORDSN)

I signal function select: PISEL (Bit6 in ORETC)

"0" ..... 15 colors specification
I pin can be used to make a half level of R, G, B signal (Dark color) through an extra circuit.

"1" ..... 8 colors specification
Contents of IDT register is disregarded.
I pin can be used as half transparency/half tone through an extra circuit.

Table 2.15.5.9 Character Color (15 colors)

| IDT | RDT | GDT | BDT | Character Color |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Black |
| | 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 |
| 1 | 0 | 0 | 0 | Black |
| | 0 | 0 | 1 | Dark blue |
| | 0 | 1 | 0 | Dark green |
| | 0 | 1 | 1 | Dark cyan |
| | 1 | 0 | 0 | Dark red |
| | 1 | 0 | 1 | Dark magenta |
| | 1 | 1 | 0 | Dark yellow |
| | 1 | 1 | 1 | Gray |

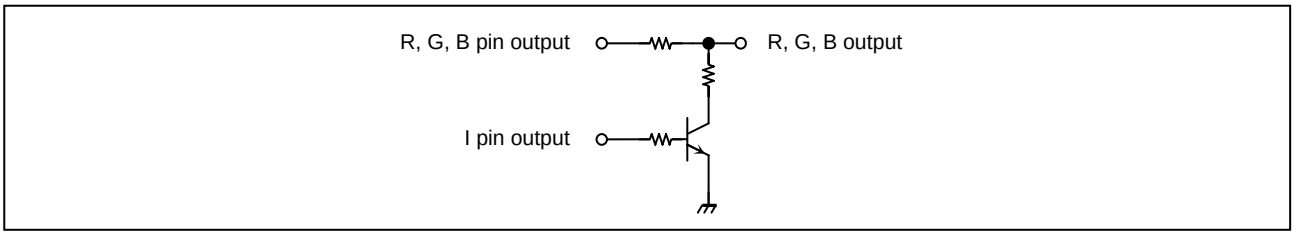



Figure 2.15-25 Example of Circuit for 15 Color by I Pin

(4) Blinking function

Blinking function is used to blink display characters.

When BKMF is "1", characters specified for blinking by BLF are not displayed. (If the background color function is used, the background color is not disappeared.)

Blinking specification unit: Character

Blinking specification register (1 bit) ..... BLF (Bit4 in ORDSN)

"0" ..... No blinking
"1" ..... Blinking

Blinking master specification register (1 bit) ..... BKMF (Bit5 in ORETC)

"0" ..... Disable blinking
"1" ..... Enable blinking (Characters whose BLF are set to "1" are not displayed.)

Note: Regarding the extra dot of the left and/or right character by fringing function, it is not enabled as blink.

(5) 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/line

Underline enable register (Character unit) (1 bit) ..... EUL (Bit5 in ORDSN)

"0" ..... No underline
"1" ..... Underline

Underline enable register (Line unit) (1 bit × 12) ..... EULAn (n: 1 to 8)(OREULA8), EULAn (n: 9 to 12) (OREULA12)

Underline colors: 8 or 15
Underline color specification registers (4 bits) ..... RDT, GDT, BDT, IDT (Bit3 to 0 in ORDSN) (Refer to Table 2.15.5.9)

Note: To use the underline function, set both the underline enable register for underlining text in characters and that for underlining text in lines. If the former register (EUL) only is set, an underline is not displayed.

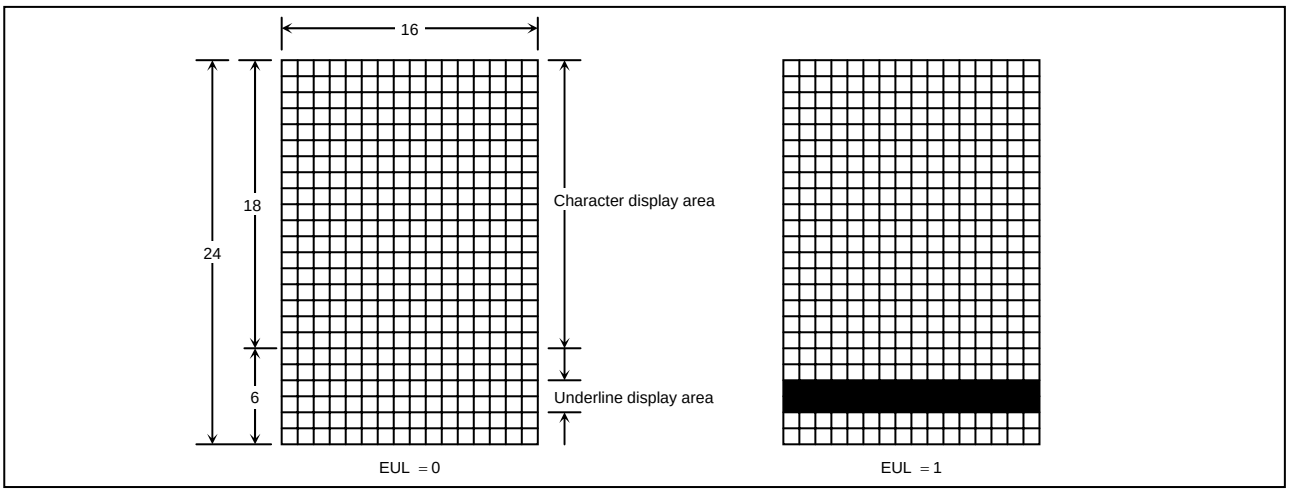



Figure 2.15-26 Underline

(6) 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 (24 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 12 | SOL121 and SOL120 (Bits 7 and 6 in ORSOL12) |

Solid space specification

The solid space control functions as follows:

SOLx1/SOLx0 (x: 1 to 12)

| | | |
|---|---|---|
| "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 (4 bits)

..... IBDT, RBDT, GBDT, BBDT (Bits 3 to 0 in ORBK)
(Same color as that of background)

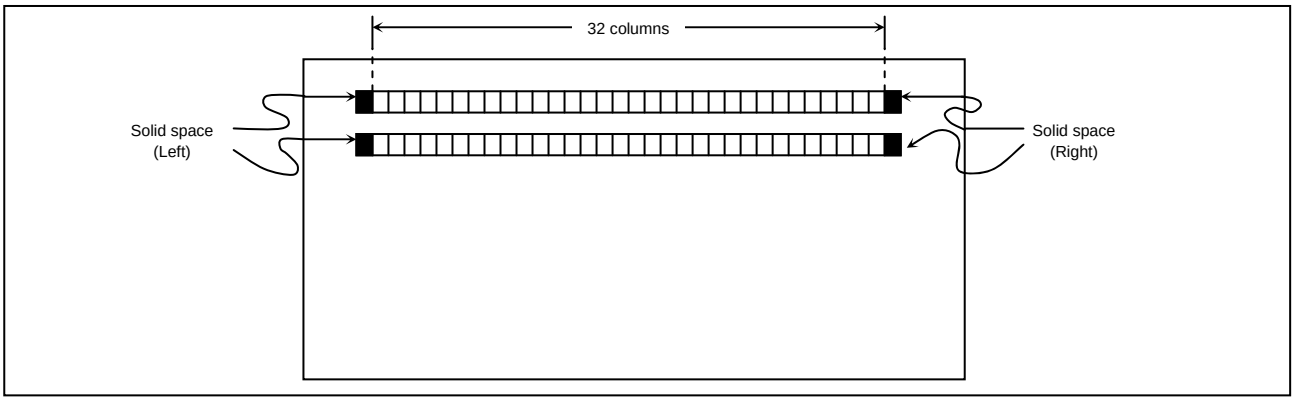



Figure 2.15-27 Solid Space

(7) Slant function

Slant function is used to slant characters for italics.

Slant specification unit: Character

Slant enable register (1 bit) ..... SLNT (Bit6 in ORDSN)

"0" ..... No slant
"1" ..... Slant

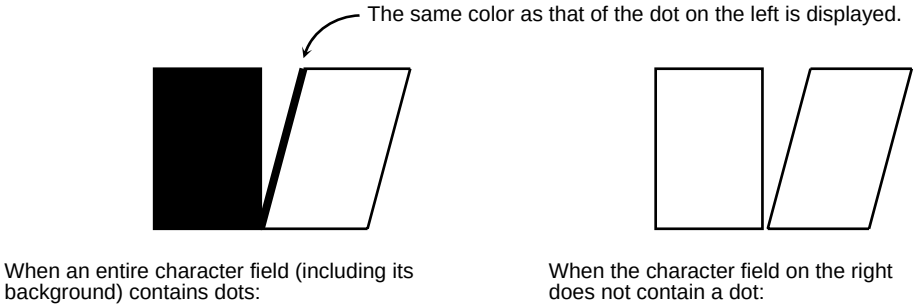
Note 1: SLANT function is enabled each characters, and therefore, in case of using background function, this color of the background is enable as slant. Regarding the extra dots of the left and/or right character by fringing function, it is not enabled as slant.

Note 2: When a character is slanted in an area, which overlaps with the character field, the overlap is also slanted.

Note 3: If slanting a character causes part of the character to get into the character field to the immediate right of the character, then this part is not displayed.

Note 4: To provide closed caption display (CCD), specify black as the background color, and set YBLCS to "1". R, G, B and Y are all slanted. Thus, if the Y signal is selected, a video signal is displayed above and to the left of the slant character.

Note 5: When a character is slanted, the dot data to the immediate left of the character is also slanted.



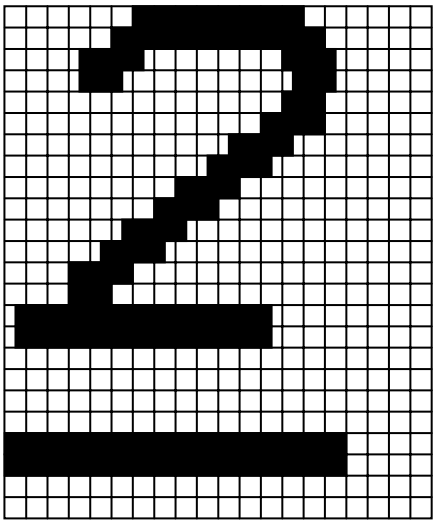


Figure 2.15-28 Slant

#### 2.15.5.8 Clock Generation for OSD Display

The TMP88CS38/CM38A/CP38A has clock generator for OSD display. It can generate a clock from 8 MHz to 24 MHz. However, note that the OSD display clock ($f_{OSC}$) frequency should not exceed multiply-by-1.6 basic clock frequency ($f_C$). (Refer to the clock frequency in "recommended operating conditions".) The frequency of display clock is specified by ORCLKC and is monitored by ORCLKF.

Display clock frequency specification register: ORCLKC (8 bits)

$f_{OSC}$ = ORCLKC × 8/$T_{Hdhigh}$ ($T_{Hdhigh}$: High period of $\overline{HD}$ signal)

Display clock frequency locked monitor: ORCLKF (8 bits)

0: Unmatched

1: Matched

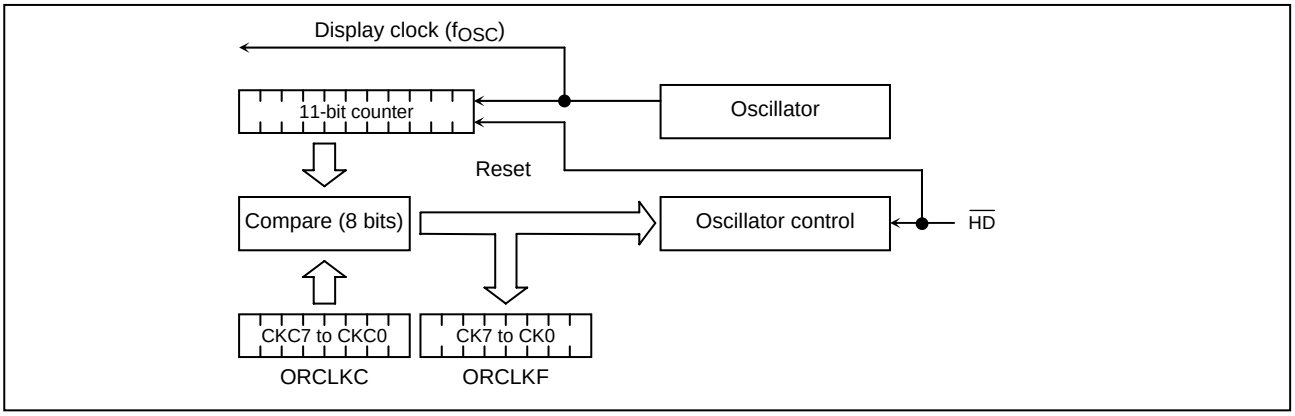



Figure 2.15-29 Clock Generation For OSD Display Control

#### 2.15.5.8.1 Operating Principle

The clock generator for OSD display consists of a ring oscillator, an 11-bit counter, a comparator and an oscillator controller. The frequency of the ring oscillator depends on the cycle of the external $\overline{HD}$ input and the ORCLKC setting. The ring oscillator generates clocks required for one scanning line and the 11-bit counter counts the number of clocks. Then the counted number of the upper-order 8 bits is compared with the value set by ORCLCK. The comparator sends the compared result, which controls the ring oscillator and changes the oscillating frequency. At the frequencies of 8 MHz, 16 MHz and 24 MHz set by ORCLKC, the error percentages are 1.65%, 0.83% and 0.55% respectively. Thus, even when the same value is rewritten to ORCLKC, the display position after refresh deviates by the error percentage (Note 1).

The oscillator controller controls the operation of the ring oscillator using a result from the comparator. To avoid display jitter created by fine adjustment, the oscillator controller receives a coincidence signal and automatically stops adjusting the ring oscillator. After the ring oscillator becomes stable, it drives a display clock at a regular frequency. The output frequency fluctuates according to changes of temperature and voltage (Note 2). The frequency of the ring oscillator can be adjusted to the required frequency (Display position) by writing value to ORCLKC in response to the change of condition.

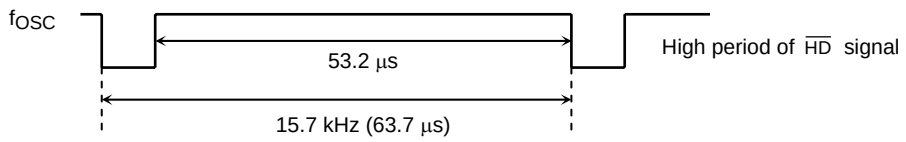
When the oscillating frequency of the ring oscillator is adjusted under software control, please writes value before adjustment to ORCLKC.The comparator compares ORCLKF (the result from comparing the counted number of the upper-order 8 bits of the 11-bit counter with the ORCLKC setting value.) with the ORCLKC setting value.

Comparing ORCLKF with ORCLKC, the deviation of the oscillating frequency is checked and please writes values before adjustment to ORCLKC.

In this case, the deviation of the display position occurs as described in Note 1. Therefore, care must be taken to the interval of adjusting the frequency of the ring oscillator.

Note 1: Since the clock generator for OSD display controls the frequency using the upper-order 8 bits of the 11-bit counter, the low-order 3 bits do not participate in determining the frequency. This yields an error equal to relative magnitude of the low-order 3 bits at maximum. Even when the same value is set to a register, up to 8-count error occurs in one line. (Errors when using clock generator for OSD display in NTSC (interlace) mode (HD = 53.2 μs) are explained subsequently.)

(Example 1) $f_{OSC}$ = 8 MHz




Frequency error when ORCLKC is 35H (at the 8 MHz setting).

When the low-order 3 bits are 000, the value of the 11-bit counter is 424 (1A8), so the frequency is 7.97 MHz.

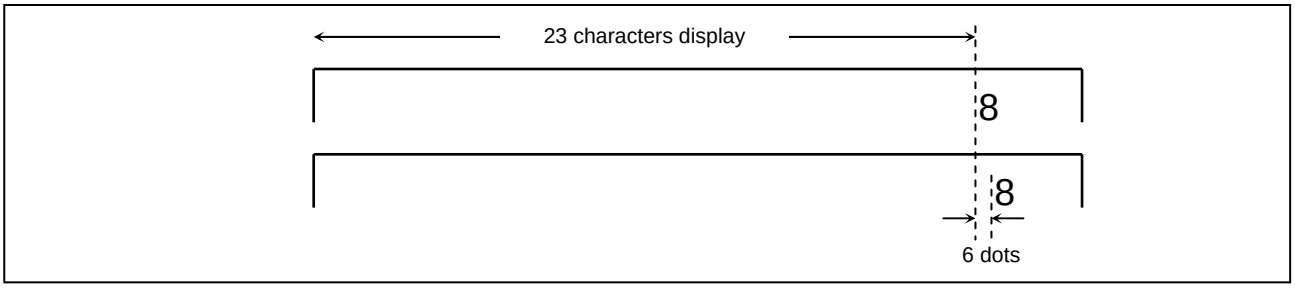
When the low-order 3 bits are 111, the value of the 11-bit counter is 431 (1AF), so the frequency is 8.10 MHz.

1 dot in the errors above is 2 ns (max).

The effect of the 24th character in the display.

Display position error: 16 dots × (24th character – 1) × 1 dot error = 16 × 23 × 2 = 736 ns.

This is approximately equivalent to 6 dot error for the character display.




(Example 2) $f_{OSC}$ = 16 MHz

Frequency error when ORCLKC is 6AH (at the 16 MHz setting).

When the low-order 3 bits are 000, the value of the 11-bit counter is 848 (350), so the frequency is 15.94 MHz.

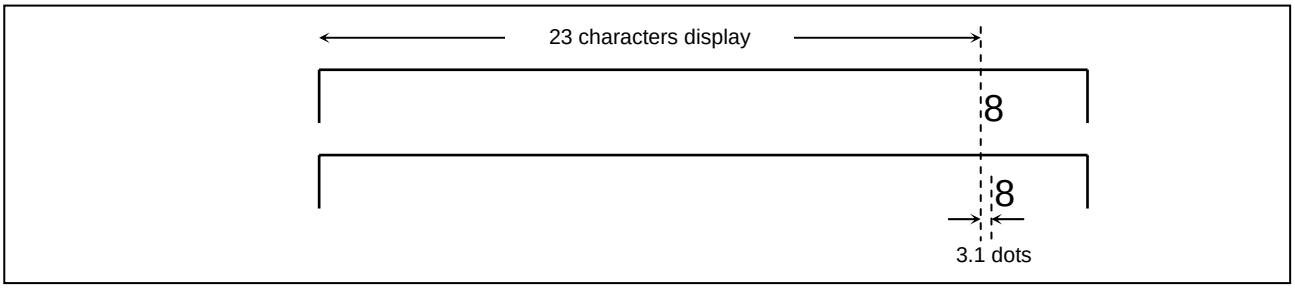
When the low-order 3 bits are 111, the value of the 11-bit counter is 855 (357), so the frequency is 16.07 MHz.

1 dot in the errors above is 0.52 (max).

The effect of the 24th character in the display.

Display position error: 16 dots × (24th character – 1) × 1 dot error = 16 × 23 × 0.52 = 191.4 ns.

This is approximately equivalent to 3.1 dot error for the character display.



Note 1: When a character is displayed in the right side towards the screen, the display position deviates significantly by number of clocks from the rising edge of the $\overline{HD}$ signal × error delay time.

Note 2: After the operation of the ring oscillator becomes stable, the cycle of the display clock changes according to temperature and voltage.The percentage of oscillator clock period by temperature fluctuation is up to 38% within the usage temperature range.

(Set the frequency of the OSD oscillator at Ta = −30°C and change the temperature to 70°C.)

The percentage of oscillator clock period by voltage fluctuation is up to 19% within the usage voltage range.

(Set the frequency of the OSD oscillator at OVDD = 4.5 V and change the voltage to 5.5 V.)

*) Please set the power supply voltage (OVDD) of the clock generator for OSD display to the same value as the system power supply voltage (VDD). It is not recommended that only the OVDD be changed intentionally. (It will affect the internal operation.) When the power supply voltage (OVDD) fluctuates for some reasons, it changes the OSD display position. Thus, make sure that the voltage does not fluctuate during operation.

When voltage and temperature fluctuate simultaneously, both errors overlaps each other.

The percentage of the both fluctuations is up to 57% (38% + 19%) within the usage range.

Frequency adjustment is performed for the display clock frequency every time when value is written to ORCLKC. Therefore, please monitor the ORCLKF register periodically and check if a deviation occurs. When it happens, the display position can be corrected when the value before adjustment is written to the ORCLKC register under software control. However, an error as described in Note 1 occurs when the display position is corrected.

2.15.5.8.2 Notes on Creating a Software Program

(1) If you want to change the OSD display screen, rewrite the same value to ORCLKC for adjustment of . the oscillating frequency.

Example) When the OSD display screen is ON.
When the content of the OSD screen is all changed.
When OFF command is received while CCD is displayed.

(2) When the content of the OSD screen is displayed for a long time, the OSD display position deviates because the oscillating frequency changes according to the power supply voltage and temperature characteristics. To adjust the deviation, monitor the oscillating frequency error using ORCLKF, then rewrite the value before adjustment to ORCLKC during OSD screen off.

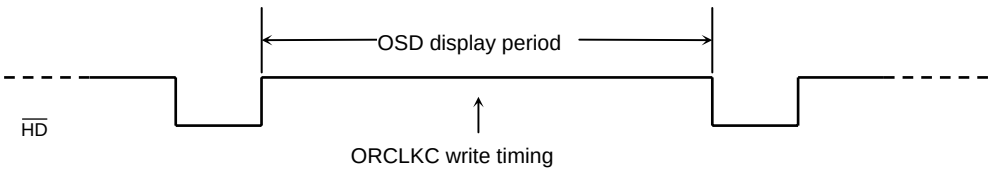
In this case, the OSD display position deviates (Refer to Note 1), please take the timing of writing a setting value to ORCLKC into consideration.

(3) The clock generator for OSD display does not stop oscillating even in STOP mode. If you want to stop it, write 00H to ORCLKC.

By reading ORCLKF, the status of a display clock frequency can be monitored. When the read value is FFH, it turns out that the display clock frequency is in a setting value. However, depending on temperature, power supply voltage, and OSD oscillation frequency, the lower 3 bits of ORCLKF may not be in agreement (the maximum difference of the lower 3 bits value is “5”), then the display clock frequency may become lower than a setting value. Please setup the oscillation frequency which has a margin in the composition of a display screen in consideration of these characteristics.

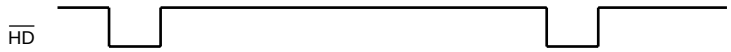
The period from the ORCLKC writing to the completion of a setting of a display clock frequency is required the time of 256 scanning lines at maximum.

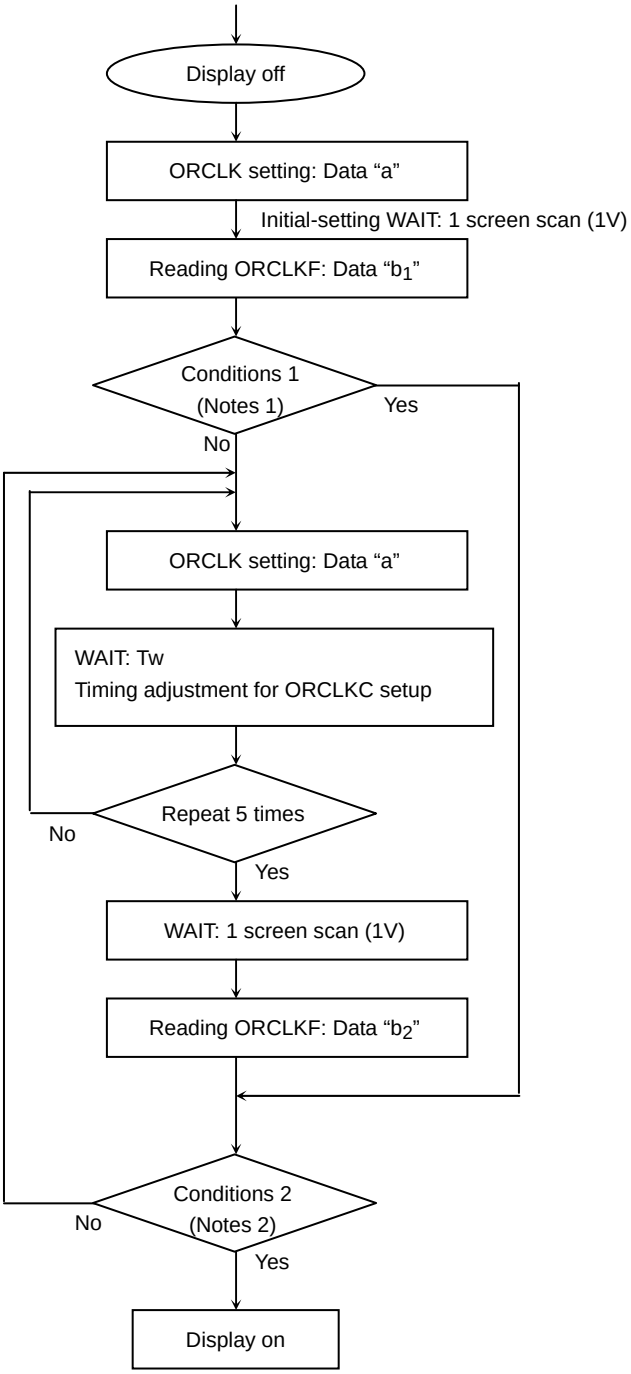
When the ORCLKC is resetup in order to change OSD oscillation frequency, OSD oscillation frequency may become higher rapidly. In order to slip out of this state, a resetup of ORCLKC is required. Please repeat a resetup of ORCLKC until the higher 5 bits of the ORCLKF or more bits are in agreement. Please write in the ORCLKC at timing of the OSD display period (near the center of the OSD display period is recommended) of a horizontal synchronizing signal, or please write in the ORCLKC two or more times.



## The example of software of an OSD clock setup

<Condition> Please execute this flow during OSD screen OFF.
The horizontal synchronizing signal: $\overline{HD}$ has described as the active Low.




Term explanation in a flow chart

Data “a”
8-bit data written in ORCLKC
Data “$b_1$”, ”$b_2$”
8-bit data which read ORCLKF
Tw: (THDx2) x2/3
THD is the cycle of a horizontal synchronizing signal ( $\overline{HD}$ ).
Example: When THD is 64 microseconds, Tw recommends about 90 microseconds.

“WAIT” after the 5 times writing of ORCLKC “1V” is about 16ms (64microsecond x 256) in the case of the interlace display.

Note 1: Condition 1

By calculating XNOR of data “a” and data “$b_1$”, the setting value inside the OSD oscillator can be calculated.

The setting value inside the OSD oscillator: (Data “$c_1$”) = data “a” XNOR data “$b_1$”

In the case so that data “$c_1$” may become extremely large to data “a” ((“$c_1$” − “a”) > 5), please move on to the flow which writes in ORCLKC 5 times.

* In the case so that data “$c_1$” may become extremely large, data “$c_1$” is more than E0h (when the high width of $\overline{HD}$ is 60 microseconds, OSD oscillation frequency becomes 30 MHz or more.).

Note 2: Condition 2

By calculating XNOR of data “a” and data “$b_i$” (i = 1, 2), the setting value inside the OSD oscillator can be calculated.

The setting value inside the OSD oscillator: (Data “$c_i$”) = data “a” XNOR data “$b_i$”

* From the difference of data “$c_i$” and data “a”, it can judge whether it is in the error range. Please set the error range to a maximum of ±5.

### 2.15.5.9 OSD Control Registers

Can not access all OSD control registers in any of read-modify-write instructions such as bit operation, etc.

| 0RHS1 (00F81H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | HS17 | HS16 | HS15 | HS14 | HS13 | HS12 | HS11 | HS10 | (Initial value: 0000 0000) |

| | | |
|---|---|---|
| | Horizontal display start position specification | Write only |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| ORVS1 (00F82H) | VS17 | VS16 | VS15 | VS14 | VS13 | VS12 | VS11 | VS10 | (Initial value: 0000 0000) |
| (00F83H) | – | – | – | – | – | – | – | VS18 | (Initial value: **** ***0) |
| ORVS2 (00F84H) | VS27 | VS26 | VS25 | VS24 | VS23 | VS22 | VS21 | VS20 | (Initial value: 0000 0000) |
| (00F85H) | – | – | – | – | – | – | – | VS28 | (Initial value: **** ***0) |
| ORVS3 (00F86H) | VS37 | VS36 | VS35 | VS34 | VS33 | VS32 | VS31 | VS30 | (Initial value: 0000 0000) |
| (00F87H) | – | – | – | – | – | – | – | VS38 | (Initial value: **** ***0) |
| ORVS4 (00F88H) | VS47 | VS46 | VS45 | VS44 | VS43 | VS42 | VS41 | VS40 | (Initial value: 0000 0000) |
| (00F89H) | – | – | – | – | – | – | – | VS48 | (Initial value: **** ***0) |
| ORVS5 (00F8AH) | VS57 | VS56 | VS55 | VS54 | VS53 | VS52 | VS51 | VS50 | (Initial value: 0000 0000) |
| (00F8BH) | – | – | – | – | – | – | – | VS58 | (Initial value: **** ***0) |
| ORVS6 (00F8CH) | VS67 | VS66 | VS65 | VS64 | VS63 | VS62 | VS61 | VS60 | (Initial value: 0000 0000) |
| (00F8DH) | – | – | – | – | – | – | – | VS68 | (Initial value: **** ***0) |
| ORVS7 (00F8EH) | VS77 | VS76 | VS75 | VS74 | VS73 | VS72 | VS71 | VS70 | (Initial value: 0000 0000) |
| (00F8FH) | – | – | – | – | – | – | – | VS78 | (Initial value: **** ***0) |
| ORVS8 (00F90H) | VS87 | VS86 | VS85 | VS84 | VS83 | VS82 | VS81 | VS80 | (Initial value: 0000 0000) |
| (00F91H) | – | – | – | – | – | – | – | VS88 | (Initial value: **** ***0) |
| ORVS9 (00F92H) | VS97 | VS96 | VS95 | VS94 | VS93 | VS92 | VS91 | VS90 | (Initial value: 0000 0000) |
| (00F93H) | – | – | – | – | – | – | – | VS98 | (Initial value: **** ***0) |
| ORVS10 (00F94H) | VS107 | VS106 | VS105 | VS104 | VS103 | VS102 | VS101 | VS100 | (Initial value: 0000 0000) |
| (00F95H) | – | – | – | – | – | – | – | VS108 | (Initial value: **** ***0) |
| ORVS11 (00F96H) | VS117 | VS116 | VS115 | VS114 | VS113 | VS112 | VS111 | VS110 | (Initial value: 0000 0000) |
| (00F97H) | – | – | – | – | – | – | – | VS118 | (Initial value: **** ***0) |
| ORVS12 (00F98H) | VS127 | VS126 | VS125 | VS124 | VS123 | VS122 | VS121 | VS120 | (Initial value: 0000 0000) |
| (00F99H) | – | – | – | – | – | – | – | VS128 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| VSn8 to VSn0 | Vertical display start position for line n | Write only |

(n: 1 to 12)

Note 1: If display lines are overlapped each other, previous display line is enabled and next line is disabled. Set the vertical display start position not to overlap display lines.

Note 2: Transfer the contents of vertical display start position registers into OSD circuit before a position of the scanning line coincides with their own vertical display start position.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| ORCS4 (00F9AH) | CS4 | | CS3 | | CS2 | | CS1 | | (Initial value: 0000 0000) |
| ORCS8 (00F9BH) | CS8 | | CS7 | | CS6 | | CS5 | | (Initial value: 0000 0000) |
| ORCS12 (00F9CH) | CS12 | | CS11 | | CS10 | | CS9 | | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| CSn | Character size and display on/off for line n | 00: Display off<br>01: Large size<br>10: Middle size<br>11: Small size | Write only |

(n: 1 to 12)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| OREULA8 (00F9DH) | EULA8 | EULA7 | EULA6 | EULA5 | EULA4 | EULA3 | EULA2 | EULA1 | (Initial value: 0000 0000) |
| OREULA12 (00F9EH) | – | – | – | – | EULA12 | EULA11 | EULA10 | EULA9 | (Initial value: **** 0000) |

| | | | |
|---|---|---|---|
| EULAn | Underline for display line for line n | 0: Display off<br>1: Display on | |

(n: 1 to 12)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| OREFR8 (00F9FH) | EFR8 | EFR7 | EFR6 | EFR5 | EFR4 | EFR3 | EFR2 | EFR1 | (Initial value: 0000 0000) |
| OREFR12 (00FA0H) | – | – | – | – | EFR12 | EFR11 | EFR10 | EFR9 | (Initial value: **** 0000) |

| | | | |
|---|---|---|---|
| EFRn | Fringing enable specification register for line n | 0: Disable fringing<br>1: Enable fringing | Write only |

(n: 1 to 12)

Note: When a display line is enabled fringing function, its vertical size is increased by one dot (by two dots when its character size is small) independent of its character font. Therefore, when a vertical display start position is specified to no space between the lines, the display line which is overlapped with increasing dot(s) is canceled.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| ORCLKF (00FA1H) | CK7 | CK6 | CK5 | CK4 | CK3 | CK2 | CK1 | CK0 | (Initial value: 0000 0000) |
| ORCLKC (00FA1H) | CKC7 | CKC6 | CKC5 | CKC4 | CKC3 | CKC2 | CKC1 | CKC0 | (Initial value: 0000 0000) |

| | | |
|---|---|---|
| CKn | Display clock frequency locked monitor | Read only |
| CKCn | Display clock frequency specification register | Write only |

(n: 0 to 7)

| | | | | | |
|---|---|---|---|---|---|
| ORSLO4 (00FA2H) | SLO4 | SLO3 | SLO2 | SLO1 | (Initial value: 0000 0000) |
| ORSLO8 (00FA3H) | SLO8 | SLO7 | SLO6 | SLO5 | (Initial value: 0000 0000) |
| ORSLO12 (00FA4H) | SLO12 | SLO11 | SLO10 | SLO9 | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| SLOn | Solid space for line n | 00: No solid space display<br>01: Solid space display left<br>10: Solid space display right<br>11: Solid space display left and right | Write only |

(n: 0 to 12)

| ORBK | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00FA5H) | IBDT | RBDT | GBDT | BBDT | IFDT | RFDT | GFDT | BFDT | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| IBDT/<br>RBDT/<br>GBDT/<br>BBDT | Background color select | 0000: Black<br>0001: Blue<br>0010: Green<br>0011: Cyan<br>0100: Red<br>0101: Magenta<br>0110: Yellow<br>0111: White<br>1000: Black<br>1001: Dark blue<br>1010: Dark green<br>1011: Dark cyan<br>1100: Dark red<br>1101: Dark magenta<br>1110: Dark yellow<br>1111: Gray | Write only |
| IFDT/<br>RFDT/<br>GFDT/<br>BFDT | Fringing color select | 0000: Black<br>0001: Blue<br>0010: Green<br>0011: Cyan<br>0100: Red<br>0101: Magenta<br>0110: Yellow<br>0111: White<br>1000: Black<br>1001: Dark blue<br>1010: Dark green<br>1011: Dark cyan<br>1100: Dark red<br>1101: Dark magenta<br>1110: Dark yellow<br>1111: Gray | |

Note: Set IBDT and IFDT to 1 when PISEL (Bit6 in ORETC) sets to 1. Then background color select and fringing color select are 8 variety.

| ORACL | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00FA6H) | ACLI2 | ACLR2 | ACLG2 | ACLB2 | ACLI1 | ACLR1 | ACLG1 | ACLB1 | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| ACLI2/<br>ACLR2/<br>ACLG2/<br>ACLB2 | Area 2 plane color select | 0000: Black<br>0001: Blue<br>0010: Green<br>0011: Cyan<br>0100: Red<br>0101: Magenta<br>0110: Yellow<br>0111: White<br>1000: Black<br>1001: Dark blue<br>1010: Dark green<br>1011: Dark cyan<br>1100: Dark red<br>1101: Dark magenta<br>1110: Dark yellow<br>1111: Gray | Write only |
| ACLI1/<br>ACLR1/<br>ACLG1/<br>ACLB1 | Area 1 plane color select | 0000: Black<br>0001: Blue<br>0010: Green<br>0011: Cyan<br>0100: Red<br>0101: Magenta<br>0110: Yellow<br>0111: White<br>1000: Black<br>1001: Dark blue<br>1010: Dark green<br>1011: Dark cyan<br>1100: Dark red<br>1101: Dark magenta<br>1110: Dark yellow<br>1111: Gray | |
| ACLI2 | | 0: Not assign half transparency for area 2 plane<br>1: Assign half transparency for area 2 plane | |
| ACLI1 | | 0: Not assign half transparency for area 1 plane<br>1: Assign half transparency for area 1 plane | |

Note: Set ACLI2 and ACLI1 to 1 when PISEL (Bit6 in ORETC) sets to 1. Then area 2 plane color select and area 1 plane color select are 8 variety.

ORIV (00FBBH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VDPOL | HDPOL | YBLII | RGBII | YIV | BLIV | RGBIV | IIV |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| VDPOL | $\overline{VD}$ input polarity select | 0: Non-invert input signal<br>1: Invert input signal | Write only |
| HDPOL | $\overline{HD}$ input polarity select | 0: Non-invert input signal<br>1: Invert input signal | |
| YBLII | Y/BLIN input polarity select | 0: Active high<br>1: Active low | |
| RGBII | RIN, GIN, BIN input polarity select | 0: Active high<br>1: Active low | |
| YIV | Y output polarity select | 0: Active high<br>1: Active low | |
| BLIV | BL output polarity select | 0: Active high<br>1: Active low | |
| RGBIV | R, G, B output polarity select | 0: Active high<br>1: Active low | |
| IIV | I output polarity select | 0: Active high<br>1: Active low | |

ORDMA

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00024H) | DMA7 | DMA6 | DMA5 | DMA4 | DMA3 | DMA2 | DMA1 | DMA0 | (Initial value: 0000 0000) |
| (00025H) | – | – | – | – | – | – | – | DMA8 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| DMAn | Display memory address | Write only |

(n: 0 to 8)

Note: It necessary to write all bits of display memory address, writng DMA7 to DMA0 after DMA8, when writing display address.

ORDSN (0001DH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | SLNT | EUL | BLF | IDT | RDT | GDT | BDT |

(Initial value: **** ****)

| | | | |
|---|---|---|---|
| SLNT | Slant enable specification register | 0: Disable slant<br>1: Enable slant | Read/ Write |
| EUL | Underline enable specification register | 0: Disable underline<br>1: Enable underline | |
| BLF | Blinking enable specification register | 0: Disable blinking<br>1: Enable blinking | |
| IDT/ RDT/ GDT/ BDT | Character color select | 0000: Black<br>0001: Blue<br>0010: Green<br>0011: Cyan<br>0100: Red<br>0101: Magenta<br>0110: Yellow<br>0111: White<br>1000: Black<br>1001: Dark blue<br>1010: Dark green<br>1011: Dark cyan<br>1100: Dark red<br>1101: Dark magenta<br>1110: Dark yellow<br>1111: Gray | |

Note: Set IDT to 1 when PISEL (Bit6 in ORETC) sets to 1. Then character color select is 8 variety.

| ORCRA | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (0001EH) | CRA7 | CRA6 | CRA5 | CRA4 | CRA3 | CRA2 | CRA1 | CRA0 | (Initial value: **** ****) |
| (0001FH) | – | – | – | – | – | – | – | CRA8 | (Initial value: **** ****) |

| | | |
|---|---|---|
| CRAn | Character code | Read/ Write |

(n: 0 to 8)

Note: Write or read CRA7 to CRA0 after write or read CRA8.

| ORWVSH | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00FBCH) | WVSH7 | WVSH6 | WVSH5 | WVSH4 | WVSH3 | WVSH2 | WVSH1 | WVSH0 | (Initial value: 0000 0000) |
| (00FBDH) | – | – | – | – | – | – | – | WVSH8 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| WVSLn | Window upper limit position | Write only |

(n: 0 to 8)

| ORWVSL | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00FBEH) | WVSL7 | WVSL6 | WVSL5 | WVSL4 | WVSL3 | WVSL2 | WVSL1 | WVSL0 | (Initial value: 0000 0000) |
| (00FBFH) | – | – | – | – | – | – | – | WVSL8 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| WVSLn | Window lower limit position | Write only |

(n: 0 to 8)

| ORDON | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00F80H) | – | – | – | – | – | RGWR | EWDW | DON | (Initial value: **** *000) |

| | | | |
|---|---|---|---|
| RGWR | Written data transfer control | 0: (Initial state)<br>1: Transfers written data to OSD circuit (after transfer, RGWR is reset to 0) | Read/ Write |
| EWDW | Window enable specification register | 0: Disable window function<br>1: Enable window function | |
| DON | Display on/off select | 0: Disable display<br>1: Enable display | |

| ORRCL | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (00FA7H) | EBKGD | EXBL | AON2 | AON1 | RCLI | RCLR | RCLG | RCLB | (Initial value: 0000 0000) |

| | | | |
|---|---|---|---|
| EBKGD | Background function enable specification register | 0: No background function<br>1: Background function enable | Write only |
| EXBL | Full-raster blanking enable specification register | 0: No Full-raster blanking<br>1: Full-raster blanking | |
| AON2 | Area 2 plane display enable specification register | 0: No area 2 plane display<br>1: Area 2 plane display enable | |
| AON1 | Area 1 plane display enable specification register | 0: No area 1 plane display<br>1: Area 1 plane display enable | |
| RCLI<br>RCLR/<br>RCLG/<br>RCLB | Raster plane color select | 0000: Black<br>0001: Blue<br>0010: Green<br>0011: Cyan<br>0100: Red<br>0101: Magenta<br>0110: Yellow<br>0111: White<br>1000: Black<br>1001: Dark blue<br>1010: Dark green<br>1011: Dark cyan<br>1100: Dark red<br>1101: Dark magenta<br>1110: Dark yellow<br>1111: Gray | |

Note: Set RCLI to 1 when PISEL (Bit6 in ORETC) sets to 1. Then transfer plane select is 8 variety.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| ORAHS1 (00FA8H) | AHS17 | AHS16 | AHS15 | AHS14 | AHS13 | AHS12 | AHS11 | AHS10 | (Initial value: 0000 0000) |
| (00FA9H) | – | – | – | – | – | – | – | AHS18 | (Initial value: **** ***0) |
| ORAHE1 (00FAAH) | AHE17 | AHE16 | AHE15 | AHE14 | AHE13 | AHE12 | AHE11 | AHE10 | (Initial value: 0000 0000) |
| (00FABH) | – | – | – | – | – | – | – | AHE18 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| AHS1n | Horizontal start point for area 1 plane | Write only |
| AHE1n | Horizontal end point for area 1 plane | |

(n: 0 to 8)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| ORAVS1 (00FACH) | AVS17 | AVS16 | AVS15 | AVS14 | AVS13 | AVS12 | AVS11 | AVS10 | (Initial value: 0000 0000) |
| (00FADH) | – | – | – | – | – | – | – | AVS18 | (Initial value: **** ***0) |
| ORAVE1 (00FAEH) | AVE17 | AVE16 | AVE15 | AVE14 | AVE13 | AVE12 | AVE11 | AVE10 | (Initial value: 0000 0000) |
| (00FAFH) | – | – | – | – | – | – | – | AVE18 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| AVS1n | Vertical start point for area 1 plane | Write only |
| AVE1n | Vertical end point for area 1 plane | |

(n: 0 to 8)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| ORAHS2 (00FB0H) | AHS27 | AHS26 | AHS25 | AHS24 | AHS23 | AHS22 | AHS21 | AHS20 | |
| (00FB1H) | – | – | – | – | – | – | – | AHS28 | |
| ORAHE2 (00FB2H) | AHE27 | AHE26 | AHE25 | AHE24 | AHE23 | AHE22 | AHE21 | AHE20 | (Initial value: 0000 0000) |
| (00FB3H) | – | – | – | – | – | – | – | AHE28 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| AHS2n | Horizontal start point for area 2 plane | Write only |
| AHE2n | Horizontal end point for area 2 plane | |

(n: 0 to 8)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| ORAVS2 (00FB4H) | AVS27 | AVS26 | AVS25 | AVS24 | AVS23 | AVS22 | AVS21 | AVS20 | (Initial value: 0000 0000) |
| (00FB5H) | – | – | – | – | – | – | – | AVS28 | (Initial value: **** ***0) |
| ORAVE2 (00FB6H) | AVE27 | AVE26 | AVE25 | AVE24 | AVE23 | AVE22 | AVE21 | AVE20 | (Initial value: 0000 0000) |
| (00FB7H) | – | – | – | – | – | – | – | AVE28 | (Initial value: **** ***0) |

| | | |
|---|---|---|
| AVS2n | Vertical start point for area 2 plane | Write only |
| AVE2n | Vertical end point for area 2 plane | |

(n: 0 to 8)

ORP6S (00FBAH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P67S | P66S | P65S | P64S | PIDS | YBLCS | MPXS | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P67S to P64S | P6 port output select | 0: R, G, B, Y/BL signal output<br>1: P67 to P64 port output | Write only |
| PIDS | I pin output select | 0: I signal output<br>1: P57 port output | |
| YBLCS | Y/BL signal select | 0: Y signal output<br>1: BL signal output | |
| MPXS | R, G, B, Y/BL signal select | 00: Simultaneous output (Signal from the OSD circuit has higher priority)<br>01: Output of signal from internal OSD circuit<br>10: Output of signal from externally input<br>11: Simultaneous output (Externally input signal has higher priority) | |

ORETC (00FB8H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VDSMD | PISEL | BKMF | ESMZ | “0” | MFYWR | MBK | RDWRV |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| VDSMD | Scan mode select | 0: Normal mode<br>1: Double scan mode | Write only |
| PISEL | I pin function select | 0: 15 colors<br>1: Half transparency/half tone | |
| BKMF | Blinking master | 0: Double blinking<br>1: Enable blinking | |
| ESMZ | Smoothing enable specification register | 0: Disable smoothing<br>1: Enable smoothing | |
| MFYWR | Display memory read mode select | 0: Normal mode<br>1: Read-modify-write mode | |
| MBK | Display memory bank switching | 0: Access to either character code or character display options<br>1: Access both character display option and character code | |
| RDWRV | Read/write mode select | 0: Data write mode for display memory<br>1: Data read mode for display memory | |

Note: Clear “0” to Bit3 in ORETC.

ORIRC (00FB9H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | SDV | ISDC | | | |

(Initial value: ***0 0000)

| | | | |
|---|---|---|---|
| SVD | Interrupt source select | 0: Interrupt request by ISDC value<br>1: Interrupt request at falling edge of $\overline{VD}$ signal | Write only |
| ISDC | Interrupt generation line select | When the line display of the ISDC value ends (with the falling edge of $\overline{HD}$ signal)<br>while SVD = 0, interrupt request is generated.<br>0000: Request interrupt when display of low-order 4 bits "0000" of DCTR ends.<br>0001: Low-order 4 bits "0001" of DCTR<br>0010: Low-order 4 bits "0010" of DCTR<br>0011: Low-order 4 bits "0011" of DCTR<br>0100: Low-order 4 bits "0100" of DCTR<br>0101: Low-order 4 bits "0101" of DCTR<br>0110: Low-order 4 bits "0110" of DCTR<br>0111: Low-order 4 bits "0111" of DCTR<br>1000: Low-order 4 bits "1000" of DCTR<br>1001: Low-order 4 bits "1001" of DCTR<br>1010: Low-order 4 bits "1010" of DCTR<br>1011: Low-order 4 bits "1011" of DCTR<br>1100: Low-order 4 bits "1100" of DCTR<br>1101: Low-order 4 bits "1101" of DCTR<br>1110: Low-order 4 bits "1110" of DCTR<br>1111: Low-order 4 bits "1111" of DCTR | |

ORIRC (00FB9H)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | DCTR |

(Initial value: **** 0000)

| | | | |
|---|---|---|---|
| DCTR | Display line counter | 0000: No line display or when the display of the 16th line ends.<br>0001: 1st line display ends.<br>0010: 2nd line display ends.<br>0011: 3rd line display ends.<br>0100: 4th line display ends.<br>0101: 5th line display ends.<br>0110: 6th line display ends.<br>0111: 7th line display ends.<br>1000: 8th line display ends.<br>1001: 9th line display ends.<br>1010: 10th line display ends.<br>1011: 11th line display ends.<br>1100: 12th line display ends.<br>1101: 13th line display ends.<br>1110: 14th line display ends.<br>1111: 15th line display ends. | Read only |

Note 1: The display line counter also increments when a line with all blank data or a line with display off is specified. If display lines are overlapped each other, previous display line is enabled and next line is disabled. At this time, the display line counter for canceled line does not increment.

Note 2: *: Don't care.

Note 3: All OSD control registers cannot use the read-modify-write instructions. (Bit manipulation instructions such as SET, CLR, etc. and logical operation such as AND, OR, etc.)

Figure 2.15-30 OSD Control Register List (1/2)

| Register Address | Register Name | Register Bit Configuration | | | | | | | | Bit Contents |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |
| 00F81 | ORHS1 | HS17 | HS16 | HS15 | HS14 | HS13 | HS12 | HS11 | HS10 | HS17 to 10: Code horizontal display base position setting |
| 00F82, 00F83 to 00F98, 00F99 | ORVSn | VSn7 | VSn6 | VSn5 | VSn4 | VSn3 | VSn2 | VSn1 | VSn0 | VSn8 to 0: Code vertical display position setting (n: 0 to 12) |
| | | – | – | – | – | – | – | – | VSn8 | |
| 00F9A | ORCS4 | CS4 | | CS3 | | CS2 | | CS1 | | CSn: Character size (n: 1 to 12)<br>00: Display off 10: Middle size<br>01: Large size 11: Small size |
| 00F9B | ORCS8 | CS8 | | CS7 | | CS6 | | CS5 | | |
| 00F9C | ORCS12 | CS12 | | CS11 | | CS10 | | CS9 | | |
| 00F9D | OREULA8 | EULA8 | EULA7 | EULA6 | EULA5 | EULA4 | EULA3 | EULA2 | EULA1 | EULAn: Underline display setting for line n (n: 0 to 12) |
| 00F9E | OREULA12 | – | – | – | – | EULA12 | EULA11 | EULA10 | EULA9 | |
| 00F9F | OREFR8 | EFR8 | EFR7 | EFR6 | EFR5 | EFR4 | EFR3 | EFR2 | EFR1 | EFRn: Fringing setting for line n (n: 0 to 12) |
| 00FA0 | OREFR12 | – | – | – | – | EFR12 | EFR11 | EFR10 | EFR9 | |
| 00FA1 | ORCLKF | CK7 | CK6 | CK5 | CK4 | CK3 | CK2 | CK1 | CK0 | CKx: Display clock frequency monitor (x: 0 to 7) |
| 00FA1 | ORCLKC | CKC7 | CKC6 | CKC5 | CKC4 | CKC3 | CKC2 | CKC1 | CKC0 | CKCx: Display clock frequency (x: 0 to 7) |
| 00FA2 | ORSOL4 | SOL4 | | SOL3 | | SOL2 | | SOL1 | | SOLn: Solid space display setting for line n (n: 0 to 12)<br>00: No solid space 10: Right<br>01: Left 11: Left and right |
| 00FA3 | ORSOL8 | SOL8 | | SOL7 | | SOL6 | | SOL5 | | |
| 00FA4 | ORSOL12 | SOL12 | | SOL11 | | SOL10 | | SOL9 | | |
| 00FA5 | ORBK | IBDT | RBDT | GBDT | BBDT | IFDT | RFDT | GFDT | BFDT | IBDT, RBDT, GBDT, BBDT: Background color setting<br>IFDT, RFDT, GFDT, BFDT: Fringing color setting |
| 00FA6 | ORACL | ACLI2 | ACLR2 | ACLG2 | ACLB2 | ACLI1 | ACLR1 | ACLG1 | ACLB1 | ACLI2/ACLR2/ACLG2/ACLB2: Area 2 plane color<br>ACLI1/ACLR1/ACLG1/ACLB1: Area 1 plane color<br>Set ACLI2 and SCLI1 to 1, when PISEL: 1 |
| 00FA7 | CRRCL | EBKGD | EXBL | AON2 | AON1 | RCLI | RCLR | RCLG | RCLB | EBKGD: Background function<br>EXBL: Full-raster blanking<br>AON2: Area 2 plane display<br>AON1: Area 1 plane display<br>RCLI/R/G/B: Raster plane color<br>Set RCLI to 1, when PISEL: 1 |
| 00FA8 | ORAHS1 | AHS17 | AHS16 | AHS15 | AHS14 | AHS13 | AHS12 | AHS11 | AHS10 | AHSx: Area 1 plane horizontal start position (n: 0 to 8) |
| 00FA9 | | – | – | – | – | – | – | – | AHS18 | |
| 00FAA | ORAHE1 | AHE17 | AHE16 | AHE15 | AHE14 | AHE13 | AHE12 | AHE11 | AHE10 | AHE1x: Area 1 plane horizontal end position (n: 0 to 8) |
| 00FAB | | – | – | – | – | – | – | – | AHE18 | |
| 00FAC | ORAVS1 | AVS17 | AVS16 | AVS15 | AVS14 | AVS13 | AVS12 | AVS11 | AVS10 | AVS1x: Area 1 plane vertical start position (n: 0 to 8) |
| 00FAD | | – | – | – | – | – | – | – | AHS18 | |
| 00FAE | ORAVE1 | AVE17 | AVE16 | AVE15 | AVE14 | AVE13 | AVE12 | AVE11 | AVE10 | AVE1x: Area 1 plane vertical end position (n: 0 to 8) |
| 00FAF | | – | – | – | – | – | – | – | AVE18 | |
| 00FB0 | ORAHS2 | AHS27 | AHS26 | AHS25 | AHS24 | AHS23 | AHS22 | AHS21 | AHS20 | AHS2x: Area 2 plane horizontal start position (n: 0 to 8) |
| 00FB1 | | – | – | – | – | – | – | – | AHS28 | |
| 00FB2 | ORAHE2 | AHE27 | AHE26 | AHE25 | AHE24 | AHE23 | AHE22 | AHE21 | AHE20 | AHE2x: Area 2 plane horizontal end position (n: 0 to 8) |
| 00FB3 | | – | – | – | – | – | – | – | AHE28 | |
| 00FB4 | ORAVS2 | AVS27 | AVS26 | AVS25 | AVS24 | AVS23 | AVS22 | AVS21 | AVS20 | AVS2x: Area 2 plane vertical start position (n: 0 to 8) |
| 00FB5 | | – | – | – | – | – | – | – | AHS28 | |
| 00FB6 | ORAVE2 | AVE27 | AVE26 | AVE25 | AVE24 | AVE23 | AVE22 | AVE21 | AVE20 | AVE2x: Area 2 plane vertical end position (n: 0 to 8) |
| 00FB7 | | – | – | – | – | – | – | – | AVE28 | |
| 00FB8 | ORETC | VDSMD | PISEL | BKMF | ESMZ | "0" | MFYWR | MBK | RDWRV | VDSMD: Scan mode select<br>PISEL: I pin function select<br>BKMF: Blinking master<br>ESMZ: Smoothing<br>MFYWR: Display memory read mode select<br>MBK: Display memory bank switching select<br>RDWRV: Read/write mode select at normal mode |
| 00FB9 | ORIRC | – | – | – | SVD | ISDC | | | | SVD: Interrupt source select<br>ISDC: Interrupt generation line select |
| 00FB9 | ORIRC | – | – | – | – | DCTR | | | | DCTR: Display line counter |
| 00FBA | ORP6S | P67S | P66S | P65S | P64S | PIDS | YBLCS | MPXS | | P6xS: P6 port output select (x: 4 to 7)<br>PIDS: I pin output select<br>YBLCS: Y/BL signal select<br>MPXS: R, G, B, Y/BL single select |

Figure 2.15-31OSD Control Register List (2/2)

| Register Address | Register Name | Register Bit Configuration | | | | | | | | Bit Contents |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |
| 00FBB | ORIV | VDPOL | HDPOL | YBLII | RGBII | YIV | BLIV | RGBIV | IIV | VDPOL: $\overline{VD}$ input polarity select<br>HDPOL: $\overline{HD}$ input polarity select<br>YBLII: Y/BLIN input polarity select<br>RGBII: RIN, GIN, BIN input select<br>YIV: Y output polarity select<br>BLIV: BL output polarity select<br>RGBIV: R, G, B output polarity select<br>IIV: I pin polarity select |
| 00024 | ORDMA | DMA7 | DMA6 | DMA5 | DMA4 | DMA3 | DMA2 | DMA1 | DMA0 | DMAx: Display memory address setting (x: 0 to 8) |
| 00025 | | – | – | – | – | – | – | – | DMA8 | |
| 0001D | ORDSN | – | SLNT | EUL | BLF | IDT | RDT | GDT | BDT | SLNT: Slant EUL: Underline<br>BLF: Blinking IDT/RDT/CDT/BDT: Character color |
| 0001E | ORCRA | CRA7 | CRA6 | CRA5 | CRA4 | CRA3 | CRA2 | CRA1 | CRA0 | CRAx: Character code (x: 0 to 8) |
| 0001F | | – | – | – | – | – | – | – | CRA8 | |
| 00FBC | ORWVSH | WVSH7 | WVSH6 | WVSH5 | WVSH4 | WVSH3 | WVSH2 | WVSH1 | WVSH0 | WVSHx: Window upper limit position (x: 0 to 8) |
| 00FBD | | – | – | – | – | – | – | – | WVSH8 | |
| 00FBE | ORWVSl | WVSL7 | WVSL6 | WVSL5 | WVSL4 | WVSL3 | WVSL2 | WVSL1 | WVSL0 | WVSL: Window lower limit position (x: 0 to 8) |
| 00FBF | | – | – | – | – | – | – | – | WVSL8 | |
| 00F80 | ORDON | – | – | – | – | – | RGWR | EWDW | DON | RGWR: Writing data transfer control<br>EWDW: Window enable<br>DON: OSD display ON/OFF |

Note 1: Except the meshed registers are changed by RGWR.

Note 2: Only lower 2 bits of the register in address 00F80H are changed by RGWR (the register in address 00F80H must not be used with any of the read-modify-write instructions as SET, CLR, etc.).

## 2.16 Jitter Elimination Circuit

The TMP88CS38/CM38A/CP38A has a built-in jitter elimination circuit which maintains the vertical stability of the OSD even when input of the vertical signal fluctuates.

And the field decision information for the OSD circuit is detected by using jitter elimination circuit.

### 2.16.1 Configuration

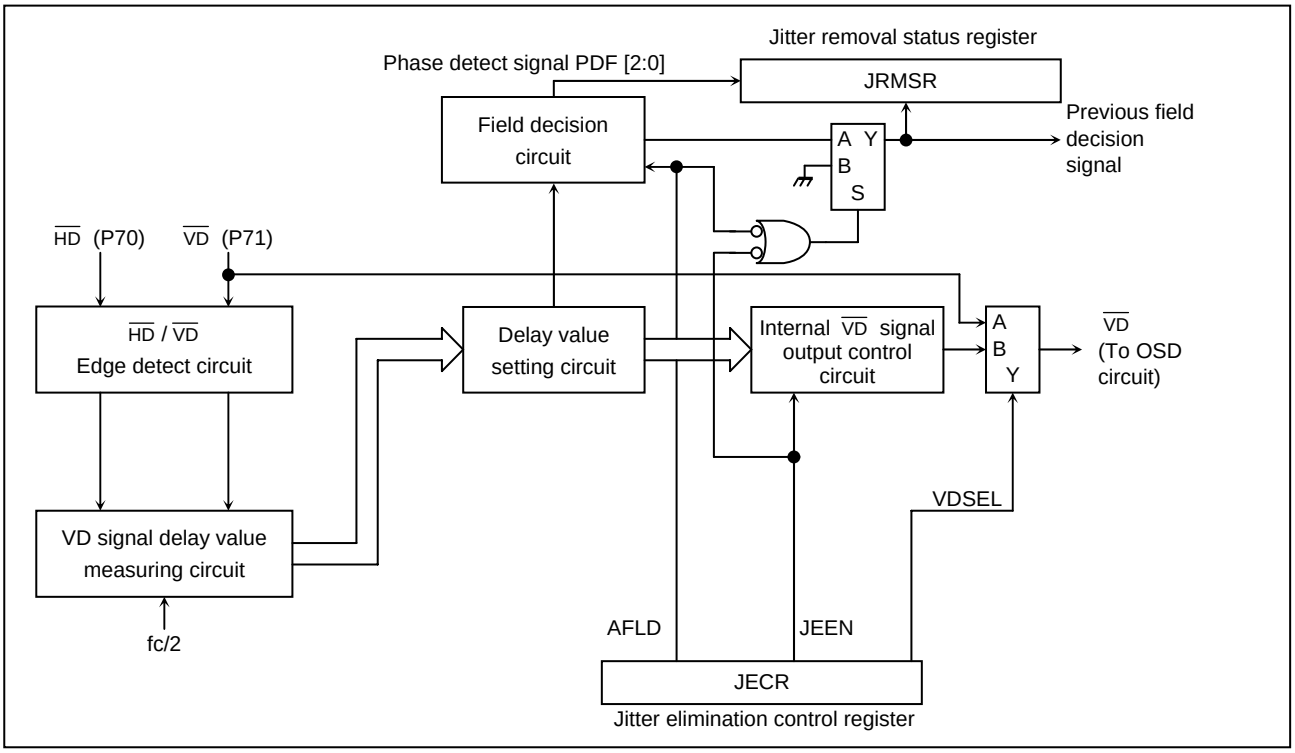



Figure 2.16.1 Jitter Elimination Circuit

### 2.16.2 Control

Jitter elimination circuit is controlled by the jitter elimination control register (JECR).

Jitter Elimination Control Register

| JECR (00FE4H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | – | – | – | VDSEL | AFLD | JEEN | "0" | "0" |

(Initial value: ***0 0000)

| | | | |
|---|---|---|---|
| VDSEL | VD select | 0: $\overline{VD}$ from P71<br>1: $\overline{VD}$ from jitter elimination circuit | Write only |
| AFLD | Automatic field decision | 0: Automatic field decision disabled<br>1: Automatic field decision enabled | |
| JEEN | Jitter elimination enable specification | 0: Jitter elimination disabled<br>1: Jitter elimination enabled | |

Note 1: Clear the AFLD to "0" to disable jitter elimination circuit.

Note 2: Always clear "0" to bit1 and bit0 of JECR.

Note 3: Clear "0" to AFLD and VDSEL if there is no phase shift in the vertical and horizontal sync. signals every other time, such as with non-interlaced TV.

Note 4: *: Don't care

Note 5: Setting JEEN to "0", OSD display is only 2nd field.

Note 6: Setting AFLD to "0", OSD display is only 2nd field.

Jitter Elimination Status Register

| JESR (00FE5H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FDSF | PDF1 | PDF0 | – | – | – | – | PDF2 |

(Initial value: 0*** ****)

| | | | |
|---|---|---|---|
| FDSF | Field detect status flag | 0: A position of a scanning line exists in the field which has a second display dot of character on an interlace TV screen.<br>1: A position of a scanning line exists in the field which has a first display dot of character on an interlace TV screen. | Read only |
| PDF2 to PDF0 | Phase detect flag between $\overline{HD}$ and $\overline{VD}$ | 000: Phase 0<br>001: Phase 1<br>010: Phase 2<br>011: Phase 3<br>100: Phase 4<br>101: Phase 5<br>110: Phase 6<br>111: Phase 7 | |

Note 1: FDSF is different from the 1st and the 2nd field. It is a unique field decided for OSD display.

Note 2: *: Don't care

Note 3:

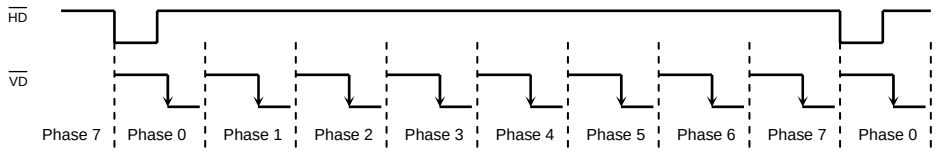



Figure 2.16.2 Jitter Elimination Control Register and Jitter Elimination Status Register

### 2.16.3 Jitter Elimination Mode

The jitter elimination circuit 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 +/–1/4HD, the jitter is automatically eliminated and internal $\overline{VD}$ signal is set to the stable location.

This function is enabled by setting JEEN (Bit2 in JECR) in the jitter elimination control register to "1".

### 2.16.4 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.

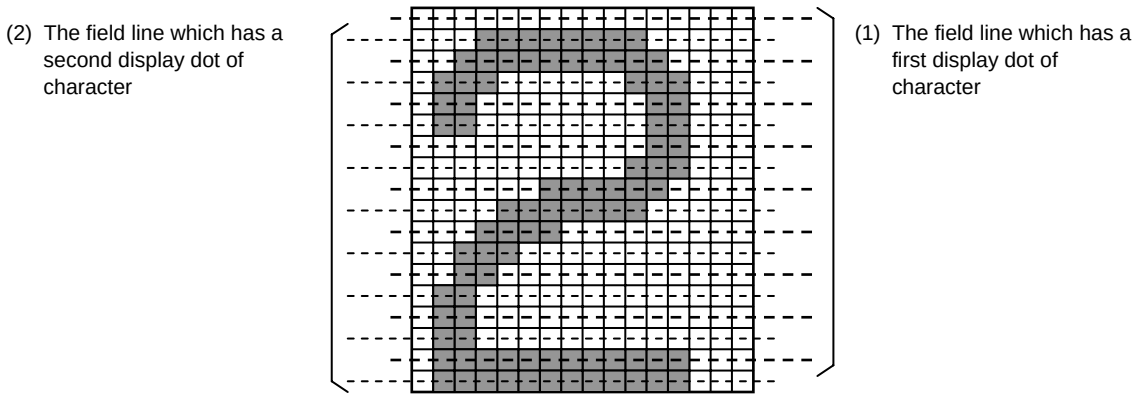
The OSD display in normal mode

Type A) When the OSD circuit is used on the TV system which has a phase shift in the vertical and horizontal sync. Signals every other filed such as the interlace TV, enable jitter elimination circuit and set "1" to AFLD and VDSEL. At this time, the field lines which have first and second display dot of character are displayed.

Type B) When the OSD circuit is used on the TV system which has no phase shift in the vertical and horizontal sync. Signals every other filed such as the non-interlace TV, enable jitter elimination circuit and clear "0" to AFLD and VDSEL. At this time, the field line which has a second display dot of character is only displayed.

The OSD display in double scan mode

Type C) Disable jitter elimination circuit and clear "0" to AFLD and VDSEL. At this time, the field lines which have first and second display dot of character are displayed.




| Scanning System | Register | Display |
|---|---|---|
| Type A | VDSEL = 1, AFLD = 1 | (1) and (2) |
| Type B | VDSEL = 0, AFLD = 0 | (2) |
| Type C | VDSEL = 0, AFLD = 0 | (1) and (2) |

Figure 2.16.3 Relation with Field Line and VDSEL, AFLD

## 2.17 Data Slicer

The TMP88CS38/CM38A/CP38A contains the data slicer to decode the caption data which multiplied during vertical flyback time of the composite video signal.

The composite video signal inputs to the data slicer circuit through P32 (VIN1) and P33 (VIN0). The caption data is decoded from the video signal. The composite video signal including negative sync-tip inputs to VIN0 and VIN1 pins. The data slicer 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.

Note: When the data slicer is used at fc = 16 MHz, set to "02H" in FC8CR.
When the data slicer is used at fc = 8 MHz, set to "00H" in FC8CR. (Refer to Figure 1.4.5)

### 2.17.1 Configuration

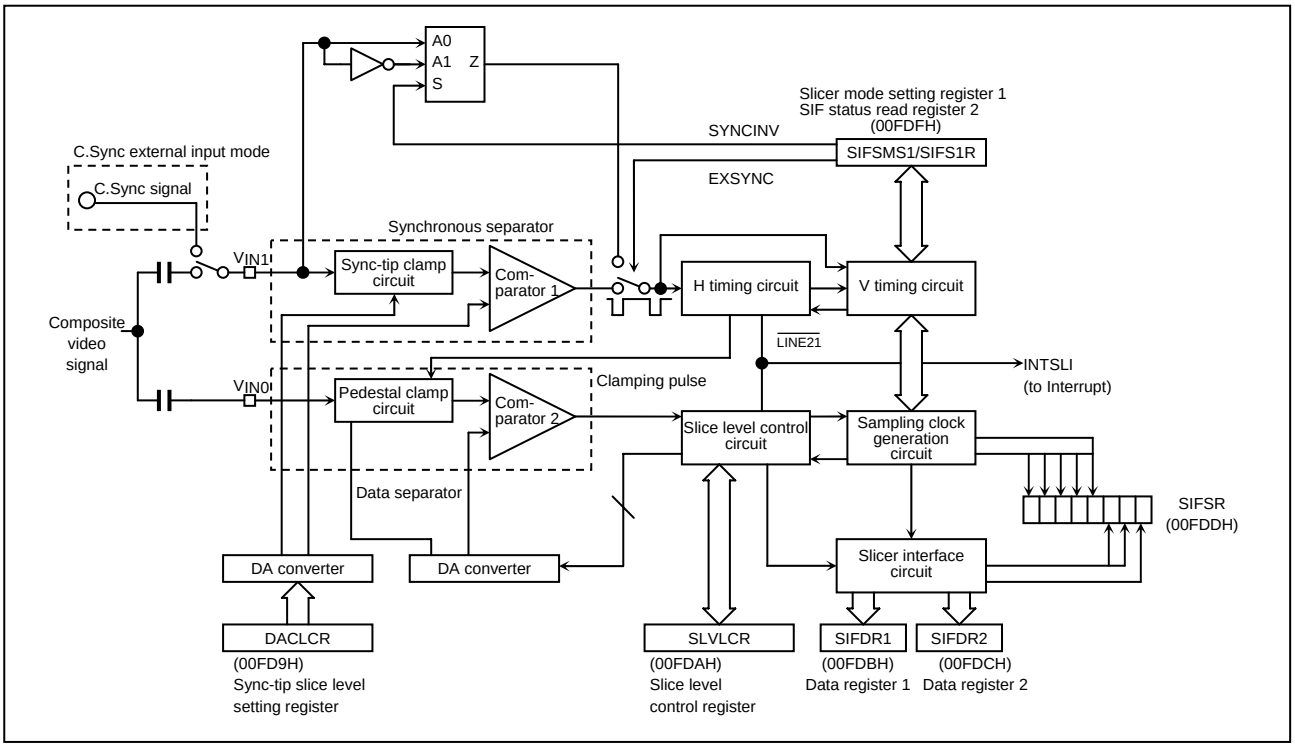



Figure 2.17.1 Data Slicer

### 2.17.2 Functions

(1) Video signal input

A low pass filter, a voltage amplifier and a condenser of about 0.1 μF are connected between the video signal and the video signal input pin of VIN1 and VIN0 pins, that is shown as Figure 2.17.3 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 VIN1 and VIN0 pins.

(2) Synchronous separator

This circuit is to separate the synchronous signal from the video signal. When DACL7 to DACL0 of DACLCR are set for the synchronous separation, the sync slice level is capable of setting. DACL7 to DACL4 set the slice level at the rising edge of the sync signal clamped data, and DACL3 to DACL0 set the slice level at the falling edge of the sync-tip clamped data. (Refer to section 2.17.5)

(3) Data separator

The data separator replaces the caption data piled on the video signal with the digital signal.

When SLVL5 to SLVL0 of SLVLCR are set to get the digital signal, the Initial value of the caption data slice level is capable of setting. (Refer to section 2.17.5)

(4) Sync-tip clamp circuit

The sync-tip 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) DA converter

This converter gets the DA 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 synchornously 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 synchornously 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.

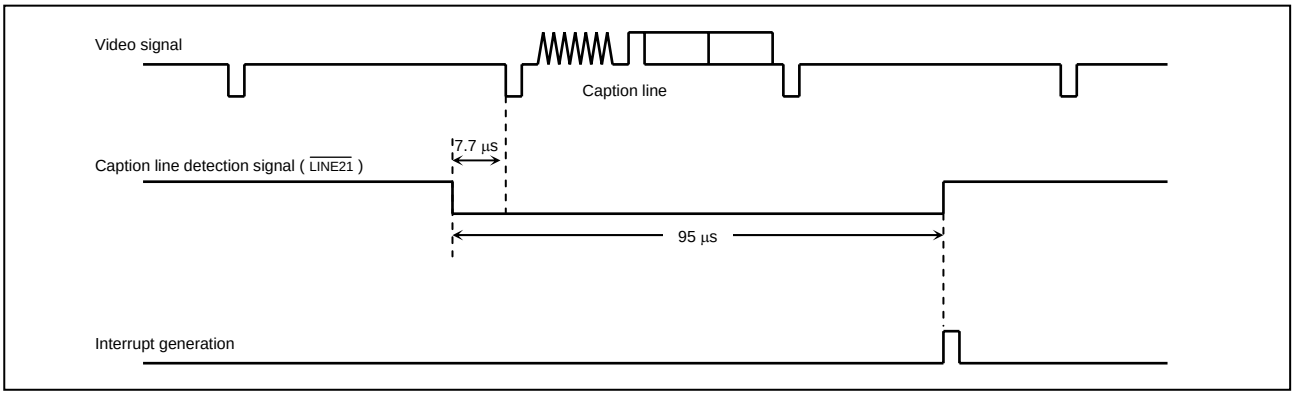



Figure 2.17.2 Interrupt Generation Timing

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 bit5) to "1".

As shown in Figure 2.17.3 (b), insert a low-pass filter ($f_T = 503$ kHz), voltage amplifier
(× 2 voltage amplification), and a capacitor of approximately 0.1 μF between the video signal and the video signal input pin VIN1 and input an external C.Sync signal to CSIN.

The polarity of the C.Sync signal is selected by SYNCINV (SIFSMS1 bit6). (Internally used as $\overline{\text{C.Sync}}$ .)

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

### 2.17.3 Video Signal Connection

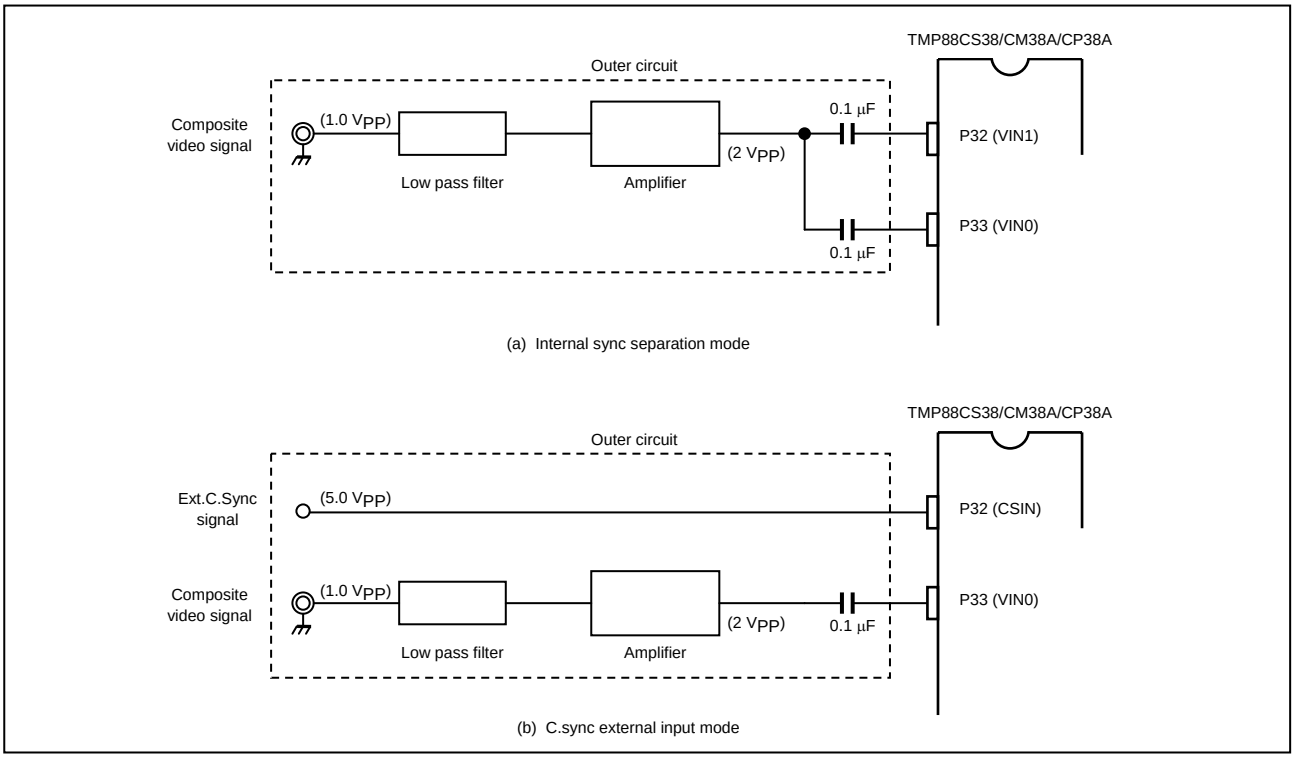



Figure 2.17.3 Video Signal Connection

Data Slicer Control Register

| SINTCR (00FD8H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | – | – | – | – | SLON | SLCR | – | – |

(Initial value: 0000 00**)

| | | | |
|---|---|---|---|
| SLON | Data slicer enable/disable | 1: Enable<br>0: Disable | Write only |
| SLCR | Data slicer interrupt control | 1: Enable interrupt<br>0: Disable interrupt | |

Data Slicer Interrupt Satus Register

| SINTCR (00FD8H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | – | – | – | SLIS | – | – | – | – |

(Initial value: ***0 ****)

| | | | |
|---|---|---|---|
| SLIS | Data slicer interrutp status | 0: –<br>1: Interrupt request | Read only |

Note 1: For setting SCLR to “1”, write “1” after SLON is set to “1”.

Note 2: SLIS is cleared to “0” after reading SINTCR.

Figure 2.17.4 Data Slicer Control (I)

SIF Data Register 1 (Caption data 1st byte read register) (Read only)

| SIFDR1 (00FDBH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | D1ST7 | D1ST6 | D1ST5 | D1ST4 | D1ST3 | D1ST2 | D1ST1 | D1ST0 |

| | | | |
|---|---|---|---|
| D1ST7-0 | Caption data 1st byte read register | | Read only |

SIF Data Register 2 (Caption data 2nd byte read register) (Read only)

| SIFDR2 (00FDCH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | D2ST7 | D2ST6 | D2ST5 | D2ST4 | D2ST3 | D2ST2 | D2ST1 | D2ST0 |

| | | | |
|---|---|---|---|
| D2ST7-0 | Caption data 2nd byte read register | | Read only |

SIF Status Register (Read only)

| SIFST (00FDDH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | STCRI | CRIN3 | CRIN2 | CRIN1 | CRIN0 | STFLD | STSB | STDE |

| | | | |
|---|---|---|---|
| STCRI | Clock run in detection | 1: Clock run in detection<br>0: No clock run in detection | Read only |
| CRIN | CRI number – 1 | Actual CRI number – 1 | |
| STFLD | Field identification | 1: 2nd field<br>0: 1st field | |
| STSB | Start bit identification flag | 1: From detection of start bit until fall in $\overline{VD}$<br>0: Other times | |
| STDE | 16-bit data receive end identification flag | 1: From end of 16-bit data reception until fall in $\overline{VD}$<br>0: Other times | |

Figure 2.17.5 Data Slicer Control (II)

Slicer Mode Setting Register 1 (Write only)

| SIFSMS1 (00FDFH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | "0" | SYNC INV | EXSYNC | "1" | CLINE3 | CLINE2 | CLINE1 | CLINE0 |

(Initial value: 0001 1011)

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

Note: Always write "0" to bit7 of SIFSMS1 and "1" to bit4 when writing to SIFSMS1.

Figure 2.17.6 Data Slicer Control (III)

SIF Status Read Register 2

| SIFS1R | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| (00FDFH) | – | – | GOODV | FLINE4 | FLINE3 | FLINE2 | FLINE1 | FLINE0 |

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

Figure 2.17.7 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 field 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)

| SLVLCR (00FDAH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | – | – | SLVL5 | SLVL4 | SLVL3 | SLVL2 | SLVL1 | SLVL0 |

(Initial value: **00 1010)

| | | | |
|---|---|---|---|
| SLVL | Slice level (Initial value:) setting<br>Sice level setting | 000000: VPCLAMP + (1/256) $V_{DD}$<br>000001: VPCLAMP + (2/256) $V_{DD}$<br>000010: VPCLAMP + (3/256) $V_{DD}$<br>000011: VPCLAMP + (4/256) $V_{DD}$<br>000100: VPCLAMP + (5/256) $V_{DD}$<br>⋮ ⋮<br>111101: VPCLAMP + (62/256) $V_{DD}$<br>111110: VPCLAMP + (63/256) $V_{DD}$<br>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 fata. 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-tip Slice Level Setting Register (Write only)

| DACLCR (00FD9H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DACL7 | DACL6 | DACL5 | DACL4 | DACL3 | DACL2 | DACL1 | DACL0 |

(Initial value: 0100 0010)

| | | | |
|---|---|---|---|
| DACL | DACL7 to DACL4: Slice level Lower limit setting<br>DACL3 to DACL0: Slice level Upper limit setting | 0000: VSCLAMP + (3/512) $V_{DD}$<br>0001: VSCLAMP + (6/512) $V_{DD}$<br>0010: VSCLAMP + (9/512) $V_{DD}$<br>0011: VSCLAMP + (12/512) $V_{DD}$<br>⋮ ⋮<br>1101: VSCLAMP + (42/512) $V_{DD}$<br>1110: VSCLAMP + (45/512) $V_{DD}$<br>1111: VSCLAMP + (48/512) $V_{DD}$ | Write only |

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

Figure 2.17.8 Data Slicer Control (V)

### 2.17.4 Clamp and Data Slicer Operation

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

| | |
|---|---|
| Sync signal separation | The composite video signal input via VIN1 (Pin 40) is clamped by the sync tip clamp circuit and the HD and VD sync signals separated by the sync separation circuit. |
| Caption line detection | Field decision and caption line detection are effected using the HD and VD sync signals |
| CRI detection | The CRI signal of the caption line interval is detected from the pedestal clamped video signal at VIN0 (Pin 41). |
| Slice level setting and generation of sampling clock | The slice level is controlled during the CRI signal interval, detected by the slice level control block, to obtain the optimum level.<br>To determine the timing for extracting the caption data, a sampling clock is generated that is phase locked to the CRI. |
| Caption signal extraction | The caption data is extracted at the selected slice level using sampling clock, which is locked to the caption data. |

The data slicer has two separation circuits:

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

b. 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 2.17.9.

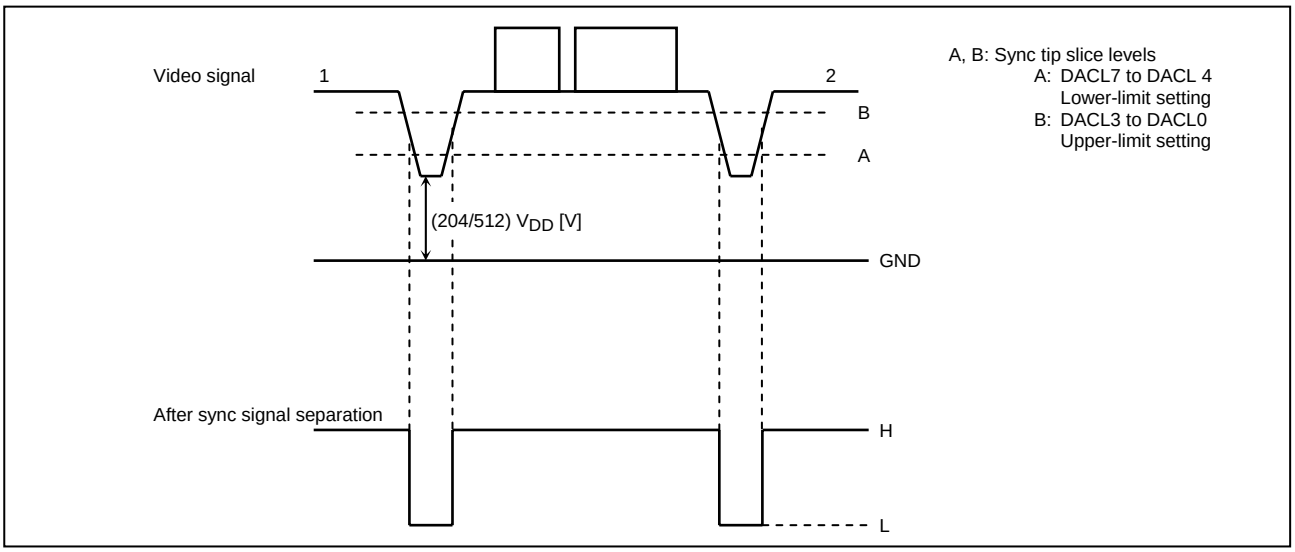



Figure 2.17.9 Sync Signal Slice

a-2 Method of sync signal slice

The sync signal is separated as shown in Figure 2.17.9.

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)

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

$V_{DD}$: Power supply voltage

VSCLAMP: Sync tip clamp voltage = (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 2.17.10.

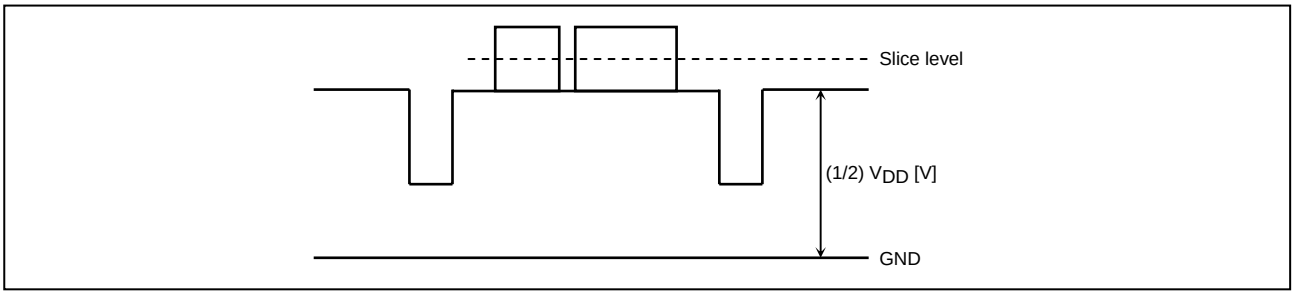



Figure 2.17.10 Pedestal Clamp

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:

Slice level = VPCLAMP + (X/256) VDD [V]

$V_{DD}$: Power supply voltage

VPCLAMP: Pedestal clamp voltage = (1/2) VDD

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.

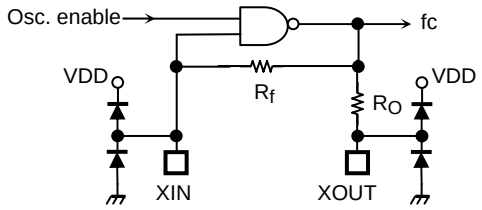
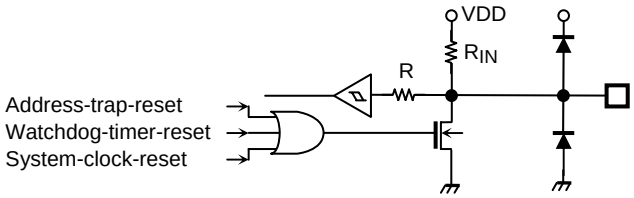
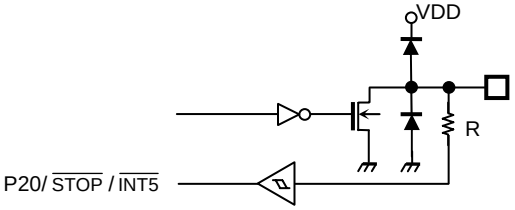
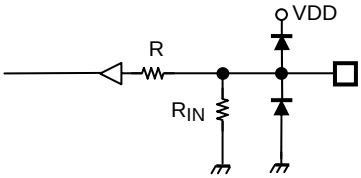
Slice level correction always begins with the setup value of SLVL (Bit5 to bit0 of SLVLCR).

If you want the last value to become the initial value of the next slice level, set it to SLVL (Bit5 to bit0 of SLVLCR).

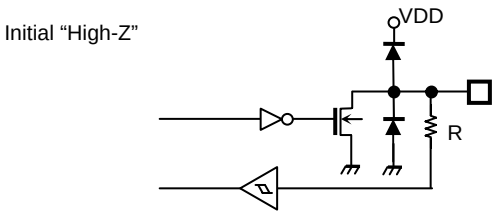
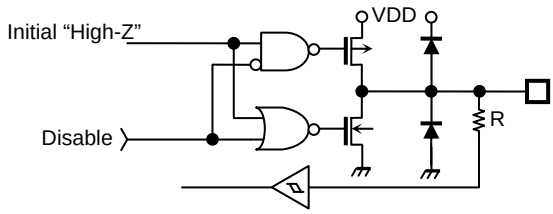
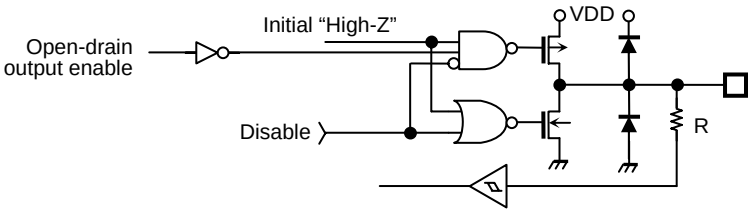
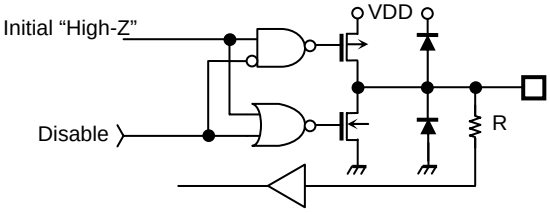
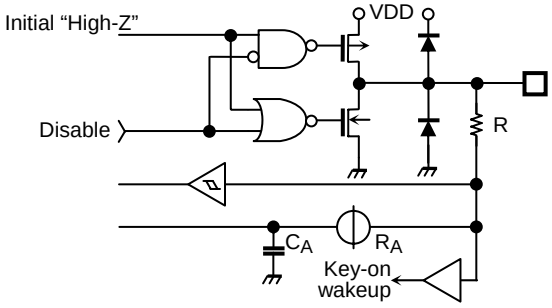
# Input/Output Circuit

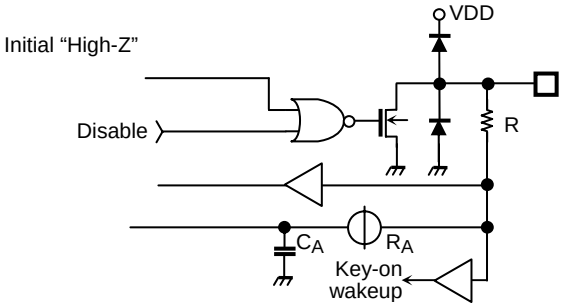
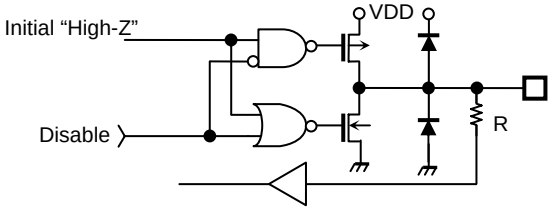
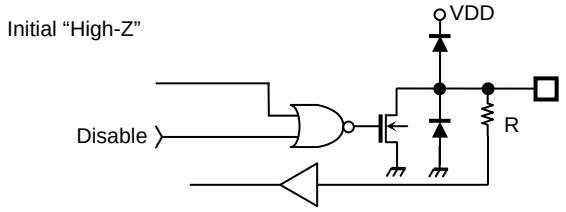
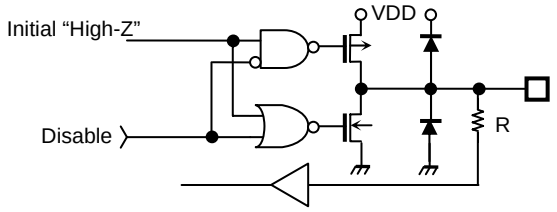
(1) Control pins

The input/output circuitries of the TMP88CS38/CM38A/CP38A control pins are shown below.

| Control Pin | I/O | Input/Output Circuitry | Remarks |
|---|---|---|---|
| XIN<br>XOUT | I/O |  | Resonator connection pins<br>(High frequency)<br><br>$R_f$ = 1.2 MΩ (typ.)<br>$R_O$ = 0.5 kΩ (typ.) |
| $\overline{\text{RESET}}$ | I/O |  | Sink open-drain output<br>Hysteresis input<br>Pull-up resistor<br><br>$R_{IN}$ = 220 kΩ (typ.)<br>R = 1 kΩ (typ.) |
| $\overline{\text{STOP}}$ / $\overline{\text{INT5}}$<br>(P20) | Input |  | Hysteresis input<br><br>R = 1 kΩ (typ.) |
| TEST | Input |  | Pull-down resistor<br><br>$R_{IN}$ = 70 MΩ (typ.)<br>R = 1 kΩ (typ.) |

(2) Input/Output ports

| Port | I/O | Input/Output Circuitry | Remarks |
|---|---|---|---|
| P20 | I/O |  | Sink open-drain output<br>Hysteresis input<br><br>R = 1 kΩ (typ.) |
| P30 to P33<br>P50, P57<br>P70, P71 | I/O |  | Tri-state I/O<br>Hysteresis input<br><br>R = 1 kΩ (typ.) |
| P34, P35, P51, P52 | I/O |  | Tri-state I/O or open-drain output programmable<br>Hysteresis input<br><br>R = 1 kΩ (typ.) |
| P40 to P47 | I/O |  | Tri-state I/O<br><br>R = 1 kΩ (typ.) |
| P53 to P56 | I/O |  | Tri-state I/O<br>Hysteresis input<br>Key-on wakeup input<br>($V_{IL4} = 0.65 \times V_{DD}$)<br><br>R = 1 kΩ (typ.)<br>$R_A$ = 5 kΩ (typ.)<br>$C_A$ = 22 pF (typ.) |

| Port | I/O | Input/Output Circuitry | Remarks |
|---|---|---|---|
| P60, P61 | I/O |  | Sink open-drain output<br>High current output<br>$I_{OL}$ = 20 mA (typ.)<br><br>R = 1 kΩ (typ.)<br>$R_A$ = 5 kΩ (typ.)<br>$C_A$ = 22 pF (typ.)<br><br>Key-on wakeup input<br>($V_{IL4}$ = 0.65 × $V_{DD}$) |
| P62 (at CSOUT) | I/O |  | Tri-state I/O<br>High current output<br>$I_{OL}$ = 20 mA (typ.)<br><br>R = 1 kΩ (typ.) |
| P62, P63 | I/O |  | Sink open-drain output<br>High current output<br>$I_{OL}$ = 20 mA (typ.)<br><br>R = 1 kΩ (typ.) |
| P64 to P67 | I/O |  | Tri-state I/O<br><br>R = 1 kΩ (typ.) |

# Electrical Characteristics

## Absolute Maximum Ratings

($V_{SS}$ = 0 V)

| Parameter | Symbol | Pins | Ratings | 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 1 pin) | $I_{OUT1}$ | Ports P2, P3, P4, P5, P64 to P67, P7 | 3.2 | mA |
| | $I_{OUT2}$ | Ports P60 to P63 | 30 | mA |
| Output current (Total) | Σ $I_{OUT1}$ | Ports P2, P3, P4, P5, P64 to P67, P7 | 120 | mA |
| | Σ $I_{OUT2}$ | Ports P60 to P63 | 120 | mA |
| Power dissipation [Topr = 70°C] | PD | – | TMP88CS38NG: 600<br>TMP88CS38FG/<br>CP38A/CM38A: 400 | mW |
| Soldering temperature (Time) | Tsld | – | 260 (10 s) | °C |
| Storage temperature | Tstg | – | –55 to 125 | °C |
| Operating temperature | Topr | – | –30 to 70 | °C |

Note: The absolute maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, 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 V, Topr = –30 to 70°C)

| Parameter | Symbol | Pins | Conditions | | Min | Max | Unit |
|---|---|---|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | | Fc = 16 MHz | NORMAL mode | 4.5 | 5.5 | V |
| | | | Fc = 16 MHz | IDLE mode | | | |
| | | | | STOP mode | | | |
| Input high voltage | $V_{IH1}$ | Except hysteresis input | $V_{DD}$ = 4.5 to 5.5V | | $V_{DD}$ × 0.70 | $V_{DD}$ | V |
| | $V_{IH2}$ | Hysteresis input | | | $V_{DD}$ × 0.75 | | |
| Input low voltage | $V_{IL1}$ | Except hysteresis input | $V_{DD}$ = 4.5 to 5.5V | | 0 | $V_{DD}$ × 0.30 | V |
| | $V_{IL2}$ | Hysteresis input | | | | $V_{DD}$ × 0.25 | |
| | $V_{IL4}$ | Key-on wakeup input | $V_{DD}$ = 4.5 to 5.5V | | | $V_{DD}$ × 0.65 | |
| Clock frequency | fc | XIN, XOUT | $V_{DD}$ = 4.5 to 5.5V | | 8.0 | 16.0 | MHz |
| | $f_{OSC}$ | Internal clock | $V_{DD}$ = 4.5 to 5.5V | fc = 8 MHz | 8.0 | 12.0 | MHz |
| | | | | fc = 16 MHz | 16.0 | 24.0 | MHz |

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: Clock frequency fc: Supply voltage range is specified in NORMAL mode and IDLE mode.

Note 3: Smaller value is alternatively specified as the maximum value.

## DC Characteristics

($V_{SS}$ = 0 V, Topr = –30 to 70°C)

| Parameter | Symbol | Pins | Conditions | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|---|
| Hysteresis voltage | $V_{HS}$ | Hysteresis inputs | | – | 0.9 | – | V |
| Input current | $I_{IN1}$ | TEST | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V/0 V | – | – | ±2 | μA |
| | $I_{IN2}$ | Open-drain ports | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V/0 V | – | – | ±2 | |
| | $I_{IN3}$ | Tri-state ports | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V/0 V | – | – | ±2 | |
| | $I_{IN4}$ | $\overline{RESET}$ , $\overline{STOP}$ | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V/0 V | – | – | ±2 | |
| Input resistance | $R_{IN2}$ | $\overline{RESET}$ | $V_{DD}$ = 5.5 V, $V_{IN}$ = 0 V | 100 | 220 | 450 | kΩ |
| Output leakage current | $I_{LO1}$ | Sink open-drain ports | $V_{DD}$ = 5.5 V, $V_{OUT}$ = 5.5 V | – | – | 2 | μA |
| | $I_{LO2}$ | Tri-state ports | $V_{DD}$ = 5.5 V, $V_{OUT}$ = 5.5 V/0 V | – | – | ±2 | |
| Output high voltage | $V_{OH2}$ | Tri-state ports | $V_{DD}$ = 4.5 V, $I_{OH}$ = – 0.7 mA | 4.1 | – | – | V |
| Output low voltage | $V_{OL}$ | Except XOUT and ports P60 to P63 | $V_{DD}$ = 4.5 V, $I_{OL}$ = 1.6 mA | – | – | 0.4 | |
| Output low current | $I_{OL3}$ | Port P60 to P63 | $V_{DD}$ = 4.5 V, $V_{OL}$ = 1.0 V | – | 20 | – | mA |
| Supply current in NORMAL mode | $I_{DD}$ | – | $V_{DD}$ = 5.5 V<br>fc = 16 MHz (Note 3)<br>$V_{IN}$ = 5.3 V/0.2 V | – | 25 | 30 | |
| Supply current in IDLE mode | | | | – | 20 | 25 | |
| Supply current in STOP mode | | | $V_{DD}$ = 5.5 V<br>$V_{IN}$ = 5.3 V/0.2 V | – | 0.5 | 10 | μA |

Note 1: Typical values show those at Topr = 25°C, $V_{DD}$ = 5 V.

Note 2: Input Current $I_{IN3}$: The current through resistor is not included.

Note 3: Supply Current $I_{DD}$: The current (Typ. 0.5 mA) through ladder resistors of ADC is included in NORMAL mode and IDLE mode.

## AD Conversion Characteristics

($V_{SS}$ = 0 V, $V_{DD}$ = 4.5 V to 5.5 V, Topr = –30 to 70°C)

| Parameter | Symbol | Conditions | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Analog reference voltage | $V_{AREF}$ | supplied from $V_{DD}$ pin. | – | $V_{DD}$ | – | V |
| | $V_{ASS}$ | supplied from $V_{SS}$ pin. | – | 0 | – | |
| Analog reference voltage range | $\Delta V_{AREF}$ | = $V_{DD}$ – $V_{SS}$ | – | $V_{DD}$ | – | |
| Analog input voltage | $V_{AIN}$ | | $V_{SS}$ | – | $V_{DD}$ | |
| Nonlinearity error | | $V_{DD}$ = 5.0 V | – | – | ±1 | LSB |
| Zero point error | | | – | – | ±2 | |
| Full scale error | | | – | – | ±2 | |
| Total error | | | – | – | ±3 | |

Note: The total error means all error except quanting error.

## AC Characteristics

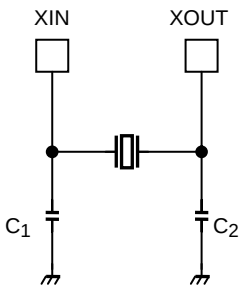
($V_{SS}$ = 0 V, $V_{DD}$ = 4.5 V to 5.5 V, Topr = –30 to 70°C)

| Parameter | Symbol | Conditions | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Machine cycle time | $t_{cy}$ | in NORMAL mode | 0.5 | – | 1.0 | μs |
| | | in IDLE mode | | | | |
| High level clock pulse width | $T_{WCH}$ | for external clock operation (XIN input), fc = 16 MHz | 31.25 | – | – | ns |
| Low level clock pulse width | $T_{WCL}$ | | | | | |

## Recommended Oscillating Conditions

($V_{SS}$ = 0 V, $V_{DD}$ = 4.5 V to 5.5 V, Topr = –30 to 70°C)

| Parameter | Oscillator | Oscillation Frequency | Recommended Oscillator | | Recommended Constant | |
|---|---|---|---|---|---|---|
| | | | | | $C_1$ | $C_2$ |
| High-frequency oscillation | Ceramic resonator | 8 MHz | Murata | CSA 8.00MTZ | 30 pF | 30 pF |
| | | 16 MHz | Murata | CSA 16.00MXZ040 | 5 pF | 5 pF |




High-frequency oscillation

Note 1: To keep reliable operation, shield the device electrically with the metal plate on its package mold surface against the high electric field, for example, by CRT (Cathode ray tube).

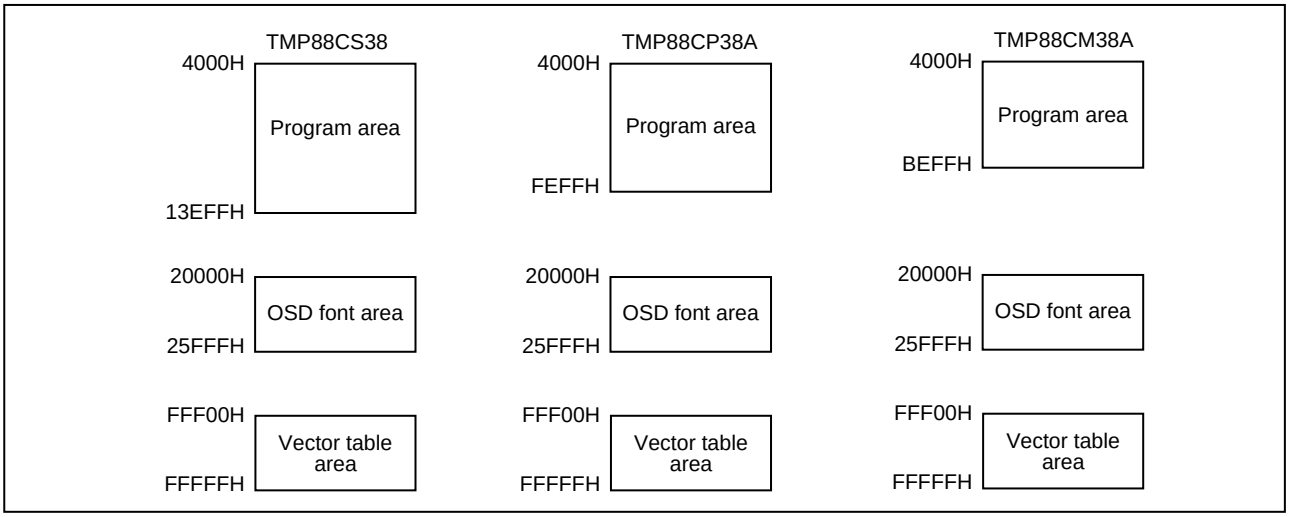
Note 2: The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following URL;

http://www.murata.co.jp/search/index.html

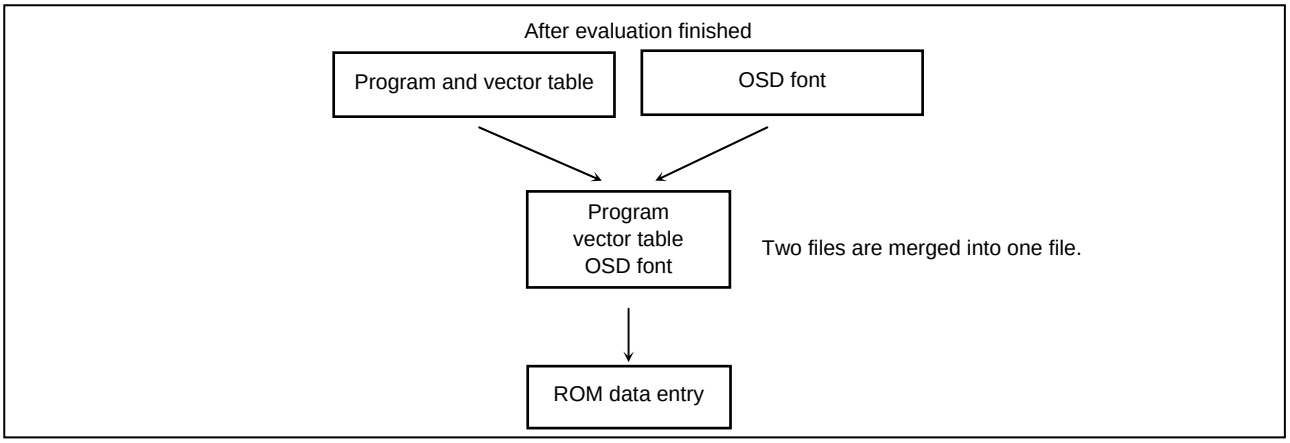
# Notice of ROM Entry

When you make a ROM data entry for TMP88CS38 and TMP88CM38A/CP38A,
Please transfer one file including program area, vector table area and OSD font area.

The ROM area must be transferred is as follows.



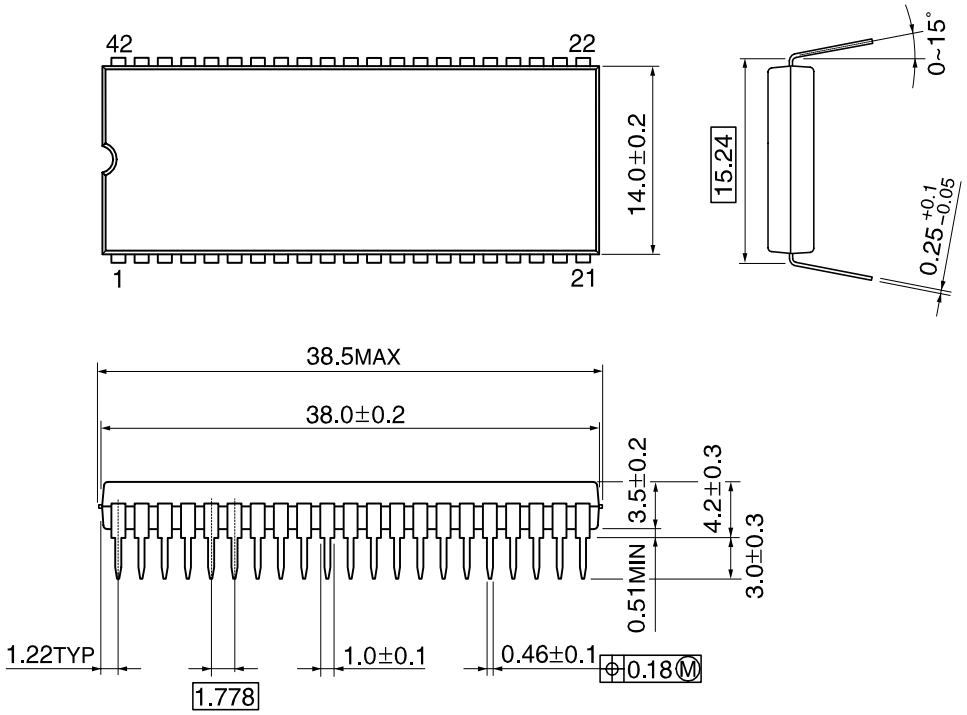

Flow of ROM data entry

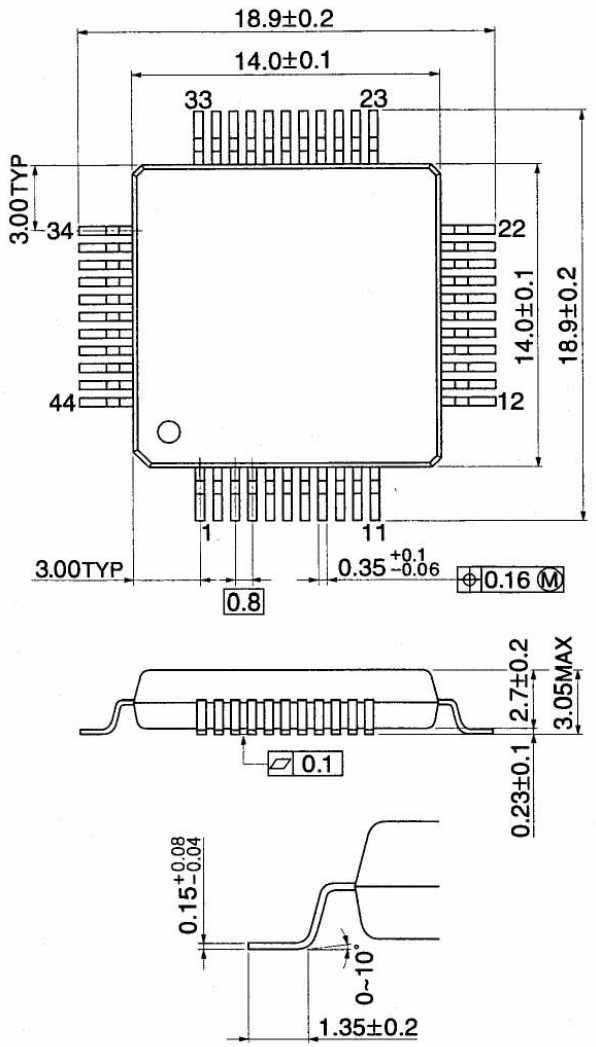


# Package

P-SDIP42-600-1.78

Unit: mm



P-QFP44-1414-0.80K

Unit: mm