CMOS 8-BIT MICROCONTROLLERS
TMP90C802AP/TMP90C802AM
TMP90C803AP/TMP90C803AM

1.   OUTLINE AND CHARACTERISTICS

The TMP90C802A is a high-speed advanced 8-bit micro controller applicable to a variety of equipment.

With its 8-bit CPU, ROM, RAM, timer/event counter and general-purpose serial interface integrated into a single CMOS chip, the TMP90C802A allows the expansion of external memories (up to 56K byte). The TMP90C803A is the same as the TMP90C802A but without the ROM.

The TMP90C802AP/803AP is in a DIP package.

The TMP90C802AM/803AM is in a SOP (Small Outline Package).

The characteristics of the TMP90C802A include:

(1)   Powerful instructions : 163 basic instructions, including
      Multiplication, division, 16-bit arithmetic operations, bit manipulation instruct-
      ions

(2)   Minimum instruction executing time: 320 ns (at 12.5 MHz oscillation frequency)

(3)   Internal ROM: 8K byte (The TMP90C803A does not have a built-in ROM.)

(4)   Internal RAM: 256 byte

(5)   Memory expansion
      External memory: 56K byte

(6)   General-purpose serial interface (1 channel)
      Asynchronous mode, I/O interface mode

(7)   8-bit timers (4 channels) : (1 external clock input)

(8)   Port with zero cross detection circuit (1 input)

(9)   Input/Output ports (TMP90C802A : 32pins, TMP90C803A : 6pins)

(10)   Interrupt function: 8 internal interrupts and 3 external interrupts

(11)   Micro Direct Memory Access (DMA) function (4 channels)

(12)   Watchdog timer

(13)   Standby function (4 HALT modes)
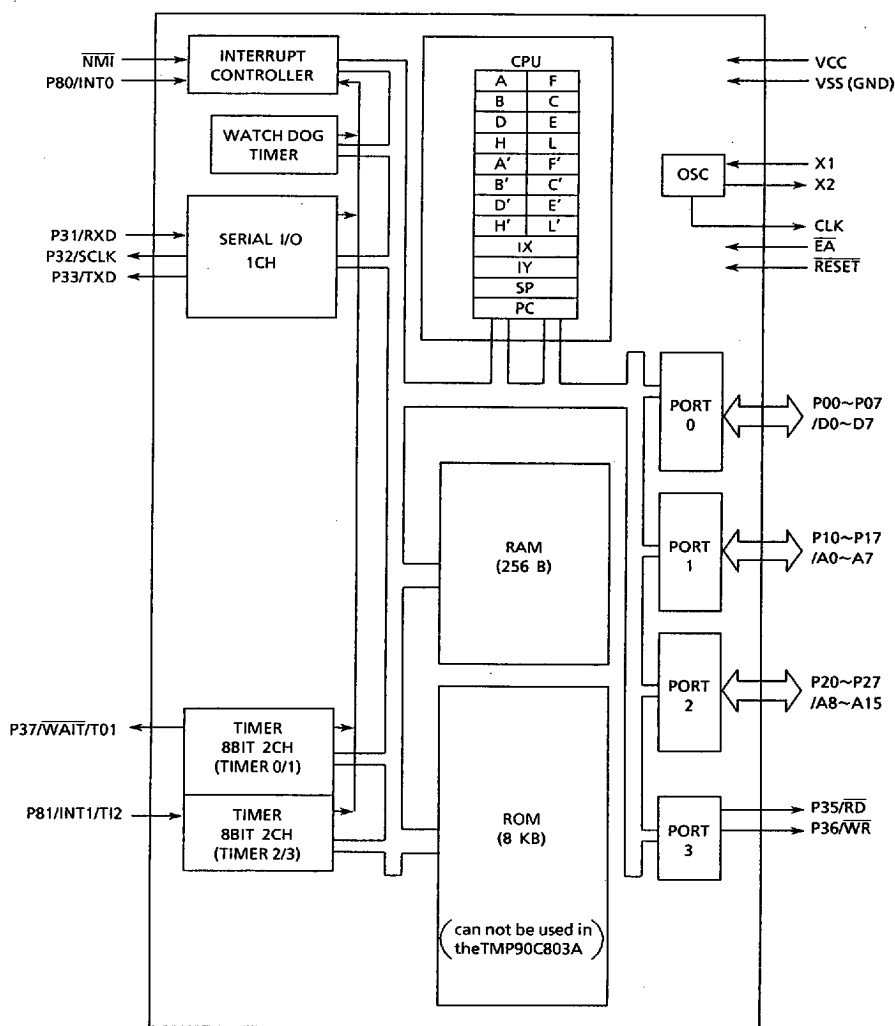
■ 9097249 0040715 963 ■
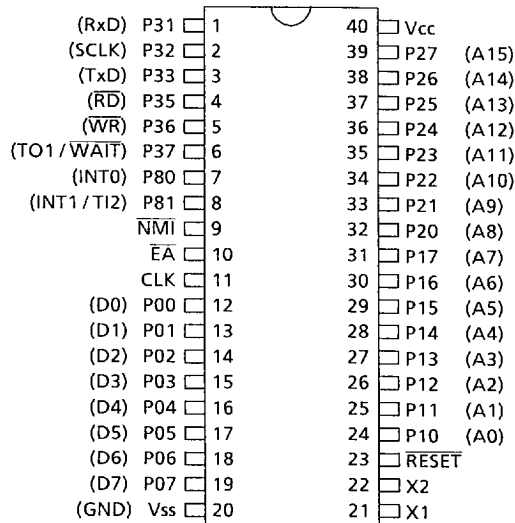
Figure 1 TMP90C802A Block Diagram

MCU90-260

9097249 0040716 8TT

2. PIN ASSIGNMENT AND FUNCTIONS

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

2.1 Pin Assignment

Figure 2.1 shows pin assignment of the TMP90C802A/803A.

Left (pin)	#		#	Right (pin)
(RxD) P31	1		40	Vcc
(SCLK) P32	2		39	P27 (A15)
(TxD) P33	3		38	P26 (A14)
($\overline{RD}$) P35	4		37	P25 (A13)
($\overline{WR}$) P36	5		36	P24 (A12)
(TO1/$\overline{WAIT}$) P37	6		35	P23 (A11)
(INT0) P80	7		34	P22 (A10)
(INT1/TI2) P81	8		33	P21 (A9)
$\overline{NMI}$	9		32	P20 (A8)
$\overline{EA}$	10		31	P17 (A7)
CLK	11		30	P16 (A6)
(D0) P00	12		29	P15 (A5)
(D1) P01	13		28	P14 (A4)
(D2) P02	14		27	P13 (A3)
(D3) P03	15		26	P12 (A2)
(D4) P04	16		25	P11 (A1)
(D5) P05	17		24	P10 (A0)
(D6) P06	18		23	$\overline{RESET}$
(D7) P07	19		22	X2
(GND) Vss	20		21	X1

240889

Figure 2.1 Pin Assignment

■ 9097249 0040717 736 ■

2.2 Pin Names and Functions

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

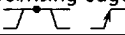### Table 2.2 Pin Names and Functions (1/2)

Pin Name	No. of pins	I/O 3 states	Function
P00~P07	8	I/O	Port 0: 8-bit I/O port that allows selection of input/output on byte basis
/D0~D7		3 states	Data bus: Also functions as 8-bit bidirectional data bus for external memory (For 803A, fixed to databus)
P10~P17	8	I/O	Port 1: 8-bit I/O port that allows selection on byte basis
/A0~A7		Output	Address bus: The lower 8 bits address bus for external memory (For 803A, fixed to address bus)
P20~P27	8	I/O	Port 2: 8-bit I/O port that allows selection on bit basis
/A8~A15		Output	Address bus: The upper 8 bits address bus for external memory (For 803A, fixed to address bus)
P31	1	Input	Port 31: 1-bit input port
/RxD			Receives Serial Data
P32	1	Output	Port 32: 1-bit output port
/SCLK			Serial clock output
P33	1	Output	Port 33: 1-bit output port
/TxD			Transmittes Serial Data
P35	1	Output	Port 35: 1-bit output port
/$\overline{RD}$			Read: Generates strobe signal for reading external memory
P36	1	Output	Port 36: 1-bit output port
/$\overline{WR}$			Write: Generates strobe signal for writing into external memory
P37	1	Input	Port 37: 1-bit input port
/$\overline{WAIT}$			Wait: Input pin for connecting slow speed memory or peripheral LSI
/TO1		Output	Timer output 1: Output of Timer 0 or 1
P80	1	Input	Port 80: 1-bit input port
/INT0			Interrupt request pin 0: interrupt request pin (Level/rising edge is programmable)
P81	1	Input	Port 81: 1-bit input port
/INT1			Interrupt request pin 1: interrupt request pin (Rising edge)
/TI2			Timer input 2: Counter input signal for Timer 2
$\overline{NMI}$	1	Input	Non-maskable interrupt request pin: Falling edge interrupt request pin

Table 2.2 Pin Names and Functions (2/2)

Pin Name	No. of pins	I/O 3 states	Function
CLK	1	Output	Clock output: Generates clock pulse at 1/4 frequency of clock oscillation. It is pulled up internally during resetting.
$\overline{EA}$	1	Input	External access: Connects with Vcc pin in the TMP90C802A using internal ROM, and with GND pin in the TMP90C803A with no internal ROM.
$\overline{RESET}$	1	Input	Reset : Initializes the TMP90C802A/803A (Built in pull-up resister)
X1/X2	2	Input/ Output	Pin for quartz crystal or ceramic resonator (1~12.5MHz)
Vcc	1		Power supply (+ 5V)
Vss (GND)	1		Ground (0V)

120889

9097249 0040719 509

3. OPERATION

This chapter describes the functions and the basic operations of the TMP90C802A in every block.

3.1 CPU

TMP90C802A includes a high performance 8-bit CPU. For the function of CPU, see the previous chapter "TLCS-90 CPU." This chapter explains exclusively the functions of the CPU of TMP90C802A which are not described in the "TLCS-90 CPU."

3.1.1 Reset

The basic timing of the reset operation is indicated in Figure 3.1. In order to reset the TMP90C802A, the $\overline{\text{RESET}}$ input must be maintained at the "0" level for at least ten system clock cycles (10 states : $2\mu s$ at 10MHz) within an operating voltage band and with as table oscillation. When a reset request is accepted, all I/O ports (Port 0 /data bus D0~D7, Port 1/address bus A0~A7, Port 2/address bus A8~A15) function as input ports (high impedance state). Output ports (P32, P33, P35 ($\overline{\text{RD}}$) and P36 ($\overline{\text{WR}}$)) and CLK turn to "1." Input ports remain unchanged.

The registers of the CPU also remain unchanged. Note, however, that the program counter "PC" and the interrupt enable flag IFF are cleared to "0". Register A shows an undefined status.

When the reset is cleared, the CPU starts executing instructions from the address 0000H.
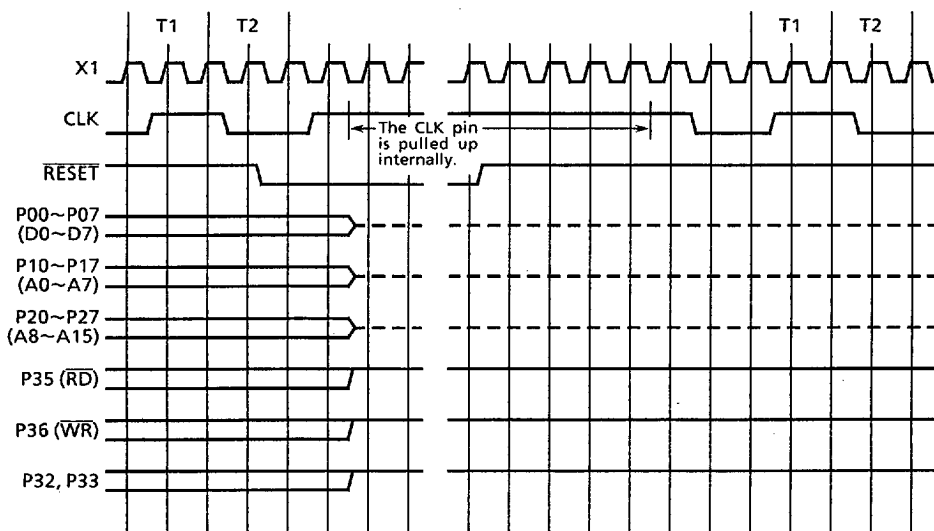
Figure 3.1 Reset Timing

3.1.2 EXF (Exchange Flag)

For TMP90C802A, "EXF", which is inverted when the command "EXX" is executed to transfer data between the main register and the auxiliary register, is allocated to the first bit of 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
	Resetting Value	1	0	0	0	0	0	Undefined	0
	Function	1 : WDT Enable	WDT Detecting time 00: 2^{14}/fc 01: 2^{16}/fc 10: 2^{18}/fc 11: 2^{20}/fc		Warming up time 0: 2^{14}/fc 1: 2^{16}/fc	Standby mode 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		Invert each time EXX instruction is executed	1:drive pins even in STOP mode

010491

3.1.3 Wait Control

For TMP90C802A, a wait control register (WAITC) is allocated to the 6th and 7th bits of memory address FFC7H.

P3CR (FFC7H)		7	6	5	4	3	2	1	0
	bit Symbol	WAITC1	WAITC0	RDE	ODE	TXDC	SCLK	RXDC	–
	Read/Write	R/W		R/W	R/W	R/W	R/W	R/W	–
	Resetting Value	0	0	0	0	0	0	0	–
	Function	Wait control 00 : 2state wait 01 : normal wait 10 : non wait 11 :Timer 0/1 Output		RD control 0 : $\overline{RD}$ for only external access 1 : Always $\overline{RD}$ Output	P33control 0 : CMOS 1 : Open drain	P33 0 : Out 1 : TxD	P32 0 : Out 1 : SCLK	P31 0 : In 1 : RxD	–

240890

9097249 0040721 167

3.2 Memory Map

The TMP90C802A supports a program or data memory of up to 64K bytes.

The program/data memory may be assigned to the address space from 0000H to FFFFH.

(1) Internal ROM

The TMP90C802A internally contains an 8Kbyte ROM. The address space 0000H~1FFFH is provided to the ROM. The CPU starts executing a program from 0000H by resetting.

The addresses 0010H~007FH in this internal ROM area are used for the entry area for the interrupt processing. The TMP90C803A dose not have a built-in ROM; therefore, the address space 0000H~1FFFH is used as external memory space.

(2) Internal RAM

The TMP90C802A also contains a 256byte RAM, which is allocated to the address space FEC0H~FFBFH. The CPU allows the access to whole RAM area (FF00H~FFBFH, 192 bytes) by a short operation code (opcode) in a "direct addressing mode".

The addresses from FF30H~FF7FH in this RAM area can be used as parameter area for micro DMA processing (and for any other purposes when the micro DMA function is not used).

(3) Internal I/O

The TMP90C802A provides a 48byte address space as an internal I/O area, whose addresses range from FFC0H~FFEFH. This I/O area can be accessed by the CPU using a short opcode in the "direct addressing mode".

Figure 3.2 is a memory map indicating the areas accessible by the CPU in the respective addressing mode.
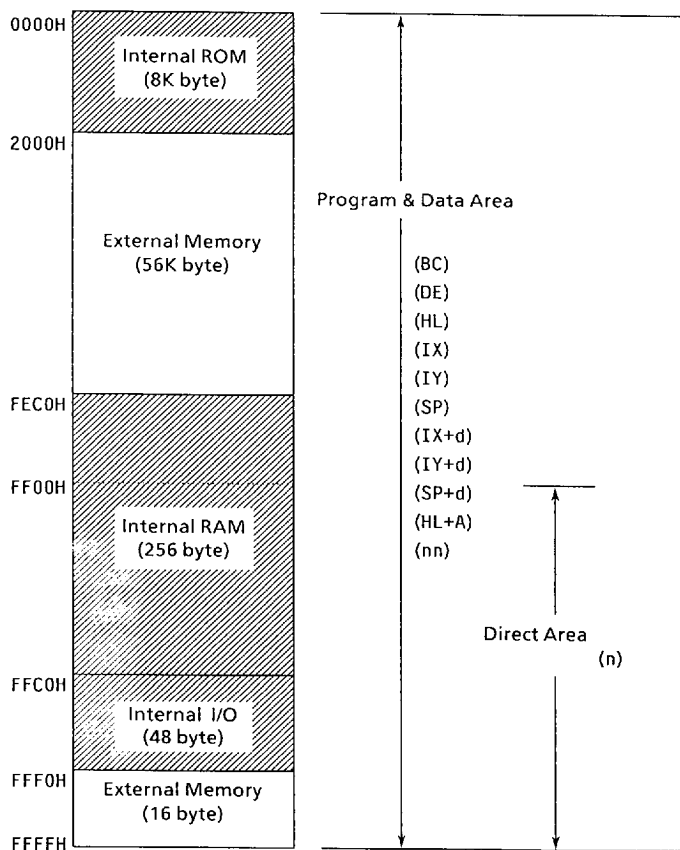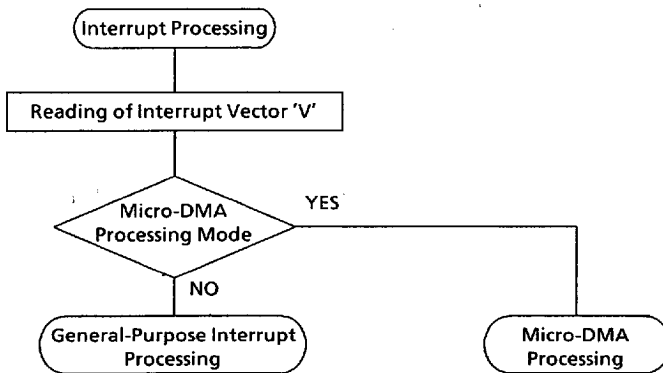
240889

Figure 3.2 Memory Map

9097249 0040723 T3T

3.3 Interrupt Functions

The TMP90C802A supports a general purpose interrupt processing mode and a micro DMA processing mode that enables automatic data transferred by the CPU for internal and external interrupt requests.

After the reset state is released, all interrupt requests are processed in the general purpose interrupt processing mode. However, they can be processed in the micro DMA processing mode by using a DMA enable register to be described later.

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

240889

Figure 3.3 (1)　Interrupt Response Flowchart

When an interrupt is requested, the source of the interrupt transmits the request to the CPU via an internal interrupt controller. The CPU starts the interrupt processing if it is a non-maskable or maskable interrupt requested in the EI state. However, a maskable interrupt requested in the DI state (IFF = '0') is ignored.

Having acknowledged an interrupt, the CPU reads out the interrupt vector from the internal interrupt controller to find out the interrupt source.

Then, the CPU checks if the interrupt requests the general purpose interrupt processing or the micro DMA processing, and proceeds to each processing.

As the reading of an interrupt vectors is performed in the internal operating cycles, the bus cycle results in dummy cycles.

3.3.1 General Purpose Interrupt Processing

A general purpose interrupt is processed as shown in Figure 3.3 (2).

The CPU stores the contents of the program counter PC and the register pair AF into the stack, and resets the interrupt enable flag IFF to "0" (disable interrupts). Then transfers the value of the interrupt vector "V" to the program counter, and the processing jumps to an interrupt processing program.

The overhead for the entire process from accepting an interrupt to jumping to an interrupt processing program is 20 states.
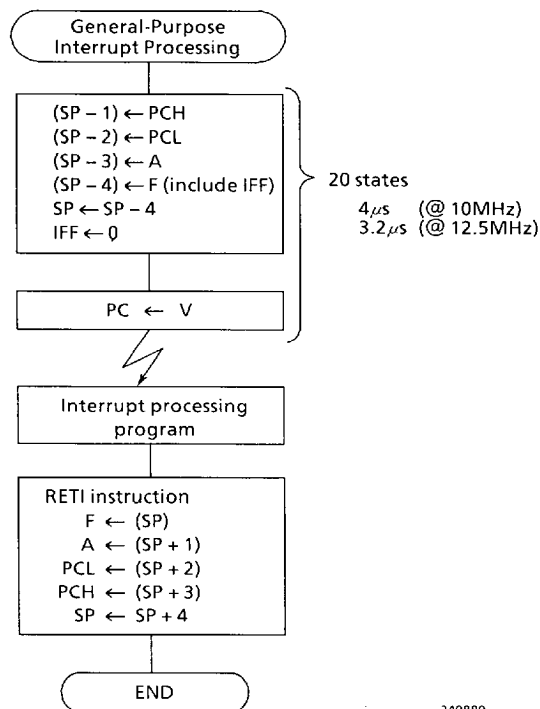
Figure 3.3 (2) General Purpose Interrupt Processing Flowchart

An interrupt (Maskable and Nonmaskable) processing program ends with a RETI instruction.

When this instruction is executed, the data previously stacked from the program counter PC and the register pair AF are restored.

After the CPU reads out the interrupt vector, the interrupt source acknowledges that the CPU accepts the request, and clears the request.

■ 9097249 0040725 802 ■

A non-maskable interrupt cannot be disabled by programming. A maskable interrupt, on the other hand, can be enabled or disabled by programming. An interrupt enable flip flop (IFF) is provided on the bit 5 of Register F in the CPU. The interrupt is enabled or disabled by setting IFF to "1" by the EI instruction or to "0" by the DI instruction, respectively. IFF is reset to "0" by the reset operation or the acceptance of any interrupt (including non-maskable interrupt). The EI instruction is executed after the subsequent instruction is executed.

Table 3.3 (1) lists the possible interrupt sources.

Table3.3 (1) Interrupt Sources

Priority order	Type	Interrupt source	Vector value ÷ 8	Vector value	Start address of general purpose interrupt processing	Start address of Micro DMA processing parameter
1	Non maskable	SWI instruction	02H	10H	0010H	–
2		NMI (Input from NMI pin)	03H	18H	0018H	–
3		INTWD (watchdog)	04H	20H	0020H	–
4	Maskable	INT0 (External input 0)	05H	28H	0028H	–
5		INTT0 (Timer 0)	06H	30H	0030H	FF30H
6		INTT1 (Timer 1)	07H	38H	0038H	FF38H
7		INTT2 (Timer 2)	08H	40H	0040H	–
8		INTT3 (Timer 3)	09H	48H	0048H	–
9		INT1 (External input 1)	0BH	58H	0058H	–
10		INTRX (End of serial receiving)	0EH	70H	0070H	FF70H
11		INTTX (End of serial transmission)	0FH	78H	0078H	FF78H

240889

The "priority order" in the table shows the order of the interrupt source to be acknowledged by the CPU when more than interrupt are requested at one time.

If interrupt of fourth and fifth orders are requested simultaneously, for example, an interrupt of the "5th" priority is acknowledged after a "4th" priority interrupt processing has been completed by a RETI instruction. However, a lower priority interrupt can be acknowledged immediately by executing an EI instruction in a program that processes a higher priority interrupt.

The internal interrupt controller merely determines the priority of the sources of interrupts to be acknowledged by the CPU when more than one interrupt are requested at a time. It is, therefore, unable to compare the priority of interrupt being executed with the one being requested.

3.3.2 Micro DMA Processing

Figure 3.3 (3) is a flowchart of the micro DMA processing. Parameters (addresses of source and destination, and transfer mode) for the data transfer between memories are loaded by the CPU from an address modified by an interrupt vector value. After the data transfer between memories according to these parameter, these 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 transfer becomes "0", the CPU proceeds to the general purpose interrupt handling described in the previous chapter.

```
            ┌────────────────────────────┐
            │   Micro DMA Processing     │
            └────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │ Loading of Parameters                   │
   │   DST ← (FF00H + V + 1) W               │
   │   SRC ← (FF00H + V + 3) W               │
   │   CMD ← (FF00H + V + 5)                 │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │ Data Transfer between Memories          │
   │   (DST) ← (SRC)                         │
   │   DST ← DST + 0 / 1 / 2                 │
   │   SRC ← SRC ± 0 / 1 / 2                 │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │ Saving of Update Parameters             │
   │   (FF00H + V + 3) W ← SRC               │
   │   (FF00H + V + 1) W ← DST               │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐      DST    : Destination
   │ Decrement of No. of Transfer            │      SRC    : Source
   │   COUNT ← (FF00H + V)                   │      CMD    : Command (Transfer mode)
   │   COUNT ← COUNT – 1                     │      COUNT  : Count (Number of transfer)
   │   (FF00H + V) ← COUNT                   │
   └─────────────────────────────────────────┘
                         │        YES
                   < COUNT = 0 >─────────┐
                         │ NO            │
                   ┌───────────┐   ┌───────────────────────┐
                   │    END    │   │ General-purpose       │
                   └───────────┘   │ Interrupt Processing  │
                                   └───────────────────────┘
```

240889

Figure 3.3 (3) Micro DMA Processing Flowchart

■ 9097249 0040727 685 ■

The micro DMA processing is performed by using only hardware to process interrupts mostly completed by simple data transfer. The use of hardware allows the micro DMA processing to handle the interrupt in a higher speed than the conventional methods using software. 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.
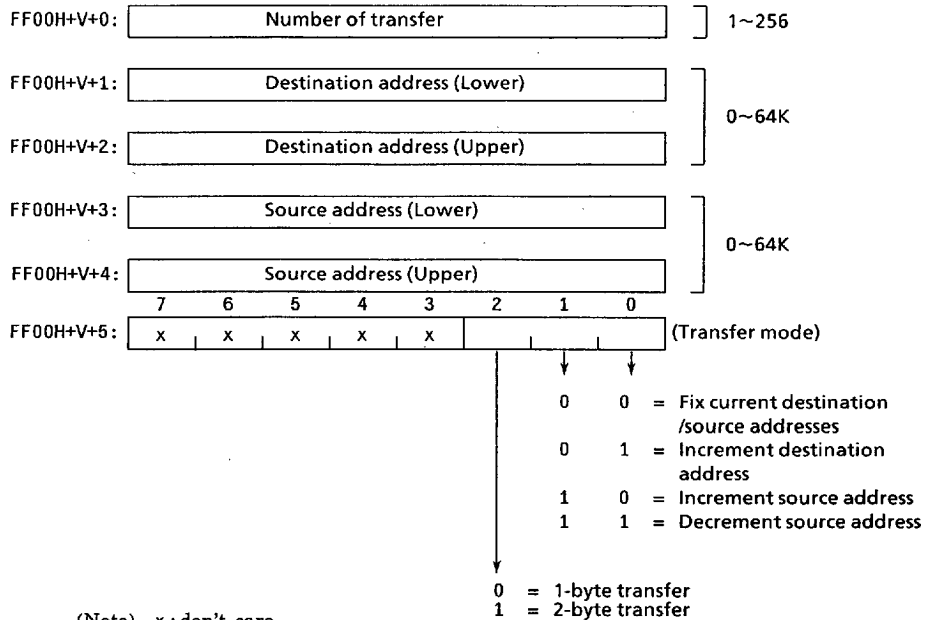
Figure 3.3 (4) Parameters for Micro DMA Processing

Parameters for the micro DMA processing are located in the internal RAM area (See Table 3.3 (1) Interrupt Sources). The start address of each parameter is "FF00H + interrupt vector value", from which a six bytes' space is used for the parameter. This space can be used for any other memory purposes if the micro DMA processing is not used.

The parameters normally consist of the number of transfer, addresses of destination and source, and transfer mode. The number of transfer indicates the number of data transfer accepted in the micro DMA processing.

The amount of data transferred by a single micro DMA processing is limited to one or two bytes. Both the destination and source addresses are specified by 2-byte data. The address space available for the micro DMA processing ranges from 0000H to FFFFH.

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

MCU90-272

■ 9097249 0040728 511 ■

Table 3.3 (2) shows the relation between the transfer mode and the result of updating 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 source/ destination 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 source/ destination 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

240889

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 results are different as shown in the table 3.3 (2).

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   1,(0FFE6H)           ;   Enable an interrupt for serial data receiving.
SET   1,(0FFE8H)           ;   Set the micro DMA processing mode for the interrupt.
LD    (0FF70H),7           ;   Set the number of transfer = 7
LDW   (0FF71H),0FF00H      ;   Set FF00H for the destination address.
LDW   (0FF73H),0FFEBH      ;   Set FFEBH for the source (serial receiving buffer) address.
LD    (0FF75H),1           ;   Set the transfer mode (1-byte transfer : Increment destination
                                address).
EI
:
:
ORG      0070H
```

┌─────────────────────────────┐
│ Interrupt processing program │
│ after serial data receiving │
└─────────────────────────────┘

240889

```
RETI
```

Figure 3.3 (5)   Example of Micro DMA Processing

■ 9097249 0040729 458 ■

The bus operation in the general purpose interrupt processing and the micro DMA processing is included in "Table 1.4 (2) Bus Operation for Executing Instructions" in the previous section."

The micro DMA processing time (when the number of transfer is not decremented to 0) is 46 states (9.2μs at 10MHz oscillation frequency) without regard to the 1-byte/2-byte transfer mode.

Figure 3.3 (6) shows the interrupt processing flowchart.
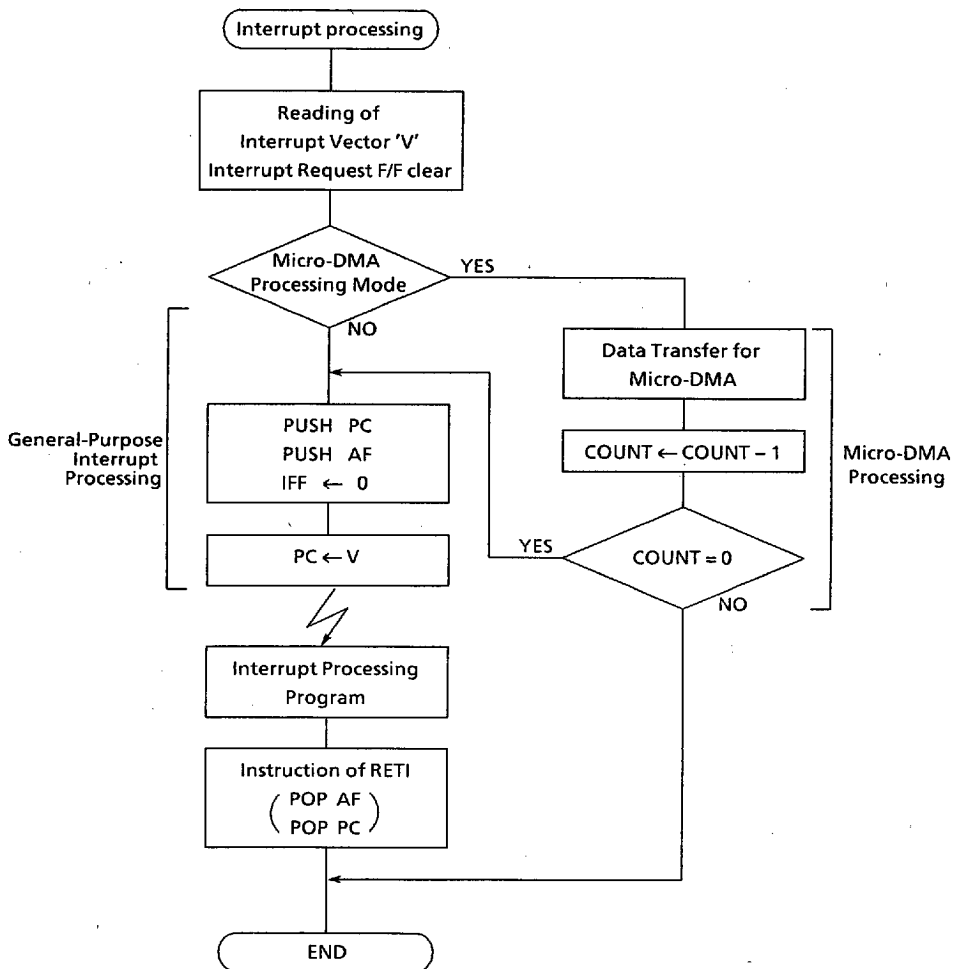
Figure 3.3 (6)   Interrupt Processing Flowchart

240889

■ 9097249 0040730 17T ■
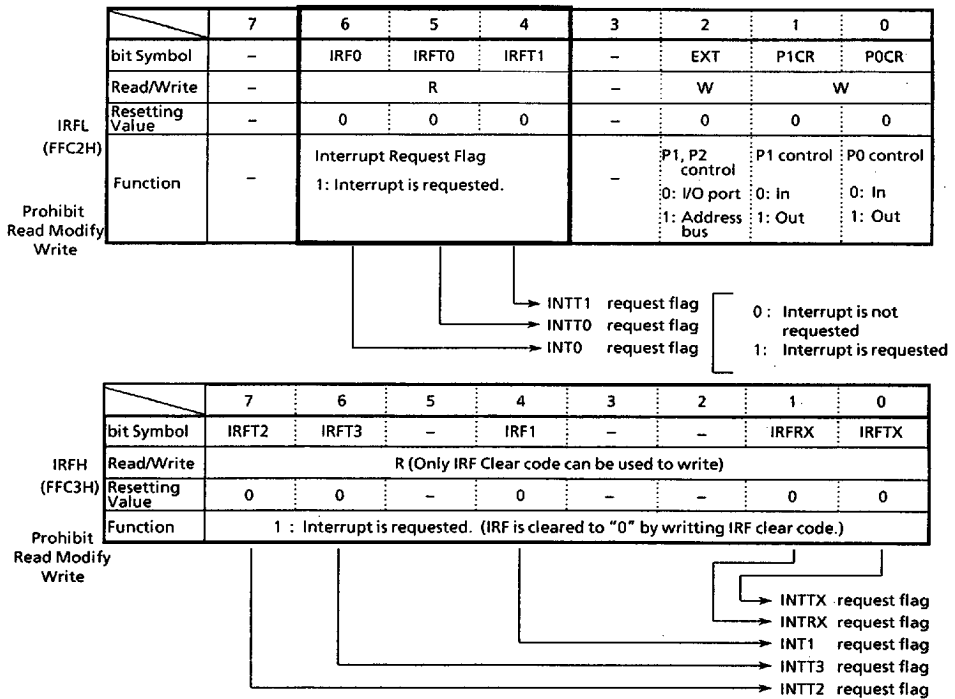
3.3.3 Interrupt Controller

Figure 3.3 (8) outlines the interrupt circuit. The left side of this figure represents an interrupt controller, and the right side comprises the CPU's interrupt request signal circuit and HALT release signal circuit.

The interrupt controller has Interrupt Request Flip-flops, interrupt enable flags, and micro DMA enable flags (4 channels) for each interrupt channel (11 channels). The Interrupt Request Flip-flops serve to latch interrupt requests from peripherals. Each flip-flop is reset to "0" when a reset or interrupt is acknowledged by the CPU and the vector of the interrupt channel is read into the CPU, or when the CPU executes an instruction that clears a request to interrupt that channel (write "vector divided by 8" into the memory address FFC3H). For example, by executing.

LD (0FFC3H), 38H/8,

The Interrupt Request Flip-flops for the interrupt channel "INTT1" whose vector is 38H is reset to "0".

The status of an Interrupt Request Flip-flops is found out by reading the memory address FFC2H or FFC3H. "0" denotes there is no interrupt request, and "1" denotes that an interrupt is requested. Figure 3.3 (7) illustrates the bit configuration indicating the status of Interrupt Request Flip-flops.

		7	6	5	4	3	2	1	0
	bit Symbol	–	IRF0	IRFT0	IRFT1	–	EXT	P1CR	P0CR
	Read/Write	–	R			–	W	W	
IRFL	Resetting Value	–	0	0	0	–	0	0	0
(FFC2H)	Function	–	Interrupt Request Flag 1: Interrupt is requested.			–	P1, P2 control 0: I/O port 1: Address bus	P1 control 0: In 1: Out	P0 control 0: In 1: Out

Prohibit Read Modify Write

INTT1 request flag
INTT0 request flag
INT0 request flag

0 : Interrupt is not requested
1 : Interrupt is requested

		7	6	5	4	3	2	1	0
	bit Symbol	IRFT2	IRFT3	–	IRF1	–	–	IRFRX	IRFTX
IRFH	Read/Write	R (Only IRF Clear code can be used to write)							
(FFC3H)	Resetting Value	0	0	–	0	–	–	0	0
	Function	1 : Interrupt is requested. (IRF is cleared to "0" by writting IRF clear code.)							

Prohibit Read Modify Write

INTTX request flag
INTRX request flag
INT1 request flag
INTT3 request flag
INTT2 request flag

(Caution) Writing "vector divided by 8" into the memory address FFC3H clears the Flip-Flop for the specified interrupt request.

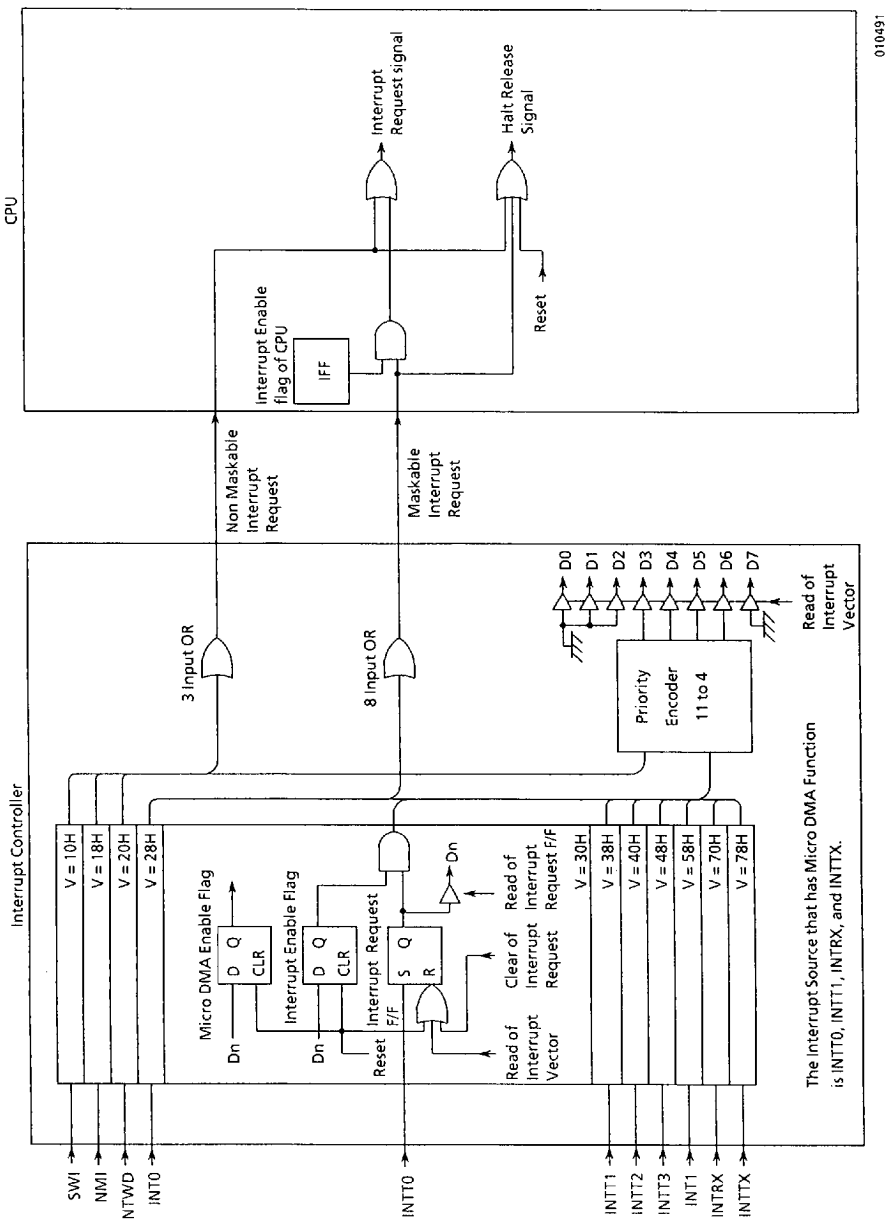### Figure 3.3 (7) Configration of Interrupt Request Flip-flops

■ 9097249 0040732 T42 ■

Figure 3.3 (8)   Block Diagram of Interrupt Controller

9097249 0040733 989

The interrupt enable flags provided for all interrupt request channels are assigned to the memory address FFE6H or FFE7H. Setting any of these flags to "1" enables an interrupt of the respective channel. These flags are initialized to "0" by resetting.

Clear the interrupt enable flag in the DI status.

The micro DMA enable flag also provided for each interrupt request channel is assigned to the memory address FFE7H or FFE8H. The interrupt processing for each channel is placed in the micro DMA processing mode by setting this flag to "1". This flag is initialized to "0" (general purpose interrupt processing mode) by resetting.

Figure 3.3 (9) shows the bit configuration of the interrupt enable flags and micro DMA enable flags.

The external interrupt functions are shown below.

Interrupt	Common terminal	Mode		setting
$\overline{\text{NMI}}$	—		Falling edge	———
INT0	P80		Level	P8CR<EDGE> = 0
			Rising edge	P8CR<EDGE> = 1
INT1	P81		Rising edge	———

FOR the pulse width for the external interrupts, see selection 4.7 "Interrupt Operation".

Attention should be paid to the following two modes having special circuits :

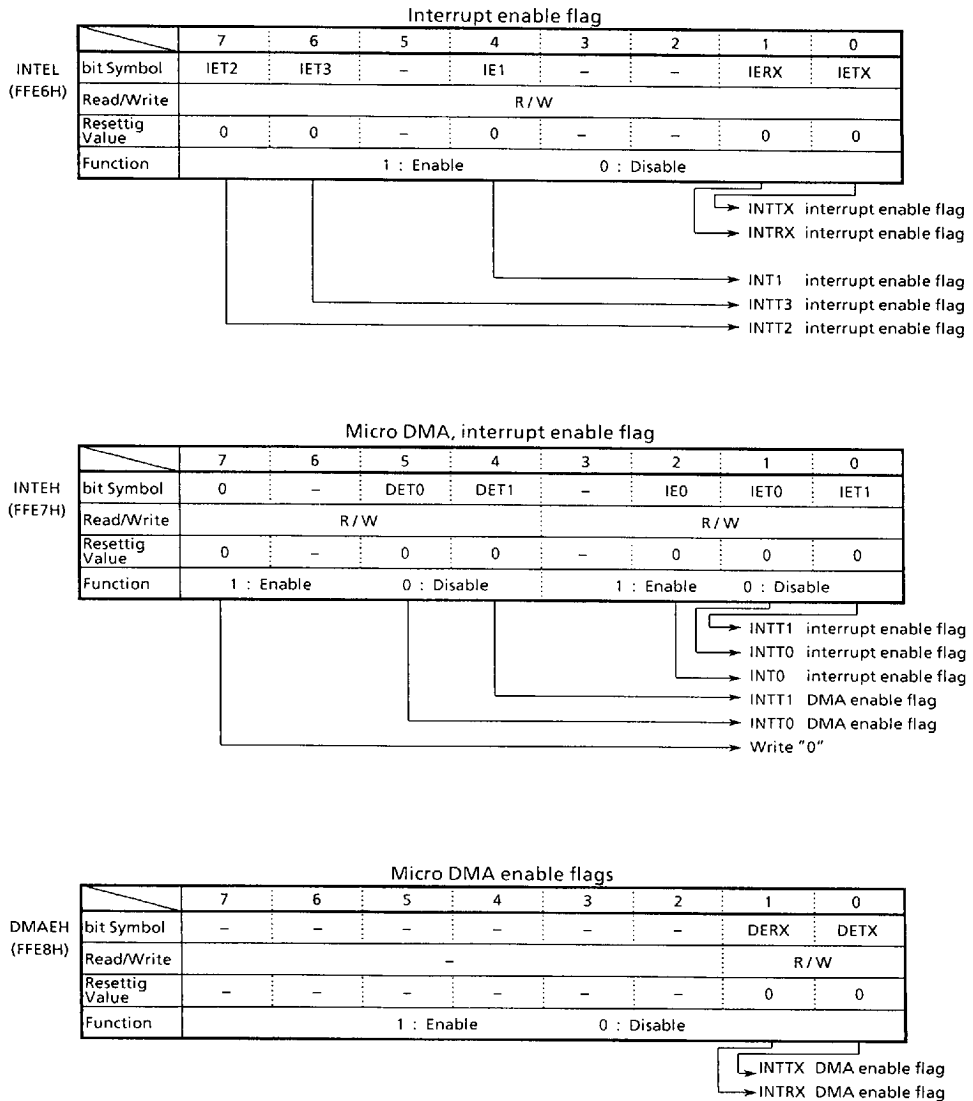| INT0 Level mode	IF INT0 is not an edge-based interrupt, the function of Interrupt Request Flip-flop is canceled. Therefore the interrupt request signal must be held until the interrupt request is acknowledged by the CPU. A change in the mode (edge to level) automatically clears the interrupt request flag. When the CPU has been put in the interrupt response sequence by changing INT0 from "0" to "1", it is necessary to leave INT0 at "1" until the second bus cycle of the interrupt response sequence is completed. Also, "1" must always be held until HALT is cleared when using the INT0 level mode to clear HALT. (Use care to prevent noise changing "1" back to "0".) When switching from the level mode to the edge mode, the interrupt request flag set in the level mode is not cleared; therefore, use the following sequence to clear the interrupt request flag. DI LD (0FFD1H), 01H: switch from level to edge LD (0FFC3H), 05H: clear interrupt request flag EI
INTRX	The Interrupt Request Flip-flop is cleared only by resetting or reading the serial channel receiving buffer, and not by an instruction.

240889

Interrupt enable flag

		7	6	5	4	3	2	1	0
INTEL (FFE6H)	bit Symbol	IET2	IET3	–	IE1	–	–	IERX	IETX
	Read/Write	R / W							
	Resettig Value	0	0	–	0	–	–	0	0
	Function	1 : Enable			0 : Disable				

INTTX interrupt enable flag
INTRX interrupt enable flag
INT1   interrupt enable flag
INTT3 interrupt enable flag
INTT2 interrupt enable flag

Micro DMA, interrupt enable flag

		7	6	5	4	3	2	1	0
INTEH (FFE7H)	bit Symbol	0	–	DET0	DET1	–	IE0	IET0	IET1
	Read/Write	R / W				R / W			
	Resettig Value	0	–	0	0	–	0	0	0
	Function	1 : Enable		0 : Disable		1 : Enable		0 : Disable	

INTT1 interrupt enable flag
INTT0 interrupt enable flag
INT0   interrupt enable flag
INTT1 DMA enable flag
INTT0 DMA enable flag
Write "0"

Micro DMA enable flags

		7	6	5	4	3	2	1	0
DMAEH (FFE8H)	bit Symbol	–	–	–	–	–	–	DERX	DETX
	Read/Write	–						R / W	
	Resettig Value	–	–	–	–	–	–	0	0
	Function	1 : Enable				0 : Disable			

INTTX DMA enable flag
INTRX DMA enable flag

Figure 3.3 (9) Interrupt/Micro DMA Enable Flags

■ 9097249 0040735 751 ■

3.4 Standby Function

When a HALT instruction is executed, the TMP90C802 selects one of the following modes as determined by the halt mode set register:

(1)RUN   :   Suspends only the CPU operation. The power consumption remains unchanged.

(2)IDLE1 :   Suspends all internal circuits except the internal oscillator. In this mode, the power consumption is less than 1/10 of that in the normal operation.

(3)IDLE2 :   Operate only the internal oscillator and specific internal I/O devices. The power consumption is less than 1/3 of that in the normal operation.

(4)STOP :   Suspends all internal circuits including the internal oscillator. In this mode, the power consumption is considerably reduced.

The HALT mode set register WDMOD <HALTM1, 0> is assigned to the bits 2 and 3 of the memory address FFD2H in the internal I/O register area (other bits are used to control other functions). The register is reset to "00" (RUN mode) by resetting.

These HALT state can be released by resetting or requesting an interrupt. The methods for releasing the HALT status are shown in Table 3.4 (2).

Either a non-maskable or maskable interrupt with EI (enable interrupt) condition is acknowledged and interrupt processing is processed. A maskable interrupt with DI (disable interrupt) condition is also acknowledged and CPU starts executing an instruction that follows the HALT instruction, but the interrupt request flag is held at "1".

But if interrupt request occur before MPU practices "HALT" command in the state of DI and it latches interrupt request flag, it causes to release HALT state and to state will be released as soon after MPU practices "HALT" command. (MPU doesn't HALT state.)

Therefore clear interrupt request flag or disable interrupt enable flag before MPU practices "HALT" command.

ex) MPU becomes "STOP" mode in the state of DI and release it by INT0 interupt.
     But "built-in I/O" used only "Timer 0".

```
DI                          ;
SET     2, (INTEH)          ; INT0 interrupt enable
RES     1, (INTEH)          ; INTT0 interrupt disable
LD      (WDMOD), 04H        ; STOP mode
HALT
```

```
After release "HALT",
Practice Program
```

When the halt status is released by a reset, the status in effect before entering the halt status (including built-in RAM) is held. The RAM contents may not be held, however, if the HALT instruction is executed within the built-in RAM.

WDMOD	bit Symbol	7	6	5	4	3	2	1	0
(FFD2H)		WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
	Read/Write	R/W	R/W		R/W	R/W		R	R/W
	Resetting Value	1	0	0	0	0	0	Undefined	0
	Function	1 : WDT Enable	WDT Detecting time 00: 2^{14}/fc 01: 2^{16}/fc 10: 2^{18}/fc 11: 2^{20}/fc		Warming up time 0: 2^{14}/fc 1: 2^{16}/fc	Standby mode 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		Invert each time EXX instruction is executed	1: to drive pin in STOP mode.

Explained in "3.8 Watchdog Timer"

Exchange flag
Explained in "3.1.2 Registers"

Explained in "3.4.4 STOP mode"

230890

Figure 3.4 (1)   HALT Mode Set Register

3.4.1 RUN Mode

Figure 3.4 (2) shows the timing for releasing the HALT state by interrupts in the RUN/ IDLE 2 mode.

In the RUN mode, the system clock in the MCU continues to operate even after a 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, an interrupt request is sampled with the rising edge of the "CLK" signal.
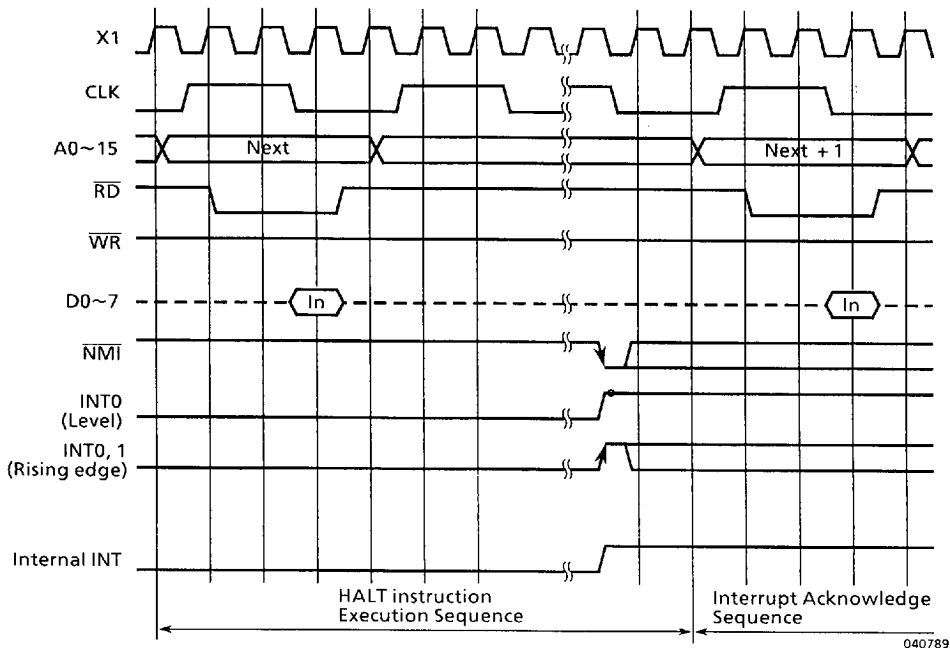

Figure 3.4 (2)   Timing Chart for Releasing the HALT State by Interrupts in RUN/IDLE 2 Modes

040789

MCU90-281

■ 9097249 0040737 524 ■

3.4.2 IDLE 1 Mode

Figure 3.4 (3) illustrates the timing for releasing the HALT state by interrupts in the IDLE 1 mode.

In the IDLE 1 mode, only the internal oscillator and the watchdog timer operate. The system clock in the MCU stops, and the CLK signal is fixed at the "1" level.

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

Note : An interrupt requested by the watchdog timer is prohibited through the HALT period in this mode.
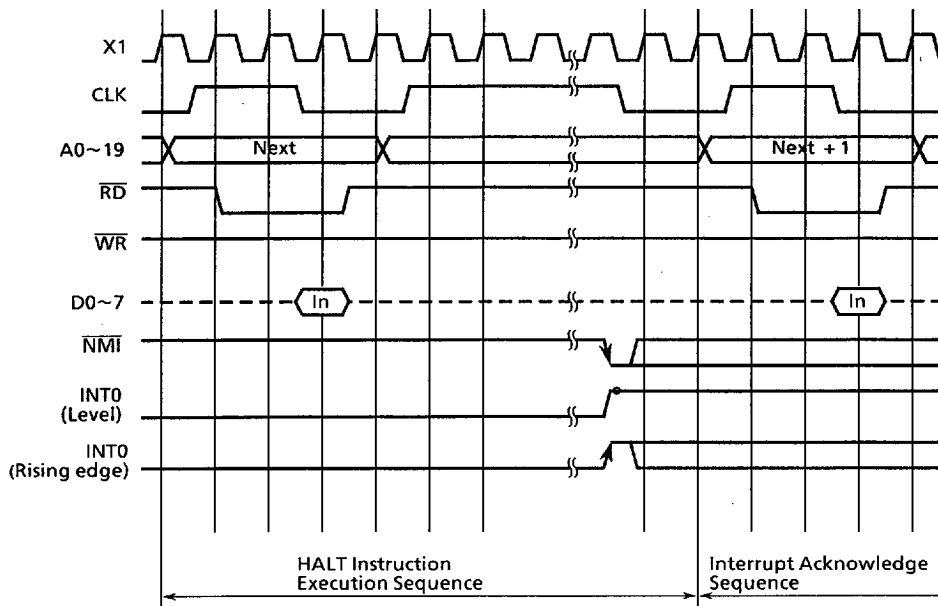
150689

Figure 3.4 (3)   Timing Chart of HALT Released by
                  Interrupts in IDLE1 Mode

9097249 0040738 460

3.4.3 IDLE 2 Mode

Figure 3.4 (2) shows the timing of HALT release caused by interrupts in the RUN/IDLE 2 mode.

In the IDLE 2 mode, the HALT state is released by an interrupt with the same timing as in the RUN mode, except the internal operation of the MCU. In the RUN mode, only the CPU stops executing the current instruction, and the system clock is supplied to all internal devices. In the IDLE 2 mode, however, the system clock is supplied to only specific internal I/O devices. As a result, the HALT state in the IDLE 2 mode requires only a 1/3 of the power consumed in the RUN mode. In the IDLE 2 mode, the system clock is supplied to the following I/O devices:

- 8-bit timer
- Serial interface
- Watchdog timer

3.4.4 STOP Mode

Figure 3.4 (4) is a timing chart for releasing the HALT state by interrupts in the STOP mode.

The STOP mode is selected to stop all internal circuits including the internal oscillator. In this mode, all pins except special ones are put in the high-impedance state, independent of the internal operation of the MCU. Table 3.4 (1) summarizes the state of these pins in the STOP mode. Note, however, that the pre-halt state (The status prior to execution of HALT instruction) of all output pins can be retained by setting the internal I/O register WDMOD <DRIVE> (Drive enable: Bit 0 of memory address FFD2H) to "1". The content of this register is initialized to "0" by resetting.

When the CPU accepts an interrupt request, the internal oscillator is restarted immediately. However, to get the stabilized oscillation, the system clock starts its output after the time set by the warming up counter WDMOD <WARM> (Warming up: Bit 4 of memory address FFD2H). A warming-up time of either the clock oscillation time $\times 2^{14}$ or $\times 2^{16}$ can be set by setting this bit to either "0' or "1". This bit is initialized to "0" by resetting.
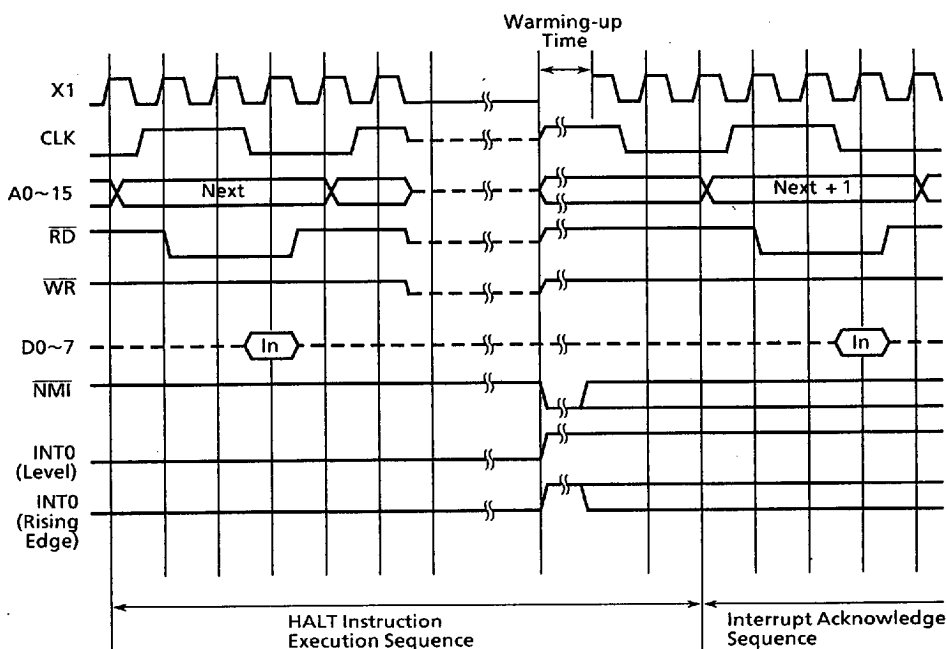
Figure 3.4 (4)   Timing Chart of HALT Released by
                  Interrupt in STOP Mode

MCU90-284

■ 9097249 0040740 019 ■

The internal oscillator can be also restarted by the input of the $\overline{\text{RESET}}$ signal at "0" to the CPU. In the Reset restart mode, however, the warming-up counter remains inactive in order to get the quick response of MCU when the power is turned on (Power on Reset). As a result, the normal clock operation may not be performed due to the unstable clock supplied immediately after restarting the internal oscillator. To avoid this, it is necessary to keep the $\overline{\text{RESET}}$ signal at "0" long enough to release the HALT state in the STOP mode.

Table 3.4 (1)   State of Pins in STOP Mode

	IN/OUT	DRVE = 0	DRVE = 1
P0	Input mode Output mode	—— OUT	—— OUT
P1	Input mode Output mode	—— ——	IN OUT
P2	Input mode Output mode	—— ——	IN OUT
P3	Input pin Output pin	—— ——	IN OUT
P80 (INT0) P81 (INT1)	Input pin Input pin	IN ——	IN IN *
$\overline{\text{NMI}}$ CLK $\overline{\text{RESET}}$ X1 X2	Input pin Output pin Input pin Input pin Output pin	IN IN —— " 1 "	IN " 1 " IN —— " 1 "

240889

* : Intermediate bias is still applied to this pin in the zero cross detect mode.

—— : Indicates that input mode/input pin cannot be used for input and that the output mode/output pin have been set to high impedance.

IN : The input enable status.

IN : The input gate is operating. Fix the input voltage at either "0" or "1" to prevent the pin floating.

OUT : The output status.

It is necessary to leave INT0 at "1" until the second bus cycle of the interrupt response sequence is completed, when the STOP mode is released by the level mode of INT0.

9097249 0040741 T55

Table 3.4 (2)   I/O Operation During Halt and How to Release the Halt Command

Halt mode			RUN	IDLE2	IDLE1	STOP
WDMOD <HALTM1,0>			00	11	10	01
Operating block	CPU		Halt			
	I/O port		Keeps the state when the halt command was executed.			See Table 3.4 (1)
	8 bit timer		Operation		Halt	
	Serial interface					
	Watchdog timer					
	Interrupt controller					
Halt releasing source	Interrupt	NMI	○	○	○	○
		INTWD	○	○	—	—
		INT0	○	○	○	○
		INTT0	○	○	—	—
		INTT1	○	○	—	—
		INTT2	○	○	—	—
		INTT3	○	○	—	—
		INT1	○	○	—	—
		INTRX	○	○	—	—
		INTTX	○	○	—	—
	Reset		○	○	○	○

010491

○ : Can be used to release the halt commnad.
— : Cannot be used to release the halt command.

3.5 Function of Ports

The TMP90C802A contains total 32 pins (TMP90C803A: 6pins) input/output ports. These ports function not only for the general-purpose I/O but also for the input/output of the internal CPU and I/O. Table 3.5 describes the functions of these ports.

Table 3.5 Functions of Ports

Port name	Pin name	No. of pins	Direction	Direction set unit	Resetting value	Pin name for internal function
Port 0	P00~P07	8	I/O	Byte	Input	D0~D7
Port 1	P10~P17	8	I/O	Byte	Input	A0~A7
Port 2	P20~P27	8	I/O	Bit	Input	A8~A15
Port 3	P31	1	Input	—	Input	RxD
	P32	1	Output	—	Output	SCLK
	P33	1	Output	—	Output	TxD
	P35	1	Output	—	Output	$\overline{RD}$
	P36	1	Output	—	Output	$\overline{WR}$
	P37	1	Input	—	Input	$\overline{WAIT}$/TO1
Port 8	P80	1	Input	—	Input	INT0
	P81	1	Input	—	Input	INT1/TI2

230890

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.
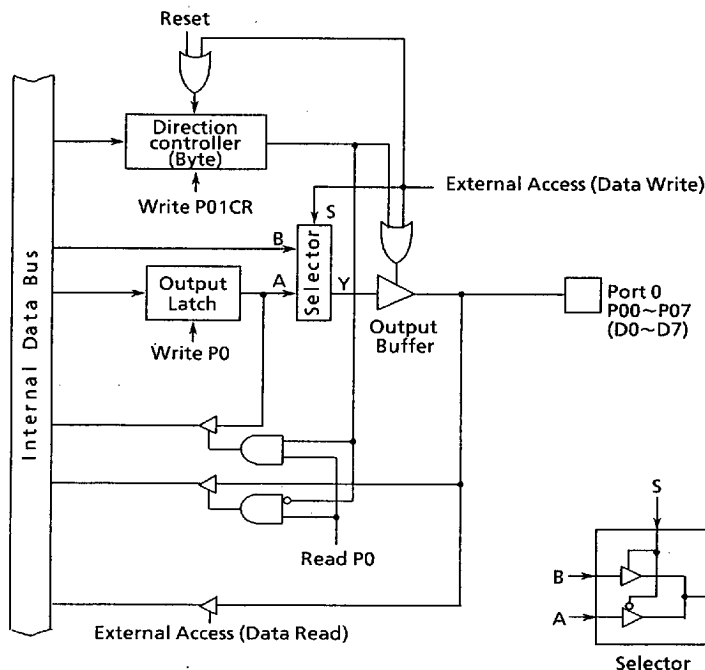
The TMP90C803A functions in the same way as the TMP90C802A except:

- Port 0 always functions as data bus (D0 to D7)
- Port 1 always functions as address bus (A0 to A7)
- Port 2 always functions as address bus (A8to A15)
- P35 and P36 of always function as $\overline{RD}$ and $\overline{WR}$ pins, respectively

3.5.1 Port 0 (P00~P07)

Port 0 is an 8-bit general-purpose I/O port P0 whose I/O function is specified by the control register P01CR <P0C> in byte. By resetting all bits of the control register are initialized to "0", whereby Port 0 turns to the input mode, and the contents of the output latch register are undefined.

In addition to the general-purpose I/O port function, it functions as a data bus (D0~D7). Access of an external memory makes it automatically function as a data bus and <P0C> are cleared to "0".

Reset

Direction controller (Byte)

Write P01CR

External Access (Data Write)

Output Latch

Write P0

Output Buffer

Port 0 P00~P07 (D0~D7)

Read P0

External Access (Data Read)

Selector

230890

Figure 3.5 (1) Port 0

9097249 0040744 764

3.5.2 Port 1 (P10~P17)

Port 1 is an 8-bit general-purpose I/O port P1 whose I/O function is specified by the control register P01CR <P1C> in byte. All bits of the output latch and the control register are initialized to "0" by resetting, whereby Port 1 is put in the input mode.

In addition to the general-purpose I/O port function, it functions as an address bus (A0~A7). The address bus function can be selected by setting only the external extension control register P01CR <EXT> to "1" regardless of the status of the above control register <P1C>. The <EXT> register is reset to "0" whereby Port 1 and Port 2 turn to the general-purpose I/O mode.

190990

Figure 3.5 (2) Port 1

■ 9097249 0040745 6T0 ■

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				

Port 1 Register

P1 (FFC1H)		7	6	5	4	3	2	1	0
	bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
	Read/Write				R / W				
	Resetting Value				Input Mode				

Port 0,1 Control Register

P01CR (FFC2H)		7	6	5	4	3	2	1	0
	bit Symbol	–	IRF0	IRFT0	IRFT1	–	EXT	P1C	P0C
	Read/Write	–		R		–	W	W	
	Resetting Value	–	0	0	0	–	0	0	0
	Function	–	Interrupt Request Flag 1: Interrupt is requested			–	P1, P2 control 0: I/O port 1: Address bus	P1 control 0: In 1: Out	P0 control 0: In 1: Out

Prohibit Read Modify Write.

Explained in "3.3.3 Interrupt controller"

Set I/O of Port 0

0	Input
1	Output

Set I/O of Port 1

0	Input
1	Output

Set general-purpose port/address bus of Ports 1 and 2

0	General-purpose port
1	Address bus

230890

Figure 3.5 (3)   Registers for Port 0 and 1
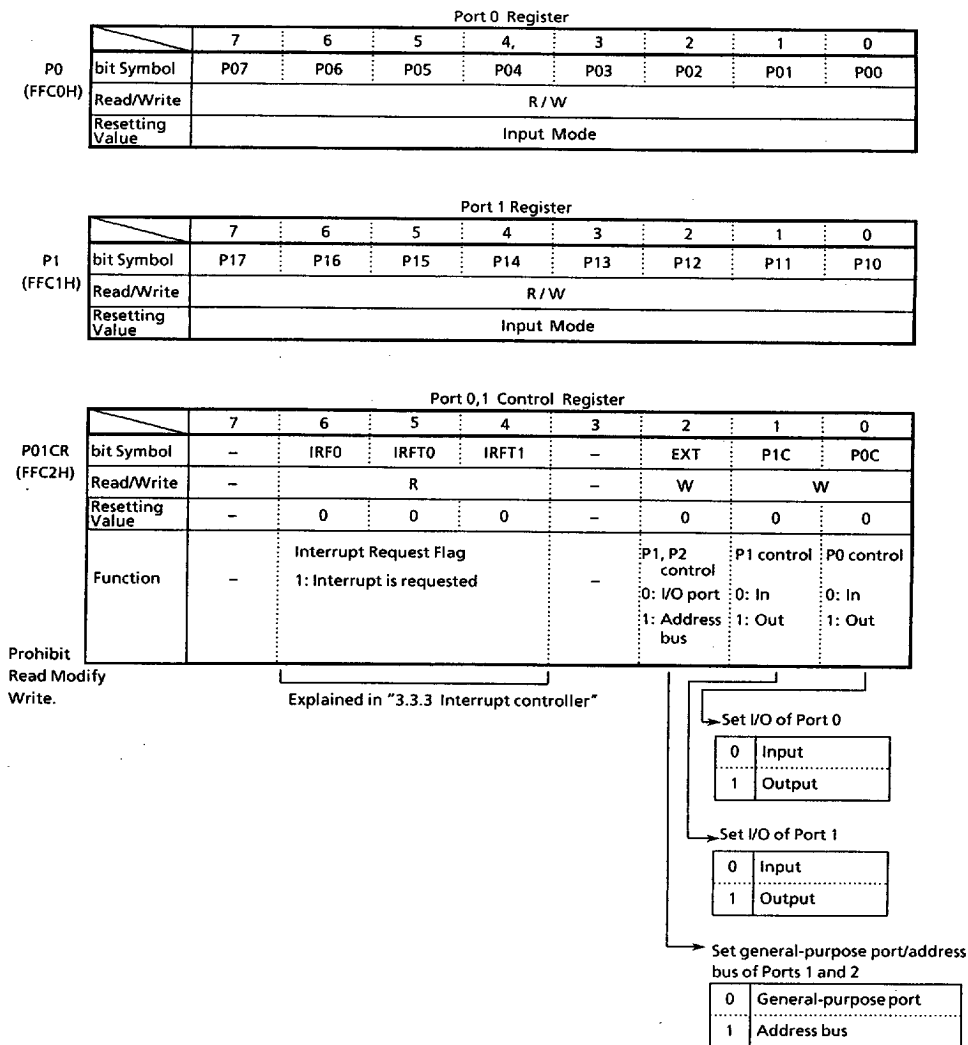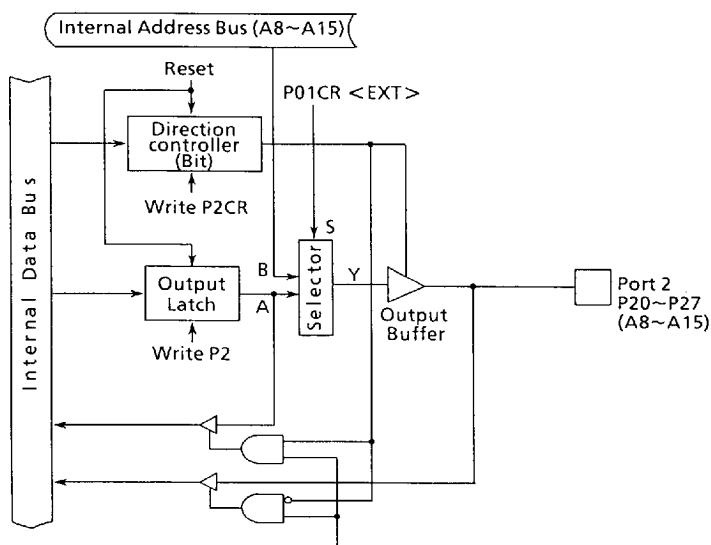
9097249 0040746 537
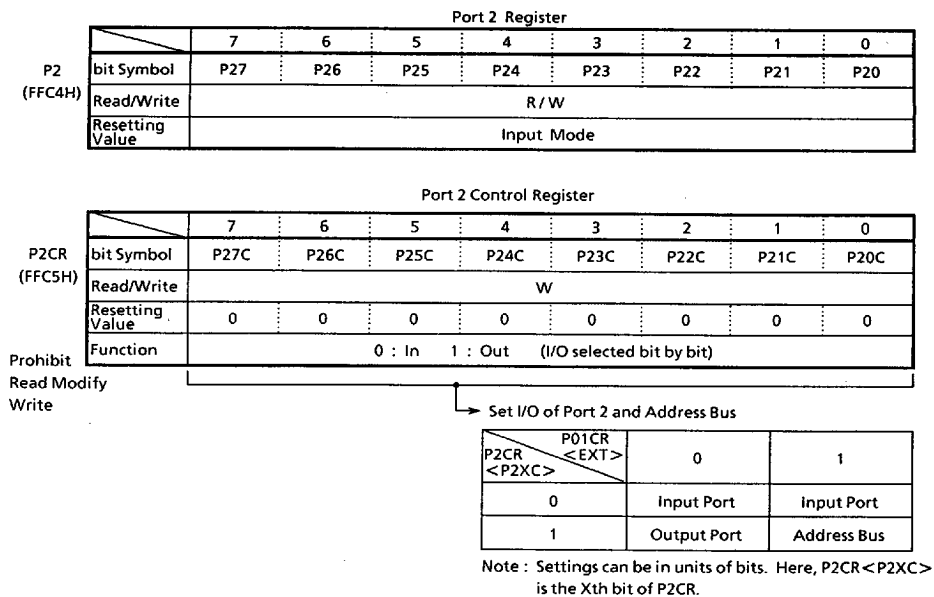
3.5.3 Port 2 (P20~P27)

Port 2 is an 8-bit general-purpose I/O port P2 whose I/O functions are specified by the control register P2CR for each bit. All bits of the output latch and the control register are initialized to "0" by resetting, where by Port 2 turns to the input mode.

In addition to the general-purpose I/O port function, it functions as an address bus (A8~A15). The address bus function can be selected by setting the register P01CR <EXT> (shared with port 1) to "1" and setting the Port 2 control register P2CR to the output mode. When the Port 2 control register P2CR is set to "0", Port 2 functions as an input port, regardless of the status of the <EXT> register.

190990

Figure 3.5 (4) Port 2

ꟼ0ꟼ7249 0040747 473

Port 2 Register

P2 (FFC4H)	bit Symbol	7	6	5	4	3	2	1	0
		P27	P26	P25	P24	P23	P22	P21	P20
	Read/Write	R / W							
	Resetting Value	Input Mode							

Port 2 Control Register

P2CR (FFC5H)	bit Symbol	7	6	5	4	3	2	1	0
		P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
	Read/Write	W							
	Resetting Value	0	0	0	0	0	0	0	0
	Function	0 : In 1 : Out (I/O selected bit by bit)							

Prohibit
Read Modify
Write

└─► Set I/O of Port 2 and Address Bus

P2CR<P2XC> \ P01CR<EXT>	0	1
0	Input Port	Input Port
1	Output Port	Address Bus

Note : Settings can be in units of bits. Here, P2CR<P2XC>
is the Xth bit of P2CR.

010491

Figure 3.5 (5)   Registers for Port 2

3.5.4 Port 3 (P31~P33, P35~P37)

Port 3 is a 6-bit general-purpose I/O port P3 with fixed I/O function. All bits of the output latch are initialized to "1" by resetting, and "High level" is generated to the output port.
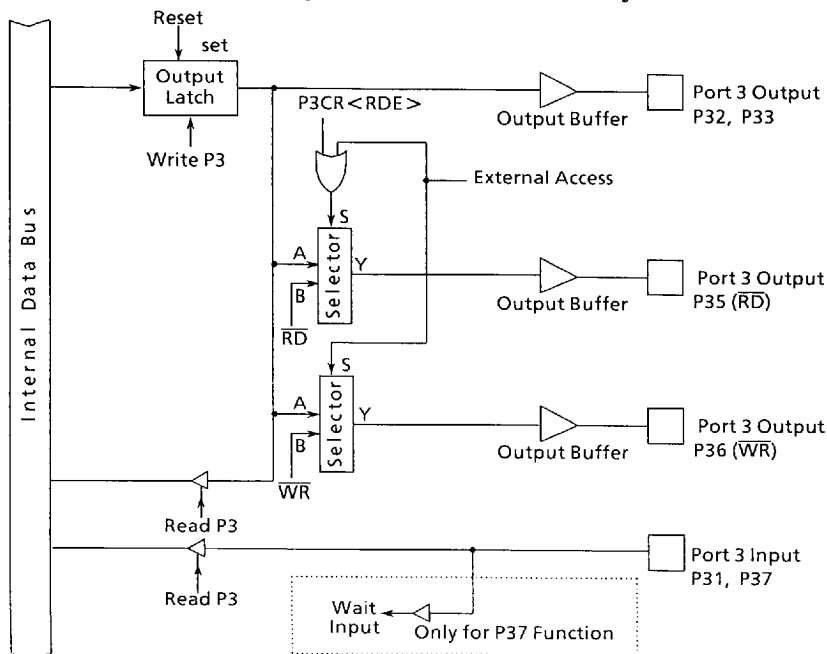
In addition to the I/O port function, P31~P33 have the I/O function for the internal serial interface, while P35~P37 have the external memory control function. The additional functions can be selected by the control register P3CR. All bits of the control register are initialized to "0" by resetting, and the port turns to the general-purpose I/O Ports mode.

However, P37 is placed in the input mode after resetting, and turns to the TO1 output port mode after writing P3CR < WAITC1, 0 > = 1, 1.

Further, access of an external memory makes P35 and P36 automatically function as the memory control pins ($\overline{\text{RD}}$ and $\overline{\text{WR}}$), and access of an internal memory makes them function as general-purpose I/O ports.

When an external memory is accessed, therefore, the output latch registers P35 ($\overline{\text{RD}}$) and P36 ($\overline{\text{WR}}$) should be kept at "1" which is the initial value after the reset.

The P3CR < RDE > of the control register is intended for a pseudostatic RAM. When set to "1", it always functions as an $\overline{\text{RD}}$ pin. Therefore the $\overline{\text{RD}}$ pin outputs "0" (Enable) when it is an internal memory read and internal I/O read cycle.
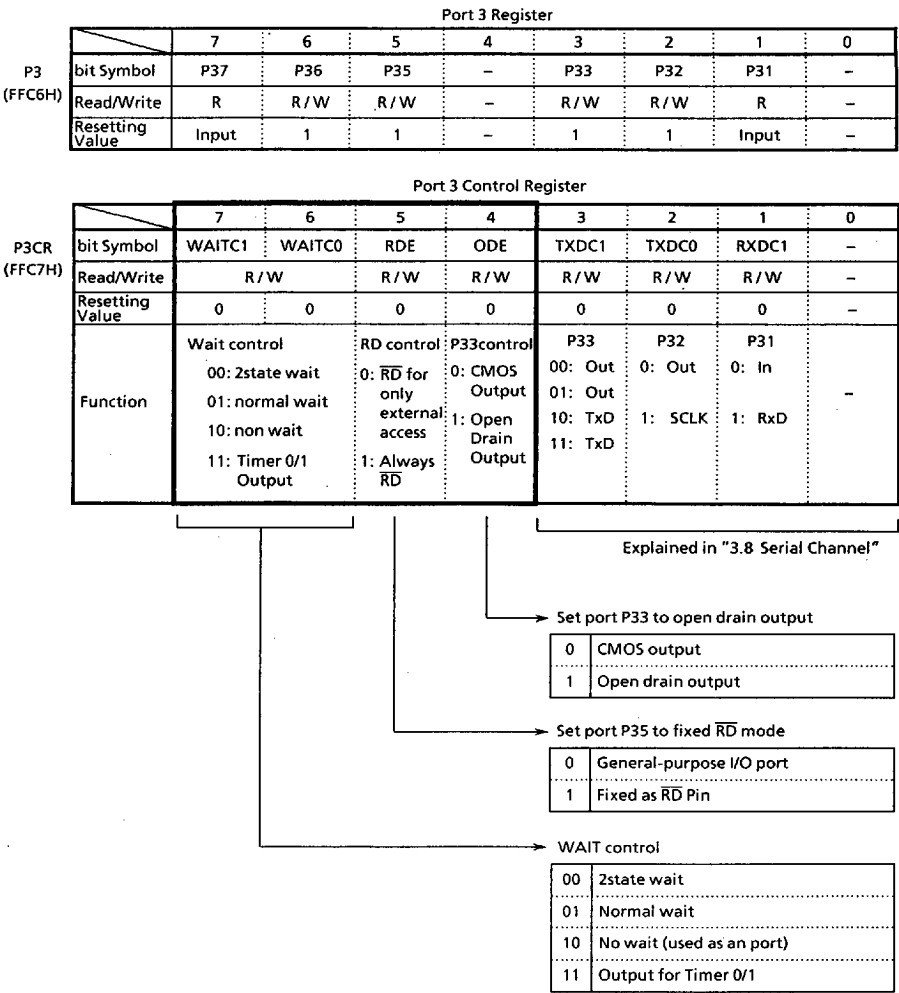

Figure 3.5 (6) Port 3

MCU90-293

Port 3 Register

P3 (FFC6H)		7	6	5	4	3	2	1	0
	bit Symbol	P37	P36	P35	–	P33	P32	P31	–
	Read/Write	R	R/W	R/W	–	R/W	R/W	R	–
	Resetting Value	Input	1	1	–	1	1	Input	–

Port 3 Control Register

P3CR (FFC7H)		7	6	5	4	3	2	1	0
	bit Symbol	WAITC1	WAITC0	RDE	ODE	TXDC1	TXDC0	RXDC1	–
	Read/Write	R/W		R/W	R/W	R/W	R/W	R/W	–
	Resetting Value	0	0	0	0	0	0	0	–
	Function	Wait control 00: 2state wait 01: normal wait 10: non wait 11: Timer 0/1 Output		RD control 0: RD for only external access 1: Always RD	P33control 0: CMOS Output 1: Open Drain Output	P33 00: Out 01: Out 10: TxD 11: TxD	P32 0: Out 1: SCLK	P31 0: In 1: RxD	–

Explained in "3.8 Serial Channel"

Set port P33 to open drain output

0	CMOS output
1	Open drain output

Set port P35 to fixed RD mode

0	General-purpose I/O port
1	Fixed as RD Pin

WAIT control

00	2state wait
01	Normal wait
10	No wait (used as an port)
11	Output for Timer 0/1

230890

Figure 3.5 (7) Register for Ports 3

9097249 0040750 T68

3.5.5 Port 8 (P80~P81)

Port 8 is a 2-bit general-purpose INPUT port P8.

Port 8 also has the functions of interrupt request input, and clock input for a timer/event counter.

(1) P80/INT0

P80 is a general-purpose input port, also used as the external interrupt request input pin INT0. INT0 allows the selection of either an "H" level interrupt or rising edge interrupt by using the control register P8CR <EDGE>.
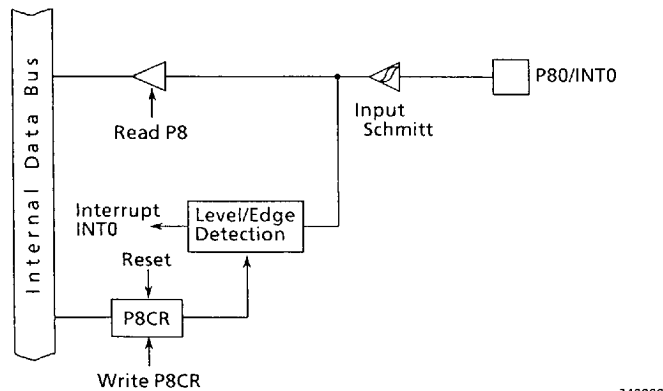
Figure 3.5 (8) Port P80/INT0

(2) P81/INT1/TI2

P81 is a general-purpose input port, also used as the external interrupt request input pin INT1 and the clock input pin TI2 for the timer/event counter.

This port incorporates a zero-cross detection circuit, and enables zero-cross detection by connecting an external capacitor. The zero-cross detection can be disabled/enabled by using the control register P8CR <ZCE1>. This control register is reset to "0", making the zero-cross detection disabled by resetting.
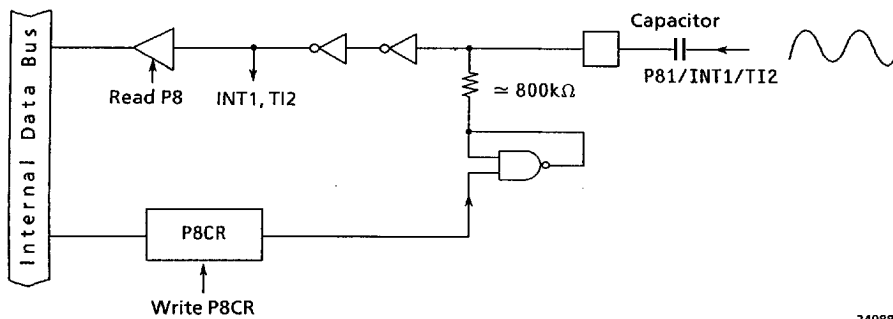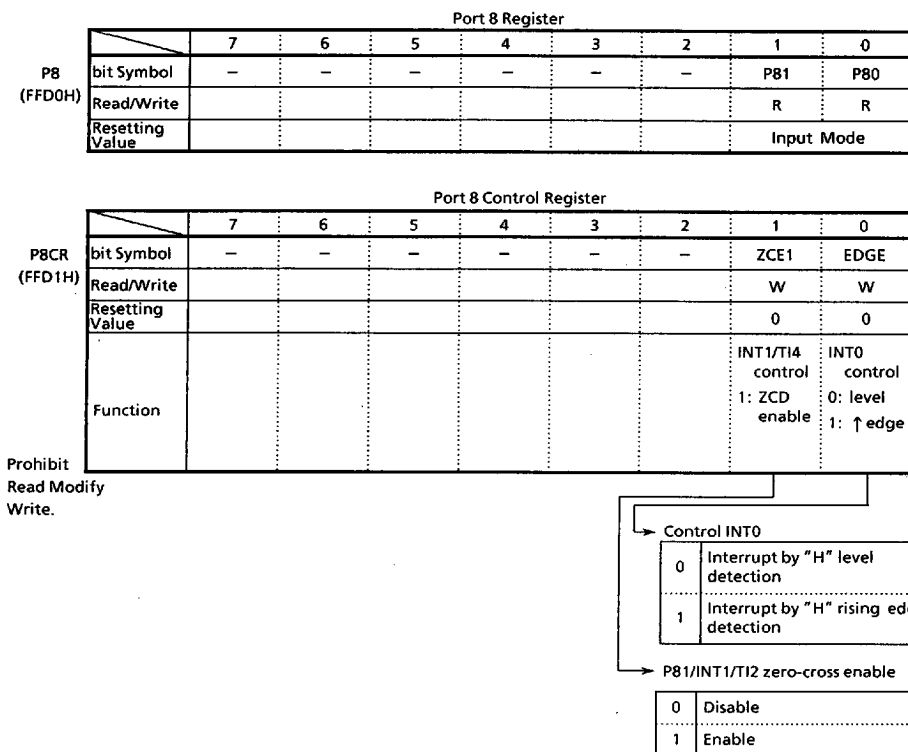
9097249 0040751 9T4

Figure 3.5 (9) Port P81/INT1/TI2

240889

Port 8 Register

		7	6	5	4	3	2	1	0
P8 (FFD0H)	bit Symbol	−	−	−	−	−	−	P81	P80
	Read/Write							R	R
	Resetting Value							Input Mode	

Port 8 Control Register

		7	6	5	4	3	2	1	0
P8CR (FFD1H)	bit Symbol	−	−	−	−	−	−	ZCE1	EDGE
	Read/Write							W	W
	Resetting Value							0	0
	Function							INT1/TI4 control 1: ZCD enable	INT0 control 0: level 1: ↑edge

Prohibit Read Modify Write.

Control INT0

0	Interrupt by "H" level detection
1	Interrupt by "H" rising edge detection

P81/INT1/TI2 zero-cross enable

0	Disable
1	Enable

Figure 3.5 (10)   Registers for Port 8

9097249 0040752 830

3.6   Timers

The TMP90C802A incorporates four 8-bit timers.

The four 8-bit timers can be operated independently, and can also be functioned as two 16-bit timer by mode setting: Timer 2 has an event counter function, so that it can also be used as an 8-bit counter. Furthermore, it can be used as a 16-bit counter cascaded with Timer 3.

- 8-bit interval timer mode (4 timers)
- 16-bit interval timer mode (2 timers)
- 8-bit programmable pulse generation (PPG) output mode (Timer 0 and Timer 1)
- 8-bit PWM output mode (Timer 1)   Possible arrangements : 8-bit×2 and 16-bit×1
- 8-bit event counter mode (Timer 2)
- 16-bit event counter mode (Timer 2 and Timer 3)
- Software counter latch function (Timer 2 and Timer 3)

3.6.1 8-bit Timers

The TMP90C802A incorporates four 8-bit interval timers (Timers 0, 1, 2 and 3) , each of which can be operated independently. The cascade connection of Timer 0 and 1, or Timer 2 and 3 allows these timers used as 16-bit internal timers.

Figure 3.6 (1) is a block diagram of the 8-bit timers (Timer 0 and Timer 1) .

Figure 3.6 (2) is a block diagram of the 8-bit timer/event counters (Timer 2 and Timer 3).

Each interval timer is composed of an 8-bit up-counter, an 8-bit comparator and an 8-bit timer register, with a Timer Flip-flop (TFF1) provided to pair of Timer 0/1.

Internal clocks (ϕT1, ϕT16 and ϕT256), some of the input clock sources for the interval timers, are generated by the 9-bit prescaler shown in Figure 3.6 (3) .

Their operating modes of the 8-bit timers and flip-flops are controlled by four control registers (TCLK, TFFCR, TMOD and TRUN) .
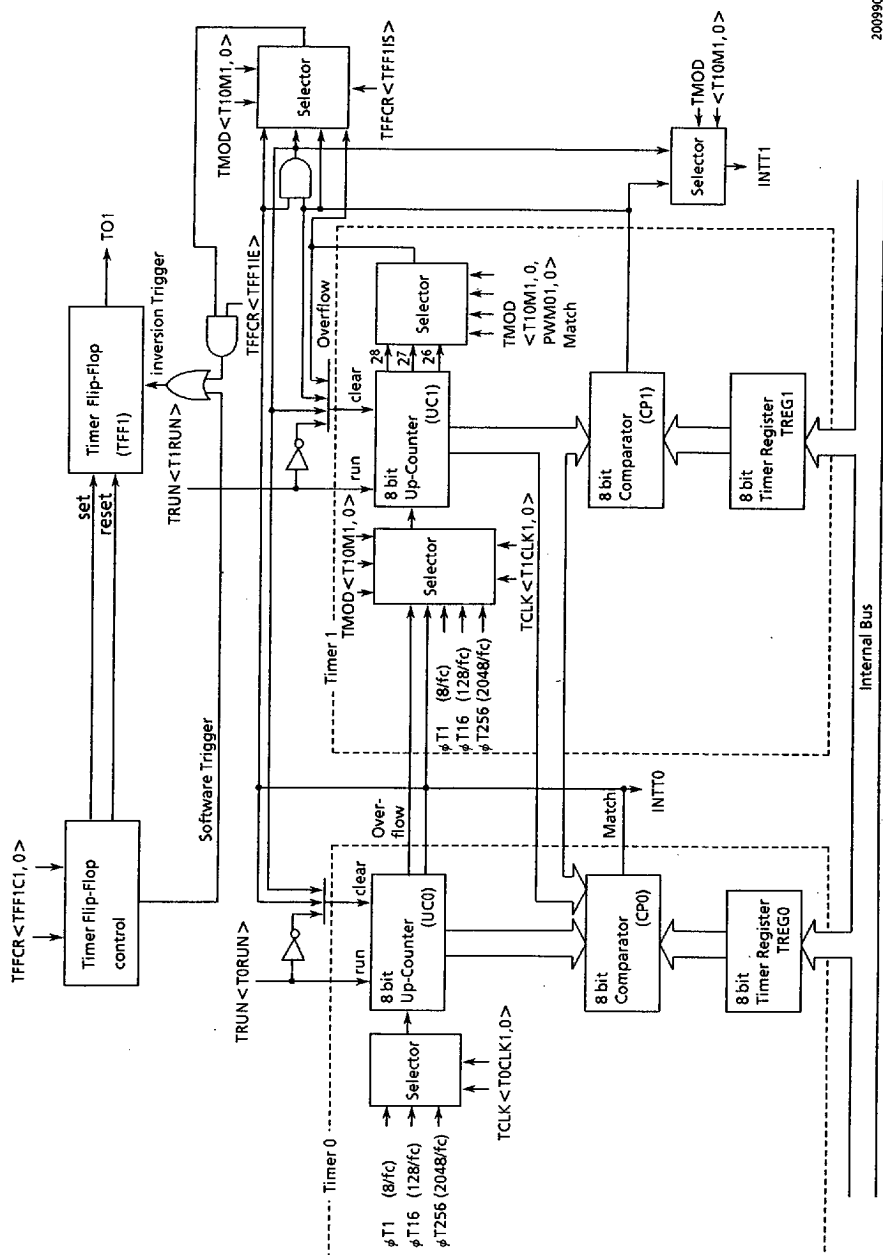
9097249 0040753 777

Figure 3.6 (1)   Block Diagram of 8-bit Timers (Timer 0 and Timer 1)
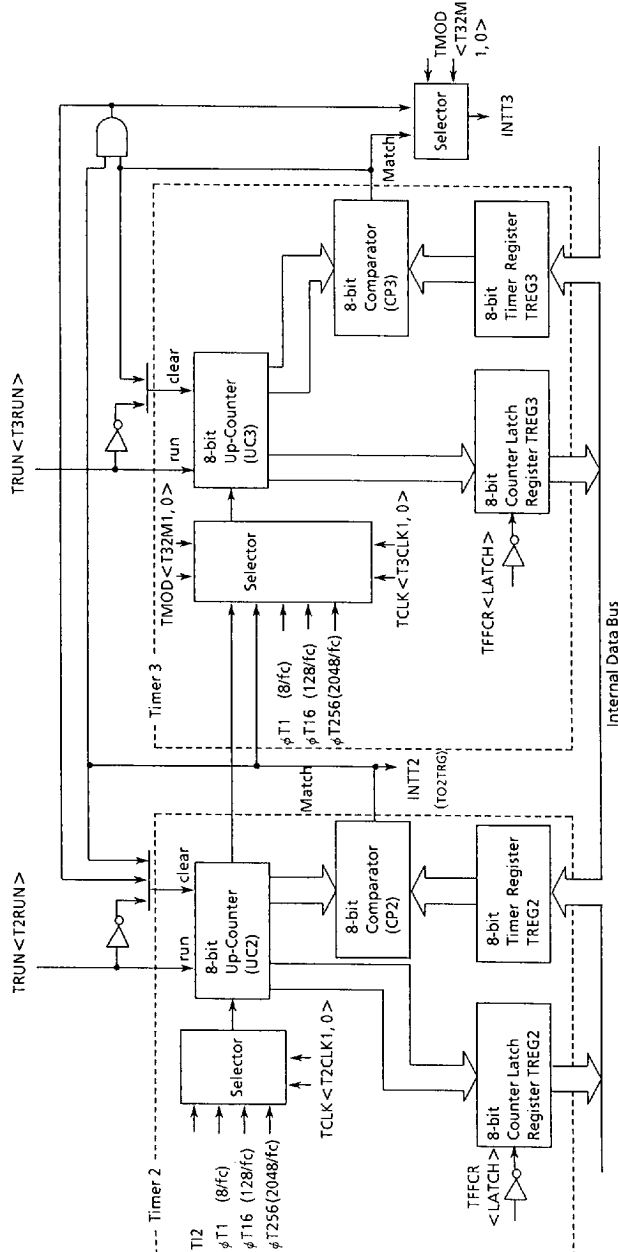
9097249 0040754 603

Figure 3.6 (2)   Block Diagram of 8-bit Timer/Counter (Timer 2 and Timer 3)

9097249 0040755 54T

① Prescaler
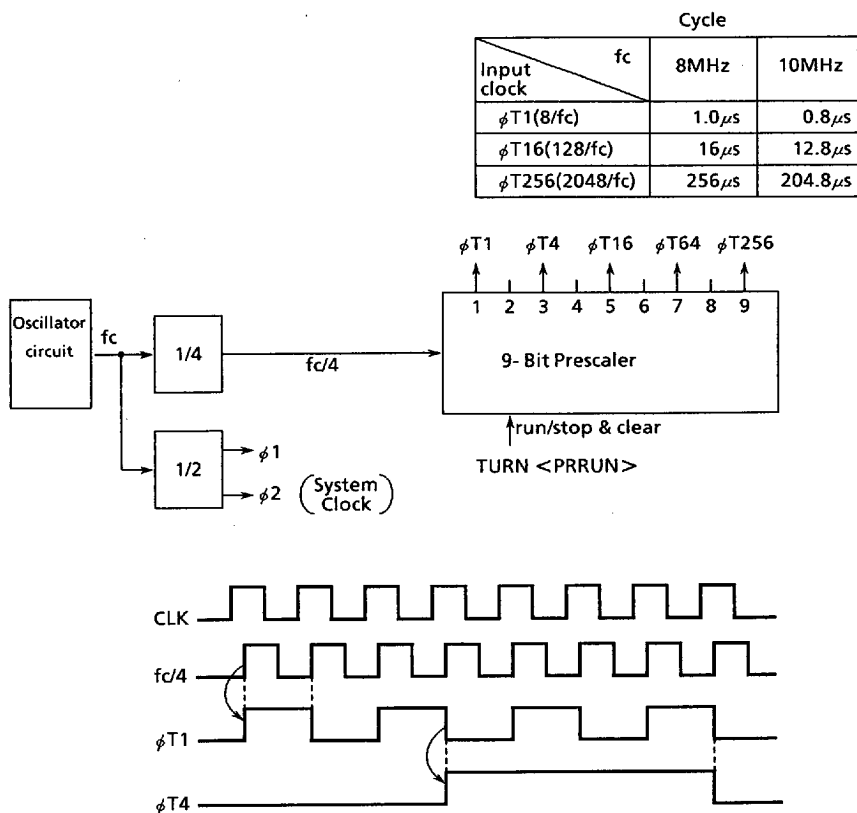
An 9-bit prescaler is provided to further devide the clock frequency already divided to a 1/4 of the frequency of the source clock (fc).

It generates an input clock pulse for the 8-bit timers, the baud-rate generator, etc.

For the 8-bit timers, three types of clock are generated (ϕT1, ϕT16 and ϕT256).

The prescaler can be run or stopped by using the 5th bit TRUN<PRRUN> of the timer control register TRUN. Setting <TRUN> to "1" makes the prescaler count, and setting it to "0" clears the prescaler to stop.

By resetting, <PRRUN> is initialized to "0", making the prescaler clear and stop.

Input clock \ fc	Cycle	
	8MHz	10MHz
ϕT1(8/fc)	1.0μs	0.8μs
ϕT16(128/fc)	16μs	12.8μs
ϕT256(2048/fc)	256μs	204.8μs

Figure 3.6 (3) Prescaler

240889

② Up-counter

This is an 8-bit binary counter that counts up by an input clock pulse specified by an 8-bit timer clock control register TCLK and an 8-bit timer mode register (TMOD).

The input clock pulse for Timer 0 and 2 is selected from ϕT1, ϕT16 and ϕT256 according to the setting of the TCLK register. When using Timer 2 as the counter, set bit 4 and bit 5 of TCLK to "0".

Example : When setting TCLK<T0CLK1, 0>=0, 1, ϕT1 is selected as the input clock pulse for Timer 0.

The input clock pulse to Timer 1 and 3 is selected according to the operating mode. In the 16-bit timer/counter mode, the overflow output of Timer 0 and 2 is automatically selected as the input clock pulse, regardless of the setting of the TCLK register.

In the other operating modes, the clock pulse is selected among the internal clocks ϕT1, ϕT16 and ϕT256, and the output of the Timer 0 and 2 comparator (match signal).
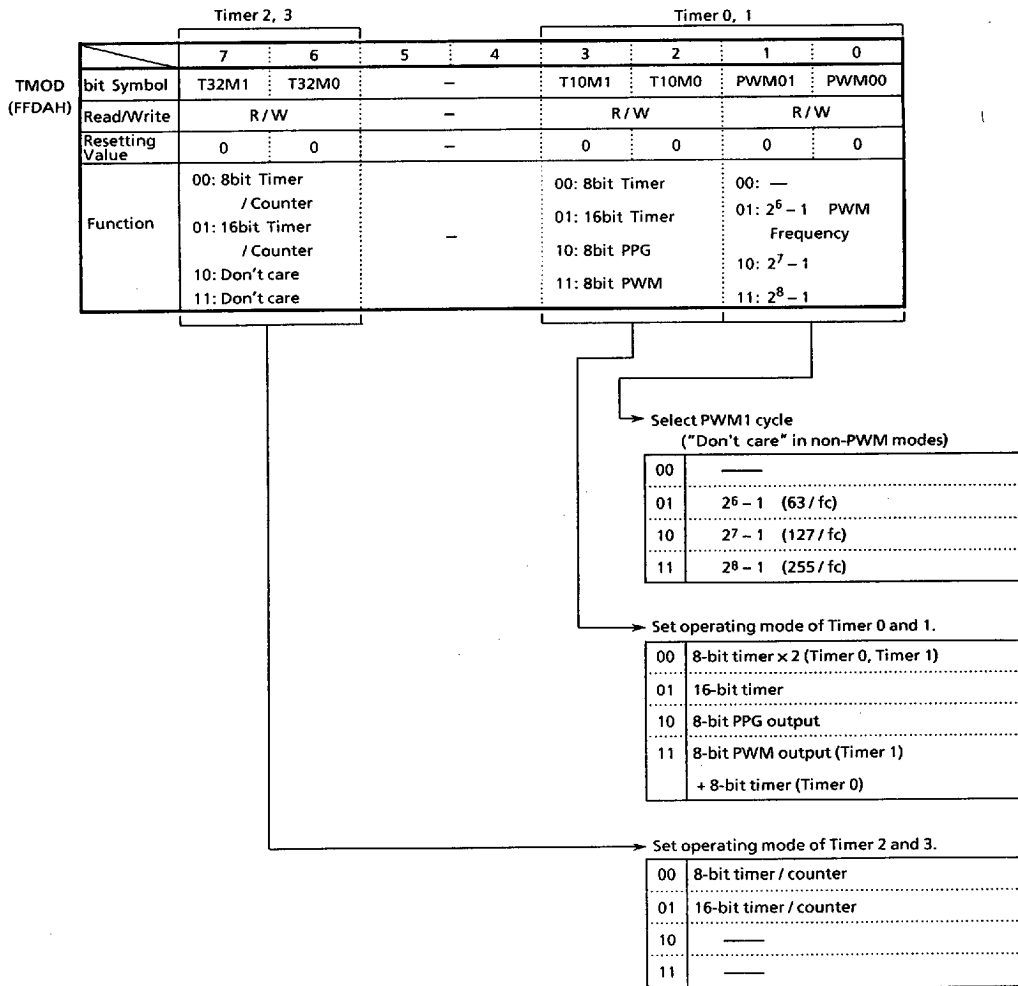
Example : If TMOD<T10M10>=0, 1, the overflow output of Timer 0 is selected as the input clock to Timer 1. (16 bit timer mode)
If TMOD<T10M1, 0>=0, 0 and TCLK<T1CLK1, 0>=0, 1, ϕT1 is selected as the input clock to Timer 1. (8 bit timer mode)

The operating mode is selected by the TMOD register. This register is initialized to TMOD<T10M1, 0>=0, 0/TMOD<T32M1, 0>=0, 0 by resetting, whereby the up-counter is placed in the 8-bit timer mode.
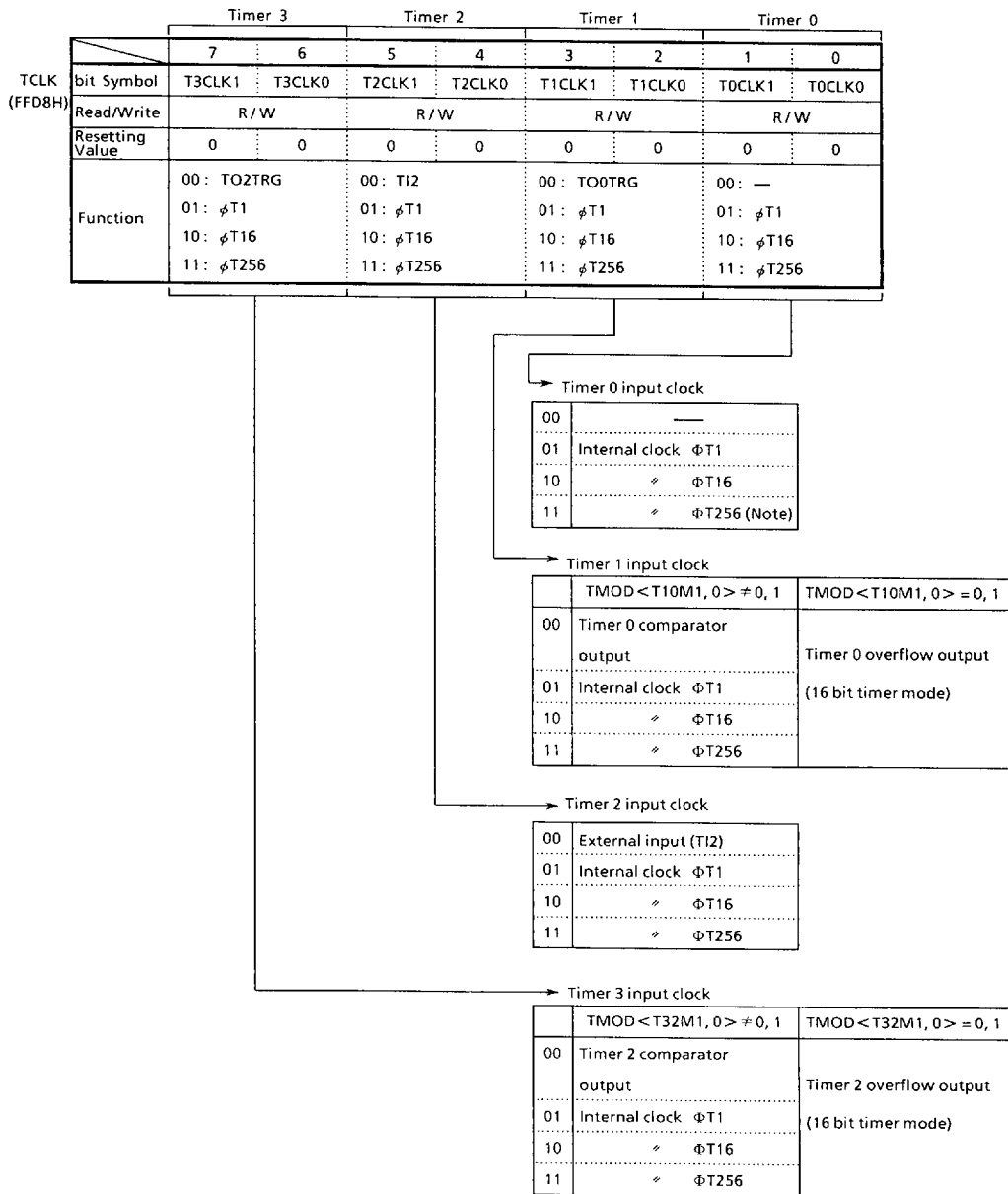
Functions, count, stop or clear of the up-counter can be controlled for each interval timer by the timer control register TRUN.

By resetting, all up-counters are cleared to stop the timers.

9097249 0040757 312

		7	6	5	4	3	2	1	0
		Timer 2, 3				Timer 0, 1			
TMOD (FFDAH)	bit Symbol	T32M1	T32M0		–	T10M1	T10M0	PWM01	PWM00
	Read/Write	R / W			–	R / W		R / W	
	Resetting Value	0	0		–	0	0	0	0
	Function	00: 8bit Timer /Counter 01: 16bit Timer /Counter 10: Don't care 11: Don't care			–	00: 8bit Timer 01: 16bit Timer 10: 8bit PPG 11: 8bit PWM		00: — 01: 2^6-1 PWM Frequency 10: 2^7-1 11: 2^8-1	

Select PWM1 cycle
("Don't care" in non-PWM modes)

00	——
01	2^6-1 (63 / fc)
10	2^7-1 (127 / fc)
11	2^8-1 (255 / fc)

Set operating mode of Timer 0 and 1.

00	8-bit timer × 2 (Timer 0, Timer 1)
01	16-bit timer
10	8-bit PPG output
11	8-bit PWM output (Timer 1) + 8-bit timer (Timer 0)

Set operating mode of Timer 2 and 3.

00	8-bit timer / counter
01	16-bit timer / counter
10	——
11	——

270890

Figure 3.5 (4) 8-bit Timer Mode Register TMOD

9097249 0040758 259

		Timer 3		Timer 2		Timer 1		Timer 0	
		7	6	5	4	3	2	1	0
TCLK (FFD8H)	bit Symbol	T3CLK1	T3CLK0	T2CLK1	T2CLK0	T1CLK1	T1CLK0	T0CLK1	T0CLK0
	Read/Write	R / W		R / W		R / W		R / W	
	Resetting Value	0	0	0	0	0	0	0	0
	Function	00 : TO2TRG 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : TI2 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : TO0TRG 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : — 01 : ϕT1 10 : ϕT16 11 : ϕT256	

Timer 0 input clock

00	—
01	Internal clock ΦT1
10	$\nearrow$ ΦT16
11	$\nearrow$ ΦT256 (Note)

Timer 1 input clock

	TMOD<T10M1, 0> ≠ 0, 1	TMOD<T10M1, 0> = 0, 1
00	Timer 0 comparator output	Timer 0 overflow output (16 bit timer mode)
01	Internal clock ΦT1	
10	$\nearrow$ ΦT16	
11	$\nearrow$ ΦT256	

Timer 2 input clock

00	External input (TI2)
01	Internal clock ΦT1
10	$\nearrow$ ΦT16
11	$\nearrow$ ΦT256

Timer 3 input clock

	TMOD<T32M1, 0> ≠ 0, 1	TMOD<T32M1, 0> = 0, 1
00	Timer 2 comparator output	Timer 2 overflow output (16 bit timer mode)
01	Internal clock ΦT1	
10	$\nearrow$ ΦT16	
11	$\nearrow$ ΦT256	

031090

Figure 3.6 (5) 8-bit Timer Clock Control Register TCLK

■ 9097249 0040759 195 ■

		8bit Timer						
		7	6	5	4	3	2	1	0

		7	6	5	4	3	2	1	0
TRUN (FFDBH)	bit Symbol	BRATE1	BRATE0	PRRUN	–	T3RUN	T2RUN	T1RUN	T0RUN
	Read/Write	R / W		R / W					
	Resetting Value	0	0	0	–	0	0	0	0
	Function	00: 300/150 bps 01: 1200/600 10: 4800/2400 11: 19200/9600 @ fc = 9.8304MHz		Prescaler & Timer Run/Stop Control 0: Stop & Clear 1: Run (Count up)					

Select Timer 0 operation

0	Stop and clear
1	Count

Select Timer 1 operation

0	Stop and clear
1	Count

Select Timer 2 operation

0	Stop and clear
1	Count

Select Timer 3 operation

0	Stop and clear
1	Count

Select prescaler operation

0	Stop and clear
1	Count

Select transfer speed of serial I/O baud rate generator (fc = 9.8304MHz)

	SCMOD<SC1,0> = 0, 1	SCMOD<SC1,0> = 1, 1
00	300 bps	150 bps
01	1200	600
10	4800	2400
11	19200	9600

Figure 3.6 (6) Timer/Serial Channel Control Registers TRUN

TFFCR (FFD9H)		7	6	5	4	3	2	1	0
						TFF1			
	bit Symbol	LATCH	—	—		TFF1C1	TFF1C0	TFF1IE	TFF1IS
	Read/Write	W	—	—		W		R/W	
	Resetting Value	1	—	—		—		0	0
	Function	0: LATCH (one shot)	—	—		00: Clear TFF1 01: Set TFF1 10: Invert TFF1 11: Don't care		1: TFF1 Invert Enable	0: Invert by 8-bit timer 0 1: Invert by 8-bit timer 1

Select inverse signal of timer flip-flop TFF1

TMOD3, 2 / FF1IS	0, 0	0, 1	1, 0	1, 1
0	8-bit timer mode (Timer 0)	—		
1	8-bit timer mode (Timer 1)	16-bit timer mode (Timers 0 & 1)	PPG mode (Timer 1)	PWM mode (Timer 1)

Invert timer flip-flop TFF1

0	Disable
1	Enable

Control timer flip-flop TFF1

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

Software latch trigger

0	Fetch the value of up-counter into the counter latch register.
1	Don't care (Always set to "1" when read out).

270890

Figure 3.6 (7)   8-bit Timer Flip-Flop Control Register (TFFCR)

③ Timer registers

TREG0	TREG1	TREG2	TREG3
8 - bits	8 - bits	8 - bits	8 - bits
FFD4H	FFD5H	FFD6H	FFD7H

Note : Only for writing

8-bit registers are provided to set the interval time. When the set value of a timer register matches that of an up-counter, the match signal of their comparators turn to the active mode. If "00H" is set, this signal becomes active when the up-counter overflows.

The values of the timer register 0 and the timer register 1 cannot be read. The values of the timer register 2 and the timer register 3, however, can be read because these registers are assigned same address with the counter latch registers. When the values of these registers are read, they become the values of the counter latch registers (Read only register). When the values of these registers are written, they become the values of the values of the timer registers (Write only register).

④ Comparators

A comparator compares the values in an up-counter and a timer register. When they matches, the up-counter is cleared to "0", and an interrupt signal (INTTn) is generated. If the Timer Flip-Flop inversion is enabled by the Timer Flip-Flop Control Register, the Timer Flip-Flop is inverted.

⑤ Timer FliF-flop (Timer F/F)

The status of the Timer Flip-Flop is inverted by the match signal (output by comparator) of each interval timer. Its status can be output to the timer output pin TO1 (also used as P37).

This Timer F/F is provided to the timer pair, Timer 0 - Timer 1, and is called TFF1. The status of TFF1 is output to TO1.

The Timer F/F are controlled by a Timer Flip-Flop Control Register (TFFCR).

● TFFCR<FF1IS> is a timer selection bit for inversion of TFF1. In the 8-bit timer mode, inversion is enabled by the match signal from Timer 0 if this bit is set to "0", or by the signal from Timer 1 if set to "1".

In any other mode, <FF1IS> must be always set to "1". It is initialized to "0" by resetting.

● TFFCR<FF1IE> controls the inversion of TFF1. Setting this bit to "1" enables the inversion and setting it to "0" disable.

<FF1IE> is initialized to "0" by resetting.

The bits TFFCR are used to set/reset TFF1 or enable its inversion by software. TFF1 is reset by writing "0, 0", set by "0, 1" and inverted by "1, 0".

The 8-bit timers operate as follows:

(1)   8-bit timer mode

The four interval timers, Timer 0, Timer 1, Timer 2 and Timer 3, can be operated independently as the 8-bit interval timers. Only the operation of Timer 1 is described in this section because the operations of all the timers are the same.

① Generating interrupts at specified intervals

Periodic interrupts can be generated by using Timer 1 (INTT1) in the following procedure: Stop Timer 1, set the desired operating mode, input clock and cycle time in, the registers TMOD, TCLK and TREG1, enable INTT1, and start the counting of Timer 1.

Example : To generate Timer 1 interrupt every 40μs at fc = 10 MHz, the registers should be set as follows :

```
         MSB                LSB
             7 6 5 4 3 2 1 0
  ┌ TRUN  ←  – – – – – – 0 –        Stop Timer 1, and clear it to "0".
  │ TMOD  ←  – – – – 0 0 X X        Set the 8-bit timer mode.
  │ TCLK  ←  – – – – 0 1 – –        Select ɸT1 (0.8μs @fc = 10 MHz) as the input clock.
  │ TREG1 ←  0 0 1 1 0 0 1 0        Set the timer register at 40μs/ɸT1 = 32H.
  │ INTEH ←  – – – – – – – 1        Enable INTT1.
  └ TRUN  ←  – – 1 – – – 1 –        Start Timer 1.
```

Note :   X : Don't care   – : No change

Use the following table for selecting the input clock:

Table 3.6 (1)   8-bit timer interrupt cycle and input clock

Interrupt cycle @fc = 10 MHz	Resolution	Input clock
0.8μs ～ 204μs	0.8μs	ɸT1 (8/fc)
12.8μs ～ 3.264ms	12.8μs	ɸT16 (128/fc)
204.8μs ～ 52.429ms	204.8μs	ɸT256 (2048/fc)

250889

② Generating pulse at 50% duty

The Timer Flip-flop is inverted at specified intervals, and its status is output to a timer output pin TO1 (only Timer 0, Timer 1).

Example: To output pulse from TO1 at fc = 10 MHz every 4.8μs, the registers should be set as follows:

This example uses Timer 1, but the same operation can be effected by using Timer 0.

```
        MSB              LSB
         7 6 5 4 3 2 1 0
 ┌ TRUN  ←  – – – – – – 0 –      Stop Timer 1, and clear it to "0".
 │ TMOD  ←  – – – – 0 0 X X      Set the 8-bit timer mode.
 │ TCLK  ←  – – – – 0 1 – –      Select φT1 as the input clock.
 │ TREG1 ←  0 0 0 0 0 0 1 1      Set the timer register at 4.8μs/φT1/2 = 3.
 │ TFFCR ←  – – – – 0 0 1 1      Clear TFF1 to "0", and set to invert by the match
 │                               signal from Timer 1.
 │ P3CR  ←  1 1 – – – – – –      Select P37 as TO1 pin.
 └ TRUN  ←  – – 1 – – – 1 –      Start Timer 1.
```

Note : X : Don't care   – : No change
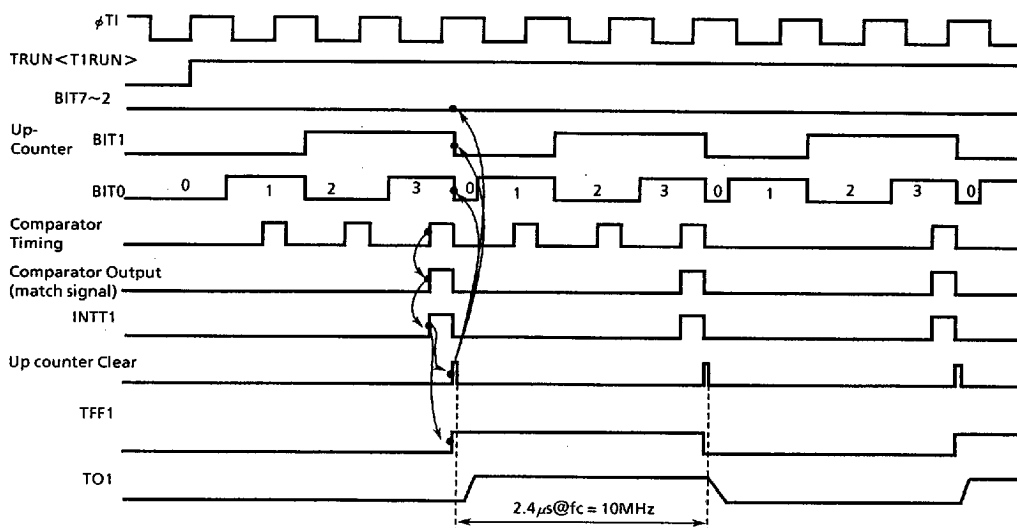
Figure 3.6 (8) Pulse Output (50% duty) Timing Chart

9097249 0040764 552

③ Making Timer 1 count up by match signal from Timer 0 comparator.

Select the 8-bit timer mode, and set the comparator output of Timer 0 as the input clock to Timer 1.
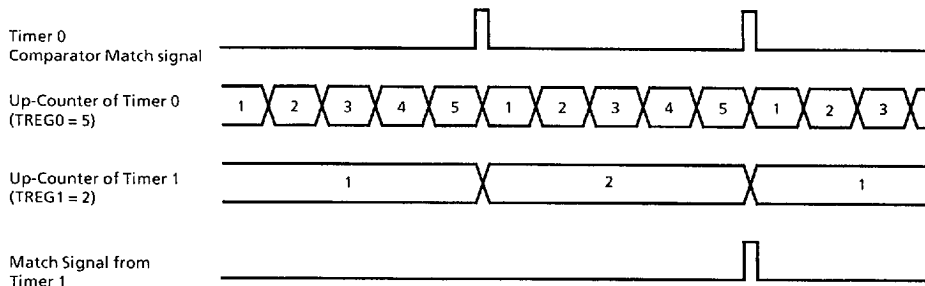
250889

Figure 3.6 (9)

④ Software inversion

The Timer Flip-Flops can be inverted by software independent of the timer operation.
Writing "1, 0" into the bits TFFCR<TFF1C1, 0> inverts TFF1.

⑤ Initial setting of Timer Flip-Flops

The Timer Flip-flops can be initialized to either "0" or "1" without regard to the timer operation.
TFF1 is initialized to "0" by writing "0, 0" into TFFCR<TFF1C1, 0>, and "1" by writing "0, 1" into these bits.

(Note) : Reading the data from the Timer Flip-flops and timer registers is prohibited.

(2) 16-bit timer mode

The Timer 0 and Timer 1 or Timer 2 and Timer 3 can be used as one 16-bit interval timer.
Only the operation of Timer 0 and Timer 1 is described in this section since the operation of Timer 2 and Timer 3 is identical with that of Timer 0 and Timer 1 except a pair of Timer 2 and 3 does not have Timer Output function.
Cascade connection of Timer 0 and Timer 1 to use them as a 16-bit interval timer requires to set the <T10M1, 0>of the mode register TMOD to "0, 1".
By selecting the 16-bit timer mode, the overflow output of Timer 0 is automatically selected as the input clock to Timer 1, regardless of the set value of the clock control register TCLK. The input clock to Timer 0 is selected by TCLK. Table 3.6 (2) shows the combinations of timer (interrupt) cycle and input clock.

Table 3.6 (2)   16-bit timer (interrupt) cycle and input clock

Timer (interrupt) cycle @fc = 10MHz	Resolution	Input clock to Timer 0
0.8μs~52.43ms	0.8μs	ϕT1 (8/fc)
12.8μs~838.86ms	12.8μs	ϕT16 (128/fc)
204.8μs~13.42s	204.8μs	ϕT256 (256/fc)

250889

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

Example: To generate interrupts INTT1 at fc=8MHz every 1 second, the timer registers TREG0 and TREG1 should be set as follows:

As ϕT16 ($=16\mu$s @8MHz) is selected as the input clock,

$$1 \text{ s}/16\mu s = 62500 = \text{F424H}$$

Therefore,

$$\text{TREG1} = \text{F4H}$$
$$\text{TREG0} = \text{24H}$$

The match signal is generated by Timer 0 comparator each time the up-counter UC0 matches TREG0. In this case, the up-counter UC0 is not cleared, but the interrupt INTT0 is generated.

Timer 1 comparator also generates the match signal each time the up-counter UC1 matchs TREG1. When the match signal is generated simultaneously from comparators of Timer 0 and Timer 1, the up-counters UC0 and UC1 are cleared to "0", and the interrupt INTT1 is generated. If the Timer Flip-Flop inversion is enabled by the Timer Flip-Flop control register, the Timer Flip-Flop TFF1 is inverted at the same time.

	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 (TO1 can't be output matching with TREG0.)	TREG0 (Continue counting when match)	Interrupt is generated.	Can output (TO1 can be output matching with both TREG0 and TREG1)	TREG1$\ast2^8$ + TREG0(16 bit) (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$\ast$TREG0 (Multiplied Valve) (Cleared by matching)

031090

Example: Given TREG1 = 04H and TREG0 = 80H,
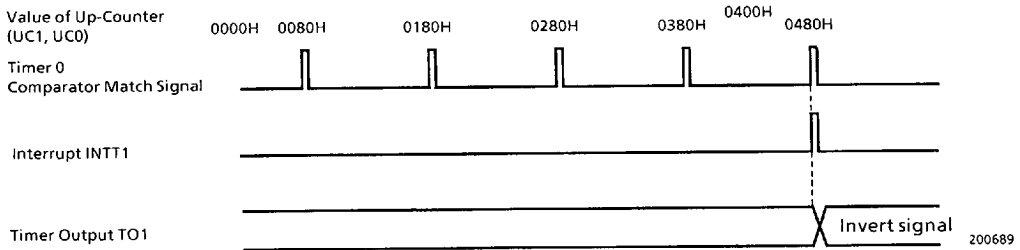
Figure 3.6 (10)

(3) 8-bit PPG (Programmable pulse generation) mode

Pulse can be generated at any frequency and duty rate by Timer 1. The output pulse may be either low-or high-active.

In this mode, Timer 0 can not be used.

Pulse is output to TO1 (shared with P37).
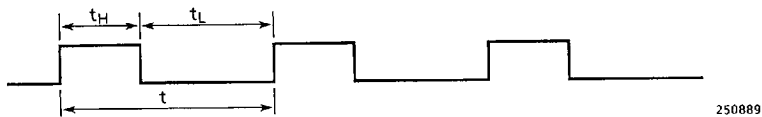
Following is the timing of Timer 1
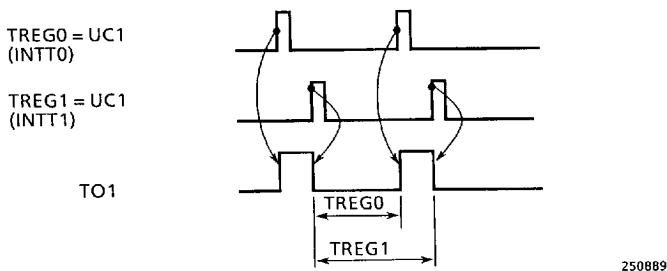
In the 8-bit PPG mode, programmable pulse is generated by the inversion of the timer output each time the 8-bit up-counter 1 (UC1) matches the timer register TREG0 or TREG1.

Note that the set value of TREG0 must be smaller than that of TREG1.

In this mode, the up-counter UC0 of Timer 0 can not be used (Set TRUN <T0RUN> = 1, and count the Timer 0).

The block diagram of the PPG mode is illustrated as follows :

■ 9097249 0040767 261 ■
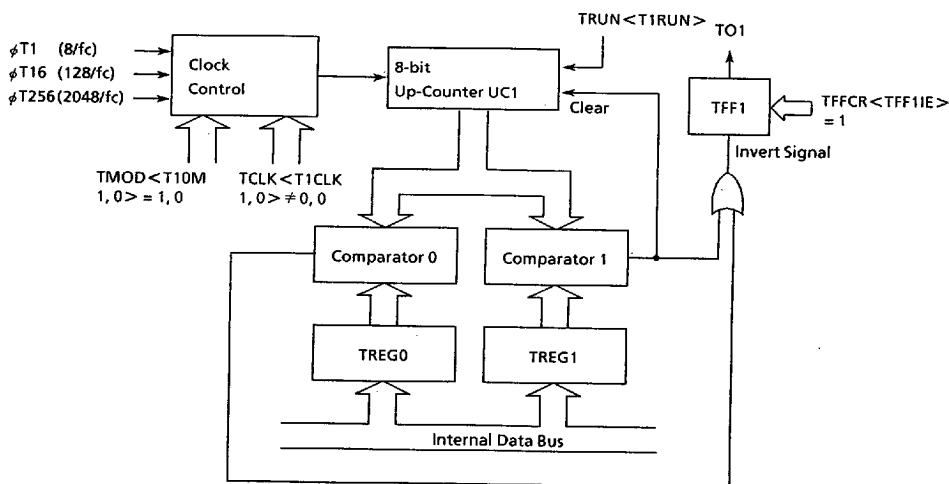
Figure 3.6 (11)   Block Diagram of 8-bit PPG Mode

Example : Generate pulse at 50kHz and 1/4 duty rate (@fc = 8MHz)

- Calculate the set values of the timer registers.

To obtain the frequency of 50kHz, the pulse cycle t should be: $1/50kHz = 20\mu s$.
Given $\phi T1 = 1\mu s$ (@8MHz),

$20\mu s/1\mu s = 20$

Consequently, the timer register 1 (TREG1) should be set to $20 = 14H$.
Given a 1/4 duty, $t \times 1/4 = 20 \times 1/4 = 5\ \mu s$

$5\ \mu s/1\ \mu s = 5$

As a result, the timer register 0 (TREG0) should be set to $5 = 05H$.

9097249 0040768 1T8

```
TRUN   ←  - - - - - - 0 0        Stop Timer 0 and Timer 1, and clear them to "0".
TCLK   ←  - - - - 0 1 x x        Select ∮T1 as the input clock.
TMOD   ←  - - - - 1 0 x x        Set 8-bit PPG mode.
TFFCR  ←  - - - - 0 1 1 1        Set the output "H", and enable the inversion by Timer 1.
                     └─┘
                      └──────────────────→ Writing "00" provides negative logic pulse
TREG0  ←  0 0 0 0 0 1 0 1        Write "5H".
TREG1  ←  0 0 0 1 0 1 0 0        Write "14H".
P3CR   ←  1 1 - - - - - -        Select P37 as the TO1 pin.
TRUN   ←  - - 1 - - - 1 1        Start Timer 1.
```

Note :   x : Don't care   - : No change

Precautions for PPG Output

By rewriting the content of the TREG (timer register), it is possible to make TMP90C802 output PPG. However, be careful, since the timing to rewrite TREG differs depending on the pulse width of PPG to be set. This problem is explained below by an example.

Example : To output PPG through 8 bit timers 0 and 1

TREG0 :   Pulse width

TREG1 :   Cycle
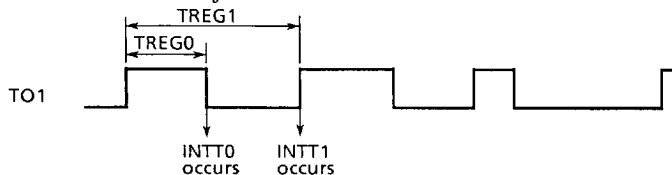
200990

The pulse width is normally changed by the interrupt (INTT1) process routine in each cycle. However, when the pulse width to be set (the value to be written in TREG0) is small, trouble may occur, in that the timer counter exceeds the value of TREG0 before the interrupt process routine is set. Therefore, it is recommended to make the following decisions in INTT0 and INTT1 interrupt processes.

INTT0 process routine :   The value of TREG0 is rewritten only when the value to be written in TREG0 is smaller than the current value of TREG0.

INTT1 process routine :   On the contrary to INTT0, TREG0 is rewritten only when the value to be written in TREG0 is larger than the current value of TREG0.

TMP90C802 cannot read the content of TREG, so it is necessary to buffer the content of TREG in a RAM (or the like) for making the above judgment.

MCU90-313

■ 9097249 0040769 034 ■

**(4) 8-bit PWM (pulse width modulation) mode**

This mode is only available for Timer 1, and can output 8-bit resolution PWM. It is output to TO1 (also used as P37). Timer 0 can be used as 8-bit timers.

The inversion of the timer output occurs when the up-counter (UC1) matches the set value of the timer register TREG1, as well as when an overflow of $2^n - 1$ (n = 6, 7 or 8 selected by TMOD<PWM1, 00> occurred at the counter. The up-counter UC1 is cleared by the occurrence of an overflow of $2^n - 1$. For example, 6-bit PWM is selected when n = 6, and 7-bit PWM is selected when n = 7.

The following condition must be obtained in this PWM mode:

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

(Set value of timer register) ≠ 0
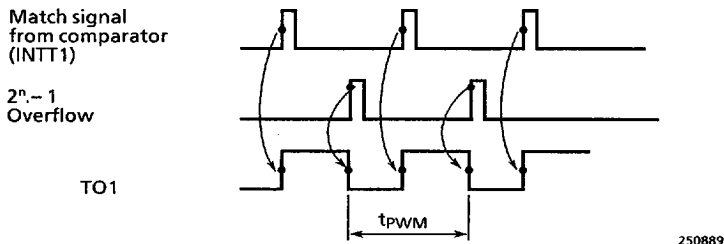
The PWM mode can be illustrated as follows:
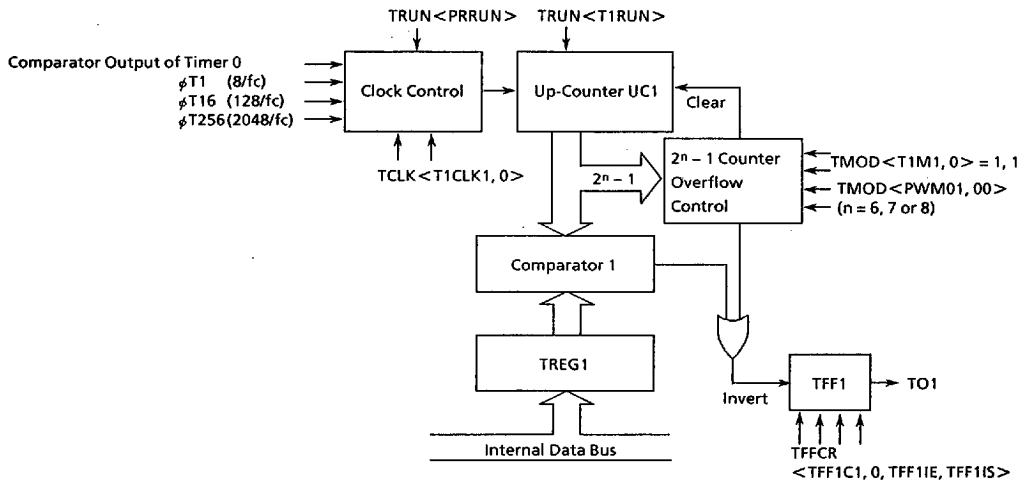
Figure 3.6 (12) Block Diagram of 8-bit PWM Mode

MCU90-314

9097249 0040770 856

Example : Generate the following PWM to the TO1 pin at fc = 10MHz.
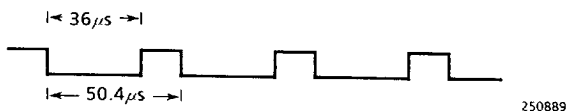
Assuming the PWM cycle is 50.4 μs when ϕT1 = 0.8 μs and @fc = 10MHz,

$50.4\mu s/0.8\mu s = 63 = 2^6 - 1$

Consequently, n should be set at 6 (TMOD < PWM01, 00 > = 0, 1).

Given the "L" level period of 36 μs, setting ϕT1 = 0.8 μs results:

$36\mu s/0.8\mu s = 45 = 2DH$

As a result, TREG1 should be set at 2DH.

TRUN ← - - - - - - 0 -	Stop Timer 1.
TCLK ← - - - - 0 1 - -	Select ϕT1 as the input clock.
TMOD ← - - - - 1 1 0 1	Set the $2^6 - 1$ cycle in the PWM mode.
TFFCR ← - - - - 0 0 1 1	Set the initial output to 0 ("L" level).
TREG1 ← 0 0 1 0 1 1 0 1	Write "2DH".
P3CR ← 1 1 - - - - - -	Select P37 as the TO1 pin.
TRUN ← - - 1 - - - 1 -	Start Timer 1.

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

Table 3.6 (3)   PWM Cycle and Selection of $2^n - 1$ counter

|	Expression	PWM cycle (@fc = 10MHz)		
		ϕT1 (8/fc)	ϕT16 (128/fc)	ϕT256 (2048/fc)
2^6-1	$(2^6$-1)$\times \phi$Tn	50.4μs	806.4μs	12.9ms
2^7-1	$(2^7$-1)$\times \phi$Tn	101.6μs	1625.6μs	26.0ms
2^8-1	$(2^8$-1)$\times \phi$Tn	204.0μs	3264.0μs	52.2ms

270890

Precautions for PWM output

TMP90C802 can output PWM by the 8 bit timer. However, changing the pulse width of PWM requires special care. This problem is explained by the following example.

Example : To output PWM by 8 bit timer

TREG1 :   Pulse width

Cycle :   Fixed ($2^6 - 1, 2^7 - 1, 2^8 - 1$)

9097249 0040771 792
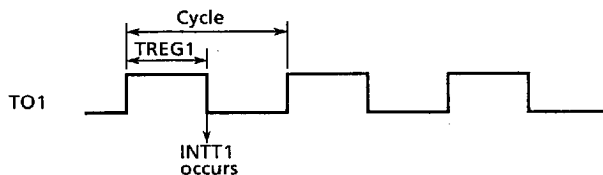
200990

In the PWM mode, INTT1 occurs at the coincidence with TREG1. However, the pulse width cannot be changed directly using this interrupt. (Depending on the value of TREG1 to be set, coincidence with TREG1 may be detected again in a single cycle, inverting the timer output.)

To eliminate this problem in changing the pulse width, it is effective to halt the timer with the INTT1 process, modify the value of TREG1, set the timer output to "1", and restart the timer. In the meantime, the output waveform loses shape when the pulse width is changed. This method is valid for a system that allows a deformed output waveform.

(5) A Table of All Timer Mode

Table 3.6 (4) Timer Mode

Register name	TMOD		TCLK		TFFCR
bit Symbol	T10M (T32M)	PWM1	T1CLK (T3CLK)	T0CLK (T2CLK)	FF1IS
Function	Timer mode	PWM cycle	Upper Input	Lower Input	Inversion select
16 bit Timer mode	01	—	—	ϕT1, 16, 256 (01, 10, 11)	1 (*)
8 bit Timer × 2ch	00	—	Comparator output from Lower timer, ϕT1, 16, 256 (00, 01, 10, 11)	ϕT1, 16, 256 (01, 10, 11)	0: Lower timer 1: Upper timer
8 bit PPG × 1ch	10	—	ϕT1, 16, 256 (01, 10, 11)	—	1
8 bit PWM × 1ch	11	$2^6 - 1, 2^7 - 1, 2^8 - 1$ (01, 10, 11)	ϕT1, 16, 256 (01, 10, 11)	—	1
8 bit Timer × 1ch		—	—	ϕT1, 16, 256 (01, 10, 11)	Impossible to output

031090

(Note) — : don't care
   * : It is possible to set to "0", when timer F/F is not used.

■ 9097249 0040772 b29 ■

3.6.2 8-bit Timer/Event Counter

(1) Event counter mode

Timer 2 has the 8-bit timer/event counter and can be used not only as the 8-bit timer but also as the counter. Timer 2 can be placed in the event counter mode by setting the input clock of Timer 2 as the external count input TI2. Timer 2 and Timer 3 can be used as an 8-bit counter and an 8-bit timer, respectively, and as a 16-bit counter through cascade connection. The counter is incremented at the rising edge of the counter input TI2. The counter input pin TI2 is also used for P81 / INT1 and has the zero-cross detection function. To use this pin as the counter input pin (TI2), set TCLK < T2CLK1, 0 > to "0, 0".

Example : Using the Timer 2 as the event counter.

```
          MSB              LSB
          7 6 5 4 3 2 1 0
  ┌ TRUN  ←  – – – – – 0 – –        Stop Timer 2.
  │ TMOD  ←  0 0 X X – – – –        Place Timer 2 in the 8-bit timer/counter mode.
  │ P8CR  ←  – – – – – – * –        *=0:  When input waveform of TI2 is pulse.
  │                                 *=1:  When input waveform of TI2 is sine wave.
  │                                        (Zero-cross detection)
  │ INTEL ←  1 – – – – – – –        Enable INTT2.
  │ TCLK  ←  – – 0 0 – – – –        Set the input clock of Timer 2 as the counter input TI2.
  │ TREG2 ←  * * * * * * * *        Set the count number.
  └ TRUN  ←  – – 1 – – 1 – –        Start Timer 2.
```

Note :   Place the prescaler in the "RUN" mode even if Timer 2 is used as the event counter.

X : Don't care   – : No change

(2) Software counter latch

In the event counter mode, the value of the up-counter can be read by software. When TFFCR<LATCH> is set to "0", the counter value at that time is loaded into the counter latch registers (TREG2 and TREG3). To read value, place the prescaler in the "RUN" mode (set TRUN<PRRUN> to "1").

Example : To latch the counter value every 40μs at a frequency of 10 MHz, set the registers as follows:

```
                MSB                LSB
                7 6 5 4 3 2 1 0
   TRUN         - - - - - 0 0 -          Stop Timer 1 and Timer 2 and clear them to "0".
   TMOD         0 0 X X 0 0 - -          Place Timer 1 in the 8-bit timer mode, and Timer 2 in
                                         the 8-bit timer/counter mode.
   TCLK         - - 0 0 0 1 - -          Set the input clock of Timer 1 as $\phi$T1 (0.8$\mu$s at
                                         fc = 10MHz), and Timer 2 as the counter input TI2.
   TREG1        0 0 1 1 0 0 1 0          Set 50 (40$\mu$s ÷ T1) in Timer register 1.
   TREG2        * * * * * * * *          Set Timer register 2.
   INTEH        - - - - - - - 1          Enable INTT1.
   TRUN         - - 1 - - 1 1 -          Start Timer 1 and Timer 2.
```

INTT 1 interrupt.

```
    TFFCR  ←   0 - - - - - - -           Latch the counter value.
```

The latched counter value can be read by reading Timer register 2.
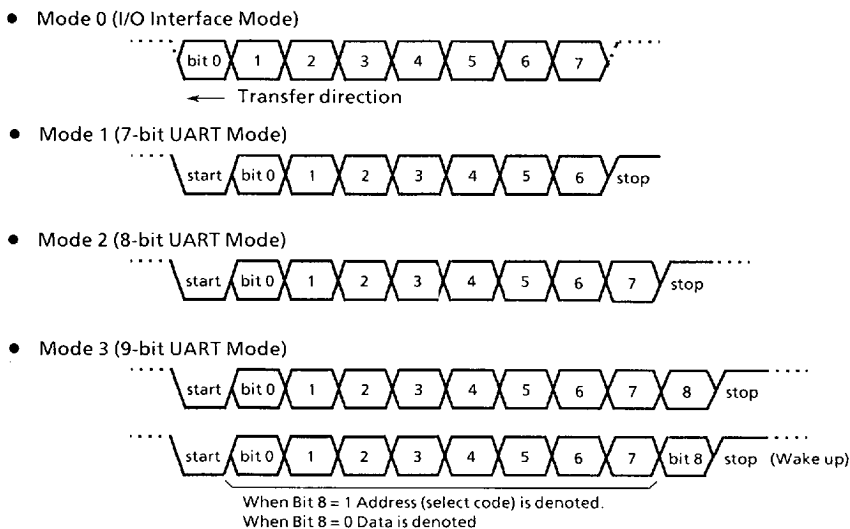

3.7   Serial Channel

The TMP90C802A incorporates a serial I/O channel for full-duplex asynchronous transmission (UART) and I/O expansion.

The serial channel has the following operating modes:

- ● I/O interface mode ——— Mode 0 :   Transmit/Receive I/O data for expand I/O and transmit its synchronous signals SCLK

- ● Asynchronous transmission (UART) mode

  - Mode1 :   7-bit data
  - Mode2 :   8-bit data
  - Mode3 :   9-bit data

The Mode 3 accommodates a wake-up function to start the slave controllers in a controller serial link (multi-controller system).

Figure 3.7 (1) shows the data format (1-frame data) in each mode.

- ● Mode 0 (I/O Interface Mode)

←— Transfer direction

- ● Mode 1 (7-bit UART Mode)

- ● Mode 2 (8-bit UART Mode)

- ● Mode 3 (9-bit UART Mode)

When Bit 8 = 1 Address (select code) is denoted.
When Bit 8 = 0 Data is denoted

250889

Figure 3.7 (1)   Data Formats

Data received and transmitted are stored temporarily into separate buffer registers to allow independent transmission and receiving (Full-duplex).

In the I/O interface mode, however, the data transfer is half-duplex due to the single SCLK (serial clock) pin is used for transmission and receiving.

■ 9097249 0040775 338 ■

The pin function (Port function or serial I/O function) is selected by the Port 3 control registers. For example, P31 can be used as the RxD pin by setting P3CR<RXDC> to **1**.

The receiving buffer register has a double-buffer structure to prevent overruns. The one buffer receives the next frame data while the other buffer stores the received 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 an request is issued to the CPU to transmit data after the transmitting buffer becomes empty or to read data after the receiving buffer completed to store data, the interrupt INTTX or INTRX occurs respectively. In receiving data, the occurrence of an overrun error, or framing error sets the flag SCCR<OERR, FERR> accordingly.

3.7.1 Control Registers

The serial channel is controlled by four control registers (SCMOD, SCCR, TRUN, and P3CR). The received/transmitted data are stored into SCBUF.
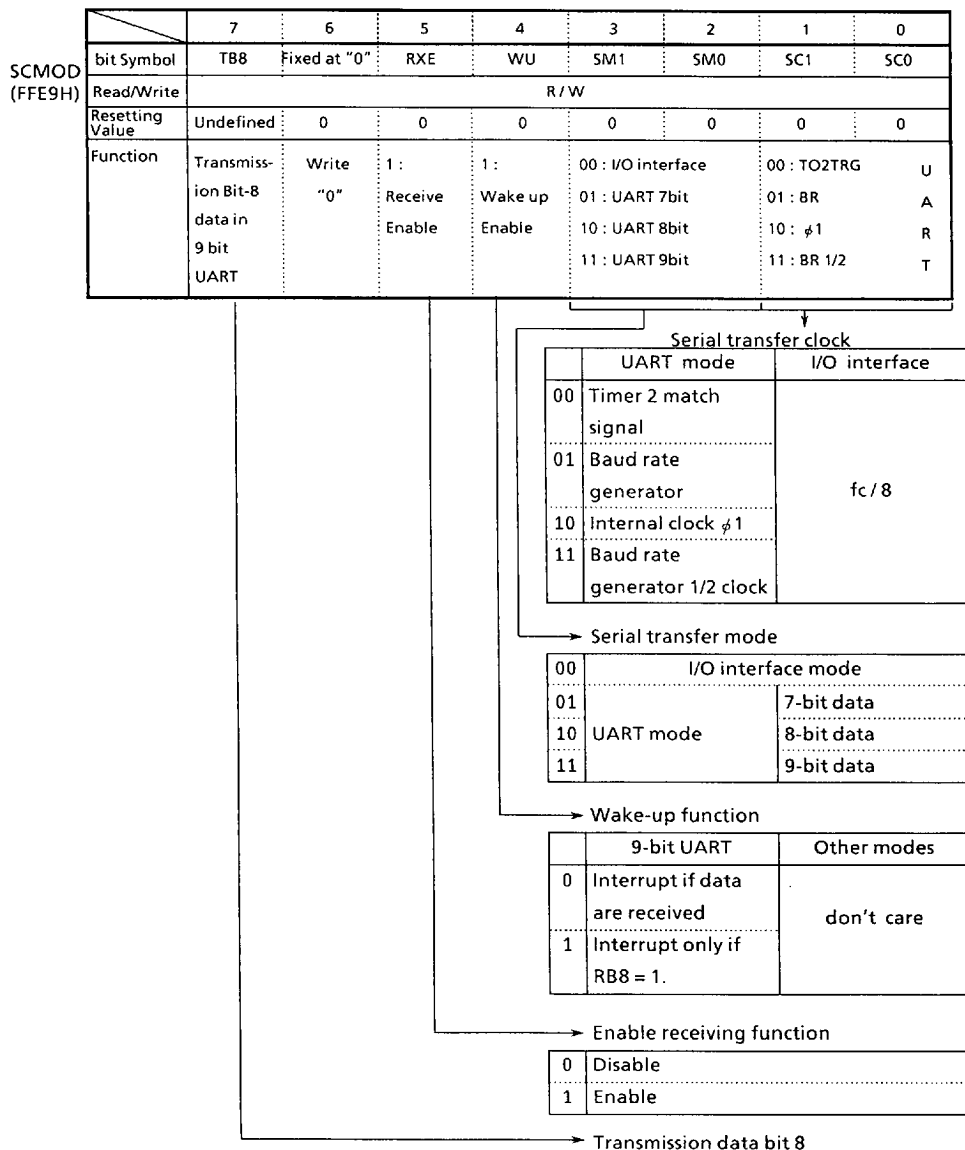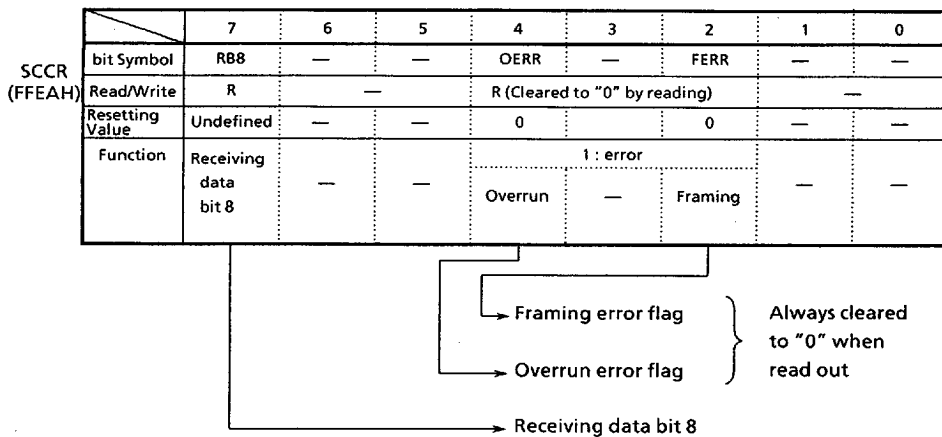
		7	6	5	4	3	2	1	0
SCMOD (FFE9H)	bit Symbol	TB8	Fixed at "0"	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write				R / W				
	Resetting Value	Undefined	0	0	0	0	0	0	0
	Function	Transmission Bit-8 data in 9 bit UART	Write "0"	1 : Receive Enable	1 : Wake up Enable	00 : I/O interface 01 : UART 7bit 10 : UART 8bit 11 : UART 9bit		00 : TO2TRG 01 : BR 10 : $\phi1$ 11 : BR 1/2	U A R T

Serial transfer clock

	UART mode	I/O interface
00	Timer 2 match signal	
01	Baud rate generator	fc / 8
10	Internal clock $\phi1$	
11	Baud rate generator 1/2 clock	

→ Serial transfer mode

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

→ Wake-up function

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

→ Enable receiving function

0	Disable
1	Enable

→ Transmission data bit 8

Figure 3.7 (2)   Serial Channel Mode Register

		7	6	5	4	3	2	1	0
SCCR (FFEAH)	bit Symbol	RB8	—	—	OERR	—	FERR	—	—
	Read/Write	R	—		R (Cleared to "0" by reading)			—	
	Resetting Value	Undefined	—	—	0		0	—	—
	Function	Receiving data bit 8	—	—	1 : error			—	—
					Overrun	—	Framing		

Framing error flag ⎫
Overrun error flag ⎬ Always cleared to "0" when read out
Receiving data bit 8 ⎭

Caution : Since all error flags are cleared after readout, avoid testing for only one bit using a bit-testing instruction

250889

Figure 3.7 (3) Serial Channel Control Register

	7	6	5	4	3	2	1	0		
SCBUF (FFEBH)	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)	(Write only)
	7	6	5	4	3	2	1	0		
	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)	(Read only)
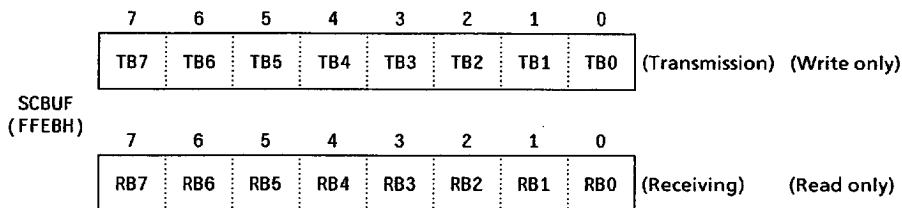
250889

Figure 3.7 (4) Serial Transmission/Receiving Buffer Registers

■ 9097249 0040778 047 ■

		7	6	5	4	3	2	1	0
TRUN (FFDBH)	bit Symbol	BRATE1	BRATE0	PRRUN	—	T3RUN	T2RUN	T1RUN	T0RUN
	Read/Write	R/W			R/W				
	Resetting Value	0	0	0	—	0	0	0	0
	Function	00: 300/150 bps 01: 1200/600 10: 4800/2400 11: 19200/9600 @ fc = 9.8304MHz		Prescaler & Timer Run / Stop Control 0: Stop & Clear 1: Run (Count up)					

Timer 2 function

0	Stop & clear
1	Count

Prescaler function

0	Stop & clear
1	Count

Select transfer speed of serial I/O baud rate generator

	SCMOD<SC1, 0> = 0, 1	SCMOD<SC1, 0> = 1, 1
00	300bps	150bps
01	1200bps	600bps
10	4800bps	2400bps
11	19200bps	9600bps

@fc = 9.8304MHz

Figure 3.7 (5) Timer/Serial Channel Operation Control Register

9097249 0040779 T83

	7	6	5	4	3	2	1	0
bit Symbol	WAITC1	WAITC0	RDE	ODE	TXDC	SCLK	RXDC	—
Read/Write	R / W		R / W	R / W	R / W	R / W	R / W	—
Resetting Value	0	0	0	0	0	0	0	—
Function	Wait control 00 : 2state wait 01 : normal wait 10 : non wait 11 : Timer 0/1 Output		RD control 0 : $\overline{RD}$ for only external access 1 : Always $\overline{RD}$	P33 control 0 : CMOS 1 : Open drain	P33 0 : Out 1 : TxD	P32 0 : Out 1 : SCLK	P31 0 : In 1 : RxD	—

P3CR (FFC7H)

Explained in "3.5.4 Port 3".

Select P31 function

0	Input port
1	RxD pin

Select P32 function

0	Output port
1	SCLK pin

Select P33 function

0	Output port
1	TxD pin

Select P33 CMOS/Open-drain output

0	CMOS output
1	Open-drain output

Figure 3.7 (6)   Port 3 Control Register

3.7.2 Architecture

Figure 3.7 (7) is a block diagram of the serial channel.
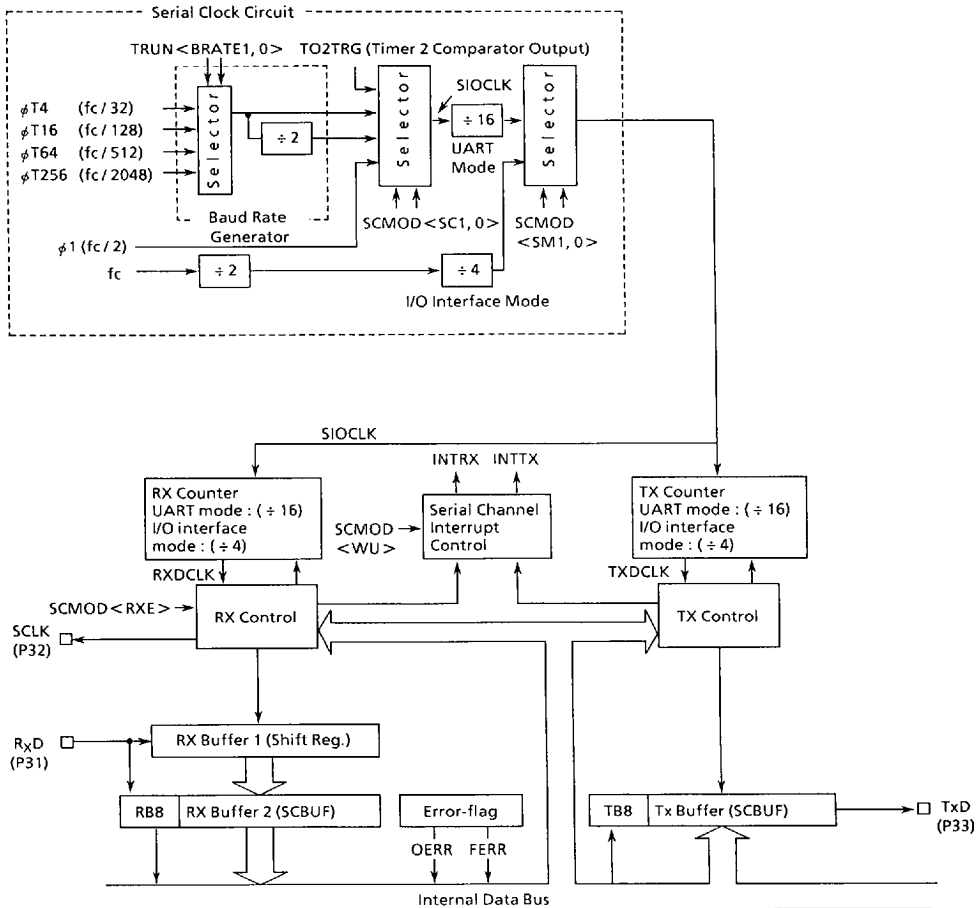
Figure 3.7 (7) Block Diagram of Serial Channel

250889

9097249 0040781 631

① Baud-rate generator

The baud-rate generator comprises a circuit that generates a clock pulse to determine the transfer speed for transmission/receiving in the asynchronous communication (UART) mode.

The input clock to the baud-rate generator ΦT4 (fc/32), ΦT16 (fc/128), ΦT64 (fc/512) or ΦT256 (fc/2048) is generated by the 9-bit prescaler. One of these input clocks is selected by the timer/serial channel control register TRUN<BRATE1, 0>.

Also, either no frequency division or 1/2 division can be selected by the serial channel mode register SCMOD<SC1, 0>.

Table 3.7 (1) shows the baud-rate when fc=9.8304 MHz.

Table 3.7 (2) shows the baud-rate when use timer 2 (input clock : ΦT1)

Talbe 3.7 (1)   Baud Rate Selection (1)   [bps]

<BRATE1, 0>	Input clock	No division (SC1, 0 = 01)	1/2 division (SC1, 0 = 11)
00	ΦT256 (fc/2048)	300	150
01	ΦT64 (fc/512)	1200	600
10	ΦT16 (fc/128)	4800	2400
11	ΦT4 (fc/32)	19200	9600

@fc = 9.8304MHz   250889

Table3.7 (2)   Baud Rate Selection (2) (When use Timer2)   [kbps]

fc / TREG2	12.288 MHz	12 MHz	9.8304 MHz	8 MHz	6.144 MHz
01H	96		76.8	62.5	48
02H	48		38.4	31.25	24
03H	32	31.25			16
04H	24		19.2		12
05H	19.2				9.6
08H	12		9.6		6
0AH	9.6				4.8
10H	6		4.8		3
14H	4.8				2.4

180790

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

Input clock of Timer 2

$$\Phi T1 = fc / 8$$
$$\Phi T16 = fc / 128$$
$$\Phi T256 = fc / 2048$$

MCU90-326

■ 9097249 0040782 578 ■

② Serial clock generating circuit

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

1) I/O interface mode

It generates a clock at a 1/8 frequency (1.25 M bit/s at 10 MHz) of the system clock (fc). This clock is output from the SCLK pin (also used as P32).

2) Asynchronous communication (UART) mode

A basic clock (SIOCLK) is generated based on the above baud rate generator clock, the internal clock $\Phi 1$ (fc/2) (SIOCLK = 5MHz, Transfer speed = 312.5kb.p.s. at 10MHz), or the match signal from Timer 2, as selected by SCMOD < SC1, 0 > register.

③ Receiving counter

The receiving counter is a 4-bit binary counter used in the asynchronous communication (UART) mode and is counted by using SIOCLK. 16 pulses of SIOCLK is used for receiving 1 bit data. The data are sampled three times at 7th, 8th and 9th pulses and evaluated by the rule of majority. For example, if data sampled at the 7th, 8th and 9th clock are "1", "0" and "1", the received data is evaluated as "1". The sampled data "0", "0" and "1" is evaluated that the received data is "0".

④ Receiving control

1) I/O interface mode

The RxD signal is sampled on the rising edge of the shift clock which is output to the SCLK pin.

2) Asynchronous communication (UART) mode

The receiving control features a circuit for detecting the start bit by the rule of majority. When two or more "0" are detected during 3 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.

■ ٩٥٩٧٢٤٩ ٥٥٤٥٧٨٣ ٤٥٤ ■

⑤ Receiving buffer

The receiving buffer has a double-buffer structure to prevent overruns. Received data are stored into the Receiving buffer 1 (shift register type) for each 1 bit. When 7 or 8 bits data are stored in the Receiving buffer 1, the stored data is transferred to the Receiving buffer 2 (SCBUF), and the interrupt INTRX occurs at the same time. The CPU reads out the Receiving buffer 2 (SCBUF). Data can be stored into the Receiving buffer 1 before the CPU reads out the Receiving buffer 2 (SCBUF).

Note, however, that an overrun occurs unless the CPU reads out the Receiving buffer 2 (SCBUF) before the Receiving buffer 1 receives all bits of the next data.

When an overrun occurred, the data in the buffer 2 and SCCR<RB8> are not lost, however, that in the buffer 1 are lost.

SCCR<RB8> stores the MSB in the 9-bit UART mode.

In the 9-bit UART mode, setting SCMOD<WU> to "1" enables the wake-up function of the slave controllers, and the interrupt INTRX occurs only if SCCR<RB8> =1.

⑥ Transmission counter

This is a 4-bit binary counter used in the asynchronous communication (UART) mode. Like the receiving counter, it counts based on SIOCLK to generate a transmission clock TXDCLK for every 16 counts.
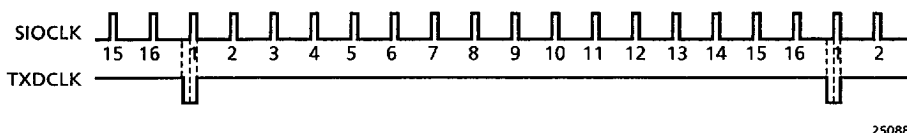
250889

⑦ Transmission control

1) I/O interface mode

Data in the transmission buffer are output to the TxD pin bit by bit at the rising edge of the shift clock output from the SCLK pin.

2) Asynchronous communication (UART) mode

When the CPU have written data into the transmission buffer, transmission is started with the next rising edge of TxDCLK, and a transmission shift clock TxDSFT is generated.

■ 9097249 0040784 340 ■

⑧ Transmission buffer

The transmission buffer SCBUF shifts out the data written by the CPU from the LSB as based on the shift clock TXDSFT (Same period as TXDCLK) generated by the transmission control unit. When all bits are shifted out, the transmission buffer becomes empty, generating the interrupt INTTX.

⑨ Error flag

There error flags are prepared to increase the reliability of received data.

1) Overrun error (SCCR<OERR>)

Overrun error occurs if all the bits of the next data are received by the receiving buffer 1 while valid data are still stored in the receiving buffer 2 (SCBUF).

2) Framing error (SCCR<FERR>)

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

⑩ Generation Timing

1) UART mode

Receiving

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

Note : The occurrence of a framing error is delayed until after interruption. Therefore, to check for framing error during interrupt operation, an additional operation, such as waiting for 1 bit time, becomes necessary.

Transmitting

mode	9 Bit	8 Bit + Parity	8 Bit, 7 Bit + Parity, 7 Bit
Interrupt timing	Just before the stop bit	←	←

070989

2) I/O interface mode

Interrupt timing of receiving	Just after the last SCLK rising ⌐Ⴑ
Interrupt timing of transmitting	↑

210689

MCU90-329

3.7.3 Operation

(1)   Mode 0 (I/O interface mode)

This mode is used to increase the number of I/O pins of the TMP90C802A.

The TMP90C802A supplies the transmitting/receiving data and a synchronous clock (SCLK) to an external shift register.
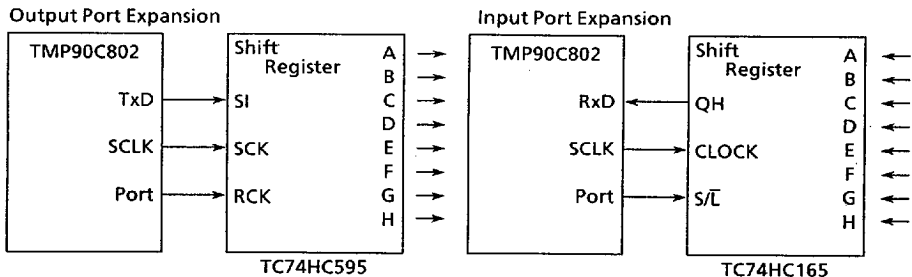
Output Port Expansion   Input Port Expansion

Figure 3. 7 (8)   I/O Interface Mode

① Transmission

Each time the CPU writes data into the transmission buffer, 8-bit data are output from TxD pin. When all data are output, IRFH<IRFTX> is set, and the interrupt INTTX occurs.
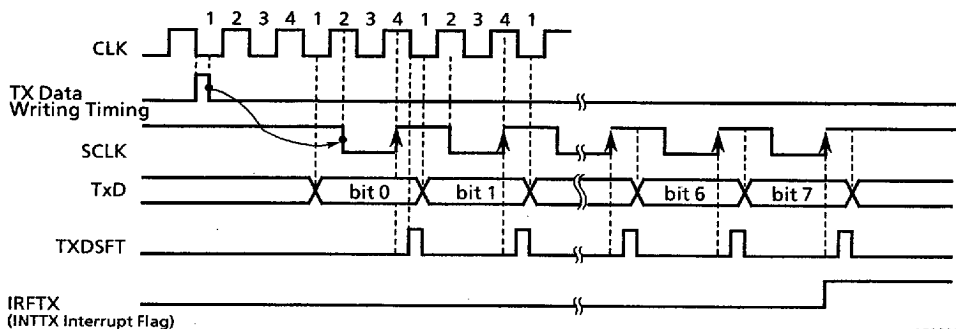
Figure 3.7 (9)   Transmitting Operation (I/O Interface Mode)

Example : When transmitting data from P33 pin, the control registers should be set as described below.

```
P3CR   ← - - - - 1 1 0 0        Select P32 as the SCLK pin, and P33 as the TXD pin.
SCMOD  ← X 0 0 0 0 0 X X        Set I/O interface Mode.
INTEL  ← - - - - - - - 1        Enable INTTX Mode.
SCBUF  ← * * * * * * * *        Set data for transmission.
```

Note : ✕ ; don't care   - ; no change

MCU90-330

② Receiving

Each time the CPU reads the receiving data and clears the receiving interrupt flag IRFH<IRFRX>, the next data are shifted into the receiving buffer 1. When 8-bit data are received, the data are transferred to the receiving buffer 2 (SCBUF), which sets IRFRX and generates interrupt INTRX.

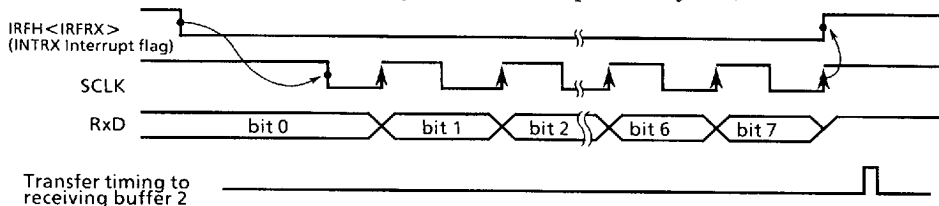For receiving data, the receiving enable state is previously set (SCMOD<RxE>=1).

180790

Figure 3.7 (10)   Receiving Operation (I/O Interface Mode)

Example : When receiving from P31 pin, the control registers should be set as described below.

```
P3CR   ← - - - - 0 1 1 X        Select P32 as the SCLK pin, and P31 as the RxD pin.
SCMOD  ← X 0 0 0 0 0 X X        Set I/O Interface Mode.
INTEL  ← - - - - - - 1 -        Enable INTRX interrupt.
SCMOD  ← - - 1 - - - - -        Set RxE to "1".
```
(注)   × ; don't care   - ; no change

240890

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

The 7-bit UART mode is selected by setting the serial channel mode register SCMOD<SM1, 0> to "01".

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

(Baud Rates 2400 bps @fc = 9.8304MHz)
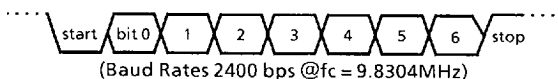
←——— Transfer direction   250889

```
P3CR   ← - - - - 1 - - -        Select P33 as the TxD pin.
SCMOD  ← X 0 - X 0 1 1 1        Set the transfer speed at 2,400 baud in
TRUN   ← 1 0 1 - - - - -        the 7-bit UART mode.
INTEL  ← - - - - - - - 1        Enable INTTX interrupt.
SCBUF  ← * * * * * * * *        Set data for transmission
```
   (Note)   X : Don't care   - : No change

MCU90-331

■ 9097249 0040787 05T ■

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

The 8-bit UART mode is selected by setting SCMOD<SM1, 0> to "1, 0".

Example: When receiving data with the following format, the control registers should be set as described below.

(Baud Rates 9600 bps @fc = 9.8304MHz)

← Transfer direction   250889
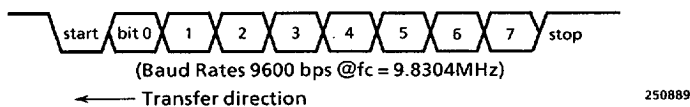
Main setting :

```
┌ P3CR  ← - - - - - - 1 -        Select P31 as the RxD pin.
│ TRUN  ← 1 1 1 - - - - -        Set the transfer speed at 9,600 bps in
│ SCMOD ← - 0 1 X 1 0 1 1        the 8-bit UART mode.
└ INTEL ← - - - - - - 1 -        Enable INTTX interrupt.
```

INTRX processing :

```
┌ Acc  ← SCCR  Λ  00010100       Check errors.
│ if Acc ≠  0 then error
└ Acc  ← SCBUF                   Read out the received data.
```

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

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

The 9-bit UART mode is selected by setting SCMOD<SM1, 0> to "11".

The MSB (9th bit) is written into SCMOD<TB8> for transmission, and into SCCR<RB8> for receiving. Writing into or reading from the buffer must begin with the MSB (9th bit) followed by SCBUF.

<u>Wake-up function</u>

In the 9-bit UART mode, setting SCMOD<WU> to "1" allows the wake-up operation as the slave controllers. The interrupt INTRX occurs only when SCCR<RB8> = 1.
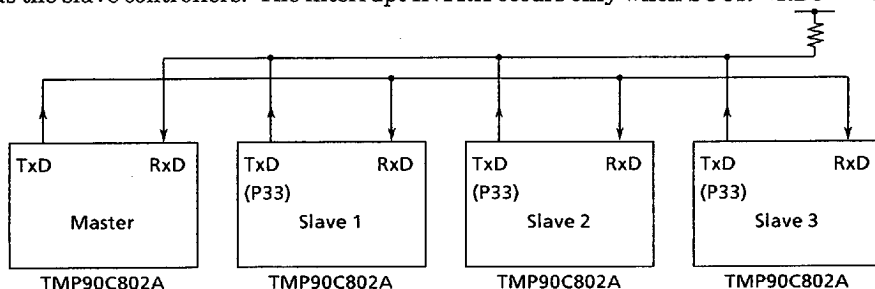

Note : For the wake-up operation, P33 should be always selected as the TxD pin of the slave controllers, and put in the open drain output mode.

250889

Figure 3.7 (11)   Serial Link Using Wake-up Function

MCU90-332

■ 9097249 0040788 T96 ■

<div style="border:1px solid">Protocol</div>

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

②   Set the SCMOD<WU> bit of each slave controller to "1" to enable data receiving.

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

Select code of slave controller.   "1"
                                                                250889
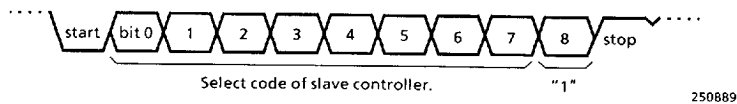
④   Each slave controller receives the above frame, and clears the <WU> bit to "0" if the above select code matches its own select code.

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

Data   "0"
   250889
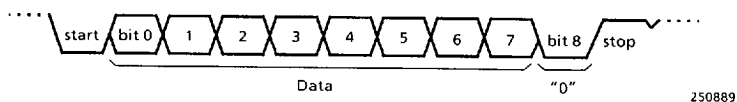
⑥   The other slave controllers (with the SCMOD<WU> bit remaining at "1") ignore the receiving data because their MSBs SCCR<RB8> are set to "0" to disable the interrupt INTRX.

When the <WU> bit is cleared to "0", the interrupt INTRX occurs, making it possible to read the receiving data.

The slave controllers (<WU> = 0) transmits data to the master controller, and it is possible to indicate the end of data receiving to the master controller by this transmission.

Example : Link two slave controllers serially with the master controller, and use the internal clock $\phi 1$ (fc/2) as the transfer clock.

                                                                250889
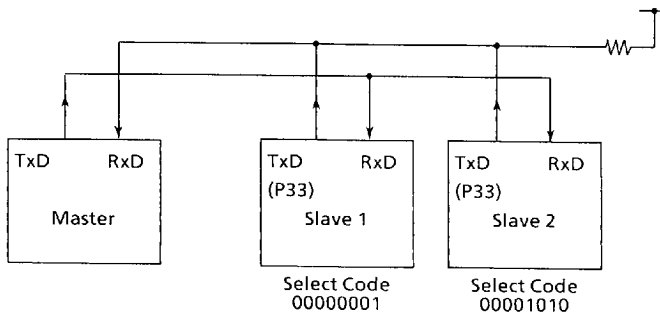
MCU90-333

■ 9097249 0040789 922 ■

● Set the master control

  Main

  ⎡ P3CR   ← - - - 0 1 0 1 X   Select P33 as TxD pin and P31 as RxD pin.
  ⎢ INTEL ← - - - - - - 1 1   Enable INTRX and INTTX.
  ⎢ SCMOD ← 1 0 1 0 1 1 1 0   Select $\phi 1$ (fc/2) as the transfer clock in the 9-bit UART
  ⎢   mode.
  ⎣ SCBUF ← 0 0 0 0 0 0 0 1   Set the select code for the slave controller 1.

  INTTX interrupt

  ⎡ SCMOD ← 0 - - - - - - -   Set SCMOD<TB8> to "0".
  ⎣ SCBUF ← * * * * * * * *   Set data for transmission.

● Set the slave 2

  Main

  ⎡ P3CR   ← - - - 1 1 0 1 X   Select P33 as TxD pin and P31 as RxD pin.
  ⎢ 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 mode
                                   (transfer clock $\phi 1$ (fc/2)).

  INTRX interrupt

  ⎡ Acc   ← SCBUF
  ⎢ if Acc = select code
  ⎣ then SCMOD ← - - - 0 - - - -   Clear <WU> to "0".

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

■ 9097249 0040790 644 ■

3.8   Watchdog Timers (Runaway Detecting Timer)

When the malfunction (runaway) of the CPU occurs due to any cause such as noise, the watchdog timer (WDT) detects it to return to the normal state. When WDT has detected malfunction, a non-maskable interrupt is generated to indicate it to the CPU.

3.8.1 Architecture

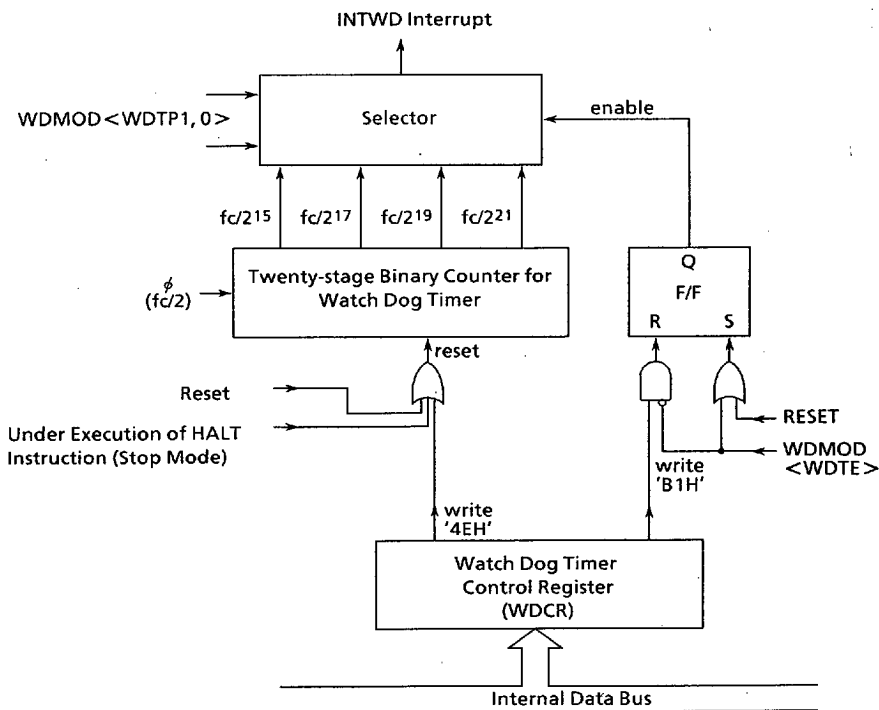Figure 3.8 (1) is a block diagram of the watchdog timer (WDT).

The watchdog timer consists of a 20-stage binary counter (input clock: ϕ @fc/2), a flip-flop that disables/enables the selector, a selector that selects one of the four output clocks generated from the binary counter, and two control registers.

The watchdog timer generates INTWD (watchdog timer interrupt) after a time specified by the register WDMOD<WDTP1, 0>. The binary counter for the watchdog timer is cleared to "0" by software (instruction) before the interrupt occurs. If the CPU caused a malfunction (runaway) for reason such as noise and fails to execute the instruction to clear the watchdog timer, the counter will overflow and the watchdog timer interrupt INTWD occurs. The CPU detects the malfunction (runaway) by this interrupt and is possible to be recovered to the normal state by the software for malfunction.

The watchdog timer starts its operation as the reset state is cleared

The watchdog timer stops its operation only in the STOP mode. When the STOP mode is released, the watchdog timer starts its operation after a specified warming-up time.

In the other standby mode (IDLE1, IDLE2 or RUN modes), the watchdog timer is enabled. However, the function can be disabled before entering any of these modes.
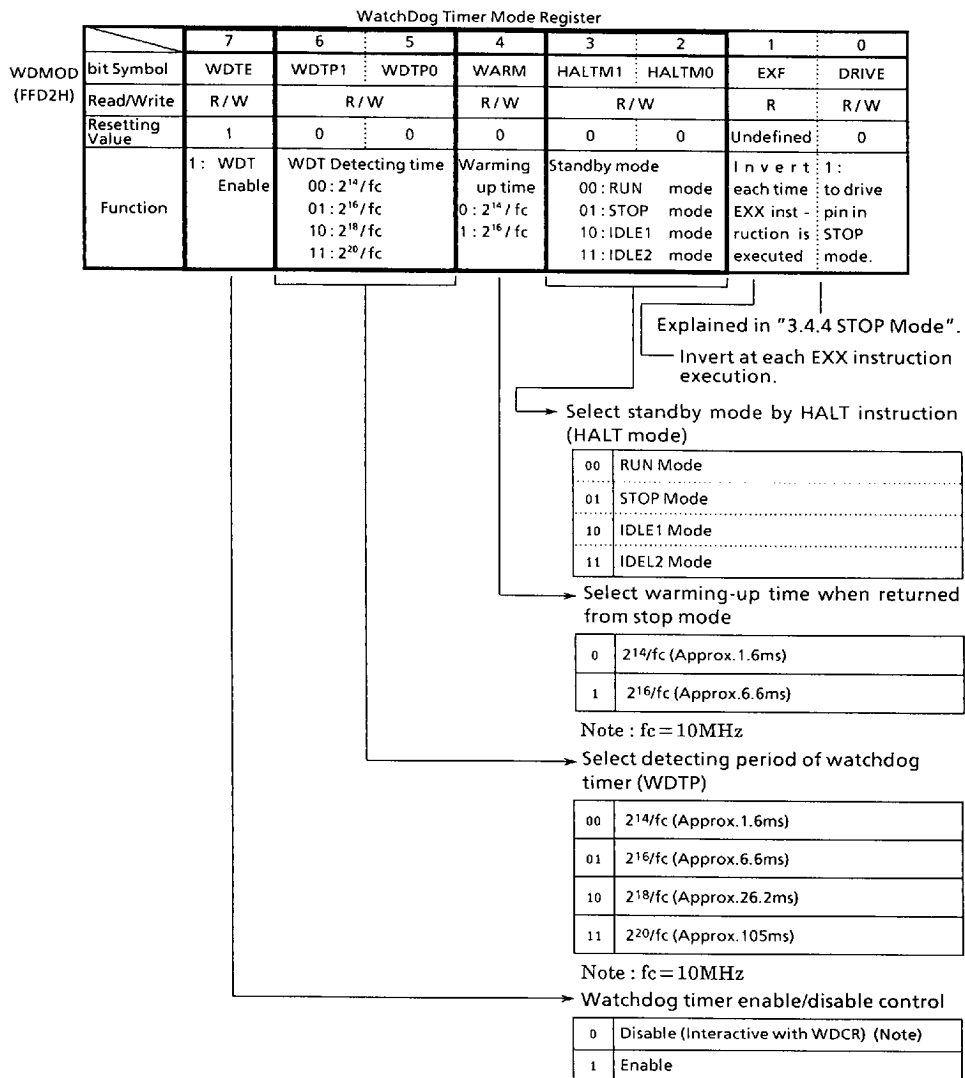
INTWD Interrupt

WDMOD<WDTP1, 0>   Selector   enable

fc/2^{15} fc/2^{17} fc/2^{19} fc/2^{21}

ϕ
(fc/2)   Twenty-stage Binary Counter for
Watch Dog Timer

Q
F/F
R   S

reset

Reset

Under Execution of HALT
Instruction (Stop Mode)

RESET

WDMOD
<WDTE>

write
'B1H'

write
'4EH'

Watch Dog Timer
Control Register
(WDCR)

Internal Data Bus

250889

Figure 3.8 (1)   Block Diagram of Watchdog Timer

■ 9097249 0040792 417 ■

3.8.2 Control Registers

WDT is controlled by two control registers (WDMOD and WDCR). Figure 3.8 (2) indicates the registers related to WDT.

WatchDog Timer Mode Register

		7	6	5	4	3	2	1	0
WDMOD (FFD2H)	bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
	Read/Write	R/W	R/W		R/W	R/W		R	R/W
	Resetting Value	1	0	0	0	0	0	Undefined	0
	Function	1: WDT Enable	WDT Detecting time $00:2^{14}/fc$ $01:2^{16}/fc$ $10:2^{18}/fc$ $11:2^{20}/fc$		Warming up time $0:2^{14}/fc$ $1:2^{16}/fc$	Standby mode $00:RUN$ mode $01:STOP$ mode $10:IDLE1$ mode $11:IDLE2$ mode		Invert each time EXX inst - ruction is executed	1: to drive pin in STOP mode.

Explained in "3.4.4 STOP Mode".

Invert at each EXX instruction execution.

Select standby mode by HALT instruction (HALT mode)

00	RUN Mode
01	STOP Mode
10	IDLE1 Mode
11	IDEL2 Mode

Select warming-up time when returned from stop mode

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

Note : $fc=10MHz$

Select detecting period of watchdog timer (WDTP)

00	$2^{14}/fc$ (Approx.1.6ms)
01	$2^{16}/fc$ (Approx.6.6ms)
10	$2^{18}/fc$ (Approx.26.2ms)
11	$2^{20}/fc$ (Approx.105ms)

Note : $fc=10MHz$

Watchdog timer enable/disable control

0	Disable (Interactive with WDCR) (Note)
1	Enable

Note : To disable, it is necessary to also write the disable code to the WDCR register. Disabling is not possible by writing to this register alone.
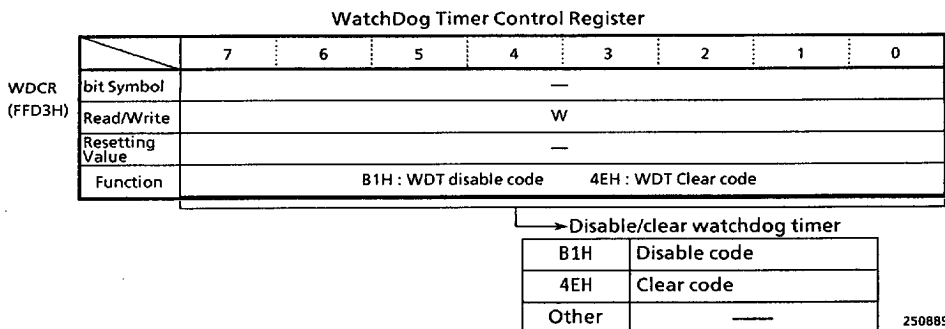
250889

Figure 3.8 (2) Watchdog Timer Mode Register

■ 9097249 0040793 353 ■

WatchDog Timer Control Register

WDCR (FFD3H)		7	6	5	4	3	2	1	0
	bit Symbol	—							
	Read/Write	W							
	Resetting Value	—							
	Function	B1H : WDT disable code 4EH : WDT Clear code							

→ Disable/clear watchdog timer

B1H	Disable code
4EH	Clear code
Other	——

250889

Figure 3.8 (3) Watchdog Timer Control Register

3.8.3 Operation

(1) Watchdog timer mode register (WDMOD)

① Set the detecting time of watchdog timer WDMOD < WDTP1, 0 >

The WDT interrupt period is set by this 2-bit registers. WDMOD < WDTP1, 0 > is initialized to "00" by resetting, providing the initial set value of 2^{14}/fc (s.) (approx. 8,192 states).

② The WDT enable/disable control WDMOD < WDTE >

< WDTE > is initialized to "1" by resetting, which enables the watchdog timer function.

To disable the function, the bit should be cleared to "0" and the disable code "B1H" should be written into the Watchdog Timer Control register. WDCR By using this dual procedure, it becomes hard to disable the WDT even if the malfunction occurs.

The disable state can be returned to the enable state easily by setting < WDTE > to "1".

(2) Watchdog timer control register (WDCR)

This register is used to disable the watchdog timer function or clear the binary counters.

① Disable Watch Dog Timer

The watchdog timer can be disabled by, after clearing WDMOD < WDTE > to "0", writing the disable code (B1H) into this WDCR register.

```
WDMOD ← 0 - - - - - X X    Clear WDTE to " 0 ".
WDCR  ← 1 0 1 1 0 0 0 1    Write disable code (B1H).
```

MCU90-338

◼ 9097249 0040794 29T ◼

② Clear Watch Dog Timer

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

WDCR ← 0 1 0 0 1 1 1 0   Write clear code (4EH).

Example :   ① Clear the binary counter.
WDCR   ← 0 1 0 0 1 1 1 0   Write clear code (4EH)

② Set 2^{16}/fc for the detecting time of watchdog timer.
WDMOD ← 1 0 1 – – – X X

③ Disable the watchdog timer.
WDMOD   ← 0 – – – – – X X   Clear WDTE to "0"
WDCR   ← 1 0 1 1 0 0 0 1   Write disable code (B1H)

④ Select the IDLE 2 mode.
WDMOD   ← 0 – – – 1 1 X X   Disable WDT and set IDLE 2 mode
WDCR   ← 1 0 1 1 0 0 0 1
Execute HALT instruction . Falling into the standby mode.

⑤ Select the STOP mode (Warming-up time: 2^{16}/fc)
WDMOD   ← – – – 1 0 1 X X   Select STOP mode
Execute HALT instruction. Falling into the standby mode.

4. ELECTRICAL CHARACTERISTICS

TMP90C802AP/TMP90C802AM
TMP90C803AP/TMP90C803AM

4.1 Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
V_{CC}	Supply voltage	$-0.5 \sim +6.5$	V
V_{IN}	Input voltage	$-0.5 \sim V_{CC} + 0.5$	V
P_D	Power dissipation (Ta = 85°C)	250	mW
T_{SOLDER}	Soldering temperature (10 s)	260	℃
T_{STG}	Storage temperature	$-65 \sim 150$	℃
T_{OPR}	Operating temperature	$-40 \sim 85$	℃

250889

4.2 DC Characteristics

Vcc = 5V ± 10% TA = −40~85°C (1~10MHz)
TA = −20~70°C (1~12.5MHz)

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{IL}	Input Low Voltage (P0)	-0.3	0.8	V	
V_{IL1}	P1, P2, P3, P8	-0.3	0.3Vcc	V	
V_{IL2}	$\overline{RESET}$, INT0, $\overline{NMI}$	-0.3	0.25Vcc	V	
V_{IL3}	$\overline{EA}$	-0.3	0.3	V	
V_{IL4}	X1	-0.3	0.2Vcc	V	
V_{IH}	Input High Voltage (P0)	2.2	Vcc + 0.3	V	
V_{IH1}	P1, P2, P3, P8	0.7Vcc	Vcc + 0.3	V	
V_{IH2}	$\overline{RESET}$, INT0, $\overline{NMI}$	0.75Vcc	Vcc + 0.3	V	
V_{IH3}	$\overline{EA}$	Vcc − 0.3	Vcc + 0.3	V	
V_{IH4}	X1	0.8Vcc	Vcc + 0.3	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 1.6mA
V_{OH} V_{OH1} V_{OH2}	Output High Voltage	2.4 0.75Vcc 0.9Vcc		V V V	I_{OH} = − 400μA I_{OH} = − 100μA I_{OH} = − 20μA
I_{DAR}	Darlington Drive Current (8 I/O pins) (Note)	− 1.0	− 3.5	mA	V_{EXT} = 1.5V R_{EXT} = 1.1 kΩ
I_{LI}	Input Leakage Current	0.02 (Typ)	± 5	μA	0.0 ≤ Vin ≤ Vcc
I_{LO}	Output Leakage Current	0.05 (Typ)	± 10	μA	0.2 ≤ Vin ≤ Vcc − 0.2
I_{CC}	Operating Current (RUN) Idle 1 Idle 2	17 (Typ) 1.5 (Typ) 6 (Typ)	30 5 15	mA mA mA	tosc = 10MHz (25%Up @12.5MHz)
	STOP (TA = −40~85°C) STOP (TA = 0~50°C)		50 10	μA μA	0.2 ≤ Vin ≤ Vcc − 0.2
V_{STOP}	Power Down Voltage (@STOP)	2 RAM BACK UP	6	V	V_{IL2} = 0.2Vcc, V_{IH2} = 0.8Vcc
R_{RST}	$\overline{RESET}$ Pull Up Registor	50	150	kΩ	
CIO	Pin Capacitance		10	pF	testfreq = 1MHz
V_{TH}	Schmitt width $\overline{RESET}$, $\overline{NMI}$, INT0	0.4	1.0 (Typ)	V	

Note : I_{DAR} is guaranteed for a total of up to 8 optional ports.

070989

MCU90-340

9097249 0040796 062

4.3 AC Characteristics

Vcc = 5V ± 10%   TA = − 40~85°C (1~10MHz)
CL = 50pF   TA = − 20~70°C (1~12.5MHz)

Symbol	Parameter	Variable		10MHz Clock		12.5MHz Clock		Unit
		Min	Max	Min	Max	Min	Max	
t_{OSC}	OSC. Period = x	80	1000	100		80		ns
t_{CYC}	CLK Period	4x	4x	400		320		ns
t_{WL}	CLK Low width	2x − 40		160		120		ns
t_{WH}	CLK High width	2x − 40		160		120		ns
t_{AC}	Address Setup to $\overline{RD}$, $\overline{WR}$	x − 45		55		35		ns
t_{RR}	$\overline{RD}$ Low width	2.5x − 40		210		160		ns
t_{CA}	Address Hold Time After $\overline{RD}$, $\overline{WR}$	0.5x − 30		20		10		ns
t_{AD}	Address to Valid Data In		3.5x − 95		255		185	ns
t_{RD}	$\overline{RD}$ to Valid Data In		2.5x − 80		170		120	ns
t_{HR}	Input Data Hold After $\overline{RD}$	0		0		0		ns
t_{WW}	$\overline{WR}$ Low width	2.5x − 40		210		160		ns
t_{DW}	Data Setup to $\overline{WR}$	2x − 50		150		110		ns
t_{WD}	Data Hold After $\overline{WR}$	30	90	30	90	30	90	ns
t_{CWA}	$\overline{RD}$, $\overline{WR}$ to Valid $\overline{WAIT}$		1.5x − 100		50		20	ns
t_{AWA}	Address to Vaild $\overline{WAIT}$		2.5x − 130		120		70	ns
t_{WAS}	$\overline{WAIT}$ Setup to CLK	70		70		70		ns
t_{WAH}	$\overline{WAIT}$ Hold After CLK	0		0		0		ns
t_{RV}	$\overline{RD}$, $\overline{WR}$ Recovery Time	1.5x − 35		115		85		ns
t_{CPW}	CLK to Port Data Output		x + 200		300		280	ns
t_{PRC}	Port Data Setup to CLK	200		200		200		ns
t_{CPR}	Port Data Hold After CLK	100		100		100		ns
t_{CHCL}	$\overline{RD}$ / $\overline{WR}$ Hold After CLK	x − 60		40		20		ns
t_{CLC}	$\overline{RD}$ / $\overline{WR}$ Setup to CLK	1.5x − 50		100		70		ns
t_{CLHA}	Address Hold After CLK	1.5x − 80		70		40		ns
t_{ACL}	Address Setup to CLK	2.5x − 80		170		120		ns
t_{CLD}	Data Setup to CLK	x − 50		50		30		ns

250889

- AC output level High 2.2V/Low 0.8V
- AC input level   High 2.4V/Low 0.45V (D0~D7)
    High 0.8Vcc/Low 0.2Vcc (excluding D0~D7)

9097249 0040797 TT9

4.4 Zero- Cross Characteristics

Vcc = 5V ± 10% TA = − 40~85°C (1~10MHz)
TA = − 20~70°C (1~12.5MHz)

Symbol	Parameter	Condition	Min	Max	Unit
Vzx	Zero- cross detection input	AC coupling C = 0.1μF	1	1.8	VAC p − p
Azx	Zero- cross accuracy	50/60Hz sine wave		135	mV
Fzx	Zero- cross detection input frequency		0.04	1	kHz

210689

4.5 Serial Channel Timing – I/O Interface Mode

Vcc = 5V ± 10% TA = − 40~85°C (1~10MHz)
CL = 50pF TA = − 20~70°C (1~12.5MHz)

Symbol	Parameter	Variable		10MHz Clock		12.5MHz Clock		Units
		Min	Max	Min	Max	Min	Max	
t_{SCY}	Serial Port Clock Cycle Time	8x		800		640		ns
t_{OSS}	Output Data Setup SCLK Rising Edge	6x − 150		450		330		ns
t_{OHS}	Output Data Hold After SCLK Rising Edge	2x − 120		80		40		ns
t_{HSR}	Input Data Hold After SCLK Rising Edge	0		0		0		ns
t_{SRD}	SCLK Rising Edge to Input DATA Valid		6x − 150		450		330	ns

210689

4.6 8-bit Event Counter

Vcc = 5V ± 10% TA = − 40~85°C (1~10MHz)
TA = − 20~70°C (1~12.5MHz)

Symbol	Parameter	Variable		10MHz Clock		12.5MHz Clock		Units
		Min	Max	Min	Max	Min	Max	
t_{VCK}	TI2 clock cycle	8x + 100		900		740		ns
t_{VCKL}	TI2 Low clock pulse width	4x + 40		440		360		ns
t_{VCKH}	TI2 High clock pulse width	4x + 40		440		360		ns

250889

4.7 Interrupt Operation

Vcc = 5V ± 10% TA = − 40~85°C (1~10MHz)
TA = − 20~70°C (1~12.5MHz)

Symbol	Parameter	Variable		10MHz Clock		12.5MHz Clock		Units
		Min	Max	Min	Max	Min	Max	
t_{INTAL}	$\overline{NMI}$, INT0 Low level pulse width (⎤⎣)	4x		400		320		ns
t_{INTAH}	$\overline{NMI}$, INT0 High level pulse width (⎣⎤)	4x		400		320		ns
t_{INTBL}	INT1 Low level pulse width (⎤⎣)	8x + 100		900		740		ns
t_{INTBH}	INT1 High level pulse width (⎣⎤)	8x + 100		900		740		ns

250889

■ 9097249 0040798 935 ■

4.8   I/O Interface Mode Timing Chart
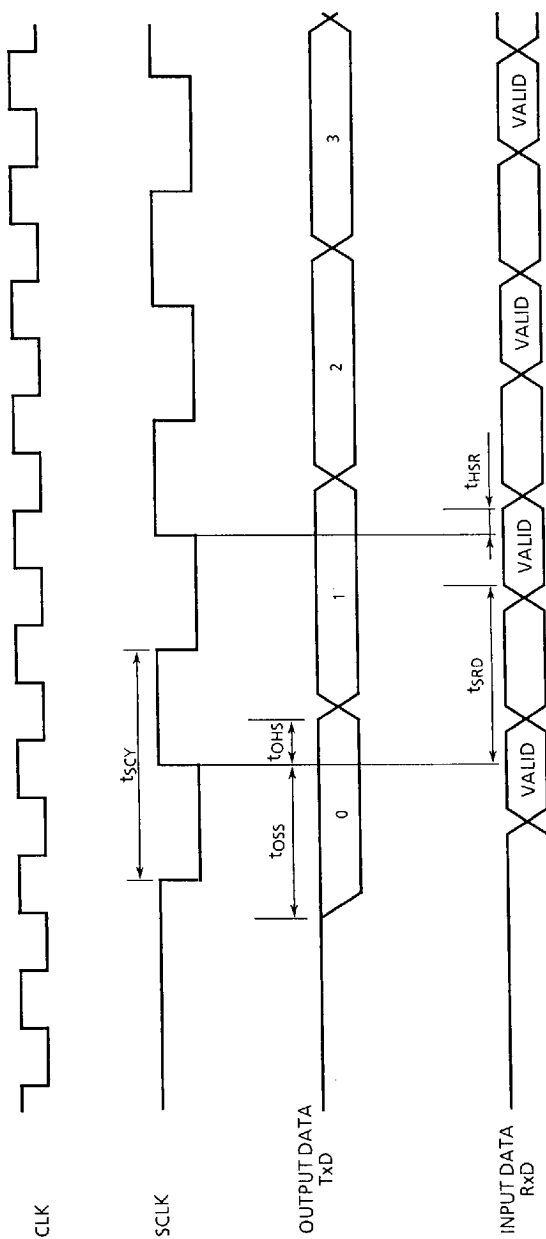
9097249 0040799 871

4.9 Timing Chart

CLK

A0~15

$\overline{RD}$

Read Data

$\overline{WR}$

Write Data

$\overline{WAIT}$

Port Output

Port Input

250889

MCU90-344

9097249 0040800 313
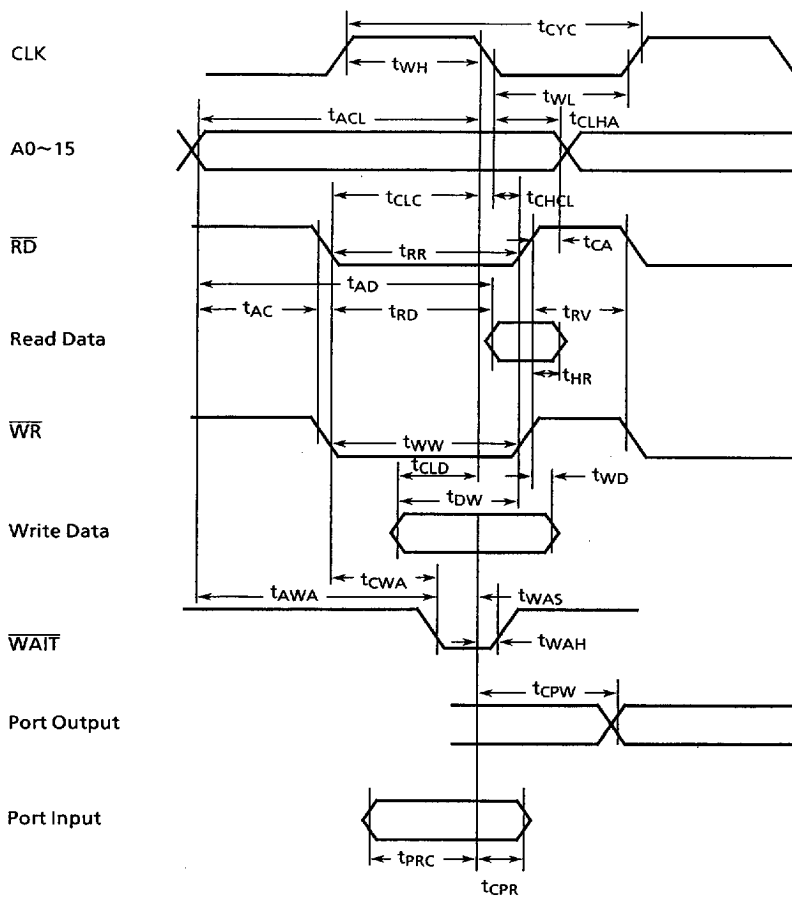
5. TABLE OF SPECIAL FUNCTION REGISTERS

The special function registers include the I/O ports, peripheral control registers allocated to the 48-byte addresses from FFC0H to FFEFH.

Format of table

Symbol	Name	Address	7	6		1	0	
								→ bit Symbol
								→ Read/Write
	.							→ Resetting value
								→ Function

TMP90C802 Special Function Register Address List

Address	Symbol	Address	Symbol	Address	Symbol
FFC0	P0	FFD0	P8	FFE0	—
FFC1	P1	FFD1	P8CR	FFE1	—
FFC2	P01CR (IRFL)	FFD2	WDMOD	FFE2	—
FFC3	IRFH	FFD3	WDCR	FFE3	—
FFC4	P2	FFD4	TREG0	FFE4	—
FFC5	P2CR	FFD5	TREG1	FFE5	—
FFC6	P3	FFD6	TREG2	FFE6	INTEL
FFC7	P3CR	FFD7	TREG3	FFE7	INTEH (DMAEL)
FFC8	—	FFD8	TCLK	FFE8	DMAEH
FFC9	—	FFD9	TFFCR	FFE9	SCMOD
FFCA	—	FFDA	TMOD	FFEA	SCCR
FFCB	—	FFDB	TRUN	FFEB	SCBUF
FFCC	—	FFDC	—	FFEC	—
FFCD	—	FFDD	—	FFED	—
FFCE	—	FFDE	—	FFEE	—
FFCF	—	FFDF	—	FFEF	—

■ 9097249 0040801 25T ■

(1) I/O Port

Symbol	Name	Address	MSB 7	6	5	4	3	2	1	LSB 0
P0	Port 0	0FFC0H	P07	P06	P05	P04	P03	P02	P01	P00
						R/W				
						Input mode				
P1	Port 1	0FFC1H	P17	P16	P15	P14	P13	P12	P11	P10
						R/W				
						Input mode				
P2	Port 2	0FFC4H	P27	P26	P25	P24	P23	P22	P21	P20
						R/W				
						Input mode				
P3	Port 3	0FFC6H	P37	P36	P35	–	P33	P32	P31	–
			R	R/W	R/W	–	R/W	R/W	R	–
			Input	1	1	–	1	1	Input	–
P8	Port8	0FFD0H	—						P81	P80
			—						R	R
			—						Input mode	

250889

Note : Read/Write

R/W   : Either read or write is possible.

R   : Only read is possible.

W   : Only write is possible.

prohibit RMW : Prohibit Read Modify Write (Prohibit RES/SET Instruction)

9097249 0040802 196

(2) I/O Port control

MSB   LSB

Symbol	Name	Address	7	6	5	4	3	2	1	0
P01CR (IRFL)	Port 0/1 Control Reg.	0FFC2H	–	IRF0	IRFT0	IRFT1	–	EXT	P1C	P0C
			–	R			–	W	W	W
			–	0	0	0	–	0	0	0
		prohibit RMW	–	Interrupt Request Flag 1 : Interrupt being requested			–	P1, P2 control 0:I/O Port 1:Address bus	P1 control 0:In 1:Out	P0 control 0:In 1:Out
P2CR	Port2 Control Reg.	0FFC5H	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
		prohibit RMW	0	0	0	0	0	0	0	0
			0: In 1:Out (I/O selected bit by bit)							
P3CR	Port3 Control Reg.	0FFC7H	WAITC1	WAITC0	RDE	ODE	TXDC	SCLK	RXDC	–
			R/W		R/W	R/W	R/W	R/W	R/W	–
			0	0	0	0	0	0	0	–
			Wait control 00 : 2state wait 01 : normal wait 10 : non wait 11 : Timer 0/1 Output		RD control 0:$\overline{RD}$ for only external access 1:Always $\overline{RD}$	P33 control 0 :CMOS 1 :Open drain	P33 0: Out 1: TxD	P32 0: Out 1: SCLK	P31 0: In 1: RxD	–
P8CR	Port8 Control Reg.	0FFD1H	—						ZCE1	EDGE
			—						W	W
			—						0	0
		prohibit RMW	—						INT1/TI2 control 1:ZCD enable	INT0 control 0 : level 1 : ↑ edge

Symbol in ( ) denotes another name.

250889

■ 9097249 0040803 022 ■

(3). Watchdog timer control

Symbol	Name	Address	MSB 7	6	5	4	3	2	1	LSB 0
WDMOD	Watch Dog Timer Mode Reg.	0FFD2H	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRVE
			R/W	R/W		R/W	R/W		R	R/W
			1	0	0	0	0	0	Undefined	0
			1:WDT Enable	WDT Detecting time $00:2^{14}/fc$ $01:2^{16}/fc$ $10:2^{18}/fc$ $11:2^{20}/fc$		Warming up time $0:2^{14}/fc$ $1:2^{16}/fc$	Standby mode 00:RUN mode 01:STOP mode 10:IDLE1 mode 11:IDLE2 mode		Invert each time EXX instruction is executed	1: to drive pin in STOP mode.
WDCR	Watch Dog Timer Control Reg.	0FFD3H prohibit RMW	—							
			W							
			—							
			B1H : WDT Disable code　　4EH : WDT Clear code							

010491

▆ 9097249 0040804 T69 ▆

(4) Timer/event counter control

MSB / LSB

Symbol	Name	Address	7	6	5	4	3	2	1	0
TERG0	8bit Timer Register 0	0FFD4H prohibit RMW	—							
			W							
			Undefined							
TERG1	8bit Timer Register 1	0FFD5H prohibit RMW	—							
			W							
			Undefined							
TERG2	8bit Timer / Counter Latch Register 2	0FFD6H prohibit RMW	—							
			R/W R: Counter Latch Register 2, W: 8 bit Timer Register 2							
			Undefined							
TERG3	8bit Timer / Counter Latch Register 3	0FFD7H prohibit RMW	—							
			R/W R: Counter Latch Register 3, W: 8 bit Timer Register 3							
			Undefined							
TCLK	8bit Timer Source Clock Control Reg.	0FFD8H	T3CLK1	T3CLK0	T2CLK1	T2CLK0	T1CLK1	T1CLK0	T0CLK1	T0CLK0
			R/W		R/W		R/W		R/W	
			0	0	0	0	0	0	0	0
			8-bit 00 : TO2TRG 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : TI2 01 : ϕT1 10 : ϕT16 11 : ϕT256 (8bit mode only)		8-bit 00 : TO0TRG 01 : ϕT1 10 : ϕT16 11 : ϕT256		00 : —— 01 : ϕT1 10 : ϕT16 11 : ϕT256 (8bit mode only)	
TFFCR	8bit Timer Flip-Flop Control Reg.	0FFD9H	LATCH	—	—	—	TFF1C1	TFF1C0	TFF1E	TFF1IS
			W	—	—	—	W		R/W	
			1	—	—	—	—		0	0
			0 : LATCH (one shot)	—	—	—	00 : Clear TFF1 01 : Set TFF1 10 : Invert TFF1 11 : Don't care		1 : TFF1 Invert Enable	0 : Invert by 8bit timer 0 1 : Invert by 8 bit Timer 1
TMOD	8bit Timer Mode reg.	0FFDAH	T23M1	T23M0	—	—	T10M1	T10M0	PWM01	PWM00
			R/W		—	—	R/W		R/W	
			0	0	—	—	0	0	0	0
			(Note) 00:8bit Timer/counter 01:16bitTimer/counter 10:Don't care 11:Don't care		—	—	00 : 8bit Timer 01 : 16bit Timer 10 : 8bit PPG 11 : 8bit PWM		PWM Frequency 00 : —— 01 : 2^6-1 10 : 2^7-1 11 : 2^8-1	

Note :   00 :   8bit Timer × 2 or 8bit counter + 8bit Timer
         01 :   16bit Timer or 16bit counter

250889

Symbol	Name	Address	MSB 7	6	5	4	3	2	1	LSB 0
TRUN	8bit Timer/ Serial Channel Baud Rate Control Reg.	0FFDBH	BRATE1	BRATE0	PRRUN	–	T3RUN	T2RUN	T1RUN	T0RUN
			R/W			R/W				
			0	0	0	–	0	0	0	0
			00 : 300/150 bps 01 : 1200/600 10 : 4800/2400 11 : 19200/9600		Prescaler & Timer Run/Stop Control 0: Stop & Clear 1: Run (Count up)					

250889

(5)   Serial channel control

Symbol	Name	Address	MSB 7	6	5	4	3	2	1	LSB 0
SCMOD	Serial Channel Mode Reg.	0FFE9H	TB8	Fixed at "0"	RXE	WU	SM1	SM0	SC1	SC0
						R/W				
			Undefined	0	0	0	0	0	0	0
			Trans-mission Bit-8 data in 9bit UART	Write "0"	1 : Receive Enable	1 : Wake up Enable	00 : I/O interface 01 : UART 7bit 10 : UART 8bit 11 : UART 9bit		00 : TO2TRG 01 : BR 10 : ø1 11 : BR 1/2	U A R T
SCCR	Serial Channel Control Register	0FFEAH	RB8	–	–	OERR	–	FERR	–	–
			R	–		R (Cleared to "0" by reading)			–	
			Undefined	–	–	0	–	0	–	–
			Receiving Bit-8 data	–	–	1 : Error Overrun	–	Framing	–	–
SCBUF	Serial Channel Buffer Register	0FFEBH prohibit RMW	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R (Receiving)/W (Transmission)							
			Undefined							

250889

Also refer to P3CR, TRUN register.   Note :   BR: Baud Rate Generator

9097249 0040806 831

(6) Interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTEL	Interrupt Enable Mask Reg.	0FFE6H	IET2	IET3	–	IE1	–	–	IERX	IETX
						R/W				
			0	0	–	0	–	–	0	0
					1:Enable			0:Disable		
INTEH (DMAEL)	Micro DMA Enable Register	0FFE7H	0	–	DET0	DET1	–	IE0	IET0	IET1
				R/W				R/W		
			0		0	0		0	0	0
				1 : Enable		0 : Disable		1 : Enable		0 : Disable
DMAEH		0FFE8H	–	–	–	–	–	–	DERX	DETX
						–			R/W	
			–	–	–	–	–	–	0	0
					1:Enable			0:Disable		
IRFL (P01CR)	Interrupt Request Flag & IRF Clear	0FFC2H	–	IRF0	IRFT0	IRFT1	–	EXT	P1CR	P0CR
			–		R		–	W		W
			–	0	0	0	–	0	0	0
		prohibit RMW	–	Interrupt Request Flag 1 : Interrupt being requested			–	P1, P2 Control 0:I/O port 1:Address bus	P1 Control 0 : In 1 : Out	P0 Control 0 : In 1 : Out
IRFH		0FFC3H	IRFT2	IRFT3	–	IRF1	–	–	IRFRX	IRFTX
					R (Only IRF Clear code can be used to write)					
		prohibit RMW	0	0	–	0	–	–	0	0
			1: Interrupt being requested (IRF is cleared to "0" by writing IRF Clear code).							

250889

Symbol in ( ) denotes another name.

6. PORT SECTION EQUIVALENT CIRCUIT DIAGRAM

- Reading The Circuit Diagram

    Basically, the gate singles written are the same as those used for the standard CMOS logic IC [74HCXX] series.

    The dedicated signal is described below.

    STOP : This signal becomes active "1" when the hold mode setting register is set to the STOP mode and the CPU executes the HALT instruction. When the drive enable bit [DRIVE] is set to "1", however, STP remains at "0".
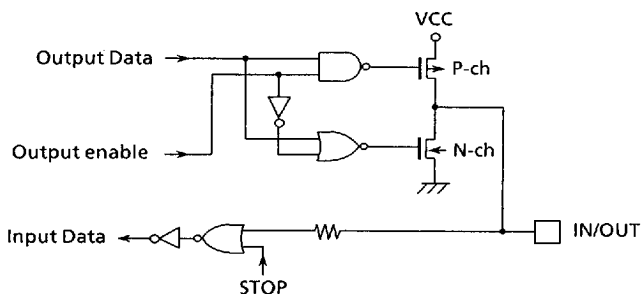
- The input protection resistans ranges from several tens of ohms to several hundreds of ohms.

■ P0 (D0~D7)
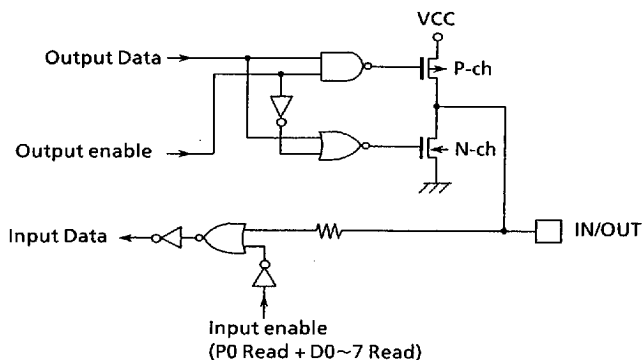
030289

■ P1, P2

030289

■ P31, P37
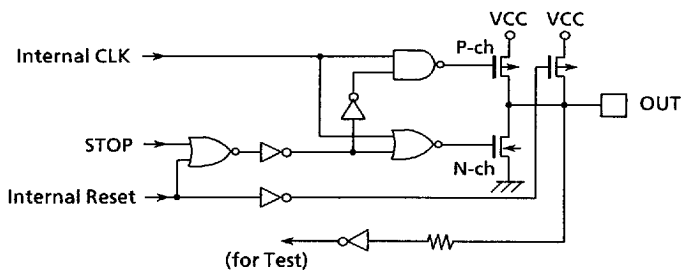
■ P32, P35, P36

■ P33

■ P80, $\overline{\text{NMI}}$

9097249 0040809 540

■ P81

030289

■ RESET

030289

■ CLK

270289
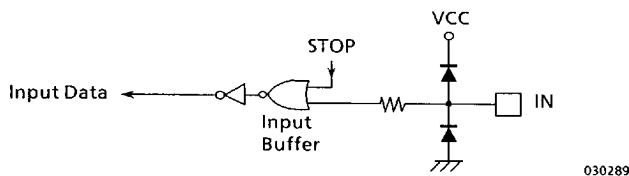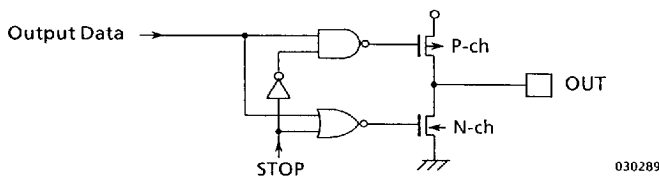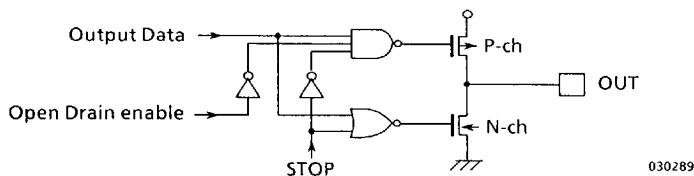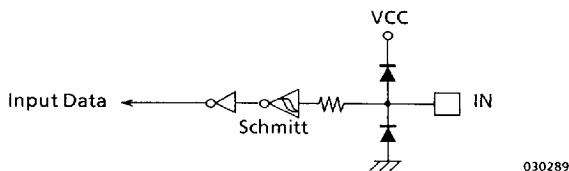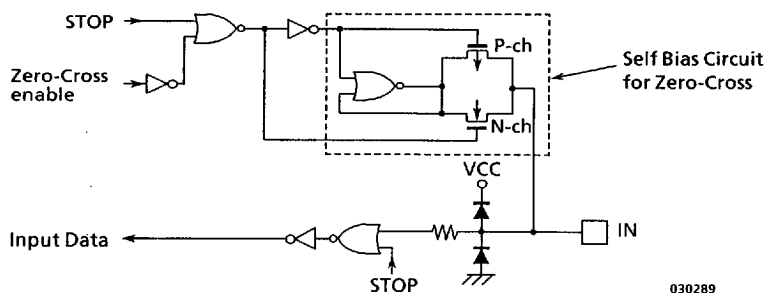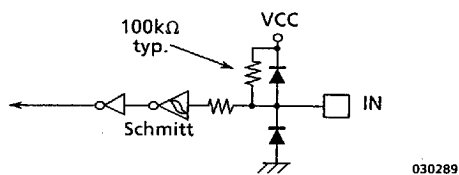
9097249 0040810 262

■ X1, X2

oscillation circuit

VCC

STOP

P-ch   N-ch

×2

VCC

×1

030289

■ $\overline{EA}$

Delete at 90P802A

VCC

IN

030289

9097249 0040811 1T9