TOSHIBA

CMOS 8–Bit Microprocessor

TMP90C051F

1. Overview and Features

The TMP90C051F is a high-speed 8-bit CMOS microprocessor with advanced functions and was developed for application in various types of device control.

The TMP90C051F is a single-chip, 8-bit CMOS microprocessor that integrates peripheral functions such as DMA, DRAM controller, serial interface, timer/event counter and real time clock with an 8-bit CPU (TLCS90) core. Memory area can also be extended up to 8M bytes depending on the MMU.

The TMP90C051F features are as follows.

(1) Highly efficient instruction group
Number of basic instructions: 167
Multiplication and division instructions, 16-bit arithmetic operation instructions and bit operation instructions.

(2) Minimum instruction execution times: 250ns (at 16.0MHz)

(3) Memory extension capability
Address space: 8M byte (max)

(4) DMA (2 channels)
Maximum transfer speed: 2-byte transfer;
0.88M byte/sec (at 16.0MHz)

(5) DRAM controller

(6) General-purpose serial interface (2 channels)
UART (transmit/receive)
I/O interface mode (transmit/receive)

(7) Thermal printer head (transmit)

(8) 8-bit timer/event counter (4 channels)

(9) Real time clock: Clock and calendar functions/battery backup capability.

(10) Stepping motor control pattern generation ports (2 channels)

(11) Input/output port (31-pin)

(12) Interrupt functions:11 internal, 9 external.

(13) Micro DMA function
Maximum transfer speed: 0.34M byte/s (at 16.0MHz)

(14) Watchdog timer function
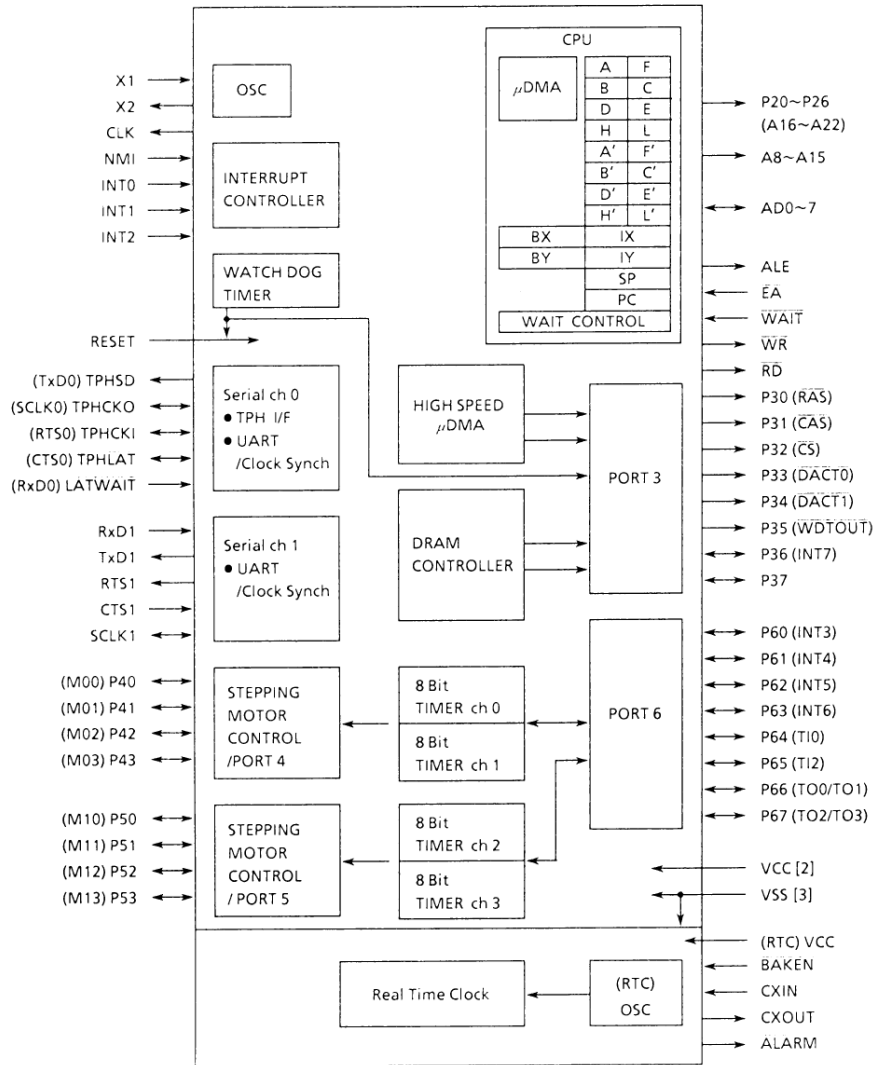
(15) Standby function (3 halt modes)

Figure 1. TMP90C051F Block Diagram

2. Pin Layout and Functions

This section shows the TMP90C051F pin layout diagram, and describes the input/output pin names and functions.

2.1 Pin Layout Diagram
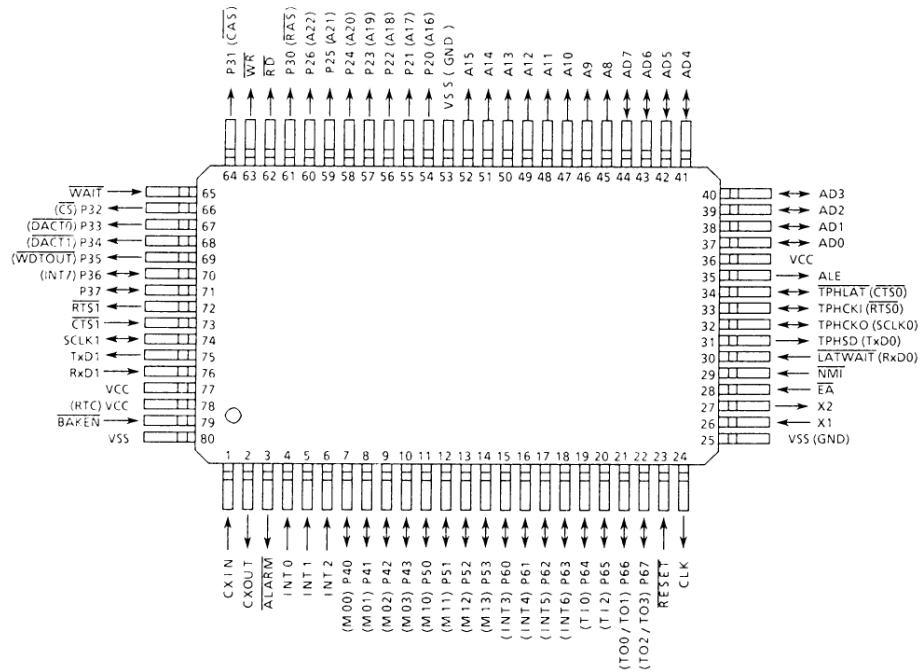
Figure 2.1 shows the TMP90C051F pin layout.

Figure 2.1. Pin Layout Diagram (80-pin Flat Package) [TOP VIEW]

2.2 Pin Names and Functions

Table 2.2 Shows the input/output pin names and functions.

Table 2.2 Pin Names and Functions (1/3)

Pin Name	Number of pins (pin no.)	Input/output or tristate	Function
AD0 ~ AD7	8 (37 ~ 44)	Input/output or tristate	Lower address/data bus: Used as both the lower 8-bit address bus (output) and data bus (bi-directional).
A8 ~ A15	8 (45 ~ 52)	Output	The upper 8-bit address bus.
P20 ~ P26 (A16 ~ A22)	7 (54 ~ 60)	Output	Port 2: A 7-bit output port. Outputs "0" after reset.
		Output	Address bus: The extension (upper) address bus. Outputs "0" after reset; there-fore, the TMP90C051F address area is 0 - 64K bytes until a value other than 000H is written to EXPA0 and EXPA1 of the MMU register group after a reset; therefore, A16 - A22 output zeros (0). When a logical address indicates within the local area (local 0 or local 1) range when the MMU is started, the physical addresses generated by the MMU are outputs to A16 - A22. Zeros (0) are output to A16 - A22 when a logical address indicates within the common area (common 0 or common 1) range. When an indirect extension address area is accessed using the CPU index regis-ter the address value loaded to the BX or BY register is output to A16 - A22. When data are transferred using high-speed micro DMA, the address value loaded to the DMA bank register is output to A16 - A22.
$\overline{RD}$	1 (62)	Output	Read: The strobe signal output used to read data from external memory.
$\overline{WR}$	1 (63)	Output	Write: The strobe signal output used to write data to external memory.
$\overline{WAIT}$	1 (65)	Input	Wait: An input pin used to connect memory with slow access time and periph-eral LSIs.
$\overline{NMI}$	1 (29)	Input	Non-maskable interrupt request pin: The input pin used to generate interrupt requests on the fall edges
P30 ($\overline{RAS}$)	(1) (61)	Output	Port 30: A 1-bit output port. Outputs "1" after reset.
			RAS: Outputs the RAS timing required for refreshing the DRAM of CAS before RAS.
P31 ($\overline{CAS}$)	1 (64)	Input	Port 31: A 1-bit output port. Outputs "1" after reset.
			CAS: Outputs the CAS timing required for refreshing the DRAM of CAS before RAS.
P32 ($\overline{CS}$)	1 (66)	Output	Port 32: A 1-bit output port. Outputs "1" after reset.
			Chip select: The external memory chip select output pin. When a start address and area size are loaded to the control register in the dynamic RAM controller, "0" is output from the CS pin whenever the TMP90C051F accesses that address range.
P33 ($\overline{DACT0}$)	1 (67)	Output	Port 33: A 1-bit output port. Outputs "1" after reset.
			DMA active 0: Outputs zeros (0) while data are being transferred via high-speed micro $\overline{DMA}$ channel 0.

Table 2.2. Pin Names and Functions (2/3)

Pin Name	Number of pins (pin no.)	Input/outpu t or tristate	Function
P34 ($\overline{DACT1}$)	1 (68)	Output	Port 34: A 1-bit output port. Outputs "1" after reset.
			DMA active 1: Outputs zeros (0) while data are being transferred via high-speed micro $\overline{DMA}$ channel 1.
P35 ($\overline{WDTOUT}$)	1 (69)	Output	Port 35: A 1-bit output port. Outputs "WDTOUT = 1" after reset.
			Watchdog timer output: used as the $\overline{WDTOUT}$ output pin when watchdog timer mode register, D1 = 1.
P36 (INT7)	1 (70)	Input/Output Input	Port 36: A 1-bit output port. Functions as an input port after reset.
			Interrupt request pin 7: The input pin for generating interrupt requests at rise edges.
P37	1 (71)	Input/output	Port 37: A 1-bit input/output port. Functions as an input port after reset.
P40 ~ P43	4 (7 ~ 10)	Input/output	Port 40 - 43: A 4-bit input/output port. Functions as an input port after reset.
		Output	Stepping motor control port 0.
P50 ~ P53 (M10 ~ M03)	4 (11 ~ 14)	Input/output	Port 50 - 53: A 4-bit input/output port. Functions as an input port after reset.
		Output	Stepping motor control port 1
P60 ~ P63 (INT3 ~ INT6)	4 (15 ~ 18)	Input/output	Port 60 ~ 63: A 4-bit input/output port. Functions as an input port after reset.
		Input	Interrupt request pins 3-6: Input pins for generating interrupt requests at rise edges.
P64 (TI0)	1 (19)	Input/output	Port 64: A 1-bit input/output port. Functions as an input port after reset.
		Input	Timer input 0: The timer 0 clock input pin.
P65 (TI2)	1 (20)	Input/output	Port 65: A 1-bit input/output port. Functions as an input port after reset
		Input	Timer input 2: The timer 2 clock input pin.
P66 (TO0/TO1)	1 (22)	Input/output	Port 66: A 1-bit input/output port. Functions as an input port after reset.
		Output	Timer output 0/1: The timer 0 or timer 1 output pin.
P67 (TO2/TO3)	1 (23)	Input/output	Port 67: A 1-bit input/output port. Functions as an input port after reset.
		Output	Timer output 2/3: The timer 2 or timer 3 output pin.
$\overline{LATWAIT}$ $\overline{RxD0}$	1 (30)	Input	Set serial channel 0 to the TPH mode after reset. The wait control input pin used to cause $\overline{TPHLAT}$ output to wait.
			Used as the receive data input pin when serial channel 0 is set to the Normal mode.
TPHSD (TxD0)	1 (31)	Output	Serial channel 0 enters the TPH mode and outputs "0" after a reset. The TPHSD pin is used to output transmit data to TPH.
			When set to the Normal mode, serial channel 0 functions as the transmit data output pin TXD0 output immediately after mode is set.
TPHCK0	1	Output	After a reset, serial channel 0 enters the TPH mode and outputs "1". The TPHCK0 pin is used to output the transmit clock to TPH.
(SCLK0)	(32)	Input/output	When set to the Normal mode by the TPH control register, serial channel 0 operates in the UART or I/O interface mode. The control register of serial channel 0 determines whether the UART or I/O interface mode is used. In the I/O interface mode, either the external clock from SCLK0 or the transmit/receive clock from the internal baud rate generator is output. Note: When operating with external clock inout in the I/O interface mode SCLK0 is output after a reset; therefore, use an open drain circuit to connect the external clock to the SCLK0 pin.

Table 2.2 Pin Names and Functions (3/3)

Pin Name	No. of pins (pin. no.)	Input/output or tristate	Function
TPHCKI	1	Input	Serial channel 0 enters the TPH mode after a reset and functions to input the transmit clock to TPH.
($\overline{\text{RTS0}}$)	(33)	Output	Serial channel 0 is the TPH control register and operates as the RTS output pin when set to the Normal mode.
$\overline{\text{TPHLAT}}$	1	Output	After a reset, serial channel 0 enters the TPH mode and the functions as the latch signal pin for outputs to TPH.
($\overline{\text{CTS0}}$)	(34)	Input	When set to the Normal mode by the TPH control register, serial channel 0 operates as the CTS0 input pin. Consequently, since the CTS0 pin outputs after a reset, use an open drain circuit to connect the CTS0 pin to CTS0 when serial channel 0 is used in the UART mode.
$\overline{\text{RTS1}}$	1 (72)	Output	The serial channel 1 $\overline{\text{RTS1}}$ output pin.
$\overline{\text{CTS1}}$	1 (73)	Input	The serial channel 1 $\overline{\text{CTS1}}$ input pin.
SCLK1	1 (74)	Input/output	Functions as the serial channel 1 transceive clock input or output pin. SCLK1 functions as an input pin after a reset.
TxD1	1 (75)	Output	The serial channel 1 transmit data output pin.
RxD1	1 (76)	Input	The serial channel 1 receive data input pin.
CXIN/CXOUT	2 (1/2)	Input/output	Used to connect the crystal oscillator for the real time clock. A 32.768kHz crystal oscillator is usually connected.
$\overline{\text{ALARM}}$	1 (3)	Output	Alarm: The output pin for alarms from the real time clock. Comparator output and either the 16Hz or 1Hz signal generated by the divider can be output to the ALARM pin using the alarm select circuit inside the real time clock.
INT0 ~ INT2	3 (4~6)	Input	Interrupt request pins 0 - 2: The input pin used to generate interrupt requests at rise edges. INT0 is an input pin with a Schmitt circuit.
ALE	1 (35)	Output	Address latch enable signal: AD0 - AD7 addresses are latched on the fall of this signal. Connected to external memory.
$\overline{\text{EA}}$	1 (28)	Input	Connect to GND.
CLK	1 (24)	Output	Clock output: Output a clock equal to the oscillation frequency divided by 4. Pulled up during resets.
$\overline{\text{RESET}}$	1 (23)	Input	Reset: The reset input pin used to initialize the TMP90C051F. (Built-in pull up register)
X1/X2	2 (26/77)	Input/Output	Used to connect the crystal oscillator that generates the TMP90C051 internal system clock.
V$_{CC}$	2 (36/77)	–	Main power supply pin (+5V)
$\overline{\text{BAKEN}}$	1 (79)	Input	Backup enable: set BUKEN = 1 when the TMP90C051F main power supply Vcc (pins 36/77) is +5V. Set to BUKRN = 0 to provide battery backup for the real time clock when the main power supply Vcc (pins 36/77) is off.
V$_{CC\ (RTC)}$	1 (78)	–	The real time clock backup power supply pin. Arrange so that the same power is supplied to Vcc (RTC) as to Vcc (pins 36/77) when power is being supplied to Vcc (pins 36/77). To provide battery backup for only the ral time clock, supply + 5V - 2V to Vcc (RTC).
V$_{SS}$ (GND)	3 (25/53/80)	–	GND pin (0V)

3. Operation

This section explains the functions and basic operations of the TMP90C051F by block.

3.1 CPU

The TMP90C051F includes a high performance 8 bit CPU. For the function of the CPU, see the book TLCS 90 Series CPU Core Architecture. This chapter explains exclusively the functions of the CPU of TMP90C051F which are not described in the the book.

3.1.1 Reset

Figure 3.1 (1) shows the basic timing for the reset operation.

To reset the TMP90C051F, it is necessary to maintain the RESET input at "0" for at least 10 system clocks (10 states: 2 microseconds with a 10MHz system clock) with the power supply voltage within the operating range and stable internal oscillator operation.

When a reset is received, the address data bus (AD0 - AD7), address bus (A8 - A15), P36, P37, port 4 port 5 and port 6 are all set to input port status (high impedance). Dedicated output port 2 (A16 - A22) ALE is set to "0"; P30 - P35, RD, WR and CLK are set to "1". The other dedicated output ports are all set to "0" and the dedicated input ports retain their current status.

The CPU registers and external memory are not changed; however, the program counter PC, interrupt enable/disable flag IFF, bank registers BX and BY are cleared to "0". The A register is indeterminate.

When the reset is released, the instruction starts from address 0000H.
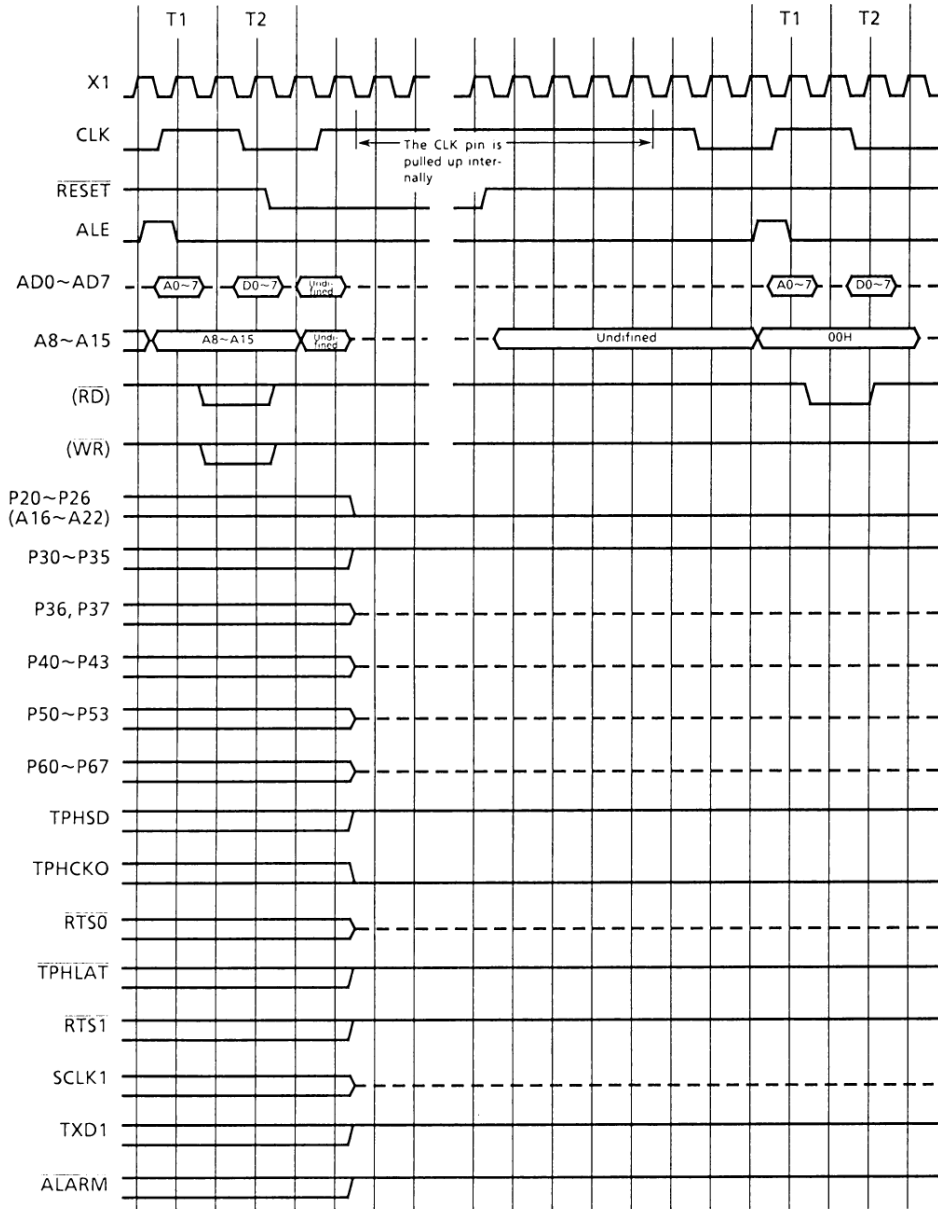
Figure 3.1 (1). Reset Timing

3.1.2 EXF (Exchange Flag)

For TMP90C051F, "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 FFBFH.

P6FR (FFBFH)		7	6	5	4	3	2	1	0
	bit Symbol	TO3C	TO1C	–	–	–	–	EXF	Fixed at "0"
	Read/Write	R / W		–	–	–	–	R	–
	Resetting value	0	0	–	–	–	–	Undefined	0
	Function	0: general purpose port 1: TO2/3 output	0: general purpose port 1: TO0/1 output					EXX instruction Reverses each time executed	Write "0"

3.1.3 Wait Control

For TMP90C051F, a wait control register P2FR <WAITC0, 1> is allocated to the 2th and 3th bits of memory address FFDFH.

P2FR (FFDFH)		7	6	5	4	3	2	1	0
	bit Symbol	SCLKC1	SCLKC0	ODE1	ODE0	WAITC1	WAITC0	–	Fixed at "1"
	Read/Write	R / W				R / W		–	R / W
	Resetting Value	0	0	0	0	0	0	–	1
	Function	SCLK1 CONTROL 0: SCLK1 input 1: SCLK1 output	SCLK0 CONTROL 0: SCLK0 input 1: SCLK0 output	TXD1 CONTROL 0: CMOS output 1: Open drain output	TXD0 CONTROL 0: CMOS output 1: Open drain output	Wait control 00: 2 state Wait 01: Normal Wait 10: non Wait 11: reserved			Write "0"

3.1.4 Bank Register

For TMP90C051F, BX and BY registers are allocated to memory addresses FFECH (BX register) FFEDH (BY register), respectively. In these registers, only the low-order 7 bits are valid, and the high-order 1 bit are undefined. These undefined bits become "1" whenever they are read.

		7	6	5	4	3	2	1	0
BX (FFECH)	bit Symbol	–	BX6	BX5	BX4	BX3	BX2	BX1	BX0
	Read/Write	–	R / W						
	Resetting Value	–	0	0	0	0	0	0	0

		7	6	5	4	3	2	1	0
BY (FFEDH)	bit Symbol	–	BY6	BY5	BY4	BY3	BY2	BY1	BY0
	Read/Write	–	R / W						
	Resetting Value	–	0	0	0	0	0	0	0

3.2 Memory Map

The TMP90C051F supports a program memory and a data memory of maximum 8M bytes.

The program and data memory may be assigned to the address space from 00000H to 7FFFFFH.

3.2.1 Internal I/O

The TMP90C051F provides a 128-byte address space as an internal I/O area, whose addresses range from FF80H to FFFFH. This I/O area can be accessed by the CPU using a short opcode in the "direct addressing mode".

Figure 3.2 (1) is a memory map indicating the areas accessible by the CPU in the respective addressing mode.
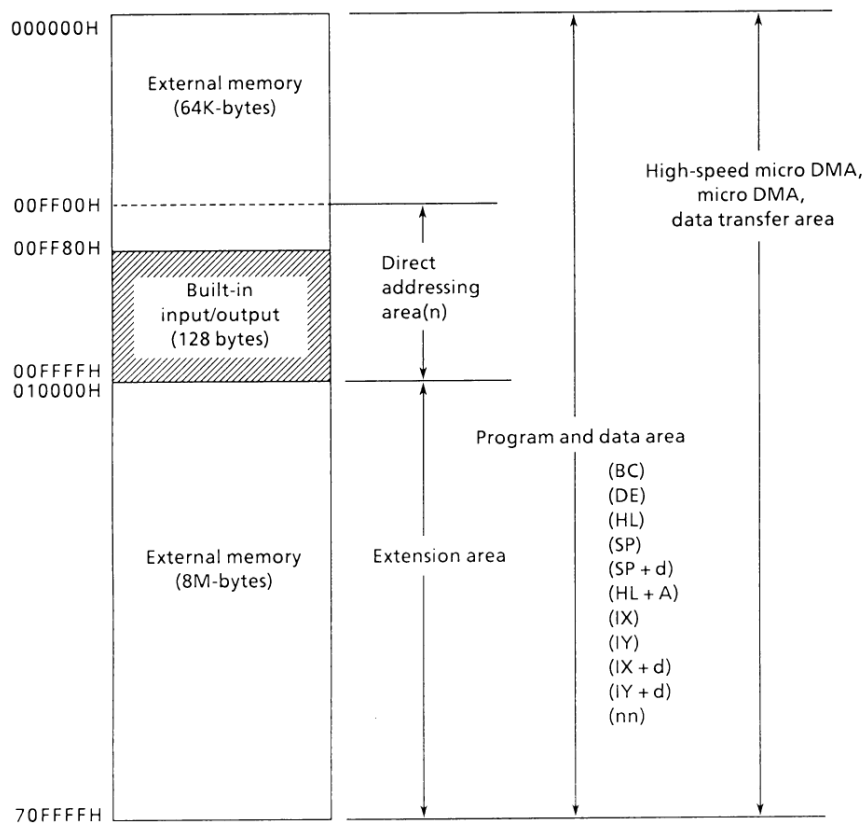
Figure 3.2 (1). Memory Map

3.3 Interrupt Functions

The TMP90C051F has a general-purpose interrupt process-ing routine for responding to both internal and external inter-rupt requests, a micro DMA processing mode in which the CPU automatically transfers data, and a high-speed micro DMA (HDMA) processing mode.

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

The high-speed DMA processing mode can be set by loading a vector value to the DMAV 0/1 register.

Figure 3.3 (1) shows the interrupt response flow.
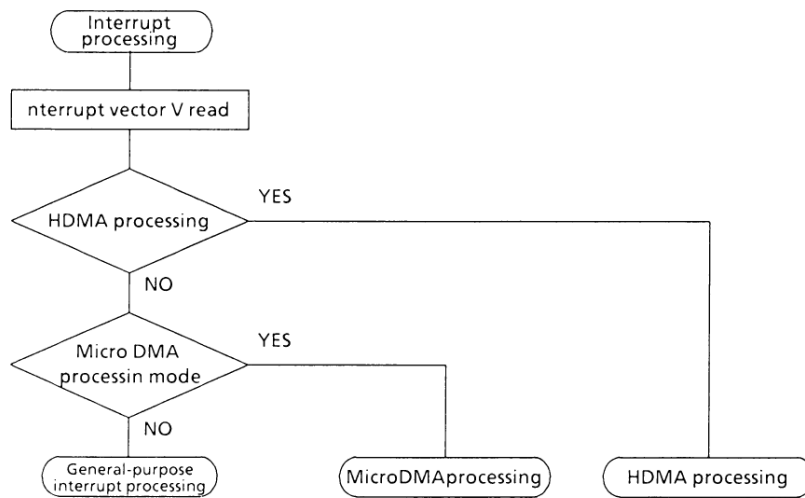
Figure 3.3 (1). Interrupt Response Flow

When an interrupt request is generated, this is reported to the CPU via the built-in interrupt controller. If the request is for a non-maskable interrupt or an enabled maskable interrupt, the CPU starts interrupt processing. If for a disabled maskable interrupt, the request is ignored and not received.

If the interrupt is received, the CPU first reads the interrupt vector from the built-in interrupt controller to determine the source of the interrupt request.

Next, a check is made as to whether this request is for general-purpose interrupt processing, micro DMA processing or high-speed DMA (HDMA) processing, and then the corre-sponding processing is performed.

The interrupt vector is read in an internal operation cycle so the bus cycle becomes a dummy cycle.

3.3.1 General-Purpose Interrupt Processing

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

The CPU first saves the contents of the program counter PC and register AF (including the interrupt enable/disable flag IFF immediately before an interrupt) to the stack and then resets the interrupt enable/disable flag IFF to "0" (interrupt disable). Finally, the interrupt vector contents [M] are transferred to the program counter and a jump is made to the interrupt processing program.

There is a 20-state overhead from the time when the interrupt is received until the jump is made to the interrupt processing program.
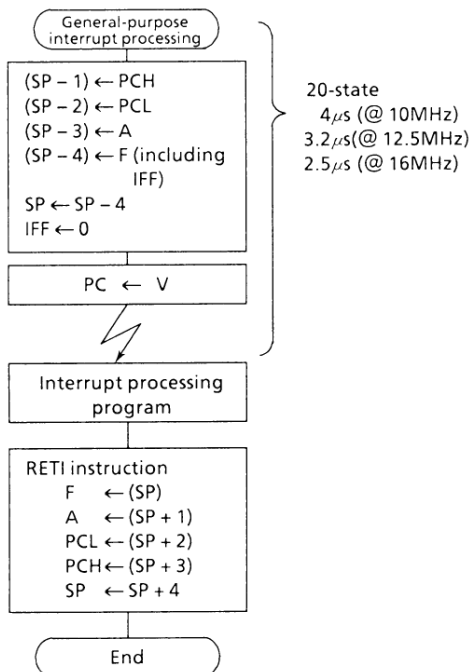
Figure 3.3 (2). General-Purpose Interrupt Processing Flow

Interrupt processing program is ended with the RETI instruction for both maskable and non-maskable interrupts.

Executing this instruction restores the program counter PC and register AF contents from the stack. (Resets the interrupt enable/disable flag immediately before an interrupt.)

When the CPU reads the interrupt vector, the interrupt request source confirms that the interrupt has been received and then clears the interrupt request. Non-maskable interrupts cannot be disabled by program. Maskable interrupts, however, can be enabled and disabled by program. Bit 5 of CPU register F is an interrupt enable/disable flipflop (IFF). Interrupts are enabled by setting this bit to "1" with the EI (interrupt enable) instruction and disabled by resetting this bit to "0" with the DI (interrupt disable) instruction. IFF is reset to "0" by resetting and when an interrupt is received (including non-maskable interrupts).

The EI instruction is actually executed after the next instruction is executed.

Table 3.3 (1) shows the interrupt sources.

Table 3.3 (1) Interrupt Sources

Priority sequence	Type	Interrupt function name	Interrupt vector value	General-purpose interrupt processing start address	Micro DMA support	Micro DMA processing parameter start address	Interrupt sources		Comments	
							Internal	External		
1 2	Non-maskable	S W I Instruction NMI	08H 10H	0008H 0010H	– –	– –	O –	– NMI pin	– –	*1)
3	Maskable	(HDMA)	–	–	–	–	–	–	* 1	
4		INT0	18H	0018H	O	FF18H	–	INT0 pin	External 0	
5		INT1	20H	0020H	O	FF20H	–	INT1 pin	External 1	
6		INT2	28H	0028H	O	FF28H	–	INT2 pin	External 2	
7		INTT0	30H	0030H	O	FF30H	Timer 0	–	Timer	
8		INTT1	38H	0038H	O	FF38H	Timer 1	–	Timer	
9		INTT2	40H	0040H	O	FF40H	Timer 2	–	Timer	
10		INTT3	48H	0048H	O	FF48H	Timer 3	–	Timer	
11		INTRX0	50H	0050H	O	FF50H	Serial receive end	–	SIO	
12		INTTX0	58H	0058H	O	FF58H	Serial send end	–	SIO	
13		INTLINE	60H	0060H	O	FF60H	TPH send	–	TPH	
14		INTRX1	68H	0068H	O	FF68H	Serial receive end	–	SIO	
15		INTTX1	70H	0070H	O	FF70H	Serial receive end	–	SIO	
16		INTALARM	78H	0078H	–	–	Alarm output	–	RTC	
17		INT3	80H	0080H	–	–	–	INT3 pin	External 3	
18		INT4	88H	0088H	–	–	–	INT4 pin	External 4	
19		INT5	90H	0090H	–	–	–	INT5 pin	External 5	
20		INT6	98H	0098H	–	–	–	INT6 pin	External 6	
21		INT7	A0H	00A0H	–	–	–	INT7 pin	External 7	

Note: * HDMA supports all maskable interrupts.

The "priority sequence" shown in Table 3.3 (1) indicates the sequence in which interrupt sources are received by the CPU when multiple interrupt requests are generated simultaneously.

For example, if interrupt requests with the priority sequences 4 and 5 are generated simultaneously, the CPU will receive the interrupt request with priority sequence 4 first. When processing of the interrupt with priority sequence 4 is ended with the RETI instruction, the CPU will then receive the interrupt with priority sequence 5.

If the interrupt processing program for the priority sequence 4 interrupt is interrupted by executing the EI instruction, the CPU will receive the priority sequence 5 interrupt request. When multiple interrupt requests are generated simultaneously, the built-in interrupt controller only determines the priority sequence of the interrupt sources received by the CPU. There is no function to compare the priority sequence of the interrupt currently being processed and the interrupt currently being requested.

Another interrupt can be enabled while another interrupt is being processed by resetting the interrupt enable/disable flag IFF to enable.

3.3.2 Micro DMA Processing

Figure 3.3 (3) shows the micro DMA processing flow. The CPU first loads the parameters (transfer source and destination addresses, transfer mode) required for transferring data between memories from the address modified by the interrupt vector value and then transfers the data between memories in accordance with those parameters. After that, the revised parameters are saved to the original location. The transfer count is then decremented and, if the value is not "0", micro DMA processing is ended. If the value is "0", the general-purpose interrupt processing described in the previous item is performed.

The transfer count is decremented by "-1" each time the micro DMA starts up.
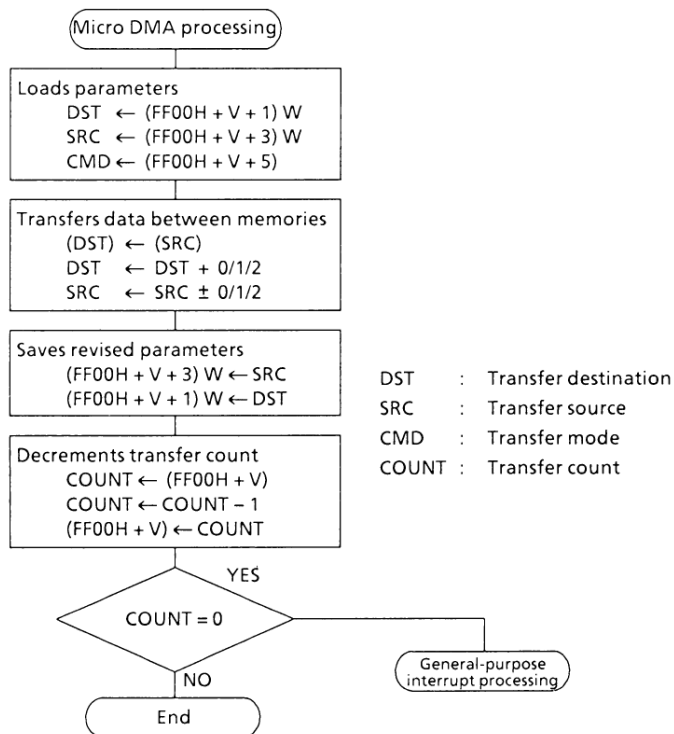
Figure 3.3 (3). Micro DMA Processing Flow

Since most interrupt processing involves only simple data transfers, micro DMA processing uses only the hardware for such processing. Consequently, micro DMA processing is faster than conventional software processing. Naturally, there is absolutely no influence on the CPU registers from the micro DMA processing.

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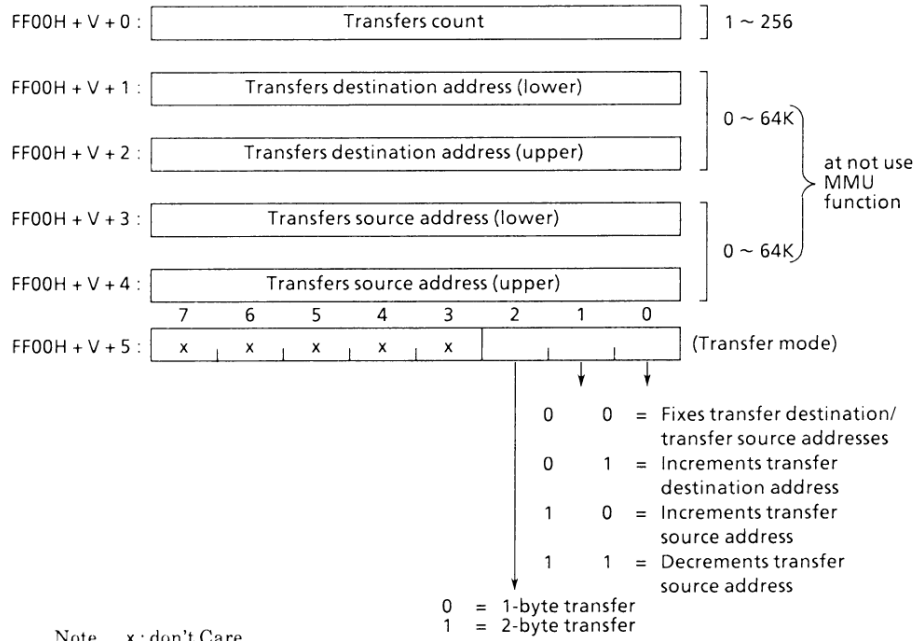
Figure 3.3 (4). Micro DMA Processing Parameters

The parameters used for the micro DMA processing are located in the external RAM area (See Table 3.3 (1) Interrupt sources). The start address of each micro DMA processing parameter is [FF00H + interrupt vector value], 6 bytes of which are used for the parameters. When the micro DMA processing mode is not used, this area can be used as user memory.

The parameters include transfer count, transfer destination address, transfer source address and transfer mode. Transfer count specifies the number of times data will be transferred by micro DMA processing. Either 1 or 2- bytes of data are trans-ferred at one time by micro DMA processing. Data are transferred 256 times with a transfer count of "00H".

The addresses 0000H - FFFFH are used in micro DMA processing (at not use MMU function). Bits 0 and 1 specify the transfer destination or transfer source address change mode. Bit 2 specifies the length (1 or 2 bytes) of the data handled.

Table 3.3 (2) shows the relationship between the transfer mode and transfer destination/transfer source address increment/decrement values.

Table 3.3 (2) Micro DMA Processing Address Changes

Transfer Mode	Function	Transfer destination address	Transfer source address
000	Transfers 1-byte, fixes transfer destination/source addresses.	0	0
001	Transfers 1-byte, increment s transfer destination address.	+1	0
010	Transfers 1-byte, increment transfer source address.	0	+1
011	Transfers 1-byte, decrement transfer source address.	0	-1
100	Transfers 2-bytes , fixes the current destination/source addresses.	0	0
101	Transfers 2-bytes , increment transfer destination address.	+2	0
110	Transfers 2-bytes, increment transfer source address.	0	+2
111	Transfers 2-bytes, decrements, transfer source address.	0	-2

Data are transferred as shown below in the 2-byte transfer mode.

(Transfer source address)←(Transfer destination address)
(Transfer source address + 1)←(Transfer destination address + 1)

Transfers are also performed as shown above in "decrement transfer source address mode" but address changes are as shown in Table 3.3.

Address incrementation and decrementation are used for the memory area but fixed addresses are used for ordinary input/output addresses. Because of that, micro DMA was designed taking into consideration input/output to memory and memory to input/output transfers.

Figure 3.3 (5) shows and example using the micro DMA processing mode. Built-in serial input/output receive data are processed in the example.

In the example, 7 frames (1 frame = 1 byte in this example) of received data are stored to memory address FF00H - FF06H and when all of the data have been received, the "receive end processing program" is executed.

```
SET    6,(0FFCEH)      ;  Sets serial receive interrupt to micro DMA
                          processing mode
LD     (0FF50H),7      ;  Sets transfer count to 7
LDW    (0FF51H),0FF00H ;  Sets transfer destination start address to FF00H
LDW    (0FF53H),0FFD2H ;  Sets transfer source (serial receive buffer)
                          address to FFEBH
LD     (FF55H),1       ;  Sets transfer mode ( transfers 1 byte,
                          increments transfer destination address)
EI
 :
 :
ORG    0050H
```

Receive end interrupt processing program

```
RETI
```

Figure 3.3 (5). Micro DMA Processing Example

"Table 3.2 Bus operations for each instruction" above shows the bus operation for general-purpose interrupt processing and micro DMA processing.

The execution time (when the transfer count is not 0 after decrementation) for micro DMA processing is 46 states (9.2 microseconds at 10MHz), regardless of whether the 1-byte or 2-byte transfer mode is used.

3.3.3 High-Speed Micro DMA Processing

The TMP90C051F has two built-in DMA channels called HDMA.

HDMA has three times the processing capacity of μDMA and is used for high-speed data transfers. HDMA execution time (decrease the value of transfer number and the value is not "0" data) is 14 states, regardless of whether the 1-byte transfer mode or 2-byte transfer mode is used. HDMA and micro DMA transfer speeds.

Table 3.3 (3) shows the high-speed micro.

Table 3.3 (3) Transfer Speeds

fxtal (MHz)	HDMA	Micro DMA
10	2.8μs	9.2μs
12.5	2.24μs	7.36μs
16	1.75μs	5.75μs

* At 1-byte transfer mode.

Table 3.3 (4) Shows the HDMA function

Number of channels	2
Transfer speed	14 states (for 1 byte) or 18 states (for 2 bytes)
Start method	By interrupt (all external and internal interrupt sources)
Transfer mode	1 byte transfer or 2 byte transfer
Address output method	Dual address (source/destination)
Access area	0 ~ 8-M byte memory area (64k-byte units)

(1)　HDMA Setting Registers

The following describes the registers required for HDMA operation.

(a) DMAS0, 1: source address register (16-bit)

A15															A0

(b) DMA0, 1: Destination address register (16-bit)

A15															A0

(c) DMAC0, 1: byte count register (16-bit)

A15															A0
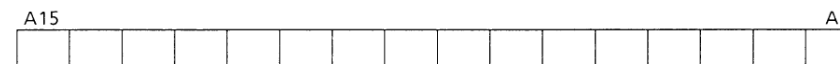
Sets the number of bytes to be transferred. The set value is decremented (−1) for each HDMA started. A generation purpose interrupt is when the value reaches "0".

(d) DMAM0, 1: transfer mode register (8-bit)

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–					

D4	Transfer byte count
0	1-byte transfer
1	2-byte transfer

D3	D2	Source address
0	0	Fixed
0	1	Increment
1	0	Decrement

D1	D0	Destination address
0	0	Fixed
0	1	Increment
1	0	Decrement
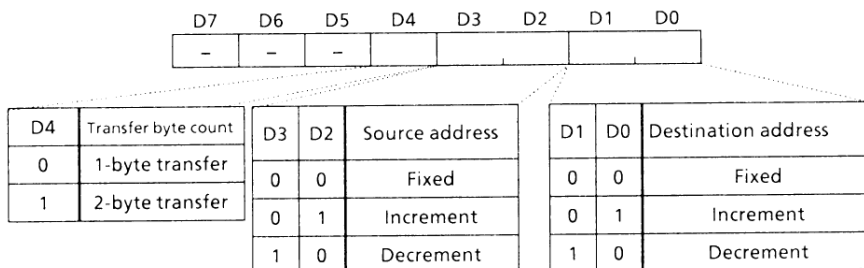
This register determines the HDMA transfer mode.
The bits of this register are as shown in the table above.
The source and destination addresses shown here are the addresses loaded to DMAS0, 1 and DMAD0, 1 above.

Example 1 : XXX00001
Transfers 1 byte, fixes transfer source address (DMAS0, 1), and increments the transfer destination address after each transfer.

Example 2: The renewal of address at 2 byte transfer mode.
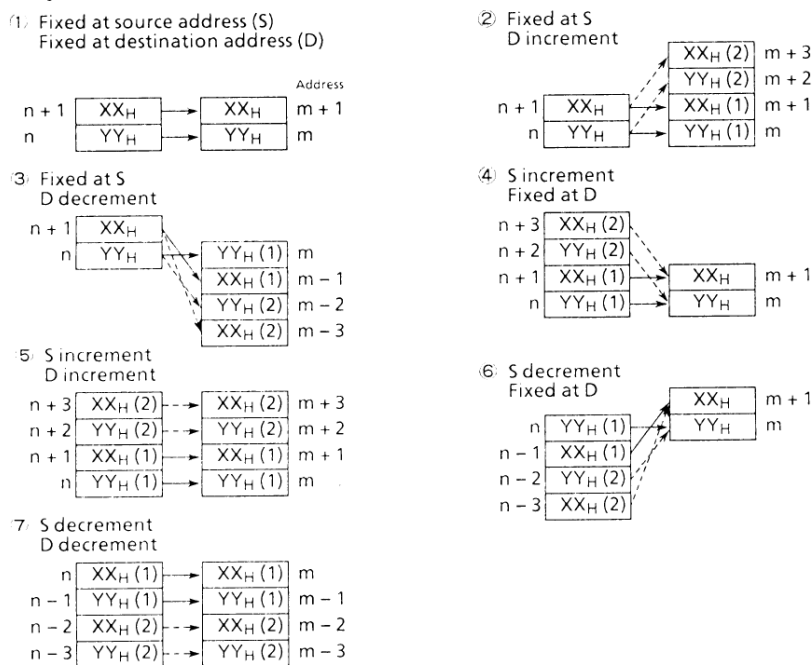
Note: It is ineffective to set decrement for a destination address when a source address being increment; and to set increment for a destination address when a source address being decrement.

(e) DMASB0, 1: source address bank register

DMASB0 (FFF6H)		7	6	5	4	3	2	1	0
	bit Symbol	–	DSB06	DSB05	DSB04	DSB03	DSB02	DSB01	DSB00
	Read/Write	–	R / W						
	Resetting Value	–	0	0	0	0	0	0	0

DMASB1 (FFF9H)		7	6	5	4	3	2	1	0
	bit Symbol	–	DSB16	DSB15	DSB14	DSB13	DSB12	DSB11	DSB10
	Read/Write	–	R / W						
	Resetting Value	–	0	0	0	0	0	0	0

Source address extension register
Used as a pair with the DMAS0, 1 register, this bank register is specified for addresses for 64k bytes or more.

This register has no increment or decrement function; therefore, it is necessary to exercise caution in specifying variable addresses (increment or decrement) with the above mode register.

A22	A16	A15	A0
DMASB		DMAS	
Bank register		Source address register	

```
          00  0000H  ⎫
                     ⎬  0~8M bytes (64k-byte unit)
          7F  FFFFH  ⎭
```
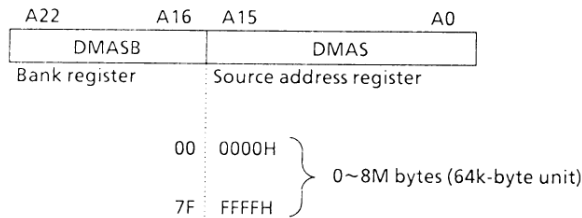
(f) DMADB0, 1: destination address bank register

DMADB0 (FFF7H)		7	6	5	4	3	2	1	0
	bit Symbol	–	DDB06	DDB05	DDB04	DDB03	DDB02	DDB01	DDB00
	Read/Write	–	R / W						
	Resetting Value	–	0	0	0	0	0	0	0

DMADB1 (FFFAH)		7	6	5	4	3	2	1	0
	bit Symbol	–	DDB16	DDB15	DDB14	DDB13	DDB12	DDB11	DDB10
	Read/Write	–	R / W						
	Resetting Value	–	0	0	0	0	0	0	0

Destination address register
Used as a pair with the DMAD0, 1 register. The function

are the same as those of the DMASB0/1.

(g) DMAV0, 1: DMA vector register

DMAV0 (FFF8H)		7	6	5	4	3	2	1	0
	bit Symbol	DV07	DVO6	DV05	DV04	DV03	DV02	DV01	DV00
	Read/Write				W				
	Resetting Value	0	0	0	0	0	0	0	0

DMAV11 (FFFBH)		7	6	5	4	3	2	1	0
	bit Symbol	DV17	DV16	DV15	DV14	DV13	DV12	DV11	DV10
	Read/Write				W				
	Resetting Value	0	0	0	0	0	0	0	0

HDMA is started by interrupts. Consequently, the vector address of the interrupt that assigns HDMA start is loaded to the DMA vector register (DMAV0/1).
HDMA compares the interrupt vector and the contents of this register. If they match, HDMA operation starts.
It is necessary to set the vector address before generating the interrupt that starts HDMA.

(2) Register Loading

(a) DMAS0, 1
        DMAD0, 1 Loaded with the LDC instruction.
        DMAC0, 1 (*The LDC instruction is a new TLCS-90
                1 instruction.)
        DMAM0,

(b) DMASB0, 1
        DMADB0, 1 Load the input/output address with
                the LD instruction.
        DMAV0,   1 (See the separate address map
                concerning input/output addresses.)

(3) HDMA Start

HDMA can be started by any of the TMP90C051 maskable interrupt sources.

(a) Internal start factors

• SWI (software)
• All internal I/O interrupts

Assign starting of HDMA channel 0 or channel 1 to the INT0 - INT7 external interrupts, connect any of the bits of ports 2 - 6 (output mode) externally to INT0 - INT7 to generate a start interrupt.

(b) External start factors

• $\overline{NMI}$ pin
• INT0 ~ 7 pin

(4) HDMA Channel 0 and Channel 1 Priority Sequence

The channel where an interrupt is generated first has priority.

Note: HDMA, regardless of an interrupt enable flag, compares the vector and the values of the DMA V0/1 register. If they match in EI mode, the HDMA starts. Do not write the vector value of the non-maskable interrupt to the DMA V0/1 register. If doing so, the HDMA does not operate normally.
To stop the HDMA being started, set DI mode before generating the interrupt to start the HDMA, or set the DMA V0/1 register to 00H.

(5)   HDMA Operation Flow
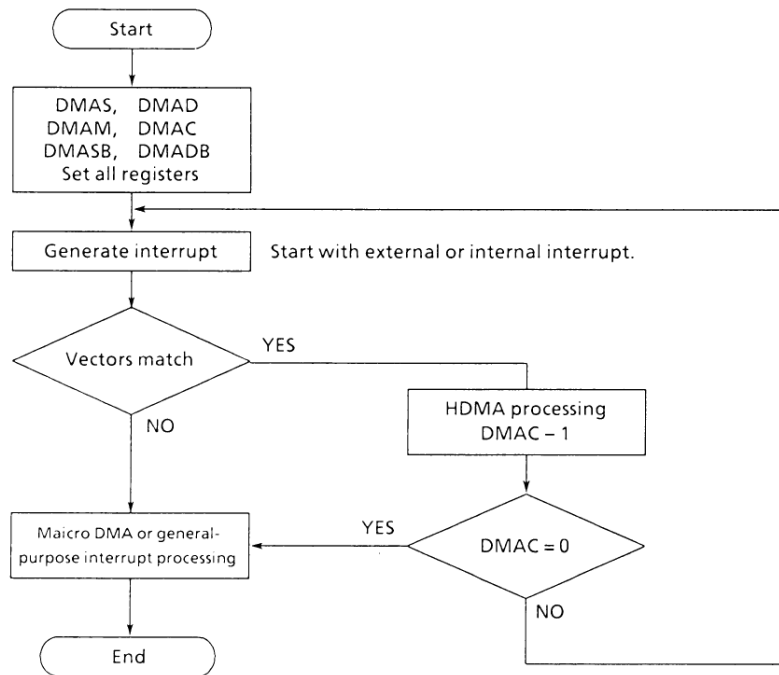
Figure 3.3 (6). HDMA Operation Flow

(6)   HDMA Operation Timing

(a) 1-byte transfer mode
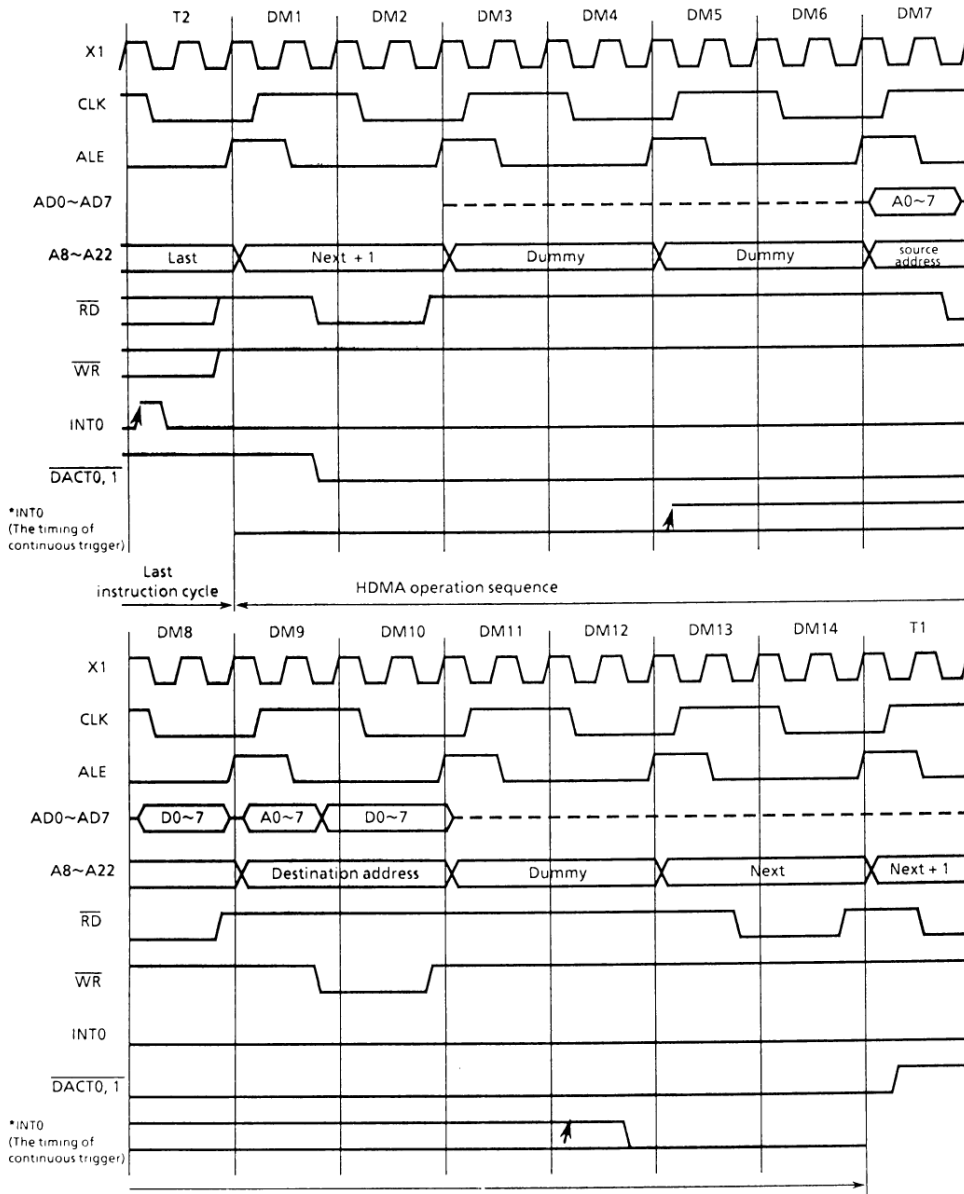
Figure 3.3 (7a). HDMA Operating Timing

(b) 2-byte transfer mode
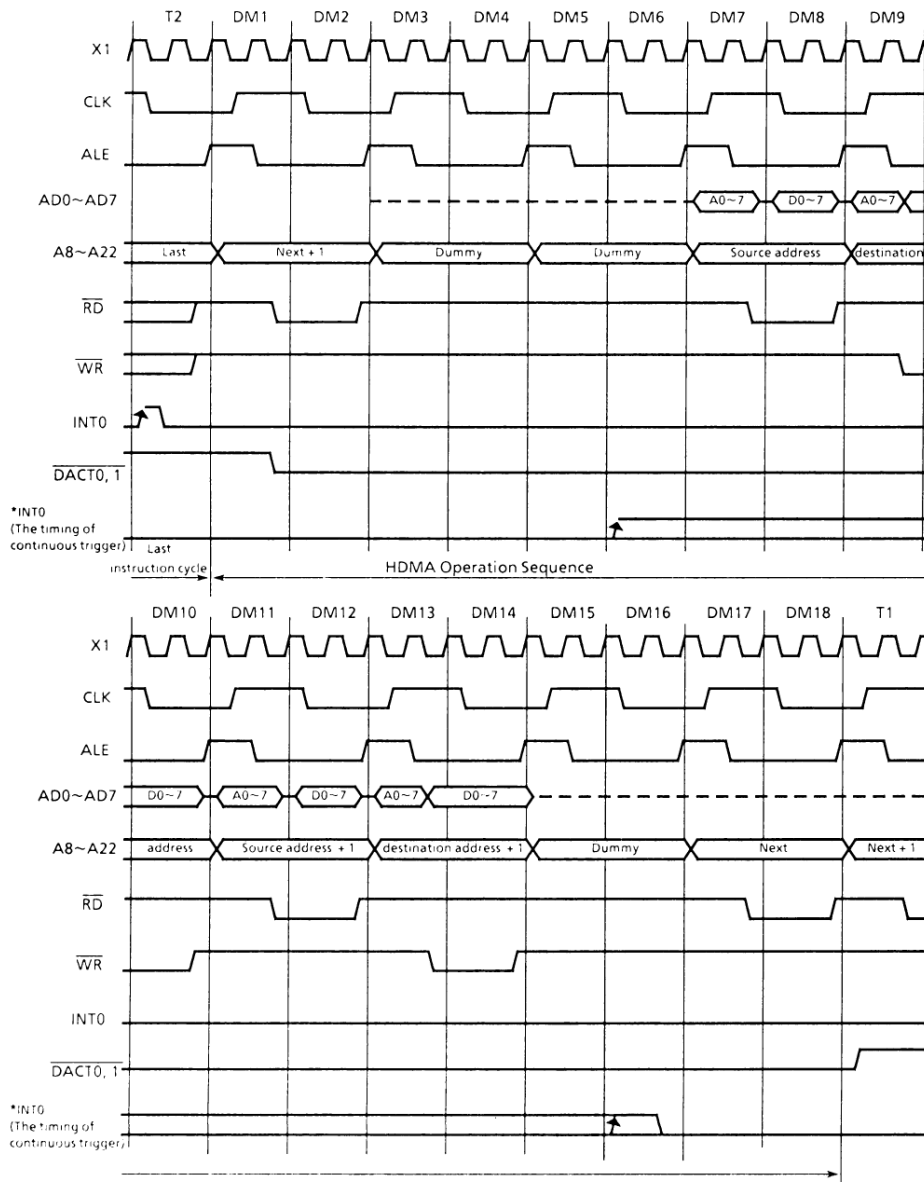
Figure 3.3 (7b). HDMA Operation Timing

3.3.4 Interrupt Controller

Figure 3.3 (9) shows an abbreviated interrupt circuit diagram. The left half of this diagram shows the interrupt controller and the right half shows the CPU interrupt request signal circuit and hold release circuit.

The interrupt controller has an interrupt request flipflop and interrupt enable/disable flag for each interrupt channel (total: 20 channels), and a micro DMA enable/disable flag. The interrupt request flip-flop latches interrupt requests that arrive from the periphery. This flipflop is reset to "0" when there is a reset, when the CPU receives an interrupt and reads the vector of that interrupt channel, and when an instruction that clears the interrupt request (writes "vector value/8" to memory address FFC9H) for that channel is executed.

LD (0FFC9H), 38H/8

For example, when LD (0FFC9H), 38H / 8 is executed, the interrupt request flipflop for the interrupt channel [INTT1] with the vector value 38H is reset to "0" (to clear the flipflop, also write to address FFC9H when the interrupt request flag is assigned to FFCAH and FFCBH).

Table 3.3 (5) shows the "interrupt vector value/8" values. The status of the interrupt request flipflop can be determined by reading memory address FFC9H, FFCAH or FFCBH. "0" means no interrupt request and "1" means an interrupt request. Figure 3.3 (8) shows the bit layout when the interrupt request flipflop is read.

Table 3.3 (5) Interrupt Vector Value/8 Values

Priority sequence	Type	Interrupt function name	Interrupt vector value	Vector value ÷ 8
1	Non	SWI instruction	08H	–
2	Maskable	NMI	10H	
3		(HDMA)	–	–
4		INT0	18H	03H
5		INT1	20H	04H
6		INT2	28H	05H
7		INTT0	30H	06H
8		INTT1	38H	07H
9		INTT2	40H	08H
10		INTT3	48H	09H
11		INTRX0	50H	0AH
12	Maskable	INTTX0	58H	0BH
13		INTLINE	60H	0CH
14		INTX1	68H	0DH
15		INTTX1	70H	0EH
16		INTALARM	78H	0FH
17		INT3	80H	10H
18		INT4	88H	11H
19		INT5	90H	12H
20		INT6	98H	13H
21		INT7	A0H	14H

Interrupt request flag

IRF0 (FFC9H)	bit Symbol	7	6	5	4	3	2	1	0
	bit Symbol	IRF7	IRF6	IRF5	IRF4	IRF3	IRF2	IRF1	IRF0
	Read/Write	R (Only IRF clear code can be written)							
	Resetting Value	0	0	0	0	0	0	0	0
	Function	Interrupt Request Flag 1: Interrupt requested (IRF is cleared to "0" by writting IRF clear Code)							

INT0 request flag
INT1 request flag
INT2 request flag
INT3 request flag
INT4 request flag
INT5 request flag
INT6 request flag
INT7 request flag

Note: The specified interrupt request flipflop is cleared by writing {vector value/8} to memory addresses FFC9H.

Figure 3.3 (8). Interrupt Request Flipflop Read (1/2)

Interrupt request flag

IRF1 (FFCAH)		7	6	5	4	3	2	1	0
	bit Symbol	–	–	–	IRFALARM	IRFT3	IRFT2	IRFT1	IRFT0
	Read/Write	–	–	–	R				
	Resetting value	–	–	–	0	0	0	0	0
	Function				1 : interrupt requested				

→ INTT0 request flag
→ INTT1 request flag
→ INTT2 request flag
→ INTT3 request flag
→ INTALARM request flag

Interrupt request flag

IRF2 (FFCBH)		7	6	5	4	3	2	1	0
	bit Symbol	–	–	–	IRFLINE	IRFRx1	IRFRx0	IRFTx1	IRFTx0
	Read/Write	–	–	–	R				
	Resetting value	–	–	–	0	0	0	0	0
	Function				1 : interrupt requested				

→ INTTX0 request flag
→ INTTX1 request flag
→ INTRX0 request flag
→ INTRX1 request flag
→ INTLINE request flag

Figure 3.3 (8). Interrupt Request Flipflop Read (2/2)

Figure 3.3 (9). Interrupt Controller Block

The interrupt enable/disable flags for each interrupt request channel are assigned to memory addresses FFCC - FFCEH. Interrupts are enabled for a channel by setting the flag to "1". The flags are reset to "0" by resets.

Disable interrupts (DI) before clearing the interrupt request flags.

The micro DMA enable/disable flags for each interrupt request channel are assigned to memory addresses FFCEH or FFCFH. The interrupt requests for a channel are set to the

micro DMA processing mode by setting the flag to "1". The flags are reset to "0" by resets ("0" is the general-purpose interrupt processing mode).

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

Disable interrupts (DI) before clearing the interrupt request flags.

Caution is required in usage because the two following points are exceptions.

INTRX0, INTRX1	Interrupt request flipflops are cleared only by resets and reading the serial channel receive buffer. They are not cleared by instructions.
INT0 ~ INT7	INT0 ~ INT7 are all of the edge type.

Interrupt enable flag

INTE0 (FFCCH)		7	6	5	4	3	2	1	0
	bit Symbol	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
	Read/Write				R / W				
	Resetting value	0	0	0	0	0	0	0	0
	Function			1 : Enable		0 : Disable			

- INT0　Interrupt enable flag
- INT1　Interrupt enable flag
- INT2　Interrupt enable flag
- INT3　Interrupt enable flag
- INT4　Interrupt enable flag
- INT5　Interrupt enable flag
- INT6　Interrupt enable flag
- INT7　Interrupt enable flag

Interrupt enable flag

INTE1 (FFCDH)		7	6	5	4	3	2	1	0
	bit Symbol	−	−	IELINE	IEALARM	IET3	IET2	IET1	IET0
	Read/Write	−	−			R / W			
	Resetting value	−	−	0	0	0	0	0	0
	Function			1 : Enable		0 : Disable			

- INTT0　Interrupt enable flag
- INTT1　Interrupt enable flag
- INTT2　Interrupt enable flag
- INTT3　Interrupt enable flag
- INTALARM　Interrupt enable flag
- INTLINE　Interrupt enable flag

Micro DMA and interrupt enable flag

INTE2 (FFCEH)		7	6	5	4	3	2	1	0
	bit Symbol	IERx1	IERx0	IETx1	IETx0	DERx1	DERx0	DETx1	DETx0
	Read/Write				R / W				
	Resetting value	0	0	0	0	0	0	0	0
	Function			1 : Enable		0 : Disable			

- INTTX0 μDMA enable flag
- INTTX1 μDMA enable flag
- INTRX0 μDMA enable flag
- INTRX1 μDMA enable flag
- INTTX0 Interrupt enable flag
- INTTX1 Interrupt enable flag
- INTRX0 Interrupt enable flag
- INTRX1 Interrupt enable flag

Figure 3.3 (10). Micro DMA Interrupt Enable Flags (1/2)

Micro DMA enable flags

DMAE (FFCFH)		7	6	5	4	3	2	1	0
	bit Symbol	DELINE	DETT3	DETT2	DETT1	DETT0	DET2	DET1	DET0
	Read/Write	R / W							
	Resetting Value	0	0	0	0	0	0	0	0
	Functionf	1 : Enable 0 : Disable							

INT0 μDMA Enable flag

INT1 μDMA Enable flag

INT2 μDMA Enable flag

INTT0 μDMA Enable flag

INTT1 μDMA Enable flag

INTT2 μDMA Enable flag
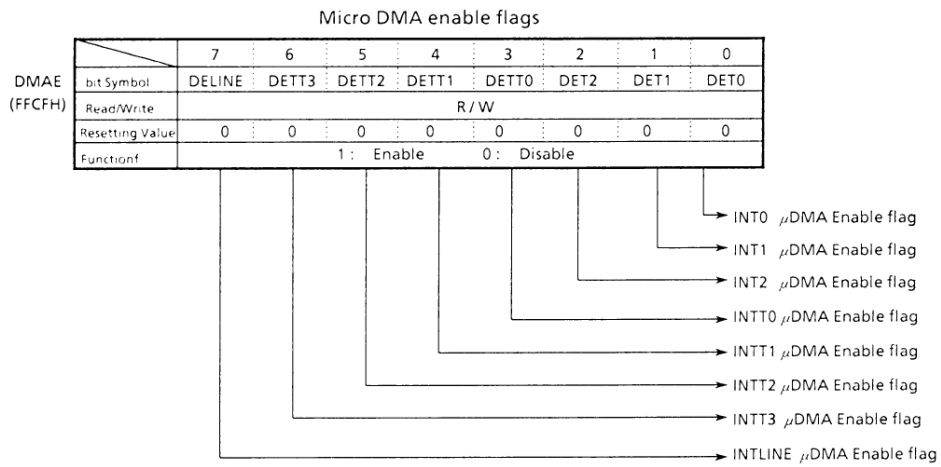
INTT3 μDMA Enable flag

INTLINE μDMA Enable flag

Figure 3.3 (10). Micro DMA Interrupt Enable Flags (2/2)

Figure 3.3 (11). Overall Interrupt Processing Flow

3.4 MMU

3.4.1 Address Area Extension

The TMP90C051F can access the extended address area (10000H -7FFFFFH) using the following three methods.

Address extension method

(1)   Extended data area access using CPU bank registers BX and BY.

(2)   Extended data area access during HDMA (high-speed DMA) transfers (source addresses bank register: DMAS0 and DMAS1; destination address bank register: DMADB0 and DMADB1).

(3)   Extended program area and extended data area access by the MMU.

As shown in figure 3.4 (1), the TMP90C051F can also use CLLAR (common local logical address register) of the MMU to specify common 0, local 0, common 1 and local 1 areas in the logical area at address 0000H - FFFFH.

Table 3.4 (1) shows in what areas of the logical address area the above three methods can be used.

Table 3.4 (1) Address Extensions for Each Area of the Logical Area

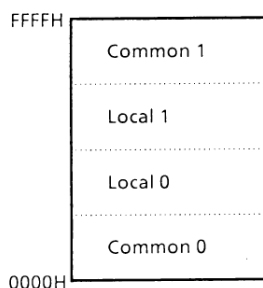| Logical area	BX, BY	MMU	DMASB0/1, DMADB0/1
Common 0	0	X	0
Local 0	0	0	0
Local 1	0	0	0
Common 1	0	X	0

Figure 3.4 (1). Logical Area Specification

i)   Extended data area access using the CPU bank registers
      BX and BY
      IX, IY, BX and registers

IX and IY are independent 16-bit registers called index registers. BX and BY are independent 7-bit registers called bank registers.

These registers are mainly used to specify memory addresses and to generate 23-bit addresses. The IX and IY registers are also used as 16-bit addition instruction registers.

The BX and BY registers are assigned to address FFECH (BX register) and address FFEDH (BY) in the built-in input/output register area. Only the lower 7 bits of these registers are effective; the uppermost bit is undefined. This undefined bit is always read as "1". The lower 7 bits of the BX and BY registers are initialized to "0" by resets.

BX (FFECH)	—	BX6	BX5	BX4	BX3	BX2	BX1	BX0	R / W
BY (FFEDH)	—	BY6	BY5	BY4	BY3	BY2	BY1	BY0	R / W
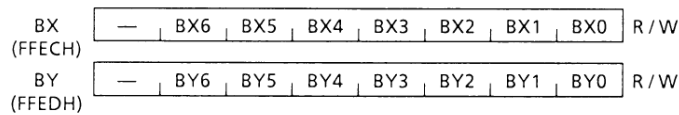
Extended data area addressing mode

The TMP90C051F can use up to 8M bytes of data memory.

Addresses 0000000H - 00FFFFH can be accessed in the normal addressing mode.

Addresses 010000H - 7FFFFFH are called the extended data area and are accessed in a special addressing mode.

To access the extended data area, an addressing mode (a mode that uses the register indirect addressing mode register pair IX or IY, or a mode that uses the index addressing mode register pair IX or IY) that uses the index addressing mode register pair IX or IY to calculate the operand address.

Thus, there are four types.

     (IX)
     (IY)
     (IX + d)
     (IY + d)

In these modes, the 23-bit address required for accessing the extended data area consists of a 16-bit offset (address bus A0 - A15) and a 7-bit bank address (address bus A16 - A22).

The 16-bit offset is the same as the calculated value of the conventional address.

The 7-bit bank address is specified with bank register BX or BY. BX is used to form a pair in modes that use the index register IX. BY is used to form a pair in modes that us the index register IY.

Note: When the extended data area addressing mode is used with the JR or CALL instruction, the value of the bank register that corresponds to that index register must be "0".

These instruction cannot use in expand memory area at MMU.

TOSHIBA CORPORATION
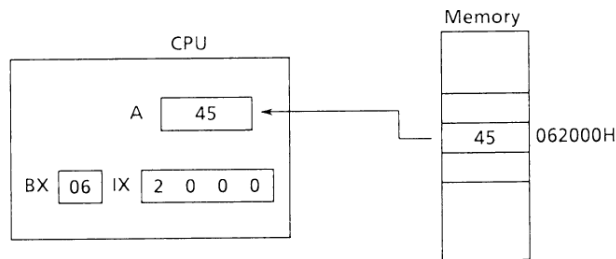
Example: LD A, (IX)

Memory data 45H at address 062000H is loaded to the A register.

In the index addressing mode, carries resulting from calculation of 16-bit offsets are ignored. There is no effect on the bank registers.

Example: LD A, (IY + 23H)
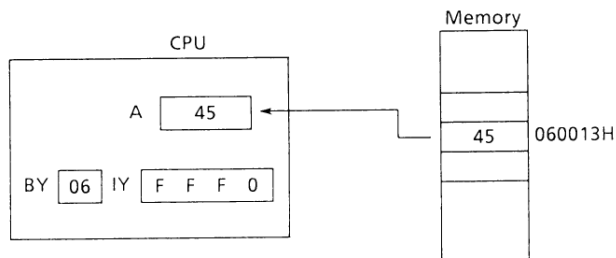
Memory data 45H at address 060013H is loaded to the A register.

In addressing modes (modes that do not use index register IX or IY to calculate the operand address) that do not access the extended data area, the 7-bit bank register (address bus A16 - A22) becomes "0" and the address range 000000H - 00FFFFH can then be accessed.

Note: When this addressing mode is used by the transfer destination, this write cycle is not performed correctly when the actual address is bank register address FFECH or FFEDH.

Example: LD (IX), 05H when the IX value is FFECH and BX value is "0".

ii)   Extended data area access during HDMA transfer

During HDMA transfers, memory addresses are indicated by the values loaded to the HDMA source address register (DMAS0 and DMAS1) and destination address register (DMAD0 and DMAD1). For extended data area access, memory addresses are indicated by the values loaded to the source bank register (DMASB0 and DMASB1) and destination bank register (DMADB0 and DMADB1).
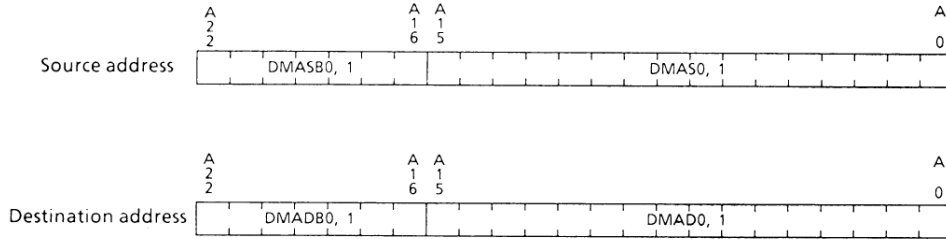
Figure 3.4 (2). Source Address/Destination Address Specification During HDMA Transfer
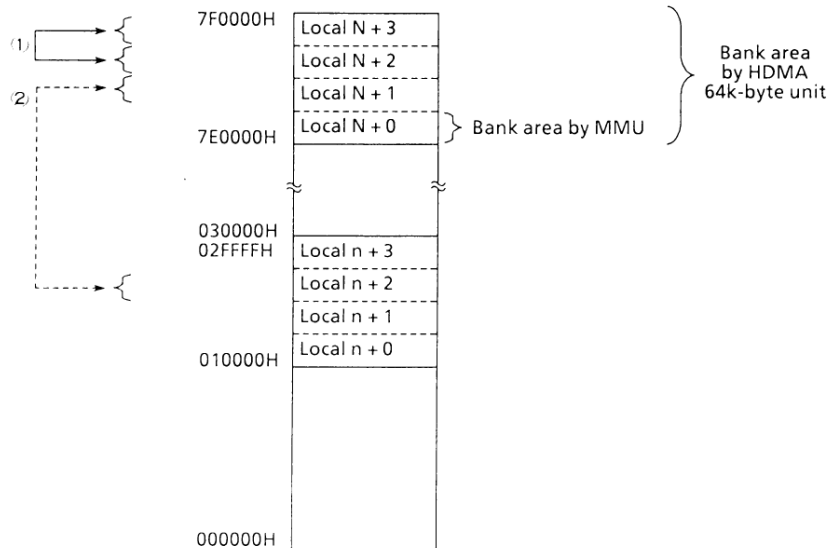
Figure 3.4 (3). Typical HDMA Data Transfer Between Banks (① ~ ②)

iii) <u>Program area and data area expansion using the MMU function</u>

The TMP90C051 offers program area and data area expansion, depending on the MMU.

- • MMU not used : Program area   0 - 64KB
                       : Data area   0 - 64KB
                       : When BX and
                           BY/HDMA are used   0 - 8MB

- • MMU used   : Program area   0 - 8MB
                       : Data area   0 - 8MB

3.4.2 MMU Functions

The following register expand the 64KB logical memory capacity (address 0000H - FFFFH) to a maximum of 8MB (addresses 000000H - 7FFFFFH).

(1) CLLAR: Common Local Logical Address Register (14-bit)

(2) EXPA0: Local 0 Expand Register (11-bit)

(3) EXPA1: Local 1 Expand Register (11-bit)

CLLAR is used to specify the logical addresses 0000H-FFFFH in the common 0, local 0, local 1and common 1 areas, as shown in Figure 3.4 (1).

EXPA0 is a bank register used to map the local 0 area logical addresses specified with CLLAR to the physical address space.

When the program area or data is indicated as being inside the local 0 area, the EXPA0 value is added to either the current program address value or memory address value to generate the physical addresses.

EXPA1 functions in the same way as EXPA0.
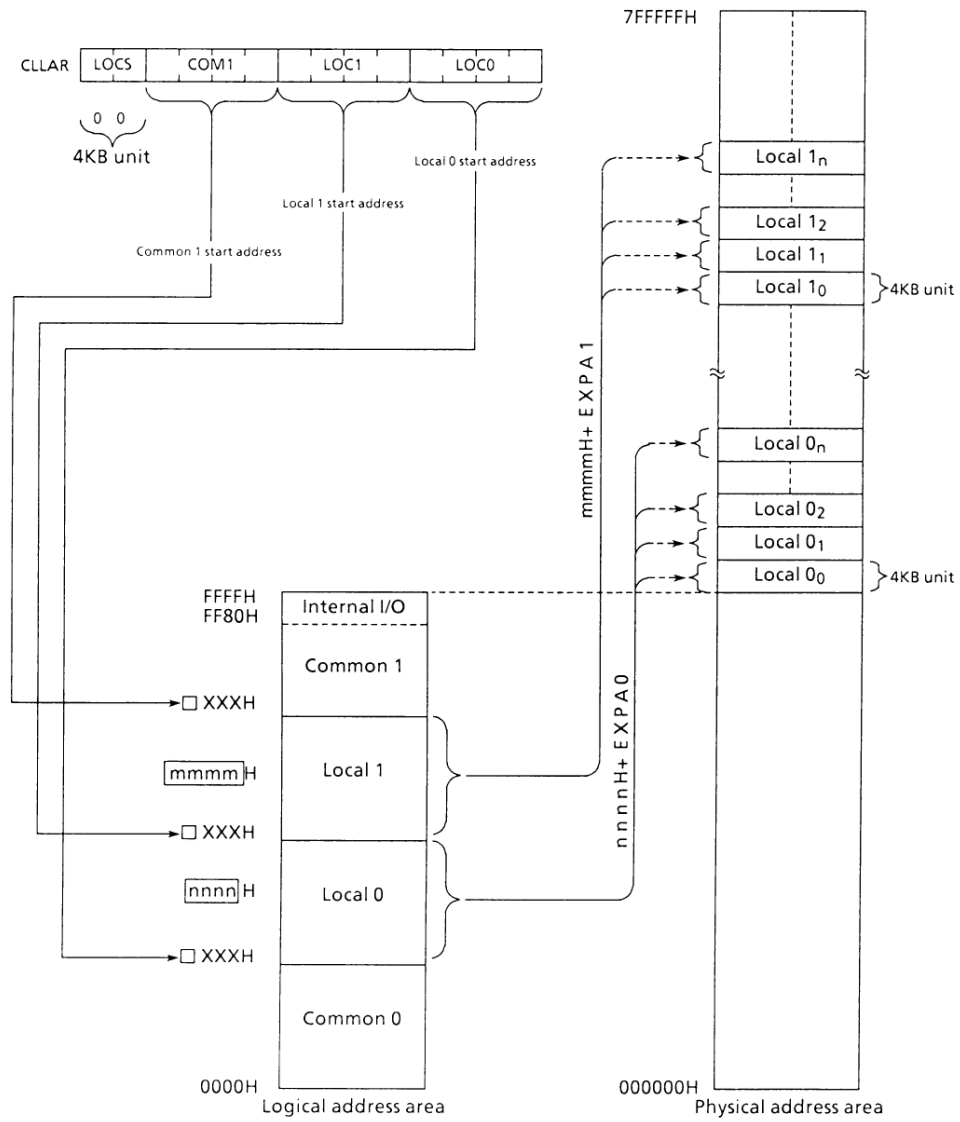
Figure 3.4 (4). Address Expansion by MMU (4KB units)

(1)   CLLAR (Common and Local Logical Address Register)

Common Local Logical Address Register L

CLLARL (FFEEH)		7	6	5	4	3	2	1	0
	bit Symbol	LOC13	LOC12	LOC11	LOC10	LOC03	LOC02	LOC01	LOC00
	Read/Write	R / W							
	Resetting Value	1	1	1	0	0	0	0	0
	Function	Specify local1 start address(A15~A12)				Specify local0 start address (A15~A12)			

Common Local Logical Register H

CLLARH (FFEFH)		7	6	5	4	3	2	1	0
	bit Symbol	—	—	LOCS1	LOCS0	COM13	COM12	COM11	COM10
	Read/Write	—	—	R / W					
	Resetting Value	—	—	0	0	1	1	1	0
	Function			Local size 00: 4K Byte 01: 8K Byte 10: 16K Byte 11: not used		Specify common 1 start address(A15~A12)			

Figure 3.4 (5). Configuration

As shown in Figure 3.4 (5), CLLAR has a 14-bit configuration (8 lower bits and 6 upper bits).

The start addresses of common 0, local 0, local 1 and common 1 in the logical address area shown in Figure 3.4 (4) are specified with CLLAR. That is, up to 4 address areas can be specified with CLLAR.

When the 16-bit transfer instruction is used, LOCS and COM1 are the upper addresses and LOC1 and LOC0 are the lower addresses of the CLLAR setting.

LOCS and COM1, LOC1 and LOC0 can also be set individually with 8-bit transfer instructions.

CLLAR can be read and written with either 8-bit or 16-bit transfer instructions.

i)   LOCS (local size)

This register specifies the minimum local 0 and local 1 unit (size).

D5	D4		
0	0	:	4 K-byte unit (after reset)
0	1	:	8 K-byte unit
1	0	:	16 K-byte unit

ii)   <u>COM1, LOC1 and LOC0 (common and local start addresses)</u>

These registers specify the 4 upper bits of the start addresses of common 1, local 1 and local 0 in the 0000H - FFFFH logical address area.

Since the minimum unit for the COM1, LOC1 and LOC0 areas is determined by LOCS, the start address of each area is as shown in Figure 3.4 (6).

After a reset, COM1 = EH, LOC = EH and LOC0 = 0H.

Note: The minimum unit for the area size is determined by LOCS, and defined by the start address of each area.

Setting example: (The minimum unit : 4K byte)

When LOCS = 00H
       LOC0 = 01H
       LOC1 = 08H
       COM1 = 0FH

FFFFH F000H	Common 1	4K-byte
	Local 1	28K-byte
8000H		
	Local 0	28K-byte
1000H		
0000H	Common 0	4K-byte

TOSHIBA CORPORATION
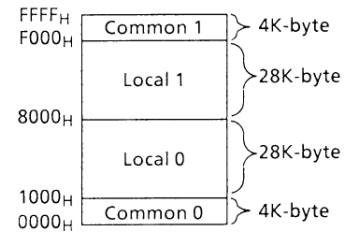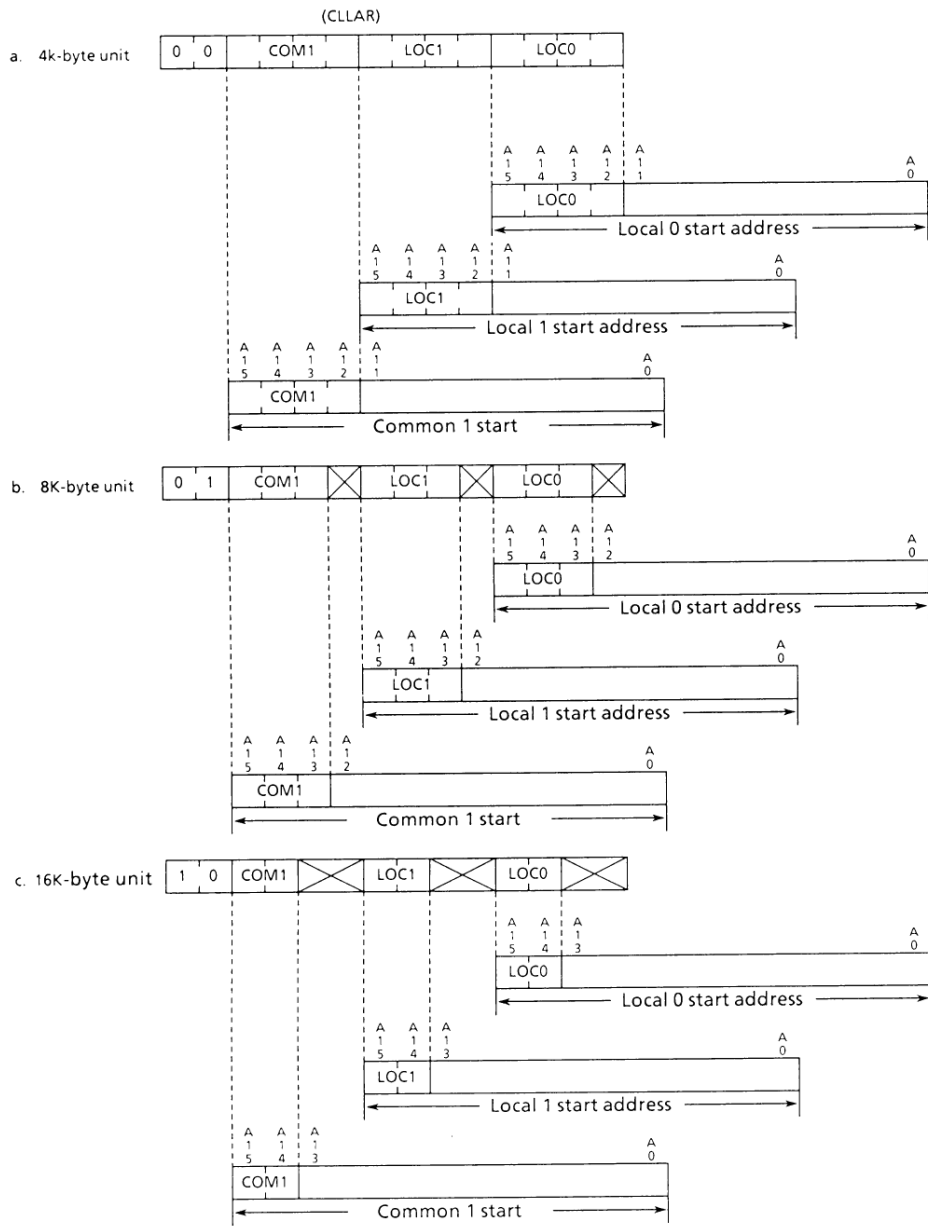
Figure 3.4 (6). Local Size (LOCS) and Start Address Specification

iii)   Address area specification and types

In the logical address area (0000H - FFFFH), the address areas are set using the common and local register (CLADR) as shown below. MMU will not operate correctly if any combination not shown here is used.

Always set the CLADR register to COM1 ≥ LOC1 ≥ LOC0.
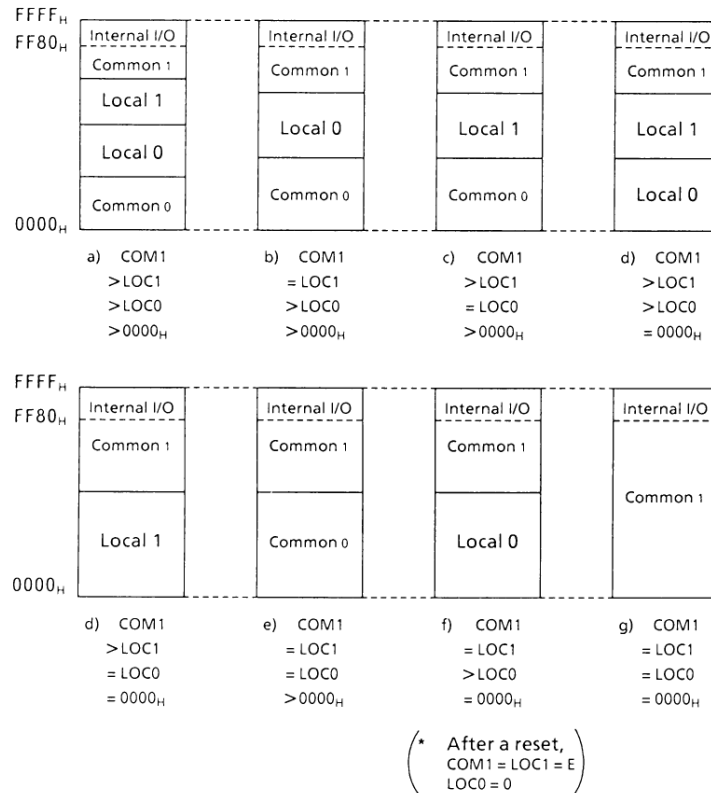
Note: COM1, LOC1 and LOC0 are set to match the local size (LOCS) as shown below.

(2)   EXPA0 (Expand Register 0) and EXPA1 (Expand Register 1)

Expand register 0H

EXPA1L (FFF0H)		7	6	5	4	3	2	1	0
	bit Symbol	—							
	Read/Write	R/W							
	Resetting Value	0	0	0	0	0	0	0	0

Expand register 1H

EXPA1H (FFF1H)		7	6	5	4	3	2	1	0
	bit Symbol	—	—	—	—	—	—		
	Read/Write	—	—	—	—	—	R/W		
	Resetting Value	—	—	—	—	—	0	0	0

Expand register 0L

EXPA0L (FFF2H)		7	6	5	4	3	2	1	0
	bit Symbol	—							
	Read/Write	R/W							
	Resetting Value	0	0	0	0	0	0	0	0

Expand register 0H

EXPA0H (FFF3H)		7	6	5	4	3	2	1	0
	bit Symbol	—	—	—	—	—	—		
	Read/Write	—	—	—	—	—	R/W		
	Resetting Value	—	—	—	—	—	0	0	0

Figure 3.4 (7). EXP0, EXPA1 Configuration

EXPA0 is set to map the logical address area in local 0 to the physical address area when the local 0 address area is accessed by the CPU. All bits are set to "0" by resets.

EXPA1 is set to map the logical address area in local 1 to the physical address area when the local 1 address area is accessed by the CPU. All bits are set to "0" by resets.

For both EXPA0 and EXPA1, the 16-bit transfer instruction loads the upper address bits to A22 - A20 and the lower address bits to A19 - A12. Loading is also possible with 8-bit transfer instructions.

Both EXPA0 and EXPA1 can be read and written.

Use the common area (in the logical address area) to write overwrite EXPA0 and EXPA1. When loading to the logical address area, the contents of EXPA0 and EXPA1 are mapped to the physical address area immediately after the instruction used to load is executed.

Note: When overwriting EXPA0 and EXPA1 in the physical address area, use caution concerning the addresses in the physical address area to which the addresses in the current logical address area are mapped. This is because the addresses are mapped to a different physical address area immediately after the instruction used to overwrite is executed.

(3)   Physical Address Area Specification

When a program area is located above 64k bytes (0000H - FFFFH), the logical address area can be mapped to the physical address area by accessing the local 0 and local 1 address area specified with the common and local address register.

When mapping to the physical area, the address is indicated by adding the values of the local 0 expand register (EXPA0), local 1 expand register (EXPA1) and the logical addresses in the local 0 and local 1 area. When a program is executed in a common area, zeros are output to A22 - A20 and the logical address is output to A12 - A15.

Physical address generation
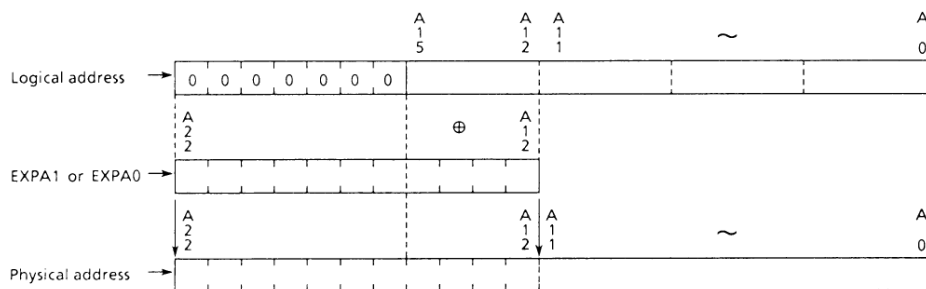
Figure 3.4 (8). Physical Address Generation Using 4k-byte Units
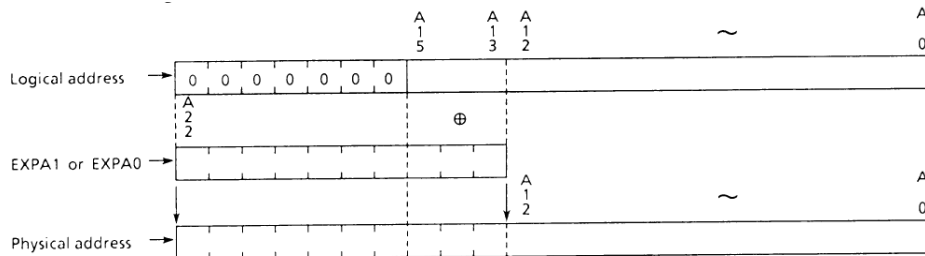
Figure 3.4 (9). Physical Address Generation Using 8k-byte Units

TOSHIBA CORPORATION
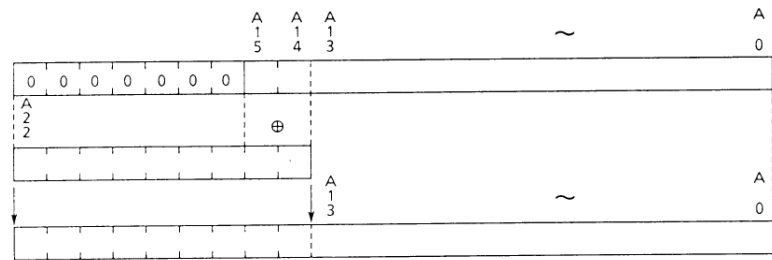
Figure 3.4 (10). Physical Address Generation Using 16k-byte Units

3.4.3 MMU Usage (Software)

(1)   Typical MMU Setting

MMU register	Setting value
CLLAR	$0EE0_H$
EXPA0	000_H
EXPA1	000_H

Logical area:
- $FFFF_H$
- Common 1
- $E000_H$
- $DFFF_H$
- Local 0
- 0000_H

Physical area:
- $7FFFFF_H$
- $00FFFF_H$
- $00E000_H$
- $00DFFF_H$
- 000000_H

Figure 3.4 (11). After Reset

MMU register	Setting value
CLLAR	$0E84_H$
EXPA0	030_H
EXPA1	058_H

Logical area:
- Common 1
- $E000_H$
- $DFFF_H$
- Local 1
- 8000_H
- $7FFF_H$
- Local 0
- 4000_H
- $3FFF_H$
- Common 0
- 0000_H

Physical area:
- $7FFFFF_H$
- 060000_H
- $065FFF_H$
- 060000_H
- $037FFF_H$
- 034000_H
- $003FFF_H$
- 000000_H

Figure 3.4 (12). Typical MMU Settings – 1
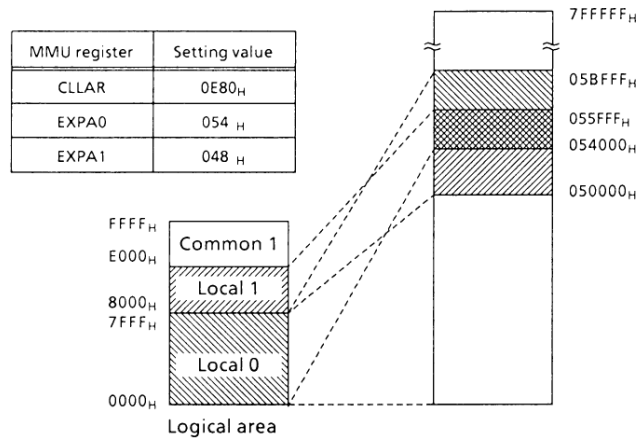
Figure 3.4 (13). MMU Settings –2
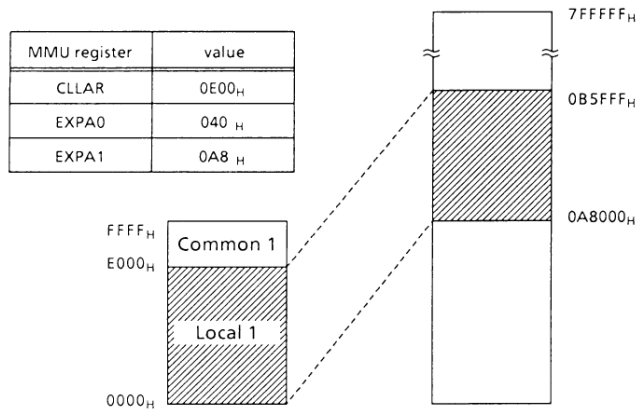
Figure 3.4 (14). MMU Settings –3

3.5 Standby Functions

Executing the HALT instruction sets the TMP90C051 to the RUN, IDLE1 or STOP mode, depending on the contents of the halt mode setting register. The features are shown below.

(1) Run: Only the CPU halts, power consumption remains the same.

(2) IDLE: Only the internal oscillators operate, all others internal circuitry halts. Power consumption is 1/10 or less of that during operation.

(3) STOP: All internal circuitry halts, including the oscillators. Power consumption is extremely reduced.

The HALT mode setting WDMOD <HLTM1, 0> is assigned to bits 2 and 3 memory address FFDDH in the built-in input/output register area (all other bits are used to control other block functions). The "00" RUN mode is set by resets.

The halt status is released by interrupt requests and resets. If interrupts are enabled, the CPU receives non-maskable and maskable interrupts and starts interrupt processing. If maskable interrupts are disabled, the CPU restarts execution from the instruction following the HALT instruction but the interrupt request flag remains at "1".

If the halt status is released by reset, however, use caution since the status in effect before entering the halt status cannot be maintained. In such cases, it is recommended that interrupt requests be used for releasing.

WDMOD (FFDDH)		7	6	5	4	3	2	1	0
	bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRIVE
	Read/Write	R / W						R / W	
	Resetting value	1	0	0	0	0	0	0	0
	Function	1 : WDT Enable	WDT Detection time 00: $2^{16}/fc$ 01: $2^{18}/fc$ 10: $2^{20}/fc$ 11: $2^{22}/fc$		Warming Up time 0: $2^{14}/fc$ 1: $2^{16}/fc$	Standby mode 00: RUN 01: STOP 10: IDLE 11: —		Watchdog timer out control 1 : Pin 69 is set as the WDTOUT ,pin and is connected internally to the RESET pin.	1: Drive the pin even in the STOP mode.

Note : <RESCR> bit is setted by "0" after Reset and P35 pin function is WDTOUT pin.

Figure 3.5 (1). Halt Mode Setting Register

3.5.1 RUN Mode

Figure 3.5 (2) shows the timing used to release halts using interrupts in the RUN mode.

In the RUN mode, the system clock in the MCU does not stop after the HALT instruction is executed; the CPU merely stops executing instructions. That is, the CPU repeats dummy cycles until the halt status is released. In the halt status, interrupt requests are sampled at CLK signal falls.
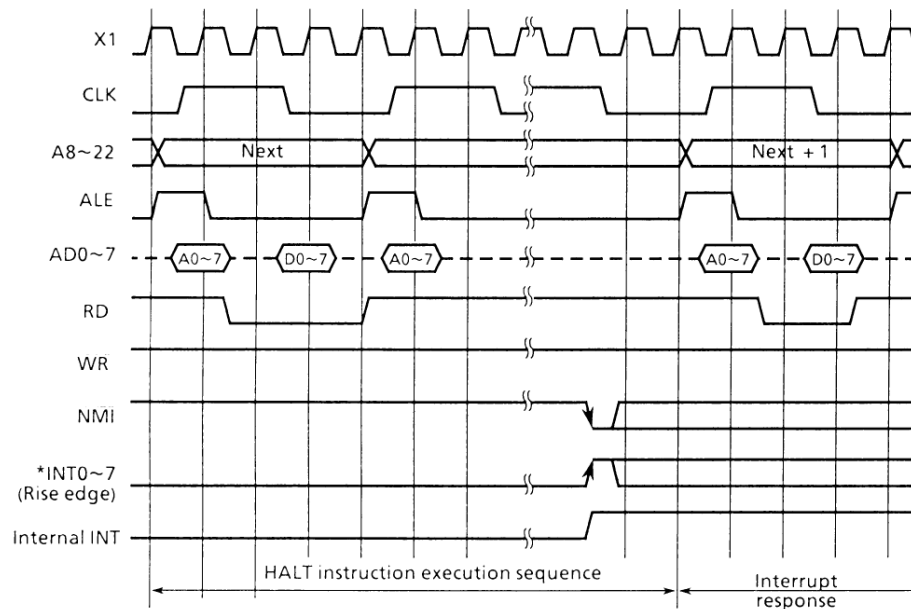
Figure 3.5 (2). Halt Release Timing Using Interrupts in the RUN Mode

*: Halt status can be released with external interrupts (INT1 - INT7) only in the RUN mode.

3.5.2 IDLE Mode

Figure 3.5 (3) shows the timing used to release halts with interrupts in the IDLE mode.

In the IDLE mode, only the internal oscillator operations, TMP90C051F system clock stops and the CLK signal is fixed at "1".

In the halt mode, interrupt requests are sampled asynchronously with the system clock but sampling is performed synchronously when the halt mode is released.

Note: In this mode, only external interrupt ($\overline{\text{NMI}}$ and INT0) are enabled during the halt interval.
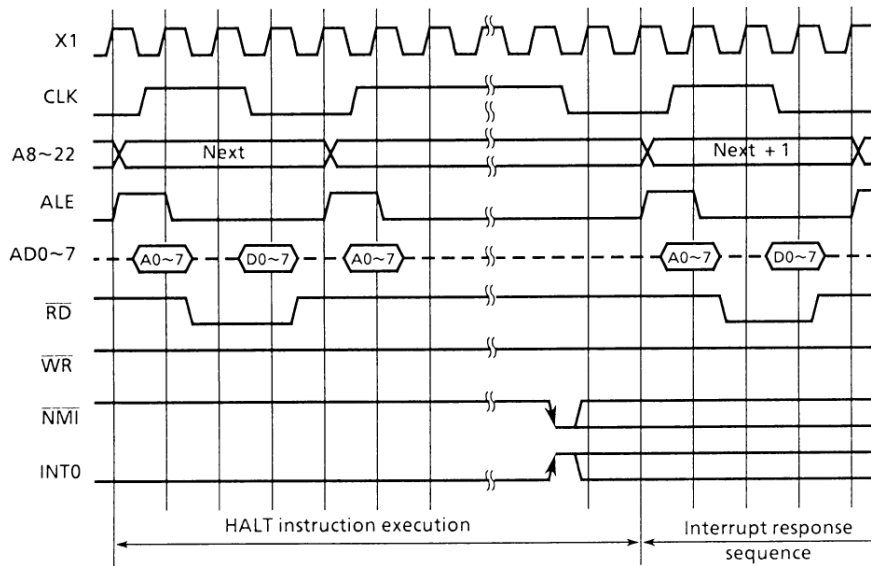
Figure 3.5 (3). Halt Release Timing Using Interrupts in the IDLE Mode

TOSHIBA CORPORATION

3.5.3 STOP Mode

In the STOP mode, all internal circuitry stops, including the internal oscillator. When the STOP mode is activated, all but certain pins are isolated from TMP90C051F by being set to high impedance.

All interrupt requests are disabled during the hat interval in this mode (leave the external non-maskable interrupt pin (NMI) at the "1" level). Consequently, this mode can only be released by resetting. Ports with programmable pull-up resistance remain pulled up.

Table 3.5 shows the status of each pin in the STOP mode. The status in effect before the halt continues if WDMOD <DRIVE> (drive enable: bit 0 of memory address FFDDH) of the built-in input/output register is set to "1". This register is cleared to "0" by resets.

The internal oscillator can also be restarted by inputting the RESET signal "0" to the CPU; however, operation is sometimes not correct after power-on due to clock instability immediately after the internal oscillator restarts. When releasing a halt by resetting in the STOP mode, it is necessary to keep the RESET signal at "0" for a sufficient time.

The internal oscillator also restarts by entering the RESET signal "0" to the CPU; however, the warming up counter does not operate due to quick power-on operation. Because of clock instability immediately after restarting the internal oscillator, the CPU may not operate correctly. When releasing a halt by resetting in STOP mode, the RESET signal must be kept at "0" for a sufficient time.
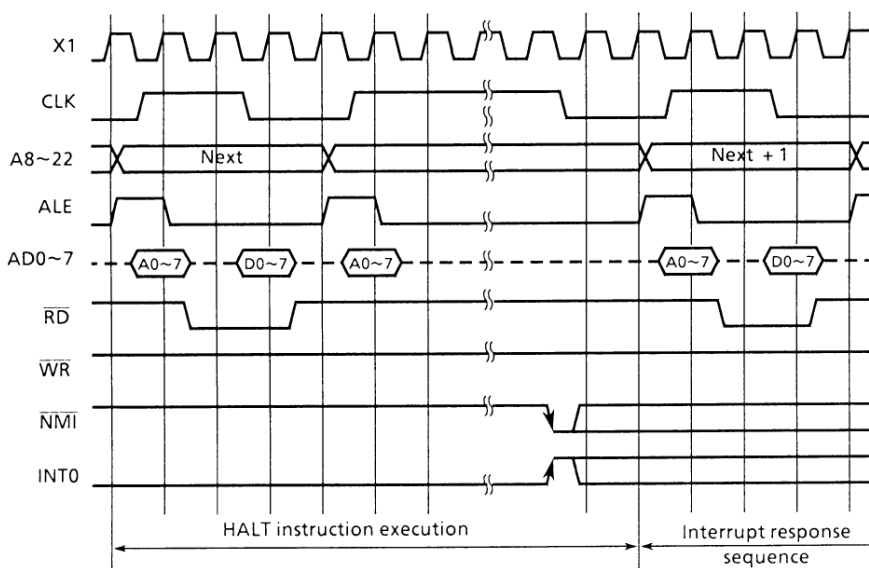
Figure 3.5 (4). Halt Release Timing Using Interrupts in the STOP Mode

Table 3.5 State of Pin in STOP Mode

	IN/OUT	DRZVE = 0	DRZVE = 1
AD0 ~ AD7	3 State	–	–
A8 ~ A15	OUT Pin	–	OUT
P20 ~ P26	OUT Pin	–	OUT
P30 ~ P35	OUT Pin	–	OUT
P36 ~ P37	INPUT Mode OUTPUT Mode	– –	IN OUT
P4	INPUT Mode OUTPUT Mode	– OUT	IN OUT
P5	INPUT Mode OUTPUT Mode	– OUT	IN OUT
P6	INPUT Mode OUTPUT Mode	– –	IN OUT
ALE	OUTPUT Pin	"0"	"0"
CLK	OUTPUT Pin	–	"1"
$\overline{\text{RESET}}$	INPUT Pin	IN	IN
X1	INPUT Pin	–	–
X2	OUTPUT Pin	"1"	"1"
RD	OUTPUT Pin	–	OUT
WR	OUTPUT Pin	–	OUT
$\overline{\text{WAIT}}$	INPUT Pin	–	IN
$\overline{\text{NMI}}$	INPUT Pin	IN	IN
INT0	INPUT Pin	IN	IN
INT1 ~ 2	INPUT Pin	-	IN
$\overline{\text{LATWAIT}}$/RXD0	INPUT Pin	–	IN
TPHSD/TXD0	OUTPUT Pin	–	OUT
TPHCKO/SCLK0	INPUT Mode OUTPUT Mode	– –	IN OUT
TPHCKI/RTS0	INPUT Mode OUTPUT Mode	– –	IN OUT
$\overline{\text{TPHLAT}}$/$\overline{\text{CTS0}}$	INPUT Mode OUTPUT Mode	– –	IN OUT
RXD1	INPUT Pin	–	IN
TXD1	OUTPUT Pin	–	OUT
SCLK1	INPUT Mode OUTPUT Mode	– –	IN OUT
$\overline{\text{RTS1}}$	OUTPUT Pin	–	OUT
$\overline{\text{CTS1}}$	INPUT Pin	–	IN
$\overline{\text{BAKEN}}$	INPUT Pin	IN	IN
CXIN	INPUT Pin	IN	IN
CXOUT	OUTPUT Pin	OUT	OUT
$\overline{\text{ALARM}}$	OUTPUT Pin	OUT	OUT

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

OUT : The output status.

IN : The input status.

3.6 Function of Ports

The TMP90C051F contains a of total 31 bits 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 3.6 describes the functions of these ports.

Table 3.6 Functions of Ports

Port name	Pin name	No. of pins	Direction	Direction set unit	Pin name for internal function
Port 2	P20	1	Output	–	A16
	P21	1	Output	–	A17
	P22	1	Output	–	A18
	P23	1	Output	–	A19
	P24	1	Output	–	A20
	P25	1	Output	–	A21
	P26	1	Output	–	A22
Port 3	P30	1	Output	–	$\overline{RAS}$
	P31	1	Output	–	$\overline{CAS}$
	P32	1	Output	–	$\overline{CS}$
	P33	1	Output	–	$\overline{DACT0}$
	P34	1	Output	–	$\overline{DACT1}$
	P35	1	Output	–	$\overline{WDTOUT}$
	P36	1	I/O	–	INT7
	P37	1	I/O	–	–
Port 4	P40 ~ P43	4	I/O	bit	M00 ~ M03
Port 5	P50 ~ P53	4	I/O	bit	M10 ~ M113
Port 6	P60	1			INT3
	P61	1			INT4
	P62	1			INT5
	P63	1	I/O	bit	INT6
	P64	1			TI0
	P65	1			TI2
	P66	1			T00/T01
	P67	1			T02/T03

These port pins function as the general-purpose input/output ports by resetting. The port pins, for which input or output is programmably selectable, function as input ports by resetting.

A separate program is required to use them for an internal function.

3.6.1 Port (P20 - P26)

Port 2 is a 7-bit general-purpose output port (P2: memory address FFB4H).

P20 - P26 are used both as output only ports and as the expansion address bus (A16 - A22). The control register P2CR: memory address FFB5H) is used to specify which mode is to be used. Reset operations clear the P20 - P26 output latch and all control register bits to "0" to set P20 - P26 to the general-purpose output port mode.

Figure 3.6 (1). Port 2 (P20 - P26)

Port 2 register

P2 (FFB4H)		7	6	5	4	3	2	1	0
	bit Symbol	−	P26	P25	P24	P23	P22	P21	P20
	Read/Write	−	R / W						
	Resetting Value	−	0	0	0	0	0	0	0

Port 2 control register

P2CR (FFB5H) (RMW disable)		7	6	5	4	3	2	1	0
	bit Symbol	−	P26C	P25C	P24C	P23C	P22C	P21C	P20C
	Read/Write	−	W						
	Resetting Value	−	0	0	0	0	0	0	0
	Function		0 : output port 1: Address Bus (control in 1 bit units)						

→ set output function of port.

0	Output port
1	Extended address output

Port 2 function register

P2FR (FFDFH)		7	6	5	4	3	2	1	0
	bit Symbol	SCLKC1	SCLKC0	ODE1	ODE0	WAITC1	WAITC0	−	Fixed at "1"
	Read/Write	R / W						−	R/W
	Resetting Value	0	0	0	0	0	0	−	1
	Function	SCLK1 control 0: SCLK1 input 1: SCLK1 output	SCLK0 control 0: SCLK0 input 1: SCLK0 output	TXD1 control 0:CMOS output 1:Open drain output	TXD1 control 0:CMOS output 1:Open drain output	Wait control 00: 2state Wait 01: normal Wait 10: non Wait 11: reserved			Write "1".

See "3.9 Serial Channel"

→ WAIT control

00	2-state wait
11	Normal wait
10	No wait
11	Reserve

Figure 3.6 (2). Register for Port 2

3.6.2 Port 3 (P30 - P37)

Port 3 is an 8-bit general-purpose I/O port (P3: memory address FFB6H). P30 - P35 are output only ports; P36 and P37 are I/O ports. The control register (P3CR: memory address FFB7H) is used to set P36 and P37 as input or output ports.

    Reset operations set the P30 - P35 output latch to "1", and clear the P36 and P37 output latch and all control register bits to "0" to set P36 and P37 to the input mode.

    In addition to the general-purpose I/O port function, there are also interrupt request input, chip select ($\overline{\text{CS}}$) and DMA active ($\overline{\text{DACT0}}$, $\overline{\text{DATC1}}$, $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, $\overline{\text{WD6OUT}}$) output functions.

(1)   P30 - P34

P30 - P34 are used both as general-purpose output ports for $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, $\overline{\text{CS}}$, $\overline{\text{DACT0}}$ and $\overline{\text{DACT1}}$. $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, $\overline{\text{CS}}$, $\overline{\text{DACT0}}$ and $\overline{\text{DACT1}}$ output is set using bits 0 - 4 of the port 3 function register (P3FR: memory address FFB8H).
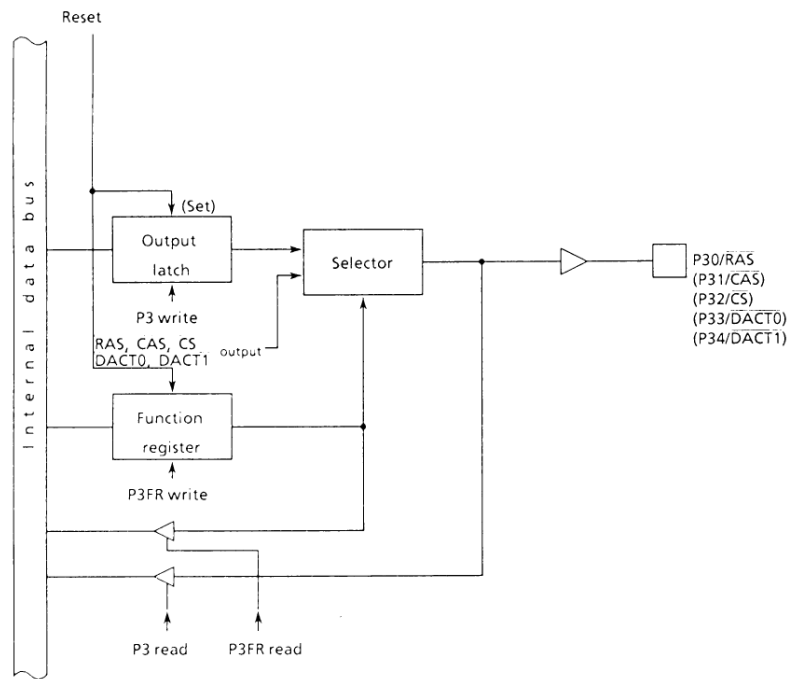
Figure 3.6 (3). Port 3 (P30 - P34)

          TOSHIBA CORPORATION

(2) P35

P35 is used both as a general-purpose output port for

$\overline{\text{WDTOUT}}$ output. Bit 1 of the watchdog timer mode register (WDMOD: memory address FFDDH) is used to set P35 for $\overline{\text{WDTOUT}}$ output.
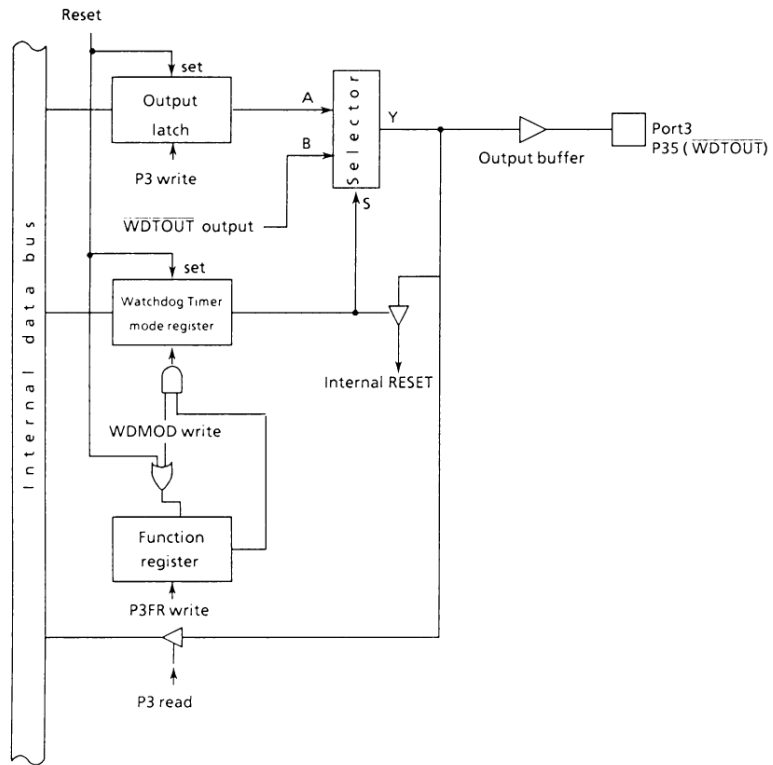
Figure 3.6 (4) Port 3 (P35)

(3)　P36

P36 is used both as a general-purpose I/O port and for external interrupt request input INT7.
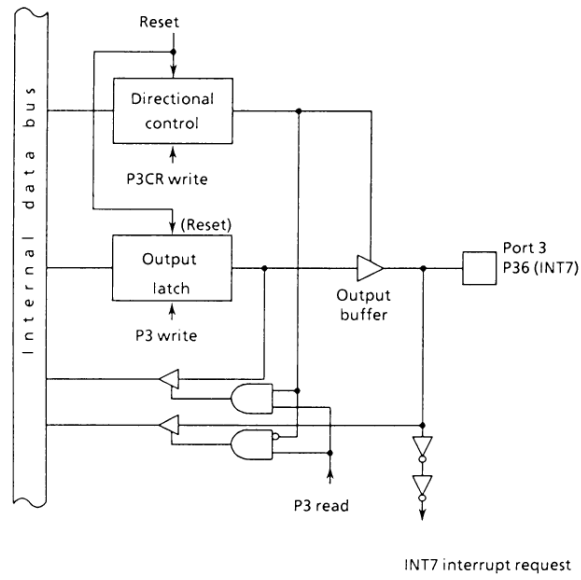
Figure 3.6 (5). Port 3 (P36)

(4)   P37

Port 37 is a general-purpose I/O port.

Figure 3.6 (6). Port 3 (P37)

Port 3 register

P3 (FFB6H)		7	6	5	4	3	2	1	0
	bit Symbol	P37	P36	P35	P34	P33	P32	P31	P30
	Read/Write				R / W				
	Resetting Value	input	input	1	1	1	1	1	1

Port 3 Control register

P3CR (FFB7H) (R/W disable)		7	6	5	4	3	2	1	0
	bit Symbol	P37C	P36C	–	–	–	–	–	–
	Read/Write	W		–	–	–	–	–	–
	Resetting Value	0	0	–	–	–	–	–	–
	Function	0 IN, 1 OUT (Specifies input/ output in 1 bit units)							

→ Select input/output of port 36, 37

0	input
1	output

Port 3 Function register

P3FR (FFB8H) (RMW disable)		7	6	5	4	3	2	1	0
	bit Symbol	–	–	WDTOUTC	DACT1C	DACT0C	CSC	CASC	RASC
	Read/Write	–	–	W			R / W		
	Resetting Value	–	–	0	0	0	0	0	0
	Function	–	–	P35 control 0: disable 1: enable	P34 control 0: input/ output port 1: $\overline{DACT1}$	P33 control 0:input/ output port 1: $\overline{DACT0}$	P32 control 0:input/ output port 1: $\overline{CS}$	P31 control 0:input/ output port 1: $\overline{CAS}$	P30 control 0:input/ output port 1: $\overline{RAS}$

→ Select P30 function

0	P30 output
1	$\overline{RAS}$ output

→ Select P31 fucntion

0	P31 output
1	$\overline{CAS}$ output

→ Select P32 function

0	P32 output
1	$\overline{CS}$ output

→ Select P33 function

0	P33 output
1	$\overline{DACT0}$ output

→ Select P34 function

0	P34 output
1	$\overline{DACT1}$ output

→ Select P35 function control

0	disable
1	enable

Figure 3.6 (7). Registers for Port 3

3.6.3 Port 4 (P40 - P43)

Port 4 is a 4-bit general-purpose I/O port (P4: memory address FFB9H), each bit of which can be set individually for input or output. The control register (bits 0 - 3 of P45CR: memory address FFBBH) is used to set the bits for input or output. Reset operations clear this control register to "0" to set port 4 to the input mode.

This port is also used as stepping motor control/pattern generation port 0 (M00 - M03). The function register (bits 0 - 3 of P45FR: memory address FFBCH) is used to select general-purpose I/O port or stepping motor control/pattern generation port. Reset operations set port 4 to the general-purpose I/O mode.
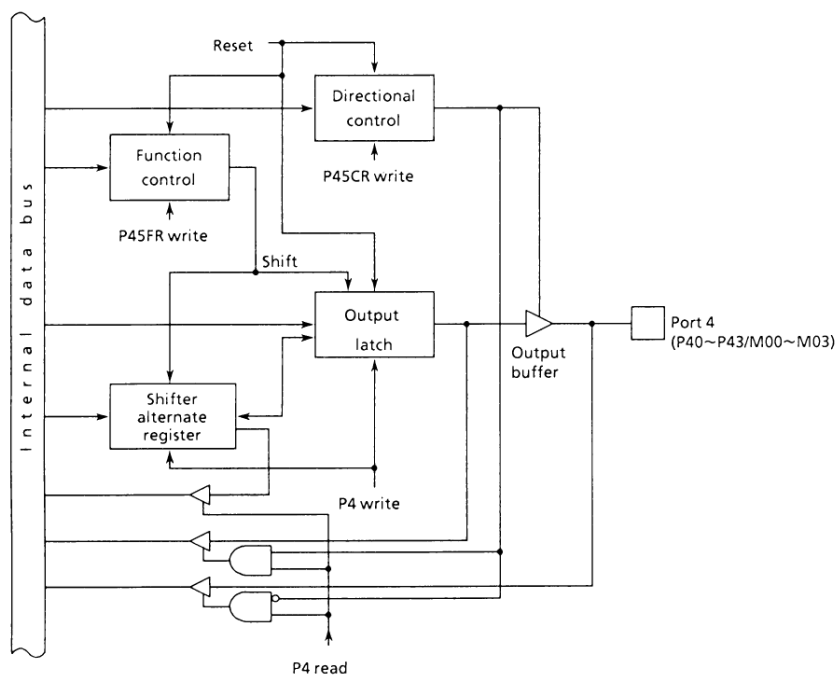
Figure 3.6 (8). Port 4

3.6.4 Port 5 (P50 - P53)

Port 5 is a 4-bit general-purpose I/O port (P5: memory address FFBAH), each bit of which can be set individually for input or output. The control register (bits 4 - 7 of P45CR: memory address FFBBH) is used to set the bits for input or output. Reset operations clear this control register to "0" to set port 5 to the input mode.

This port is also used as stepping motor control/pattern generation port 1 (M10 - M13). The function register (bits 4 - 7 of P45FR: memory address FFBCH) is used to select general-purpose I/O port or stepping motor control/pattern generation port. Reset operations set port 5 to the general-purpose I/O mode.

Figure 3.6 (9). Port 5

Port 4 register

P4 (FFB9H)		7	6	5	4	3	2	1	0
	bit Symbol	SA43	SA42	SA41	SA40	P43	P42	P41	P40
	Read/Write	R / W				R / W			
	Resetting Value	Undefind				input			
	Function	Shift Alternate reg.0				Stepping motor control port 0 (M0) / Pattern Generation Port 0			Used both

Shift Alternate reg.0 ⎯⎯ Port 4

Port 5 register

P5 (FFBAH)		7	6	5	4	3	2	1	0
	bit Symbol	SA53	SA52	SA51	SA50	P53	P52	P51	P50
	Read/Write	R / W				R / W			
	Resetting Value	Undefind				input			
	Function	Shift Alternate reg.1				Stepping motor control port 1 (M1) / Pattern Generation Port 1			Used both

Shift Alternate reg.1 ⎯⎯ Port 5

Port 4, 5 control register

P45CR (FFBBH) (RMW disable)		7	6	5	4	3	2	1	0
	bit Symbol	P53C	P52C	P51C	P50C	P43C	P42C	P41C	P40C
	Read/Write	W							
	Resetting Value	0	0	0	0	0	0	0	0
	Function	0: IN 1: OUT (Specifies input/output in 1 bit units)							

→ Select input/output of port 4

0	input
1	output

→ Select input/output of port 5

0	input
1	output

Port 4, 5 Function register

P45FR (FFBCH)		7	6	5	4	3	2	1	0
	bit Symbol	PAT1	CCW1	M1M	M1S	PAT0	CCW0	M0M	M0S
	Read/Write	R / W							
	Resetting Value	0	0	0	0	0	0	—	0
	Function	0: 8-bit write 1: 4-bit write	0: forward 1: reverse	0: 1 or 2 excitation 1: 1-2 excitation	0: port 1: Stepping motor control/pattern generation port1(M1)	0: 8-bit write 1: 4-bit write	0: forward 1: reverse	0: 1 or 2 excitation 1: 1-2 excitation	0: port 1: Stepping motor control/pattern generation port 0 (M0)

See "3.7 Stepping moter control"

↳ Select function of port 4

0	P4
1	Stepping moter control/pattern generation port 0.

→ Select function of port 5

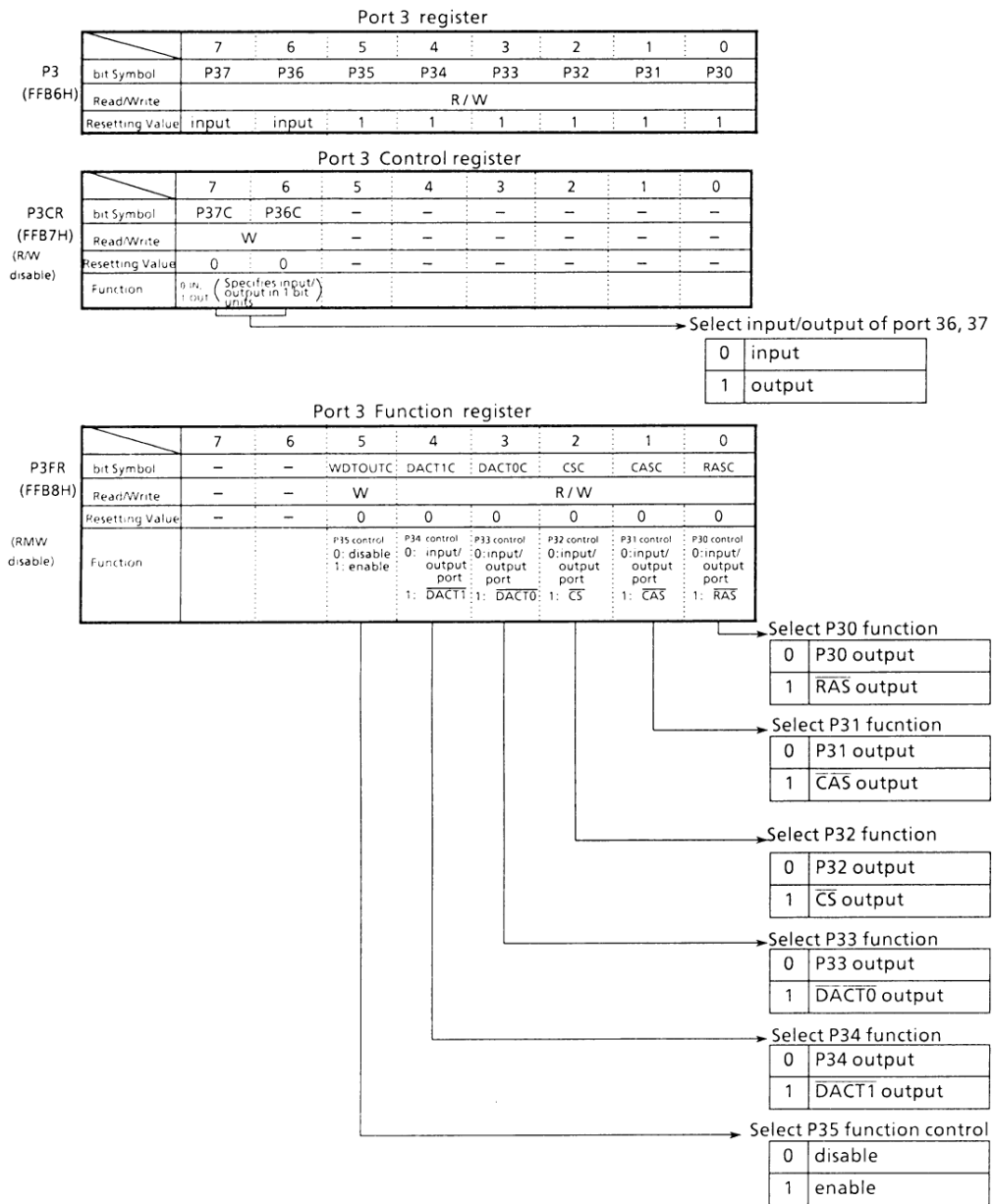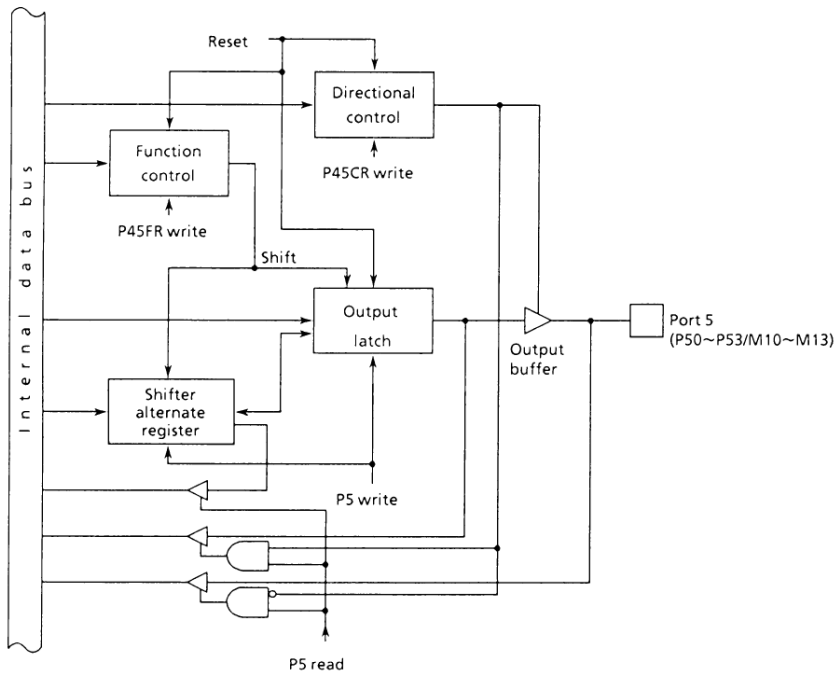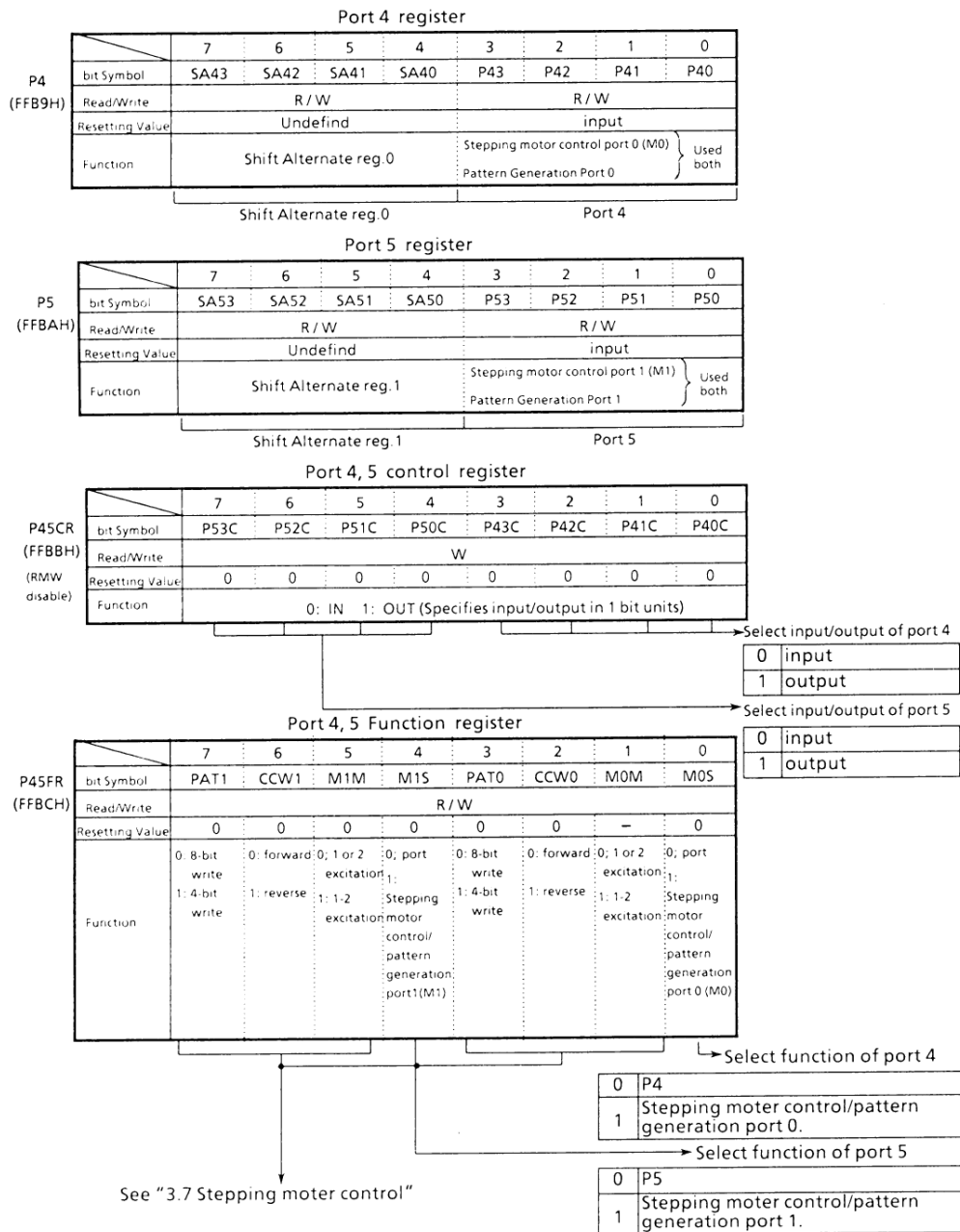| 0	P5
1	Stepping moter control/pattern generation port 1.

Figure 3.6 (10). Registers for Port 4, 5

3.6.5 Port 6 (P60 - P67)

Port 6 is an 8-bit general-purpose I/O port (P6: memory address FFBDH), each bit of which can be set individually for input or output. The control register (P6CR: memory address FFBEH) is used to set the bits for input or output.

    Reset operations clear all bits of the output latch and control register to "0" to set port 6 to the input mode.

    In addition to its general-purpose I/O port function, port 6 also has interrupt request input, time clock input and timer output functions that can be specified with the port 6 function register (P6FR: memory address FFBFH).

(1)   P60 - P63

    P60 - P63 are also used for external interrupt request input (INT3 - I NT6).

Figure 3.6 (11). Port 6 (P60 - P63)

               TOSHIBA CORPORATION
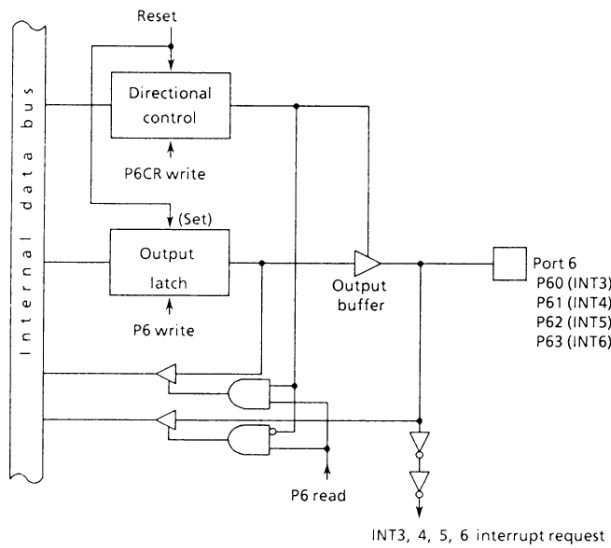
(2)   P64 - P65

P64 and P65 are also used for 8-bit timer clock input.

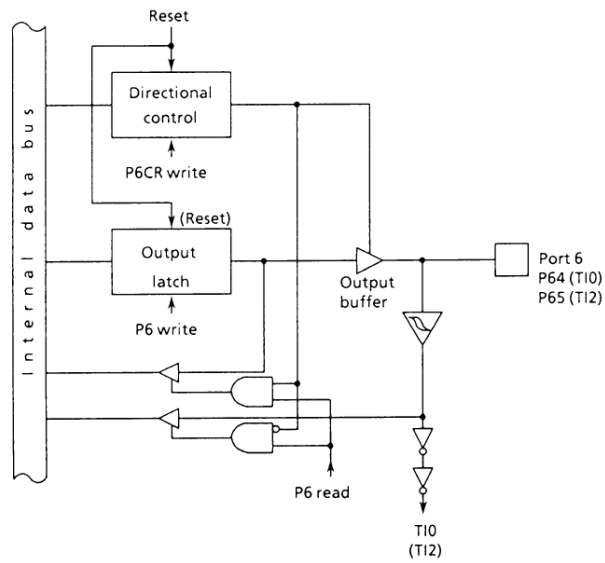**Figure 3.6 (12). Port 6 (P64 - P65)**

(3)   P66 - P67   timer output.
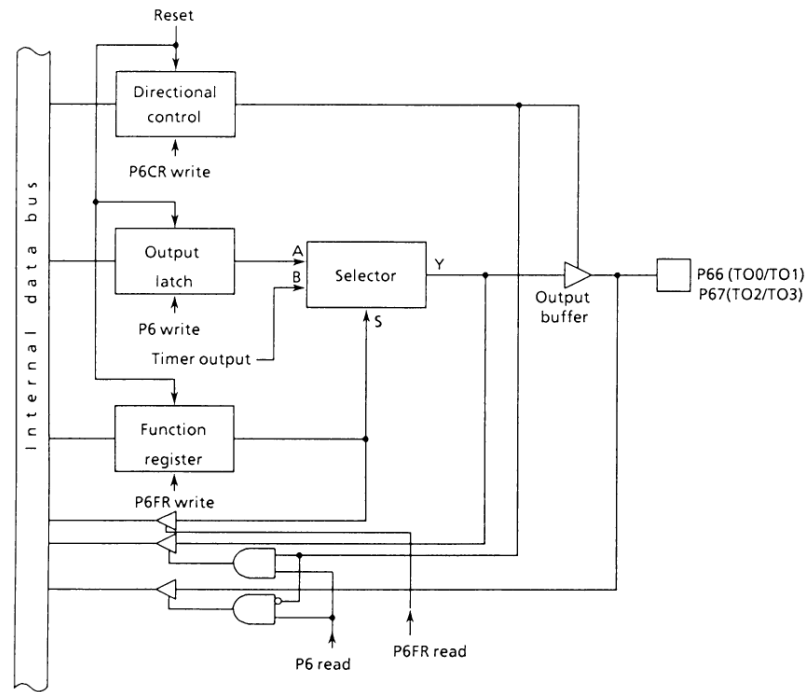
P66 and P67 are also used for 8-bit timer 0/1, 2/3
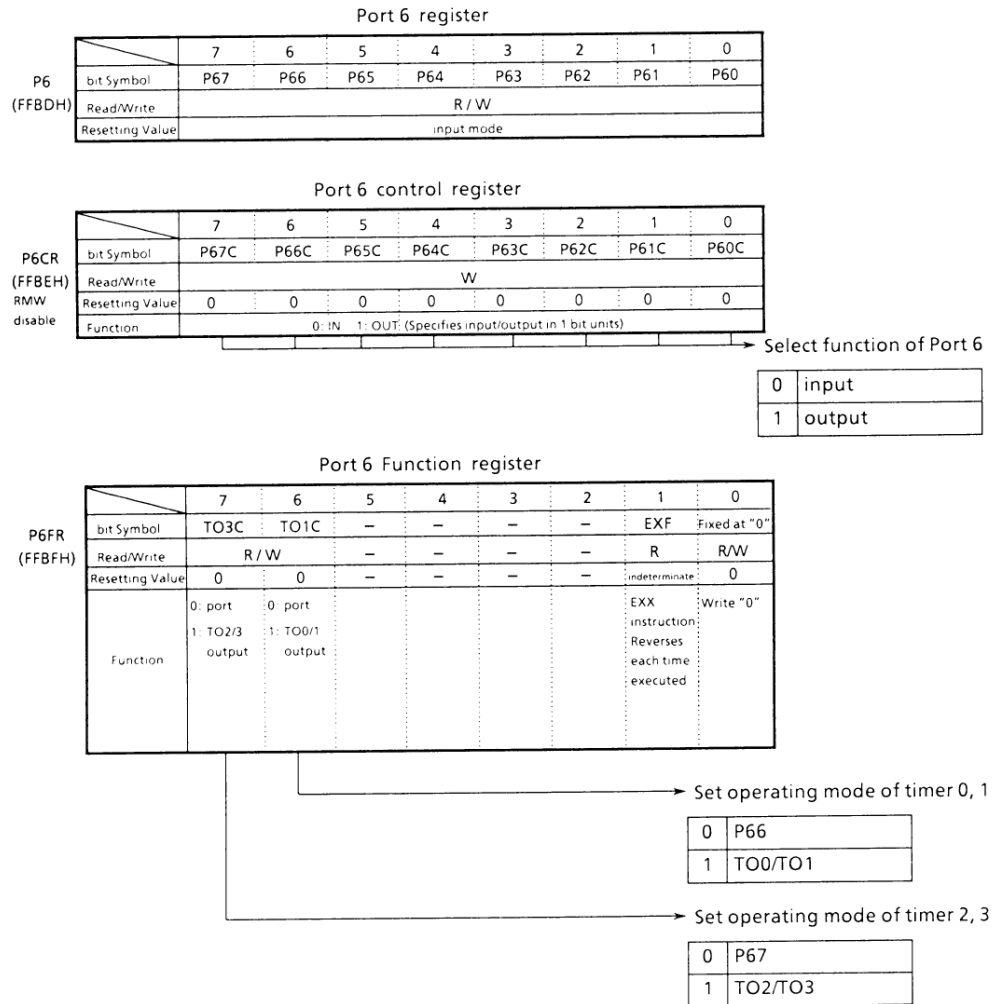
Figure 3.6 (13). Port 6 (P66 - P67)

Port 6 register

P6 (FFBDH)		7	6	5	4	3	2	1	0
	bit Symbol	P67	P66	P65	P64	P63	P62	P61	P60
	Read/Write	R / W							
	Resetting Value	input mode							

Port 6 control register

P6CR (FFBEH) RMW disable		7	6	5	4	3	2	1	0
	bit Symbol	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
	Read/Write	W							
	Resetting Value	0	0	0	0	0	0	0	0
	Function	0: IN 1: OUT (Specifies input/output in 1 bit units)							

→ Select function of Port 6

0	input
1	output

Port 6 Function register

P6FR (FFBFH)		7	6	5	4	3	2	1	0
	bit Symbol	TO3C	TO1C	–	–	–	–	EXF	Fixed at "0"
	Read/Write	R / W		–	–	–	–	R	R/W
	Resetting Value	0	0	–	–	–	–	indeterminate	0
	Function	0: port 1: TO2/3 output	0: port 1: TO0/1 output					EXX instruction Reverses each time executed	Write "0"

→ Set operating mode of timer 0, 1

0	P66
1	TO0/TO1

→ Set operating mode of timer 2, 3

0	P67
1	TO2/TO3

Figure 3.6 (14). Register of Port 6

3.7 Stepping Motor Control/Pattern Generation Port (P4, P5)

The TMP90C051F has a built-in, timer-coupled, 4-bit, 2-channel (M0 and M1) hardware stepping motor control/pattern generation port (hereafter called SMC). SMC (M0 and M1) is also used as 4-bit input/output ports P4 and P5.

Channel 0 (M0) is coupled to 8-bit timer 0 or timer 1, and channel 1 (M1) is coupled to 8-bit timer 2 or timer 3, to alter output.

SMC is controlled by two control registers (P45CR and P45FR) and is used to select the stepping motor control mode and pattern generation mode.

3.7.1 Control Registers

(1)   Ports 4 and 5 Input/Output Specification Register (P45CR)

This register is specifies 4-bit input/output port 5 input and output in 1-bit units. All P45CR bits are cleared to "0" by resets, which sets ports 4 and 5 for input. To use ports 4 and 5 as SMC, set all P45CR bits to "1" for output.
The P45CR register can only be written, and cannot be read.

(2)   Ports 4 and 5 Function Control Register (P45FR)

This register is used to set ports 4 and 5 for use as SMC. To use ports 4 and 5 as SMC, set P45FR <M0S, M1S> and P45FR4 (M1S) to "1".

SMC is set to the 8-bit write mode or 4-bit write mode with P45FR <PAT0> and P45FR <PAT1>. When writing to SMC in the 4-bit write mode, it is only possible to write to the 4-bit shift alternate register and SMC functions as the pattern generation port.

To use SMC as the stepping motor control port, the method of excitation is selected with P45FR <M0M> and P45FR <M1M>; P45FR <CCW0> and P45FR < CCW1> control the direction of rotation.

(3)   Port 4

This is a 4-bit input/output port assigned to address FFB9.
The 4 lower bits correspond to port 4 and the 4 upper bits are a shift alternate register P4 <SA43 ~ 40> used to drive the stepping motor in the pattern generation mode of with 1-2 excitations.

(4)   Port 5

This is a 4-bit input/output port assigned to address FFBA.
The 4 lower bits correspond to port 5, and the 4 upper bits are a shift alternate register P5 <SA53 ~ 50> used to drive the stepping motor in the pattern generation mode of with 1-2 excitations.
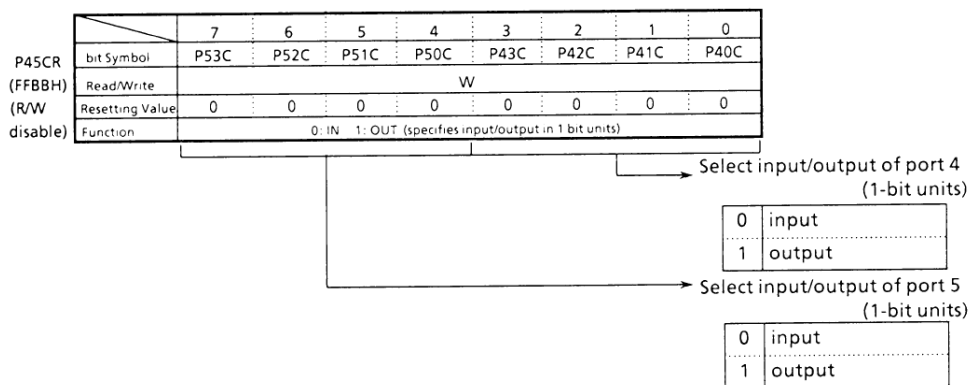
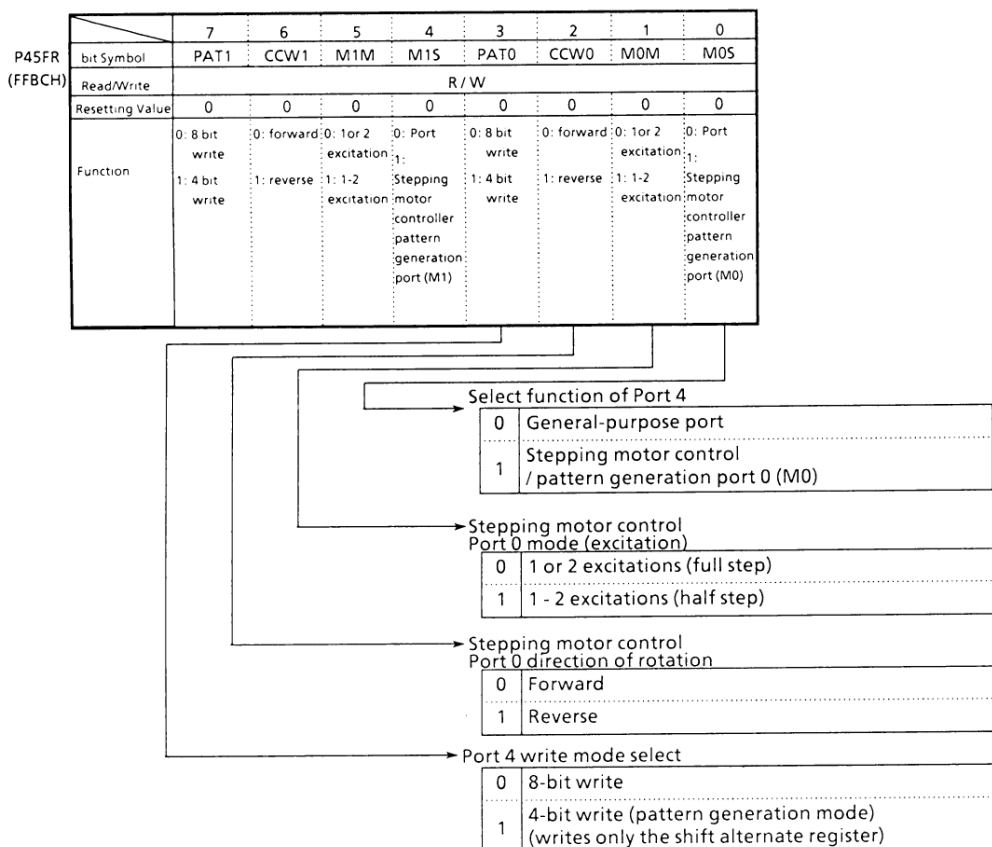**Figure 3.7 (1). Ports 4, 5 Input/Output Specification Register (P45CR)**

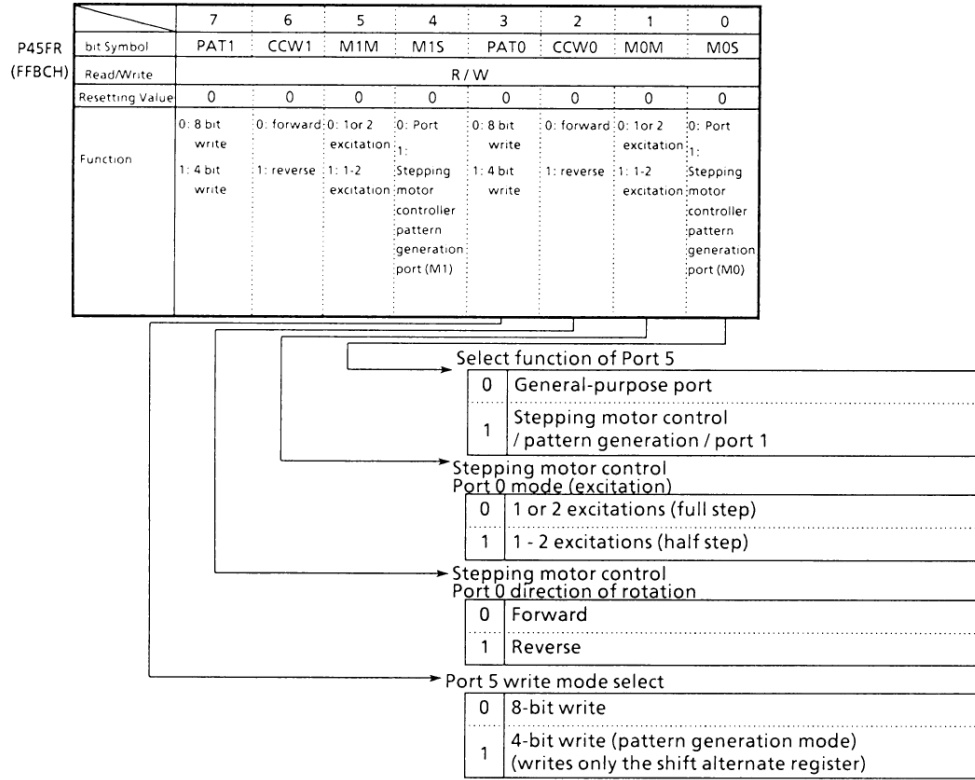**Figure 3.7 (2a). Port 4 and 5 Function Control Register (P45FR)**

P45FR (FFBCH)		7	6	5	4	3	2	1	0
	bit Symbol	PAT1	CCW1	M1M	M1S	PAT0	CCW0	M0M	M0S
	Read/Write	R / W							
	Resetting Value	0	0	0	0	0	0	0	0
	Function	0: 8 bit write 1: 4 bit write	0: forward 1: reverse	0: 1or 2 excitation 1: 1-2 excitation	0: Port 1: Stepping motor controller pattern generation port (M1)	0: 8 bit write 1: 4 bit write	0: forward 1: reverse	0: 1or 2 excitation 1: 1-2 excitation	0: Port 1: Stepping motor controller pattern generation port (M0)

Select function of Port 5

0	General-purpose port
1	Stepping motor control / pattern generation / port 1

Stepping motor control
Port 0 mode (excitation)

0	1 or 2 excitations (full step)
1	1 - 2 excitations (half step)

Stepping motor control
Port 0 direction of rotation

0	Forward
1	Reverse

Port 5 write mode select

0	8-bit write
1	4-bit write (pattern generation mode) (writes only the shift alternate register)

Figure 3.7 (2b). Port 4 and 5 Function Control Register (P45FR)

P4 (FFB9H)		7	6	5	4	3	2	1	0
	bit Symbol	SA43	SA42	SA41	SA40	P43	P42	P41	P40
	Read/Write	R / W				R / W			
	Resetting Value	Undefined				Input Mode			
	Function	Shift Alternate reg.0				Stepping motor control port 0 (M0) Pattern Generation Port 0			use both

→ Port 4
→ Shift Alternate reg.0

P5 (FFBAH)		7	6	5	4	3	2	1	0
	bit Symbol	SA53	SA52	SA51	SA50	P53	P52	P51	P50
	Read/Write	R / W				R / W			
	Resetting Value	Undefined				Input Mode			
	Function	Shift Alternate reg.1				Stepping motor control port 1 (M1) Pattern Generation Port 1			use both

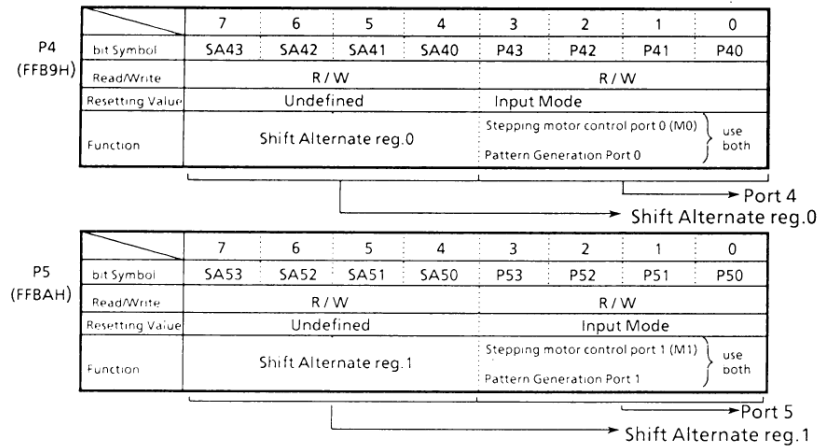→ Port 5
→ Shift Alternate reg.1

Figure 3.7 (3). Port 4 and 5

3.7.2 Pattern Generation Mode

SMC can be set to function as a pattern generation port with bits 3 and 7 (PAT0 and PAT1) of P45FR. In the mode, the CPU writes only to the shift alternate register; therefore, during shift trigger timer interrupt processing, it is possible to write to ports 4 and 5, and to output patterns in real time by coupling with the timer.

In this mode, it is also necessary to always set P45FR1 and P45FR5 (M0M and M1M) to "1".
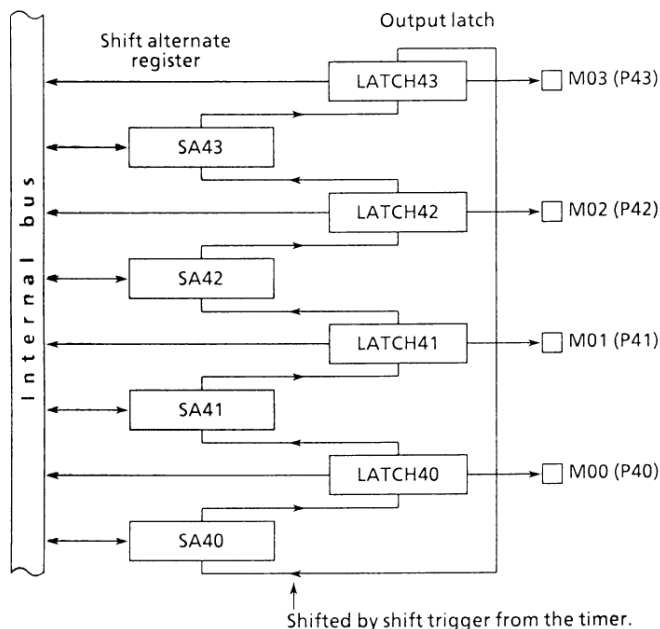
Figure 3.7 (4) shows the mode block diagram.

Figure 3.7 (4). Pattern Generation Mode Block Diagram (Port 4)

In this pattern generation mode, only writing to the output latch is disabled by the hardware. All other operations are the same as for 1-2 excitations in the stepping motor control port mode. Consequently, after a trigger signal from the timer, it is always necessary to write data before the next trigger signal is generated.
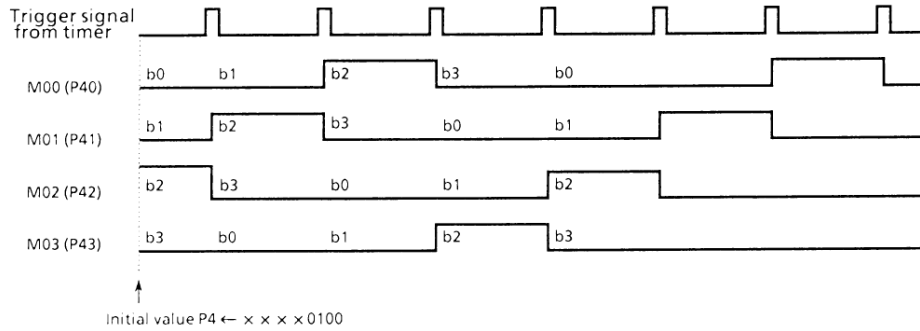
3.7.3 Stepping Motor Control Mode

(1)   4-Phase 1-Excitation/2-Excitation

Figure 3.7 (5) and (6) show the channel 0 output waveforms for 4-phase 1-excitation and 4-phase 2-excitation.
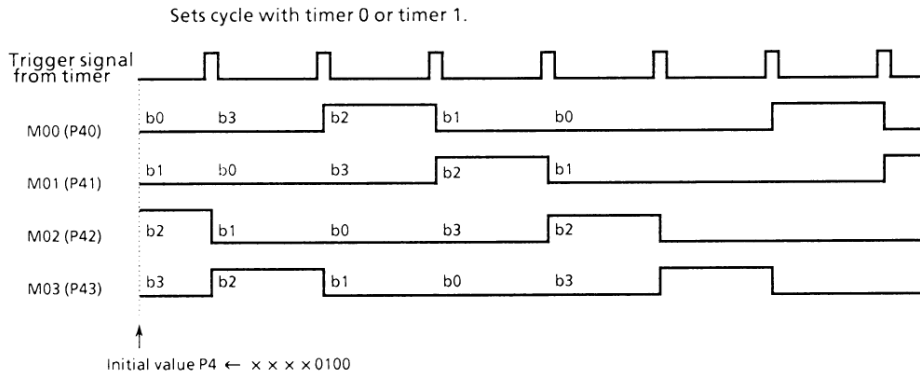
① Forward rotation

② Reverse rotation

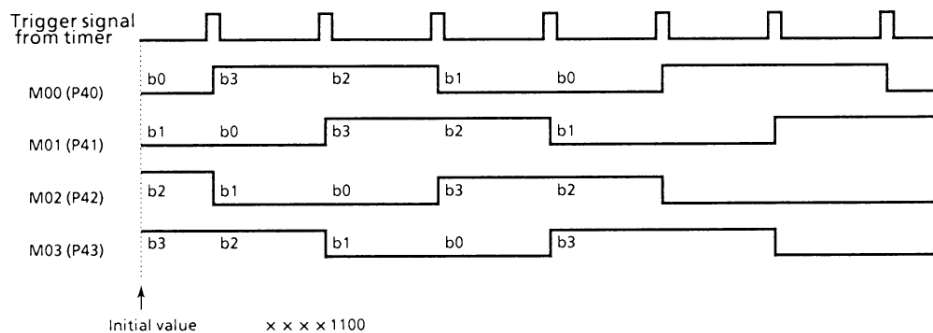**Figure 3.7 (5). 4-phase, 1-Excitation Output Waveform (Forward/Reverse)**

Figure 3.7 (6). 4-Phase, 2-Excitation Output Waveform (Forward)

The following is an explanation of channel 0 operation.

The M0 (also used P4) output latch is shifted and output to the port at the rise of the trigger signal from the timer.

The shift direction is set with P45FR5 (CCW0). The direction of rotation is forward when CCW0 is "0", and is reverse (M00←M01←M02←M03) when CCW0 is "1". Setting only one bit to "1" when making the port 4 initial setting sets 4-phase, 1-excitation; setting 2 consecutive bits to "1" sets 4-phase, 2-excitation.

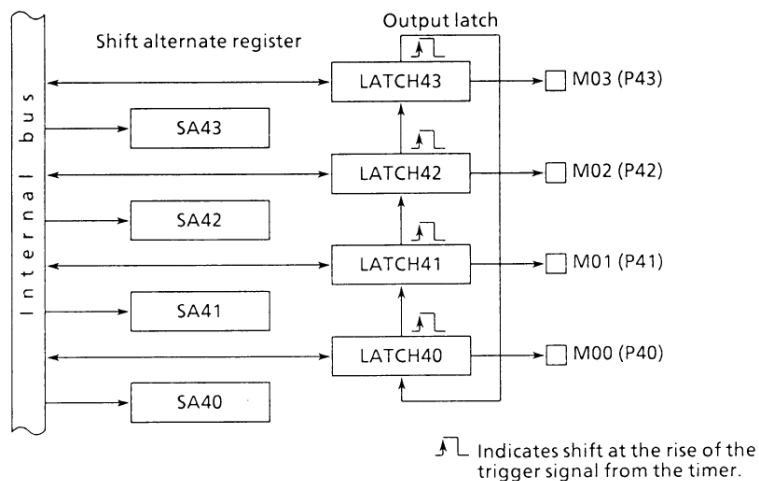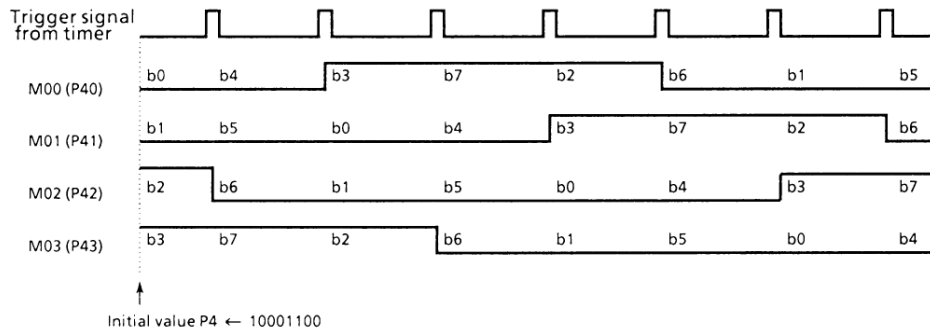Figure 3.7 (7) shows the block diagram.

Figure 3.7 (7). 4-Phase, 2-Excitation (Forward) Block Diagram
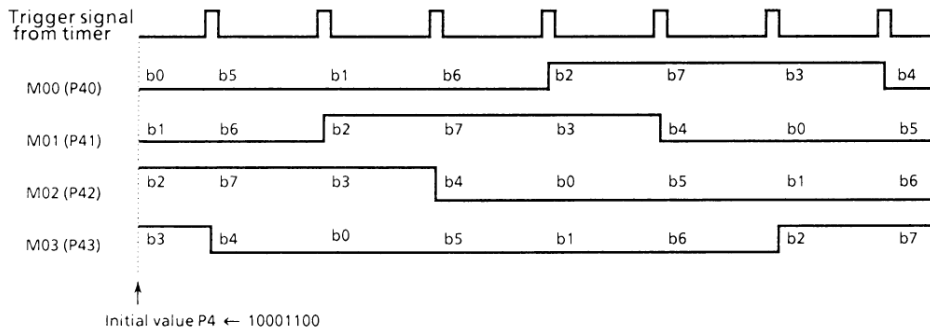
(2)　4-Phase, 2-Excitation

Figure 3.7 (8) shows the channel 0 4-phase, 2-excitation output waveform
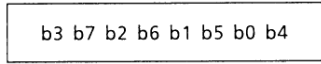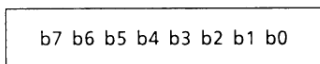

① Forward rotation

② Reverse rotation

Figure 3.7 (8). 4-Phase, 1-2 Excitation Output Waveform (Forward/Reverse)

The 4-phase, 1-2 excitation settings are as shown below.

Intial value

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |

| b3 | b7 | b2 | b6 | b1 | b5 | b0 | b4 |

When the initial value is rearranged as, 3 consecutive bits are set to "1" and the others to "0" (positive logic).

For example, if b3, b7 and b2 are "1", the initial value is 10001100 and the waveform shown in Figure 3.7 (8) is obtained.

To obtain a negative logic output waveform, reverse the "1s" and "0s" of the initial value. For example, to obtain the waveform shown in Figure 3.7 (8) with negative logic, the initial value must be 01110011.

The following is an explanation of channel 0 operation.

The M0 (also used as P4) output latch and stepping motor control shift alternate register (SA4) is shifted and output to the port at the rise of the trigger signal from the timer. The shift direction is set with P45FR <CCW0>.
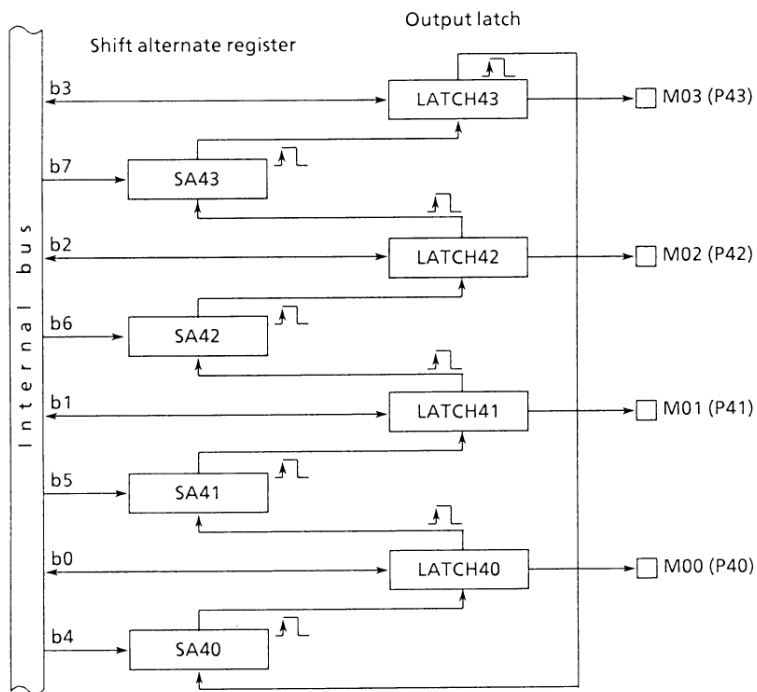
Figure 3.7 (9) shows the block diagram.

Figure 3.7 (9). 4-Phase, 1-2 Excitation (Forward) Block Diagram

Setting example: Setting the registers as shown below when driving channel 0 (M0) with timer 0 at 4-phase, 1-2 excitations.

```
            7 6 5 4 3 2 1 0
TRUN   ←  - - - - - - - 0      Stops timer 0 and clears it to zero.
T01MOD ←  0 0 X X - - 0 1      Sets the 8-bit timer mode and sets the timer 0 input
                               clock to ⌀T1.
TFFCR  ←  - - - - 1 0 1 0      Clears TFF1 to zero and enables the inversion trigger
                               with timer 0.
TREG0  ←  * * * * * * * *      Loads the cycle to the timer counter.
P45CR  ←  - - - - 1 1 1 1      Sets all P4 bits to output mode.
P45FR  ←  - - - - 0 0 1 1      Sets 4-phase, 1 - 2 excitations, forward rotation.
P4     ←  1 0 0 0 1 1 0 0      Sets the initial value.
TRUN   ←  - - 1 - - - - 1      Starts timer 0.

Note: X ; don't care     - ; no change
```

3.7.4 Timer Trigger Signals

The trigger signals used with SMC are not the same as the timer flip-flop (TFF1, 3) inversion trigger signals, but differ as shown in Table 3.7 (1) depending on the timer operation mode.

Table 3.7 (1) Using 8-bit timers 0 and 1 (same for timers 2 and 3)

	TFF1 Inversion	SMC shift
8-bit timer mode	Selected with TFFCR0 (FF1IS) when the up counter matches TREG0 or TREG1.	←
16-bit timer mode	Selected when the up counter matches both TREG0 and TREG1 (up counter value = TREG*28 + TREG0).	←
PPG output mode	Selected when the up counter matches TREG0 and TREG1, respectively.	Selected when the up counter matches TREG1 (PPG cycle).
PWM output mode	Selected when the up counter matches TREG0.	The SMC shift trigger signal is not generated.

Note: It is necessary to TFFCR1 (FF1IE) to "1" and enable TFF1 inversion when SMC is shifted.

3.7.5 SMC and Timer Output Application

As was explained in "3.7.4 Timer Trigger Signal", SMC shift and TFF inversion timing differ depending on the timer mode. The following is a description of a typical application in which SMC is operated while the 8-bit timer is operated in the PPG output mode.

When driving a stepping motor, a sync signal at the excitation switchover timing is often necessary together with each phase value (SMC output). In view of this, in this application the port 4 is used as the stepping motor control port and the sync clock is output to TO1 (also used as P61).
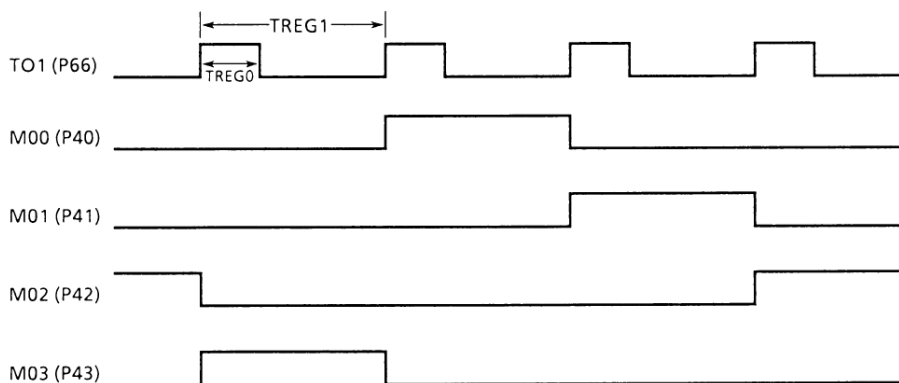
4-phase, 1-Excitation Output Waveform

Setting example:

```
              7 6 5 4 3 2 1 0
TRUN    ← - - - - - - 0 0      Stops timer 0 and 1, and clears to zero.
T01MOD  ← 1 0 X X X X 0 1      Sets timer 0 and 1 to the PPG output mode and sets the
                               input clock to T1.
TFFCR   ← - - - - 0 1 1 X      Enables TFF1 inversion and sets to "1."
TREG0   ← * * * * * * * *      Sets TREG0 to TO1 duty.
TREG1   ← * * * * * * * *      Sets TREG1 to TO1 cycle.
P6FR    ← - 1 - - - - - -  ⎫
                           ⎬   Sets P66 as TO1 pin.
P6CR    ← - 1 - - - - - -  ⎭
P45FR   ← - - - - 0 0 0 1      Sets P4 as SMC and 4-phase, 1-excitation.
P45CR   ← - - - - 1 1 1 1      Sets all P4 bits to output mode.
P4      ← * * * * * * * *      Sets the initial value.
TRUN    ← - - 1 - - - 1 1      Starts timers 0 and 1.

Note: X ; don't care    - ; no change
```

3.8 Timers

The TMP90C051F has four 8-bit timers (timers 0, 1, 2 and 3).

These four 8-bit timers can be operated independently or cascade connected to form two 16-bit timers. These 8-bit timers have the following four operation modes.

- 8-bit internal timer modes (4)　} Can be combined
- 16-bit interval timer modes (2)　} (8 bits x 2, 16 bits x 1)
- 8-bit programmable square wave (PPG: variable duty at fixed interval) (2).
- 8-bit PWM (pulse width modulation: variable duty at fixed interval) output modes (2)

Figure 3.8 (1) shows the 8-bit timer (timers 0 and 1) block diagram.

Timers 2 and 3 have the same circuit configuration as timers 0 and 1.

The configuration of each interval timer consists of an 8-bit up counter.

8-bit comparator and 8-bit timer register. Timer flipflops (TFF1 and TFF3) are available for the timer 0 and 1 pair, and the timer 2 and 3 pair. The internal clocks øT1, øT4, øT16 and øT256 used as the input clocks to the interval timers are obtained from the 9-bit prescaler shown in Figure 3.6 (2).

The operating modes and timer flipflops for the 8-bit timers are controlled by 5 control registers (TO1MOD, T23MOD, TFFCR, TRUN and TRDC).
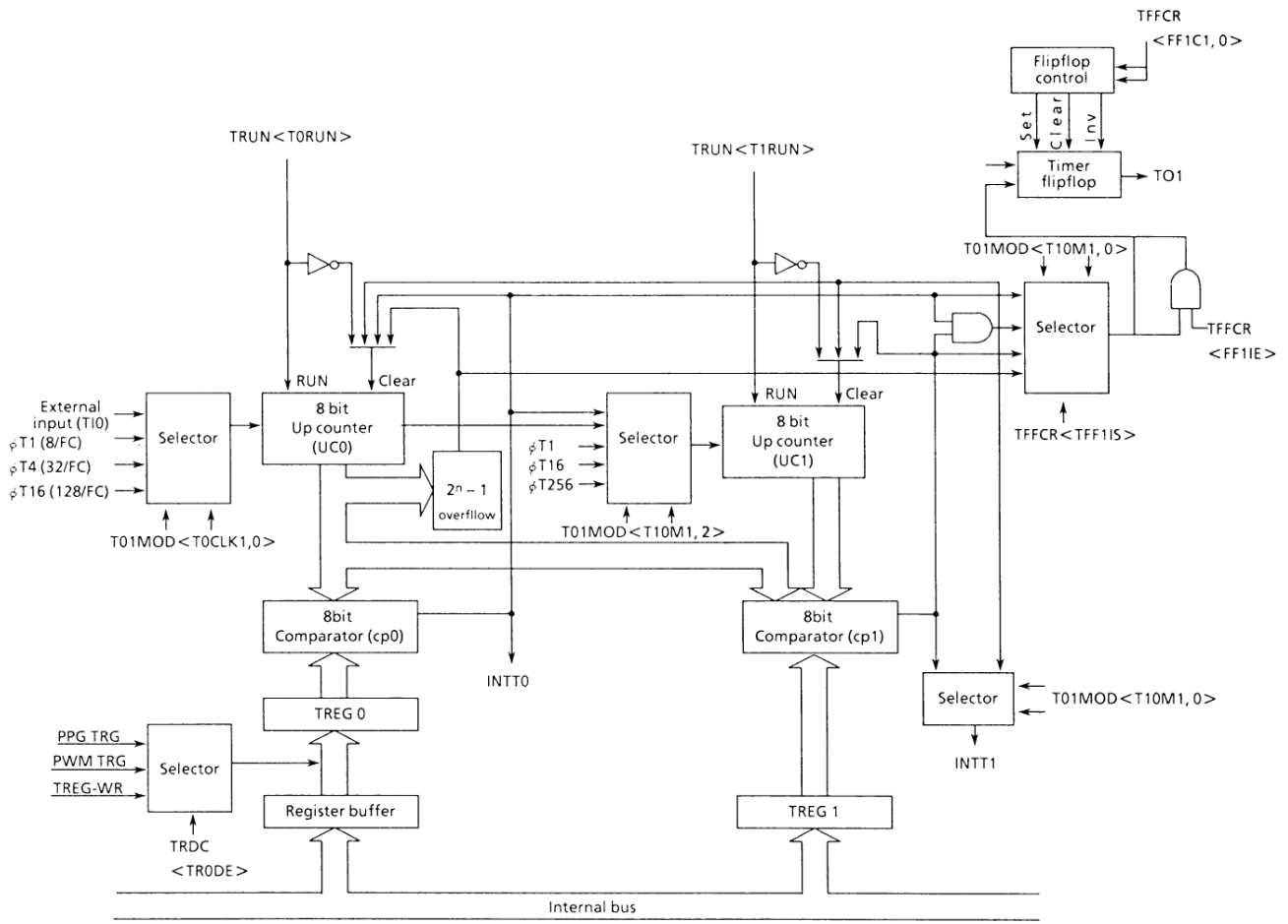
Figure 3.8 (1). Timer Block Diagram (Timers 0 and 1)

① Prescaler

This 9-bit prescaler generates the clocks used for input to the 8-bit timers and baud rate generator, TPH serial I/F by further dividing the fundamental clock after it has been divided by 4 (fc/4)

The four clock øT1, øT4, øT16 and øT256 are used for the 8-bit timers.

This prescaler can be enabled/disabled with the 5th bit TRUN <PRRUN> of the timer operation control register (TRUN). Setting <PRRUN> to "1" starts counting and setting it to "0" clears and stops the timer. Resets clear <PRRUN> to "0", so the prescaler is cleared and stopped.

	Cycle	
Input clock fc	12.5MHZ	16MHZ
ϕT1 (8/FC)	0.64μs	0.5μs
ϕT4 (32/FC)	2.56μs	2.0μs
ϕT16 (128/FC)	10.24μs	8.0μs
ϕT256 (2048/FC)	163.84μs	128 μs

Figure 3.8 (2). Prescaler

② Up-counter

This is an 8-bit binary counter that counts up in accordance with the input clock specified with the timer 0 and 1 mode register (T01MOD) and timer 2 and 3 mode register (T23MOD).

The timer 0/timer 2 input clock can be selected from external clocks (TI0 and TI2) and the three internal clocks øT1, øT4 and øT16 by setting the T01MOD/ T23MOD registers.

The timer 1/timer 3 input clock differs depending on the operating mode. In the 16-bit timer mode, the timer 0/timer 2 output overflow is used as the input clock.
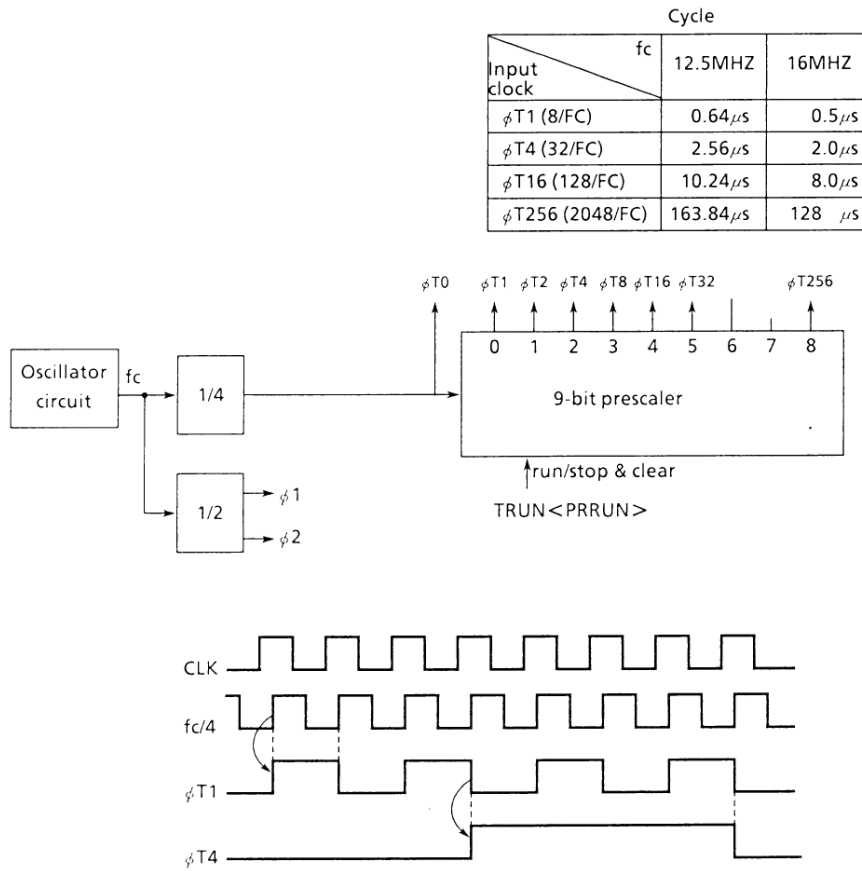
In other than the 16-bit timer mode, the input clock is selected from the internal clocks øT1, øT16 and øT256 by setting the T01MOD/T23MOD registers, and the timer 0/timer 1 comparator output (match detection).

Examples: If T01MOD <T10M1, 0> = 01, the timer 0 overflow output is used as the timer 1 input clock (16-bit timer).

If T01MOD <T10M1, 0> = 00, and T01MOD <T1CLK1, 0> = 01, øT1 (8/fc) is used as the timer 1 input clock (8-bit timer).

The operating mode is also set using the T01MOD/ T23MOD registers. <T10M1, 0> and T23M1, 0> are initialized to "00" by resets. This sets the 8-bit timer mode.

Concerning the up counters, each interval timer can be enabled, disabled or cleared with timer operation control register TRUN. All counters are cleared and timers are stopped by resets.

③ Timer register

These are 8-bit registers used to set interval times. When the values to which the timer registers (TREG0/ TREG1 and TREG2/TREG3) are set and the up counter value match, the comparator match detect signal becomes active. When the setting value is 00H, the match detect signal becomes active in case of an up counter overflow.

Timer registers TREG0/TREG2 have a double-buffer configuration and these registers and buffers are used as pairs.

The TREG0/TREG2 double buffer is enabled and disabled with <TR0DE> and <TR2DE> of the timer register double buffer control register (TRDC). The double buffer is disabled when <TR0DE>/<TR2DE> = "0" and enabled when <TR0DE>/<TR2DE> = "1".

The timing for data transfers from a register buffer to a timer register when double buffer has been enabled is determined by compare matching of PWM mode 2^n - 1 overflow or PPG mode cycle.

<TR0DE>/<TR2DE> are initialized to "0" by resets, which disable double buffers. When using a double buffer, write data to the timer register, set <TR0DE>/ <TR2DE> to "1" and then write the following data to the register buffer.
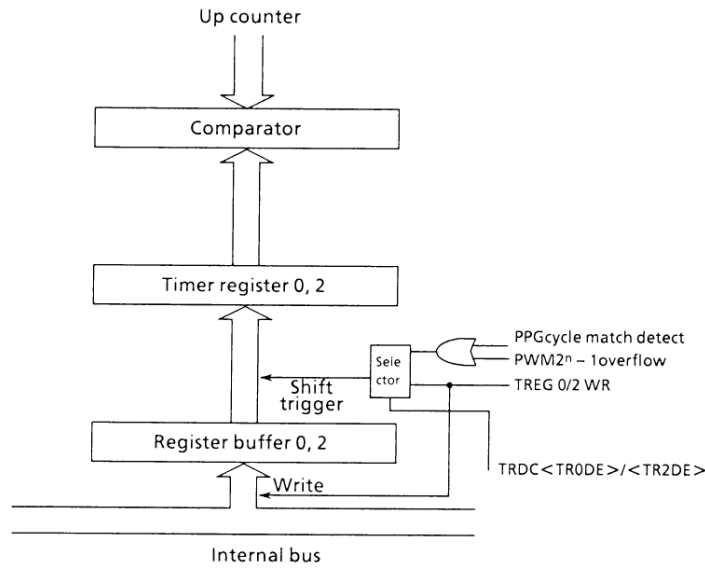
Figure 3.8 (3). Timer Register 0, 2 Configuration

Note: The timer registers and register buffers are assigned to the same memory addresses.

When <TR0DE>/<TR2DE> = 0, the same values are written to the register buffers and timer registers.
When <TR0DE>/<TR2DE> = 1, values are written only to the register buffers.

The timer register memory addresses are as follows.

TREG0: FFC0H
TREG1: FFC1H
TREG2: FFC2H
TREG3: FFC3H

These are write-only registers.

T01MOD (FFC4H)		7	6	5	4	3	2	1	0
	bit Symbol	T10M1	T10M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
	Read/Write				R / W				
	Resetting Value	0	0	0	0	0	0	0	0
	Function	00: 8Bit Timer 01: 16Bit Timer 10: 8Bit PPG 11: 8Bit PWM		00: – 01: 2^6-1 PWM 10: 2^7-1 cycle 11: 2^8-1		00: T0OTRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI0 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 0 input clock

00	External clock TI0
01	Internal clock ϕT1
10	Internal clock ϕT4
11	Internal clock ϕT16

Timer 1 input clock

	T01MOD7, 6 ≠ 01	T01MOD7, 6 = 01
00	Timer 0 comparator output	Timer 0 overflow output
01	Internal clock ϕT1	(16-bit timer
10	Internal clock ϕT16	mode)
11	Internal clock	

PWM0 cycle select
(Don't care for other than the PWM mode.)

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

Timers 0 and 1 operating mode setting

00	8-bit timer x 2 (timer 0/1)
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM output (timer 0) + 8-bit timer (timer 1)

Figure 3.8 (4). Timer 0/1 Mode Register (T01MOD)

T23MOD	bit Symbol	7	6	5	4	3	2	1	0
		T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
(FFC5H)	Read/Write				R / W				
	Resetting Value	0	0	0	0	0	0	0	0
	Function	00: 8Bit Timer 01: 16Bit Timer 10: 8Bit PPG 11: 8Bit PWM		00: − 01: 2^6-1 PWM 10: 2^7-1 cycle 11: 2^8-1		00: TO2TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI2 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 2 input clock

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

Timer 3 input clock

	T23MOD7, 6 ≠ 01	T01MOD7, 6 = 01
00	Timer 2 comparator output	Timer 2 overflow output
01	Internal clock ϕT1	(16-bit timer mode)
10	Internal clock ϕT16	
11	Internal clock	

PWM2 cycle select
(don't care in other than the PWM mode.)

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

Timer 2/3 operating mode setting

00	8-bit timer x 2 (timer 2/3)
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM output (timer 2) + 8-bit timer (timer 3)

Figure 3.8 (5). Timer 2/3 Mode Register (T23MOD)

		7	6	5	4	3	2	1	0
TFFCR	bit Symbol	TFF3C1	TFF3C0	TFF3IE	TFF3IS	TFF1C1	TFF1C0	TFF1IE	TFF1IS
(FFC6H)	Read/Write	W		R / W		W		R / W	
	Resetting Value	—		0	0	—		0	0
(RMW disable)	Function	00: Invent TFF3 01: Set TFF3 10: Clear TFF3 11: Dont Care.		1: TFF3 Invert Enable	0: Invers-ion with timer 2	00: Invent TFF1 01: Set TFF1 10: Clear TFF1 11: Dont Care.		1: TFF1 Invert Enable	0: Invers-ion with timer 0

Timer flipflop TFF1 inversion signal reset
(Don't care for other than the 8-bit timer mode.)

0	Inversion with timer 0
1	Inversion with timer 1

Timer flipflop TFF1 inversion

0	Disable
1	Enable

Timer flipflop TFF1 control

00	Inverts the TFF1 value (software inversion)
01	Sets TFF1 to "1".
10	Clears TFF1 to "0".
11	Don't care (Always read as "11")

Timer flipflop TFF3 inversion signal reset
(Don't care in other than the 8-bit timer mode.)

0	Inversion with timer 0
1	Inversion with timer 3

Timer flipflop TFF3 inversion

0	Disable
1	Enable

Timer flipflop TFF3 control

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

Figure 3.8 (6). 8-bit Timer Flip-flop Control Register (TFFCR)

TRUN (FFC8H)		7	6	5	4	3	2	1	0
	bit Symbol	–	–	PRRUN	–	T3RUN	T2RUN	T1RUN	T0RUN
	Read/Write	–	–	R/W	–			R/W	
	Resetting Value	–	–	0	–	0	0	0	0
	Function					Prescaler & Timer RUN / STOP Control 0: Stop & Clear 1: RUN			

Timer 0 operation

0	Stop and clear
1	Count

Timer 1 operation

0	Stop and clear
1	Count

Timer 2 operation

0	Stop and clear
1	Count

Timer 3 operation

0	Stop and clear
1	Count

Prescaler operation

0	Stop and clear
1	Count

Figure 3.8 (7). Timer Operation Control Register (TRUN)

TRDC (FFC7H)		7	6	5	4	3	2	1	0
	bit Symbol	Fixed at "0"	–	–	–	–	–	TR2DE	TR0DE
	Read/Write	R /W	–	–	–	–	–		R / W
	Resetting Value	0	–	–	–	–	–	0	0
	Function	Write "0"					Timer Reg. Double Buffer Control 0: Double Buffer Disable 1: Double Buffer Enable		

Timer register 0 double buffer control

0	Disable
1	Enable

Timer register 2 double buffer control

0	Disable
1	Enable

Figure 3.8 (8). Timer Register Double Buffer Control Register (TRDC)

　　　　　　　　　　　　　　　　　　　TOSHIBA CORPORATION

④ Comparator

When a comparison of the up counter value and timer register values shows a match, the up counter is cleared to "0" and an interrupt (INTT0-INTT3) is generated. If timer flip-flop inversion is enabled, the timer flip-flop value is inverted at the same time.

⑤ Timer Flip-flop

This flip-flop is inverted by the interval timer match detect signals (comparator output) and the value can be output to the timer output pins TO1 (also used a P66) and TO3 (also used as P67).
One each of these timer flip-flops is provided for the timer 0/1 pair and the timer 2/3 pair. These timer flip-flops are called TFF1 and TFF3. TFF1 outputs to TO1 and TFF3 outputs to TO3.

The following explains the operation of the 8-bit timers.

(1)   8-Bit Timer Mode

The four 8-bit interval timers 0, 1, 2 and 3 can be used independently. The operation is the same for all and timer 1 will be used for explanatory purposes.

① When interrupts are generated in a fixed cycle

To use timer 1 for generating interrupts (INTT1) in a fixed cycle, first stop timer 1 and then load the operation mode, input clock and cycle to the T01MOD TREG1 registers. Next, enable interrupt INTT1 and then start timer 1 counting.

Example: Use the following procedure to load the registers to generate timer 1 interrupts every 40 microseconds at fc = 16MHz.

```
                MSB              LSB
                7 6 5 4 3 2 1 0
TRUN    ←   X X – X – – 0 –      Stops timer 1 and clears it to zero.
T01MOD  ←   0 0 X X 0 1 – –      Sets the 8-bit timer mode and sets the input clock to øT1
                                 (0.5 µs @ fc = 16MHz).
TREG1   ←   0 1 0 1 0 0 0 0      Loads 40µs / øT1 = 50H to the timer register.
INTE1   ←   X X – – – – 1 –      Enables INTT1.
TRUN    ←   X X 1 X – – 1 –      Starts timer 1 counting.

Note:     X : don't care     - : no change
```

Refer to the table below concerning input clock selection

Table 3.8 (1) Interrupt Cycle and Input Clock Selection Using an 8-Bit Timer

Interrupt cycle (@fc = 16MHz)	Resolution	Input clock
0.5µs ~ 128µs	0.5 µs	øT1 (8/fc)
2 µs ~ 512µs	2 µs	øT4 (32/fc)
8 µs ~ 2.048ms	8 µs	øT16 (8128/fc)
128 µs ~ 32.768ms	128 µs	øT256 (2048/fc)

② Outputting a 50% duty square waveform

Invert the timer flipflop at a fixed cycle and output the timer flipflop value to the timer output pin (TO1).

Example: Use the following procedure to load the registers to output a square waveform from the TO1 pin in a 3.0 microsecond cycle at fc = 16MHz. Timers 0 and 1 are used in this case but timer 1 will be used here for explanatory purposes.

```
                MSB           LSB
                7 6 5 4 3 2 1 0
TRUN      ←     - - - - - - 0 -      Stops timer 1 and clears it to zero.
T01MOD    ←     0 0 X X 0 1 - -      Sets the 8-bit timer mode and sets the input clock to
                                     φT1 (0.5μs@fc = 16MHz)
TREG1     ←     0 0 0 0 0 0 1 1      Loads 3.0μs ÷ φT1 ÷ 2 = 3 to the timer register.
                                     Clears TFF1 to "0" and sets for inversion with the
TFFCR     ←     - - - - 1 0 1 1      match detect signal from timer 1.

P6CR      ←     - 1 - - - - - -   ⎫  Sets P66 as TO1 pin.
P6FR      ←     - 1 X X X X - -   ⎬
TRUN      ←     X X 1 X - - 1 -   ⎭  Starts timer 1 counting.
```

Note :   X ; don't care   - ; no change
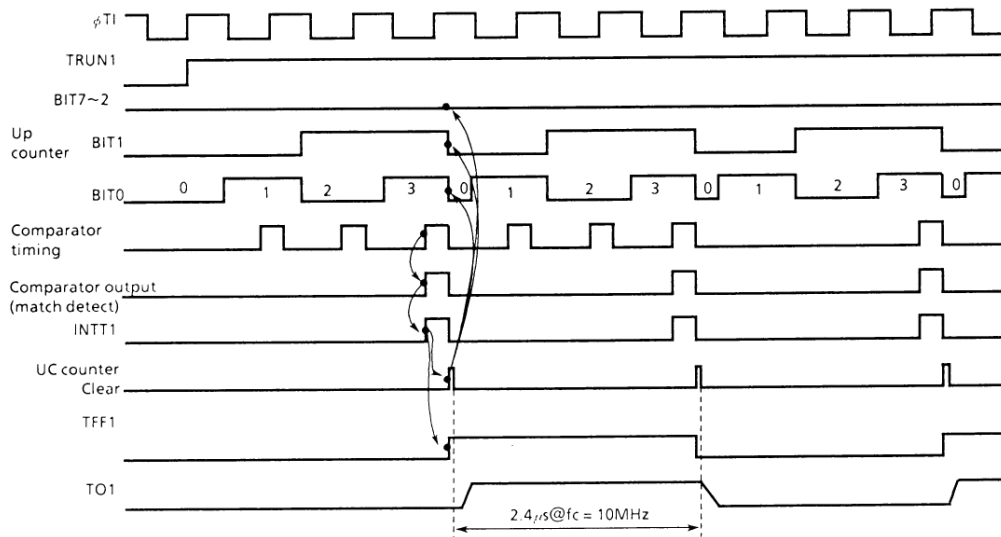
Figure 3.8 (9). Square Waveform (50% Duty) Output Timing Chart

③ Starting timer 1 counting with timer 0 match output

Set the 8-bit timer mode and set the timer 0 comparator

output as the timer 1 input clock.
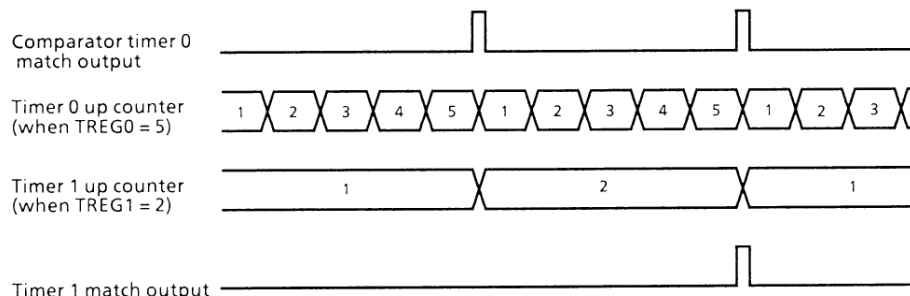
Figure 3.8 (10)

④ Output inversion with software

The timer flipflop value can be inverted regardless of timer operation.
Writing 00 to FF1C (bit 3 and 2 of TFFCR at memory address FFC6) inverts the TFF1 value. Writing 00 to FF3C (bits 7 and 6 of TFFCR at memory address FF3C) inverts the TFF3 value.

⑤ Timer Flipflop initial setting

The timer flipflop initial value can be set to either "0" or "1" regardless of timer operation.
For example, write 10 to FF1C (bits 3 and 2 of TFFCR) at memory address FF1C to set TFF1 to "0" and write 01 to memory address FF1C to set TFF1 to "1".

Note: Timer Flipflop and timer register values cannot be read.

(2)　16-Bit Timer Mode

16-bit interval timers can be created by using timers 0 and 1 as a pair and timers 2 and 3 as a pair.
Timers 0 and 1, and timers 2 and 3 have the same operation, so timers 0 and 1 are used here for explanatory purposes.
Cascade connect timers 0 and 1 and set bits 7 and 6 of the timer 0 and 1 mode register T01MOD to "00" to enable use as a 16-bit interval timer.
When the 16-bit timer mode is set, the timer 0 overflow is used as the timer 1 input clock regardless of the T1CLK (bits 3 and 2 of T01MOD) setting value. Table 3.8 (2) shows the relationship between timer (interrupt) cycles and input clock selection.

Table 3.8 (2) 16-Bit Timer (Interrupt) Cycle and Input Clock Selection

Interrupt cycle (@fc = 16MHz)	Resolution	Input clock
0.5 μs ~ 32.768ms	0.5μs	øT1
2 μs ~ 131.072ms	2 μs	øT4
8 μs ~ 524.288ms	2 μs	øT16

The timer (interrupt) cycle is set by loading the 8 lower bits to timer register TREG0 and the 8 upper bits to timer register TREG1. In this case, always set TREG0 first. (Comparison is temporarily halted by writing data to TREG0 and comparison is started by writing data to TREG1.)

Setting example: To generate the interrupt INTT1 every 0.5s at fc = 16MHz, load the following values to TREG0 and TREG1.
When counting using øT16 (= 8µs @16MHz),
    0.5s ÷ 8µs = 62500 = F424H
therefore, set TREG1 = F4H, TREG0 = 24H.

The timer 0 comparator match detect signal is output each time the time counter UC0 and TREG0 values match; however, up counter UC0 is not cleared.

When the timer 1 comparator match detect signal is output at each comparator timing cycle, if up counter UC1 and TREG1 match, up counters UC0 and UC1 are cleared to "0" when the match detect signals of both timer 0 and 1 comparator are output at the same time, and interrupt INTT1 is generated. If inversion is enabled, the value of the timer flipflop TFF1 is inverted.

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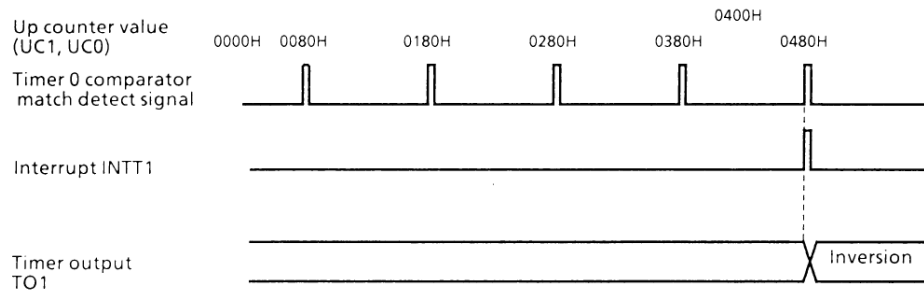

Figure 3.8 (11)

(3)   8-bit PPG (Programmable Square Waveform) Output Mode

Timer 0 of timer 2 can be used to output square wave-forms of optional frequency and optional duty. The output pulses can be either low-active or high-active. Timers 0 and 3 cannot be used when this mode is set.

When timer 0 is used, the output goes to TO1 (also used as P66) and when timer 2 is used, the output goes to TO3 (also used as P67).
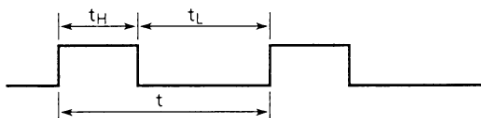
Timer 0 will be used for explanatory purposes. (Operation is the same with timer 2.)
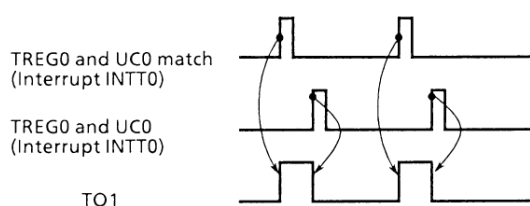
This mode outputs a programmable square waveform by inverting the timer output each time the 8-bit up counter 0 (UC0) matches timer registers (TREG0 and TREG1.

It is necessary, however, to satisfy the condition (TREG0 setting value) < (TREG1 setting value).

The timer 1 up counter (UC1) cannot be used in this mode but timer 1 can be used for counting by setting TRUN <T1RUN> = 1.

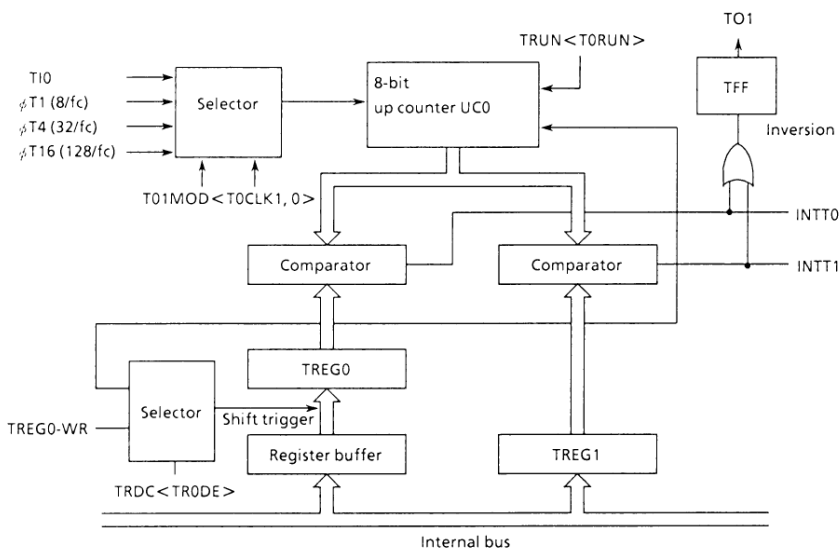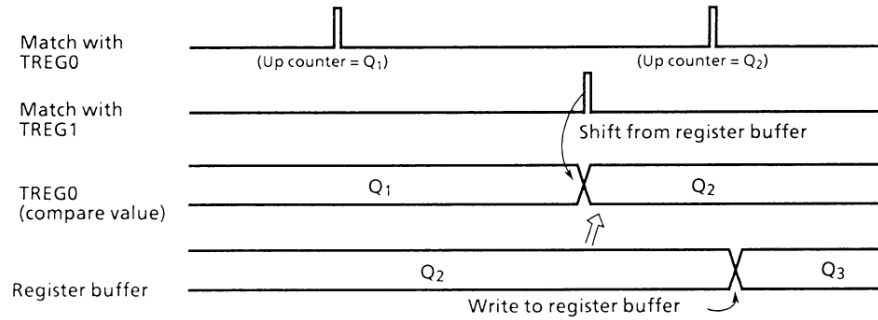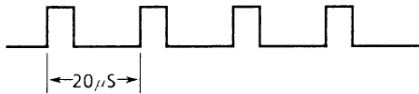Figure 3.8 (12) shows the block diagram for this mode.

Figure 3.8 (12). 8-Bit PPG Output Mode Block Diagram

In this mode, the register buffer value is shifted in to TREG0 by TREG1 and UC0 matching when TREG0 is enabled as a double buffer.

A double buffer facilitates small duties (when changing the duty).

Example: Outputting a 1/4 duty 50kHz pulse (@fc = 16MHz)

• Determining timer register setting values

The frequency is set to 50kHz by making a $t = 1/50\text{kHz} = 20\mu\text{s}$ waveform.
When $\phi T1 = 0.5\mu\text{s}$ (@16MHz) is used $20\mu\text{s} \div 0.5\mu\text{s} = 40$ therefore, timer register 0 (TREG0) is set to 10 = 0AH.

	MSB							LSB	
	7	6	5	4	3	2	1	0	
TRUN	← x	x	-	x	-	-	0	0	Stops timer 0 and timer 1, and clears them to zero.
TO1MOD	← 1	0	x	x	x	x	0	1	Sets the 8-bit PPG mode and sets the input clock to ϕT1
TFFCR	← -	-	-	-	0	1	1	x	Sets TTF1 and enables inversion.
									→ A negative logic output waveform is obtained when set to "10".
TREG0	← 0	0	0	0	1	0	1	0	Writes 0A_H
TREG1	← 0	0	1	0	1	0	0	0	Writes 28_H
P6CR	← -	1	-	-	-	-	-	-	Sets P66 as the TO1 pin.
P6FR	← -	1	x	x	x	x	-	-	
TRUN	← x	x	1	x	-	-	1	1	Starts timer 0 counting.

Note: x ; don't care   - ; no change

(4)   8-bit PWM output mode

Only timers 0 and 2 can be used in this mode. Up to two 8-bit resolution PWM outputs are possible (PWM0 and PWM2).

When timer 0 is used, the PWM waveform is output to TO1 (also used as P66) and when timer 2 is used, the PWM waveform is output to TO3 (also used as P67).

Timer 0 (PWM0) is used for explanatory purposes. (Operation is the same for timer 2).

Timer output is inverted when the up counter (UC0) matches the timer register TREG0 value and 2^n - 1 (set to n = 6, 7 or 8 with the T01MOD register) counter overflows. Up counter UC0 is cleared by 2^n - 1 counter overflows.

The following conditions must be satisfied when the PWM mode is used.

(Timer register setting value) $<$ (2^n - 1 counter overflow setting value)

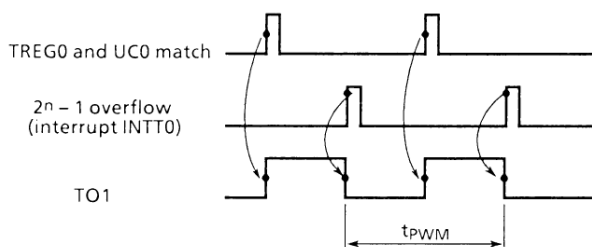(timer register setting value) $\neq$ 0

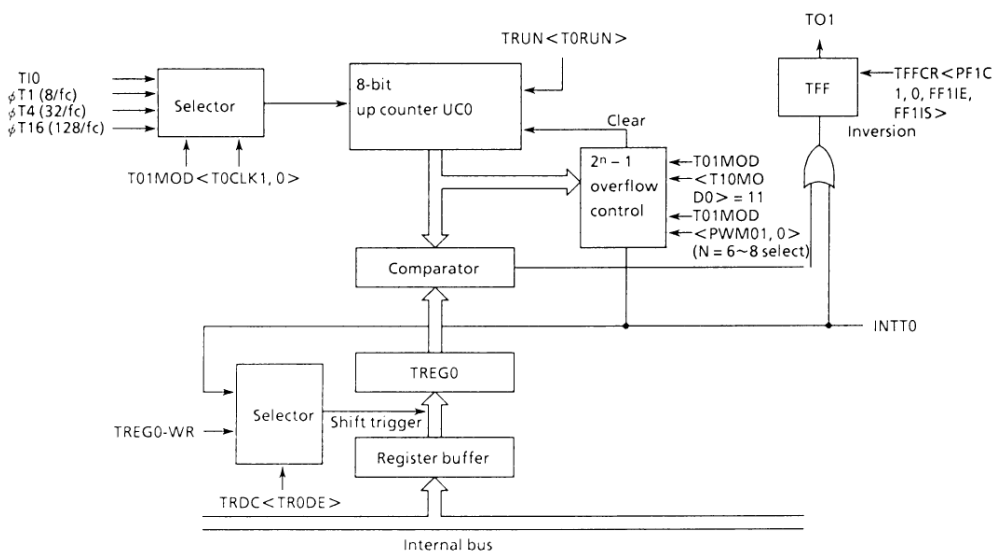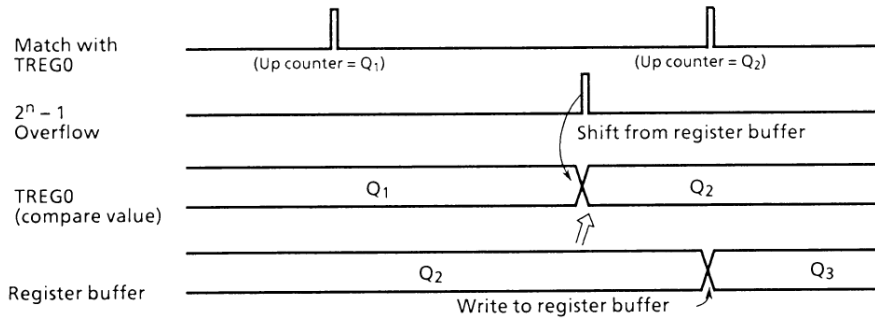Figure 3.8 (13) shows the block diagram for this mode
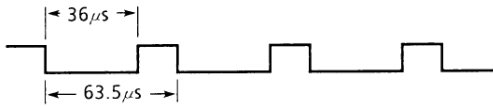

Figure 3.8 (13). 8-Bit PWM Output Mode Block Diagram

In this mode, the register buffer value is shifted in to TREG0 by $2^n - 1$ overflow detection when the TREG0 double buffer is enabled.

A double buffer facilitates small duties.

Example: Outputting the PWM waveform shown below to the TO1 pin using timer 0 at fc = 16MHz.

Setting a PWM cycle of 63.5 with μs øT1 = 0.5μs (@fc = 16MHz)

$$63.5\mu s \div 0.5\mu s = 127 = 2^7 - 1$$

Thus, n = 7 is set.

The "LOW" level cycle is 36μs; therefore, at øT1 = 0.5μs

$36\mu s \div 0.5\mu s = 72 = 48H$ is loaded to TREG0.

```
            MSB          LSB
            7 6 5 4 3 2 1 0
    TRUN  ← x x - x - - - 0    Stops timer 0 and timer 1, and clears them to zero.
    TO1MOD ← 1 1 1 0 - - 0 1   Sets the 8-bit PWM mode (cycle = 2⁷ - 1) and sets the
                               input clock to øT1
    TFFCR ← - - - - 1 0 1 x    Clears TFF1 and enables inversion.
    TREG0 ← 0 1 0 0 1 0 0 0    Writes 48H
    P6CR  ← - 1 - - - - - -  ⎫
    P6FR  ← - 1 x x x x - -  ⎬ Sets P66 as the TO1 pin.
    TRUN  ← x x 1 x - - - 1    Starts timer 0 counting.
```

Note : x ; don't care   - ; no change

Table 3.8 (3) PWM Cycle and Selection of $2^n - 1$ Counter

	PWM cycle (@fc = 122.5MHz)			PWM cycle (@fc = 16MHz		
	øT1	øT4	øT16	øT1	øT4	øT16
$2^6 - 1$	40.3µs	161µs	645µs	31.5µs	126µs	504µs
$2^7 - 1$	81.2µs	325µs	1.30ms	63.5µs	254µs	1.01ms
$2^8 - 1$	163µs	652µs	2.61ms	127µs	510µs	2.04ms

(5)   Table 3.8 (4) Shows the Settings for Each 8-bit Timer Mode

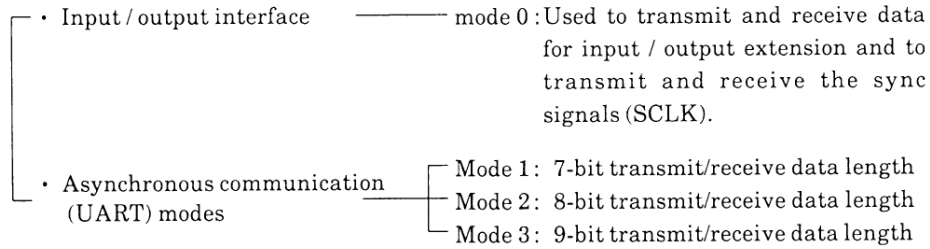### Table 3.8 (4) Setting Register for Each Timer Mode

Register	T01MOD (T23MOD)				TFFCR
Bit Name	T10M (T23M)	PWM0 (PWM2)	T1CLK (T3CLK)	T0CLK (T2CLK)	FF1IS (FF3IS)
Function	Timer mode	PWM cycle	Upper timer input clock	lower timer input clock	reverse signal selection of timer F/F
16-bit timer mode	01	–	–	External, øT1, øT4, øT16, (01, 10, 11)	–
8-bit timer x 2ch	00	–	–	External, øT1, øT4, øT16, (00, 01, 10, 11)	0: lower timer output 1: upper timer output
8-bit PPG x 1ch	10	–	–	External, øT1, øT4, øT16, (00, 01, 10, 11)	–
8-bit PWM x 1ch	11	$2^6 - 1, 2^7 - 1, 2^8 - 1$ (01, 10, 11)	–	External, øT1, øT4, øT16, (00, 01, 10, 11)	–
8-bit timer x 1 ch	11	–	øT1, øT16, øT256, (01, 10, 11)	–	Output disenable

(Note) –: Don't care

3.9 Serial Channels

The TMP90C051F has two built-in serial input channels (CH0 and CH1) for full-duplex asynchronous communication (UART) and input/output extension. Operation is the same for both channels, so channel 0 will be used for explanatory purposes. The serial channels have the following operation modes.

· Input / output interface ——— mode 0 : Used to transmit and receive data for input / output extension and to transmit and receive the sync signals (SCLK).

· Asynchronous communication (UART) modes

Mode 1 : 7-bit transmit/receive data length
Mode 2 : 8-bit transmit/receive data length
Mode 3 : 9-bit transmit/receive data length

A parity bit can be added in modes 1 and 2. Mode 3 has a wake-up function with which the master controller starts the slave controller with a serial link (multi-controller system).

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

· Mode 0 (input/output interface mode)

bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
← Direction of transfer

· Mode 1 (7-bit UART mode)

start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | stop

start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | parity | stop

· Mode 2 (8-bit UART mode)

start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | stop

start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | parity | stop

· Mode 3 (9-bit UART mode)

start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | stop

start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | bit 8 | stop (Wake up)

Address (select code) if bit 8 = 1
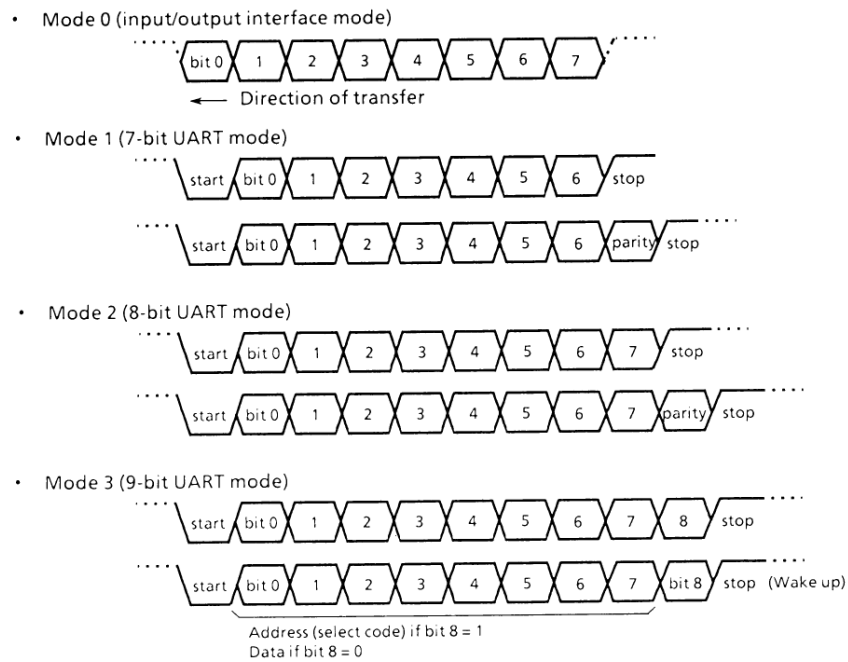Data if bit 8 = 0

Figure 3.9 (1). Data Formats

The serial channels have buffer registers for the temporary storage of transmit and receive data; therefore, transmitting and receiving can be performed independently (full duplex).

In the input/output interface mode, however, the SCLK (serial clock) pin is used for both transmitting and receiving, so only half duplex operation is possible.

The receive buffer register has a double buffer construction to prevent overflow errors and has a one frame margin until the CPU reads the received data. That is, one buffer stores the data already received and the other buffer receives the next frame data.

In the UART mode, a check function has been added that prevents the receive operation being started by mistake by the start bit due to noise. The start bit is sampled three times and the receive operation is started only when a normal start bit is detected at least two times.

The INTTX interrupt is generated when the transmit buffer is empty and a request for the next data to be sent is sent to

the CPU. The INTRX interrupt is generated when data are stored in the receive buffer and a read request is sent to the CPU. The flags (SCCR <OERR, PERR, FERR>) are set when an overrun error, parity error or framing error occurs during the receive operation.

In the input/output interface mode, sync signal (SCLK) input is also possible and data can be sent and received using an external clock. (Note)

The serial channels have a dedicated built-in baud rate generator. Any desired baud rate can be set by dividing by 2 - 16 4 clocks (øT0, øT2, øT8, øT32) from the prescaler (also used for the 8-/16-bit timers).

Note: When transmitting and receiving using serial channel 0 in the input/output interface mode with external clock input, an external circuit and control are required for the SCLK0 pin. This circuit and control are shown below.

(Typical circuit)   (Typical control)
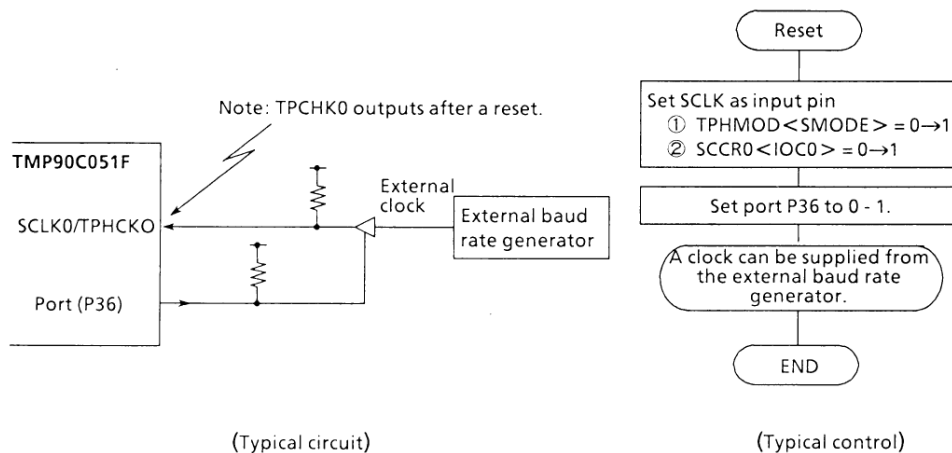
3.9.1 Control Registers

All serial channels are controlled by 4 control registers

(SCMOD, SCCR, BRGCR and P2FR). Transmit and receive data are stored to the SCBUF register.
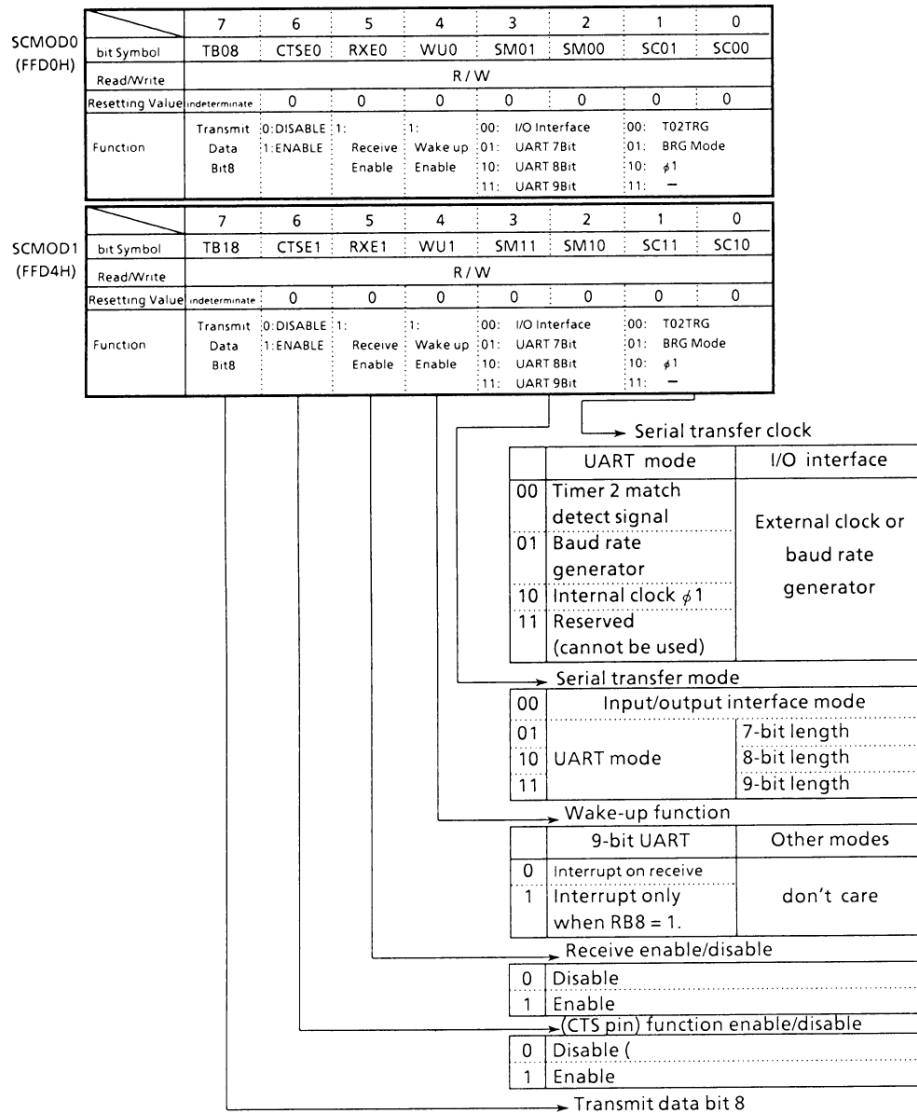

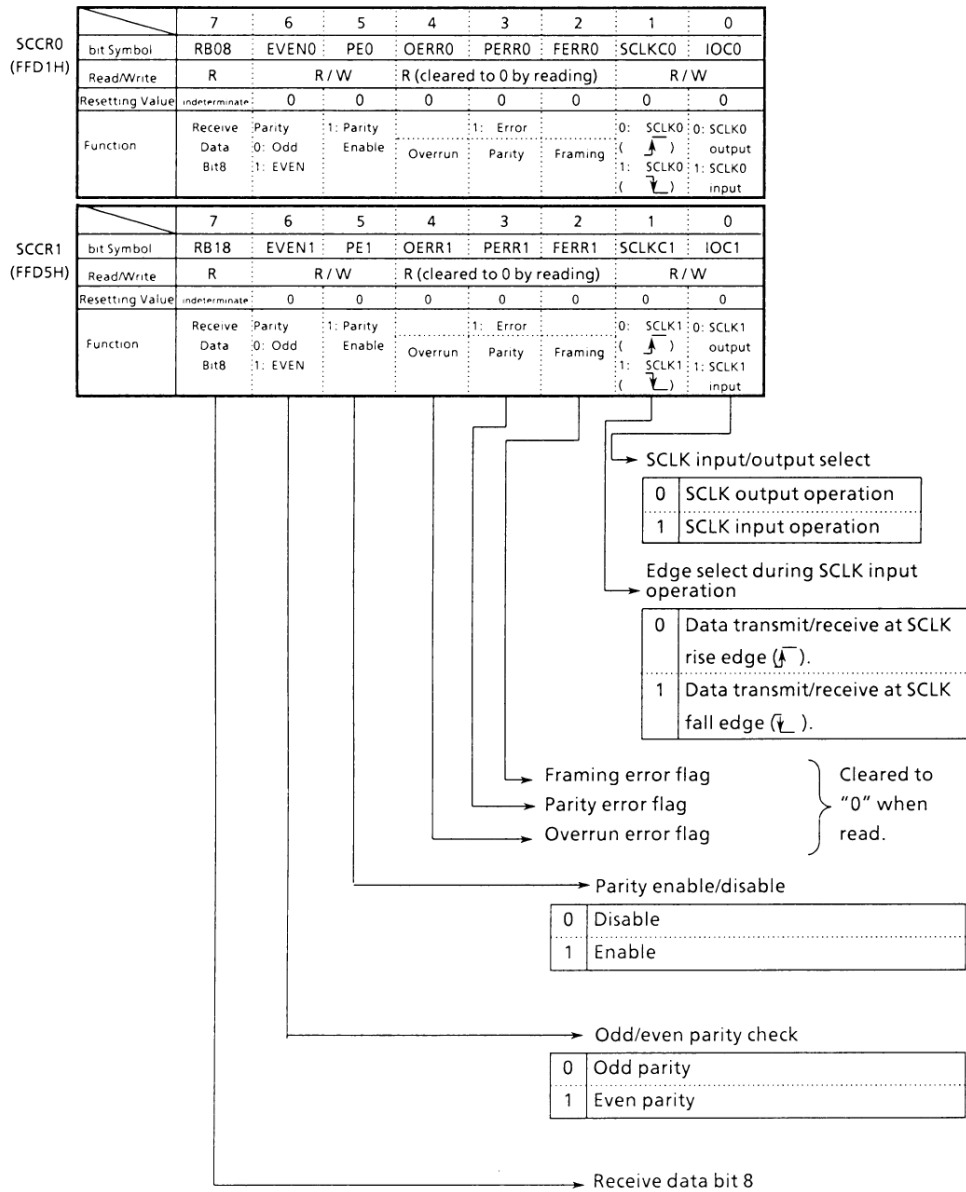**Figure 3.9 (2). Serial Channel Mode Register (SCMOD0/1)**

SCCR0 (FFD1H)		7	6	5	4	3	2	1	0
	bit Symbol	RB08	EVEN0	PE0	OERR0	PERR0	FERR0	SCLKC0	IOC0
	Read/Write	R	R / W		R (cleared to 0 by reading)			R / W	
	Resetting Value	indeterminate	0	0	0	0	0	0	0
	Function	Receive Data Bit8	Parity 0: Odd 1: EVEN	1: Parity Enable	1: Error Overrun	Parity	Framing	0: SCLK0 (↑) 1: SCLK0 (↓)	0: SCLK0 output 1: SCLK0 input

SCCR1 (FFD5H)		7	6	5	4	3	2	1	0
	bit Symbol	RB18	EVEN1	PE1	OERR1	PERR1	FERR1	SCLKC1	IOC1
	Read/Write	R	R / W		R (cleared to 0 by reading)			R / W	
	Resetting Value	indeterminate	0	0	0	0	0	0	0
	Function	Receive Data Bit8	Parity 0: Odd 1: EVEN	1: Parity Enable	1: Error Overrun	Parity	Framing	0: SCLK1 (↑) 1: SCLK1 (↓)	0: SCLK1 output 1: SCLK1 input

SCLK input/output select

0	SCLK output operation
1	SCLK input operation

Edge select during SCLK input operation

0	Data transmit/receive at SCLK rise edge (↑).
1	Data transmit/receive at SCLK fall edge (↓).

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

Parity enable/disable

0	Disable
1	Enable

Odd/even parity check

0	Odd parity
1	Even parity

Receive data bit 8
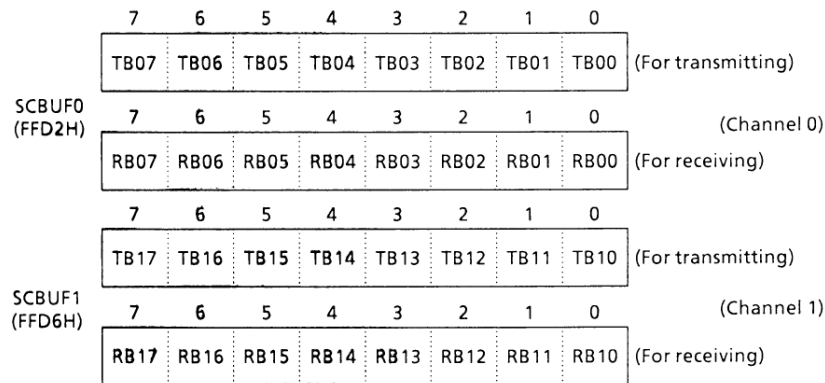
Figure 3.9 (3). Serial Channel Mode Register (SCCR0/1)

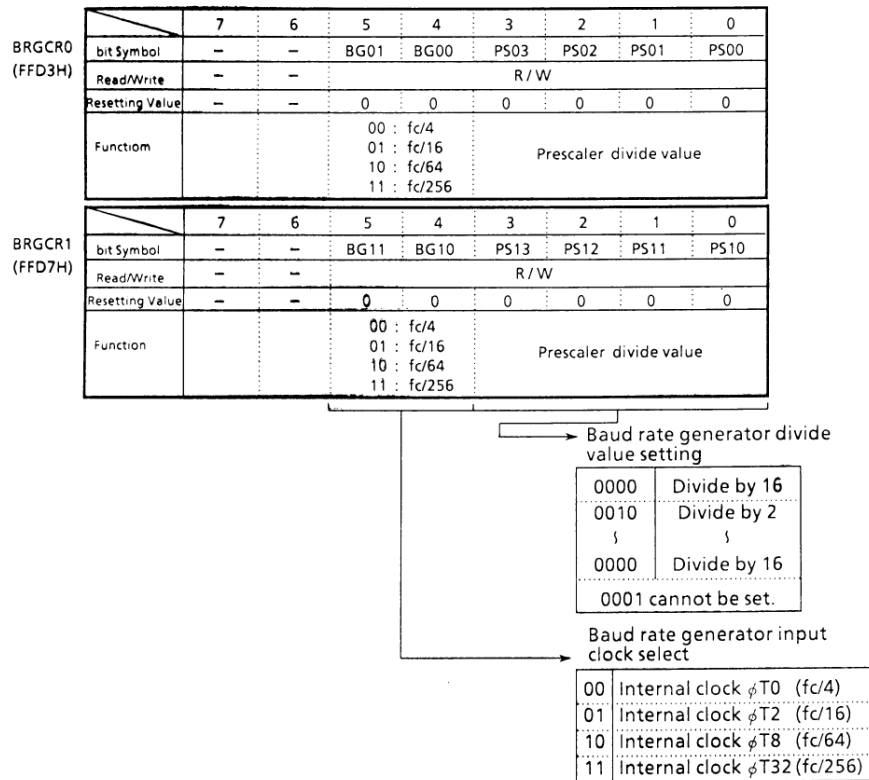**Figure 3.9 (4). Serial Transmit/Receive Buffer Register (SCBUF0/1)**

Figure 3.9 (5). Baud Rate Generator Control Register (BRGCR0/1)

P2FR (FFDFH)		7	6	5	4	3	2	1	0
	bit Symbol	SCLKC1	SCLKC0	ODE1	ODE0	WAITC1	WAITC0	—	Fixed at "1"
	Read/Write	R / W						—	R / W
	Resetting Value	0	0	0	0	0	0	—	1
	Function	SCLK1 control 0: SCLK1 input 1: SCLK1 output	SCLK0 control 0: SCLK0 input 1: SCLK0 output	TXD1 control 0:CMOS input 1:open drain output	TXD0 control 0:CMOS output 1:open drain output	Wait control 00: 2state Wait 01: normal Wait 10: non Wait 11: reserved			Write "1"

Explained in "3.1.3 Wait control"

TXD0 CMOS/open-drain output setting

0	CMOS output
1	Open drain output

TXD1 CMOS/open-drain output setting

0	CMOS output
1	Open drain output

SCLK0 setting

0	SCLK0 input
1	SCLK0 output

SCLK1 pin setting

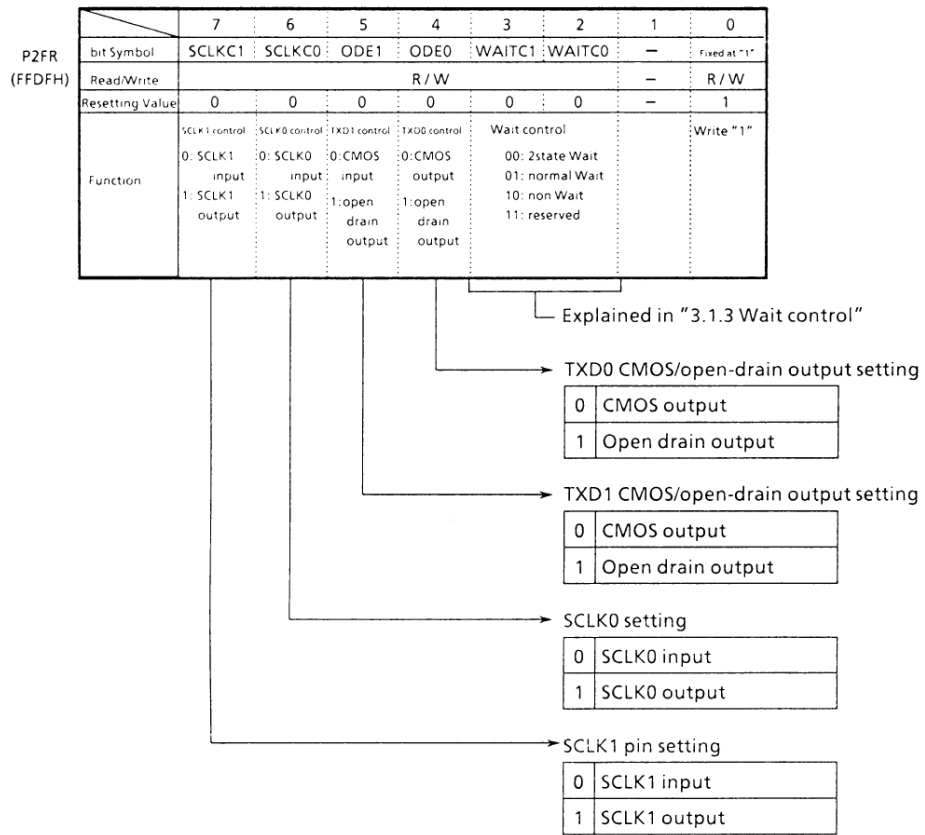| 0	SCLK1 input
1	SCLK1 output

Figure 3.9 (6). Port 2 Function Register (P2FR)

3.9.2 Configuration

Figure 3.9 (7) shows the serial channel (channel 0) block diagram.

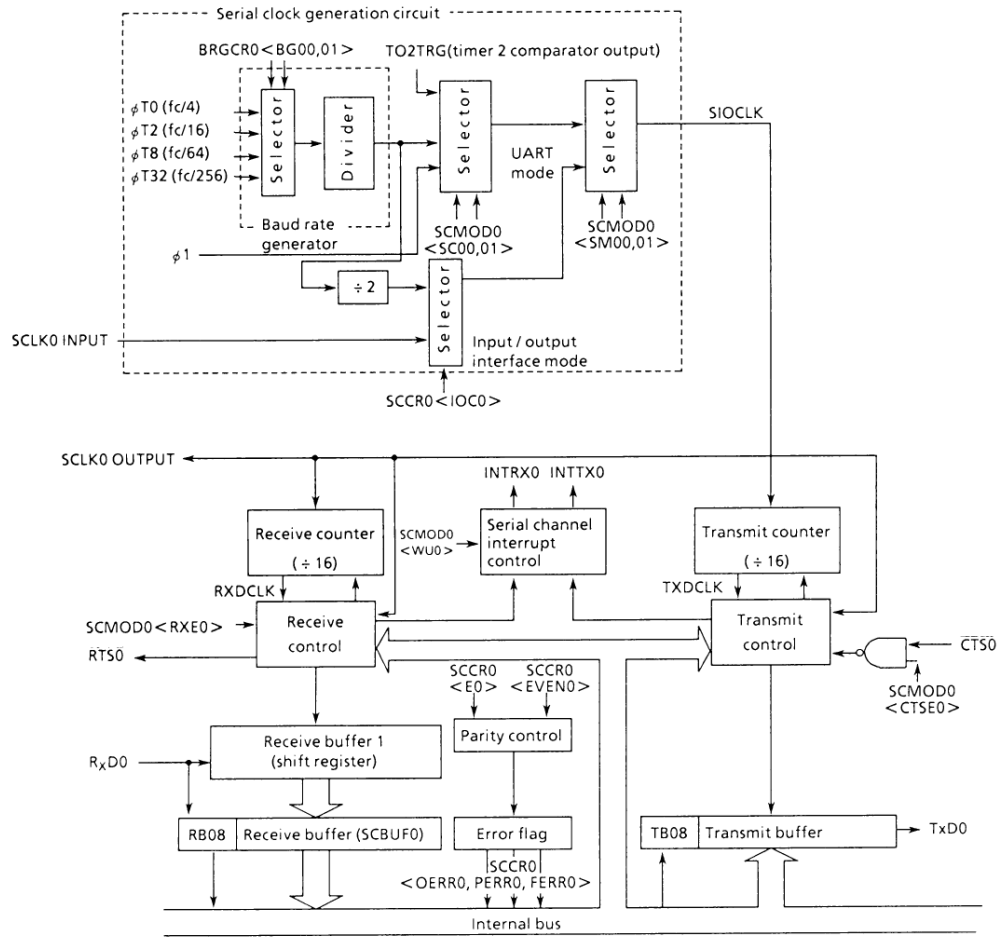Channels 0 and 1 have the same circuit configuration.

Figure 3.9 (7). Serial Channel (Channel 0) Block Diagram

① Baud rate generator

The baud rate generator is a circuit that generates the transmit/receive clocks that determine the transfer speed of the serial channels.

øT0 (fc/4), øT2 (fc/16), øT8 (fc/64) and øT32 (fc/256) from the 9-bit prescaler also used as a timer are used as the baud rate generator input clocks. The input clocks are selected by setting bits 5 and 4 (BG1, 0) of the baud rate generator control register BRGCR.

The baud rate generator has a built-in 4-bit divider. This divider determines the transfer speed by dividing by 2 ~ 16.

Next is the method used to calculate the baud rate when the baud rate generator is used.

• UART mode

$$\text{Baud Rate} \ = \ \frac{\text{baud rate generator input clock}}{\text{baud rate generator divide value}} \div 16$$

• Input/output interface mode

$$\text{Baud Rate} \ = \ \frac{\text{baud rate generator input clock}}{\text{baud rate generator divide value}} \div 2$$

The relationship to the input clock fundamental clock (fc) is as follows.

$$\begin{aligned} øT0 &= fc/4 \\ øT2 &= fc/16 \\ øT8 &= fc/64 \\ øT32 &= fc/256 \end{aligned}$$

Consequently, with a fundamental clock of fc = 12.288MHz, input clock øT2 (fc/16), and divider value of 5, the UART mode baud rate is as follows.

$$\text{Baud Rate} \ = \ \frac{fc/16}{5} \div 16$$

$$= 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600 \ (\text{bps})$$

Table 3.9 (1) shows a typical UART mode baud rate.

The serial channel baud rate can also be determined using the 8-bit timer 2. Table 3.9 (2) shows a typical baud rate determined using timer 2.

Table 3.9 (1) Baud Rate Selection (1) (Using the Baud Rate Generator) Unit: Kbps

FC	Input clock	øT0	øT2	øT8	øT32
	Divide value	(fc/4)	(fc/16)	(fc/64)	(fc/256)
9.8304MHz	–	2457.6	614.4	153.6	38.4
–	2	76.8	19.2	4.8	1.2
–	4	38.4	9.6	2.4	0.6
–	8	19.2	4.8	1.2	0.3
–	16	9.6	2.4	0.6	0.15
12.288MHz	–	3072.0	786.0	192.0	48.0
–	5	38.4	9.6	2.4	0.6
–	10	19.2	4.8	1.2	0.3
14.7456MHz	–	3686.4	921.6	230.4	57.6
–	3	76.8	19.2	4.8	1.2
–	6	38.4	9.6	2.4	0.6
–	12	19.2	4.8	1.2	0.3

Table 3.9 (2) Baud Rate Selection (2) (Using Timer: Input CLK øT1) Unit: Kbps

fc TREG2	12.288 MHz	12 MHz	9.8304 MHz	8 MHz	6.144 MHz
1H	96	–	76.8	62.5	48
2H	48	–	38.4	31.25	24
3H	32	31.25	–	–	16
4H	24	–	19.2	–	12
5H	19.2	–	–	–	9.6
8H	12	–	9.6	–	6
AH	9.6	–	–	–	4.8
10H	6	–	4.8	–	3
14H	4.8	–	–	–	2.4

Baud rate calculation method (using timer 2)

$$\text{Transmit/receive Rate} = \frac{1}{\text{TREG2}} \times \frac{1}{16} \times \text{timer2 (Input CLK)}$$

Input CLK of timer 2

øT1 = fc/8
øT4 = fc/32
øT16 = fc/128

② Serial clock generation circuit

This circuit generates the basic transmit/receive clock.

1) Input/output interface mode

In the SCCR0 <IOC0> = "1" SCLK0 output mode, the output from the baud rate generator as described above is divided by 2 to make the basic clock.
In the SCCR0 <IOC0> = "1" SCLK0 input mode, either the rise or fall edge, as set with the SCCR0 <SCLK0> register, is detected to make the basic clock.

2) Asynchronous communication (UART) mode

Either the baud rate generator clock described above, the internal clock 1 (312.5k-baud @ 10MHz), or the match detect signal from timer 2 is selected, as determined by the SCMOD <SC01, 00> register settings, to make the basic clock (SIOCLK).

③ Receive counter

The receive counter is a 4-bit binary counter used in the asynchronous communication (UART) mode and is counted up by SIOCLK. 16 SIOCLK clocks are used to receive 1 bit of data and the data are sampled at the 7th, 8th and 9th clocks.
The receive data are identified by majority logic by sampling 3 times. For example, if the data 1, 0, 1 are sampled by the 7th, 8th and 9th clocks, the data is identified as "1". If the values 0, 0, 1 are sampled, the data is identified as "0".

④ Receive control block

1) Input/output interface mode

In the SCCR0 <IOC0> = "0" SCLK0 output mode, the RxD0 pin is sampled at the rise of the shift clock output to the SCLK0 pin.
In the SCCR0 <IOC0> = "1" SCLK0 input mode, the RxD0 pin is sampled at the rise/fall of the SCLK0 input in accordance with the setting of the SCCR0 <IOC0> register.

2) Asynchronous communication (UART) mode

The receive control block has a start bit detection circuit that used majority logic. If "0" is detected twice or more by sampling three times, the start bit is judged to be correct and receiving starts.
Receive data are identified by majority logic even while receiving data.

⑤ Receive buffers

The receive buffers have a redundant construction to prevent overruns. Receive data are stored 1 bit at a time to receive buffer 1 (shift register type). When 7 or 8 bits of data have been stored, the are shifted to the other receive buffer (SCBUF0) and interrupt INTRX is generated. The CPU reads receive buffer 2 (SCBUF0). The next receive data can be stored to receive buffer 1 even before the CPU reads receive buffer 2 (SCBUF0). However an overrun error will occur unless receive buffer 2 (SCBUF0) is read before all bits of the next data have been received by receive buffer 1. If an overrun error occurs, the contents of receive buffer 2 and RB08 are held but the contents of receive buffer 1 are lost.

In the case of 8-bit UART with parity, the parity bit is stored to SCCR07 (RB08). In the case of 9-bit UART, the uppermost bit is stored to SCCR0 <RB08>.

In the case of 9-bit UART, wake-up operation of the slave controller can be enabled by setting SCMOD0 <WU0> to "1". After that, interrupt INTRX will be generated only when RB08 = 1.

⑥ Transmit counter

The transmit counter is a 4-bit binary counter used in the asynchronous communication (UART) mode and, like the receive counter, is counted up by SIOCLK. The transmit clock TXDCLK is generated every 16 clocks.
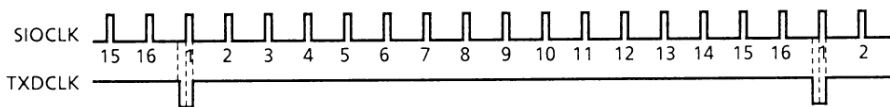
⑦ Transmit control block

1) Input/output interface mode

In the SCCR0 <IOC0> = "0" SCLK0 output mode, the transmit buffer data are output to the TxD0 pin one bit at a time at the rise of the shift clock output to the SCLK0 pin.
In the SCCR0 <IOC0> = "1" SCLK0 input mode, the transmit buffer data are output to the TxD0 pin one bit at a time at the rise/fall of the SCLK0 input in accordance with the setting of the SCCR01(SCLK0) register.

2) Asynchronous communication (UART) mode

When the CPU writes transmit data to the transmit buffer, transmitting starts from the next TxDCLK rise edge to make the transmit shift clock TxDSFT.

Hand-shake function

The TMP90C051F supports a hand-shake function by the connection of $\overline{CTS0}$ and $\overline{RTS0}$ of the other TMP90C051F.
The hand-shake function allows receiving/transmitting data on a frame basis to prevent overrun errors. This function is enabled or disabled by the control register SCMOD0 <CTSE0>.
When the last bit (parity bit or MSB) of 1-frame data is received by the receiving unit, the $\overline{RTS0}$ pin turns to the "H" level to request the transmission unit to halt transmission.
When the $\overline{CTS0}$ pin turned to the "H" level, the transmission unit halts transmission, after completing the current data transmission, until the pin turns to the "L" level. At this time, the interrupt INTTX is generated, to request the CPU to transfer data. Then the data is written into the transmission buffer, and the transmission unit is placed in the standby until the $\overline{CTS0}$ pin turned to the "L" level.
When the received data are read by the CPU, the $\overline{RTS0}$ pin returns to the "L" level, requesting that the transmission is restarted.
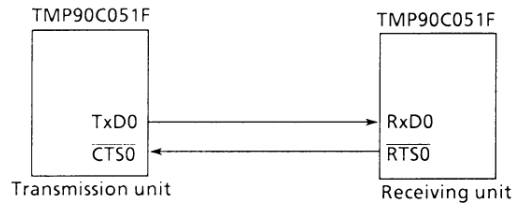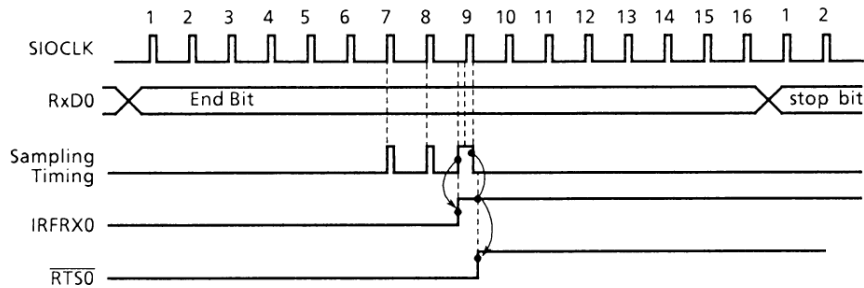
Figure 3.9 (8). Hand-shake Function

Note : In case of 8-bit asynchronous communication (UART), the last bit is the bit 7 in the non-parity mode, and the parity bit in the parity-added mode.

Figure 3.9 (9). Timing Chart of RTS (request to send) Signal
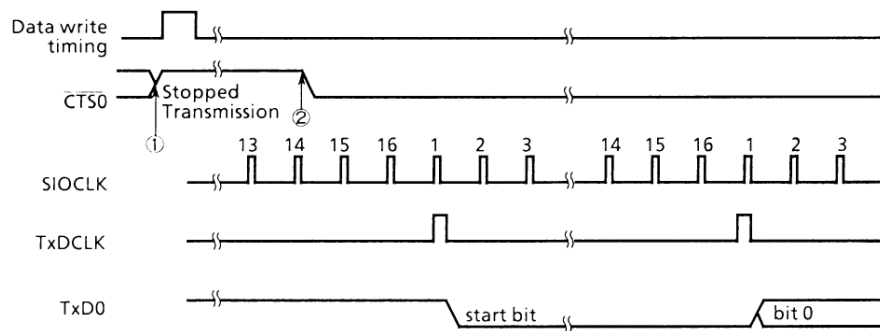
Note: 1) A Rise of the $\overline{CTS0}$ signal during the data transmission halts the transmission of the next data after the current data transmission.
2) The transmisison is restarted from the first fall of TXDCLK after a fall of the $\overline{CTS0}$ signal.

Figure 3.9 (10). Hand-shake by CTS (clear to send) Signal

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 Parity control circuit

Setting the serial channel control register SCCR <PE> to "1" allows the addition of a parity bit in transmitting/receiving data, only in the 7-bit UART or 8-bit UART mode. Either even or odd parity can be selected by the SCCR <EVEN> register.

In the transmission mode, the parity is automatically generated as based on the data written into the transmission buffer SCBUF, storing into the SCBUF <TB7> in the 7-bit UART mode or into <TB8> in the 8-bit UART mode for transmission. <PE> and <EVEN> should be designated before writing data into the transmission buffer.

In the receiving mode, the receiving data is shifted into the receiving buffer 1 and transferred to the receiving buffer 2 (SUBUF). The parity is generated from the data in the receiving buffer 2. A parity error is detected and the SCCR <PERR> flag is set if the parity status mismatches the SCBUF <RB7> in the 7-bit UART mode or <RB8> in the 8-bit UART mode.

⑩ Error flag

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

1) Overrun error (SCCR0 <OERR0>)

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) Parity error (SCCR0 <PERR0>)

The parity generated from the data that is transferred

to the receiving buffer 2 (SCBUF0) is compared with the parity bit received from the RxD terminal. Parity error occurs if they are not equal.

3) Framing error (SCCR0 <FERR0>)

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

⑪ 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	↑
Parity error timing	Center of last bit (Bit 8)	Center of last bit (Parity 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 addition 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	←	←

2) I/O interface mode

Receiving

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

3.9.3 Operation

(1)   Mode 0 (input/output interface mode)

This mode is used to increase the number of TMP90C051F input/output pins. This is done by using externally connected shift registers to transmit and receive data.

This mode has an SCLK0 output mode in which the sync clock (SCLK0) is output and an SCLK0 input mode in which an external. Sync clock (SCLK0) is input.

This mode has an SCLK0 output mode in which the sync clock (SCLK0) is output and SCLK0 input mode in which an external sync clock (SCLK0) is input.

• Typical SCLK0 output mode connection
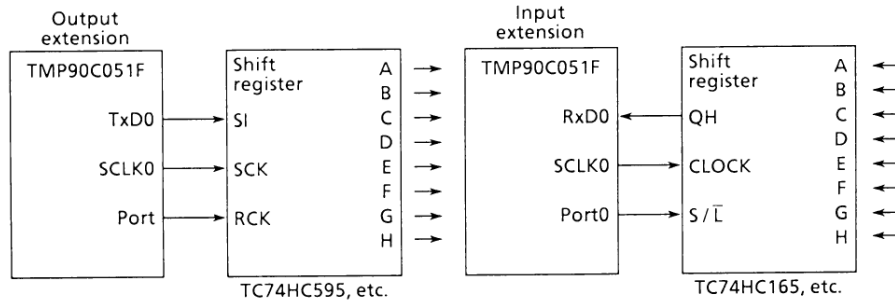
• SCLK0 input mode connection
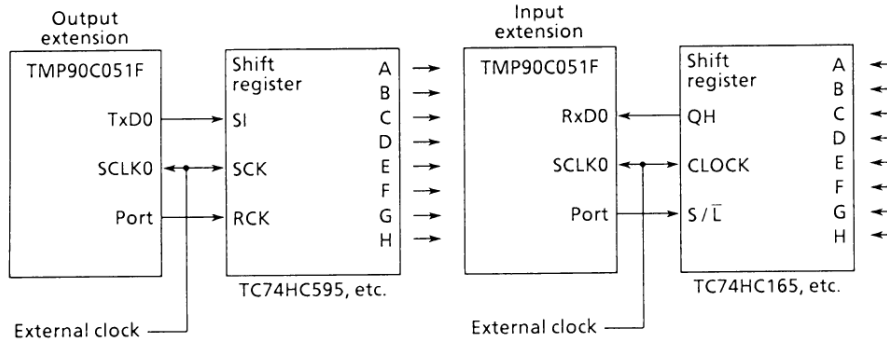
Figure 3.9 (11). Input/Output Interface Mode

TOSHIBA CORPORATION

① Transmitting

In the SCLK0 output mode, each time the CPU writes data to the transmit buffer, 8 bits of data are output to the TxD0 pin and the sync clock is output from the SCLK0 pin.
When all of the data have been output, IRF2 <IRFTX0> is set and interrupt INTTX0 is generated.
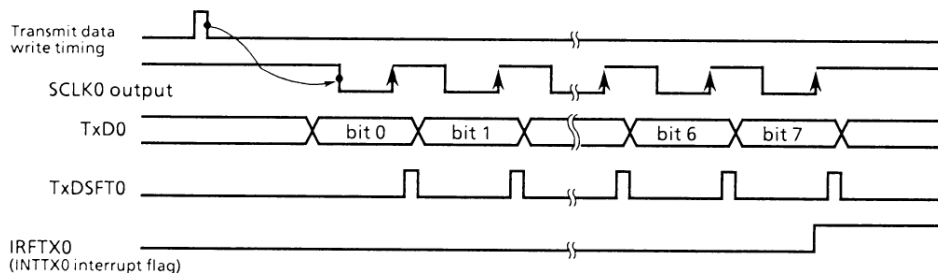
Figure 3.9 (12). Input/Output Interface Mode Transmit Operation (SCLK0 Output Mode)

In the SCLK0 input mode, 8 bits of data are output from the TxD0 pin when SCLK0 input is active while data are being written by the CPU to the transmit buffer.
When all of the data have been output, IRF2 <IRFTX0> is set and interrupt INTTX0 is generated.
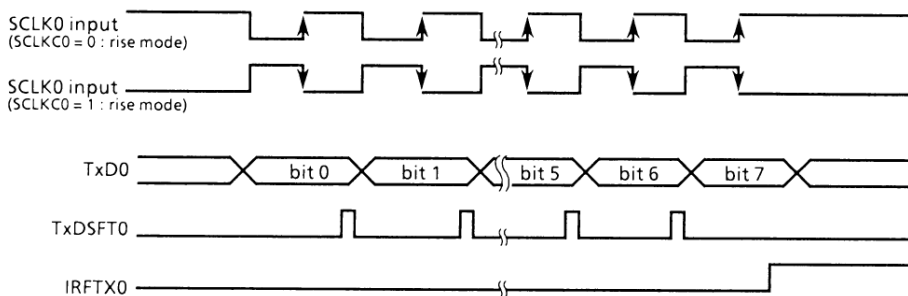
Figure 3.9 (13). Input/Output Interface Mode Transmit Operation (SCLK0 Input Mode)

② Receiving

In the SCLK0 output mode, the sync clock is output from the SCLK0 pin and the next data are shifted into receive buffer 1 each time the CPU reads the receive data and the receive interrupt flag IRFRX0 is cleared. When 8 bits of data have been received, they are shifted to receive buffer 2 (SCBUF0), IRF2 <IRFRX0> is again set and interrupt INTTX0 is generated.
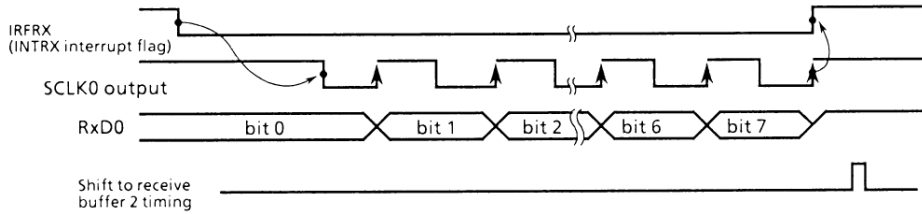
Figure 3.9 (14). Input/Output Interface Mode Receive Operation (SCLK0 Output Mode)

In the SCLK0 input mode, the next data are shifted into receive buffer 1 when SCKL0 input is active while the CPU is reading receive data and the receive interrupt flag IRFRX0 is cleared. When 8 bits of data have been received, they are shifted to receive buffer 2 (SCBUF0), IRFRX0 is again set and interrupt INTRX0 is generated.
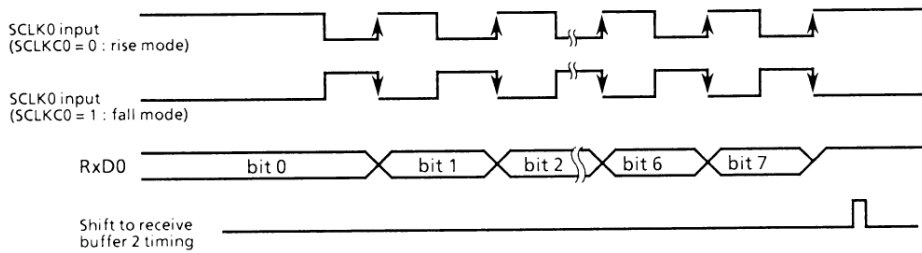
Figure 3.9 (15). Input/Output Interface Mode Receive Operation (SCLK0 Input Mode)

It is necessary to enable receiving (SCMOD <RXE0> = 1) before receiving data.

(2)   Mode 1 (7-Bit UART Mode)

The 7-bit UART mode is entered by setting serial channel mode register SCMOD0 <SM01, 00> to 01.

In this mode, a parity bit can be added. The parity bit is enabled and disabled with serial channel control register SCCR0 <PE0>. <PE0> = 1 enables parity; <PE> = 0 disables parity; and even or odd parity is selected SCCR0 <EVEN0>.

Setting example: The control register settings for transmitting data in the following format.

Direction of transfer (transfer speed: 2400bps @fc = 12.2884MHz)
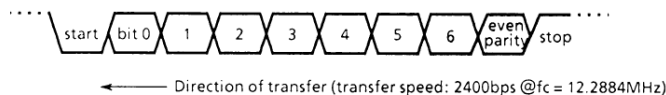
```
              7 6 5 4 3 2 1 0
SCMOD0 ← X 0 - X 0 1 0 1     Sets CM0 to 7-bit UART mode.
SCCR0  ← X 1 1 X X X X X     Sets even parity.
BRGCR0 ← 0 X 1 0 0 1 0 1     Sets 2400 bps.
TRUN   ← X X 1 - X - - -     Starts the prescaler for use by the baud rate generator.

INTE2  ← - - - 1 - - - -     Enables INTTX0 interrupt.
SCBUF0 ← * * * * * * * *     Loads the transmit data.
```

Note: X ; don't care   - ; no change

(3)   Mode 2 (8-Bit UART Mode)

The 8-bit UART mode is entered by setting serial channel mode register SCMOD0 <SM01, 00> to 10. In this mode, a parity bit can be added. The parity bit is enabled and disabled with serial channel control register SCCR0 <PE0>. <PE0> = 1 enables parity; PE = 0 dis-ables parity; and even or odd parity is selected with SCCR0 <EVEN0>.

Setting example: The control register settings for receiving data in the following format

Direction of transfer (transfer speed: 9600 baud @ fc = 12.288MHz)
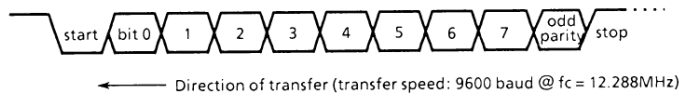
```
Main settings          7 6 5 4 3 2 1 0

           SCMOD0 ←  - 0 1 X 1 0 0  1    Sets channel 0 to the 8-bit UART mode and enables receiving.
           SCCR0  ←  X 0 1 X X X X X     Sets odd parity.
           BRGCR0 ←  0 X 0 1 0 1 0 1     Sets 9600 bps
           TRUN   ←  X X 1 X - - - -     Starts the prescaler for use by the baud rate generator.
           INTE2  ←  - 1 - - - - - -     Enables INTRX1 interrupt.
```

```
Typical processing with INTRX

           Acc    ← SCCR0 AND 00011100   Performs error check.
           if Acc ≠  0 then error
           Acc    ← SCBUF0               Reads receive data.

     Note :   X ; don't care    - ; no change
```

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

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

The uppermost (9th) bit is written to SCMOD0 <TB08> when transmitting and is stored to SCCR0 <RB08> when receiving. Always start with the uppermost bit and proceed toward SCBUF0 when reading or writing the buffer.

Wake-up function

In the 9-bit UART mode, slave controller wake-up operation is enabled by setting SCMOD0 <WU0> to "1". Interrupt INTRX0 is generated when SCCR0 <RB08> = 1.
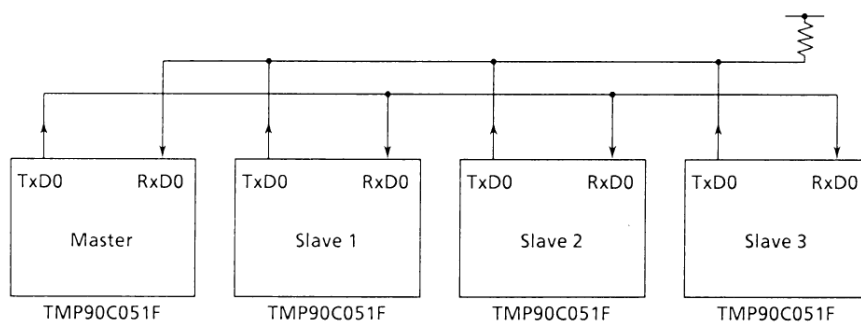
Caution: Always set the TxD0 pin of the slave controller to the open drainoutput mode

Figure 3.9 (16). Serial Link Using the Wake-up Function

Protocol

① Set the master and slave controller to the 9-bit UART mode.

② Set the SCMOD0 <WU0> bit of a slave controller to "1" to enable receiving.

③ The master controller transmits 1 frame, including the select code (8 bits) of the slave controller. The uppermost bit SCMOD0 <TB08> is set to "1" at this time.

Slave controller select code   "1"
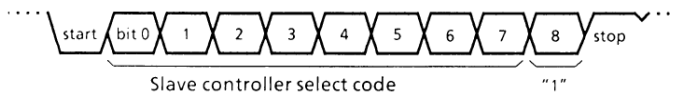
④ The slave controller receives the frames and, if its own select code matches the received data, clears the <WU0> bit to "0".

⑤ The master controller transmits data to the specified slave controller (with <WU0> cleared to "0"). At this time, the uppermost bit <TB08> is cleared to "0".

Data   "0"
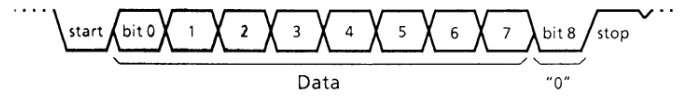
⑥ Even with <WU0> set to "1", the slave controller will neither generate interrupt INTRX0 nor read receive data as long as <RB08> is cleared to "0".

When both <WU0> and <RB08> are set to "1", the slave controller will always generate interrupt INTRX0 and read the receive data.

Then, when <WU0> is cleared to "0", this is reported to the master controller to inform it that receiving has ended.

Setting example: Serially linking two slave controllers using internal clock ø1 as the transfer clock at using ch0.
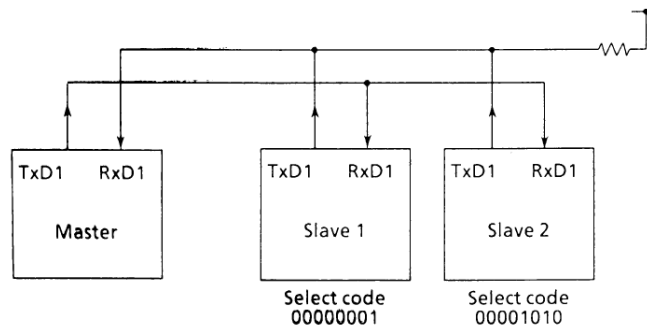
• Master controller setting

```
Main    ┌ INTE2  ←  1 - 1 - - - - -        Enables INTRX1 and INTTX1.
        │ SCMOD1 ←  1 0 1 0 1 1 1 0        Sets 9-bit UART mode transfer clock to φ1.
        └ SCBUF1 ←  0 0 0 0 0 0 0 1        Sets the slave 1 select code.

INTTX1  ┌ SCMOD1 ←  0 - - - - - - -        Sets SCMOD1<TB18> to "0".
interrupt│ SCBUF1 ←  * * * * * * * *        Loads the transmit data.
```

• Slave 2 setting

```
Main    ┌ INTE2  ←  1 - 1 - - - - -        Enables INTRX0 and INTTX0.
        └ SCMOD1 ←  0 0 1 1 1 1 1 0        Sets <WU1> to "1" with the 9-bit UART mode transfer
                                           clock φ1 (fc/2).

INTRX   ┌ Acc   ← SCBUF0
interrupt│ if Acc = select code
        └ then SCMOD0 ← - - - 0 - - - -    <WU1> is cleared to 0
```

Note :   X ; don't care   - ; no change

3.10 TPH Serial Interface

When copying or storing receive data, the TPH (Thermal Printer Head) serial interface converts the parallel data from MPU and DMA to serial data and transmits it to the TPH driver. The TPH serial interface uses SIO channel 0.

Thus, SIO channel 0 has independent control registers for the serial interface mode, UART mode and input/output interface mode. Any of these three can be selected by program.

The TPH serial interface mode is in effect after a reset.

3.10.1 Features

(1)   Three transmit modes can be selected. An interrupt is generated after the specified number of bytes has been sent.

- A4 mode (1728 bits): 216 bytes are sent
- B4 mode (2048 bits): 256 bytes are sent
- A3 mode (2592 bits): 324 bytes are sent

(2)   Either an internal or external can be selected as the transmit clock.

- In the internal clock mode, fxtal 1/4, 1/8, 1/32, 1/128 baud rates can be selected. (@ fxtal = 12.5MHz: 3.125MHz/bit sent at 1/4 rate.)
- In the external clock mode, an external 4MHz (max.) input clock can be used.

(3)   TPH driver and interface pins

- TPHCKO   :transmit clock output pin
- TPHSD   :transmit data output pin
- $\overline{\text{TPHLAT}}$:latch output pin
- $\overline{\text{LATWAIT}}$:$\overline{\text{TPHLAT}}$ output wait control input pin
- TPHCK1   :external clock input pin

These pins are multiplexed internally to function as TXD0, RXD0, $\overline{\text{RTS0}}$, $\overline{\text{CTS0}}$ and SCLK0 when the input/output interface issued in the UART mode.
For details, see 3.10.3 (6) Operation (pin multiplexing).

3.10.2 Control Registers

(1) Transmit Data Write Register

TPHBUF (FFE9H)		7	6	5	4	3	2	1	0
	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
	Read/Write	W							
	Resetting Value	1	1	1	1	1	1	1	1

(2) TPH Control Register

TPHMOD (FFEAH)		7	6	5	4	3	2	1	0
	bit Symbol	LACHWAIT	SMODE	TPHCK	BR1	BR0	TRM1	TRM0	Fixed at "0"
	Read/Write	W							
	Resetting Value	0	0	0	0	0	0	0	0
	Function	WAIT for $\overline{TPHLAT}$ output 0: release of WAIT 1: hold of WAIT	0: TPH mode 1: normal serial mode	TPHoutput transfer clock 0: internal 1: external	Transfer rate 00: fc/4 01: fc/8 10: fc/32 11: fc/128		Transfer mode 00: 216 byte count (A4-size) 01: 256 byte count (B4-size) 10: 324 byte count (A3-size)		Write "0"

00	216 bytes	(A4)
01	256 bytes	(B4)
10	324 bytes	(A3)

00	1/4 of fxtal
01	1/8 of fxtal
10	1/32 of fxtal
11	1/128 of fxtal

0	Internal clock mode
1	External clock mode

0	TPH mode
1	UART, input/output interface mode The UART and input/output interface modes are set with the serial interface control register of SIO channel 0.

0	Release of WAIT for $\overline{TPHLAT}$ output.
1	Hold of WAIT for $\overline{TPHLAT}$ output.

Note: also TMP90C51F can control the wait of $\overline{TPHLAT}$ output by external pin ($\overline{LATWAIT}$).

(3)   TPH Serial Control Register

		7	6	5	4	3	2	1	0
TPHSCR (FFEBH)	bit Symbol	SCR7	SCR6	SCR5	SCR4	SCR3	SCR2	SCR1	EXINT
	Read/Write				W				
	Resetting Value	0	0	0	0	0	0	0	0
	Function	Shift register area The TPH shift register is cleared by writing "1" to SXR7~SCR1							0: internal interrupt mode 1: external interrupt mode

(a) The TPH shift register (TPH BUF; Byte Counter) can be cleared by writing "1" to D7 ~ D1. When TPH serial interface transmitting has been stopped on the way, this command must be written again before writing the next transmit data.

(Note) Write data "1" to <SCR7 ~ 1> before the user uses firstly TPH serial I/F.

(b) Interrupt generate mode

The <EXINT> bit is set to "0" after a reset.
The <EXINT> bit is used to select the internal interrupt generation mode (transmit interrupts [INTTX0] are generated only internally) or the external interrupt generation mode (started by the AND of the start signal from INT3 and the internal transmit interrupt enable status).

<EXINT> = 0: internal interrupt generation mode
<EXINT> = 1: external interrupt generation mode.

3.10.3 Operation

(1) Clock Selector

- In the internal clock mode, it is possible to output the transmit clock from TPHCK0 and select a clock rate of øT0 (fc/4), øT1 (fc/8), T4 (fc/32), T16 (fc/128) from the internal 9 bit prescaler (Timer Common use) (@ x'tal 12.5MHz: max. 3.125MHz/bit transfer at the 1/4 rate).
- In the external clock mode, the clock input from TPHCK1 is used for transmit operations (max. 4.0MHz/bit). If a user uses the internal clock mode, operate the prescaler.

(2) Byte Counter

Bits TPHMOD <TRM1, 0> (Mode = 0) of the control register (#FFEAH) are used to specify the number of bytes to be transferred. The byte counter is reset to "0" after one line has been transmitted.

(3) $\overline{\text{TPHLAT}}$ Output

$\overline{\text{TPHLAT}}$ is set to "Low" active when the byte counter has counted the number of bytes in one line.

(4) $\overline{\text{LATWAIT}}$ Input

Wait control by software or from the external pin ($\overline{\text{LATWAIT}}$) can be used to wait for the $\overline{\text{TPHLAT}}$ output. Waits are applied by keeping $\overline{\text{LATWAIT}}$ at low active from the TPHCK0 fall that transmits the last bit of the last byte of the transmits data to the next TPHCK0 rise.

(5) Interrupt Generation

INTTX0: Generate at the trigger of the buffer empty after TPH I/F transfer the 1 byte data

INTLINE: Generate at the trigger of 1 line data end with TPHLAT output

Interrupt generation is enabled and disabled by setting the INTE 1/2 register at address #FFCD/FFCEH. Interrupt generation is disabled after a reset.

Table 3.10 (1) shows the interrupt generation sources for SIO channel 0. Set SIO channel 0 interrupts as shown below in accordance with the mode setting (operating mode).

a) TPH Serial Interface Mode

Set the mode bit of TPHMOD <SMODE> to "0".

When controlling the writing of transmit data to the TPH serial interface from external pin INT3, set the TPHSCR <EXINT> to "1". In this case, transmit interrupts are generated in accordance with the AND logic of the internal transmit shift register status and external pin INT3 status.

When the TPHSCR <EXINT> is set to "0", transmit interrupts are generated in accordance with the internal transmit register status only.

Table 3.10 (1) Serial Channel 0 Interrupt Generation Sources

Serial channel 0 mode	UART, Input/Output interface mode		TPH mode	
Start	**External**	**Internal**	**External**	**Internal**
Interrupt source				
INTTX	–	INTTX	INTTX•INT3	INTTX
INTRX	–	INTRX	–	–
INTLINE	–	–	INTLINE	

(b) UART, I/O Interface Mode

Set the TPHMOD <SMODE> to "1" and the TPHSCR <EXINT> to "0". Figure 3.10 (1) shows the SIO channel 0 interrupt generation circuit.
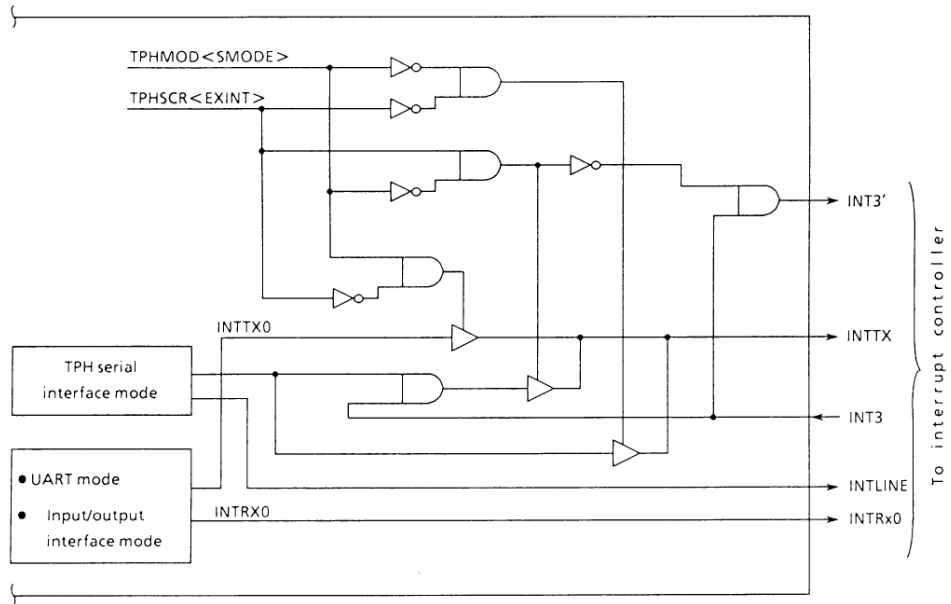
Figure 3.10 (1). SIO Channel 0 Interrupt Generation Circuit

(6)   Pin Multiplexing   channel 0 pins.

Table 3.10 (2) shows the multiplexing of the serial

Table 3.10 (2) Pin multiplexing

Pin name	Pin no.	After reset and in TPH serial interface mode			UART mode		
		TPHMOD <SMODE>	Function	Input/output	TPHMOD <SMODE>	Function	Input/output
RxD0/LATWAIT	30		LATWAIT	Input		RxD0	Input
TxD0/TPHSD	31		TPHSD	Output		TxD0	Output
SCLK0/TPHCK0	32	0	TPHCK0	Output	1	SCLK0	Input/Output
RTS0/TPHCKI	33		TPHCKI	Input		RTS0	Output
CTS0/TPHLAT	34		TPHLAT	Output		CTS0	Input

Pin name	Pin no.	Input/output interface mode (1)		
		TPHMOD <SMODE>	Function	Input/output
RxD0/LATWAIT	30		RxD0	Input
TxD0/TPHSD	31		TxD0	Output
SCLK0/TPHCK0	32	1	SCLK0	Input...in external clock mode Output...when using the internal baud rate generator
RTS0/TPHCKI	33		RTS0	Output
CTS0/TPHLAT	34		CTS0	Input

3.10.4 Transmit Operation Timing

Typical transmit operations using the TPH serial interface are shown below.

(1)   Send Format Example 1

: A4 mode transmitting using an internal clock and external $\overline{\text{LATWAIT}}$ insertion.
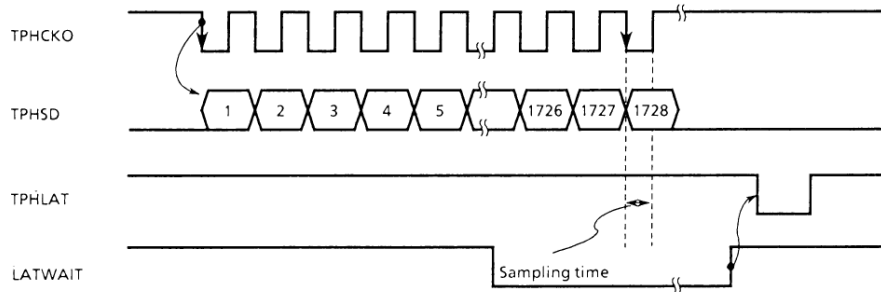
(* Waits are applied by keeping $\overline{\text{LATWAIT}}$ at low active from the fall of TPHCK0 that transmits the last bit of the last byte of the transmit data to the next TPHCK rise.)

(2)   Send Format Example 2

: B4 mode transmitting using an external clock.
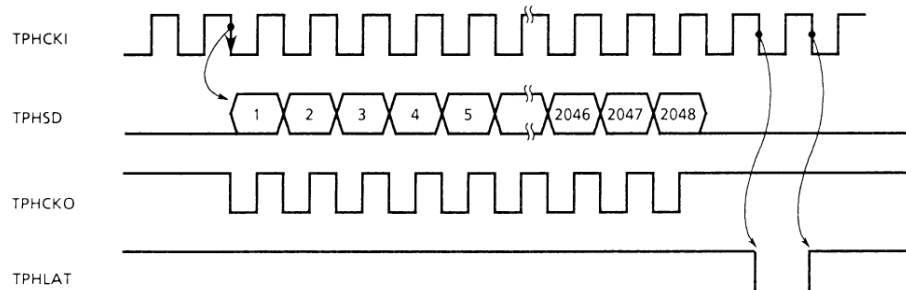
(3)   Send Format Example 3
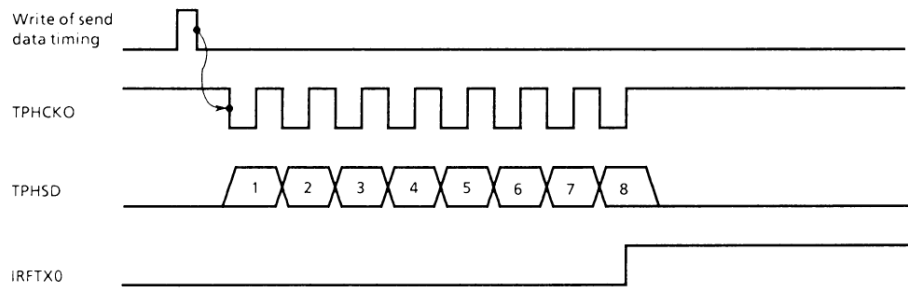
(4)   Send Format Example 4
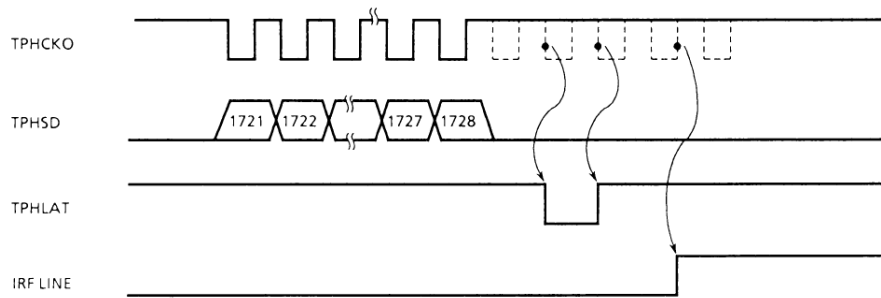
3.10.5 Typical TPH Driver Connections

(1)   Using an External Clock
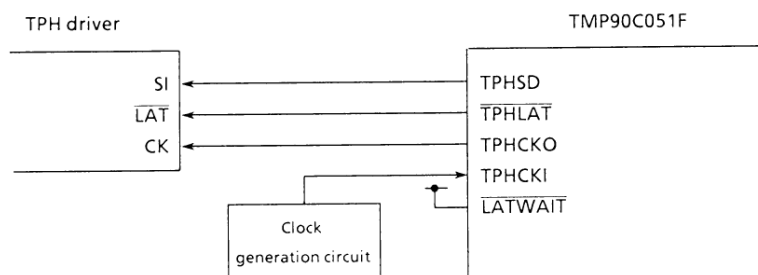
(2)   Using LATWAIT and an Internal Clock
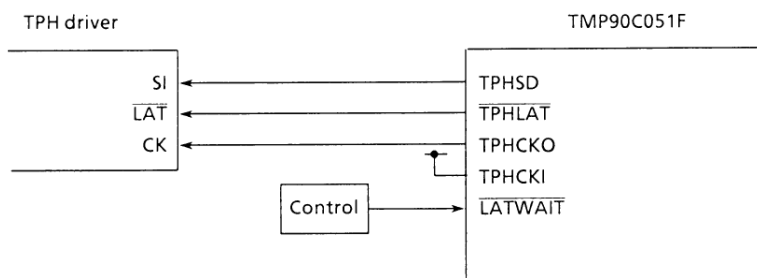
(3)   Send Format Example 3
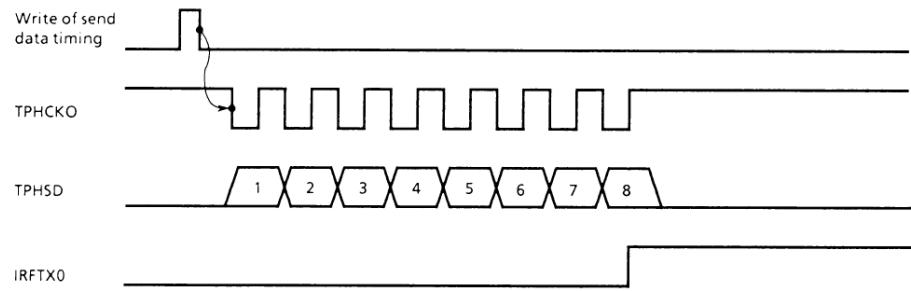
(4)   Send Format Example 4
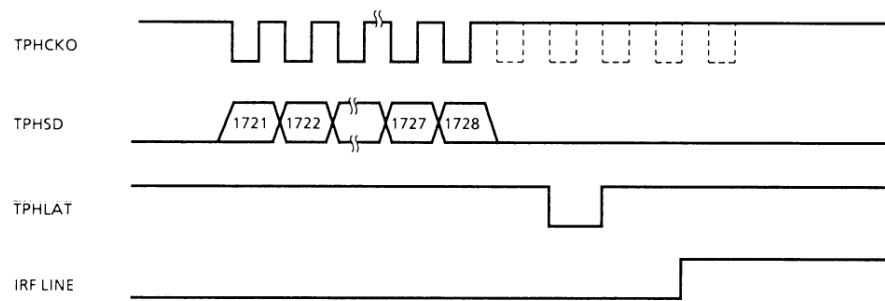
3.11 Real Time Clock (RTC)

3.11.1 RTC Function Overview

1) Clock (hour, minute, second, month, day [youbi], date [hinichi], year, leap year)

2) Auto-calendar

3) Choice of 24-hour clock or 12-hour clock (A.M./P.M.)

4) ±30 seconds correction function (software adjustment)

5) Alarm output ($\overline{\text{ALARM}}$)

6) Internal interrupt generation by alarm output

3.11.2 RTC Block Diagram
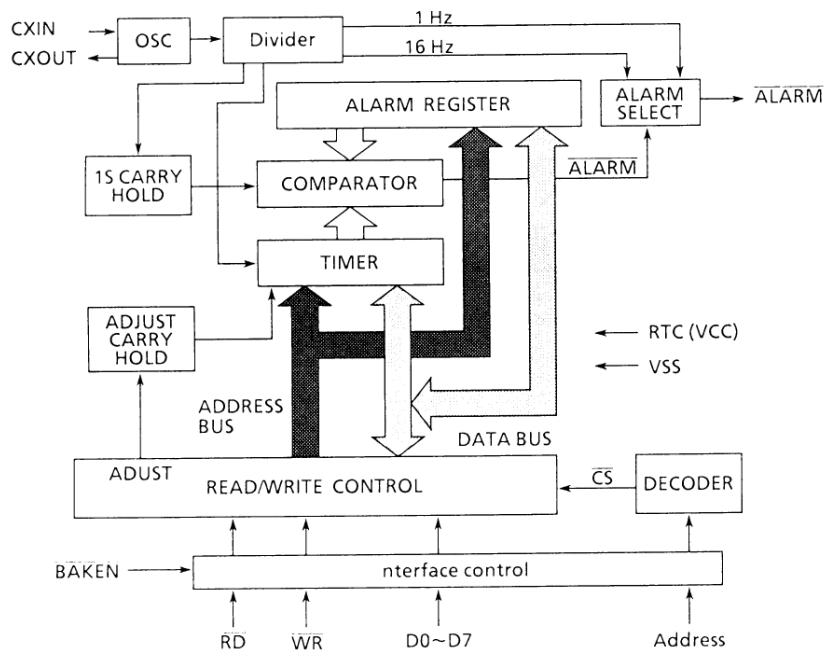
Figure 3.11 (1). RTC Block Diagram

TOSHIBA CORPORATION

3.11.3 RTC Input/Output Address Map and Control Registers

Table 3.11 (1) PAGE 0 (timer function) Register

Symbol	I/O address	D7	D6	D5	D4	D3	D2	D1	D0	Setting Contents	Read/Write
SECR	#FFE0H	(Note 1) –	40 seconds	20 seconds	10 seconds	8 seconds	4 seconds	2 seconds	1 seconds	Second	R/W
MINR	#FFE1H	–	40 minutes	20 minutes	10 minutes	8 minutes	4 minutes	2 minutes	1 minutes	Minute	R/W
HOUR	#FFE2H	–	–	20 hours (PM/AM)	10 hours	8 hours	4 hours	2 hours	1 hours	Hour	R/W
DAYR	#FFE3H	–	–	–	–	–	W2	W1	W0	Day	R/W
DATER	#FFE4H	–	–	20 days	10 days	8 days	4 days	2 days	1 days	Date	R/W
MONTHR	#FFE5H	—	–	–	10 months	8 months	4 mnonths	2 months	1 months	Month	R/W
YEARR	#FFE6H	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 years	Year (last 2 digits)	R/W
PAGER	#FFE7H	–	–	–	ADJUST	TIMER En/$\overline{\text{dis}}$	ALARM EN/$\overline{\text{dis}}$	–	Page	Page register (Note 2)	R/W
RESTR	#FFE8H	1Hz dis/$\overline{\text{En}}$	16Hz dis/$\overline{\text{En}}$	TIMER REST	ALARM REST	0	0	0	0	Reset register	Write only

Table 3.11 (2) Page 1 (alarm function) Registers

Symbol	I/O address	D7	D6	D5	D4	D3	D2	D1	D0	Setting Contents	Read/Write
SECR	#FFE0H	–	–	–	–	–	–	–	–	–	R/W
MINR	#FFE1H	–	40 minutes	20 minutes	10 minutes	8 minutes	4 minutes	2 minutes	1 minutes	Alarm minute	R/W
HOUR	#FFE2H	–	–	20 hours (PM/AM)	10 hours	8 hours	4 hours	2 hours	1 hours	Alarm hour	R/W
DAYR	#FFE3H	–	–	–	–	–	W2	W1	W0	Alarm day	R/W
DATER	#FFE4H	–	–	20 days	10 days	8 days	4 days	2 days	1 day	Alarm date	R/W
MONTHR	#FFE5H	–	–	–	–	–	–	–	24/$\overline{12}$	24-hour clock select bit	R/W
YEARR	#FFE6H	–	–	–	–	–	–	LEAP1	LEAP0	Leap year	R/W
PAGER	#FFE7H	–	–	–	–	TIMER En/$\overline{\text{dis}}$	ALARM EN/$\overline{\text{dis}}$	–	Page	Page register (Note 2)	R/W
RESTR	#FFE8H	1Hz dis/$\overline{\text{En}}$	16Hz dis/$\overline{\text{En}}$	TIMER REST	ALARM REST	0	0	0	0	Reset register (Note 3)	Write only

(Note 1) "–" is ignored when writing but is read as "1".
(Note 2) Page is specified with the D0 bit (Page) of #FFE7H.
　　　　Page = 0 sets PAGE 0
　　　　Page = 1 sets PAGE 1
(Note 3) All register bits are not initialized by $\overline{\text{RESET}}$.

3.11.4 Register Explanation

The real timer clock (RTC) is not initialized by RESET. RTC operates with unstable values after power-on; therefore, load the time, month, day, date, year and leap year to the registers before starting operation.

(1)　Setting the Second Register (Page 0 only)

SECR (FFE0H)		7	6	5	4	3	2	1	0
	bit Symbol	—	SE6	SE5	SE4	SE3	SE2	SE1	SE0
	Read/Write	—	R/W						
	Resetting Value	—	undefined						
	Function	—	40 seconds	20 seconds	10 seconds	8 seconds	4 seconds	2 seconds	1 second

6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0 second
0	0	0	0	0	0	1	1 second
0	0	0	0	0	1	0	2 seconds
0	0	0	0	0	1	1	3 seconds
0	0	0	0	1	0	0	4 seconds
0	0	0	0	1	0	1	5 seconds
0	0	0	0	1	1	0	6 seconds
0	0	0	0	1	1	1	7 seconds
0	0	0	1	0	0	0	8 seconds
0	0	0	1	0	0	1	9 seconds
0	0	1	0	0	0	0	10 seconds

6	5	4	3	2	1	0	
0	0	1	1	0	0	1	19 seconds
0	1	0	0	0	0	0	20 seconds

6	5	4	3	2	1	0	
0	1	0	1	0	0	1	29 seconds
0	1	1	0	0	0	0	30 seconds

6	5	4	3	2	1	0	
0	1	1	1	0	0	1	39 seconds
1	0	0	0	0	0	0	40 seconds

6	5	4	3	2	1	0	
0	1	1	1	0	0	1	39 seconds
1	0	0	0	0	0	0	40 seconds

6	5	4	3	2	1	0	
1	0	1	1	0	0	1	59 seconds

Bit D7 is ignored when writing
but is read as "1".

(2) Setting the Minute Register (Page 0/Page1)

		7	6	5	4	3	2	1	0
MINR (FFE1H)	bit Symbol	—	M16	M15	M14	M13	M12	M11	M10
	Read/Write	—	R / W						
	Resetting Value	—	undefined						
	Function	—	40 minutes	20 minutes	10 minutes	8 minutes	4 minutes	2 minutes	1 minute

0	0	0	0	0	0	0	0 minute
0	0	0	0	0	0	1	1 minute
0	0	0	0	0	1	0	2 minutes
0	0	0	0	0	1	1	3 minutes
0	0	0	0	1	0	0	4 minutes
0	0	0	0	1	0	1	5 minutes
0	0	0	0	1	1	0	6 minutes
0	0	0	0	1	1	1	7 minutes
0	0	0	1	0	0	0	8 minutes
0	0	0	1	0	0	1	9 minutes
0	0	1	0	0	0	0	10 minutes

0	0	1	1	0	0	1	19 minutes
0	1	0	0	0	0	0	20 minutes

0	1	0	1	0	0	1	29 minutes
0	1	1	0	0	0	0	30 minutes

0	1	1	1	0	0	1	39 minutes
1	0	0	0	0	0	0	40 minutes

1	0	0	1	0	0	1	49 minutes
1	0	1	0	0	0	0	50 minutes

1	0	1	1	0	0	1	59 minutes

Bit D7 is ignored when writing but is read as "1".

(3)   Setting the Hour Register (Page 0/Page 1)   ① When D0 of FFE5H = 1 (24-hr. clock display)

		7	6	5	4	3	2	1	0
HOUR	bit Symbol	—	—	HO5	HO4	HO3	HO2	HO1	HO0
(FF22H)	Read/Write	—	—	R / W					
	Resetting Value	—	—	undefined					
	Function	—	—	20 hours (P/A)	10 hours	8 hours	4 hours	2 hours	1 hour

0	0	0	0	0	0	0	hour
0	0	0	0	0	1	1	hour
0	0	0	0	1	0	2	hours
0	0	0	0	1	1	3	hours
0	0	0	1	0	0	4	hours
0	0	0	1	0	1	5	hours
0	0	0	1	1	0	6	hours
0	0	0	1	1	1	7	hours
0	0	1	0	0	0	8	hours
0	0	1	0	0	1	9	hours
0	1	0	0	0	0	10	hours
0	1	1	0	0	1	19	hours
1	0	0	0	0	0	20	hours
1	0	0	0	1	1	23	hours

Bits D7 and D6 are ignored when writing but are read as "1".

② When D0 of FFE5H = 0 (12-hr. clock display)

		7	6	5	4	3	2	1	0
HOUR	bit Symbol	—	—	HO5	HO4	HO3	HO2	HO1	HO0
(FF22H)	Read/Write	—	—	R / W					
	Resetting Value	—	—	undefined					
	Function	—	—	20 hours (P/A)	10 hours	8 hours	4 hours	2 hours	1 hour

		0	0	0	0	0	0	0 hour (AM)
		0	0	0	0	0	1	1 hour
		0	0	0	0	1	0	2 hours
		0	0	0	0	1	1	3 hours
		0	0	0	1	0	0	4 hours
		0	0	0	1	0	1	5 hours
		0	0	0	1	1	0	6 hours
		0	0	0	1	1	1	7 hours
		0	0	1	0	0	0	8 hours
		0	0	1	0	0	1	9 hours
		0	1	0	0	0	0	10 hours
		0	1	0	0	0	1	11 hours
		1	0	0	0	0	0	0 hour
		1	0	0	0	0	1	1 hour

Bits D7 and D6 are ignored when writing but are read as "1".

(4)   Setting the Date Register (Page 0/Page 1)

		7	6	5	4	3	2	1	0
DAYR	bit Symbol	—	—	—	—	—	WE2	WE1	WE0
(FFE3H)	Read/Write	—	—	—	—	—	R / W		
	Resetting Value	—	—	—	—	—	undefined		
	Function	—	—	—	—	—	W2	W1	W0

						0	0	0	Sunday
						0	0	1	Monday
						0	1	0	Tuesday
						0	1	1	Wednesday
						1	0	0	Thursday
						1	0	1	Friday
						1	1	0	Saturday

Bits D7 - D3 are ignored when writing but are read as "1".

(5)   Setting the Date Register (Page 0/Page 1)

		7	6	5	4	3	2	1	0
DATER (FFE4H)	bit Symbol	—	—	DA5	DA4	DA3	DA2	DA1	DA0
	Read/Write	—	—	R / W					
	Resetting Value	—	—	undefined					
	Function	—	—	20 days	10 days	8 days	4 days	2 days	1 day

		5	4	3	2	1	0	
		0	0	0	0	0	1	1 day
		0	0	0	0	1	0	2 days
		0	0	0	0	1	1	3 days
		0	0	0	1	0	0	4 days
		0	0	0	1	0	1	5 days
		0	0	0	1	1	0	6 days
		0	0	0	1	1	1	7 days
		0	0	1	0	0	0	8 days
		0	0	1	0	0	1	9 days
		0	1	0	0	0	0	10 days
		0	1	0	0	0	1	11 days
		0	1	1	0	0	1	19 days
		1	0	0	0	0	0	20 days
		1	0	1	0	0	1	29 days
		1	1	0	0	0	0	30 days
		1	1	0	0	0	1	31 days

Bits D7 and D6 are ignored
when writing but are read as "1".

(6)   Setting the Month Register (Page 0 only)

MONTHR (FFE5H)		7	6	5	4	3	2	1	0
	bit Symbol	−	−	−	MO4	MO3	MO2	MO1	MO00
	Read/Write	−	−	−	R / W				
	Resetting Value	−	−	−	undefined				
	Function	−	−	−	10 months	8 months	4 months	2 months	1 month

			0	0	0	0	1	1 month
			0	0	0	1	0	2 months
			0	0	0	1	1	3 months
			0	0	1	0	0	4 months
			0	0	1	0	1	5 months
			0	0	1	1	0	6 months
			0	0	1	1	1	7 months
			0	1	0	0	0	8 months
			0	1	0	0	1	9 months
			1	0	0	0	0	10 months
			1	0	0	0	1	11 months
			1	0	0	1	0	12 months

Bits D7, D6 and D5 are ignored
when writing but are read as "1".

(7)   Setting 24-Hour/12-Hour Clock (Page 1 only)

The MO0 bit of address #FFE5H is used to set the 24- hour/12-hour clock of the timer function.

MONTHR (FFE5H)		7	6	5	4	3	2	1	0
	bit Symbol	−	−	−	−	−	−	−	MO0
	Read/Write	−	−	−	−	−	−	−	R / W
	Resetting Value	−	−	−	−	−	−	−	undefined
	Function								1: 24-Hour 0: 12-Hour

Bit 7~1 are ignored when
writing but are read as "1".

0	sets the 12-hour clock
1	sets the 24-hour clock
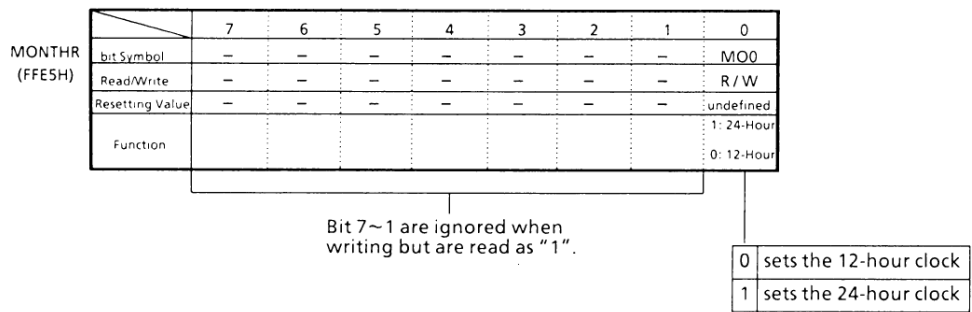
(8)   Setting the Year (Last 2 Digits) Register (Page 0 only)

YEARR (FFE6H)		7	6	5	4	3	2	1	0
	bit Symbol	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
	Read/Write	R / W							
	Resetting Value	undefined							
	Function	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 year

1	0	0	1	0	0	0	0	90 years
1	0	0	1	0	0	0	1	91
1	0	0	1	0	0	1	0	92
1	0	0	1	0	0	1	1	93
1	0	0	1	0	1	0	0	94
1	0	0	1	0	1	0	1	95
1	0	0	1	0	1	1	0	96
1	0	0	1	0	1	1	1	97
1	0	0	1	1	0	0	0	98
1	0	0	1	1	0	0	1	99
0	0	0	0	0	0	0	0	0 year

(9)   Setting the Leap Year Register (Page 1 only)

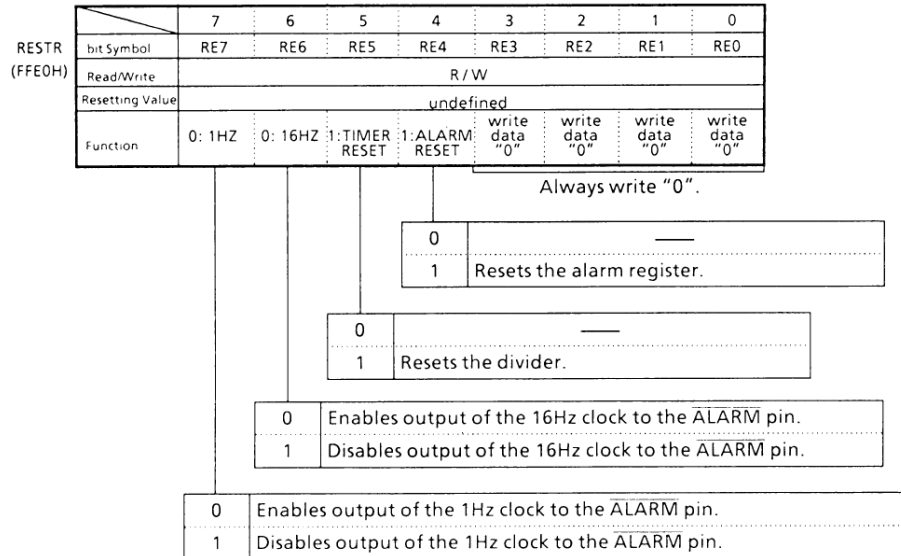| YEARR (FFE6H)		7	6	5	4	3	2	1	0
	bit Symbol	—	—	—	—	—	—	YE1	YE0
	Read/Write	—	—	—	—	—	—	R / W	
	Resetting Value	—	—	—	—	—	—	—	—
	Function	—	—	—	—	—	—	00: leap year 01: 1st year after leap year 10: 2nd year after leap year 11: 3rd year after leap year	

Ignored when writing but read as "1".

0	0	During a leap year.
0	1	One year after a leap year.
1	0	Two years after a leap year.
1	1	Three years after leap year.

(10)　Setting the Page Register (Page 0/Page 1)

PAGER (FFE7H)		7	6	5	4	3	2	1	0
	bit Symbol	—	—	—	PA4	PA3	PA2	—	PA0
	Read/Write	—	—	—	W	R / W		—	R / W
(RMW disable)	Resetting Value	—	—	—	—	undefined		—	undefined
	Function	—	—	—	1: ADJUST	TIMER 1: ENABLE 0: DISABLE	TIMER 1: ENABLE 0: DISABLE	—	PA0 1: PAGE0 0: PAGE1

	0	Specifies Page 0.
	1	Specifies page 1.

	0	Disables alarms.
	1	Enables alarms.

	0	Disables timer.
	1	Enables timer.

	0	——
	1	Corrects the seconds. Writing "1" to this bit resets the number of seconds to "0" when the value is 0 - 29, and increments the number of minutes by "1" and resets the number of seconds to "0" when the value is 35 - 59.

(Page0 only)

Bits D7 - D5 and D1 are ignored
when writing but are read as "1".

(11)　Setting the Reset Register (Page 0/Page 1)

RESTR (FFE0H)		7	6	5	4	3	2	1	0
	bit Symbol	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
	Read/Write	R / W							
	Resetting Value	undefined							
	Function	0: 1HZ	0: 16HZ	1:TIMER RESET	1:ALARM RESET	write data "0"	write data "0"	write data "0"	write data "0"

Always write "0".

	0	——
	1	Resets the alarm register.

	0	——
	1	Resets the divider.

	0	Enables output of the 16Hz clock to the $\overline{\text{ALARM}}$ pin.
	1	Disables output of the 16Hz clock to the $\overline{\text{ALARM}}$ pin.

	0	Enables output of the 1Hz clock to the $\overline{\text{ALARM}}$ pin.
	1	Disables output of the 1Hz clock to the $\overline{\text{ALARM}}$ pin.

3.11.5 Alarms

(1)   ALARM Output Pin

Any one of the following three signals can be selected with SELECT circuit and output to the ALARM pin.

① The ALARM signal output from the comparator drops to "Low" when the contents of the timer and alarm register match
Set as follows to output the ALARM signal to the ALARM pin.

PAGER	<PA2>: ALARM Enable = 1
RESTER	<RE7>: 1Hz Output Disable = 1
	<RE6>: 16Hz Output Disable = 1

② A 1Hz signal obtained by dividing 32kHz signal from

OSC with the divider is output from the ALARM pin. Set as follows.

PAGER	<PA2>: ALARM DISABLE = 0
RESTER	<RE7>: 1Hz Output enable = 0
	<RE6>: 16Hz Output Disable = 1

③ A 16Hz signal obtained by dividing the 32kHz signal from OSC with the divider is output from the ALARM pin. Set as follows.

PAGER	<PA2>: ALARM DISABLE = 0
RESTER	<RE7>: 1Hz Output disable = 1
	<RE6>: 16Hz Output enable = 0
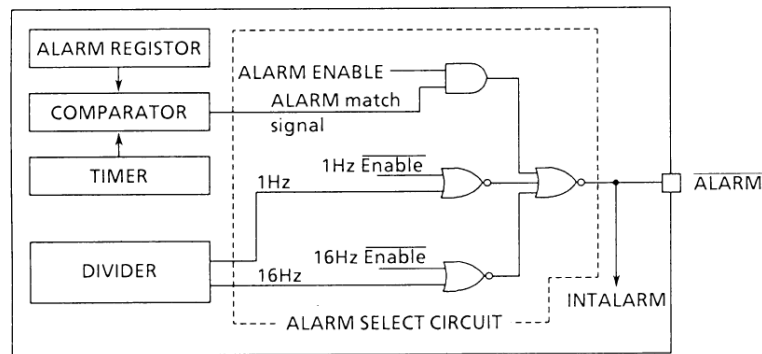
Figure 3.11 (2). Alarm SELECT Circuit

(2)   ALARM Reset Cautions

"0" is output to $\overline{\text{ALARM}}$ when the timer and alarm register contents match for the minute, hour, day and date items while alarms are enabled.

After an alarm reset, however, items for which nothing has been written are considered to match, regardless of the timer contents.

However, the two digits (10's and units) of the minute, hour and date items form a single item, so set both digits when writing.

For example, to output an alarm at the same time every day, it is only necessary to set the hour (both the 10's and units digits) and minute (both the 10's and units digits) after executing an alarm reset. An alarm reset clears the contents of the alarm register to "0". To compare "0" (to compare "0" for hour and minute,

for example, 00 hour, 00 minute), however, all items enter the "don't care" status when an alarm reset is executed with alarms enabled and the $\overline{\text{ALARM}}$ signal is output until an alarm is set. Because of that, it is necessary to disable alarms before executing an alarm reset.

(3)   INTALARM (Internal)

As with $\overline{\text{ALARM}}$ signal output, interrupts can be generated with any of the three following signals. 1) Comparator output, 2) 1Hz, 3) 16Hz.

Interrupts are generated internally in synchronization with falls of the $\overline{\text{ALARM}}$ pin output.

INTALARM is reset by system resets (when $\overline{\text{RESET}}$ = "Low").
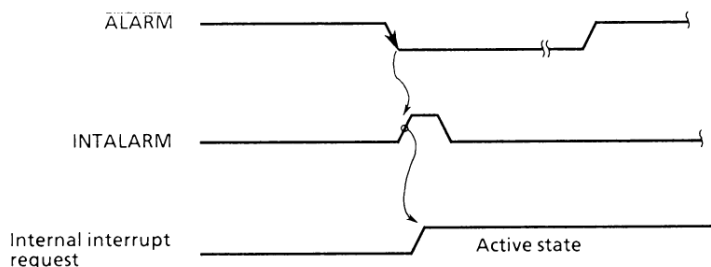
3.11.6 Typical Programming Sequence

(1)   When Reading Timer Data

① Errors are possible when a carry to the next higher digit of an internal counter occurs while reading timer data. Thus, use the following method to read the data correctly.
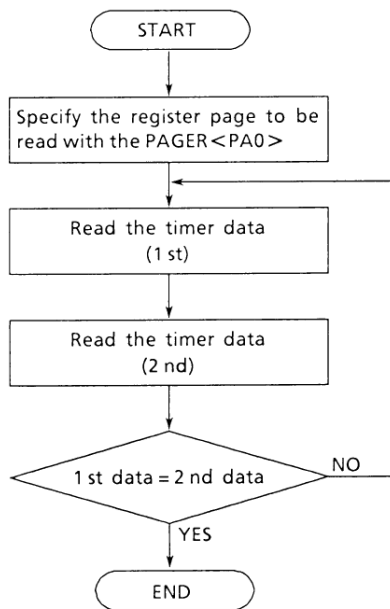

**Figure 3.11 (3). Typical Timer Data Read Sequence**

When reading timer data as shown in Figure 3.11 (3), the timer data are read twice and the contents are then compared to determine if a carry operation A carry operation is indicated when a comparison shows that the occurred. contents differ, so the data are read again.

② Reading timer data when $\overline{\text{ALARM}}$ output was used

Data can be read at the $\overline{\text{ALARM}}$ output rise edge by

detecting $\overline{\text{ALARM}}$ = 1 with a 1Hz INTALARM interrupt routine.
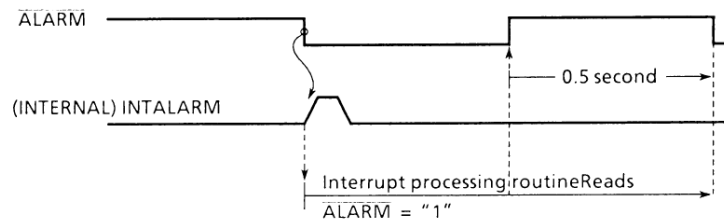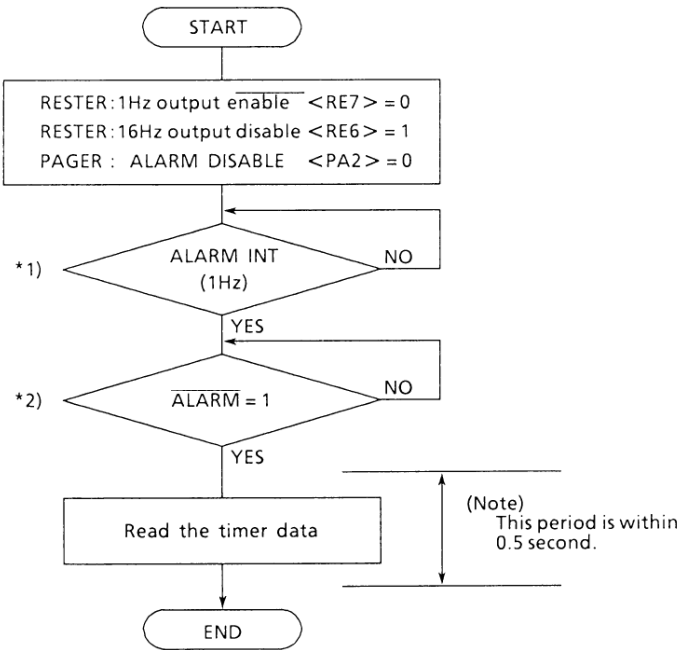
(Note:) This is because an RTC timer carry operation occurs at the 1Hz pulse rise edge when the RTC timer is read after reading ALARM = 1 from a port with an interrupt routine at *2). When reading timer data, the prescribed timer (timer value) can be read by reading for 0.5 sec. after the carry operation.

Figure 3.11 (4)

(2)   When Writing Data

① Data will not be written correctly if a carry signal
appears while writing a series of data. Use the follow-
ing method to write the data correctly.

The RTC contains a 15-stage divider that generates
the 1Hz signal from the 32.768kHz signal. Carry operations
are not performed during the 1-s. interval after this
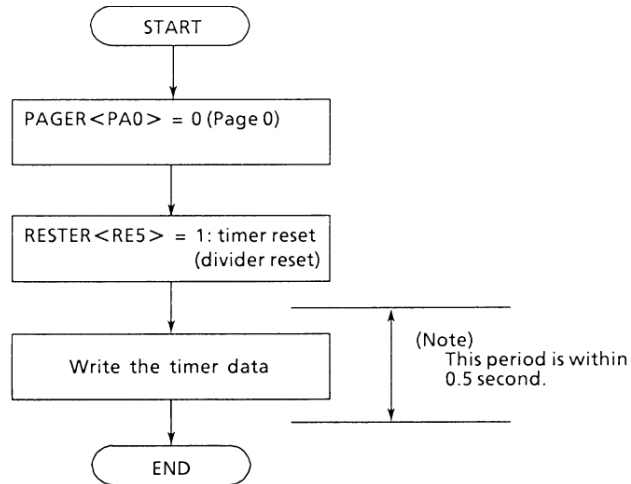divider is reset. Data are written during this interval.

```
         ┌─────────────┐
         │    START    │
         └──────┬──────┘
                │
  ┌─────────────▼─────────────┐
  │ PAGER<PA0> = 0 (Page 0)   │
  └─────────────┬─────────────┘
                │
  ┌─────────────▼─────────────┐
  │ RESTER<RE5> = 1: timer reset
  │              (divider reset) │
  └─────────────┬─────────────┘
                │
  ┌─────────────▼─────────────┐          (Note)
  │   Write  the  timer data  │    This period is within
  └─────────────┬─────────────┘    0.5 second.
                │
         ┌──────▼──────┐
         │     END     │
         └─────────────┘
```

Figure 3.11 (5). Typical Data Write Sequence

② When disabling the timer

Writing "0" to bit <PA3> of address #FFE7H disables the timer, disables carry operations and prevents misoperation due to the CLOCK HOLD circuit.

The CLOCK HOLD circuit holds once only the 1-second carry signal generated by the divider while the timer is disabled, corrects the time by outputting the carry signal to the timer after the timer is enabled, and continues operating. If the timer disable continues for longer than one second, however, the timer is delayed. Caution must be exercised here in that the TMP90C051F system power supply is halted while the timer is disabled. In this case, the timer remains stopped and the time is delayed; therefore, when a system power-down is detected while the timer is disabled, always return to timer enable and then set $\overline{\text{BAKEN}}$ (Back Up Enable) to "0".

```
        ┌───────────┐
        │   START   │
        └─────┬─────┘
              │
     ┌────────▼────────┐
     │ Disable the timer│
     └────────┬────────┘
              │              ──────────────┐
     ┌────────▼────────┐                   │  (Note)
     │ Read the timer data│                │  This period is within
     └────────┬────────┘                   │  one second.
              │              ──────────────┘
     ┌────────▼────────┐
     │  Enable the timer │
     └────────┬────────┘
              │
        ┌─────▼─────┐
        │    END    │
        └───────────┘
```
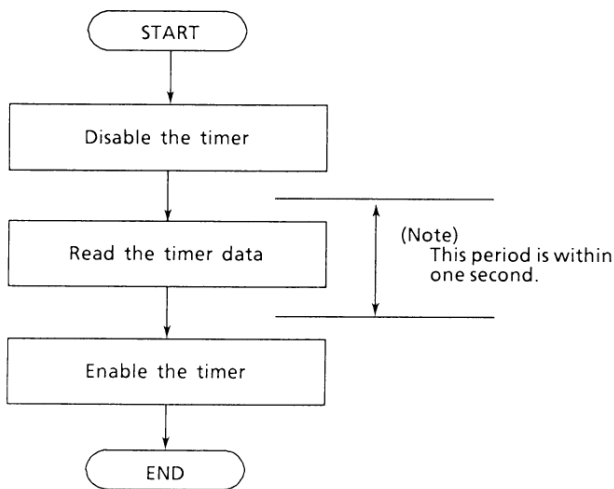
Figure 3.11 (6). Typical Timer Disable Sequence

3.11.7 Battery Backup

The RTC section has an independent (RTC) Vcc power supply pin. Operation can be continued in case of loss of Vcc (pins 36/77), the main power supply, if an external battery backup is connected to (RTC) Vcc.

The battery backup is activated by setting $\overline{\text{BUKEN}}$ = "0" when loss of the main power supply, Vcc (pins 36/77), is detected externally.

A minimum (RTC) Vcc voltage of 2V is guaranteed by the battery backup.

Use a backup circuit that provides the same potential at (RTC) Vcc found at Vcc (pins 36/77) when the main power supply is on.

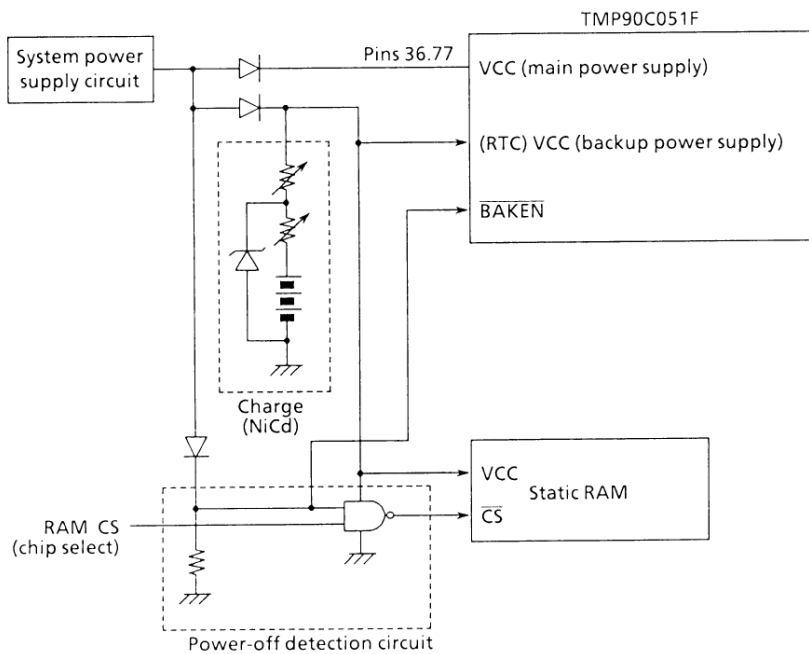Figures 3.11 (7) and (8) below show battery backup circuit configurations.

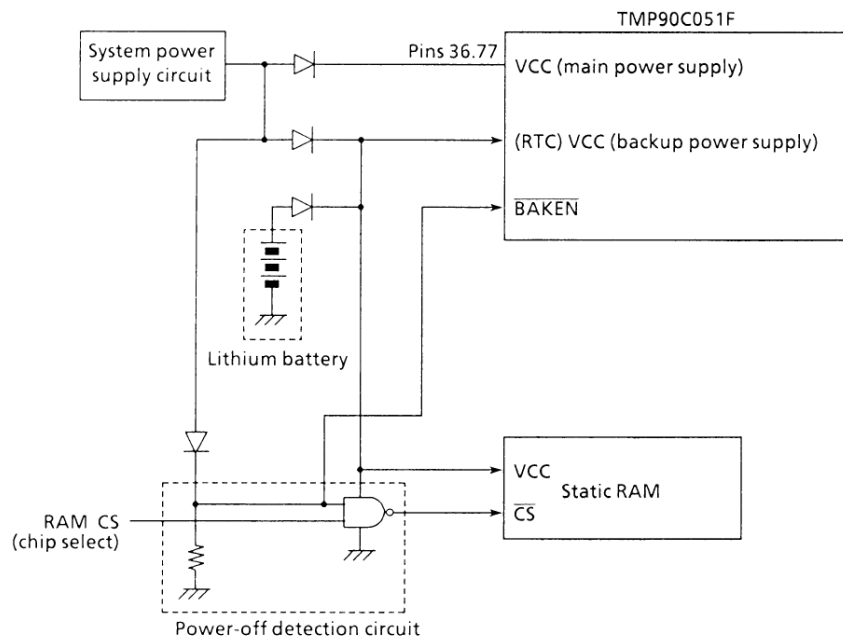**Figure 3.11 (7). NiCad Battery Backup Circuit Configuration**

Figure 3.11 (8). Lithium Battery Backup Circuit Configuration

3.12 Dynamic RAM Controller

The TMP90C051F configuration includes a controller circuit for refreshing the dynamic RAM, an access circuit for reading and writing, and an address decoder circuit.

Figure 3.12 (1) shows the dynamic RAM controller block diagram.
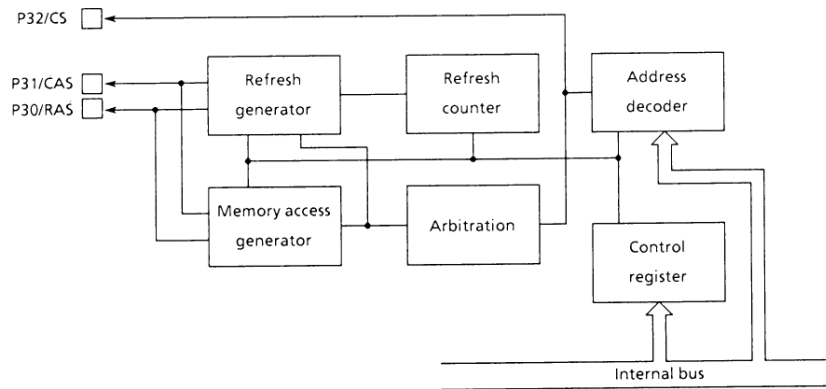

Figure 3.12 (1). Dynamic RAM Controller Block Diagram

3.12.1 Refresh Controller Block

The TMP90C051F can drive RAS and CAS outputs used for DRAM refreshing.

DRAM refresh is easy because the output cycle and output pulse width of the $\overline{RAS}$ and $\overline{CAS}$ outputs can be set by program.

The refresh controller block has the following features.

1) Refresh method: $\overline{CAS}$ before $\overline{RAS}$ interval refresh
2) Refresh interval: 15 ~ 154 states (programmable)
3) Refresh cycle width: 2 ~ 9 states (programmable)
4) Dummy cycles can be generated.

Figure 3.12 (2) shows typical refresh cycle timing.

a) With the refresh cycle width set to 2 states.

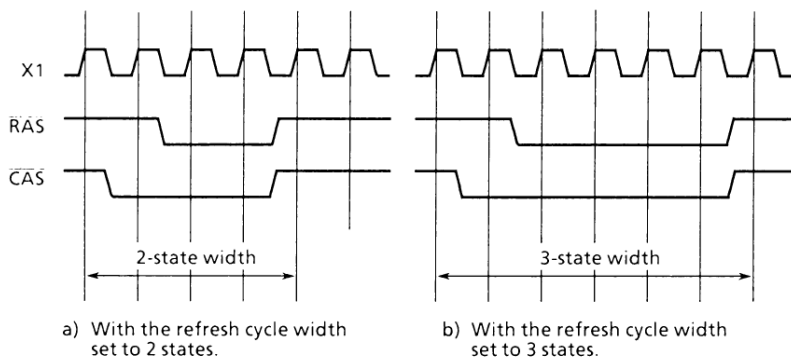b) With the refresh cycle width set to 3 states.

Figure 3.12 (2). Typical Refresh Cycle Timing

The method for using the refresh controller block is described below.

i)   Refresh Method

Refreshing is performed using the $\overline{CAS}$ and $\overline{RAS}$ refresh method using the DRAM built-in refresh counter.

ii)   Register Setting Method

With the $\overline{CAS}$ before $\overline{RAS}$ refresh method, the refresh interval and refresh cycle width differ depending on the DRAM used.
The TMP90C051 sets the $\overline{RAS}$ and $\overline{CAS}$ output in accordance with the system clock and DRAM used by changing the refresh control register value.
Figure 3.12 (3) shows the bit configuration of the refresh control register used to control $\overline{RAS}$ and $\overline{CAS}$ output.
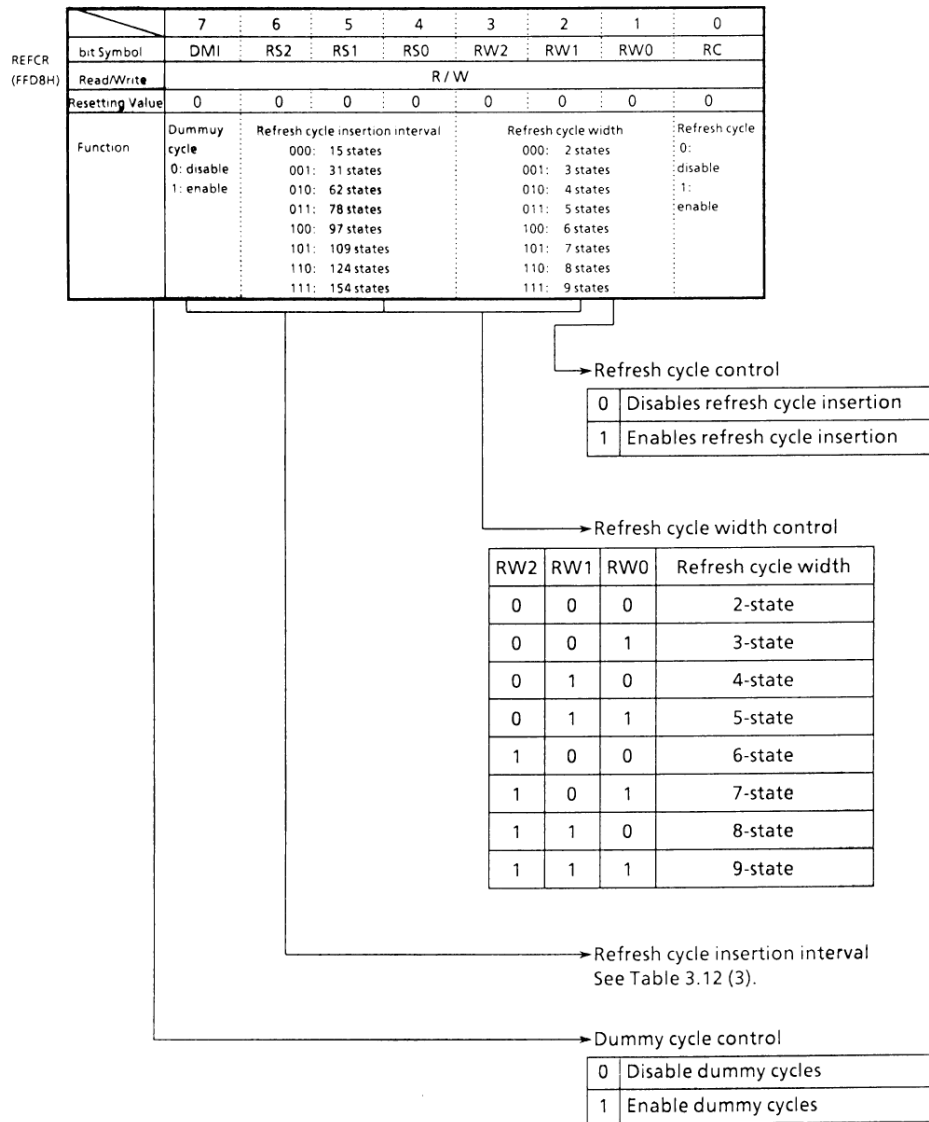
		7	6	5	4	3	2	1	0
REFCR	bit Symbol	DMI	RS2	RS1	RS0	RW2	RW1	RW0	RC
(FFD8H)	Read/Write				R / W				
	Resetting Value	0	0	0	0	0	0	0	0
	Function	Dummuy cycle 0: disable 1: enable	Refresh cycle insertion interval 000: 15 states 001: 31 states 010: 62 states 011: 78 states 100: 97 states 101: 109 states 110: 124 states 111: 154 states			Refresh cycle width 000: 2 states 001: 3 states 010: 4 states 011: 5 states 100: 6 states 101: 7 states 110: 8 states 111: 9 states			Refresh cycle 0: disable 1: enable

Refresh cycle control

0	Disables refresh cycle insertion
1	Enables refresh cycle insertion

Refresh cycle width control

RW2	RW1	RW0	Refresh cycle width
0	0	0	2-state
0	0	1	3-state
0	1	0	4-state
0	1	1	5-state
1	0	0	6-state
1	0	1	7-state
1	1	0	8-state
1	1	1	9-state

Refresh cycle insertion interval
See Table 3.12 (3).

Dummy cycle control

0	Disable dummy cycles
1	Enable dummy cycles

Figure 3.12 (3). Refresh Control Register

Table 3.12 (3) Refresh Cycle Insertion Intervals

Resfresh cycle			Insertion interval (state)	Frequecncy					
RS2	RS2	RS0		4MHz	8MHz	10MHz	12.5MHz	14MHz	16MHz
0	0	0	15	7.5	3.75	3.0	2.4	2.14	1.88
0	0	1	31	15.5	7.55	6.2	4.96	4.43	3.88
0	1	0	62	31.0	15.5	12.4	9.92	8.86	7.75
0	1	1	78	39.0	19.5	15.6	12.48	11.14	9.75
1	0	0	97	48.5	24.25	19.4	15.52	13.86	12.13
1	0	1	109	54.5	27.25	21.8	17.44	15.57	13.63
1	1	0	124	62.0	31.0	24.8	19.84	17.72	15.5
15.5	1	1	154	77.0	38.5	30.8	24.7	22.0	19.3

(Unit μs)

The refresh control register (REFCR) is assigned to address FFD8H and the bits are as follows.

<DM1>: Dummy cycle control bit (DMI)
The dummy cycles required fro DRAM intialization are genrated by setting this bit to "1" (see Figure 3.12 (4)).

<RS2 ~ 0>: Refresh cycle insertion interval control bits (RS2 ~ RS0)
These 3 bits are used to change the system clock used and the insertion interval.

Example: When using the 12.5MHz system clock, set these bits to "100" to specify a DRAM refresh cycle of 16 microseconds.

<RW2 ~ 0>: Refresh cycle width control bits (RW2 ~ RW0)
These 3 bits are used to change the refresh cycle width (RAS, CAS output) (2 ~ 9 states).

<RC>: Refresh cycle control bit (RC)
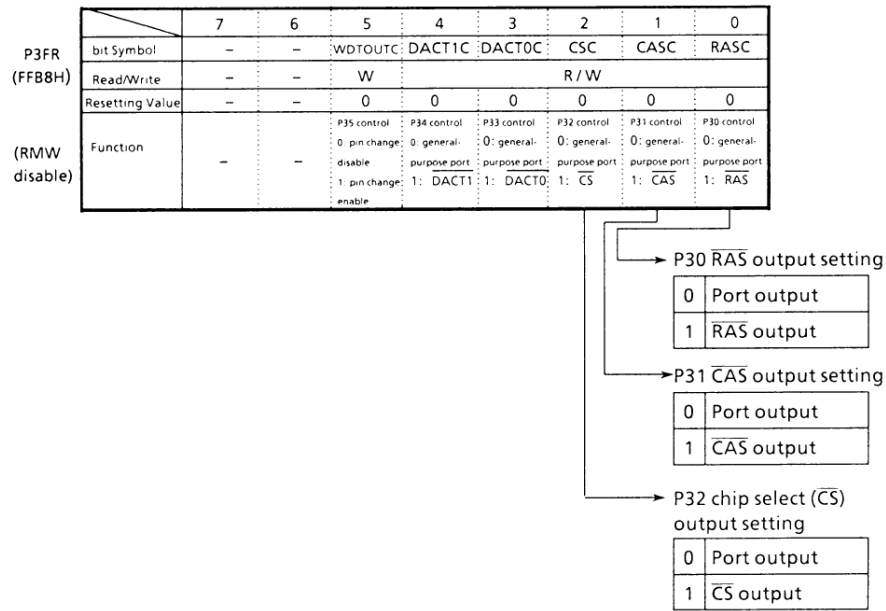This bit is used to control refresh cycle insertion.

P3FR (FFB8H)		7	6	5	4	3	2	1	0
	bit Symbol	–	–	WDTOUTC	DACT1C	DACT0C	CSC	CASC	RASC
	Read/Write	–	–	W			R / W		
	Resetting Value	–	–	0	0	0	0	0	0
(RMW disable)	Function	–	–	P35 control 0: pin change disable 1: pin change enable	P34 control 0: general-purpose port 1: DACT1	P33 control 0: general-purpose port 1: DACT0	P32 control 0: general-purpose port 1: $\overline{CS}$	P31 control 0: general-purpose port 1: $\overline{CAS}$	P30 control 0: general-purpose port 1: $\overline{RAS}$

P30 $\overline{RAS}$ output setting

0	Port output
1	$\overline{RAS}$ output

P31 $\overline{CAS}$ output setting

0	Port output
1	$\overline{CAS}$ output

P32 chip select ($\overline{CS}$) output setting

0	Port output
1	$\overline{CS}$ output

Figure 3.12 (4). Port 3 Function Registers

iii) Dynamic RAM Initialize

The dynamic RAM controller generates consecutive $\overline{CAS}$ before $\overline{RAS}$ dummy cycles when using DRAM. Figure 3.12 (4) shows the $\overline{CAS}$ before $\overline{RAS}$ dummy cycle timing.
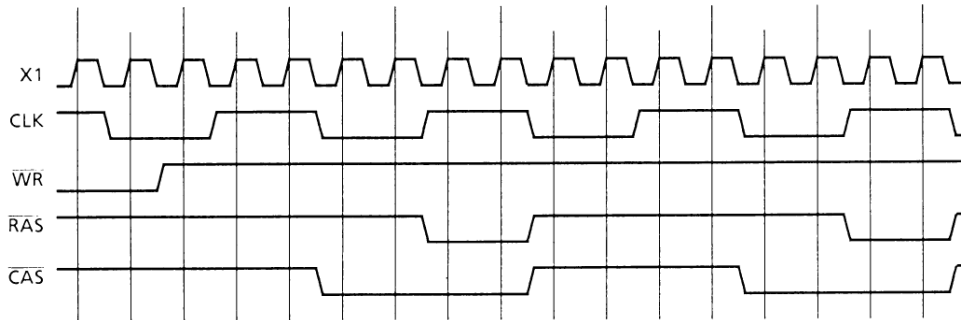
Figure 3.12 (5) $\overline{CAS}$ Before $\overline{RAS}$ Dummy Cycle Timing

3.12.2 Read/Write Control Block

The read/write control block outputs the memory access cycle $\overline{RAS}$ and $\overline{CAS}$ signals when the address area set with the decoder block described in the next section is selected.

Figure 3.12 (6) shows the $\overline{RAS}$ and $\overline{CAS}$ output timing during the memory access cycle.

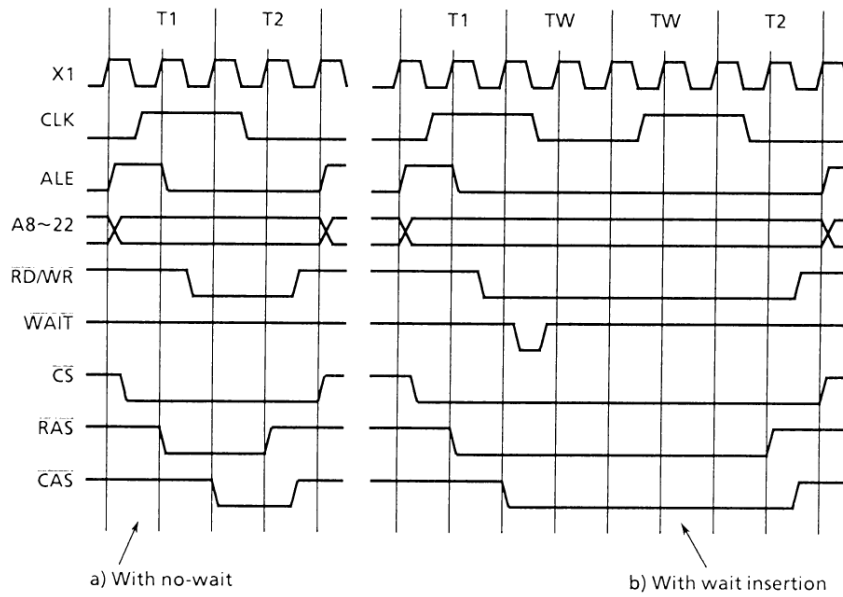**Figure 3.12 (6). Memory Access Cycle Timing**

i)   Register Setting Method

Figure 3.4 (7) shows the bit configuration of the memory access control register. The memory access control register (MACR) is assigned to address FFDBH.
CS output is enabled and disabled by setting

<CSEN>. Additionally, RAS and CAS output during memory access is enabled and disabled by setting <MC>. When <CSEN> is set to "0" (CS output disable), RAS and CAS output during memory access is disabled, regardless of <MC> value.
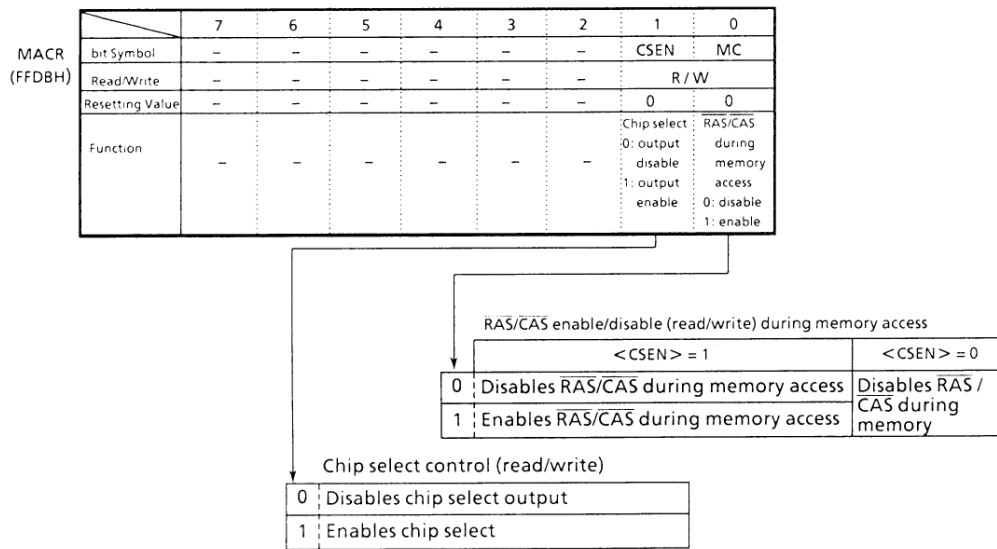
Figure 3.12 (7). Memory Access Control Register (MACR)

3.12.3 Address Decoder Block

The address decoder block enables DRAM access and simultaneously outputs the $\overline{CS}$ signal when the start address and mask address are loaded to the register. When a DRAM is not used, the $\overline{CS}$ signal can be used as the area select signal.
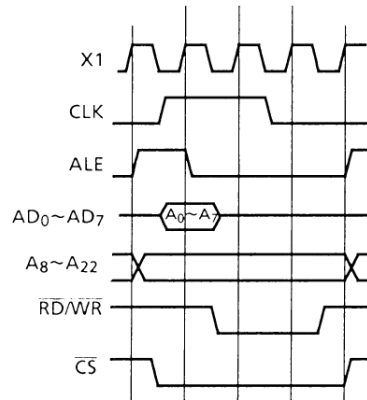

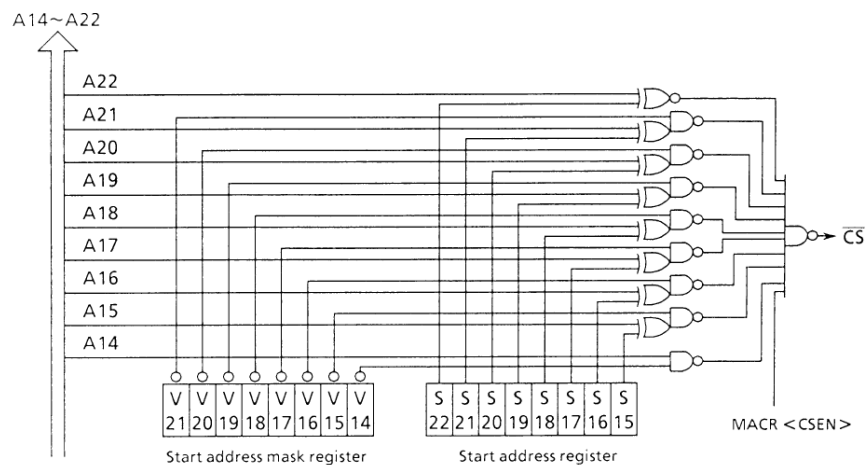**Figure 3.12 (8). Chip Select ($\overline{CS}$) Operation Timing**

Figure 3.12 (9). Address Decoder Block Diagram

ii)   Start Address Setting Method

The address decoder outputs the $\overline{CS}$ signal when the start address and area size are specified.
As shown in the diagram, the start address is decoded

by A15 - A22 and is therefore set every 32K bytes.
In other words, the DRAM start address is set every 32K bytes, starting at 000000H.

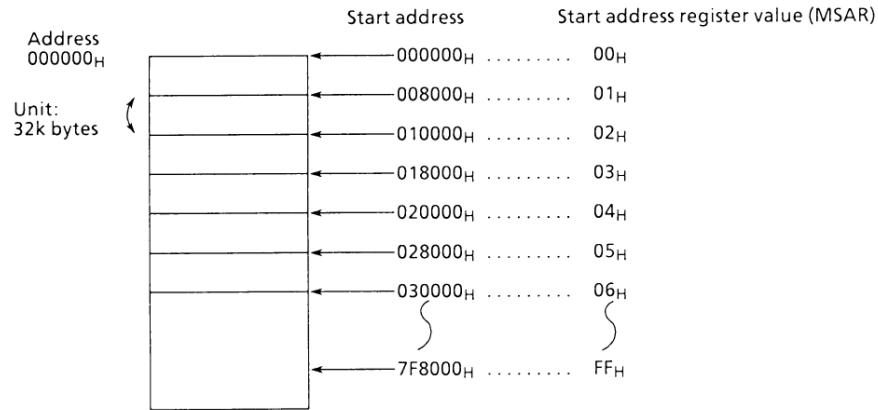<Note> The start address is changed to set the value of MSAMR register.

Figure 3.12 (10). Start Address Setting Locations

iii)   Address Area Setting Method

The address area is specified by setting the start address mask register (MSAMR).
As can be seen from the address decoder block diagram (Fig. 3.12 (9), the address area that can output CS is specified depending on whether or not the A14 - A21 address values are compared.
In other words, it is possible to specify an area of up to 4M bytes in 16K byte units.

iii)   Procedures for Setting Start Address/Address Area

(1)   Set the memory start address mask register (MSAMR) (Setting the address area)

(2)   Set the memory start address mask register (MSAR) (Setting the area start address)

(3)   Compare values of the same address bit of MSAMR and MSAR. (Ex.) Compare values of MSAMR <V15> and MSAR <S15>.

(4)   If the values of the same address bit are "1", the bit in MSAR becomes ineffective and is regarded as "0".
The start address is changed.
If the values of the same address bit are not matched by "1", the set-up is complete. (The specified address area and start address are decoded.)

(5)   When the start address is changed correctly, the set-up is complete. If not so, change the MSAR value.

(6)   Set MSAR again and compare. (Start from (3).)

Set-up example

When address area = 64K bytes, start address = 18000$_H$
(Memory map: 18000$_H$ to 27FFF$_H$)
Set
MSAMR = 03$_H$ <-> address area: 64K bytes
MSAR = 03$_H$ <-> start address: 18000$_H$
*: When values of the same address bit are "1", the start
address is changed to 10000$_H$.
If not so, change the start address to 20000$_H$.
(Memory map: 20000$_H$ to 2FFFF$_H$)
MSAMR =03$_H$
MSAR = 04$_H$

When the start address is 20000$_H$, the address bits do
not match as "1" and the start address is not changed.
This enables to decode under the start address = 20000$_H$
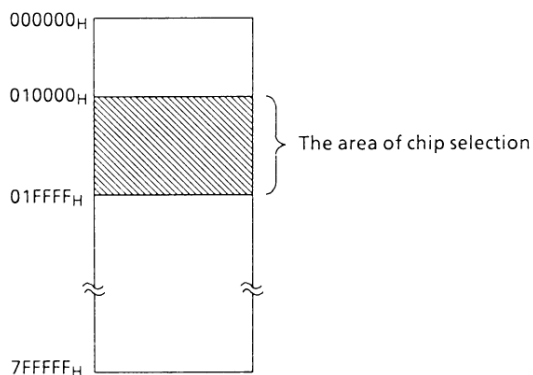and the address area = 64K bytes.

(Set-up example 1)

Case of MSAR = 03$_H$ and MSAMR = 03$_H$

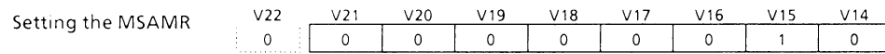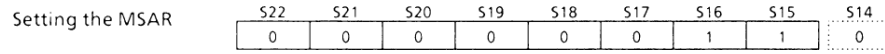The output of chip selection is the memory map as below.

	S22	S21	S20	S19	S18	S17	S16	S15	S14
Setting the MSAR	0	0	0	0	0	0	1	1	0
									default "0"

	V22	V21	V20	V19	V18	V17	V16	V15	V14
Setting the MSAMR	0	0	0	0	0	0	0	1	1
	default "0"								

V22 ~ V16 = "0", so these values of S22 ~ S16 are valid-
ness but V15 and V14 = "1", so these values of S15 and S14
are invalidity.

000000$_H$
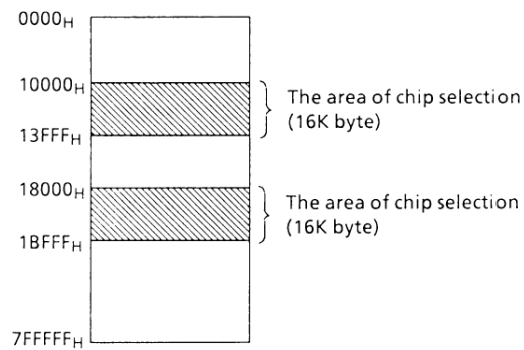
010000$_H$

The area of chip selection

01FFFF$_H$

7FFFFF$_H$

(Set-up example 2)   Case of MSAR = 03H and MSAMR = 02H

Setting the MSAR

S22	S21	S20	S19	S18	S17	S16	S15	S14
0	0	0	0	0	0	1	1	0

Setting the MSAMR

V22	V21	V20	V19	V18	V17	V16	V15	V14
0	0	0	0	0	0	0	1	0

V22 ~ V16, V14 = "0", so these value of S22 ~ S16 and   S14 are validness but S15 = "1", so this value of S15 is invalidity.

Address	Area
0000_H	
10000_H	The area of chip selection (16K byte)
$13FFF_H$	
18000_H	The area of chip selection (16K byte)
$1BFFF_H$	
$7FFFFF_H$	

(Set-up example 3)   MSAMR (partly)

Chip select area by setting value of MSAR and

MSAMR / MSAR	00	01	02	03
00	0000 ∫ 3FFF (16K byte)	0000 ∫ 7FFF (32K byte)	0000 8000 ∫ ∫ 3FFF 1BFFF (16K byte x 2)	0000 ∫ FFFF (64K byte)
01	8000 ∫ BFFF (16K byte)	8000 ∫ FFFF (32K byte)	10000 18000 ∫ ∫ 3FFF 1BFFF (16K byte x 2)	0000 ∫ FFFF (64K byte)
02	10000 ∫ 13FFF (16K byte)	10000 ∫ 17FFF (32K byte)	10000 18000 ∫ ∫ 13FFF 1BFFF (16K byte x 2)	10000 ∫ 1FFFF (64K byte)
03	18000 ∫ 1BFFF (16K byte)	18000 ∫ 1FFFF (32K byte)	10000 18000 ∫ ∫ 13FFF 1BFFF (16K byte x 2)	10000 ∫ 1FFFF (64K byte)

( ) is area size

MSAMR / MSAR	03	07	0F	1F	3F	7F	FF
04	20000 ∫ 2FFFF (64K byte)	20000 ∫ 3FFFF (128K byte)	00000 ∫ 3FFFF (256K byte)	00000 ∫ 7FFFF (512K byte)	00000 ∫ FFFFF (1M byte)	00000 ∫ 1FFFFF (2M byte)	00000 ∫ 3FFFFF (4M byte)
08	40000 ∫ 4FFFF (64K byte)	40000 ∫ 5FFFF (128K byte)	40000 ∫ 7FFFF (256K byte)				
10	80000 ∫ 8FFFF (64K byte)	80000 ∫ 9FFFF (128K byte)	80000 ∫ BFFFF (256K byte)	80000 ∫ FFFF (512K byte)			
20	100000 ∫ 10FFFF (64K byte)	100000 ∫ 11FFFF (128K byte)	100000 ∫ 13FFFF (256K byte)	100000 ∫ 17FFFF (512K byte)	100000 ∫ 1FFFFF (1M byte)		
40	200000 ∫ 20FFFF (64K byte)	200000 ∫ 21FFFF (128K byte)	200000 ∫ 23FFFF (256K byte)	200000 ∫ 27FFFF (512K byte)	200000 ∫ 2FFFF (1M byte)	200000 ∫ 3FFFFF (2M byte)	
80	400000 ∫ 40FFFF (64K byte)	400000 ∫ 41FFFF (128K byte)	400000 ∫ 43FFFF (256K byte)	400000 ∫ 47FFFF (512K byte)	400000 ∫ 4FFFF (1M byte)	400000 ∫ 5FFFFF (2M byte)	400000 ∫ 7FFFFF (4M byte)

Iv)   Register Setting Method

Figure 3.12 (11) and Figure 3.12 (12) show the bit con-
figurations of the start address register (MSAR) and
start address mask register (MSAMR).
MSAR is assigned to address FFD9H and MSAMR is

assigned to address FFDAH.
CS output can be enabled and disabled regardless of
the above registers by setting <CSEN> of the memory
access control register (MACR) shown in Figure 3.12
(6).

MSAR (FFD9H)		7	6	5	4	3	2	1	0
	bit Symbol	S22	S21	S20	S19	S18	S17	S16	S15
	Read/Write	R / W							
	Resetting Value	1	1	1	1	1	1	1	1
	Function	Set memory starty address A22~A15							

Memory start address setting

Figure 3.12 (10)   Memory start address register

MSAMR (FFDAH)		7	6	5	4	3	2	1	0
	bit Symbol	V21	V20	V19	V18	V17	V16	V15	V14
	Read/Write	R / W							
	Resetting Value	1	1	1	1	1	1	1	1
	Function	0: address A14~A21 comparision valid 1: address A14~A21 comparison invalid (Specify in 1 bit units.)							

Address A14~A21 comparison control

0	A14~A21 comparison enabled
1	A14~A21 comparison disabled

Figure 3.12 (11). Memory Start Address Mask Register

The MSAR <S22> ~ <S15> correspond to the A22 ~
A15 of address and the S14 correspond to the A14
(S14 is "0" at default). The MSAMR <V21> ~ <V14>
point to the validness/invalidity of comparison with the
value of the setting MSAR and the address.
The MSAMR <V21> ~ <V14> correspond to the
MSAMR <V21> ~ <V15>, S14 and the MSAMR
<V22> corresponds to the MSAR <S22> (S22 is "0" at
default). The comparison of the A22 and <S22> is
usually validness. The meaning of comparison (valid-
ness and invalidity).

EX.   The case of invalidity at <V15> = 1

The comparison of <S15> value and address (A15) is
invalidity and <S15> value is validity. So the value of
A15 is unrelated ness for "0"/"1".
The case of validness at <V15> = 0

The comparison of <S15> value and address (A15) is
validness.
When the value of <S15> only agree with address
(A15), CS signal is enabled.

3.12.4 Priority Sequence

Since the dynamic RAM refresh cycle is asynchronous with the CPU operation cycle, it may overlap the read/write cycle in some cases. In such cases, the dynamic RAM controller gives priority to the cycle that enters first. When the refresh cycle is given priority, a waits automatically inserted in the memory access cycle. Figure 3.12 (13) shows the timing in such cases.
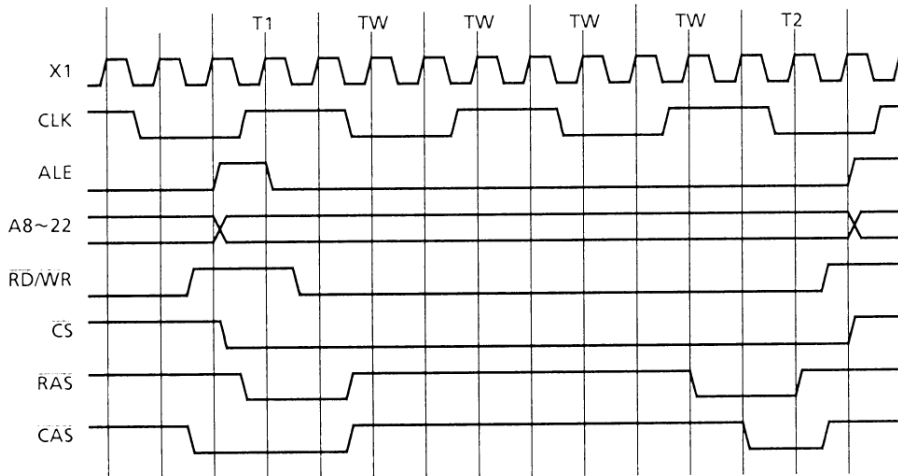
.

Figure 3.12 (13). Timing Chart for When a Memory Access Cycle Enters During a Refresh Cycle
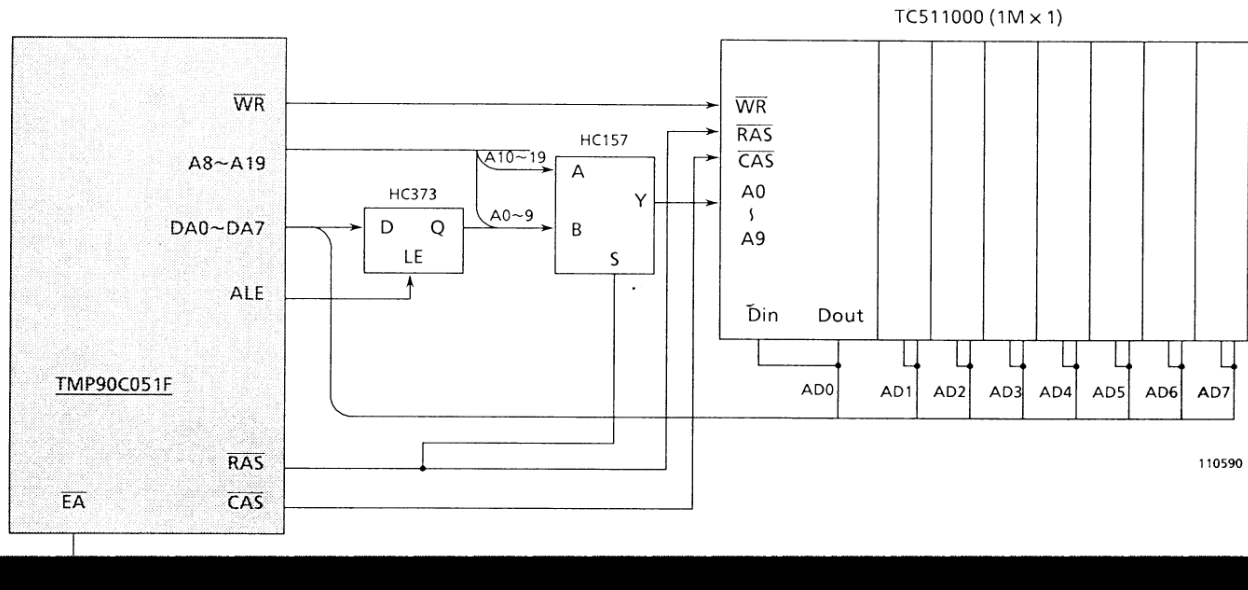
3.12.5 Typical DRAM Connection

Figure 3.12 (14). Typical DRAM Connection