Toshiba 東芝

INTEGRATED CIRCUIT

TECHNICAL DATA

TOSHIBA MOS DIGITAL INTEGRATED CIRCUIT

TMP8085AP

N-CHANNEL SILICON GATE MOS

8-BIT SINGLE CHIP MICROPROCESSOR

GENERAL DESCRIPTION

The TMP8085AP, from here on referred to as the TMP8085A, is a new generation, complete 8 bit parallel central processing unit (CPU). Its instruction set is 100% software compatible with the TMP9080A (8080A) microprocessor, and it is designed to improve the present 9080's performance by higher system speed. Its high level of system integration allows a minimum system of there IC's : TMP8085A (CPU), TMP8155P/TMP8156P (RAM/IO) and TMP8755AC (EPROM/IO)/ TMP8355P (ROM/IO). The TMP8085A uses a multiplexed data bus. The address is plit between the 8 bit address bus and the 8 bit data bus. The on-chip address latches of TMP8155P/TMP8156P/TMP8755AC/TMP8355P memory products allow a direct interface with TMP8085P.

FEATURES

- 100% Software Compatible with TMP9080A
- 1.3 s Instruction Cycle
- Single +5V Power Supply
- On-Chip Clock Generator (with External Crystal or RC Network)
- On-Chip System Controller; Advanced Cycle status information available for Large System Control
- 4 Vectored Interrupts (One is Non-Maskable) Plus an TMP9080A compatible interrupt
- Decimal, Binary and Double Precision Arithmetic
- Serial In/Serial Out Port
- Direct Addressing Capability to 64K Bytes of Memory
- Compatible with Intel's 8085A

PIN CONNECTION

(TOP VIEW)

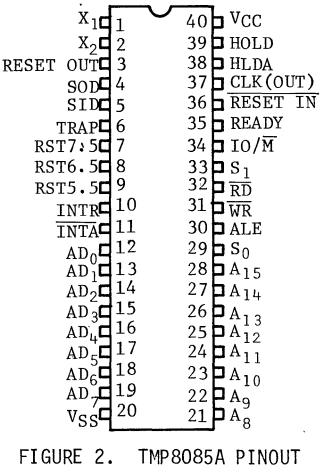


FIGURE 2. TMP8085A PINOUT DIAGRAM

BLOCK DIAGRAM

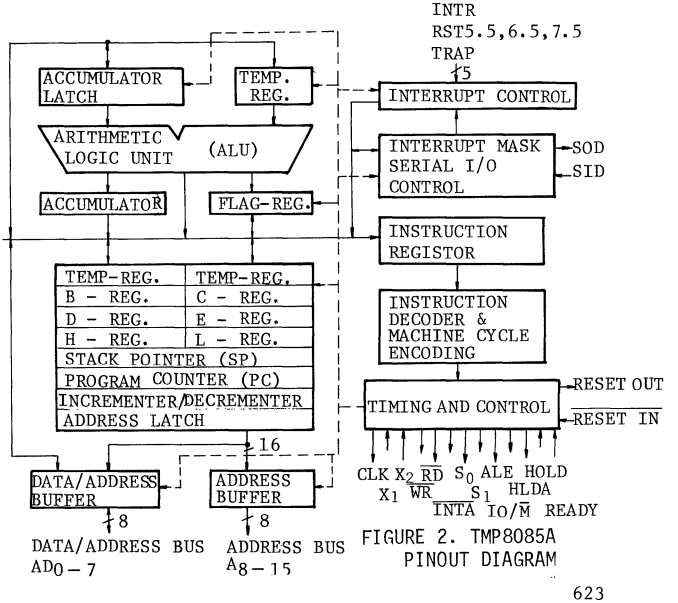


FIGURE 2. TMP8085A PINOUT DIAGRAM

PIN NAME AND PIN DESCRIPTION

X_1, X_2 (Input)

Crystal, LC, or RC network are connected to X_1 and X_2 to drive the internal clock generator. X_1 and X_2 can also be driven from an externally derived frequency source. The input frequency is devided by 2 to give the processor's internal operating frequency.

CLK (Output)

Clock Output for use as a system clock. The period of CLK is twice the X_1, X_2 input period.

$\overline{\text{RESET IN}}$ (Input)

The RESET Input initialize the processor by clearing the program counter, instruction register, SOD latch, Interrupt Enable flip-flop and HLDA flip-flop. The address and data buses and the control lines are 3-stated during RESET and because of the asynchronous nature of RESET, the processor's internal registers and flags may be altered by RESET with unpredictable results. $\overline{\text{RESET IN}}$ is a Schmitt-triggered input, allowing connection to an RC network for power on RESET delay. The TMP8085A is held in the reset condition as long as $\overline{\text{RESET IN}}$ is applied.

RESET OUT (OUTPUT)

The RESET OUT signal indicates that the TMP8085A is being reset. It can be used as a system reset. It is synchronized to the processor clock and lasts an integral number of clock periods.

SOD (Output)

Serial output data line. The output SOD is set or reset as specified by the SIM instruction.

SID (Input)

Serial input data line. The data on this line is loaded into accumulator bit 7 whenever a RIM instruction is executed.

INTR (Input)

INTERRUPT REQUEST signal provides a mechanism for external devices to modify the instruction flow of the program in progress. It is sampled only during the next to the last clock cycle of an instruction and during Hold and Halt states. If it is recognized, the processor will complete the execution of the current instruction, and then the Program Counter (PC) will be inhibited from incrementing and an $\overline{\text{INTA}}$ will be issued. During this cycle a RESTART or CALL instruction can be inserted to jump to the interrupt service routine. The INTR is enabled and disabled by software. It is disabled by RESET and immediately after an interrupt is accepted.

$\overline{INTA}$ (Output)

INTERRUPT ACKNOWLEDGE: Occurs in response to an Interrupt input and indicates that the processor will be ready for an interrupt instruction on the data bus. It is used instead of (and has the same timing as) $\overline{RD}$ during the instruction cycle after an INTR is accepted.

RST 5.5
RST 6.5 } (Inputs)
RST 7.5

RESTART INTERRUPTS: These three inputs have the same timing as INTR exept they cause an internal RESTART to be automatically inserted. These interrupts have a higher priority than INTR. The priority of these interrupts is ordered as shown Table 1. They may be individually masked out using the SIM instruction.

TRAP (Input)

Trap interrupt is a nonmaskable RESTART interrupt. It is sampled at the same timing as INTR or RST 5.5 - 7.5. It is unaffected by any mask or Interrupt Enable. It has the highest priority of any interrupt.

AD_0 - AD_7 (Input/Output, 3 - state)

Lower 8 bits of the memory address (or I/O address) appear on the bus during the first clock cycle (T_1 state) of a machine cycle. It then becomes the data bus during the second and third clock cycles.

A_8 - A_{15}(Output, 3-state)

The most significant 8 bits of the memory address or the 8 bits of the I/O address, 3-stated during Hold and Halt modes and during RESET.

S_0, S_1, and $IO/\overline{M}$ (Output)
Machine cycle status:

$IO/\overline{M}$	S_1	S_0	Status
0	1	1	Opcode fetch
0	1	0	Memory read
0	0	1	Memory write
1	1	0	I/O read
1	0	1	I/O write
1	1	1	Interrupt Acknowledge
TS	0	0	Halt
TS	X	X	Hold
TS	X	X	Reset

Note: TS = 3-state (high impedance)
X = unspecified

ALE (Output)

Address Latch Enable: It occurs during the first clock state of a machine cycle and enables the address to get latched into the on-chip latch of peripherals. The falling edge of ALE can be used to strobe the status information. ALE is never 3-stated.

$\overline{WR}$ (Output, 3-state)

WRITE control: A low level on $\overline{WR}$ indicates the data on the Data Bus is to be written into the selected memory or I/O location. Data is set up at the trailing edge of $\overline{WR}$. It is 3-stated during Hold and Halt modes and during RESET.

$\overline{RD}$ (Output, 3-state)

READ control: A low level on $\overline{RD}$ indicates the selected memory or I/O device to be read and that the Data Bus is available for the data transfer, 3-stated during Hold and Halt modes and during RESET.

READY(Input)

When READY is absent (low), indicating the external operation is not complete, the processor will enter the Wait state. It will wait an integral number of clock cycles for READY to go high before completing the read or write cycle.

HOLD (Input)

The Hold input allows an external signal to cause the processor to relinquish control over the address bus and the data bus. When Hold goes active, the processor completes its current operation, activates the HLDA output, and puts the Address, Data, $\overline{RD}$, $\overline{WR}$, and IO/$\overline{M}$ lines into their high-impedance state. Internal processing can continue. The Holding device can then utilize the address and data buses without interference. The processor can regain the bus only after the Hold is removed.

HLDA (Output)

The Hold Acknowledge output signal is a response to a Hold input. It indicates that the processor has received the HOLD request and it will relinquish the bus in the next cycle. HLDA goes low after the Hold request is moved. The processor takes the bus one half clock cycle after HDLA goes low.

V_{CC}
+5 volt supply

V_{SS}
Ground Reference

FUNCTIONAL DESCRIPTION

The TMP8085A is a complete 8-bit parallel central processor. Its basic clock speed is 3 MHz. Also it is designed to fit into a minimum system of three IC's: The CPU (TMP8085A), a RAM I/O (TMP8155P or TMP8156P), and a ROM or EPROM I/O chip (TMP8355P or TMP8755AC).

The TMP8085A is provided with internal 8-bit registers and 16-bit registers. The TMP8085A has eight addressable 8-bit registers. Six of them can be used either as 8-bit registers or as 16-bit register pairs. In addition to the register pairs, the TMP8085A contains two more 16-bit registers. The TMP8085A register set is as follows:

- The accumulator (A Register) is the focus of all of the accumulator instructions, which include arithmetic, logic, load and store, and I/O instructions.
- The program counter (PC) always points to the memory location of the next instruction to be executed.
- General - purpose registers BC, DE, and HL may be used as 8-bit registers or as three 16-bit registers, interchangeably, depending on the instruction being performed.
- The stack pointer (sp) is a special data pointer that always points to the stack top (next available stack address).
- The flag register contains five one-bit flags, each of which records processor status information and may also control processor operation.

The five flags in the TMP8085A CPU are shown below:

(MSB)

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		C

- The carry flag (C) is set and reset by arithmetic operations. An addition operation that resutls in an overflow out of the high-order bit of the accumulator sets the carry flag. The carry flag also acts as a "borrow" flag for subtract instruction.
- The auxiliary carry flag (AC) indicates overflow out of bit 3 of the accumulator in the same way that C flag indicates overflow out of bit 7. This flag is commonly used in BCD arithmetic.
- The sign flag (S) is set to the condition of the most significant (MSB) bit of the accumulator following the execution of arithmetic or logic instructions.
- The zero flag (Z) is set if the result generated by certain instructions is zero. The zero flag is cleared if the result is not zero.
- The parity flag (P) is set to 1 if the parity (number of 1-bits) of the accumulator is even. If odd, it is cleared.

In the TMP8085A microprocessor are contained the functions of clock generation, system bus control, and interrupt priority selection, in addition to execution of the instruction set. The TMP8085A uses a multiplexed Data Bus. The address is split between the higher 8-bit Address Bus and the lower 8-bit Address/Data Bus. During the first T state (T_1 clock cycle) of a machine cycle the lower order address is sent out on the Address/Data Bus. These lower 8 bits may be latched externally by the Address Latch Enable signal (ALE). During the rest of the machine cycle the data bus is used for memory or I/O data.

INTERRUPT AND SERIAL I/O

The TMP8085A has 5 interrupt inputs: INTR, RST 5.5, RST 6.5, RST 7.5, and TRAP. INTR is identical in function to 9080A INT. Each of three RESTART inputs 5.5, 6.5, 7.5, has a programmable mask. TRAP is also a RESTART interrupt but it is nonmaskable.

The three RESTART interrupts cause the internal execution of RESTART if the interrupts are enabled and if the interrupt mask is not set. The nonmaskable TRAP causes internal execution independent of the state of the interrupt enable or masks.

There are two different types of inputs in the restart interrupts. RST 5.5 and RST 6.5 are high levelsensitive like INTR and are recognized with the same timing as INTR.RST 7.5 is rising edge-sensitive.

For RST 7.5, only a pulse is required to set an internal flip-flop which generates the interrupt request. The RST 7.5 request flip-flop remains set until the request is serviced. Then it is reset automatically. This flip-flop may also be reset by using the SIM instruction or by issuing a $\overline{\text{RESET IN}}$ to the TMP8085A. The RST 7.5 internal flip-flop will be set by a pulse on the RST 7.5 pin even when the RST 7.5 interrupt is masked out.

The interrupts are arranged in a fixed priority that determines which interrupt is to be recognized if more than one is pending : TRAP-highest priority. RST 7.5, RST 6.5, RST 5.5, INTR - lowest priority. This priority scheme does not take into account the priority of a routine that was started by a higher priority interrupt. RST 5.5 can interrupt a RST 7.5 routine if the interrupts were reenabled before the end of the RST 7.5 routine.

The TRAP interrupt is useful for catastrophic errors such as power failure or bus error. It is not affected by any flag or mask. The TRAP input is both edge and level sensitive. The TRAP input must go high and remain high until it is acknowledged. It will not be recognized again until it goes low, then high again. This avoids any false triggering due to noise or logic glitches.

TABLE 1. INTERRUPT PRIORITY, RESTART ADDRESS, AND SENSITIVITY

Name	Priority	Address Branched to When Interrupt Occurs	Type Trigger
TRAP	1	24 (Hex.)	Rising edge and high level until sampled.
RST 7.5	2	3C (Hex.)	Rising edge (latched).
RST 6.5	3	34 (Hex)	High level until sampled.
RST 5.5	4	2C (Hex.)	High level until sampled.
INTR	5	See Note (2)	High level until sampled.

Notes: (1) The processor pushes the PC on the stack before branching to the indicated address.

(2) The address branched to depends on the instruction provided to the TMP8085A when the interrupt is acknowledged.

The TRAP interrupt is special in that it disables interrupts, but preserves the previous interrupt enable status. Performing the first RIM instruction following a TRAP interrupt allows you to determine whether interrupts were enabled or disabled prior to the TRAP. All subsequent RIM instruction provide current interrupt enable status. Performing a RIM instruction following INTR, or RST 5.5 - 7.5 will provide current interrupt enable status, revealing that interrupts are disabled.

The serial I/O system is also controlled by the RIM and SIM instructions. SID is read by RIM, and SIM sets the SOD date.

BASIC TIMING

The execution of each instruction by the TMP8085A consists of a sequence of from one to five machine cycles, and each machine cycle consists of a minimum of from three to six clock cycles. Most machine cycles consist of three T states, (cycles of the CLK output) with the exception of opecode fetch, which normally has either four or six T states (unless WAIT or HOLD states are forced by the receipt of $\overline{\text{READY}}$ or HOLD inputs). Any T state must be one of ten possible states, shown in Table 3.
At the beginning of every machine cycle, the TMP8085A sends out three status signals ($IO/\overline{M}$, S_1, S_0) that define what type of machine cycle is about to take place. The TMP8085A also sends out a 16-bit address at the beginning of every machine cycle to identify the particular memory location or I/O port that the machine cycle applies to.

The special timing signal, ADDRESS LATCH ENABLE (ALE), is used a strobe to sample the lower 8-bits of address on the $AD_0 - AD_7$ lines. ALE is present during T_1 of every machine cycle. Control lines $\overline{RD}$ ($\overline{INTA}$) and $\overline{WR}$ become active later, at the time when the transfer of data is to take plece. Figure 3 shows an instruction fetch, memory read and I/O write cycle (as would occur during processing of the OUT instruction).

TABLE 2. TMP8085A MACHINE CYCLE CHART

MACHINE CYCLE	IO/$\overline{M}$	S_1	SO	$\overline{RD}$	$\overline{WR}$	$\overline{INTA}$
OPCODE FETCH	0	1	1	0	1	1
MEMORY READ	0	1	0	0	1	1
MEMORY WRITE	0	0	1	1	0	1
I/O READ	1	1	0	0	1	1
I/O WRITE	1	0	1	1	0	1
ACKNOWLEDGE OF INTR	1	1	1	1	1	0
BUS IDLE : DAD	0	1	0	1	1	1
ACK. OF RST, TRAP	1	1	1	1	1	1
HALT	TS	0	0	TS	TS	1

NOTE: 0 = Logic "0", 1 = Logic "1", TS = High Impedance

TABLE 3. TMP8085A MACHINE STATE CHART

MACHINE STATE	S_1, S_0	IO/$\overline{M}$	A_8-A_{15}	AD_0-AD_7	$\overline{RD}, \overline{WR}$	$\overline{INTA}$	ALE
T_1	X	X	X	X	1	1	1°
T_2	X	X	X	X	X	X	0
T_{WAIT}	X	X	X	X	X	X	0
T_3	X	X	X	X	X	X	0
T_4	1	0†	X	TS	1	1	0
T_5	1	0†	X	TS	1	1	0
T_6	1	0†	X	TS	1	1	0
T_{RESET}	X	TS	TS	TS	TS	1	0
T_{HALT}	0	TS	TS	TS	TS	1	0
T_{HOLD}	X	TS	TS	TS	TS	1	0

NOTES: (1) 0 = Logic "0", 1 = Logic "1", TS = High Impedance, X = Unspecified

(2) °ALE not generated during 2nd and 3rd machine cycles of DAD instruction

(3) † IO/$\overline{M}$ = 1 during T_4 - T_6 of INA machine cycle

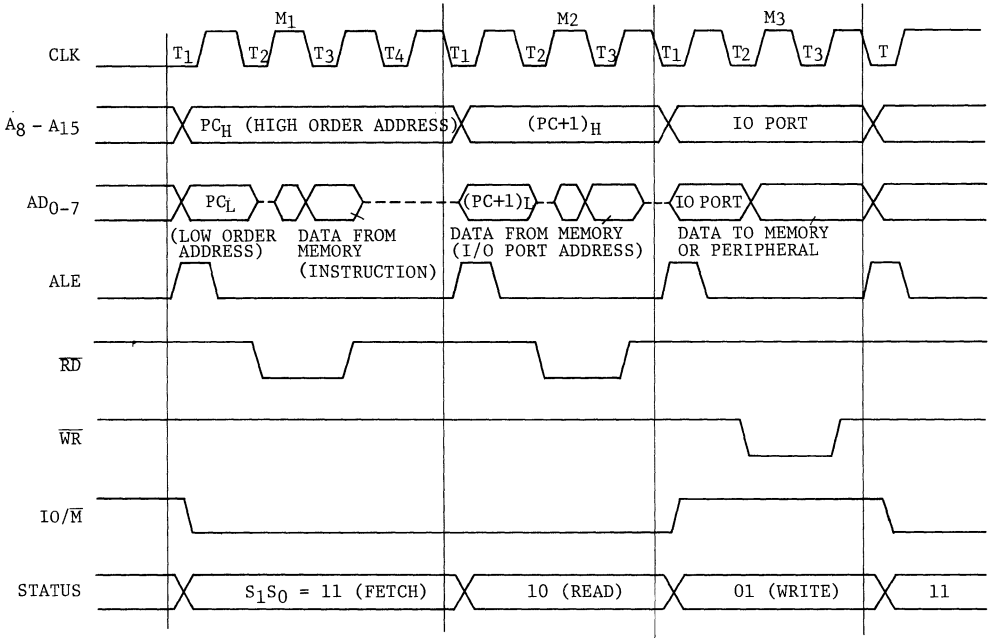


FIGURE 3. TMP8085A BASIC SYSTEM TIMING

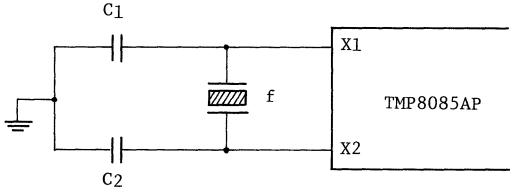
DRIVING THE X1 AND X2 INPUTS

You may drive the clock inputs of the TMP8085A with a crystal, an LC tuned circuit, an RC network or an external clock source. The driving frequency must be at least 1 MHz, and must be twice the desired internal clock frequency.

A. Quartz Crystal Clock Driver

If a crystal used, it must have the following characteristics.

- Parallel resonance at twice the clock frequency desired
- C_S (shunt capacitance) $\leq$ 7 PF
- R_S (equivalent shunt resistance) $\leq$ 75 Ohms



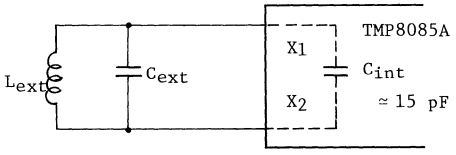
Note a value of the external capacitors C_1 and C_2 between X1, X2 and ground. In case of the crystal frequency above 4 MHz, it is recommended that you choose a value of 10pF for C_1 and C_2 and less than 4 MHz, 20pF capacitors are recommended.

B. LC Turned Circuit Clock Driver

A parallel-resonant LC circuit may be used as the frequency-determining network for the TMP8085A, providing that its frequency tolerance of approximately 10% is acceptable. The components are chosen from the formula.

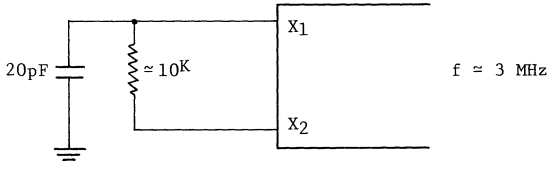
$$f = \frac{1}{2\pi \sqrt{L\ (C_{ext} + C_{int})}}$$

The use of an LC circuit is not recommended for frequencies higher than approximately 5 MHz.

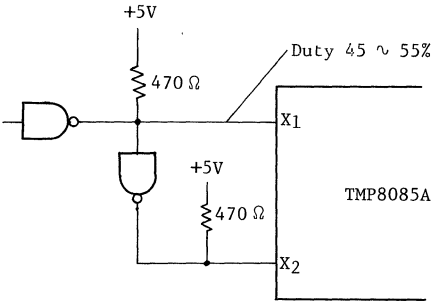


C. RC Circuit Clock Driver

An RC circuit may be used as the frequency - determining network for the TMP 8085A if maintaining a precise clock frequency is of no importance. Variations in the on-chip timing generation can cause a wide variation in frequency when using RC circuit. The driving frequency generated by the circuit shown is approximately 3 MHz. It is not recommended that frequencies greatly higher or lower than this be attempted.



D. External clock Driver Circuit



POWER ON AND $\overline{\text{RESET IN}}$

The TMP 8085A is not guaranteed to work until 10 ms after V_{CC} reaches 4.75 V. It is suggested that $\overline{\text{RESET IN}}$ be kept low during this period. Note that the 10 ms period does not include the time it takes for the power supply to reach its 4.75 V level.

INSTRUCTION SET

Symbols and Abbreviations

SYMBOLS	DEFINITION
ddd,sss	The bit pattern designating one of the registers A,B,C,D,E,H,L (ddd=destination, sss=source):

ddd or sss	REGISTER NAME
111	A
000	B
001	C
010	D
011	E
100	H
101	L
110	M (Memory)

r,r1,r2	One of the registers A,B,C,D,E,H,L
d8	8-bit data quantity
d16	16-bit data quantity
addr8	8-bit address of an I/O device
addr	16-bit address quantity
RP	The bit pattern designating one of the register pairs B,D,H,SP:

RP	rp	REGISTER PAIR (rpH)(rpL)
00	B	B-C
01	D	D-E
10	H	H-L
11	SP	SP

B_2	The second byte of the instruction
B_3	The third byte of the instruction
0	Affected
S	Set
R	Reset
-	Not affected

Data Transfer

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
MOV r1, r2	0	1	d	d	d	S	S	S	(r1) ← (r2)	1	4	-	-	-	-	-
MOV M, r	0	1	1	1	0	S	S	S	[(H)(L)] ← (r)	1	7	-	-	-	-	-
MOV r, M	0	1	d	d	d	1	1	0	(r) ← [(H)(L)]	1	7	-	-	-	-	-
MVI r, d8	0	0	d	d	d	1	1	0	(r) ← (B_2)	2	7	-	-	-	-	-
					B_2											
MVI M, d8	0	0	1	1	0	1	1	0	[(H)(L)] ← (B_2)	2	10	-	-	-	-	-
					B_2											
LDA addr	0	0	1	1	1	0	1	0	(A) ← [(B_3)(B_2)]	3	13	-	-	-	-	-
					B_2											
					B_3											
LDAX B	0	0	0	0	1	0	1	0	(A) ← [(B)(C)]	1	7	-	-	-	-	-
LDAX D	0	0	0	1	1	0	1	0	(A) ← [(D)(E)]	1	7	-	-	-	-	-
LHLD addr	0	0	1	0	1	0	1	0	(L) ← [(B_3)(B_2)]	3	16	-	-	-	-	-
					B_2				(H) ← [(B_3)(B_2)+1]							
					B_3											
LXI H, d16	0	0	1	0	0	0	0	1	(H) ← (B_3)	3	10	-	-	-	-	-
					B_2				(L) ← (B_2)							
					B_3											
LXI D, d16	0	0	0	1	0	0	0	1	(D) ← (B_3)	3	10	-	-	-	-	-
					B_2				(E) ← (B_2)							
					B_3											
LXI B, d16	0	0	0	0	0	0	0	1	(B) ← (B_3)	3	10	-	-	-	-	-
					B_2				(C) ← (B_2)							
					B_3											
LXI SP, d16	0	0	1	1	0	0	0	1	$(SP)_H$ ← (B_3)	3	10	-	-	-	-	-
					B_2				$(SP)_L$ ← (B_2)							
					B_3											
SHLD addr	0	0	1	0	0	0	1	0	[(B_3)(B_2)] ← (L)	3	16	-	-	-	-	-
					B_2				[(B_3)(B_2)+1] ← (H)							
					B_3											
STA addr	0	0	1	1	0	0	1	0	[(B_3)(B_2)] ← (A)	3	13	-	-	-	-	-
					B_2											
					B_3											

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
STAX B	0	0	0	0	0	0	1	0	[(B)(C)] ← (A)	1	7	-	-	-	-	-
STAX D	0	0	0	1	0	0	1	0	[(D)(E)] ← (A)	1	7	-	-	-	-	-
SPHL	1	1	1	1	1	0	0	1	(SP) ← (H)(L)	1	6	-	-	-	-	-
XCHG	1	1	1	0	1	0	1	1	(H) ← (D) (L) ← (E)	1	4	-	-	-	-	-
XTHL	1	1	1	0	0	0	1	1	(L) ← [(SP)] (H) ← [(SP)+1]	1	16	-	-	-	-	-
IN addr8	1	1	0	1	1	0	1	1	(A) ← (data)	2	10	-	-	-	-	-
					B_2											
OUT addr8	1	1	0	1	0	0	1	1	(data) ← (A)	2	10	-	-	-	-	-
					B_2											

Branch

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
JMP addr	1	1	0	0	0	0	1	1	(PC) ← $(B_3)(B_2)$	3	10	-	-	-	-	-
					B_2											
					B_3											
JNZ addr	1	1	0	0	0	0	1	0	If Z = 0	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If Z = 1							
									(PC) ← (PC) + 3							
JZ addr	1	1	0	0	1	0	1	0	If Z = 1	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If Z = 0							
									(PC) ← (PC) + 3							
JNC addr	1	1	0	1	0	0	1	0	If C = 0	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If C = 1							
									(PC) ← (PC) + 3							
JC addr	1	1	0	1	1	0	1	0	If C = 1	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If C = 0							
									(PC) ← (PC) + 3							

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
JPO addr	1	1	1	0	0	0	1	0	If P = 0	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If P = 1							
									(PC) ← (PC) + 3							
JPE addr	1	1	1	0	1	0	1	0	If P = 1	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If P = 0							
									(PC) ← (PC) + 3							
JP addr	1	1	1	1	0	0	1	0	If S = 0	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If S = 1							
									(PC) ← (PC) + 3							
JM addr	1	1	1	1	1	0	1	0	If S = 1	3	7/10	-	-	-	-	-
					B_2				(PC) ← $(B_3)(B_2)$,							
					B_3				If S = 0							
									(PC) ← (PC) +3							
CALL addr	1	1	0	0	1	1	0	1	[(SP)-1] ← (PCH)	3	18	-	-	-	-	-
					B_2				[(SP)-2] ← (PCL)							
					B_3				(SP) ← (SP) - 2							
									(PC) ← $(B_3)(B_2)$							
CNZ addr	1	1	0	0	0	1	0	0	If Z = 0, the actions specified in the CALL instruction are performed.	3	9/18	-	-	-	-	-
					B_2											
					B_3											
									If Z = 1							
									(PC) ← (PC) + 3							
CZ addr	1	1	0	0	1	1	0	0	If Z = 1, the actions specified in the CALL instruction are performed.	3	9/18	-	-	-	-	-
					B_2											
					B_3											
									If Z = 0,							
									(PC) ← (PC) + 3							

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
CNC addr	1	1	0	1	0	1	0	0	If C = 0, the actions specified in the CALL instruction are performed. If C = 1 (PC) ← (PC) + 3	3	9/18	-	-	-	-	-
				B_2												
				B_3												
CC addr	1	1	0	1	1	1	0	0	If C = 1, the actions specified in the CALL instruction are performed. If C = 0 (PC) ← (PC) + 3	3	9/18	-	-	-	-	-
				B_2												
				B_3												
CPO addr	1	1	1	0	0	1	0	0	If P = 0, the actions specified in the CALL instruction are performed. If P = 1 (PC) ← (PC) + 3	3	9/18	-	-	-	-	-
				B_2												
				B_3												
CPE addr	1	1	1	0	1	1	0	0	If P = 1, the actions specified in the CALL instruction are performed. If P = 0 (PC) ← (PC) + 3	3	9/18	-	-	-	-	-
				E_2												
				B_3												
CP addr	1	1	1	1	0	1	0	0	If S = 0, the actions specified in the CALL instruction are performed. If S = 1 (PC) ← (PC) + 3	3	9/18	-	-	-	-	-
				B_2												
				B_3												

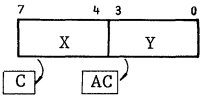
Mnemonic	Instruction Code								Operation	Bytes	States	Flag
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C Z S P AC
CM addr	1	1	1	1	1	1	0	0	If S = 1, the actions specified in the CALL instruction are performed. If S = 0 (PC) ← (PC) + 3	3	9/18	- - - - -
				B_2								
				B_3								
RET	1	1	0	0	1	0	0	1	(PCL) ← [(SP)] (PCH) ← [(SP)+1] (SP) ← (SP) + 2	1	10	- - - - -
RNZ	1	1	0	0	0	0	0	0	If Z = 0, the actions specified in the RET instruction are performed. If Z = 1 (PC) ← (PC) + 1	1	6/12	- - - - -
RZ	1	1	0	0	1	0	0	0	If Z = 1, the actions specified in the RET instruction are performed. If Z = 0 (PC) ← (PC) + 1	1	6/12	- - - - -
RNC	1	1	0	1	0	0	0	0	If C = 0, the actions specified in the RET instruction are performed. If C = 1 (PC) ← (PC) + 1	1	6/12	- - - - -

Mnemonic	Instruction Code								Operation	Bytes	States	Flag					
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
RC	1	1	0	1	1	0	0	0	If C = 1, the actions specified in the RET instruction are performed. If C = 0 (PC) ← (PC) + 1	1	6/12	-	-	-	-	-
RPO	1	1	1	0	0	0	0	0	If P = 0, the actions specified in the RET instruction are performed. If P = 1 (PC) ← (PC) + 1	1	6/12	-	-	-	-	-
RPE	1	1	1	0	1	0	0	0	If P = 1, the actions specified in the RET instruction are performed. If P = 0 (PC) ← (PC) + 1	1	6/12	-	-	-	-	-
RP	1	1	1	1	0	0	0	0	If S = 0, the actions specified in the RET instruction are performed. If S = 1 (PC) ← (PC) + 1	1	6/12	-	-	-	-	-
RM	1	1	1	1	1	0	0	0	If S = 1, the actions specified in the RET instruction are performed. If S = 0 (PC) ← (PC) + 1	1	6/12	-	-	-	-	-

Mnemonic	Instruction Code								Operation	Bytes	States	Flag					
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
PCHL	1	1	1	0	1	0	0	1	(PCH) ← (H) (PCL) ← (L)	1	6	-	-	-	-	-
RST	1	1	A	A	A	1	1	1	[(SP)-1] ← (PCH) [(SP)-2] ← (PCL) (SP) ← (SP) - 2 (PC) ← (00000000 00AAA000)	1	12	-	-	-	-	-

Arithmetic

Mnemonic	Instruction Code								Operation	Bytes	States	Flag					
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
ADD r	1	0	0	0	0	S	S	S	(A) ← (A) + (r)	1	4	0	0	0	0	0
ADC r	1	0	0	0	1	S	S	S	(A) ← (A) + (r) + (C)	1	4	0	0	0	0	0
ADD M	1	0	0	0	0	1	1	0	(A) ← (A)+[(H)(L)]	1	7	0	0	0	0	0
ADC M	1	0	0	0	1	1	1	0	(A)←(A)+[(H)(L)]+(C)	1	7	0	0	0	0	0
ADI d8	1	1	0	0	0	1	1	0 B_2	(A) ← (A) + (B_2)	2	7	0	0	0	0	0
ACI d8	1	1	0	0	1	1	1	0 B_2	(A) ← (A)+(B_2)+(C)	2	7	0	0	0	0	0
DAD rp	0	0	R	P	1	0	0	1	(H)(L) ← (H)(L) + (rH)(rL)	1	10	0	-	-	-	-
SUB r	1	0	0	1	0	S	S	S	(A) ← (A) - (r)	1	4	0	0	0	0	0
SBB r	1	0	0	1	1	S	S	S	(A) ← (A) - (r) - (C)	1	4	0	0	0	0	0
SUB M	1	0	0	1	0	1	1	0	(A) ← (A)-[(H)(L)]	1	7	0	0	0	0	0
SBB M	1	0	0	1	1	1	1	0	(A)←(A)-[(H)(L)]-(C)	1	7	0	0	0	0	0
SUI d8	1	1	0	1	0	1	1	0 B_2	(A) ← (A) - (B_2)	2	7	0	0	0	0	0
SBI d8	1	1	0	1	1	1	1	0 B_2	(A) ← (A) - (B_2) - (C)	2	7	0	0	0	0	0

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
DAA	0	0	1	0	0	1	1	1	The 8-bit number in the accumulator is adjusted to form two 4-bit BCD digits by the following process. Accumulator  1. If $Y \geq 10$ or AC=1, $(A) \leftarrow (A) + 6$ 2. If $X \geq 10$ or C=1, $(A)_{4-7} \leftarrow (A)_{4-7} + 6$	1	4	0	0	0	0	0

Logical Instruction

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
ANA r	1	0	1	0	0	S	S	S	(A) ← (A) ∧ (r)	1	4	R	0	0	0	S
ANA M	1	0	1	0	0	1	1	0	(A) ← (A) ∧ [(H)(L)]	1	7	R	0	0	0	S
ANI d8	1	1	1	0	0	1	1	0	(A) ← (A) ∧ (B_2)	2	7	R	0	0	0	S
				B_2												
XRA r	1	0	1	0	1	S	S	S	(A) ← (A) ∀ (r)	1	4	R	0	0	0	R
XRA M	1	0	1	0	1	1	1	0	(A) ← (A) ∀ [(H)(L)]	1	7	R	0	0	0	R
XRI d8	1	1	1	0	1	1	1	0	(A) ← (A) ∀ (B_2)	2	7	R	0	0	0	R
				B_2												
ORA r	1	0	1	1	0	S	S	S	(A) ← (A) ∨ (r)	1	4	R	0	0	0	R
ORA M	1	0	1	1	0	1	1	0	(A) ← (A) ∨ [(H)(L)]	1	7	R	0	0	0	R
ORI d8	1	1	1	1	0	1	1	0	(A) ← (A) ∨ (B_2)	2	7	R	0	0	0	R
				B_2												
CMP r	1	0	1	1	1	S	S	S	(A) − (r)	1	4	0	0	0	0	0
CMP M	1	0	1	1	1	1	1	0	(A) − [(H)(L)]	1	7	0	0	0	0	0
CPI d8	1	1	1	1	1	1	1	0	(A) − (B_2)	2	7	0	0	0	0	0
				B_2												
CMA	0	0	1	0	1	1	1	1	(A) ← $(\overline{A})$	1	4	-	-	-	-	-
RLC	0	0	0	0	0	1	1	1	(A_{n+1}) ← (An) (A_0) ← (A_7) (C) ← (A_7)	1	4	0	-	-	-	-
RRC	0	0	0	0	1	1	1	1	(An) ← (A_{n+1}) (A_7) ← (A_0) (C) ← (A_0)	1	4	0	-	-	-	-
RAL	0	0	0	1	0	1	1	1	(A_{n+1}) ← (An) (C) ← (A_7) (A_0) ← (C)	1	4	0	-	-	-	-
RAR	0	0	0	1	1	1	1	1	(An) ← (A_{n+1}) (C) ← (A_0) (A_7) ← (C)	1	4	0	-	-	-	-

Increment and Decrement

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
INR r	0	0	d	d	d	1	0	0	(r) ← (r) + 1	1	4	-	O	O	O	O
INR M	0	0	1	1	0	1	0	0	[(H)(L)] ← [(H)(L)]+1	1	10	-	O	O	O	O
INX rp	0	0	R	P	0	0	1	1	(rH)(rL) ← (rH)(rL)+1	1	6	-	-	-	-	-
DCR r	0	0	d	d	d	1	0	1	(r) ← (r) - 1	1	4	-	O	O	O	O
DCR M	0	0	1	1	0	1	0	1	[(H)(L)] ← [(H)(L)]-1	1	10	-	O	O	O	O
DCX rp	0	0	R	P	1	0	1	1	(rH)(rL) ← (rH)(rL)-1	1	6	-	-	-	-	-

Stack

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
PUSH rp	1	1	R	P	0	1	0	1	[(SP)-1] ← (rH) [(SP)-2] ← (rL) (SP) → (SP) - 2 Note: Register pair rp=SP may not be specified.	1	12	-	-	-	-	-
PUSH PSW	1	1	1	1	0	1	0	1	[(SP)-1] ← (A) [(SP)-2] ← $D_7D_6D_5D_4D_3D_2D_1D_0$ S Z X AC X P X C MSB (SP) ← (SP) - 2	1	12	-	-	-	-	-
POP rp	1	1	R	P	0	0	0	1	(rL) ← [(SP)] (rH) ← [(SP)+1] (SP) ← (SP)+2	1	10	-	-	-	-	-
POP PSW	1	1	1	1	0	0	0	1	(C) ← $[(SP)]_0$ (P) ← $[(SP)]_2$ (AC) ← $[(SP)]_4$ (Z) ← $[(SP)]_6$ (S) ← $[(SP)]_7$ (A) ← [(SP)+1] (SP) ← (SP) + 2	1	10	O	O	O	O	O

Control

Mnemonic	Instruction Code								Operation	Bytes	States	Flag				
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				C	Z	S	P	AC
HLT	0	1	1	1	0	1	1	0	Halt	1	5	-	-	-	-	-
STC	0	0	1	1	0	1	1	1	(C) ← 1	1	4	0	-	-	-	-
CMC	0	0	1	1	1	1	1	1	(C) ← $(\overline{C})$	1	4	0	-	-	-	-
EI	1	1	1	1	1	0	1	1	Enable interrupts Note: Interrupts are not recognized during the EI instruction.	1	4	-	-	-	-	-
DI	1	1	1	1	0	0	1	1	Disable interrupts Note: Interrupts are not recognized during the DI instruction.	1	4	-	-	-	-	-
NOP	0	0	0	0	0	0	0	0	No operation is performed.	1	4	-	-	-	-	-
RIM	0	0	1	0	0	0	0	0	(A) ← d_7 = SID d_6 = I7 d_5 = I6 d_4 = I5 d_3 = IE d_2 = M7 d_1 = M6 d_0 = M5	1	4	-	-	-	-	-
SIM	0	0	1	1	0	0	0	0	IF(A)$_6$ = 1; SOD ← (A)$_7$,IF(A)$_3$ = 1; M7 ← (A)$_2$ M6 ← (A)$_1$ M5 ← (A)$_0$,IF(A)$_4$ = 1; RST7.5 RESET	1	4	-	-	-	-	-

ABSOLUTE MAXIMUM RATINGS

Symbol	Item	Ratings
V_{CC}	V_{CC} Supply Voltage	-0.5V to 7.0V
V_{IN}	Input Voltage with Respect to V_{SS}	-0.5V to 7.0V
V_{OUT}	Output Voltage with Respect to V_{SS}	-0.5V to 7.0V
P_D	Power Dissipation	1.5W
T_{solder}	Soldering Temperature (Soldering Time 10 sec.)	260°C
T_{stg}	Storage Temperature	-55°C to 150°C
T_{opr}	Operating Temperature	0°C to 70°C

DC CHARACTERISTICS

T_A = 0°C to 70°C, V_{CC} = 5V±5%

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Units
V_{IL}	Input Low Voltage		-0.5		0.8	V
V_{IH}	Input High Voltage		2.0		V_{CC} +0.5	V
V_{OL}	Output Low Voltage	I_{OL} = 2mA			0.45	V
V_{OH}	Output High Voltage	I_{OH} = -400μA	2.4			V
I_{CC}	Power Supply Current				170	mA
I_{IL}	Input Leakage	$V_{IN} = V_{CC}$			±10	μA
I_{LO}	Output Leakage	$0.45 \leq V_{OUT} \leq V_{CC}$			10	μA
V_{ILR}	Input Low Level (RESET)		-0.5		0.8	V
V_{IHR}	Input High Level (RESET)		2.4		V_{CC} +0.5	V
V_{HY}	Hysteresis (RESET)		0.25			V

AC CHARACTERISTICS

T_A = 0°C to 70°C, V_{CC} = 5V ± 5%, V_{SS} = 0V, Unless Otherwise Noted.

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Units
t_{CYC}	CLK Cycle Period		320		2000	ns
t_L	CLK Low Time - Standard 150pF Loading - Lightly Loaded [2]		80 100			ns ns
t_H	CLK High Time - Standard 150pF Loading - Lightly Loaded[2]		120 150			ns ns
t_r, t_f	CLK Rise and Fall Time				30	ns
t_{XKR}	X_1 Rising to CLK Rising		30		120	ns
t_{XKF}	X_1 Rising to CLK Falling		30		150	ns
t_{AC}	A_{8-15} Valid to Leading Edge of Control[1]		270			ns
t_{ACL}	A_{0-7} Valid to Leading of Control		240			ns
t_{AD}	A_{0-15} Valid to Valid Data In				575	ns
t_{AFR}	Address Float after Leading Edge of $\overline{READ}$ ($\overline{INTA}$)				0	ns
t_{AL}	A8-15 Valid before Trailing Edge of ALE[1]	C_L=150pF	115			ns
t_{ALL}	A0-7 Valid before Trailing Edge of ALE		90			ns
t_{ARY}	READY Valid from Address Valid	t_{CYC}=320ns			220	ns
t_{CA}	Address (A8 - A15) Valid after Control		120			ns
t_{CC}	Width of Control Low ($\overline{RD}$, $\overline{WR}$, $\overline{INTA}$) Edge of ALE		400			ns
t_{CL}	Trailing Edge of Control to Leading Edge of ALE		50			ns
t_{DW}	Data Valid to Trailing Edge of $\overline{WRITE}$		420			ns
t_{HABE}	HLDA to Bus Enable				210	ns
t_{HABF}	Bus Float after HLDA				210	ns
t_{HACK}	HLDA Valid to Trailing Edge of CLK		110			ns
t_{HDH}	HOLD Hold Time		0			ns
t_{HDS}	HOLD Setup Time to Trailing Edge of CLK		170			ns
t_{INH}	INTR Hold Time		0			ns

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Units
t_{INS}	INTR, RST and TRAP Setup Time to Falling Edge of CLK		160			ns
t_{LA}	Address Hold Time after ALE		100			ns
t_{LC}	Trailing Edge of ALE to Leading Edge of Control		130			ns
t_{LCK}	ALE Low during CLK High		100			ns
t_{LDR}	ALE to Valid Data during Read				460	ns
t_{LDW}	ALE to Valid Data during Write				200	ns
t_{LL}	ALE Width		140			ns
t_{LRY}	ALE to READY Stable				110	ns
t_{RAE}	Trailing Edge of $\overline{READ}$ to Re-Enabling of Address		150			ns
t_{RD}	$\overline{READ}$ (or $\overline{INTA}$) to Valid Data				300	ns
t_{RV}	Control Trailing Edge to Leading Edge of Next Control		400			ns
t_{RDH}	Data Hold Time After $\overline{READ}$ $\overline{INTA}$		0			ns
t_{RYH}	READY Hold Time		0			ns
t_{RYS}	READY Setup Time to Leading Edge of CLK		110			ns
t_{WD}	Data Valid After Trailing Edge of $\overline{WRITE}$		100			ns
t_{WDL}	LEADING Edge of WRITE to Data Valid				40	ns

Notes:
1. A_{8-15} address specs apply to IO/$\overline{M}$, S_0 and S_1 except A_{8-15} are undifined during T_4 - T_6 of OF cycle whereas IO/$\overline{M}$, S_0, and S_1 are stable.
2. Loading equivalent to 50 pF + 1 TTL input.
3. All timings are measured at output voltage V_L = 0.8 V, V_H = 2.0 V.
4. To calculate timing specifications at other value of t_{CYC} use Table 4.

TABLE 4. BUS TIMING SPECIFICATION AS A T_{CYC} DEPENDENT

t_{AL}	(1/2) T - 45	MIN
t_{LA}	(1/2) T - 60	MIN
t_{LL}	(1/2) T - 20	MIN
t_{LCK}	(1/2) T - 60	MIN
t_{LC}	(1/2) T - 30	MIN
t_{AD}	(5/2 + N) T - 225	MAX
t_{RD}	(3/2 + N) T - 180	MAX
t_{RAE}	(1/2) T - 10	MIN
t_{CA}	(1/2) T - 40	MIN
t_{DW}	(3/2 + N) T - 60	MIN
t_{WD}	(1/2) T - 60	MIN
t_{CC}	(3/2 + N) T - 80	MIN
t_{CL}	(1/2) T - 110	MIN
t_{ARY}	(3/2) T - 260	MAX
t_{HACK}	(1/2) T - 50	MIN
t_{HABF}	(1/2) T + 50	MAX
t_{HABE}	(1/2) T + 50	MAX
t_{AC}	(2/2) T - 50	MIN
t_{L}	(1/2) T - 80	MIN
t_{H}	(1/2) T - 40	MIN
t_{RV}	(3/2) T - 80	MIN
t_{LDR}	(4/2) T - 180	MAX

Note: N is equal to the total WAIT states.

$T = t_{CYC}$

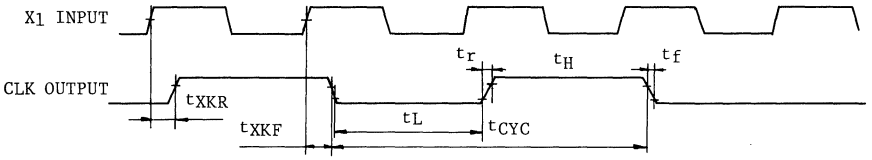


FIGURE 4. CLOCK TIMING WAVEFORM

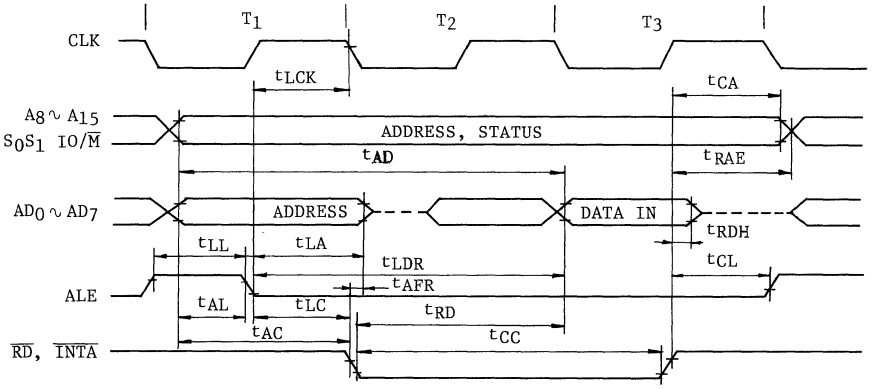


FIGURE 5. READ OPERATION

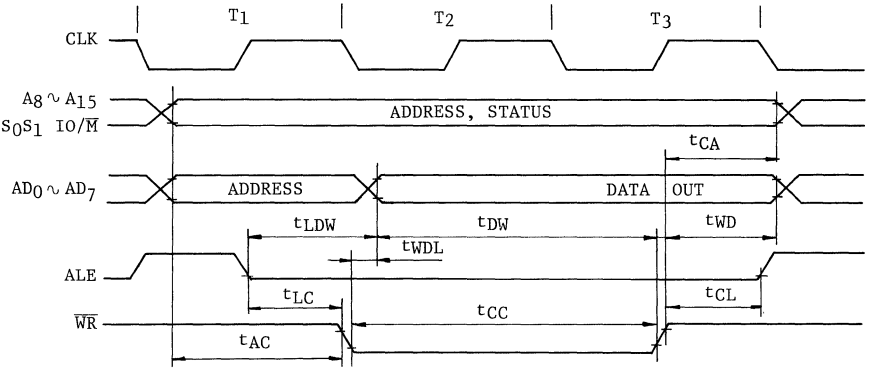


FIGURE 6. WRITE OPERATION

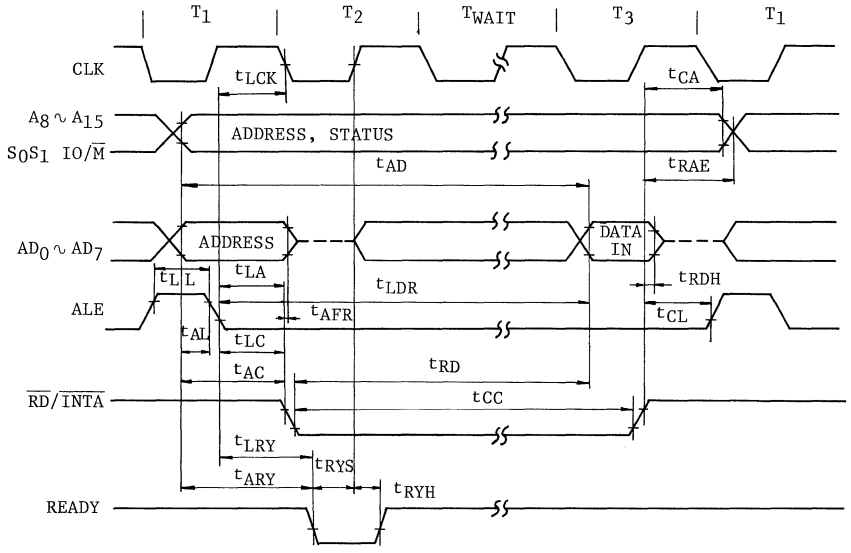


FIGURE 7. READ OPERATION WITH WAIT CYCLE (TYPICAL)
- SAME READY TIMING APPLIES TO WRITE OPERATION

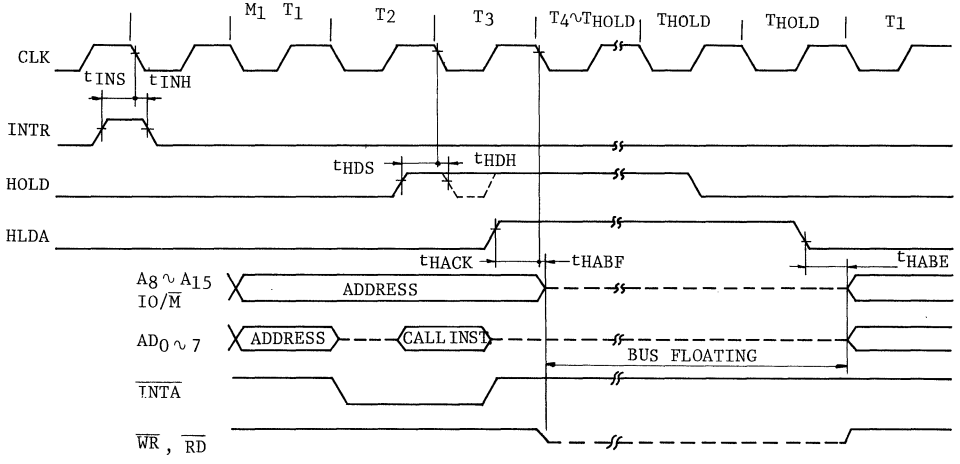
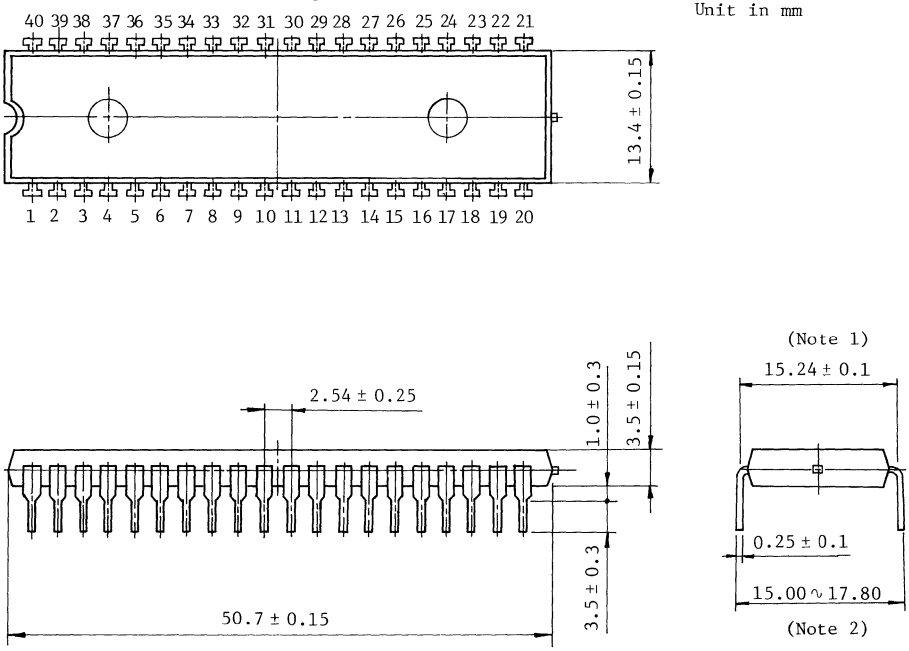


FIGURE 8. INTERRUPT AND HOLD TIMING

OUTLINE DRAWING

40 Pins Plastic Package

Unit in mm



Notes: 1. This dimension shows the center of curvature of leads
2. This dimension shows spread of leads.
3. All dimensions are in millimeters.