CMOS 8–Bit Microcontrollers

TMP90C848F

1. Outline and Characteristics

The TMP90C848F is a high-end 8-bit microcontroller developed in order to control power supply switching. With built-in high-speed 16-channel flush A/D converter (conversion time: 1.6 microseconds @10MHz*), high-speed 8-channel PWM (8-bit resolution, oscillation frequencies of 80KHz, 160KHz, and 320KHz @10MHz*) and a clock switching function, the TMP90C848F is ideal for controlling compact personal computer power supplies. (* indicates a CPU clock with a 20MHz external clock (X1), which is divided by 2 during high-speed operation.)

The TMP90C848F is a CMOS 8-bit microcontroller which integrates an 8-bit CPU, ROM, RAM, an A/D converter, a multi-function timer, serial interface, high-speed PWM, and a clock switching function in a single chip.

The characteristics of the TMP90C848F are as follows: (Assuming high-speed external clock X1 = 20MHz, CPU clock fc = 10MHz)

(1) Highly efficient instructions (TLCS-90 instruction sets)
163 types of basic instructions, including
Multiplication/division arithmetic, 16-bit arithmetic operation, bit manipulation instruction.

(2) Minimum instruction executing time: 400ns

(3) Built-in ROM: 8K bytes

(4) Built-in RAM: 512 bytes

(5) High-speed flash A/D converter (standard conversion time: 1.6 microseconds, 16 channels)

(6) General-purpose serial interface (1 channel)

(7) High-speed PWM output (8 channels, effective only at high-speed clock operation)
Oscillation frequency: 80KHz, 160KHz, 320KHz
Resolution: 8 bits (50ns, 25ns, 12.5ns)

(8) Multi-function 16-bit timer/event counter: 1

(9) 8-bit timer: 4 channel

(10) Input/Output port: 62 pins

(11) Micro DMA function (11 channels)

(12) Watchdog timer function (internally connectable to hardware reset)

(13) Interrupt function: 10 internal; 3 external

(14) Clock switching function (high-speed/low-speed switching)

(15) Standby function (3 HALT modes, possible to disable setting STOP mode.)

(16) Standby function (3 HALT modes, standby mode disable.)

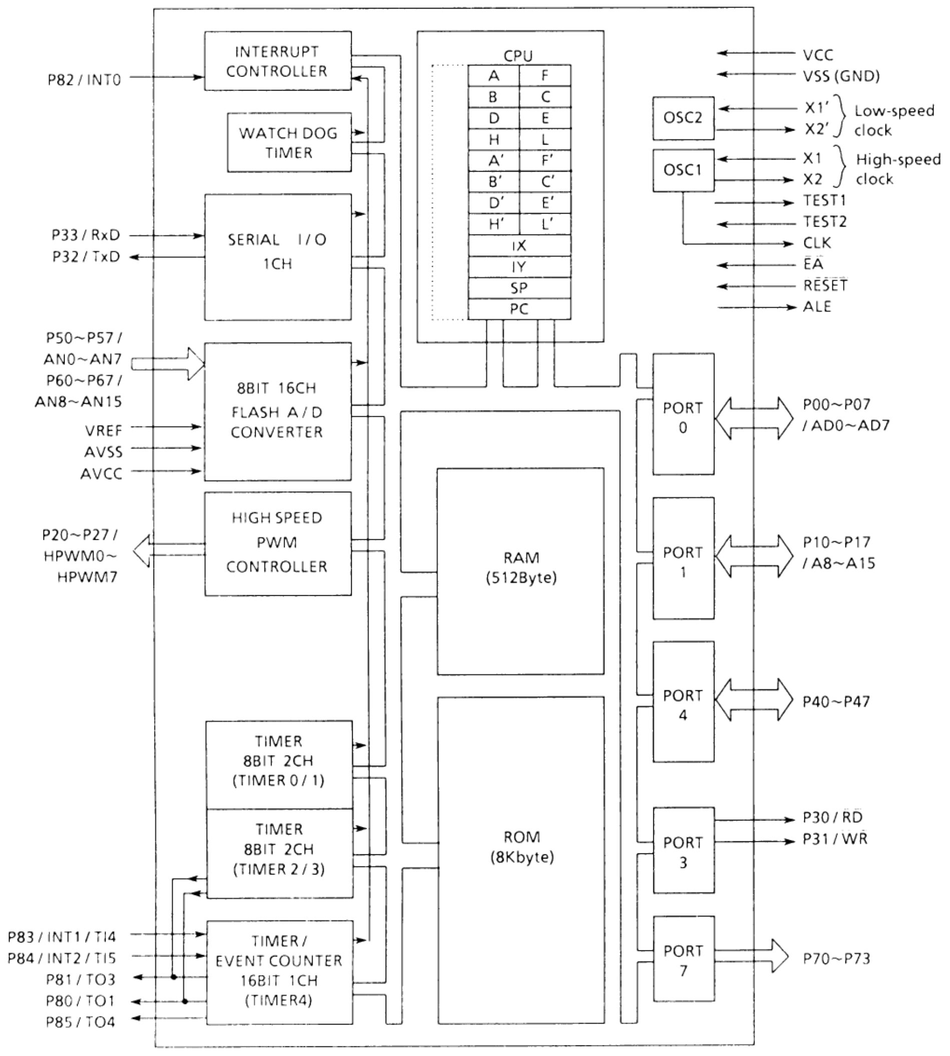


Figure 1. TMP90C848F Block Diagram

2. Pin Assignment and Functions

The assignment of input/output pins for TMP90C848F, their names and functions are described below.

2.1 Pin Assignment

Figure 2.1 (1) shows pin assignment of the TMP90C848F.

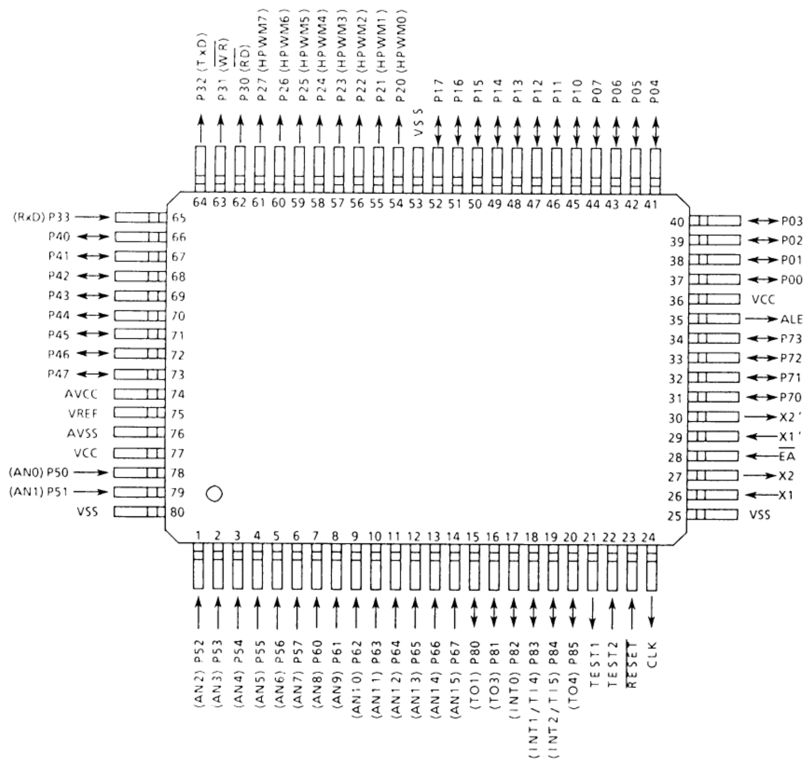


Figure 2.1. Pin Assignment (80-pin Flat Package)

2.2 Pin Names and Functions

The names of input/output pins and their functions are summarized in Table 2.2.

Table 2.2 Pin Names and Functions (1/2)

Pin Name	No. of pins	I/O or tristate	Function
P00 ~ P07	8	I/O	Port 0: 8-bit I/O port. Each bit can be set for input or output.
P10 ~ P17	8	I/O	Port 1: 8-bit I/O port. Each bit can be set for input or output. Pull-up resistance included.
P20 ~ P27/ HPWM0 ~ 7	8	Output	Port 2: 8-bit output port.
		Output	High-speed PWM output: High-speed 8
P30	1	Output	Port 30: 1-bit output port.
P31	1	Output	Port 31: 1-bit output port.
P32/ TxD	1	Output	Port 32: 1-bit output port.
		Output	Used to transmit serial data.
P33/ RxD	1	Input	Port 33: 1-bit output port.
		Input	Used to receive serial data.
P40 ~ P47	8	I/O	Port 4: 8-bit I/O port. Each bit can be set for input or output (P40 - P43 O.D. 4mA sink, P44 - 47 10mA source).
P50 ~ P57 /AN0 ~ AN7	8	Input	Port 5: 8- bit input port.
		Input	Analog input: 8-bit analog input to the A/D converter.
P60 ~ 67 /AN8 ~ AN15	8	Input	Port 6: 8-bit input port.
		Input	Analog input: 8-bit analog input the A/D converter.
P70 ~ P73	4	I/O	Port 7: 4-bit I/O port. Each bit can be set for input or output. Programmable pull-up resistance included.
P80 /T01	1	I/O	Port 80: 1-bit I/O port.
		Output	Timer output 1: Used for timer 0 or timer 1 output.
P81 /T03	1	I/O	Port 81: 1-bit I/O port.
		Output	Timer output 3: Used for timer 2 or timer 3 output.
P82 /INT0	1	I/O	Port 82: 1-bit I/O port.
		Input	Interrupt request pin 0: Level/rising edge programmable interrupt request pin.
P83 /INT1 /TI4	1	I/O	Port 83: 1-bit I/O port.
		Input	Interrupt request pin 1: Rising/falling edge programmable interrupt request pin.
		Input	Timer input 4: Count input/capture trigger signal for timer 4.
P84 /INT2 /TI5	1	I/O	Port 84: 1-bit I/O port.
		Input	Interrupt request pin 2: Rising/falling edge programmable interrupt request pin.
		Input	Timer input 5: Count input/capture trigger signal for timer 5.
P85 /T04	1	I/O	Port 85: 1-bit I/O port.
		Output	Timer output 4: Used as the timer 4 output.
ALE	1	Output	Address latch enable signal: No use.

Table 2.2 (2/2)

Pin name	No. of pins	I/O or tristate	Function
CLK	1	Output	Clock output: Generates clock pulse at 1/4 frequency of clock oscillation. Pulled up internally during resetting.
EA	1	Input	External access: Connected to the V_{CC} pin when using the TMP90C848F with built-in ROM.
RESET	1	Input	Reset: Initializes the TMP90C848F.
X1/X2	2	I/O	High-speed crystal oscillator connection pin.
X1′/X2′	2	I/O	Low-speed crystal oscillator connection pin.
TEST1/TEST2	2	I/O	Testing pins: Connects directly TEST1 and TEST2 at a normal state operation.
A VCC	1	–	Comparator power supply for the A/D converter.
VREF	1	–	A/D converter reference voltage input.
AVSS	1	–	Analog GND pin (0V)
V_{CC}	2	–	Power supply pin (+ 5V ± 10%)
V_{SS}	3	–	GND pin (0V)

3. Operation

This section explains the various functions and basic operations of the TMP90C848F.

3.1 CPU

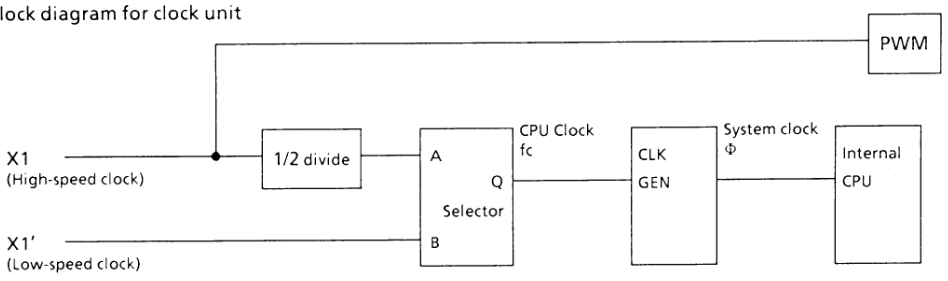
The TMP90C848F has an internal built-in, high-performance 8-bit CPU. For a description of the CPU operation, see the book TLCS 90 Series CPU Core Architecture .

This section explains CPU functions dedicated to the TMP90C848F, which are not described in that book.

3.1.1 Clock

During high-speed operation of the TMP90C848F, the external clock (X1) divided by 2 is used as the CPU clock fc. During low-speed operation, the external clock (X1') is used as the CPU clock fc. In this manual, a clock indicates the CPU clock, unless stated otherwise.

Block diagram for clock unit



3.1.2 Reset

Figure 3.1 shows the basic timing of reset.

To reset the TMP90C848F, it is required that, 1) the power supply voltage is within the specifed operating range, 2) the internal oscillator is stabe, and 3) the $\overline{\text{RESET}}$ input is kept at "0" at least 10 system clocks (10 states: 2µsec at 10MHz system clock).

When reset is accepted, the following I/O ports are set to input status (with high impedance): port 0 (address data bus AD0 ~ AD7), port 1 (address bus A8 ~ A15), port 4, port 7 and port 8. The output ports P30 ($\overline{\text{RD}}$), P31 ($\overline{\text{WR}}$) and P32, and CLK are set to "1", and ALE is cleared to "0". The input ports remain the same.

CPU registers and external memory are not changed. However, the program counter PC and the interrupt enable/disable flag IFF are cleared to "0". Register A becomes undefined.

When the reset is released, instruction start executing from address 0000H.

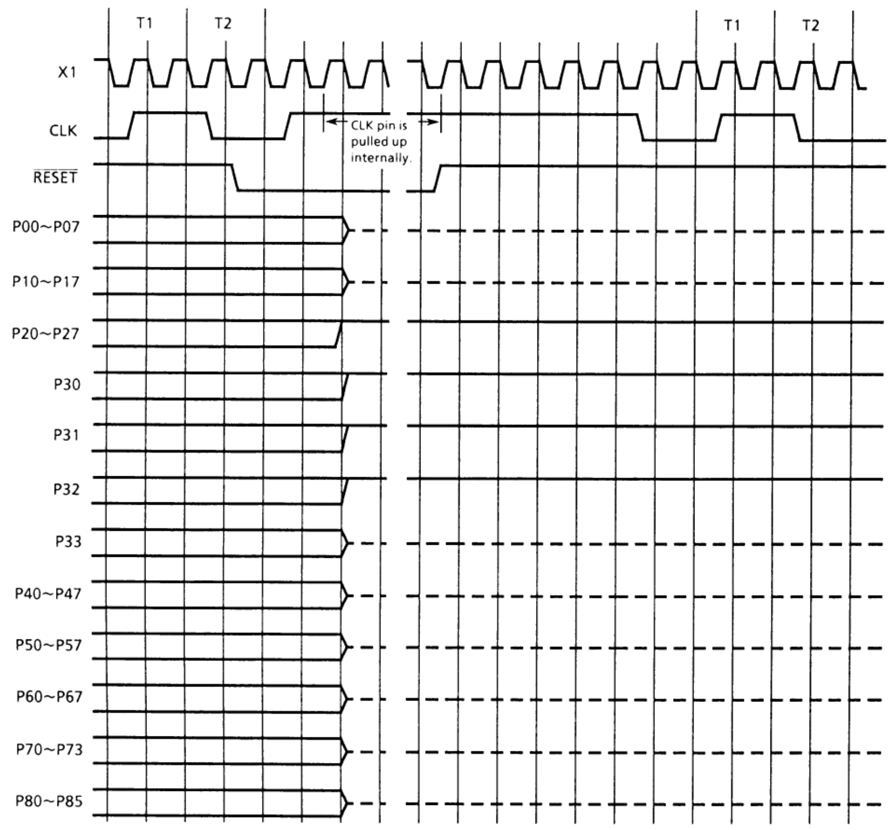


Figure 3.1. TMP90C848F Reset Timing

3.1.3 EXF (Exchange Flag)

The exchange flag EXF is inverted when the EXX instruction is executed to exchange data between the TMP90C848 main registers and auxiliary registers. This flag is assigned to bit 1 at memory address FFD2H.

WDMOD (FFD2H)

	7	6	5	4	3	2	1	0
bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
Read/Write	R/W	R/W		R/W	R/W		R	R/W
After reset	1	0	0	0	0	0	Un-defined	0
Function	0: WDT Enable	WDT detection time 00: 2^{16}/fc 01: 2^{18}/fc 10: 2^{20}/fc 11: 2^{22}/fc		Warming up time (To release stop mode 0: 2^{11}/fc 1: 2^{13}/fc) (To switch clock 0: 2^{11}/fc 1: 2^{13}/fc)	Standby mode 00: RUN 01: STOP 10: IDEL1 11: –		Inverts each time the EXX instructio n is executed.	1: Drives the pin even in the STOP mode

3.2 Memory Map

The TMP90C848F can provide a maximum 64K byte program and data memory.
The program and data memories may be allocated to the addresses 0000H ~ FFFFH.

(1) Built-in ROM

The TMP90C848F has an internal 8-byte ROM. This ROM is located at addresses 0000H ~ 1FFFH. Program execution starts from address 0000H after a reset operation.

The addresses 0008H ~ 0068H in the internal ROM area are used as the interrupt processing entry area.

(2) Built-in RAM

TMP90C848F contains a 512-byte RAM which is allocated to the addresses FFC0H ~ FFBFH. The CPU can also access some portions of the RAM (192 byte area FF00H ~ FFBFH) using short instruction codes in the direct addressing mode.

Addresses of FF18H ~ FF6DH this RAM area are used as the parameter area for micro DMA processing. (This area can freely be used when the micro DMA function is not used.)

(3) Built-in I/O

TMP90C848F uses 56 bytes of the address space as a built-in I/O area. The area is allocated to the addresses FFC0H ~ FFF7H. The CPU can access the built-in I/O using short instruction codes in the direct addressing mode.

Figure 3.2 shows the memory map and the access ranges of the CPU for each addressing mode.

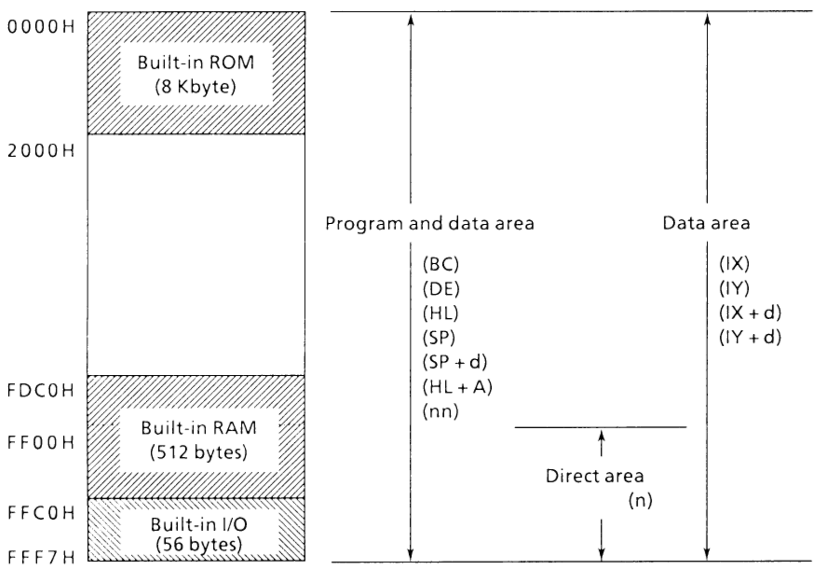


Figure 3.2. TMP90C848F Memory Map

3.3 Interrupt Functions

The TMP90C848F provides the two processing modes for internal and external interrupt requests; a general-purpose interrupt processing mode and a micro DMA processing mode in which the CPU can automatically transfer data.

Immediately after a reset is released, all the responses to interrupt requests are set in the general-purpose interrupt processing mode. Using DMA enable/disable register which will be described later, each interrupt request can be set to the micro DMA processing mode.

Figure 3.3 (1) shows a flowchart of the interrupt response sequence.

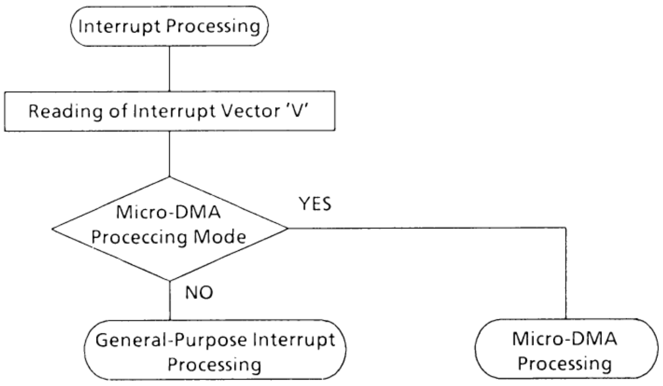

Interrupt Processing
Reading of Interrupt Vector 'V'
Micro-DMA Procecciing Mode
YES
NO
General-Purpose Interrupt Processing
Micro-DMA Processing

Figure 3.3 (1). Interrupt Response Flowchart

When an interrupt is generated, it is reported to the CPU via the built-in interrupt controller. The CPU starts the interrupt processing if it is a non-maskable or maskable interrupt requested in the EI state (interrupt enable/disable flag (IFF bit of the F register) = “1”).

However, a maskable interrupt requested in the DI state (IFF = “0”) is ignored and not accepted. (The CPU samples interrupt requests at the fall edge of CLK signal of the last bus cycle of each instruction.)

When an interrupt is accepted, the CPU first reads the interrupt vector from the built-in internal interrupt controller to find out the source of the interrupt request.

Then, the CPU checks if the request should be processed in the general-purpose interrupt processing mode or micro DMA processing mode, and proceeds to the appropriate process.

The interrupt vector is read in an internal operation cycle, so the bus cycle results in dummy cycles.

3.3.1 General Purpose Interrupt Processing

Figure 3.3 (2)shows the flow of general-purpose interrupt processing.

The CPU first saves the contents of the program counter PC and register AF (including the interrupt enable/disable flag just before the interrupt is issued) into the stack, and resets the interrupt enable/disable flag IFF to "0" (interrupt disable). Finally, it transfers the contents "V" of interrupt vector to the program counter and jumps to the interrupt processing program.

The overhead for the entire process from accepting an interrupt to jumping to an interrupt processing program is 20 states.

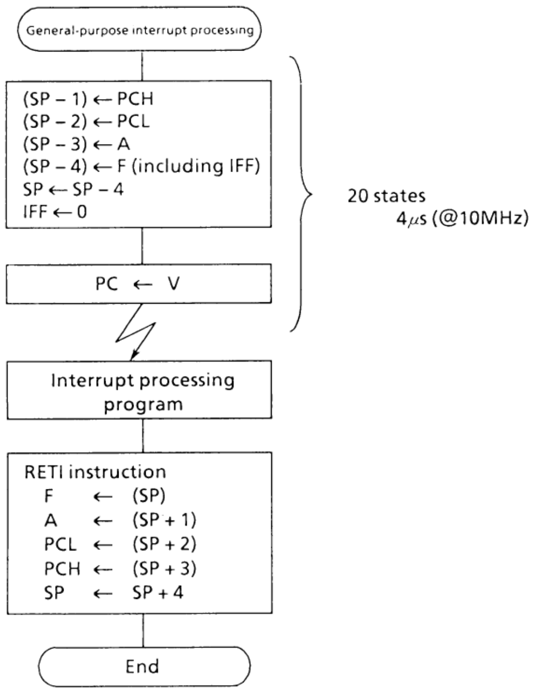


Figure 3.3 (2). General Purpose Interrupt Processing Flowchart

The interrupt processing program ends with RETI instruction for both non-maskable and maskable interrupts.

When this instruction is executed, the contents of the program counter PC and register AF will be restored from the stack (returns to the interrupt enable/disable flag just before the interrupt was issued).

When the CPU reads an interrupt vector, the interrupt request source acknowledges that the CPU accepts the request, and clears the request.

Non-maskable interrupts cannot be disabled by program. Maskable interrupts, however, can be enabled or disabled by programming. An interrupt enable/disable flip-flop (IFF) is provided on the bit 5 of Register F in the CPU.

Interrupts are enabled by setting IFF to "1" with the EI instruction and disabled by resetting IFF to "0" with the DI instruction. IFF is reset to "0" by the reset operation or the acceptance of any interrupt (including non-maskable interrupts).

Interrupt enabled with the EI instruction become effective when the instruction after the EI is executed.

Table 3.3 (1) shows the interrupt sources.

Table 3.3 (1) Interrupt Sources

Priority order	Type	Interrupt source	Vector value	Start address of general-purpose interrupt processing	Start address of Micro DMA processing parameter
1	Non-maskable	SWI instruction	08H	0008H	–
2		INTWD (watchdog)	10H	0010H	–
3	Maskable	INT0 (External input 0)	18H	0018H	FF18H
4		INTT0 (Timer 0)	20H	0020H	FF20H
5		INTT1 (Timer 1)	28H	0028H	FF28H
6		INTT2 (Timer 2)	30H	0030H	FF30H
7		INTT3 (Timer 3)	38H	0038H	FF38H
8		INTT4 (Timer 4)	40H	0040H	FF40H
9		INT1 (External input 1)	48H	0048H	FF48H
10		INTT5 (Timer 5)	50H	0050H	FF50H
11		INT2 (External input 2)	58H	0058H	FF58H
12		INTRX (Serial receiving end)	60H	0060H	FF60H
13		INTTX (Serial transmission end)	68H	0068H	FF68H

The “priority order” in the Table 3.3 (1) shows the order of the interrupt source to be acknowledge by the CPU when more than one interrupt are requested at one time.

If interrupt request of 4th and 5th orders are generated at the same time, for example, an interrupt of the “5th” priority is acknowledged after the “4th” priority interrupt processing has been completed by a RETI instruction. However, the “5th” priority interrupt can be acknowledged immediately by executing an EI instruction in a program that processes the “4th” priority interrupt.

The built-in interrupt controller only determines the priority of the interrupt sources which are to be accepted by the CPU when two or more interrupts are requested at a time.

It is, therefore, unable to compare the priority of interrupt being executed with the one being requested.

To enable other interrupt while an interrupt is being processed, set an interrupt enable/disable flag for the interrupt source to be enabled and execute EI instruction.

3.3.2 Micro DMA Processing

Figure 3.3 (3) shows the flowchart of the micro DMA processing. The CPU first loads parameters (addresses of source and destination, and transfer mode) necessary for the data transfer between memories from an address modified by an interrupt vector value. After the data transfer between memories according to these parameters, the parameters are updated and saved into the original locations. The CPU then decrements the number of transfers, and completes the micro DMA processing unless the result is “0”. If the number of transfers become “0”, the CPU proceeds to the general-purpose interrupt handling described in the previous section.

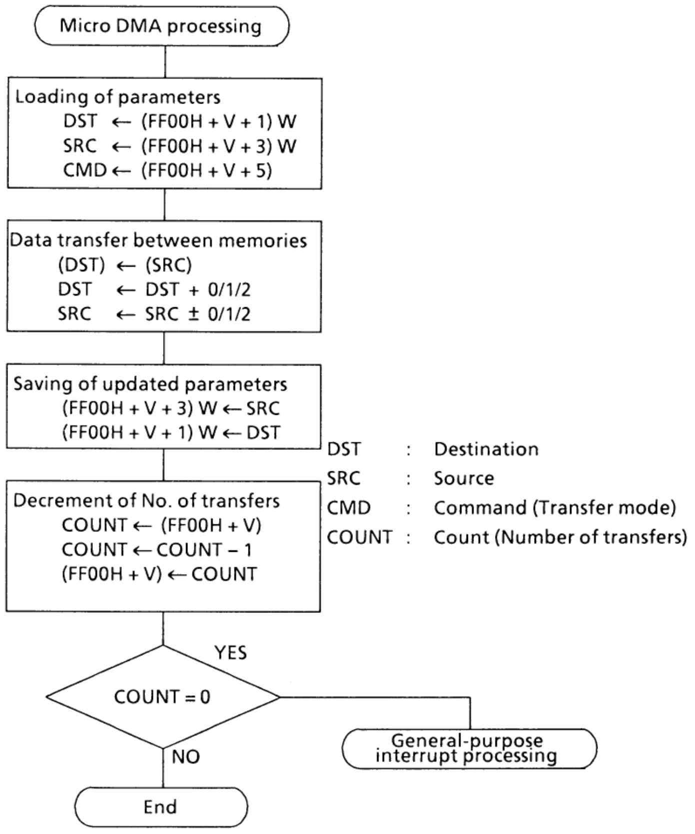


Figure 3.3 (3). Micro DMA Processing Flowchart

Since most interrupt processing involves only simple transfers, the micro DMA processing executes such processing only by hardware. Accordingly, the micro DMA processing can handle the interrupt in a higher speed than the conventional process using software. Naturally, the CPU registers are not affected by the micro DMA processing.

Figure 3.3 (4) shows the functions of parameters used in the micro DMA processing.

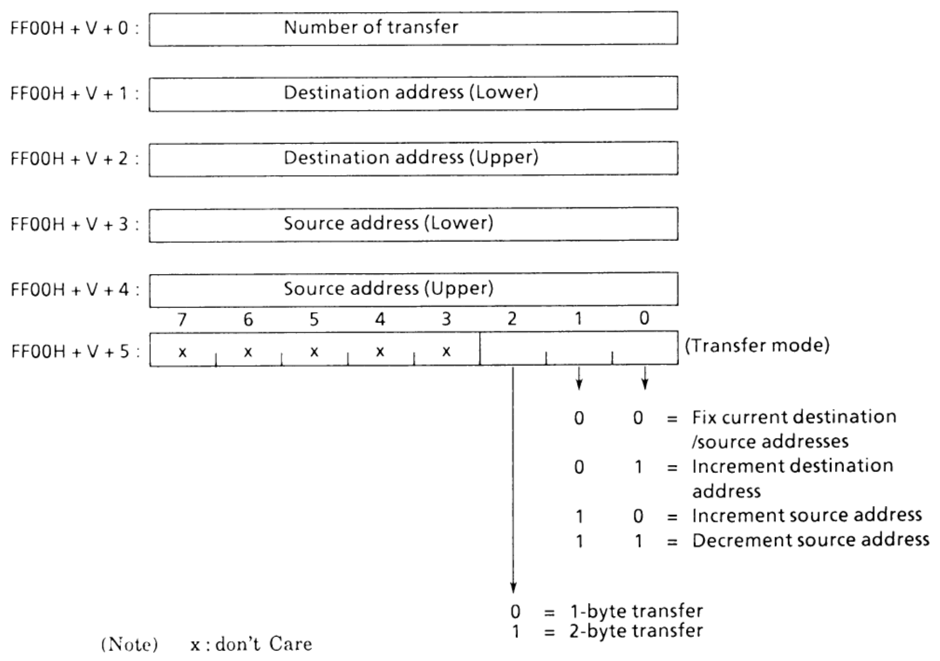


Figure 3.3 (4). Parameters for Micro DMA Processing

Parameters for micro DMA processing are located in the internal RAM area (See Table 3.3 (1) Interrupt Sources). The start address of each parameter becomes "FF00H + interrupt vector value", 6 bytes space are used for the parameter. When micro DMA processing is not used, the area can be freely used as user memory.

The parameters consist of the number of transfer, destination of addresses and source, and transfer mode. The number of transfer specifies the number of data transfers accepted by the micro DMA processing. A single time micro DMA processing transfers 1-byte or 2-byte data. The number of transfers is 256 when the number of transfers value is "00H". Both the destination and source addresses are specified by 2-byte data. The address space available for the micro DMA processing ranges from 0000H ~ FFFFH.

Bits 0 and 1 of the transfer mode indicates the mode updating the source and/or destination, and the bit 2 indicates the data length (one byte or two bytes).

Table 3.3 (2) shows the relation between the transfer modes and the decremented/incremented values of the destination/source addresses.

Table 3.3 (2) Addresses Updated by Micro DMA Processing

Transfer Mode	Function	Destination address	Source address
000	1-byte transfer: Fix the current destination /source addresses	0	0
001	1-byte transfer: Increment the destination address	+1	0
010	1-byte transfer: Increment the source address	0	+1
011	1-byte transfer: Decrement the source address	0	-1
100	2-byte transfer: Fix the current destination /sourc addresses	0	0
101	2-byte transfer: Increment the destination address	+2	0
110	2-byte transfer: Increment the source address	0	+2
111	2-byte transfer: Decrement the source address	0	-2

In the 2-byte transfer mode, data are transferred as follows:

(Destination address)←(Source address)
(Destination address + 1)←(Source address + 1)

Similar data transfers are made in the modes that “decrement the source address”, but the updated address are different as shown in the Table 3.3 (2).

Address increment/decrement modes are applied to memory address space and fixed addressing modes are applied to the I/O address space. Because of that, the micro DMA was designed for both I/O to memory transferes and memory to I/O transfers.

Figure 3.3 (5) shows an example of the micro DMA processing that handles data receiving of internal serial I/O.

This is an example of executing “an interrupt processing program after serial data receiving” after receiving 7-frame data (Assume 1 frame = 1 byte for this example) and saving them into the memory addresses from FF00H to FF06H.

```
CALL   SIOINIT           ;  Initial setting for serial receiving.
SET    3,(0FFF6H)        ;  Enable an interrupt for serial data receiving.
SET    3,(0FFF8H)        ;  Set the micro DMA processing mode for serial receiving
                            interrupt.
LD     (0FF60H),7        ;  Set the number of transfers = 7.
LDW    (0FF61H),0FF00H   ;  Set FF00H for the destination start address.
LDW    (0FF63H),0FFDFH   ;  Set FFDFH for the source (serial receiving buffer) address.


LD     (0FF65H),1        ;  Set the transfer mode (1-byte transfer; increment
                            destination address).
EI
 :
 :
ORG    0060H
+------------------------------------------------------------+
| Interrupt processing program after serial data receiving   |
+------------------------------------------------------------+

RETI
```

Figure 3.3 (5). Example of Micro DMA Processing

For the bus operation in the general-purpose interrupt processing and the micro DMA processing, see "Table 1.4 (2) Bus Operation for Executing Instructions" in the previous section "TLCS-90 CPU".

Execution time for micro DMA processing (when decremented number of transfers is not zero) is 46 states (5.75µs at 16MHz oscillation), regardless of whether 1-byte/2-byte transfer mode is used.

Figure 3.3 (6) shows the flowchart of overall interrupt processing.

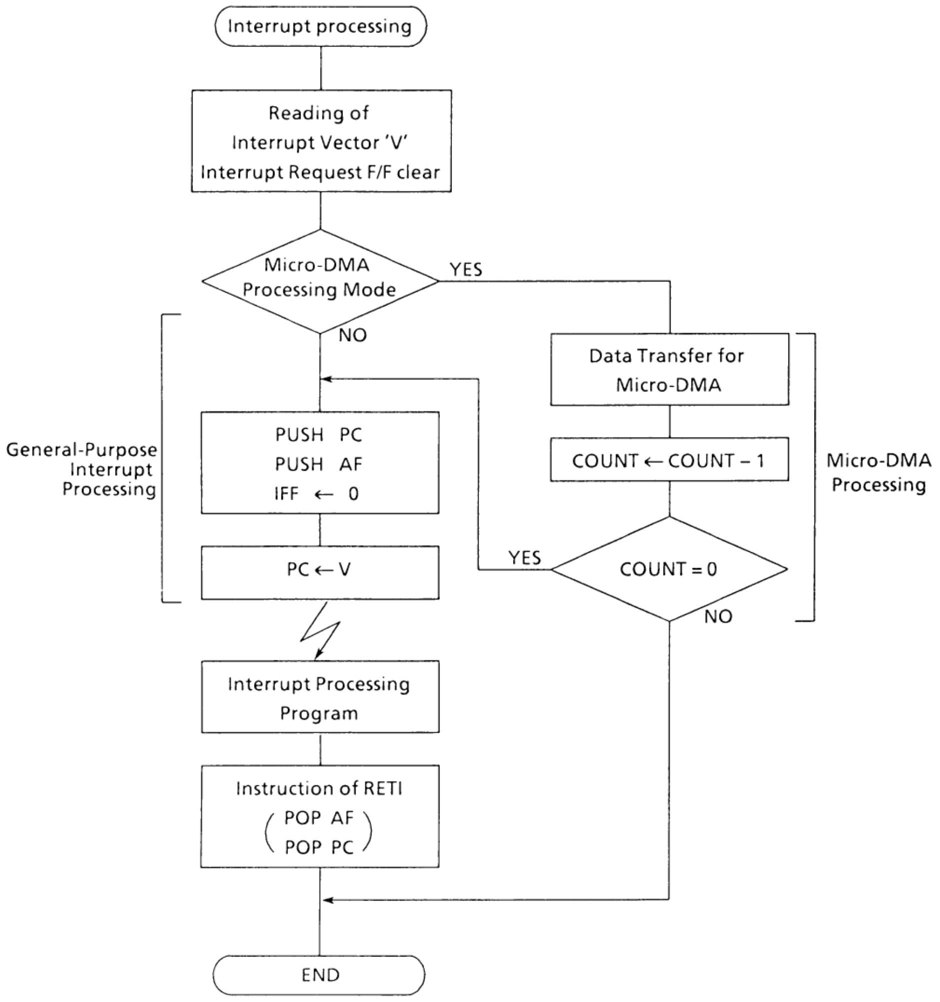

Interrupt processing
Reading of Interrupt Vector 'V' Interrupt Request F/F clear
Micro-DMA Processing Mode
YES
NO
Data Transfer for Micro-DMA
COUNT ← COUNT − 1
COUNT = 0
YES
NO
Micro-DMA Processing
General-Purpose Interrupt Processing
PUSH PC
PUSH AF
IFF ← 0
PC ← V
Interrupt Processing Program
Instruction of RETI
(POP AF
POP PC)
END

Figure 3.3 (6). Interrupt Processing Flowchart

3.3.3 Interrupt Controller

Figure 3.3 (8) shows the block diagram of interrupt circuit. The left half of this diagram shows the interrupt controller, and the right half includes the CPU's interrupt request signal circuit and HALT release signal circuit.

The interrupt controller has an interrupt request flip-flops, interrupt enable/disable flag, and micro DMA enable/disable flag for each o interrupt channel (total; 13 channels). The interrupt request flip-flop latches an interrupt request when it is issued from the peripheral devices. This flip-flop is reset to "0" when reset operation or interrupt is accepted by the CPU and the vector of that interrupt channel is read by the CPU, or when the CPU executes an instruction that clears the interrupt request for the specified channel (write "vector divided by 8" into the memory address FFEAH). For example, when executing

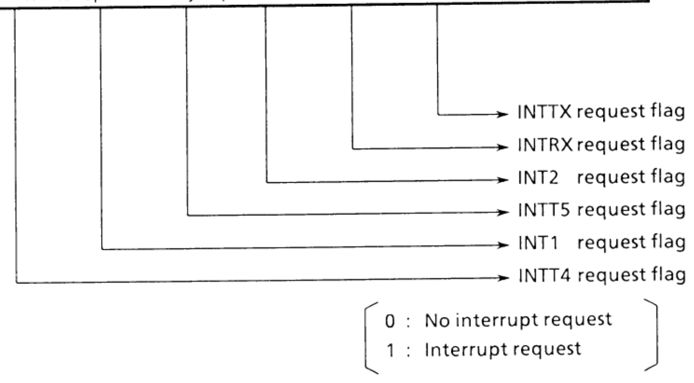
```
LD (0FFEAH), 38H/8,
```

the interrupt request flip-flop of the interrupt channel "INTT3" whose vector is 38H will be reset to "0". (Write to FFEAH even when clearing the interrupt request flag that is assigned to FFEBH.)

The status of an interrupt request flip-flop is can be known by reading the memory address FFEAH or FFEBH. "0" denotes there is no interrupt request, and "1" denotes that an interrupt is requested. Figure 3.3 (7) shows the bit configuration of the interrupt request flip-flops.

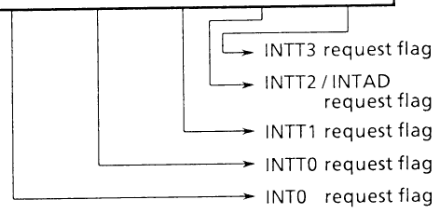
IRFL (FFEAH)

	7	6	5	4	3	2	1	0
bit Symbol	IRFT4	IRF1	IRFT5	IRF2	IRFRX	IRFTX	0	0
Read/Write	R (Write is possible only for IRF clear code.)							
After reset	0	0	0	0	0	0	0	0
Function	1: Interrupt is currently requested. (IRF is cleared by writing IRF clear code.)							



IRFH (FFEBH)

	7	6	5	4	3	2	1	0
bit Symbol				IRF0	IRFT0	IRFT1	IRFT2	IRFT3
Read/Write				R				
After reset				0	0	0	0	0
Function				Interrupt Request Flag 1: Interrupt is currently requested.				



Note: When "vector value/8" is written in memory address FFEAH, the specified interrupt request flag will be cleared.

Figure 3.3 (7). Interrupt Request Flip-flops

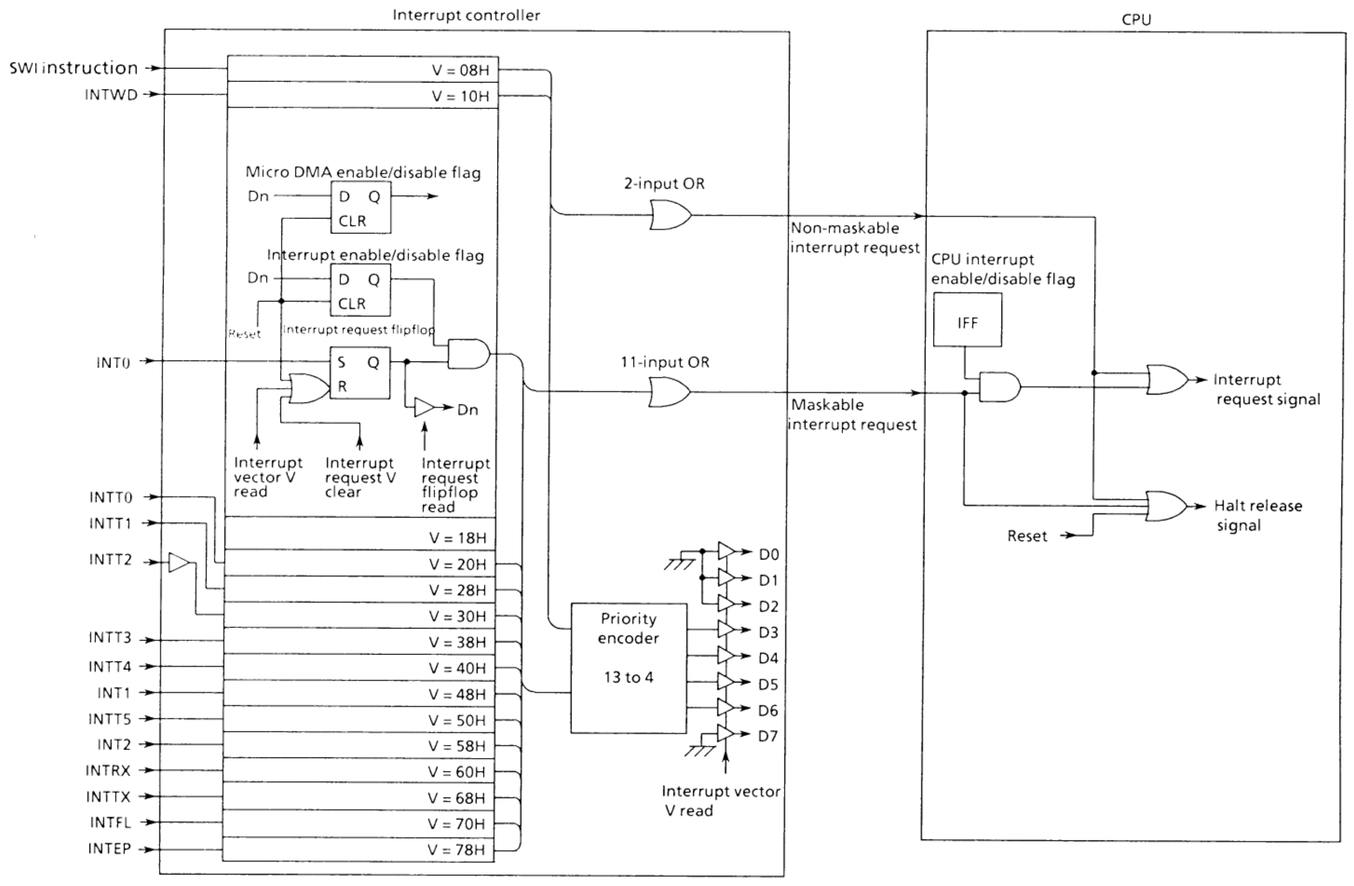


Figure 3.3 (8). Block Diagram of Interrupt Controller

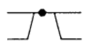
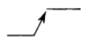
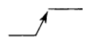
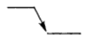
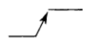
The interrupt enable/disable flags provided for all interrupt request channels are assigned to the memory address FFF6H or FFF7H. Interrupts for a channel are enabled by setting the flag to “1”. The flags are cleared to “0” by resetting.

Clear the interrupt enable flag in the DI status.

The micro DMA enable/disable flag also provided for each interrupt request channel is assigned to the memory address FFE8H or FFE9H. The interrupt processing for each channel is placed in the micro DMA processing mode by setting this flag to “1”. This flag is cleared to “0” (general-purpose interrupt processing mode) by resetting.

Figure 3.3 (9) shows the bit configurations for interrupt enable/disable flag and micro DMA enable/disable flag.

External interrupt features are as follows.

Interrupt	Common pin	Mode	How to set
INT0	P82	 Level	INTEH <EDGE> = 0
		 Rise edge	INTEH <EDGE> = 0
INT1	P83	 Rise edge	T4MOD <CAPM1, 0> = 0, 0 or 0, 1 or 1, 1
		 Fall edge	T4MOD <CAPM1, 0> = 1, 0
INT2	P84	 Rise edge	–

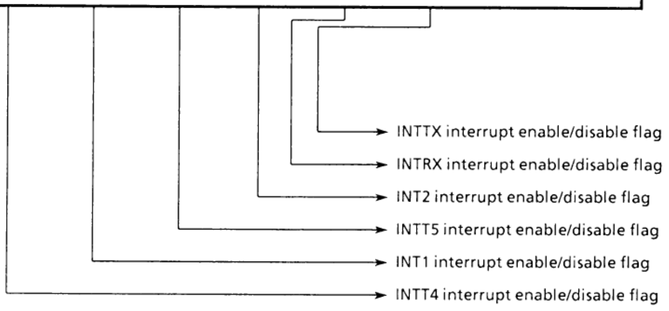
For the pulse width for the external interrupts, refer to “4.7 Interrupt Operations”.

Be careful that the following two are exceptional circuits.

INT0 Level mode	As the INT0 is not an edge based interrupt, the interrupt request flip-flop function is cancelled and thus an interrupt request from peripheral devices passes through S input of the flip-flop to become Q output. When the mode is changed over (from edge type to level type), the previous interrupt request flag will be cleared automatically. When the mode is changed from level to edge, the interrupt request flag set in the level mode is not cleared. Thus, use the following sequence to clear the interrupt request flag. DI SET 6, (0FFE7H) : Switch the mode from level to edge LD (0FFEAH), 03H : Clear interrupt request flag EI
INTRX	The interrupt request flip-flop is cleared only by reset or reading the serial channel receiving buffer, and cannot by an instruction.

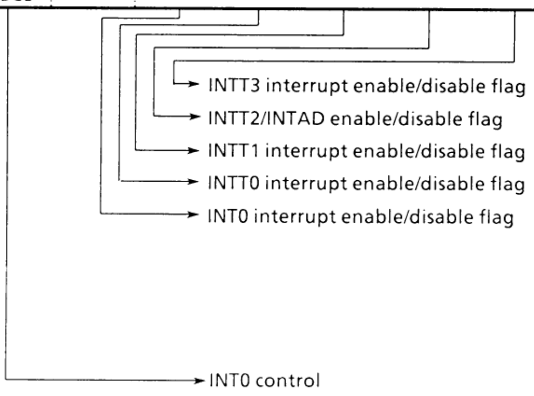
INTEL (FFE6H)

	7	6	5	4	3	2	1	0
bit Symbol	IET4	IE1	IET5	IE2	IERX	IETX	Fixed to "0"	Fixed to "0"
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	1: Enable 0: Disable							


INTTX interrupt enable/disable flag
INTRX interrupt enable/disable flag
INT2 interrupt enable/disable flag
INTT5 interrupt enable/disable flag
INT1 interrupt enable/disable flag
INTT4 interrupt enable/disable flag

INTEH (FFE7H)

	7	6	5	4	3	2	1	0
bit Symbol		EDGE		IE0	IET0	IET1	IET2	IET3
Read/Write		R/W		R/W				
After reset		0		0	0	0	0	0
Function		INT0 0: Level 1: EDGE		1: Enable 0: Disable				

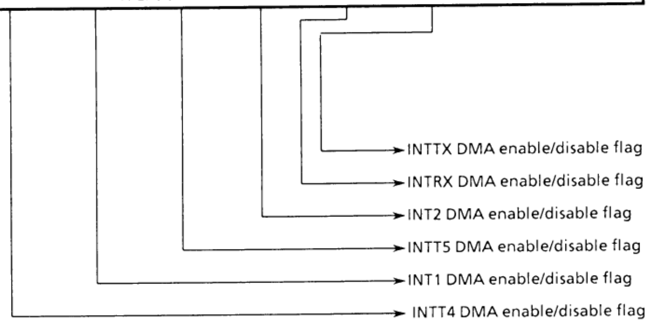

INTT3 interrupt enable/disable flag
INTT2/INTAD enable/disable flag
INTT1 interrupt enable/disable flag
INTT0 interrupt enable/disable flag
INT0 interrupt enable/disable flag
INT0 control

0	"H" level detection interrupt
1	Rise edge detection interrupt

Figure 3.3 (9). Interrupt Enable/Disable Flags

DMAEL
(FFE8H)

	7	6	5	4	3	2	1	0
bit Symbol	DET4	DE1	DET5	DE2	DERX	DETX	Fixed to "0"	Fixed to "0"
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	1: Enable 0: Disable							


INTTX DMA enable/disable flag
INTRX DMA enable/disable flag
INT2 DMA enable/disable flag
INTT5 DMA enable/disable flag
INT1 DMA enable/disable flag
INTT4 DMA enable/disable flag

DMAEH
(FFE9H)

	7	6	5	4	3	2	1	0
bit Symbol				DE0	DET0	DET1	DET2	DET3
Read/Write				R/W				
After reset				0	0	0	0	0
Function				1: Enable 0: Disable				

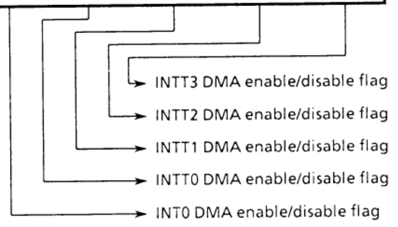

INTT3 DMA enable/disable flag
INTT2 DMA enable/disable flag
INTT1 DMA enable/disable flag
INTT0 DMA enable/disable flag
INT0 DMA enable/disable flag

Figure 3.3 (10). Micro DMA Enable/Disable Flag

3.4 Clock Switching Function

The TMP90C848F has the following clock modes.

① Low speed clock mode

The high-speed clock stops and the CPU operates in low-speed clock mode.

② High-speed clock mode

Both the high-speed clock and the low-speed clock generate, and the CPU operates in high-speed clock mode.

3.4.1 Operation

The clock mode setting register (CLKMOD) is assigned at bit 0 of the memory address “FFC7H” in the internal I/O register area.

Switching between low-speed clock mode and high-speed clock mode is controlled by commands.

When resetting DCLK (bit 0 of the clock mode setting register CLKMOD) is initialized to “0”, and the CPU operates in low-speed clock mode.

In low-speed clock mode, SLS (bit 3 of the clock mode register CLKMOD) is set to “0”; and in high-speed clock mode, “1”.

Because the watchdog timer counter is also used as the warm-up timer, the watchdog timer counter is cleared when the clock mode is switched from low-speed to high-speed.

CLKMOD (FFC7H)

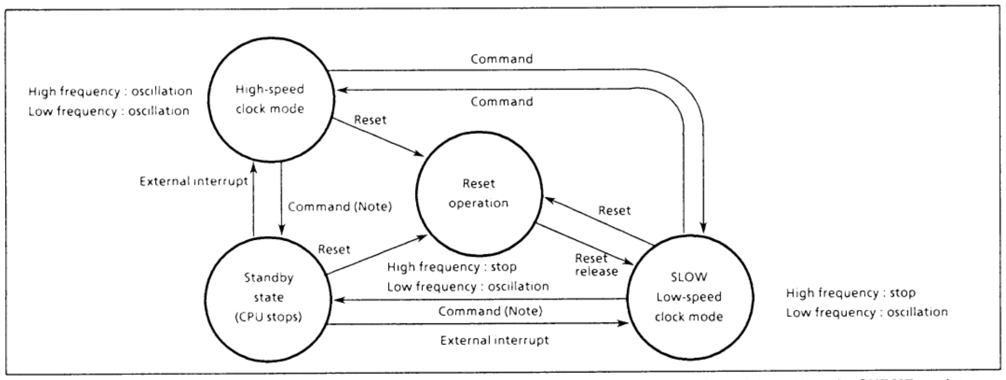
	7	6	5	4	3	2	1	0
bit Symbol			STBYD	WDRESE	SLS			DCLK
Read/Write			R/W	R/W	R			W
After reset			1	1	0			0
Function			1: Standby enable 0: Standby disable	1: Interrupt 0: RESET	0: low-speed 1: high-speed			0: low-speed 1: high-speed

Switching from low-speed clock mode to high-speed clock mode

In order to switch to WARM (bit 4 of watchdog mode register address FFD2), set the warm-up period to 0: 2^{11}/fc or 1: 2^{13}/fc, and DCLK to “1”. After completing the specified warm-up period, the clock mode is switched to high-speed clock mode. In this case, the low-speed clock does not stop.

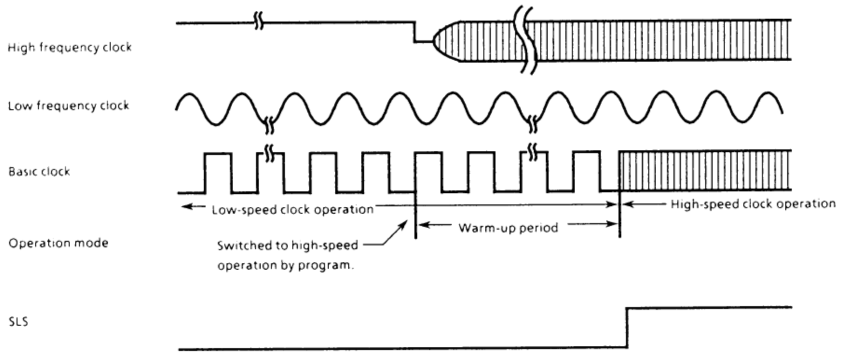
Switching from high-speed clock mode to low-speed clock mode

This clock mode is switched to low-speed by setting DCLK to “0”. The high-speed clock stops.



(Note) Specifying "0" to the STBYD register does not switch to HALT mode.

Figure 2.17. Transition of Operation Mode



High frequency clock
Low frequency clock
Basic clock
Low-speed clock operation
Warm-up period
High-speed clock operation
Operation mode
Switched to high-speed operation by program.
SLS

3.5 Standby Function

When a HALT instruction is executed, the TMP90C848F enters the RUN, IDLE1, or STOP mode according to the contents of the halt mode setting register. The features are as follows:

(1) RUN: Only the CPU halts, and the power consumption remains unchanged.

(2) IDLE1: Only the internal oscillator operates, while all other internal circuits halt. Power consumption is 1/10 or less than that during normal operation.

(3) STOP: All internal circuits halt, including the internal oscillator. The power consumption is extremely reduced.

The HALT mode setting register WDMOD <HALTM 1, 0> is assigned to the bits 2 and 3 of memory address FFD2H in the built-in I/O register area (all other bits are used to control other block functions). The RUN mode ("00") is entered by resetting.

These HALT states can be released by resetting an interrupt or resetting. Table 3.4 (2) shows how to release the HALT state. If the CPU is in the EI state for non-maskable or maskable interrupt, the interrupt will be acknowledged by the CPU and the CPU starts interrupt processing. If the CPU is in the DI state for maskable interrupt, the CPU restarts execution from the instruction following HALT instruction, but the interrupt request flag remains at "1".

Even when HALT state is released by reset operation, the state (including the built-in RAM) just before entering the HALT state can be retained. However, if HALT instruction has already been executed in the built-in RAM, the RAM contents may not be retained.

WDMOD (FFD2H)

	7	6	5	4	3	2	1	0
bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
Read/Write	R/W	R/W		R/W	R/W		R	R/W
After reset	1	0	0	0	0	0	Undefined	0
Function	1: WDT Enable	00 : 2^{16}/fc 01 : 2^{18}/fc 10 : 2^{20}/fc 11 : 2^{22}/fc	WDT Detecting time	Warming up time 0 : 2^{14}/fc 1 : 2^{16}/fc	Standby mode 00 : RUN mode 01 : STOP mode 10 : IDLE1 mode 11 : —		Undefined Inverts each time EXX instruction is executed.	1 : Drives the pin even in STOP mode

Figure 3.5 (1). HALT Mode Setting Register

3.5.1 Standby Disable Function

The standby disable function prevents the watchdog timer from being stopped and enabled it to escape from a malfunction (runaway), when the TMP90C848F enters standby mode because of a CPU runaway.

STBYD (bit 5 of clock mode register FFC7H) is reset to "0", standby mode is disabled and the TMP90C848F does not enter standby mode even if the HALT instruction is executed. The watchdog timer continues its operation, without stopping.

When standby mode is not used, STBYD is reset to "0". In this case, although standby mode is set by the halt mode register (HALTM), executing the halt instuction does not access standby mode (it accesses RUN mode).

Once STBYD is reset to "0", it cannot be set to "1" (standby mode enable). Standby mode becomes effective only by resetting. Table 3.5 (1) shows a standby state when the standby disable function is executed.

		7	6	5	4	3	2	1	0
CLKMOD (FFC7H)	bit Symbol			STBYD	WDRESE	SLS			DCLK
	Read/Write			R/W	R/W	R			W
	After reset			1	1	0			0
	Function			1: Standby enable 0: Standby disable	1: Interrupt 0: RESET	0: low-speed 1: high-speed			0: low-speed 1: high-speed

Figure 3.5 (2). Clock Mode Setting Register

Table 3.5 (1) Standby State at Standby Disable Function

	Standby mode		
STBYD	**RUN**	**IDLE1**	**STOP**
1	O	O	O
O	O	X	X

O: Executes specified standby mode.
X: Switched to RUN mode.

3.5.2 RUN Mode

Figure 3.5 (3) shows the timing for releasing the halt state by interrupts in the RUN/IDLE2 mode.

In the RUN mode, the system clock in the MCU continues to operate after the HALT instruction is executed. Only the CPU stops executing the instruction. Until the halt state is released, the CPU repeats dummy cycles. In the halt state, interrupt request is sampled with the rising edge of CLK signal.

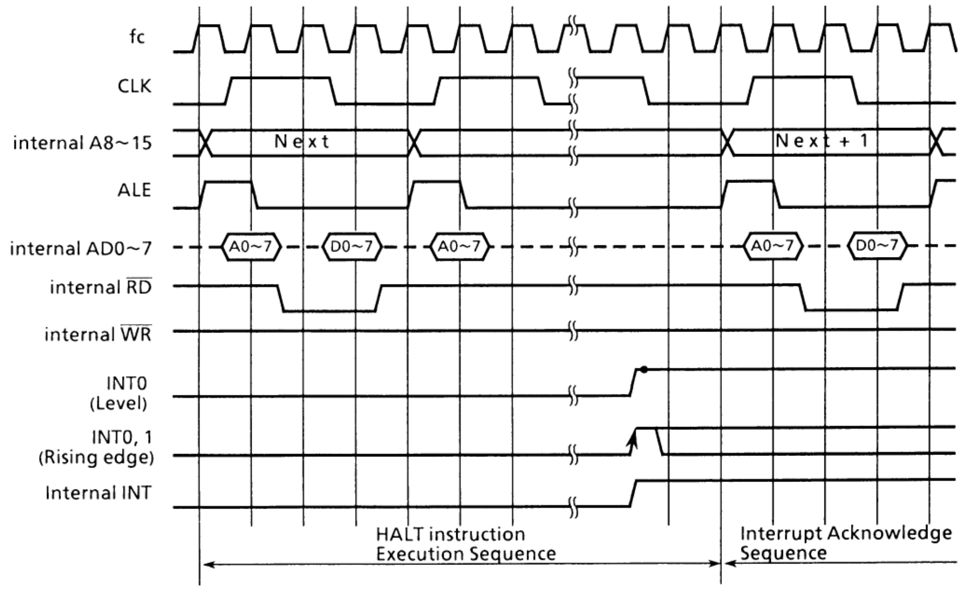


Figure 3.5 (3). HALT Release Timing Using Interrupts in RUN Mode

3.5.3 IDLE 1 Mode

Figure 3.5 (4) shows the timing for releasing the HALT state by interrupts in the IDLE1 mode.

In the IDLE1 mode, only the internal oscillator operates, the system clock inside MCU stops, and CLK signal is fixed to "1".

In the HALT state, interrupt request are sampled asynchronously with the system clock, whereas the HALT release (restart of operation) is performed synchronously with it.

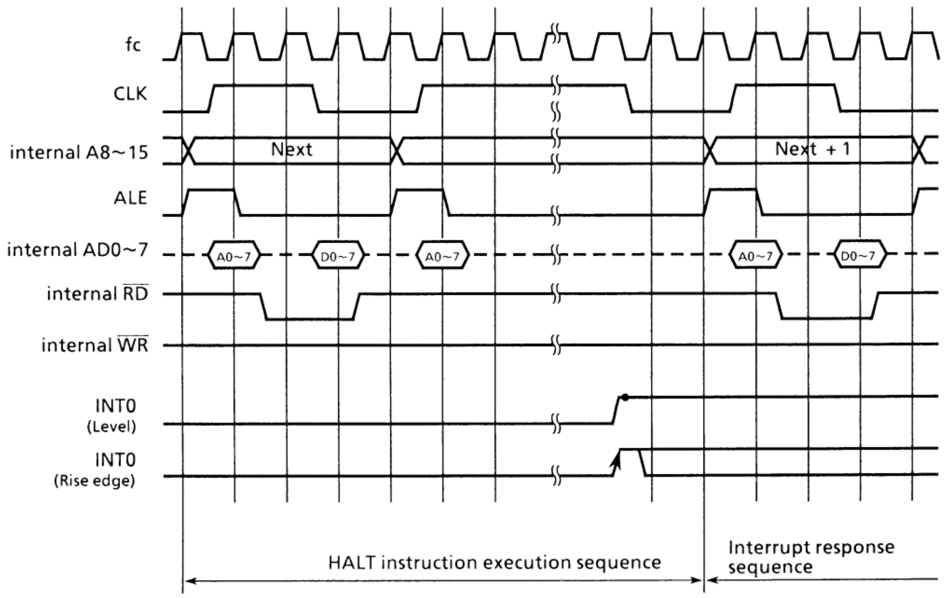


Figure 3.5 (4). HALT Release Timing Using Interrupts in IDLE1 Mode

3.5.4 STOP Mode

Figure 3.5 (5) shows the timing of HALT release caused by interrupts in STOP mode.

In the STOP mode, all internal circuits stop including the internal oscillator. When the STOP mode is activated, all pins except special ones are put in the high-impedance state, isolated from the internal operation of MCU. Table 3.4 (1) shows the state of each pin in the STOP mode. However, if WDMOD <DRVE> (drive enable: bit 0 of memory address FFD2H) of the built-in I/O register is set to "1", the pre-halt state of the pins can be retained. The register is cleared to "0" by reset operation.

When the CPU accepts an interrupt request, the internal oscillator first restarts. However, to get the stabilized oscillation, the system clock starts its output after the time set by the warming up counter has passed. WDMOD <WARM> (warming up: bit 4 at memory address FFD2H) is used to set the warming up time. Warming up is executed for 2^{14} clock oscillation time when this bit is set to "0", while 2^{16} clock oscillation time when set to "1". This bit is initialized to "0" by reset operation.

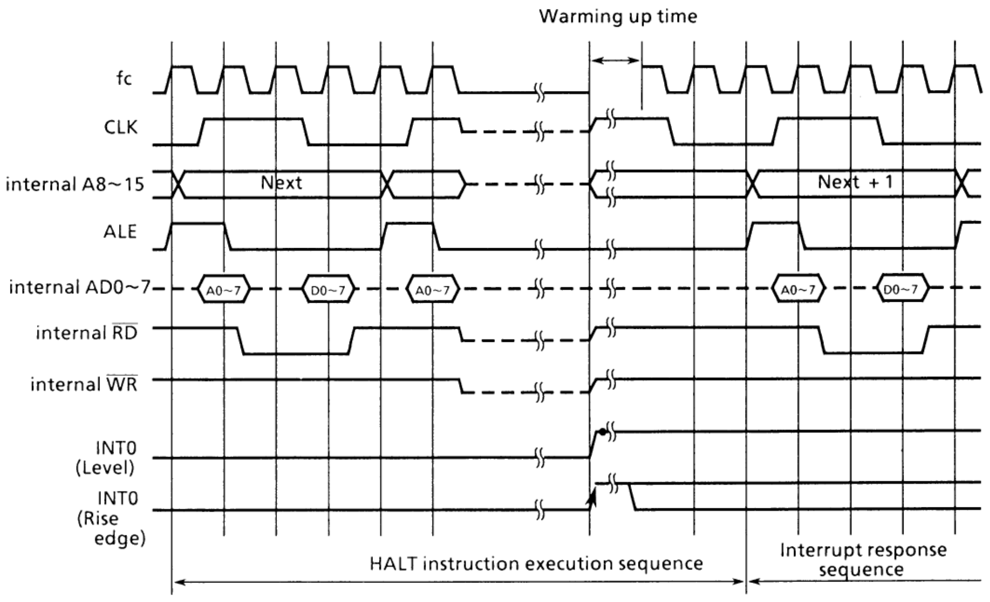


Figure 3.5 (5). HALT Release Timing Using Interrupt in STOP Mode

The internal oscillator can be also restarted by inputting $\overline{\text{RESET}}$ signal at “0” to the CPU.

However, the warming-up counter remains inactive in order to make the CPU rapidly operate when the power is turned on. Accordingly, wrong operation may occur due to unstable clocks immediately after the internal oscillator has restated. To release the HALT state by resetting the STOP mode, $\overline{\text{RESET}}$ signal must be kept at “0” for a sufficient period of time.

Table 3.5 (2) State of Pins in STOP Mode

	I/O	DRVE = 0	DRVE = 1
P0	Input mode	–	–
	Output mode	–	Output
P1	Input mode	Pull-up	Pull-up
	Output mode	Pull-up	Output
P2	Input mode	–	Input
	Output mode	–	Output
P5	Input mode	–	Input
		–	
P6	Input pin	–	Input
		–	
P3	Input pin	–	Input
	Output pin	–	Output
P4	Input mode	–	Input
	Output mode	–	Output
P7	Input mode	–	Input
	Output mode	–	Output
P82 (INT0)	Input mode	▨ Input	▨ Input
	Output mode	–	Output
P80, P81	Input mode	–	Input *
P83 ~ P85	Output mode	–	Output
$\overline{\text{RESET}}$	Input pin	▨ Input	▨ Input
ALE	Output pin	“0”	“0”
CLK	Output pin	Input	Input
X1′	Input pin	–	–
X2′	Output pin	“1”	“1”
X1	Input pin	–	–
X2	Output pin	“1”	“1”

*:	When in zero cross detect mode, intermediate bias is still applied to this pin.
–:	Indicates that input mode/input pin cannot be used for input and that the output mode/output pin have been set to high impedance.
▨ Input:	Input is enabled.
Input:	The input gate is operating. Fix the input voltage at either “0” or “1” to prevent input pin floating.
Output:	Output status.

3.6 Function of Ports

The TMP90C848F has a total of 62 I/O port pins. These ports function not only as the general-purpose I/O ports but also as the I/O ports for the internal CPU and built-in I/O. Table 3.6 shows the functions of these port pins.

Table 3.6 Functions of Ports

Port Name	Pin Name	No. of Pins	Direction	Direction Setting Unit	Pin Name for Internal Function
Port 0	P00 ~ P07	8	I/O	Bit	–
Port 1	P10 ~ P17	8	I/O	Bit	–
Port 2	P20 ~ P27	8	Output	Bit	HPWM0 ~ HPWM7
Port 3	P30	1	Output	–	TxD
	P31	1	Output	–	RxD
	P32	1	Output	–	
	P34	1	Output	–	
Port 4	P40 ~ P47	8	I/O	Bit	–
Port 5	P50 ~ P57	8	Input	–	AN0 ~ AN7
Port 6	P60 ~ P67	8	Input	–	AN8 ~ AN15
Port 7	P70 ~ P73	4	I/O	Bit	–
Port 8	P80	1	I/O	–	TO1
	P81	1	I/O	–	TO3
	P82	1	I/O	–	INT0
	P83	1	I/O	–	INT1/TI4
	P84	1	I/O	–	INT2/TI5
	P85	1	I/O	–	TO4

These port pins function as the general-purpose input/output ports by resetting. The port pins, for which input or output is programmably selectable, function as input ports by resetting. A separate program is required to use them for an internal function.

3.6.1 Port 0 (P00 ~ P07)

Port 0 is the 8-bit general-purpose I/O port P0, each bit of which can be set independently for input or output. The control register P0CR is used to set input or output. Reset operations clear all output latch and control register bits to “0” and set port 0 to the input mode.

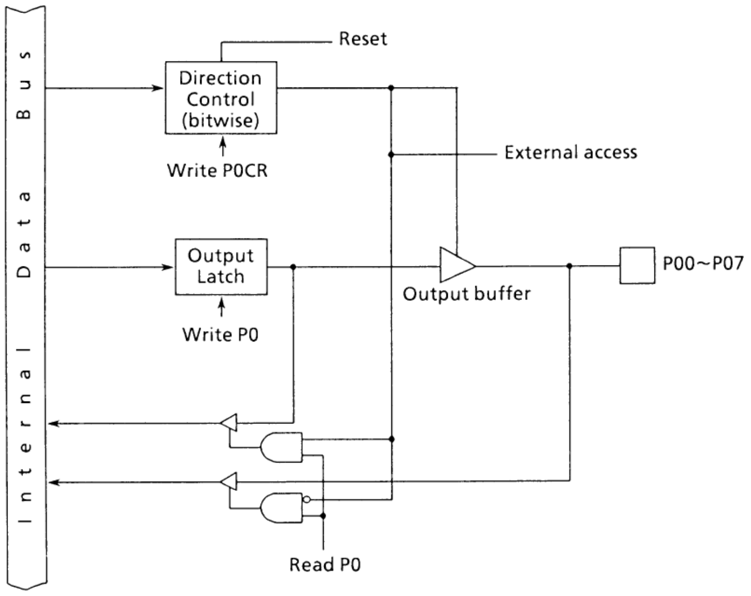


Figure 3.6 (1). Port 0 (P00 ~ P07)

3.6.2 Port 1 (P10 ~ P17)

Port 1 is an 8-bit general-purpose I/O port (P1: memory address FFC2H) which can be set to input or output bits. The port 1 control register (P1CR: memory address FFC3H) is used to set input or output.

Executing the reset operations clears all output latch and control register bits to “0” and sets all bits of port 1 to input mode.

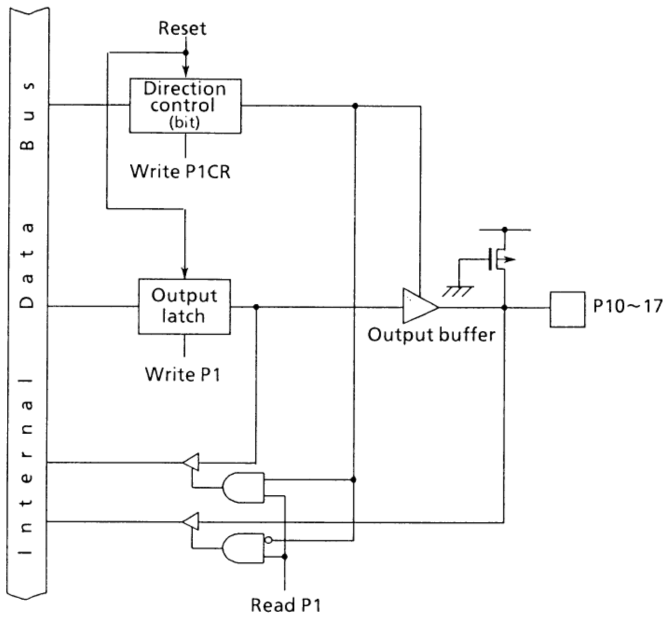


Figure 3.6 (2). Port 1 (P10 ~ P17)

Port 0 register

P0 (FFC0H)

	7	6	5	4	3	2	1	0
bit Symbol	P07	P06	P05	P04	P03	P02	P01	P00
Read/Write	R / W							
Resetting value	Input mode (output latch register is undefined)							

Port 0 control register

P0CR (FFC1H)

	7	6	5	4	3	2	1	0
bit Symbol	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
Read/Write	W							
Resetting value	0							
Function	0 : IN 1 : OUT (I/O specification in bit)							

Read modify write are not possible.

→ Port 0 I/O settings.

0	input
1	output

Port 1 register

P1 (FFC2H)

	7	6	5	4	3	2	1	0
bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R / W							
Resetting value	0 (Input mode)							

Port 1 control register

P1CR (FFC3H)

	7	6	5	4	3	2	1	0
bit Symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
Resetting value	0							
Function	0 : IN 1 : OUT (I/O specification in bit)							

Read modify write are not possible.

3.6.3 Port 2 (P20 ~ P27)

Port 2 is an 8-bit port (P2: memory address FFC4H). Executing the reset operation sets all output latch bits to “1” and sends “1” from the port.

In addition to the output port functin, port 2 also functions as the high-speed PWM output (HPWM0 ~ HPWM7). This is specified by the function register (P2FR: memory address FFC5H). The output port and high-speed PWM output can be selected by bits. Executing the reset operation resets all function register bits to “0” and sets the port to output port mode.

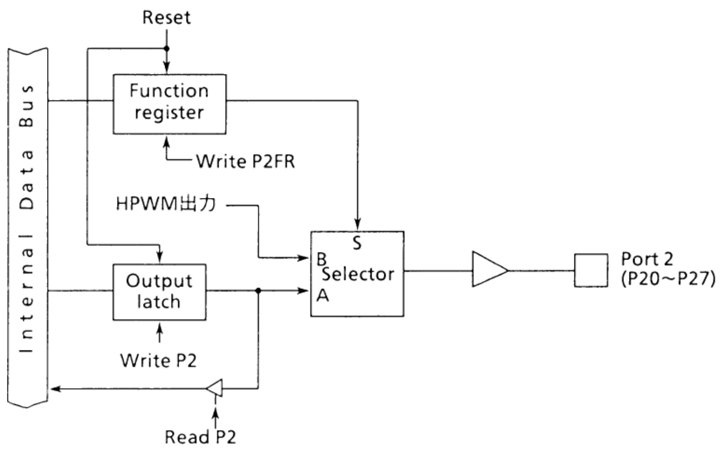


Figure 3.6 (3). Port 2

Port 2 register

		7	6	5	4	3	2	1	0
P2 (FFC4H)	bit Symbol	P27	P26	P25	P24	P23	P22	P21	P20
	Read/Write	W							
	After reset	1	1	1	1	1	1	1	1

Port 2 function register

		7	6	5	4	3	2	1	0
P2FR (FFC5H)	bit Symbol	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
	Read/Write	W							
	After reset	0							
	Function	Control P2 0: Output port 1: Output HPWM							

Registers for Port 2

3.6.4 Port 3 (P30 ~ P33)

Port 3 is a 4-bit general-purpose I/O port (P3: memory address FFC6H) which can be set to input or output by bits.

Executing the reset operation sets all output latch bits to "1" and sends "1" to the output port.

In addition to the I/O port function, P32 ~ P33 also function as the I/O port for the internal serial interface, and P30 ~ P31 function as the extrernal memory control. This is specified by the function register (P3FR: memory address FFC8H). Executing the reset operation resets all function register bits to "0" and sets the port to general-purpose I/O port mode.

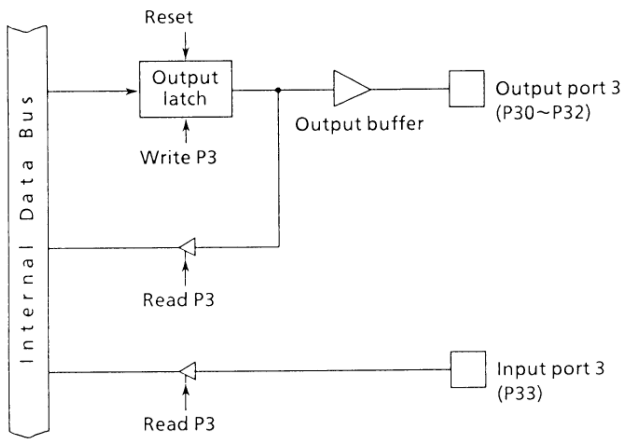


Figure 3.6 (4). Port 3

Port 3 register

		7	6	5	4	3	2	1	0
P3 (FFC6H)	bit Symbol					P33	P32	P31	P30
	Read/Write					R	W		
	After reset					入力モード	1	1	1

Port 3 function register

		7	6	5	4	3	2	1	0
P3FR (FFC8H)	bit Symbol	EXT					TxDC	ODE	
	Read/Write	W					R/W		
	After reset	0					0	0	
	Function	Control P1 0: IN/OUT 1: IN/ address					Control P32 0: PORT 1: Output TxD	P32 0: CMOS 1: OPEN DRAIN	

Figure 3.6 (5). Registers for Port 3

3.6.5 Port 4 (P40 ~ 47)

Port 4 is an 4-bit general-purpose I/O (P4: memory address FFC9H) which can be set to input or output by bits. The port 4 control register (P4CR: memory addres FFCAH) is used to set input or output.

Executing the reset operation sets all output latch bits to "1", resets all control register bits to "0", and sets the port to input port mode.

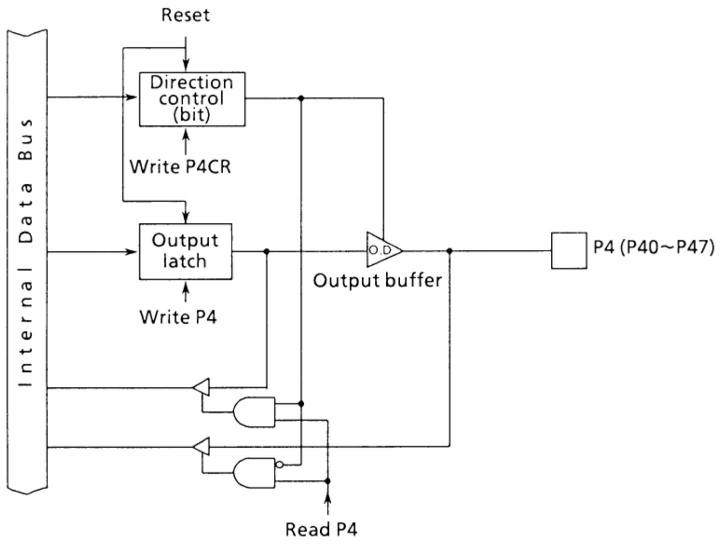


Figure 3.6 (6) A. Port 4

Port 4 is an open drain pin (as shown in Figure 3.6 (6) B): P40 ~ P43 function as an open drain sink, and P44 ~ P47 function as an open drain source.

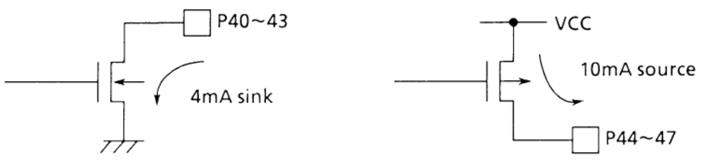


Figure 3.6 (6) B. Port 4 Open Drain

Port 4 register

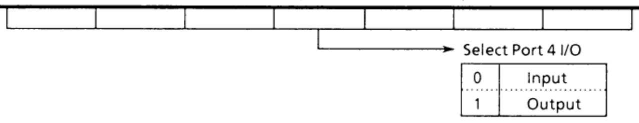
P4 (FFC9H)

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

Port 4 control register

P4CR (FFCAH)

	7	6	5	4	3	2	1	0
bit Symbol	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0 : IN 1 : OUT (Select I/O on bit basis)							



0	Input
1	Output

Figure 3.6 (7). Registers for Port 4

3.6.6 Port 5 (P50 ~ P57), Port 6 (P60 ~ P67)

Port 5 and port 6 are 8-bit input ports (P5: memory address FFCBH, P6: FFCCH) and also used as analog input pins (AN0 ~ AN15).

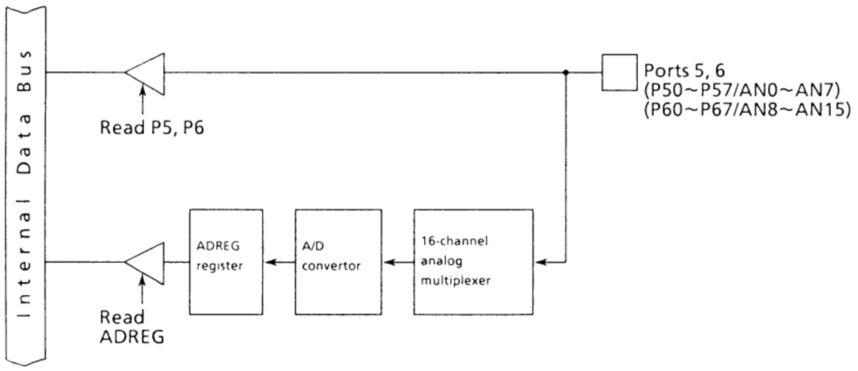


Figure 3.6 (8). Ports 5 and 6

Port 5 register

P5 (FFCBH)

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

Port 6 register

P6 (FFCCH)

	7	6	5	4	3	2	1	0
bit Symbol	P67	P66	P65	P64	P63	P62	P61	P60
Read/Write	R							
After reset	Input mode							

Figure 3.6 (9). Registers for Ports 5 and 6

3.6.7 Port (P70 ~ P73)

Port 7 is a 4-bit general-purpose I/O port, each bit of which can be set for input or output. The control register P67CR <P73C ~ P70C> is used to set for input or output. When reset, this control register will be cleared to “0”, placing the port 7 in the input mode. Each port is with programmable pull-up (FET).

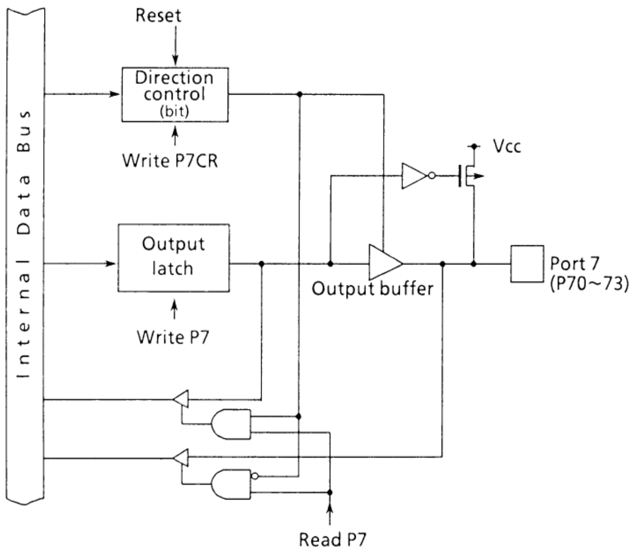


Figure 3.6 (10). Port 7

Port 7 register

P7 (FFCDH)		7	6	5	4	3	2	1	0
	bit Symbol		Fixed to "0"			P73	P72	P71	P70
	Read/Write					R / W			
	After reset		0			Input mode			

Port 7 control register

P7CR (FFCEH)		7	6	5	4	3	2	1	0
	bit Symbol					P73C	P72C	P71C	P70C
	Read/Write					W			
	After reset					0	0	0	0
	Function					0 : IN 1 : OUT (Select I/O on bit basis)			

Select Port 7 I/O

0	Input
1	Output

Figure 3.6 (11). Registers for Port 7

3.6.8 Port 8 (P80 ~ P85)

Port 8 is a 6-bit general-purpose I/O port (P8: memory address FFCFH). The port 8 control register (P8CR: bits 0 ~ 5 memory address FFD0) is used to set input or output.

In addition to the general-purpose port function, port 8 also functions as an interrupt request input, clock input for a timer/event counter, and timer output.

(1) P80, P81, P85

P80, P81 and P85 are general-purpose I/O ports and are also used as timer output pins (T01, 3, 4). The timer output function is set by the function register (P8FR: bits 0, 1 and of memory address FFD1H).

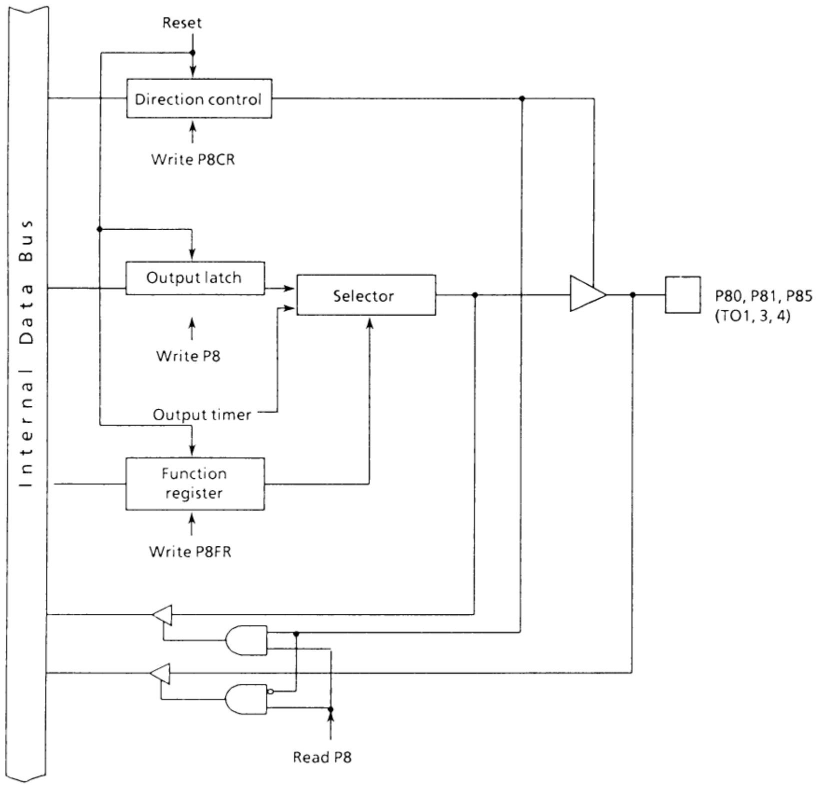


Figure 3.6 (12). Port 8 (P80, P81, P85)

(2) P82/INT0

P82 is a general-purpose I/O port and is also used as an external interrupt request input pin INT0. INT0 selects either an "H" level interrupt or a rising edge interrupt from the control register (INTEH: bit 6 of memory address FFE7H).

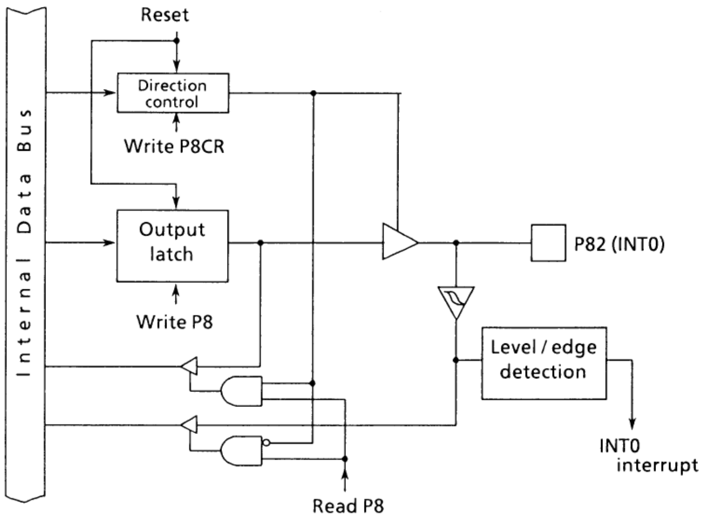


Figure 3.6 (13). Port 8 (P82)

(3) P83/INT1/TI4

P83 is a general-purpose input port and is also used as an external interrupt request input pin INT1 an a clock input pin TI4 for a timer/event counter.

P83 has a built-in zero-cross detection circuit which enables it to detect a zero cross, when it is connected to an external capacitor. The zero-cross detection function can be set to disable/enable by the function register (P8FR: bit 3 of memory address FFD1H). Executing the reset operation resets the control register to "0" and sets the zero-cross detection function to disable.

The rising/falling edge interrupt control is assigned to bit 3/ 4 of the 16-bit timer T4MOD register, which is also used for a capture control.

(4) P84/INT2/TI5

P84 is a general-purpose I/O port, similar to P83, and is also used as an external interrupt request input pin INT2 and a clock input pin TI5 for a timer/event counter. P84 also has a built-in zero-cross detection circuit which is set to disable/enable by the function register (P8FR: bit 4 of memory address FFD1H). When reset, the port is set to disable.

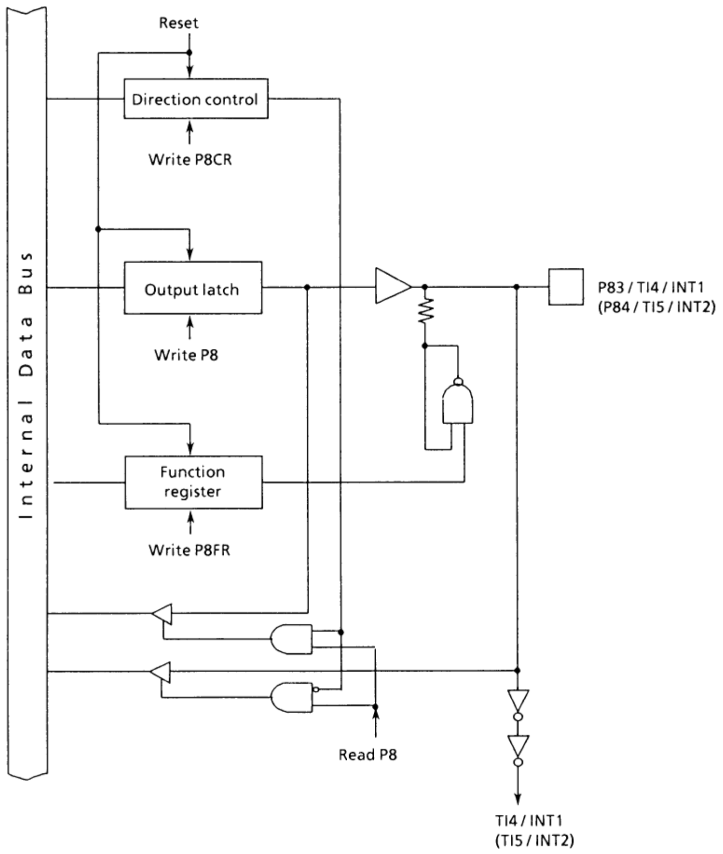


Figure 3.6 (14). Port (P83, P84)

P8 (FFCFH)

	7	6	5	4	3	2	1	0
bit Symbol			P85	P84	P83	P82	P81	P80
Read/Write			R / W					
After reset			Input mode					

P8CR (FFD0H)

	7	6	5	4	3	2	1	0
bit Symbol			P85	P84	P83C	P82C	P81C	P80C
Read/Write			W					
After reset			0	0	0	0	0	0
Function	0 : IN 1 : OUT (Select I/O on bit basis)							

Select Port 8 I/O

0	Input
1	Output

P8FR (FFD1H)

	7	6	5	4	3	2	1	0
bit Symbol			TO4S	ZCE2	ZCE1		TO3S	TO1S
Read/Write			W				W	
After reset			0	0	0		0	0
Function			P85/TO4 control 0: Port Output 1: TO4 Output	P84/INT2/TI5 Zero cross enable 0: Disable 1: Enable	P83/INT1/TI4 Zero cross enable 0: Disable 1: Enable		P81/PO3 control 0: Port Output 1: TO3 Output	P80/PO1 control 0: Port Output 1: TO1 Output

Setting P80 as the output of timer 1

0	Port Output
1	TO1 Output

P84/ INT2/ TI5 Zero cross enable

0	Disable
1	Enable

Setting P81 as the output of timer 3

0	Port Output
1	TO3 Output

Setting P85 as the output of timer 4

0	Port Output
1	TO4 Output

P83/ INT1/ TI4 Zero cross enable

0	Disable
1	Enable

Figure 3.6 (15). Registers for Port 8

3.7 8-bit Timers

The TMP90C848F contains four 8-bit timers (timers 0, 1, 2 and 3), each of which can be operated independently. The cascade connection allows these timers to be used as 16-bit timers.

The following four operating modes are provided for the 8-bit timers:

- 8-bit interval timer mode (4 timers)
- 16-bit interval timer mode (2 timers)
- 8-bit programmable square wave pulse generation (PPG: variable duty with variable cycle) output mode (2 timers)
- 8-bit pulse width modulation (PWM: variable duty constant with cycle) output mode (2 timers)

The upper two can be combined (two 8-bit timers and one 16-bit timer).

Figure 3.7 (1) shows the block diagram of the 8-bit timer (timer 0 and timer 1).

Timer 2 and timer 3 have the same circuit configuration as timer 0 and timer 1. Each interval timer consists of an 8-bit up-counter, 8-bit comparator, and 8-bit timer register. Besides, one timer flip-flop (TFF1 or TFF3) is provided for each pair of Timer 0 and timer 1 as well as timer 2 and timer 3.

Among the input clock sources for the interval timers, the internal clocks of øT1, øT4, øT16, and øT256 are obtained from the 9-bit prescaler shown in Figure 3.7 (2).

The operation modes and timer flip-flops of the 8-bit timer are controlled by five control registers T01MOD, T23MOD, TFFCR, TRUN, and TRDC.

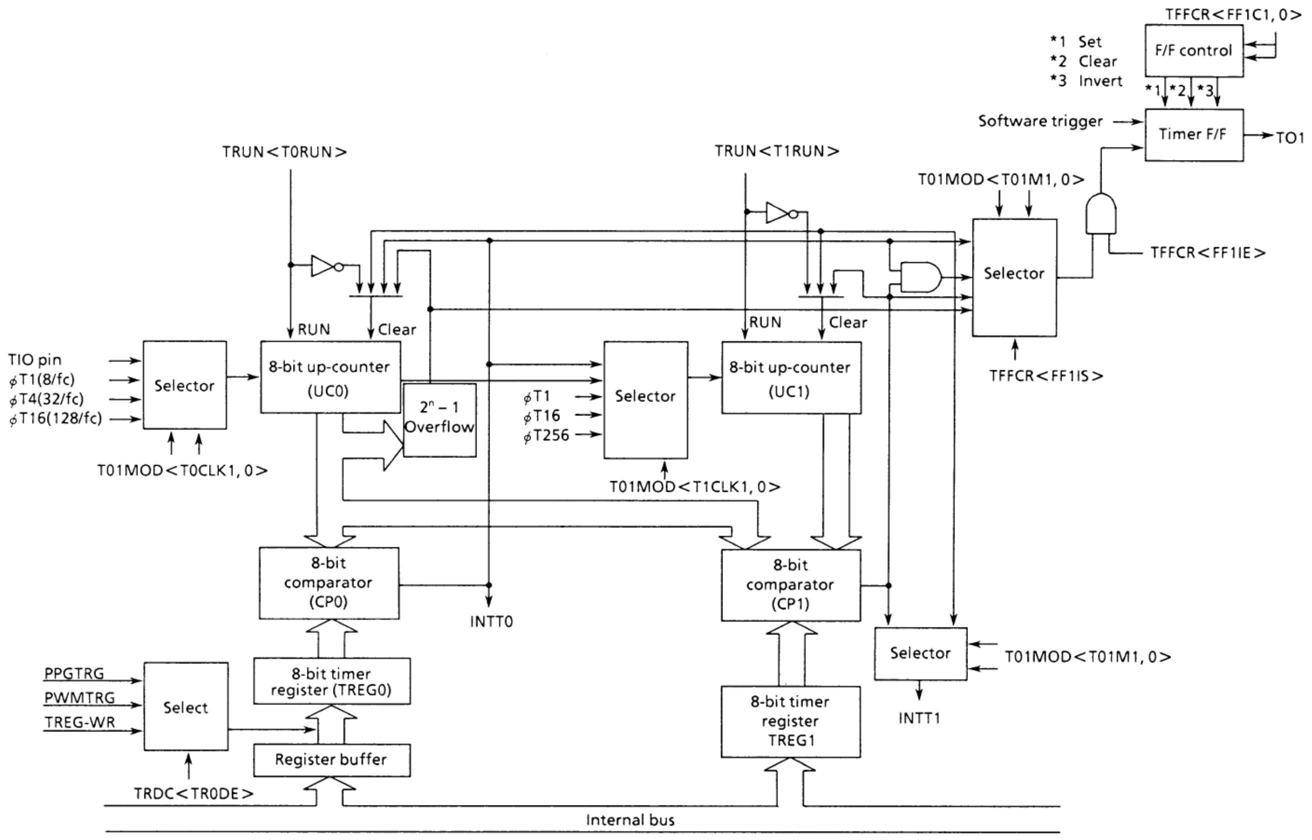


Figure 3.7 (1). Block Diagram of 8-bit Timers (Timers 0 and 1)

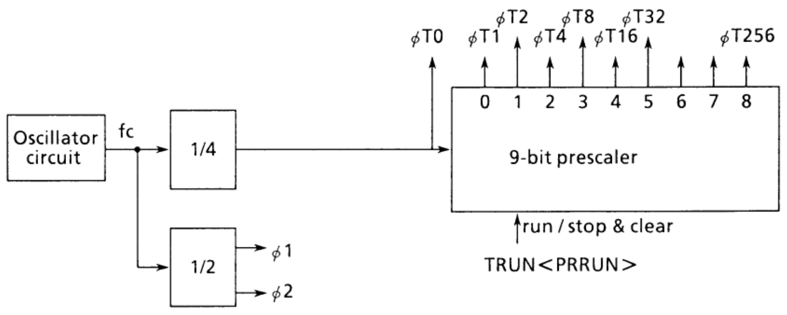
① Prescaler

This 9-bit prescaler generates the clock input to the 8-bit, 16-bit timer/event counters, and baud rate generators by further dividing the fundamental clock (fc) after it has been divided by 4 (fc/4).

Among them, 8-bit timer uses 4 types of clock: øT1, øT4, øT16 and øT256.

This prescaler can be run or stopped by the timer operation control register TRUN <PRRUN>. Counting starts when <PRRUN> is set to "0". Resetting clears <PRRUN> to "0", which clears and stops the prescaler.

	Cycle		
Input clock \ fc	1MHz	8MHz	10MHz
øT1 (8/fc)	8μs	1.0μs	0.8μs
øT4 (32/fc)	32μs	4.0μs	3.2μs
øT16 (128/fc)	128μs	16μs	12.8μs
øT256 (2048/fc)	2048μs	256μs	204.8μs



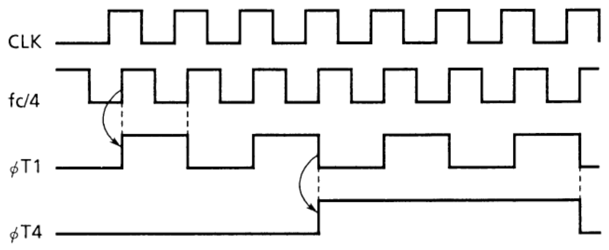


Figure 3.7 (2). Prescaler

② Up-counter

This is an 8-bit binary counter which counts up by the input clock pulse specified by the timer 0/timer 1 mode register T01MOD and timer 2/timer 3 mode register T23MOD.

The input clock of timer 0 and timer 2 is selected from the external clock from TI0 pin (commonly used as P44) and TI2 pin (commonly used as P45 or INT0) and the three internal clocks øT1 (8/fc), øT4 (32/fc), and øT16 (128/fc), according to the set value of T01MOD and T23MOD.

The input clock of timer 1 and timer 3 differs depending on the operation mode. When set to 16-bit timer mode, the overflow output of timer 0 and timer 2 is used as the input cock.

When set to any other mode than 16-bit timer mode, the input clock is selected from the internal clocks øT1 (8/fc), øT4 (32/fc), øT16 (128/fc),and øT256 (2048/fc) as well as the comparator output (match detection signal) of timer 0 and timer 2, according to the set value of T01MOD and T23MOD.

Example: When T01MOD <T01M1,0> = 01 the overflow output of timer 0 becomes the input clock of Timer 1. (16-bit timer.) When T01MOD <T01M1, 0> = 00 and T01MOD <T1CLK1,0> = 0, 1, øT1 (8/fc) becomes the input clock to timer 1. Operation mode is also set by the T01MOD and T23MOD. When reset, it is initialized to T01MOD <T01M1, 0> = 00, T23MOD <T23M1, 0> = 00, whereby the up-counter is placed in the 8-bit timer mode.

The counting, halt, and clear of up-counter can be controlled for each interval timer by the timer operation control register TRUN. When reset, all up-counters will be cleared to stop the timers.

③ Timer register

This is an 8-bit register for setting an interval timer. When the set value of timer registers TREG0, TREG1, TREG2 and TREG3 matches the value of up-counter, the comparator match detect signal becomes active. If the set value is 00H, this signal becomes active when the up-counter overflows.

Timer registers TREG0 and TREG2 are of double buffer structure, each of which makes a pair with register buffer.

The TREG0 and TREG2 control whether the double buffer should be enabled or disabled through the timer register double buffer control register TRDC <TR0DE, TR2DE>. It is disabled when <TR0DE>/TR2DE> =0 , and enabled when they are set to 1.

The timing to transfer data from the register buffer to the timer register in the double buffer enable state is the moment 2^n - 1 overflow occurs in PWM mode or the moment compare cycles will be equal in the PPG mode.

When reset, it will be initialized to <TR0DE>/TR2DE> = 0 to disable the double buffer. To use the double buffer, write data in the timer register, set <TR0DE> and TR2DE> to 1, and write the following data in the register buffer.

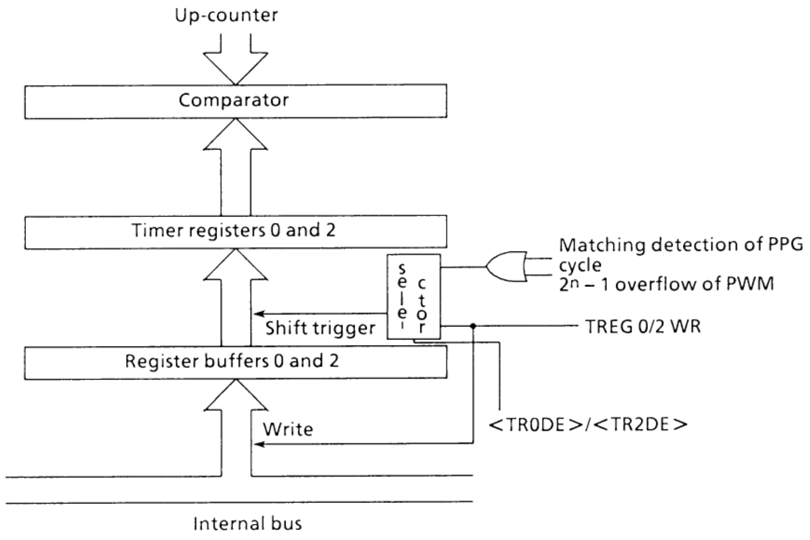


Figure 3.7 (3). Configuration of Timer Registers 0 and 2

Note: Timer register and the register buffer are allocated o the same memory address. When <TR0DE>/TR2DE> = 0, the same value is written in the register buffer as well as the timer register, while when <TR0DE>/TR2DE> = 1 only the register buffer is written.

The memory address of each timer register is as follows.

TREG0: FFD4H
TREG1: FFD5H
TREG2: FFD6H
TREG3: FFD7H

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

T01MOD
(FFD8H)

	7	6	5	4	3	2	1	0
bit Symbol	T01M1	T01M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
Read/Write	R/W		R/W		R/W		R/W	
After reset	0	0	0	0	0	0	0	0
Function	00: 8bit Timer 01: 16bit Timer 10: 8bit PPG 11: 8bit PWM		00: – 01: $2^6 - 1$ PWM 10: $2^7 - 1$ cycles 11: $2^8 - 1$		00: TO0TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI0 01: ϕT1 10: ϕT4 11: ϕT16	

Input clock of timer 0

00	External clock T10
01	Internal clock ϕT1
10	Internal clock ϕT4
11	Internal clock ϕT16

Input clock of timer 1

	T01MOD7, 6 ≠ 01	T01MOD7, 6 = 01
00	Comparator output of timer 0	Overflow output of timer 0 (16-bit timer mode)
01	Internal clock ϕT1	
10	Internal clock ϕT16	
11	Internal clock ϕT256	

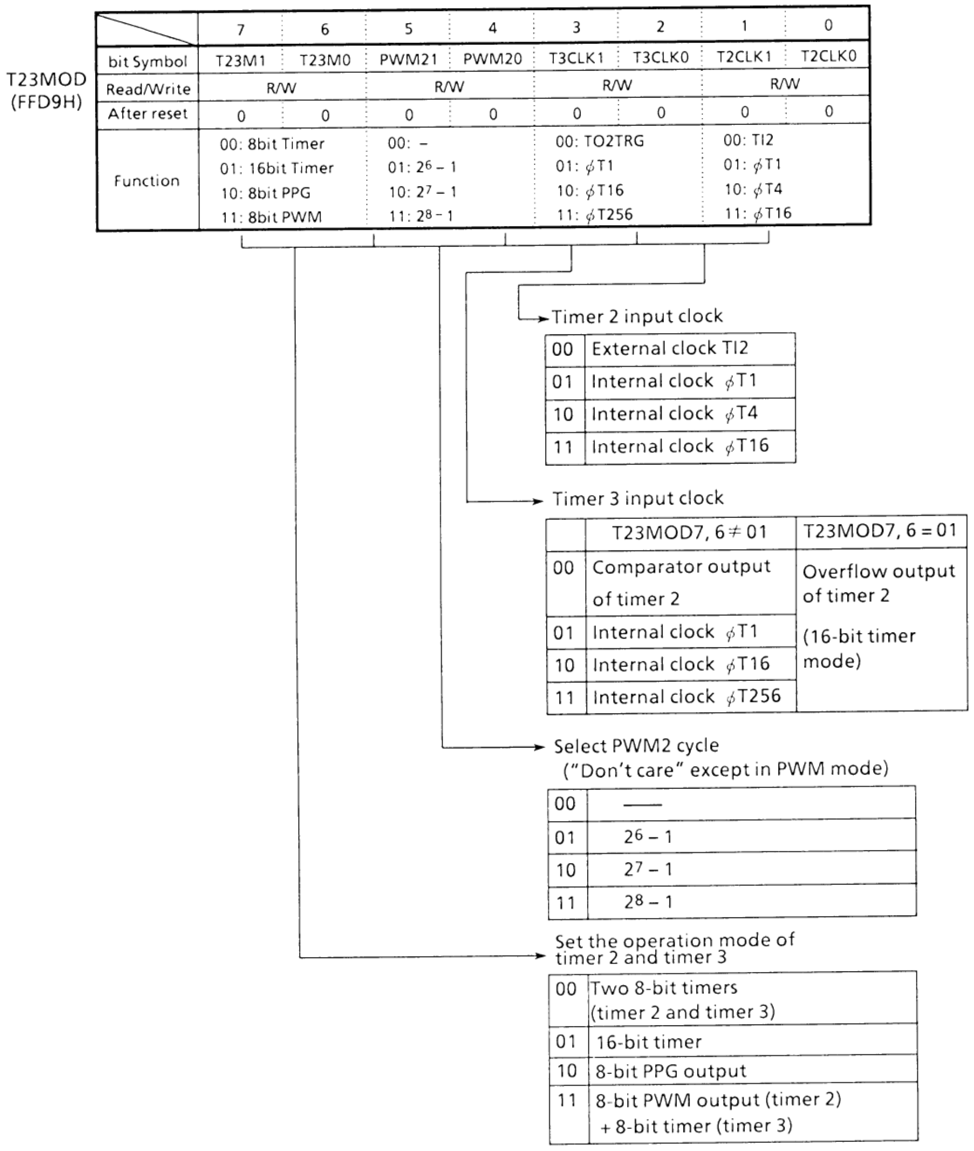
Select PWM0 cycle
("Don't care" except in PWM mode)

00	——
01	$2^6 - 1$
10	$2^7 - 1$
11	$2^8 - 1$

Set the operation mode of timer 0 and 1.

00	Two 8-bit timers (timer 0 and timer 1)
01	16-bit timer
10	8-bit PPG output
11	8-bit PWM output (timer 0) + 8-bit timer (timer 1)

Figure 3.7 (4). Timer 0/Timer 1 Mode Register (T01MOD)



T23MOD (FFD9H)

	7	6	5	4	3	2	1	0
bit Symbol	T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
Read/Write	R/W		R/W		R/W		R/W	
After reset	0	0	0	0	0	0	0	0
Function	00: 8bit Timer 01: 16bit Timer 10: 8bit PPG 11: 8bit PWM		00: – 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		00: TO2TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI2 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 2 input clock

00	External clock TI2
01	Internal clock ϕT1
10	Internal clock ϕT4
11	Internal clock ϕT16

Timer 3 input clock

	T23MOD7, 6 ≠ 01	T23MOD7, 6 = 01
00	Comparator output of timer 2	Overflow output of timer 2 (16-bit timer mode)
01	Internal clock ϕT1	
10	Internal clock ϕT16	
11	Internal clock ϕT256	

Select PWM2 cycle
("Don't care" except in PWM mode)

00	——
01	$2^6 - 1$
10	$2^7 - 1$
11	$2^8 - 1$

Set the operation mode of timer 2 and timer 3

00	Two 8-bit timers (timer 2 and timer 3)
01	16-bit timer
10	8-bit PPG output
11	8-bit PWM output (timer 2) + 8-bit timer (timer 3)

Figure 3.6 (5). Timer 2/Timer 3 Mode Register (T23MOD)

TFFCR (FFDAH)

	TFF3				TFF1			
	7	6	5	4	3	2	1	0
bit Symbol	FF3C1	FF3C0	FF3IE	FF3IS	FF1C1	FF1C0	FF1IE	FF1IS
Read/Write	W		R/W		W		R/W	
After reset	–		0	0	–		0	0
Function	00: Invert TFF3 01: Set TFF3 10: Clear TFF3 11: Don't care		1: TFF3 Invert Enable	0: Inverts by 8-bit timer 2 1: Inverts by timer 3	00: Invert TFF1 01: Set TFF1 10: Clear TFF1 11: Don't care		1: TFF1 Invert Enable	0: Inverts by 8-bit timer 0 1: Inverts by timer 1

Select inverse signal of timer flip-flop TFF1
("Don't care" except in 8-bit timer mode)

0	Inversion by timer 0
1	Inversion by timer 1

Invert of timer flip-flop TFF1

0	Disabled
1	Enabled

Control timer flip-flop TFF1

00	Invert the value of TFF1 (software inversion).
01	Set TFF1 to "1".
10	Clear TFF1 to "0".
11	Don't care (Always set at "11" when read)

Select inverse signal of timer flip-flop TFF3
(Don't care except in 8-bit timer mode)

0	Inversion by timer 2
1	Inversion by timer 3

Invert of timer flip-flop TFF3

0	Disabled
1	Enabled

Control of timer flip-flop TFF3

00	Inverts the value of TFF3 (software inversion).
01	Sets TFF3 to "1".
10	Clears TFF3 to "0".
11	Don't care (Always set at "11" when read)

Figure 3.7 (6). 8-Bit Timer Flip-flop Control Register (TFFCR)

TRUN
(FFDCH)

				16-bit timer	8-bit timer			
	7	6	5	4	3	2	1	0
bit Symbol			PRRUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Prescaler & Timer Run/Stop Control 0: Stop & Clear 1: Run (Count up)					

→ Operation of timer 0

0	Stop and clear
1	Count

→ Operation of timer 1

0	Stop and clear
1	Count

→ Operation of timer 2

0	Stop and clear
1	Count

→ Operation of timer 3

0	Stop and clear
1	Count

→ Operation of 16-bit timer (timer 4)

0	Stop and clear
1	Count

→ Operation of prescaler

0	Stop and clear
1	Count

Figure 3.7 (7). Timer Operation Control Register (TRUN)

TRDC
(FFDBH)

	7	6	5	4	3	2	1	0
bit Symbol	MIT					TR4DE	TR2DE	TR0DE
Read/Write	R/W						R/W	
After reset	0					0	0	0
Function	0: Timer 2 or 3 1: Timer 4					0: Double Buffer Disable 1: Double Buffer Enable		

Control the double buffer of timer register 0

0	Disabled
1	Enabled

Control the double buffer of timer register 2

0	Disabled
1	Enabled

Control the double buffer of timer register 4

0	Disabled
1	Enabled

Select the shift trigger of port 7
(M10~M13)

0	Timer 2 or 3
1	Timer 4

(Shift trigger generating timer for stepping motor control/pattern generation port)

Figure 3.7 (8). Timer Register Double Buffer Control Register (TRDC)

④ Comparator

A comparator compares the value in the up-counter with the values to which the timer register is set. When they match, the up-counter is cleared to zero and an interrupt signal (INTT0 ~ INTT3) is generated. If the timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

⑤ Timer flip-flop (timer F/F)

The status of the timer flip-flop is inverted by the match detect signal (comparator output) of each interval timer and the value can be output to the timer output pins TO1 (also used as P80) and TO3 (also used as P81).
A timer F/F is provided for each pair of timer 0 and timer 1 as well as that of timer 2 and timer 3 and is called TFF1 and TFF3. TFF1 is output to TO1 pin, while TFF3 is output to TO3 pin.

The operation of 8-bit timers will be as follows:

(1) 8-bit Timer Mode
Four interval timers 0, 1, 2 and 3 can be used independently as an 8-bit interval timer. All interval timers operate in the same manner, and thus, only the operation of timer 1 will be explained below.

① Generating interrupts in a fixed cycle
To generate timer 1 interrupt at constant intervals using timer 1 (INTT1), first stop timer 1, then set the operation mode, input clock, and synchronization to T01MOD and TREG1, respectively. Then, enable interrupt INTT1 and start the counting of timer 1.

Example: To generate timer 1 interrupt every 40 microseconds at fc = 10MHz, set each register in the following manner.

```
            MSB           LSB
            7 6 5 4 3 2 1 0
TRUN    ←   - - - - - - 0 -      Stop timer 1, and clear it to "0".
T01MOD  ←   0 0 X X 0 1 - -      Set the 8-bit timer mode, and select φT1 (0.8 μs @
                                 fc = 10 MHz) as the input clock.
TREG1   ←   0 1 0 1 0 0 0 0      Set the timer register at 40 μs φT1 = 50.
INTEH   ←   X - - - - 1 - -      Enable INTT1.
TRUN    ←   X X 1 - - - 1        Start timer 1 counting.
```

(Note) X: Don't Care -: No change

Use the following table for selecting the input clock:

Table 3.7 (1) 8-Bit Timer Interrupt Cycle and Input Clock

Interrupt cycle @fc = 10MHz	Resolution	Input clock
0.8μs ~ 204.8μs	0.8μs	øT1 (8/fc)
12.8μs ~ 3.2768μs	12.8μs	øT16 (128/fc)
204.8μs ~ 52.4288ms	204.8μs	øT256 (2048/fc)

② Generating a 50% duty square wave pulse

The timer flip-flop is inverted at constant intervals, and its status is output to a timer output pin TO1.

Example: To output a 4.8μs square wave pulse from TO1 pin at fc = 10MHz, set each register in the following procedures. Either timer 0 or timer 1 may be used, but this example uses timer 1.

```
              MSB           LSB
              7 6 5 4 3 2 1 0
TRUN    ←     - - - - - - 0 -      Stop timer 1, and clear it to "0".
T01MOD  ←     0 0 X X 0 1 - -      Set the 8-bit timer mode, and select ϕT1 (0.5 μs @
                                   fc = 16 MHz) as the input clock.
TREG1   ←     0 0 0 0 0 0 1 1      Set the timer register at 3.0 μs ÷ ϕT1 ÷ 2 = 3.
                                   Clear TFF1 to "0", and set to invert by the match
TFFCR   ←     - - - - 1 0 1 1      detect signal from timer 1.

P4CR    ←     - - - - - - - 1   }  Select P40 as TO1 pin.
P4FR    ←     - - X X - - - 1   }
TRUN    ←     X X 1 - - - 1 -      Start timer 1 counting.

(Note)   X; Don't care     - ; No change
```

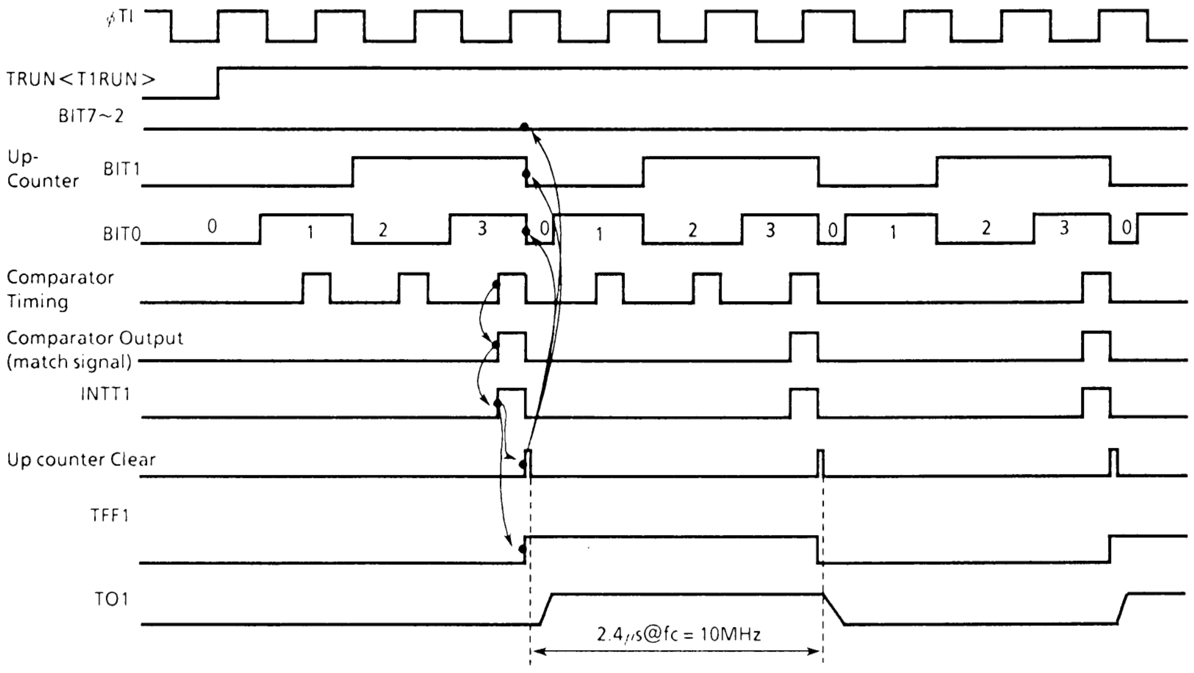
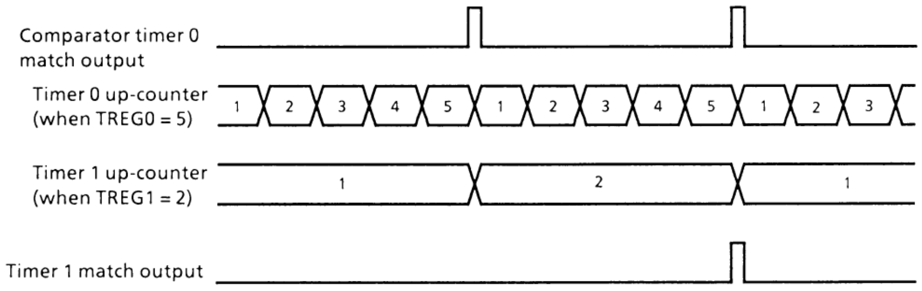


Figure 3.7 (9). Square Wave (50% Duty) Output Timing Chart

③ Making timer 1 count up by match signal from timer 0 comparator

Set the 8-bit timer mode, and set the comparator output of timer 0 as the input clock to timer 1.



Comparator timer 0 match output
Timer 0 up-counter (when TREG0 = 5)
1 2 3 4 5 1 2 3 4 5 1 2 3
Timer 1 up-counter (when TREG1 = 2)
1 2 1
Timer 1 match output

Figure 3.7 (10)

④ Output inversion with software

The value of timer flip-flop (timer F/F) can be inverted, independent of timer operation.
Writing “00” into TFFCR <FF1C1, 0> inverts the value of TFF1, and writing “00” into TFFCR <FF3C1, 0> inverts TFF3.

⑤ Initial setting of timer flip-flops (timer F/F)

The value of timer F/F can be initialized to “0” or “1”, independent of timer operation.
For example, write “10” in TFFCR <FF1C1, 0> to clear TFF1 to “0”, while write “01” in TFFCR <FF3C1, 0> to set TFF1 to “1”.

Note: The value of timer F/F and timer register cannot be read.

(2) 16-bit timer mode

A 16-bit interval timer is configurated by using the pair of timer 0 and timer 1 or that of time 2 and timer 3.

As the above two pairs operate in the same manner, only the case of combining timer 0 and timer 1 is discussed.

To make a 16-bit interval timer by cascade connection timer 0 and timer 1, set timer 0/timer 1 mode register T01MOD <T01M1, 0> to "0, 1".

When set in 16-bit timer mode, the overflow output of timer 0 will become the input clock of timer 1, regardless of the set value of T01MOD <T1CLK1, 0>. Table 3.6 (2) shows the relation between the cycle of timer (interrupt) and the selection of input clock.

Table 3.7 (2) 16-bit Timer (Interrupt) Cycle and Input Clock

Interrupt cycle (@fc = 10MHz)	Resolution	Input clock
0.8µs ~ 52.43ms	0.8µs	øT1 (8/fc)
12.8µs ~ 838.86ms	12.8µs	øT16 (128/fc)

The lower 8 bits of the timer (interrupt) cycle are set by the timer register TREG0, and the upper 8 bits are set by TREG1. Note that TREG0 always must be set first (Writing data into TREG0 disables the comparator temporarily, which is restarted by writing data into TREG1).

Setting example: To generate interrupts INTT1 every 1 seconds at fc = 8 MHz,
set the following values for timer register TREG0 and TREG1.
When counting with input clock of øT16 (16µs @ 8MHz)
1sec ÷16µs = 62500 = F424H
Therefore, set
TREG1 = F4H and TREG0 = 24H, respectively.

The comparator match signal is output from timer 0 each time the up-counter matches UC0, where the up-counter UC0 is not be cleared.

With the timer 1 comparator, the match detect signal is output at each comparator timing when up-counter UC1 and TREG1 values match. When the match detect signal is output simultaneously from both comparators of timer 0 and timer 1, the up-counters UC0 and UC1 are cleared to "0", and the interrupt INTT1 is generated. If inversion is enabled, the value of the timer flip-flop TFF1 is inverted

	Timer 0			Timer 1		
	INTT0	TO1	match	INTT1	TO1	match
16 bit Timer Mode (Count-up Timer 1 by overflow of Timer 0)	Interrupt is generated.	Can't output (Can't Output the matching with TREG0)	TREG0 (Continue counting when match)	Interrupt is generated.	Can output (Can't Output the matching with both TREG0 and TREG1)	TREG1 $*2^8$ + TREG0(16bit) (Cleared by matching with both registers)
8 bit Timer Mode (Count-up Timer 1 by matching of Timer 0)	Interrupt is generated.	Can output (Timer 0 or Timer 1)	TREG0 (Clear when match)	Interrupt is generated.	Can output (Timer 0 or Timer 1)	TREG1* TREG0 (Multiplied Value) (Cleared by matching)

Example: When TREG1 = 04H and TREG0 = 80H,

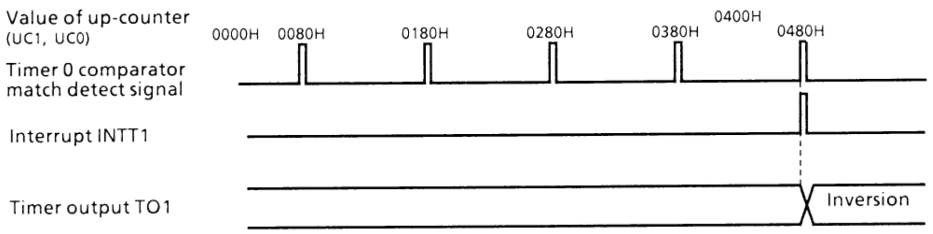
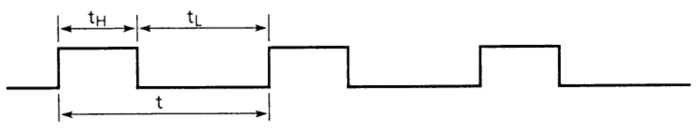


Figure 3.7 (11)

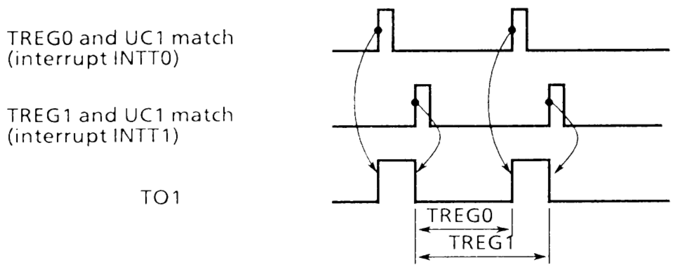
(3) 8-bit PPG (Programmable Pulse Generation) Mode

Square wave pulse can be generated at any frequency and duty by timer 1 and timer 3. The output pulse may be either low-active or high-active.

In this mode, timer 0 and timer 2 cannot be used. Timer 1 outputs pulse to TO1 pin (also used as P80), and timer 3 outputs pulse to TO3 pin (also used as P81).



As an example, the case of timer 1 will be explained below. (Timer 3 also functions in the same way.)



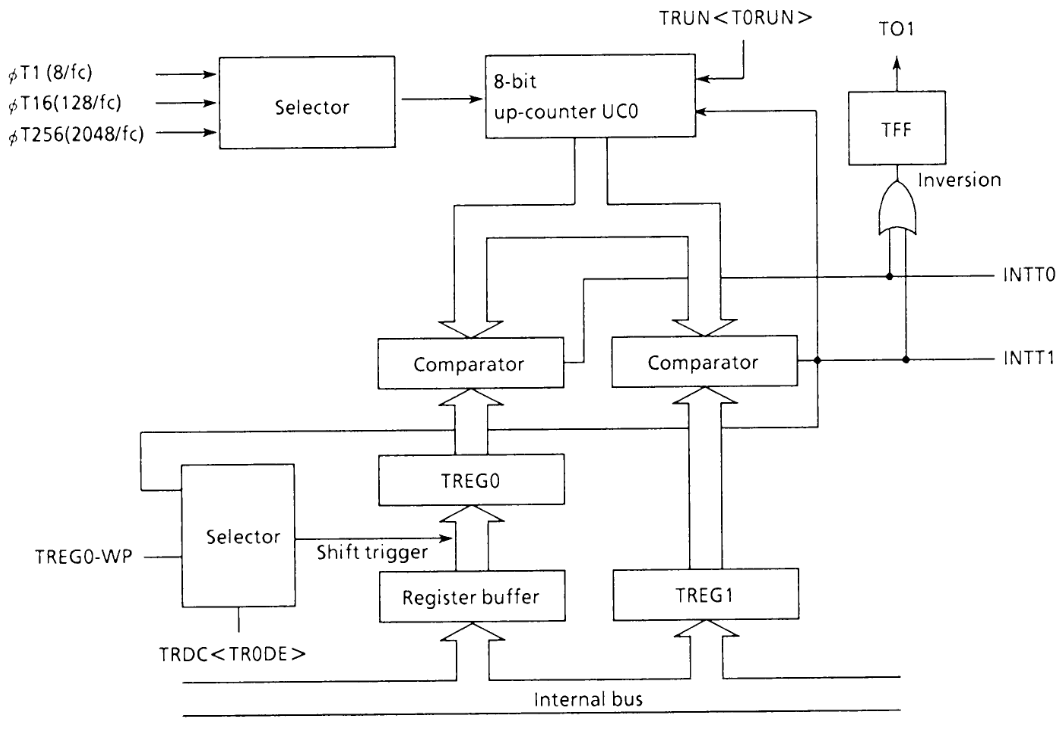
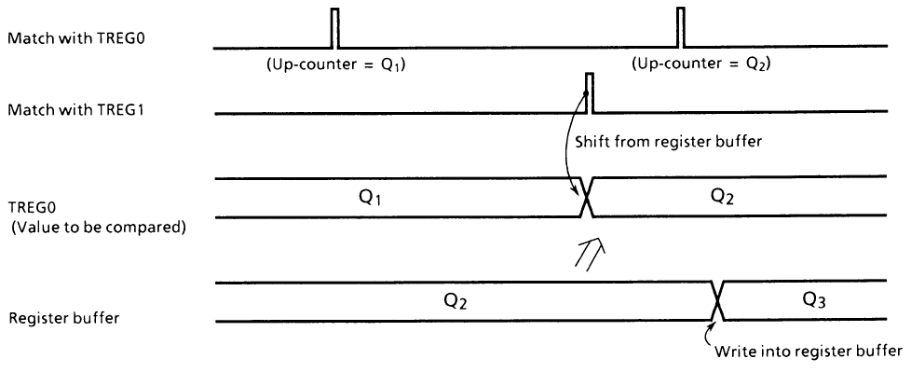


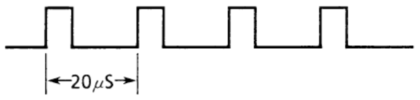
Figure 3.7 (12). Block Diagram of 8-bit PPG Mode

When the double buffer of TREG0 is enabled in this mode, the value of register buffer will be shifted in TREG0 each TREG1 matches UC0.

Use of the double buffer makes easy the handling of low duty waves (when duty is varied).



Example: Generate 1/4 duty 50KHz pulse (@fc = 8MHz
)



Calculate the value to be set for timer registers.

To obtain the frequency of 50kHz, the pulse cycle t should be: 1/50KHz = 20µs.

Given øT1 = 1.0µs (@ 8MHz),

20µs ÷ 1.0µs =20

Consequently, to set the timer register 1 (TREG1) to TREG1= 40 = 28H and then duty to 1/4, t x 1/4 = 20µs x 1/4 = 5µs

5µs ÷ 1.0µs = 5

Therefore, set timer register 0 (TREG0) to TREG0 = 5 = 5AH.

```
          MSB             LSB

        ←  7 6 5 4 3 2 1 0
TRUN    ←  X X - - - - 0 0      Stop timer 0, and clear it to "0".
T01MOD  ←  1 0 X X X X 0 1      Set the 8-bit PPG mode, and select øT1 as input clock.
TFFCR   ←  - - - - 0 1 1 x      Sets TFF1 and enable the inversion.
                   └┬┘
                    └─────────→ Writing "10" provides negative logic pulse.
TREG0   ←  0 0 0 0 0 1 0 1      Write "05H".
TREG1   ←  0 0 0 1 0 1 0 0      Write "14H".
P8CR    ←  X X - - - - - 1   }
P8FR    ←  X X - - - X - 1   }  Set P80 as the TO1 pin.
TRUN    ←  X X 1 - - - 1 1      Start timer 0 and timer 1 counting.
```

(Note) X ; Don't care   - ; No change

(4) 8-bit PWM (Pulse Width Modulation) Mode

This mode is valid only for timer 1 and timer 3. In this mode, maximum two PWMs of 8-bit resolution (PWM1 and PWM3) can be output.

PWM pulse is output to TO1 pin (also used to P80) when using timer 1, and to TO3 pin (also used as P81) when using timer 3.

Timer 0 and timer 2 can also be used as 8-bit timer.

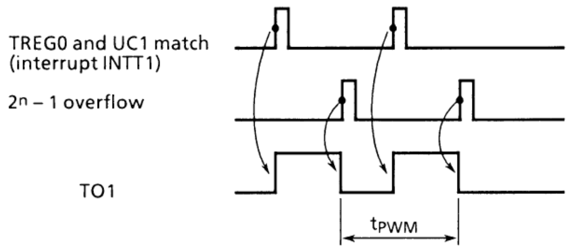
As an example, the case of timer 1 will be explained below. (Timer 3 also operates in the same way.)

Timer output is inverted when up-counter (UC0) matches the set value of timer register TREG0 or when 2n - 1 (n = 6, 7 or 8; specified by T01MOD <PWM01, 0>) counter overflow occurs. Up-counter UC1 is cleared when 2n - 1 counter overflow occurs.

To use this PWM mode, the following conditions must be satisfied.

(Set value of timer register) < (set overflow value of 2^n - 1 counter)

(Set value of timer register) ≠ 0



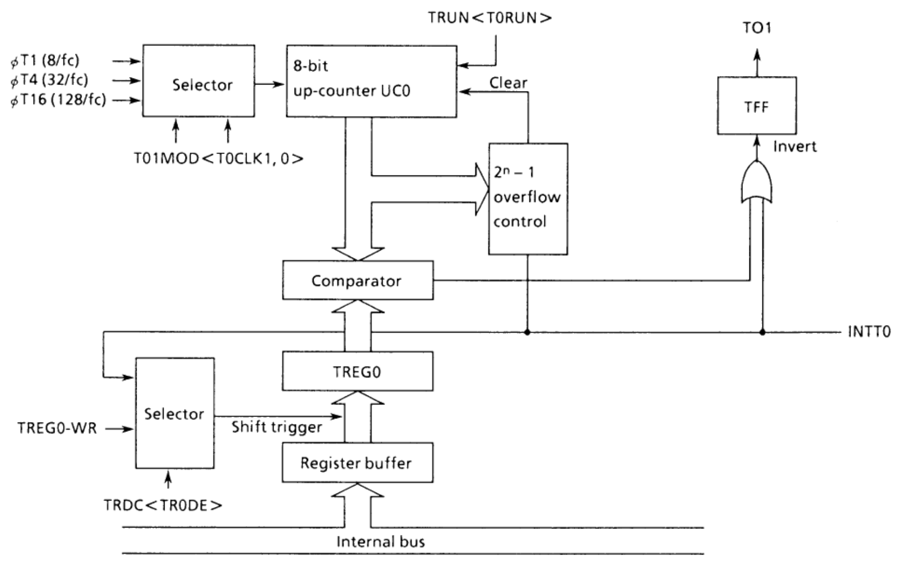
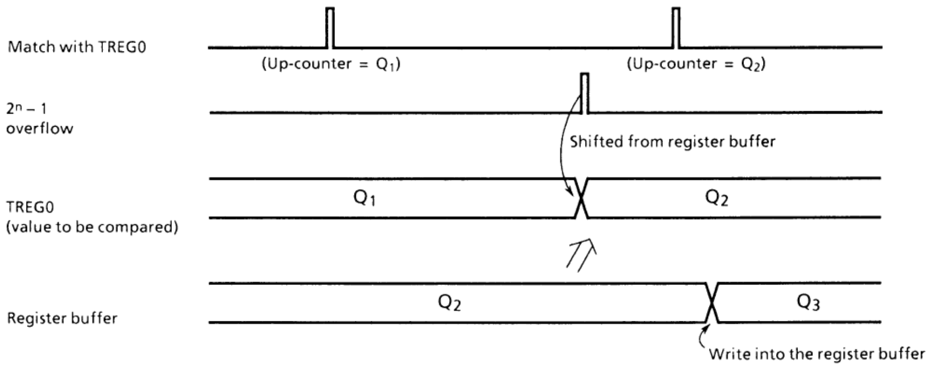


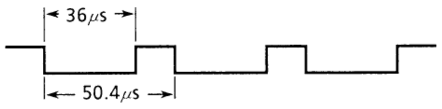
Figure 3.7 (13). Block Diagram of 8-bit PWM Mode

In this mode, the value of register buffer will be shifted in TREG0 if 2^n - 1 overflow is detected when the double buffer of TREG0 is enabled.

Use the double buffer makes easy the handling of small duty waves.



Example: To output the following PWM waves to TO1 pin using timer 0 at fc = 10MHz



To realize 50.4μs of PWM cycle by øT1 = 0.8μs (@fc = 10MHz),

50.4μs ÷ 0.8μs = 63 = 2^6 - 1

Consequently, n should be set to 6.

As the period of low level is 36μs, for T1 = 0.8μs, set the following value for TREG0.

36μs ÷ 0.8μs = 45 = 2dH

```
              MSB           LSB
               7 6 5 4 3 2 1 0
TRUN    ←      X X - - - - - 0       Stop timer 0, and clear it to "0".
T01MOD ←       1 1 0 1 - - 0 1       Set 8-bit PWM mode (cycle: 2⁶ − 1) and select ϕT1 as the
                                     input clock.

TFFCR   ←      - - - - 1 0 1 X       Clears TFF1 to enable the inversion.
TREG0   ←      0 0 1 0 1 1 0 1       Writes "2dH".
P8CR    ←      X X - - - - - 1    }  Set P80 as the TO1 pin.
P8FR    ←      X X - - - X - 1    }
TRUN    ←      X X 1 - - - - 1       Start timer 0 counting.

(Note)    X ; Don't care       - ; No change
```

Table 3.7 (3) PWM Cycle and Selection of 2^n - 1 Counter

	PWM cycle (@fc = 10MHz)		
	øT1	**øT16**	**øT256**
2^6 - 1	50.4μs	806.4μs	12.9μs
2^7 - 1	101.6μs	1625.6μs	26.0ms
2^8 - 1	204.0μs	3264.0μs	52.2ms

(5) Table 3.7 (4) shows the list of 8-bit timer modes.

Table 3.7 (4) Timer Mode Setting Register

Timer mode (8-bit timer x 2 channels)	T01M (T23M)	PWM0 (PWM2)	Upper input T1CLK (T3CLK)	Lower input T0CLK (T2CLK)	Invert select FF1IS (FF31S)
16-bit timer mode (16-bit) x 1 chanells	01	–	–	(External clock øT1, 4, 16)	–
8-bit timer (8-bit x 8-bit mode) x 1 channels (The comparator of the lower timer outputs operation clock to the upper timer.)	00	–	00	(External clock øT1, 4, 16)	0: Lower timer 1: Upper timer
8-bit x 2 channels	10	–	(øT1, 16, 256)	(External clock øT1, 4, 16)	0: Lower timer 1: Upper time
8-bit x 1 channels	11	–	–	(External clock øT1, 4, 16)	–
8-bit PWM x 1 channel (Lower) 8-bit timer x 1 channel (Upper)	11	PWM cycle	(øT1, 16, 256)	(External clock øT1, 4, 16)	– Output PWM

3.8 Multi-Function 16-bit Timer/Event Counter (Timer 4)

The TMP90C845 contains one multifunctional 16-bit timer/ event counter with the following operating modes:

- 16-bit timer
- 16-bit event counter
- 16-bit programmable pulse generation (PPG)
- Frequency measurement
- Pulse width measurement
- Time differential measurement

Figure 3.8 (1) shows the block diagram of the 16-bit timer/ event counter.

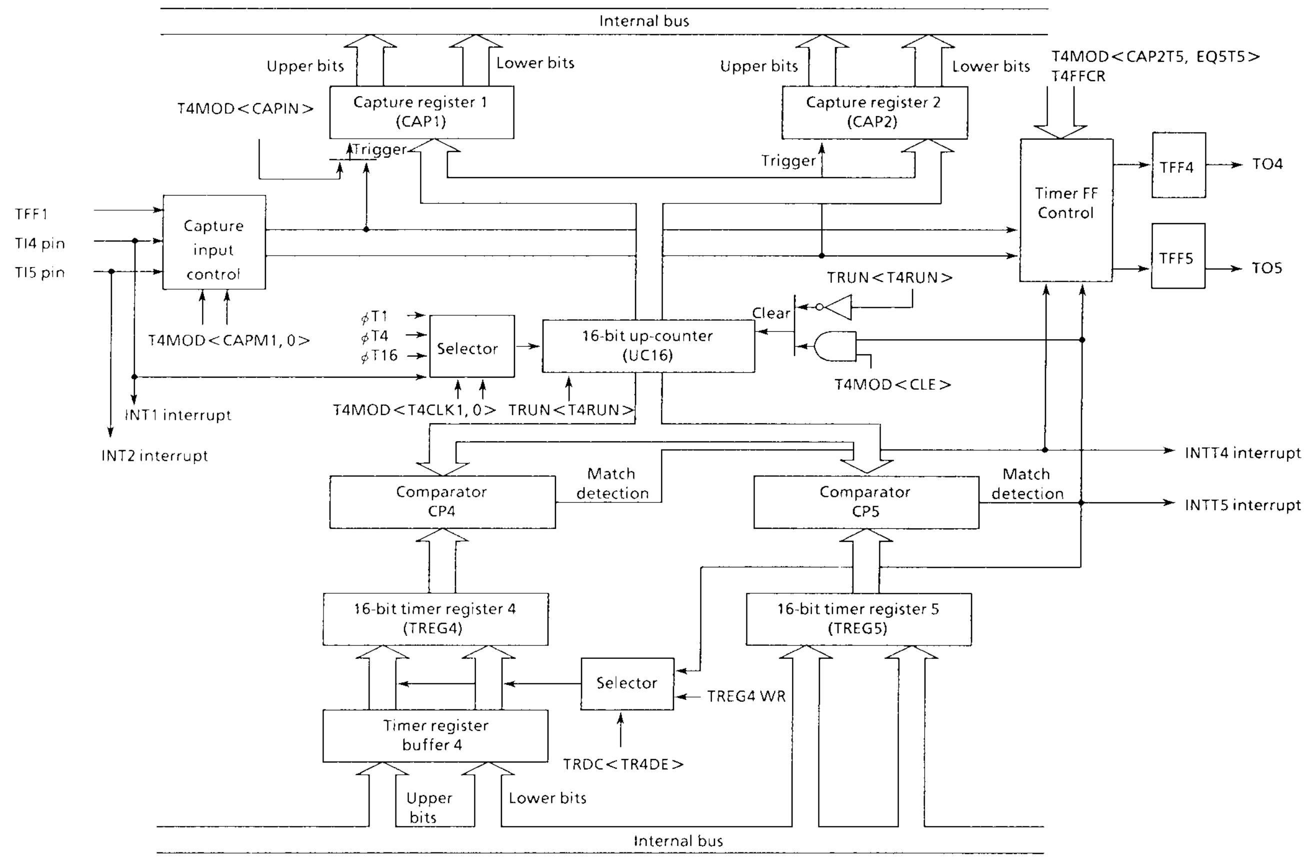


Figure 3.8 (1). Block Diagram of 16-Bit Timer/Event Counter (Timer 4)

Timer/event counter consists of 16-bit up-counter, two 16-bit timer registers, two 16-bit capture registers, two comparators, register buffer, capture input controller, timer flip-flop, and the control circuit.

Timer/event counter is controlled by 4 control registers T4MOD, T4FFCR, TRUN, and TRDC. TRUN register includes 8-bit timer controller. For TRUN and TRDC registers, see Figure 3.7 (7) and Figure 3.7 (8).

T4MOD
(FFE4H)

	7	6	5	4	3	2	1	0
bit Symbol			CAP1IN	CAPM1	CAPM0	CLE	T4CLK1	T4CLK0
Read/Write			W	R/W		R/W	R/W	
After reset			1	0	0	0	0	0
Function			0: Soft-Capture 1: don't care	Capture timing 00: Disable INT1 occurs at rise edge. 01: T14↑ T15↑ INT1 occurs at rise edge. 10: T14↑ T14↓ INT1 occurs at fall edge. 11: TFF1↑ TFF1↓ INT1 occurs at rise edge.		1: UC16 Clear Enable	Timer 4 source clock 00: TI4 01: ϕT1 (8/fc) 10: ϕT4 (32/fc) 11: ϕT16 (128/fc)	

Timer 4 input clock

00	External clock (TI4)
01	Internal clock ϕT1
10	Internal clock ϕT4
11	Internal clock ϕT16

Clearing the up-counter UC16

0	Clear disable
1	Clear by match with TREG5.

Capture control/INT1 interrupt control

	Capture control	INT1 control
00	Capture disable	Interrupt occurs at the rise edge of TI4 (INT1) input.
01	CAP1 at TI4 rise CAP2 at TI5 rise	
10	CAP1 at TI4 rise CAP2 at TI4 fall	Interrupt occurs at the fall edge of TI4 (INT1) input.
11	CAP1 at TFF1 rise CAP2 at TFF1 fall	Interrupt occurs at the rise edge of TI4 (INT1) input.

Software capture trigger

0	The up-counter value is loaded to CAP1 (software capture).
1	Always read as "1".

Figure 3.8 (2). 16-bit Timer/Event Counter (Timer 4) Controller/Mode Registers (1/2)

T4FFCR
(FFE5H)

	7	6	5	4	3	2	1	0
bit Symbol			CAP2T4	CAP1T4	EQ5T4	EQ4T4	TFF4C1	TFF4C0
Read/Write			R/W				W	
After reset			0	0	0	0	—	—
Function			TFF4 invert trigger 0: Disable trigger 1: Enable trigger When the up-counter value is loaded to CAP2	When the up-counter value is loaded to CAP1	When the up-counter matches TREG5	When up-counter matches TREG4	00: Invert TFF4 01: Set TFF4 10: Clear TFF4 11: Don't care ※ Always read as "11"	

→ Timer flip-flop 4 (TFF4) control

00	Inverts the TFF4 value (software inversion).
01	Sets TFF4 to "1".
10	Clear TFF4 to "0".
11	Don't care (Always read as "11") .

→ Timer flip-flop 4 (TFF4) invert trigger

0	Trigger disable
1	Trigger enable

CAP2T4	;	When the up-counter value is loaded to CAP2
CAP1T4	;	When the up-counter value is loaded to CAP1
EQ5T4	;	When up-counter matches TREG5
EQ4T4	;	When up-counter matches TREG4

Figure 3.8 (3). 16-Bit Timer/Event Counter Timer Flip-flop Control Register

① Up-counter (UC16)

UC16 is a 16-bit binary counter which counts up according to the input clock specified by T4MOD <T4CLK1, 0> register.

As the input clock, one of the internal clocks øT1 (8fc), øT4 (32fc), and øTI6 (128fc) from 9-bit prescaler (also used as 8-bit timer), and external clock from TI4 pin (commonly used as P46/INT1 pin) can be selected. When reset, it will be initialized to <T4CLK1, 0> = 00 to select TI4 input mode. Counting, stop, or clearing of the counter is controlled by timer operation control register TRUN <T4RUN>.

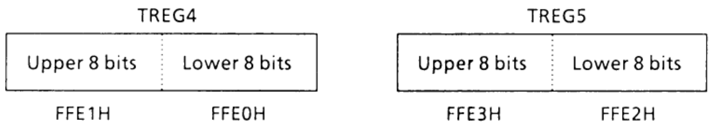
Up-counter UC16 will be cleared to zero each time it coincides or matches the timer register TREG5. The "clear enable/disable" is set by T4MOD <CLE>.

If clearing is disabled, the counter operates as a free-running counter.

② Timer Registers (TREG4 and TREG5)

These two 16-bit registers are used to set the value of counter. When the value of up-counter UC16 matches the set value of this timer register, the comparator match detect signal will be active.

Setting data for timer register (TREG4 and TREG5) is executed using 1-byte date road instruction twice for lower 8 bits and upper 8 bits in order.



TREG4 timer register is of double buffer structure, which is paired with register buffer. TREG4 controls whether the double buffer should be enabled or disabled, using the timer register double buffer control register TRDC <TR4DE>: disable when <TR4DE> = 0, while enable when <TR4DE> = 1.

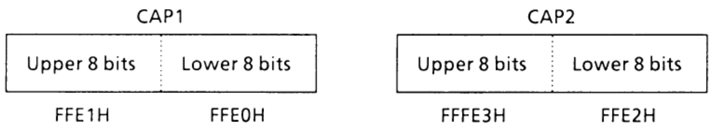
When the double buffer is enabled, the timing to transfer data from the register buffer to the timer register is at the match between the up-counter and TREG5.

When reset, it will be initialized to <TR4DE> = 0, whereby the double buffer is disabled. To use the double buffer, write data in the timer register to set to <TR4DE> = 1 then write the following data in the register buffer.

TREG4 and register buffer 4 are allocated to the same memory addresses FFE0H and FFE1H. When <TR4DE> = 0, same value will be written in both the TREG4 and register buffer 4. When <TR4DE> = 1, the value is written into only the register buffer 4.

③ Capture Register (CAP1 and CAP2)

These 16-bit registers are used to hold the values of the up-counter UC16. Data in the capture registers should be read by a 2-byte load instruction or two 1-byte data load instruction, from the lower 8 bits followed by the upper 8 bits.



④ Capture Input Control Circuit

This circuit controls the timing to latch the value of up-counter UC16 into CAP1 and CAP2.
The latch timing of capture register is controlled by register T4MOD <CAPM1, 0>.

• When T4MOD <CAPM1, 0> = 0 0
Capture function is disabled. Disable is the default on reset.

• When T4MOD <CAPM1, 0> = 01
Data is loaded to CAP1 at the rise edge of TI4 pin (commonly used as P83/INT1) input, while data is loaded to CAP2 at the TI5 pin (also used as P25/INT2) rising edge (Time difference measurement)

• When T4MOD <CAPM1, 0> = 10
Data is loaded to CAP1 at the rise edge of the TI4 pin input while data is loaded to CAP2 at the fall edge. Only in this setting, interrupt INT1 occurs at the falling edge. (Pulse width measurement)

• When T4MOD <CAP1, 0> = 11
Data is loaded to CAP1 at the rise edge of timer flip-flop TFF1, while to CAP2 at the fall edge. (Frequency measurement)

Besides, the value of up-counter can be loaded to capture registers by software. Whenever “0” is written in T4MOD <CAPIN>, the current value of up-counter will be loaded to capture register CAP1. It is necessary to keep the prescaler in RUN mode (TRUN <PRUN>to be “1”).

⑤ Comparator (CP4 and CP5)

These are 16-bit comparators which compare the up-counter UC16 value with the set value of TREG4 or TREG5 to detect the match. When a match is detected, the comparators generate an interrupt INTT4 and INTT5, respectively. The up-counter UC16 is cleared only when UC16 matches TREG5. (The clearing of up-counter UC16 can be disabled by setting T4MOD <CLE> = 0).

⑥ Timer Flip-flop (TFF4)

This flip-flop is inverted by the match detect signal from the comparators (CP4 and CP5) and the latch signal to the capture registers (CAP1 and CAP2).
Disable/enable the inversion can be set for each element by T4FFCR <CAP2T4, CAP1T4, EQ5T4, EQ4T4>. TFF4 will be inverted when “00” is written in T4FFCR < TFF4C1, 0>. Also it is set to “1” when “10” is written, and cleared to “0” when “10” is written. The value of TFF4 can be output to the timer output pin TO4 (commonly used as P85).

(1) 16-bit Timer Mode

In this example, the interval time is set in the timer register TREG5 to generate the interrupt INTT5.

```
┌TRUN   ←  X X - 0 - - - -          Stop timer 4.
│INTEL  ←  0 - 1 - - - - -          Enable INTT5 and disable INTT4.
│
│T4FFCR←  X X 0 0 0 0 1 1          Disable trigger.
│T4MOD  ←  X X 1 0 0 1 * *          Select internal clock for input and
│                 (**=01,10,11)      disable the capture function.
│TREG5  ←  **** **** **** ****      Set the interval time (16 bits).
└TRUN   ←  X X 1 1 - - - -          Start timer 4.
```

(Note) X ; Don’t care - ; No change

(2) 16-bit Event Counter Mode

In timer mode as described in above (1), the timer can be used as an event counter by selecting the external clock (TI4 pin input) as the input clock. To read the value of the counter, first perform “software capture” once and read the captured value.

The counter counts at the rise edge of TI4 pin input.

TI4 pin can also be used as P83/INT1.

```
TRUN   ←  X X - 0 - - - -          Stop timer 4.
P8FR   ←  X X - - * X - -          * = 0 :TI4 input pulse is square wave
                                   * = 1 : TI4 input pulse is sine wave (zero-cross)

INTEL  ←  0 0 1 - - - - -          Enable INTT5, while disables INTT4 and INT1.
T4FFCR←   X X 0 0 0 0 1 1          Disable trigger.
T4MOD  ←  X X 1 0 0 1 0 0          Select TI4 as the input clock.
TREG5  ←  **** **** **** ****      Set the number of counts (16 bits).
TURN   ←  - - 1 1 - - - -          Start timer 4.
```

(Note) When used as an event counter, set the prescaler in RUN mode.

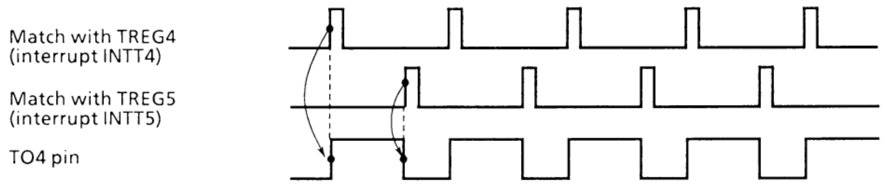
(3) 16-bit Programmable Pulse Generation (PPG) Mode

The PPG mode is entered by inversion of the timer flip-flop TFF4 that is to be enabled by match of the up-counter UC16 with the timer register TREG 4 or 5 and to be output to TO4 (also used as P85). In this mode, the following conditions must be satisfied.

(Set value of TREG4) < (Set value of TREG5)

```
TRUN   ←  X X - 0 - - - -          Stop timer 4.
TREG4  ←  **** **** **** ****      Set the duty.
TREG5  ←  **** **** **** ****      Set the cycle.
T4FFCR←   X X 0 0 1 1 0 0          Set the TFF4 inversion to be effected by match with TREG4 or
                                   TREG5. Initialize TFF4 to "0".
T4MOD  ←  X X 1 0 0 1 * *          Select the internal clock for the input, and disable the capture
                    (**=01,10,11)  function.
P8CR   ←  X X 1 - - - - -        ⎫
                    (**=00,01,10)⎬ Assign P85 as TO4.
P8FR   ←  X X 1 - - X - -        ⎭
TRUN   ←  X X 1 1 - - - -          Start timer 4.
```

(Note) X ; Don’t care - ; No change



When the double buffer of TREG4 is enabled in this mode, the value of register buffer 4 will be shifted in TREG4 at match with TREG5. This feature makes easy the handling of low duty waves (when duty rate is varied).

(4) Application examples of capture function

The loading of up-counter (UC16) values into the capture registers CAP1 and CAP2, the timer flip-flop TFF4 inversion due to the match detection by comparators CP4 and CP5, and the output of the TFF4 status to TO4 pin can be enabled or disabled. Combined with interrupt function, they can be applied in many ways, for example:

① One-shot pulse output from external trigger pulse
② Frequency measurement
③ Pulse width measurement
④ Time difference measurement

① One-shot pulse output from external trigger pulse

Set the up-counter UC16 in free-running mode with the internal input clock, input the external trigger pulse from TI4 pin, and load the value of up-counter into the capture register CAP1 at the rising edge of TI4 pin. Then set to T4MOD <CAPM1, 0> = 01.

When the interrupt INT1 is generated at the rising edge of TI4 input, set the CAP1 value (c) plus a delay time (d) to TREG4 (= c + d). and set the above set value (c + d) pulse a one shot pulse width (p) the TREG5 (= c + d + p). When the interrupt INT1 occurs the T4FFCR (BIT 2 ~ 5) register should be set that the TFF4 inversion is enabled only when the up-counter value matches TREG4 or 5. When interrupt INTT5 occurs, this inversion will be diabled.

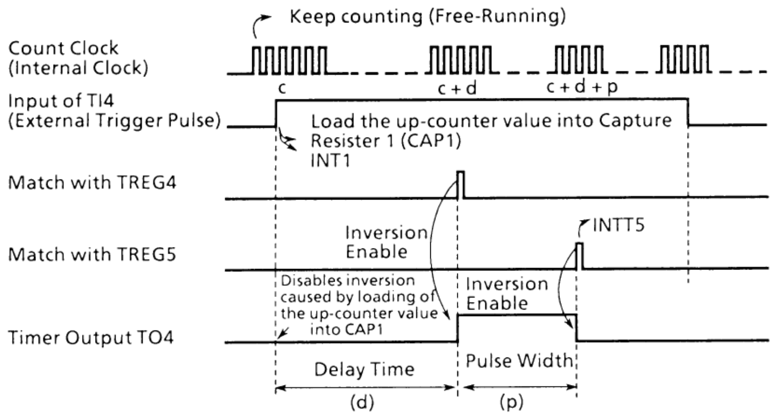

Keep counting (Free-Running)
Count Clock (Internal Clock)
c
c + d
c + d + p
Input of TI4 (External Trigger Pulse)
Load the up-counter value into Capture Resister 1 (CAP1)
INT1
Match with TREG4
Inversion Enable
INTT5
Match with TREG5
Disables inversion caused by loading of the up-counter value into CAP1
Inversion Enable
Timer Output TO4
Delay Time
(d)
Pulse Width
(p)

Figure 3.8 (4). One-Shot Output (with Delay)

Setting example: To output 2ms one-shot pulse with a 3ms delay to the external trigger pulse to TI4 pin.

Main setting

```
T4MOD  ←  X X 1 0 1 0 0 1         Keep counting (Free-runnig)
                                   Count with φT1.
                                   Load the up-counter value into CAP1 at the rise  edge
T4FFCR←   X X 0 0 0 0 0 0         of T14 pin input.
                                   Clear TFF4 to zero.
                                   Disable TFF4 inversion.
P8CR   ←  X X 1 - 0 - - -
                (**=00,01,10)      Select P85 as the TO4 pin.
P8FR   ←  X X 1 - - X - -
INTEL  ←  0 1 0 - - - - -          Enable INT1, and disable INTT4 and INTT5.

TRUN   ←  X X 1 1 - - - -          Start timer 4.
```

Setting of INT1

```
TREG4  ←  CAP1+3ms/φT1
TREG5  ←  TREG4+2ms/φT1
T4FFCR←   X X - - 1 1 - -
                                   Enable TFF4 inversion when the up-counter value
                                   matches TREG4 or 5.
INTEL  ←  - - 1 - - - - -          Enable INTT5.
```

Setting of INT5

```
T4FFCR←   X X - - 0 0 - -
                                   Disable TFF4 inversion when the up-counter value
                                   matches TREG4 or 5.
INTEL  ←  - - 0 - - - - -          Disable INTT5.
```

(Note) X ; Don't care - ; No change

When delay time is unnecessary, invert timer flip-flop TFF4 when the up-counter value is loaded into capture register 1 (CAP1), and set the CAP1 value (c) plus the one-shot pulse width (p) to TREG5 when the interrupt INT1 occurs. The TFF4 inversion should be enabled before the up-counter (UC16) value matches TREG5, and disabled when generating the interrupt INTT5.

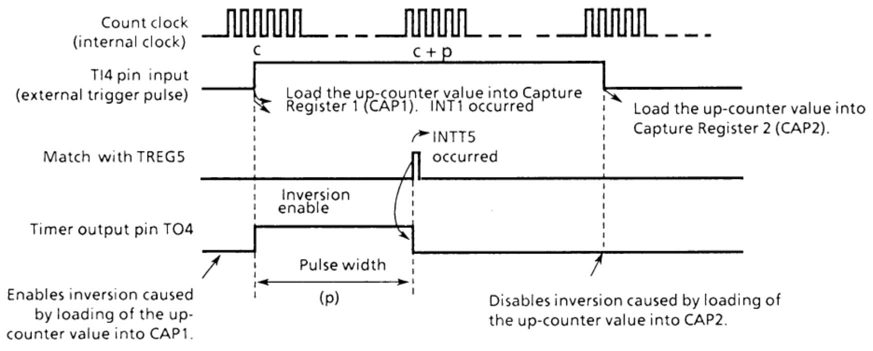

Count clock
(internal clock)
c
c + p
TI4 pin input
(external trigger pulse)
Load the up-counter value into Capture Register 1 (CAP1). INT1 occurred
Load the up-counter value into Capture Register 2 (CAP2).
INTT5 occurred
Match with TREG5
Inversion enable
Timer output pin TO4
Pulse width
(p)
Enables inversion caused by loading of the up-counter value into CAP1.
Disables inversion caused by loading of the up-counter value into CAP2.

Figure 3.8 (5). One-Shot Pulse Output (without Delay)

② Frequency measurement

The frequency of the external clock can be measured in this mode. The clock is input through the TI4 pin, and its frequency is measured by using the 8-bit timers (Timer 0 and Timer 1) and the 16-bit timer/event counter (Timer4).

The TI4 pin input should be selected for the input clock of Timer 4. The value of the up-counter is loaded into the capture register CAP1 at the rise edge of the timer flip-flop TFF1 of 8-bit timers (Timer 0 and Timer 1), and into CAP2 at its fall edge.

The frequency is calculated by the difference between the loaded values CAP1 and CAP2 when the interrupt (INTT0 or INTT1) is generated by either 8-bit timer.

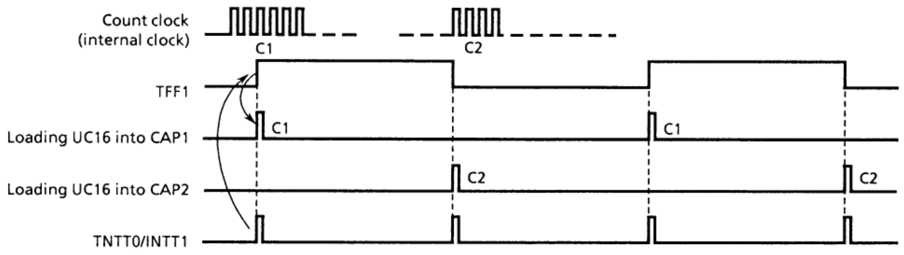


Figure 3.8 (6). Frequency Measurement

For example, if the value for the level “1” width of TFF1 of the 8-bit timer is set to 0.5 sec. and the difference between CAP1 and CAP2 is 100, the frequency will be 100/0.5 [s] - 200 [Hz].

③ Pulse width measurement

This mode allows to measure the “H” level width of an external pulse. While keeping the 16-bit timer/event counter counting (free-running) with the internal clock input, the external pulse is input through the TI4 pin.

Then the capture function is used to load the UC16 values into CAP1 and CAP2 at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT1 occurs at the falling edge of TI4.

The pulse width is obtained from the difference between the values of CAP1 and CAP2 and the internal clock cycle.

For example, if the internal clock is 8.0 microseconds and the difference between CAP1 and CAP2 is 100, the pulse width will be 100 x 0.8 = 80 microseconds.

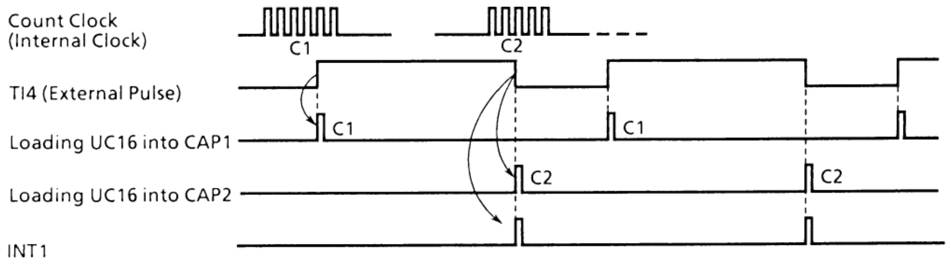


Figure 3.8 (7) Pulse Width Measurement

Note: Only in this pulse width measuring mode (T4MOD <CAPM1, 0> = 10), external interrupt INT1 occurs at the falling edge of TI4 pin input. In other modes, it occurs at the rising edge.

The width of “L” level can be measured from the difference between the first C2 and the second C1 at the second INT1 interrupt.

④ Time difference measurement

This mode is used to measure the difference in time between the rising edges of external pulses input through TI4 and TI5.

Keep the 16-bit timer/event counter (Timer 4) counting (free-running) with the internal clock, and load the UC16 value into CAP1 at the rising edge of the input pulse to TI4. Then the interrupt INT1 is generated.

Similarly, the UC16 value is loaded into CAP2 at the rising edge of the input pulse to TI5, generating the interrupt INT2.

The time difference between these pulses can be obtained from the difference between the time counts at which loading the up-counter value into CAP1 and CAP2 has been done.

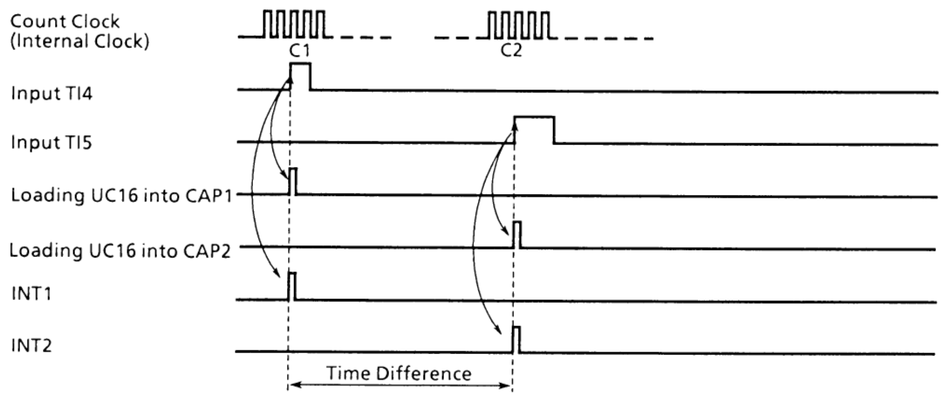


Figure 3.8 (8). Time Difference Measurement

3.9 Serial Channel

The TMP90C848F contains a serial I/O channel for full duplex asynchronous transmission (UART) as well as for I/O extension.

The serial channel has the following operating modes:

- Asynchronous transmission (UART) mode
  - Mode 1: 7-bit data
  - Mode 2: 8-bit data
  - Mode 3: 9-bit data

The mode 1 and mode 2, a parity bit can be added. Mode 3 has wake-up function for making the master controller start slave controllers serial link (multi-controller system).

Figure 3.9 (1) shows the data format (for one frame) in each mode.

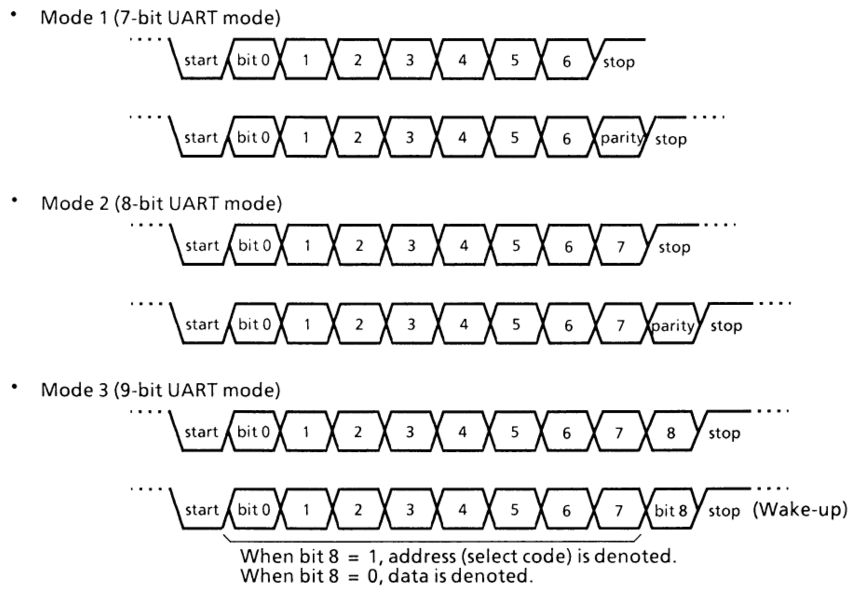

• Mode 1 (7-bit UART mode)
start
bit 0
1
2
3
4
5
6
stop
start
bit 0
1
2
3
4
5
6
parity
stop
• Mode 2 (8-bit UART mode)
start
bit 0
1
2
3
4
5
6
7
stop
start
bit 0
1
2
3
4
5
6
7
parity
stop
• Mode 3 (9-bit UART mode)
start
bit 0
1
2
3
4
5
6
7
8
stop
start
bit 0
1
2
3
4
5
6
7
bit 8
stop
(Wake-up)
When bit 8 = 1, address (select code) is denoted.
When bit 8 = 0, data is denoted.

Figure 3.9 (1). Data Formats

The serial channel has a buffer register for transmitting and receiving operations, in order to temporarily store transmitted or received data, so that transmitting and receiving operations can be done independently (full duplex).

The receiving buffer register is of a double buffer structure to prevent the occurrence of overrun error and provides one frame of margin before CPU reads the received data. Namely, the one buffer stores the already received data while the other buffer receives the next frame data.

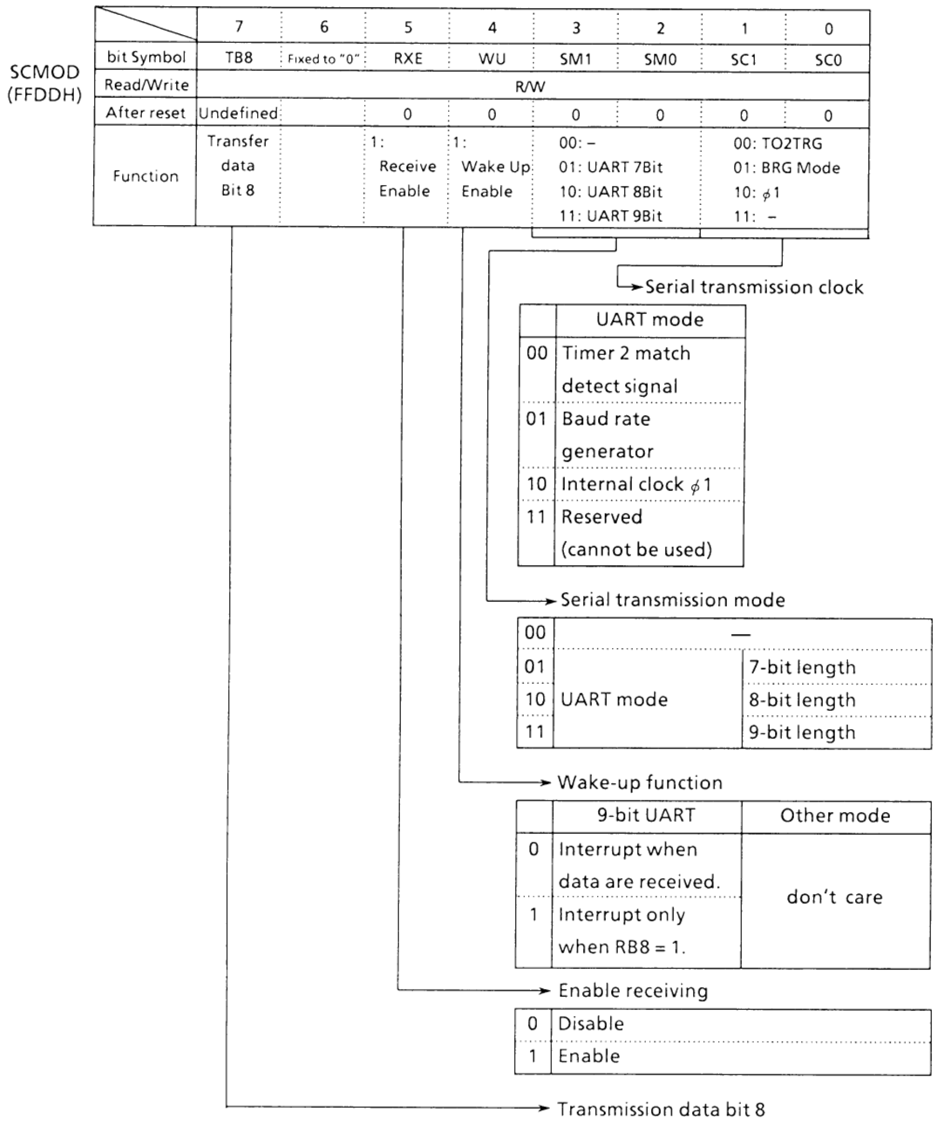
In the UART mode, a check function is added not to start the receiving operation by error start bits due to noise. The channel starts receiving data only when the start bit is detected to be normal at least twice in three samplings.

When the transmission buffer becomes empty and requests the CPU to send the next transmission data or when data is stored in the transmission buffer and the CPU is requested to read the data, INTTX or INTRX interrupt occurs. Besides, if an overrun error, parity error, or framing error occurs during receiving operation, flag SCCR <OERR, PERR, FERR> will be set.

The serial channel includes a special baud rate generator, which can set any baud rate by dividing the frequency of 4 clocks (øT0, øT2, øT8, and øT32) from the internal prescaler (shared by 8-bit/16-bit timer) by the value 2 to 16.

3.9.1 Control Registers

The serial channel is controlled by 4 control registers SCMOD, SCCR, TRUN, BRGCR, and P3FR. Transmitted and received data are stored into SCBUF.



SCMOD (FFDDH)

	7	6	5	4	3	2	1	0
bit Symbol	TB8	Fixed to "0"	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	Undefined		0	0	0	0	0	0
Function	Transfer data Bit 8		1: Receive Enable	1: Wake Up Enable	00: – 01: UART 7Bit 10: UART 8Bit 11: UART 9Bit		00: TO2TRG 01: BRG Mode 10: $\phi 1$ 11: –	

Serial transmission clock

	UART mode
00	Timer 2 match detect signal
01	Baud rate generator
10	Internal clock $\phi 1$
11	Reserved (cannot be used)

Serial transmission mode

00	—	
01	UART mode	7-bit length
10		8-bit length
11		9-bit length

Wake-up function

	9-bit UART	Other mode
0	Interrupt when data are received.	don't care
1	Interrupt only when RB8 = 1.	

Enable receiving

0	Disable
1	Enable

Transmission data bit 8

Figure 3.9 (2). Serial Channel Mode Register (SCMOD)

SCCR (FFDEH)

	7	6	5	4	3	2	1	0
bit Symbol	RB8	EVEN	PE	OERR	PERR	FERR		
Read/Write	R	R/W		R (Cleared to zero when read)				
After reset	Undefined	0	0	0	0	0		
Function	Received data Bit 8	Parity 0: Odd 1: EVEN	1: Parity Enable	1: Error Overrun	Parity	Framing		

Framing error flag
Parity error flag
Overrun error flag
} Cleared to zero when read.

Enable parity addition

0	Disable
1	Enable

Addition/check of even parity

0	Odd parity
1	Even parity

Receiving data bit 8

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

Figure 3.9 (3). Serial Channel Control Register (SCCR)

SCBUF (FFDFH)

	7	6	5	4	3	2	1	0
bit Symbol	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0
Read/Write	W							
After reset	Undefined							

	7	6	5	4	3	2	1	0
bit Symbol	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
Read/Write	R							
After reset	Undefined							

Figure 3.9 (4). Serial Transmission/Receiving Buffer Register (SCBUF)

BRGCR
(FFEDH)

	7	6	5	4	3	2	1	0
bit Symbol	Fixed to "0"		BG1	BG0	PS3	PS2	PS1	PS0
Read/Write			R/W					
After reset	0		0	0	0	0	0	0
Function			00: fc/4 01: fc/16 10: fc/64 11: fc/256		Divided frequency of prescaler			

Setting of the divided frequency of baud rate generator

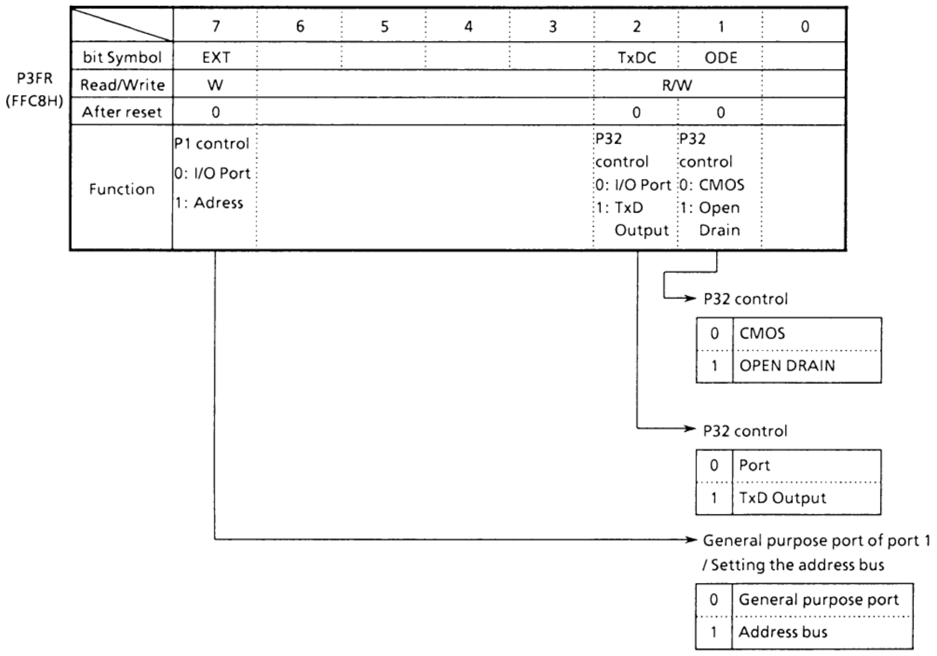
0000	16 divisions
0010 ~ 0000	2 to 16 divisions
"0001" cannot be set.	

Selecting the input clock of baud rate generator

00	Internal clock ϕT0 (fc/4)
01	Internal clock ϕT2 (fc/16)
10	Internal clock ϕT8 (fc/64)
11	Internal clock ϕT32 (fc/256)

Note: To use the baud rate generator, set TRUN<PRRUN> to "1", putting the prescaler in RUN mode.

Figure 3.9 (5). Baud Rate Generator Control Register (BRGCR)



P3FR (FFC8H)		7	6	5	4	3	2	1	0
	bit Symbol	EXT					TxDC	ODE	
	Read/Write	W					R/W	R/W	
	After reset	0					0	0	
	Function	P1 control 0: I/O Port 1: Adress					P32 control 0: I/O Port 1: TxD Output	P32 control 0: CMOS 1: Open Drain	

P32 control

0	CMOS
1	OPEN DRAIN

P32 control

0	Port
1	TxD Output

General purpose port of port 1 / Setting the address bus

0	General purpose port
1	Address bus

Figure 3.9 (6). Port 3 Function Registers (P3FR)

3.9.2 Configuration

Figure 3.9 (7) is a block diagram of the serial channel.

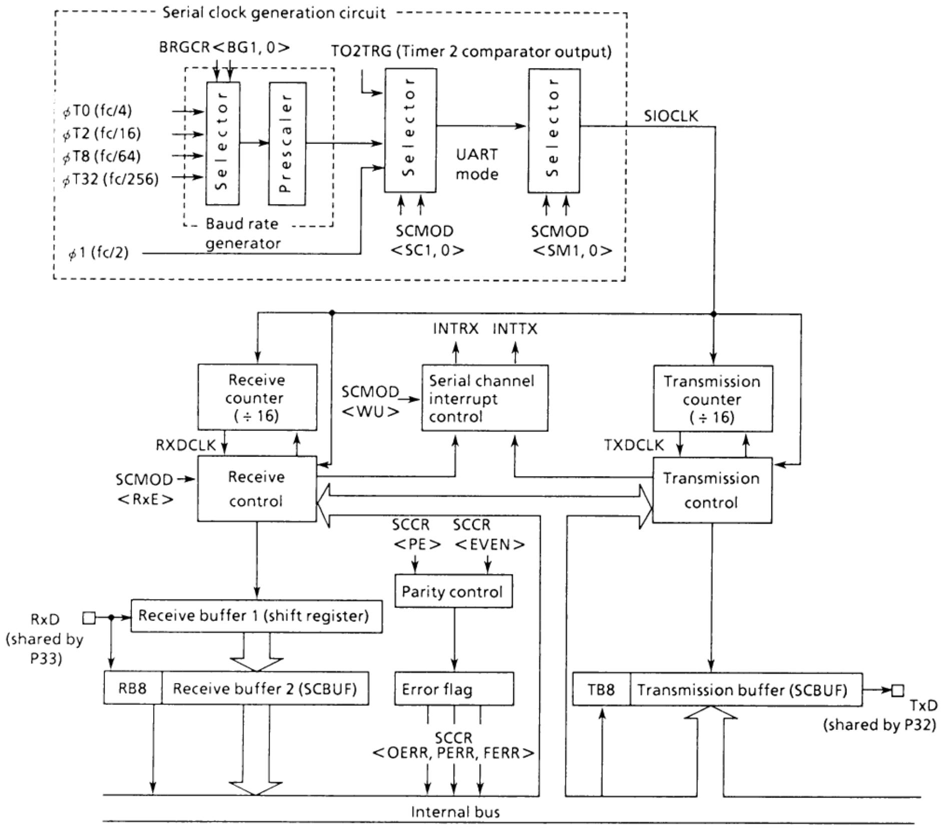


Figure 3.9 (7). Block Diagram of the Serial Channel

① Baud-rate generator

Baud-rate generator comprises a circuit that generates transmission and receiving to determine the transfer rate of the serial channel.

The input clock to the baud-rate generator øT0(fc/4), øT2 (fc/16), øT8 (fc/64), or øT32 (fc/256) is generated by the 9-bit prescaler which is shared by the timers. One of these input clocks is selected by the baud rate generator control register BRGCR <BG1, 0>.

The baud rate generator includes a 4-bit frequency divider, which divides frequency by 2 to 16 values to determine the transfer rate.

How to calculate a transfer rate when the baud rate generator is used is explained below.

$$\text{Transfer rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divisor of baud rate generator}} \div 16$$

The relation between the input clock and the source clock (fc) is as follows.

øT0 = fc/4
øT2 = fc/16
øT8 = fc/64
øT32 = fc/256

Accordingly, when source clock fc is 9.8304MHz, input clock is øT2 (fc/16), and frequency divisor is 4, the transfer rate in UART mode becomes as follows:

$$\text{Transfer rate} = \frac{\text{fc}/16}{5} \div 16$$

= 9.8304 x 10^6/16/4/16 = 9600 (bps)

Table 3.9 (1) shows shows an example of the transfer rate UART mode.

Also with 8-bit timer 2, the serial channel can get a transfer rate. Table 3.9 (2) shows an example of baud rate using timer 2.

Table 3.9 (1) Selection of Baud Rate (1) (When Baud Rate Generator is Used) (Units [Kbps}

fc	Input clock / Frequency divisor	øT0 (fc/4)	øT2 (fc/16)	øT8 (fc/64)	øT32 (fc/256)
9.8304MHz	2	76.800	19.200	4.800	1.200
–	4	38.400	9.600	2.400	0.600
–	8	19.200	4.800	1.200	0.300
–	0	9.600	2.400	0.600	0.150

Table 3.9 (2) Selection of Transfer Rate (2) (When Timer 2 (Input Clock T1) is Used)

fc / TREG2	9.8304 MHz	8 MHz	6.144 MHz
1H	76.8	62.5	48
2H	38.4	31.25	24
3H	–	–	16
4H	19.2	–	12
5H	–	–	9.6
8H	9.6	–	6
AH	–	–	4.8
10H	4.8	–	3
14H	–	–	2.4

How to calculate the baud rate (when timer 2 is used):

$$\text{Baud rate} = \frac{1}{\text{TREG2}} \times \frac{1}{16} \times (\text{Input clock of timer 2})$$

Input CLK of timer 2
øT1 = fc/8
øT4 = fc/32
øT16 = fc/128

② Serial clock generation circuit

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

According to the setting of SCMOD <SC1, 0>, the above baud rate generator clock, internal clock ø1(fc/2) (312.5Kbaud at 10MHz), or the match detect signal from timer 2 will be selected to generate the basic clock SIOCK.

③ Receiving counter

The receiving counter is a 4-bit binary counter used in asynchronous communication (UART) mode and counts up by SIOCLK clock. 16 pulses of SIOCLK are used for receiving 1 bit of data, and the data bit is sampled three times at the 7th, 8th and 9th clock.

With the three samples, the received data is evaluated by the rule of majority.

For example, if the sample data bits is “1”, “0” and “1” at 7th, 8th and 9th clock, respectively, the received data is evaluated as “1”. The sampled data “0”, “0” and “1” is evaluated the received data is “0”.

④ Receiving control

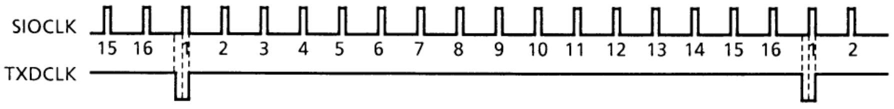
The receiving control has a circuit for detecting the start bit by the rule of majority. When two or more "0" are detected during three samples, it is recognized as normal start bit and the receiving operation is started.
Data being received are also evaluated by the rule of majority.

⑤ Receiving buffer

To prevent overrun from occurring, the receiving buffer has a double structure. Received data are stored one bit by on bit in the receiving buffer 1 (shift register type). When 7 or 8 bits of data is stored in the receiving buffer 1, the stored data are transferred to another receiving buffer 2 (SCBUF), generating an interrupt INTRX. The CPU reads receive buffer 2 (SCBUF). Even before the CPU reads receive buffer 2 (SCBUF), the received data can be stored in the receiving buffer 1. However, unless the receiving buffer 2 (SCBUF) is read before all bits of the next data received by the receiving buffer 1, an overrun error occurs. If an overrun error occurs, the contents of the receiving buffer 1 will be lost, although the contents of the receiving buffer 2 and SCCR <RB8> is still preserved.
When 9-bit UART, wake-up function of the slave controllers is enabled by setting SCMOD <WU> to "1" and interrupt INTRX occurs only SCCR <RB8> is set to "1".
The parity bit added in 8-bit UART mode and the mosy significant bit (MSB) in 9-bit UART mode are stored in SCCR <RB8>.

⑥ Transmission counter

Transmission counter is a 4-bit binary coutner which is used in asynchronous communication (UART) mode and, like the receiving counter, counts by SIOCLK clock, generating TXDCLK is generated every 16 clock pulses.



⑦ Transmission controller

When transmission data are written in the transmission buffer sent from the CPU, transmission starts at the rising edge of the next TxDCLK, generating a transmission shift clock TxDSFT.

⑧ Transmission buffer

Transmission buffer SCBUF shifts out and sends the transmission data written by the CPU from the least signicant bit (LSB) in order, using transmission shift clock TxDSFT which is generated by the transmission control. When all bits are shifted out, the transmission buffer becomes empty and generates INTTX interrupt.

⑨ Parity control circuit

When serial channel control register SCCR <PE> is set to "1", it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART or 8-bit UART mode. When SCCR <EVEN> register, even (odd) parity can be selected.
For transmission mode, parity is automatically generated according to the data written into the transmission buffer SCBUF, and data are transmitted after being stored in SCBUF <TB7>when in 7-bit UART mode while in SCMOD <TB8> when in 8-bit UART mode. <PE> and <EVEN> must be set before transmission data are written in the transmission buffer.
For receiving, data are shifted in the receiving buffer 1 and parity is added after the data are transferred in the receiving buffer 2 (SCBUF), and then compared with <RB7> of SCBUF when in 7-bit UART mode and with SCCR <RB8> when in 8-bit UART mode. If they are not equal, a parity error occurs, and SCCR <PERR> flag is set.

⑩ Error flag

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

1. Overrun error (SCCR <OERR>)

If all bits of the next data are received in receiving buffer 1 while valid data are still stored in receiving buffer 2 (SCBUF), an overrun error will occur.

2. Parity error (SCCR <PERR>)

The parity generated for the data shifted in receiving buffer 2 (SCBUF) is compared with the parity bit received from the RxD pin. If they are not equal, a parity error occurs.

3) Framing error (SCCR <FERR>)

The stop bit of received data is sampled three times around the center. If the majority is "0", a framing error occurs.

⑪ Generating Timing

Receiving

Mode	9 Bit	8 Bit + Parity	8 bit, 7 Bit + Parity, 7 Bit
Interrupt timing	Center of last bit	Center of last bit (Parity Bit)	Center of stop bit
Framing error timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error timing	Center of last bit (Bit 8)	Center of last bit (Parity Bit)	Center of stop bit
Over-run error timing	Center of last bit (Bit 8)	Center of last bit (Parity Bit)	Center of stop bit

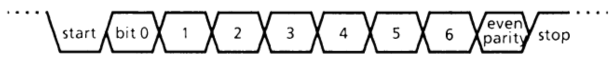
Transmitting

Mode	9 Bit	8 Bit + Parity	8 Bit, 7 Bit + Parity, 7 Bit
Interrupt timing	Just before last bit is transmitted	←	←

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

The 7-bit mode can be set by setting serial channel mode register SCMOD <SM1, 0> to "01".
In this mode, a parity bit can be added, and the addition of a parity bit can be enabled or disabled by serial channel control register SCCR <PE>, and even parity or odd parity is selected by SCCR <EVEN> when <PE> is set to "1" (enable).

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



← Transfer direction (transfer rate: 2400 bps at fc = 9.8304 MHz)

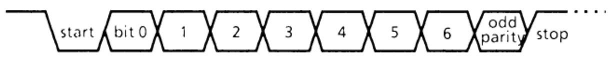
```
        7 6 5 4 3 2 1 0
P3FR  ← - X X X X 1 - X       Select P32 as the TxD pin.
SCMOD ← X 0 - X 0 1 0 1       Set 7-bit UART mode.
SCCR  ← X 1 1 X X X X X       Add an even parity.
BRGCR ← 0 X 1 0 0 1 0 0       Set transfer rate at 2400 bps.
TRUN  ← X X 1 - - - - -       Start the prescaler for the baud rate generator.
INTEL ← - - - - - 1 - -       Enable INTTX interrupt.
SCBUF ← * * * * * * * *       Set data for transmission.
```

(Note) X ; Don't care - ; No change

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

The 8-bit UART mode can be specified by setting SCMOD <SM1, 0> to "10". In this mode, parity bit can be added. the addition of a parity bit is enabled or disabled by SCCR <PE>, and even parity or odd parity is selected by SCCR <EVEN> when <PE> is set to "1" (enable).

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



← Direction of transmission (transmission rate: 9600 bps @ fc = 9.8304 MHz)

Main setting

```
              7 6 5 4 3 2 1 0
  SCMOD  ←  - 0 1 X 1 0 0 1     Enable receiving in 8-bit UART mode.
  SCCR   ←  X 0 1 X X X X X     Add an odd parity.
  BRGCR  ←  0 X 0 1 0 1 0 0     Set transfer rate at 9600 bps.
  TRUN   ←  X X 1 - - - - -     Start the prescaler for the baud rate generator.
  INTEL  ←  - - - - 1 - - -     Enable INTRX interrupt.
```

INTRX processing

```
  Acc    ←  SCCR AND 00000011     Check for error.
  if Acc ≠  0 then error
  Acc    ←  SCBUF                 Read the received data.
```

(Note) X ; don't care - ; no change

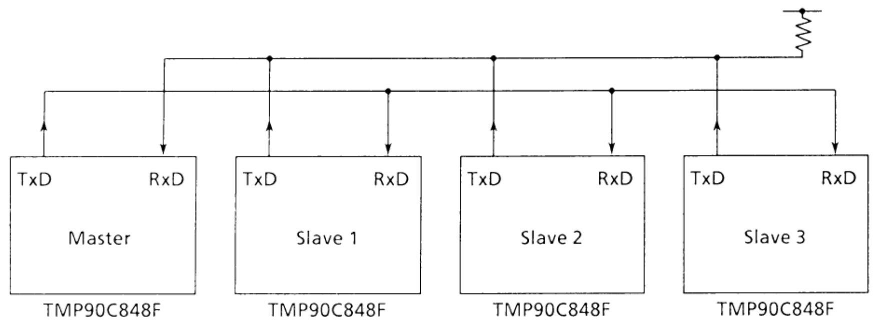
(3) Mode 3 (9-Bit UART Mode)

The 9-bit UART mode can be specified by setting serial channel mode register SCMOD0 <SM01, 00> to "11". In this mode, a parity bit cannot be added.

For transmission, the MSB (9th) bit is written to SCMOD <TB8>, while in receiving it is stored in SCCR <RB8>. For writing and reading the buffer, the MSB is read or written first then SCBUF.

Wake-up function

In 9-bit UART mode, the wake-up function of the slave controllers is enabled by the setting SCMOD <WU> to "1". The interrupt INTRX is generated when SCCR <RB8> = 1.



Note : TxD pin of the slave controllers must be in open drain output mode.

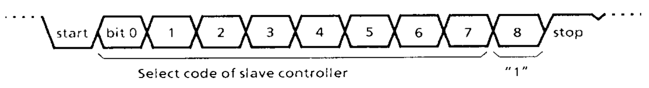
Figure 3.9 (13). Serial Link Using Wake-up Function

Protocol

① Select the 9-bit UART mode for the master and slave controllers.

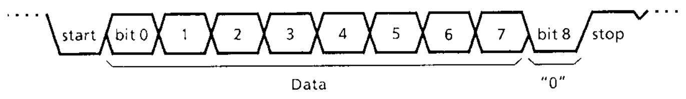
② Set SCMOD <WU> bit of each slave controller to “1” to enable receiving.

③ The master controller transmits one frame data including the 8-bit select code for the slave controllers. The MSB (bit 8) SCMOD <TB8> is set to “1”.



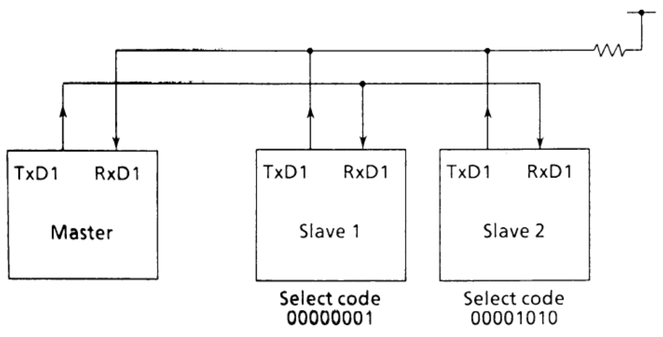
④ Each slave controller receives the above frame, and clears the WU bit to “0” if the above select code matches its own select code.

⑤ The master controller transmits data to the specified slave controller whose SCMOD <WU> bit is cleared to “0”. The MSB (bit 8) SCMOD <TB8> is cleared to “0”.



⑥ The other slave controllers (with the SCMOD <WU> bit remaining at “1”) ignore the receiving data because their MSBs (bit 8 or SCCR <RB8>) are set to “0” is disable the interrupt INTRX.
When the WU bit is cleared to “0”, the interrupt INTRX occurs, so that the slave controller can read the receiving data.
The slave controllers (WU = 0) transmit data to the master controller, and it is possible to indicate the end of data receiving to the master controller by this transmission.

Setting example: To link two slave controllers serially with the master controller, and use the internal clock ø1 (fc/2) as the transfer clock.



- Setting the master controller

```
        ┌ P3FR  ← - X X X X 1 - X     Select P35 as TxD pin and P32 as RxD pin.
        │ INTEL ← - - - - 1 1 - -     Enable INTRX and INTTX.
Main    │ SCMOD ← 1 0 1 0 1 1 1 0     Set ϕ1 (fc/2) as the transmission clock in 9-bit UART mode.
        └ SCBUF ← 0 0 0 0 0 0 0 1     Set the select code for slave controller 1.

INTTX     ┌ SCMOD ← 0 - - - - - - -   Sets SCMOD<TB> to"0".
interrupt └ SCBUF ← * * * * * * * *   Set data for transmission.
```

- Setting the slave controller 2

```
        ┌ P3FR  ← - X X X X - 1 X     Select P32 as RxD pin and P37 as TxD pin (open drain
        │                             output).
Main    │ INTEL ← - - - - 1 1 - -     Enable INTRX and INTTX.
        └ SCMOD ← 0 0 1 1 1 1 1 0     Set <WU> to "1" in the 9-bit UART transmission mode
                                      with transfer clock ϕ1 (fc/2).
          ┌ Acc  ← SCBUF
INTRX     │ if Acc = Select code
interrupt └ then SCMOD ← - - - 0 - - - -     Clear <WU> to "0".
```

(Note) X ; Don't care - ; No change

3.10 8-Bit Half-Flash A/D Converter

The TMP90C848 has an 8-bit high-speed, half-flash A/D converter with 16-channel analog input. The features are as follows.

- 8-bit half-flash A/D converter with 16-channels analog input pins
- Minimum sampling rate: 4 states (800ns @fc =10MHz)
- Software start (register write) trigger with single or repeat conversion mode

The A/D converter block diagram is shown in Figure 3.10 (1).

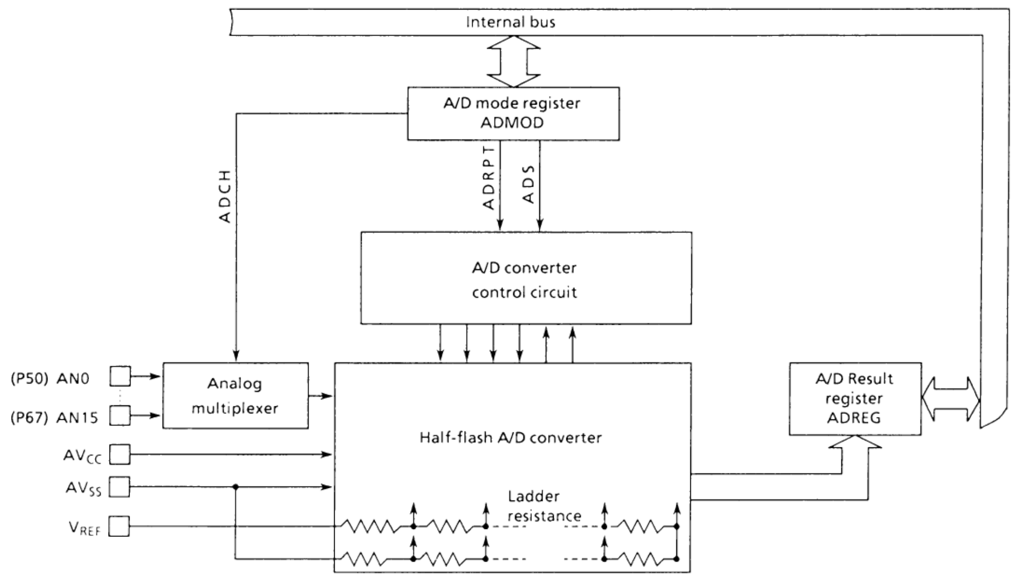


Figure 3.10 (1). A/D Converter Block Diagram

3.10.1 Basic Operation of the Half-Flash A/D Converter

This is a two-time conversion type A/D converter that converts the upper four bits and lower four bits separately. The function outline is shown in Figure 3.8 (2).

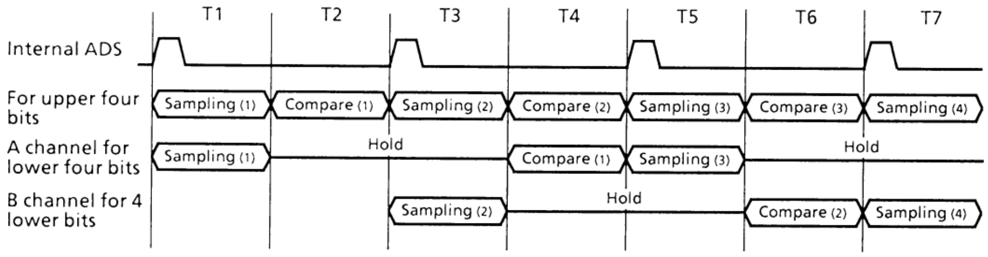


Figure 3.8 (1). Half-Flash A/D Converter Outline

When the A/D converter start signal (ADS) is input, the analog input voltage in T1 is sampled by the A/D converter for the upper four bits and the A channel for the lower four bits. The A/D converter for the upper four bits compares the output voltage from the internal ladder resistance with input voltage in T2, and outputs the conversion results of the upper four bits. The A channel A/D converter for the lower four bits compares in T4 in the same way that held the voltage in T1. The 8-bit conversion results can be obtained in T5.

The A/D converter for the lower four bits has two channels (A and B). The next analog input voltage in T3 is sampled by the A/D converter for the upper four bits and B channel A/D converter for the lower four bits.

This type of processing enables the high-speed A/D conversion with a minimum sampling rate of 4 states (800ns @ 10MHz).

3.10.2 Operation

(1) A/D converter start operation

The A/D converter starts by writing “1” to ADMOD <ADS>.
<ADS> is always read as “0”.

(2) A/D converter repeat specification

In the repeat mode, the A/D converter starts automatically after completion each conversion.
The A/D conversion in repeat mode is started by writing “1” to both <ADS> and <ADRPT>.
To end the repeat mode operation, write “0” to <RPT>. The repeat mode will end when the current conversion is completed.
Read the A/D conversion result storage register ADREG in the repeat mode since it contains the newest conversion data.

The repeat mode operation timing is shown in Figure 3.10 (3).

<ADRPT> is cleared to “0” by resetting: therefore, the A/D converter becomes the one-time conversion mode.

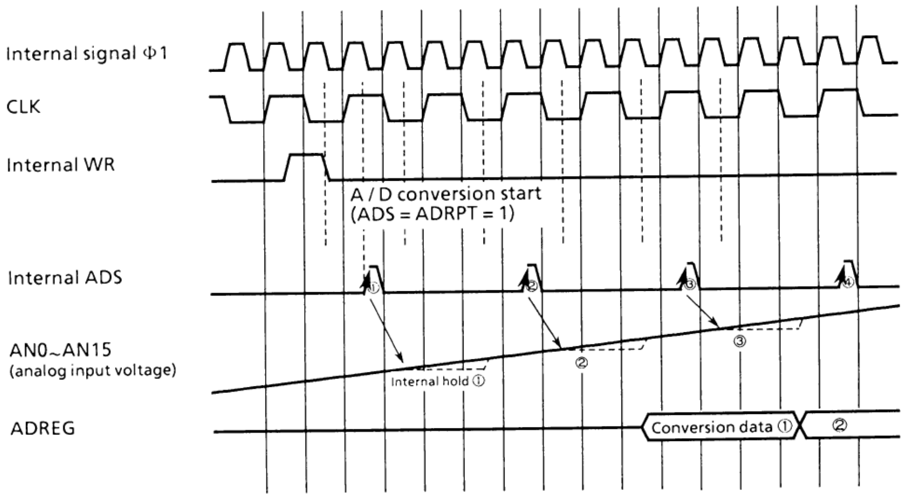


Figure 3.10 (3). Repeat Mode Operation Timing

(4) Analog input channel

Before starting the A/D conversion, select one of the 16 analog input channels (AN0 ~ AN15) with ADMOD <ADCH>.

AN0 (P30) is set as the analog input pin by clearing <ADCH> to “0” by reset. To use AN1 (P31), write “1” to <ADCH>.

The pin which is not used as an analog input can be used as an ordinary input port.

3.10.3 Control Register

ADMOD (FFECH)

	7	6	5	4	3	2	1	0
bit Symbol	ADRPT		ADS	Fixed at "0"		ADCH		
Read/Write	R/W		R/W			R/W		
After reset	0		0	0	0	0	0	0
Function	A/D conversion repeat mode 0 : One-time conversion 1 : REPEAT		1 : A/D conversion start 0 : Don't care		Analog Input Channel Select			

Analog input channel

0000	AN0
0001	AN1
0010	AN2
0011	AN3
0100	AN4
0101	AN5
0110	AN6
0111	AN7
1000	AN8
1001	AN9
1010	AN10
1011	AN11
1100	AN12
1101	AN13
1110	AN14
1111	AN15

A/D conversion start

0	——
1	A/D conversion start

A/D conversion repeat mode

0	one-time conversion mode
1	Repeat conversion mode

Figure 3.10 (4). A/D Mode Register

ADREG (FFF0H)

	7	6	5	4	3	2	1	0
bit Symbol	——	——	——	——	——	——	——	——
Read/Write	R							
After reset	Undefined							

Figure 3.10 (5). A/D Converter Related Register

3.10.4 Analog Reference Voltage

The VREF pin is the High A/D converter analog reference voltage input pin. The A Vcc and A Dss pins are used as the A/D converter power supply.

The VREF pin is variable ($3.5 \leq VREF \leq Vcc$); however, when the VREF voltage is below 5V, the conversion error for the LSB tends to increase. Refer to "4.3 A/D Converter Electrical Characteristics" for the specifications.

Program Example

To A/D convert the analog input voltage of the AN1 (P51) pin in the repeat mode:

```
ADMOD←1 X 100001  Starts conversion with channel 1
                  in the repeat mode.
```

(Note) X; Don't care

3.11 Watchdog Timers (Runaway Detecting Timer)

When CPU operation malfunctions (runaway) usually caused by noise or other disturbances, the watchdog timer (WDT) detects the situation and returns it to a normal state. When the WDT detects a malfunction, a non-maskable interrupt or a hardware reset is generated.

3.11.1 Configuration

Figure 3.11 (1) shows the block diagram of the watchdog timer (WDT).

The watchdog timer consists of: a 20-stage binary counter which uses ϕ1 (@fc/2) as the input clock, a selector, which slects one of four outputs sent from the binary counter, a flip-flop, for enable/disable control, and two control registers.

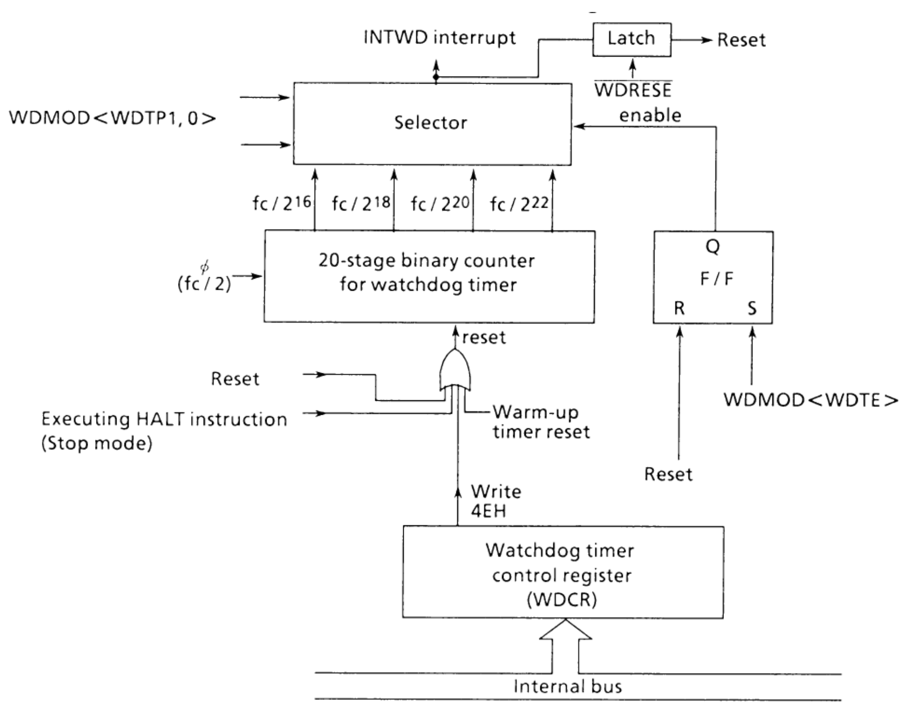


Figure 3.11 (1). Block Diagram of Watchdog Timer

3.11.2 Control Registers

The watchdog timer (WDT) is controlled by two control registers WDMOD, WDCR, and CLKMOD.

(1) Watchdog Timer Mode Register (WDMOD)

① Setting a watchdog timer detection period (WDTP)

A 2-bit register is used to set the watchdog timer interrupt time or hardware reset time in order to detect a runaway. The WDTP is initialized to “00” when reset, and therefore 2^{16}/fc is set. (The number of states is approx. 32,768 [state].)

② Watchdog Timer Enable/Disable Control Register (WDTE)

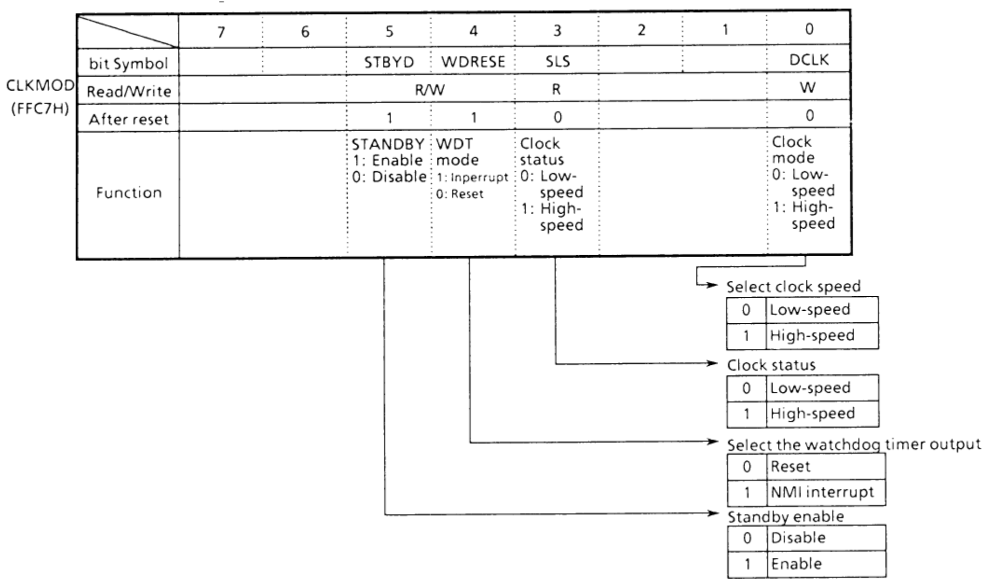
When reset WDTE is initialized to “1”, which sets the watchdog timer to disable.

To enable the watchdog timer, it is necessary to clear this bit to “0”.

The mode cannot be switched back from enable to disable.

③ Setting Watchdog Timer Mode

When reset, the WDRESE is initialized to “1”, which sets the watchdog timer to interrupt mode. When the WDRESE is reset to “0”, the watchdog timer initiates a hardware reset. Once the hardware reset mode is set, the mode cannot be changed to interrupt mode. (Interrupt mode is set only by reset.)



CLKMOD (FFC7H)

	7	6	5	4	3	2	1	0
bit Symbol			STBYD	WDRESE	SLS			DCLK
Read/Write			R/W		R			W
After reset			1	1	0			0
Function			STANDBY 1: Enable 0: Disable	WDT mode 1: Inperrupt 0: Reset	Clock status 0: Low-speed 1: High-speed			Clock mode 0: Low-speed 1: High-speed

Select clock speed

0	Low-speed
1	High-speed

Clock status

0	Low-speed
1	High-speed

Select the watchdog timer output

0	Reset
1	NMI interrupt

Standby enable

0	Disable
1	Enable

WDMOD (FFD2H)

	7	6	5	4	3	2	1	0
bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
Read/Write	R / W	R / W		R / W	R / W		R	R / W
After reset	1	0	0	0	0	0	Undefined	0
Function	WDT Enable 0 : Enable 1 : Disable	00 : 2^{16} / fc 01 : 2^{18} / fc 10 : 2^{20} / fc 11 : 2^{22} / fc	Detection period	Warm-up period (When stop mode is released) 0 : 2^{14} / fc 1 : 2^{16} / fc (When clock mode is switched 0 : 2^{11} / fc 1 : 2^{13} / fc)	Standby Mode 00 : RUN mode 01 : STOP mode 10 : IDLE1 mode 11 : ——		Inverts each time the EXX instruction is executed.	1 : Drives the pin even in STOP mode.

180790

Explained in "3.4.4 STOP mode".

Inverts each time the EXX instruction is executed.

Select standby mode by HALT instruction (HALT Mode)

00	RUN mode
01	STOP mode
10	IDLE1 mode
11	——

Select warm-up period when returned from STOP mode

0	2^{14} / fc (approx. 1.6ms)
1	2^{16} / fc (approx. 6.6ms)

(Note) The above times are given for the case fc = 10MHz.

Warm-up period when switched from low-speed to high-speed clock

0*	2^{11} / fc
1	2^{13} / fc

Select the watchdog timer detection period (Watch Dog Timer Period)

00	2^{16} / fc (approx. 6.6ms)
01	2^{18} / fc (approx. 26.2ms)
10	2^{2C} / fc (approx. 105ms)
11	2^{22} / fc (approx. 419ms)

(Note) The above times are given for the case fc = 10MHz.

Watchdog timer enable / disable control

0	Enable (Note)
1	Disable

(Note) The watchdog timer cannot be reset from enable to disable.

Figure 3.11 (2). Watchdog Timer Mode Register

(2) Watchdog Timer Control Register (WDCR)

This register is used to disable the watchdog timer function and clear the binary counter.

• Disable control

The watchdog timer cannot be reset from enable to disable.

• Enable control

Set the WDMOD <WDTE> to "0".

• Clear control of binary counter

The binary counter can be cleared and resumes counting by writing the clear code (4EH) in the WDCR register.

```
WDCR  ← 0 1 0 0 1 1 1 0          Write a clear code (4EH)
```

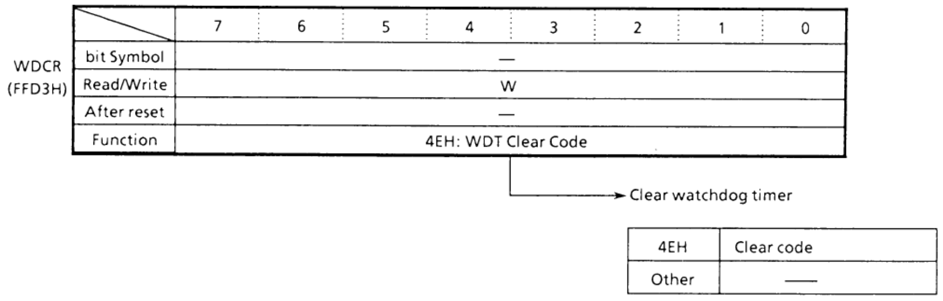


Figure 3.11 (3). Watchdog Timer Control Register

3.11.3 Operation

The watchdog timer generates an interrupt INTWD or a hardware reset after a specified detection timewhich is set with the WDMOD <WDTP1,0> register, and software (instruction) clears the watchdog timer binary counter to zero before the interrupt INTWD or the hardware reset occurs. If the CPU malfunctions (runaway) due to noise or other reasons and does not execute an instruction to clear the binary counter, the binary counter overflows. This results in an interrupt INTWD or hardware reset execution. The CPU is notified of malfunction (runaway) by the interrupt INTWD and executes the anti-malfunction (runaway) program to return to normal operation.

The watchdog timer does not operate after a reset is released. The watchdog a timer starts by writing “0” to the WDMOD <WDTE>, and does not change to disable afterward.

The watchdog timer stops its operation in the STOP and IDLE1 modes. After STOP is released, the watchdog timer resumes its operation as soon as the warm-up period is finished.

The watchdog timer operates in RUN mode.

Example : ① Clear the binary counter.

```
WDCR  ← 0 1 0 0 1 1 1 0          Write a clear code (4EH)
```

② Set the watchdog timer detection period to 2^{18}/fc.

```
WDMOD ← 1 0 1 - - - X X
```

③ Set STOP mode. (Warm-up time : 2^{16}/fc) (Except that HALTD is not set to “1”.)

```
WDMOD ← - - - 1 0 1 X X          Set STOP mode.
Execute the HALT instruction.    Set standby mode.
```

(Note) Since the watchdog timer is also used as a warm-up timer, the watchdog timer is cleared when switching from low-speed to high-speed clock mode, or when releasing standby mode.

3.12 8-Bit High-speed PWM

The TMP90C848F has an internal 8-channel high-speed PWM. The 8-channel high-speed PWM can generated different waves from a data register of each channel by writing data to the register. The high-speed PWM operates only in high-speed clock mode, not in low-speed clock mode.

Figure 3.12 (1) shows the block diagram of the high-speed PWM.

Each channel of the HPWM consists of an 8-bit data register, 8-bit comparator and additionally, a pulse generation circuit. Also, there is an 8-bit up-counter which is used commonly by all channels. The clock input of the up-counter is sent by the high-speed clock (X1).

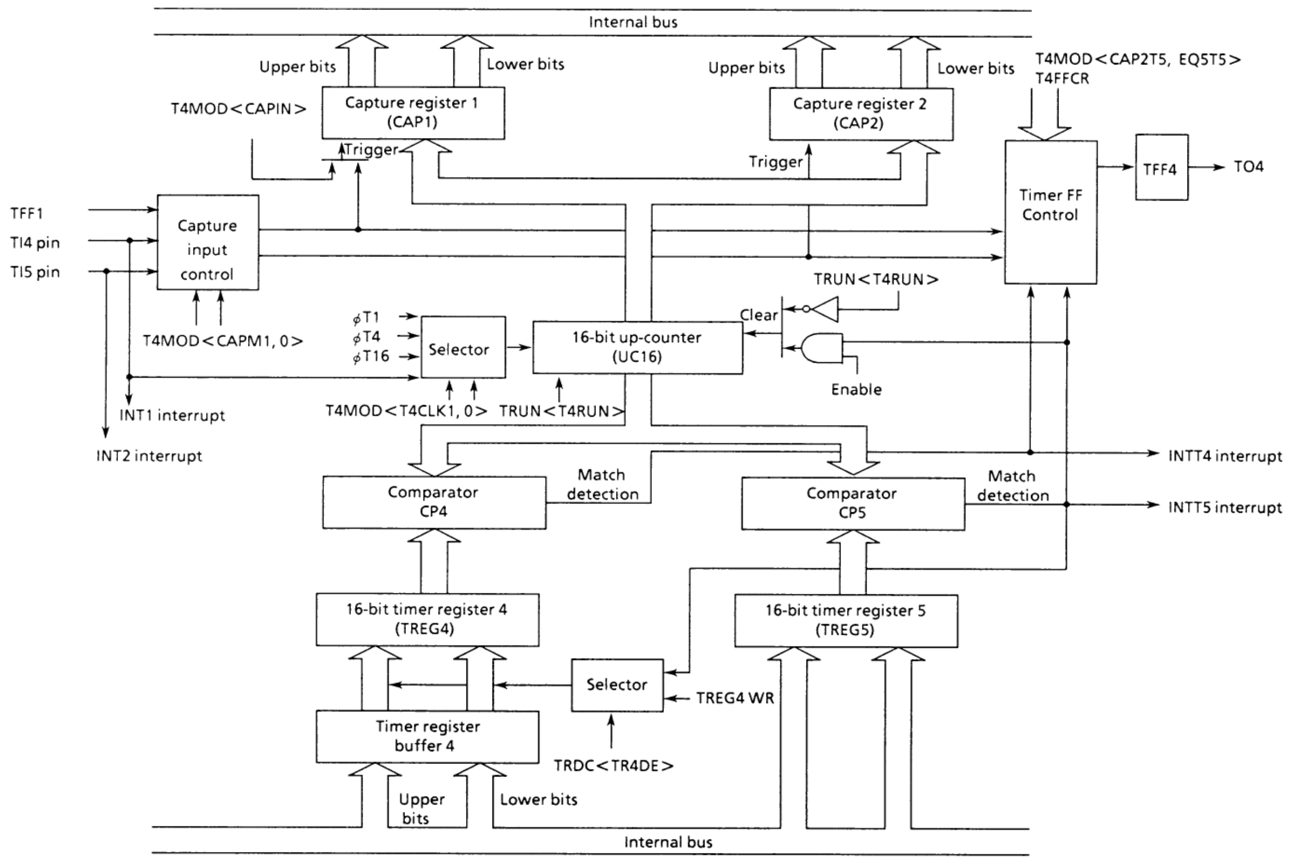


Figure 3.12 (1). Block Diagram for High-Speed PWM

HPWM data register

HPWMR0~7 (FFF0 ~FFF7H)		7	6	5	4	3	2	1	0
	bit Symbol	——	——	——	——	——	——	——	——
	Read/Write	W							
	After reset	Undefined							

HPWM enable register

HPWMER (FFEEH)		7	6	5	4	3	2	1	0
	bit Symbol	HPE7	HPE6	HPE5	HPE4	HPE3	HPE2	HPE1	HPE0
	Read/Write	R/W							
	After reset	0							
	Function	HPWM output enable setting 0 : Disable 1 : Enable							

Set HPWM output

0	Disable
1	Enable

HPWM control register

HPWMCR (FFEFH)		7	6	5	4	3	2	1	0
	bit Symbol						PWCCR	PWMMOD	
	Read/Write						R/W		
	After reset						1	0	0
	Function						PWM Reset 0: Set 1: Reset	PWM mode 00: Mode 0 01: Mode 1 10: Mode 2 11: ——	

Set PWM mode

00	Mode 0 (8bit)
01	Mode 1 (7bit)
10	Mode 2 (6bit)
11	——

PWM reset flag

0	Set
1	Reset

Figure 3.12 (2). Registers of High-Speed PWM

3.12.1 Operation

The high-speed PWM is controlled by the controlled register (HPWMCR), output enable register (HPWMER), and data register (HPWMR0 - 7). To write data to the above registers, set the HPWMCR <PWCCR> to “0”, which is enable mode. When the HPWMCR <PWCCR> is set to “1”, these registers are in reset mode, which sets up the high-speed PWM for a software reset.

(1) Operation Mode

The high-speed PWM has three operation modes.

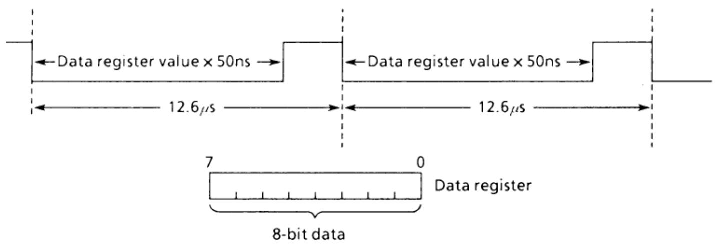
- 8-bit mode: (T = 2^8 x clock cycle, f = *80KHz)
- 7-bit mode: (T = 2^7 x clock cycle, f = *160KHz)
- 6-bit mode: (T = 2^6 x clock cycle, f = *320KHz)

(Note)* indicates the value when the high-speed clock (x1) operates at 20MHz.

Operation mode is set by HPWMCR <PWMMOD0, 1>. Operation mode applies commonly to all channels. Two modes cannot be used at any given time.

① 8-bit mode

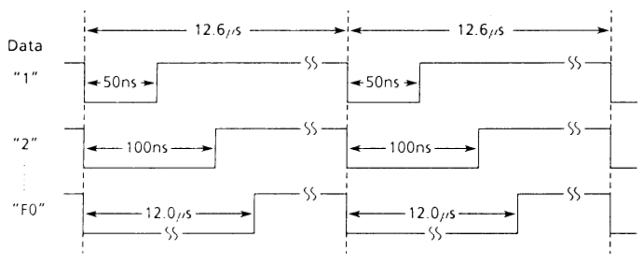
8-bit mode generates a pulse with 12.8µs cycle at a frequency of approximately 80KHz (x1 = 20MHz).



The minimum pulse width is 50ns (data “1”) and the maximum pulse width is 12.0µs (data “F0”).

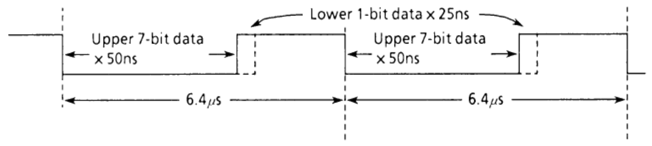
Pulse width = 8-bit data x 50ns

A wave cycle example is shown below. (The value is when x1 = 20MHz.)



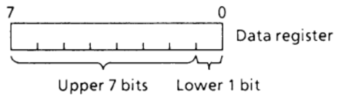
② 7-bit mode

The 7-bit mode generates a pulse with 6.4µs cycle at a frequency of approximately 160KHz (x1 = 20MHz).



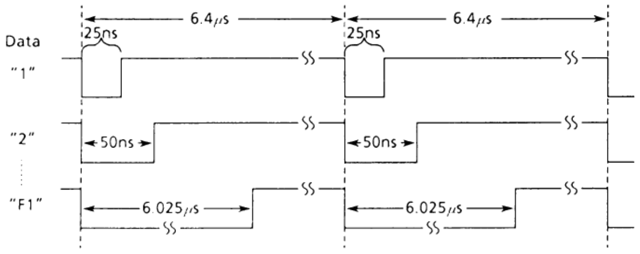
7-bit mode has for a cycle (2^7 x 5ns*/cycle) and 1 bit for a 25ns resolution (1/2 cycle of high-speed clock (x1)). When the lower 1 bit is “1”, the additional 25ns pulse is output.

The minimum pulse width is 25ns (data “1”) and the maximum pulse width is 6.025µs (data “F1”).



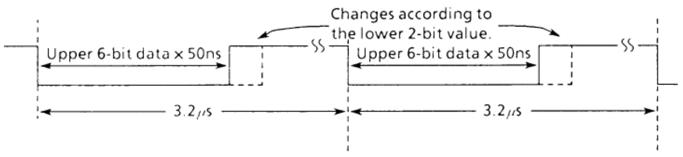
Pulse width = (Upper 7-bit data x 50ns) + (Lower 1-bit data x25ns)

A wave cycle example is shown below. (The value is when x1 = 20MHz.)

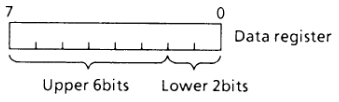


③ 6-bit mode

6-bit mode generates a pulse with 3.2µs cycle at a frequency of approximately 320KHz (x1 = 20 MHz).



6-bit mode has 6 bits for a cycle (2^6 x 50ns */cycle) and 2 bits for a 12.5ns resolution. Although the actual resolution every other cycle is 25ns, a 12.5ns resolution is simulated in the following way: The first cycle outputs a 25ns pulse, which is averaged with the second cycle of 0ns. The pattern alternates continually and averages to a 12.5ns (data “1”) and the maximum equivalent pulse width is 3.0375µs (data “F3”).

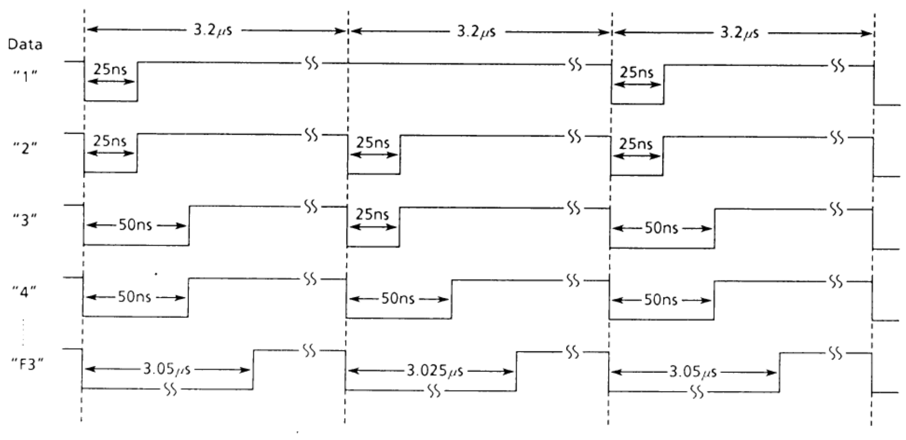


Pulse width = (Upper 6-bit data x 50ns + (*Lower 2-bit data)

Equivalent time added for lower 2-bit data is shown below.

2-bit data	Equivalent time added
0 0	0ns
0 1	12.5ns
1 0	25.0ns
1 1	37.5ns

A wave cycle example is shown below. (The value is when x1 = 20MHz.)



(2) Output port setting

The high-speed PWM is used together with P2 and can output PWM waves by setting P2FR <P20F ~ P27F> to “1”>.

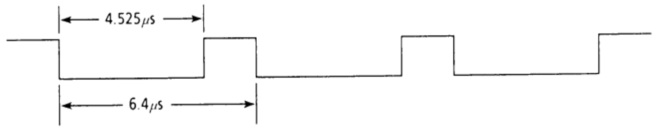
(3) Output enable bit register setting

To output PWM waves, the output enable bit register HPWMER <HPE0 ~ 7> needs to be set to “1”. (This register is cleared to “0” after reset.)

(4) Output data setting

Output data is set by writing to HPWMR0 ~ 7 (FFF0H ~ FFF7H).

Example: To output 4.525μs wave with HPWM0 in 7-bit mode with a high-speed clock (x 1) = 20MHz:



When the resolution in 7-bit mode is 25ns, a 4.525μs pulse is output by setting the following to HPWM0:
4.525μs ÷ 25ns =181 = B5H

	MSB 7	6	5	4	3	2	1	LSB 0	
P2FR	-	-	-	-	-	-	-	1	Set P20 to HPWM.
HPWMCR	X	X	X	X	X	0	-	-	Specify setting enable mode.
HPWMCR	X	X	X	X	X	0	0	1	Set 7-bit mode.
HPWMR0	1	0	1	1	0	1	0	1	Write “B5H”.
HPWMER	-	-	-	-	-	-	-	1	Switch HPWM output to enable.

(Note) x ; don’t care - : no change

4. Electrical Characteristics (Preliminary)

TMP90C848F

4.1 Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
V_{CC}	Power supply voltage	-0.5 ~ + 7	V
V_{IN}	Input voltage	-0.5 ~ V_{CC} + 0.5	V
P_D	Power dissipation (Ta = 70°C)	500	mW
T_{SOLDER}	Soldering temperature (10s)	260	°C
T_{STG}	Storage temperature	-65 ~ 150	°C
T_{OPR}	Operating temperature	-20 ~ 70	°C

4.2 DC Characteristics

V_{CC} = 5V ± 10% TA = -20 ~ 70°C
High-speed clock: 16 ~ 20MHz, Low-speed clock: 0.5 ~ 1MHz
Typical values are for TA = 25°C and Vcc = 5V.

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL}	Input Low Voltage (P0)	-0.3	0.8	V	–
V_{IL1}	P1, P3, P4, P5, P6, P7, P8	-0.3	0.3V_{CC}	V	–
V_{IL2}	$\overline{\text{RESET}}$, P82 (INTO)	-0.3	0.25V_{CC}	V	–
V_{IL3}	$\overline{\text{EA}}$	-0.3	0.3	V	–
V_{IL4}	X1, X1′	-0.3	0.2V_{CC}	V	–
V_{IH}	Input High Voltage (P0)	2.2	V_{CC} + 0.3	V	–
V_{IH1}	P1, P3, P4, P5, P6, P7, P8	0.7V_{CC}	V_{CC} + 0.3	V	–
V_{IH2}	$\overline{\text{RESET}}$, P82 (INTO)	0.75V_{CC}	V_{CC} + 0.3	V	–
V_{IH3}	$\overline{\text{EA}}$	V_{CC} - 0.3	V_{CC} + 0.3	V	–
V_{IH4}	X1, X1′	0.8V_{CC}	V_{CC} + 0.3	V	–
V_{OL} V_{OL1}	Output Low Voltage P40 ~ P43 (OPEN DRAIN Sink)	–	0.45	V	I_{OL} = 1.6mA
V_{OH} V_{OH1} V_{OH2} V_{OH3}	Output High Voltage P44 ~ 47 (OPEN DRAIN Source)	2.4 0.75V_{CC} 0.9V_{CC} 2.4	–	V V V	I_{OH} = -400μA I_{OH} = -100μA I_{OH} = -20μA I_{OH} = 10mA
I_{LI}	Input Leakage Current	0.02 (Typ)	±5	μA	0.0 ≤ Vin ≤ V_{CC}
I_{LO}	Output Leakage Current	0.05 (Typ)	±10	μA	0.2 ≤ Vin ≤ V_{CC} - 0.2
I_{CC} (Vcc - Vss)	Operating Current (RUN) Idle 1	15 (Typ) 1.5 (Typ)	30 5	mA mA	High-speed clock: 20MHz Low-speed clock: 1MHz
	STOP (TA = -20 ~ 7°C) STOP (TA = 0 ~ 50°C)	0.2 (Typ)	40 10	μA μA	0.2 ≤ Vin ≤ V_{CC} - 0.2
AIcc (A Vcc - A Vss)	Operating Current	7 (Typ)	15	mA	High-speed clock: 20MHz Low-speed clock: 1MHz (Repeat mode) A Vcc = 5V
V_{STOP}	Power Down Voltage (@STOP) (RAM back up)	2.0-	6.0	V	V_{IL2} = 0.2V_{CC}, V_{IH2} = 0.8V_{CC}
R_{RST}	$\overline{\text{RESET}}$, P1, P7, Pull Up Register	30	130	KΩ	–
CIO	Pin Capacitance	–	10	pF	testfreq = 1MHz
V_{TH}	Schmitt width ($\overline{\text{RESET}}$, P82)	0.4	1.0 (Typ)	V	–

4.3 A/D Converter Characteristics

V_{CC} = A Vcc = 5V ± 10% TA = -20 ~ 70°C
High-speed clock: 16 ~ 20MHz, Low-speed clock: 0.5 ~ 1MHz

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{REF}	Analog reference voltage	–	3.5	Vcc	Vcc	V
ΔV_{REF}	Analog reference voltage range	V_{REF} - Vss	3.5	Vcc	Vcc	
A Vss	Analog power supply voltage	–	Vss	Vss	Vss	
V_{AIN}	Analog input voltage range	–	Vss	–	Vcc	
I_{REFAD}	Supply current for analog reference voltage	–	–	0.8	2	mA

This A/D Converter is guaranteed only monotonicity because it has an offset value (when VAIN = 0V), but the 8-bit resolution is gotten except an offset value.

The A/D converted data is recommended to be processed relatively.

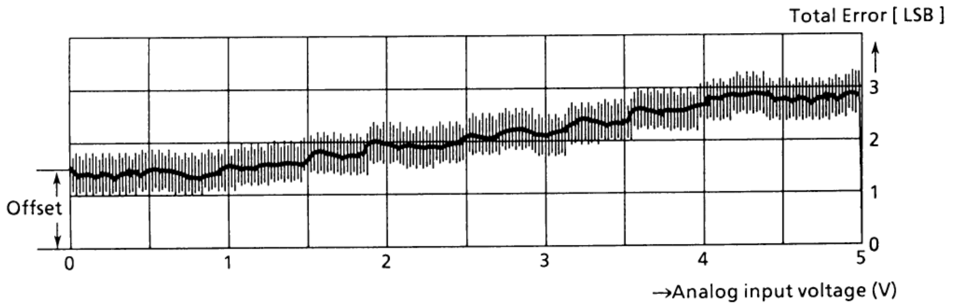


Figure 4.3 (1). A/D Converter Typical Conversion Characterics (V_{REF} = 5V, Vss = 0V)

4.4 Zero-Cross Characteristics

V_{CC} = 5V ± 10% TA = -20 ~ 70°C
High-speed clock: 16 ~ 20MHz, Low-speed clock: 0.5 ~ 1MHz

Symbol	Parameter	Condition	Min	Max	Unit
V_{ZX}	Zero-cross detection input	For AC, C = 0.1μF	1	1.8	VAC $_{p-p}$
A_{ZX}	Zero-cross accuracy	50/60Hz sine wave	–	135	mV
F_{ZX}	Zero-cross detection input frequency	–	0.04	1	kHz

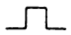
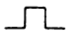
4.5 Timer/Counter Input Clock (TI0, TI2, and TI4)

V_{CC} = 5V ± 10% TA = -20 ~ 70°C
High-speed clock: 16 ~ 20MHz, Low-speed clock: 0.5 ~ 1MHz

Symbol	Parameter	Variable		16MHz Clock		Unit
		Min	Max	Min	Max	
t_{VCK}	Clock cycle	8x + 100	–	900	–	ns
t_{VCKL}	Low clock pulse width	4x + 40	–	440	–	ns
t_{VCKH}	High clock pulse width	4x + 40	–	440	–	ns

4.6 Interrupt Operation

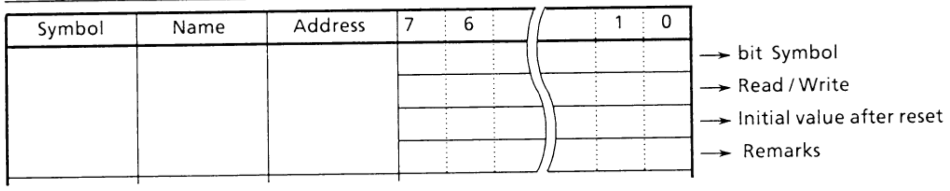
V_{CC} = 5V ± 10% TA = -20 ~ 70°C
High-speed clock: 16 ~ 20MHz, Low-speed clock: 0.5 ~ 1MHz

Symbol	Parameter	Variable		10MHz Clock		Unit
		Min	Max	Min	Max	
t_{INTAL}	$\overline{\text{NMI}}$, INT0 Low level pulse width	4x	–	400	–	ns
t_{INTAH}	$\overline{\text{NMI}}$, INT0 High level pulse width 	4x	–	400	–	ns
t_{INTBL}	INT1, INT2 Low level pulse width	8x + 100	–	900	–	ns
t_{INTBH}	INT1, INT2 High level pulse width 	8x + 100	–	900	–	ns

5. Table of Special Function Registers (SFRs)

The special function registers (SFR) include the I/O ports and peripheral control registers allocated to the 56-byte addresses from 0FFC0H to 0FFF7H.

Configuration of the table



TMP90C848F Register Map

Address	Symbol	Address	Symbol
0FFC0H	P0	0FFE0H	CAP1L/TREG4L
0FFC1H	P0CR	0FFE1H	CAP1H/TREG4H
0FFC2H	P1	0FFE2H	CAP2L/TREG5L
0FFC3H	P1CR	0FFE3H	CAP2H/TREG5H
0FFC4H	P2	0FFE4H	T4MOD
0FFC5H	P2CR	0FFE5H	T4FFCR
0FFC6H	P3	0FFE6H	INTEL
0FFC7H	CLKMOD	0FFE7H	INTEH
0FFC8H	P3FR	0FFE8H	DMAEL
0FFC9H	P4	0FFE9H	DMAEH
0FFCAH	P4CR	0FFEAH	IRFL
0FFCBH	P5	0FFEBH	IRFH
0FFCCH	P6	0FFECH	ADMOD
0FFCDH	P7	0FFEDH	BRGCR
0FFCEH	P7CR	0FFEEH	HPWMER
0FFCFH	P8	0FFEFH	HPWMCR
0FFD0H	P8CR	0FFF0H	HPWMR0/ADREG
0FFD1H	P8FR	0FFF1H	HPWMR1
0FFD2H	WDMOD	0FFF2H	HPWMR2
0FFD3H	WDCR	0FFF3H	HPWMR3
0FFD4H	TREG0	0FFF4H	HPWMR4
0FFD5H	TREG1	0FFF5H	HPWMR5
0FFD6H	TREG2	0FFF6H	HPWMR6
0FFD7H	TREG3	0FFF7H	HPWMR7
0FFD8H	T01MOD	0FFF8H	
0FFD9H	T23MOD	0FFF9H	
0FFDAH	TFFCR	0FFFAH	
0FFDBH	TRUN	0FFFBH	
0FFDCH	TRDC	0FFFCH	
0FFDDH	SCMOD	0FFFDH	
0FFDEH	SCCR	0FFFEH	
0FFDFH	SCBUF	0FFFFH	

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0	Port0	0FFC0H	P07	P06	P05	P04	P03	P02	P01	P00
			R/W							
			Input mode							
P0CR	Port0 Control Reg.	0FFC1H	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
			W							
			0	0	0	0	0	0	0	0
			0 : IN 1 : OUT (Select I/O on bit basis)							
P1	Port1 (With Pull-up resistor)	0FFC2H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Input mode							
P1CR	Port1 Control Reg.	0FFC3H	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT (Select I/O on bit basis)							
P2	Port2	0FFC4H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			1	1	1	1	1	1	1	1
P2FR	Port2 Function Reg.	0FFC5H	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0: Port output 1: PWM output (Select on bit basis)							
P3	Port3	0FFC6H	–	–	–	–	P33	P32	P31	P30
							R	R/W	R/W	R/W
							Input	1	1	1
P3FR	Port3 Function Reg.	0FFC8H	EXT	–	–	–	–	TXDC	ODE	–
			W					R/W	R/W	
			0					0	0	
			P1 control 0: I/O Port 1: Address					P32 control 0: Port 1: TxD output	P32 control 0: CMOS 1: Open Drain	
P4	Port4	0FFC9H	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			Input mode							
P4CR	Port4 Control Reg.	0FFCAH	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT (Select I/O on bit basis)							

Symbol	Name	Address	7	6	5	4	3	2	1	0
P5	Port5	0FFCBH	P57	P56	P55	P54	P53	P52	P51	P50
			R							
			Input only							
			Shared with analog input pin (AN0~7)							
P6	Port6	0FFCCH	P67	P66	P65	P64	P63	P62	P61	P60
			R							
			Input only							
			Shared with analog input pin (AN8~15)							
P7	Port7	0FFCDH		Fixed at "0"			P73	P72	P71	P70
							R/W			
							Input mode			
P7CR	Port7 Control Reg.	0FFCEH					P73C	P72C	P71C	P70C
							W			
							0	0	0	0
			0: IN 1: OUT (Select I/O on bit basis)							
P8	Port8	0FFCFH			P85	P84	P83	P82	P81	P80
					R/W					
					Input mode					
P8CR	Port8 Control Reg.	0FFD0H			P85C	P84C	P83C	P82C	P81C	P80C
					W					
					0	0	0	0	0	0
					0 : IN 1 : OUT (Select I/O on bit basis)					
P8FR	Port8 Function Reg.	0FFD1H			TO4S	ZCE2	ZCE1		TO3S	TO1S
			W							
					0	0	0		0	0
					P85/TO4 control 0: Port output 1: TO4 output	P84/INT2/TI5 ZCD Enable 0: Disable 1: Enable	P83/INT1/TI4 ZCD Enable 0: Disable 1: Enable		P81/PO3 control 0: Port output 1: TO3 output	P80/TO1 control 0: Port output 1: TO1 output

Symbol	Name	Address	7 (MSB)	6	5	4	3	2	1	0 (LSB)
SCMOD	Serial Channel Mode Reg.	0FFDDH	TB8	Fixed at "0"	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			Undefined		0	0	0	0	0	0
			Transmiss-ion bit-8 data		1 : Receive Enable	1 : Wake up Enable	00 : —— 01 : UART 7bit 10 : UART 8bit 11 : UART 9bit		00 : TO2TRG 01 : BR 10 : $\phi 1$ 11 : ——	
SCCR	Serial Channel Control Register	0FFDEH	RB8	EVEN	PE	OERR	PERR	FERR		
			R	R/W		R (Cleared to "0" by reading)				
			Undefined	0	0	0	0	0		
			Bit 8 of receiving data	Parity 0 : Odd 1 : Even	1 : Parity Enable	1 : error Overrun	Parity	Flaming		
SCBUF	Serial Channel Buffer Register	0FFDFH	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
			TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0
			R (Receiving) / W (Transmission)							
			Undefined							
BRGCR	BRG CONTROL REGISTER	0FFEDH	Fixed at "0"	—	BG1	BG0	PS3	PS2	PS1	PS0
			—	—	R/W			R/W		
			—	—	0	0	0	0	0	0
			—	—	00 : fc/4 01 : fc/16 10 : fc/64 11 : fc/256		Divided frequency from prescaler			

			MSB							LSB
Symbol	Name	Address	7	6	5	4	3	2	1	0
INTEL	Interrupt Enable Mask Reg.	0FFE6H	IET4	IE1	IET5	IE2	IERX	IETX	Fixed at "0"	Fixed at "0"
			R/W							
			0	0	0	0	0	0	0	0
			1:Enable			0:Disable				
INTEH		0FFE7H		EDGE		IE0	IET0	IET1	IET2	IET3
				R/W		R/W				
				0		0	0	0	0	0
				INT0 0 : Level 1 : EDGE				1 : Enable 0 : Disable		
DMAEL	Micro DMA Enable Register	0FFE8H	DET4	DE1	DET5	DE2	DERX	DETX	Fixed at "0"	Fixed at "0"
			R/W							
			0	0	0	0	0	0	0	0
			1:Enable			0:Disable				
DMAEH		0FFE9H				DE0	DET0	DET1	DET2	DET3
						R/W				
						0	0	0	0	0
						1:Enable		0:Disable		
IRFH	Interrupt Request Flag & IRF Clear	0FFEBH				IRF0	IRFT0	IRFT1	IRFT2	IRFT3
						R				
						0	0	0	0	0
						Interrupt Request Flag 1 : Interrupt being requested				
IRFL		0FFEAH	IRFT4	IRF1	IRFT5	IRF2	IRFRX	IRFTX	—	—
			R (Only IRF clear code can be used to write)							
			0	0	0	0	0	0	0	0
			1: Interrupt being requested (IRF is cleared to "0" by writing IRF clear code)							

CAP1L	16bit Timer/ Event Counter Capture Register 1	0FFE0H	—
			R
			Undefined
CAP1H		0FFE1H	—
			R
			Undefined
CAP2L	16bit Timer/ Event Counter Capture Register 2	0FFE2H	—
			R
			Undefined
CAP2H		0FFE3H	—
			R
			Undefined
TREG4L	16bit Timer/ Event Counter Register 4	0FFE0H	—
			W
			Undefined
TREG4H		0FFE1H	—
			W
			Undefined
TREG5L	16bit Timer/ Event Counter Register 5	0FFE2H	—
			W
			Undefined
TREG5H		0FFE3H	—
			W
			Undefined

MSB LSB

Symbol	Name	Address	7	6	5	4	3	2	1	0
T4MOD	16bit Timer/ Event Counter Mode reg.	0FFE4H			CAPIN	CAPM1	CAPM0	CLE	T4CLK1	T4CLK0
					W	R/W		R/W	R/W	
					—	0	0	0	0	0
					0 : Soft-Capture 1: don't care	Capture timing 00 : Disable 01 : TI4 ↑ TI5 ↑ 10 : TI4 ↑ TI4 ↓ 11 : TFF1 ↑ TFF1 ↓		1: TIMER4 Clear Enable	Timer 4 clock 00 : TI4 01 : ϕT1 10 : ϕT4 11 : ϕT16	
T4FFCR	16bit Timer Flip-Flop4 Control reg.	0FFE5H			CAP2T4	CAP1T4	EQ5T4	EQ4T4	TFF4C1	TFF4C0
					R/W				W	
					0	0	0	0	—	
					TFF4 inversion trigger 0 : Disable trigger 1 : Enable trigger				10 : Clear TFF4 01 : Set TFF4 00 : Invert TFF4 11 : don't care	
					When the up-counter value is loaded to CAP2	When the up-counter value is loaded to CAP1	When the up-counter matches TREG5	When the up-counter matches TREG4		

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	Watch Dog Timer Mode Reg.	0FFD2H	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
			R / W	R / W		R / W	R / W		R	R / W
			1	0	0	0	0	0	Undefined	0
			0:WDT Enable	WDT detecting time 00 : 2^{16}/fc 01 : 2^{18}/fc 10 : 2^{20}/fc 11 : 2^{22}/fc		Warming up time 0 : 2^{14}/fc 1 : 2^{16}/fc	Standby mode 00:RUN mode 01:STOP mode 10: IDLE1 mode 11: —		Inverts each time EXX instru-ction is executed.	1: To drive the pin even in STOP mode.
WDCR	Watch Dog Timer Control Reg.	0FFD3H	—							
			W							
			—							
			4EH : WDT Clear code							
CLK MOD	Clock Mode Reg.	0FFC7H			STBYD	WDRESE	SLS			DCLK
					R / W		R			W
					1	1	0			0
					STANBY 1 : Enable 0 : Disable	WDT mode 1 : Interrupt 0 : Reset	Clock status 0 : Low-speed 1 : High-speed			Clock mode 0 : Low-speed 1 : High-speed

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREG0	8bit Timer Register 0	0FFD4H	—							
			W							
			Undefined							
TREG1	8bit Timer Register 1	0FFD5H	—							
			W							
			Undefined							
TREG2	8bit Timer Register 2	0FFD6H	—							
			W							
			Undefined							
TREG3	8bit Timer Register 3	0FFD7H	—							
			W							
			Undefined							
T01MOD	Timer 0, 1 Hode Reg.	0FFD8H	T01M1	T01M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00 : 8bit Timer 01 : 16bit Timer 10 : 8bit PPG 11 : 8bit PWM		00 : — 01 : 2^6-1 10 : 2^7-1 11 : 2^8-1	PWM cycle	00 : TO0TRG 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : — 01 : ϕT1 10 : ϕT4 11 : ϕT16	
T23MOD	Timer 2, 3 Hode Reg.	0FFD9H	T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00 : 8bit Timer 01 : 16bit Timer 10 : 8bit PPG 11 : 8bit PWM		00 : — 01 : 2^6-1 10 : 2^7-1 11 : 2^8-1	PWM cycle	00 : TO2TRG 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : — 01 : ϕT1 10 : ϕT4 11 : ϕT16	
TFFCR	8bit Timer Flip-Flop Control reg.	0FFDAH	TFF3C1	TFF3C0	TFF3IE	TFF3IS	TFF1C1	TFF1C0	TFF1IE	TFF1IS
			W		R/W		W		R/W	
			—		0	0	—		0	0
			00 : Invent TFF3 01 : Set TFF3 10 : Clear TFF3 11 : Don't care		1 : TFF3 Invert Enable	0 : Inverts by Timer 2	00 : Invent TFF1 01 : Set TFF1 10 : Clear TFF1 11 : Don't care		1 : TFF1 Invert Enable	0 : Inverts by Timer 0
TRUN	Timer RUN Control Reg.	0FFDBH			PRRUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
					R/W					
					0	0	0	0	0	0
					Prescaler & Timer RUN/STOP Control 0: Stop & Clear 1: RUN					
TRDC	Timer Reg. Double Buffer Control Reg.	0FFDCH						TR4DE	TR2DE	TR0DE
									R/W	
									0	0
								Timer Reg. Double Buffer Control 0: Double Buffer Disable 1: Double Buffer Enable		

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD	A / D Mode Reg.	0FFECH	ADRPT		ADS	Fixed at "0"	ADCH			
			R / W		R / W		R / W			
			0		0		0	0	0	0
			A/D conversion repeat mode 0: One-time conversion 1: REPEAT		1: A/D conversion start 0: Don't care		0000 : AN0 0001 : AN1 0010 : AN2 0011 : AN3 0100 : AN4 0101 : AN5 0110 : AN6 0111 : AN7 1000 : AN8 1001 : AN9 1010 : AN10 1011 : AN11 1100 : AN12 1101 : AN13 1110 : AN14 1111 : AN15			
ADREG0	A / D Result Register	0FFF0H	—							
			R							
			Undefined							

Symbol	Name	Address	7	6	5	4	3	2	1	0
HPWMR0	HIGH PWM Register	0FFF0H	—							
			W							
			Undefined							
HPWMR1		0FFF1H	Same as above							
HPWMR2		0FFF2H	Same as above							
HPWMR3		0FFF3H	Same as above							
HPWMR4		0FFF4H	Same as above							
HPWMR5		0FFF5H	Same as above							
HPWMR6		0FFF6H	Same as above							
HPWMR7		0FFF7H	Same as above							
HPWMER	HIGH PWM ENABLE REGISTER	0FFEEH	HPE7	HPE6	HPE5	HPE4	HPE3	HPE2	HPE1	HPE0
			R / W							
			0							
			0 : Disable 1 : Enable							
HPWMCR	HIGH PWM RESET REGISTER	0FFEFH	—	—	—	—	—	PWCCR	PWM MOD	
								R/W		
								1	0	0
								PWM Reset 0: Set 1: Reset	PWM mode 00: Mode 0 01: Mode 1 10: Mode 2 11: ——	