T-49-19-08

CMOS 8-BIT MICRO CONTROLLERS

TMP90C845N/TMP90C845F

1. OUTLINE AND CHARACTERISTICS

TMP90C845 is a high speed, high performance 8-bit microcontroller developed for application in the control of various devices.

TMP90C845, CMOS 8-bit microcontroller, integrates an 8-bit CPU, RAM, A/D converter, multi-function timer/event counter, general-purpose serial interface, and programmable chip selector features in a single chip. In addition, it can expand to 4M byte external program memory as well as 8M byte external data memory.

TMP90C845N is a device with a 64 pin shrink DIP.

TMP90C845F is a device with a 64 pin flat package.

The following are the features of TMP90C845:

(1) Highly efficient instructions
163 types of basic instructions, including
Multiplication, division, 16-bit arithmetic operations, bit manipulation instructions

(2) Minimum instruction executing time: 250 ns (at 16 MHz oscillation frequency)

(3) Built-in RAM: 256 bytes

(4) Memory expansion
External program memory : 4M bytes
External data memory : 8M bytes

(5) Highly accurate 8-bit A/D converter (4 channels)

(6) General-purpose serial interface (1 channel)
With asynchronous mode and I/O interface mode

(7) Multi-function 16-bit timer/event counter (1 channel)

(8) 8-bit timer (4 channels)

(9) Stepping motor control and pattern generation ports (2 channels)

(10) Input/Output ports: 36 pins

(11) Programmable chip select function

(12) Interrupt function: 10 internal, 3 external

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

(14) Watchdog timer function

(15) Standby function (3 HALT modes)

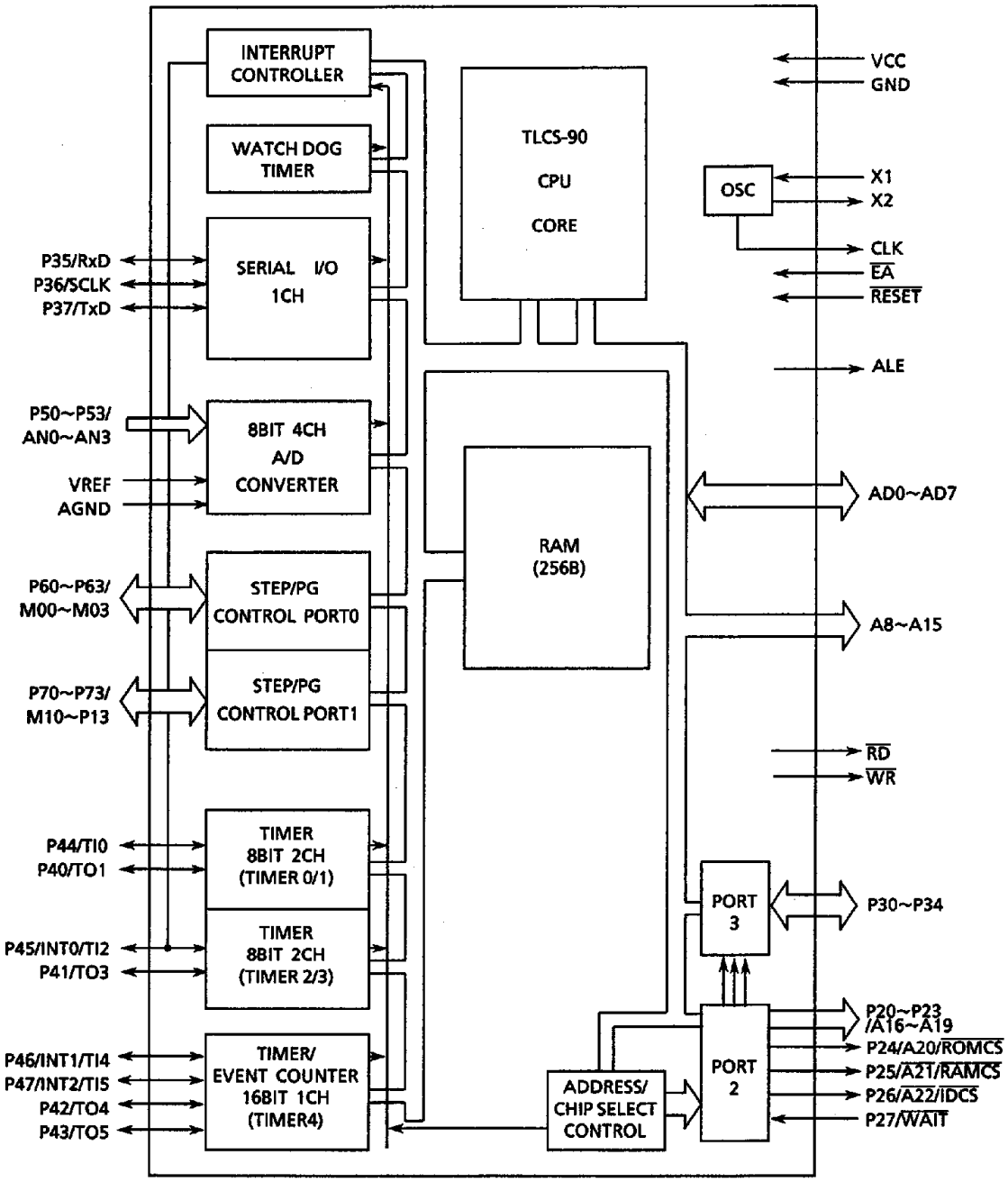


Figure 1 TMP90C845 Block Diagram

2. PIN ASSIGNMENT AND FUNCTIONS

The assignment of input/output pins for TMP90C845, their names and outline functions are described below.

2.1 Pin Assignment

Figure 2.1 (1) shows pin assignment of the TMP90C845N.

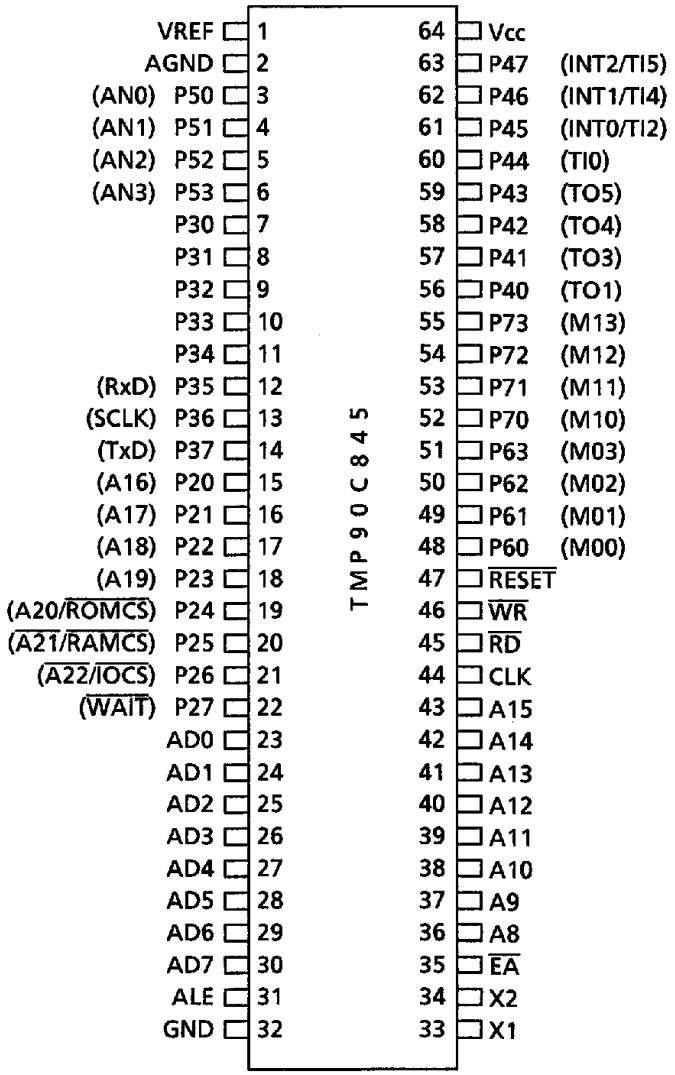


Figure 2.1 (1) Pin Assignment (Shrink DIP)

Figure 2.1 (2) shows the pin assignment of TMP90C845F.

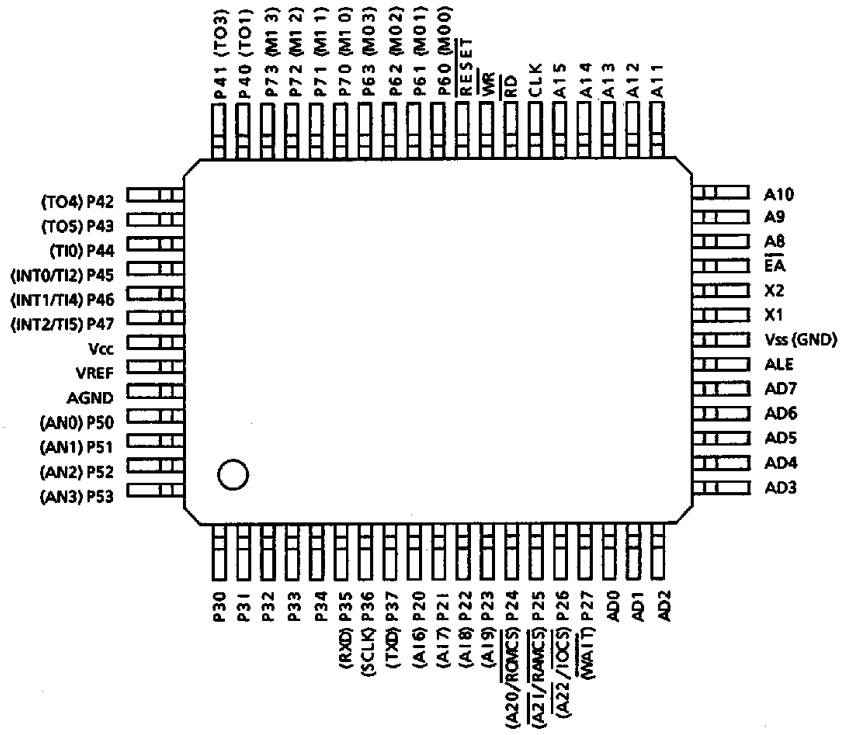


Figure 2.1 (2) Pin Assignment (Flat Package)

2.2 Pin Names and Functions

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

Table 2.2 (1/2)

Pin name	No. of pins	I/O or tristate	Function
AD0~AD7	8	Tristate	Address/data bus: 8-bit time sharing bus which transmits address (lower 8 bits) and data
A8~A15	8	Output	Address bus: The upper 8 bits address bus
P20~P26	7	Output	Ports 20~26: 7-bit output port
/A16~A20	(5)	/Output	Addresses 16~20: Address bus which is used to expand the program and data areas
/$\overline{A21}$~$\overline{A22}$	(2)	/Output	Addresses 21 and 22: Address bus which is used to extend the program and data areas (inverted output)
/$\overline{ROMCS}$, $\overline{RAMCS}$, $\overline{IOCS}$	(3)	/Output	Programmable chip select
P27	1	Input	Port 27: 1-bit input port
/$\overline{WAIT}$		/Input	Wait: Input pin for connecting a memory or peripheral LSI with delayed access time
P30~P37	8	I/O	Port 3: 8-bit I/O port which allows I/O selection on bit basis (with programmable pull-up resistor)
/RxD	(1)	/Input	Receiver of serial data
/SCLK	(1)	I/O	Serial clock
/TxD	(1)	/Output	Transmitter of serial data
P40~P47	8	I/O	Port 4: 8-bit I/O port which allows I/O selection on bit basis (with programmable pull-up resistor)
/TO1, 3, 4, 5	(4)	/Output	Timer outputs 1,3,4, and 5: Output ports for timer 0, and timer 1, timer 2 or timer 3, and timer 4 (two lines)
/TI0, 2, 4, 5	(4)	/Input	Timer inputs 0, 2, 4, and 5: Input ports for timer 0, timer 2 and timer 4 (2 lines)
/INT0	(1)	/Input	Interrupt request terminal 0: Interrupt request pin 0: Level/rise edge programmable interrupt request pin
/INT1	(1)	/Input	Interrupt request terminal 1: Interrupt request pin 1: Rise/fall edge programmable interrupt request pin
/INT2	(1)	/Input	Interrupt request terminal 2: Interrupt request pin 2: Rise edge interrupt request pin

Table 2.2 (2/2)

Pin name	No. of pins	I/O or tristate	Function
P50~P53	4	Input	Port 5: 4-bit input port
/AN0~AN3			Analog input: 4 analog inputs to A/D converter
P60~P63	4	I/O	Port 6: 4-bit I/O port which allows I/O selection on bit basis
/M00~M03		/Output	Stepping motor control port 0 or pattern generation port 0
P70~P73	4	I/O	Port 7: 4-bit I/O port which allows I/O selection on bit basis
/M10~M13		/Output	Stepping motor control port 1 or pattern generation port 1
$\overline{RD}$	1	Output	Read: Strobe signal output for reading external memory
$\overline{WR}$	1	Output	Write: Strobe signal output for writing an external memory
ALE	1	Output	Address latch enable
CLK	1	Output	Clock output: Generates clock pulse at 1/4 frequency of clock oscillation. It is pulled up during resetting.
$\overline{EA}$	1	Input	External access: Connects to GND pin in the TMP90C845 without built-in ROM.
$\overline{RESET}$	1	Input	Reset: Initializes TMP90C845. (pull-up resistance is built-in)
X1, X2	2	I/O	Crystal oscillator connection pin
VREF	1		Input of reference voltage to A/D converter
AGND	1		GND pin for A/D converter
Vcc	1		Power supply pin (+5V +/− 10%)
GND	1		GND pin (0V)

3. OPERATION

This section explains the functions and basic operations of the TMP90C845 in blocks.

3.1 CPU

TMP90C845 has a built-in, high performance 8-bit CPU.

For the operation of the CPU, see "TLCS-90 CPU" described in the previous section.

This section explains the CPU functions unique to the TMP90C845 that are not explained in "TLCS-90 CPU".

3.1.1 Reset

Figure 3.1 (1) shows the basic timing of reset.

To reset TMP90C845, it is required that power supply voltage is within operating range, the internal oscillator is stably functioning, and $\overline{\text{RESET}}$ input be kept at "0" at least 10 system clocks (10 states: 2 microseconds with 10 MHz system clock).

When reset is accepted, among I/O common ports, port 6 and port 7 are set to the input status (with high impedance), while port 3 and port 4 become input ports with pull-up resistor. Dedicated output ports P20 to P24 are cleared to "0" while P25 and P26 are set to "1". Besides, CLK is set to "1" and ALE is cleared to "0".

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

When the reset is released, instruction execution starts from address 0000H.

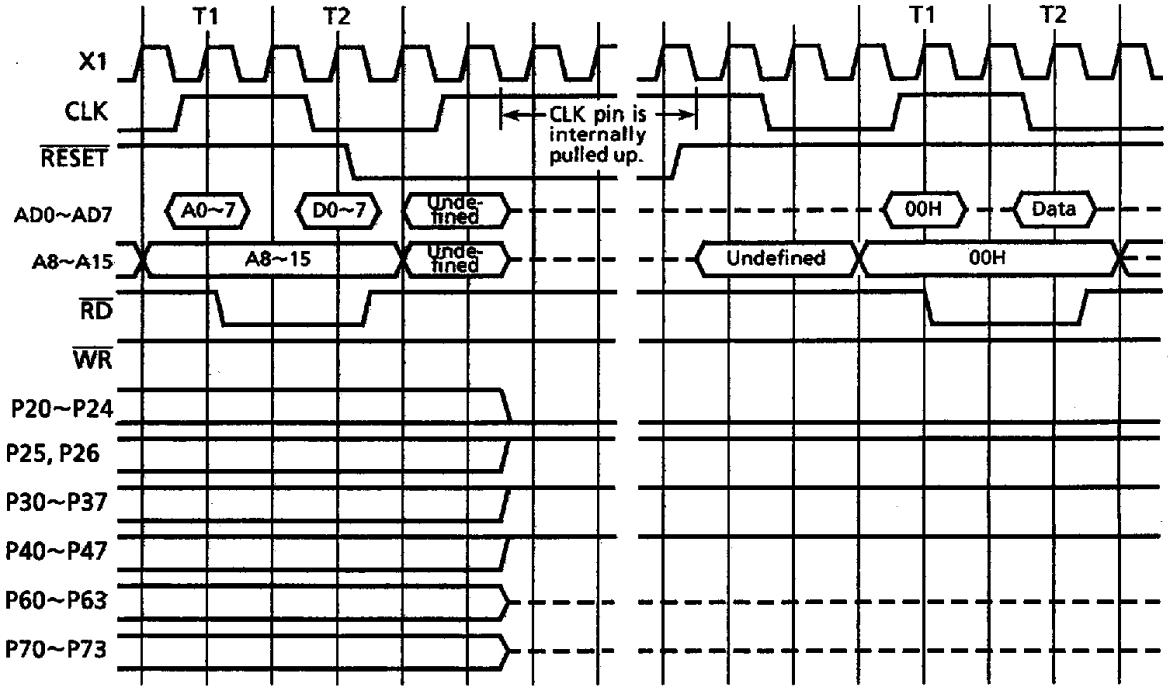


Figure 3.1 (1) Reset Timing of TMP90C845

3.1.2 EXF (Exchange flag)

The exchange flag EXF is inverted when the EXX instruction is executed to exchange data between the TMP90C845 main registers and auxiliary registers. This flag is assigned to bit 1 at memory address FFD2H.

WDMOD (FFD2H)

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

3.1.3 Wait Control

For TMP90C845, wait control register (WAITC) is assigned to bits 0 and 1 at memory address FFDH1.

P25FR (FFD1H)

	7	6	5	4	3	2	1	0
bit Symbol	–	–	–	–	–	RDE	WAITC1	WAITC0
Read/Write							R/W	
After reset						0	0	0
Function						RD control 1: Always RD output	Wait control 00: 2 state wait 01: Normal Wait 10: Non wait 11: –	

3.1.4 Bank register

For TMP90C845, BX and BY registers are allocated to memory addresses FFECH (BX register) and FFEDH (BY register), respectively. For these registers, only the lower 7 bits are valid and the upper 1 bit is undefined. This undefined bit will be "1" whenever it is read.

BX (FFECH)

	7	6	5	4	3	2	1	0
bit Symbol	–	BX6	BX5	BX4	BX3	BX2	BX1	BX0
Read/Write	R/W							
After reset		0	0	0	0	0	0	0

BY (FFEDH)

	7	6	5	4	3	2	1	0
bit Symbol	–	BY6	BY5	BY4	BY3	BY2	BY1	BY0
Read/Write	R/W							
After reset		0	0	0	0	0	0	0

3.2 Memory Map

TMP90C845 can provide a maximum 4M byte program memory and maximum 8M byte data memory.

The program memory may be allocated to the addresses 000000H~3FFFFFH, while the data memory may be allocated to any address 000000H~7FFFFFH.

(1) Built-in RAM

TMP90C845 contains a 256-byte built-in RAM which is allocated to the addresses FEC0H~FFBFH. The CPU can also access some portions of the RAM (192 byte area FF00H~FFBFH) using short instruction codes in the direct addressing mode.

Addresses of FF18H~FF68H this RAM area are used as the parameter area for micro DMA processing. (This area can freely be used when micro DMA function is not used.)

(2) Built-in I/O

TMP90C845 uses 56 bytes of the address space as a built-in I/O area. The area is allocated to the addresses FFC0H~FFF7H. The CPU can access the built-in I/O using short instruction codes in the direct addressing mode.

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

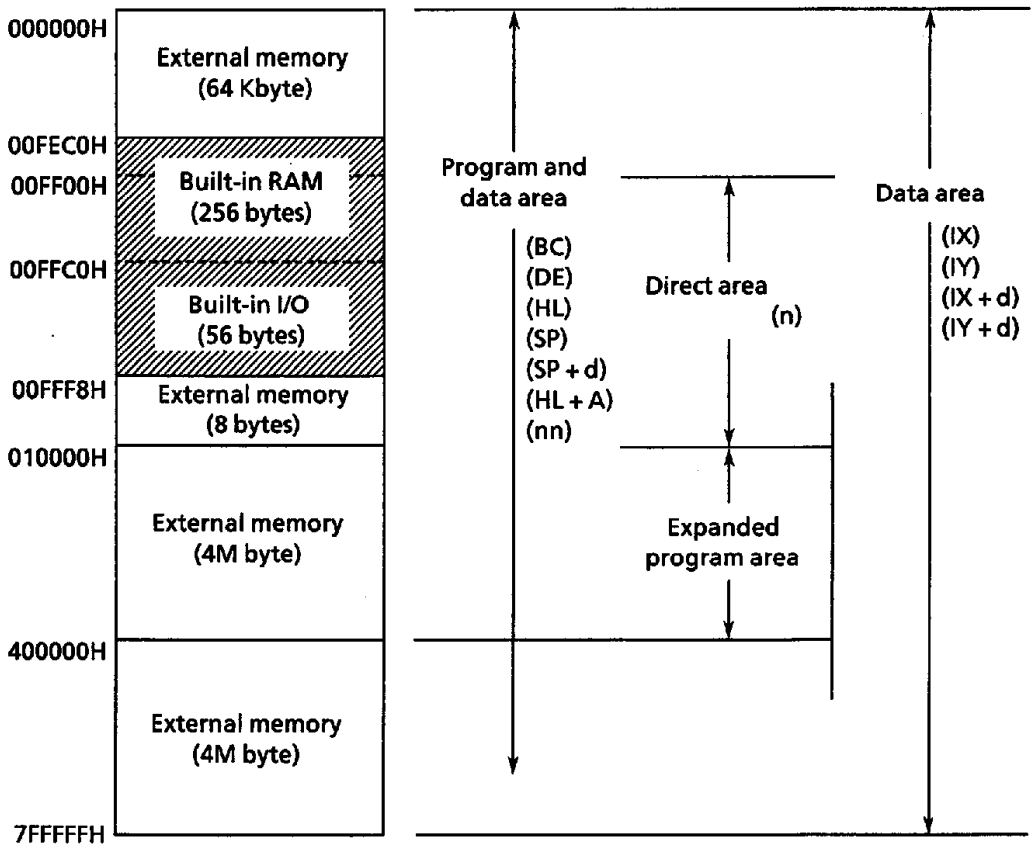


Figure 3.2 Memory Map

3.3 Interrupt Functions

TMP90C845 provides the two processing modes for internal and external interrupt requests; a general-purpose interrupt processing mode and a micro DMA processing mode in which the CPU can automatically transfer data.

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

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

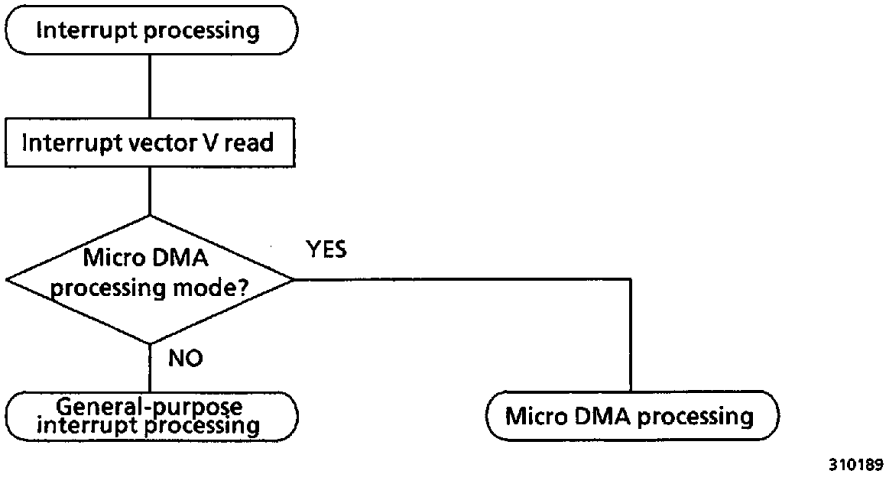


Figure 3.3 (1) Interrupt Response Flowchart

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

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

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

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

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

3.3.1 General purpose Interrupt Processing

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

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

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

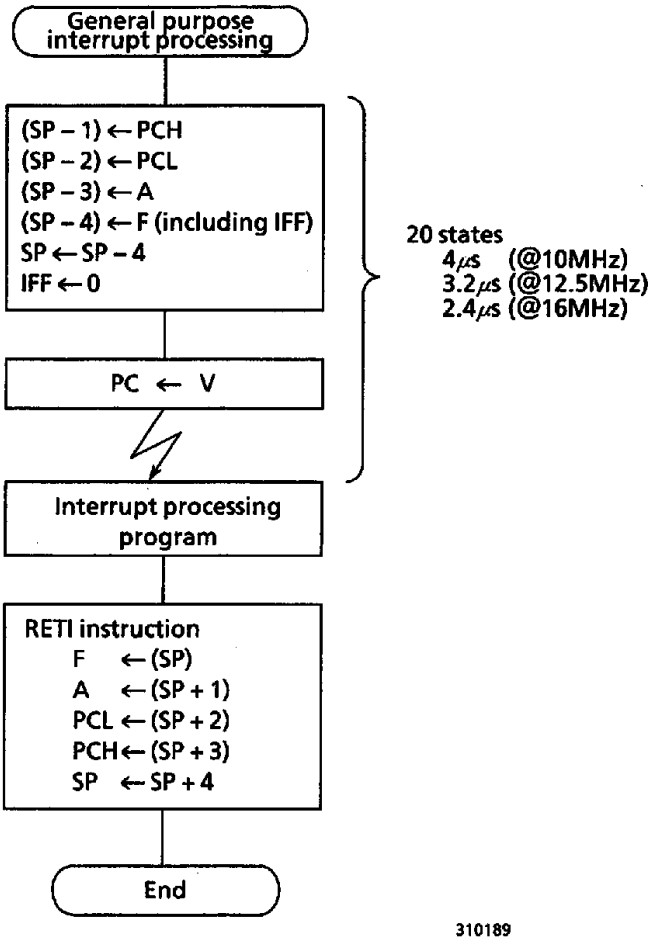


Figure 3.3 (2) General Purpose Interrupt Processing

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

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

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

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

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

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

Table 3.3 (1) shows the interrupt sources.

Table 3.3 (1) Interrupt Sources

Priority order	Type	Interrupt source	Vector value/8	Vector value	Start address of general-purpose interrupt processing	Start address of micro DMA processing parameter
1	Non-	SWI instruction		08H	0008H	—
2	maskable	INTWD (watchdog)		10H	0010H	—
3		INT0 (External input 0)	03H	18H	0018H	FF18H
4		INTT0 (Timer 0)	04H	20H	0020H	FF20H
5		INTT1 (Timer 1)	05H	28H	0028H	FF28H
6		INTT2 (Timer 2)	06H	30H	0030H	FF30H
6		INTAD (A/D conversion end)	06H	30H	0030H	FF30H
7	Maskable	INTT3 (Timer 3)	07H	38H	0038H	FF38H
8		INTT4 (Timer 4)	08H	40H	0040H	FF40H
9		INT1 (External input 1)	09H	48H	0048H	FF48H
10		INTT5 (Timer 5)	0AH	50H	0050H	FF50H
11		INT2 (External input 2)	0BH	58H	0058H	FF58H
12		INTRX (Serial receiving end)	0CH	60H	0060H	FF60H
13		INTTX (Serial transmission end)	0DH	68H	0068H	FF68H

Note: Either INTT2 or INTAD interrupt is selected by the A/D interrupt selection register INTEH<ADIS>.

The "priority order" in the table 3.3 (1) indicates the order of the interrupt sources to be acknowledged by the CPU when two or more interrupts are requested at one time.

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

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

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

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

3.3.2 Micro DMA Processing

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

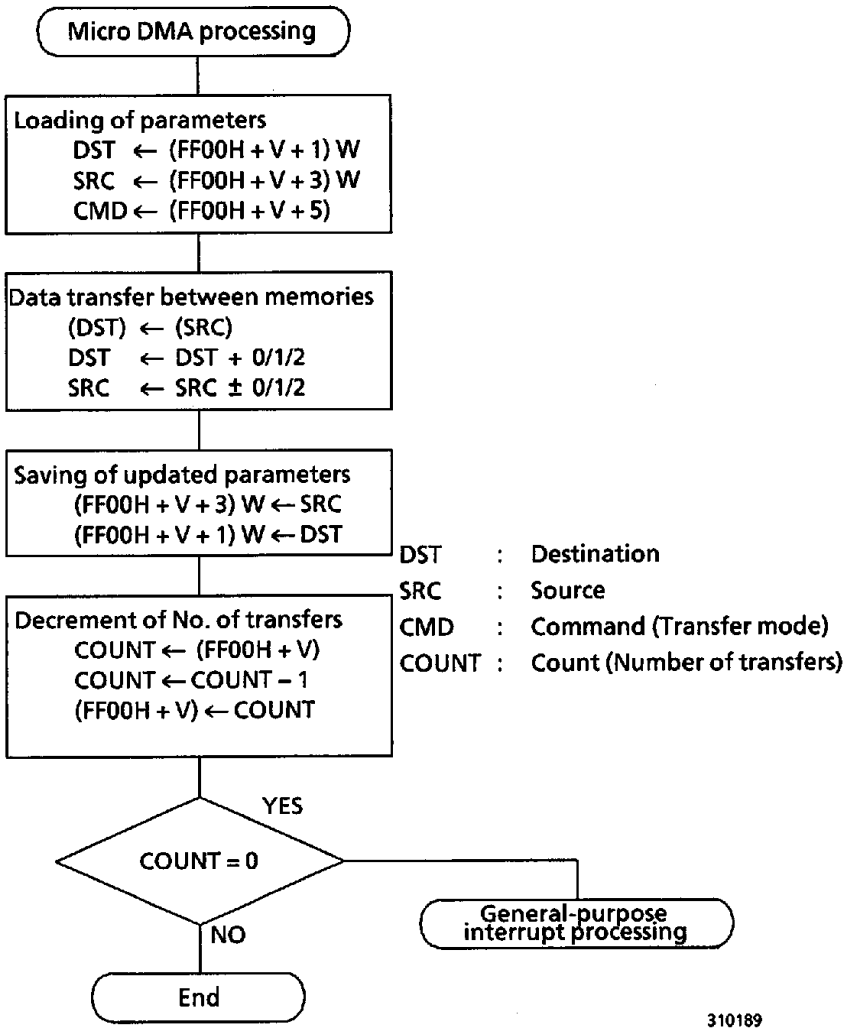


Figure 3.3 (3) Micro DMA Processing Flowchart

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

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

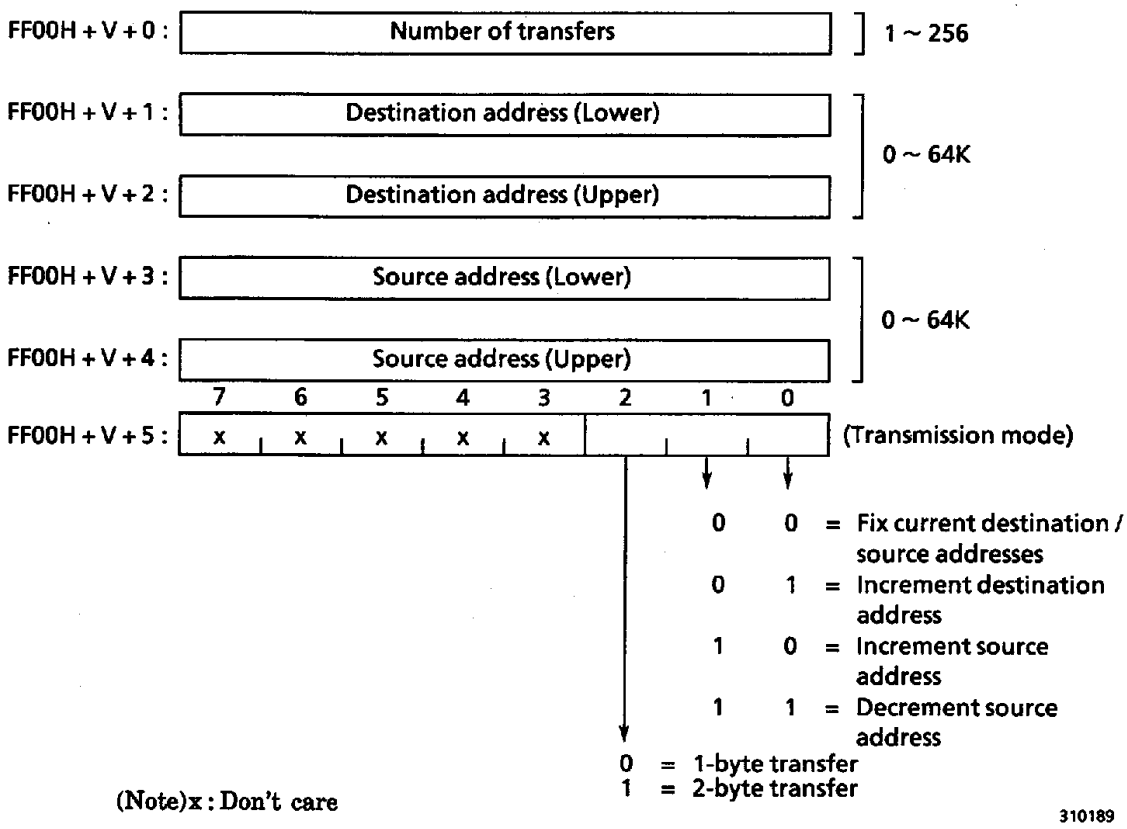


Figure 3.3 (4) Parameters for Micro DMA Processing

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

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

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

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

Table 3.3 (2) Addresses Updated by Micro DMA Processing

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

021290

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

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

Similar data transfers are made in the modes that "decrement the source address" and addresses are updated as shown in the table 3.3 (2).

Address updating in micro DMA processing is designed considering the I/O transfer from/to memory. Therefore, at least either destination or source address is fixed.

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

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

```
CALL    SIOINIT          ;  Initial setting for serial receiving.
SET     3,(0FFF4H)       ;  Enable an interrupt for serial data receiving.
SET     3,(0FFF6H)       ;  Set the micro DMA processing mode for serial receiving
                            interrupt.
LD      (0FF60H),7       ;  Set the number of transfers = 7.
LDW     (0FF61H),0FF00H  ;  Set FF00H for the destination start address.
LDW     (0FF63H),0FFE8H  ;  Set FFEBH for the source (serial receiving buffer) address.
LD      (FF65H),1        ;  Set the transfer mode (1-byte transfer; increment
                            destination address).

EI
 :
 :
ORG     0060H

Interrupt processing program after serial data receiving

RETI
```

060890

Figure 3.3 (5) Example of Micro DMA Processing

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

Execution time for micro DMA processing (when decremented number of transfers is not zero) is 46 states (9.2 μs at 10 MHz oscillation), regardless of whether 1-byte or 2-byte transfer mode is used.

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

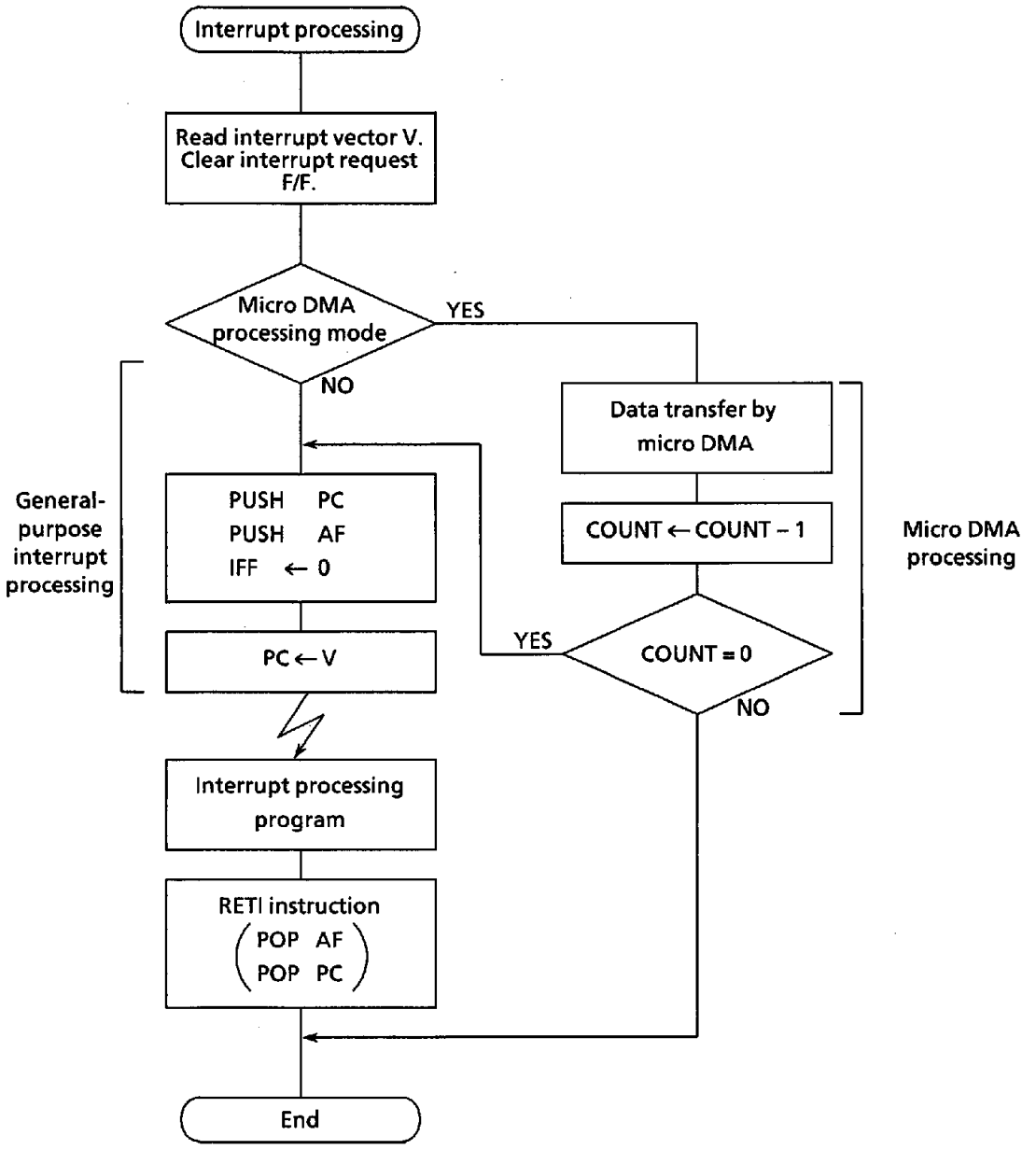


Figure 3.3 (6) Overall Interrupt Processing Flowchart

3.3.3 Interrupt Controller

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

The interrupt controller has an interrupt request flip-flop, interrupt enable/disable flag, and micro DMA enable/disable flag for each interrupt channel (total; 15 channels). The interrupt request flip-flop latches an interrupt request when it is issued from the peripheral devices. This flip-flop is reset to "0" when reset operation or interrupt is accepted by the CPU and the vector of that interrupt channel is read by the CPU, or when the CPU executes an instruction that clears the interrupt request for the specified channel (write "vector divided by 8" into the memory address FFEAH). For example, when executing

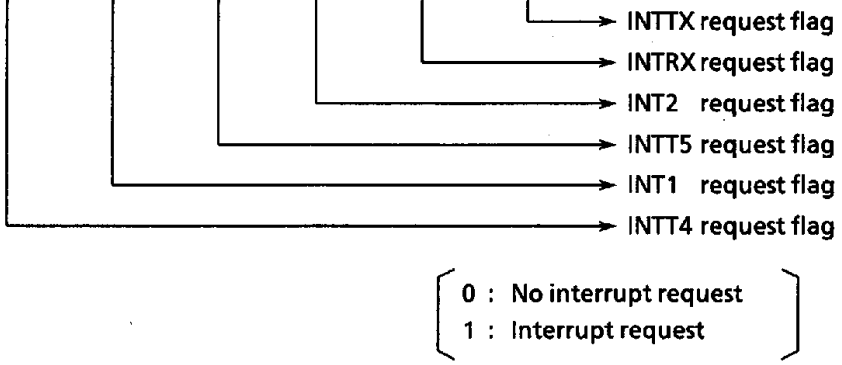
LD (0FFEAH), 38H/8

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

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

IRFL (FFEAH)

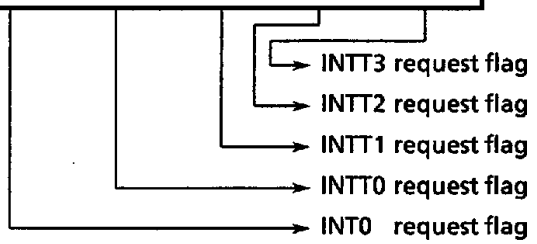
	7	6	5	4	3	2	1	0
bit Symbol	IRFT4	IRF1	IRFT5	IRF2	IRFRX	IRFTX	0	0
Read/Write	R (Write is possible only for IRF clear code.)							
After reset	0	0	0	0	0	0	0	0
Function	1: Interrupt is currently requested. (IRF is cleared by writing IRF clear code.)							



0 : No interrupt request
1 : Interrupt request

IRFH (FFEBH)

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



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

Figure 3.3 (7) Interrupt Request Flip-flops

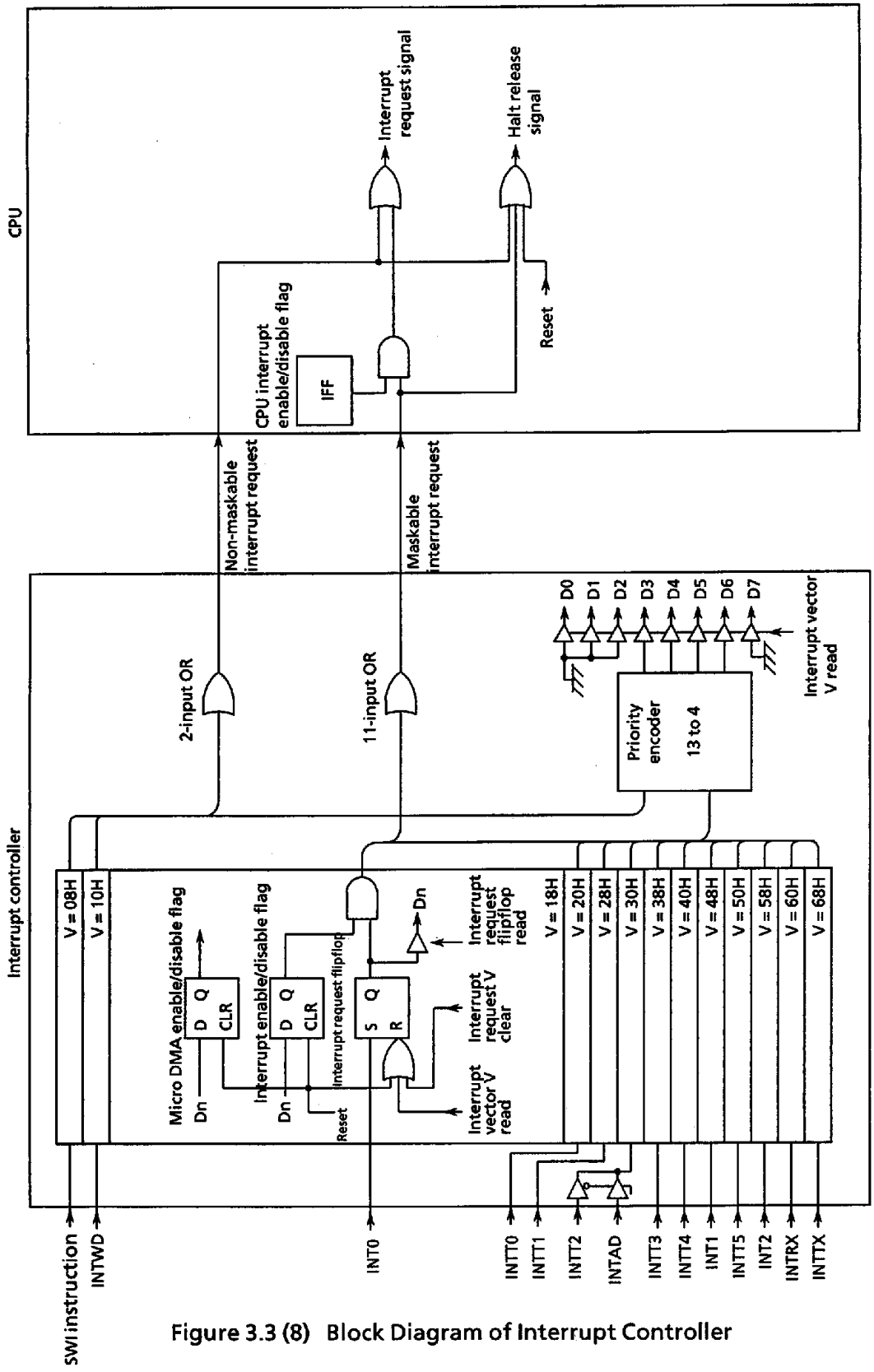


Figure 3.3 (8) Block Diagram of Interrupt Controller

The interrupt enable/disable flags provided for all interrupt request channels are assigned to the memory address FFF4H or FFF5H. Interrupts for a channel are enabled by setting the flag to "1". The flags are cleared to "0" by resetting.

The micro DMA enable/disable flag also provided for each interrupt request channel is assigned to the memory address FFF6H or FFF7H. The interrupt processing for each channel is placed in the micro DMA processing mode by setting this flag to "1". This flag is cleared to "0" (general-purpose interrupt processing mode) by resetting.

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

Interrupt by timer 2 (INTT2) and that by A/D converter (INTAD) use a common interrupt request channel. Immediately after resetting, INTT2 is input in the interrupt controller. To use INTAD, set "INTT2/INTAD selection bit" (ADIS: bit 5 of memory address FFF5H) to "1".

External interrupt features are as follows.

Interrupt	Common pin	Mode	Setting
INT0	P45	Level	INTEH<EDGE> = 0
		Rise edge	INTEH<EDGE> = 1
INT1	P46	Rise edge	T4MOD<CAPM1, 0> = 0,0 or 0,1 or 1,1
		Fall edge	T4MOD<CAPM1, 0> = 1, 0
INT2	P47	Rise edge	———

For the pulse width for external interrupt, refer to "4.7 Interrupt Operation".

Be careful that the following three are exceptional circuits.

INT0 level mode	As the INT0 is not edge type interrupt, the interrupt request flip-flop function is canceled, and thus an interrupt request from peripheral devices passes through S input of the flip-flop to become Q output. When the mode is changed over (from edge type to level type), the previous interrupt request flag will be cleared automatically. When the mode is changed from level to edge, the interrupt request flag set in the level mode is not cleared. Thus, use the following sequence to clear the interrupt request flag. DI SET 6, (0FFF5H) : Switch the mode from level to edge LD (0FFEAH), 03H : Clear interrupt request flag EI
INTAD	The interrupt request flip-flop can be cleared only by reset operation or reading the register that stores the A/D conversion value, and cannot be cleared by instruction. When the interrupt source is changed (from INTAD to INTT2), the previous interrupt request flag is cleared automatically.
INTRX	The interrupt request flip-flop is cleared only by reset operation or reading the serial channel receiving buffer, and cannot be cleared by an instruction.

INTEL (FFF4H)

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

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

INTEH (FFF5H)

	7	6	5	4	3	2	1	0
bit Symbol	–	EDGE	ADIS	IE0	IET0	IET1	IET2	IET3
Read/Write		R/W	R/W	R/W				
After reset		0	0	0	0	0	0	0
Function		INT0 0: Level 1: EDGE	0: INTT2 1: INTAD	1: Enable 0: Disable				

- INTT3 interrupt enable/disable flag
- INTT2/INTAD enable/disable flag
- INTT1 interrupt enable/disable flag
- INTT0 interrupt enable/disable flag
- INT0 interrupt enable/disable flag
- INTT2/INTAD selection

0	INTT2
1	INTAD

- INT0 control

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

Figure 3.3 (9) Interrupt Enable/Disable Flags

DMAEL (FFF6H)

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

- 3 → INTTX DMA enable/disable flag
- 4 → INTRX DMA enable/disable flag
- 5 → INT2 DMA enable/disable flag
- 6 → INTT5 DMA enable/disable flag
- 7 → INT1 DMA enable/disable flag
- 1 → INTT4 DMA enable/disable flag

DMAEH (FFF7H)

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

- 3 → INTT3 DMA enable/disable flag
- 4 → INTT2 DMA enable/disable flag
- 5 → INTT1 DMA enable/disable flag
- 6 → INTT0 DMA enable/disable flag
- 7 → INT0 DMA enable/disable flag

Figure 3.3 (10) Micro DMA Enable/Disable Flag

3.4 Standby Function

When a HALT instruction is executed, TMP90C845 enters the RUN, IDLE1, or STOP mode according to the contents of the halt mode setting register. The features are as follows:

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

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

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

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

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

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

WDMOD (FFD2H)

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

Explained in 3.13 "Watchdog Timer"

Explained in 3.4.2 "STOP mode"

Exchange flag (Explained in 3.1.2 "Register")

310189

Figure 3.4 (1) HALT Mode Setting Register

3.4.1 RUN Mode

Figure 3.4 (2) shows the timing for releasing the HALT state by an interrupt during RUN mode. In the RUN mode, the system clock inside MCU does not stop even after HALT instruction has been executed; the CPU merely stops executing instructions. Accordingly, the CPU repeats dummy cycle until HALT state is released. In the HALT state, interrupt requests are sampled at the fall edge of CLK signal.

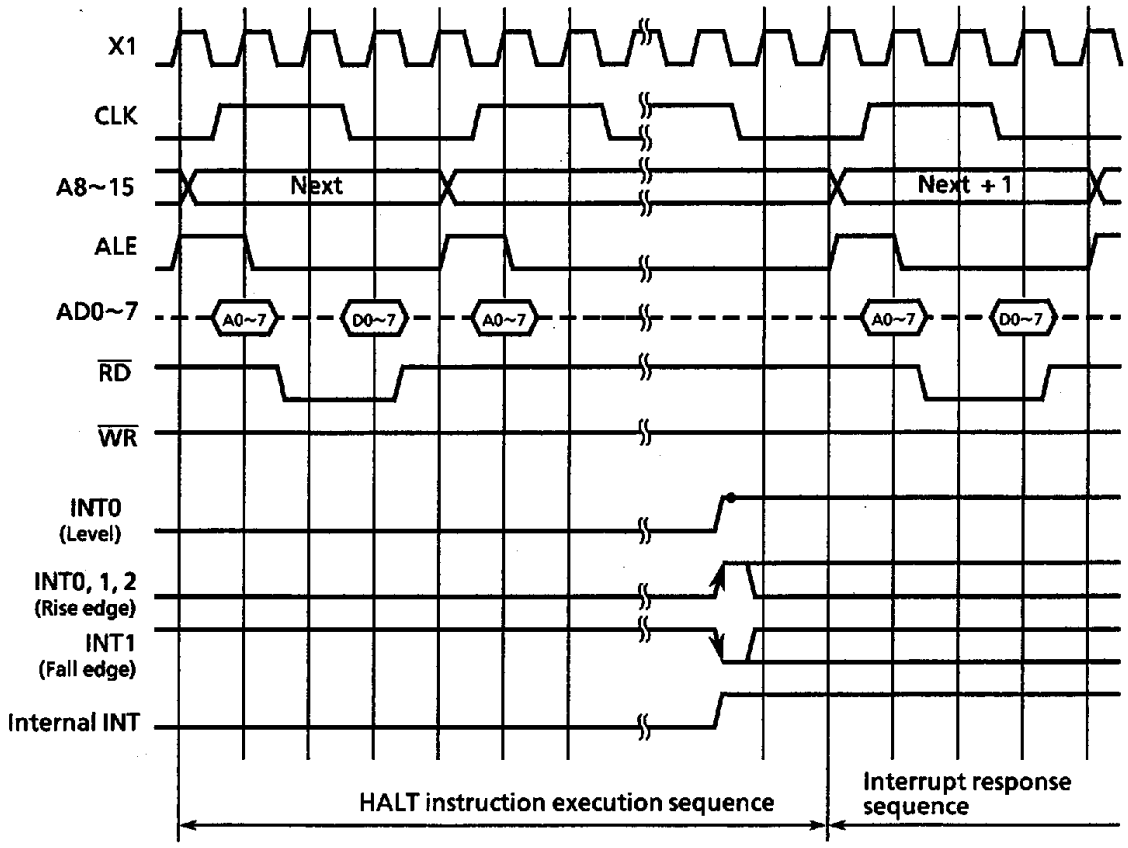


Figure 3.4 (2) HALT Release Timing Using Interrupts in RUN Mode

3.4.2 IDLE1 Mode

Figure 3.4 (3) shows the timing for releasing the HALT mode by interrupts in the IDLE1 mode.

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

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

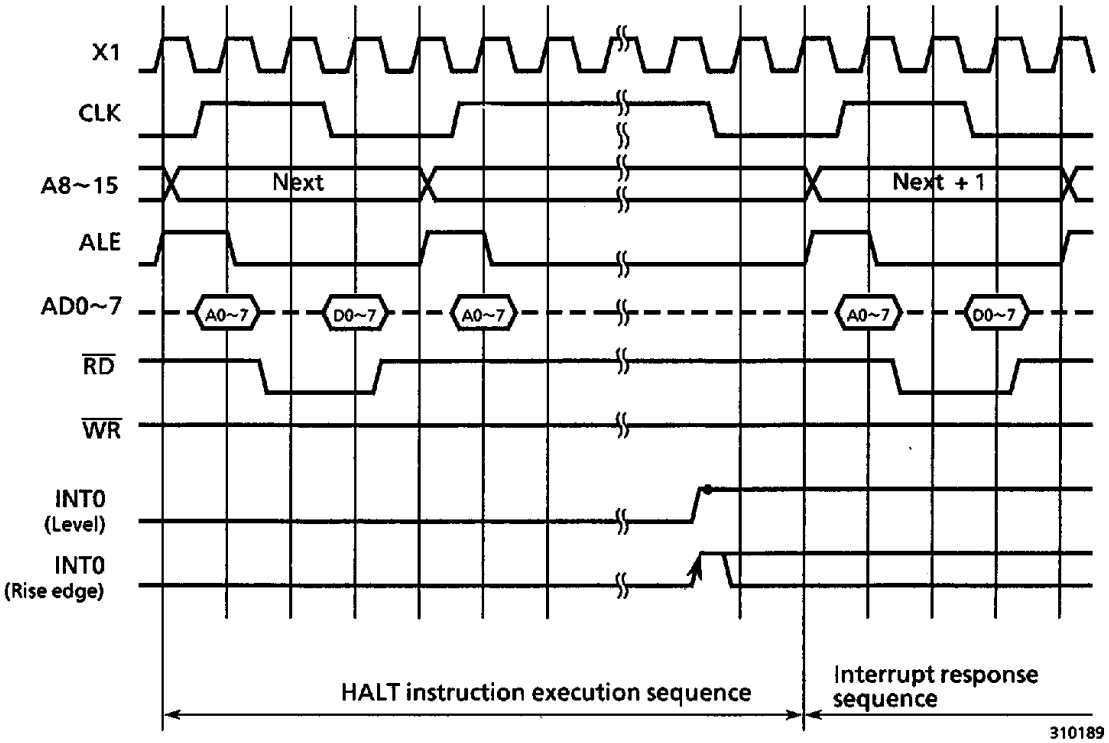


Figure 3.4 (3) HALT Release Timing Using Interrupts in IDLE1 Mode

3.4.3 STOP Mode

Figure 3.4 (4) shows the timing of HALT release caused by interrupts in STOP mode.

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

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

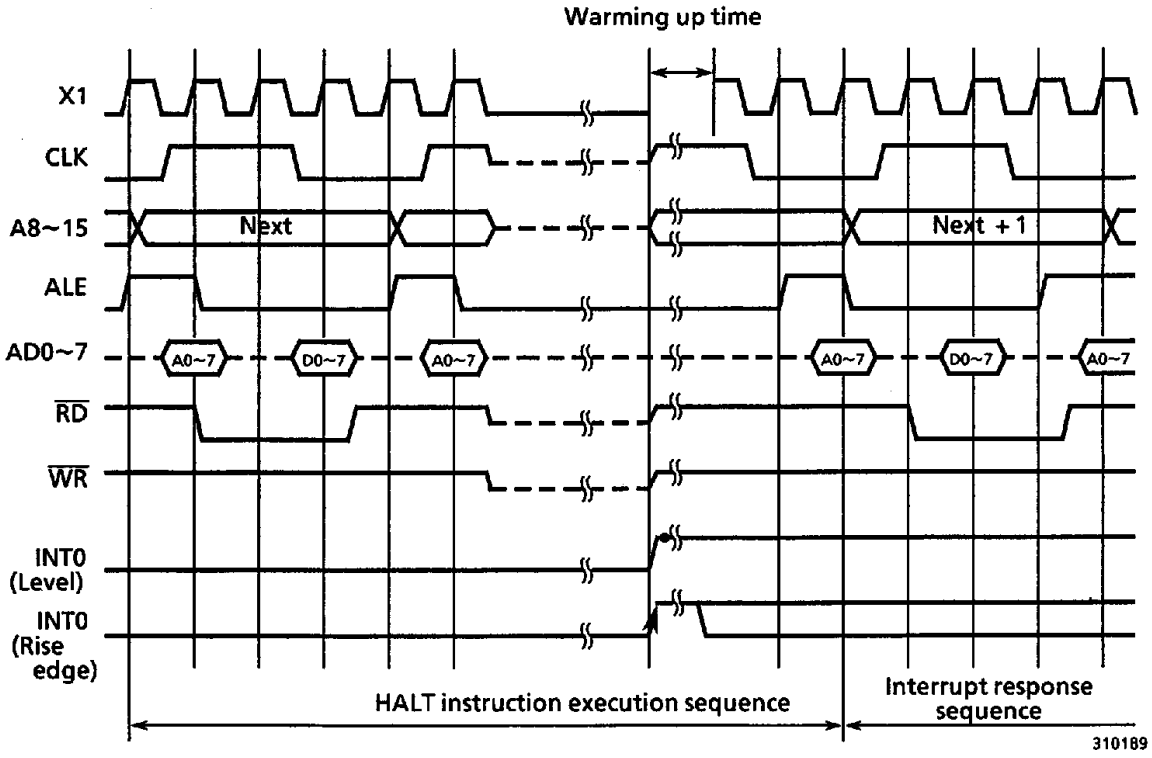


Figure 3.4 (4) HALT Release Timing Using Interrupts in STOP Mode

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

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

Table 3.4 (1) State of Pins in STOP Mode

Pin name	I/O	DRVE = 0	DRVE = 1
AD0~AD7	Tristate	——	——
A8~A15	Output pin	——	Output
P20~P26	Output pin	——	Output
P27	Input pin	——	Input
P30~P37	Input mode Output mode	—— ——	Input Output
P40~P44	Input mode Output mode	—— ——	Input Output
P45 (INT0)	Input mode Output mode	Input ——	Input Output
P46, P47	Input mode Output mode	—— ——	Input* Output
P50~P53	Input pin	——	——
P56, P57	Output pin	——	Output
P60~P63	Input mode Output mode	—— Output	—— Output
P70~P73	Input mode Output mode	—— Output	—— Output
CLK	Output pin	——	"1"
$\overline{\text{RESET}}$	Input pin	Input	Input
ALE	Output pin	"0"	"0"
$\overline{\text{EA}}$	Input pin	Input	Input
X1	Input pin	——	——
X2	Output pin	"1"	"1"

* : When in zero cross detect mode, intermediate bias is still applied to this pin.

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

Input : Input is enabled.

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

Output : Output status

Table 3.4 (2) I/O Operation and Release in HALT Mode

HALT mode			RUN	IDLE1	STOP
WDMOD<HALTM1, 0>			00	10	01
Operating block	CPU		Halt		
	I/O port		Retains the state when HALT instruction is executed.		See Table 3.4 (1)
	8-bit timer		Operation	Halt	
	16-bit timer				
	Stepping motor controller				
	Serial interface				
	A/D converter				
	Watchdog timer				
	Interrupt controller				
HALT releasing source	Interrupt	INTWD	○	—	—
		INT0	○	○	○
		INTT1	○	—	—
		INTT2	○	—	—
		INTAD	○	—	—
		INTT3	○	—	—
		INTT4	○	—	—
		INT1	○	—	—
		INTT5	○	—	—
		INT2	○	—	—
		INTRX	○	—	—
		INTTX	○	—	—
Reset			○	○	○

○ : Can be used for HALT release
— : Cannot be used for HALT release

3.5 Functions of Ports

TMP90C845 has a total of 36 I/O port pins. These port pins function not only as the general-purpose I/O ports but also as the I/O ports for the internal CPU and built-in I/O. Table 3.5 shows the functions of these port pins.

Table 3.5 Function of Ports

Port name	Pin name	No. of pins	Direction	Direction setting 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/$\overline{\text{ROMCS}}$
	P25	1	Output	—	$\overline{\text{A21}}$/$\overline{\text{RAMCS}}$
	P26	1	Output	—	$\overline{\text{A22}}$/$\overline{\text{IOCS}}$
	P27	1	Input	—	$\overline{\text{WAIT}}$
Port 3	P30~P34	5	I/O	Bit	
	P35	1			RxD
	P36	1			SCLK
	P37	1			TxD
Port 4	P40	1	I/O	Bit	TO1
	P41	1			TO3
	P42	1			TO4
	P43	1			TO5
	P44	1			TI0
	P45	1			INT0/TI2
	P46	1			INT1/TI4
	P47	1			INT2/TI5
Port 5	P50~P53	4	Input	—	AN0~AN3
Port 6	P60~P63	4	I/O	Bit	M00~M03
Port 7	P70~P73	4	I/O	Bit	M10~M13

These port pins function as the general-purpose I/O 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.

3.5.1 Port 2 (P20~P27)

Port 2 is an 8-bit general-purpose I/O port 2, each bit of which can be set to input or output. P20~P26 are output-only ports and commonly used for extended address (A16~A20, $\overline{A21}$~$\overline{A22}$) and chip select output ($\overline{ROMCS}$, $\overline{RAMCS}$, and $\overline{IOCS}$). By reset operation, the contents of the output latch of P20~P24 are cleared to "0", while those of P25 and P26 are set to "1". Resetting clears all bits of the control register to "0" and brings P20~P26 into the general-purpose output ports.

P27 is an input-only port, and automatically functions as a $\overline{WAIT}$ pin when external memory is accessed.

(1) P20~P23

These pins become the output for extended address by setting bit 0 to bit 3 of the port 2 control register P2CR.

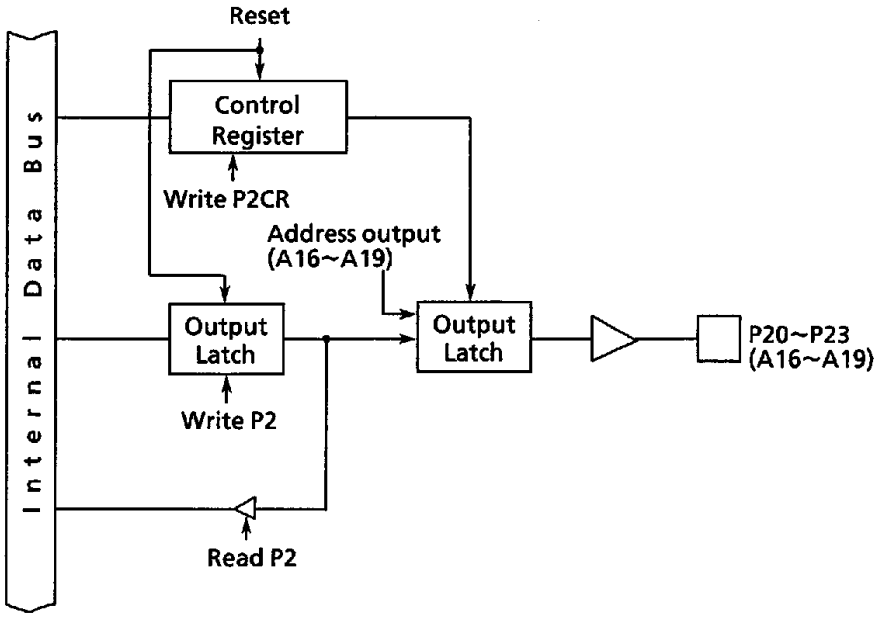


Figure 3.5 (1) Port 2 (P20 ~ P23)

(2) P24~P26

By setting bit 4 to bit 6 of the port 2 control register, these ports become the output for extended address (A20, $\overline{A21}$, and $\overline{A22}$) or chip select ($\overline{ROMCS}$, $\overline{RAMCS}$, and $\overline{IOCS}$).

Selection of the extended address output and chip select output are made by bit 0 to bit 3 of the port 2 and 3 function registers P23FR.

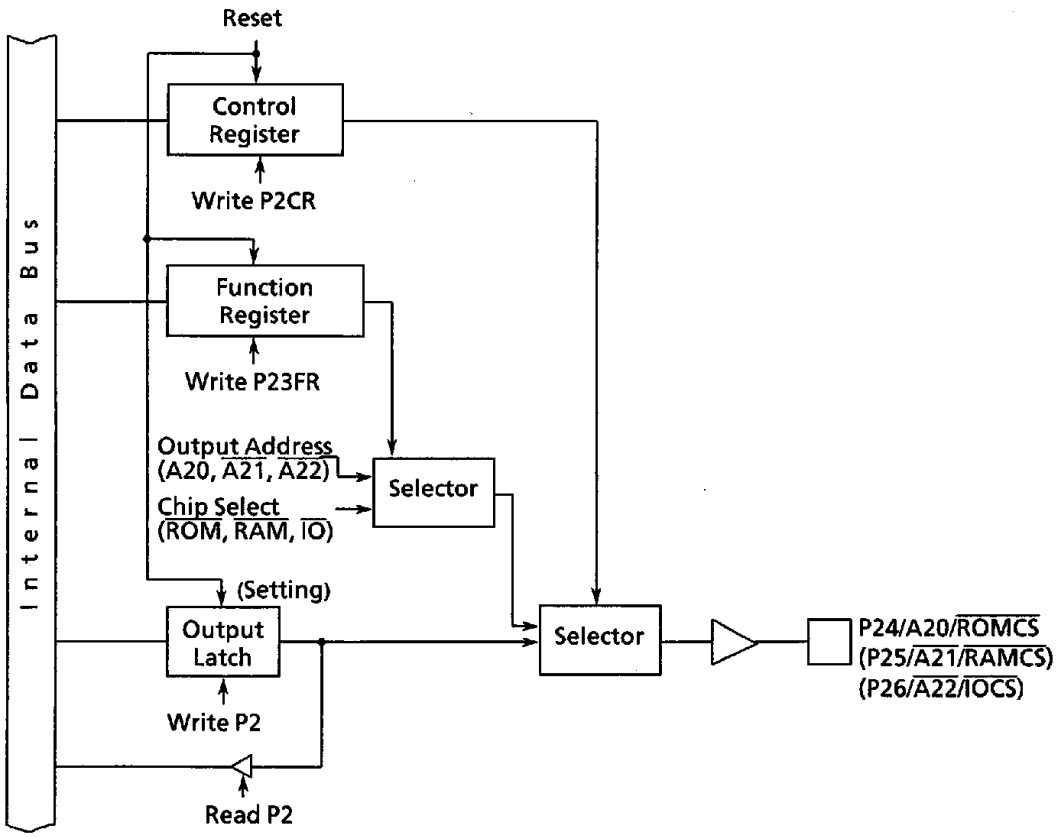


Figure 3.5 (2) Port 2 (P24~P26)

(3) P27

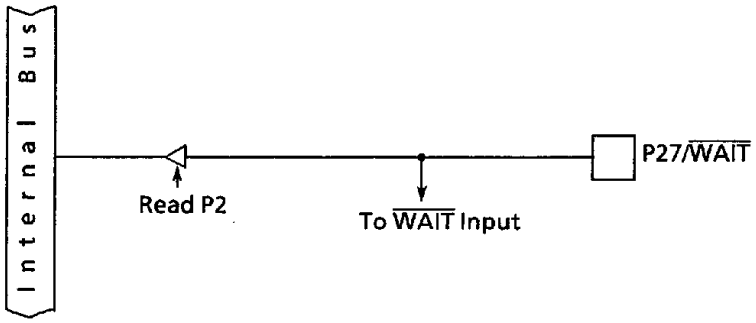


Figure 3.5 (3) Port 2 (P27)

Port 2 register

P2 (FFC4H)

	7	6	5	4	3	2	1	0
bit Symbol	P27	P26	P25	P24	P23	P22	P21	P20
Read/Write	R	R/W						
After reset	Input-only	1	1	1	0	0	0	0

Port 2 control register

P2CR (FFC5H)

	7	6	5	4	3	2	1	0
bit Symbol	–	P26C	P25C	P24C	P23C	P22C	P21C	P20C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0:Output Port 1: Address/CS							

Setting the output function of port 2

0	General-purpose output port
1	Extended address/Chip select output

Ports 2 and 3 function register

P23FR (FFCEH)

	7	6	5	4	3	2	1	0
bit Symbol	ODE	TXDC	SCLKC	Fixed to "0"	Fixed to "0"	IOCS	RAMCS	ROMCS
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	P37 0: CMOS 1: Open Drain	P37 control 0: Port 1: TxD Output	P36 control 0: Port 1: SCLK Output			P26 control 0: $\overline{A22}$ output 1: $\overline{IOCS}$ output	P25 control 0: $\overline{A21}$ output 1: $\overline{RAMCS}$ output	P24 control 0: A20 output 1: $\overline{ROMCS}$ output

Explained in "Port 3"

P24 function control

0	Extended address output(A20)
1	$\overline{ROMCS}$ output

P25 function control

0	Extended address output($\overline{A21}$)
1	$\overline{RAMCS}$ output

P26 function control

0	Extended address output($\overline{A22}$)
1	$\overline{IOCS}$ output

Figure 3.5 (4) Registers for Port 2 (1/2)

P25FR (FFDIH)

	7	6	5	4	3	2	1	0
bit Symbol	–	–	–	–	–	RDE	WAITC1	WAITC0
Read/Write						R/W		
After reset						0	0	0
Function						RD control 1: Always RD output	WAIT control 00: 2 state wait 01: Normal Wait 10: Non wait 11: –	

Read enable bit

(Outputs RD even while accessing the internal RAM or I/O)

Wait control

00	2 state wait
01	Normal wait
10	No wait
11	Reserved (cannot be used)

Figure 3.5 (4) Registers for Port 2 (2/2)

3.5.2 Port 3 (P30~P37)

Port 3 is an 8-bit general-purpose I/O port, each bit of which can be set for input or output. The control register P3CR is used to set input or output. Port 3 has the programmable pull-up function that enable pull-up when the programmable pull-up function that enable pull-up when the value of output latch is "1". By reset operations, all bits of the output latch are set to "1", while all bits of the control register are reset to "0", and port 3 is placed in the input mode with pull-up function.

In addition to the general-purpose I/O port function, P35~P37 have the I/O function for the internal serial interface. This is specified by function register P23FR. All bits of the function register are cleared to "0" by resetting, and the port turns to general-purpose I/O port mode.

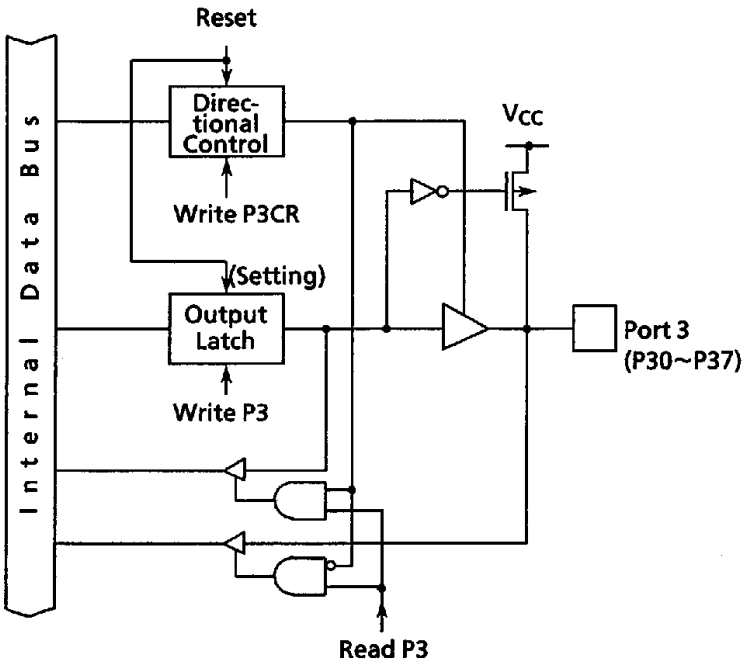


Figure 3.5 (5) Port 3

Port 3 register

P3 (FFC6H)

	7	6	5	4	3	2	1	0
bit Symbol	P37	P36	P35	P34	P33	P32	P31	P30
Read/Write	R/W							
After reset	Input mode							

Port 3 control register

P3CR (FFC7H)

	7	6	5	4	3	2	1	0
bit Symbol	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: IN 1: OUT (Select I/O on bit basis)							

Select port 3 I/O

0	Input
1	Output

Ports 2 and 3 function register

P23FR (FFCEH)

	7	6	5	4	3	2	1	0
bit Symbol	ODE	TXDC	SCLKC	Fixed to "0"	Fixed to "0"	IOCS	RAMCS	ROMCS
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	P37 0: CMOS 1: Open Drain	P37 control 0: Port 1: TxD Output	P36 control 0: Port 1: SCLK Output			P26 control 0: $\overline{A22}$ 1: $\overline{IOCS}$	P25 control 0: $\overline{A21}$ 1: $\overline{RAMCS}$	P24 control 0: A20 1: $\overline{ROMCS}$

Explained in 3.5.1 "Port 2"

Setting P36 as SCLK output

0	Port output
1	SCLK output

Setting P37 as TxD output

0	Port output
1	TxD output

Setting P37 as open drain output

0	CMOS output
1	Open drain output

Figure 3.5 (6) Registers for Port 3

3.5.3 Port 4 (P40~P47)

Port 4 is an 8-bit general-purpose I/O port, each bit of which can be set for input or output port. The control register is used to set for input or output. Port 4 has the programmable pull-up function that enables pull-up when the value of output latch is "1". By reset operation, all bits of the output register are set to "1", all bits of the control register are reset to "0", port 4 is placed in the input mode with pull-up function.

In addition to the general-purpose I/O port function, these ports function as interrupt request input, clock input for timer or event counter, or timer output.

(1) P40~P43

When specified by port 4 function register P4FR<TO1S to TO5S>, these ports become the timer output.

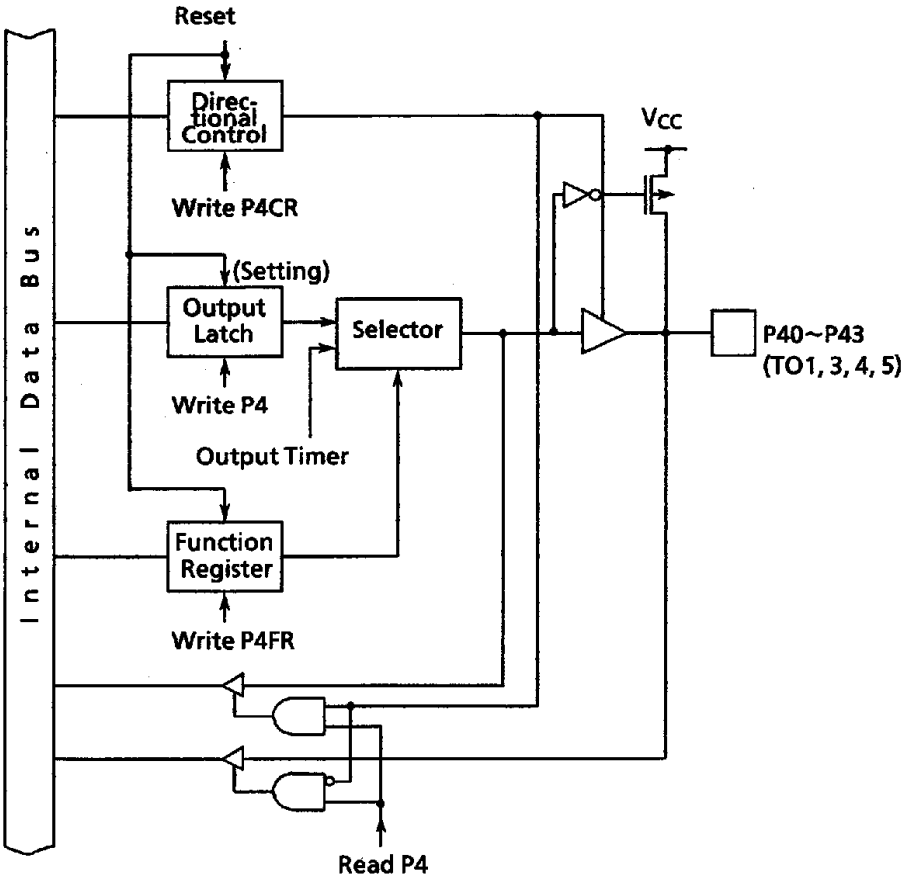


Figure 3.5 (7) Port 4 (P40~P43)

(2) P44

P44 is also used as clock input (TI0) for 8-bit timer 0.

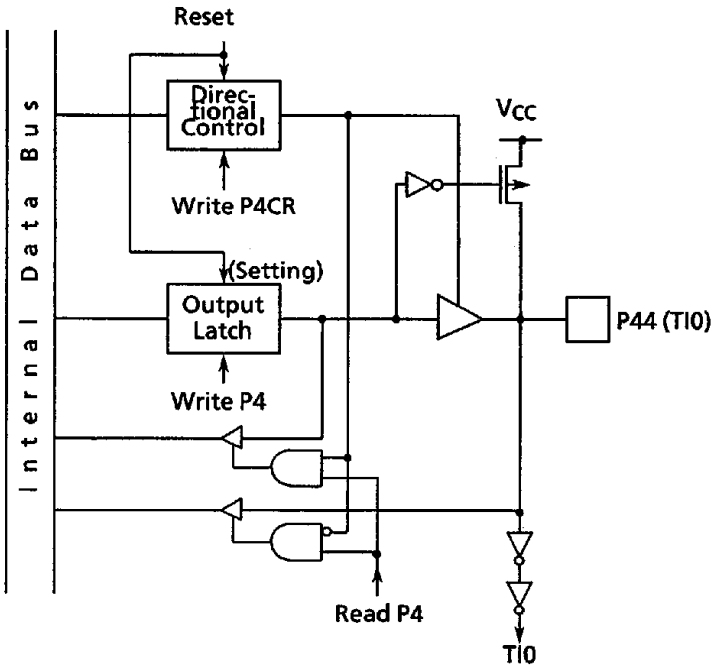


Figure 3.5 (8) Port 4 (P44)

(3) P45

P45 is also used as clock input (TI2) for 8-bit timer 2 as well as external interrupt request input (INT0).

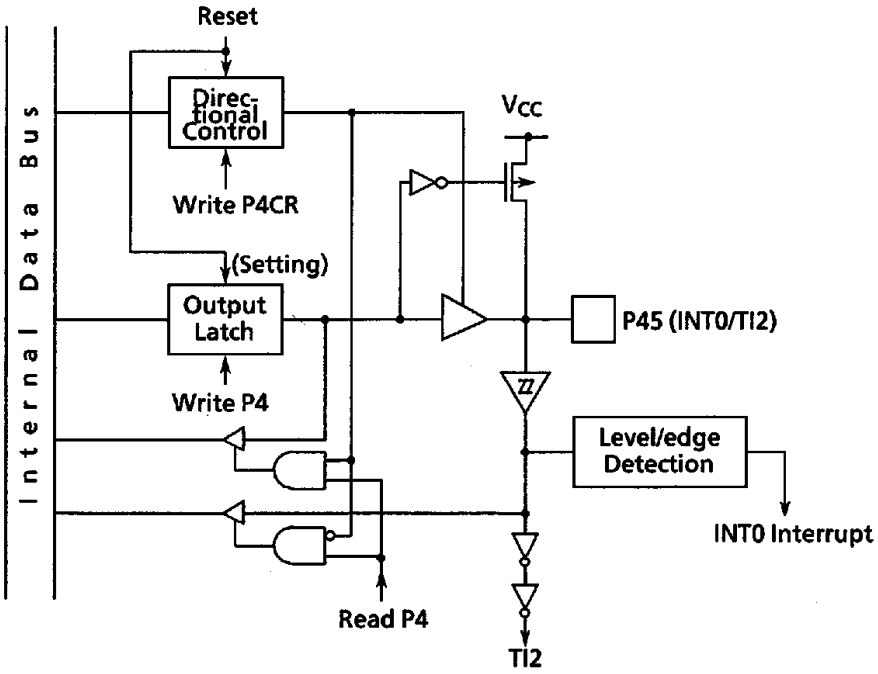


Figure 3.5 (9) Port 4 (P45)

(4) P46 and P47

These ports are also used as the clock input for 16-bit timer or event counter as well as external interrupt request input. These ports include zero cross detection circuit and can be disabled or enabled by setting the port 4 function register P4FR<ZCE1, ZCE2>.

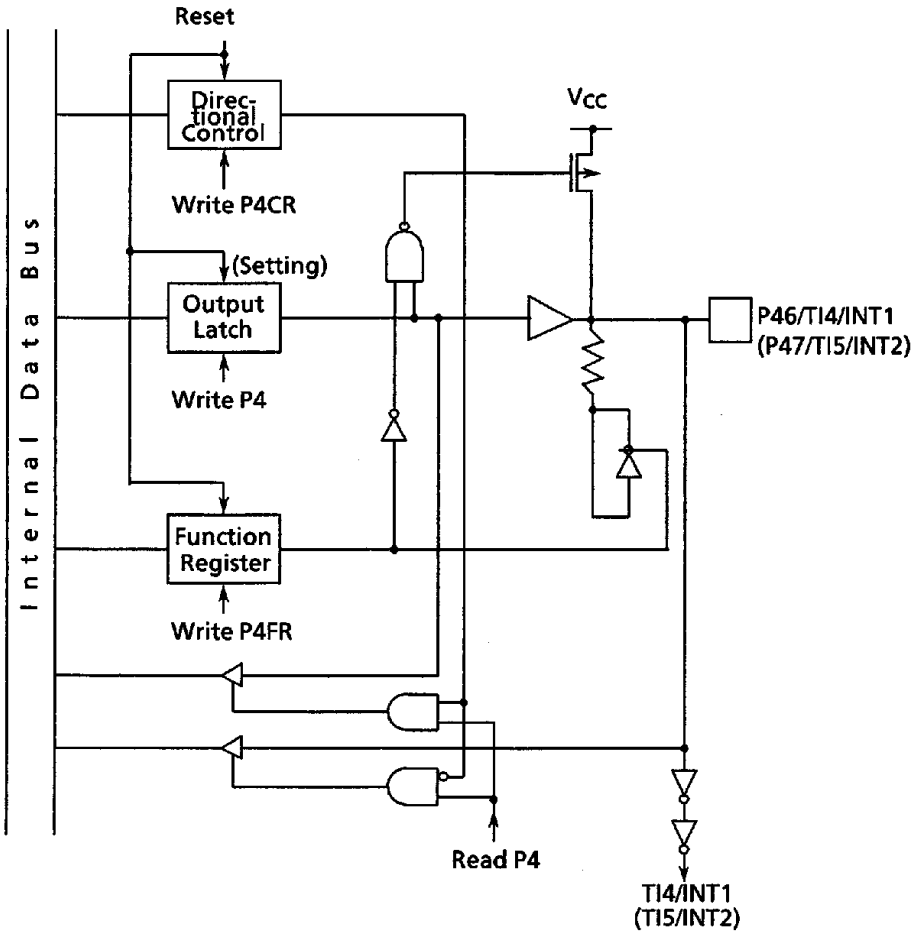


Figure 3.5 (10) Port 4 (P46 and P47)

Port 4 register

P4 (FFC8H)

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

Port 4 control register

P4CR (FFC9H)

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

→ Select port 4 I/O

0	Input
1	Output

Port 4 function register

P4FR (FFCFH)

	7	6	5	4	3	2	1	0
bit Symbol	ZCE2	ZCE1	–	–	TO5S	TO4S	TO3S	TO1S
Read/Write	R/W				R/W			
After reset	0	0			0	0	0	0
Function	P47 control 1: ZCD Enable	P46 control 1: ZCD Enable			P43 control 0: Port 1: TO5	P42 control 0: Port 1: TO4	P41 control 0: Port 1: TO3	P40 control 0: Port 1: TO1

→ P46 zero cross enable

0	Disable
1	Enable

→ P47 zero cross enable

0	Disable
1	Enable

→ Setting P40 as the output of timer 1

0	Port output
1	TO1 output

→ Setting P41 as the output of timer 3

0	Port output
1	TO3 output

→ Setting P42 as the output of timer 4

0	Port output
1	TO4 output

→ Setting P43 as the output of timer 5

0	Port output
1	TO5 output

Figure 3.5 (11) Registers for Port 4

3.5.4 Port 5 (P50~P53)

Port 5 is a 4-bit general-purpose input port.
P50 to P53 are also used as analog input pins (AN0~AN3).

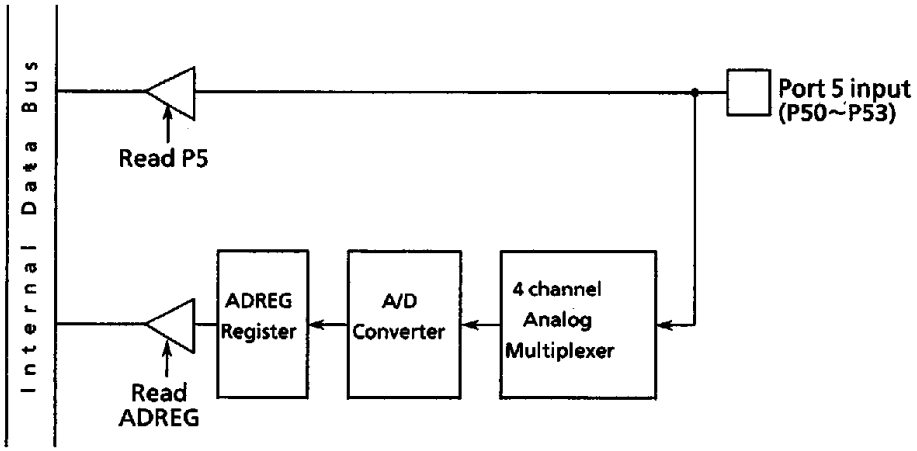


Figure 3.5 (12) Port 5

Port 5 register

P5 (FFCAH)

	7	6	5	4	3	2	1	0
bit Symbol	Fixed to "1"	Fixed to "1"	–	–	P53	P52	P51	P50
Read/Write	R/W				R			
After reset	1	1			Input-only			
Function					Used also as analog input pins (AN0~AN3)			

Figure 3.5 (13) Registers for Port 5

3.5.5 Port 6 (P60~P63)

Port 6 is a 4-bit general-purpose I/O port each bit of which can be set for input or output. The control register P67CR<P63C~P60C> is used for input or output. By reset operation, this control register is reset to "0", and port 6 is placed in the input mode.

This port can be used also as stepping motor control or pattern generation port 0 (M00~M03). Function register P67FR<PAT0, CCW0, M0M, and M0S> specifies whether the port is to be used as the general-purpose I/O port or stepping motor control/pattern generation port. When reset, it becomes a general-purpose I/O port.

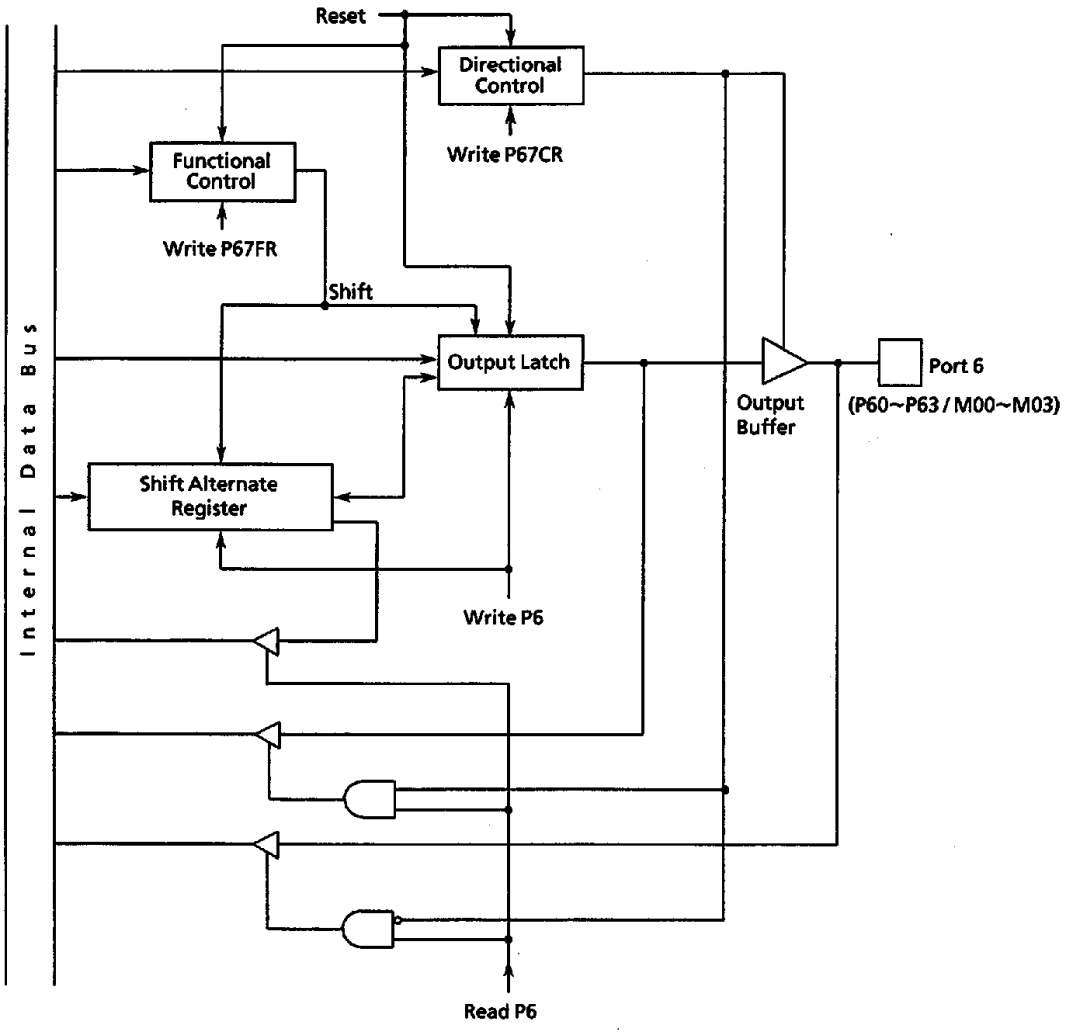


Figure 3.5 (14) Port 6

3.5.6 Port 7 (P70~P73)

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

This port can also be used as a stepping motor control/pattern generation port 1 (M10~M13). The function register P67FR<PAT1, CCW1, M1M, M1S> specifies whether it is to be used as the general-purpose I/O port or stepping motor control/pattern generation port. When reset, it becomes a general-purpose I/O port.

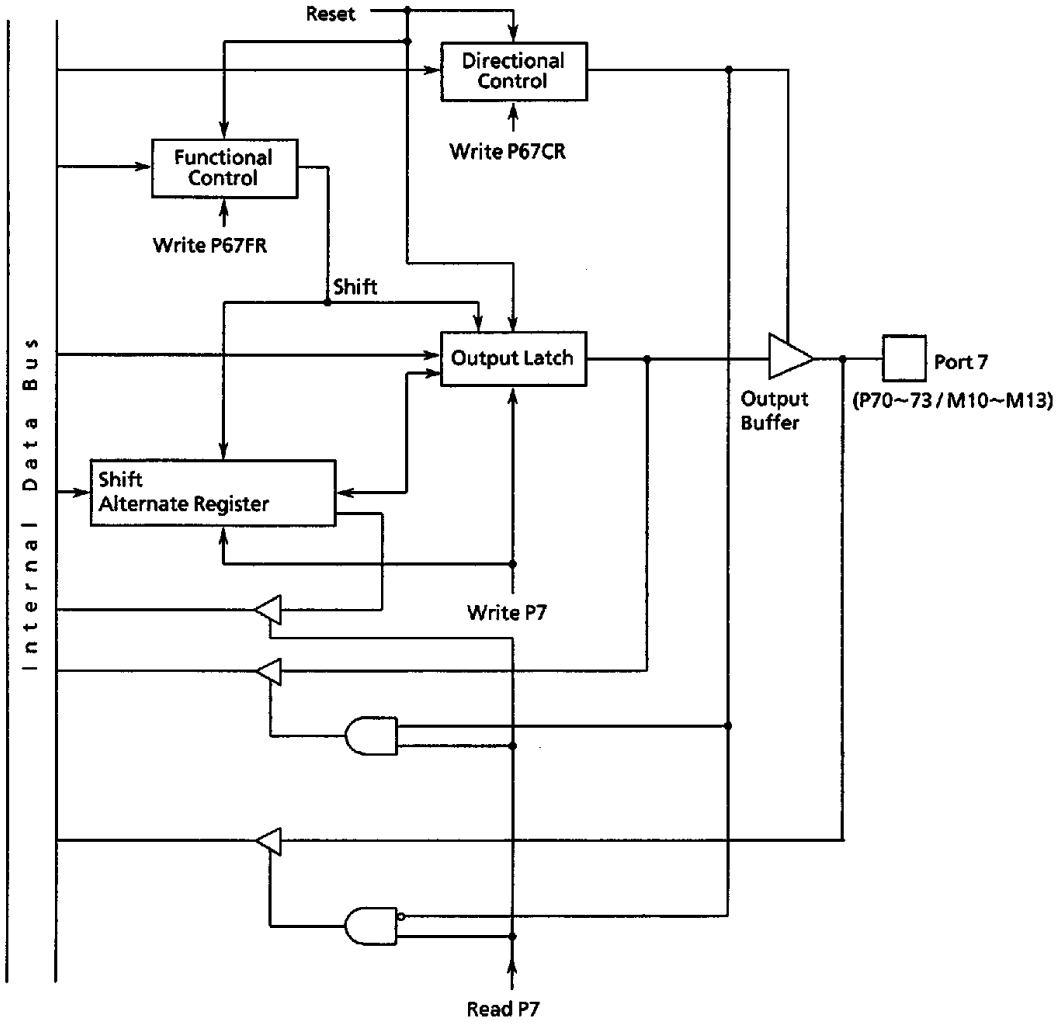


Figure 3.5 (15) Port 7

Port 6 register

P6 (FFCBH)

	7	6	5	4	3	2	1	0
bit Symbol	SA63	SA62	SA61	SA60	P63	P62	P61	P60
Read/Write	R/W				R/W			
After reset	Undefined				Input mode			
Function	Shift Alternate reg.0				Commonly used as stepping motor control port 0 as well as pattern generation port 0			

Shift alternate register 0 Port 6

Port 7 register

P7 (FFCCH)

	7	6	5	4	3	2	1	0
bit Symbol	SA73	SA72	SA71	SA70	P73	P72	P71	P70
Read/Write	R/W				R/W			
After reset	Undefined				Input mode			
Function	Shift Alternate reg.1				Commonly used as stepping motor control port 1 as well as pattern generation port 1			

Shift alternate register 1 Port 7

Ports 6 and 7 control register

P67CR (FFCDH)

	7	6	5	4	3	2	1	0
bit Symbol	P73C	P72C	P71C	P70C	P63C	P62C	P61C	P60C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	(Select I/O on bit basis)							

Select port 6 I/O

0	Input
1	Output

Select port 7 I/O

0	Input
1	Output

Figure 3.5 (16) Registers for Port 6 and Port 7 (1/2)

P67FR (FFD0H)

	7	6	5	4	3	2	1	0
bit Symbol	PAT1	CCW1	M1M	M1S	PAT0	CCW0	M0M	M0S
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: 8BIT 1: 4BIT WR	0: Normal rotation 1: Reverse rotation (For port 7)	0: 4Step 1: 8Step	0: Port 1: Step	0: 8BIT 1: 4BIT WR	0: Normal rotation 1: Reverse rotation (For port 6)	0: 4Step 1: 8Step	0: Port 1: Step

Setting the port 6 function

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

Setting the port 7 function

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

Explained in 3.10 "Stepping Motor Control/Pattern Generation Port".

Figure 3.5 (16) Registers for Port 6 and Port 7 (2/2)

3.6 Extending the Program Space

TMP90C845 has a simple MMU function which can extend the program space up to 4M bytes.

Figure 3.6 (1) shows the block diagram of the program space extension feature.

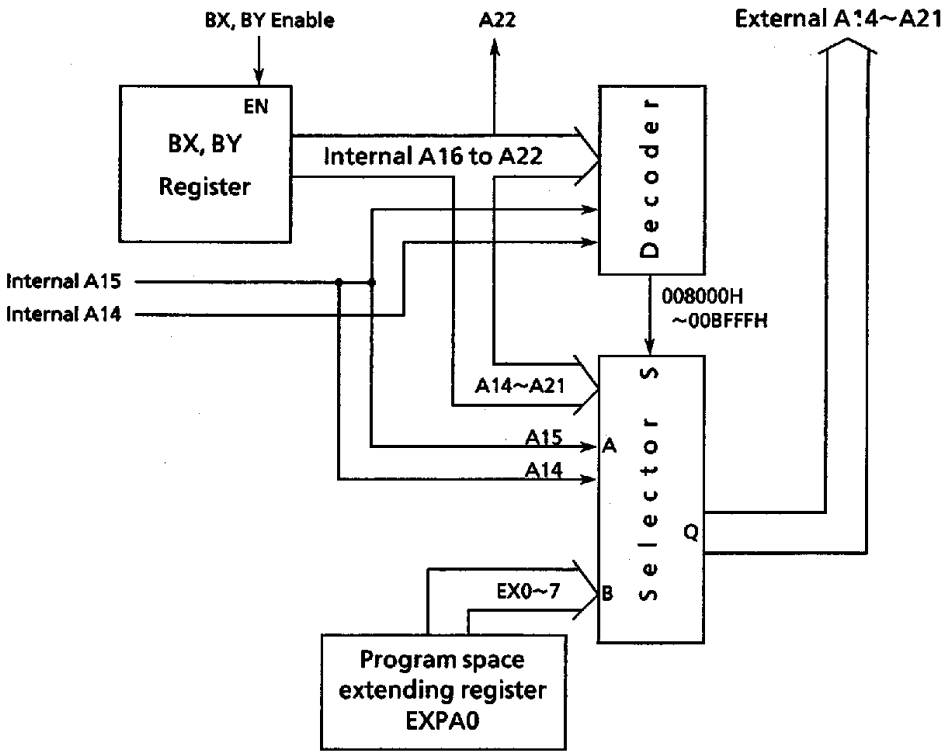


Figure 3.6 (1) Block Diagram of Program Space Extension Feature

3.6.1 Operation

For TMP90C845, when CPU accesses the logical address 008000H~00BFFFH, the contents of address extension register EXPA0 (at memory address FFDFH) will automatically be output to A14~A21.

008000H~00BFFFH are called local area, while other space within 64K byte is called common area. Figure 3.6 (2) shows an example of the memory map where EXPA0 is set to 06H.

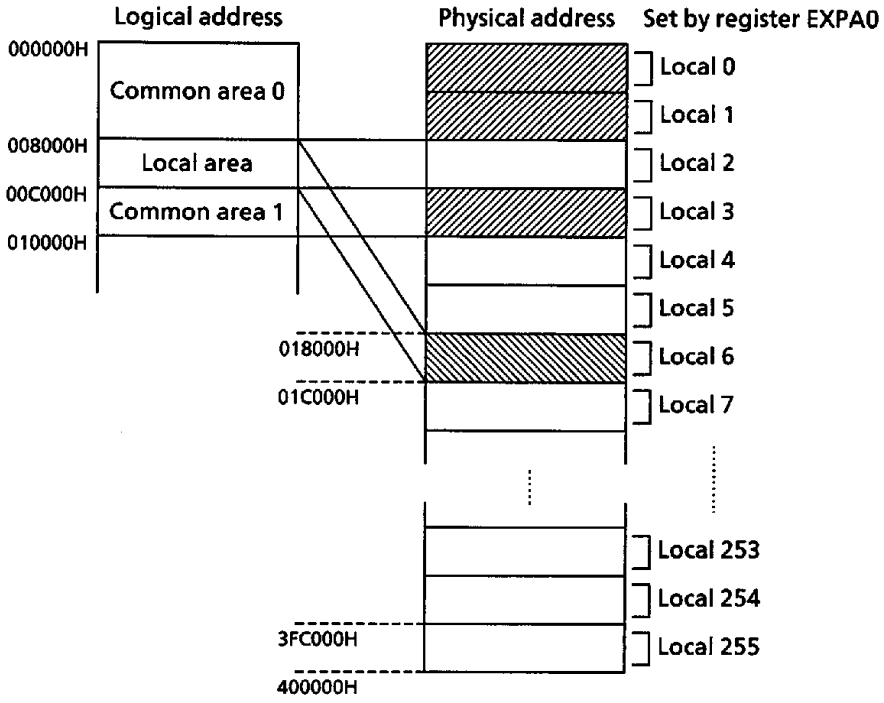
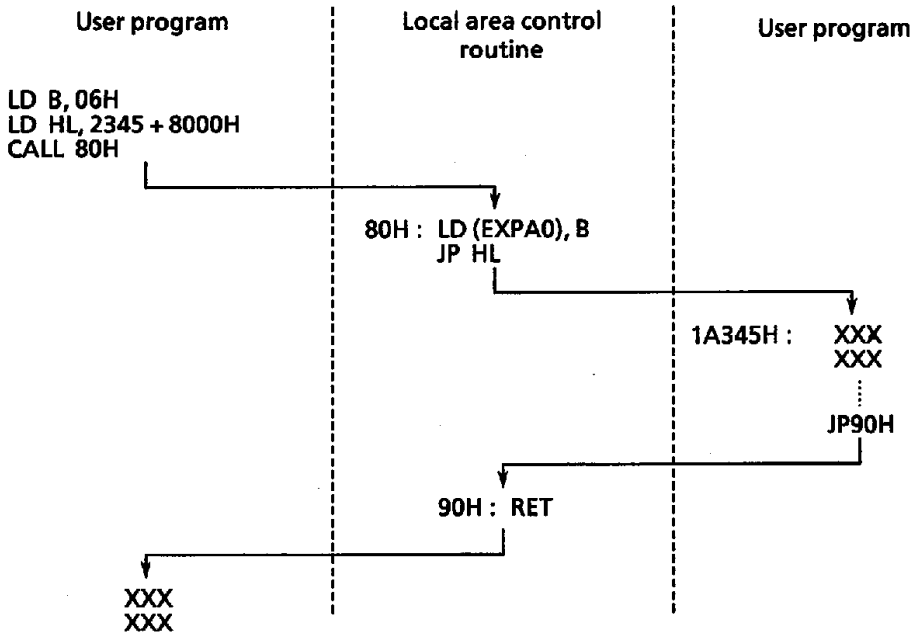


Figure 3.6 (2) Memory Map Using the Program Space Extension Function

Setting example :



In the above example, address 2345H in local 6 (physical address: 6 x 4000H + 2345H = 1A345H) is called a subroutine.

This can be implemented by specifying a local page to Breg and an offset value in the local area to HLreg and calling 80H within the local area control routine. Return can be done by jumping to 90H.

As in this example, by having a local area control routine in a common area, program can easily be located in an extended program space.

Besides, with this extension feature, micro DMA function and repeat instruction (LDIR, etc.) can be used for the data in a space which exceeds 64K byte.

3.6.2 Control Registers

The program extension feature is enabled when a value is written in the program space extension register (EXPA0). When reset, the value of EXPA0 register becomes "02H", and logical and physical addresses become equal.

EXPA0 must be written in a common area. If the value of EXPA0 is changed in a local area 008000H~00BFFFH, the physical address is immediately shifted to other local area.

EXPA0 (FFDFH)

	7	6	5	4	3	2	1	0
bit Symbol	EX7	EX6	EX5	EX4	EX3	EX2	EX1	EX0
Read/Write	R/W							
After reset	0	0	0	0	0	0	1	0
Function	Program Area Extension Bits (Corresponds to A14~A21.)							

3.7 Programmable Chip Select

TMP90C845 has three chip select outputs for facilitating the supply of chip select signals to ROM, RAM, and I/O which are to be connected to external devices.

The three chip select outputs are called $\overline{\text{ROMCS}}$, $\overline{\text{RAMCS}}$, and $\overline{\text{IOCS}}$.

$\overline{\text{IOCS}}$ is allocated to 8 bytes of external memory (00FFF8H~00FFFH) and controls external I/O in direct addressing.

$\overline{\text{RAMCS}}$ can be set in units of 16 Kbyte to 128 Kbyte by setting the control register PCSR. $\overline{\text{ROMCS}}$ occupies the entire space except for the above $\overline{\text{IOCS}}$ and $\overline{\text{RAMCS}}$ spaces.

Figure 3.7 (1) shows the block diagram of programmable chip select function.

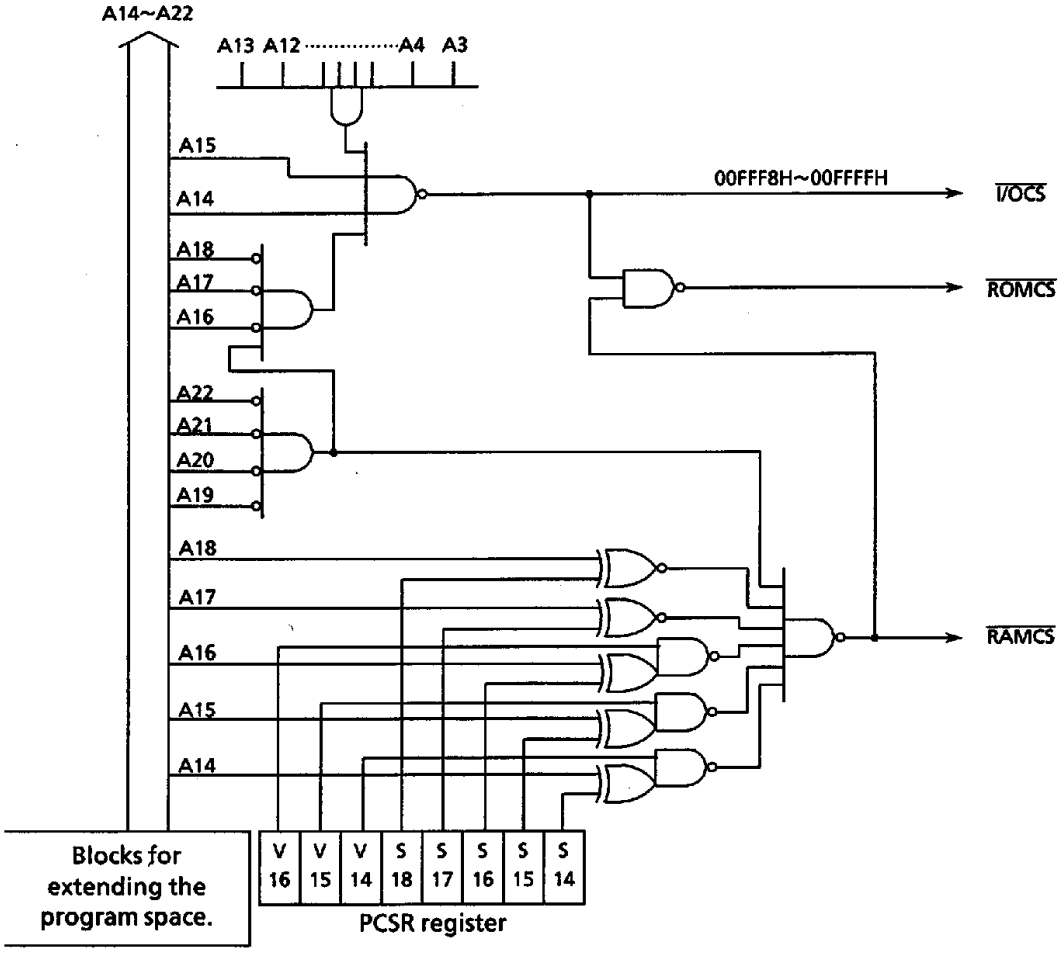


Figure 3.7 (1) Block Diagram of Programmable Chip Select Function

3.7.1 Control Register

$\overline{IOCS}$ is fixed to 00FFF8H~00FFFFH, and $\overline{RAMCS}$ specifies an area in which chip select is output by the control register PCSR.

Bit 0 to bit 4 of PCSR correspond to A14~A18, while bit 5 to bit 7 specify whether the values specified for bit 0 to bit 2 are valid or not.

PCSR (FFDEH)

	7	6	5	4	3	2	1	0
bit Symbol	V16	V15	V14	S18	S17	S16	S15	S14
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	1: S16 Valid	1: S15 Valid	1: S14 Valid	A18 Set	A17 Set	A16 Set	A15 Set	A14 Set

→ Setting A14~A18 as RAMCS (Always "1" after reset)

→ Setting whether S14~S16 are valid or not

0	Set values of S14~S16 are invalid.
1	Set values of S14~S16 are valid.

Note : For setting RAMCS, A19~A22 are preset to "0000".

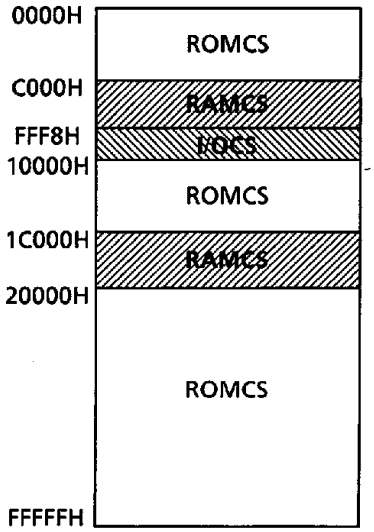
3.7.2 Example of Setting

When PCSR is set to "63H", the output of each chip select is as shown in the following memory map.

PCSR setting

V16	V15	V14	S18	S17	S16	S15	S14
0	1	1	0	0	0	1	1

As V14 and V15 are "1" and V16 is "0", the settings of S14 and S15 are valid, whereas the setting of S16 is invalid.



3.8 8-bit Timers

TMP90C845 contains four 8-bit timers (timers 0, 1, 2 and 3), each of which can be operated independently. The cascade connection allows these timers to be used as 16-bit timers. The following four operating modes are provided for the 8-bit timers.

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

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

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

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

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

Among the input clock sources for the interval timers, the internal clocks of ϕT1, ϕT4, ϕT16, and ϕT256 are obtained from the 9-bit prescaler shown in Figure 3.8 (2).

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

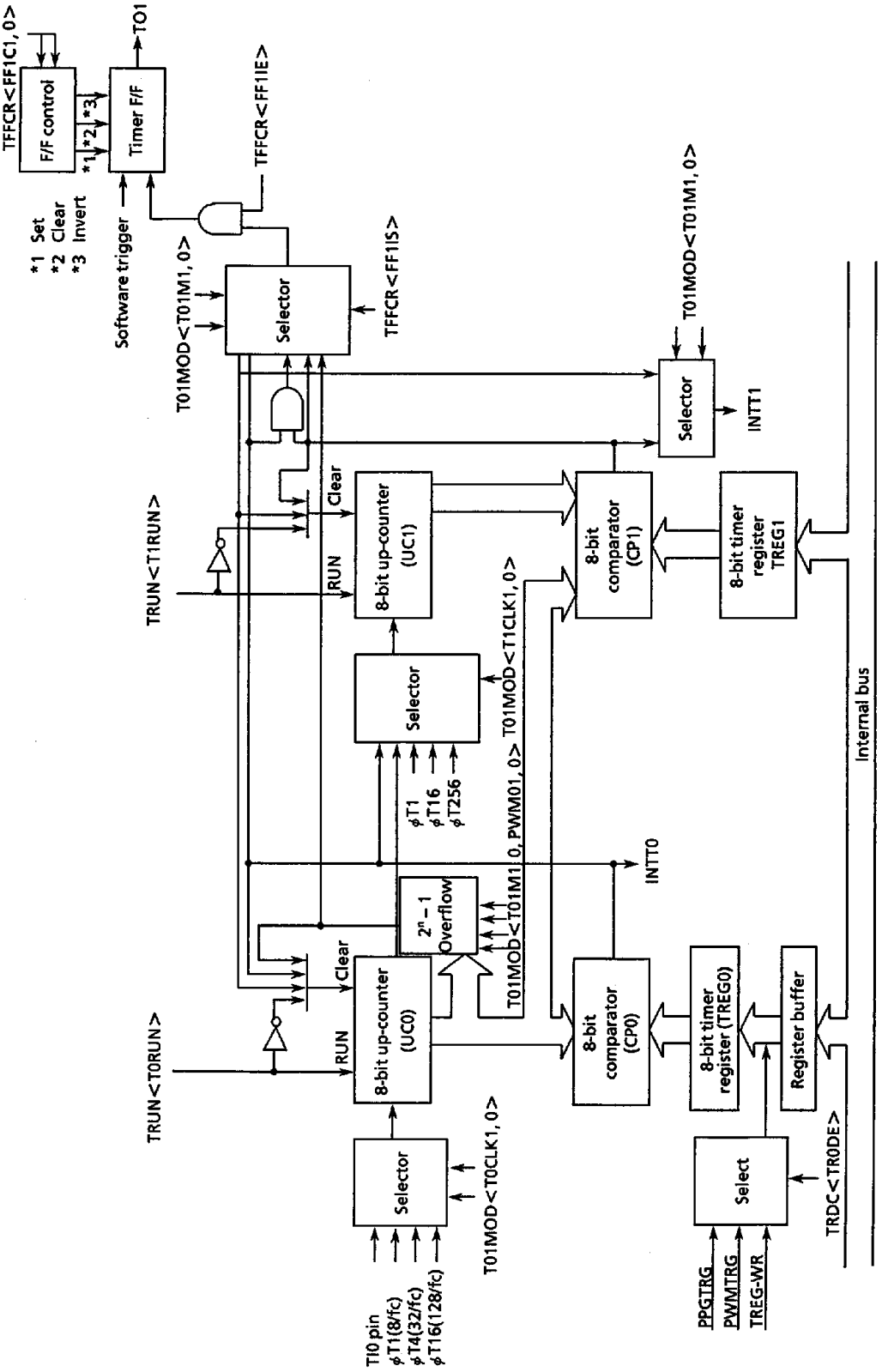


Figure 3.8 (1) Block Diagram of 8-bit Timers (Timers 0 and 1)

① Prescaler

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

Among them, 8-bit timer uses 4 types of clock: ϕT1, ϕT4, ϕT16, and ϕT256.

This prescaler can be run or stopped by the timer operation control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to "1", while the prescaler is cleared to zero and stops operation when <PRRUN> is set to "0". Resetting clears <PRRUN> to "0", which clears and stops the prescaler.

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	128μs

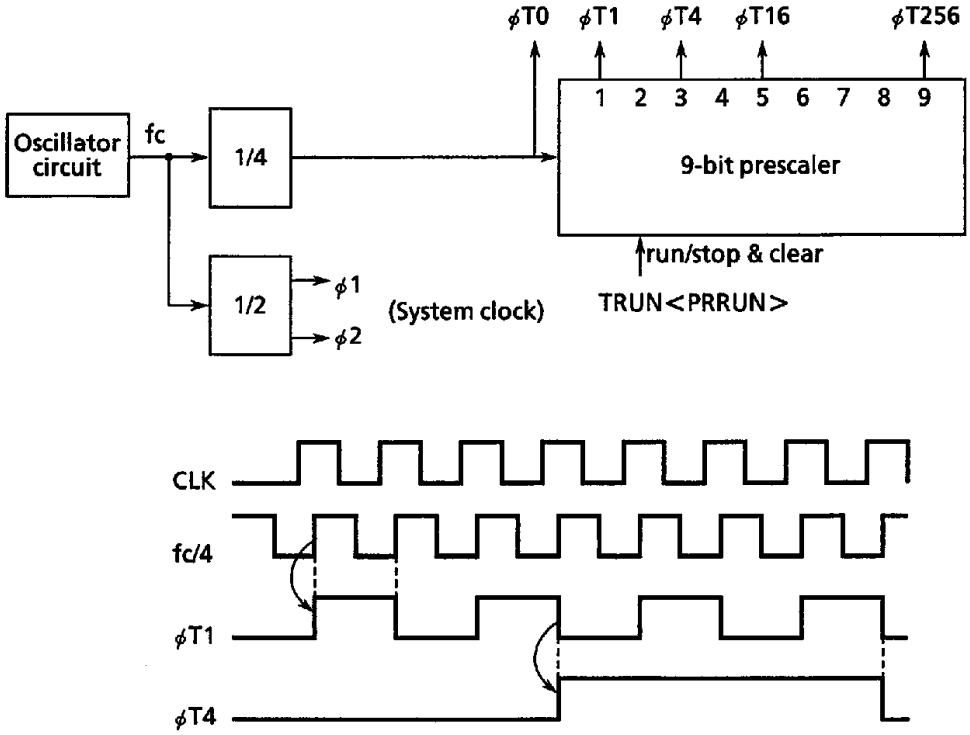


Figure 3.8 (2) Prescaler

② Up-counter

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

The input clock of timer 0 and timer 2 is selected from the external clock from TI0 pin (commonly used as P44) and TI2 pin (commonly used as P45 or INT0) and the three internal clocks ϕT1 (8/fc), ϕT4 (32/fc), and ϕT16 (128/fc), according to the set value of T01MOD and T23MOD.

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

When set to any other mode than 16-bit timer mode, the input clock is selected from the internal clocks ϕT1 (8/fc), ϕT16 (128/fc), and ϕT256 (2048/fc) as well as the comparator output (match detection signal) of timer 0 and timer 2, according to the set value of T01MOD and T23MOD.

Example: When T01MOD<T01M1,0>=01, the overflow output of timer 0 becomes the input clock of timer 1 (16-bit timer). When T01MOD<T01M1,0>=00 and T01MOD<T1CLK1,0>=01, ϕT1 (8/fc) becomes the input of timer 1.
Operation mode is also set by T01MOD and T23MOD. When reset, it is initialized to

T01MOD<T01M1, 0>=00 and T23MOD<T23M1, 0>=00, whereby the up-counter is placed in the 8-bit timer mode.

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

③ Timer register

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

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

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

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

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

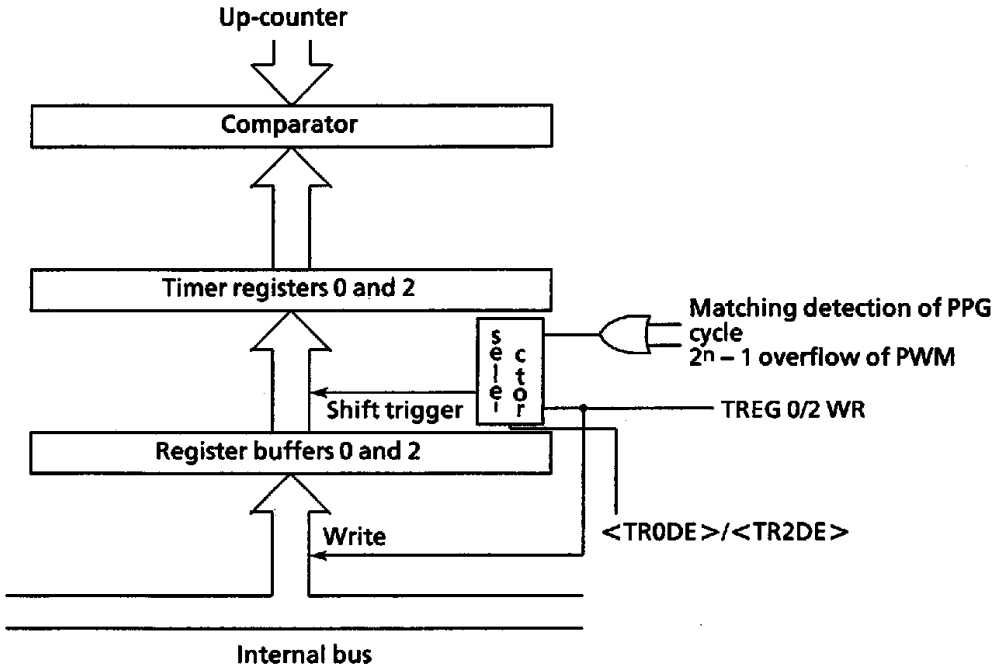


Figure 3.8 (3) Configuration of Timer Registers 0 and 2

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

The memory address of each timer register is as follows.

TREG0: FFD4H

TREG1: FFD5H

TREG2: FFD6H

TREG3: FFD7H

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

T01MOD (FFD8H)

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

Input clock of timer 0

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

Input clock of timer 1

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

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

00	——
01	2^6-1
10	2^7-1
11	2^8-1

Set the operation mode of timer 0 and 1.

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

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

T23MOD (FFD9H)

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

Timer 2 input clock

00	External clock T12
01	Internal clock ϕT1
10	Internal clock ϕT4
11	Internal clock ϕT16

Timer 3 input clock

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

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

00	——
01	2^6-1
10	2^7-1
11	2^8-1

Set the operation mode of timer 2 and timer 3

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

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

TFFCR (FFDAH)

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

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

0	Inversion by timer 0
1	Inversion by timer 1

Invert of timer flip-flop TFF1

0	Disabled
1	Enabled

Control timer flip-flop TFF1

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

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

0	Inversion by timer 2
1	Inversion by timer 3

Invert of timer flip-flop TFF3

0	Disabled
1	Enabled

Control of timer flip-flop TFF3

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

Figure 3.8 (6) 8-Bit Timer Flip-Flop Control Register (TFFCR)

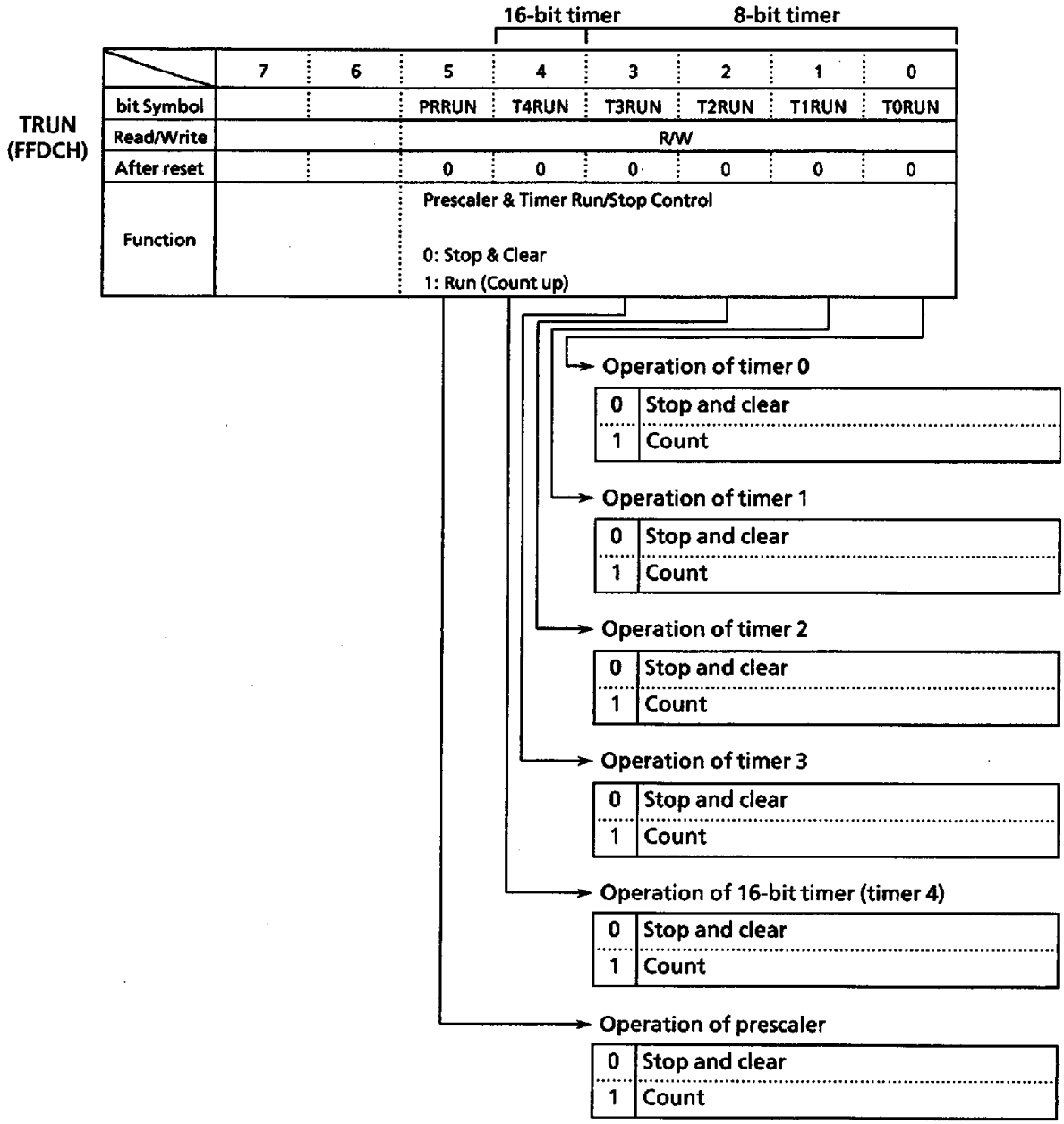


Figure 3.8 (7) Timer Operation Control Register (TRUN)

TRDC (FFDBH)

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

Control the double buffer of timer register 0

0	Disabled
1	Enabled

Control the double buffer of timer register 2

0	Disabled
1	Enabled

Control the double buffer of timer register 4

0	Disabled
1	Enabled

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

0	Timer 2 or 3
1	Timer 4

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

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

④ Comparator

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

⑤ Timer flip-flop (timer F/F)

The status of the timer flip-flop is inverted by the match detect signal (comparator output) of each interval timer and the value can be output to the timer output pins TO1 (also used as P40) and TO3 (also used as P41).

A timer F/F is provided for each pair of timer 0 and timer 1 as well as that of timer 2 and timer 3 and is called TFF1 and TFF3. TFF1 is output to TO1 pin, while TFF3 is output to TO3 pin.

The operation of 8-bit timers will be described below:

(1) 8-bit timer mode

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

① Generating interrupts in a fixed cycle

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

Example : To generate timer 1 interrupt every 40 microseconds at fc=16 MHz, set each register in the following manner.

	MSB 7 6 5 4 3 2 1 0 LSB	
TRUN ←	– – – – – – 0 –	Stop timer 1, and clear it to "0".
T01MOD←	0 0 X X 0 1 – –	Set the 8-bit timer mode, and select ϕT1 (0.5 μs @ fc = 16 MHz) as the input clock.
TREG1 ←	0 1 0 1 0 0 0 0	Set the timer register at 40 μs/ϕT1 = 50.
INTEH ←	X – – – – 1 – –	Enable INTT1.
TRUN ←	X X 1 – – – 1	Start timer 1 counting.

(Note) X ; Don't care – ; No change

Use the following table for selecting the input clock.

Table 3.8.(1) 8-Bit Timer Interrupt Cycle and Input Clock

Interrupt cycle (at fc = 16 MHz)	Resolution	Input clock
0.5μs ~ 128μs	0.5μs	φT1 (8/fc)
2μs ~ 512μs	2μs	φT4 (32/fc)
8μs ~ 2,048ms	8μs	φT16 (128/fc)
128μs ~ 32,768ms	128μs	φT256 (2048/fc)

Precautions for Using Timer 2

Timer 2 interrupt (INTT2) uses the same interrupt enable/disable flag INTEH<IET2> as used for A/D converter interrupt (INTAD). Change-over between them is executed by INTEH<ADIS>. When <ADIS> is set to "0", timer 2 interrupt will be enabled, disabling A/D converter interrupt.

② Generating a 50% duty square wave pulse

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

Example : To output a 3.0 us square wave pulse from TO1 pin at fc=16 MHz, set each register in the following procedures. Either timer 0 or timer 1 may be used, but this example uses timer 1.

	MSB 7 6 5 4 3 2 1 0 LSB	
TRUN ←	- - - - - - 0 -	Stop timer 1, and clear it to "0".
T01MOD←	0 0 X X 0 1 - -	Set the 8-bit timer mode, and select φT1 (0.5 μs @ fc = 16 MHz) as the input clock.
TREG1 ←	0 0 0 0 0 0 1 1	Set the timer register at 3.0 μs ÷ φT1 ÷ 2 = 3.
TFFCR ←	- - - - 1 0 1 1	Clear TFF1 to "0", and set to invert by the match detect signal from timer 1.
P4CR ←	- - - - - - - 1	} Select P40 as TO1 pin.
P4FR ←	- - X X - - - 1	
TRUN ←	X X 1 - - - 1 -	Start timer 1 counting.

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

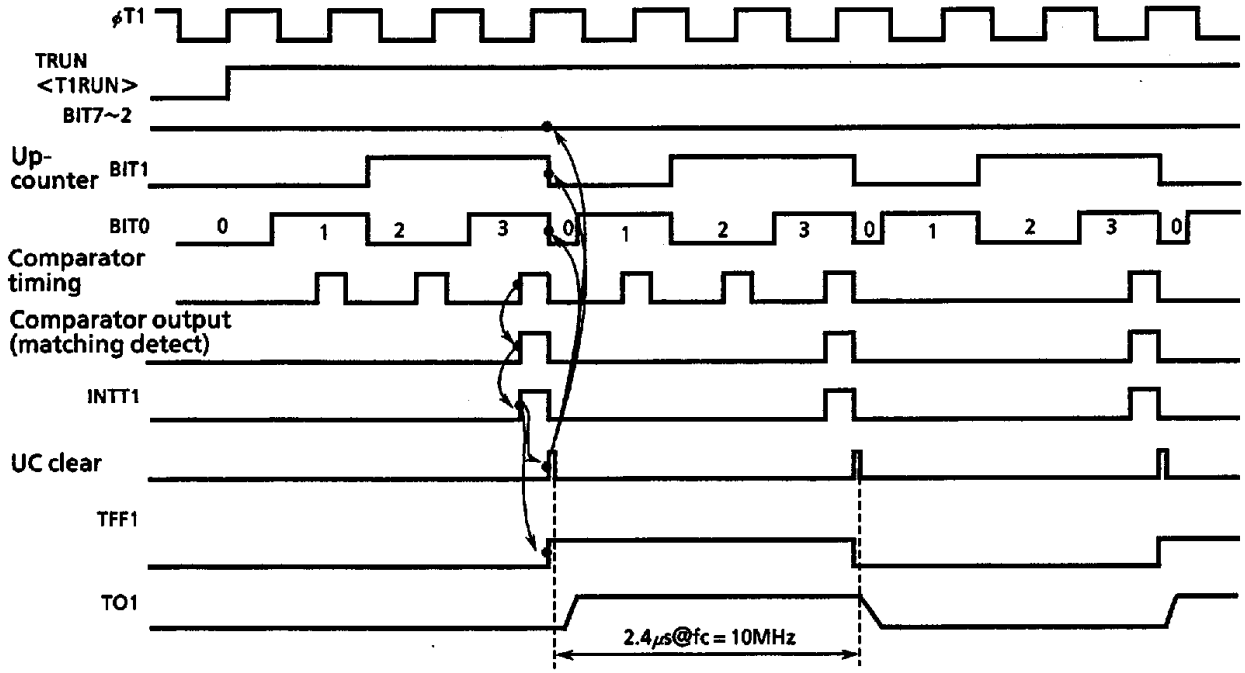


Figure 3.8 (9) Square Wave (50% Duty) Output Timing Chart

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

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

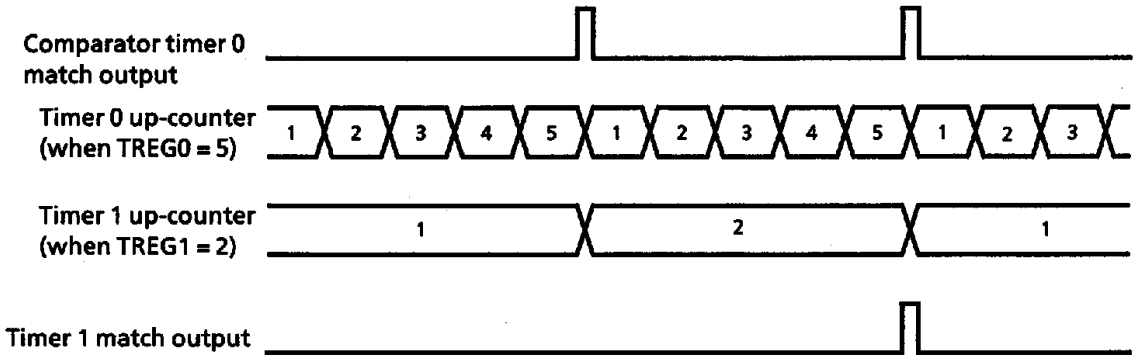


Figure 3.8 (10)

④ Output inversion with software

The value of timer flip-flop (timer F/F) can be inverted, independent of timer operation.

Writing "00" into TFFCR<FF1C1, 0> inverts the value of TFF1, and writing "00" into TFFCR<FF3C1,0> inverts TFF3.

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

The value of timer F/F can be initialized to "0" or "1", independent of timer operation.

For example, write "10" in TFFCR<FF1C1,0> to clear TFF1 to "0", while write "01" in TFFCR<FF3C1,0> to set TFF1 to "1".

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

(2) 16-bit timer mode

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

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

To make a 16-bit interval timer by cascade connecting timer 0 and timer 1, set timer 0/timer 1 mode register T01MOD<T01M1,0> to "00".

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

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

Interrupt cycle (@ fc = 16 MHz)	Resolution	Input clock
0.5μs ~ 32.768ms	0.5μs	φT1 (8/fc)
2μs ~ 131.072ms	2μs	φT4 (32/fc)
8μs ~ 524.288ms	8μs	φT16 (128/fc)

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

Setting example: To generate an interrupt INTT1 every 0.5 seconds at fc = 16 MHz, set the following values for timer registers TREG0 and TREG1.

When counting with input clock of φT16 (8μs @ 16 MHz)

0.5sec ÷ 8μs = 62500 = F424H

Therefore, set TREG1 = F4H and TREG0 = 24H, respectively.

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

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

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

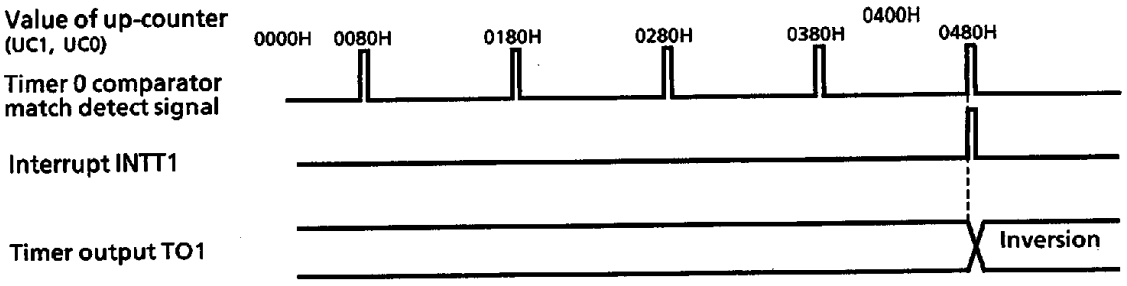


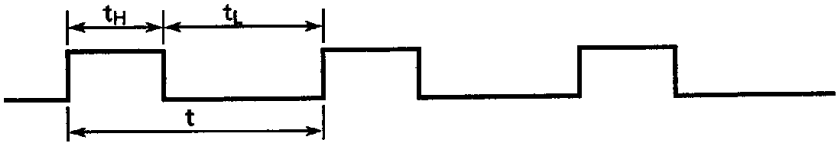
Figure 3.8 (11)

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

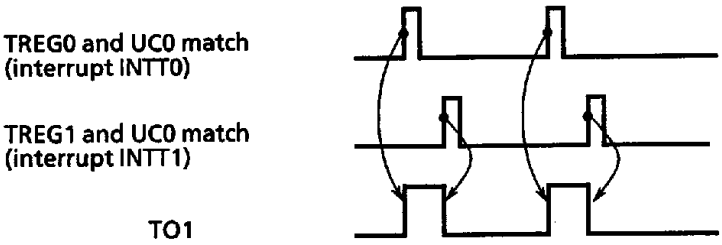
Square wave pulse can be generated at any frequency and duty by timer 0 or timer 2. The output pulse may be either low-active or high-active.

In this mode, timer 1 and timer 3 cannot be used.

Timer 0 outputs pulse to TO1 pin (also used as P40), and timer 2 outputs pulse to TO3 pin (also used as P41).



As an example, the case of timer 0 will be explained below. (Timer 2 also functions in the same way.)



In this mode, a programmable square wave is generated by inverting timer output each time the 8-bit up-counter (UC0) matches the timer registers TREG0 and TREG1.

However, it is required that the set value of TREG0 is smaller than that of TREG1.

Though the up-counter (UC1) of timer 1 cannot be used in this mode, timer 1 can be used for counting by setting TRUN<TIRUN>to 1.

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

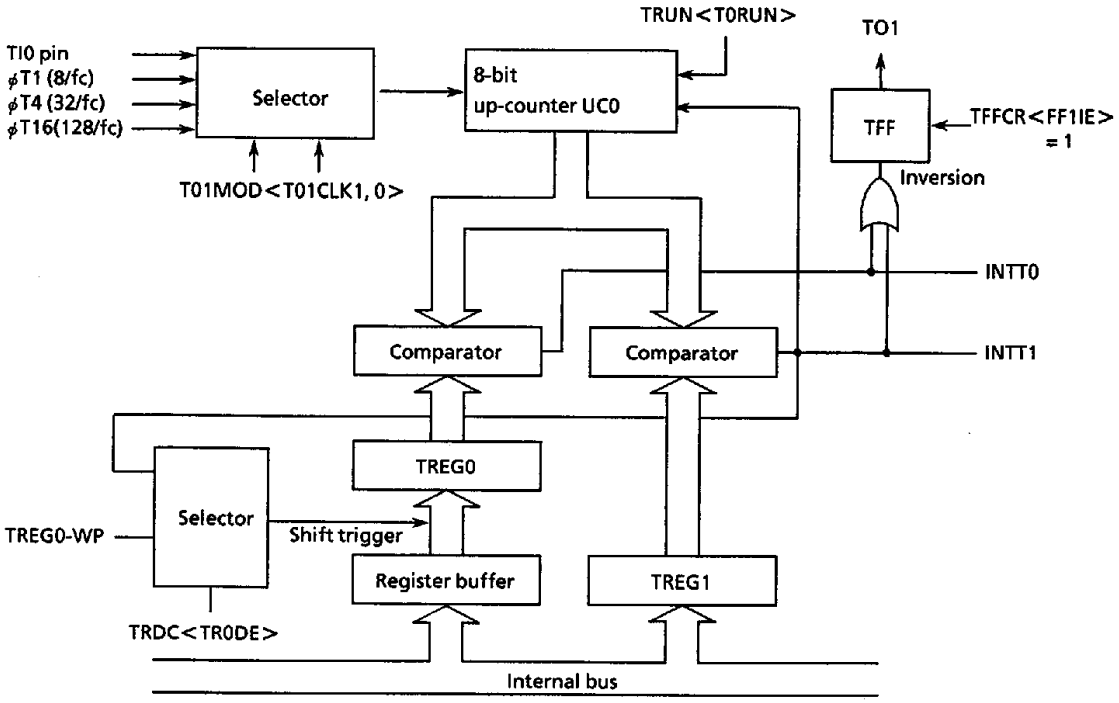
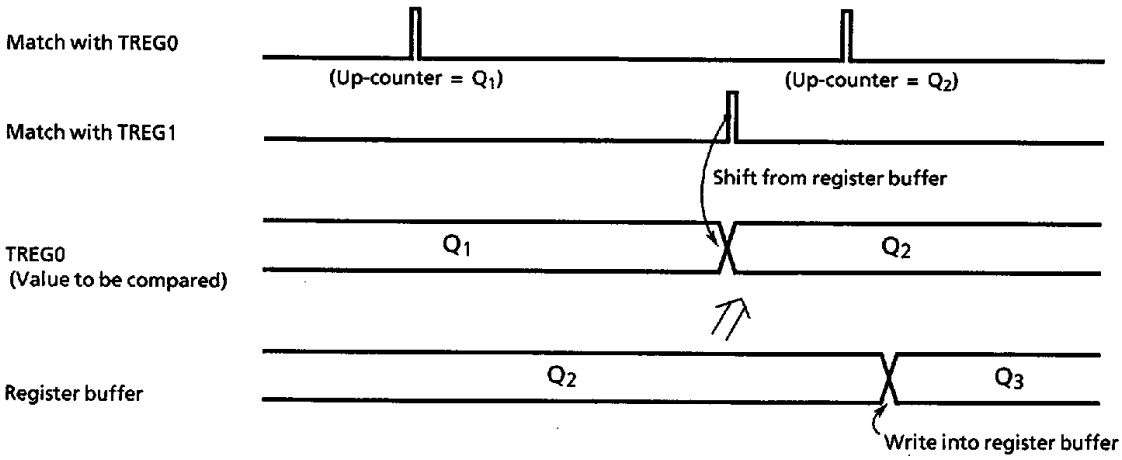


Figure 3.8 (12) Block Diagram of 8-Bit PPG Mode

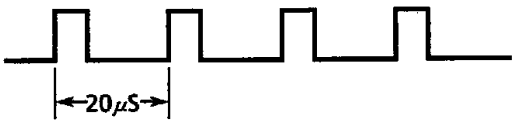
When the double buffer of TREG0 is enabled in this mode, the value of register buffer will be shifted in TREG0 each time TREG1 matches UC0.

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



varied).

Example: Generating 1/4 duty 50 KHz pulse (@ fc=16 MHz)



Calculate the value to be set for timer register.
To obtain the frequency 50 KHz, the pulse cycle t should be : 1/50 KHz = 20μs.
Given φT1=0.5μs (@ 16 Hz),

$20\mu s \div 0.5\mu s = 40$

Consequently, to set the timer register 1 (TREG1) to TREG1=40=28H
and then duty to 1/4, $t \times 1/4 = 20\mu s \times 1/4 = 5\mu s$

$5\mu s \div 0.5\mu s = 10$

Therefore, set timer register 0 (TREG0) to TREG0=10=0AH.

	MSB 7 6 5 4 3 2 1 LSB 0	
←		
TRUN ←	X X - - - - 0 0	Stop timer 0, and clear it to "0".
TO1MOD ←	1 0 X X X X 0 1	Set the 8-bit PPG mode, and select φT1 as input clock.
TFFCR ←	- - - - 0 1 1 x	Sets TFF1 and enable the inversion.
		Writing "10" provides negative logic pulse.
TREG0 ←	0 0 0 0 1 0 1 0	Write "0AH".
TREG1 ←	0 0 1 0 1 0 0 0	Write "28H".
P4CR ←	- - - - - - - 1	Set P40 as the TO1 pin.
P4FR ←	- - X X - - - 1	
TRUN ←	X X 1 - - - 1 1	Start timer 0 counting.

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

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

This mode is valid only for timer 0 and timer 2. In this mode, maximum two PWMs of 8-bit resolution (PWM0 and PWM2) can be output.

PWM pulse is output to TO1 pin (also used as P40) when using timer 0, and to TO3 pin (also used as P41) when using timer 2.

Timer 1 and timer 3 can also be used as 8-bit timer.

As an example, the case of timer 0 will be explained below. (Timer 2 also operates in the same way.)

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

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

(Set value of timer register) < (Set value of 2^n-1 counter overflow)
(Set value of timer register) ≠ 0

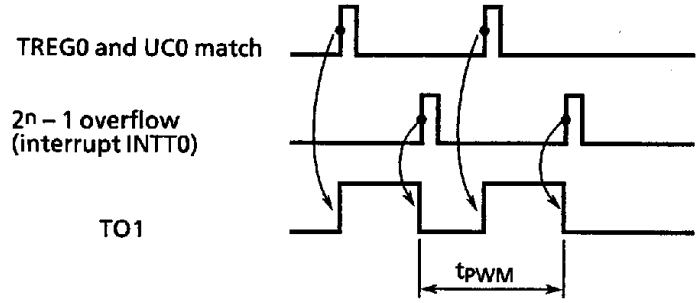


Figure 3.8 (13) shows the block diagram of this mode.

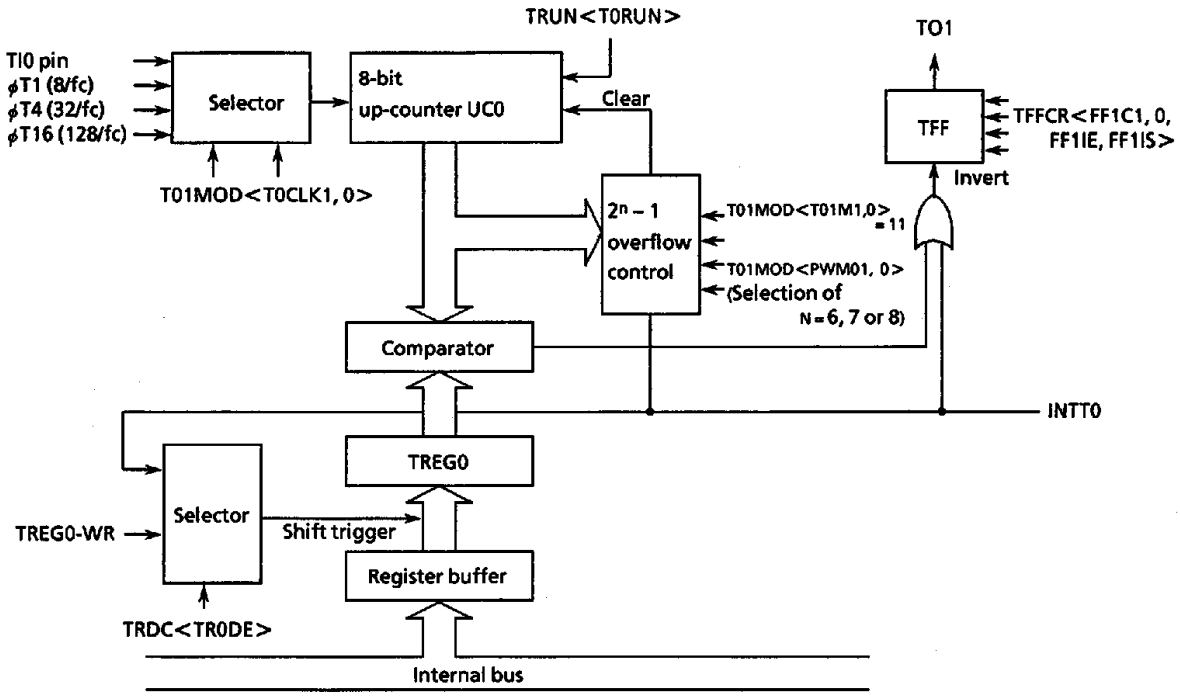
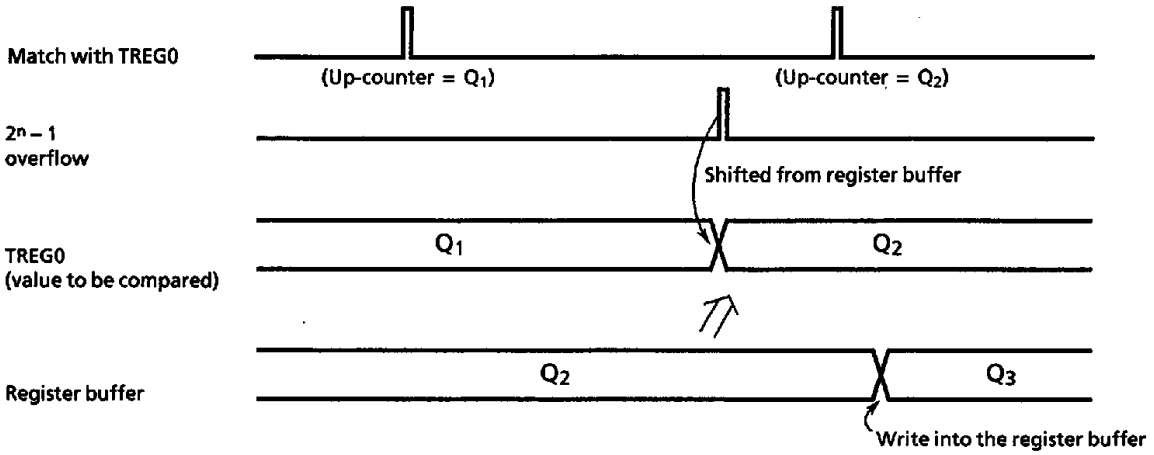


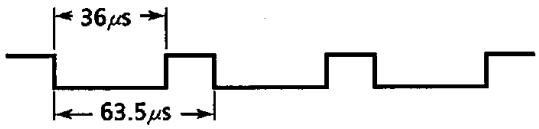
Figure 3.8 (13) Block Diagram of 8-Bit PWM Mode

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

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



Example: To output the following PWM waves to TO1 pin using timer 0 at fc = 16 MHz.



To realize 63.5μs of PWM cycle by φT1 = 0.5μs (@fc = 16 MHz),

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

Consequently, n should be set to 7.

As the period of low level is 36μs, for φT1 = 0.5μs,

set the following value for TREG0.

$36\mu s \div 0.5\mu s = 72 = 48H$

	MSB 7	6	5	4	3	2	1	LSB 0	
TRUN ←	X	X	-	-	-	-	-	0	Stop timer 0, and clear it to "0".
T01MOD ←	1	1	1	0	-	-	0	1	Set 8-bit PWM mode (cycle: $2^7 - 1$) and select φT1 as the input clock.
TFFCR ←	-	-	-	-	1	0	1	X	Clears TFF1 to enable the inversion.
TREG0 ←	0	1	0	0	1	0	0	0	Writes "48H".
P4CR ←	-	-	-	-	-	-	-	1	Set P40 as the TO1 pin.
P4FR ←	-	-	-	-	-	-	-	1	
TRUN ←	X	X	1	-	-	-	-	1	Start timer 0 counting.

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

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

	Formula	PWM cycle (@fc = 16 MHz)		
		φT1 (8/fc)	φT4 (32/fc)	φT16 (128/fc)
$2^6 - 1$	$(2^6 - 1) \times \phi Tn$	31.5μs	126μs	504μs
$2^7 - 1$	$(2^7 - 1) \times \phi Tn$	63.5μs	254μs	1.01ms
$2^8 - 1$	$(2^8 - 1) \times \phi Tn$	127μs	510μs	2.04ms

(5) Table 3.8 (4) shows the list of 8-bit timer modes.

Table 3.8 (4) Timer Mode Setting Registers

Register name	T01MOD (T23MOD)				TFFCR
Name of function in register	T01M (T23M)	PWM0 (PWM2)	T1CLK (T3CLK)	T0CLK (T2CLK)	FF1IS (FF31S)
Function	Timer mode	PWM cycle	Upper timer input clock	Lower timer input clock	Timer F/F invert signal select
16-bit timer mode	01	–	–	External clock, ϕT1, ϕT4, ϕT16 (01, 10, 11)	–
8-bit timer × 2 channels	00	–	Lower timer match: ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
8-bit PPG × 1channel	10	–	–	External clock, ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	–
8-bit PWM × 1channel	11	2^6-1, 2^7-1, 2^8-1 (01, 10, 11)	–	External clock, ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	–
8-bit timer × 1channel	11	–	ϕT1, ϕT4, ϕT16 (01, 10, 11)	–	Output disabled

(Note) – : Don't care

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

TMP90C845 contains one multifunctional 16-bit timer/event counter with the following operation modes.

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

Figure 3.9 (1) shows the block diagram of 16-bit timer/event counter.

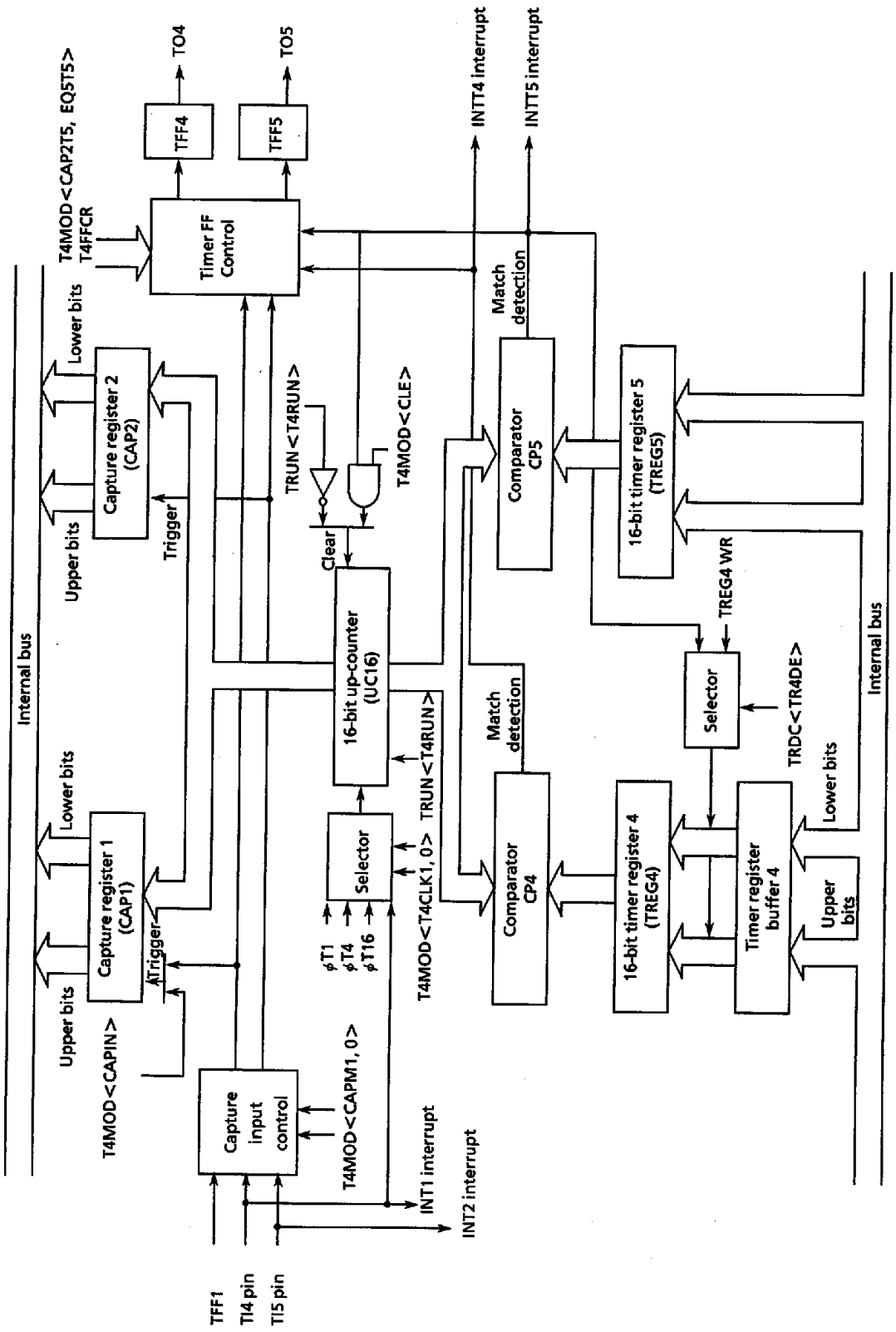


Figure 3.9 (1) Block Diagram of 16-Bit Timer/Event Counter (Timer 4)

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

Timer/event counter is controlled by 4 control registers: T4MOD, T4FFCR, TRUN, and TRDC. TRUN register includes 8-bit timer controller. For TRUN and TRDC registers, see Figure 3.8 (7) and Figure 3.8 (8).

T4MOD (FFE4H)

	7	6	5	4	3	2	1	0
bit Symbol	CAP2T5	EQ5T5	CAP1IN	CAPM1	CAPM0	CLE	T4CLK1	T4CLK0
Read/Write	R/W		W	R/W		R/W	R/W	
After reset	0	0	1	0	0	0	0	0
Function	TFF5 invert trigger 0: Disable 1: Enable		0: Soft-Capture 1: don't care	Capture timing 00: Disable INT1 occurs at rise edge. 01: T14↑ T15↑ INT1 occurs at rise edge. 10: T14↑ T14↓ INT1 occurs at fall edge. 11: TFF1↑ TFF1↓ INT1 occurs at rise edge.		1: UC16 Clear Enable	Timer 4 source clock 00: TI4 01: φT1 (8/fc) 10: φT4 (32/fc) 11: φT16 (128/fc)	

Timer 4 input clock

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

Clearing the up-counter UC16

0	Clear disable
1	Clear by match with TREG5.

Figure 3.9 (2) 16-Bit Timer/Event Counter (Timer 4) Controller/Mode Register (1/2)

T4MOD (FFE4H)

	7	6	5	4	3	2	1	0
bit Symbol	CAP2T5	EQ5T5	CAP1IN	CAPM1	CAPM0	CLE	T4CLK1	T4CLK0
Read/Write	R/W		W	R/W		R/W	R/W	
After reset	0	0	1	0	0	0	0	0
Function	TFF5 invert trigger 0: Disable 1: Enable		0: Soft-Capture 1: don't care	Capture timing 00: Disable INT1 occurs at rise edge. 01: TI4↑ TI5↑ INT1 occurs at rise edge. 10: TI4↑ TI4↓ INT1 occurs at fall edge. 11: TFF1↑ TFF1↓ INT1 occurs at rise edge.		1: UC16 Clear Enable	Timer 4 source clock 00: TI4 01: φT1 (8/fc) 10: φT4 (32/fc) 11: φT16 (128/fc)	

→ Capture control/INT1 interrupt control

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

→ Software capture trigger

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

→ Timer flip-flop 5 (TFF5) invert trigger

0	Trigger disable
1	Trigger enable

CAP2T5 : When the up-counter value is loaded to CAP2
EQ5T5 : When the up-counter matches TREG5

Figure 3.9 (2) 16-Bit Timer/Event Counter (Timer 4) Controller/Mode Register (2/2)

T4FFCR (FFE5H)

	7	6	5	4	3	2	1	0
bit Symbol	TFF5C1	TFF5C0	CAP2T4	CAP1T4	EQ5T4	EQ4T4	TFF4C1	TFF4C0
Read/Write	W		R/W				W	
After reset			0	0	0	0		
Function	00: Invert TFF5 01: Set TFF5 10: Clear TFF5 11: Don't care Always read as "11".		TFF4 invert trigger 0: Disable trigger 1: Enable trigger				00: Invert TFF4 01: Set TFF4 10: Clear TFF4 11: Don't care Always read as "11"	
			When the up-counter value is loaded to CAP2	When the up-counter value is loaded to CAP1	When the up-counter matches TREG5	When up-counter matches TREG4		

→ Timer flip-flop 4 (TFF4) control

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

→ Timer flip-flop 4 (TFF4) invert trigger

0	Trigger disable
1	Trigger enable

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

→ Timer flip-flop 5 (TFF5) control

00	Inverts the TFF5 value (software inversion).
01	Set TFF5 to "1".
10	Clear TFF5 to "0".
11	Don't care (Always read as "11".)

Figure 3.9 (3) 16-Bit Timer/Event Counter Timer Flip-flop Control Register

① Up-counter (UC16)

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

As the input clock, one of the internal clocks ϕT1 (8fc), ϕT4 (32fc), and ϕT16 (128fc) from 9-bit prescaler (also used as 8-bit timer), and external clock from TI4 pin (commonly used as P46/INT1 pin) can be selected. When reset, it will be initialized to <T4CLK1,0>=00 to select TI4 input mode. Counting, stop, or clearing of the counter is controlled by timer operation control register TRUN<T4RUN>.

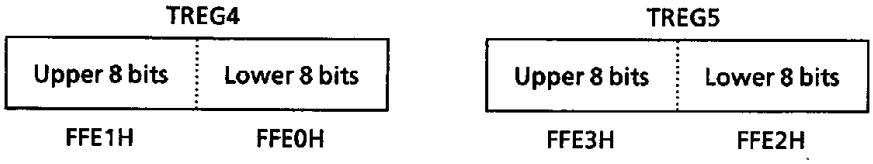
When clearing is enabled, up-counter UC16 will be cleared to zero each time it coincides matches the timer register TREG5. The "clear enable/disable" is set by T4MOD<CLE>.

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

② Timer registers (TREG4 and TREG5)

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

Setting data for timer register (TREG4 and TREG5) is executed using 16-bit transfer instruction or using 8-bit transfer instruction twice for lower 8 bits and upper 8 bits in order.



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

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

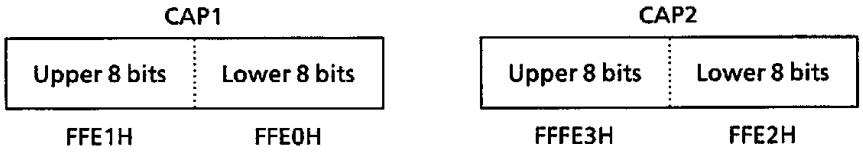
When reset, it will be initialized to <TR4DE>=0, whereby the double buffer is disabled. To use the double buffer, write data in the timer register to set to <TR4DE>=1 then write the following data in the register buffer.

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

③ Capture register (CAP1 and CAP2)

These 16-bit registers are used to hold the values of the up-counter UC16.

Data in the capture registers should be read by a 2-byte data load instruction or two 1-byte data load instruction, from the lower 8 bits followed by the upper 8 bits.



④ Capture input control circuit

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

- When T4MOD<CAPM1, 0> = 00

  Capture function is disabled. Disable is the default on reset.

- When T4MOD<CAPM1, 0> = 01

  Data is loaded to CAP1 at the rise edge of TI4 pin (commonly used as P46/INT1) input, while data is loaded to CAP2 at the rise edge of TI5 pin (commonly used as P47/INT2) input. (Time difference measurement)

- When T4MOD<CAPM1, 0> = 10

  Data is loaded to CAP1 at the rise edge of TI4 pin input, while to CAP2 at the fall edge. Only in this setting, interrupt INT1 occurs at fall edge. (Pulse width measurement)

- When T4MOD<CAP1, 0> = 11

  Data is loaded to CAP1 at the rise edge of timer flip-flop TFF1, while to CAP2 at the fall edge. (Frequency measurement)

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

⑤ Comparator (CP4 and CP5)

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

⑥ Timer flip-flop (TFF4)

This flip-flop is inverted by the match detect signal from the comparators (CP4 and CP5) and the latch signals to the capture registers (CAP1 and CAP2). Disable/enable of inversion can be set for each element by T4FFCR<CAP2T4, CAP1T4, EQ5T4, EQ4T4>. TFF4 will be inverted when "00" is written in T4FFCR<TFF4C1,0>. Also it is set to "1" when "10" is written, and cleared to "0" when "10" is written. The value of TFF4 can be output to the timer output pin TO4 (commonly used as P42).

⑦ Timer flip-flop (TFF5)

This flip-flop is inverted by the match detect signal from the comparator CP5 and the latch signal to the capture register CAP2. TFF5 will be inverted when "00" is written in T4FFCR<TFF5C1,0>. Also it is set to "1" when "10" is written, and cleared to "0" when "10" is written. The value of TFF5 can be output to the timer output pin TO5 (commonly used as P43).

(1) 16-bit timer mode

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

```
TRUN  ←  - - - 0 - - - -        Stop timer 4.
INTEL ←  0 - 1 - - - - -        Enable INTT5 and disable INTT4.

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

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

(2) 16-bit event counter mode

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

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

TI4 pin can also be used as P46/INT1.

```
TRUN  ←  - - - 0 - - - -        Stop timer 4.
P4FR  ←  - * - - - - - -        * = 0 :TI4 input pulse is square wave
                                * = 1 : TI4 input pulse is sine wave (zero-cross)

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

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

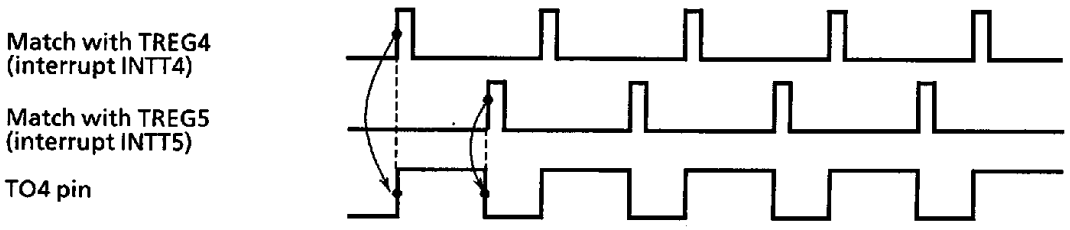
(3) 16-bit programmable pulse generation (PPG) mode

The PPG mode is entered by inversion of the timer flip-flop TFF4 that is to be enabled by the match of the up-counter UC16 with the timer register TREG4 or 5 and to be output to TO4 (also used as P42). In this mode, the following conditions must be satisfied.

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

```
TRUN  ←  - - - 0 - - - -        Stop timer 4.
TREG4 ←  **** **** **** ****    Set the duty.
TREG5 ←  **** **** **** ****    Set the cycle.
T4FFCR←  1 1 0 0 1 1 0 0        Set the TFF4 inversion to be effected by match with TREG4 or
                                TREG5. Initialize TFF4 to "0".
T4MOD ←  0 0 1 0 0 1 * *        Select the internal clock for the input, and disable the capture
                   (**=01,10,11) function.
P4CR  ←  - - - - - 1 - -
                   (**=00,01,10) Assign P42 as TO4.
P4FR  ←  - - - - - 1 - -
TRUN  ←  - - 1 1 - - - -        Start timer 4.
```

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



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

(4) Application examples of capture function

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

① One-shot pulse output from external trigger pulse

② Frequency measurement

③ Pulse width measurement

④ Time difference measurement

① One-shot pulse output from external trigger pulse

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

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

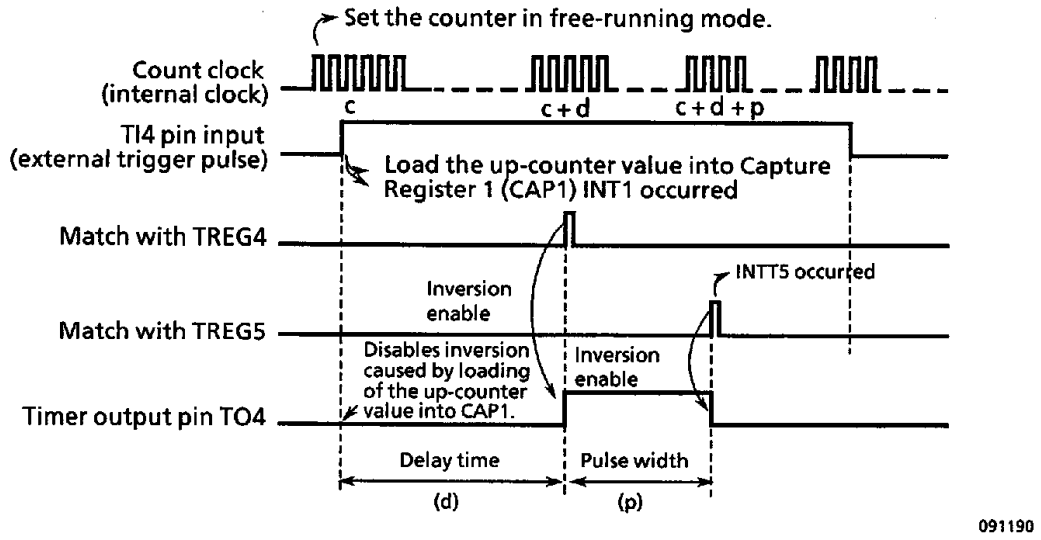


Figure 3.9 (4) One-Shot Pulse Output (with Delay)

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

Main setting

```
T4MOD ← 0 0 1 0 1 0 0 1   Keep counting (Free-runnig)
                          Count with φT1.
T4FFCR← 1 1 0 0 0 0 0 0   Load the up-counter value into CAP1 at the rise edge
                          of T14 pin input.
                          Clear TFF4 to zero.
                          Disable TFF4 inversion.
P4CR  ← - - - - - 1 - -
        (**=00,01,10)     Select P42 as the TO4 pin.
P4FR  ← - - - - - 1 - -
INTEL ← 0 1 0 - - - - -   Enable INT1, and disable INTT4 and INTT5.
TRUN  ← - - 1 1 - - - -   Start timer 4.
```

Setting of INT1

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

Setting of INT5

```
T4FFCR← 1 1 - - 0 0 - -
                    └──────→ Disable TFF4 inversion when the up-counter value
                             matches TREG4 or 5.
INTEL ← - - 0 - - - - -      Disable INTT5.
```

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

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

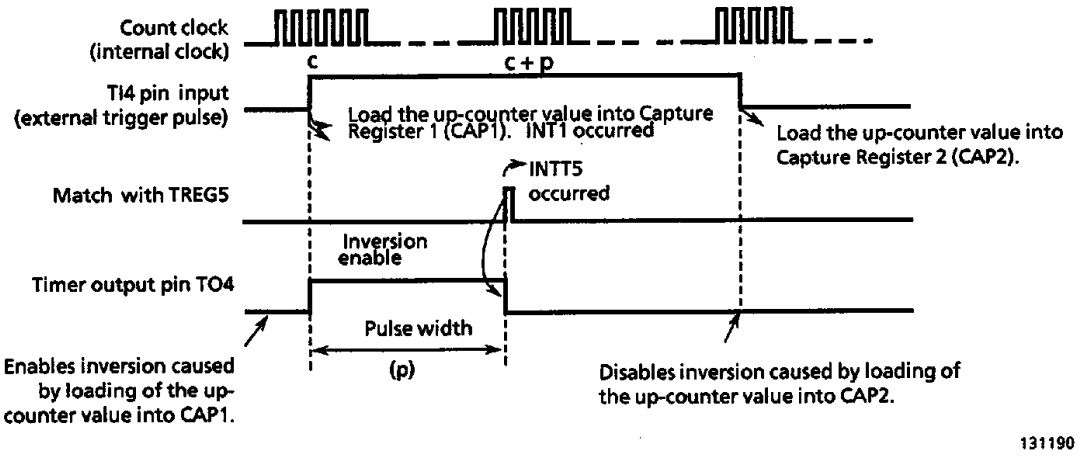


Figure 3.9 (5) One-Shot Pulse Output (without Delay)

② Frequency measurement

The frequency of the external clock can be measured in this mode. The clock is input through the TI4 pin, and its frequency is measured by the 8-bit timers (Timer 0 and Timer 1) and the 16-bit timer/event counter (Timer 4).

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

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

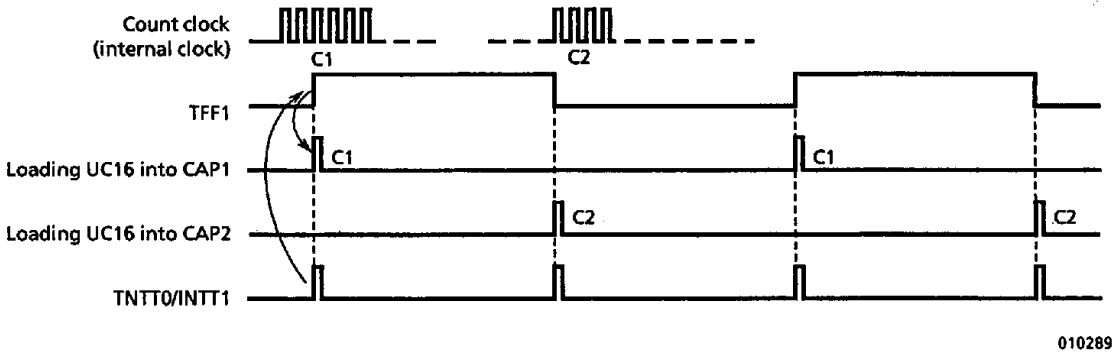


Figure 3.9 (6) Frequency Measurement

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

③ Pulse width measurement

This mode allows to measure the "H" level width of an external pulse. While keeping the 16-bit timer/event counter counting (free-running) with the internal clock input, the external pulse is input through the TI4 pin. Then the capture function is used to load the UC16 values into CAP1 and CAP2 at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT1 occurs at the falling edge of TI4.

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

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

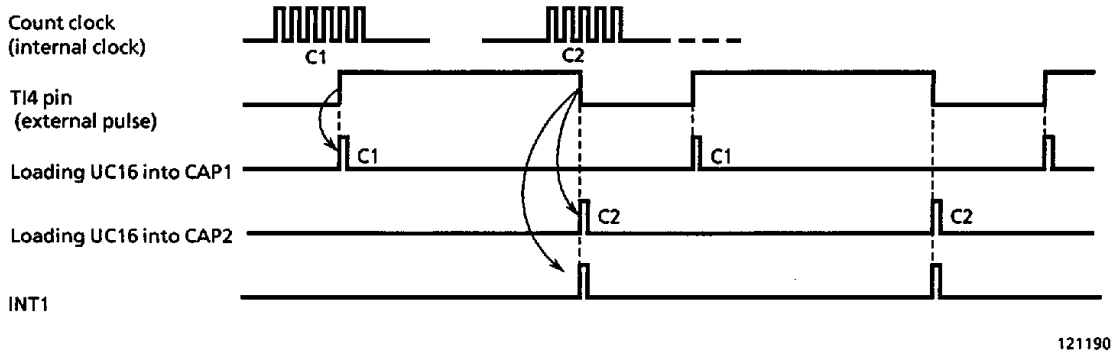


Figure 3.9 (7) Pulse Width Measurement

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

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

④ Time difference measurement

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

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

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

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

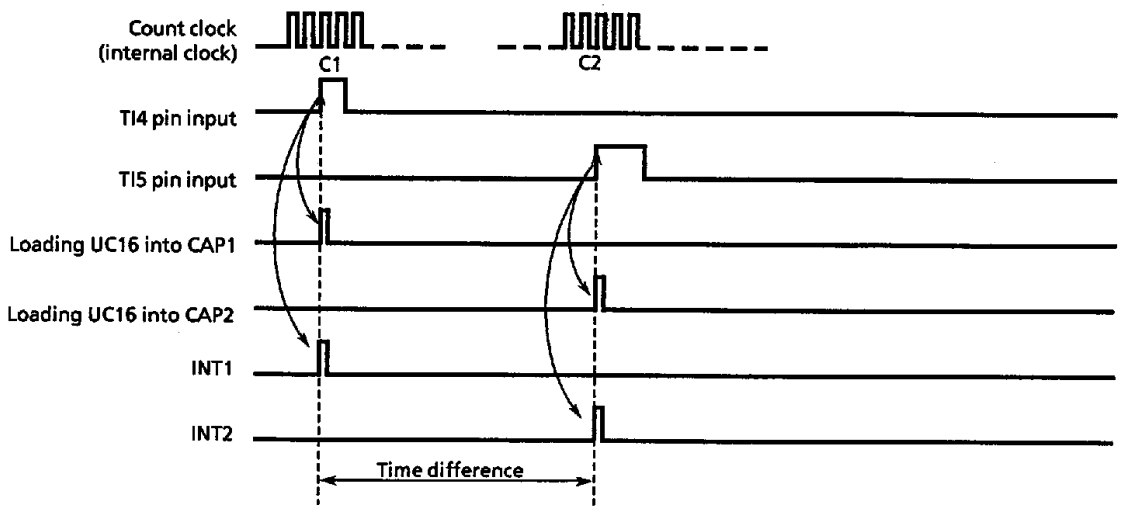


Figure 3.9 (8) Time Difference Measurement

3.10 Stepping Motor Control/Pattern Generation Port (P6 and P7)

TMP90C845 contains 2 channels (M0 and M1) of 4-bit hardware stepping motor control/pattern generation ports (herein after called SMC) which actuate in synchronization with the (8-bit/16-bit) timers. The SMCs (M0 and M1) are shared by 4-bit I/O ports P6 and P7.

Channel 0 (M0) is synchronous with 8-bit timer 0 or timer 1, and channel 1 (M1) is synchronous with 8-bit timer 2 or timer 3 or 16-bit timer 4, to update the output.

The SMC ports are controlled by three control registers P67CR, P67FR, and TRDC and can select either stepping motor control mode or pattern generation mode.

3.10.1 Control Registers

(1) Ports 6 and 7 I/O selection register (P67CR)

This register specifies either input or output for each bit of the 4-bit I/O ports 6 and 7. When reset, all bits of P67CR are cleared to "0", so that port 6 and port 7 function as input ports. To use port 6 and port 7 as SMC, set all bits of P67CR to "1", specifying them as output pins.

P67CR is a write-only register and so cannot be read.

(2) Ports 6 and 7 function control register (P67FR)

This register is used for setting port 6 and port 7 as SMC. To use port 6 and port 7 as SMC, set P67FR<M0S>/<M1S> to "1".

With P67<PAT0>/<PAT1>, set SMC in either 8-bit write mode or 4-bit write mode. In 4-bit write mode, writing the SMC is executed only on the 4-bit shift alternate register, and SMC functions as a pattern generation port.

To use SMC as a stepping motor control port, select the method of excitation and the method to control the direction of rotation by P67FR<M0M>/<M1M> and P67FR<CCW0>/<CCW1>, respectively.

(3) Selecting the channel 1 synchronizing timer (TRDC)

With TRDC<M1T>, select a timer which synchronizes SMC channel 1 (M1). When <M1T>=0, M1 is synchronous with timer 2 or timer 3, while when <M1T>=1 it is synchronous with timer 4.

(4) Port 6

This is a 4-bit I/O port allocated to address FFCB.

The lower 4 bits are assigned as port 6, while the upper 4 bits function as the shifter alternate register SA6 which is used in pattern generation mode or to drive the stepping motor by 1-2 excitation.

(5) Port 7

This is a 4-bit I/O port and allocated to address FFCC.

The lower 4 bits are assigned as port 7, while the upper 4 bits function as the shifter alternate register SA7 which is used in pattern generation mode or to drive the stepping motor by 1-2 excitation.

F67CR (FFCDH)

	7	6	5	4	3	2	1	0
bit Symbol	P73C	P72C	P71C	P70C	P63C	P62C	P61C	P60C
Read/Write	W				W			
After reset	0	0	0	0	0	0	0	0
Function	0 : In		1 : Out		0 : In		1 : Out	

Select Port 6 I/O (on bit basis)

0	Input
1	Output

Select Port 7 I/O (on bit basis)

0	Input
1	Output

121190

Figure 3.10 (1) Port 6 and Port 7 I/O Selection Register (P67CR)

F67FR (FFD0H)

	7	6	5	4	3	2	1	0
bit Symbol	PAT1	CCW1	M1M	M1S	PAT0	CCW0	M0M	M0S
Read/Write	R/W				R/W			
After reset	0	0	0	0	0	0	0	0
Function	0: 8-bit write 1: 4-bit write	0:Normal rotation 1:Reverse rotation	0:1 excitation or 2 excitation 1:1-2 excitation	0:General-purpose port 1:SMC	0: 8-bit write 1: 4-bit write	0:Normal rotation 1:Reverse rotation	0:1 excitation or 2 excitation 1: 1-2 excitation	0:General-purpose port 1:SMC

Selecting the function of Port 6

0	General-purpose port
1	Stepping motor control/pattern generation port 0 (M0)

Stepping motor control port 0 mode (excitation)

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

Rotating direction of stepping motor control port 0

0	Normal rotation
1	Reverse rotation

Selecting port 6 write mode

0	8-bit write
1	4-bit write (Pattern generation mode) (Only shifter alternate register can be written.)

Figure 3.10 (2a) Port 6 and Port 7 Function Control Register (P67FR)

F67FR (FFD0H)

	7	6	5	4	3	2	1	0
bit Symbol	PAT1	CCW1	M1M	M1S	PAT0	CCW0	M0M	M0S
Read/Write	R/W				R/W			
After reset	0	0	0	0	0	0	0	0
Function	0: 8-bit write 1: 4-bit write	0: Normal rotation 1: Reverse rotation	0: 1 excitation or 2 excitation 1: 1-2 excitation	0: General-purpose port 1: SMC	0: 8-bit write 1: 4-bit write	0: Normal rotation 1: Reverse rotation	0: 1 excitation or 2 excitation 1: 1-2 excitation	0: General-purpose port 1: SMC

Selecting the function of port 7

0	General-purpose port
1	Stepping motor control/pattern generation port 1 (M1)

Stepping motor control port 1 mode (excitation)

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

Rotating direction of stepping motor control port 1

0	Normal rotation
1	Reverse rotation

Selecting port 7 write mode

0	8-bit write
1	4-bit write (Pattern generation mode) (Only shifter alternate register can be written.)

Figure 3.10 (2b) Port 6 and Port 7 Function Control Register (P67FR)

TRDC (FFDBH)

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

Explained in 3.8 "8-Bit Timer"

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

0	Timer 2 or 3
1	Timer 4

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

Figure 3.10 (3) Timer Register Double Buffer Control Register (TRDC)

P6 (FFCBH)

	7	6	5	4	3	2	1	0
bit Symbol	SA63	SA62	SA61	SA60	P63	P62	P61	P60
Read/Write	W				R/W			
After reset	Undefined				Input mode			
	Shift alternate register 0 for stepping motor control				Shared by stepping motor control port 0			

Port 6

Shift alternate register 0 for stepping motor control

P7 (FFCCH)

	7	6	5	4	3	2	1	0
bit Symbol	SA73	SA72	SA71	SA70	P73	P72	P71	P70
Read/Write	W				R/W			
After reset	Undefined				Input mode			
	Shift alternate register 1 for stepping motor control				Shared by stepping motor control port 1			

Port 7

Shift alternate register 1 for stepping motor control

Figure 3.10 (4) Port 6 and Port 7

3.10.2 Pattern Generation Mode

SMC functions as a pattern generation port according to the setting of P67FR<PAT1>/<PAT0>. In this mode, because writing from CPU is executed only on the shifter alternate register, writing ports 6 and 7 can be done during the interrupt operation of the timer for shift trigger and a pattern can be output, synchronous with the timer.

In this mode, P67FR<M1M>/<M0M> must be always set to "1".

Figure 3.10 (5) shows the block diagram of this mode.

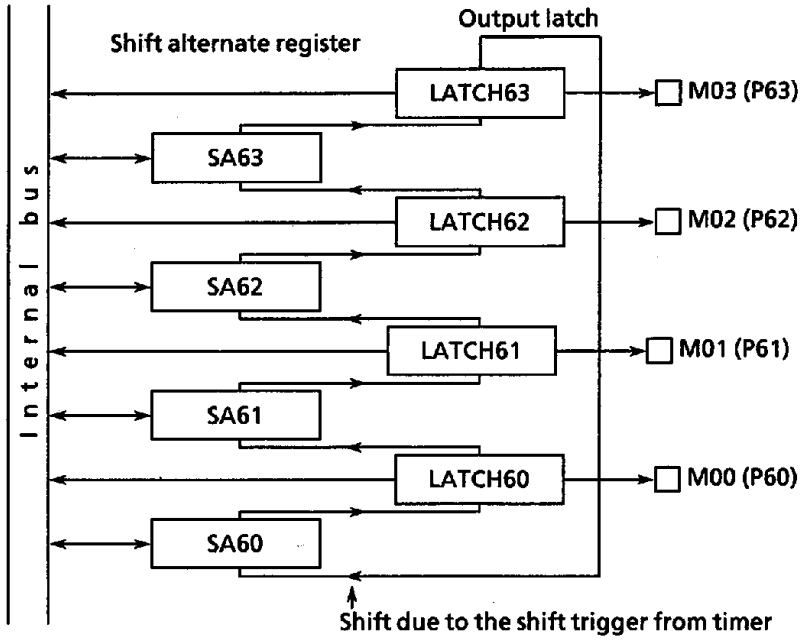


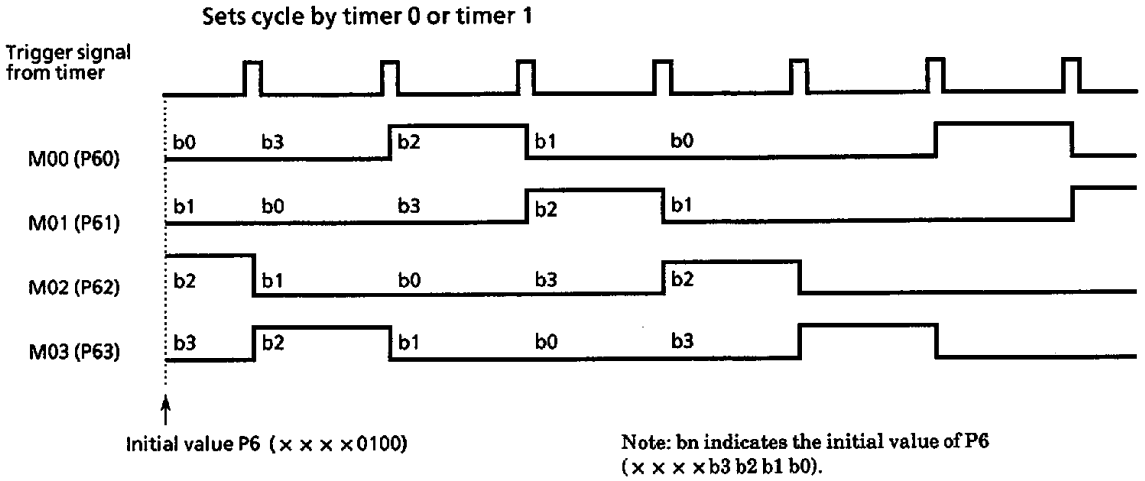
Figure 3.10 (5) Pattern Generation Mode Block Diagram (Port 6)

In this pattern generation mode, only writing the output latch is disabled by hardware, but other functions do the same operation as 1-2 excitation in stepping motor control port mode. Accordingly, the data shifted by trigger signal from a timer must be written before the next trigger signal is output.

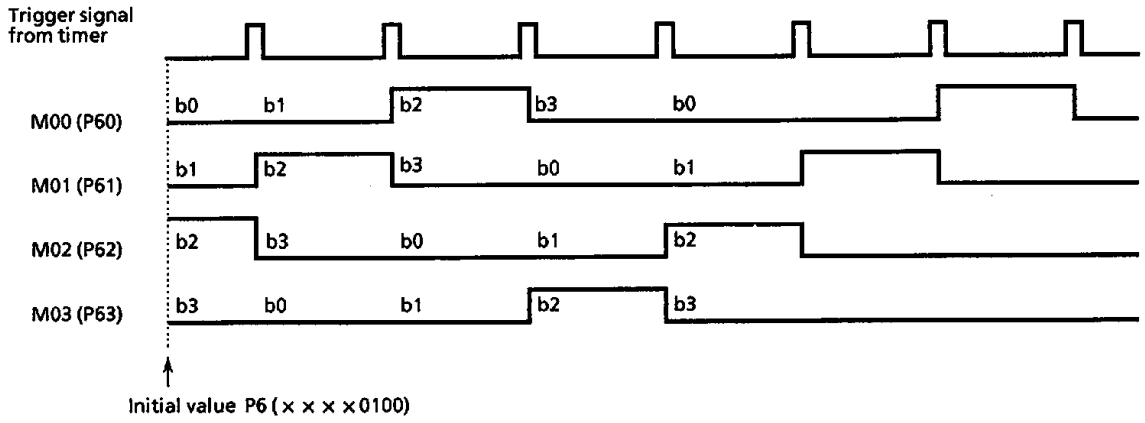
3.10.3 Stepping Motor Control Mode

(1) 4-phase 1-Step/2-Step Excitation

Figure 3.10 (6) and Figure 3.10 (7) show the output waveforms of 4-phase 1 excitation and 4-phase 2 excitation, respectively when channel 0 is selected.



① Normal Rotation



② Reverse Rotation

Figure 3.10 (6) Output Waveforms of 4-Phase 1-step Excitation (Normal Rotation and Reverse Rotation)

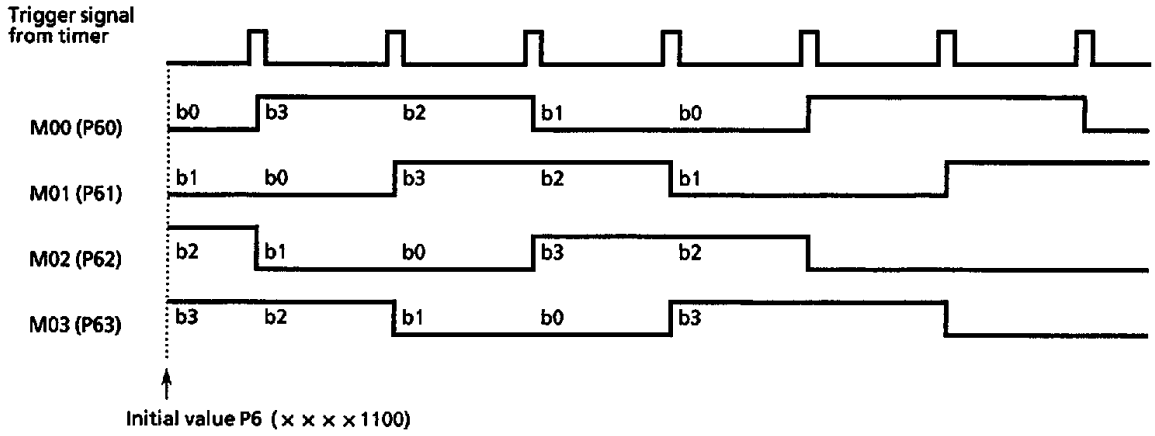


Figure 3.10 (7) Output Waveforms of 4-Phase 2-step Excitation (Normal Rotation)

The operation when channel 0 is selected is explained below.

The output latch of M0 (also used as P6) is shifted at the rising edge of the trigger signal from the timer to be output to the port.

The direction of shift is specified by P67FR<CCW0>: Normal rotation (M00→M01→M02→M03) when <CCW0> is set to "0"; reverse rotation (M00←M01←M02←M03) when "1". 4-phase 1-step excitation will be selected when only one bit is set to "1" during the initialization of port 6, while 4-phase 2-step excitation will be selected when two consecutive bits are set to "1".

Figure 3.10 (8) shows the block diagram.

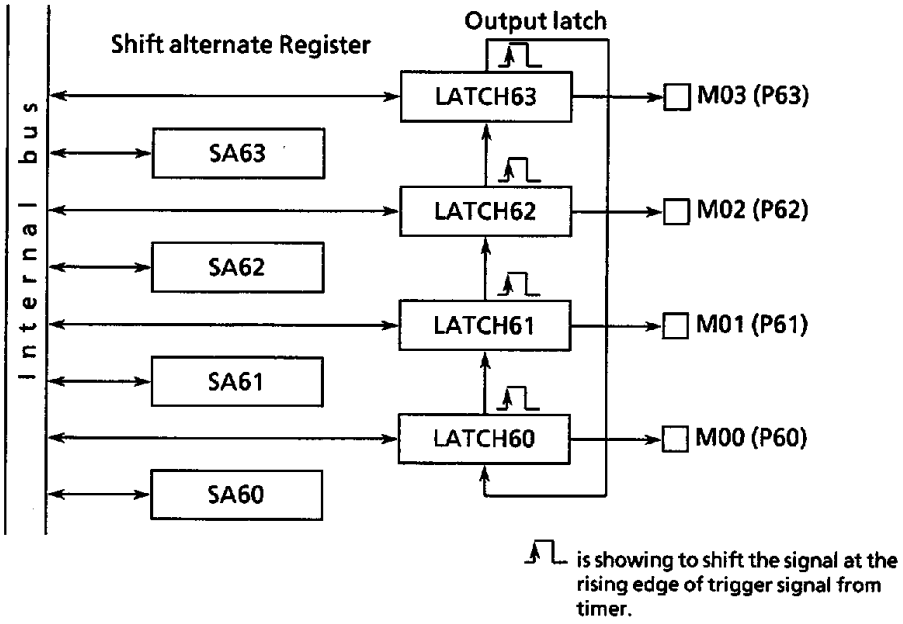
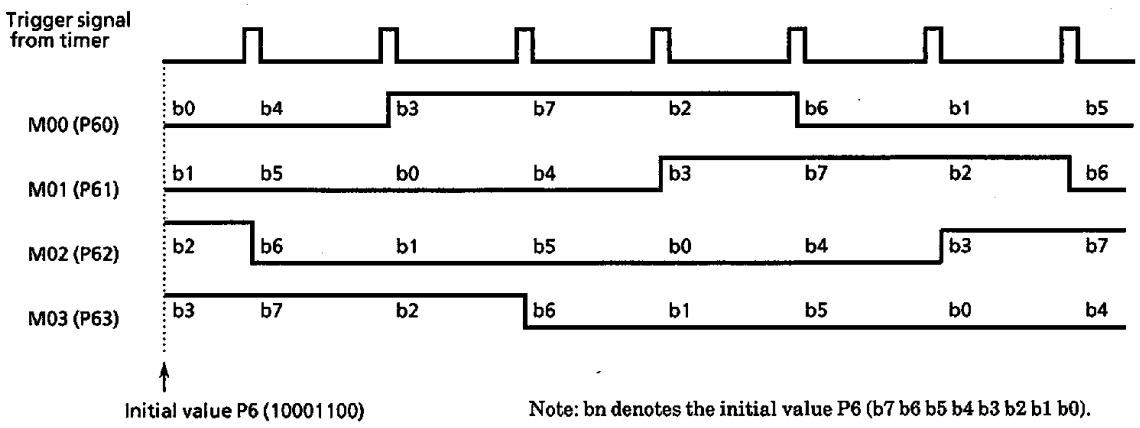


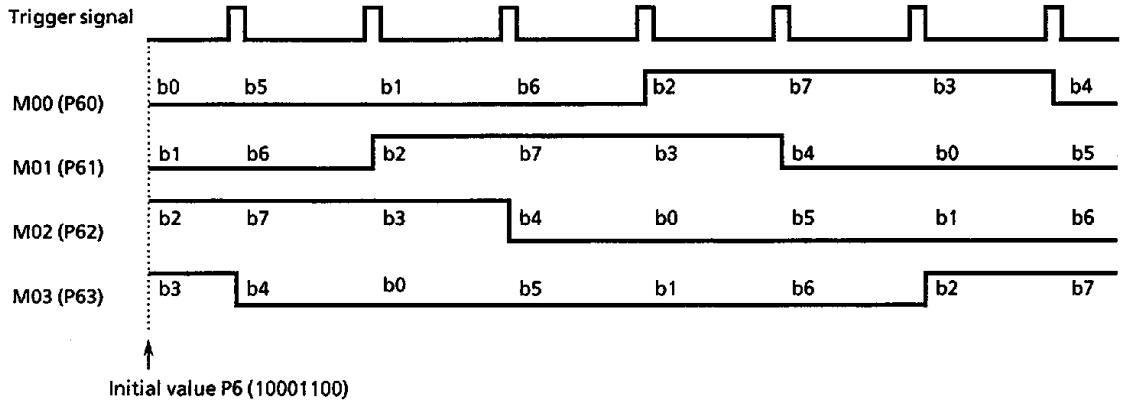
Figure 3.10 (8) Block Diagram of 4-Phase 1-step Excitation/2-step Excitation (Normal Rotation)

(2) 4-Phase 1-2 step Excitation

Figure 3.10 (9) shows the output waveforms of 4-phase 1-2 step excitation when channel 0 is selected.



① Normal Rotation



② Reverse Rotation

Figure 3.10 (9) Output Waveforms of 4-Phase 1-2 step Excitation (Normal Rotation and Reverse Rotation)

The initialization for 4-phase 1-2 step excitation is as follows.

By rearranging the initial value "b7 b6 b5 b4 b3 b2 b1 b0" to "b3 b7 b2 b6 b1 b5 b0 b4", the consecutive 3 bits are set to "1" and other bits are set to "0" (positive logic).

For example, if b3, b7, and b2 are set to "1", the initial value becomes "10001100", obtaining the output waveforms as shown in Figure 3.10 (9).

To get an output waveform of negative logic, set values 1's and 0's of the initial value should be inverted. For example, to change the output waveform shown in Figure 3.10 (9) into negative logic, change the initial value to "01110011".

The operation will be explained below for channel 0.

The output latch of M0 (shared by P6) and the shifter alternate register (SA6) for stepping motor control are shifted at the rising edge of trigger signal from the timer to be output to the port. The direction of shift is set by P67FR<CCW0>.

Fig 3.10 (10) shows the block diagram.

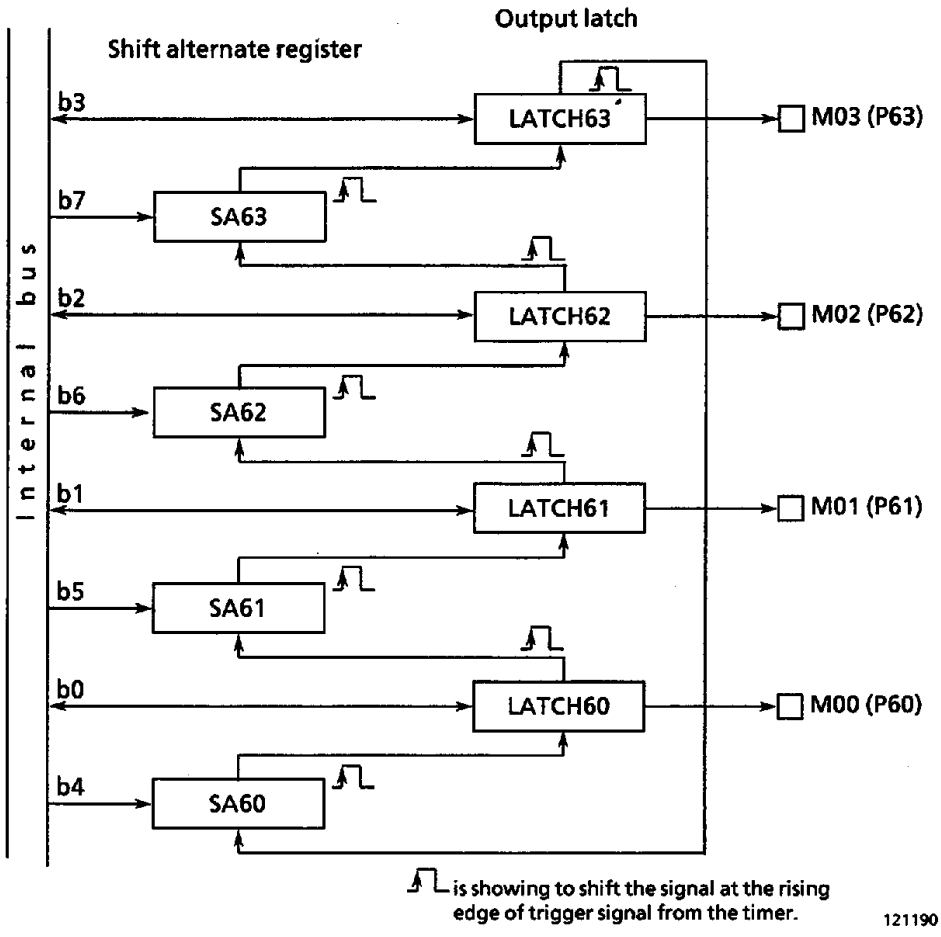


Figure 3.10 (10) Block Diagram of 4-Phase 1-2 step Excitation (Normal Rotation)

Setting example: To drive channel 0 (M0) by 4-phase 1-2 step excitation (normal rotation) when timer 0 is selected, set each register as follows.

```
          7 6 5 4 3 2 1 0
TRUN   ← - - - - - - - - 0      Stop timer 0, and clear it to zero.
T01MOD ← 0 0 X X - - 0 1        Set 8-bit timer mode and select φT1 as the input clock of timer 0.

TFFCR  ← - - - - 1 0 1 0        Clear TFF1 to zero and enable the inversion trigger by timer 0.

TREG0  ← * * * * * * * *        Set the cycle in timer register.
P67CR  ← - - - - 1 1 1 1        Set all P6 bits to the output mode.
P67FR  ← - - - - 0 0 1 1        Select 4-phase 1-2 step excitation mode and normal rotation .
P6     ← 1 0 0 0 1 1 0 0        Set an initial value.
TRUN   ← - - 1 - - - - 1        Start timer 0.
```

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

3.10.4 Trigger Signal From Timer

The trigger signal from the timer which is used by SMC is not equal to the reverse trigger signal of each timer flip-flop (TFF1, TFF3, TFF4, and TFF5) and differs as shown in Table 3.10 (1) depending on the operation mode of the timer.

3.10 (1) 8-Bit Timers 0 and 1 (Same for timers 2 and 3)

	TFF1 inversion	SMC shift
8-bit timer mode	Selected by TFFCR0 (FF1IS) when the up-counter value matches TREG0 or TREG1 value.	←
16-bit timer mode	When the up-counter value matches with both TREG0 and TREG1 values (The value of up-counter = TREG1*2^8 + TREG0)	←
PPG output mode	When the up-counter value matches with both TREG0 and TREG1	When the up-counter value matches TREG1 value (PPG cycle)
PWM output mode	When the up-counter value matches TREG0 value at PWM cycle.	Trigger signal for SMC shift is not output.

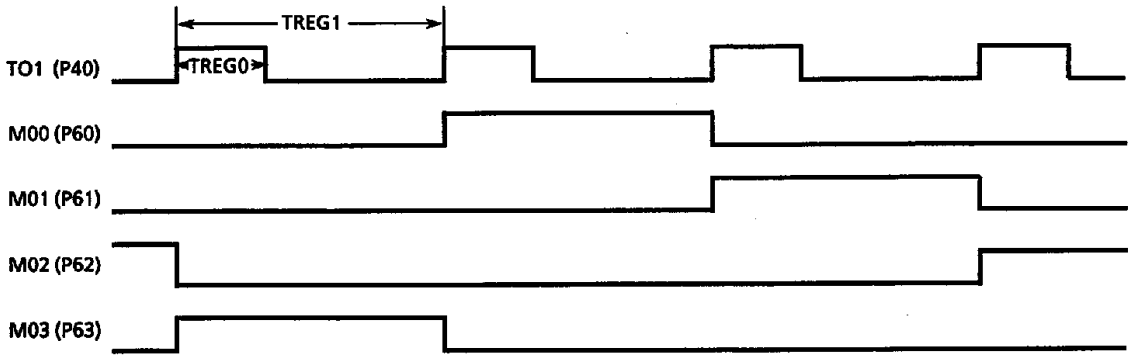
Note: To shift SMC, TFFCR<FF1IE> must be set to "1" to enable TFF1 inversion.

Channel 1 of SMC can be synchronized with the 16-bit timer. In this case, the SMC shift trigger signal from the 16-bit timer is output only when the up-counter value matches TREG5. Set either T4FFCR<EQ5T4> or T4MOD<EQ5T5> to "1".

3.10.5 Application of SMC and Timer Output

As explained in 3.10.4 "Trigger signal from timer", the timing to shift SMC and invert TFF differs depending on the mode of timer. An application to operate SMC while operating an 8-bit timer in PPG mode will be explained below.

To drive a stepping motor, in addition to the value of each phase (SMC output), synchronizing signal is often required at the timing when excitation is changed over. In this application, noting this fact, port 6 is used as a stepping motor control port to output a synchronizing signal to the TO1 pin (shared by P40).



Output Waveforms of 4-Phase 1-step Excitation

Setting example:

		7	6	5	4	3	2	1	0	
TRUN	←	-	-	-	-	-	-	0	0	Stop timer 0, and clear it to zero.
T01MOD	←	1	0	X	X	X	X	0	1	Set timer 0 and timer 1 in PPG output mode and select ϕT1 as the input clock.
TFFCR	←	1	1	-	-	0	1	1	X	Enable TFF1 inversion and set TFF1 to "1".
TREG0	←	*	*	*	*	*	*	*	*	Set the duty of TO1 for TREG0.
TREG1	←	*	*	*	*	*	*	*	*	Set the cycle of TO1 for TREG1.
P4FR	←	-	-	X	X	-	-	-	1	} Assign P40 as T01.
P4CR	←	-	-	-	-	-	-	-	1	
P67FR	←	-	-	-	-	0	0	0	1	Set P6 as SMC in 4-phase 1-step excitation mode.
P67CR	←	-	-	-	-	1	1	1	1	Set all P6 bits to output mode.
P6	←	*	*	*	*	*	*	*	*	Set an initial value.
TRUN	←	-	-	1	-	-	-	1	1	Start timer 0 and timer 1.

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

3.11 Serial Channel

TMP90C845 contains a serial I/O channel for full duplex asynchronous transmission (UART) as well as for I/O extension.

The serial channel has the following operation modes.

- I/O interface mode ——— Mode 0: To transmit and receive I/O data as well as the synchronizing signal SCLK for extending I/O.
- Asynchronous transmission (UART) mode
  - Mode 1: 7-bit data
  - Mode 2: 8-bit data
  - Mode 3: 9-bit data

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

Figure 3.11 (1) shows the data format (for one frame) in each mode.

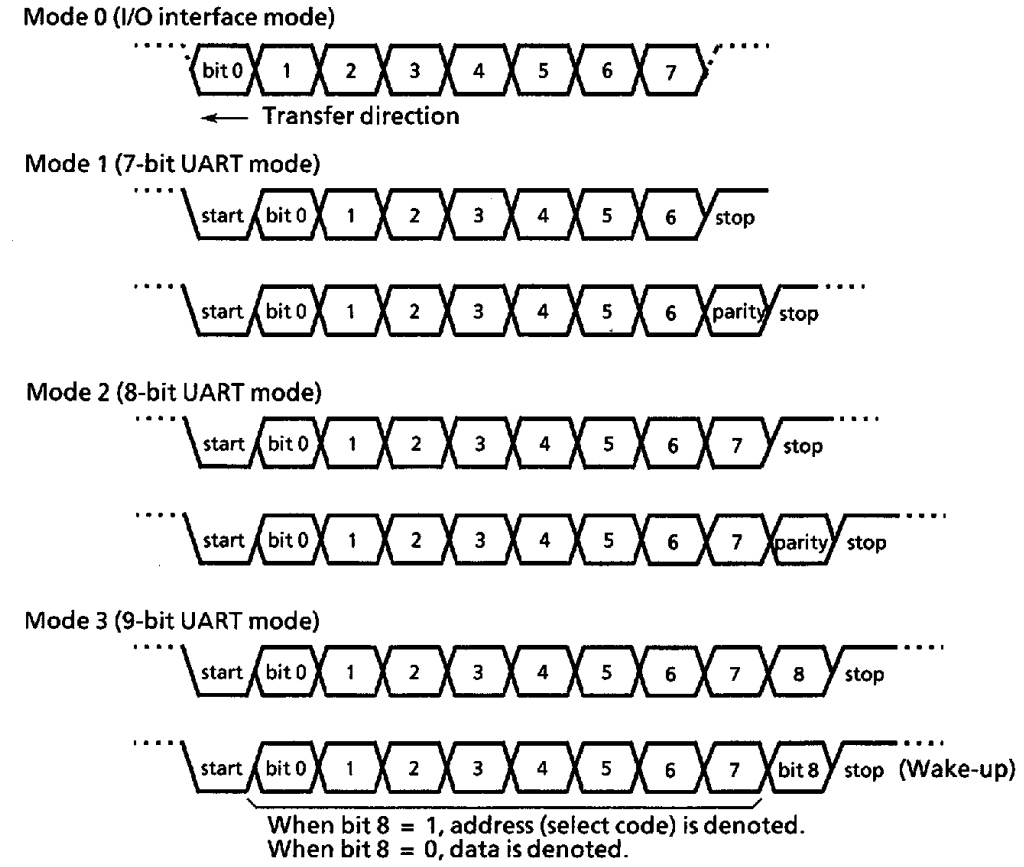


Figure 3.11 (1) Data Formats

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

However, in I/O interface mode, SCLK (serial clock) pin is commonly used for both transmission and receiving, the channel becomes half-duplex.

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

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

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

In I/O interface mode, it is possible to input synchronous signals as well as to transmit or receive data by external clock.

The serial channel includes a special baud rate generator, which can set any baud rate by dividing the frequency of 4 clocks (ϕT0, ϕT2, ϕT8, and ϕT32) from the internal prescaler (shared by 8-bit/16-bit timer) by the value 2 to 16.

3.11.1 Control Registers

The serial channel is controlled by 4 control registers SCMOD, SCCR, BRGCR, and P23FR. Transmitted and received data are stored in register SCBUF.

SCMOD (FFE6H)

	7	6	5	4	3	2	1	0
bit Symbol	TB8	Fixed to "0"	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	Undefined	0	0	0	0	0	0	0
Function	Transfer data Bit 8		1: Receive Enable	1: Wake Up Enable	00: I/O Interface 01: UART 7Bit 10: UART 8Bit 11: UART 9Bit		00: TO2TRG 01: BRG Mode 10: ϕ1 11: –	

Serial transmission clock

	UART mode	I/O interface
00	Timer 2 match detect signal	External clock or baud rate generator
01	Baud rate generator	
10	Internal clock ϕ1	
11	Reserved (cannot be used)	

Serial transmission mode

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

Wake-up function

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

Enable receiving

0	Disable
1	Enable

Transmission data bit 8

Figure 3.11 (2) Serial Channel Mode Register (SCMOD)

SCCR (FFE7H)

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

Select SCLK I/O

0	SCLK output mode
1	SCLK input mode

Edge selection in SCLK input mode

0	Transmits and receives data at the rise edge of SCLK (↑).
1	Transmits and receives data at the fall edge of SCLK(↓).

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

Enable parity addition

*	0	Disable
	1	Enable

Addition/check of even parity

*	0	Odd parity
	1	Even parity

Receiving data bit 8

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

Figure 3.11 (3) Serial Channel Control Register (SCCR)

SCBUF (FFE8H)

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

131190

Figure 3.11 (4) Serial Transmission/Receiving Buffer Registers (SCBUF)

BRGCR (FFE9H)

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

Setting of the divided frequency of baud rate generator

0000	16 divisions
0010 ↓ 1111	2 to 15 divisions
"0001" cannot be set.	

Selecting the input clock of baud rate generator

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

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

Figure 3.11 (5) Baud Rate Generator Control Registers (BRGCR)

P23FR (FFCEH)

	7	6	5	4	3	2	1	0
bit Symbol	ODE	TXDC	SCLKC	Fixed to "0"	Fixed to "0"	IOCS	RAMCS	ROMCS
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	P37 0: CMOS 1: Open Drain	P37 control 0: Port output 1: TxD output	P36 control 0: Port output 1: SCLK output			P26 control 0: 1: $\overline{IOCS}$ output	P25 control 0: 1: $\overline{RAMCS}$ output	P24 control 0: 1: $\overline{ROMCS}$ output

Explained in 3.7 "Programmable Chip Select"

Setting P36 as SCLK output

0	Port output
1	SCLK output

Setting P37 as TxD output

0	Port output
1	TxD output

Setting P37 as open drain output

0	CMOS output
1	Open drain output

Figure 3.11 (6) Port 2 and Port 3 Function Registers (P23FR)

3.11.2 Configuration

Figure 3.11 (7) shows the block diagram of the serial channel.

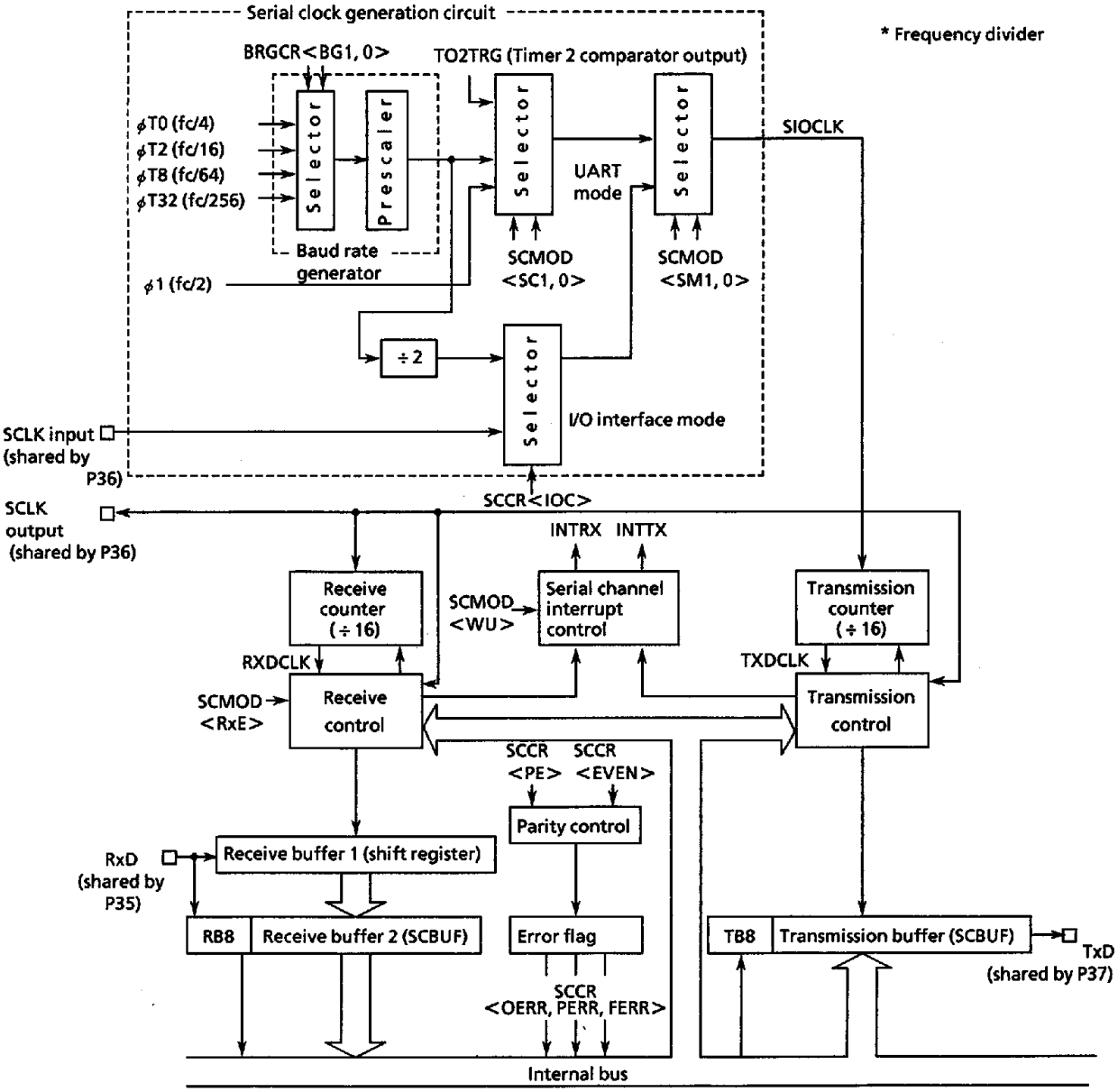


Figure 3.11 (7) Block Diagram of the Serial Channel

① Baud rate generator

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

The input clock to the baud rate generator, ϕT0 (fc/4), ϕT2 (fc/16), ϕT8 (fc/64), or ϕT32 (fc/256) is generated by the 9-bit prescaler which is shared by the timers. One of these input clocks is selected by the baud rate generator control register BRGCR<BG1,0>.

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

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

- UART mode

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

- I/O interface mode

$$\text{Transfer rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divisor of baud rate generator}} \div 2$$

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

ϕT0 = fc/4
ϕT2 = fc/16
ϕT8 = fc/64
ϕT32 = fc/256

Accordingly, when source clock fc is 12.288 MHz, input clock is ϕT2 (fc/16), and frequency divisor is 5, the transfer rate in UART mode becomes as follows:

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

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

Table 3.11 (1) shows an example of the transfer rate in UART mode.

Also with 8-bit timer 2, the serial channel can get a transfer rate. Table 3.11 (2) shows an example of baud rate using timer 2.

Table 3.11 (1) Selection of Transfer Rate (1) (When Baud Rate Generator Is Used)

Source clock (fc)	Input clock / Frequency divisor	φT0 (FC/4)	φT2 (FC/16)	φT8 (FC/64)	φT32 (FC/256)
9.8304MHZ					
	2	76.800	19.200	4.800	1.200
	4	38.400	9.600	2.400	0.600
	8	19.200	4.800	1.200	0.300
	0	9.600	2.400	0.600	0.150
12.288MHZ					
	5	38.400	9.600	2.400	0.600
	A	19.200	4.800	1.200	0.300
14.7456MHZ					
	3	76.800	19.200	4.800	1.200
	6	38.400	9.600	2.400	0.600
	C	19.200	4.800	1.200	0.300

Units: Kbps

Table 3.11 (2) Selection of Transfer Rate (2)
(When Timer 2 (Input Clock φT1) Is Used) Units:Kbps

TREG2 \ fc	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

131190

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

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

Input clock of timer 2

$\phi T1 = fc/8$
$\phi T4 = fc/32$
$\phi T16 = fc/128$

② Serial clock generation circuit

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

1) I/O interface mode

When in SCLK output mode with the setting of SCCR<IOC>="0", the basic clock will be generated by dividing by 2 the output of the baud rate generator described before. When in SCLK input mode with the setting of SCCR<IOC>="1", the rising edge or falling edge will be detected according to the setting of SCCR<SCLKC> register to generate the basic clock.

2) Asynchronous communication (UART) mode

According to the setting of SCMOD<SC1,0>, the above baud rate generator clock, internal clock $\phi 1$ (fc/2) (312.5 Kbaud at 10 MHz), or the match detect signal from timer 2 will be selected to generate the basic clock SIOCK.

③ Receiving counter

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

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

For example, if the sampled data bit is "1", "0" and "1" at 7th, 8th and 9th clock respectively, the received data is evaluated as "1". The sampled data "0", "0" and "1" is evaluated that the received data is "0".

④ Receiving control

1) I/O interface mode

When in SCLK output mode with the setting of SCCR<IOC>="0", RxD signal will be sampled at the rising edge of shift clock which is output to SCLK pin.

When in SCLK input mode with the setting SCCR<IOC>="1", RxD signal will be sampled at the rising edge or falling edge of SCLK input according to the setting of SCCR<SCLKC> register.

2) Asynchronous communication (UART) mode

The receiving control has a circuit for detecting the start bit by the rule of majority. When two or more "0" are detected during 3 samples, it is recognized as normal start bit and the receiving operation is started.

Data being received are also evaluated by the rule of majority.

⑤ Receiving buffer

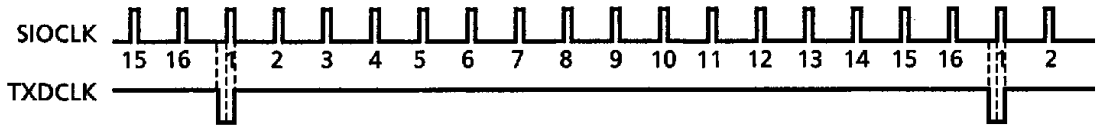
To prevent overrun from occurring, the receiving buffer has a double structure. Received data are stored one bit by one bit in the receiving buffer 1 (shift register type). When 7 bits or 8 bits of data is stored in the receiving buffer 1, the stored data are transferred to another receiving buffer 2 (SCBUF), generating an interrupt INTRX. The CPU reads only receiving buffer 2 (SCBUF). Even before the CPU reads the receiving buffer 2 (SCBUF), the received data can be stored in the receiving buffer 1. However, unless the receiving buffer 2 (SCBUF) is read before all bits of the next data are received by the receiving buffer 1, an overrun error occurs. If an overrun error occurs, the contents of the receiving buffer 1 will be lost, although the contents of the receiving buffer 2 and SCCR<RB8> is still preserved.

The parity bit added in 8-bit UART mode and the most significant bit (MSB) in 9-bit UART mode are stored in SCCR<RB8>.

When in 9-bit UART mode, the wake-up function of the slave controllers is enabled by setting SCMOD<WU> to "1", and interrupt INTRX occurs only when SCCR<RB8> is set to "1".

⑥ Transmission counter

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



⑦ Transmission controller

1) I/O interface mode

In SCLK output mode with the setting of SCCR<IOC>="0", the data in the transmission buffer are output bit by bit to TxD pin at the rising edge of shift clock which is output from SCLK pin.

In SCLK input mode with the setting of SCCR<IOC>="1", the data in the transmission buffer are output bit by bit to TxD pin at the rising edge or falling edge of SCLK input according to the setting of SCCR<SCLKC> register.

2) Asynchronous communication (UART) mode

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

⑧ Transmission buffer

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

⑨ Parity control circuit

When serial channel control register SCCR<PE> is set to "1", it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART or 8-bit UART mode. With SCCR<EVEN> register, even (odd) parity can be selected.

For transmission, parity is automatically generated according to the data written in the transmission buffer SCBUF, and data are transmitted after being stored in SCBUF<TB7> when in 7-bit UART mode while in SCMOD<TB8> when in 8-bit UART mode. <PE> and <EVEN> must be set before transmission data are written in the transmission buffer.

For receiving, data are shifted in the receiving buffer 1, and parity is added after the data are transferred in the receiving buffer 2 (SCBUF), and then compared with <RB7> of SCBUF when in 7-bit UART mode and with SCCR<RB8> when in 8-bit UART mode. If they are not equal, a parity error occurs, and SCCR<PERR> flag is set.

⑩ Error flag

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

1. Overrun error (SCCR<OERR>)

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

2. Parity error (SCCR<PERR>)

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

3. Framing error (SCCR<FERR>)

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

⑪ Generating Timing

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	Center of stop bit
Parity error timing	Center of last bit (Bit 8)	Center of last bit (parity bit)	Center of stop bit
Overrun error timing	Center of last bit (Bit 8)	Center of last bit (parity bit)	Center of stop bit

Note: Framing error occurs after an interrupt has occurred. Therefore, to check for framing error during interrupt operation, it is necessary to wait for 1 bit period of time.

Transmitting

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

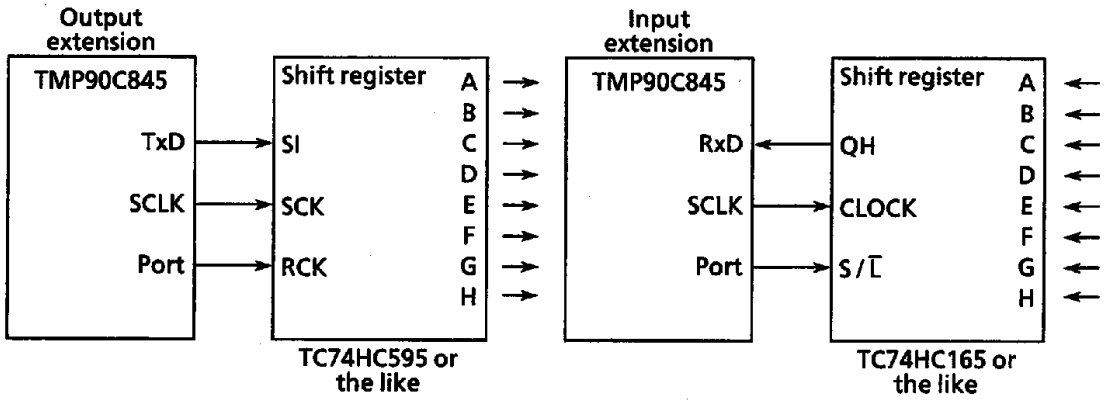
3.11.3 Operational Description

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

This mode is used to increase the number of I/O pins of TMP90C844 for transmitting or receiving data to or from the external shifter register.

This mode includes SCLK output mode to output synchronous clock SCLK and SCLK input mode to input external synchronous clock SCLK.

- Example of SCLK output mode connection



- Example of SCLK input mode connection

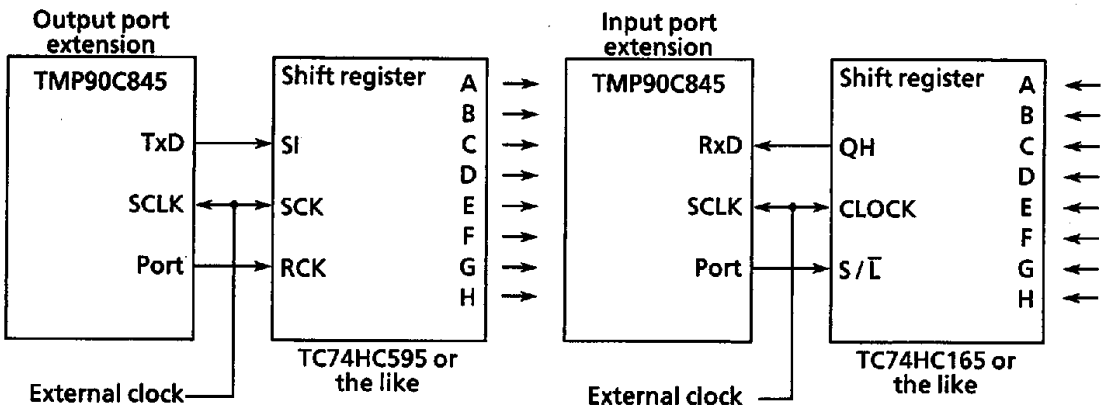


Figure 3.11 (8) I/O Interface Mode

① Transmission

In SCLK output mode, 8-bit data and synchronous clock are output from TxD pin and SCLK pin, respectively, each time the CPU writes data in the transmission buffer. When all data is output, IRFL<IRFTX> will be set to generate INTTX interrupt.

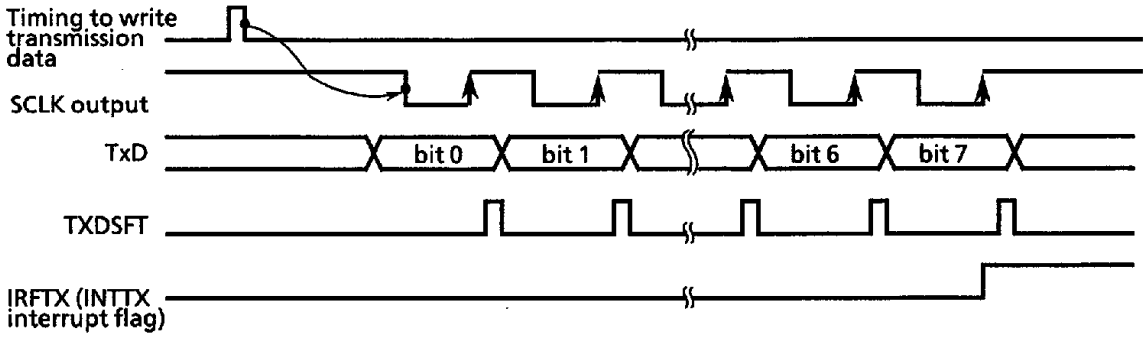


Figure 3.11 (9) Transmitting Operation in I/O Interface Mode (SCLK Output Mode)

In SCLK output mode, 8-bit data are output from TxD pin when SCLK input becomes active while data are written in the transmission buffer by CPU.

When all data are output, <IRFTX> will be set to generate INTTX interrupt.

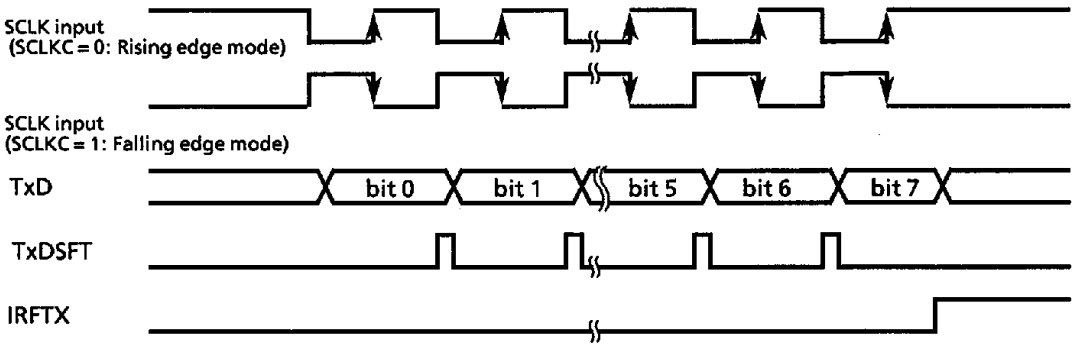


Figure 3.11 (10) Transmitting Operation in I/O Interface Mode (SCLK Input Mode)

② Receiving

In SCLK output mode, received data are read by the CPU, and synchronous clock is output from SCLK pin and the next data are shifted in the receiving buffer 1 whenever the receive interrupt flag IRFL<IRFRX> is cleared. When 8-bit data are received, the data will be transferred in the receiving buffer 2 (SCBUF), and <IRFRX> will be set again to generate INTRX interrupt.

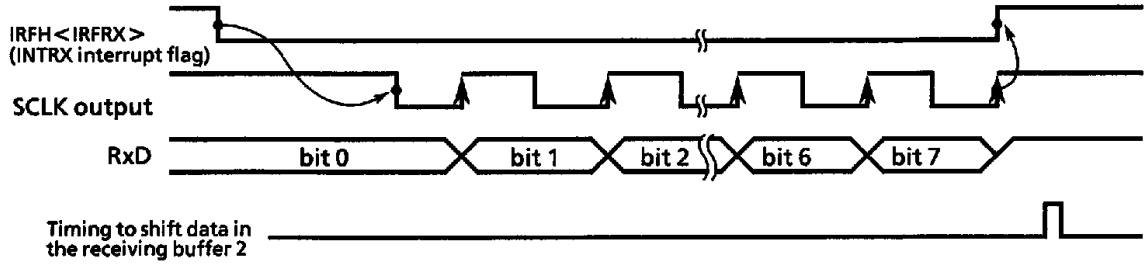


Figure 3.11 (11) Receiving Operation in I/O Interface Mode (SCLK Output Mode)

In SCLK input mode, received data are read by the CPU, and the next data is shifted in the receiving buffer 1 when SCLK input becomes active while the receive interrupt flag <IRFRX> is cleared. When 8-bit data is received, the data will be shifted in the receiving buffer 2 (SCBUF), and <IRFRX> will be set again to generate INTRX interrupt.

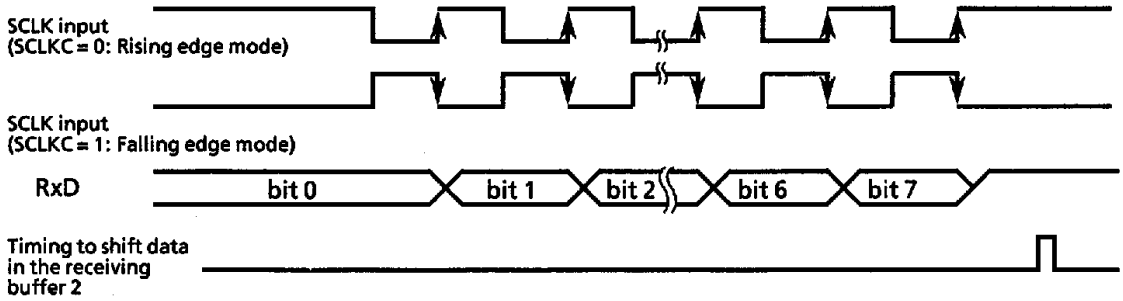


Figure 3.11 (12) Receiving Operation in I/O Interface Mode (SCLK Input Mode)

For data reception, the system must be placed in the receive enable state (SCMOD<RXE>="1")

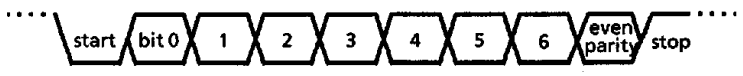
(2) Mode 1 (7-bit UART Mode)

7-bit mode can be set by setting serial channel mode register SCMOD<SM1,0>to "01".

In this mode, a parity bit can be added, and the addition of a parity bit can be enabled or disabled by serial channel control register.

SCCR<PE>, and even parity or odd parity is selected by SCCR<EVEN> when <PE> is set to "1" (enable).

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



← Transfer direction (transfer rate: 2400 bps at fc=12.288 MHz)

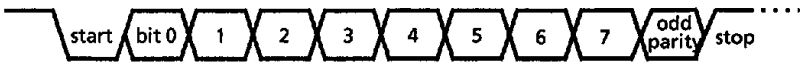
	7 6 5 4 3 2 1 0	
P3CR	← 1 - - - - - - -	} Select P37 as the TxD pin.
P23FR	← - 1 - - - 0 0 0	
SCMOD	← X 0 - X 0 1 0 1	Set 7-bit UART mode.
SCCR	← X 1 1 X X X X X	Add an even parity.
BRGCR	← 0 X 1 0 0 1 0 1	Set transfer rate at 2400 bps.
TRUN	← X X 1 - - - - -	Start the prescaler for the baud rate generator.
INTEL	← - - - - - 1 - -	Enable INTTX interrupt.
SCBUF	← * * * * * * * *	Set data for transmission.

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

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

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

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



← Direction of transmission (transmission rate: 9600 bps @ fc=12.288 MHz)

Main setting

```
        7 6 5 4 3 2 1 0
P3CR  ← - - 0 - - - - -    Specify P35 (RxD) as the input pin.
SCMOD ← - 0 1 X 1 0 0 1    Enable receiving in 8-bit UART mode.
SCCR  ← X 0 1 X X X X X    Add an odd parity.
BRGCR ← 0 X 0 1 0 1 0 1    Set transfer rate at 9600 bps.
TRUN  ← X X 1 - - - - -    Start the prescaler for the baud rate generator.
INTEL ← - - - - - 1 - - -  Enable INTRX interrupt.
```

INTRX processing

```
Acc  ← SCCR AND 00011100   Check for error.
if Acc ≠ 0 then error
Acc  ← SCBUF               Read the received data.
```

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

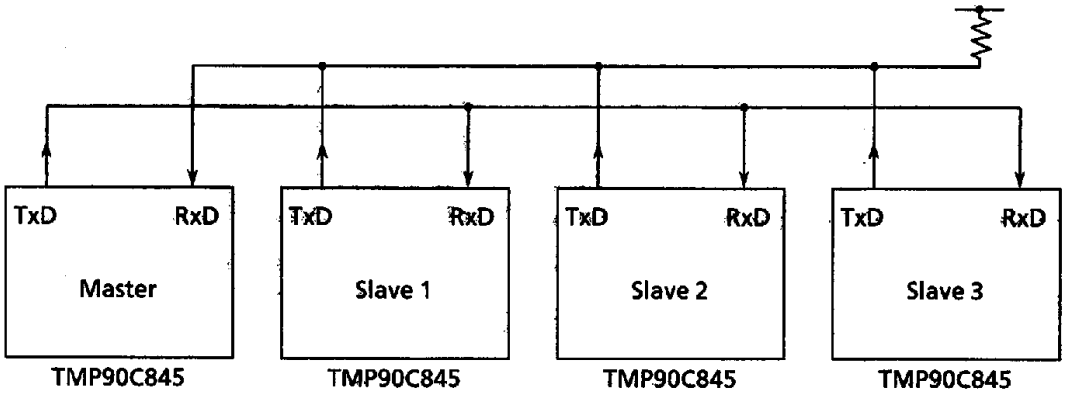
(4) Mode 3 (9-bit UART Mode)

9-bit UART mode can be specified by setting SCMOD<SM1,0> to "11". In this mode, parity bit cannot be added.

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

Wake-up function

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



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

Figure 3.11 (13) Serial Link Using Wake-Up Function

Protocol

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

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

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

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

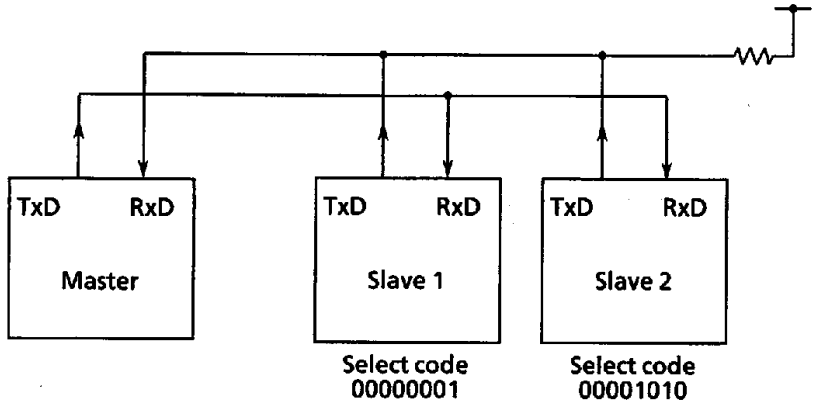
⑤ The master controller transmits data to the specified slave controller whose SCMOD<WU> bit is cleared to "0". The MSB (bit 8) SCMOD<TB8> is cleared to "0".

⑥ The other slave controllers (with the SCMOD<WU> bit remaining at "1") ignore the receiving data because their MSBs (bit 8 or SCCR<RB8>) are set to "0" to disable the interrupt INTRX.

When the WU bit is cleared to "0", the interrupt INTRX occurs, so that the slave controller can read the receiving data.

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

Setting example: To link two slave controllers serially with the master controller, and use the internal clock ϕ/1 (fc/2) as the transfer clock.



- Setting the master controller

Main

```
P3CR  ← 1 - 0 - - - - -
P23FR ← - 1 - - - 0 0 0
INTEL ← - - - - - 1 1 -
SCMOD ← 1 0 1 0 1 1 1 0
SCBUF ← 0 0 0 0 0 0 0 1
```

P3CR, P23FR: Select P35 as RxD pin and P37 as TxD pin.
INTEL: Enable INTRX and INTTX.
SCMOD: Set ϕ1 (fc/2) as the transmission clock in 9-bit UART mode.
SCBUF: Set the select code for slave controller 1.

INTTX interrupt

```
SCMOD ← 0 - - - - - - -
SCBUF ← * * * * * * * *
```

SCMOD: Sets SCMOD<TB8> to "0".
SCBUF: Set data for transmission.

- Setting the slave controller 2

Main

```
P3CR  ← 1 - 0 - - - - -
P23FR ← 1 1 - - - 0 0 0
INTEL ← - - - - - 1 1 -
SCMOD ← 0 0 1 1 1 1 1 0
```

P3CR, P23FR: Select P35 as RxD pin and P37 as TxD pin (open drain output).
INTEL: Enable INTRX and INTTX.
SCMOD: Set <WU> to "1" in the 9-bit UART transmission mode with transfer clock ϕ1 (fc/2).

INTRX interrupt

```
Acc ← SCBUF
if Acc = Select code
then SCMOD ← - - - 0 - - - -
```

Clear <WU> to "0".

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

3.12 Analog/Digital Converter

TMP90C845 contains a high-speed, high-accuracy analog/digital converter (A/D converter) with 4-channel analog input that features 8-bit sequential comparison.

Figure 3.12 (1) shows the block diagram of the A/D converter. 4-channel analog input pins (AN3 to AN0) are shared by input-only port P5 and so can be used as input port.

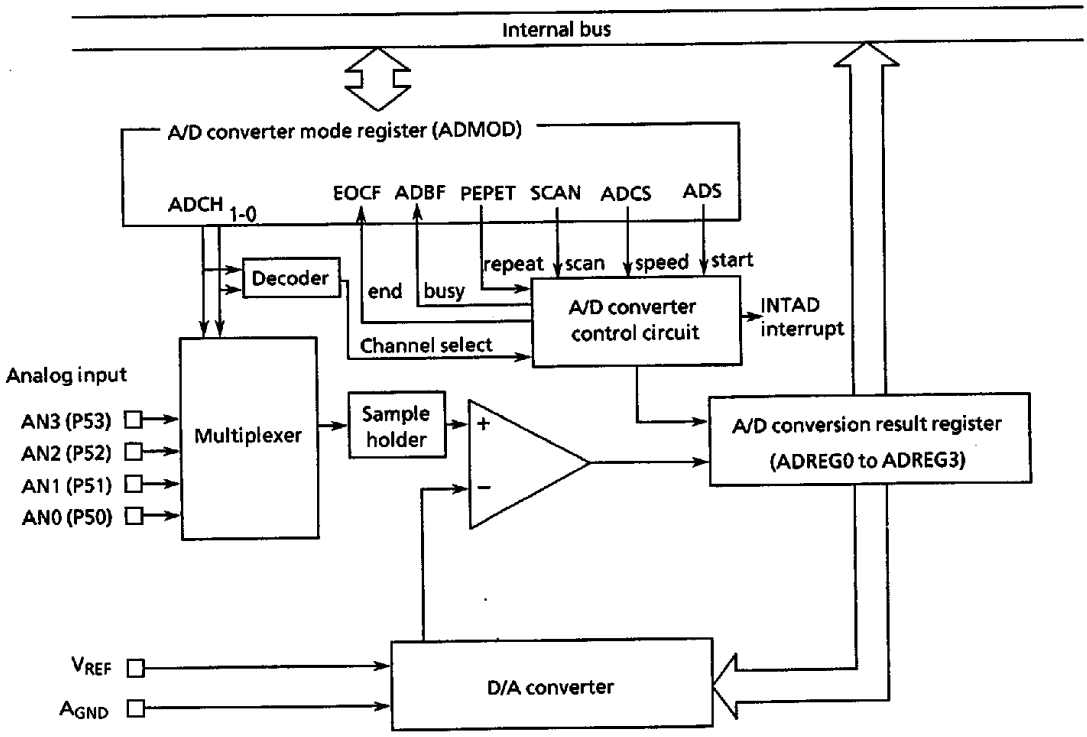


Figure 3.12 (1) Block Diagram of A/D Converter

3.12.1 Control Registers

ADMOD (FFEFH)

	7	6	5	4	3	2	1	0
bit Symbol	EOCF	ADBF	REPET	SCAN	ADCS	ADS	ADCH1	ADCH0
Read/Write	R		R/W	R/W	R/W		R/W	
After reset	0	0	0	0	0	0	0	0
Function	1: END	1: BUSY	1: Repeat Mode Set	1: Scan Mode Set	1:Low speed mode	1: START	Analog Input Channel Select	

Analog input channel select

	Normal	Scan mode
00	AN0	AN0
01	AN1	AN0→AN1
10	AN2	AN0→AN1→AN2
11	AN3	AN0→AN1→AN2→AN3

A/D conversion start

0	——
1	Start A/D conversion.

Note: Always "0" when data is read.

A/D conversion speed selection

0	High speed conversion mode (95 states)
1	Low speed conversion mode (190 states)

Specification of A/D scan mode

0	Fixed A/D conversion channel mode
1	A/D conversion channel scan mode

Selecting A/D repeat mode

0	A/D conversion single mode
1	A/D conversion repeat mode

A/D conversion busy flag

0	A/D conversion not busy
1	A/D conversion busy

A/D conversion end flag

0	A/D conversion not ended nor started
1	A/D conversion ended

Figure 3.12 (2) A/D Conversion Mode Register (ADMOD)

ADREG0 (FFF0H)

	7	6	5	4	3	2	1	0
bit Symbol								
Read/Write	R							
After reset	Undefined							
Function	A/D conversion result of channel 0 is stored.							

A/D conversion result register for AN0 (Read only)

ADREG1 (FFF1H)

	7	6	5	4	3	2	1	0
bit Symbol								
Read/Write	R							
After reset	Undefined							
Function	A/D conversion result of channel 1 is stored.							

A/D conversion result register for AN1 (Read only)

ADREG2 (FFF2H)

	7	6	5	4	3	2	1	0
bit Symbol								
Read/Write	R							
After reset	Undefined							
Function	A/D conversion result of channel 2 is stored.							

A/D conversion result register for AN2 (Read only)

ADREG3 (FFF3H)

	7	6	5	4	3	2	1	0
bit Symbol								
Read/Write	R							
After reset	Undefined							
Function	A/D conversion result of channel 3 is stored.							

A/D conversion result register for AN3 (Read only)

Figure 3.12 (3) A/D Conversion Result Register (ADREG0~ADREG3)

3.12.2 Operation

(1) Analog Reference Voltage

High analog reference voltage is applied to the VREF pin, and the low analog voltage is applied to AGND pin.

The reference voltage between VREF and AGND is divided by 256 using ladder resistance, and compared with the analog input voltage for A/D conversion.

(2) Analog Input Channels

Analog input channel is selected by ADMOD<ADCH1,0>. However, which channel to select depends on the operation mode of the A/D converter.

In fixed analog input mode, one channel is selected by ADMOD<ADCH1,0> among three pins: AN0 to AN3.

In analog input channel scan mode, the number of channels to be scanned from AN0 is specified by ADMOD<ADCH1,0>, such as AN0→AN1, AN0→AN1→AN2, and AN0→AN1→AN2→AN3.

When reset, A/D conversion channel register will be initialized to ADMOD<ADCH1,0>=00, so that AN0 pin will be selected.

The pins which are not used as analog input channel can be used as ordinary input port P5.

(3) Starting A/D conversion

A/D conversion starts when A/D conversion register ADMOD<ADS> is written "1". When A/D conversion starts, A/D conversion busy flag ADMOD<ADBF> which indicates "A/D conversion is in progress" will be set to "1".

(4) A/D Conversion Mode

Both fixed A/D conversion channel mode and A/D conversion channel scan mode have two conversion modes, i.e., single and repeat conversion modes.

In fixed channel repeat mode, conversion of specified one channel is executed repeatedly.

In scan repeat mode, scanning from AN0, ···→AN3 is executed repeatedly.

A/D conversion mode is selected by ADMOD<REPET, SCAN>.

(5) A/D Conversion Speed Selection

There are two A/D conversion speed modes: high speed mode and low speed mode. The selection is executed by ADMOD<ADCS> register.

When reset, ADMOD<ADCS> will be initialized to "0", so that high speed conversion mode will be selected.

(6) A/D Conversion End and Interrupt

- A/D conversion single mode

ADMOD<EOCF> for A/D conversion end will be set to "1", ADMOD<ADBF> flag will be reset to "0", and INTAD interrupt will be enabled when A/D conversion of specified channel ends in fixed conversion channel mode or when A/D conversion of the last channel ends in channel scan mode.

A/D interrupt INTAD is controlled by the interrupt mask INTEH<IET2> commonly used for timer 2 INTT2 interrupt, and INTAD or INTT2 is selected by INTEH<ADIS>. To enable INTAD, set both INTEH<IET2> and INTEH<ADIS> to "1".

Both INTAD and INTT2 interrupts jump to the same vector address (0030H), so that it is judged by <ADIS> whether INTAD interrupt or INTT2 interrupt is being requested now.

Interrupt requesting flip-flop is cleared only by resetting operation or reading the A/D conversion result storing register and cannot be cleared by instruction. When interrupt source is changed (between INTAD or INTT2), the previous interrupt requesting flag will automatically be cleared.

- A/D conversion repeat mode

For both fixed conversion channel mode and conversion channel scan mode, INTAD will be disabled when in repeat mode. Always leave the INTEH<ADIS> flag at "0".

Write "0" to ADMOD<REPET> to end the repeat mode. Then, the repeat mode will be exited as soon as the conversion in progress is completed.

(7) Storing the A/D Conversion Result

The results of A/D conversion are stored in ADREG0 to ADREG3 registers for each channel. In repeat mode, the registers are updated whenever conversion ends.

ADREG0 to ADREG3 are read-only registers.

(8) Reading the A/D Conversion Result

The results of A/D conversion are stored in ADREG0 to ADREG3 registers. When the contents of one of ADREG0 to ADREG3 registers are read, ADMOD<EOCF> will be cleared to "0".

Setting example: ① When the analog input voltage of the AN3 pin is A/D converted and the results are read in the memory at FF10H by A/D interrupt INTAD routine

Main setting

```
INTEH ← X - 1 - - - 1 -
ADMOD ← X X 0 0 0 1 1 1
```

Enable INTAD.
Specify AN3 pin as an analog input channel and starts A/D conversion in high speed mode.

INTAD routine

```
A       ← ADREG3
(FF10H)← A
```

Read the value of ADREG3 into the accumulator and store the value of the accumulator in the memory at FF10H.

② When the analog input voltage of AN0~AN2 pins is kept A/D converted in high speed conversion channel scan repeat mode

```
INTEH ← X - 0 - - - - -
```

Disable INTAD.

```
ADMOD ← X X 1 1 0 1 1 0
```

Start the A/D conversion of analog input channels AN0~AN2 in the high-speed scan repeat mode.

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

3.13 Watchdog Timer (Runaway Detecting Timer)

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

3.13.1 Configuration

Figure 3.13 (1) shows the block diagram of the watchdog timer (WDT).

The watchdog timer consists of 22-stage binary counter which uses $\phi 1$ (fc/2) as the input clock, selector that selects one from the four outputs generated from the binary counter, flip-flop for enable/disable control, and two control registers.

The watchdog timer generates interrupt INTWD after the detection time set with watchdog timer detection time selection register WDMOD<WDTP1,0> and clears to zero by software (instruction) the watchdog timer binary counter before INTWD interrupt occurs. If the CPU malfunctions (runs away) due to causes such as noise and does not execute the instruction to clear the watchdog timer, the binary counter will overflow, and an INTWD interrupt will be generated. The CPU is notified of malfunction (runaway) by the INTW interrupt and runs the anti-malfunction (runaway) program to return to normal operation.

The watchdog timer restarts operation immediately after resetting is released.

The watchdog timer stops only in the STOP mode. After the STOP mode is released and the warming up time has elapsed, the watchdog timer resumes operation.

The watchdog timer operates in the other standby modes (IDLE1 and RUN mode), whereas it can be disabled when entering one of these standby modes.

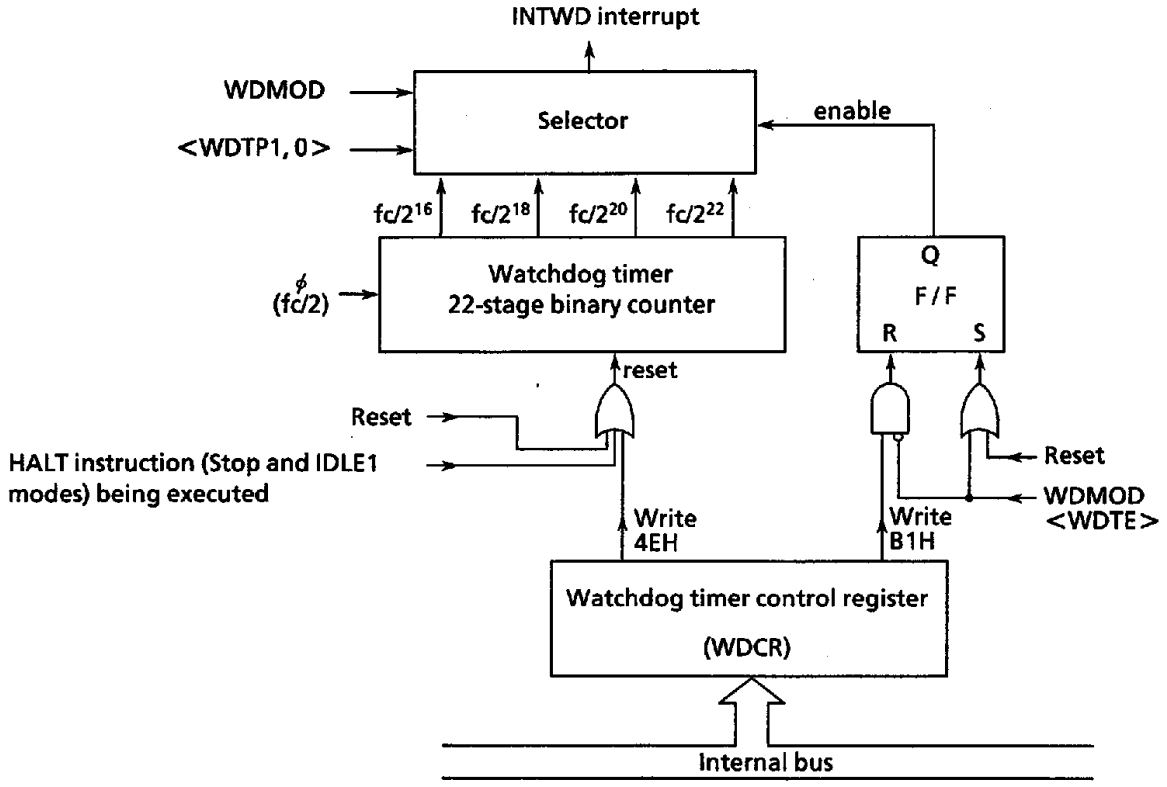


Figure 3.13 (1) Block Diagram of Watchdog Timer

3.13.2 Control Registers

Watchdog timer WDT is controlled by two control registers WDMOD and WDCR.

(1) Watchdog Timer Mode Register (WDMOD)

① Setting the detecting time of watchdog timer

This 2-bit register is used to set the watchdog timer interrupt time for detecting the runaway. This register is initialized to WDMOD<WDTP1, 0>=00 when reset, and therefore 2^{16}/fc is set. (The number of states is approx. 32,768.)

② Watchdog timer enable/disable control register WDMOD<WDTE>

When reset, WDMOD<WDTE> is initialized to "1" enable the watchdog timer.

To disable, it is necessary to clear this bit to "0" and write the disable code (B1H) in the watchdog timer control register WDCR. This makes it difficult for the watchdog timer to be disabled by runaway.

However, it is possible to return from the disable state to enable state by merely setting <WDTE> to "1".

WDMOD (FFD2H)

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

→ Explained in 3.4.4 "STOP mode"

→ Inverts each time the EXX instruction is executed.

→ Select the standby mode by HALT instruction

*	00	RUN mode
	01	STOP mode
	10	IDLE1 mode
	11	Cannot be used.

→ Select warming-up time when returned from STOP mode

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

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

→ Select the detecting period of watchdog timer

*	00	2^{16} / fc (approx. 6.6 ms)
	01	2^{18} / fc (approx. 26.2 ms)
	10	2^{20} / fc (approx. 105 ms)
	11	2^{22}/fc (approx. 419 ms)

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

→ Watchdog timer enable/disable control

	0	Disable (Note)
*	1	Enable

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

Figure 3.13 (2) Watchdog Timer Mode Register

(2) Watchdog Timer Control Register (WDCR)

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

- Disable control

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

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

- Enable control

Set WDMOD<WDTE> to "1".

- Watchdog timer clear control

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

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

WDCR (FFD3H)

	7	6	5	4	3	2	1	0
bit Symbol	–							
Read/Write	W							
After reset	–							
Function	B1H: WDT Disable Code				4EH: WDT Clear Code			

→ Disable/clear watchdog timer.

B1H	Disable code
4EH	Clear code
Other	——

Figure 3.13 (3) Watchdog Timer Control Register

3.13.3 Operation

The watchdog timer generates interrupt INTWD after the detecting time set with watchdog timer detecting time selection register WDMOD<WDTP1,0> and clears to zero by software (instruction) the watchdog timer binary counter before INTWD interrupt occurs. If the CPU malfunctions (runs away) due to causes such as noise and does not execute the instruction to clear the watchdog timer, the binary counter will overflow, and an INTWD interrupt will be generated. The CPU is notified of malfunction (runaway) by the INTW interrupt and runs the anti-malfunction (runaway) program to return to normal operation.

The watchdog timer restarts operation immediately after resetting is released.

The watchdog timer stops its operation in the IDLE1 and STOP modes. In the RUN mode, the watchdog timer is enabled.

However, the function can be disabled when entering the RUN mode.

Example : ① Clear the binary counter

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

② Set the watchdog timer detecting time to 2^{18}/fc

```
WDMOD ← 1 0 1 - - - X X
```

③ Disable the watchdog timer.

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

Set the STOP mode (warming up time: 2^{16}/fc)

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

4. ELECTRICAL CHARACTERISTICS (PRELIMINARY)

TMP90C845N/TMP90C845F

4.1 Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
V_{CC}	Power supply voltage	−0.5 ~ +7	V
V_{IN}	Input voltage	−0.5 ~ Vcc + 0.5	V
P_D	Power consumption (at Ta = 85°C)	F 500 N 600	mW
T_{SOLDER}	Soldering temperature (10 sec)	260	°C
T_{STG}	Storage temperature	−65 ~ 150	°C
T_{OPR}	Operating temperature	−40 ~ 85	°C

4.2 DC Characteristics

Vcc = 5V ± 10% TA = −20~70°C (1~16MHz)
Typical values are for TA = 25°C and Vcc = 5 V.

Symbol	Parameter	Min	Max	Unit	Condition
V_{IL}	Input Low Voltage (AD0~AD7)	−0.3	0.8	V	
V_{IL1}	P2, P3, P4, P5, P6, P7	−0.3	0.3Vcc	V	
V_{IL2}	$\overline{RESET}$, P45 (INT0)	−0.3	0.25Vcc	V	
V_{IL3}	$\overline{EA}$	−0.3	0.3	V	
V_{IL4}	X1	−0.3	0.2Vcc	V	
V_{IH}	Input High Voltage (AD0~AD7)	2.2	Vcc + 0.3	V	
V_{IH1}	P2, P3, P4, P5, P6, P7	0.7Vcc	Vcc + 0.3	V	
V_{IH2}	$\overline{RESET}$, P45 (INT0)	0.75Vcc	Vcc + 0.3	V	
V_{IH3}	$\overline{EA}$	Vcc − 0.3	Vcc + 0.3	V	
V_{IH4}	X1	0.8Vcc	Vcc + 0.3	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 1.6mA
V_{OH} V_{OH1} V_{OH2}	Output High Voltage	2.4 0.75Vcc 0.9Vcc		V V V	I_{OH} = −400μA I_{OH} = −100μA I_{OH} = −20μA
I_{DAR}	Darlington Drive Current (8 I/O Pins max)	−1.0	−3.5	mA	V_{EXT} = 1.5V R_{EXT} = 1.1 kΩ
I_{LI}	Input Leakage Current	0.02 (Typ)	±5	μA	0.0 ≤ Vin ≤ Vcc
I_{LO}	Output Leakage Current	0.05 (Typ)	±10	μA	0.2 ≤ Vin ≤ Vcc − 0.2
I_{CC}	Operating Current (RUN) Idle 1	35 (Typ) 1.5 (Typ)	50 5	mA mA	tosc = 16MHz
	STOP (TA = −20~70°C) STOP (TA = 0~50°C)	0.2 (Typ)	40 10	μA μA	0.2 ≤ Vin ≤ Vcc − 0.2
V_{STOP}	Power Down Voltage (@STOP) (RAM back Up)	2.0	6.0	V	V_{IL2} = 0.2Vcc, V_{IH2} = 0.8Vcc
R_{RST}	$\overline{RESET}$ Pull Up Register	50	150	KΩ	
C_{IO}	Pin Capacitance		10	pF	testfreq = 1MHz
V_{TH}	Schmitt width ($\overline{RESET}$, P45)	0.4	1.0 (Typ)	V	

4.3 AC Characteristics

Vcc = 5V ± 10% TA = −20~70℃ (1~16MHz)

Symbol	Parameter	Variable		12.5MHz Clock		16MHz Clock		Unit
		Min	Max	Min	Max	Min	Max	
t_{OSC}	Oscillation cycle (=X)	62.5	1000	80		62.5		ns
t_{CYC}	CLK Period	4X	4X	320		250		ns
t_{WH}	CLK High width	2X − 40		120		85		ns
t_{WL}	CLK Low width	2X − 40		120		85		ns
t_{AL}	A0~7 effective address → ALE fall	0.5X − 15		25		16		ns
t_{LA}	ALE fall → A0~7 hold	0.5X − 15		25		16		ns
t_{LL}	ALE Pulse width	X − 40		40		23		ns
t_{LC}	ALE fall → $\overline{RD}/\overline{WR}$ fall	0.5X − 30		10		1		ns
t_{CL}	$\overline{RD}/\overline{WR}$ rise → ALE rise	0.5X − 20		20		11		ns
t_{ACL}	A0~7 effective address → $\overline{RD}/\overline{WR}$ fall	X − 25		55		38		ns
t_{ACH}	Upper effective address → $\overline{RD}/\overline{WR}$ fall	1.5X − 50		70		44		ns
t_{CA}	$\overline{RD}/\overline{WR}$ fall → Upper address hold	0.5X − 20		20		11		ns
t_{ADL}	A0~7 effective address → Effective data input		3.0X − 35		205		153	ns
t_{ADH}	Upper effective address → Effective data input		3.5X − 55		225		164	ns
t_{RD}	$\overline{RD}$ fall → Effective data input		2.0X − 50		110		75	ns
t_{RR}	$\overline{RD}$ Pulse width	2.0X − 40		120		85		ns
t_{HR}	$\overline{RD}$ rise → Data hold	0		0		0		ns
t_{RAE}	$\overline{RD}$ rise → Address enable	X − 15		65		48		ns
t_{WW}	$\overline{WR}$ pulse width	2.0X − 40		120		85		ns
t_{DW}	Effective data → $\overline{WR}$ rise	2.0X − 50		110		75		ns
t_{WD}	$\overline{WR}$ rise → Effective data hold	0.5X − 10		30		21		ns
t_{ACKH}	Upper address → CLK fall	2.5X − 50		150		106		ns
t_{ACKL}	Lower address → CLK fall	2.0X − 50		110		75		ns
t_{CKHA}	CLK fall → Upper address hold	1.5X − 80		40		13		ns
t_{CCK}	$\overline{RD}/\overline{WR}$ fall → CLK fall	X − 25		55		37		ns
t_{CKHC}	CLK fall → $\overline{RD}/\overline{WR}$ rise	X − 60		20		2		ns
t_{DCK}	Valid data → CLK fall	X − 50		30		12		ns
t_{CWA}	$\overline{RD}/\overline{WR}$ fall → Valid $\overline{WAIT}$		X − 40		40		22	ns

150491

Vcc = 5V ± 10% TA = −20~70℃ (1~16MHz)

Symbol	Parameter	Variable		12.5MHz Clock		16MHz Clock		Unit
		Min	Max	Min	Max	Min	Max	
t_{AWAL}	Lower address → Valid $\overline{WAIT}$		2.0X − 70		90		55	ns
t_{WAH}	CLK fall → Valid $\overline{WAIT}$ hold	0		0		0		ns
t_{AWAH}	Upper address → Valid $\overline{WAIT}$		2.5X − 70		130		86	ns
t_{CPW}	CLK fall → Port Data Output		X + 200		280		262	ns
t_{PRC}	Port Data Input → CLK fall	200		200		200		ns
t_{CPR}	CLK fall → Port Data hold	100		100		100		ns

150491

AC Measuring Conditions

- Output level : High 2.2V / Low 0.8V, C_L = 50pF
  (However, CL = 100pF for AD0~7, A8~15, ALE, $\overline{RD}$, $\overline{WR}$)
- Input level : High 2.4V/Low 0.45V (AD0~AD7)
  High 0.8Vcc/Low 0.2Vcc (except for AD0~AD7)

4.4 A/D Conversion Characteristics

Vcc = 5V ± 10% TA = −20~70℃
f = 1~16MHz

Symbol	Parameter	Min	Typ	Max	Unit
V_{REF}	Analog reference voltage	Vcc − 1.5	Vcc	Vcc	V
A_{GND}	Analog reference voltage	V_{SS}	V_{SS}	Vss	
V_{AIN}	Analog input voltage range	V_{SS}		Vcc	
I_{REF}	Supply current for analog reference voltage		0.5	1.0	mA
Error (Quantize error of ±0.5 LSB not included)	Total error (TA = 25℃, Vcc = V_{REF} = 5.0V)			1.0	LSB
	Total error			2.5	

4.5 Zero-Cross Characteristics

Vcc = 5V ± 10% TA = −20~70℃
f = 1~16MHz

Symbol	Parameter	Condition	Min	Max	Unit
V_{ZX}	Zero-cross detection input	AC coupling C = 0.1μF	1	1.8	V_{ACP-P}
A_{ZX}	Zero-cross accuracy	50/60 Hz sine wave		135	mV
F_{ZX}	Zero-cross detection input frequency		0.04	1	KHz

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

Vcc = 5V ± 10% TA = −20~70℃
f = 1~16MHz

Symbol	Parameter	Variable		12.5MHz		16MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_{VCK}	Clock cycle	8X + 100		740		600		ns
t_{VCKL}	Low level clock pulse width	4X + 40		360		290		ns
t_{VCKH}	High level clock pulse width	4X + 40		360		290		ns

4.7 Interrupt Operation

Vcc = 5V ± 10% TA = −20~70℃
f = 1~16MHz

Symbol	Parameter	Variable		12.5MHz		16MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_{INTAL}	INT0 low level pulse width (⊓)	4X		320		250		ns
t_{INTAH}	INT0 high level pulse width (⊔)	4X		320		250		ns
t_{INTBL}	INT1, INT2 low level pulse width (⊓)	8X + 100		740		600		ns
t_{INTBH}	INT1, INT2 high level pulse width (⊔)	8X + 100		740		600		ns

4.8 Serial Channel Timing – I/O Interface Mode

(1) SCLK Input Mode

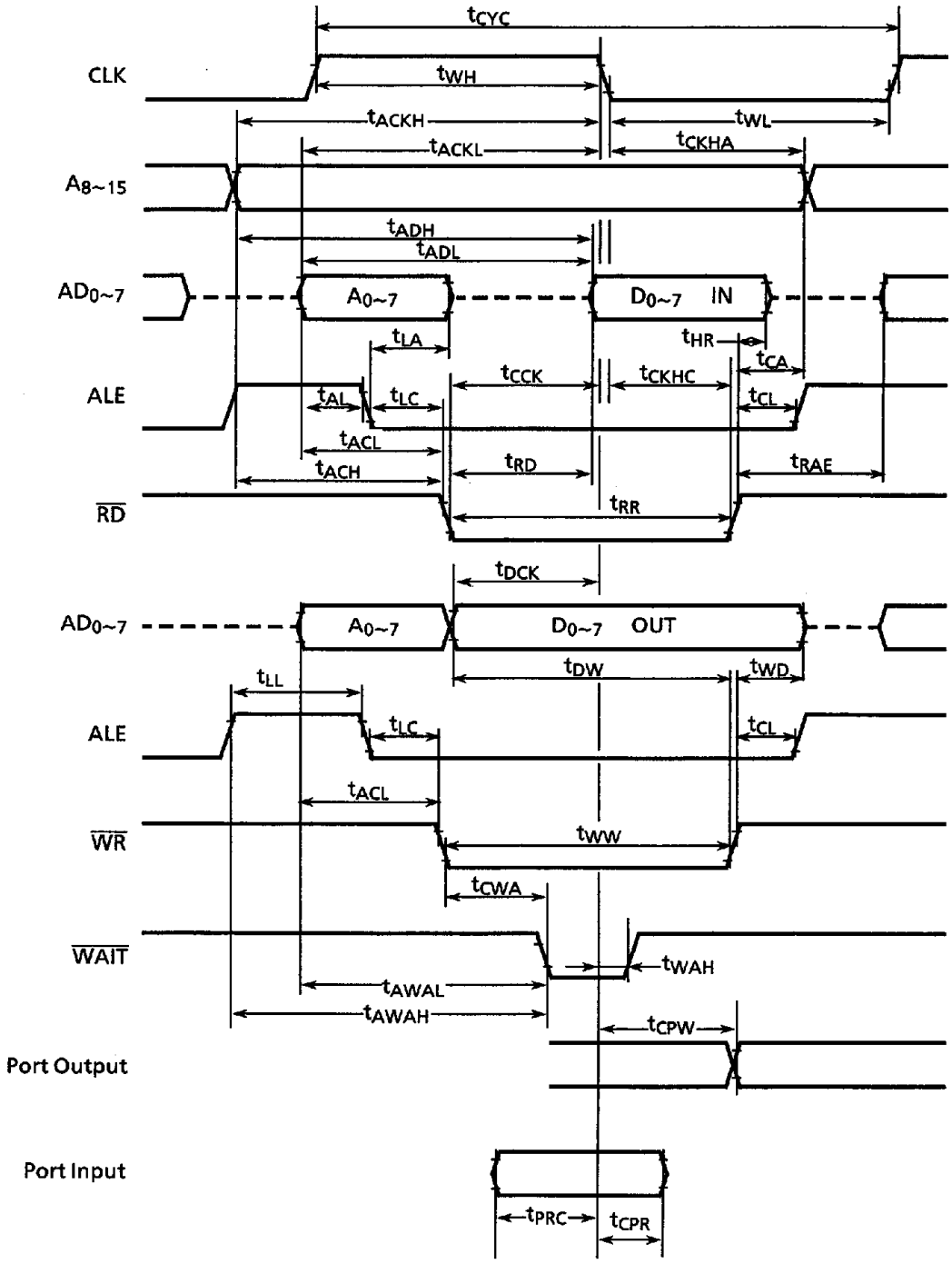
Vcc = 5V ± 10% TA = −20~70℃
f = 1~16MHz

Symbol	Parameter	Variable		12.5MHz		16MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_{SCY}	SCLK cycle	16X		1.28		1		μs
t_{OSS}	Output data → Rising edge of SCLK	$t_{SCY}/2 - 5X - 50$		190		137		ns
t_{OHS}	SCLK rising edge → Output data hold	5X – 100		300		212		ns
t_{HSR}	SCLK rising edge → Input data hold	0		0		0		ns
t_{SRD}	SCLK rising edge → Effective data input		$t_{SCY} - 5X - 100$		780		587	ns

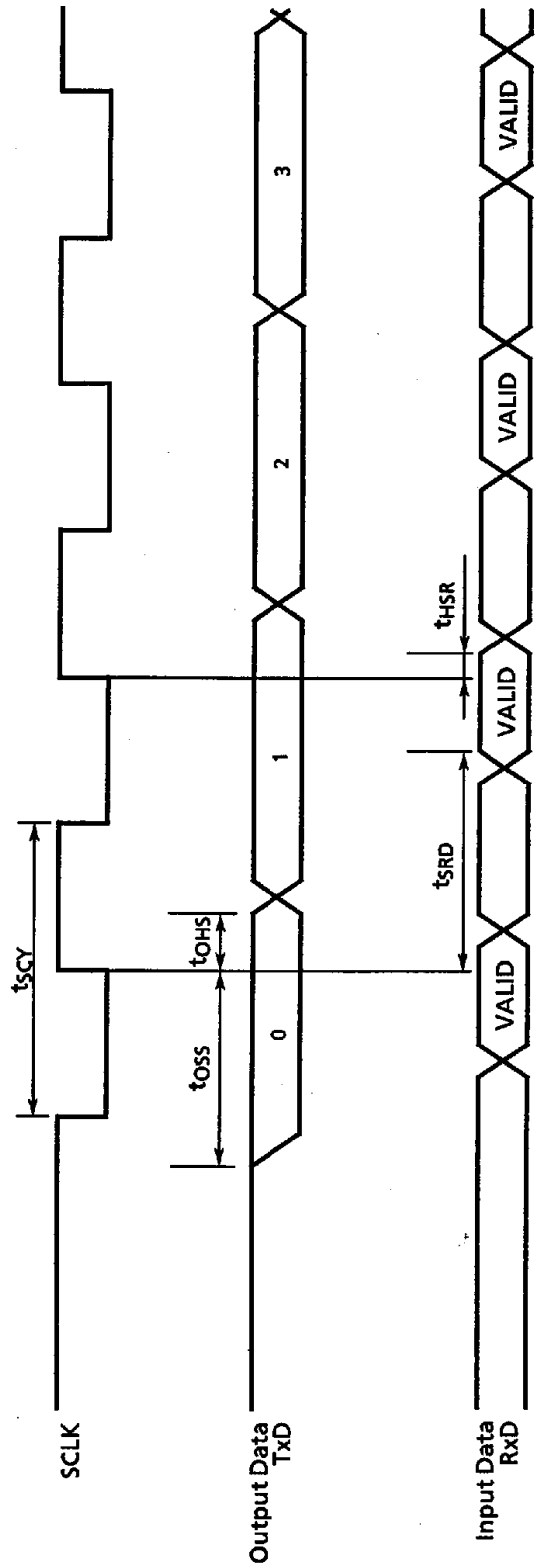
(2) SCLK Output Mode

Symbol	Parameter	Variable		12.5MHz		16MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_{SCY}	SCLK cycle (programmable)	16X	8192X	1.28	655.4	1	512	μs
t_{OSS}	Output data → SCLK rising edge	$t_{SCY} - 2X - 50$		970		725		ns
t_{OHS}	SCLK rising edge → Output data hold	2X – 80		80		45		ns
t_{HSR}	SCLK rising edge → Input data hold	0		0		0		ns
t_{SRD}	SCLK rising edge → Effective data input		$t_{SCY} - 2X - 150$		970		725	ns

4.9 Timing Chart



4.10 Timing Chart for I/O Interface Mode



5. TABLE OF SPECIAL FUNCTION REGISTERS (SFRs)

The special function registers (SFRs) include the I/O ports, peripheral control registers and bank registers (BX and BY) allocated to the 56-byte addresses 0FFC0H~0FFF7H.

Configuration of the table

Symbol	Name	Address	7	6			1	0	
									→ bit Symbol
									→ Read / Write
									→ Initial value after reset
									→ Remarks

141190

Symbol	Name	Address	7	6	5	4	3	2	1	0
P2	Port2	FFC4	P27	P26	P25	P24	P23	P22	P21	P20
			R	R/W						
			Input only	1	1	1	0	0	0	0
P2CR	Port2 Control Reg.	FFC5	—	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0: Output Port 1: Address/CS							
P3	Port3	FFC6	P37	P36	P35	P34	P33	P32	P31	P30
			R/W							
			Input mode							
P3CR	Port3 Control Reg.	FFC7	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT (I/O selected bit by bit)							
P23FR	Port2, 3 Function Reg.	FFCE	ODE	TXDC	SCLKC	Fixed to "0".	Fixed to "0".	IOCS	RAMCS	ROMCS
			R/W							
			0	0	0	0	0	0	0	0
			P37 control 0: CMOS 1: Open Drain	P37 control 0: Port 1: TxD output	P36 control 0: Port 1: SCLK output			P26 control 0: 1: $\overline{IOCS}$ output	P25 control 0: 1: $\overline{RAMCS}$ output	P24 control 0: 1: $\overline{ROMCS}$ output
P4	Port4	FFC8	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			Input mode							
P4CR	Port4 Control Reg.	FFC9	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT (I/O selected bit by bit)							
P4FR	Port4 Function Reg.	FFCF	ZCE2	ZCE1	–	–	TO5S	TO4S	TO3S	TO1S
			R/W				R/W			
			0	0			0	0	0	0
			P47 control 1: ZCD Enable	P46control 1: ZCD Enable			P43control 0: Port 1: TO5	P42control 0: Port 1: TO4	P41control 0: Port 1: TO3	P40control 0: Port 1: TO1
P5	Port5	FFCA	Fixed to "1"	Fixed to "1"	–	–	P53	P52	P51	P50
			R/W				R			
			1	1			Input only			
							Shared with analog input pin (AN0 ~ AN3)			

Symbol	Name	Address	7	6	5	4	3	2	1	0
P6	Port6	FFCB	SA63	SA62	SA61	SA60	P63	P62	P61	P60
			R/W				R/W			
			Undefined				Input mode			
			Shift Alternate reg. 0				• Stepping Motor Control Port 0 • Pattern Generation Port 0			}Shared with
P7	Port7	FFCC	SA73	SA72	SA71	SA70	P73	P72	P71	P70
			R/W				R/W			
			Undefined				Input mode			
			Shift Alternate reg. 1				• Stepping Motor Control Port 1 • Pattern Generation Port 1			}Shared with
P67CR	Port6, 7 Control Reg.	FFCD	P73C	P72C	P71C	P70C	P63C	P62C	P61C	P60C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT (I/O selected bit by bit)							
P67FR	Port6, 7 Function Reg.	FFD0	PAT1	CCW1	M1M	M1S	PAT0	CCW0	M0M	M0S
			R/W							
			0	0	0	0	0	0	0	0
			0: 8Bit 1: 4Bit	0: Normal rotation 1: Reverse rotation (For port 7)	0: 4Step 1: 8Step	0: Port 1: Step	0: 8Bit 1: 4Bit	0: Normal rotation 1: Reverse rotation (For port 6)	0: 4Step 1: 8Step	0: Port 1: Step
P25FR	Port2, 5 Function Reg.	FFD1	–	–	–	–	–	RDE	WAITC1	WAITC0
								R/W		
								0	0	0
								RD control 1:Always RD output	WAIT control 00: 2 state wait 01: Nomal Wait 10: Non wait 11: –	
WDMOD	Watch Dog Timer Mode Reg.	FFD2	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	EXF	DRIVE
			R/W						R	R/W
			1	0	0	0	0	0	Undefined	0
			1: WDT Enable	WDT detecting time 00: 2^{16}/fc 01: 2^{18}/fc 10: 2^{20}/fc 11: 2^{22}/fc		Warming up time 0: 2^{14}/fc 1: 2^{16}/fc	Standby mode 00: RUN 01: STOP 10: IDLE1 11: –		Inverts each time EXX instruction is executed.	1:To drive the pin even in STOP mode.
WDCR	Watch Dog Timer Control Reg.	FFD3	–							
			W							
			–							
			B1H: WDT Disable Code 4EH: WDT Clear Code							

Symbol	Name	Address	7	6	5	4	3	2	1	0
PCSR	Programmable CS Reg.	FFDE	V16	V15	V14	S18	S17	S16	S15	S14
			R/W							
			1	1	1	1	1	1	1	1
			1: S16 Valid	1: S15 Valid	1: S14 Valid	A18 Set	A17 Set	A16 Set	A15 Set	A14 Set
EXPA0	EXpand Program Area Reg.	FFDF	EX7	EX6	EX5	EX4	EX3	EX2	EX1	EX0
			R/W							
			0	0	0	0	0	0	1	0
			Program area expanding bit (corresponds to A14~A21)							

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREG0	8bit Timer Reg 0	FFD4	–							
			W							
			Undefined							
TREG1	8bit Timer Reg 1	FFD5	–							
			W							
			Undefined							
TREG2	8bit Timer Reg 2	FFD6	–							
			W							
			Undefined							
TREG3	8bit Timer Reg 3	FFD7	–							
			W							
			Undefined							
T01MOD	Timer 0, 1 Mode Reg.	FFD8	T10M1	T10M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8 Bit Timer 01: 16 Bit Timer 10: 8 Bit PPG 11: 8 Bit PWM		00: – 01: 2^6 – 1 PWM 10: 2^7 – 1 Cycle 11: 2^8 – 1		00: TO0TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI0 01: ϕT1 10: ϕT4 11: ϕT16	
T23MOD	Timer 2, 3 Mode Reg.	FFD9	T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8 Bit Timer 01: 16 Bit Timer 10: 8 Bit PPG 11: 8 Bit 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	
TFFCR	8bit Timer Flip-Flop Control Reg.	FFDA	TFF3C1	TFF3C0	TFF3IE	TFF3IS	TFF1C1	TFF1C0	TFF1IE	TFF1IS
			W		R/W		W		R/W	
			–		0	0	–		0	0
			00: Invert TFF3 01: Set TFF3 10: Clear TFF3 11: Dont Care.		1: TFF3 Invert Enable	1: Inverts by timer 2.	00: Invert TFF1 01: Set TFF1 10: Clear TFF1 11: Dont Care.		1: TFF1 Invert Enable	1: Inverts by timer 0.
TRUN	Timer Run Control Reg.	FFDC			PRRUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
					R/W					
					0	0	0	0	0	0
					Prescaler & Timer RUN/STOP Control 0: Stop & Clear 1: RUN					
TRDC	Timer Reg. Double Buffer Control Reg.	FFDB	M1T					TR4DE	TR2DE	TR0DE
			R/W					R/W		
			0					0	0	0
			0:Timer 2 or timer 3 1: Timer 4					Timer Reg. Double Buffer Control 0: Double Buffer Disable 1: Double Buffer Enable		

Symbol	Name	Address	7	6	5	4	3	2	1	0
CAP1L	Capture Reg. 1	FFE0	—							
			R							
			Undefined							
CAP1H		FFE1	—							
			R							
			Undefined							
CAP2L	Capture Reg. 2	FFE2	—							
			R							
			Undefined							
CAP2H		FFE3	—							
			R							
			Undefined							
TREG4L	16Bit Timer Reg. 4	FFE0	—							
			W							
			Undefined							
TREG4H		FFE1	—							
			W							
			Undefined							
TREG5L	16Bit Timer Reg. 5	FFE2	—							
			W							
			Undefined							
TREG5H		FFE3	—							
			W							
			Undefined							
T4MOD	16Bit Timer Mode Reg. 5	FFE4	CAP2T5	EQ5T5	CAP1IN	CAPM1	CAPM0	CLE	T4CLK1	T4CLK0
			R/W		W	R/W		R/W	R/W	
			0	0	—	0	0	0	0	0
			TFF5 inversion trigger 0: Disable trigger 1: Enable trigger		0: Soft-Capture	Capture Timing 00: Disable 01: TI4 ↑T15 ↑ 10: TI4 ↑TI4 ↓ 11: TFF1 ↑TFF1 ↓		1: Timer 4 Clear Enable	Timer 4 clock 00: TI4 01: ϕT1 10: ϕT4 11: ϕT16	
T4FFCR	16Bit Timer F/F Control Reg. 5	FFE5	TFF5C1	TFF5C0	CAP2T4	CAP1T4	EQ5T4	EQ4T4	TFF4C1	TFF4C0
			W		R/W				W	
			—		0	0	0	0	—	
			00: Invert TFF5 01: Set TFF5 10: Clear TFF5 11: Don't Care		TFF4 inversion trigger 0: Disable trigger 1: Enable trigger				00: Invert TFF4 01: Set TFF4 10: Clear TFF4 11: Don't Care	

*) CAP1 and TREG4 as well as CAP2 and TREG5 are allocated to the same address.

Symbol	Name	Address	7	6	5	4	3	2	1	0
SCMOD	Serial Channel Mode Reg.	FFE6	TB8	Fixed to "0"	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			Undefined	0	0	0	0	0	0	0
			Transmission bit-8 data		1: Receive Enable	1: Wake Up Enable	00: I/O Interface 01: UART 7Bit 10: UART 8Bit 11: UART 9Bit		00: TO2TRG 01: BRG Mode. 10: ϕ1 11: —	
SCCR	Serial Channel Control Reg.	FFE7	RB8	EVEN	PE	OERR	PERR	FERR	SCLKC	IOC
			R	R/W		R (Cleared to "0" by reading)			R/W	
			Undefined	0	0	0	0	0	0	0
			Bit 8 of receiving data	Parity 0: Odd 1: EVEN	1: Parity Enable	Overrun	1: Error Parity	Framing	0: SCLK (↓) 1: SCLK (↑)	0: SCLK output 1: SCLK input
SCBUF	Serial Channel Buffer Reg.	FFE8	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
			TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0
			R (Receiving) /W (Transmission)							
			Undefined							
ADMOD	A/D Converter Mode Reg.	FFEF	EOCF	ADBF	REPET	SCAN	ADCS	ADS	ADCH1	ADCH0
			R		R/W	R/W	R/W		R/W	
			0	0	0	0	0	0	0	0
			1: END	1: BUSY	1: Repeat Mode Set	1: Scan Mode Set	1: Low speed mode	1: START	Analog Input Channel Select	
ADREG0	A/D Result Reg. 0	FFF0	—							
			R							
			—							
ADREG1	A/D Result Reg.1	FFF1	—							
			R							
			—							
ADREG2	A/D Result Reg.2	FFF2	—							
			R							
			—							
ADREG3	A/D Result Reg. 3	FFF3	—							
			R							
			—							
BX	Bank Reg. X	FFFC		BX6	BX5	BX4	BX3	BX2	BX1	BX0
			R/W							
				0	0	0	0	0	0	0
BY	Bank Reg. Y	FFED		BY6	BY5	BY4	BY3	BY2	BY1	BY0
			R/W							
				0	0	0	0	0	0	0

Symbol	Name	Address	7	6	5	4	3	2	1	0
BRGCR	Baud Rate Generater Control Reg.	FFE9	Fixed to "0"		BG1	BG0	PS3	PS2	PS1	PS0
			R/W							
			0		0	0	0	0	0	0
					00: fc/4 01: fc/16 10: fc/64 11: fc/256		Divided frequency from prescaler			
INTEL	Interrupt Enable Mask Reg.	FFF4	IET4	IE1	IET5	IE2	IERX	IETX	Fixed to "0"	Fixed to "0"
			R/W							
			0	0	0	0	0	0	0	0
			1: Enable				0: Disable			
INTEH		FFF5		EDGE	ADIS	IE0	IET0	IET1	*IET2	IET3
				R/W	R/W	R/W				
				0	0	0	0	0	0	0
				INT0 0: Level 1: EDGE	1: INTAD	1: Enable		0: Disable		
DMAEL	Micro DMA Enable Reg.	FFF6	DET4	DE1	DET5	DE2	DERX	DETX	Fixed to "0"	Fixed to "0"
			R/W							
			0	0	0	0	0	0	0	0
			1: Enable				0: Disable			
DMAEH		FFF7				DE0	DET0	DET1	DET2	DET3
						R/W				
						0	0	0	0	0
						1: Enable		0: Disable		
IRFH	Interrupt Request Flag & IRF Clear	FFEB				IRF0	IRFT0	IRFT1	IRFT2	IRFT3
						R				
						0	0	0	0	0
						Interrupt Request Flag 1: Interrupt being requested				
IRFL		FFEA	IRFT4	IRF1	IRFT5	IRF2	IRFRX	IRFTEX	–	–
			R (Only IRF clear code can be used to write)							
			0	0	0	0	0	0	0	0
			1: Interrupt being requested (IRF is cleared to "0" by writing IRF clear code.)							

Address	Symbol	Address	Symbol
FFC0	(Reserved)	FFE0	CAP1L/TREG4L
FFC1	(Reserved)	FFE1	CAP1H/TREG4H
FFC2	(Reserved)	FFE2	CAP2L/TREG5L
FFC3	(Reserved)	FFE3	CAP2H/TREG5H
FFC4	P2	FFE4	T4MOD
FFC5	P2CR	FFE5	T4FFCR
FFC6	P3	FFE6	SCMOD
FFC7	P3CR	FFE7	SCCR
FFC8	P4	FFE8	SCBUF
FFC9	P4CR	FFE9	BRGCR
FFCA	P5	FFEA	IRFL
FFCB	P6	FFEB	IRFH
FFCC	P7	FFEC	BX
FFCD	P67CR	FFED	BY
FFCE	P23FR	FFEE	(Reserved)
FFCF	P4FR	FFEF	ADMOD
FFD0	P67FR	FFF0	ADREG0
FFD1	P25FR	FFF1	ADREG1
FFD2	WDMOD	FFF2	ADREG2
FFD3	WDCR	FFF3	ADREG3
FFD4	TREG0	FFF4	INTEL
FFD5	TREG1	FFF5	INTEH
FFD6	TREG2	FFF6	DMAEL
FFD7	TREG3	FFF7	DMAEH
FFD8	T01MOD		
FFD9	T23MOD		
FFDA	TFFCR		
FFDB	TRDC		
FFDC	TRUN		
FFDD	(Reserved)		
FFDE	PCSR		
FFDF	EXPA0		

Writing to the (Reserved) register is disabled.