

TMPZ84C011AF-6 TLCS-Z80 MICROPROCESSOR

1. OVERVIEW AND FEATURES

The TMPZ84C011A is a high-performance CMOS 8-bit microprocessor that contains the functional peripherals such as counter timer circuit (CTC) parallel I/O port, and clock generator/controller (CGC) around the TLCS-Z80 MPU. This microprocessor directly inherits the basic architecture of the TLCS-Z80 MPU, making available the software resources and development tools accumulated so far.

The TMPZ84C011A is based on the new CMOS process and housed in a standard 100-pin mini-flat package, greatly contributing to system minituarization and power saving.

The TMPZ84C011A has many I/O ports, making it available for a wide range of system applications such as the controller, the measuring instrument, and the word processor.

Features:

- Built-in MPU, CTC, and CGC functions of the TLCS-Z80 family.
- High speed operation (6 MHz operation)
- Built-in clock generator (CGC. Clock Generator/Controller)
- Built-in standby capability (with the controller built in)
- Low power consumption

operation	15 mA typ. (4 MHz) 22mA typ. (6MHz)
idle	1.0 mA typ. (4 MHz) 1.5mA typ. (6MHz)
standby.....	500 nA typ.
- Wide operational power voltage range ($5V \pm 10\%$, 3 to 6V operation supported)
- Wide operational temperature range (-40°C to $+85^{\circ}\text{C}$)
- Three selectable operational modes

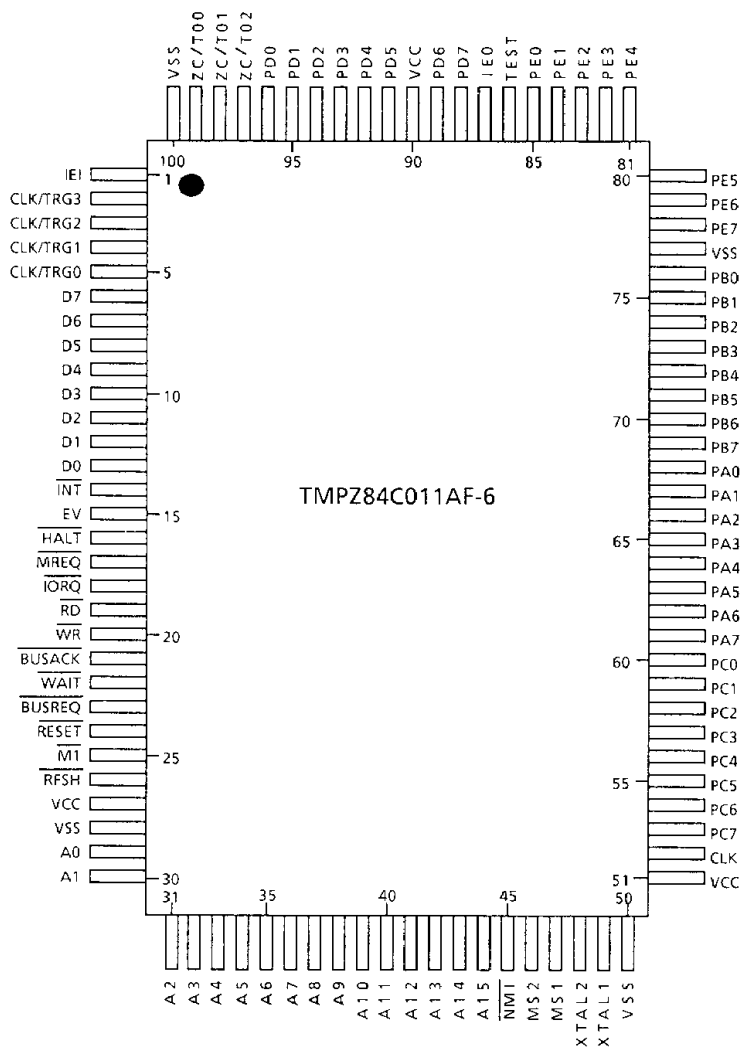
Run mode	Normal operation
Idle mode	Only clock generation goes on
Stop mode	Clock generation stops; standby state
- Many general-purpose ports (5 ports, 40 bits) which support I/O operation on a bit basis.
- Four channels of timer/counter (the same capability as the TLCS-Z80 CTC).
- Built-in dynamic RAM refresh controller
- Daisy-chain interrupt control pin

- Housed in a compact 100-pin mini-flat package
- Emulation by Z80 ICE (ex. Toshiba Real Time Emulator RTE80) is available.
- The Toshiba evaluator board (BM8024) and adapter board (BM8026) are available.

note : Z80 is a trade mark of Zilog Inc.

2. PIN ASSIGNMENTS AND FUNCTIONS

2.1 PIN ASSIGNMENTS (TOP VIEW)



110489

Figure 2.1.1 Pin Assignments

(1/3)

Pin	Number	Type	Function
D0-D7	8 (6-13)	I/O 3-state	The 8-bit bi-directional data bus.
A0-A15	16 (29-44)	Output 3-state	The 16-bit address bus. These pins specify memory and I/O port addresses. During a refresh cycle, the refresh address is output to the low-order 7 bits.
PA0-PA7	8 (61-68)	I/O 3-state	The general-purpose I/O port (port A) which can be set by program for input or output on a bit basis. When it is set for output, it provides latch output.
PB0-PB7	8 (69-76)	I/O 3-state	The general-purpose I/O port (port B) which can be set by program for input or output on a bit basis. When it is set for output, it provides latch output.
PC0-PC7	8 (53-60)	I/O 3-state	The general-purpose I/O port (port C) which can be set by program for input or output on a bit basis. When it is set for output, it provides latch output.
PD0-PD7	8 (88-89) (91-96)	I/O 3-state	The general-purpose I/O port (port D) which can be set by program for input or output on a bit basis. When it is set for output, it provides latch output.
PE0-PE7	8 (78-85)	I/O 3-state	The general-purpose I/O port (port E) which can be set by program for input or output on a bit basis. When it is set for output, it provides latch output.
$\overline{M1}$	1 (25)	Output 3-state	Machine cycle 1 signal pin. In an Op code fetch cycle, this pin goes "0" with the \overline{MREQ} signal. At the execution of a 2-byte Op code, this pin goes "0" for each operation fetch. In a maskable interrupt acknowledge cycle, this pin goes "0" with the \overline{IORQ} signal. When the EV signal is applied, this pin is put in the high-impedance state.
\overline{RD}	1 (19)	Output 3-state	The read signal pin. This signal indicates that the MPU is ready for accepting data from memory or an I/O device. The data from the addressed memory or I/O device is gated by this signal onto the MPU data bus. When the \overline{BUSREQ} signal is applied, this pin is put in the high impedance state.
\overline{WR}	1 (20)	Output 3-state	The write signal pin. This signal active when the data to be stored in the addressed memory or I/O device is on the data bus. When the \overline{BUSREQ} signal is applied, this pin is put in the high-impedance state.
\overline{MREQ}	1 (17)	Output 3-state	The memory request signal pin. When the execution address for memory access is on the address bus, this pin goes "0". During a memory refresh cycle, this pin also goes "0" with the \overline{RFSH} signal.

110489

(2/3)

Pin	Number	Type	Function
$\overline{\text{IORQ}}$	1 (18)	Output 3-state	The input/output request signal pin. This pin goes "0" when the address for an I/O operation is on the low-order 8 bits (A0 through A7) for the address bus. The $\overline{\text{IORQ}}$ signal is also output with the $\overline{\text{MT}}$ signal at interrupt acknowledge to tell an I/O device that it can put the interrupt response vector onto the data bus.
$\overline{\text{WAIT}}$	1 (22)	Input	The wait request signal pin. This signal indicates to MPU that the addressed memory or I/O device is not ready for data transfer. As long as this signal is "0", the MPU continues to be in the wait state.
$\overline{\text{BUSREQ}}$	1 (23)	Input	The bus request signal pin. The $\overline{\text{BUSREQ}}$ signal forces the MPU address bus, data bus, and control signals $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ to get in the high-impedance state. This signal is normally wire-ORed and required an external pullup resistor for these applications.
$\overline{\text{BUSACK}}$	1 (21)	Output	The bus acknowledge signal pin. In response to the $\overline{\text{BUSREQ}}$ signal, the $\overline{\text{BUSACK}}$ signal indicates to the requesting peripheral LSI that the MPU address bus, data bus, and control signals $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ have been put in the high-impedance state.
$\overline{\text{HALT}}$	1 (16)	Output 3-state	The halt signal pin. This pin goes "0" when the MPU has executed a HALT instruction and is in the halt state. It is put in the high-impedance state when the EV signal is applied.
$\overline{\text{RFSH}}$	1 (26)	Output	The refresh signal pin. When the dynamic memory refresh address is on the low-order 7 bits of the address bus, this signal goes "0". At the same time, the $\overline{\text{MREQ}}$ signal also goes active ("0").
EV	1 (15)	Input	The evaluator signal pin. When "1" is applied to this pin, the $\overline{\text{MT}}$ and $\overline{\text{HALT}}$ pins are put in the high-impedance state. In using the TMPZ84C011A for an evaluator chip, giving the signal with $\overline{\text{BUSREQ}}$ electrically disconnects the MPU section and the chip operates following the instructions from other MPUs (such as the MPU of the ICE).
TEST	1 (86)	Input	The test signal pin. Normally, this pin should be "0".
CLK/TRG0 ┆ CLK/TRG3	4 (2-5)	Input	The external clock/timer trigger input pins. These 4 CLK/TRG pins correspond to 4 channels. In the counter mode, the down-counter is decremented by 1 and, in the timer mode, the timer is activated by each active edge (a rising or falling edge) which are input at these pins. Whether the active edge is a rising or falling edge can be selected by program.
ZCT/O0 ┆ ZCT/O2	3 (97-99)	Output	The zero count/time out signal output pin. In either the timer mode or counter mode, pulses are output from these pins when the counter value has reached zero.

110489

(3/3)

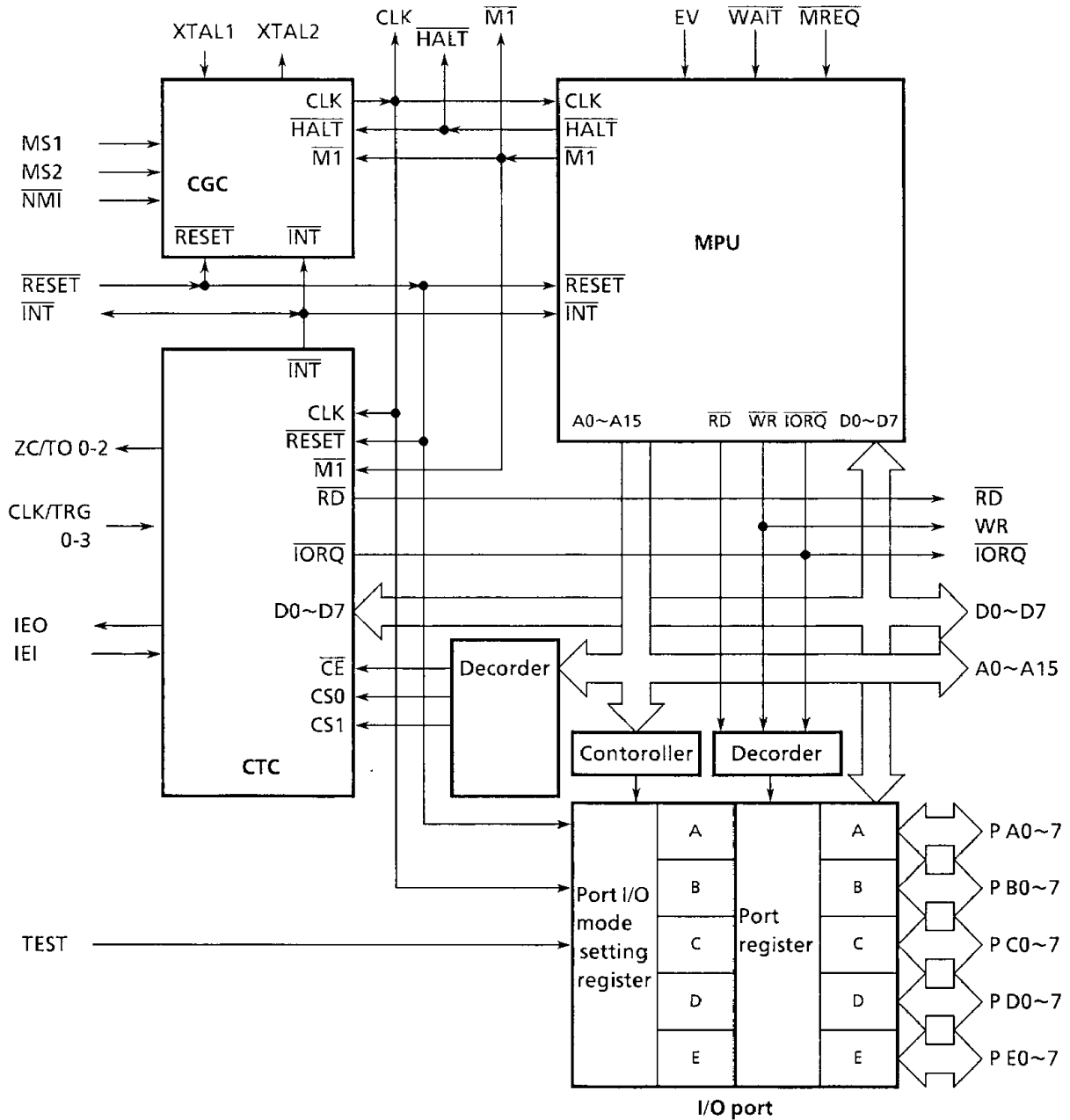
Pin	Number	Type	Function
IEI	1 (1)	Input	The CTC interrupt enable input signal pin. This signal indicates whether there is an interrupt by an upper peripheral LSIs in daisy-chaining.
IEO	1 (87)	Output	The CTC interrupt enable output signal pin. This signal controls the interrupts by lower peripheral LSIs in daisy chaining. This signal goes high only when the IEI pin is high and the MPU is not servicing the interrupt from the channel in the CTC.
XTAL1 XTAL2	2 (49) (48)	Input Output	The crystal oscillator connection pins. Connect a resonator whose oscillation frequency is double the system clock (CLK output) frequency.
MS1 MS2	2 (47) (46)	Input	The mode select input pins. The states of these 2 pins determine one of the 3 modes (Run, Idle, and Stop).
CLK	1 (52)	Output	The single-phase clock output pin. When the HALT instruction is executed in the Stop or Idle mode, the MPU and the CTC stop operating with the clock (CLK) output held at "0". This output is used for the clock to other peripheral LSIs
$\overline{\text{RESET}}$	1 (24)	Input	The reset signal input pin. This signal is commonly given to the MPU, CTC, I/O and CGC. It is also used for the restart signal from the halt state.
$\overline{\text{INT}}$	1 (14)	Input	The maskable interrupt request signal pin. The interrupt is caused by the CTC or peripheral LSIs. The interrupt is accepted when the interrupt enable flip-flop (IFF) is set to "1" by software. The $\overline{\text{INT}}$ pin is normally wire-ORed, requiring to attach a pullup resistor externally. This signal is also used for the restart signal from the halt state.
$\overline{\text{NMI}}$	1 (45)	Input	The non-maskable interrupt request signal pin. This interrupt request has a higher priority than a maskable interrupt and independent of the interrupt enable flip-flop (IFF) state.
VCC	3 (27, 51, 90)		The power supply pin (+ 5 V).
VSS	4 (28, 50, 77, 100)		The ground pin (0 V).

110489

3. OPERATIONAL DESCRIPTIONS

3.1 ENTIRE BLOCK DIAGRAM AND OPERATION OF EACH BLOCK

3.1.1 Entire Block Diagram



110489

Figure 3.1.1 Entire Block Diagram

3.1.2 Operation of Each Block

The TMPZ84C011A largely consists of a processor (MPU), a counter/timer circuit (CTC), an input/output port section (I/O), and a clock generator/controller (CGC).

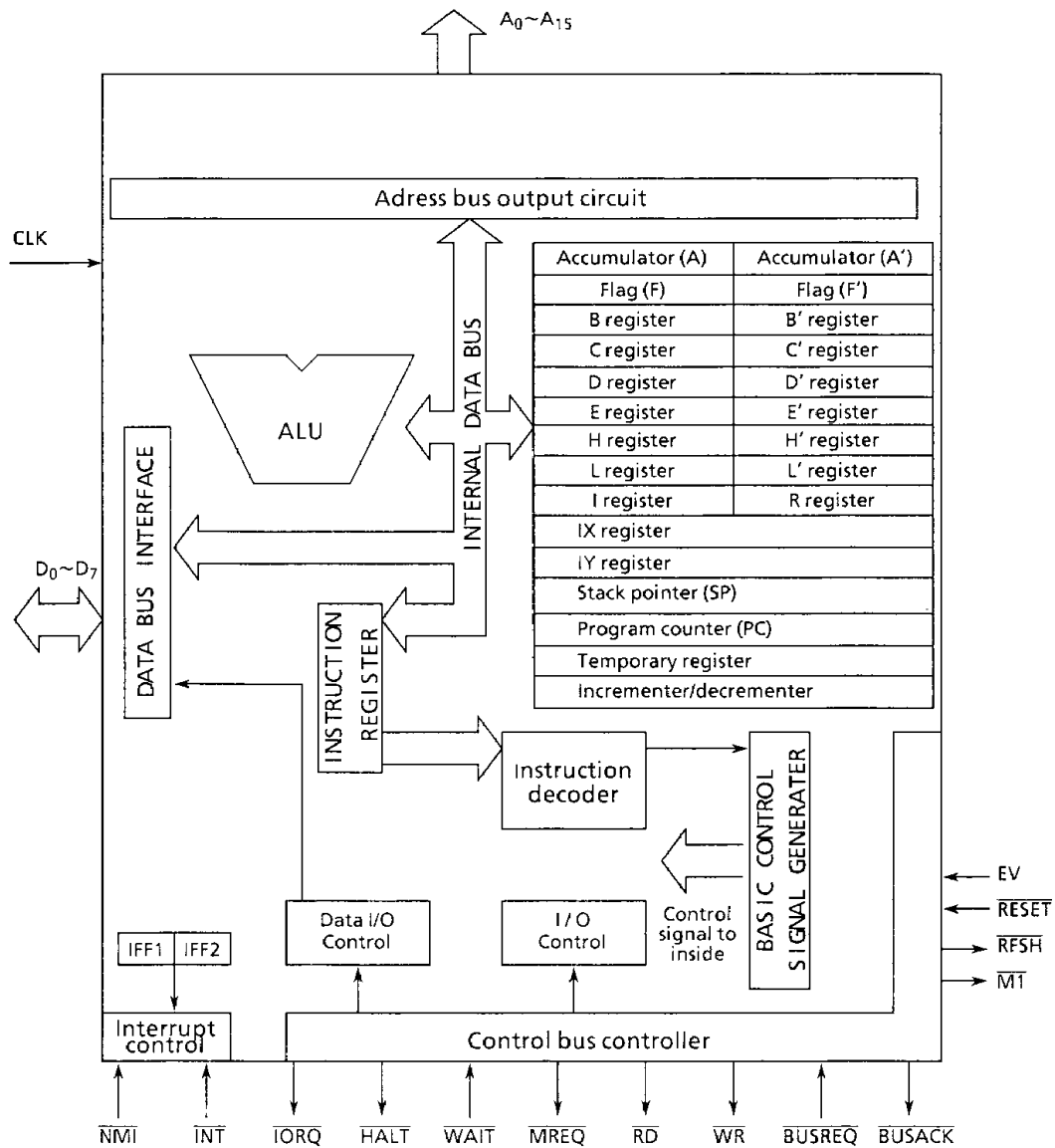
- The MPU has all pins of the Toshiba TLCS-Z80 MPU, and its function is fully compatible with TLCS-Z80 microprocessor.
- The CTC provides the capabilities of the Toshiba TLCS-Z80 CTC (TMPZ84C30A) and has the pins required to perform the necessary operations as a TLCS-Z80 peripheral LSI. The four independent timer channels are I/O-addressed internally. It also has the IEI and IEO pins required for the daisy-chain interrupt to provide daisy-chaining with other peripheral LSIs.
- The I/O consists of 5 general-purpose I/O ports, each made up of 8 bits. Each port can be specified by program for input or output on a bit basis. The 5 ports are assigned with I/O addresses, which are specified by program to make the ports perform I/O operations. When these ports are used for output ports, the output is latched, so that the data once output is held as it is until changed.
- The CGC provided the 3 kinds of operation modes to control the entire TMPZ84C011A chip; the Run, Idle, and Stop modes. The CGC provides the same function as the Toshiba TLCS-Z80 CGC (TMPZ84C60) and has the pin to select the Run, Idle, or Stop mode.
 - In the Run mode, the clock generated by the CGC is supplied to the TMPZ84C011A and peripheral LSIs to perform the normal programmed microcomputer operations.
 - In the Idle mode, clock oscillation is going on but the clock is not supplied to the TMPZ84C011A and peripheral LSIs, thereby saving the system power consumption. This mode shortens the time required for system restart.
 - In the Stop mode, clock oscillation is not performed and the system operation can be stopped completely. In this mode, the system can be restarted with the internal data retained with an extremely low power consumption level unique to the CMOS technology. Note that these modes can be set only when the MPU has executed a HALT instruction.
- Additionally, the TMPZ84C011A has also the EV pin which is used to put the MPU in the high-impedance state for electrical disconnection, thus providing an evaluator chip. That is, the MPU in the TMPZ84C011A is electrically disconnected by the EV pin and the $\overline{\text{BUSREQ}}$ pin to implement the emulation by the signal from the in-circuit emulator (ICE).

3.2 MPU OPERATIONS

This subsection describes the system configuration, functions and the basic operations of the MPU of the TMPZ84C011A.

3.2.1 Block Diagram

Figure 3.2.1 shows the block diagram of the MPU.



270489

Figure 3.2.1 MPU Block Diagram

3.2.2 MPU System Configuration

The MPU has the configuration shown in Fig. 3.2.1. The address signal is put on the address bus via the address buffer. The data bus is controlled for input or output by the data bus interface. Both the address and data buses are put in the high-impedance state by the $\overline{\text{BUSREQ}}$ signal input to make them available for other peripheral LSIs. The Opcode read from memory via the data bus is written to the instruction register. This Opcode is decoded by the instruction decoder. According to the result of the decoding, control signals are sent to the relevant devices. Receiving these control signals, the ALU performs various arithmetic operations. The register array temporarily holds the information required to perform a given operation.

The following describes the MPU's internal registers and functions:

[1] Internal Register Groups

The configuration of the internal register groups is as follows:

(1) Main registers

A, F, B, C, D, E, H, L

(2) Alternate registers

A', F', B', C', D', E', H', L'

(3) Special registers

I, R, IX, IY, SP, PC

Figure 3.2.3 shows the configuration of the internal register groups. The register groups, each being of a static RAM, consists of eighteen 8-bit registers and four 16-bit registers. The following describes the function of each register:

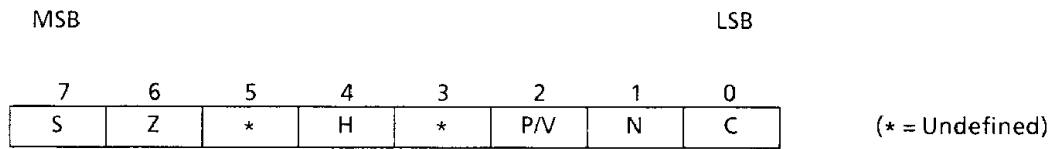
(1) Main registers (A, F, B, C, D, E, H, L)

(a) Accumulator (A)

The accumulator is an 8-bit register used for arithmetic and data transfer operations.

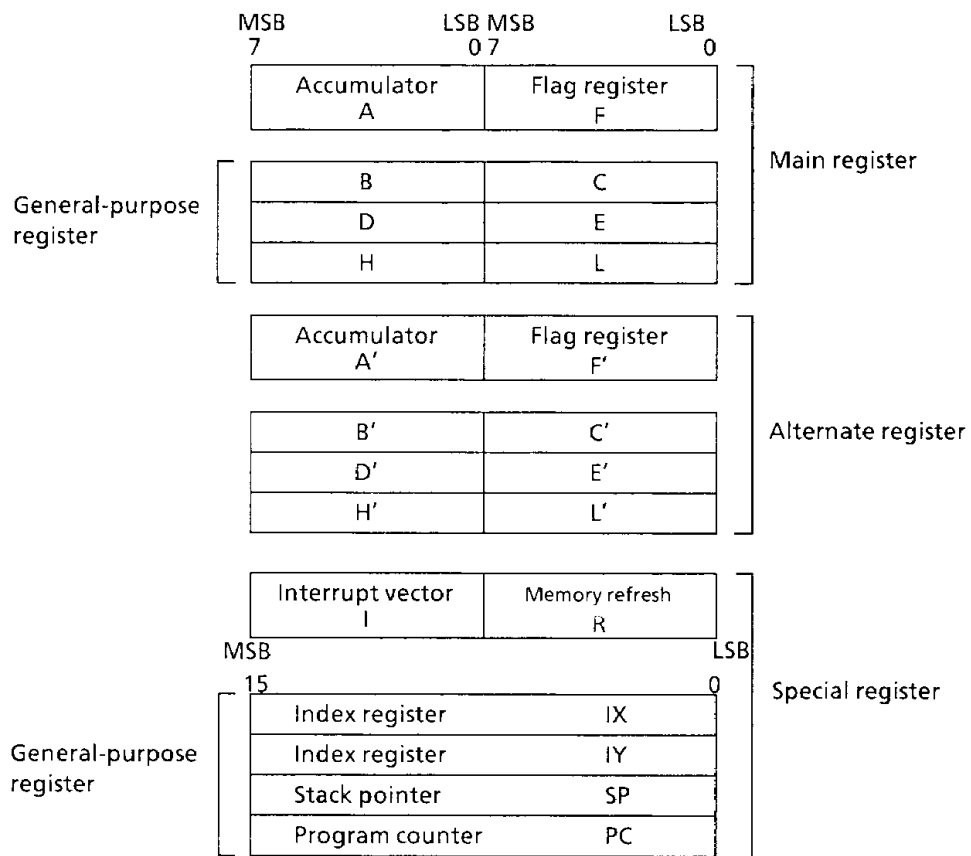
(b) Flag register (F) (see Figure 3.2.2)

The flag register is an 8-bit register to hold the result of each arithmetic operation. Actually, the 6 of the 8 bits are set ("1")/reset ("0") according to the condition specified by an instruction.



270489

Figure 3.2.2 Flag Register Configuration



110489

Figure 3.2.3 Flag Register Configuration

The following 4 bits are directly available to the programmer for setting the jump, call and return instruction conditions:

- Sign flag (S)

When the result of an operation is negative, the S flag is set to "1". Actually, the content of bit 7 of accumulator is stored in this flag.

- Zero flag (Z)

When all bits turn out to be "0" after operation, the Z flag is set to "1". Otherwise, it is set to "0". With a block search instruction (CPI, CPIR, CPD or CPDR), the Z flag is set to "1" if the source data and the accumulator data match. With a block I/O instruction (INI, IND, OUTI or OUTD), the Z flag is set to "1" if the content of the B register used as the byte counter is "0" at the end of comparison.

- Parity overflow flag (P/V)

This flag has two functions. One is the parity flag (P) that indicates the result of a logic operation (AND A,B etc.). The P flag is set to "1" if the parity is even as a result of the operation on signed values by two's complement. It is set to "0" if the parity is odd. With a block search instruction (CPI, CPIR, CPD or CPDR) and a block transfer instruction (LDI or LDD), the P flag indicates the state of the byte counter (register pair BC). It is set to "1" if the byte counter is not "0" and to "0" when the byte counter becomes "0" (at the end of comparison or data transfer). The content of the interrupt enable flip-flop (IFF) is saved to the P flag when the contents of the R register or I register are transferred to the accumulator.

The other use of the P/V flag is the overflow flag (V) that indicates whether an overflow has occurred or not as a result of an arithmetic operation. The V flag is set to "1" when the value in the accumulator gets out of a range of the maximum value +127 and the minimum value -128 and therefore cannot be correctly represented as a two's complement notation.

Whether the P/V flag operates as the P flag or V flag is determined by the type of the instruction executed.

- Carry flag (C)

The C flag is set to "1" if a carry occurs from bit 7 of the accumulator or a borrow occurs as a result of an operation.

The following two flags are not available to the programmer for the test and set ("1")/reset ("0") purposes. They are internally used by the MPU for BCD arithmetic operations.

- Half carry flag (H)

The H flag is used for holding the carry or borrow from the low-order 4 bits of a BCD operation result. When a DAA instruction (decimal adjust) is executed, the MPU automatically uses the H flag to adjust the result of a decimal addition or subtraction.

- Add/subtract flag (N)

In BCD operation, algorithm is different between addition and subtraction. The N flag indicates whether the executed operation is addition or subtraction.

For how the flags change depending on the instruction, see 3.2.4 "TMPZ84C011A Instruction Set".

(c) General-purpose registers (B, C, D, E, H, L)

General-purpose registers consist of 8 bits each. They are used as 16-bit register pairs (BC, DE, HL) as well as separate 8-bit registers to supplement the accumulator. The B register and the register pair BC are used as a counter when a block I/O, block transfer, or search instruction is executed. The register pair HL has various memory addressing features as compared with the register pairs BC and DE.

(2) Alternate registers (A', F', B', C', D', E', H', L')

The configuration of the alternate registers is exactly the same as that of the main registers. There is no instruction that handles the alternate registers directly. The data in the alternate registers are processed by moving them into the main registers by means of exchange instructions as shown below:

EX AF, AF' (A↔A', F↔F')

EXX (B↔B', C↔C', D↔D', E↔E', H↔H', L↔L')

When a high-speed interrupt response has been requested within the system, these instructions can be used to quickly move the data from the accumulator, flag registers, and general-purpose registers on the main or alternate side into the corresponding registers. This eliminates the need for transferring the register contents to/from the external stack during execution of the interrupt handling routine, thereby shortening the interrupt servicing time greatly.

(3) Special registers (I, R, IX, IY, SP, PC)

(a) Interrupt page address register (I)

The TMPZ84C011A provides two kinds of interrupts: maskable interrupt (\overline{INT}) and non-maskable interrupt (\overline{NMI}). The maskable interrupt provides three modes (0, 1, and 2) in which the interrupt is handled. These modes can be selected by instructions IM0, IM1, and IM2 respectively. In mode 2, any memory location can be called indirectly depending on the interrupt. For this purpose, the I register stores the high-order 8 bits of the indirect address. The low-order 8 bits are supplied from the interrupting peripheral LSI. This scheme permits calling the interrupt handling routine from any memory location in an extremely short access time. For the details of interrupts, see (IV) "Interrupt Capability".

(b) Memory refresh register (R)

The R register is used as the memory refresh counter when the dynamic RAM is used for memory. This permits using of the dynamic memory in the same manner as the static memory. This 8-bit register is automatically incremented for each instruction fetch. While the MPU decoder executes the fetched instruction, the contents of the R register are synchronized with the refresh signal to place the low-order 7 bits on the address bus. This operation is all performed by the MPU and, therefore, need not perform a special processing by program. The MPU operation is not delayed by this operation. During refresh, the contents of the I register are placed on the high-order 8 bits of the address bus.

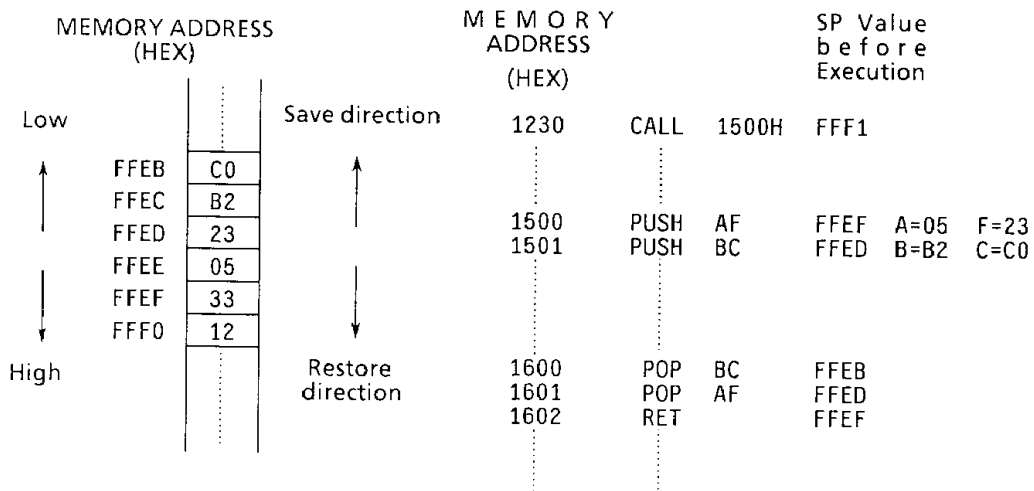
(c) Index registers (IX, IY)

The two independent index registers IX and IY holds the 16-bit base address when used in the index addressing mode. In this addressing mode, the memory address obtained by adding the contents of an index register to the displacement value (for example, LD IX + 40H) is specified. This mode is convenient for using data tables. Also these registers can be used separately for memory addressing and data retaining registers.

(d) Stack pointer (SP)

The stack pointer is a 16-bit register to provide the start address information in the stack area in the external RAM. The content of the stack pointer is decremented at the execution of a call instruction or PUSH instruction or interrupt handling and is incremented at the execution of a return instruction or POP instruction. At the execution of a call instruction or interrupt handling, the current content of the program counter is saved into the stack. At the execution of a return instruction, the content is restored from the stack to the program counter. These operations are all performed by the MPU automatically. However, the other registers are not saved or restored automatically. For the storing of these registers, alternate register exchange instruction (EX and EXX) or a PUSH and a POP instructions must be used. When a PUSH instruction is executed, the contents of the specified register are saved into the stack. When a POP instruction is executed, the contents of the stack are moved to the specified register.

These data are restored on a last-in, first-out basis. Use of the stack permits processing of multiple-level interrupts, very deep subroutine nestings, and various data manipulation very easily. The stack pointer is not initialized in the hardware approach. Therefore, it is required to specify initialization (to the highest address in the stack) in the initialization program by allocating the stack area in RAM in during programming.



The above example shows the stack pointer and stack operations in which the instructions starting with the CALL at address 1230H and ending at with the RET at address 1602H have been executed. However, it is assumed that there is no instruction or interrupt other than shown above that uses the above stack during the execution. When the value of the stack pointer before executing the CALL instruction at address 1230H indicates address FFF1H, address 1233H is stored at addresses FFF0H and FFEFH because the CALL instruction consists of 3 bytes, then the stack pointer is decremented. Similarly, the data are saved or restored sequentially according to the instructions. These stack and stack pointer operations are all performed automatically.

(e) Program counter (PC)

The program counter holds, in 16 bits, the memory address of the instruction to be executed next. The MPU fetches the instruction from the memory location indicated by the program counter. When the content of the program counter is put on the address bus, the program counter is incremented automatically. However, it is not incremented with a jump instruction, a call instruction, or interrupt processing. Instead, the specified new address set on it. With a return instruction, the content restored from the stack is set on the program counter. These operations are all performed automatically and therefore, transparent to the programmer.

[2] Halt capability

When a HALT instruction has been executed, the MPU is put in the halt state. The halt capability can be used to halt the MPU against the external interrupts, thereby reducing the power dissipation. In the halt state the states of MPU's internal registers are all retained. The halt state is cleared by reset or when an interrupt is accepted. For the details of halt operation, see [3] "Basic Timing".

(1) Halt operation

When a HALT instruction has been executed, the MPU sets the $\overline{\text{HALT}}$ signal to "0" to tell the outside that the MPU is going to get in the halt state. Actually, the MPU in the halt state continues executing NOPs if there is the system clock input. However, the program counter is not incremented. This keeps the refresh signal generated when the dynamic memory is used. During halt, the MPU's internal states are retained. The TMPZ84C011A contains the clock generator/controller, easily implementing the clock input control for these halt operations.

(2) Clearing the halt state

The halt state is cleared by accepting an interrupt (the $\overline{\text{INT}}$ or $\overline{\text{NMI}}$ signal input) or by reset (the $\overline{\text{RESET}}$ signal input). When an interrupt is accepted, the halt state is cleared and the interrupt handling routine is executed. However, a maskable interrupt (INT) cannot be accepted unless the interrupt enable flip-flop (IFF) is set.

Note that when the halt state is cleared by the $\overline{\text{RESET}}$ signal, the MPU is reset and the program counter is set to "0".

[3] RESET Signal

Holding the $\overline{\text{RESET}}$ pin at the low level ("0") under the following conditions, the MPU's internal states are reset:

- (1) The power voltage level is within the operational voltage range.
- (2) System clock stabilization.
- (3) Holding the $\overline{\text{RESET}}$ signal at the low level ("0") for at least 3 full clock cycles. When the $\overline{\text{RESET}}$ signal goes high ("1"), the MPU starts executing instructions from address 0000H after at least 2T state dummy cycles.

When reset, the MPU performs the following processing:

- Program counter
0000H is set.

- Interrupt

The interrupt enable flip-flop (IFF) is reset to "0" to disable the maskable interrupt. For the maskable interrupt processing, mode 0 is specified.

- Control output

All control outputs are made inactive ("1"). Therefore, the halt state is also cleared.

- Interrupt page address register (I register)

The content of the I register becomes 00H.

- Refresh register (R register)

The content of the R register becomes 00H.

The registers other than above and the external memory do not change. Therefore, they must be initialized as required.

[4] Interrupt Capability

The interrupt capability is used to pause the execution of the currently executed program upon request from a peripheral LSI, executing the requested processing before. Normally, this interrupt processing routine contains the data exchange and transfer of status and control information between the MPU and the peripheral LSI. When this routine has been completed, the MPU returns to the state before the interrupt was accepted.

The TMPZ84C011A provides the non-maskable interrupt (NMI) and maskable interrupt (INT) capabilities which are detected by the $\overline{\text{NMI}}$ and $\overline{\text{INT}}$ interrupt request signals, respectively. A non-maskable interrupt, when caused by a peripheral LSI, is accepted unconditionally. This interrupt is used to support critical functions such as the protection of the system from unpredictable happening including power outages. A maskable interrupt can be enabled or disabled by program. For example, if the timer is used and, therefore, an interrupt is not desired, the system can be programmed to disable the interrupt. Table 3.2.1 lists the processing by interrupt cause.

(1) Interrupt enable/disable

A non-maskable interrupt cannot be disabled by program, while a maskable interrupt can be enabled or disabled by program. The MPU has the interrupt enable flip-flop (IFF). A maskable interrupt can be enabled or disabled by setting this flip-flop to "1" (set) or "0" (reset) through an EI instruction (enable) or a DI instruction (disable) in program. Actually, the IFF consists of two flip-flops IFF1 and IFF2. IFF1 is used to select between the enable and disable of a maskable interrupt. IFF2 holds the state of IFF1 effective before a nonmaskable interrupt has been accepted. Both IFF1 and IFF2 are reset to "0" when any of the following conditions occurs, disabling an interrupt:

Actually, the waiting maskable interrupt request is accepted after the execution of the instruction that follows the EI instruction. This delay by one instruction is caused by accepting an interrupt after completion of the execution of a return instruction if the instruction following the EI instruction is a return instruction. In the following operations, the contents of IFF1 and IFF2 are the same.

- MPU reset
- Execution of DI instruction
- Acceptance of maskable interrupt

Both IFF1 and IFF2 are set to "1" when the following condition occurs, enabling an interrupt:

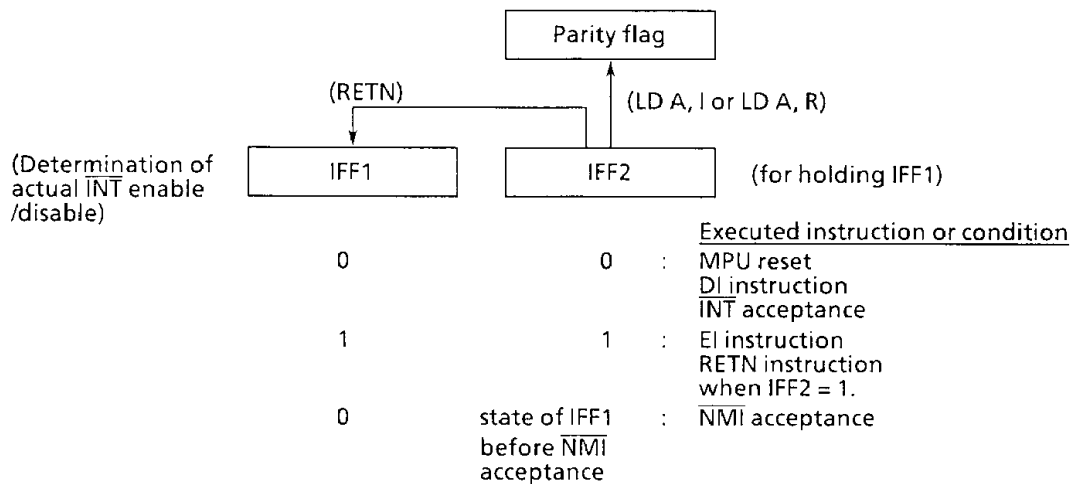
- Execution of EI instruction

Table 3.2.1 Processing by Interrupt Cause

Interrupt Source	Priority	Programmed condition		Vector address	Interrupt return instruction
Non-maskable interrupt (the falling edge of \overline{NMI})	1	None		Address 66H	RETN
Maskable interrupt (\overline{INT} becomes "0" at instruction's last clock)	2	IFF = 1	Mode 0	Instruction from peripheral LSI. Normally, CALL or RST instruction.	(Note) RETI
			Mode 1	Address 38H.	
			Mode 2	The address indicated by the data table (memory) at the address specified by I register (high-order 8 bits) and data from peripheral LSI (low-order 8 bits, LSB = "0").	

110489

Note : Mode 0 applies when the instruction from peripheral LSI is CALL or RST instruction.



110489

Figure 3.2.4 Interrupt Enable Flip-Flop (IFF)

When a non-maskable interrupt has been accepted, IFF1 is reset to "0" (interrupt disable) until an EI or RETI instruction is executed to prevent a next interrupt accepting. For this purpose, the state (interrupt enable/disable) of IFF1 immediately before non-maskable interrupt acceptance must be stored. This state is copied into IFF2 upon acceptance of a non-maskable interrupt. The content of IFF2 is copied into the parity flag at the execution of the following instructions, so that the copied data can be tested or stored:

- The load instruction (LD A, I) to load from the I register to the accumulator.
- The load instruction (LD A, R) to load from the R register to the accumulator.

When the return instruction from the non-maskable interrupt (RETN) is executed, the content of IFF2 is copied back to IFF1. If an operation which changes the contents of IFF2 (due to the execution of EI or DI instruction, for example) has not been performed during interrupt handling, IFF1 automatically returns to the state which was effective immediately before the interrupt acceptance. Table 3.2.1 lists the states of IFF1 and IFF2 after the execution of interrupt-related instructions.

Table 3.2.1 State of IFF1 and IFF2

Operation sequence	IFF1	IFF2	Remarks
MPU reset	0	0	
EI	1	1	
NMI acceptance	0	1	
LD A, I	*	*	Parity flag←IFF2
RETN	1	1	IFF1←IFF2
LD A, R	*	*	Parity flag←IFF2
INT acceptance	0	0	
RETI	*	*	
EI	1	1	
NMI acceptance	0	1	
DI	0	0	
RETN	*	*	

Note : * = no change.

110489

(2) Interrupt processing

With a non-maskable interrupt, the internal NMI flip-flop is set to "1" at the falling edge of the interrupt signal, $\overline{\text{NMI}}$. The state of this flip-flop is sampled at the rising edge of the last clock of each instruction to accept an interrupt. A maskable interrupt is accepted if the interrupt signal $\overline{\text{INT}}$ is low ("0") at the rising edge of the last clock of each instruction and the interrupt enabled state (IFF = 1 and $\overline{\text{BUSREQ}}$ signal = inactive ("1")) is on. Described below is the processing to be performed after a non-maskable interrupt and a maskable interrupt are accepted:

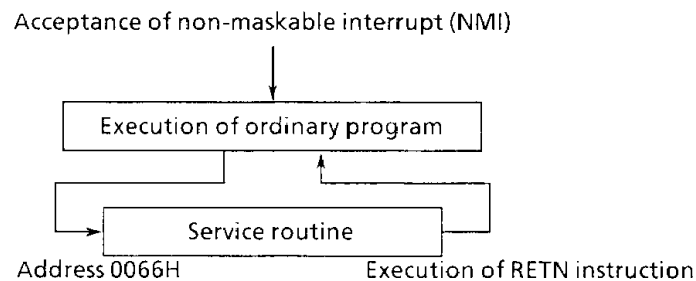
(a) Non-maskable interrupt (NMI)

When a non-maskable interrupt has been accepted, the MPU performs the following processing:

- ① The internal NMI flip-flop is reset "0".
- ② IFF1 is reset to "0", disabling the maskable interrupt.
The contents of the IFF1 immediately before the interrupt acceptance are copied into the IFF2.
- ③ The current contents of the program counter are saved into the stack.
- ④ The instructions starting from non-maskable interrupt vector address 66H are executed.

The non-maskable interrupt processing program terminates by executing the RETN instruction. This return instruction performs the followings:

- ① The contents of the current IFF2 are copied into IFF1.
- ② The contents of the program counter are restored from the stack.



110489

Figure 3.2.5 Non-Maskable Interrupt Processing

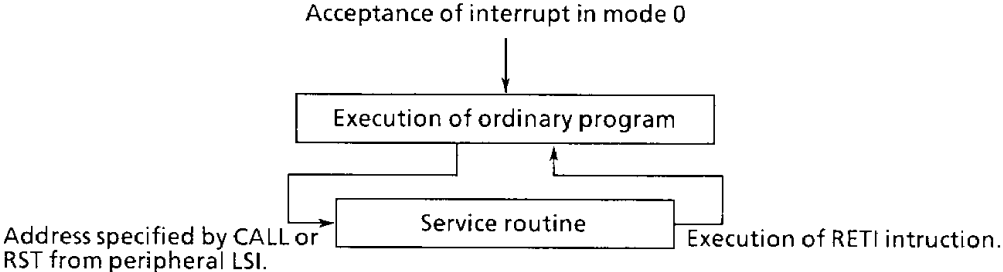
(b) Maskable interrupt (INT)

When a maskable interrupt has been accepted, the MPU performs the following processings:

- ① Both IFF1 and IFF2 are reset to "0", disabling the maskable interrupts.
- ② The current contents of the program counter are saved into the stack.
- ③ A maskable interrupt is serviced in one of the three modes 0, 1 and 2. A mode is selected by executing the instruction IM0, IM1 or IM2 before the interrupt is serviced. The instructions are executed starting from the vector address corresponding to the selected mode.

• Mode 0

In mode 0, the interrupting peripheral LSI puts a restart instruction (RST) or a call instruction (CALL) on the data bus and the MPU executes the interrupt service routine according to that instruction.

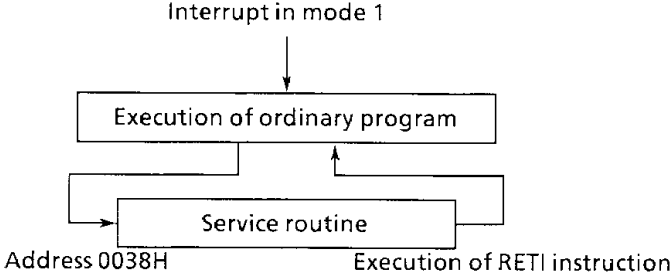


110489

Figure 3.2.6 Interrupt Processing in Mode 0

• Mode 1

When an interrupt is accepted in mode 1, restart is performed from address 0038H. Therefore, the service routine for this interrupt is programmed from the address 0038H.

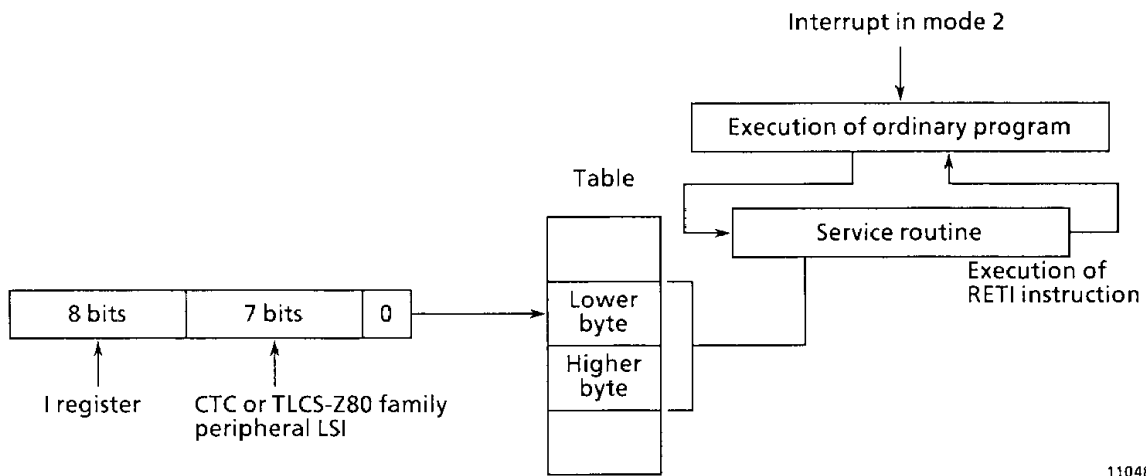


110489

Figure 3.2.7 Interrupt Processing in Mode 1

- Mode 2

The interrupt processing in mode 2 requires a 16-bit pointer consisting of the high-order 8 bits of the I register and the low-order 8 bits (with the LSB = "0") of the data captured from the interrupting CTC or TLCS-Z80 family peripheral LSI. Therefore, the necessary value must be loaded in the I register beforehand. This pointer is used to specify the memory address in the table. The contents of the specified addresses and the next address provide the start address of the service routine. Therefore, use of this mode requires to present table of the service routine's start address (16 bits) by program at convenient location. This location can be anywhere in memory. The LSB of the table pointer is set to "0" because a 2-byte data is needed to specify the service routine start address just in 16 bits and start that address from an even-number address. In the table, the start address begins with the low-order byte followed by the high-order byte as shown in Figure 3.2.8



110489

Figure 3.2.8 Interrupt Processing in Mode 2

Mode 2 is used in the daisy chain interrupt processing using the CTC and/or TLCS-Z80 family LSI. The CTC and TLCS-Z80 family peripheral LSIs all contain the interrupt priority controller of daisy chain structure. In this interrupt structure, the interrupt request signals are put in the series order and given priorities for processing when two or more maskable interrupt requests occur at a time. Only the interrupt vector from the peripheral LSI having the highest priority is put on the data bus. By receiving the vector interrupt in mode 2, the processing for that peripheral LSI can be performed. When an interrupt comes from a peripheral LSI having a priority higher than that of the current peripheral LSI during the execution of its interrupt processing routine, the new interrupt can be enabled by the EI instruction to form an interrupt nesting.

The maskable interrupt processing program terminates by executing an RETI instruction. This return instruction performs the following processing:

- Restores the content of the program counter from the stack.
- Notifies the requesting peripheral LSI of the termination of interrupt processing.

3.2.3 MPU Status Transition Diagram and Basic Timing

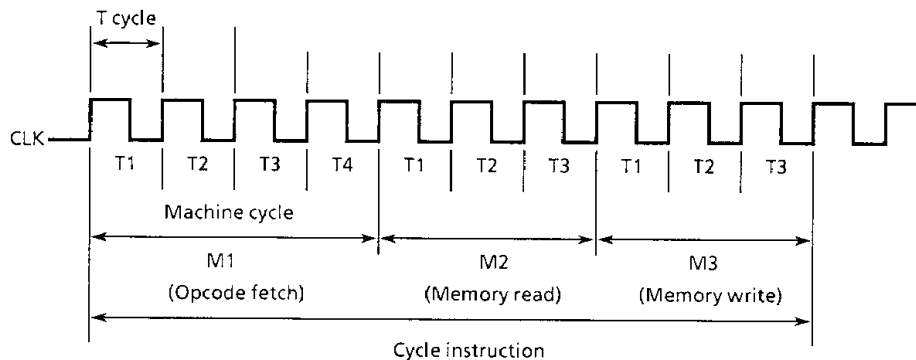
The following describes the MPU status transition and the basic timing of each MPU operation.

[1] Instruction Cycle

Each TMPZ84C011A instruction is executed by combining the basic operations of memory read/write, input/output, bus request/acknowledge, and interrupt. These basic operations are performed in synchronization with the system clock (the CLK signal).

One clock period is called a state (T). The smallest unit of each basic operation is called a machine cycle (M). Each instruction consists of 1 to 6 machine cycles and each machine cycle, 3 to 6 clock states basically. However, the number of clock states in a machine cycle can be increased by the $\overline{\text{WAIT}}$ signal described later on. Figure 3.2.9 shows an example of the basic timing of a 3-machine-cycle instruction.

The first machine cycle (M1) of each instruction is the cycle in which the Opcode of the instruction to be executed next is read (this is called the Opcode fetch cycle). The Opcode fetch cycle basically consists of 4 to 6 clock states. In the machine cycle that follows the Opcode fetch cycle, data is transferred between the MPU and the memory or peripheral LSIs. This operation basically consists of 3 to 5 clock states.



110489

Figure 3.2.9 Example of MPU Basic Timing (3-Machine-Cycle Instruction)

[2] Status Transition Diagram

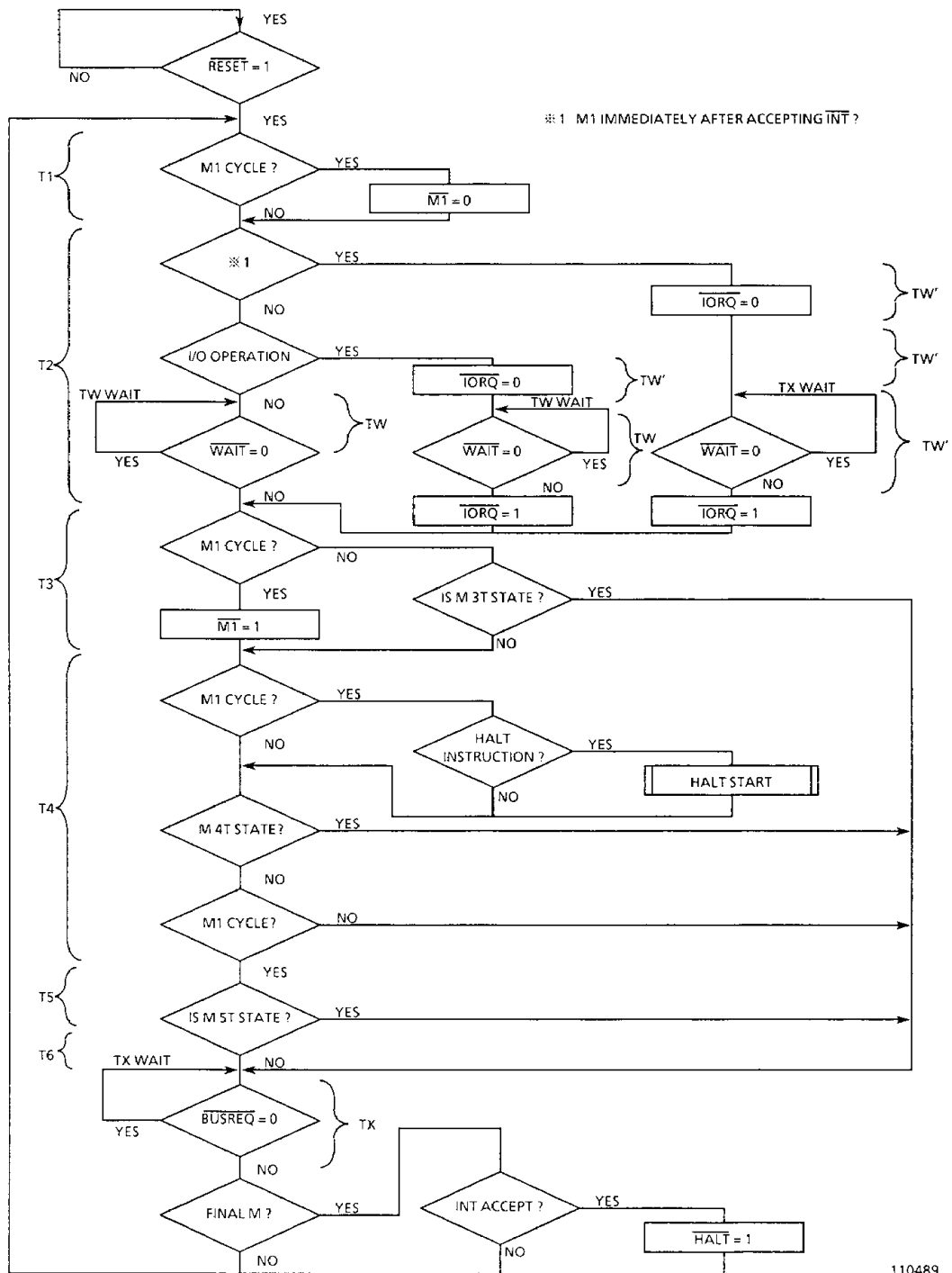


Figure 3.2.10 Status Transition Diagram

[3] Basic Timings

(1) Opcode fetch cycle (M1)

In the Opcode fetch cycle, MPU reads an Opcode from among the machine-language codes in memory. This is also called the M1 cycle for it is the first machine cycle to execute each instruction.

Figure 3.2.11 shows the basic timing of a basic Opcode fetch cycle. In clock state T1, the content of the program counter is put on the address bus. The $\overline{M1}$ signal goes "0", telling the outside the MPU that this is the Opcode fetch cycle. At the same time, \overline{MREQ} and \overline{RD} signals go "0". When the \overline{MREQ} signal goes "0", the address signal has already been stabilized. Therefore, this signal can be used for the memory chip enable signal. The \overline{RD} signal indicates that the MPU is ready to accept the data from memory. By these signals, the MPU accesses memory to put the Opcode in the instruction register. The MPU samples the \overline{WAIT} signal at the falling edge of clock state T2. If the \overline{WAIT} signal is "0" at the falling edges of clock state T2 and the following wait state (T_W), the next state becomes clock state T_W . Figure 3.2.12 shows the delay state of the Opcode fetch cycle caused by the \overline{WAIT} signal.

The data (Opcode) on the data bus is fetched at the rising edge of clock state T3 then, at the rising edges of the same state, the \overline{MREQ} , \overline{RD} , and $\overline{M1}$ signals go "1". In clock state T3, a memory refresh address is put on the low-order 7 bits of the address bus and the \overline{RFSH} signal goes "0" and the \overline{MREQ} signal goes "0" again. This signal indicates that the memory refresh cycle is on. At this time, the contents of the I register are on the high-order 8 bits of the address bus and the 7 bits of the R register contents are on the low-order 7 bits of the address bus. By using the \overline{RFSH} and \overline{MREQ} signals, memory refresh is performed in clock states T3 and T4. However, the \overline{RD} signal remains "1" because the contents of the memory refresh address are not put on the data bus.

In clock state T4, the \overline{MREQ} signal returns to "1". The refresh address is kept output until the rising edge of the clock state T1 of the next machine cycle, during which the \overline{RFSH} signal is also "0". The cycle delay state caused by setting the \overline{WAIT} signal to "0" is the same with the memory read/write, input/output, and maskable interrupt acknowledge cycles. The following description the diagram of the cycle delay state caused by the \overline{WAIT} signal's going "0" is omitted.

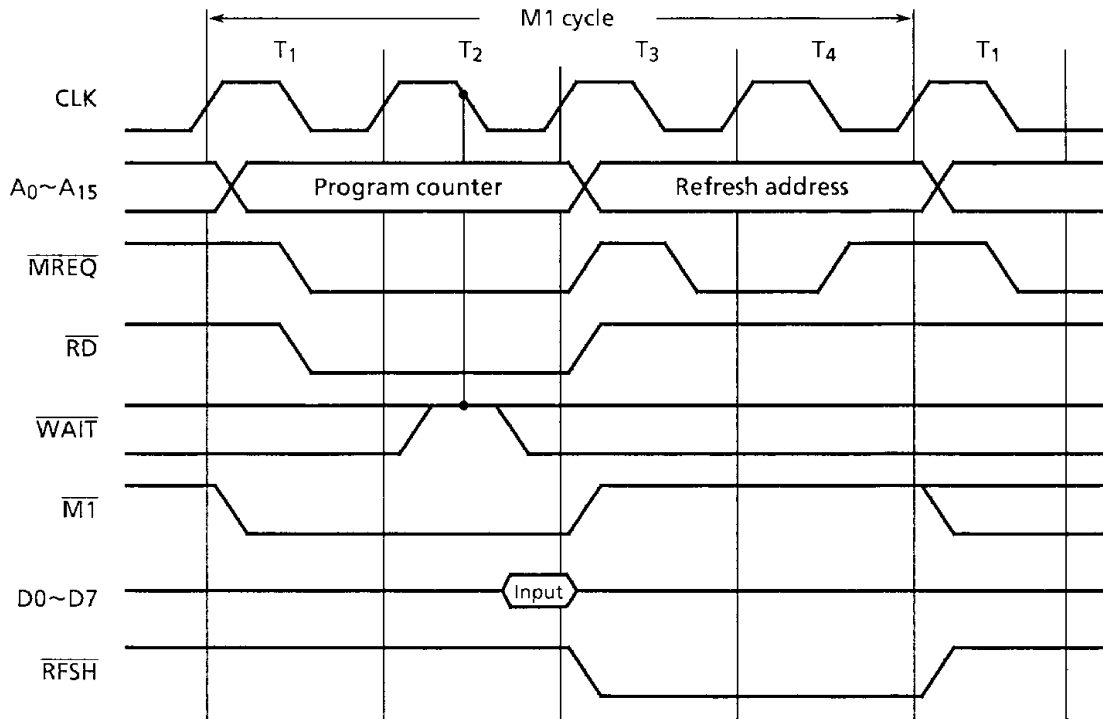


Figure 3.2.11 Opcode Fetch Timing

110489

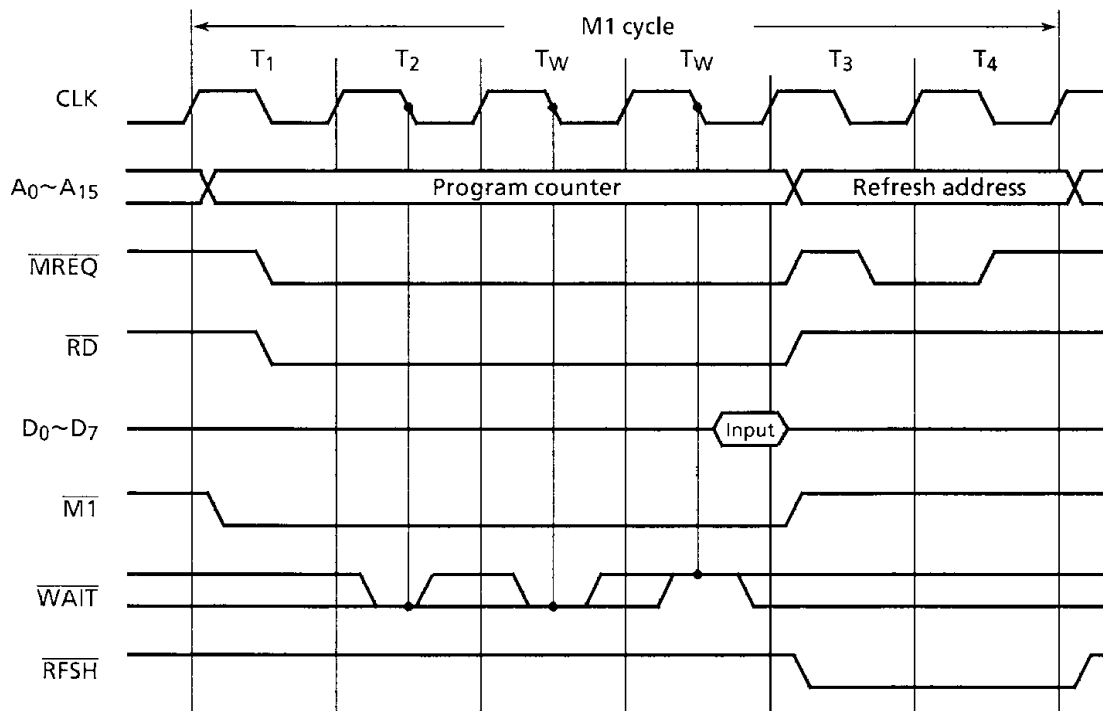


Figure 3.2.12 Opcode Fetch Timing Including Wait State

110489

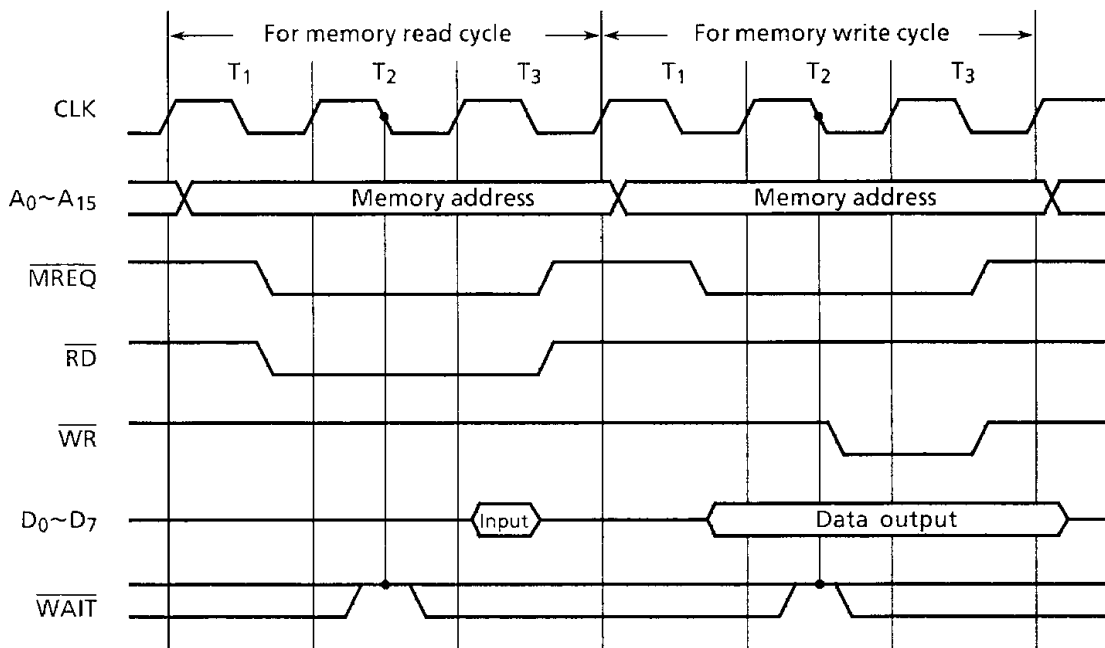
(2) Memory read/write operations

Figure 3.2.13 shows the basic timing of memory read/write operations (except for the Opcode fetch cycle) in the same diagram for convenience.

In each case, the memory address signal to read/write data on the address bus is output in clock state T1. The operation in which the $\overline{\text{WAIT}}$ signal is sampled in clock state T2 and the following T_W state is the same as the Opcode fetch cycle.

In memory read, memory data is put on the data bus by the address, $\overline{\text{MREQ}}$, and $\overline{\text{RD}}$ signals. The MPU reads this data.

In memory write, the memory address signals is put on the address bus then the $\overline{\text{MREQ}}$ signal is set to "0" to put the write data onto the data bus. When the data bus has been stabilized, the $\overline{\text{WR}}$ signal is output in clock state T2. The $\overline{\text{WR}}$ signal can be used for the memory write signal.



110489

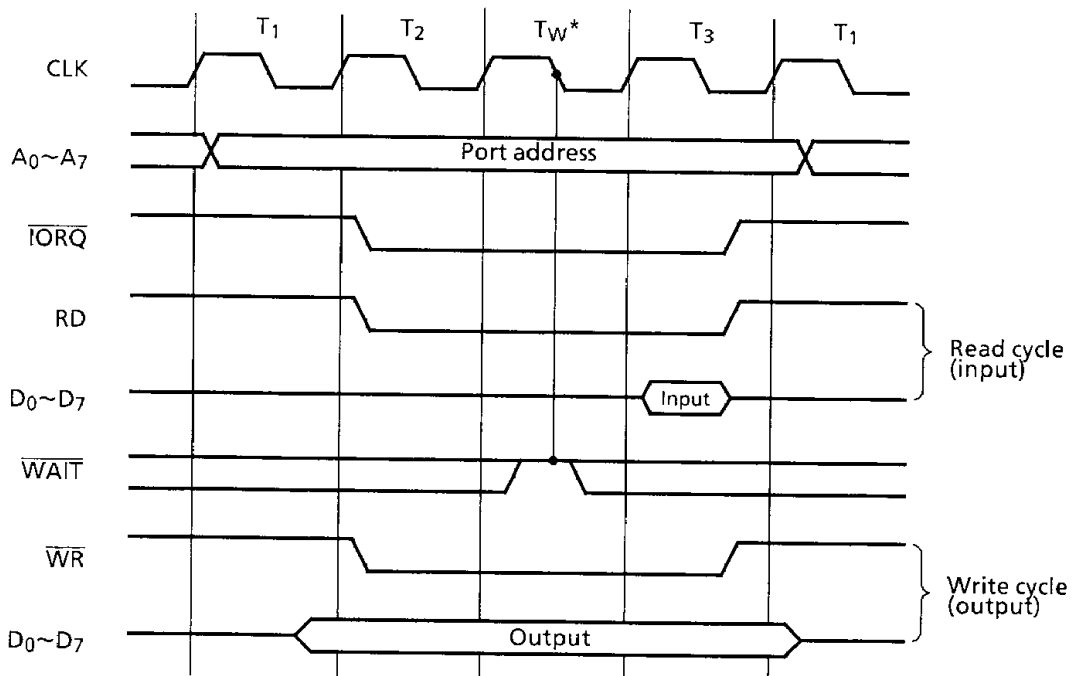
Figure 3.2.13 Memory Read / Write Cycle Timing

(3) Input/output operations

Figure 3.2.14 shows the basic timing of input/output operations. The feature of the I/O operation timing is that, regardless of the state of the $\overline{\text{WAIT}}$ signal in clock state T2, the I/O cycle automatically gets in the wait state (T_{W^*}) after clock T2. The $\overline{\text{WAIT}}$ signal is sampled at the falling edge of T_{W^*} . If the $\overline{\text{WAIT}}$ signal is "0" at the falling edges of T_{W^*} and the following clock state, the I/O cycle moves into clock state T_W . Clock state T_{W^*} is inserted because the $\overline{\text{IORQ}}$ signal goes "0" in clock state T2, so that it is too late to sample the $\overline{\text{WAIT}}$ signal after decoding the I/O port address. In each of input and output operations, the I/O port address is put on the low-order 8 bits of the address bus in clock state T1. On the high-order 8 bits, the contents of the accumulator or B register are output. In clock state T2, the $\overline{\text{IORQ}}$ signal goes "0" instead of the $\overline{\text{MREQ}}$ signal. The $\overline{\text{IORQ}}$ signal can be used as the chip enable signal for a peripheral LSI.

In an input operation, the contents of the input port are read onto the data bus by the address, $\overline{\text{IORQ}}$, and $\overline{\text{RD}}$ signals. The MPU reads this data.

In an output operation, the output port address and the output data are respectively put on the address bus and data bus in clock state T1; the $\overline{\text{IORQ}}$ and $\overline{\text{WR}}$ signals goes "0" in clock state T2. The $\overline{\text{WR}}$ signal can be used as the output port write signal.



110489

Figure 3.2.14 I/O Operation Timing

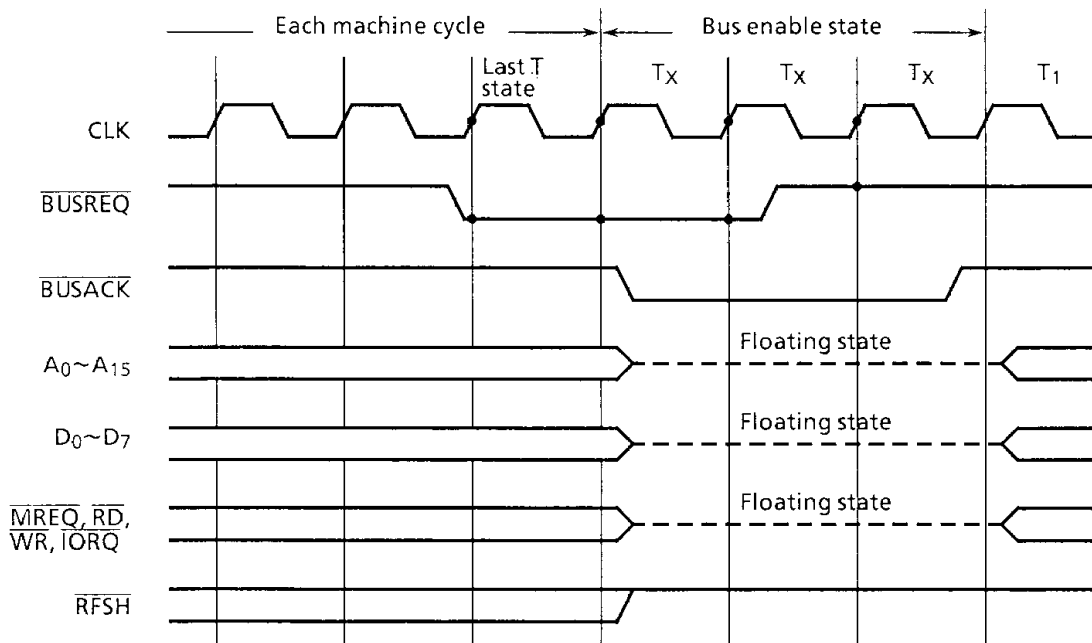
(4) Bus request and bus acknowledge operations

Figure 3.2.15 shows the basic timings of bus request and bus acknowledge operations.

The 3-state address bus (A0 through A15), data bus (D0 through D7), $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ signals controlled by the MPU can be put in the high-impedance state (floating) to electrically disconnect them from the MPU. This operation, by sampling the $\overline{\text{BUSREQ}}$ signal at the rising edge of the last clock of each machine cycle, starts with the rising edge of the next clock if this signal is found "0". Subsequently, these buses are controlled by peripheral LSIs. For example, data can be directly transferred between memory and these peripheral LSIs. This state is cleared if the $\overline{\text{BUSREQ}}$ signal is found "1" by sampling it at the rising edge of each subsequent clock state (T_X), executing the next machine cycle. During the floating state, the $\overline{\text{BUSACK}}$ signal is "0" to tell the peripheral LSIs of it.

In this state, however, no memory refresh is performed and, therefore, the $\overline{\text{RFSH}}$ signal is set to "1". Hence, to maintain this state for a long time with a system using dynamic memory, memory refresh must be performed by the external controller.

Note that, in the floating state, neither maskable (INT) nor non-maskable (NMI) interrupts can be accepted.



110489

Figure 3.2.15 Bus Request and Bus Acknowledge Timing

(5) Maskable interrupt acknowledge operation

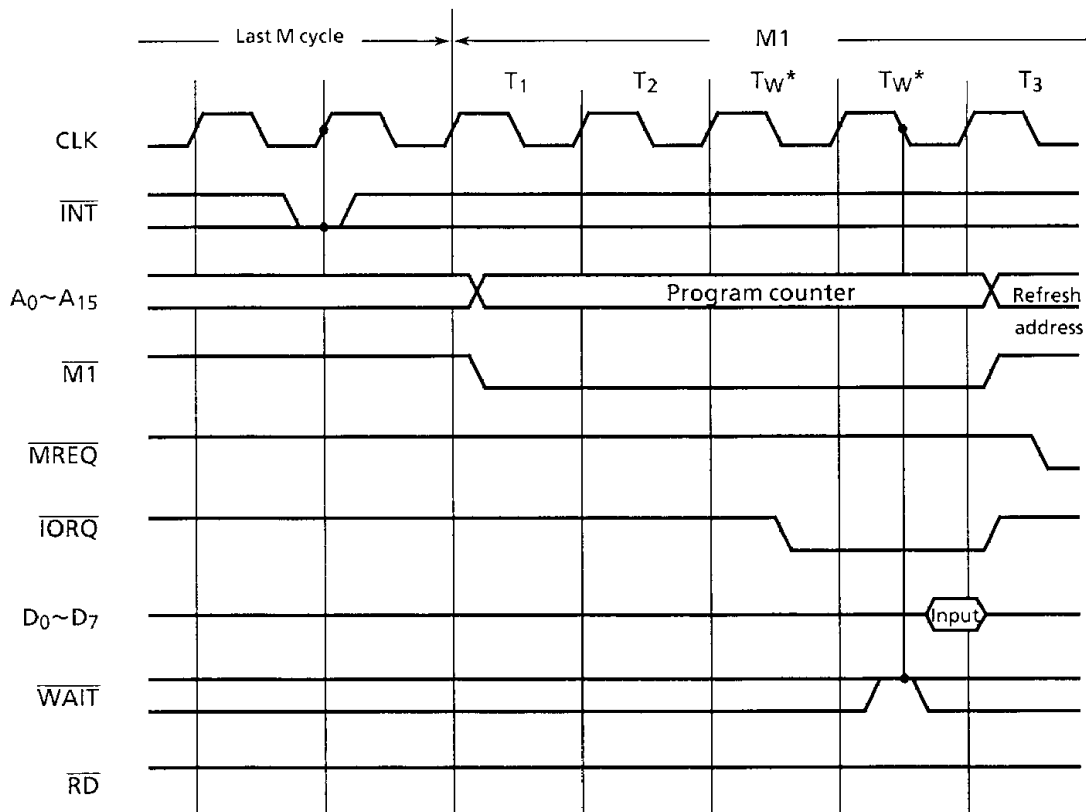
Figure 3.2.16 shows the basic timing of the maskable interrupt acknowledge.

The MPU samples the maskable interrupt request signal ($\overline{\text{INT}}$) at the rising edge of the last clock of each instruction execution. If the $\overline{\text{INT}}$ signal is found "0", a maskable interrupt is accepted except in the following cases:

- The interrupt enable flip-flop is reset to "0".
- The $\overline{\text{BUSREQ}}$ signal is "0".

When a maskable interrupt has been accepted, a special Opcode fetch cycle is generated. In this cycle, 2 clock states of wait state (T_W^*) is automatically inserted after the clock state T2. The $\overline{\text{WAIT}}$ signal is sampled at the falling edge of the second clock state T_W^* and the following clock state T_W and, if the $\overline{\text{WAIT}}$ signal is found "0", the instruction cycle gets in the next clock state T_W . In this Opcode fetch cycle, the $\overline{\text{IORQ}}$ signal goes "0" in the first T_W^* state instead of the $\overline{\text{MREQ}}$ signal while in a normal Opcode fetch cycle the $\overline{\text{MREQ}}$ signal goes "0" in clock state T1. This tells the maskable interrupt requesting LSI that it can put the 8-bit interrupt vector on the data bus, then reads this data to perform interrupt processing. Therefore, the contents of the program counter put on the address bus are not used. Unlike an ordinary I/O operation, the $\overline{\text{RD}}$ signal does not go "0".

In the clock state T3, the memory refresh address signal is put on the address bus for memory refresh like normal Opcode fetch cycle and the $\overline{\text{RFSH}}$ signal goes "0". In the subsequent machine cycles (M2 and M3), the current contents of the program counter are saved into the stack. In the machine cycles M4 and M5, the contents of the I register (the high-order 8 bits) and the contents of the address indicated by the address of the vector (the low-order 8 bits) from the CTC or the peripheral LSIs connected externally are put in the program counter.



120489

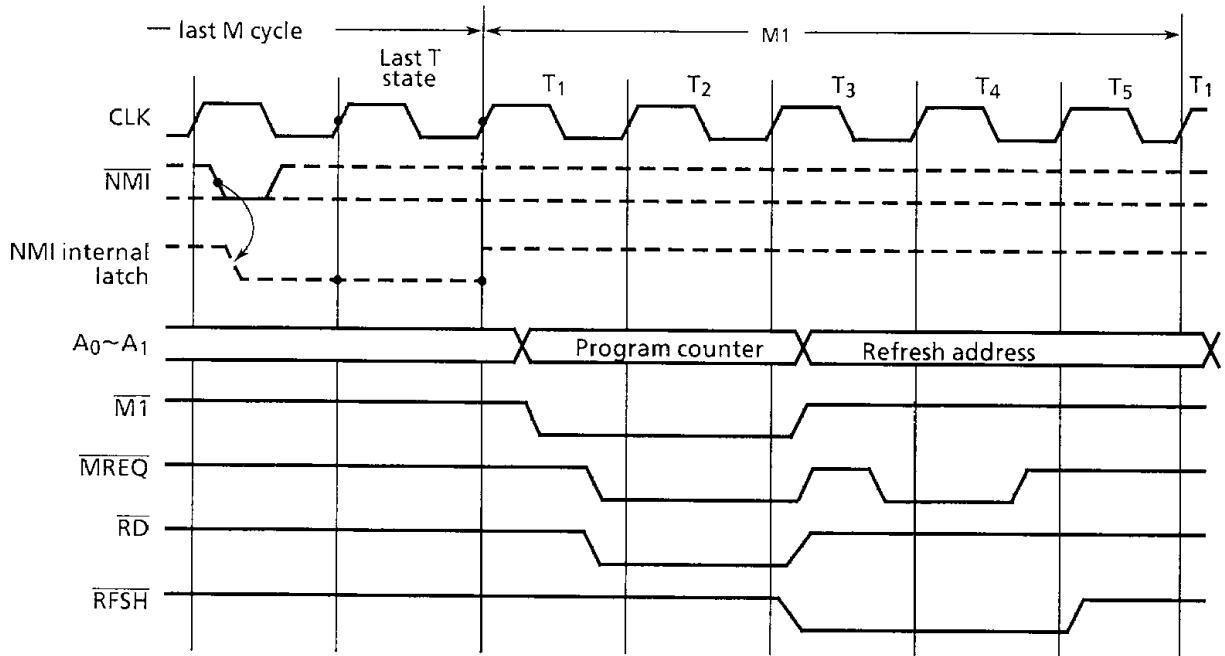
Figure 3.2.16 Maskable Interrupt Acknowledge Timing

(6) Non-maskable interrupt acknowledge operation

Figure 3.2.17 shows the basic timing of non-maskable interrupt acknowledge.

When the non-maskable interrupt request signal ($\overline{\text{NMI}}$) goes low, the internal non-maskable flip-flop is set to "1". The $\overline{\text{NMI}}$ signal is detected in any timing of each instruction. However, the internal NMI flip-flop is sampled at the rising edge of the last clock of each instruction. Therefore, the $\overline{\text{NMI}}$ signal should go low until the last clock state of an instruction

The Opcode fetch cycle for non-maskable interrupt acknowledge is generally the same as the ordinary Opcode fetch cycle. However, the Opcode on the data bus at the time is ignored. The current contents of the program counter are saved into the stack in the subsequent machine cycles (M2 and M3). In the following machine cycle, the operation jumps to address 0066H, the non-maskable interrupt vector address. The machine cycles after these depend on the contents of the fetched Opcode.



120489

Figure 3.2.17 Non-Maskable Interrupt Acknowledge Timing

(7) Halt operation

Having fetched a HALT instruction in the Opcode fetch cycle, the MPU sets the $\overline{\text{HALT}}$ signal to "0" in synchronization with the falling edge of clock state T4 to tell peripheral LSIs of the fetch and stops operating. If the system clock is kept supplied in the halt state, the MPU continues executing NOP instructions. This is done to output refresh signals when the memory is used. The NOP instruction execution cycle is the same as the ordinary Opcode fetch cycle in which the data on the data bus are ignored.

The halt state is cleared when an interrupt is accepted or the $\overline{\text{RESET}}$ signal is set to "0" to reset the MPU. Figure 3.2.18 shows the halt state clear operation by interrupt acknowledge. An interrupt is sampled at the rising edge of the last clock (clock state T4) of the NOP instruction. A maskable interrupt can be accepted when the $\overline{\text{INT}}$ signal is "0". A non-maskable interrupt is accepted when the internal NMI flip-flop which is set at the falling edge of the $\overline{\text{NMI}}$ signal is set at "1". However, it is required that the interrupt enable flip-flop is set to "1" for a maskable interrupt to be accepted. The interrupt processing for the type of the accepted interrupt gets started from the following cycle.

However, when the supply of the system clock from the CGC has been stopped by the power down operation, it is required to restart the supply of the system clock and input the $\overline{\text{INT}}$ signal until the execution of one instruction is completed or the $\overline{\text{RESET}}$ signal at least 3 clocks are output. Figure 3.2.9 shows the timing of clearing the halt state caused by power down.

For the reset operation, see (8) "Reset operation" below. Note that the $\overline{\text{INT}}$ and $\overline{\text{NMI}}$ signals are shown on the same diagram in Figures 3.2.18 and 3.2.19 for convenience.

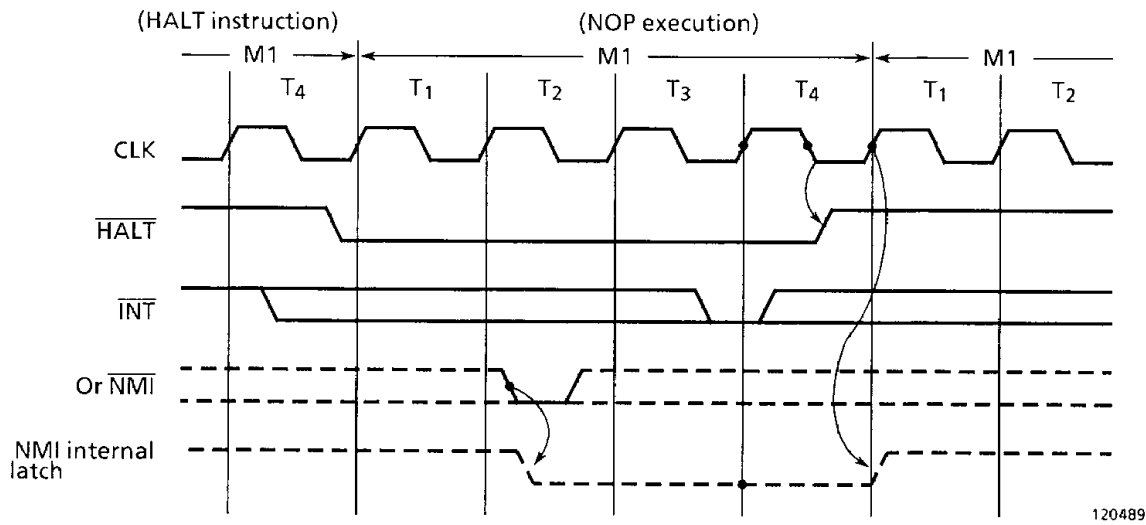


Figure 3.2.18 Timing of Clearing Halt State Caused by Interrupt Acknowledge

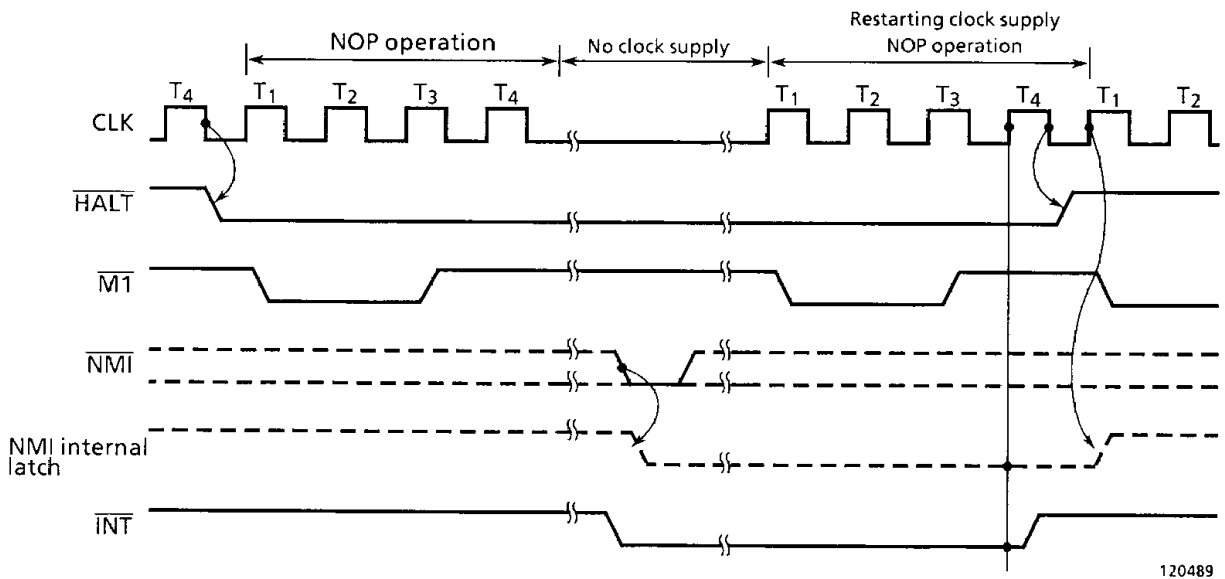
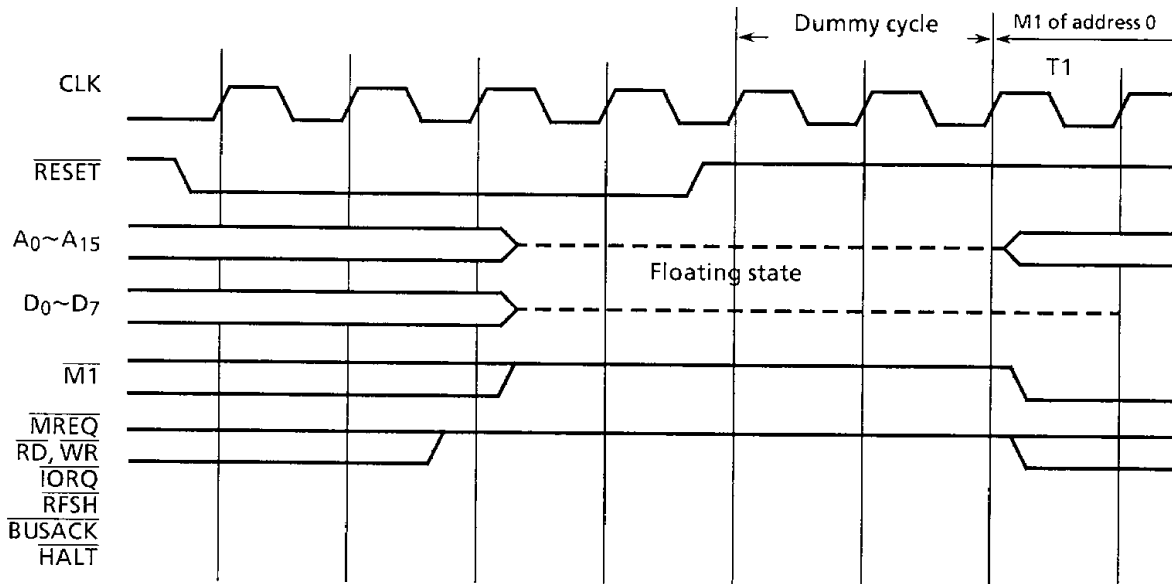


Figure 3.2.19 Timing of Clearing Halt State Caused by Power Down

(8) Reset operation

Figure 3.2.20 shows the basic timing of reset operation.

To reset the MPU, the $\overline{\text{RESET}}$ signal must be kept at "0" for at least 3 clocks. When the $\overline{\text{RESET}}$ signal goes "1", instruction execution starts from address 0000H after a dummy cycle of at least 2 clock states.



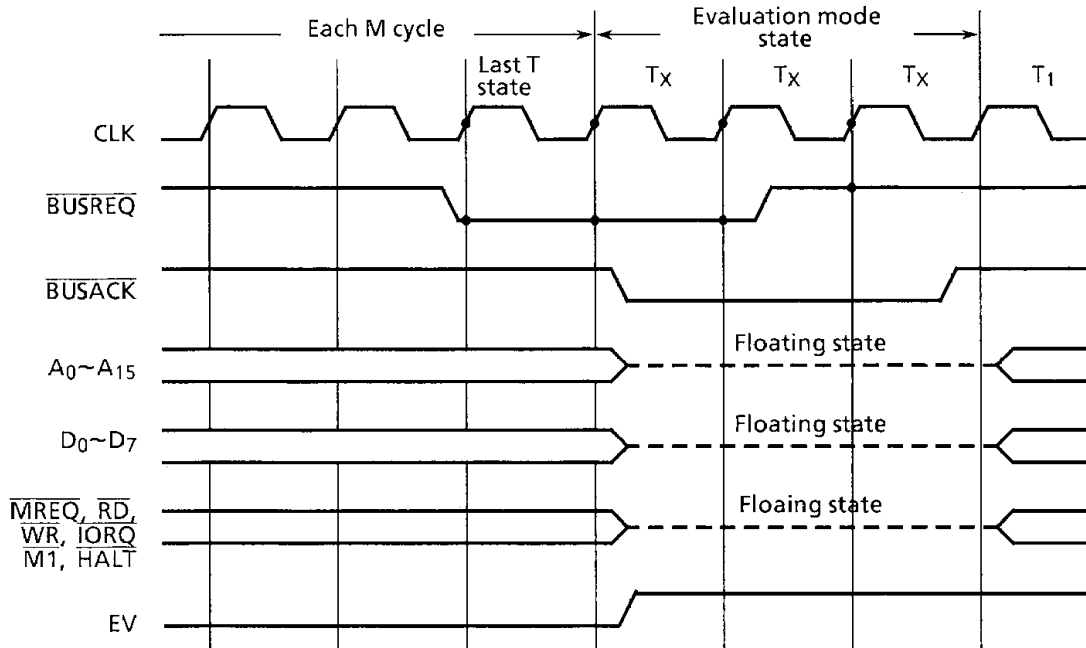
120489

Figure 3.2.20 Reset Timing

To clear the power down state by the $\overline{\text{RESET}}$ signal, the $\overline{\text{RESET}}$ signal must be input until 3 clocks or more are supplied by restarting the supply of the system clock from the CGC.

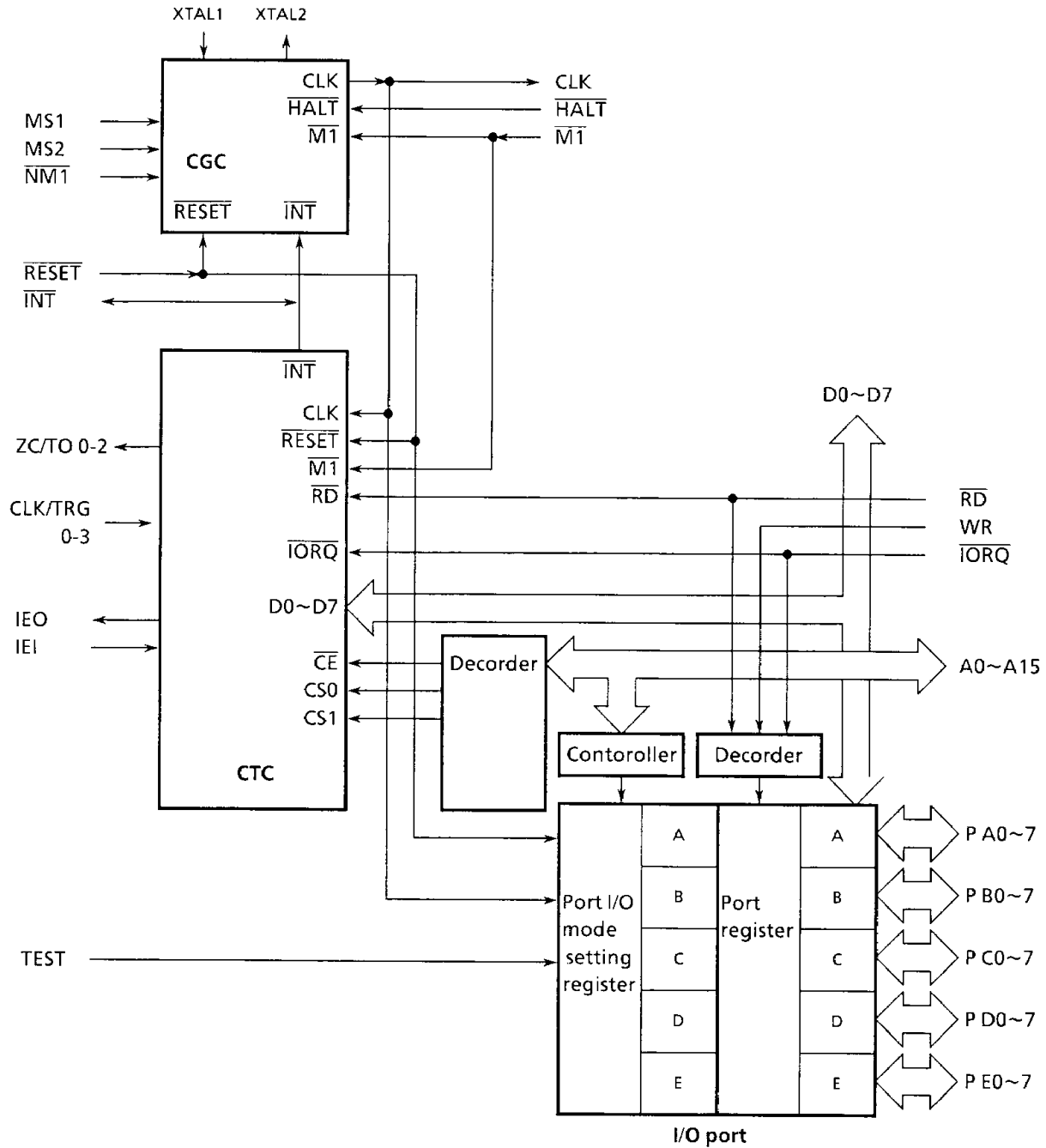
(9) Evaluation operation

Each of the MPU signals (A0 through A15, D0 through D7, $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{HALT}}$ and $\overline{\text{M1}}$) can be put in the high-impedance state by EV and $\overline{\text{BUSREQ}}$ signals to electrically disconnect them from the system.



120489

Figure 3.2.21 Evaluation Timing



120489

Figure 3.2.22 Block Diagram of the TMPZ84C011A Functioning As Evaluator

3.2.4 TMPZ84C011A Instruction Set

This subsection lists the TMPZ84C011A instruction codes and their functions. The table below lists the symbols and abbreviations used to describe the instruction set. The symbols which require special attention are described in the locations in which they appear.

- Symbols (1/2)

Classification	Symbol	Meaning
Register	r, g	Register B, C, D, E, H, L, A,
	t	Register pair BC, DE, HL Stack pointer SP
	q	Register pair BC, DE, HL, AF
	p	Register pair BC, DE Index register IX Stack pointer SP
	s	Register pair BC, DE Index register IY Stack pointer SP
	t _H	Higher register of register pair (B, D, H) Higher 8 bits of stack pointer (SP)
	q _H	Higher register of register pair (B, D, H, A)
	IX _H	Higher 8 bits of index register IX
	IY _H	Higher 8 bits of index register IY
	PC _H	Higher 8 bits of program counter (PC)
	t _L	Lower register of register pair (C, E, L) Lower 8 bits of stack pointer (SP)
	q _L	Lower register of register pair (C, E, L, F)
	IX _L	Lower 8 bits of index register IX
	IY _L	Lower 8 bits of index register IY
	PC _L	Lower 8 bits of program counter (PC)
	rb	Bit b (0-7) of register (B, C, D, E, H, L, A)

120489

- Symbols (2/2)

Classification	Symbol	Meaning
Memory	mn (HL) _b (IX + d) _b (IY + d) _b	Memory address represented in 16 bits. m indicates higher 8 bits and n, lower 8 bits. Bit b (0-7) of the contents of the memory address indicated by register pair HL. Bit b (0-7) of the contents of the memory address indicated by the value obtained by adding 8-bit data d to the content of index register IX. Bit b (0-7) of the contents of the memory address indicated by the value obtained by adding 8-bit data d to the content of index register IY.
Flag change symbol	0 1 - * X P V	Reset to "0" by operation. Set to "1" by operation. No change Affected by operation Undefined Handled as parity flag. P = 0: odd parity P = 1: even parity Handled as overflow flag. V = 0: No overflow V = 1: Overflow
Operator	← ↔ + - ^ v ⊕	Transfer Exchange Add Subtract Logical and between bits. Logical or between bits. Exclusive or between bits
Others	IFF CY Z	Interrupt enable flip-flop Carry flag Zero flag

120489

TMPZ84C011A Instruction Set (1/9)

ITEM/ CLASSI- FICATION	Assembler mnemonic	Object code		Function	Flag							No. OF CY- CLES	No. OF STA- TES	
		Binary	Hex		S	Z	H	P/V	N	C				
		76 543 210												
8 - BIT DATA LOAD	LD r,g	01 rrr ggg	40+r×8+g	r←g	-	-	X	-	X	-	-	-	1	4
	LD r,n	00 rrr 110 nn nnn nnn	06+r×8 n	r←n	-	-	X	-	X	-	-	-	2	7
	LD r,(HL)	01 rrr 110	46+r×8	r←(HL)	-	-	X	-	X	-	-	-	2	7
	LD r,(IX+d)	11 011 101 01 rrr 110 dd ddd ddd	DD 46+r×8 d	r←(IX+d)	-	-	X	-	X	-	-	-	5	19
	LD r,(IY+d)	11 111 101 01 rrr 110 dd ddd ddd	FD 46+r×8 d	r←(IY+d)	-	-	X	-	X	-	-	-	5	19
	LD (HL),r	01 110 rrr	70+r	(HL)←r	-	-	X	-	X	-	-	-	2	7
	LD (IX+d),r	11 011 101 01 110 rrr dd ddd ddd	DD 70+r d	(IX+d)←r	-	-	X	-	X	-	-	-	5	19
	LD (IY+d),r	11 111 101 01 110 rrr dd ddd ddd	FD 70+r d	(IY+d)←r	-	-	X	-	X	-	-	-	5	19
	LD (HL),n	00 110 110 nn nnn nnn	36 n	(HL)←n	-	-	X	-	X	-	-	-	3	10
	LD (IX+d),n	11 011 101 00 110 110 dd ddd ddd nn nnn nnn	DD 36 d n	(IX+d)←n	-	-	X	-	X	-	-	-	5	19
	LD (IY+d),n	11 111 101 00 110 110 dd ddd ddd nn nnn nnn	FD 36 d n	(IY+d)←n	-	-	X	-	X	-	-	-	5	19
	LD A,(BC)	00 001 010	9A	A←(BC)	-	-	X	-	X	-	-	-	2	7
	LD A,(DE)	00 011 010	1A	A←(DE)	-	-	X	-	X	-	-	-	2	7
	LD A,(mn)	00 111 010 nn nnn nnn mm mmm mmm	3A n m	A←(mn)	-	-	X	-	X	-	-	-	4	13
	LD (BC),A	00 000 010	02	(BC)←A	-	-	X	-	X	-	-	-	2	7
	LD (DE),A	00 010 010	12	(DE)←A	-	-	X	-	X	-	-	-	2	7
	LD (mn),A	00 110 010 nn nnn nnn mm mmm mmm	32 n m	(mn)←A	-	-	X	-	X	-	-	-	4	13
	LD A,I	11 101 101 01 010 111	ED 57	A←I	*	*	X	0	X	IFF	0	-	2	9
	LD A,R	11 101 101 01 011 111	ED 5F	A←R	*	*	X	0	X	IFF	0	-	2	9
	LD I,A	11 101 101 01 000 111	ED 47	I←A	-	-	X	-	X	-	-	-	2	9
LD R,A	11 101 101 01 001 111	ED 4F	R←A	-	-	X	-	X	-	-	-	2	9	
16-BIT DATA LOAD	LD t,mn	00 tt0 001 nn nnn nnn mm mmm mmm	01+t×10 n m	t←mn	-	-	X	-	X	-	-	-	3	10
	LD IX,mn	11 011 101 00 100 001 nn nnn nnn mm mmm mmm	DD 21 n m	IX←mn	-	-	X	-	X	-	-	-	4	14

r	rrr
g	ggg
B	000
C	001
D	010
E	011
H	100
L	101
A	111

t	tt
BC	00
DE	01
HL	10
SP	11

Note : r,g means any of the registers A, B, C, D, E, H, L.
IFF in "Flag" column indicates that the content of the interrupt enable flip-flop is copied into the P/V flag.

TMPZ84C011A Instruction Set (2/9)

ITEM/ CLASSI- FICA- TION	Assembler mnemonic	Object code		Function	Flag						No. OF CY- CLES	No. OF STA- TES		
		Binary			Hex	S	Z	H	P/V	N			C	
		76	543											210
16-BIT DATA LOAD	LD IY, mn	11 111 101 00 100 001 nn nnn nnn mm mmm mmm	FD 21 n m	IY+mn	-	-	X	-	X	-	-	-	4	14
	LD HL, (mn)	00 101 010 nn nnn nnn mm mmm mmm	2A n m	H+(mn+1) L+(mn)	-	-	X	-	X	-	-	-	5	16
	LD t, (mn)	11 101 101 01 tt1 011 nn nnn nnn mm mmm mmm	ED 4B+t×10 n m	tH+(mn+1) tL+(mn)	-	-	X	-	X	-	-	-	6	20
	LD IX, (mn)	11 011 101 00 101 010 nn nnn nnn mm mmm mmm	DD 2A n m	IXH+(mn+1) IXL+(mn)	-	-	X	-	X	-	-	-	6	20
	LD IY, (mn)	11 111 101 00 101 010 nn nnn nnn mm mmm mmm	FD 2A n m	IYH+(mn+1) IYL+(mn)	-	-	X	-	X	-	-	-	6	20
	LD (mn), HL	00 100 010 nn nnn nnn mm mmm mmm	22 n m	(mn+1)+H (mn)-L	-	-	X	-	X	-	-	-	5	16
	LD (mn), t	11 101 101 01 tt0 011 nn nnn nnn mm mmm mmm	ED 43+t×10 n m	(mn+1)+tH (mn)+tL	-	-	X	-	X	-	-	-	6	20
	LD (mn), IX	11 011 101 00 100 010 nn nnn nnn mm mmm mmm	DD 22 n m	(mn+1)+IXH (mn)-IXL	-	-	X	-	X	-	-	-	6	20
	LD (mn), IY	11 111 101 00 100 010 nn nnn nnn mm mmm mmm	FD 22 n m	(mn+1)+IYH (mn)-IYL	-	-	X	-	X	-	-	-	6	20
	LD SP, HL	11 111 001	F9	SP+HL	-	-	X	-	X	-	-	-	1	6
	LD SP, IX	11 011 101 11 111 001	DD F9	SP+IX	-	-	X	-	X	-	-	-	2	10
	LD SP, IY	11 111 101 11 111 001	FD F9	SP+IY	-	-	X	-	X	-	-	-	2	10
	PUSH q	11 qq0 101	C5+q×10	{SP-2}+qL, {SP-1}+qH, SP-SP-2	-	-	X	-	X	-	-	-	3	11
	PUSH IX	11 011 101 11 100 101	DD E5	{SP-2}+IXL, {SP-1}+IXH SP+SP-2	-	-	X	-	X	-	-	-	4	15
	PUSH IY	11 111 101 11 100 101	FD E5	{SP-2}+IYL, {SP-1}+IYH SP+SP-2	-	-	X	-	X	-	-	-	4	15
	POP q	11 qq0 001	C1+q×10	qH+(SP+1), qL+(SP), SP+SP+2	-	-	X	-	X	-	-	-	3	10
	POP IX	11 011 101 11 100 001	DD E1	IXH+(SP+1), IXL+(SP) SP+SP+2	-	-	X	-	X	-	-	-	4	14
	POP IY	11 111 101 11 100 001	FD E1	IYH+(SP+1), IYL+(SP) SP+SP+2	-	-	X	-	X	-	-	-	4	14
	*1 EX DE, HL	11 101 011	EB	DE↔HL	-	-	X	-	X	-	-	-	1	4
	EX AF, AF'	00 001 000	08	AF↔AF'	-	-	X	-	X	-	-	-	1	4
EXX	11 011 001	D9	BC↔BC', DE↔DE', HL↔HL'	-	-	X	-	X	-	-	-	1	4	

t	tt
BC	00
DE	01
HL	10
SP	11

q	qq
BC	00
DE	01
HL	10
AF	11

Note : t is any of the register pairs BC, DE, HL, SP.
q is any of the register pairs AF, BC, DE, HL.
(PAIR)_H, (PAIR)_L refer to high order and low order eight bits of the register pair respectively. (Ex) BC_L = C, AF_H = A.
*1 : EXCHANGE

TMPZ84C011A Instruction Set (3/9)

ITEM/ CLASSI- FICATION	Assembler mnemonic	Object code		Function	Flag							No. OF CY- CLES	No. OF STA- TES				
		Binary			Hex	S	Z	H	P/V	N	C						
		76	543											210			
EXCHANGE	EX (SP),HL	11	100	011	E3	H \leftrightarrow (SP+1),L \leftrightarrow (SP)	-	-	X	-	X	-	-	-	5	19	
	EX (SP),IX	11	011	101	DD	IX \leftrightarrow (SP+1)	-	-	X	-	X	-	-	-	6	23	
		11	100	011	E3	IX \leftrightarrow (SP)	-	-	X	-	X	-	-	-	6	23	
	EX (SP),IY	11	111	101	FD	IY \leftrightarrow (SP+1)	-	-	X	-	X	-	-	-	6	23	
		11	100	011	E3	IY \leftrightarrow (SP)	-	-	X	-	X	-	-	-	6	23	
BLOCK TRANSFER BLOCK SEARCH	LDI	11	101	101	ED	(DE)+(HL),DE+DE+1	-	-	X	0	X	*M	0	-	4	16	
		10	100	000	A0	HL+HL+1,BC+BC-1	-	-	X	0	X	0	0	-	5	21	
	LDIR	11	101	101	ED	(DE)+(HL),DE+DE+1	-	-	X	0	X	0	0	-	4	16	
		10	110	000	B0	HL+HL+1,BC+BC-1 Repeat until BC=0	-	-	X	0	X	0	0	-	4	16	
	LDD	11	101	101	ED	(DE)+(HL),DE+DE-1	-	-	X	0	X	*M	0	-	4	16	
		10	101	000	A8	HL+HL-1,BC+BC-1	-	-	X	0	X	0	0	-	5	21	
	LDDR	11	101	101	ED	(DE)+(HL),DE+DE-1	-	-	X	0	X	0	0	-	4	16	
		10	111	000	B8	HL+HL-1,BC+BC-1 Repeat until BC=0	-	-	X	0	X	0	0	-	4	16	
	CPI	11	101	101	ED	A-(HL)	*	*N	X	*	X	*M	1	-	4	16	
		10	100	001	A1	HL+HL+1,BC+BC-1	-	-	X	0	X	0	0	-	5	21	
CPIR	11	101	101	ED	A-(HL),HL+HL+1,BC+BC-1	*	*N	X	*	X	*M	1	-	4	16		
	10	110	001	B1	Repeat until A=(HL) or BC=0	-	-	X	0	X	0	0	-	5	21		
CPD	11	101	101	ED	A-(HL)	*	*N	X	*	X	*M	1	-	4	16		
	10	101	001	A9	HL+HL-1,BC+BC-1	-	-	X	0	X	0	0	-	5	21		
CPDR	11	101	101	ED	A-(HL),HL+HL-1,BC+BC-1	*	*N	X	*	X	*M	1	-	4	16		
	10	111	001	B9	Repeat until A=(HL) or BC=0	-	-	X	0	X	0	0	-	5	21		
8-BIT ARITHMETIC AND LOGICAL	ADD A,r	10	000	rrr	B0+r	A+A+r	*	*	X	*	X	V	0	*	1	4	
	ADD A,n	11	000	110	C6	A+A+n	*	*	X	*	X	V	0	*	2	7	
		nn	nnn	nnn	n												
	ADD A,(HL)	10	000	110	B6	A+A+(HL)	*	*	X	*	X	V	0	*	2	7	
	ADD A,(IX+d)	11	011	101	DD	A+A+(IX+d)	*	*	X	*	X	V	0	*	5	19	
		10	000	110	B6												
		dd	ddd	ddd	d												
	ADD A,(IY+d)	11	111	101	FD	A+A+(IY+d)	*	*	X	*	X	V	0	*	5	19	
		10	000	110	B6												
		dd	ddd	ddd	d												
	ADC A,r	10	001	rrr	B5+r	A+A+r+CY	*	*	X	*	X	V	0	*	1	4	
	ADC A,n	11	001	110	CE	A+A+n+CY	*	*	X	*	X	V	0	*	2	7	
		nn	nnn	nnn	n												
	ADC A,(HL)	10	001	110	BE	A+A+(HL)+CY	*	*	X	*	X	V	0	*	2	7	
	ADC A,(IX+d)	11	011	101	DD	A+A+(IX+d)+CY	*	*	X	*	X	V	0	*	5	19	
		10	001	110	BE												
	dd	ddd	ddd	d													
ADC A,(IY+d)	11	111	101	FD	A+A+(IY+d)+CY	*	*	X	*	X	V	0	*	5	19		
	10	001	110	BE													
	dd	ddd	ddd	d													
SUB r	10	010	rrr	90+r	A+A-r	*	*	X	*	X	V	1	*	1	4		
SUB n	11	010	110	D6	A+A-n	*	*	X	*	X	V	1	*	2	7		
	nn	nnn	nnn	n													
SUB (HL)	10	010	110	96	A+A-(HL)	*	*	X	*	X	V	1	*	2	7		
SUB (IX+d)	11	011	101	DD	A+A-(IX+d)	*	*	X	*	X	V	1	*	5	19		
	10	010	110	96													
	dd	ddd	ddd	d													
SUB (IY+d)	11	111	101	FD	A+A-(IY+d)	*	*	X	*	X	V	1	*	5	19		
	10	010	110	96													
	dd	ddd	ddd	d													

+ [BC < > 0]
+ [BC=0]
+ [BC < > 0]
+ [BC=0]
+ [BC < > 0 & A < > (HL)]
* [BC=0 or A = (HL)]
+ [BC < > 0 & A < > (HL)]
* [BC=0 or A = (HL)]

r	rrr
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Note : *M P/V flag is 0 if the result of BC-1=0, otherwise P/V=1.
*N Z flag is 1 if A=(HL), otherwise Z=0.
[] indicates the total condition of the number of cycles and states indicated by arrow.
r means any of the registers A, B, C, D, E, H, L.

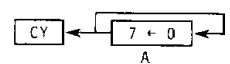
TMPZ84C011A Instruction Set (4/9)

ITEM/ CLASSI- FICATION	Assembler mnemonic	Object code		Function	Flag							No. OF CY- CLES	No. OF STA- TES					
		Binary			Hex	S	Z	H	P/V	N	C			r	rrr			
		76	543													210		
8 - BIT ARITHMETIC AND LOGICAL	SBC A,r	10	011	rrr	98+r	A←A-r-CY	*	*	X	*	X	V	1	*	1	4	r	rrr
	SBC A,n	11	011	110	DE	A←A-n-CY	*	*	X	*	X	V	1	*	2	7	B	000
		nn	nnn	nnn	n												C	001
	SBC A,(HL)	10	011	110	9E	A←A-(HL)-CY	*	*	X	*	X	V	1	*	2	7	D	010
	SBC A,(IX+d)	11	011	101	DD	A←A-(IX+d)-CY	*	*	X	*	X	V	1	*	5	19	E	011
		10	011	110	9E												H	100
		dd	ddd	ddd	d												L	101
	SBC A,(IY+d)	11	111	101	FD	A←A-(IY+d)-CY	*	*	X	*	X	V	1	*	5	19	A	111
		10	011	110	9E													
		dd	ddd	ddd	d													
	AND r	10	100	rrr	A0+r	A←A∧r	*	*	X	1	X	P	0	0	1	4		
	AND n	11	100	110	E6	A←A∧n	*	*	X	1	X	P	0	0	2	7		
		nn	nnn	nnn	n													
	AND (HL)	10	100	110	A6	A←A∧(HL)	*	*	X	1	X	P	0	0	2	7		
	AND (IX+d)	11	011	101	DD	A←A∧(IX+d)	*	*	X	1	X	P	0	0	5	19		
		10	100	110	A6													
		dd	ddd	ddd	d													
	AND (IY+d)	11	111	101	FD	A←A∧(IY+d)	*	*	X	1	X	P	0	0	5	19		
		10	100	110	A6													
		dd	ddd	ddd	d													
	OR r	10	110	rrr	B0+r	A←A∨r	*	*	X	0	X	P	0	0	1	4		
	OR n	11	110	110	F6	A←A∨n	*	*	X	0	X	P	0	0	2	7		
		nn	nnn	nnn	n													
	OR (HL)	10	110	110	B6	A←A∨(HL)	*	*	X	0	X	P	0	0	2	7		
	OR (IX+d)	11	011	101	DD	A←A∨(IX+d)	*	*	X	0	X	P	0	0	5	19		
		10	110	110	B6													
		dd	ddd	ddd	d													
	OR (IY+d)	11	111	101	FD	A←A∨(IY+d)	*	*	X	0	X	P	0	0	5	19		
		10	110	110	B6													
		dd	ddd	ddd	d													
XOR r	10	101	rrr	A8+r	A←A⊕r	*	*	X	0	X	P	0	0	1	4			
XOR n	11	101	110	EE	A←A⊕n	*	*	X	0	X	P	0	0	2	7			
	nn	nnn	nnn	n														
XOR (HL)	10	101	110	AE	A←A⊕(HL)	*	*	X	0	X	P	0	0	2	7			
XOR (IX+d)	11	011	101	DD	A←A⊕(IX+d)	*	*	X	0	X	P	0	0	5	19			
	10	101	110	AE														
	dd	ddd	ddd	d														
XOR (IY+d)	11	111	101	FD	A←A⊕(IY+d)	*	*	X	0	X	P	0	0	5	19			
	10	101	110	AE														
	dd	ddd	ddd	d														
CP r	10	111	rrr	B8+r	A-r	*	*	X	*	X	V	1	*	1	4			
CP n	11	111	110	FE	A-n	*	*	X	*	X	V	1	*	2	7			
	nn	nnn	nnn	n														
CP (HL)	10	111	110	BE	A-(HL)	*	*	X	*	X	V	1	*	2	7			
CP (IX+d)	11	011	101	DD	A-(IX+d)	*	*	X	*	X	V	1	*	5	19			
	10	111	110	BE														
	dd	ddd	ddd	d														
CP (IY+d)	11	111	101	FD	A-(IY+d)	*	*	X	*	X	V	1	*	5	19			
	10	111	110	BE														
	dd	ddd	ddd	d														
INC r	00	rrr	100	04+r×8	r←r+1	*	*	X	*	X	V	0	-	1	4			
INC (HL)	00	110	100	34	(HL)←(HL)+1	*	*	X	*	X	V	0	-	3	11			
INC (IX+d)	11	011	101	DD	(IX+d)←(IX+d)+1	*	*	X	*	X	V	0	-	6	23			
	00	110	100	34														
	dd	ddd	ddd	d														

Note : r means any of the registers A, B, C, D, E, H, L.

120489

TMPZ84C011A Instruction Set (5/9)

ITEM/ CLASSI- FICA- TION	Assembler mnemonic	Object code		Function	Flag						No. OF CY- CLES	No. OF STA- TES			
		Binary			Hex	S	Z	H	P/V	N			C		
		76 543 210													
8-BIT ARITHMETIC AND LOGICAL	INC (IY+d)	11 111 101 00 110 100 dd ddd ddd	FD 34 d	(IY+d)+(IY+d)+1	*	*	X	*	X	V	0	-	6	23	
	DEC r	00 rrr 101	05+r×8	r-r-1	*	*	X	*	X	V	1	-	1	4	r rrr
	DEC (HL)	00 110 101	35	(HL)+(HL)-1	*	*	X	*	X	V	1	-	3	11	B 000
	DEC (IX+d)	11 011 101 00 110 101 dd ddd ddd	DD 35 d	(IX+d)+(IX+d)-1	*	*	X	*	X	V	1	-	6	23	C 001 D 010 E 011
	DEC (IY+d)	11 111 101 00 110 101 dd ddd ddd	FD 35 d	(IY+d)+(IY+d)-1	*	*	X	*	X	V	1	-	6	23	H 100 L 101 A 111
GENERAL-PURPOSE ARITHMETIC AND MPU CONTROL	DAA	00 100 111	27	Decimal adjust accumulator	*	*	X	*	X	P	-	*	1	4	
	CPL	00 101 111	2F	A←Ā	-	-	X	1	X	-	1	-	1	4	
	NEG	11 101 101 01 000 100	ED 44	A←0-A	*	*	X	*	X	V	1	*	2	8	
	CCF	00 111 111	3F	CY←CY	-	-	X	X	X	-	0	*	1	4	
	SCF	00 110 111	37	CY←1	-	-	X	0	X	-	0	1	1	4	
	NOP	00 000 000	00	no operation	-	-	X	-	X	-	-	-	1	4	
	HALT	01 110 110	76	MPU Halted	-	-	X	-	X	-	-	-	1	4	
	DI	11 110 011	F3	IFF←0	-	-	X	-	X	-	-	-	1	4	
	EI	11 111 011	FB	IFF←1	-	-	X	-	X	-	-	-	1	4	
	IM 0	11 101 101 01 000 110	ED 46	Set interrupt mode 0	-	-	X	-	X	-	-	-	2	8	
	IM 1	11 101 101 01 010 110	ED 56	Set interrupt mode 1	-	-	X	-	X	-	-	-	2	8	
	IM 2	11 101 101 01 011 110	ED 5E	Set interrupt mode 2	-	-	X	-	X	-	-	-	2	8	
16-BIT ARITHMETIC	ADD HL,t	00 tt1 001	09+t×10	HL+HL+t	-	-	X	X	X	-	0	*	3	11	t tt
	ADC HL,t	11 101 101 01 tt1 010	ED 4A+t×10	HL+HL+t+CY	*	*	X	X	X	V	0	*	4	15	BC 00 DE 01 HL 10 SP 11
	SBC HL,t	11 101 101 01 tt0 010	ED 42+t×10	HL+HL-t-CY	*	*	X	X	X	V	1	*	4	15	
	ADD IX,p	11 011 101 00 pp1 001	DD 09+p×10	IX+IX+p	-	-	X	X	X	-	0	*	4	15	p pp
	ADD IY,s	11 111 101 00 ss1 001	FD 09+s×10	IY+IY+s	-	-	X	X	X	-	0	*	4	15	BC 00 DE 01
	INC t	00 tt0 011	03+t×10	t←t+1	-	-	X	-	X	-	-	-	1	6	IX 10
	INC IX	11 011 101 00 100 011	DD 23	IX+IX+1	-	-	X	-	X	-	-	-	2	10	SP 11
	INC IY	11 111 101 00 100 011	FD 23	IY+IY+1	-	-	X	-	X	-	-	-	2	10	s ss
	DEC t	00 tt1 011	0B+t×10	t←t-1	-	-	X	-	X	-	-	-	1	6	BC 00 DE 01
	DEC IX	11 011 101 00 101 011	DD 2B	IX+IX-1	-	-	X	-	X	-	-	-	2	10	IY 10 SP 11
	DEC IY	11 111 101 00 101 011	FD 2B	IY+IY-1	-	-	X	-	X	-	-	-	2	10	
	ROTATE	RLCA	00 000 111	07		-	-	X	0	X	-	0	*	1	4

Note : ss is any of the register pairs BC, DE, HL, SP. PP is any of the register pairs BC, DE, IX, SP.
rr is any of the register pairs BC, DE, IY, SP.

TMPZ84C011A Instruction Set (6/9)

ITEM/ CLASSI- FICATION	Assembler mnemonic	Object code		Function	Flag							No. OF CY- CLES	No. OF STA- TES		
		Binary	Hex		S	Z	H	P/V	N	C					
		76 543 210													
ROTATE SHIFT	RLA	00 010 111	17		-	-	X	0	X	-	0	*	1	4	
	RRCA	00 001 111	0F		-	-	X	0	X	-	0	*	1	4	
	RRA	00 011 111	1F		-	-	X	0	X	-	0	*	1	4	
	RLC r	11 001 011 00 000 rrr	CB 00+r		*	*	X	0	X	P	0	*	2	8	
	RLC (HL)	11 001 011 00 000 110	CB 06		*	*	X	0	X	P	0	*	4	15	
	RLC (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 000 110	DD CB d 06		*	*	X	0	X	P	0	*	6	23	
	RLC (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 000 110	FD CB d 06		*	*	X	0	X	P	0	*	6	23	
	RL r	11 001 011 00 010 rrr	CB 10+r		*	*	X	0	X	P	0	*	2	8	
	RL (HL)	11 001 011 00 010 110	CB 16		*	*	X	0	X	P	0	*	4	15	
	RL (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 010 110	DD CB d 16		*	*	X	0	X	P	0	*	6	23	
	RL (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 010 110	FD CB d 16		*	*	X	0	X	P	0	*	6	23	
	RRC r	11 001 011 00 001 rrr	CB 08+r		*	*	X	0	X	P	0	*	2	8	
	RRC (HL)	11 001 011 00 001 110	CB 0E		*	*	X	0	X	P	0	*	4	15	
	RRC (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 001 110	DD CB d 0E			*	*	X	0	X	P	0	*	6	23
	RRC (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 001 110	FD CB d 0E		*	*	X	0	X	P	0	*	6	23	
	RR r	11 001 011 00 011 rrr	CB 18+r		*	*	X	0	X	P	0	*	2	8	
	RR (HL)	11 001 011 00 011 110	CB 1E			*	*	X	0	X	P	0	*	4	15
	RR (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 011 110	DD CB d 1E			*	*	X	0	X	P	0	*	6	23

r	rrr
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Note : r means any of the registers A, B, C, D, E, H, L.

120489

TMPZ84C011A Instruction Set (7/9)

ITEM CLASSIFICATION	Assembler mnemonic	Object code		Function	Flag						No. OF CYCLES	No. OF STATES				
		Binary	Hex		S	Z	H	P/V	N	C						
		76 543 210														
ROTATE SHIFT	RR (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 011 110	FD CB d 1E		*	*	X	0	X	P	0	*	6	23		
	SLA r	11 001 011 00 100 rrr	CB 20+r		*	*	X	0	X	P	0	*	2	8	r rrr	
	SLA (HL)	11 001 011 00 100 110	CB 26		*	*	X	0	X	P	0	*	4	15	B 000 C 001 D 010 E 011 H 100 L 101 A 111	
	SLA (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 100 110	DD CB d 26		*	*	X	0	X	P	0	*	6	23		
	SLA (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 100 110	FD CB d 26		*	*	X	0	X	P	0	*	6	23		
	SRA r	11 001 011 00 101 rrr	CB 28+r		*	*	X	0	X	P	0	*	2	8		
	SRA (HL)	11 001 011 00 101 110	CB 2E		*	*	X	0	X	P	0	*	4	15		
	SRA (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 101 110	DD CB d 2E		*	*	X	0	X	P	0	*	6	23		
	SRA (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 101 110	FD CB d 2E		*	*	X	0	X	P	0	*	6	23		
	SRL r	11 001 011 00 111 rrr	CB 38+r		*	*	X	0	X	P	0	*	2	8		
	SRL (HL)	11 001 011 00 111 110	CB 3E		*	*	X	0	X	P	0	*	4	15		
	SRL (IX+d)	11 011 101 11 001 011 dd ddd ddd 00 111 110	DD CB d 3E		*	*	X	0	X	P	0	*	6	23		
	SRL (IY+d)	11 111 101 11 001 011 dd ddd ddd 00 111 110	FD CB d 3E		*	*	X	0	X	P	0	*	6	23		
	RLD	11 101 101 01 101 111	ED 6F		*	*	X	0	X	P	0	-	5	18	*1	
	RRD	11 101 101 01 100 111	ED 67		*	*	X	0	X	P	0	-	5	18	*1	
	BIT SET RESET AND TEST	BIT b, r	11 001 011 01 bbb rrr	CB 40+b x 8+r	Z←r _b	X	*	X	1	X	X	0	-	2	8	b bbb
		BIT b, (HL)	11 001 011 01 bbb 110	CB 46+b x 8	Z←(HL) _b	X	*	X	1	X	X	0	-	3	12	0 000 1 001 2 010 3 011 4 100 5 101 6 110 7 111

Note : *1 : Rotate digit left and right between the accumulator and location (HL).
 The content of the upper half of the accumulator is unaffected.
 The notation (HL)_b indicates bit _b (0 to 7) within the contents of the HL register pair.
 The notation r_b indicates bit _b (0 to 7) within the r register.

TMPZ84C011A Instruction Set (8/9)

ITEM CLASSIFICATION	Assembler mnemonic	Object code		Function	Flag							No. OF CYCLES	No. OF STATES				
		Binary			Hex		S	Z	H	P/V	N			C	r	rrrr	
		76	543		210												
BIT SET RESET AND TEST	BIT b,(IX+d)	11 011 101	DD	$Z+(IX+d)_b$	X	*	X	1	X	X	0	-	5	20			
		11 001 011	CB														
		dd ddd ddd	d														
	BIT b,(IY+d)	01 bbb 110	46+b×8														
		11 111 101	FD	$Z-(IY+d)_b$	X	*	X	1	X	X	0	-	5	20			
		11 001 011	CB														
	JUMP	SET b,r	dd ddd ddd	d													
			01 bbb 110	46+b×8													
			11 001 011	CB	r_b+1	-	-	X	-	X	-	-	-	2	8	r	rrrr
		SET b,(HL)	11 bbb rrr	C0+b×8+r													
			11 001 011	CB	$(HL)_b+1$	-	-	X	-	X	-	-	-	4	15	B	000
			11 bbb 110	C6+b×8												C	001
SET b,(IX+d)		11 011 101	DD	$(IX+d)_b+1$	-	-	X	-	X	-	-	-	6	23	D	010	
		11 001 011	CB												E	011	
		dd ddd ddd	d												H	100	
SET b,(IY+d)		11 bbb 110	C6+b×8												L	101	
		11 111 101	FD	$(IY+d)_b+1$	-	-	X	-	X	-	-	-	6	23	A	111	
		11 001 011	CB														
JUMP	RES b,r	dd ddd ddd	d														
		11 bbb 110	C6+b×8														
		11 001 011	CB	r_b+0	-	-	X	-	X	-	-	-	2	8	b	bbb	
	RES b,(HL)	10 bbb rrr	80+b×8+r														
		11 001 011	CB	$(HL)_b+0$	-	-	X	-	X	-	-	-	4	15	0	000	
		10 bbb 110	86+b×8												1	001	
	RES b,(IX+d)	11 011 101	DD	$(IX+d)_b+0$	-	-	X	-	X	-	-	-	6	23	2	010	
		11 001 011	CB												3	011	
		dd ddd ddd	d												4	100	
	RES b,(IY+d)	10 bbb 110	86+b×8												5	101	
		11 111 101	FD	$(IY+d)_b+0$	-	-	X	-	X	-	-	-	6	23	6	110	
		11 001 011	CB												7	111	
JUMP	JP mn	dd ddd ddd	d														
		10 bbb 110	86+b×8														
		11 001 011	CB	$PC+mn$	-	-	X	-	X	-	-	-	3	10			
	JP c,ma	11 ccc 010	C2+c×8														
		nn nnn nnn	n	$PC+mn$	-	-	X	-	X	-	-	-	3	10			
		mm mmm mmm	m	(Only when condition is met)													
	JR \$+e	00 011 000	18	$PC+e$	-	-	X	-	X	-	-	-	3	12			
		aa aaa aaa	a														
		00 111 000	38	If C=0, continue	-	-	X	-	X	-	-	-	2	7			
	JR C,\$+e	aa aaa aaa	a	If C=1, PC+\$+e	-	-	X	-	X	-	-	-	3	12			
		00 110 000	30	If C=0, PC+\$+e	-	-	X	-	X	-	-	-	3	12			
		aa aaa aaa	a	If C=1, continue	-	-	X	-	X	-	-	-	2	7			
JR Z,\$+e	00 101 000	28	If Z=0, continue	-	-	X	-	X	-	-	-	2	7				
	aa aaa aaa	a	If Z=1, PC+\$+e	-	-	X	-	X	-	-	-	3	12				
	00 100 000	20	If Z=0, PC+\$+e	-	-	X	-	X	-	-	-	3	12				
JR NZ,\$+e	aa aaa aaa	a	If Z=1, continue	-	-	X	-	X	-	-	-	2	7				
	00 010 000	10	B+B-1, if B=0, continue	-	-	X	-	X	-	-	-	2	8				
	aa aaa aaa	a	B+B-1, if B<0, continue	-	-	X	-	X	-	-	-	3	13				
JP (HL)	11 101 001	E9	$PC+HL$	-	-	X	-	X	-	-	-	1	4				

Note: • a = e - 2 in the Opcode provides an effective address of PC + e as PC is incremented by 2 before the addition of e.
 • \$ indicates the reference to the location counter value of the current segment.
 • The notation $(HL)_b$, $(IX + d)_b$ indicates bit b (0 to 7) within the contents of the register pair.
 • The notation r_b indicates bit b (0 to 7) within the r register.
 • a = e - 2 in the op-code provides effective address of PC + e as PC is incremented by 2 prior to the addition of e.

c	ccc	Condition
NZ	000	Non-Zero
Z	001	Zero
NC	010	No-carry
C	011	Carry
PO	100	Odd Parity
PE	101	Even Parity
P	110	Sign Positive
M	111	Sign Negative

TMPZ84C011A Instruction Set (9/9)

ITEM/ CLASSI- FICA- TION	Assembler mnemonic	Object code		Function	Flag							No. OF CY- CLES	No. OF STA- TES	
		Binary	Hex		S	Z	H	P/V	N	C				
		76 543 210												
JUMP	JP (IX)	11 011 101	DD	PC+(IX)	-	-	X	-	X	-	-	-	2	8
	JP (IY)	11 111 101	FD	PC+(IY)	-	-	X	-	X	-	-	-	2	8
CALL AND RETURN	CALL mn	11 001 101	CD	(SP-1)+PC _H , (SP-2)+PC _L	-	-	X	-	X	-	-	-	5	17
		nn nnn nnn	n	PC+mn										
		mm mmm mmm	m	SP+SP-2										
	CALL c, mn	11 ccc 100	C4+cX8	If condition c is met, same as CALL mn	-	-	X	-	X	-	-	-	5	17
		nn nnn nnn	n	If condition c is not met, continue										
		mm mmm mmm	m	PC _L ←(SP), PC _H ←(SP+1)										
	RET	11 001 001	C9	SP+SP+2	-	-	X	-	X	-	-	-	3	10
CALL AND RETURN	RET c	11 ccc 000	C0+cX8	If condition c is met, same as RET	-	-	X	-	X	-	-	-	3	11
		nn nnn nnn	n	If condition c is not met, continue										
	RETI	11 101 101	ED	Return from interrupt Processing routine	-	-	X	-	X	-	-	-	4	14
	RETN	01 101 101	4D	Return from non-maskable interrupt Processing routine	-	-	X	-	X	-	-	-	4	14
		01 000 101	45											
	RST j	11 kkk 111	C7+kX8	(SP-1)+PC _H , (SP-2)+PC _L	-	-	X	-	X	-	-	-	3	11
				PC _H ←0, PC _L ←j, SP+SP-2										
INPUT AND OUTPUT	IN A, (n)	11 011 011	DB	A←(n)	-	-	X	-	X	-	-	-	3	11
		nn nnn nnn	n	n→A0-A7, A→A8-A15										
	IN r, (C)	11 101 101	ED	r←(C) if r=110, only the flags will be affected.	*	*	X	*	X	P	0	-	3	12
	INI	11 101 101	ED	(HL)←(C), B←B-1, HL←HL+1	X	*M	X	X	X	X	1	X	4	16
		10 100 010	A2											
	INIR	11 101 101	ED	(HL)←(C), B←B-1, HL←HL+1	X	1	X	X	X	X	1	X	5	21
		10 110 010	B2	Repeat until B=0										
	IND	11 101 101	ED	(HL)←(C), B←B-1, HL←HL-1	X	*M	X	X	X	X	1	X	4	16
		10 101 010	AA											
	INDR	11 101 101	ED	(HL)←(C), B←B-1, HL←HL-1	X	1	X	X	X	X	1	X	5	21
		10 111 010	BA	Repeat until B=0										
	OUT (n), A	11 010 011	D3	(n)←A	-	-	X	-	X	-	-	-	3	11
		nn nnn nnn	n	n→A0-A7, A→A8-A15										
OUT (C), r	11 101 101	ED	(C)←r	-	-	X	-	X	-	-	-	3	12	
	01 rrr 001	41+rX8												
OUTI	11 101 101	ED	(C)←(HL), B←B-1, HL←HL+1	X	*M	X	X	X	X	1	X	4	16	
	10 100 011	A3												
OTIR	11 101 101	ED	(C)←(HL), B←B-1, HL←HL+1	X	1	X	X	X	X	1	X	5	21	
	10 110 011	B3	Repeat until B=0											
OUTD	11 101 101	ED	(C)←(HL), B←B-1, HL←HL-1	X	*M	X	X	X	X	1	X	4	16	
	10 101 011	AB												
OTDR	11 101 101	ED	(C)←(HL), B←B-1, HL←HL-1	X	1	X	X	X	X	1	X	5	21	
	10 111 010	BB	Repeat until B=0											

Note: • *M If the result of B-1 is zero, the Z flag is set, otherwise it is reset.
 • A0 through A15 indicate the address bus.
 • [] indicates the total condition of the number of cycles and states indicated by arrow.

*1	C→A0-A7	c	ccc	Condition
	B→A8-A15	NZ	000	Non-Zero
		Z	001	Zero
		MC	010	No-Carry
		C	011	Carry
		PO	100	Odd Parity
		PE	101	Even Parity
		P	110	Sign Positive
		M	111	Sign negative

j	kkk
00H	000
08H	001
10H	010
18H	011
20H	100
28H	101
30H	110
38H	111
r	rrr
B	000
C	001
D	010
E	011
H	100
L	101
A	111

TMPZ84C011A Instruction Map (1/7)

MPU Instruction Table (I)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0	NOB	LD BC, mn	LD (BC), A	INC BC	INC B	DEC B	LD B, n	RLCA	EX AF, AF	ADD HC, BC	LD A, (BC)	DEC BC	INC C	DEC C	LD C, n	RRCA
	1	DJNZ e	LD DE, mn	LD (DE), A	INC DE	INC D	DEC D	LD D, N	RLA	JE e	ADD HL, DE	LD A, (DE)	DEC DE	INC E	DEC E	LD E, n	RRA
	2	JR NZ, e	LD HL, mn	LD (mn), HL	INC HL	INC D	DEC H	LD H, n	DAA	JR Z e	ADD HL, HL	LD HL, (mn)	DEC HL	INC L	DEC L	LD L, n	CPL
	3	JR NC, e	LD SP, mn	LD (mn), A	INC SP	INC (HL)	DEC {HL}	LD {HL}, n	SCF	JR C e	ADD HL, SP	LD A, (mn)	DEC SP	INC A	DEC A	LD A, n	CCF
	4	LD B, B	LD B, C	LD B, D	LD B, E	LD B, H	LD B, L	LD B, (HL)	LD B, A	LD C, B	LD C, C	LD C, D	LD C, E	LD C, H	LD C, L	LD C, (HL)	LD C, A
	5	LD D, B	LD D, C	LD D, D	LD D, E	LD D, H	LD D, L	LD D, (HL)	LD D, A	LD E, B	LD E, C	LD E, D	LD E, E	LD E, H	LD E, L	LD E, (HL)	LD E, A
	6	LD H, B	LD H, C	LD H, D	LD H, E	LD H, H	LD H, L	LD H, (HL)	LD H, A	LD L, B	LD L, C	LD L, D	LD L, E	LD L, H	LD L, L	LD L, (HL)	LD L, A
	7	LD (HL), B	LD (HL), C	LD (HL), D	LD (HL), E	LD (HL), H	LD (HL), L	HALT	LD (HL), A	LD A, B	LD A, C	LD A, D	LD A, E	LD A, H	LD A, L	LD A, (HL)	LD A, A
	8	ADD A, B	ADD A, C	ADD A, D	ADD A, E	ADD A, H	ADD A, L	ADD A, (HL)	ADD A, A	ADC A, B	ADC A, C	ADC A, D	ADC A, E	ADC A, H	ADC A, L	ADC A, (HL)	ADC A, A
	9	SUB B	SUB C	SUB D	SUB E	SUB H	SUB L	SUB (HL)	SUB A	SBC A, B	SBC A, C	SBC A, D	SBC A, E	SBC A, H	SBC A, L	SBC A, (HL)	SBC A, A
	A	AND B	AND C	AND D	AND E	AND H	AND L	AND (HL)	AND A	XOR B	XOR C	XOR D	XOR E	XOR H	XOR L	XOR (HL)	XOR A
	B	OR B	OR C	OR D	OR E	OR H	OR L	OR (HL)	OR A	CP B	CP C	CP D	CP E	CP H	CP L	CP (HL)	XOR A
	C	RET NZ	POP BC	JP NZ, mn	JP mn	CALL NZ, mn	PUSH BC	ADD A, n	RST 00H	RET C	RET	JP Z, mn	①	CALL Z, mn	CALL mn	ADC A, n	RST 08H
	D	RET NC	POP DE	JP NC, mn	OUT (n), A	CALL NC, mn	PUSH DE	SUB n	RST 10H	RET C	EXX	JP C, mn	IN A, (n)	CALL C, mn	③	SBC A, n	RST 18H
	E	RET PO	POP HL	JP PO, mn	EX (SP), HL	CALL PO, mn	PUSH HL	AND n	RST 20H	RET PE	JP (HL)	JP PE, mn	EX DE, HL	CALL PI, mn	②	XOR n	RST 28H
	F	RET P	POP AF	JP P, mn	DI	CALL P, mn	PUSH AF	OR n	RST 30H	RET M	LD SP, HL	JP M, mn	EI	CALL M, mn	④	CP n	RST 38H

120489

Note ①~④: Multi-Opcode Instructions (ref. Table (II)~(VI))

TMPZ84C011A Instruction Map (2/7)

① Byte 1 “CB”

Instruction Table (II) (Byte 2 of 2-byte Opcode)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		RLC B	RLC C	RLC D	RLC E	RLC H	RLC L	RLC (HL)	RLC A	RRC B	RRC C	RRC D	RRC E	RRC H	RRC L	RRC (HL)	RRC A
1		RL B	RL C	RL D	RL E	RL H	RL L	RL (HL)	RL A	RR B	RR C	RR D	RR E	RR H	RR L	RR (HL)	RR A
2		SLA B	SLA C	SLA D	SLA E	SLA H	SLA L	SLA (HL)	SLA A	SRA B	SRA C	SRA D	SRA E	SRA H	SRA L	SRA (HL)	SRA A
3										SRL B	SRL C	SRL D	SRL E	SRL H	SRL L	SRL (HL)	SRL A
4		BIT 0, B	BIT 0, C	BIT 0, D	BIT 0, E	BIT 0, H	BIT 0, L	BIT 0, (HL)	BIT 0, A	BIT 1, B	BIT 1, C	BIT 1, D	BIT 1, E	BIT 1, H	BIT 1, L	BIT 1, (HL)	BIT 1, A
5		BIT 2, B	BIT 2, C	BIT 2, D	BIT 2, E	BIT 2, H	BIT 2, L	BIT 2, (HL)	BIT 2, A	BIT 3, B	BIT 3, C	BIT 3, D	BIT 3, E	BIT 3, H	BIT 3, L	BIT 3, (HL)	BIT 3, A
6		BIT 4, B	BIT 4, C	BIT 4, D	BIT 4, E	BIT 4, H	BIT 4, L	BIT 4, (HL)	BIT 4, A	BIT 5, B	BIT 5, C	BIT 5, D	BIT 5, E	BIT 5, H	BIT 5, L	BIT 5, (HL)	BIT 5, A
7		BIT 6, B	BIT 6, C	BIT 6, D	BIT 6, E	BIT 6, H	BIT 6, L	BIT 6, (HL)	BIT 6, A	BIT 7, B	BIT 7, C	BIT 7, D	BIT 7, E	BIT 7, H	BIT 7, L	BIT 7, (HL)	BIT 7, A
8		RES 0, B	RES 0, C	RES 0, D	RES 0, E	RES 0, H	RES 0, L	RES 0, (HL)	RES 0, A	RES 1, B	RES 1, C	RES 1, D	RES 1, E	RES 1, H	RES 1, L	RES 1, (HL)	RES 1, A
9		RES 2, B	RES 2, C	RES 2, D	RES 2, E	RES 2, H	RES 2, L	RES 2, (HL)	RES 2, A	RES 3, B	RES 3, C	RES 3, D	RES 3, E	RES 3, H	RES 3, L	RES 3, (HL)	RES 3, A
A		RES 4, B	RES 4, C	RES 4, D	RES 4, E	RES 4, H	RES 4, L	RES 4, (HL)	RES 4, A	RES 5, B	RES 5, C	RES 5, D	RES 5, E	RES 5, H	RES 5, L	RES 5, (HL)	RES 5, A
B		RES 6, B	RES 6, C	RES 6, D	RES 6, E	RES 6, H	RES 6, L	RES 6, (HL)	RES 6, A	RES 7, B	RES 7, C	RES 7, D	RES 7, E	RES 7, H	RES 7, L	RES 7, (HL)	RES 7, A
C		SET 0, B	SET 0, C	SET 0, D	SET 0, E	SET 0, H	SET 0, L	SET 0, (HL)	SET 0, A	SET 1, B	SET 1, C	SET 1, D	SET 1, E	SET 1, H	SET 1, L	SET 1, (HL)	SET 1, A
D		SET 2, B	SET 2, C	SET 2, D	SET 2, E	SET 2, H	SET 2, L	SET 2, (HL)	SET 2, A	SET 3, B	SET 3, C	SET 3, D	SET 3, E	SET 3, H	SET 3, L	SET 3, (HL)	SET 3, A
E		SET 4, B	SET 4, C	SET 4, D	SET 4, E	SET 4, H	SET 4, L	SET 4, (HL)	SET 4, A	SET 5, B	SET 5, C	SET 5, D	SET 5, E	SET 5, H	SET 5, L	SET 5, (HL)	SET 5, A
F		SET 6, B	SET 6, C	SET 6, D	SET 6, E	SET 6, H	SET 6, L	SET 6, (HL)	SET 6, A	SET 7, B	SET 7, C	SET 7, D	SET 7, E	SET 7, H	SET 7, L	SET 7, (HL)	SET 7, A

120489

TMPZ84C011A Instruction Map (3/7)

② Byte 1 "ED"

Instruction Table (III) (Byte 2 of 2-byte Opcode)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																	
1																	
2																	
3																	
4	IN B, (C)	OUT (C), B	SBC HL,BC	LD (mn),BC	NEG	RETN	IM0	LD I, A	IN C, (C)	OUT (C), C	ADC HL,BC	LD BC,(mn)		RETI		LD R, A	
5	IN D, (C)	OUT (C), D	SBC HL,DE	LD (mn),DE			IM1	LD A, I	IN E, (C)	OUT (C), E	ADC HL,DE	LD DE,(mn)			IM2	LD A, R	
6	IN H, (C)	OUT (C), H	SBC HL,HL	LD (mn),HL				RRD	IN L, (C)	OUT (C), L	ADC HL,HL	LD HL,(mn)				RLD	
7			SBC HL,SP	LD (mn),SP					IN A, (C)	OUT (C), A	ADC HL,SP	LD SP,(mn)					
8																	
9																	
A	LDI	CPI	INI	OUTI						LDD	CPD	IND	OUTD				
B	LDIR	CPIR	INIR	OTIR						LDDR	CPDR	INDR	OTDR				
C																	
D																	
E																	
F																	

120489

TMPZ84C011A Instruction Map (4/7)

③ Byte 1 “DD”

Instruction Table (IV) (Byte 2 of 2-byte Opcode)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0										ADD IX, BC						
	1										ADD IX, DE						
	2		LD IX, mn	LD (mn), IX	INC IX						ADD IX, IX	LD IX, (mn)	DEC IX				
	3					INC (IX + d)	DEC (IX + d)	LD (IX + d) , P			ADD IX, SP						
	4							LD B, (IX + d)									LDC, (IX + d)
	5							LD D, (IX + d)									LDE, (IX + d)
	6							LD H, (IX + d)									LD L, (IX + d)
	7	LD (IX + d) , B	LD (IX + d) , C	LD (IX + d) , D	LD (IX + d) , E	LD (IX + d) , H	LD (IX + d) , L		LD (IX + d) , A								LDA, (IX + d)
	8							ADD (IX + d)									ADC I, (IX + d)
	9							SUB (IX + d)									SBC A, (IX + d)
	8							AND (IX + d)									XOR (IX + d)
	9							OR (IX + d)									CP (IX + d)
	C													Ⓢ			
	D																
	E		POP IX		EX (SP), IX		PUSH IX				JP (IX)						
	F										LD SP, IX						

120489

Note Ⓢ: Special 2 byte Opcode Instructions (ref. Table (VI))

TMPZ84C011A Instruction Map (5/7)

④ Byte 1 “FD”

Instruction Table (V) (Byte 2 of 2-byte Opcode)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0										ADD IY,BC						
	1										ADD IY,DE						
	2		LD IY,nn	ID (mn),IY	INC IY						ADD IY,IY	LD IY,(mn)	DEC IY				
	3					INC (IY + d)	DEC (IY + d)	LD (IY + d) ,n			ADD IY,SP						
	4							LD B, (IY + d)								LD C, (IY + d)	
	5							LD D, (IY + d)								LD E, (IY + d)	
	6							LD H, (IY + d)								LD L, (IY + d)	
	7	LD (IY + d) ,B	LD (IY + d) ,C	LD (IY + d) ,D	LD (IY + d) ,E	LD (IY + d) ,H	LD (IY + d) ,L		LD (IY + d) ,A							LDA, (IY + d)	
	8							ADD (IY + d)								ADDC A, (IY + d)	
	9							SUB (IY + d)								SUBC A, (IY + d)	
	8							AND (IY + d)								XOR (IY + d)	
	9							OR (IY + d)								CP (IY + d)	
	C												Ⓢ				
	D																
	E		POP IY		EX (SP),IY		PUSH IY				JP (IY)						
	F										LD SP,IY						

Note Ⓢ: Special 2 byte Opcode Instructions (ref. Table (VII))

120489

TMPZ84C011A Instruction Map (6/7)

- ⑤ Byte 1 “DD”
- Byte 2 “CB”

Instruction Table (VI) (Special case of 2-byte Opcode : Byte 3)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0								RLC (IX + d)								RRC (IX + d)	
1								RL (IX + d)								RR (IX + d)	
2								SLA (IX + d)								SRA (IX + d)	
3																SRL (IX + d)	
4								BIT 0, (IX + d)								BIT 1, (IX + d)	
5								BIT 2, (IX + d)								BIT 3, (IX + d)	
6								BIT 4, (IX + d)								BIT 5, (IX + d)	
7								BIT 6, (IX + d)								BIT 7, (IX + d)	
8								RES 0, (IX + d)								RES 1, (IX + d)	
9								RES 2, (IX + d)								RES 3, (IX + d)	
A								RES 4, (IX + d)								RES 5, (IX + d)	
B								RES 6, (IX + d)								RES 7, (IX + d)	
C								SET 0, (IX + d)								SET 1, (IX + d)	
D								SET 2, (IX + d)								SET 3, (IX + d)	
E								SET 4, (IX + d)								SET 5, (IX + d)	
F								SET 6, (IX + d)								SET 7, (IX + d)	

120489

TMPZ84C011A Instruction Map (7/7)

⑥ Byte 1 “FD”

Byte 2 “CB”

Instruction Table (VII) (Special case of 2-byte Opcode : Byte 3)

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0								RLC (IY + d)								RRC (IY + d)	
1								RL (IY + d)								RR (IY + d)	
2								SLA (IY + d)								SRA (IY + d)	
3																SRL (IY + d)	
4								BIT 0, (IY + d)								BIT 1, (IY + d)	
5								BIT 2, (IY + d)								BIT 3, (IY + d)	
6								BIT 4, (IY + d)								BIT 5, (IY + d)	
7								BIT 6, (IY + d)								BIT 7, (IY + d)	
8								RES 0, (IY + d)								RES 1, (IY + d)	
9								RES 2, (IY + d)								RES 3, (IY + d)	
A								RES 4, (IY + d)								RES 5, (IY + d)	
B								RES 6, (IY + d)								RES 7, (IY + d)	
C								SET 0, (IY + d)								SET 1, (IY + d)	
D								SET 2, (IY + d)								SET 3, (IY + d)	
E								SET 4, (IY + d)								SET 5, (IY + d)	
F								SET 6, (IY + d)								SET 7, (IY + d)	

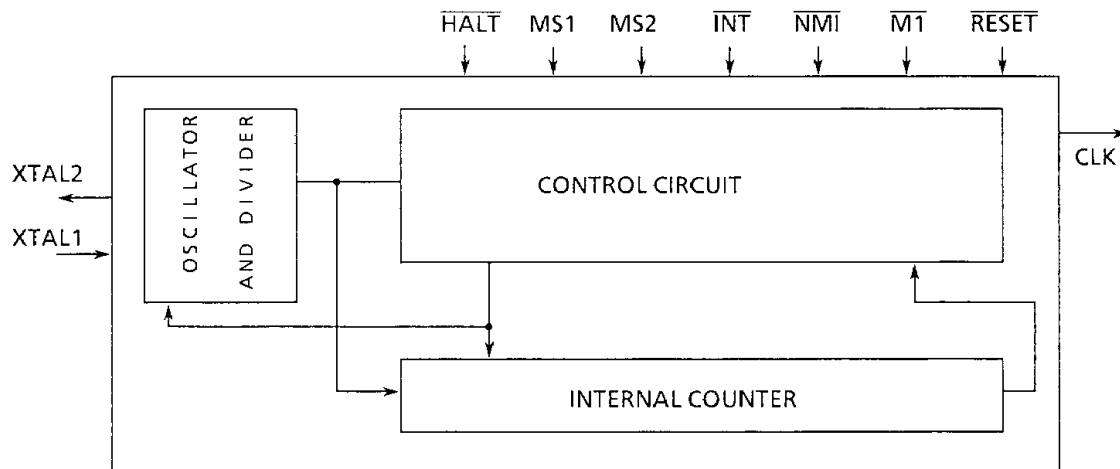
120489

3.3 CGC Operations

This subsection describes the system configuration, functions, and basic operations of the clock generator/controller (CGC).

3.3.1 Block Diagram

Figure 3.3.1 shows the block diagram of the CGC.



120489

Figure 3.3.1 Block Diagram

3.3.2 CGC System Configuration

The internal configuration of the CGC is shown in Figure 3.3.1. The waveform from the external oscillated by the internal oscillator and divided by the divider is converted into the square waveform for clock. The clock is controlled by the controller and the counter to be sent to the outside the CGC. The following describes the CGC's main components and their functions.

- (1) Clock Generation
- (2) Operation Modes

[1] Clock Generation

The CGC contains an oscillator. By connecting oscillating elements to external pins (XTAL1 and XTAL2), the required clock can be produced easily. The CGC provides the clock whose frequency is 1/2 of the oscillation frequency. Figure 3.3.2 shows an example of crystal connection.

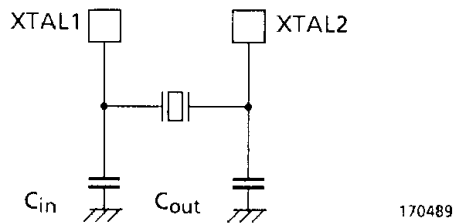


Figure 3.3.2 (a) Example of Crystal Connection

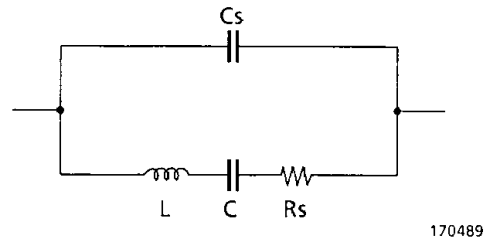


Figure 3.3.2 (b) Oscillator Equivalent Circuit

- (1) For the quartz crystal oscillator, use the MR8000-C20 (oscillation frequency 8 MHz) or MR12000-C20 (oscillation frequency 12 MHz) manufactured by Tokyo Denpa Company Ltd., or the equivalent;

Product No.	Holder Type	Frequency (MHz)	Cin (pF)	Cout (pF)	Quartz Crystal Parameter (Typ.)			Drive Level (mW)	Condition Load Capacitance (pF)
					C ₁ (pF)	C ₀ (pF)	R ₁ (Ω)		
MR8000-C20	HC-49-U	8	22	33	—	4.00	30.0	—	—
MR8000-C14		8	20	20	0.0189	3.87	6.0	0.5	12.67
MR12000-C20		12	33	33	—	4.00	25.0	—	—
MR12000-C14	(TR-49)	12	20	20	0.0190	3.81	6.9	0.5	12.55

310589

Note : The load capacitance in the condition does not include any stray capacitance.

- (2) For the ceramic resonator, use the CSA8.00MT, CST8.00MT (oscillation frequency 8 MHz) or CSA12.00MT, CST12.00MT (oscillation frequency 12 MHz) manufactured by Murata MFG Co., Ltd.

Product No.	Frequency (MHz)	Cin (pF)	Cout (pF)
CSA8.00MT	8	30	30
CST8.00MT	8		
CSA12.00MT	12	30	30
CST12.00MT	12		

190589

Note : The CST8.00MT and CST12.00MT need no outer capacitance.

[2] Operation Modes

The CGC has the capability to control 3 types of operational modes: Run, Idle and Stop. Anyone of them can be selected by MS1, MS2 (Mode Select 1 and 2) pin.

These modes become valid when the MPU executes a HALT instruction. Fetching a HALT instruction, the MPU sets the $\overline{\text{HALT}}$ signal to "0", telling that it has been put in the halt state. After the execution of the HALT instruction, CGC performs the operation of the specified mode. Table 3.3.1 shows the operations of these modes.

Table 3.3.1 CGC Operation Modes

MS1	MS2	Operation Mode	Description
0	0	Idle mode	Only the internal oscillator operates, stopping the supply of clock outside. The clock output (CLK) is held at "0".
0	1	—	Not used.
1	0	Stop mode	All internal operations are stopped. The clock output (CLK) is held at "0".
1	1	Run mode	The supply of clock outside is continued.

120489

The restart from the clock stop state in Idle or Stop mode is performed by reset ($\overline{\text{RESET}}$ signal) or acknowledge of maskable interrupt ($\overline{\text{INT}}$ signal) or non-maskable interrupt ($\overline{\text{NMI}}$ signal).

[3] Warm-up Time for Restart (from Stop mode)

Clearing the halt state by interrupt acknowledge, the MPU begins executing interrupt processing. Therefore, when restarting the clock by the $\overline{\text{NMI}}$ or $\overline{\text{INT}}$ restart signal in the Stop mode, its oscillation must be fully stabilized before being supplied outside. The CGC provides, by means of the internal counter, the warm-up time enough for the clock to reach the stabilize frequency. The warm-up ends at the rising edge of the internal counter output derived from dividing the oscillation frequency to start clock output. The warm-up time is equal to the time derived by dividing the frequency of the externally attached oscillator by 2^{14} .

Figure 3.3.3 shows the block diagram of the internal counter. Table 3.3.2 shows the relationship between the oscillation frequency and the warm-up time.

In the restart by the $\overline{\text{RESET}}$ signal, no warm-up is performed for the quick operation at power-up. Therefore, expand the width of the $\overline{\text{RESET}}$ signal adequately to provide the warm-up time.

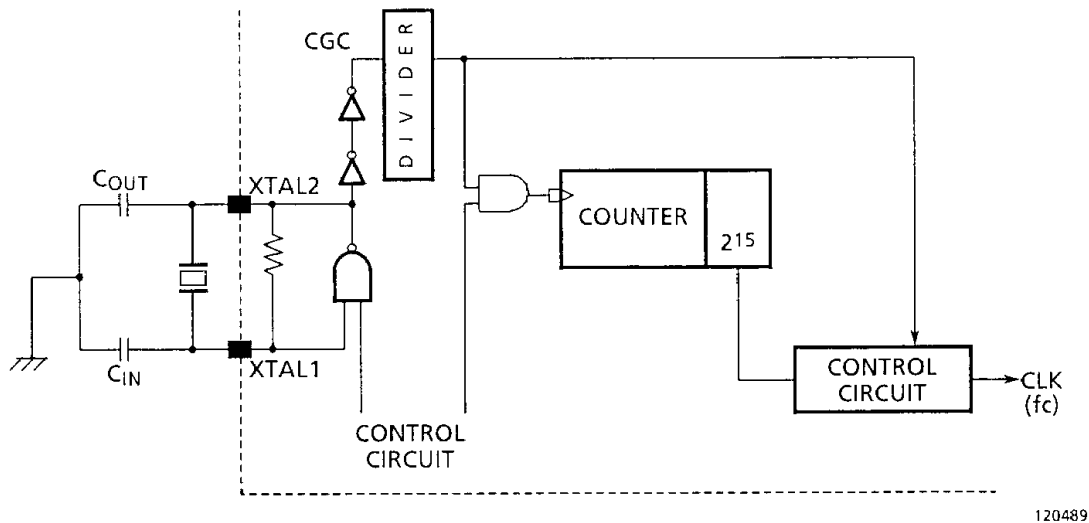


Figure 3.3.3 Block Diagram of Internal Counter

Table 3.3.2 Warm-up Time

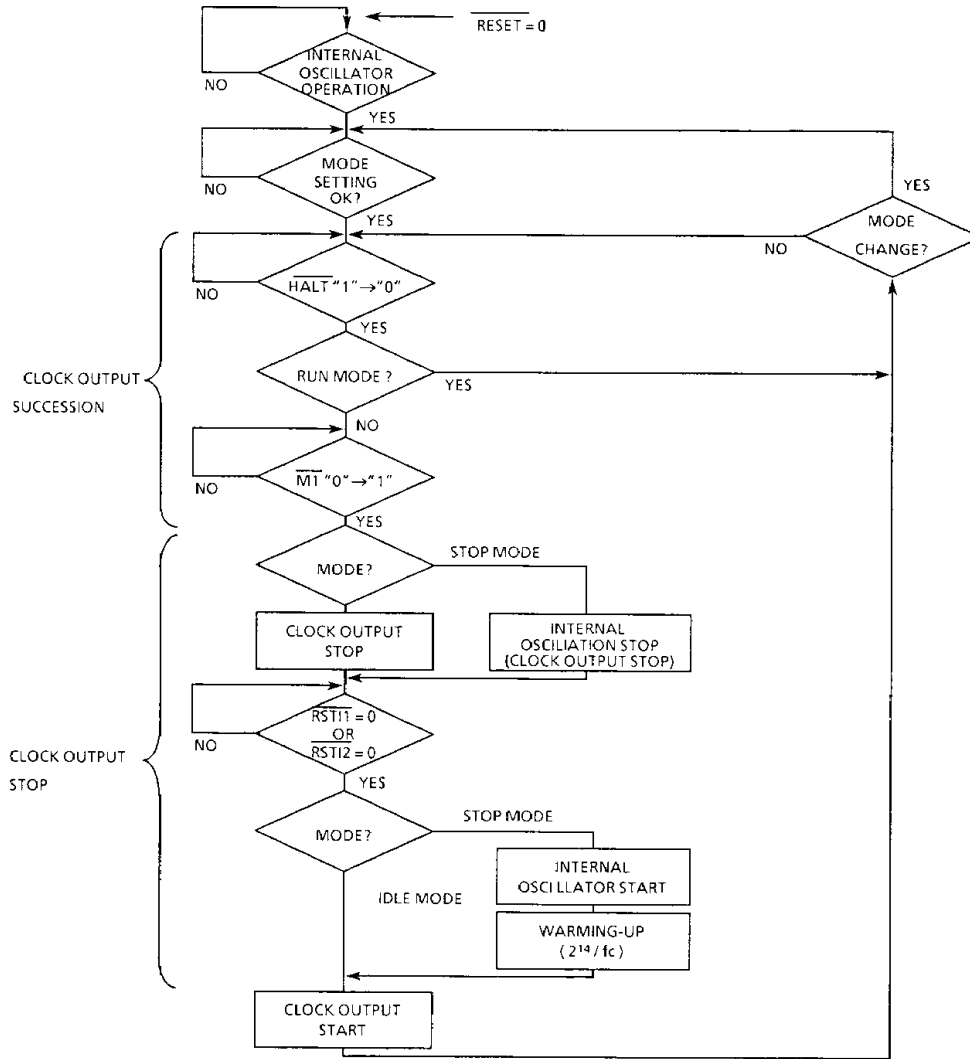
Counter output	Warm-up time		
	215	214/fc	fXTAL = 12MHz
		2.7 ms	4 ms

* fc = fXTAL/2

3.3.3 CGC Status Transition Diagram and Basic Timing

The following describes the status transition and basic timing to be provided when the CGC operates.

[1] Status Transition Diagram



120489

Figure 3.3.4 Status Transition Diagram

[2] Basic Timing

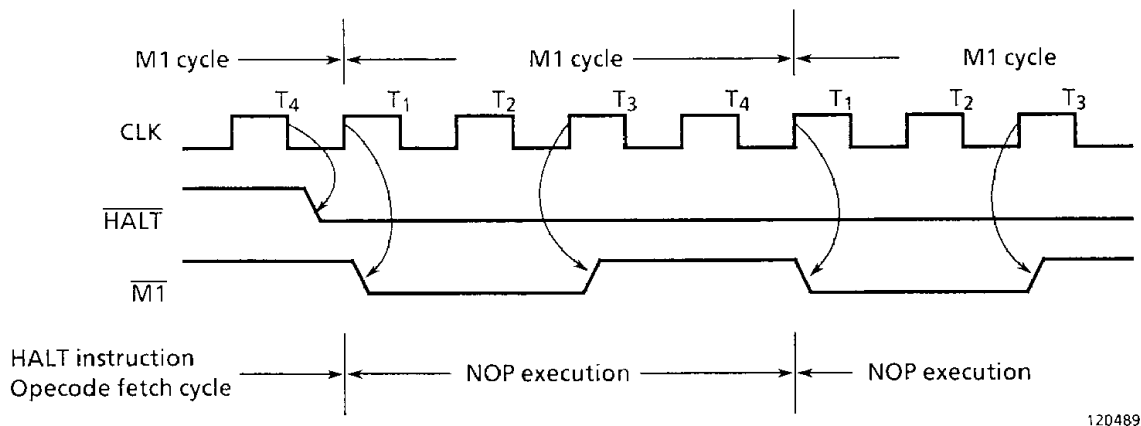
The following describes the CGC basic timing.

(1) Operation at execution of HALT instruction

The following describes the basic timing in each mode to be provided when the MPU executes a HALT instruction. The MPU sets the $\overline{\text{HALT}}$ signal to "0" in synchronization with the falling edge of clock state T4 of the HALT instruction Opcode fetch cycle (M1). This signal tells the CGC that the MPU is going to get in the halt state.

(a) Run mode (MS1 = 1, MS2 = 1)

Figure 3.3.5 shows the basic timing in the Run mode. In the Run mode, the CGC continues supplying the clock to the outside even when the MPU is in the halt state. Therefore, the MPU continues executing NOPs during the halt state. The systems which need memory address refresh use this mode.



120489

Figure 3.3.5 Basic Timing in Run Mode

(b) Idle mode (MS1 = MS2 = 0), and Stop mode (MS1 = 1, MS2 = 0)

Figure 3.3.6 shows the basic timing in the Idle and Stop modes. In these modes, the clock output is stopped with clock state T4 being "0" by the $\overline{\text{HALT}}$ signal and the $\overline{\text{M1}}$ signal which follows the HALT instruction.

However, in the Stop mode, the CGC's internal oscillator also stops.

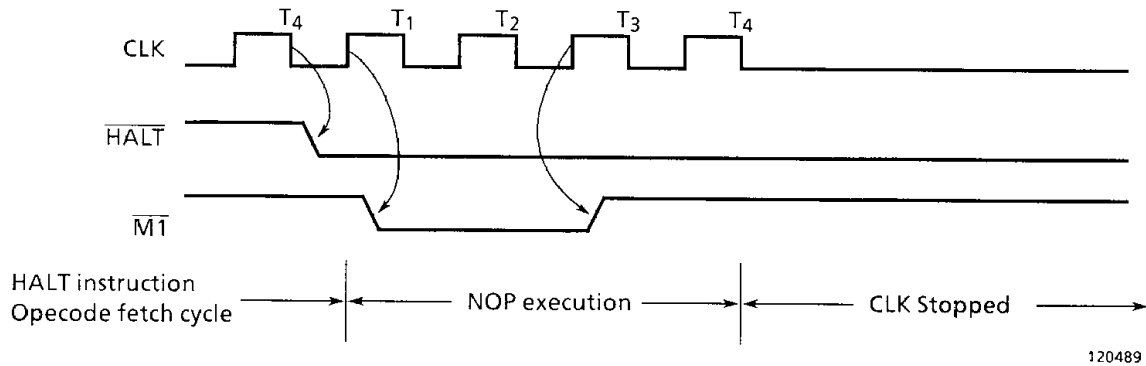


Figure 3.3.6 Basic Timing in Idle and Stop Modes

(2) Clock output restart from each mode

The clock stopped state in the Idle or Stop mode is cleared by setting any of the following signals to “0” (for the system restart operation, see Subsection 3.3.4) :

- \overline{INT} (level trigger input)
- \overline{NMI} (edge trigger input)
- \overline{RESET} (level trigger input)

(a) Clock output restart from Idle mode

Figure 3.3.7 shows the basic timing for the sequence of the output restart from the clock stopped state in the Idle mode. In the restart in the Idle mode, the clock output is restarted in a relatively short delay time because the internal oscillator is operating even in the clock stopped state.

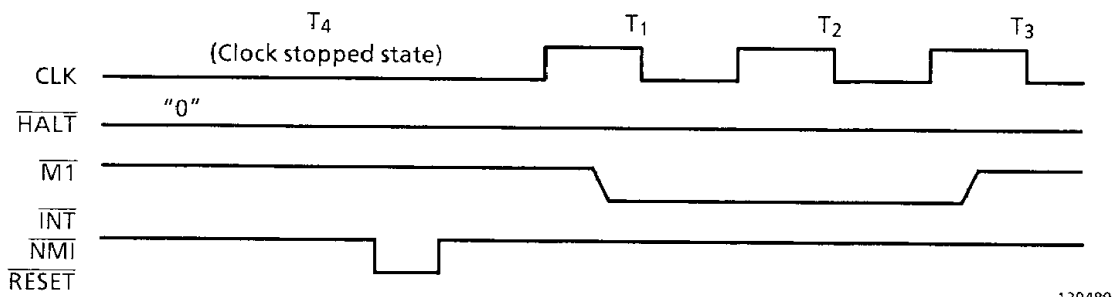
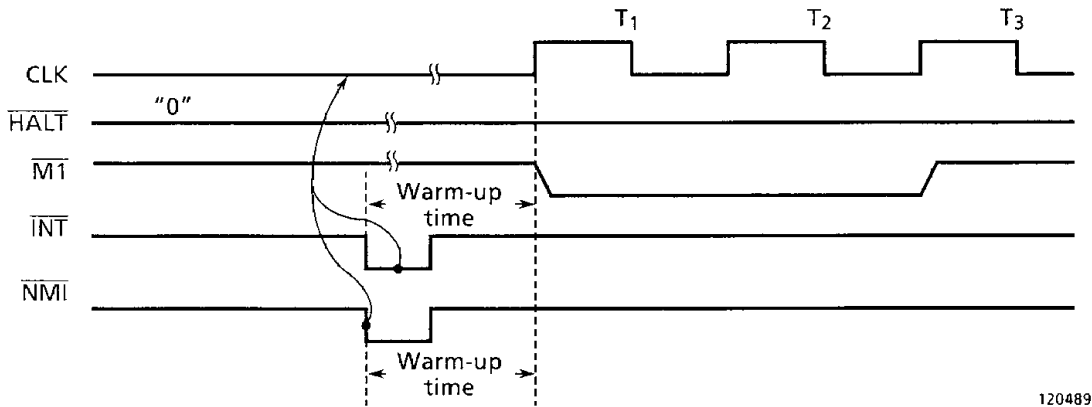


Figure 3.3.7 Basic Timing for Sequence of Restart from Clock stopped State (Idle Mode)

(b) Clock output restart from Stop mode

Figure 3.3.8 shows the basic timing for the sequence of the restart from the clock stopped state in the Stop mode. When restarting by setting the $\overline{\text{INT}}$ or $\overline{\text{NMI}}$ signal to “0”, the warm-up time is automatically created by the internal counter. In the restart by the $\overline{\text{RESET}}$ signal, oscillation restarts without warm-up.



120489

Figure 3.3.8 Basic Timing for Sequence of Restart from Clock Stopped State (Stop Mode)

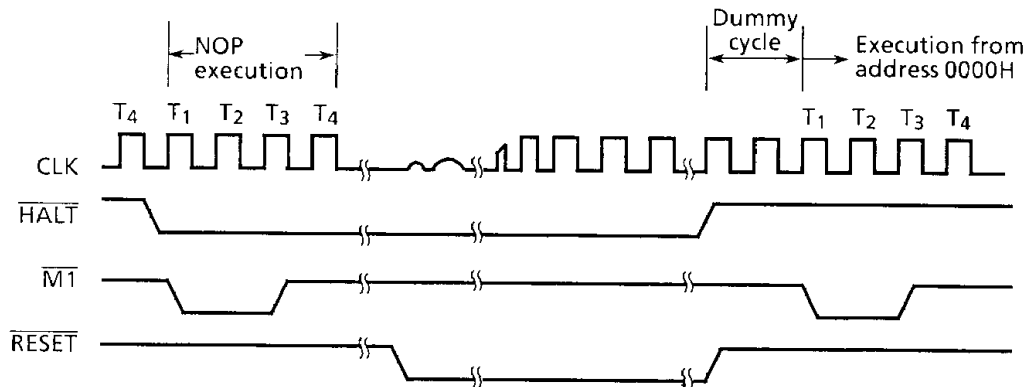
3.3.4 Relationship with MPU

The following describes the relationship between the CGC and the MPU mainly in terms of the halt clear operation.

[1] RESET Signal

Figure 3.3.9 shows an example of the timing for the restart from the Stop mode on the TMPZ84C011A sharing the MPU and CGC $\overline{\text{RESET}}$ signals. To reset the MPU, the $\overline{\text{RESET}}$ signal must be set to “0” for at least 3 stable clocks. When the $\overline{\text{RESET}}$ signal goes “1”, the MPU gets out of the halt state after a dummy cycle of 2 clock states to start executing instructions from address 0000H.

To restart the clock output by the $\overline{\text{RESET}}$ signal in the Stop mode, the internal counter to determine the warm-up time does not operate. Therefore, if the MPU does not restart correctly due to the unstable clock output immediately after the restart of the internal oscillator, or when the stability of the crystal at power-up is considered, the $\overline{\text{RESET}}$ signal must be held at “0” for a time long enough for the MPU to be reset securely.



120489

Figure 3.3.9 Example of Clock Restart Timing by $\overline{\text{RESET}}$ Signal

[2] Clearing Halt State by Interrupt Signal

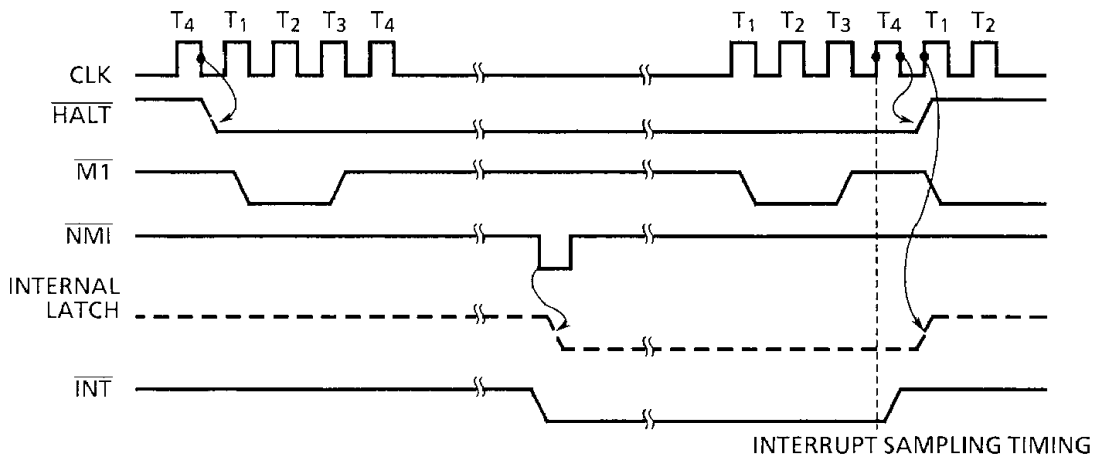
The CGC restarts the clock output from the Idle or Stop mode by the input of $\overline{\text{INT}}$ or $\overline{\text{NMI}}$ signal. By this clock, the MPU starts operating. However, when the CGC restarts the clock output, the MPU is still in the halt state executing NOPs. To clear the halt state, the interrupt signal must be entered into the MPU (in the case of the $\overline{\text{INT}}$ signal, the signal for at least one instruction). The MPU interrupt is detected at the rising edge of the last clock of each instruction (NOP for the halt state).

(1) When using non-maskable interrupt (NMI)

MPU's non-maskable interrupt is triggered by edge. The MPU contains the flip-flop to detect an interrupt. That is, the state of this internal NMI flip-flop is sampled at the rising edge of the last clock of each instruction. Therefore, when a short active low ("0") pulse has been inserted before the interrupt detection timing, the interrupt is acknowledged. The $\overline{\text{NMI}}$ input of the TMPZ84C011A is connected to the $\overline{\text{NMI}}$ input of the MPU via the CGC, performing the same operations as above. (See Figure 3.3.11.)

(2) When using maskable interrupt (INT)

With a maskable interrupt, the maskable interrupt enable flip-flop (IFF) must be set to "1" by program before "0" of the $\overline{\text{INT}}$ input signal is detected. Even if the CGC accepts the $\overline{\text{INT}}$ signal to restart supply the clock, no interrupt is acknowledge unless the $\overline{\text{INT}}$ signal is kept inserted until one instruction (NOP) has been executed. Figure 3.3.10 shows the timing for clearing the halt state by the interrupt signal.

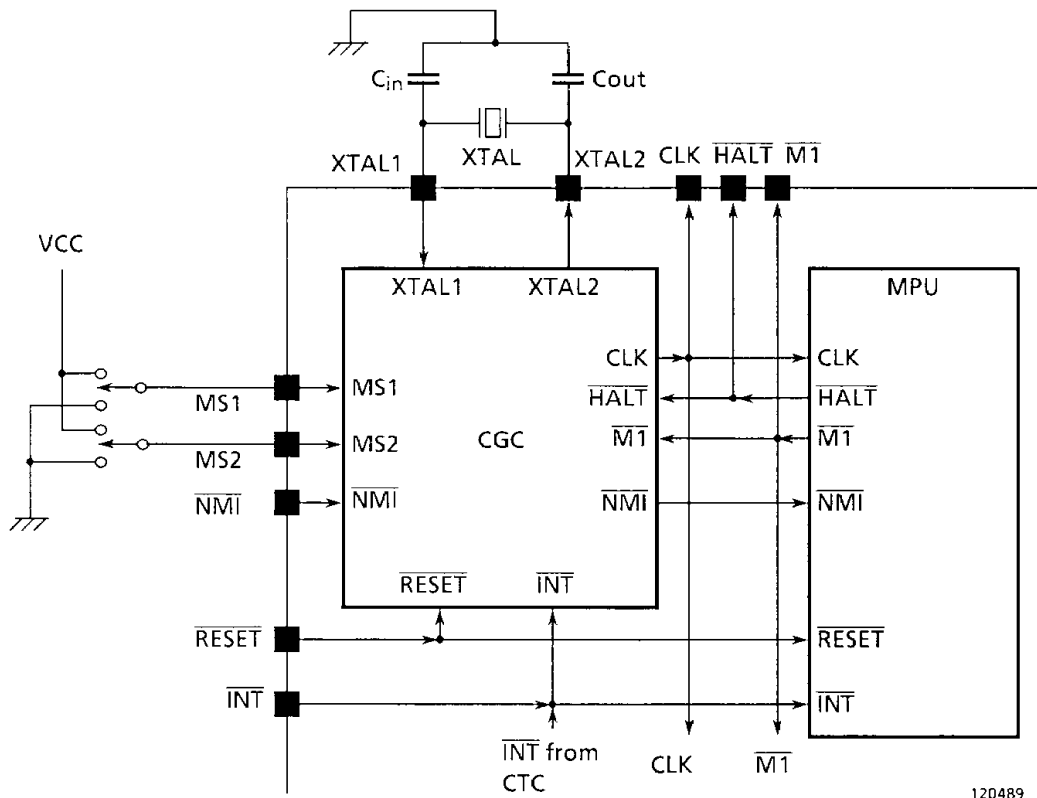


120489

Figure 3.3.10 Timing for Clearing Halt State by Interrupt Signal

[3] Connecting CGC to MPU on TMPZ84C011A

Figure 3.3.11 shows the connection between the CGC and the MPU on the TMPZ84C011A.



120489

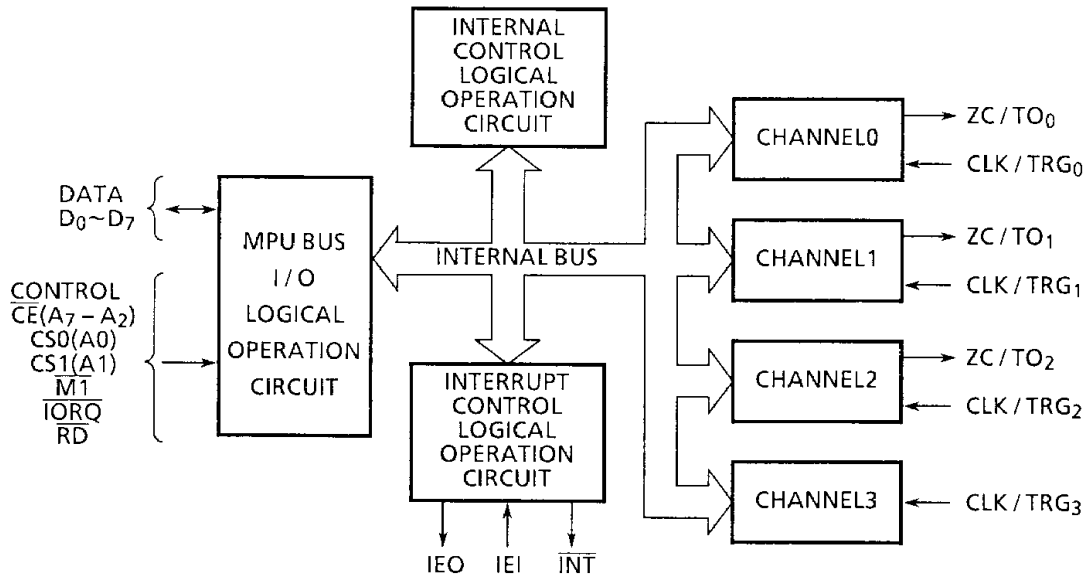
Figure 3.3.11 Connection Between CGC and MPU

3.4 CTC Operational Description

The CTC has 4 independent channels. To these channels, addresses are allocated on the TMPZ84C011A's I/O map, permitting the read/write of the channels in the MPU's I/O cycle. (See Figure 3.4.1.) This subsection mainly describes the operation to be performed after the CTC is accessed.

3.4.1 CTC Block Diagram

Figure 3.4.1 shows the block diagram of the CTC.



120489

Figure 3.4.1 Block Diagram of CTC

3.4.2 CTC System Configuration

The CTC system consists of the following 4 logic circuits:

- (1) MPU bus I/O logic circuit
- (2) Internal control logic circuit
- (3) Interrupt control logic circuit
- (4) Four independent counter/timer channel logic circuits

[1] MPU Bus I/O Logic Circuit

This circuit transfer data between the MPU and the CTC.

[2] Internal Control Logic Circuit

This circuit controls the CTC's operation functions such as the CTC selector chip enable, reset, and read/write circuits.

[3] Interrupt Control Logic Circuit

This circuit performs the MPU interrupt related processing such as priority determination. The priority with other LSIs is determined by the physical location in daisy chain connection.

[4] Counter/Timer Channel Logic Circuit

This circuit consists of the following 2 registers and 2 counters. Figure 3.4.2 shows the configuration of this circuit.

- Time-constant register (8 bits)
- Channel control register (8 bits)
- Down-counter (8 bits)
- Prescaler (8 bits)

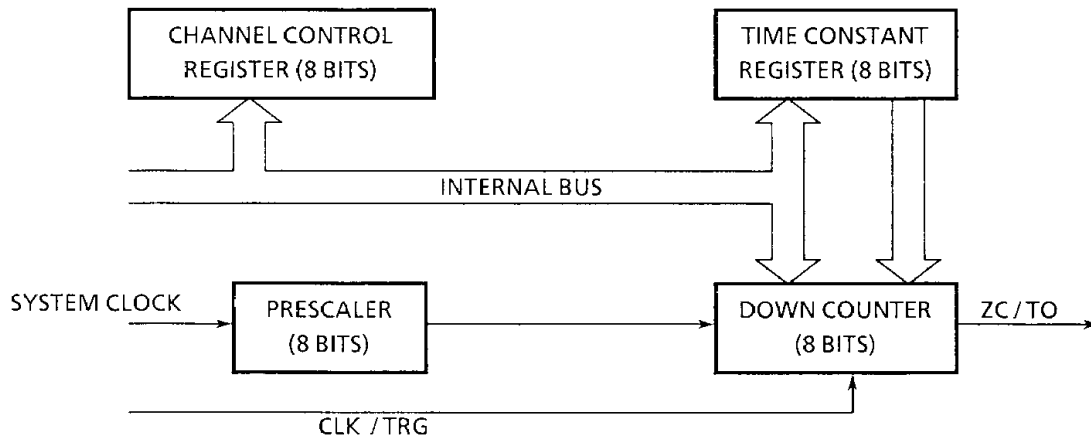


Figure 3.4.2 Configuration of Counter/Timer Channel Logic Circuit

(1) Time-constant register

This register holds the time constant to be written in the down counter. When the CTC is initialized or the down-counter has reached zero, the time constant is loaded into down-counter. The time constant is set immediately after the MPU has written the channel control word to the channel control register. For a time constant, an integer from 1 to 256 can be used.

(2) Channel control register

This register is used to choose the channel mode or condition according to the channel control word sent from the MPU.

(3) Down-counter

The contents of the time-constant register are loaded into the down counter. In the counter mode, these contents are decremented at each edge of the external clock; in the timer mode, they are decremented for each prescaler clock output. The contents of the time-constant register are loaded at initialization or when the down-counter has reached zero.

The contents of the down-counter can be read any time. Also, the system can be programmed so that an interrupt request is generated each time the down-counter has reached zero.

(4) Prescaler

The prescaler, used only in the timer mode, divides the system clock by a factor of 16 or 256. The prescaler factor can be programmed by channel control word. The output of the prescaler becomes the clock input to the down-counter.

3.4.3 CTC Basic Operations

[1] Reset

The state of the CTC is unstable after it is powered on. To initialize the CTC, the low level signal needs to be applied to the $\overline{\text{RESET}}$ pin. On any channel, the channel control word and time-constant data must be written to the channel to be started before it is started in the counter or timer mode. To program the system to enable interrupts, the interrupt vector word must be written to the interrupt controller. When these data have been written to the CTC, it is ready to start.

[2] Interrupt

The CTC can cause an interrupt when the MPU is operating in the mode 2. The CTC interrupt can be programmed for each channel. Each time the channel's down-counter has reached zero, the CTC outputs the interrupt request signal ($\overline{\text{INT}}$). When the MPU accepts the CTC's interrupt request, the CTC outputs the interrupt vector. Based on this interrupt vector, the MPU specifies the start address of the interrupt processing routine and calls it to start interrupt processing.

Because the MPU specifies the start address of the interrupt processing routine by the interrupt vector from the CTC, the user can change the vector value to call any desired address.

The interrupt processing is terminated when the MPU executes an RETI instruction. The CTC has the circuit which decodes the RETI instruction. By constantly monitoring the data bus CTC can know the termination of the interrupt processing.

The interrupt priority with the Z80 peripheral LSIs is determined by the daisy chain connection. That is, the peripheral LSIs are connected in series and physically nearer the MPU, a higher priority is given. Inside the CTC, channel 0 is given the highest priority followed by channels 1, 2 and 3 in this order.

The CTC has the signal lines IEO and IEI. Connect the IEO of a higher peripheral LSI to the IEI of a lower peripheral LSI. Connect the IEI of the highest peripheral LSI to VCC. Leave the IEO of the lowest peripheral LSI unused. In this connection, the CTC interrupt is caused under the following conditions:

- When both IEI and IEO are high, no interrupt is caused. At this time, the $\overline{\text{INT}}$ signal is high. An interrupt can be requested in this state.
- When the CTC outputs the interrupt request signal ($\overline{\text{INT}}$), the IEO of the CTC becomes low. When the MPU accepts the interrupt, the $\overline{\text{INT}}$ pin goes high again.
- When the IEI goes low, the IEO also goes low.
- While the IEI is low, no interrupt can be requested.
- When the IEI goes low while an interrupt is being serviced, the interrupt processing is aborted.

[3] Operation Modes

The CTC operates in either the counter mode or the timer mode. Mode selection is programmed by writing the channel control word.

(1) Counter mode

In the counter mode, the number of edge of the pulses applied to the channel's CLK/TRG pin is counted. When pulses have been input, the contents of the down-counter are decremented in synchronization with the rising edge of the next system clock. The pulse's rising edge or falling edge to be counted can be specified by the channel control word.

When the down-counter has read zero, the high level pulses is output from the ZC/TO pin. When the interrupt is enabled by the channel control word, the $\overline{\text{INT}}$ pin goes low and an interrupt is requested. When the down-counter has reached zero, the time constant data written in the time constant register is automatically loaded into the down-counter. To load a new time constant value into the down-counter, write the data to the time constant register, and it is loaded into the down-counter after the end of the current count operation.

(2) Timer mode

In the timer mode, the time intervals which are integral multiples of the system clock period. A timer interval is measured against the system clock. The system clock is supplied to the prescaler which divides it by a factor of 16 or 256. The output of the prescaler provides the clock to decrement the down-counter by 1. The time constant data is automatically loaded in the down-counter each time it has reached zero as in the counter mode. When the down-counter has reached zero, the high level pulse is output from the ZC/TO pin.

The period of this pulse is given by the following expression:

$$tc \cdot P \cdot TC$$

Where, tc = System clock period

P = Prescaler value (16 or 256)

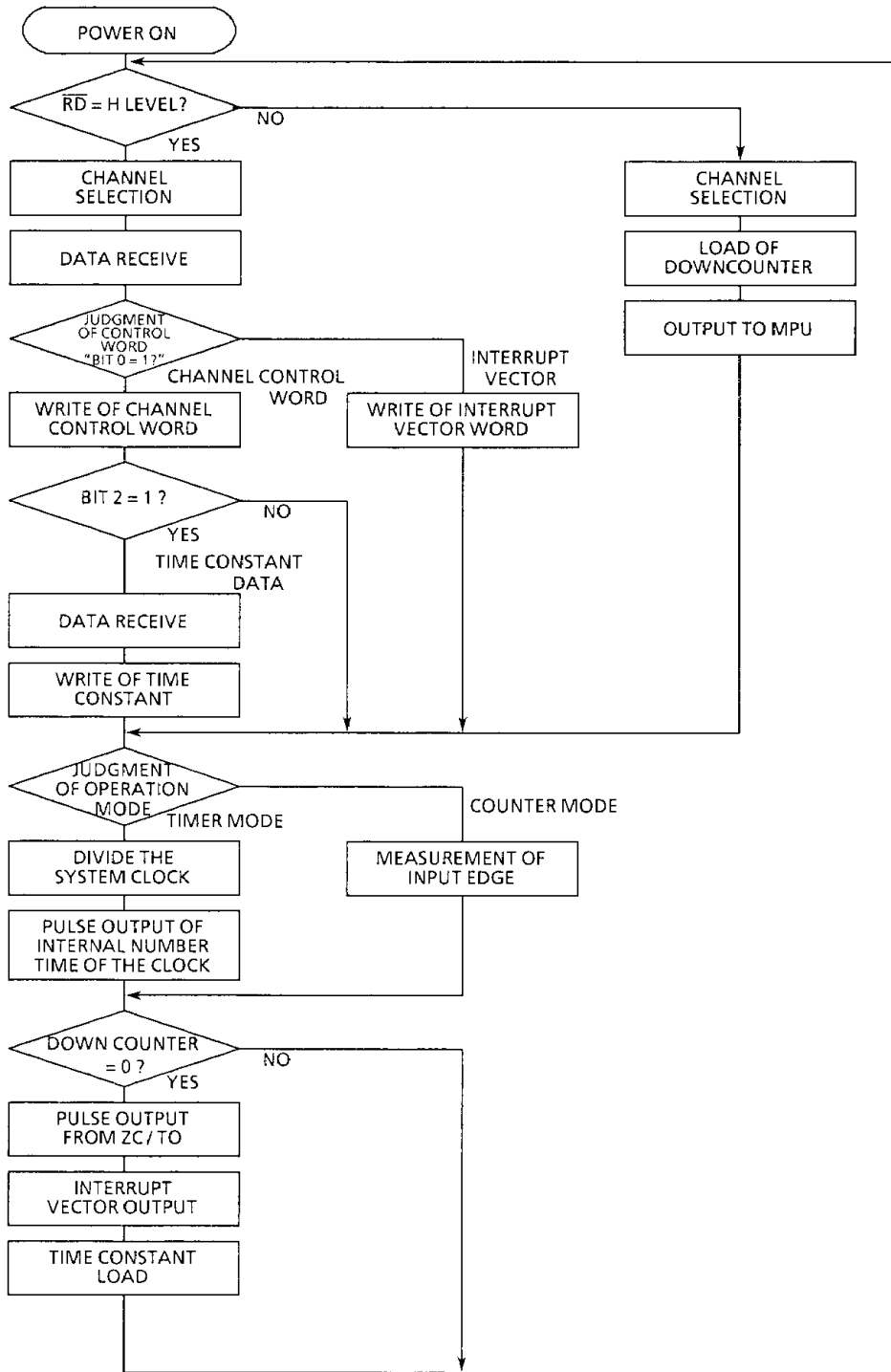
TC = Time constant data (256 for 00H)

The user can specify, by means of the channel control word, whether to start the timer automatically or not and start the timer at the edge of the pulse at CLK/TRG pin or not, and, if the edge is used, start the timer at the rising edge or the falling edge.

3.4.4 CTC Status Transition Diagram and Basic Timing

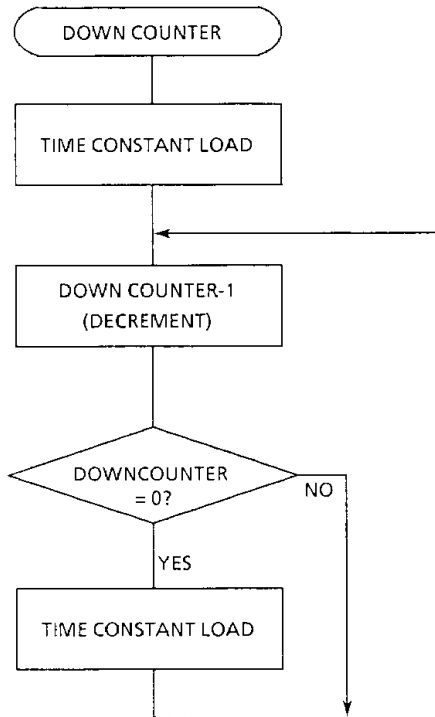
[1] Transition Diagram

Figure 3.4.3 shows the CTC status transition diagram.



120489

Figure 3.4.3 (a) CTC Transition Diagram (a)



120489

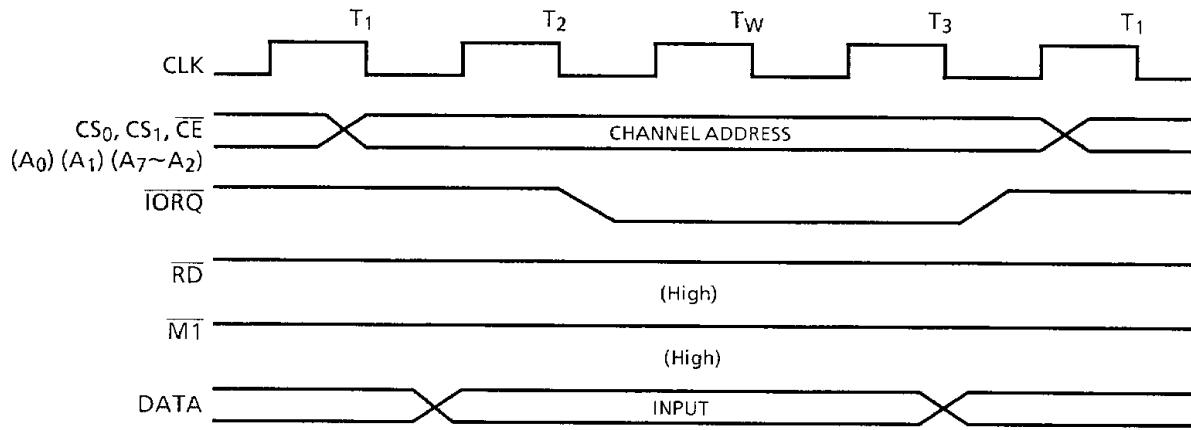
Figure 3.4.3 (b) CTC Transition Diagram (b)

[2] Basic Timing

(1) Write cycle

The write cycle is used to write a channel control word, an interrupt vector, or a time constant. The MPU makes low the $\overline{\text{IORQ}}$ pin of the CTC in the subsequent system clock cycle T2 to start the write cycle. It is required to make high the M1 pin of the CTC to indicate that the write cycle is on.

At the start of the cycle, the channel is specified by CS₁ (A₁) and CS₀ (A₀) of the CTC. Thus, the CTC's internal registers are ready to acquire data in system clock T3. Tw is the wait state to be automatically inserted by the MPU.

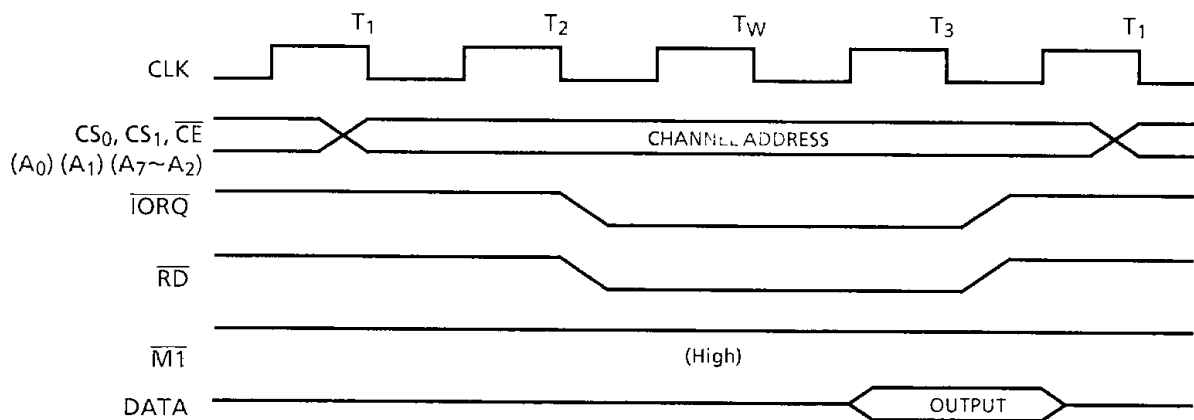


120489

Figure 3.4.4 Write Timing

(2) Read cycle

The read cycle is used to read the contents of the down-counter. In system clock cycle T2, the MPU makes low the \overline{RD} and \overline{IORQ} pins of the CTC to start the read cycle. It is required to make high the \overline{MI} pin to indicate that the read cycle is on. At the start of the read cycle, the channel is specified by CS_1 (A_1) and CS_0 (A_0) of the CTC. On the rising edge of system clock T_W , the contents of the down-counter at the time of the rising edge of T2 are put on the data bus. T_W is the wait state to be automatically inserted by the MPU.



120489

Figure 3.4.5 Read Timing

[3] Counter mode

In the counter mode, the down-counter is decremented in synchronization with the system clock, at the edge of the pulse applied from the external circuit connected to the CLK/TRG pin. The period of the pulse to be applied to the CLK/TRG pin must be 2 times greater than the system clock period. Also, it is required to insert the setup time between the active edge of the CLK/TRG pin signal and the rising edge of the succeeding system clock. When the interval between these pulses is short, the down-counter is decremented one system clock later. When the down-counter has reached zero, a high level pulse is output from the ZC/TO pin.

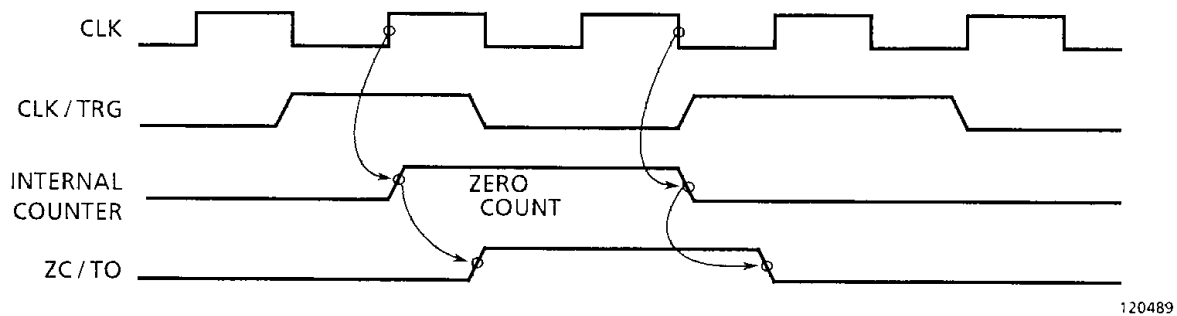


Figure 3.4.6 Counter Mode Timing

[4] Timer mode

The timer starts operating at the second rising edge of the system clock from the rising edge of the pulse applied from the external circuit connected to the CLK/TRG pin. The period of the pulse to be applied to the CLK/TRG pin must be greater than 2 times the system clock period. Also, it is required to insert the setup time between the active edge of the CLK/TRG pin signal and the rising edge of the succeeding system clock. When the interval between these pulses is short, the timer starts one system clock cycle later.

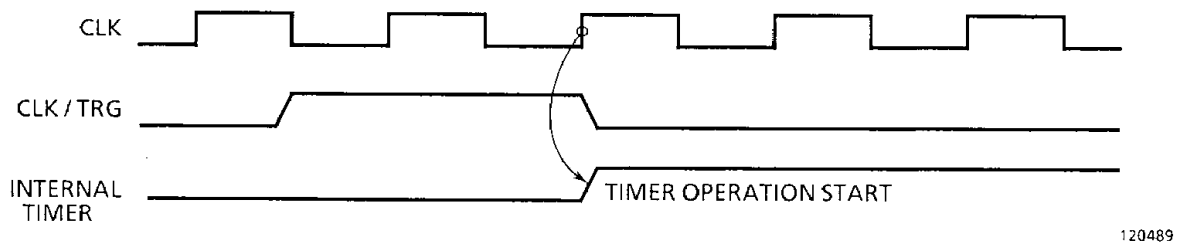


Figure 3.4.7 Timer Mode Timing

[5] Interrupt acknowledge cycle

Having received the interrupt request signal (\overline{INT}) from the CTC, the MPU makes low the CTC's \overline{MI} pin and \overline{IORQ} pin to provide the acknowledge signal. The \overline{IORQ} pin goes low 2.5 system clocks later than the \overline{MI} pin. To stabilize the signal lines (IEI and IEO) in daisy chain connection, the interrupt request cannot be changed on each channel while the \overline{MI} pin is low. The \overline{RD} pin is held high to make distinction between the instruction fetch cycle and the interrupt acknowledge cycle. While the \overline{RD} pin is high, the CTC's interrupt control circuit determines the interrupt-requesting channel with the highest priority. When the CTC's IEI is high and the \overline{MI} pin and \overline{IORQ} pin go low, the interrupt vector is output from the interrupt requesting channel with the highest priority on the data bus. At this time, 2 system clock cycles are automatically inserted by the MPU as a wait state to maintain the stabilization of the daisy chain connection.

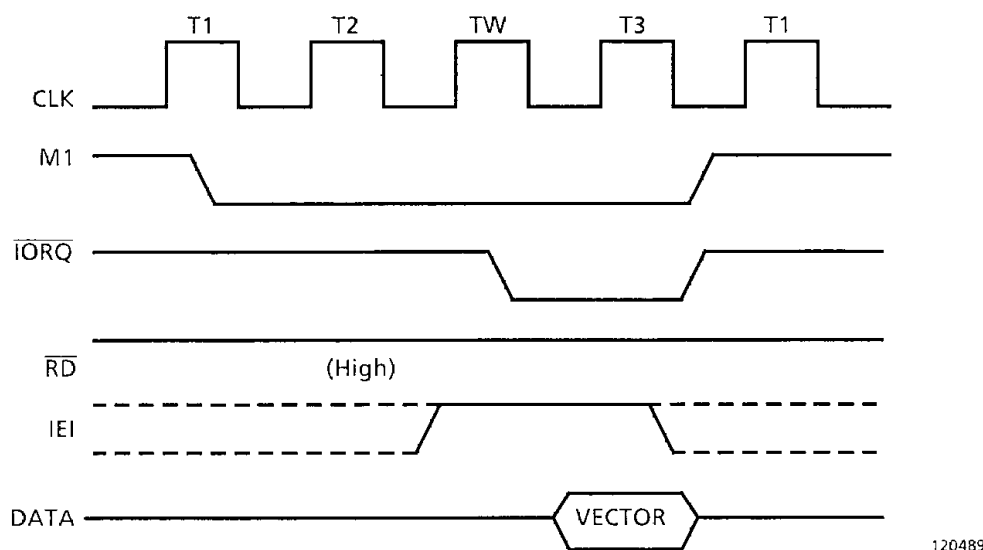


Figure 3.4.8 Interrupt Acknowledge Timing

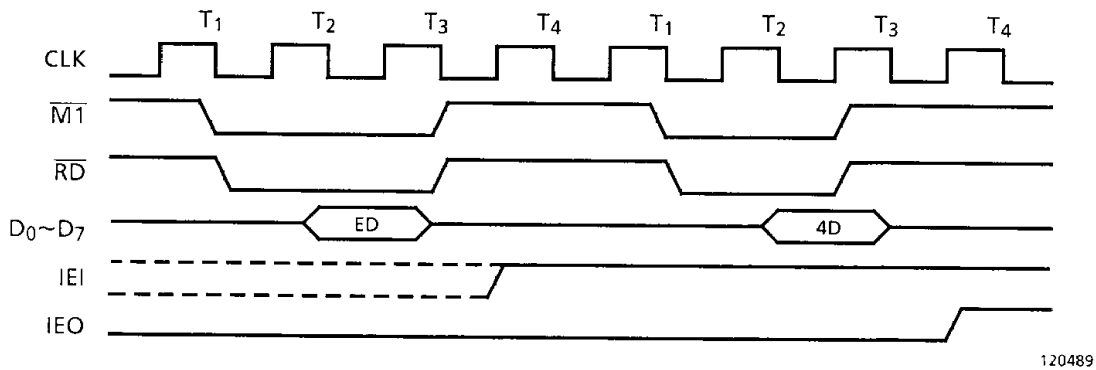
[6] Return from interrupt processing

Return from the interrupt processing is performed by MPU's executing the RETI instruction. This RETI instruction must be used at the end of the interrupt processing routine. When this instruction is executed by the MPU, the CTC's IEI and IEO return to the state active before the interrupt has been serviced.

The RETI instruction is a 2-byte instruction. Its code is EDH 4DH. The CTC decodes this instruction to check if there is the next interrupt request channel.

In the diasy chain structure, the interrupting LSI's IEI and IEO are held high and low respectively at the time the instruction code EDH has been decoded.

The code following EDH is 4DH, only the peripheral LSI which has sent the interrupt vector (that is, the LSI whose IEI is high and IEO is low) returns from the interrupt processing. This restarts the processing of the pending interrupt of the next higher peripheral LSI.



Figureure 3.4.9 Interrupt Return Timing

120489

3.4.5 CTC Operational Procedure

To operate the CTC in the counter mode or the timer mode, the channel control word and time-constant data must be written to the CTC. To enable interrupts by the channel control word, the interrupt vector must be written to the CTC.

[1] I/O Address and Channel Control Word

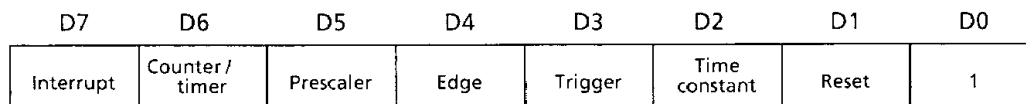
To write the channel control word to the CTC, the channel must be specified by the corresponding channel I/O address. Table 3.4.1 shows the channel I/O addresses.

Table 3.4.1 Channel I/O Addresses

Channel	I/O address
0	#10
1	#11
2	#12
3	#13

120489

The channel control word to be written to the CTC consists of 8 bits. The system data bus bit D0 through D7 correspond to bit 0 through 7 respectively. Figure 3.4.10 shows the meaning of each bit. Table 3.4.2 shows the function of each bit.



120489

Figure 3.4.10 Channel Control Word

For the channel control word, D0 must always be 1.

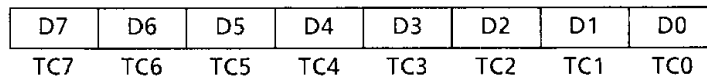
(2/2)

Bit	Meaning and function	
	0	1
Bit 2 (D2)	Indicates that the channel control word is not followed by the writing of time constant. When the channel is in the reset state, this bit cannot be set to "0" in the first channel control word. To change states with the time constant unchanged, the channel control word with this bit set to "0" must be entered.	Indicates that the channel control word is followed by the writing of time constant. If the time constant is written during down-counter operation, the new time constant is set to the time constant register with proceeding the current counting. When the down-counter has reached zero, the new time constant is loaded into the down-counter.
Bit 1 (D1)	Continues the current channel operation.	Stops the down-counter operation. When this bit is set to "1", the channel operation stops but the channel control register bits remain unchanged. When bit 2 = "1" and bit 1 = "1", the channel remains stopped until a new time constant is written. Channel restart is set up after the new time constant is programmed. The channel is restarted according to the state of bit 3. When bit 2 = "0" and bit 1 = "1", the channel operation does not start until a new channel control word is written.

120489

[2] Time-Constant Data

In either the time mode or the counter mode, the time-constant data must be loaded into the time constant register. When bit 2 (D2) of the channel control word is "1", the time constant is loaded into the time constant register immediately after the channel control word is written. A time-constant value must be an integer in a range of 1 to 256. When the 8 bits of a time constant are all "0"s, such a time constant is assumed to be 256. Figure 3.4.11 shows the bit configuration of time-constant data.



120489

Figure 3.4.11 Time-Constant Data

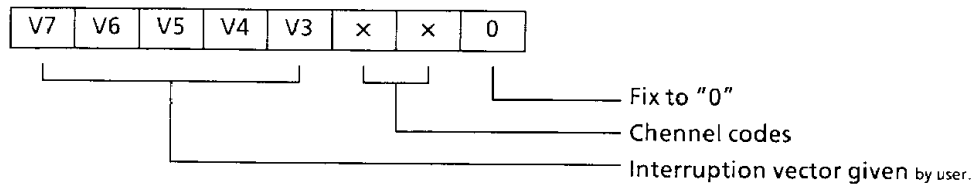
[3] Interrupt Vector

In the MPU mode-2 interrupt, the interrupting channel must give an interrupt vector to the MPU. An interrupt vector is written to the channel-0 interrupt vector register with bit 0 (D0) = "0". The vector is written in the same way as the channel control word is written to channel 0. However, bit 0 (D0) of the vector should always be "0". Bit 7 (D7) through bit 3 (D3) are user-defined values. Bit 2 (D2) and bit 1 (D1) are automatically given and contain the code of the interrupt- requesting channel having the highest priority. Table 3.4.3 shows the channel codes. Figure 3.4.12 shows the interrupt vector bit configuration.

Table 3.4.3 Channel Codes

Bit 2(D2)	Bit 1(D1)	Channel number	Interrupt priority
0	0	0	(High)
0	1	1	↑ ↓ (low)
1	0	2	
1	1	3	

120489



120489

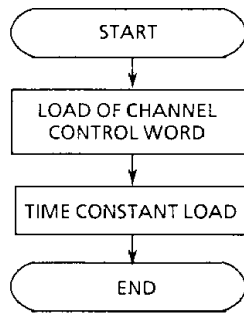
Figure 3.4.12 Interrupt Vector

3.4.6 Using CTC

[1] Counter Mode

The following describes how to use the CTC by referring to a program using channel 0 with interrupt disabled.

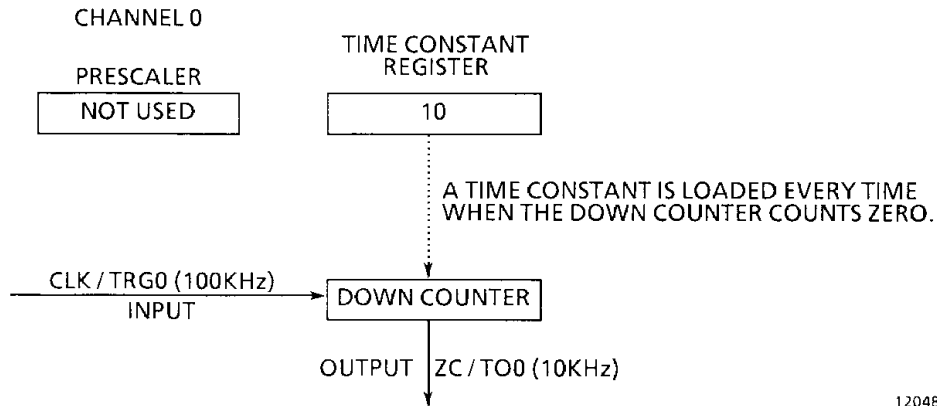
- (a) The counter programming procedure is shown in Figure 3.4.13.



120489

Figure 3.4.13 Counter Programming Procedure

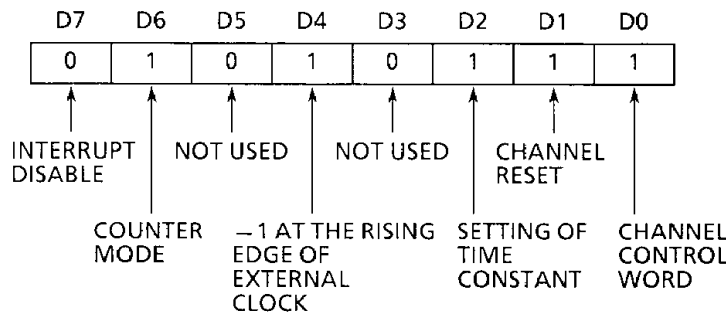
- (b) The block diagram for converting the 100 kHz system clock into the 10 kHz equivalent is shown in Figure 3.4.14.



120489

Figure 3.4.14 Down-Counter Block Diagram

- (c) The channel control word configuration is shown in Figure 3.4.15.

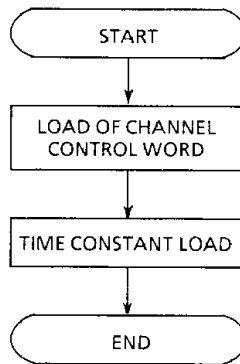


120489

Figure 3.4.15 Channel Control Word Configuration

[2] Timer Mode

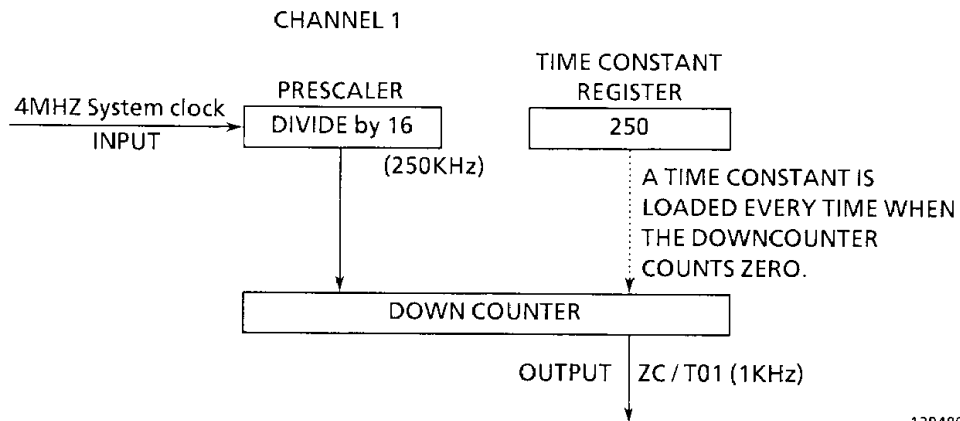
- (a) The timer programming procedure without using interrupt is shown in Figure 3.4.16.



120489

Figure 3.4.16 Timer Programming Procedure

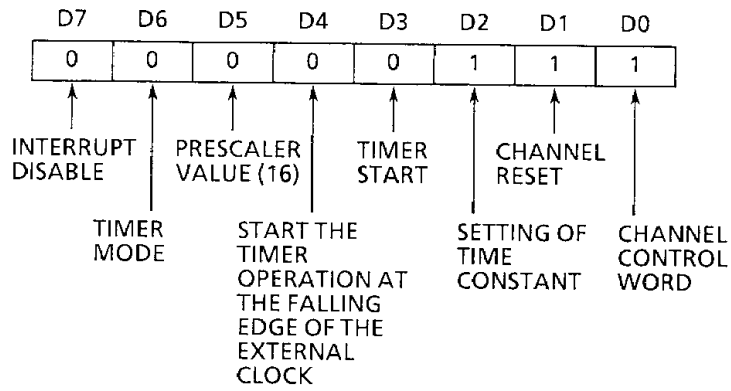
- (b) The block diagram for converting the 4 MHz system clock into the 1 kHz equivalent is shown in Figure 3.4.17.



120489

Figure 3.4.17 Timer Block Diagram

(c) The channel control word is shown in Figure 3.4.18.



120489

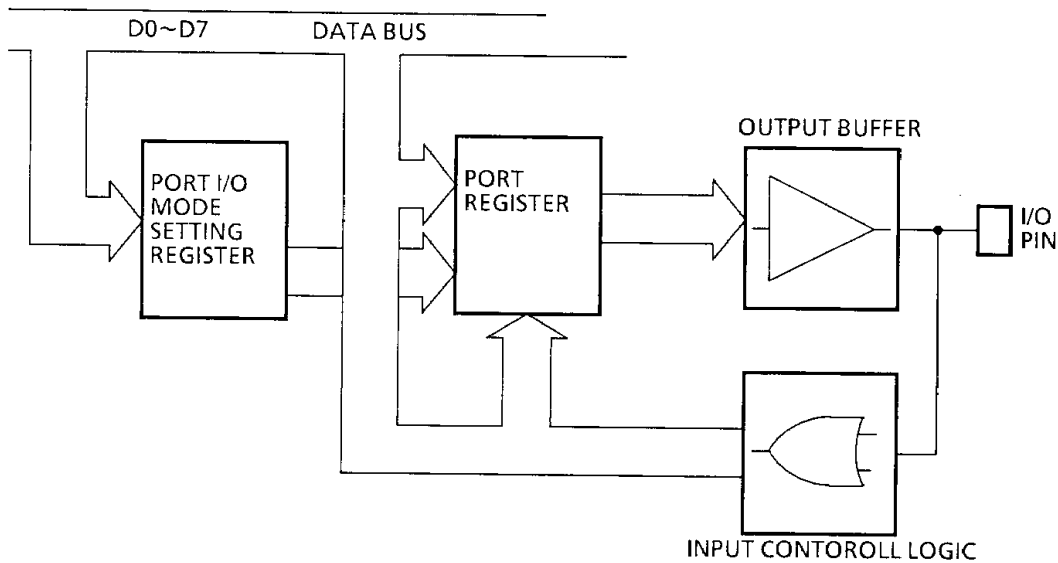
Figure 3.4.18 Channel Control Word Configuration

3.5 Description of I/O Operation

This subsection describes the system configuration and operations of the input/output (I/O) section of the TMPZ84C011A. The I/O consists of five 8-bit general-purpose ports (a total of 40 I/Os). Each bit can be programmed for an input or output port.

3.5.1 I/O Block Diagram

Figure 3.5.1 shows the I/O block diagram.



120489

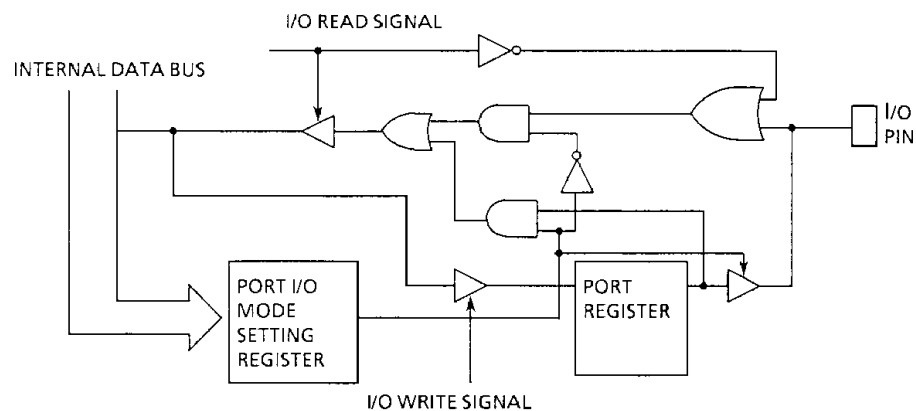
Figure 3.5.1 I/O Block Diagram

3.5.2 I/O System Configuration

The I/O system consists of the I/O setting registers, the port registers, the output buffers, and the input control logics.

The I/O setting registers specify whether the port to be used is to operate as an input port or an output port. Each register corresponds to each port bit. The port registers latch the output data to the port pin. The latched output data can also be taken by program into the TMPZ84C011A. These registers and control logics are controlled by the program-controlled $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, and I/O address signals.

Figure 3.5.2 shows the logic diagram of the I/O port.



- Notes:
- (1) At reset
 - I/O register = "0" ... Input mode
 - Port register = "0"
 - (2) I/O read ... Enabled by CPU's I/O input instruction.
 - (3) I/O write ... Enabled by CPU's I/O output instruction.

120489

Figure 3.5.2 I/O Port Logic Diagram

3.5.3 I/O Port Basic Operations

The I/O setting registers and port registers of the 5 port of the TMPZ84C011A are assigned with the addresses listed in Table 3.5.1 and are enabled by the MPU's I/O instructions.

Table 3.5.1 I/O Addresses of I/O Port

Register	I/O address
PA I/O setting register	#54
PB I/O setting register	#55
PC I/O setting register	#56
PD I/O setting register	#34
PE I/O setting register	#44
PA port register	#50
PB port register	#51
PC port register	#52
PD port register	#30
PE port register	#40

120489

[1] I/O Setting Registers

The I/O setting registers specify whether an I/O port is to operate as an input port or an output port. As listed in Table 3.5.2, when the register content is "0", the specified port is an input port; when it is "1", the specified port is an output port.

Table 3.5.2 I/O Setting Register Contents and Port Types

I/O setting register contents	Port type
0 (Note)	Input port
1	Output port

Note: At reset

120489

[2] Port Registers / I/O Buffer

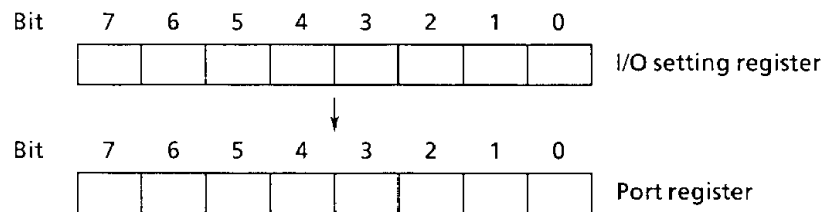
The port registers latch the data put on the data bus from the MPU and output it to the port pin. They hold the data until it is rewritten.

When using a port register for input, the data can be rewritten by the MPU's I/O output instruction without directly affecting the register operation. When the type of a port registers are switched from input to output during operation, the contents of the port registers are output to the port pins immediately.

When reading the port output data during an output operation, the outputs of the port registers in front of the output buffer are read, if the port pin's load of the port pin gets heavy, making it possible to read the correct data when the output level goes low. When a port is used for input, it becomes the non-latch type in which the input port state of time when the MPU's I/O input instruction has been executed is read. At reset, the port register contents are cleared to "0".

[3] I/O Setting Registers vs. Port Registers

As shown in Figure 3.5.3, the I/O setting registers correspond to the port register one by one.



120489

Figure 3.5.3 I/O Setting Registers vs. Port registers

Therefore, the partial change of port bits requires to rewrite contents of the I/O setting registers of the port containing the change bit.

[4] Precautions

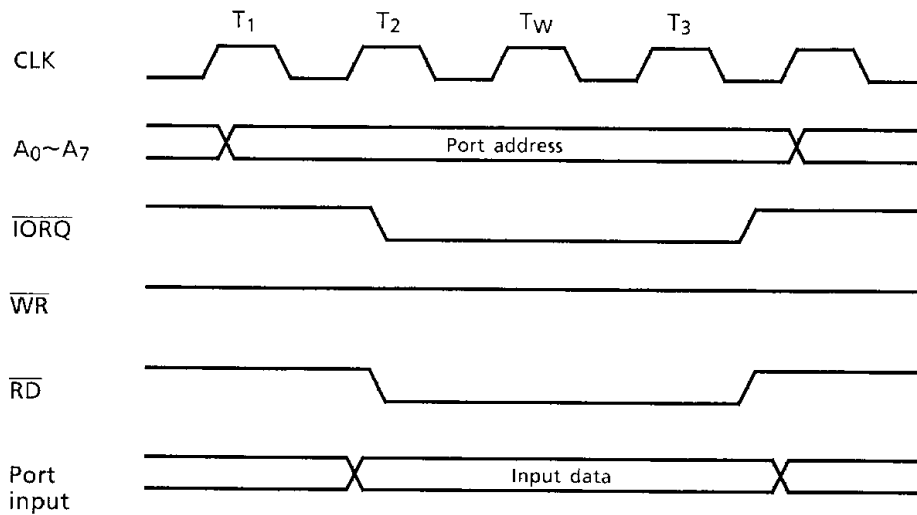
The following points must be taken into account in using the I/O ports:

- (1) When the port types are changed from input to output, the old contents of the port register are output. To avoid this, the contents of the port register must be rewritten with the data to be output before changing the port types. Then, when the port types are changed from input to output, the desired data are output to the port pins, preventing the operational errors of the peripheral devices connected to the pin.
- (2) After reset, all ports are in the input mode and the port register contents are cleared to "0". Therefore, port type change must not be performed immediately. Instead, the content of the port register corresponding the bit to be used for an output port must be replaced with the desired data by the MPU's I/O output instruction before changing the port types to the output port. This streamlines the port operation.

3.5.4 I/O System Basic Timing

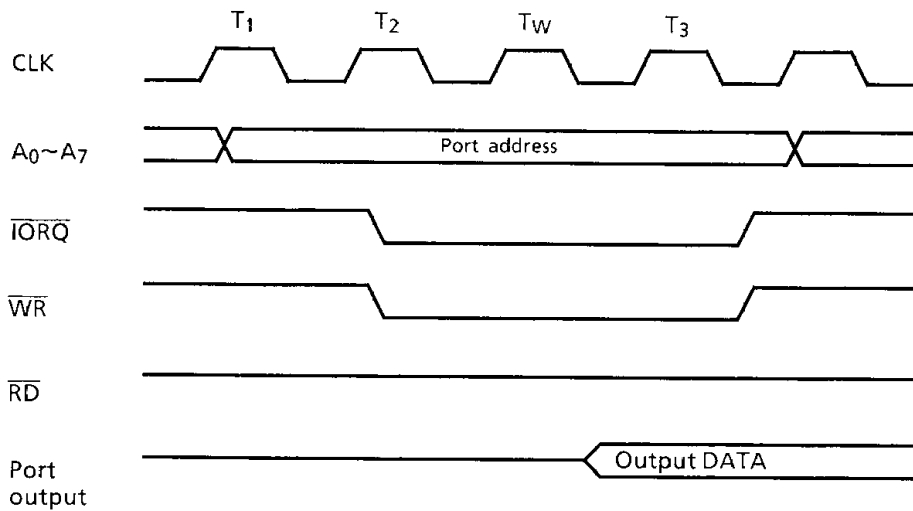
Figure 3.5.4 and 3.5.5 show the basic timings of the I/O ports. The I/O ports of the TMPZ84C011A perform I/O operations according to the MPU's I/O instructions. Therefore, an I/O port address is specified by the low-order 8 bits of the address bus for I/O operations.

In the I/O operation timing, a wait state (T_W) is automatically inserted after clock state T_2 like the operation of the TLCS-Z80 MPU.



120489

Figure 3.5.4 Input Cycle Timing Chart



120489

Figure 3.5.5 Output Cycle Timing Chart

4. ELECTRICAL CHARACTERISTICS

4.1 MAXIMUM RATINGS

SYMBOL	ITEM	RATING
VCC	Vcc Supply Voltage with respect to Vss	- 0.5V to 7V
VIN	Input Voltage	- 0.5V to VCC + 0.5V
PD	Power Dissipation (TA = 85°C)	250mW
TSOLDER	Soldering Temperature (Soldering Time 10 sec)	260°C
TSTG	Storage Temperature	- 65°C to 150°C
TOPR	Operating Temperature	- 40°C to 85°C

120489

4.2 DC ELECTRICAL CHARACTERISTICS (1)

TA = -40°C ~ +85°C, Vcc = 5V ± 10%, Vss = 0V

SYMBOL	PARAMETER	TEST CONDITION	MIN.	TYP.	MAX.	UNIT.	
VIL	Input Low Voltage (except RESET, XTAL1)		-0.5	-	0.8	V	
VIH	Input High Voltage (except RESET, XTAL1)		2.2	-	VCC	V	
VILR	Input Low Voltage (RESET)		-0.5	-	0.45	V	
VIHR	Input High Voltage (RESET)		VCC - 0.6	-	VCC	V	
VOLC	Output Low Voltage (CLK)	IOL = 2.0mA	-	-	0.6	V	
VOHC	Output High Voltage (CLK)	IOH = -2.0mA	VCC - 0.6	-	-	V	
VOL	Output Low Voltage	IOL = 2.0mA	-	-	0.4	V	
VOH1	Output High Voltage 1	IOL = -1.6mA	2.4	-	-	V	
VOH2	Output High Voltage 2	IOH = -250µA	VCC - 0.8	-	-	V	
ILI	Input Leakage Current	Vss ≤ VIN ≤ Vcc	-	-	± 10	µA	
ILO	3-state Output Leakage Current in Float	Vss + 0.4 ≤ Vout ≤ VCC	-	-	± 10	µA	
ICC1	Power Supply Current	Vcc = 5V, fCLK = (1) VIH = VCC - 0.2V VIL = 0.2V	4MHz	-	15	20	mA
			6MHz	-	22	30	
ICC2	Stand-by supply current	Vcc = 5V, (2) VIH = VCC - 0.2V VIL = 0.2V	4MHz / 6MHz	-	0.5	50	µA
ICC3	Power Supply Current (IDLE MODE)	Vcc = 5V, fCLK = (1) VIH = VCC - 0.2V VIL = 0.2V	4MHz	-	1.0	2.5	mA
			6MHz	-	1.5	4	

120489

Note : (1) fCLK = 1/TCC (MIN)

(2) ICC2 Stand-by Supply Current is guaranteed only when the supplied clock is stopped at a low level during T4 state of the following machine Cycle (M1) next to OP code fetch Cycle of HALT instruction.

4.2 DC ELECTRICAL CHARACTERISTICS (2)

$T_A = -40^\circ\text{C} \sim +85^\circ\text{C}$, $V_{CC} = 2.7\text{V} \pm 5.5\text{V}$, $V_{SS} = 0\text{V}$ (Low Voltage operation)

SYMBOL	PARAMETER	TEST CONDITION	MIN.	TYP.	MAX.	UNIT
VIL	Input Low Voltage (except RESET)		-0.3	-	0.2	V
VIH	Input High Voltage (except RESET)		$V_{CC} - 0.2$	-	$V_{CC} + 0.3$	V
VOL	Output Low Voltage	$I_{OL} = 1.2\text{mA}$	-	-	0.4	V
VOH	Output High Voltage	$I_{OL} = -0.7\text{mA}$	2.2	-	-	V
ILI	Input Leakage Current	$V_{SS} \leq V_{IN} \leq V_{CC}$	-	-	± 2	μA
ILO	3-state Output Leakage Current in Float	$V_{SS} \leq V_{out} \leq V_{CC}$	-	-	± 2	μA
ICC1	Power Supply Current	$V_{CC} = 3\text{V}$, $f_{CLK} = 2\text{MHz}$ $V_{IH} = V_{CC}$ $V_{IL} = 0.0\text{V}$	-	-	10	mA
ICC2	Stand-by supply current	$V_{CC} = 3\text{V}$, (2) $V_{IH} = V_{CC}$ $V_{IL} = 0.0\text{V}$	-	-	10	μA
ICC3	Power Supply Current (IDLE MODE)	$V_{CC} = 3\text{V}$, $f_{CLK} = 2\text{MHz}$ $V_{IH} = V_{CC}$ $V_{IL} = 0.0\text{V}$	-	1.0	2.5	mA

310589

4.3 AC ELECTRICAL CHARACTERISTICS (1) (IN ACTIVE STATE)

$T_A = -40^\circ\text{C} \sim +85^\circ\text{C}$, $V_{CC} = 5\text{V} \pm 10\%$, $V_{SS} = 0\text{V}$

4.3.1 AC Characteristics of MPU (in Active State)

(1/3)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 $V_{CC} = 2.7\text{V} \sim 5.5\text{V}$ (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
1	TcC	Clock Cycle Time	500	-	DC		165		ns
2	TwCh	Clock Pulse Width (High)	220	-	DC	-	70	-	ns
3	TwCl	Clock Pulse Width (Low)	220	-	DC	-	70	-	ns
4	TfC	Clock Fall Time	-	-	30	-	12	-	ns
5	TrC	Clock Rise Time	-	-	30	-	12	-	ns
6	TdCr (A)	Clock \uparrow to Address Valid Delay	-	-	180	-	-	90	ns
7	TdA (MREQf)	Address Valid to $\overline{\text{MREQ}} \downarrow$ Delay	180	-	-	-	32	-	ns
8	TdCf (MREQf)	Clock \downarrow to $\overline{\text{MREQ}} \downarrow$ Delay	-	-	150	-	-	70	ns
9	TdCr (MREQr)	Clock \uparrow to $\overline{\text{MREQ}} \uparrow$ Delay	-	-	180	-	-	70	ns
10	TwMREQh	$\overline{\text{MREQ}}$ Pulse Width (High)	190	-	-	-	62	-	ns
11	TwMREQl	$\overline{\text{MREQ}}$ Pulse Width (Low)	460	-	-	-	135	-	ns
12	TdCf (MREQr)	Clock \downarrow to $\overline{\text{MREQ}} \uparrow$ Delay	-	-	190	-	-	70	ns

290589

(2/3)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
13	TdCf (RDf)	Clock ↓ to \overline{RD} ↓ Delay	—	—	180	—	—	80	ns
14	TdCr (RDr)	Clock ↑ to \overline{RD} ↑ Delay	—	—	180	—	—	70	ns
15	TsD (Cr)	Data Setup Time to Clock ↑	60	—	—	30	—	—	ns
16	ThD (RDr)	Data Hold Time to \overline{RD} ↑	0	—	—	0	—	—	ns
17	TsWAIT (Cf)	\overline{WAIT} Setup Time to Clock ↓	80	—	—	60	—	—	ns
18	ThWAIT (Cf)	\overline{WAIT} Hold Time after Clock ↓	15	—	—	10	—	—	ns
19	TdCr (M1f)	Clock ↑ to $\overline{M1}$ ↓ Delay	—	—	170	—	—	80	ns
20	TdCr (M1r)	Clock ↑ to $\overline{M1}$ ↑ Delay	—	—	170	—	—	80	ns
21	TdCr (RFSHf)	Clock ↑ to \overline{RFSH} ↓ Delay	—	—	220	—	—	110	ns
22	TdCr (RFSHr)	Clock ↑ to \overline{RFSH} ↑ Delay	—	—	220	—	—	100	ns
23	TdCr (RDr)	Clock ↓ to \overline{RD} ↑ Delay	—	—	180	—	—	70	ns
24	TdCr (RDf)	Clock ↑ to \overline{RD} ↓ Delay	—	—	150	—	—	70	ns
25	TsS (Cf)	Data Setup to Clock ↓ during M2, M3, M4 or M5 Cycles	70	—	—	40	—	—	ns
26	TdA (IORQf)	Address Stable prior \overline{IORQ} ↓	400	—	—	—	110	—	ns
27	TdCr (IORQf)	Clock ↑ to \overline{IORQ} ↓ Delay	—	—	150	—	—	65	ns
28	TdCf (IORQr)	Clock ↓ to \overline{IORQ} ↑ Delay	—	—	180	—	—	70	ns
29	TdD (WRf)	Data Stable Prior to \overline{WR} ↓	240	—	—	—	25	—	ns
30	TdCf (WRf)	Clock ↓ to \overline{WR} ↓ Delay	—	—	150	—	—	70	ns
31	TwWR	\overline{WR} Pulse Width	420	—	—	—	135	—	ns
32	TdCf (WRr)	Clock ↓ to \overline{WR} ↑ Delay	—	—	180	—	—	70	ns
33	TdD (WRf)	Data Stable Prior to \overline{WR} ↓	70	—	—	—	58	—	ns
34	TdCr (WRf)	Clock ↑ to \overline{WR} ↓ Delay	—	—	120	—	—	60	ns
35	TdWRr (D)	Data Stable from \overline{WR} ↑	200	—	—	—	27	—	ns
36	TdCf (HALT)	Clock ↓ to \overline{HALT} ↑ or ↓	—	—	350	—	—	260	ns
37	TwNMI	\overline{NMI} Pulse Width	180	—	—	70	—	—	ns
38	TsBUSREQ (Cr)	\overline{BUSREQ} Setup Time to Clock ↑	100	—	—	50	—	—	ns
39	ThBUSREQ (Cr)	\overline{BUSREQ} Hold Time after Clock ↑	15	—	—	10	—	—	ns
40	TdCr (BUSACKf)	Clock ↑ to \overline{BUSACK} ↓ Delay	—	—	180	—	—	90	ns
41	TdCf (BUSACKr)	Clock ↓ to \overline{BUSACK} ↑ Delay	—	—	160	—	—	90	ns
42	TdCr (Dz)	Clock ↑ to Data Float Delay	—	—	120	—	—	80	ns
43	TdCr (CTz)	Clock ↑ to Control Outputs Float Delay (MREQ, IORQ, \overline{RD} , and WR)	—	—	140	—	—	70	ns
44	TdCr (Az)	Clock ↑ to Address Float Delay	—	—	140	—	—	80	ns

290589

(3/3)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
45	TdCr (A)	MREQ ↑, IORQ ↑, RD ↑ and WR ↑ to Address Hold Time	200	—	—	—	32	—	ns
46	TsRESET (Cr)	RESET to Clock ↑ Setup Time	140	—	—	60	—	—	ns
47	ThRESET (Cr)	RESET to Clock ↑ Hold Time	20	—	—	10	—	—	ns
48	TsINTf (Cr)	INT to Clock ↑ Setup Time	100	—	—	70	—	—	ns
49	TsINTr (Cr)	INT to Clock ↑ Hold Time	15	—	—	10	—	—	ns
50	TdM1f (IORQf)	M1 ↓ to IORQ ↓ Delay	1050	—	—	—	362	—	ns
51	TdCf (IORQf)	Clock ↓ to IORQ ↓ Delay	—	—	185	—	—	70	ns
52	TdCr (IORQr)	Clock ↑ to IORQ ↑ Delay	—	—	200	—	—	70	ns
53	TdCf (D)	Clock ↓ to Data Valid Delay	—	—	300	—	—	130	ns

290589

4.3.2 AC Characteristics of CGC (in Active State)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
54	TRST (INT) S	CLK restart time by INT (STOP Mode)	—	214 + 2.5T _{CC}	—	—	(214 + 2.5)T _{CC}	—	ns
55	TRST (NMI) S	CLK restart time by NMI (STOP Mode)	—	214 + 2.5T _{CC}	—	—	(214 + 2.5)T _{CC}	—	ns
56	TRST (INT) I	CLK restart time by INT (IDLE Mode)	—	2.5 *T _{CC}	—	—	2.5 T _{CC}	—	ns
57	TRST (NMI) I	CLK restart time by NMI (IDLE Mode)	—	2.5 *T _{CC}	—	—	2.5 T _{CC}	—	ns
58	TRST (RESET) I	CLK restart time by RESET (IDLE Mode)	—	T _{CC}	—	—	T _{CC}	—	ns

290589

4.3.3 AC Characteristics of CTC (in Active State)

(1/2)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
59	TdM1 (IEO)	Delay from M1 fall to IEO fall (in case of generating only interrupt immediately before M1 cycle)	—	—	350	—	—	130	ns
60	TdIEI (IEOf)	Delay from IEI fall to IEO fall	—	—	200	—	—	100	ns
61	TdIEI (IEOr)	Delay from IEI rise to IEO rise (after ED decode)	—	—	250	—	—	110	ns

290589

(2/2)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
62	TsCTR (INT)	CLK/TRG setup for detection of interrupt TsCTR (C) Satisfied	T _{cC} + 200 + T ₆₈ + T ₄₈	—	—	T _{cC} + 120 + T ₆₈ + T ₄₈	—	—	ns
		TsCTR (C) not Satisfied	2T _{cC} + 200 + T ₆₈ + T ₄₈	—	—	2T _{cC} + 120 + T ₆₈ + T ₄₈	—	—	ns
63	T _{cCTR}	CLK/TRG Frequency	2T _{cC}	—	—	2T _{cC}	—	—	ns
64	T _{rCTR}	CLK/TRG rising time	—	—	50	—	—	40	ns
65	T _{fCTR}	CLK/TRG falling time	—	—	50	—	—	40	ns
66	T _{wCTRI}	Low CLK/TRG Pulse Width	240	—	—	120	—	—	ns
67	T _{wCTRh}	High CLK/TRG Pulse Width	240	—	—	120	—	—	ns
68	T _{sCTR (Cs)}	Set-up time up to CLK/TRG rise for clock rise requiring immediate count (counter mode)	300	—	—	150	—	—	ns
69	T _{sCTR (CT)}	Set-up time up to CLK/TRG rise for clock rise requiring immediate start of prescaler (timer mode)	300	—	—	150	—	—	ns
70	T _{dC (ZC / TO_r)}	Delay from clock rise to ZC/ TO rise	—	—	300	—	—	140	ns
71	T _{dC (ZC / TO_f)}	Delay from clock fall to ZC/ TO fall	—	—	220	—	—	140	ns

290589

4.3.4 AC Characteristics of I/O (in Active State)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (6MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
72	T _{dCf (Pout)}	Delay from clock fall to port data output	—	—	650	—	—	300	ns
73	T _{sPIN (IORDf)}	Port Input to $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ fall Set-up time	0	—	—	0	—	—	ns
74	T _{hPIN}	Port Input to $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ fall Hold Time	0	—	—	0	—	—	ns

290589

4.4 CAPACITANCE

T_a = 25°C

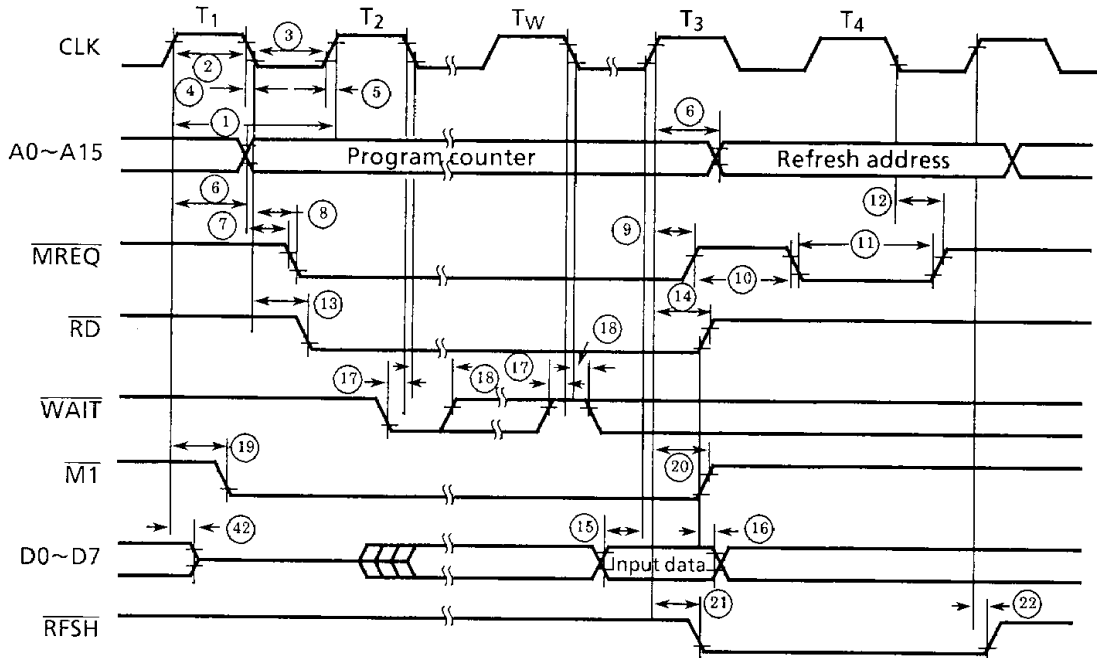
SYMBOL	Item	TEST COINDITION	MIN.	TUP.	MAX.	UNIT.
C _{IN}	Input Capacitance	f = 1MHz ALL terminals	—	5	—	PF
C _{out}	Output Capacitance	except that to be measured should be earthed	—	30	—	PF

290589

4.5 AC TIMING CHARTS (1) (IN ACTIVE STATE)

4.5.1 AC Timing Charts of CPU (in Active State)

Figure 4.5.1 through 4.5.8 show the basic timing charts. The circled numbers in these charts correspond to the numbers under the number column of the AC Electrical Characteristics Tables.



120489

Figure 4.5.1 Opcode Fetch Cycle

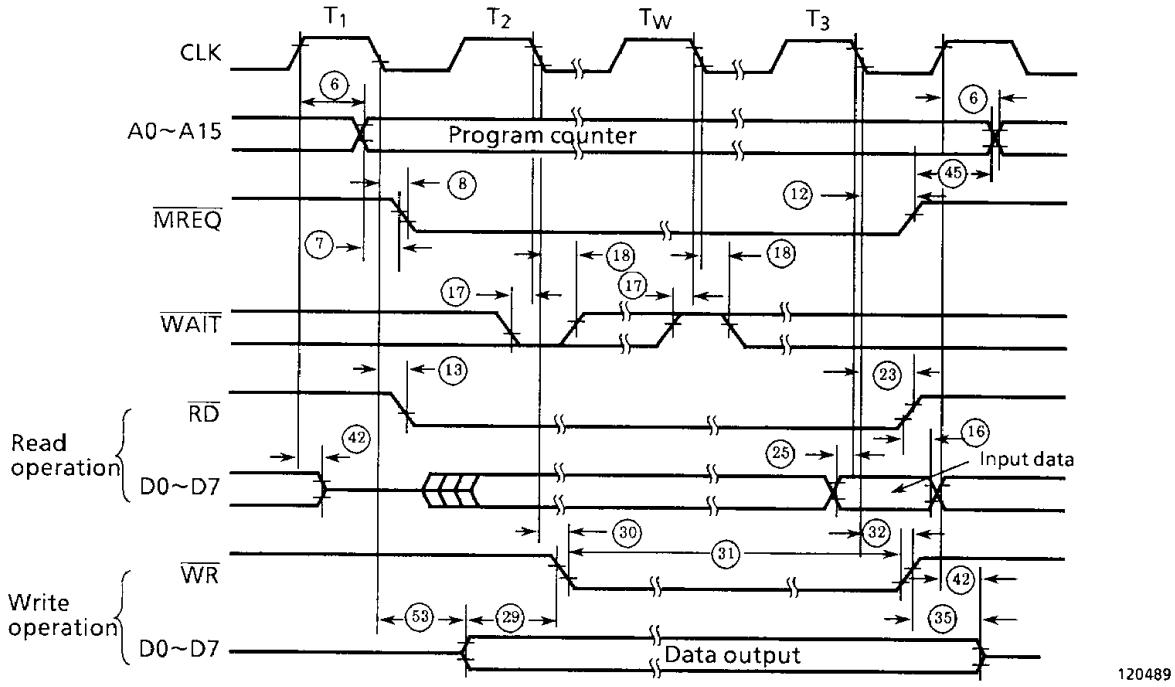
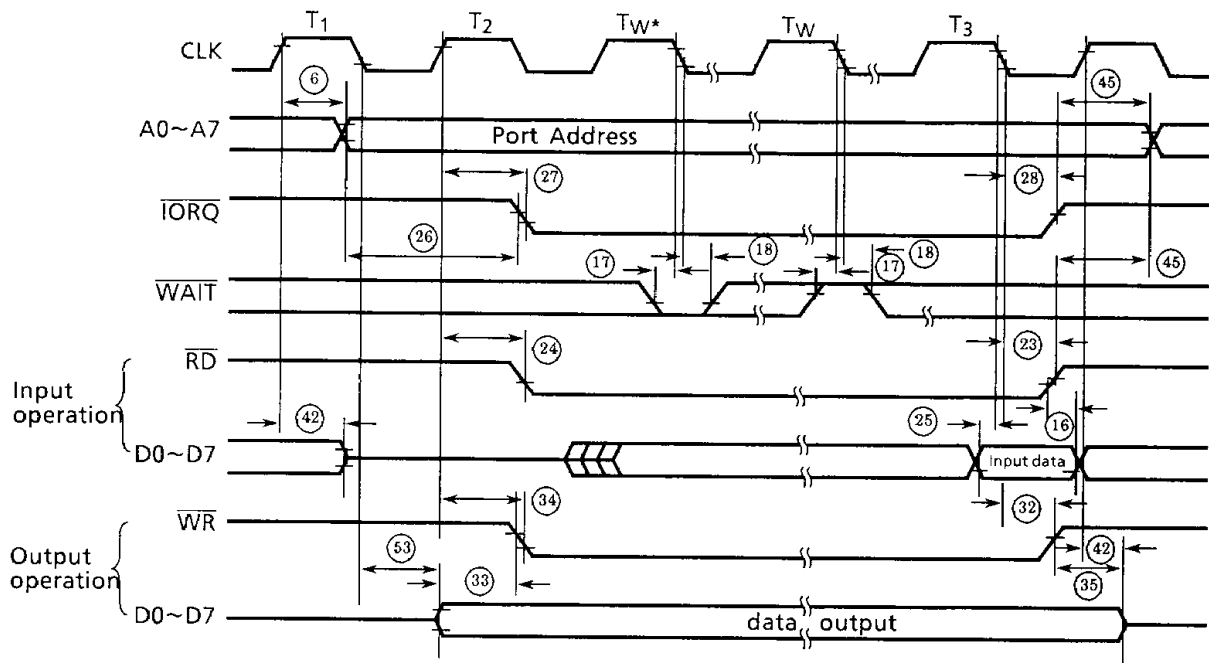


Figure 4.5.2 Memory Read/Write Cycle

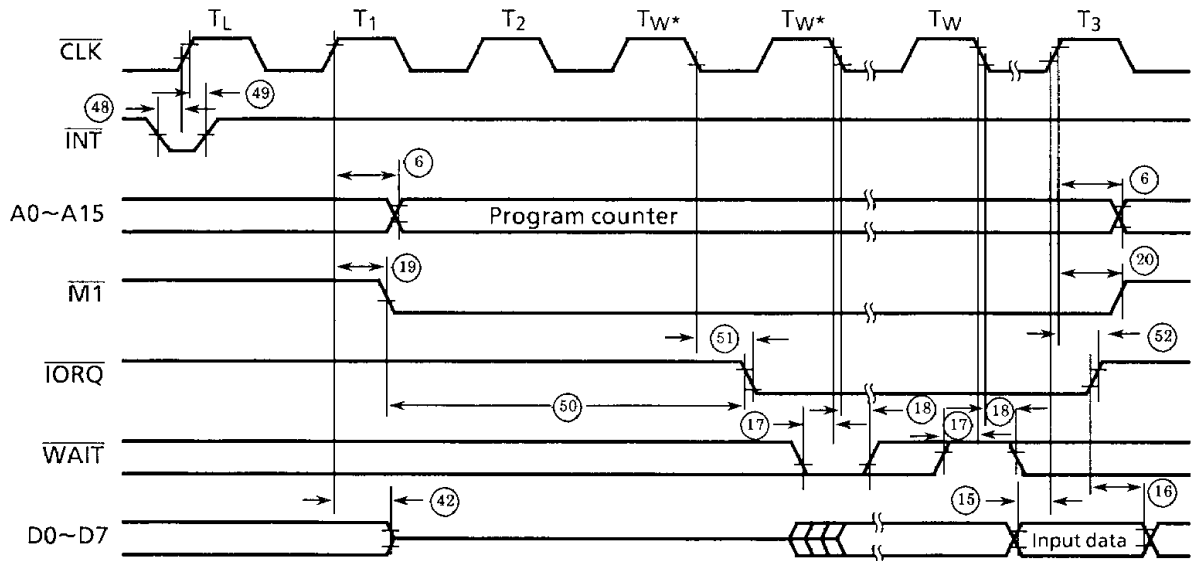
120489



Note: 1-wait state (Tw*) is inserted automatically by MPU

120489

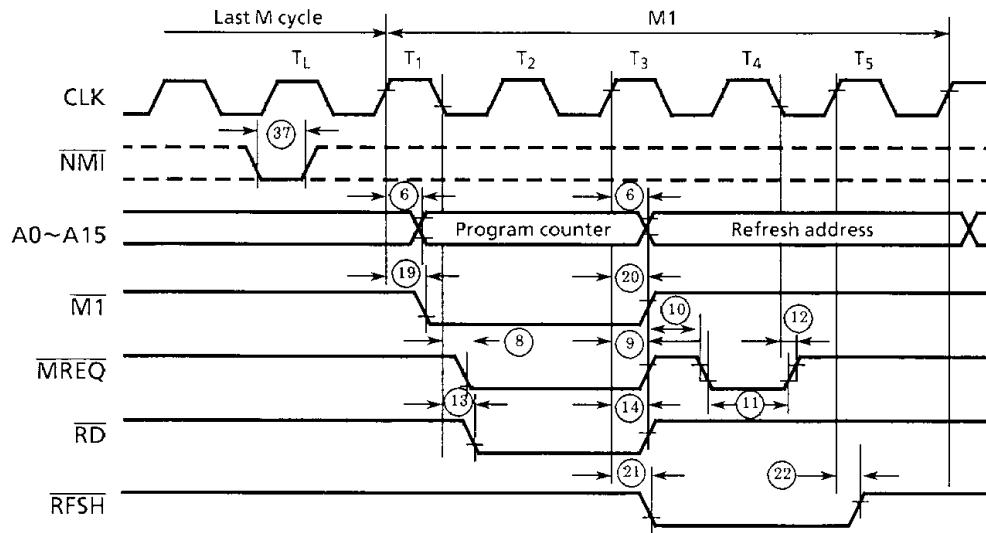
Figure 4.5.3 I/O Cycle



- Note: 1. T_L : Last state of instruction
 2. 2-wait state (T_{W^*}) is inserted automatically by MPU

120489

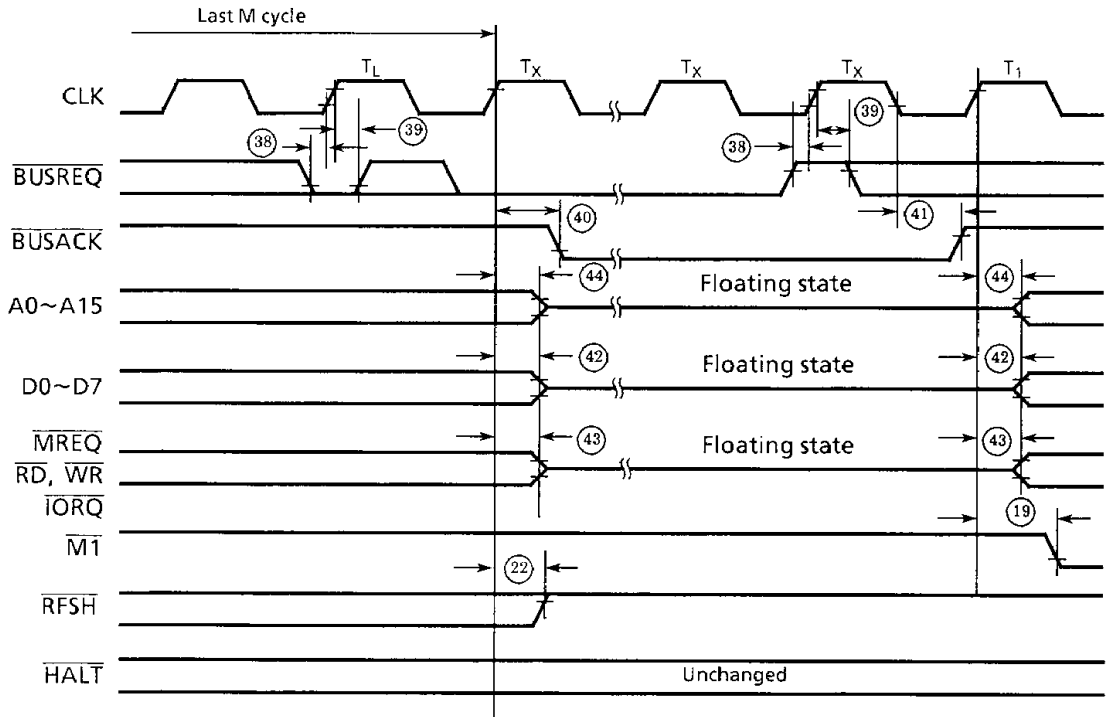
Figure 4.5.4 Interrupt Request/Acknowledge Cycle



- Note: \overline{NMI} is asynchronous, but its falling edge signal must occur synchronously with the rising edge of previous T_L state for correct response to the subsequent machine cycle.

120489

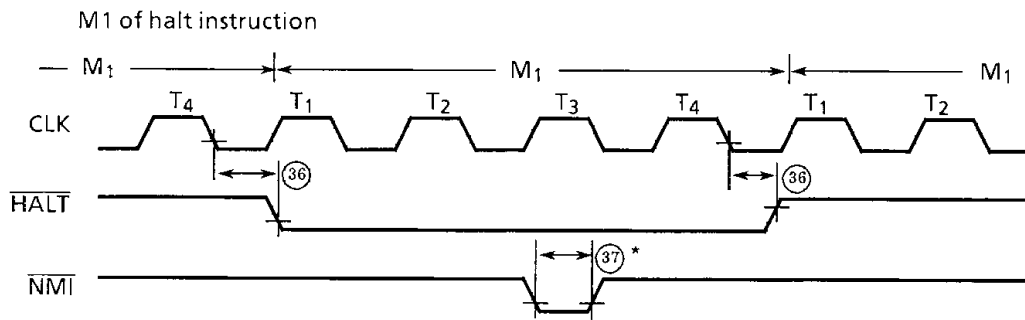
Figure 4.5.5 Non-maskable Interrupt Request Cycle



- Note: 1. T_L : Last state of a given machine cycle
 2. T_x : Clock used by peripheral LSI that made request.

120489

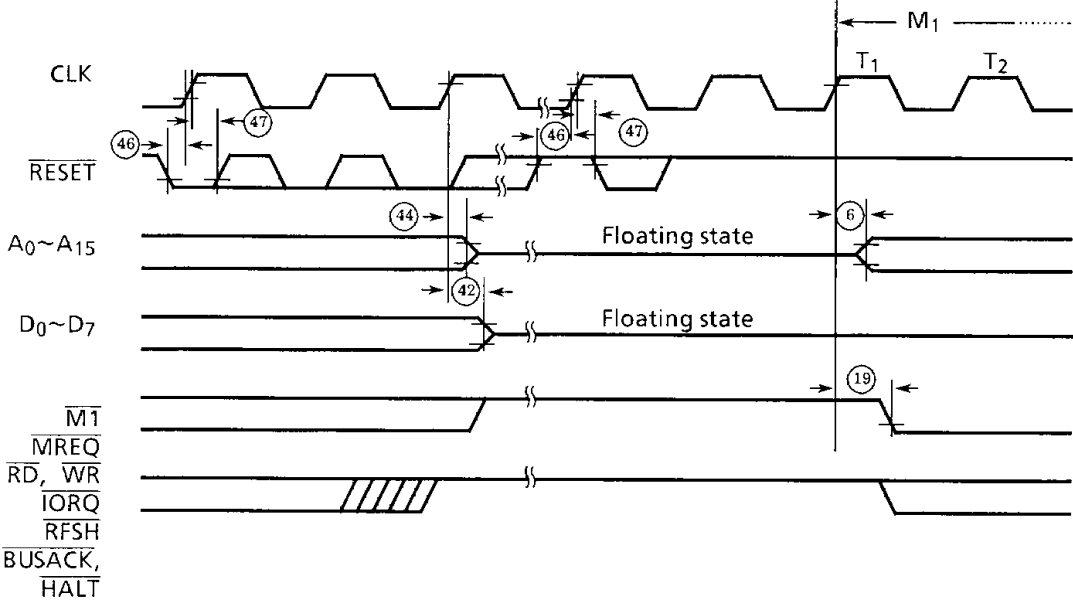
Figure 4.5.6 Bus Request/Acknowledge Cycle



- Note: \overline{INT} signal is also used for releasing HALT state.

120489

Figure 4.5.7 Halt Acknowledge Cycle



120489

Figure 4.5.8 Reset Cycle



4.5.2 AC Timing Cfacts of CGC (in Active State)

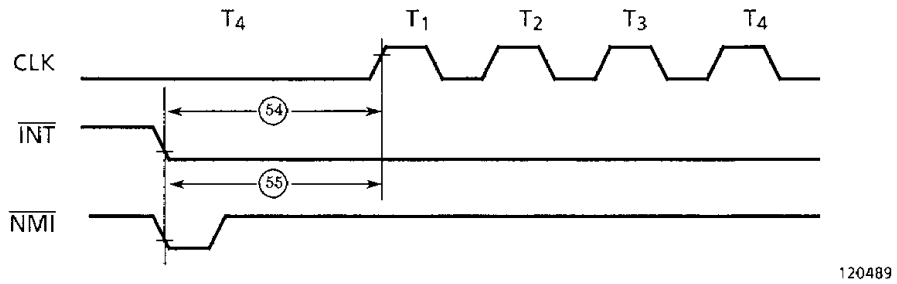


Figure 4.5.9 Clock Restart Timing (STOP mode)

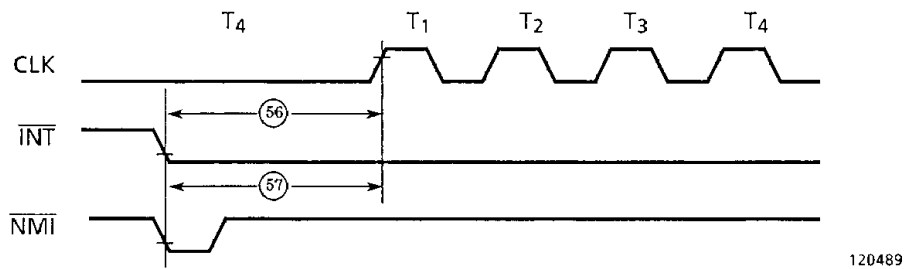


Figure 4.5.10 Clock Restart Timing (IDLE mode)

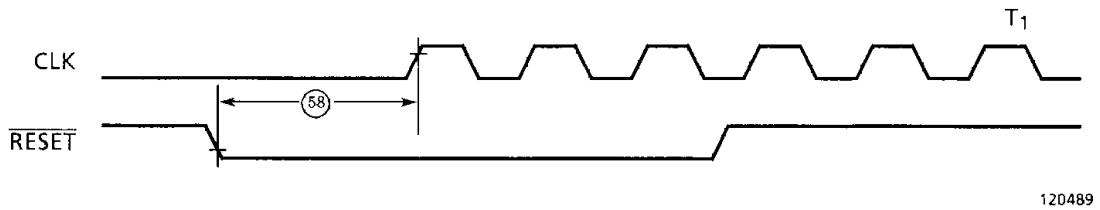
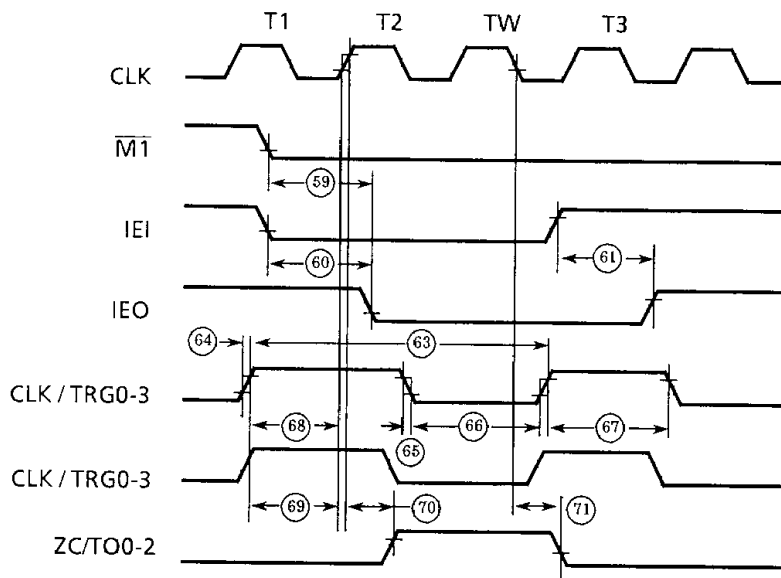


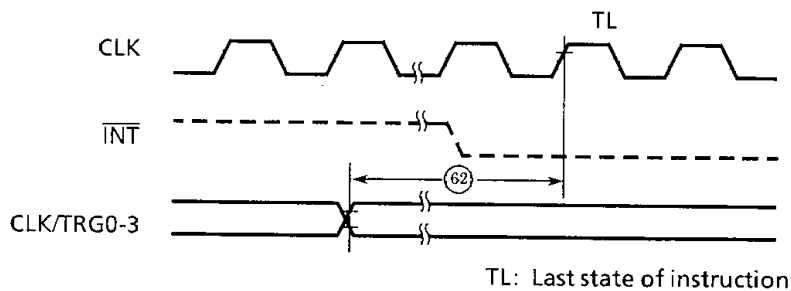
Figure 4.5.11 Timing of Clock Start by $\overline{\text{RESET}}$ (IDLE mode)

4.5.3 AC Timing Charts of CTC (in Active State)



120489

Figure 4.5.12 CTC Timing Diagram



120489

Figure 4.5.13 CTC Interrupt Occurrence Timing

4.5.4 AC Timing Charts of I/O (in Active State)

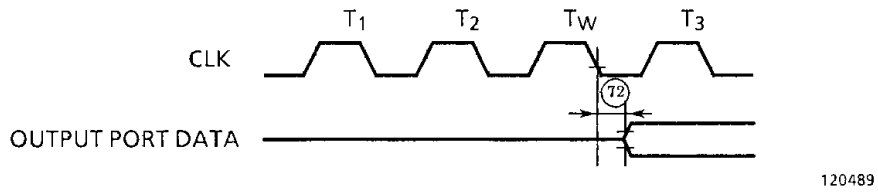


Figure 4.5.14 I/O OUTPUT Timing

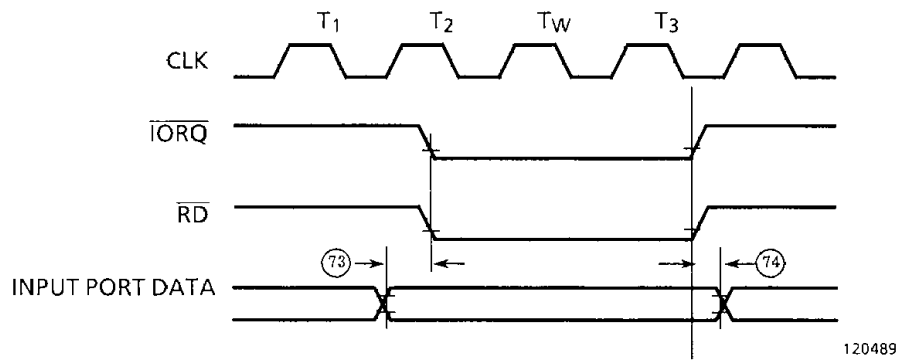


Figure 4.5.15 I/O INPUT Timing

4.6 AC ELECTRICAL CHARACTERISTICS (2) (IN INACTIVE STATE)

$T_a = -40^{\circ}\text{C} \sim +85^{\circ}\text{C}$, $V_{CC} = 5\text{V} \pm 10\%$, $V_{SS} = 0\text{V}$

4.6.1 AC Characteristics of CGC (in Inactive State)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 $V_{CC} = 2.7\text{V} \sim 5.5\text{V}$ (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
1	T_{cC}	CLOCK Cycle	500	—	DC	—	165	—	ns
2	T_{wCh}	CLOCK Width (High)	220	—	DC	—	70	—	ns
3	T_{wCl}	CLOCK Width (Low)	220	—	DC	—	70	—	ns
4	T_{fC}	CLOCK Fall Time	—	—	30	—	12	—	ns
5	T_{rC}	CLOCK Rise Time	—	—	30	—	12	—	ns
6	TRST (INT) S	Clock (CLK) restart Time by $\overline{\text{INT}}$ (STOP mode)	—	$2^{14} + 2.5T_{cC}$	—	—	$2^{14} + 2.5T_{cC}$	—	ns
7	TRST (NMI) S	Clock (CLK) restart Time by $\overline{\text{NMI}}$ (STOP mode)	—	$2^{14} + 2.5T_{cC}$	—	—	$2^{14} + 2.5T_{cC}$	—	ns
8	TRST (INT) I	Clock (CLK) restart Time by $\overline{\text{INT}}$ (IDLE mode)	—	$2.5 * T_{cC}$	—	—	$2.5 T_{cC}$	—	ns
9	TRST (NMI) I	Clock (CLK) restart Time by $\overline{\text{NMI}}$ (IDLE mode)	—	$2.5 * T_{cC}$	—	—	$2.5 T_{cC}$	—	ns
10	TRST (RESET) I	Clock (CLK) restart Time by $\overline{\text{RESET}}$ (IDLE mode)	—	$1T_{cC}$	—	—	$1T_{cC}$	—	ns
11	$T_{sHALT} (M1r)$	$\overline{\text{HALT}}$ set-up Time	10	—	—	10	—	—	ns

290589

4.6.2 AC Characteristics of CTC (in Inactive State)

(1/2)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
12	Th	Hold Time	20	—	—	0	—	—	ns
13	TsCS (C)	CS (A1, A0) set-up time to clock ↑	280	—	—	100	—	—	ns
14	TsCE (C)	CE (A7 to A2) set-up time to clock ↑	220	—	—	150	—	—	ns
15	TsIO (C)	$\overline{\text{IORQ}}$ ↓ set-up time to clock ↑	280	—	—	70	—	—	ns
16	TsRD (C)	$\overline{\text{RD}}$ ↓ set-up time to clock ↑	260	—	—	70	—	—	ns
17	TdC (DO)	clock ↑ to Data Valid Delay	—	—	260	—	—	130	ns
18	TdC (DOz)	$\overline{\text{IORQ}}$, $\overline{\text{RD}}$ ↑ to Data Float Delay	—	—	250	—	—	90	ns
19	TsDI (C)	Data Input set-up time to clock ↑	90	—	—	40	—	—	ns
20	TsM1 (C)	$\overline{\text{M1}}$ set-up time to clock ↑	240	—	—	70	—	—	ns
21	TdM1 (IEO)	clock ↑ $\overline{\text{M1}}$ ↓ to IEO ↓ Delay (in case of generating only interrupt immediately before M1) cycle	—	—	350	—	—	130	ns
22	TdIO (DOI)	$\overline{\text{IORQ}}$ ↓ to Data out delay (INTA cycle)	—	—	380	—	—	110	ns
23	TdIEI (IEOf)	IEI ↓ to IEO ↓ Delay	—	—	200	—	—	100	ns
24	TdIEI (IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)	—	—	250	—	—	110	ns
25	TdC (INT)	Clock ↑ to $\overline{\text{INT}}$ ↓ Delay	—	—	T _{cC} + 200	—	—	T _{cC} + 120	ns
26	TdCLK (INT)	CLK/TRG ↑ to $\overline{\text{INT}}$ ↓ Delay TsCTR(c) Satisfied	T _{cC} + 200 + T32	—	—	—	T _{cC} + 120 + 70 + T31	—	ns
		TsCTR (c) not Satisfied	2T _{cC} + 200 + T32	—	—	—	2T _{cC} + 120 + 70 + T31	—	ns
27	TcCTR	CLK/TRG Frequency	2T _{cC}	—	—	—	2T _{cC}	—	ns
28	TrCTR	CLK/TRG rising time	—	—	50	—	—	40	ns
29	TfCTR	CLK/TRG falling time	—	—	50	—	—	40	ns
30	TwCTRI	LOW CLK/TRG Pulse width	240	—	—	120	—	70	ns
31	TwCTRh	High CLK/TRG Pulse width	240	—	—	120	—	—	ns
32	TsCTR (Cs)	Set-up time up to CLK/TRG rise for clock rise requiring immediate (counter mode)	300	—	—	150	—	—	ns

290589

(2/2)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
33	TsCTR (Ct)	Set-up time up to CLK/TRG rise for clock rise requiring immediate start of prescaler (timer mode)	300	—	—	150	—	—	ns
34	TdC (ZC / TO _r)	Delay from clock rise to ZC/TO rise	—	—	300	—	—	140	ns
35	TdC (ZC / TO _f)	Delay from clock fall to ZC/TO fall	—	—	220	—	—	140	ns

290589

4.6.3 AC Characteristics of I/O (in Inactive State)

NO.	SYMBOL	PARAMETER	TMPZ84C011AF-6 V _{CC} = 2.7V~5.5V (2MHz)			TMPZ84C011AF-6 (6MHz)			UNIT
			MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	
36	TsCE (Cr)	CE (A7~A0) to clock ↑ Set-up time	300	—	—	150	—	—	ns
37	TsIO (C)	Set-up time from clock ↑ to $\overline{\text{IORQ}} \downarrow$	150	—	—	70	—	—	ns
38	TsRD (C)	Set-up time from clock ↑ to $\overline{\text{RD}} \downarrow$	150	—	—	70	—	—	ns
39	TdC (DO)	Delay from clock ↑ to Data Output	—	—	300	—	—	130	ns
40	TdC (DO _z)	Delay from clock $\overline{\text{IORQ}}$ and $\overline{\text{RD}} \uparrow$ to Data Float	—	—	150	—	—	90	ns
41	TsDI (C)	Data Input to clock ↑ Set-up time	0	—	—	0	—	—	ns
42	TsWR (C)	Set-up time from $\overline{\text{WR}} \downarrow$ to clock ↑	150	—	—	70	—	—	ns
43	TdIOWR _r (D)	Output Data Stable from $\overline{\text{IORQ}}$, $\overline{\text{WR}} \downarrow$	100	—	—	20	—	—	ns
44	TdC _f (Pout)	Delay from clock ↓ to Data Output	—	—	650	—	—	300	ns
45	TSPIN (IORD _f)	Port Input to $\overline{\text{IORQ}}$ and $\overline{\text{RD}} \downarrow$ Set-up time	0	—	—	0	—	—	ns
46	ThPIN (IORD _r)	Port Input to $\overline{\text{IORQ}}$ and $\overline{\text{RD}} \uparrow$ Hold Time	0	—	—	0	—	—	ns

290589

Note : Timing Measurements are made at the following voltage.

Input : V_{IH} = 2.4V V_{IL} = 0.4V
 Output : V_{OHC} = V_{CC} - 0.6V V_{OLC} = 0.6V (CLK)
 Output : V_{OH} = 2.4V V_{OL} = 0.8V (Except CLK)
 C_L = 100pF

4.7 AC TIMING CHARTS (2) (IN INACTIVE STATE)

4.7.1 AC Timing Charts of CGC (in Inactive State)

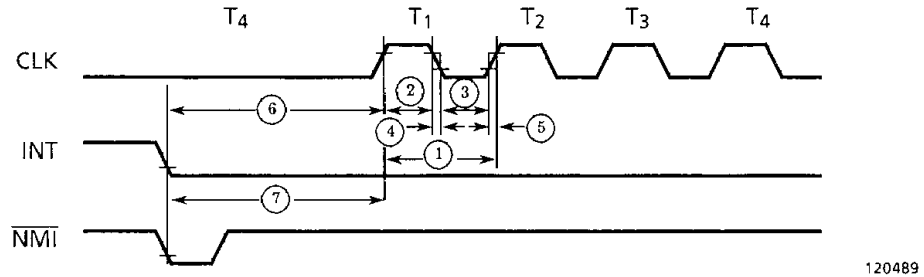


Figure 4.7.1 Clock Restart Timing (STOP mode)

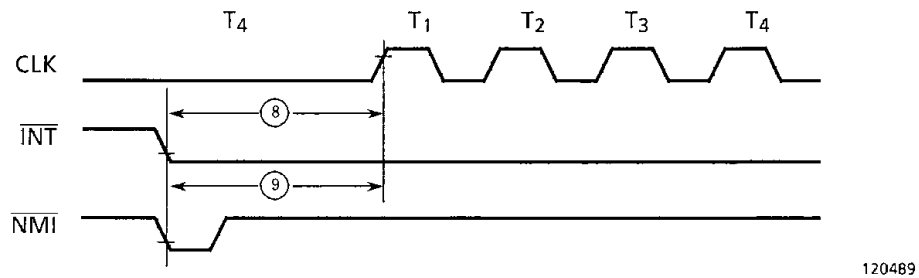


Figure 4.7.2 Clock Restart Timing (IDLE mode)

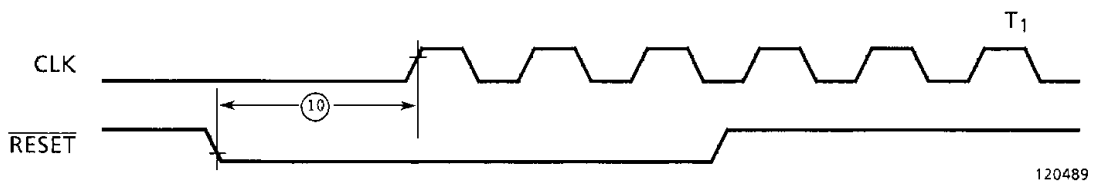


Figure 4.7.3 Timing if Clock Restart by RESET (IDLE mode)

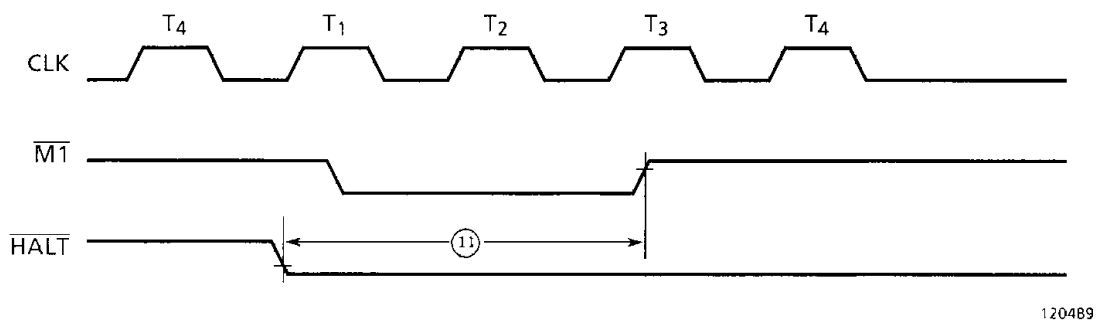
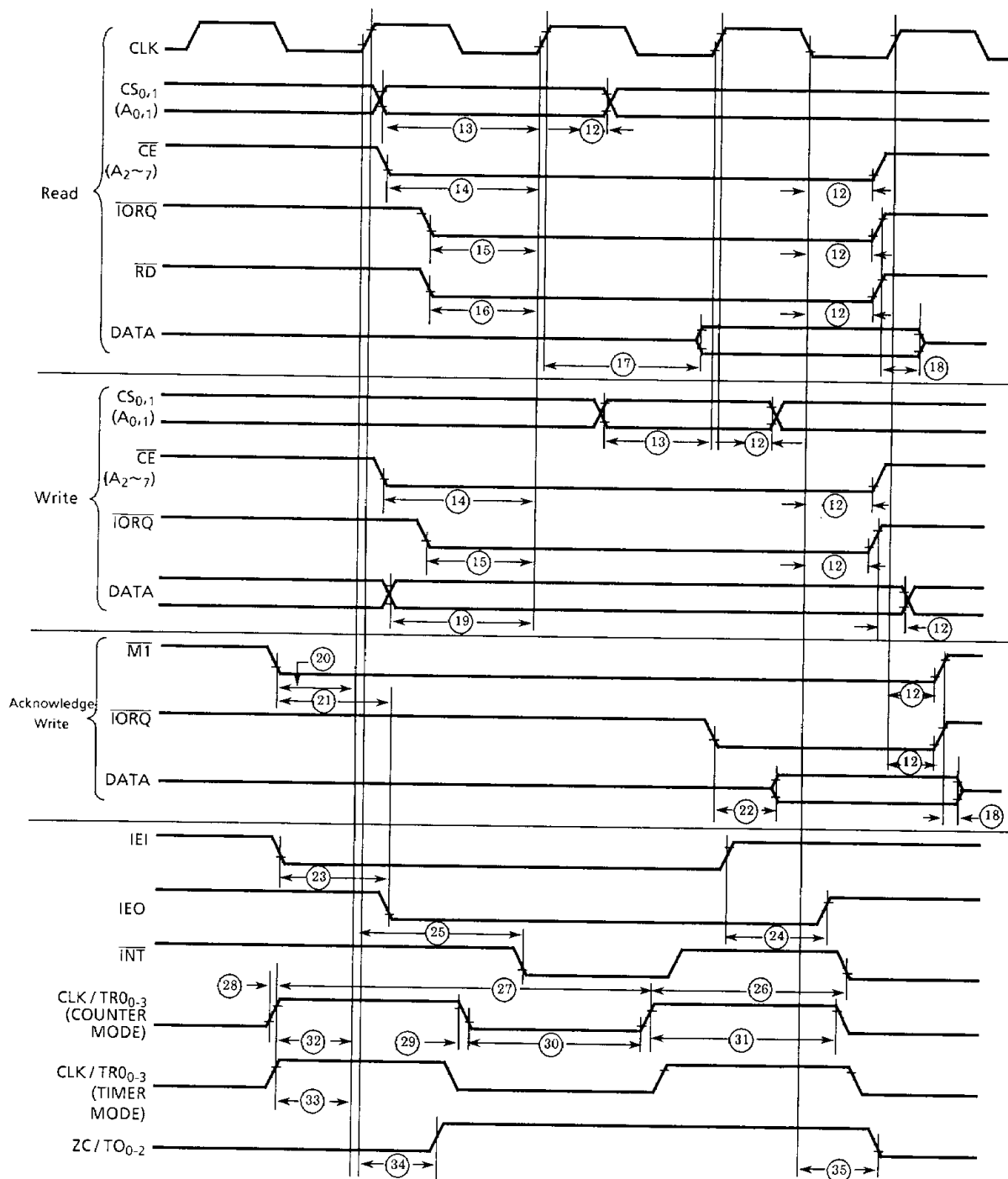


Figure 4.7.4 Clock Suspension Timing (IDLE/STOP modes)

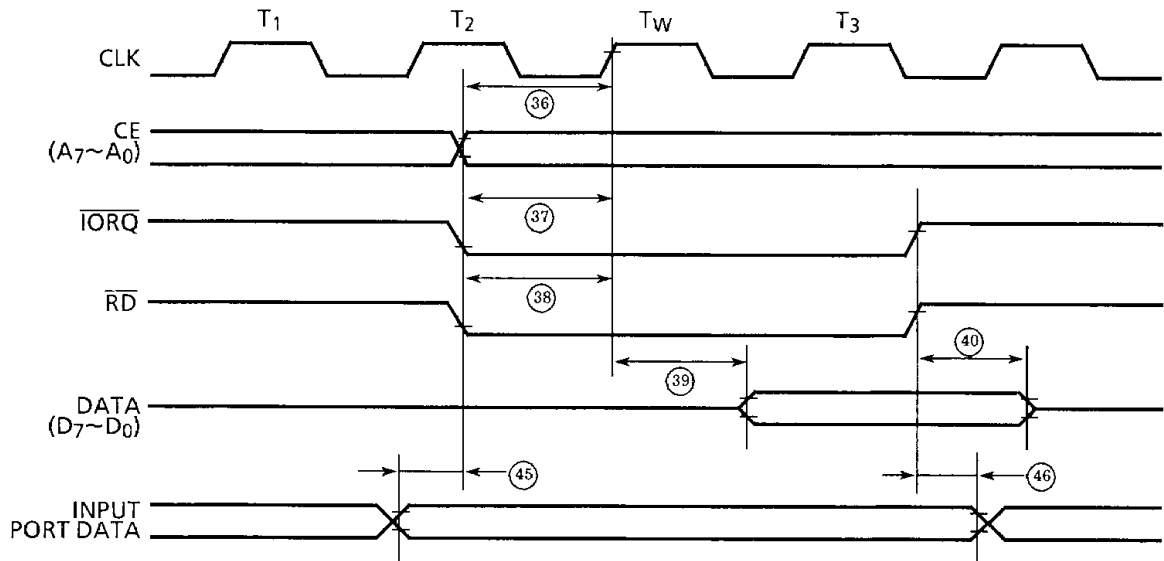
4.7.2 AC Timing Charts of CTC (in Inactive State)



120489

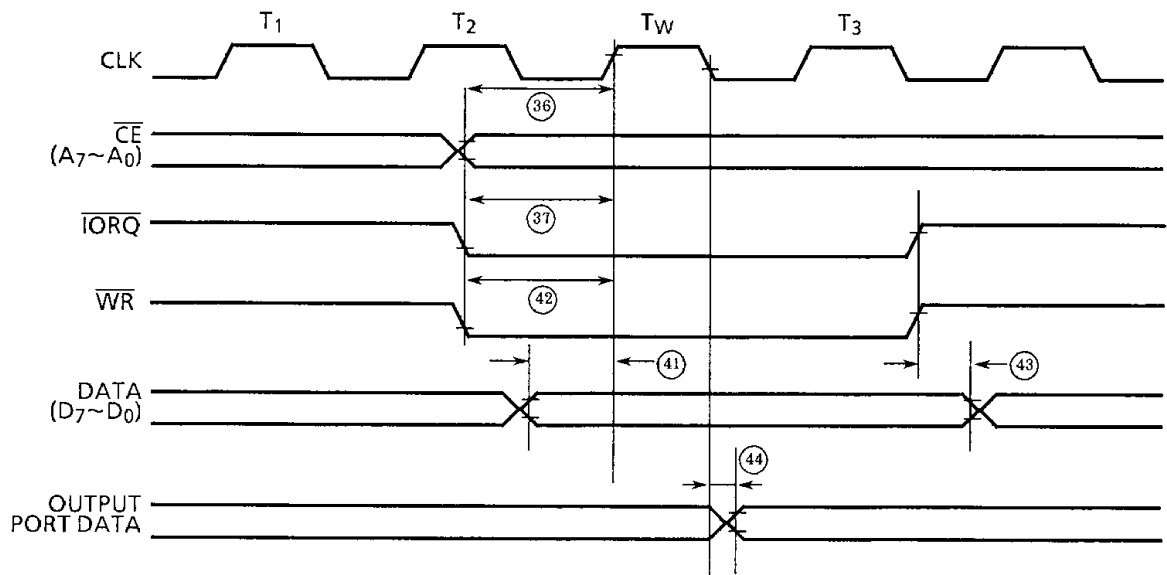
Figure 4.7.5 CTC Timing Diagram (Inactive)

4.7.3 AC Timing Charts of I/O (in Inactive State)



120489

Figure 4.7.6 (a) I/O READ Timing



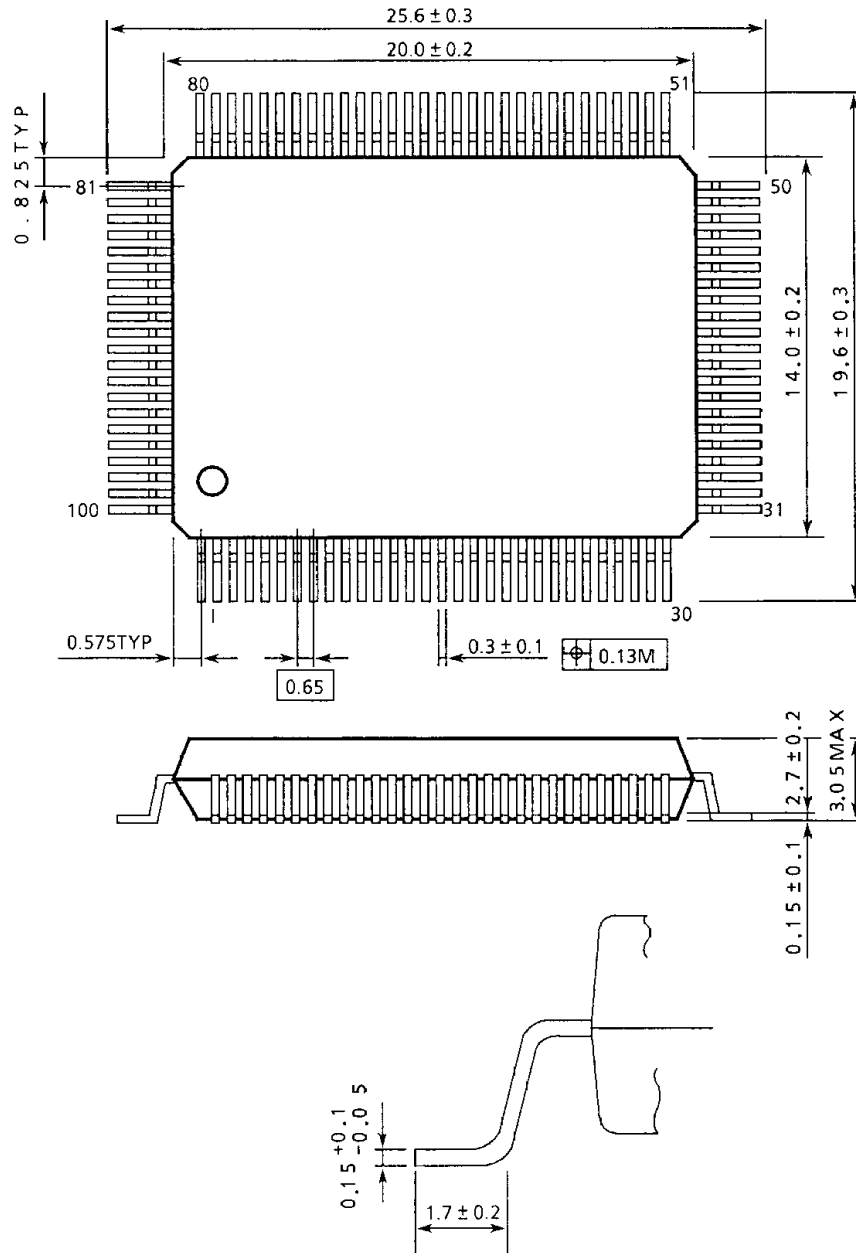
120489

Figure 4.7.6 (b) I/O WRITE Timing

4.8 OUT LINE DRAWING

QFP100-P-1420B

Unit : mm



290589