intel®

8080

SINGLE CHIP
EIGHT-BIT PARALLEL
CENTRAL PROCESSOR UNIT

APRIL 1974

CONTENTS

APPENDICES

1. INTRODUCTION

The 8080 is a complete 8-bit parallel central processing unit (CPU) for use in general purpose digital computer systems. It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process, thus offering much higher performance than conventional microprocessors (2μs instruction cycle). A complete micro computer system is formed when the 8080 CPU is interfaced with I/O ports (up to 256 input and 256 output ports) and any type or speed of semi-conductor memory.

Although significantly higher in performance than existing microprocessors, the 8080 has been designed to be software compatible at the source code level with Intel's 8008 micro-processor. Like the 8008, the 8080 contains six 8-bit data registers, an 8-bit accumulator, four 8-bit temporary registers, four testable flag bits, and an 8-bit parallel binary arithmetic unit. The 8080 also provides decimal arithmetic capability, and it includes sixteen bit arithmetic and immediate operators which greatly simplify memory address calculations, and high speed arithmetic operations.

The 8080 has a stack architecture wherein any portion of the external memory can be used as a last in/first out stack to store/retrieve the contents of the accumulator, the flags, or any of the data registers.

The 8080 also contains a 16-bit stack pointer to control the addressing of this external stack. One of the major advantages of the stack is that multiple level interrupts can easily be handled since complete system status can easily be saved when an interrupt occurs and then be restored after the interrupt. Another major advantage is that almost unlimited subroutine nesting is possible.

This processor has been designed to greatly simplify system design. Separate 16-line address and 8-line bidirectional data busses are used to allow direct interface to memories and I/O ports. Control signals, which require no decoding, are provided directly by the processor. All busses, including control, are TTL compatible.

Communication on the address lines and the data lines can be interlocked by using the HOLD input. When the HLDA (Hold Acknowledge) signal is issued by the CPU, CPU operation is suspended and the address and data lines are forced to be in the FLOATING state. This permits "OR-tying" the address and data busses with other devices such as direct memory access channels (DMA).

The 8080 has many instructions which are extremely useful and extend the range of applicability of the CPU. The instruction groups are as follows:

- Data register and memory transfers
- Conditional or unconditional branches and subroutine calls
- I/O operations
- Direct Load/Store Accumulator
- Save, Restore Data Registers, Accumulator and Flags
- Double Length Operation in Data Registers
  - Increment/Decrement/Addition
  - Direct Load/Store (H and L)
  - Load Immediate
  - Index Register Modification
- Indirect Jump
- Stack Pointer Modification
- Logical Operations
- Binary Arithmetic
- Decimal Arithmetic
- Set and reset interrupt enable flip-flop
- Increment/Decrement Memory or data registers

The purpose of this publication is to present the basic microprocessor operation, instruction set, and electrical characteristics. in addition, other memory and peripheral circuits which have been designed, and specified for use with the 8080 are presented.

8080 ADDRESSING MODES:

- DIRECT
- REGISTER
- REGISTER INDIRECT
- IMMEDIATE

2. PROCESSOR TIMING

The following describes the function of all of the 8080 I/O pins. Several of the descriptions refer to internal timing periods. For a definition of the timing periods refer to section 2-3.

2-1. Pin Configuration and Control Signal

The pin configuration is shown in Figure 1.

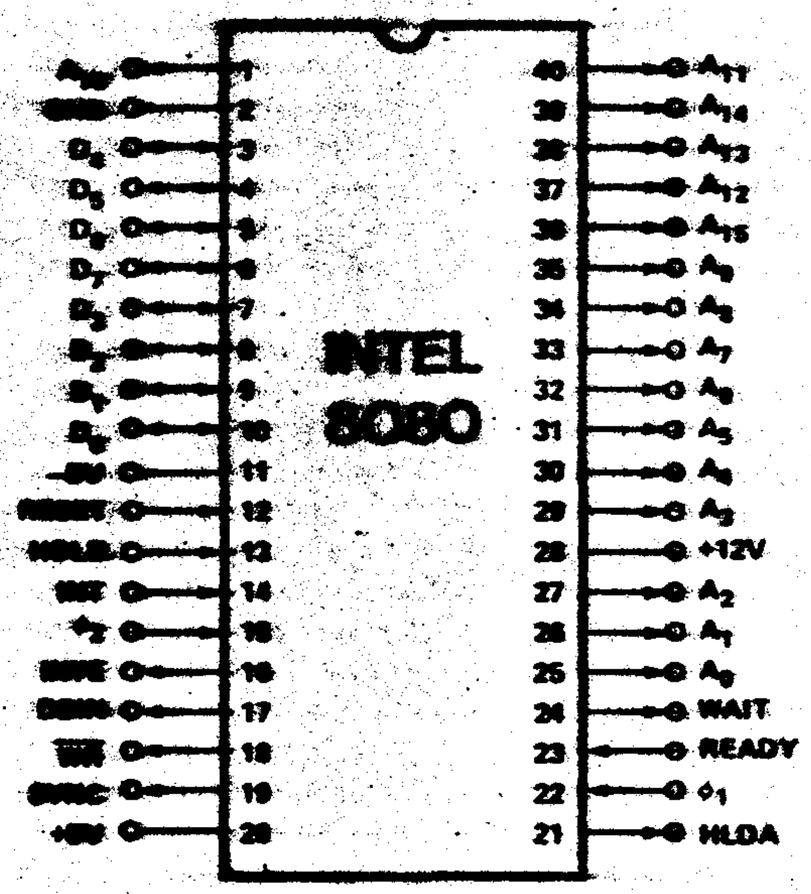


Figure 1. Pin Configuration

PIN DEFINITION

Symbols	Meaning
A_{15}-A_0 (output tri-state)	ADDRESS BUS; the address bus provides the address to memory (up to 64K 8-bit words) or denotes the I/O device number for up to 256 input and 256 output devices. A_0 is the least significant address bit.
D_7-D_0 (input/output) tri-state	DATA BUS; the data bus provides bidirectional communication between memory and I/O devices for instructions and data transfers. D_0 is the least significant bit.
SYNC (output)	SYNCHRONIZING SIGNAL; the SYNC pin provides a signal to indicate the beginning of each machine cycle. (Instructions can be executed in 1, 2, 3, 4 or 5 machine cycles, and the status information of each machine cycle is sent to external latches at SYNC time.) See 2-2.
DBIN (output)	DATA BUS IN; the DBIN signal indicates to external circuits that the data bus is in the input mode. This signal should be used to enable the gating of data onto the 8080 data bus from memory or I/O.
READY (input)	READY; the READY signal indicates to the 8080 that valid memory or input data is available on the 8080 data bus. This signal is used to synchronize the CPU with slower memory or I/O devices. If after sending an address out the 8080 does not receive a READY input, the 8080 will enter a WAIT state for as long as the READY line is low.
WAIT (output)	WAIT; the WAIT signal acknowledges that the CPU is in a WAIT state.
$\overline{WR}$ (output)	WRITE; the $\overline{WR}$ signal is used for memory WRITE or I/O output control. The data on the data bus is stable while the $\overline{WR}$ signal is active ($\overline{WR}$ = 0).
HOLD (input)	HOLD; the HOLD signal requests the CPU to enter the HOLD state. The HOLD state allows an external device to gain control of the 8080 address and data bus as soon as the 8080 has completed its use of these buses for the current machine cycle. It is recognized under the following conditions: • the CPU is in the HALT state • the CPU is in the T2 or TW state and the READY signal is active As a result of entering the HOLD state the CPU ADDRESS BUS (A_{15}-A_0) and DATA BUS (D_7-D_0) will be in their high impedance state. The CPU acknowledges its state with the HOLD ACKNOWLEDGE (HLDA) pin. See Figure 3. The CPU will always finish the execution of the current machine cycle. When the HOLD signal is removed, the operation will resume from the T1 time of the next machine cycle. (See attached timing charts, Figures 3 and h in Appendix III.)

Symbols	Meaning
HLDA (output)	HOLD ACKNOWLEDGE; the HLDA signal appears in response to the HOLD signal and indicates that the data and address bus will go to the high impedance state. The HLDA signal begins at: T3 for READ memory or input The Clock Period following T3 for WRITE memory or OUTPUT operation In either case, the HLDA signal appears after the rising edge of ϕ_1 and high impedance occurs after the rising edge of ϕ_2. See Appendix III-f for timing diagram and $\overline{WO}$ status information for mode determination.
INTE (output)	INTERRUPT ENABLE; indicates the content of the internal interrupt enable flip/flop. This flip/flop may be set or reset by the EI and DI instructions and inhibits interrupts from being accepted by the CPU if it is reset. It is automatically reset (disabling further interrupts) at time T1 of the instruction fetch cycle (M1) when an interrupt is accepted and is also reset by the RESET signal.
INT (input)	INTERRUPT REQUEST; the CPU recognizes an interrupt request on this line at the end of the current instruction or while halted. If the CPU is in the HOLD state or if the Interrupt Enable flip/flop is reset it will not honor the request. The CPU acknowledges acceptance of an interrupt by sending out the INTA (Interrupt Acknowledge) status signal at SYNC time. During the next instruction fetch cycle the program counter is not advanced and a 1 byte instruction (usually RESTART) can be inserted. See Appendix I.
RESET (input)	RESET; while the RESET signal is activated, the content of the program counter is cleared and the instruction register is set to 0. After RESET, the program will start at location 0 in memory. The INTE and HLDA flip/flops are also reset. Note that the flags, accumulator, and registers are not cleared as with the 8008. The HL and DE registers may be exchanged.
V_{ss}	Ground Reference.
V_{dd}	+ 12 ± 5% Volts
V_{cc}	+ 5 ± 5% Volts
V_{bb}	−5 ± 5% Volts (substrate bias)
ϕ_1, ϕ_2	2 externally supplied clock phases. (non TTL compatible)

2-2. Status Information

Instructions for the 8080 require from one to five machine cycles for complete execution. The 8080 sends out 8 bit of status information on the data bus at the beginning of each machine cycle (during SYNC time). The following table defines the status information.

STATUS INFORMATION DEFINITION

Symbols	Data Bus Bit	Definition
HLTA	D_3	Acknowledge signal for HALT instruction.
INTA*	D_0	Acknowledge signal for INTERRUPT request. Signal should be used to gate a restart instruction onto the data bus when DBIN is active.
INP*	D_6	Indicates that the address bus contains the address of an input device and the input data should be placed on the data bus when DBIN is active.
OUT	D_4	Indicates that the address bus contains the address of an output device and the data bus will contain the output data when $\overline{WR}$ is active.
MEMR*	D_7	Designates that the data bus will be used for memory read data.
M_1	D_5	Provides a signal to indicate that the CPU is in the fetch cycle for the first byte of an instruction.
STACK	D_2	Indicates that the address bus holds the pushdown stack address from the Stack Pointer.
$\overline{WO}$	D_1	Indicates that the operation in the current machine cycle will be a WRITE memory or OUTPUT function ($\overline{WO}$ = 0). Otherwise, a READ memory or INPUT operation will be executed.

*These three status bits can be used to control the flow of data onto the 8080 data bus.

2-3. Timing

Instructions in the 8080 contain one to three bytes. Each instruction requires from one to five machine or memory cycles for fetching and execution. Machine cycles are called M1, M2, . . . , M5. Each machine cycle requires from three to five states T1, T2, . . . , T5 for its completion. Each state has the duration of one clock period (0.5 micro-second). There are three other states (WAIT, HOLD, and HALT) which last one to an indefinite number of clock periods, as controlled by external signals. Machine cycle M1 is always the operation-code fetch cycle and lasts four or five clock periods. Machine cycles M2, M3, M4, and M5 normally last three clock periods each.

To understand the basic operation of the 8080, refer to the simplified state diagram shown in Figure 3 and the timing diagram of Figure 2.

During T1 the content of the program counter is sent to the address bus, SYNC is true, and the data bus contains the status information pertaining to the cycle that is currently being initiated. T1 is always followed by another state, T2, during which the condition of the READY, HOLD and HALT Acknowledge Signals are tested. If READY is true, T3 can be entered; otherwise, the CPU will go into the wait state (TW) and stay there for as long as READY is false.

READY thus allows the CPU speed to be synchronized to a memory with any access time or to any input device. Furthermore, by properly controlling the READY line, the user can single-step through his program.

During T3, the data coming from memory is available on the data bus and is transferred into the instruction register (during M1 only) as shown in the 8080 block diagram of Figure 4. The instruction decoder and control sections then generate the basic signals to control the internal data transfers, the timing, and the machine cycle requirements of the new instructions.

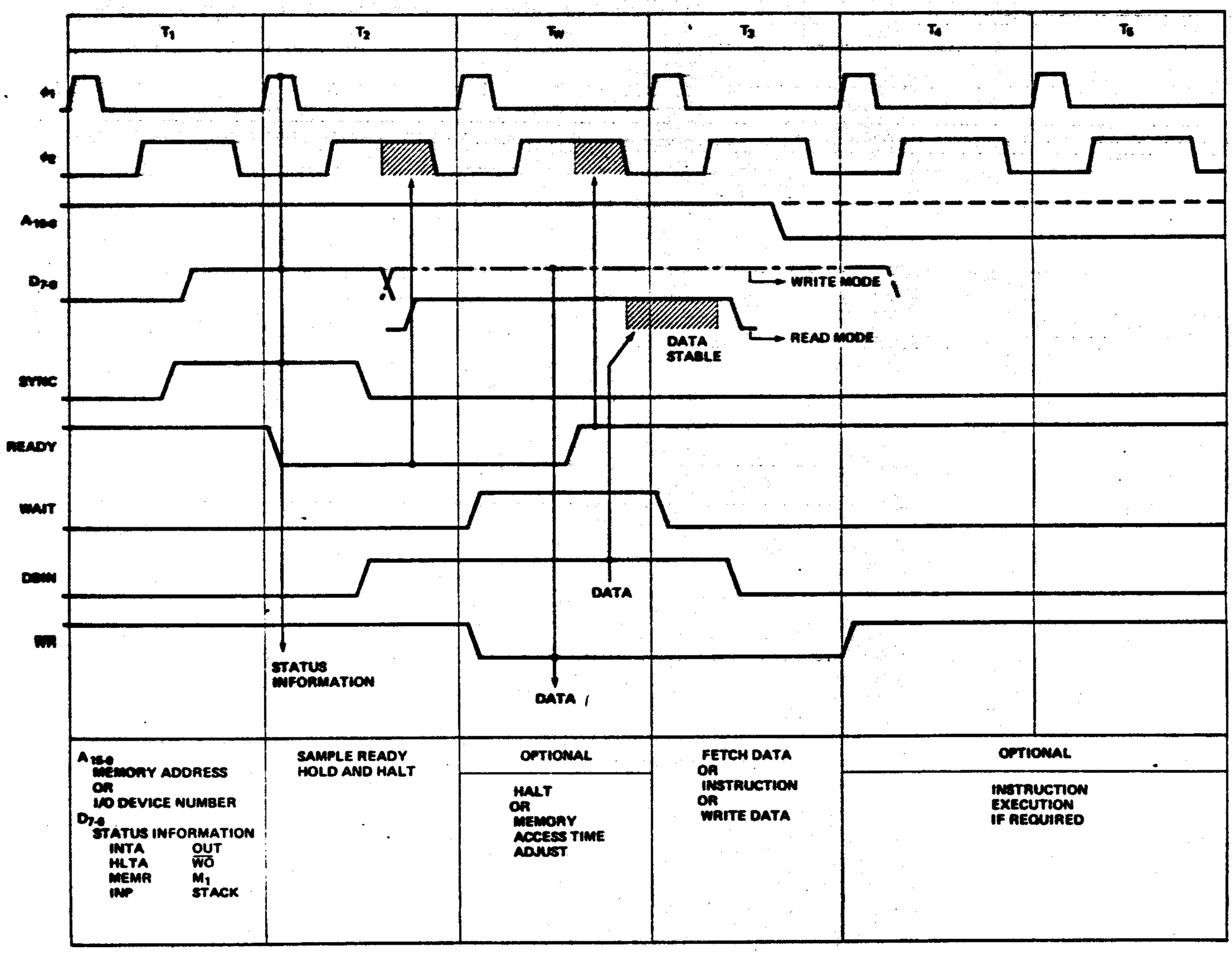
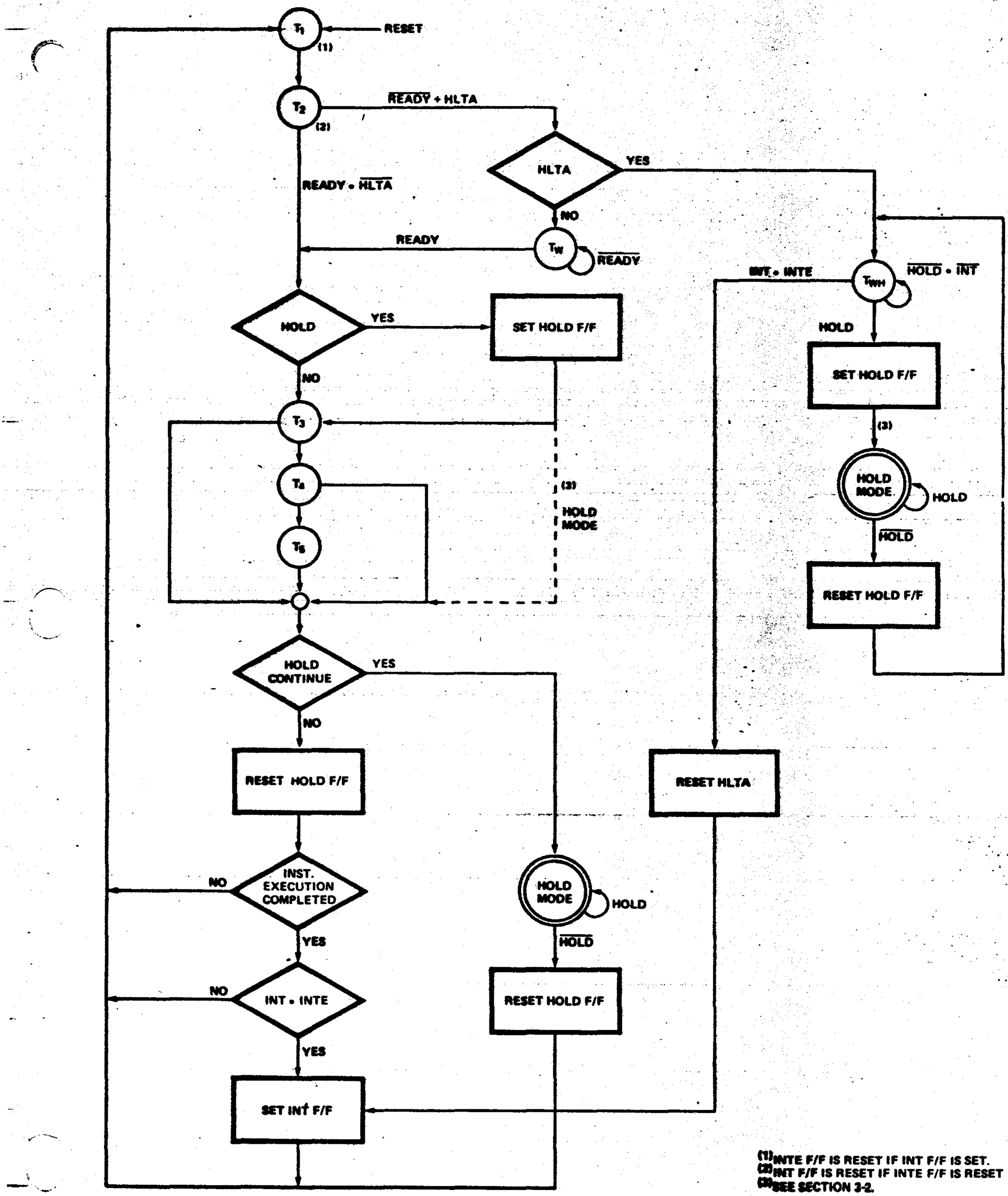


Figure 2. Basic 8080 Instruction Cycle



(1) INTE F/F IS RESET IF INT F/F IS SET.
(2) INT F/F IS RESET IF INTE F/F IS RESET
(3) SEE SECTION 3-2.

Figure 3. CPU State Transition Diagram

At the end of T4, if the cycle is complete, or else at the end of T5, the 8080 goes back to T1 and enters machine cycle M2, unless the instruction required only one machine cycle for its execution. In such cases, a new M1 cycle is entered. The loop is repeated for as many cycles and states as required by the instruction.

It is only during the last state of the last machine cycle that the interrupt request line is tested and a special M1 cycle is entered, during which no program-counter incrementing takes place and INTERRUPT ACKNOWLEDGE status is sent out. During this cycle, one of eight possible restart instructions will be sent to the CPU by the interrupting device.

Instruction state requirements range from a minimum of four states for non-memory referencing instructions, like register and accumulator arithmetic instructions, up to a maximum of 18 states for the most complex instructions (exchange the contents of registers H and L with the content of the top two locations of the stack). At the maximum clock frequency of 2 megahertz, this means that all instructions will be executed in intervals ranging from 2 μs to 9 μs. If a HALT instruction is executed, the processor enters a WAIT state and remains there until an interrupt is received.

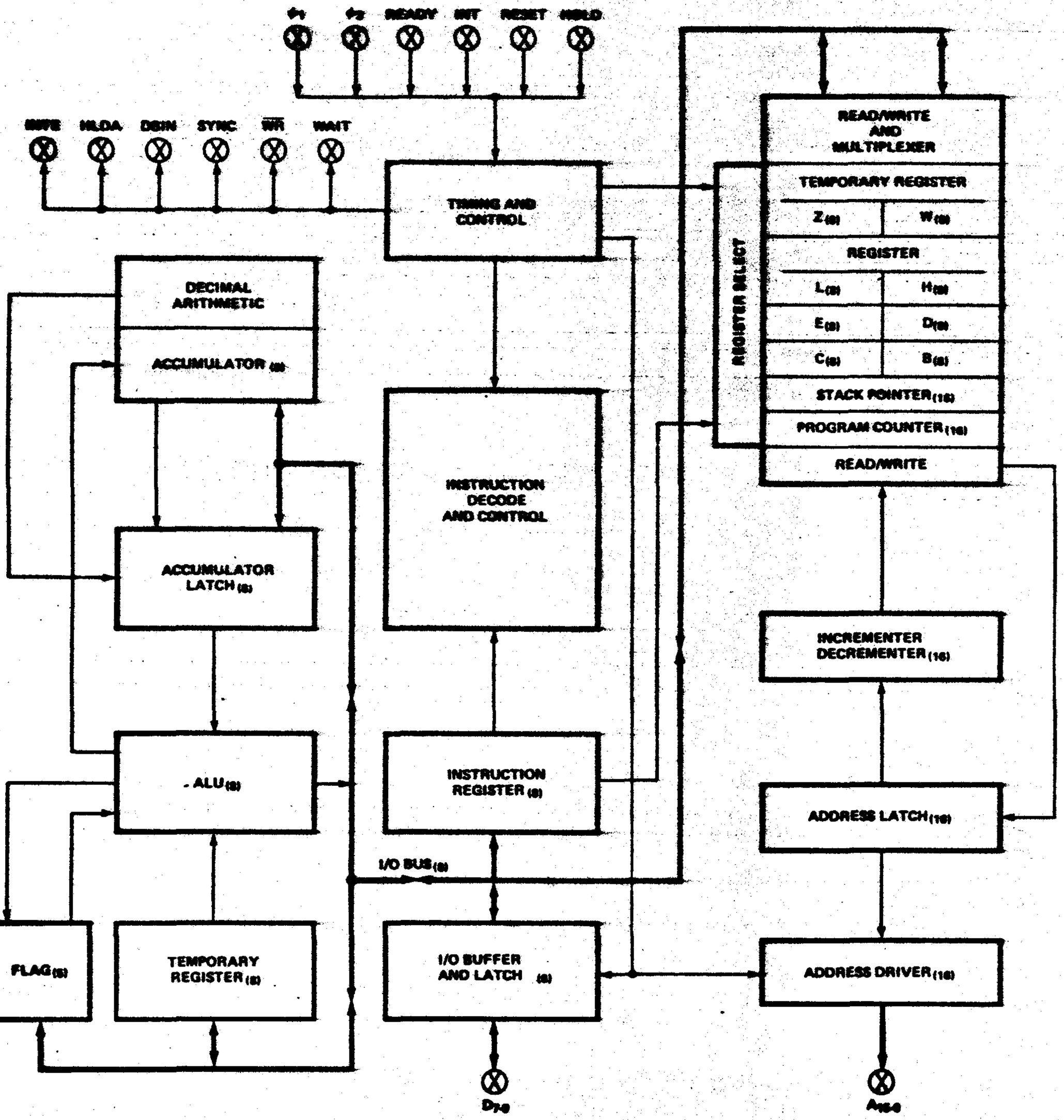


Figure 4. 8080 Block Diagram

3. PROCESSOR PROGRAMMING INSTRUCTIONS

3-1. Complete Functional Definition

The following pages present a detailed description of the complete 8080 Instruction Set.

Symbols	Meaning
<B2>	Second byte of the instruction
<B3>	Third byte of the instruction
r	One of the scratch pad register references: A, B, C, D, E, H, L
c	One of the following flag flip-flop references:

flag flip-flops Condition for True	
carry	—Overflow, underflow
zero	—Result is zero
sign	—MSB of result is "1"
parity	—Parity of result is even

M	Memory location indicated by the contents of registers H and L
( )	Contents of location or register
∧	Logical product
⊻	Exclusive "or"
∨	Inclusive "or"
A_m	Bit m of the A-register
SP	Stack Pointer
PC	Program Counter
←	is transferred to
XXX	A "don't care"
SSS	Source register for data
DDD	Destination register for data

Register # (SSS or DDD)	Register Name
000	B
001	C
010	D
011	E
100	H
101	L
110	Memory
111	ACC

8080 INSTRUCTION SET

Mnemonic	Bytes	Cycles	Description of Operation
MOV r_1, r_2	1	1	$(r_1) \leftarrow (r_2)$ Load register r_1 with the content of r_2. The content of r_2 remains unchanged.
MOV r, M	1	2	(r) ← (M) Load register r with the content of the memory location addressed by the contents of registers H and L.
MOV M, r	1	2	(M) ← (r) Load the memory location addressed by the contents of registers H and L with the content of register r.
MVI r <B_2>	2	2	(r) ← <B_2> Load byte two of the instruction into register r.
MVI M <B_2>	2	3	(M) ← <B_2> Load byte two of the instruction into the memory location addressed by the contents of registers H and L.

Mnemonic	Bytes	Cycles	Description of Operation
INR r	1	1	(r) ← (r) + 1 The content of register r is incremented by one. All the condition flip-flops except carry are affected by the result.
DCR r	1	1	(r) ← (r) − 1 The content of register r is decremented by one. All of the condition flip-flops except carry are affected by the result.
ADD r	1	1	(A) ← (A) + (r) Add the content of register r to the content of register A and place the result into register A. (All flags affected.)
ADC r	1	1	(A) ← (A) + (r) + (carry) Add the content of register r and the contents of the carry flip-flop to the content of the A register and place the result into Register A. (All flags affected.)
SUB r	1	1	(A) ← (A) − (r) Subtract the content of register r from the content of register A and place the result into register A. Two's complement subtraction is used. (All flags affected.)
SBB r	1	1	(A) ← (A) − (r) − (borrow) Subtract the content of register r and the content of the carry flip-flop from the content of register A and place the result into register A. (All flags affected.)
ANA r	1	1	(A) ← (A) ∧ (r) Place the logical product of the register A and register r into register A. (Resets carry.)
XRA r	1	1	(A) ← (A) ∀ (r) Place the "exclusive - or" of the content of register A and register r into register A. (Resets carry.)
ORA r	1	1	(A) ← (A) ∨ (r) Place the "inclusive - or" of the content of register A and register r into register A. (Resets carry.)
CMP r	1	1	(A) − (r) Compare the content of register A with the content of register r. The content of register A remains unchanged. The flag flip-flops are set by the result of the subtraction. Equality (A = r) is indicated by the zero flip-flop set to "1." Less than (A < r) is indicated by the carry flip-flop, set to "1."
ADD M	1	2	(A) ← (A) + (M) ADD
ADC M	1	2	(A) ← (A) + (M) + (carry) ADD with carry
SUB M	1	2	(A) ← (A) − (M) SUBTRACT
SBB M	1	2	(A) ← (A) − (M) − (borrow) SUBTRACT with borrow
ANA M	1	2	(A) ← (A) ∧ (M) Logical AND
XRA M	1	2	(A) ← (A) ∀ (M) Exclusive OR
ORA M	1	2	(A) ← (A) ∨ (M) Inclusive OR
CMP M	1	2	(A) − (M) COMPARE
ADI $<B_2>$	2	2	(A) ← (A) + $<B_2>$ ADD
ACI $<B_2>$	2	2	(A) ← (A) + $<B_2>$ + (carry) ADD with carry
SUI $<B_2>$	2	2	(A) ← (A) − $<B_2>$ SUBTRACT
SBI $<B_2>$	2	2	(A) ← (A) − $<B_2>$ − (borrow) SUBTRACT with borrow
ANI $<B_2>$	2	2	(A) ← (A) ∧ $<B_2>$ Logical AND
XRI $<B_2>$	2	2	(A) ← (A) $<B_2>$ Exclusive OR
ORI $<B_2>$	2	2	(A) ← (A) $<B_2>$ Inclusive OR
CPI $<B_2>$	2	2	(A) − $<B_2>$ COMPARE
RLC	1	1	$A_{m+1} \leftarrow A_m$, $A_0 \leftarrow A_7$, (carry) $\leftarrow A_7$ Rotate the content of register A left one bit. Rotate A_7 into A_0 and into the carry flip-flop.
RRC	1	1	$A_m \leftarrow A_{m+1}$, $A_7 \leftarrow A_0$, (carry) $\leftarrow A_0$ Rotate the content of register A right one bit. Rotate A_0 into A_7 and into the carry flip-flop.

(M) addressed by the contents of registers H and L.

Flags affected are same as ARr.

Mnemonic	Bytes	Cycles	Description of Operation
RAL	1	1	$A_{m+1} \leftarrow A_m$, $A_0 \leftarrow$ (carry), (carry) $\leftarrow A_7$ Rotate the content of Register A left one bit. Rotate the content of the carry flip-flop into A_0. Rotate A_7 into the carry flip-flop.
RAR	1	1	$A_m \leftarrow A_{m+1}$, $A_7 \leftarrow$ (carry), (carry) $\leftarrow A_0$ Rotate the content of register A right one bit. Rotate the content of the carry flip-flop into A_7. Rotate A_0 into the carry flip-flop.
JMP <B_2> <B_3>	3	3	(PC) $\leftarrow$ <B_3> <B_2> Jump unconditionally to the instruction located in memory location addressed by byte two and byte three.
JC <B_2> <B_3>	3	3	If (Carry) = 1 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JNC <B_2> <B_3>	3	3	If (Carry) = 0 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JZ <B_2> <B_3>	3	3	If (Zero) = 1 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JNZ <B_2> <B_3>	3	3	If (Zero) = 0 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JP <B_2> <B_3>	3	3	If (Sign) = 0 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JM <B_2> <B_3>	3	3	If (Sign) = 1 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JPE <B_2> <B_3>	3	3	If (Parity) = 1 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
JPO <B_2> <B_3>	3	3	If (Parity) = 0 (PC) $\leftarrow$ <B_3> <B_2> Otherwise (PC) = (PC) + 3
HLT	1	1	On receipt of the Halt Instruction, the activity of the processor is immediately suspended in the STOPPED state. The content of all registers and memory is unchanged and the PC has been updated.
CALL <B_2> <B_3>	3	5	[SP − 1] [SP − 2] $\leftarrow$ (PC), (SP) = (SP) − 2 (PC) $\leftarrow$ <B_3> <B_2> Transfer the content of PC to the pushdown stack in memory addressed by the register SP. The content of SP is decremented by two. Jump unconditionally to the instruction located in memory location addressed by byte two and byte three of the instruction.
CC <B_2> <B_3>	3	3/5	If (carry) = 1 [SP − 1] [SP − 2] $\leftarrow$ PC, (SP) = (SP) − 2, (PC) $\leftarrow$ <B_3> <B_2>; otherwise (PC) = (PC) + 3
CNC <B_2> <B_3>	3	3/5	If (carry) = 0 [SP − 1] [SP − 2] $\leftarrow$ PC, (SP) = (SP) − 2, (PC) $\leftarrow$ <B_3> <B_2>; otherwise (PC) = (PC) + 3
CZ <B_2> <B_3>	3	3/5	If (zero) = 1 [SP − 1] [SP − 2] $\leftarrow$ PC, (SP) = (SP) − 2, (PC) $\leftarrow$ <B_3> <B_2>; otherwise (PC) = (PC) + 3
CNZ <B_2> <B_3>	3	3/5	If (zero) = 0 [SP − 1] [SP − 2] $\leftarrow$ PC, (SP) = (SP) − 2, (PC) $\leftarrow$ <B_3> <B_2>; otherwise (PC) = (PC) + 3
CP <B_2> <B_3>	3	3/5	If (sign) = 0 [SP − 1] [SP − 2] $\leftarrow$ PC, (SP) = (SP) − 2, (PC) $\leftarrow$ <B_3> <B_2>; otherwise (PC) = (PC) + 3

Mnemonic	Bytes	Cycles	Description of Operation
CM <B_2> <B_3>	3	3/5	If (sign) = 1 [SP − 1] [SP − 2] ← PC, (SP) = (SP) − 2, (PC) ← <B_3> <B_2>; otherwise (PC) = (PC) + 3
CPE <B_2> <B_3>	3	3/5	If (parity) = 1 [SP − 1] [SP − 2] ← PC, (SP) = (SP) − 2, (PC) ← <B_3> <B_2>; otherwise (PC) = (PC) + 3
CPO <B_2> <B_3>	3	3/5	If (parity) = 0 [SP − 1] [SP − 2] ← PC, (SP) = (SP) − 2, (PC) ← <B_3> <B_2>; otherwise (PC) = (PC) + 3
RET	1	3	(PC) ← [SP] [SP + 1] (SP) = (SP) + 2. Return to the instruction in the memory location addressed by the last values shifted into the pushdown stack addressed by SP. The content of SP is incremented by two.
RC	1	1/3	If (carry) = 1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RNC	1	1/3	If (carry) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RZ	1	1/3	If (zero) = 1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RNZ	1	1/3	If (zero) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RP	1	1/3	If (sign) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RM	1	1/3	If (sign) = 1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RPE	1	1/3	If (parity) = 1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RPO	1	1/3	If (parity) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1
RST	1	3	[SP − 1] [SP − 2] ← (PC), (SP) = (SP) − 2 (PC) ← (00000000 00AAA000)
IN <B_2>	2	3	(A) ← (Input data) At T_1 time of third cycle, byte two of the instruction, which denotes the I/O device number, is sent to the I/O device through the address lines*, and the INP status information, instead of MEMR, is sent out at sync time. New data for the accumulator is loaded from the data bus when DBIN control signal is active. The condition flip-flops are not affected.
OUT <B_2>	2	3	(Output data) ← (A) At T_1 time of the third cycle, byte two of the instruction, which denotes the I/O device number, is sent to the I/O device through the address lines*, and the OUT status information is sent out at sync time. The content of the accumulator is made available on the data bus when the $\overline{WR}$ control signal is 0.
LXI B <B_2> <B_3>	3	3	(C) ← <B_2>; (B) ← <B_3> Load byte two of the instruction into C. Load byte three of the instruction into B.
LXI D <B_2> <B_3>	3	3	(E) ← <B_2>, (D) ← <B_3> Load byte two of the instruction into E. Load byte 3 of the instruction into D.
LXI H <B_2> <B_3>	3	3	(L) ← <B_2>, (H) ← <B_3> Load byte two of the instruction into L. Load byte three of the instruction into H.

*The device address appears on $A_7 - A_0$ and $A_{15} - A_8$

Mnemonic	Bytes	Cycles	Description of Operation
LXI SP <B_2> <B_3>	3	3	$(SP)_L \leftarrow <B_2>$, $(SP)_H \leftarrow <B_3>$ Load byte two of the instruction into the lower order 8-bit of the stack pointer and byte three into the higher order 8-bit of the stack pointer.
PUSH PSW	1	3	[SP − 1] ← (A), [SP − 2] ← (F), (SP) = (SP) − 2 Save the contents of A and F (5-flags) into the pushdown stack addressed by the SP register. The content of SP is decremented by two. The flag word will appear as follows: D_0: CY_2 (Carry) D_1: 1 D_2: Parity (even) D_3: 0 D_4: CY_1 D_5: 0 D_6: Zero D_7: MSB (sign)
PUSH B	1	3	[SP − 1] ← (B) [SP − 2] ← (C), (SP) = (SP) − 2
PUSH D	1	3	[SP − 1] ← (D) [SP − 2] ← (E), (SP) = (SP) − 2
PUSH H	1	3	[SP − 1] ← (H) [SP − 2] ← (L), (SP) = (SP) − 2
POP PSW	1	3	(F) ← [SP], (A) ← [SP + 1], (SP) = (SP) + 2 Restore the last values in the pushdown stack addressed by SP into A and F. The content of SP is incremented by two.
POP B	1	3	(C) ← [SP], (B) ← [SP + 1], (SP) = (SP) + 2
POP D	1	3	(E) ← [SP], (D) ← [SP + 1], (SP) = (SP) + 2
POP H	1	3	(L) ← [SP], (H) ← [SP + 1], (SP) = (SP) + 2
STA <B_2> <B_3>	3	4	[<B_3> <B_2>] ← (A) Store the accumulator content into the memory location addressed by byte two and byte three of the instruction.
LDA <B_2> <B_3>	3	4	(A) ← [<B_3> <B_2>] Load the accumulator with the content of the memory location addressed by byte two and byte three of the instruction.
XCHG	1	1	(H) ⟷ (D) (E) ⟷ (L) Exchange the contents of registers H and L and registers D and E.
XTHL	1	5	(L) ⟷ [SP], (H) ⟷ [SP + 1] Exchange the contents of registers H, L and the last values in the pushdown stack addressed by registers SP. The SP register itself is not changed. (SP) = (SP)
SPHL	1	1	(SP) ← (H) (L) Transfer the contents of registers H and L into register SP.
PCHL	1	1	(PC) ← (H) (L) JUMP INDIRECT
DAD SP	1	3	(H) (L) ← (H) (L) + (SP) Add the content of register SP to the content of registers H and L and place the result into registers H and L. If the overflow is generated, the carry flip-flop is set; otherwise, the carry flip-flop is reset. The other condition flip-flops are not affected. This is useful for addressing data in the stack.
DAD B	1	3	(H) (L) ← (H) (L) + (B) (C)
DAD H	1	3	(H) (L) ← (H) (L) + (H) (L) (double precision shift left H and L)
DAD D	1	3	(H) (L) ← (H) (L) + (D) (E)
STAX B	1	2	[(B) (C)] ← (A) Store the accumulator content in the memory location addressed by the content of registers B and C.
STAX D	1	2	[(D) (E)] ← (A) Store the accumulator content into the memory location addressed by the content of register D and E.
LDAX B	1	2	(A) ← [(B) (C)] Load the accumulator with the content of the memory location addressed by the content of registers B and C.

[illegible]	1	2	(A) ← [(D) (E)] Load the accumulator with the content of memory location addressed by the content of register D and E.
INX B	1	1	(B) (C) ← (B) (C) + 1 The content of register pair B and C is incremented by one. All of the condition flip-flops are not affected.
INX H	1	1	(H) (L) ← (H) (L) + 1 The content of register H and L is incremented by one. All of the condition flip-flops are not affected.
INX D	1	1	(D) (E) ← (D) (E) + 1
INX SP	1	1	(SP) ← (SP) + 1
DCX B	1	1	(B) (C) ← (B) (C) − 1
DCX H	1	1	(H) (L) ← (H) (L) − 1
DCX D	1	1	(D) (E) ← (D) (E) − 1
DCX SP	1	1	(SP) ← (SP) − 1
CMA	1	1	(A) ← $\overline{(A)}$ The content of accumulator is complemented. The condition flip-flops are not affected.
STC	1	1	(Carry) ← 1 Set the carry flip-flop to 1. The other condition flip-flops are not affected.
CMC	1	1	(carry) ← $\overline{(carry)}$ The content of carry is complemented. The other condition flip-flops are not affected.

DAA	1	1	Decimal Adjust Accumulator
			The 8-bit value in the accumulator containing the result from an arithmetic operation on decimal operands is adjusted to contain two valid BCD digits by adding a value according to the following rules:
			7——4 3——0
			X | Y
			Accumulator
			If (Y ≥ 10) or (carry from bit 3) then Y = Y + 6 with carry to X digit.
			If (X ≥ 10) or (carry from bit 7) or [(Y ≥ 10) and (X = 9)] then X = X + 6 (which sets the carry flip-flop).
			Two carry flip-flops are used for this instruction. CY_1 represents the carry from bit 3 (the fourth bit) and is accessible as a fifth flag. CY_2 is the carry from bit 7 and is the usual carry bit.
			All condition flip-flops are affected by this instruction.

SHLD <B_2> <B_3>	3	5	[<B_3> <B_2>] ← (L), [<B_3> <B_2> + 1] ← (H) Store the contents of registers H and L into the memory location addressed by byte two and byte three of the instructions.
LHLD <B_2> <B_3>	3	5	(L) ← [<B_3> <B_2>], (H) ← [<B_3> <B_2> + 1] Load the registers H and L with the contents of the memory location addressed by byte two and byte three of the instruction.
EI	1	1	Interrupt System Enable
DI	1	1	Interrupt System Disable The Interrupt Enable flip-flop (INTE) can be set or reset by using the above mentioned instructions. The INT signal will be accepted if the INTE is set. When the INT signal is accepted by the CPU, the INTE will be reset immediately. During interrupt enable or disable instruction executions, an interrupt will not be accepted.
INR M	1	3	[M] ← [M] + 1. The content of memory designated by registers H and L is incremented by one. All of the condition flip-flops except carry are affected by the result.
DCR M	1	3	[M] ← [M] −1. The content of memory designated by registers H and L is decremented by one. All of the condition flip-flops except carry are affected by the result.

3-2. Data and Instruction Formats

Data in the 8080 is stored in the form of 8-bit binary integers. All data transfers to the system data bus will be in the same format.

$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$
DATA WORD

The program instructions may be one, two, or three bytes in length. Multiple byte instructions must be stored in successive words in program memory. The instruction formats then depend on the particular operation executed.

One Byte Instructions

$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$	OP CODE

TYPICAL INSTRUCTIONS

Register to register, memory reference, arithmetic or logical, rotate return, PUSH, POP, ENABLE or DISABLE

Two Byte Instructions

INTERRUPT INSTRUCTIONS

$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$	OP CODE
$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$	OPERAND

Immediate mode or I/O instructions

Three Byte Instructions

$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$	OP CODE
$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$	LOW ADDRESS OR OPERAND 1
$D_7\ D_6\ D_5\ D_4\ D_3\ D_2\ D_1\ D_0$	HIGH ADDRESS OR OPERAND 2

JUMP, CALL or DIRECT LOAD AND STORE INSTRUCTIONS

For the 8080 a logic "1" is defined as a high level and a logic "0" is defined as a low level.

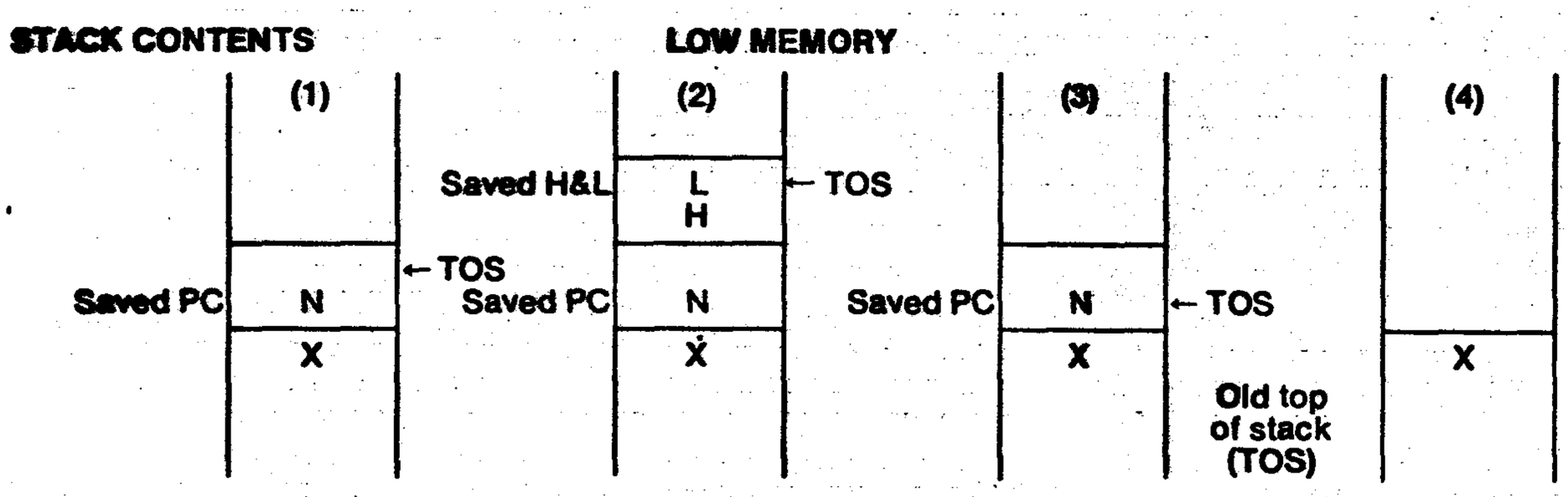
3-3. Summary of Processor Instructions

Mnemonic	Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	States Required	Notes
MOV r_1, r_2	0	1	D	D	D	S	S	S	5	
MOV M, r	0	1	1	1	0	S	S	S	7	
MOV r, M	0	1	D	D	D	1	1	0	7	
HLT	0	1	1	1	0	1	1	0	7	
MVI r	0	0	D	D	D	1	1	0	7	
MVi M	0	0	1	1	0	1	1	0	10	
INR r	0	0	D	D	D	1	0	0	5	
DCR r	0	0	D	D	D	1	0	1	5	1.
ADD r	1	0	0	0	0	S	S	S	4	DDD or SSS
ADC r	1	0	0	0	1	S	S	S	4	000 B
SUB r	1	0	0	1	0	S	S	S	4	001 C
SBB r	1	0	0	1	1	S	S	S	4	010 D
ANA r	1	0	1	0	0	S	S	S	4	011 E
XRA r	1	0	1	0	1	S	S	S	4	100 H
ORA r	1	0	1	1	0	S	S	S	4	101 L
CMP r	1	0	1	1	1	S	S	S	4	110 Memory
ADD M	1	0	0	0	0	1	1	0	7	111 ACC
ADC M	1	0	0	0	1	1	1	0	7	
SUB M	1	0	0	1	0	1	1	0	7	2.
SBB M	1	0	0	1	1	1	1	0	7	1 Time State =
ANA M	1	0	1	0	0	1	1	0	7	1 Clock Period
XRA M	1	0	1	0	1	1	1	0	7	≥500ns
ORA M	1	0	1	1	0	1	1	0	7	
CMP M	1	0	1	1	1	1	1	0	7	
ADI	1	1	0	0	0	1	1	0	7	
ACI	1	1	0	0	1	1	1	0	7	
SUI	1	1	0	1	0	1	1	0	7	
SBI	1	1	0	1	1	1	1	0	7	
NDI	1	1	1	0	0	1	1	0	7	
XRI	1	1	1	0	1	1	1	0	7	
ORI	1	1	1	1	0	1	1	0	7	
CPI	1	1	1	1	1	1	1	0	7	
RLC	0	0	0	0	0	1	1	1	4	
RRC	0	0	0	0	1	1	1	1	4	
RAL	0	0	0	1	0	1	1	1	4	
RAR	0	0	0	1	1	1	1	1	4	
JMP	1	1	0	0	0	0	1	1	10	
JC	1	1	0	1	1	0	1	0	10	
JNC	1	1	0	1	0	0	1	0	10	
JZ	1	1	0	0	1	0	1	0	10	
JNZ	1	1	0	0	0	0	1	0	10	
JP	1	1	1	1	0	0	1	0	10	
JM	1	1	1	1	1	0	1	0	10	

Mnemonic	D_7	D_6	Instruction Code D_5	D_4	D_3	D_2	D_1	D_0	States Required	Notes
JPE	1	1	1	0	1	0	1	0	10	
JPO	1	1	1	0	0	0	1	0	10	
CALL	1	1	0	0	1	1	0	1	17	
CC	1	1	0	1	1	0	0	0	11/17	
CNC	1	1	0	1	0	0	0	0	11/17	
CZ	1	1	0	0	1	0	0	0	11/17	
CNZ	1	1	0	0	0	0	0	0	11/17	
CP	1	1	1	1	0	0	0	0	11/17	
CM	1	1	1	1	1	0	0	0	11/17	
CPE	1	1	1	0	1	0	0	0	11/17	
CPO	1	1	1	0	0	0	0	0	11/17	
RET	1	1	0	0	1	0	0	1	10	
RC	1	1	0	1	1	0	0	0	5/11	
RNC	1	1	0	1	0	0	0	0	5/11	
RZ	1	1	0	0	1	0	0	0	5/11	
RNZ	1	1	0	0	0	0	0	0	5/11	
RP	1	1	1	1	0	0	0	0	5/11	
RM	1	1	1	1	1	0	0	0	5/11	
RPE	1	1	1	0	1	0	0	0	5/11	
RPO	1	1	1	0	0	0	0	0	5/11	
RST	1	1	A	A	A	1	1	1	11	
IN	1	1	0	1	1	0	1	1	10	
OUT	1	1	0	1	0	0	1	1	10	
LXI B	0	0	0	0	0	0	0	1	10	
LXI D	0	0	0	1	0	0	0	1	10	
LXI H	0	0	1	0	0	0	0	1	10	
LXI SP	0	0	1	1	0	0	0	1	10	
PUSH B	1	1	0	0	0	1	0	1	11	
PUSH D	1	1	0	1	0	1	0	1	11	
PUSH H	1	1	1	0	0	1	0	1	11	
PUSH PSW	1	1	1	1	0	1	0	1	11	
POP B	1	1	0	0	0	0	0	1	10	
POP D	1	1	0	1	0	0	0	1	10	
POP H	1	1	1	0	0	0	0	1	10	
POP PSW	1	1	1	1	0	0	0	1	10	
STA	0	0	1	1	0	0	1	0	13	
LDA	0	0	1	1	1	0	1	0	13	
XCHG	1	1	1	0	1	0	1	1	4	
XTHL	1	1	1	0	0	0	1	1	18	
SPHL	1	1	1	1	1	0	0	1	5	
PCHL	1	1	1	0	1	0	0	1	5	
DAD B	0	0	0	0	1	0	0	1	10	
DAD D	0	0	0	1	1	0	0	1	10	
DAD H	0	0	1	0	1	0	0	1	10	
DAD SP	0	0	1	1	1	0	0	1	10	
STAX B	0	0	0	0	0	0	1	0	7	
STAX D	0	0	0	1	0	0	1	0	7	
LDAX B	0	0	0	0	1	0	1	0	7	
LDAX D	0	0	0	1	1	0	1	0	7	
INX B	0	0	0	0	0	0	1	1	5	
INX D	0	0	0	1	0	0	1	1	5	
INX H	0	0	1	0	0	0	1	1	5	
INX SP	0	0	1	1	0	0	1	1	5	
DXC B	0	0	0	0	1	0	1	1	5	
DXC D	0	0	0	1	1	0	1	1	5	
DCX H	0	0	1	0	1	0	1	1	5	
DCX SP	0	0	1	1	1	0	1	1	5	
CMA	0	0	1	0	1	1	1	1	4	
STC	0	0	1	1	0	1	1	1	4	
CMC	0	0	1	1	1	1	1	1	4	
DAA	0	0	1	0	0	1	1	1	4	
SHLD	0	0	1	0	0	0	1	0	16	
LHLD	0	0	1	0	1	0	1	0	16	
EI	1	1	1	1	1	0	1	1	4	
DI	1	1	1	1	0	0	1	1	4	
NOP	0	0	0	0	0	0	0	0	4	

APPENDIX I
HOW TO USE THE PUSHDOWN STACK

Addr.	Location	PC Contents	SP Contents	
N − 1	INSTR. N − 1	N	SP	← (INTERRUPT ARRIVES HERE)
N	INSTR. N			Restart instruction inserted here
N + 1	INSTR. N + 1		---- (1) SP − 2	Save PC value N in stack using restart instruction to jump to S.
	SUBROUTINE FOR HANDLING INTERRUPT			
S	PUSH H	S + 1	---- (2) SP − 4	Save HL in stack if desired
⋮	EI			Enable further interrupts if desired.
S + n	POP H	S + n + 1	---- (3) SP − 2	Restore HL from stack
S + n + 1	RET	N	---- (4) SP	Return PC from stack



NOTE: The user can initialize the stack point SP register with a LXI SP instruction to use any section of read-write memory as a stack. The SP is decremented when data is pushed onto the stack, and incremented when data is popped (that is the stack grows "downward").

APPENDIX II
PROGRAMMING EXAMPLES

(Decimal operation)

a. Decimal Addition:

Memory address of Augend; D and E is (ALPHA)
Memory address of Addend; H and L is (BETA)

Mnemonic	Operand	Explanation	Bytes	Comment
LXI	D, ALPHA	Load D and E Immediate	3	Set address to DE
LXI	H, BETA	Load H and L Immediate	3	Set address to HL
MVI	C, 8	Load C with "8"		
XRA		Exclusive or A with A	1	Clear Carry
LOOP: LDAX	D	Load A with (DE)	1	Load Augend to Acc
ADC	M	Add M to A (HL)	1	Add Addend to Augend
DAA		Decimal Adjust	1	
STAX	D	Store A to (DE)	1	Replace Result
INX	H	Increment HL	1	Renew address HL
INX	D	Increment DE	1	Renew address DE
DCR	C	Decrement C	1	Check end of calculation
JNZ	LOOP	If not zero go to loop	3	

Calculation time (16 digits) ~ 230 μsec

b. Decimal Subtraction

Memory address of Minuend; D and E (ALPHA)
Memory address of Subtrahend; H and L (BETA)

Mnemonic	Operand	Explanation	Bytes	Comment
LXI	D, ALPHA	Load D and E Immediate	3	Set address to DE
LXI	H, BETA	Load H and L Immediate	3	Set address to HL
MVI	C, 8	Load C with "8"	2	
STC		Set Carry	1	
LOOP: MVI	A, 99H	Load A with 99 HEX	2	$99_{16} + 1 = 9A_{16}$
ACI	0	Add with carry	2	
SUB	M	Subtract M from A	1	
XCHG		Exchange DE and HL	1	Actually
ADD	M	Add M to A	1	
DAA		Decimal Adjust	1	$3 - 2 = 10 - 2 + 3 = 11$
MOV	M, A	Load A to M	1	
XCHG		Exchange DE and HL	1	No borrow occurs here
INX	D	Increment DE	1	
INX	H	Increment HL	1	
DCR	C	Decrement C	1	
JNZ	LOOP		3	

Calculation time (16 digits) ~ 330 μsec

Programming examples

c. Binary Multiplication Loop

A contains Multiplier, D and E is Multiplicand, H and L are Partial Product

Mnemonic	Operand	Explanation	Bytes
LXI	H, 0	Initialize Partial Product to 0	3
MVI	B, 8	8 → B to control loop	2
LOOP: DAD	H	Shift partial product left and into carry	1
RAL		Rotate multiplier bit to carry	1
JNC	DEC	Test multiplier at carry	3
DAD	D	Add multiplicand to partial product if carry = 1	1
ACI	0		
DEC: DCR	B	Decrement B loop counter	1
JNZ	LOOP	Test to see if B = 0 to iterate 8 times	3

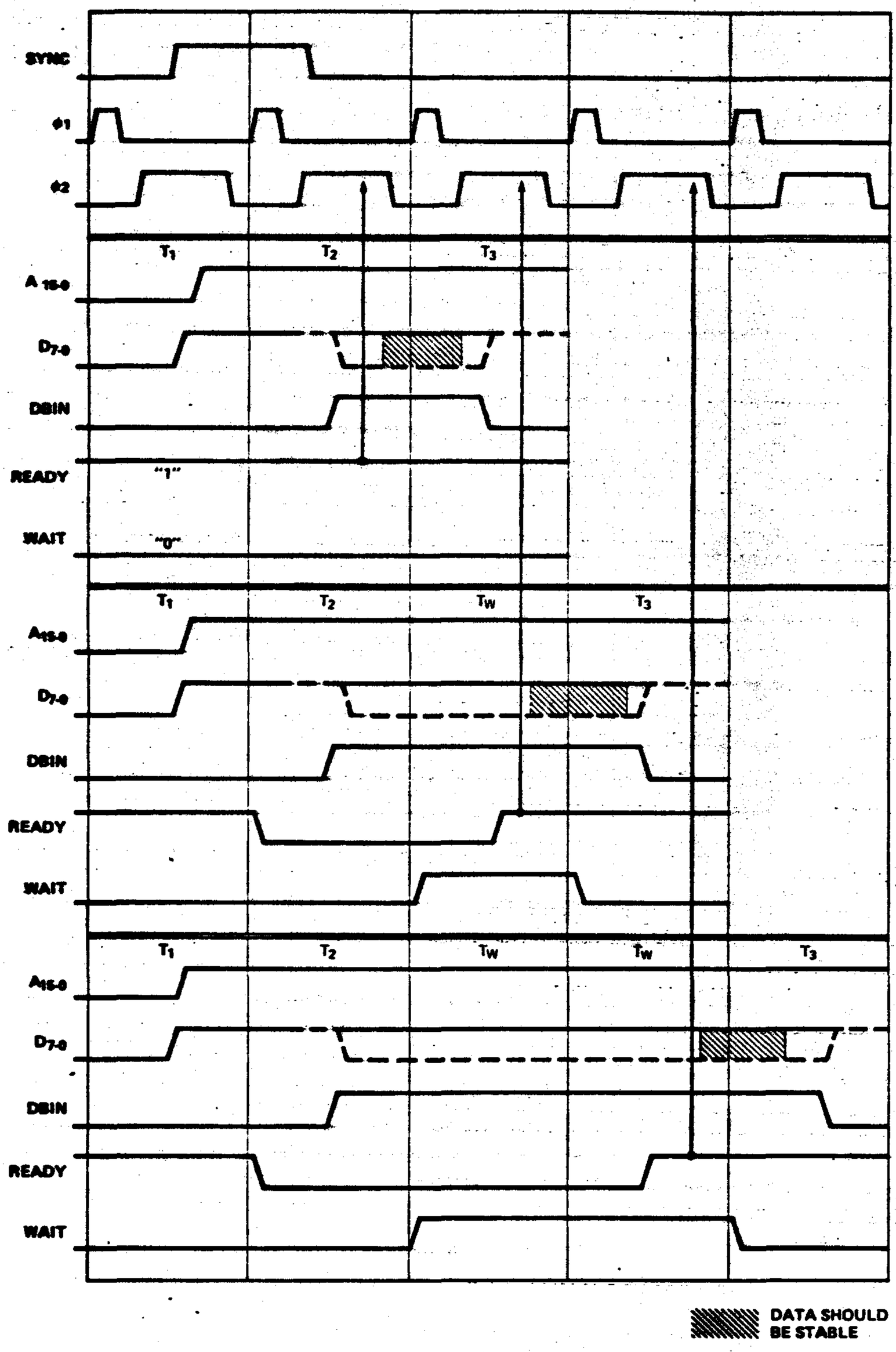
Calculation time for 8 x 16 multiply ~ 230 μsec

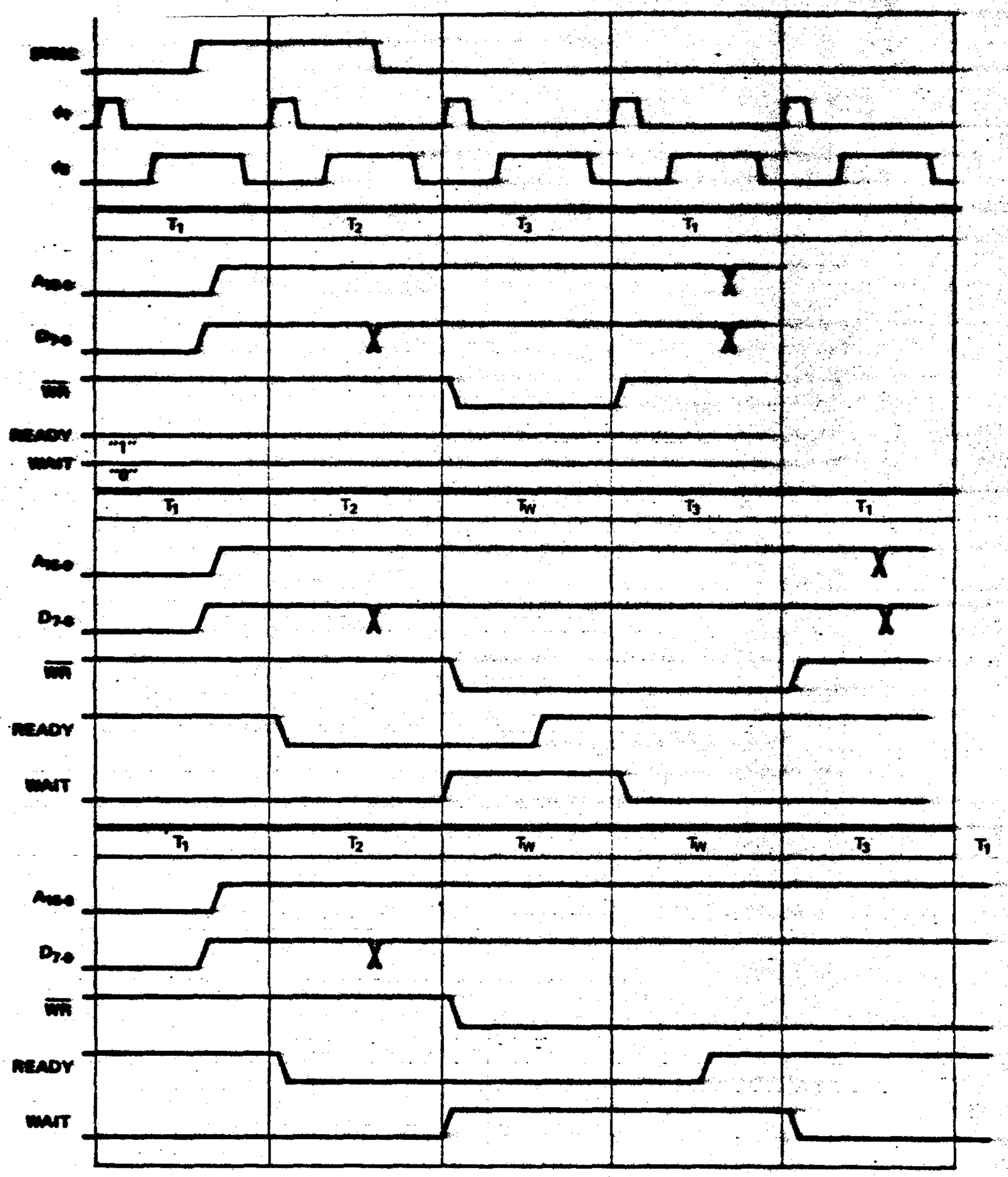
d. Accumulator Loading

Mnemonic	Operand	Explanation	Bytes
MOV	A, B	Load A with Register B	1
MVI	A, 23	Load A with Data immediate "23"	2
LDA	4098	Load A with contents of memory LOC 4098	3
MOV	A, M	Load A using H and L as address	1
LDAX	B	Load A using B and C as address	1
LDAX	D	Load A using D and E as address	1
LHLD	4098	Load A indirect using LOC 4098	4
MOV	A, M		
POP	A	Load A with data from stack	1
IN	10	Load A with data from Device #10	2

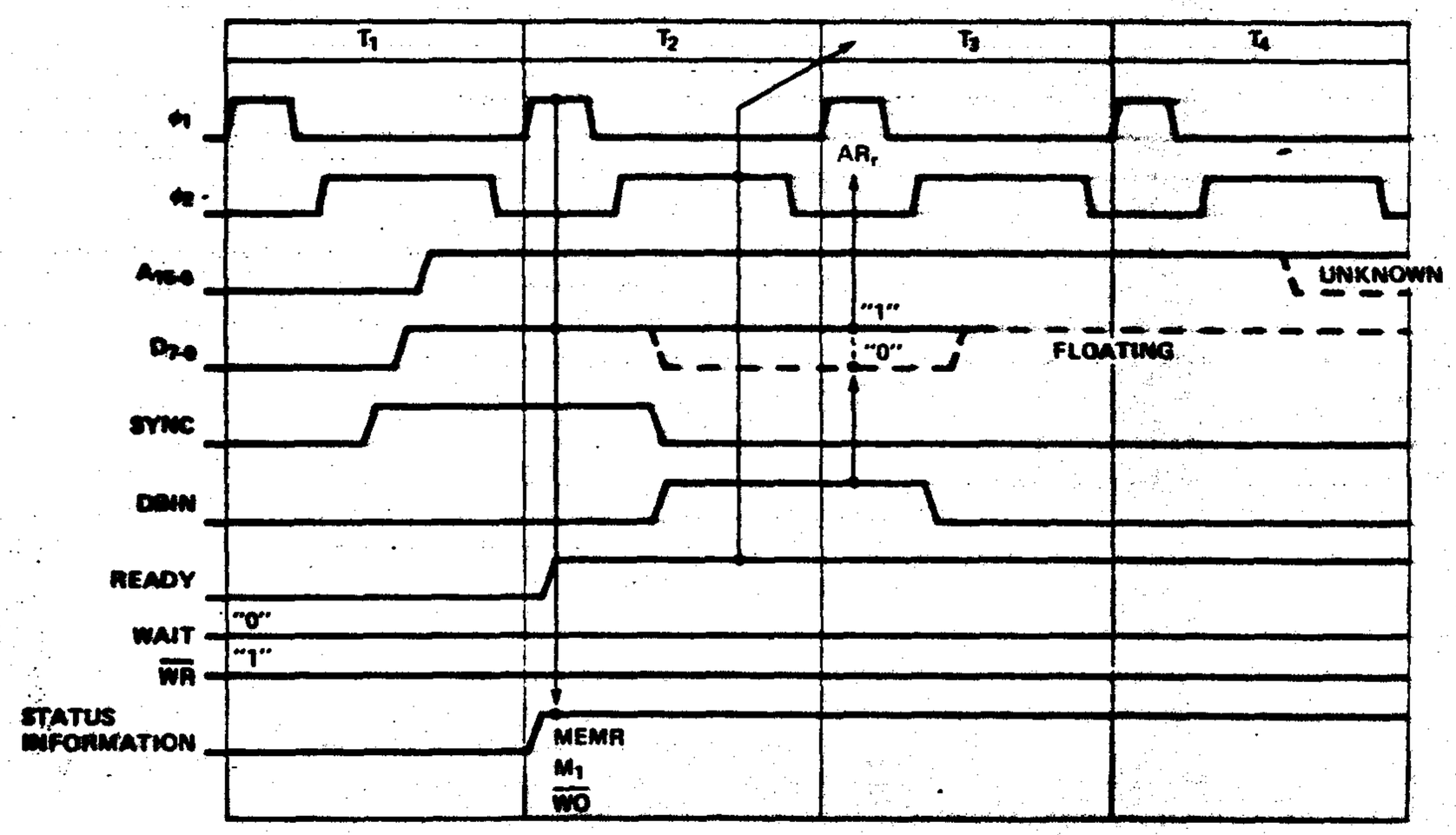
APPENDIX III
TIMING DIAGRAMS

a. Relation between READY and DBIN

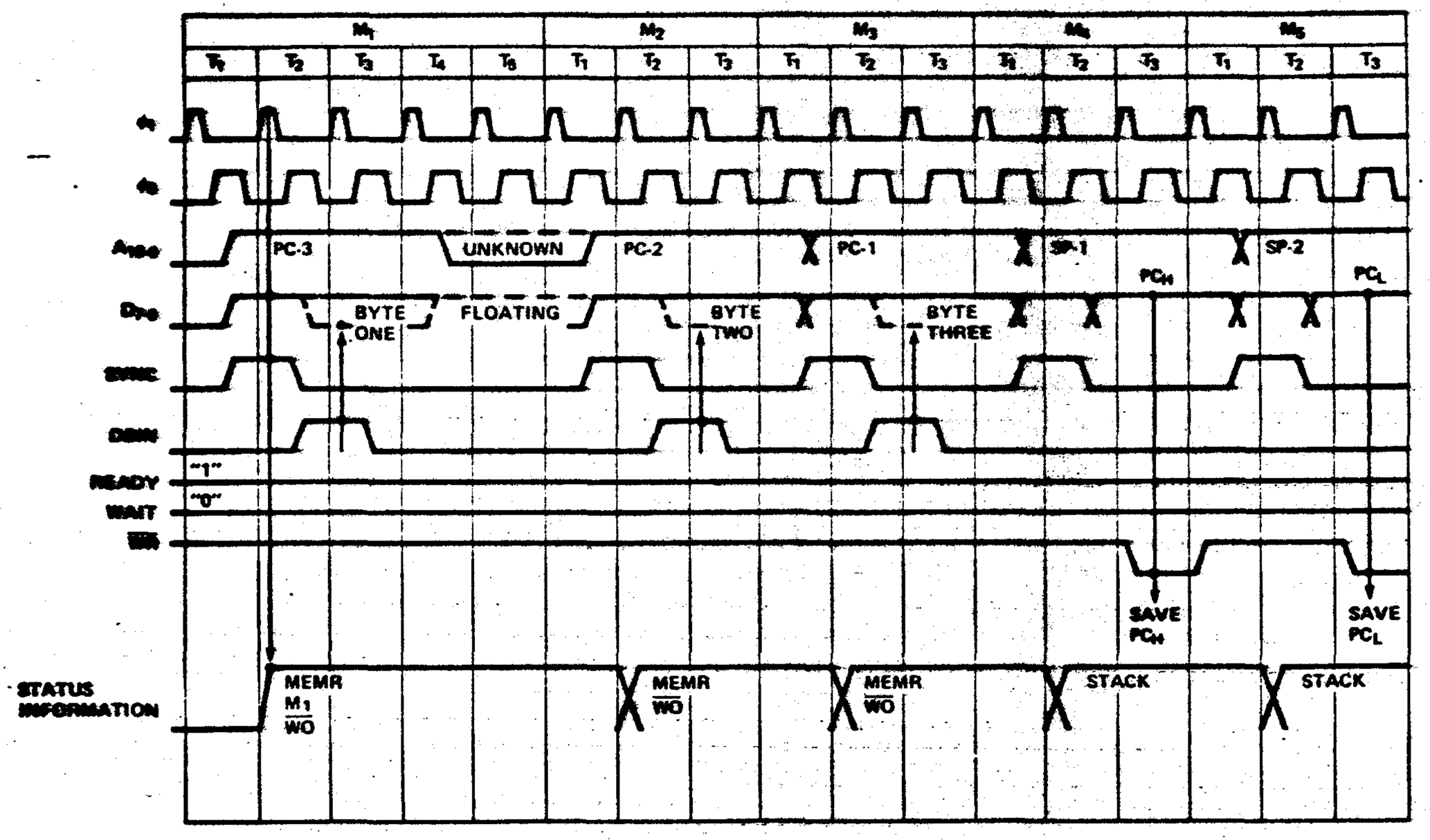




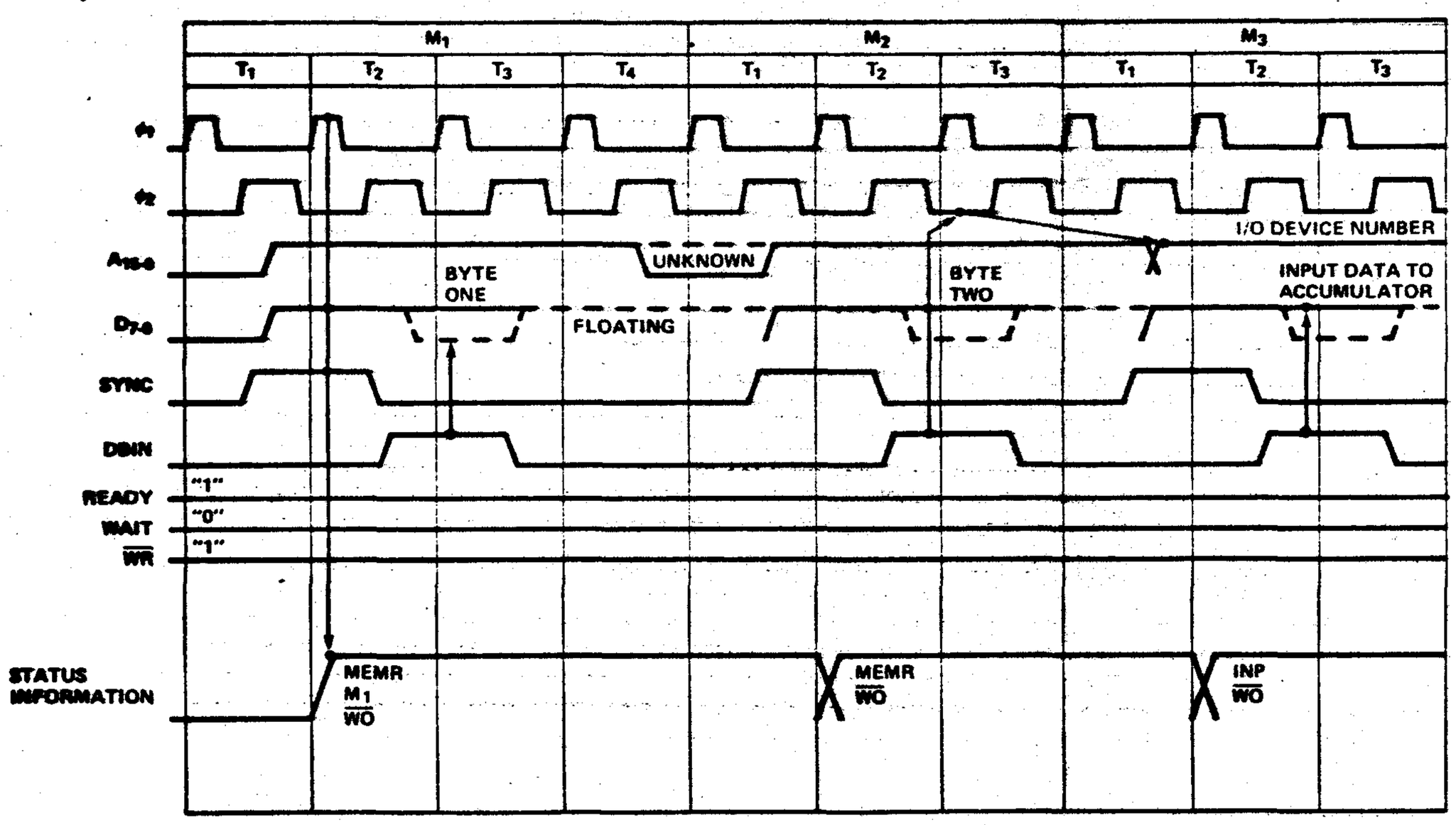
b. Non-Memory Reference Instruction (AR_r)



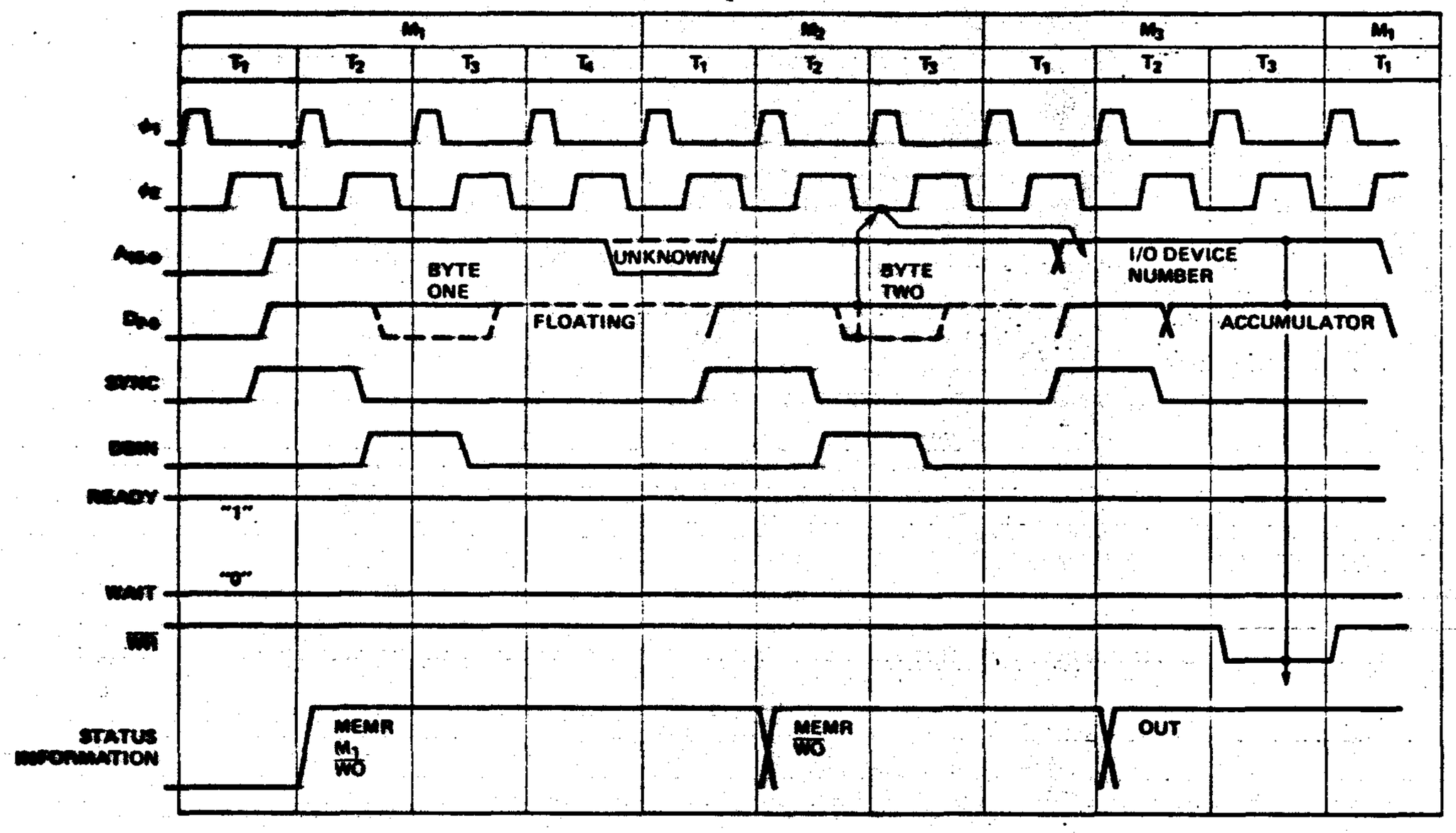
c. Memory Reference Instruction (CALL)



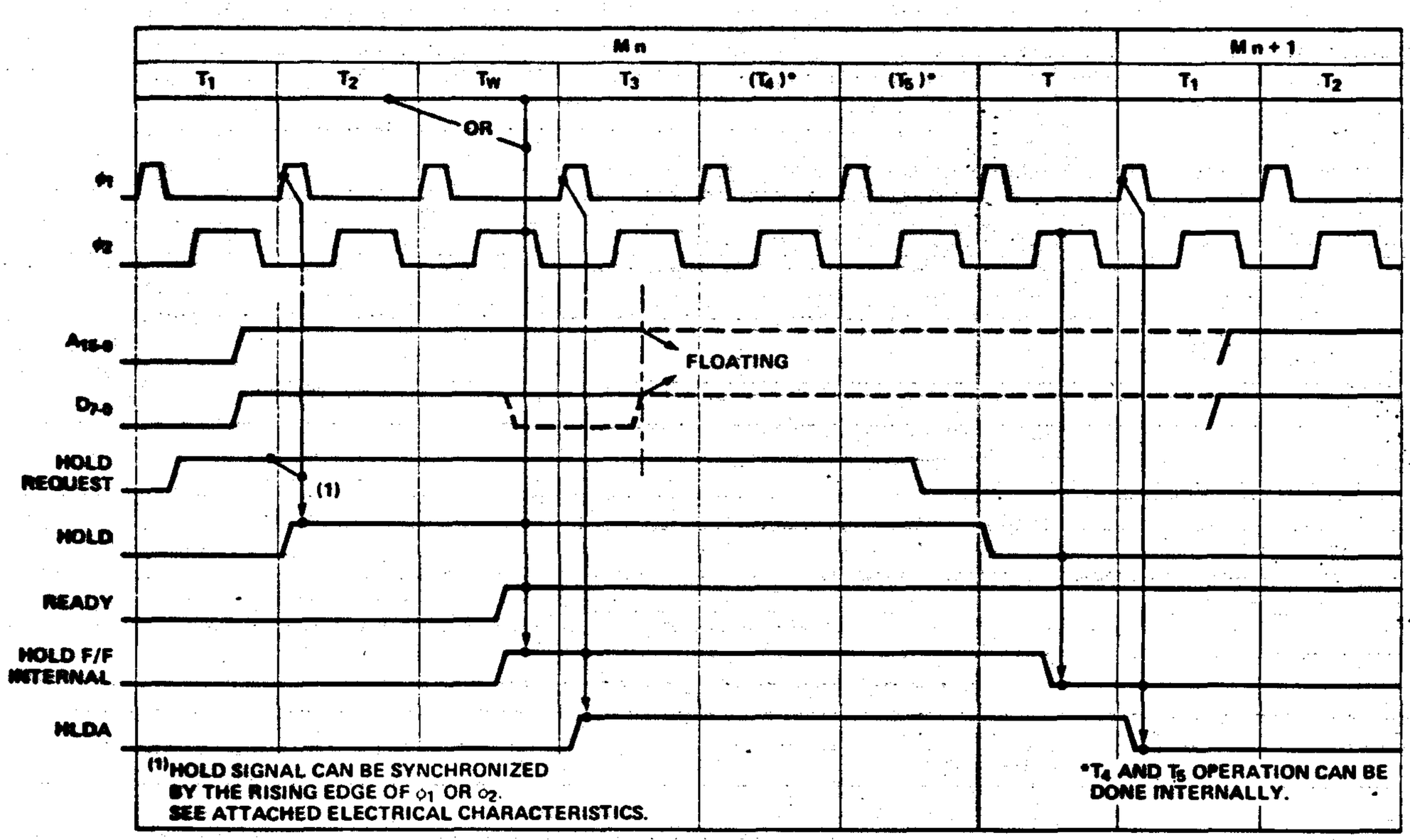
d. Input Instruction



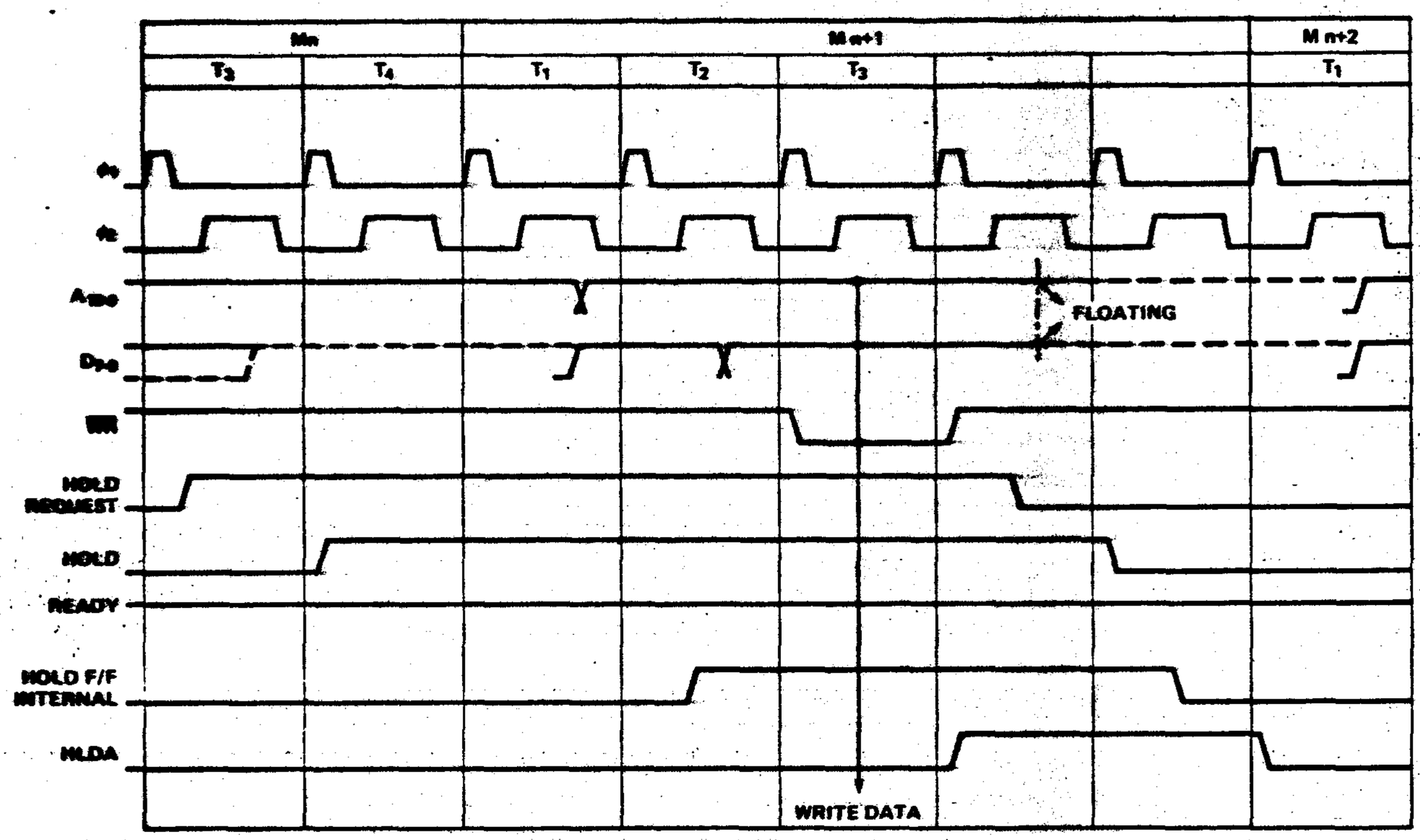
e. OUTPUT INSTRUCTION



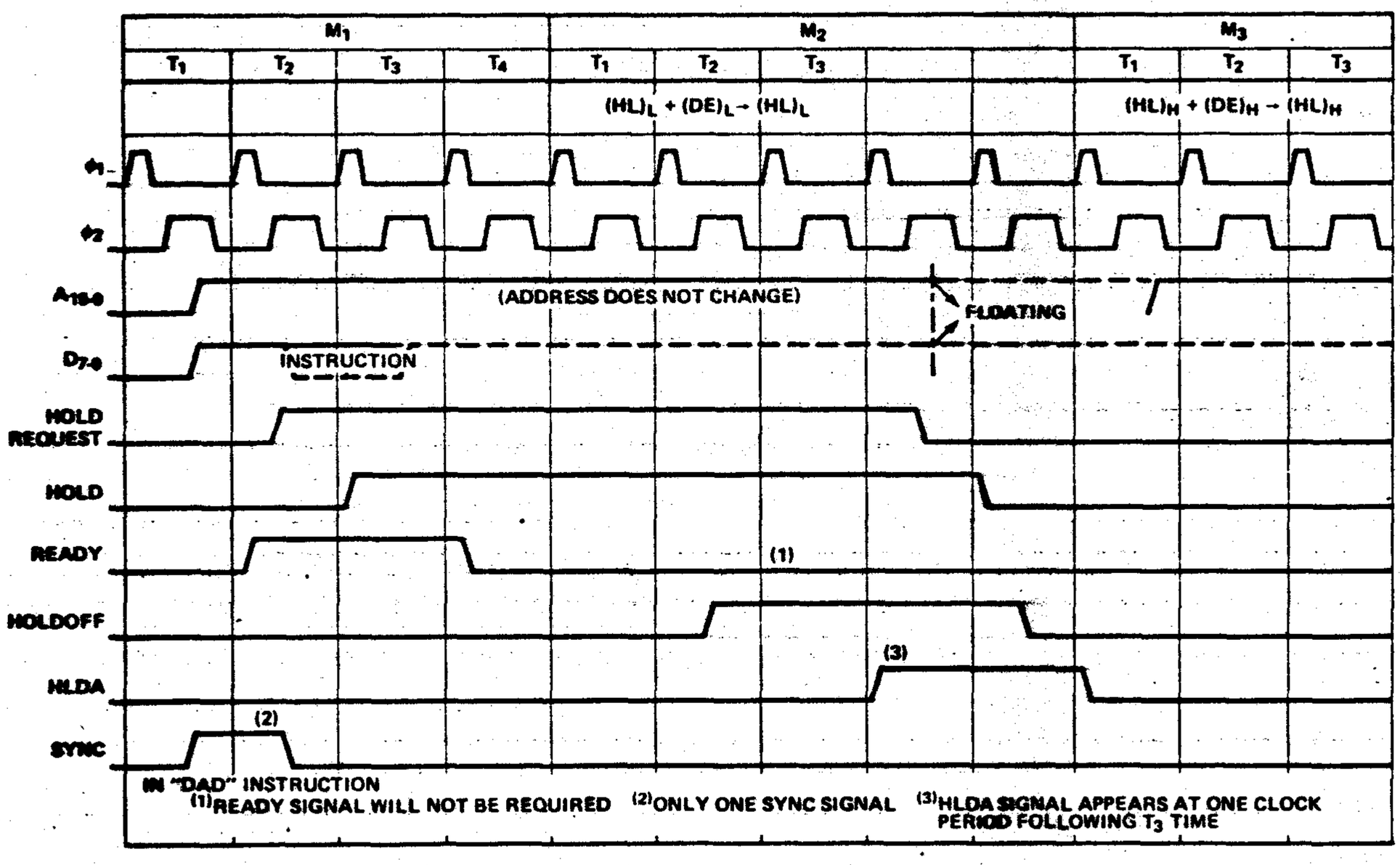
f. HOLD OPERATION (READ MODE)



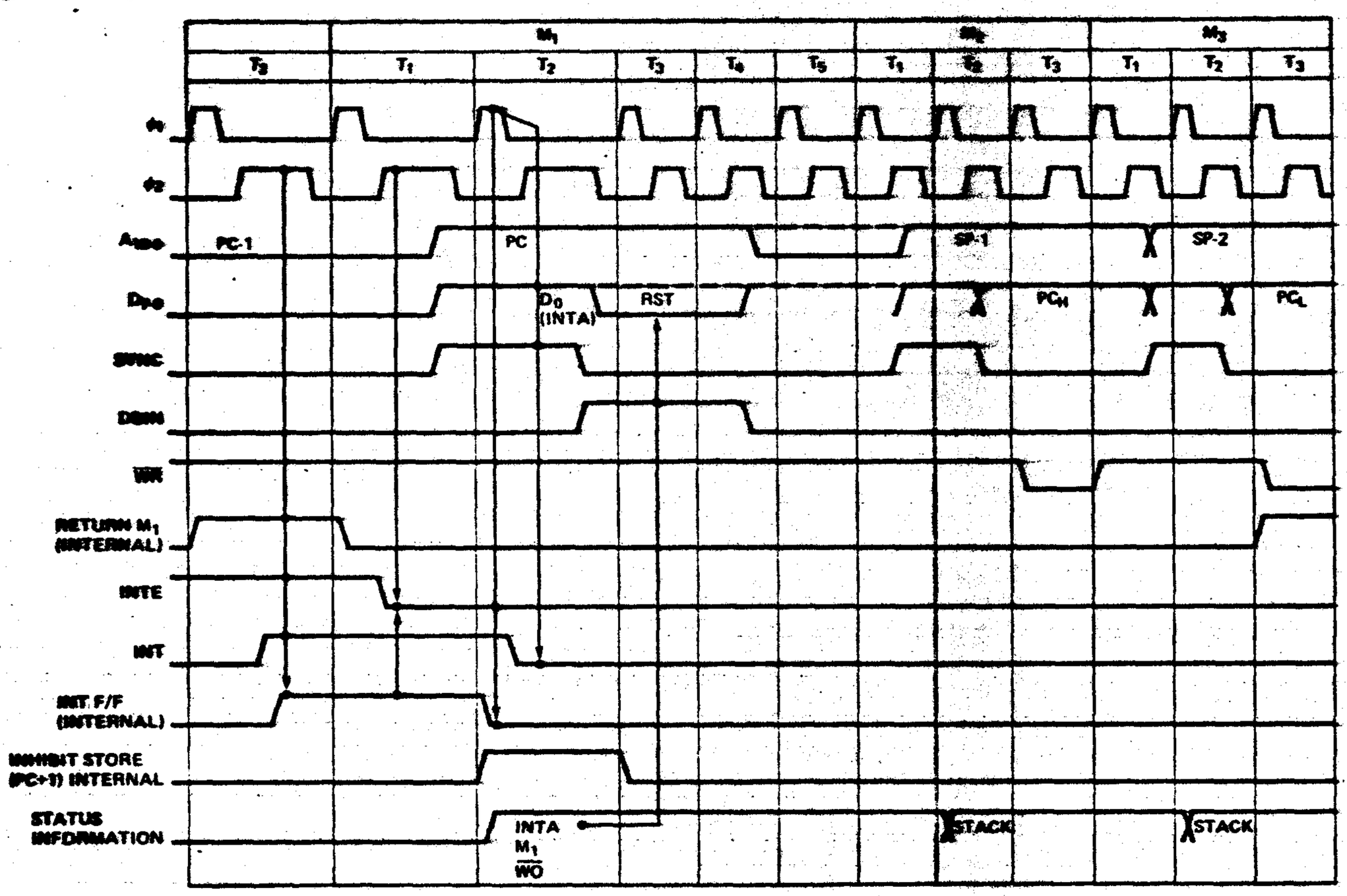
HOLD Operation (Write mode)



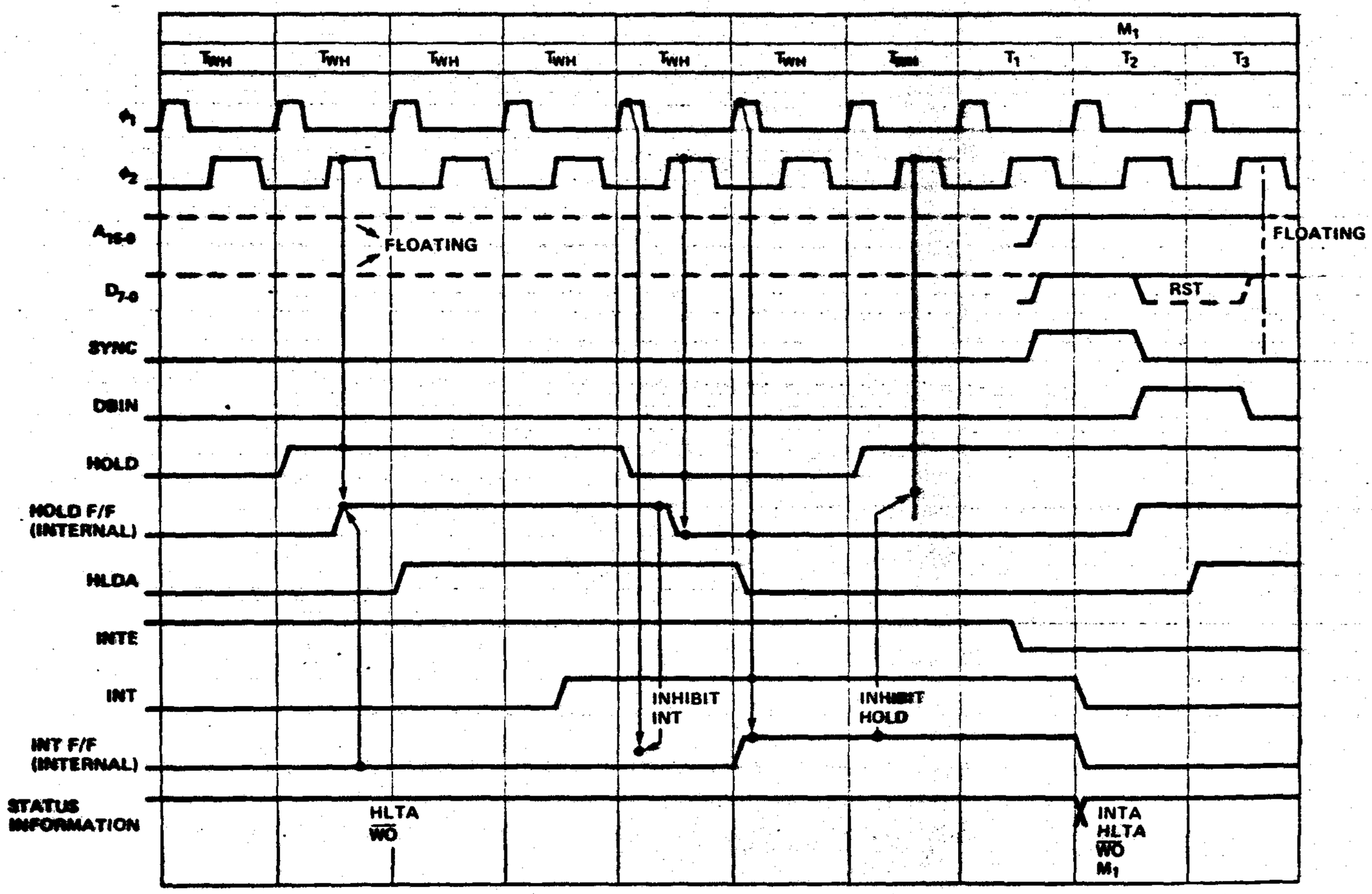
HOLD Operation (DAD)



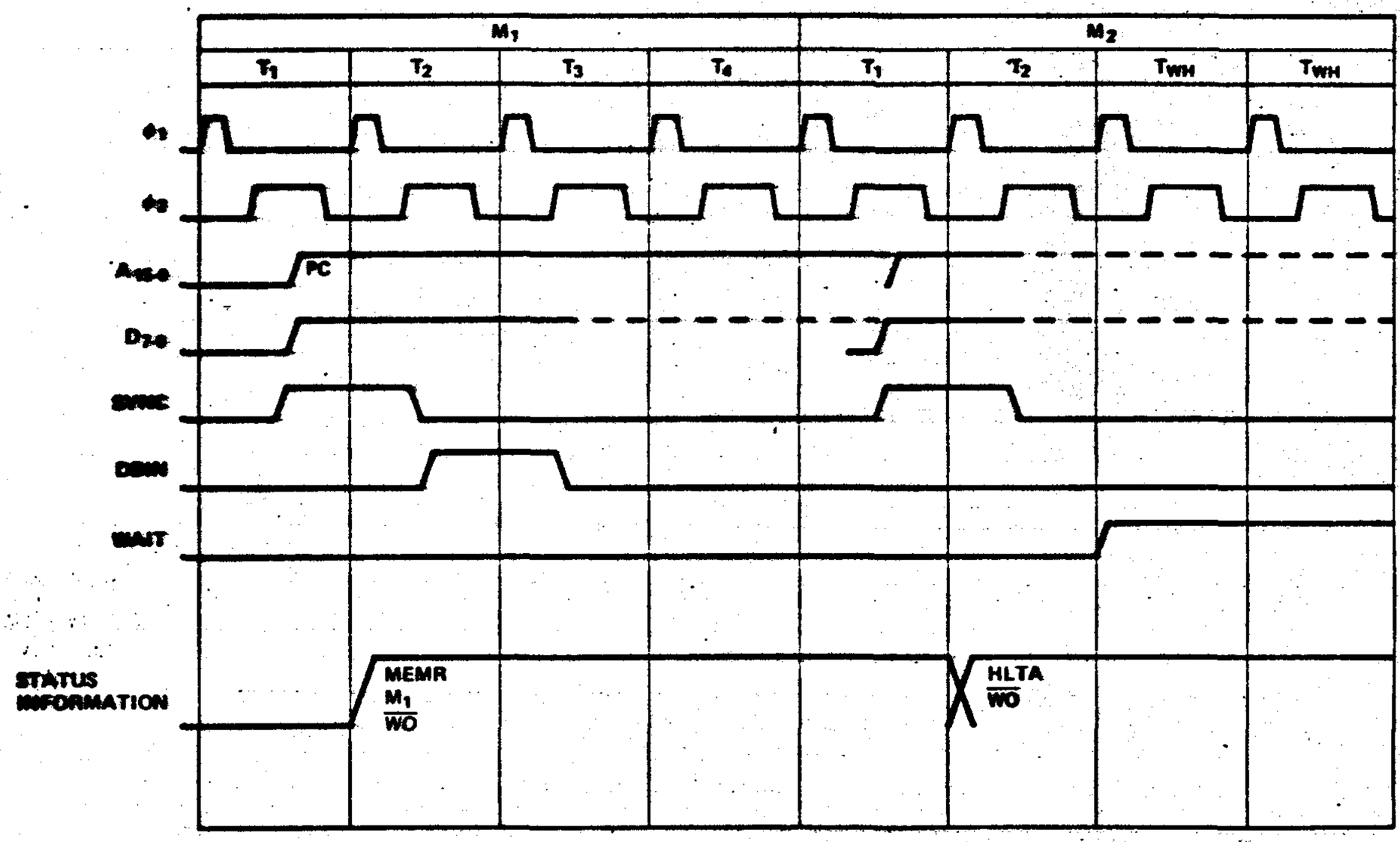
g. Interrupt



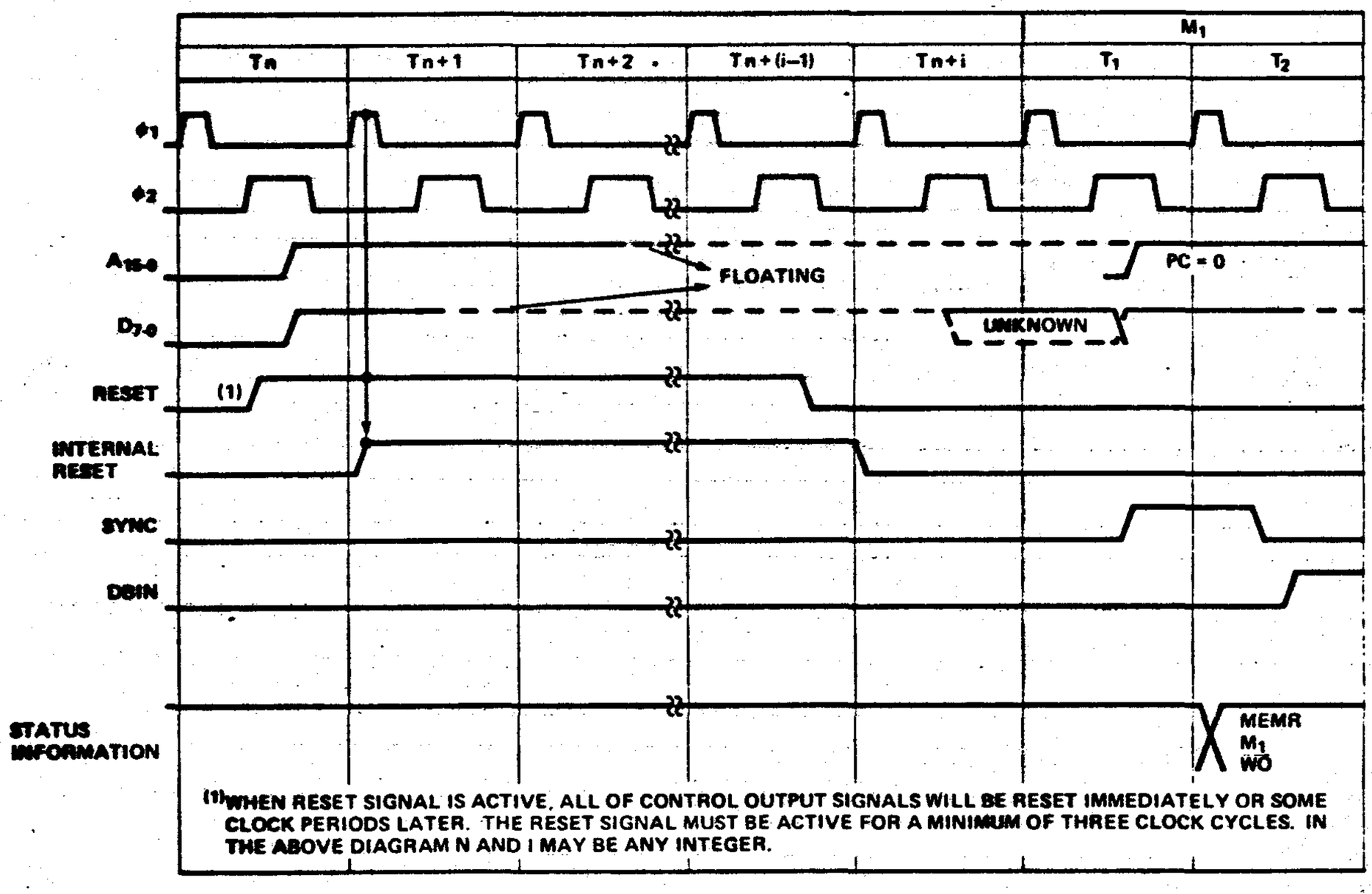
h. Relation between HOLD and INT in the HALT state



i. HALT Instruction

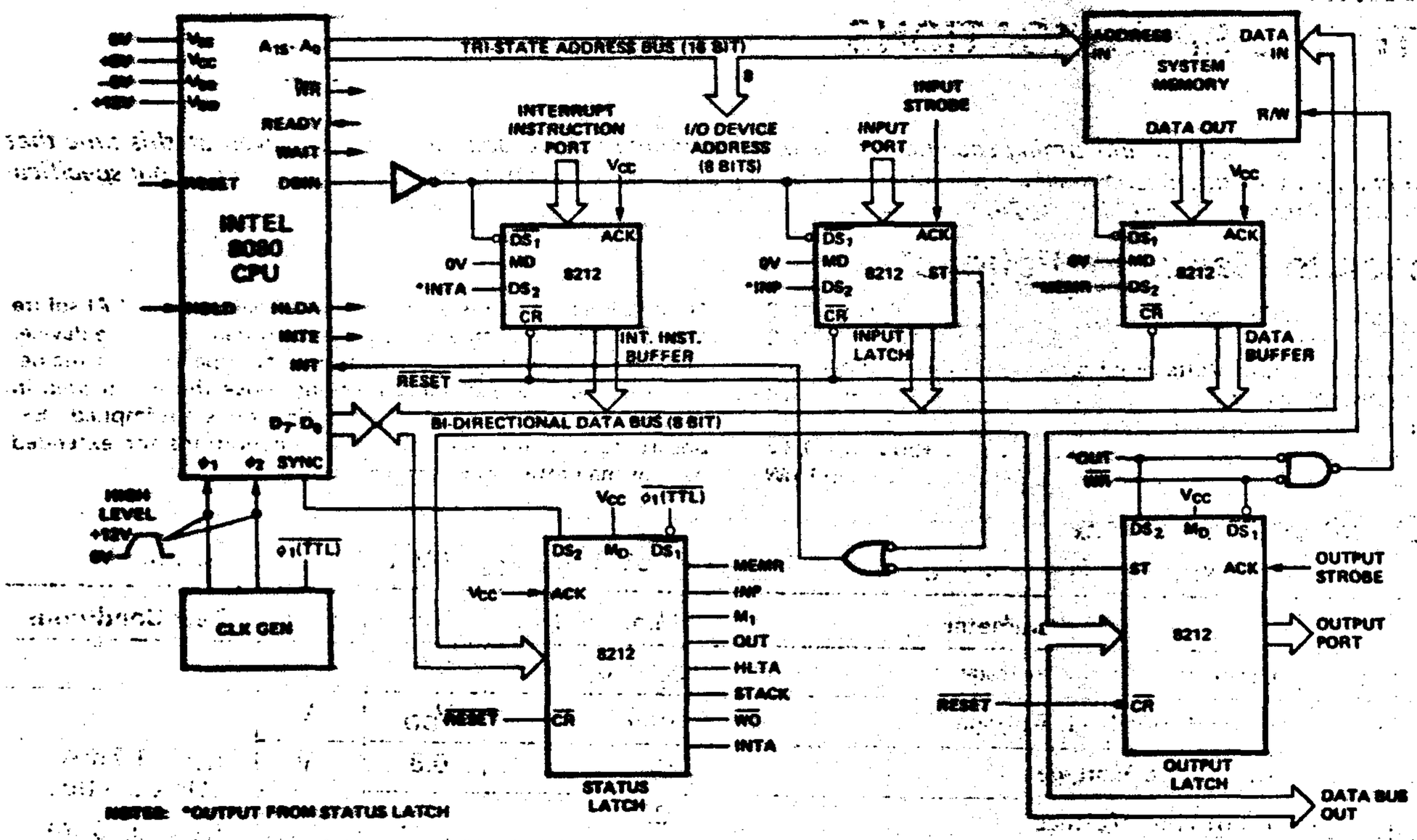


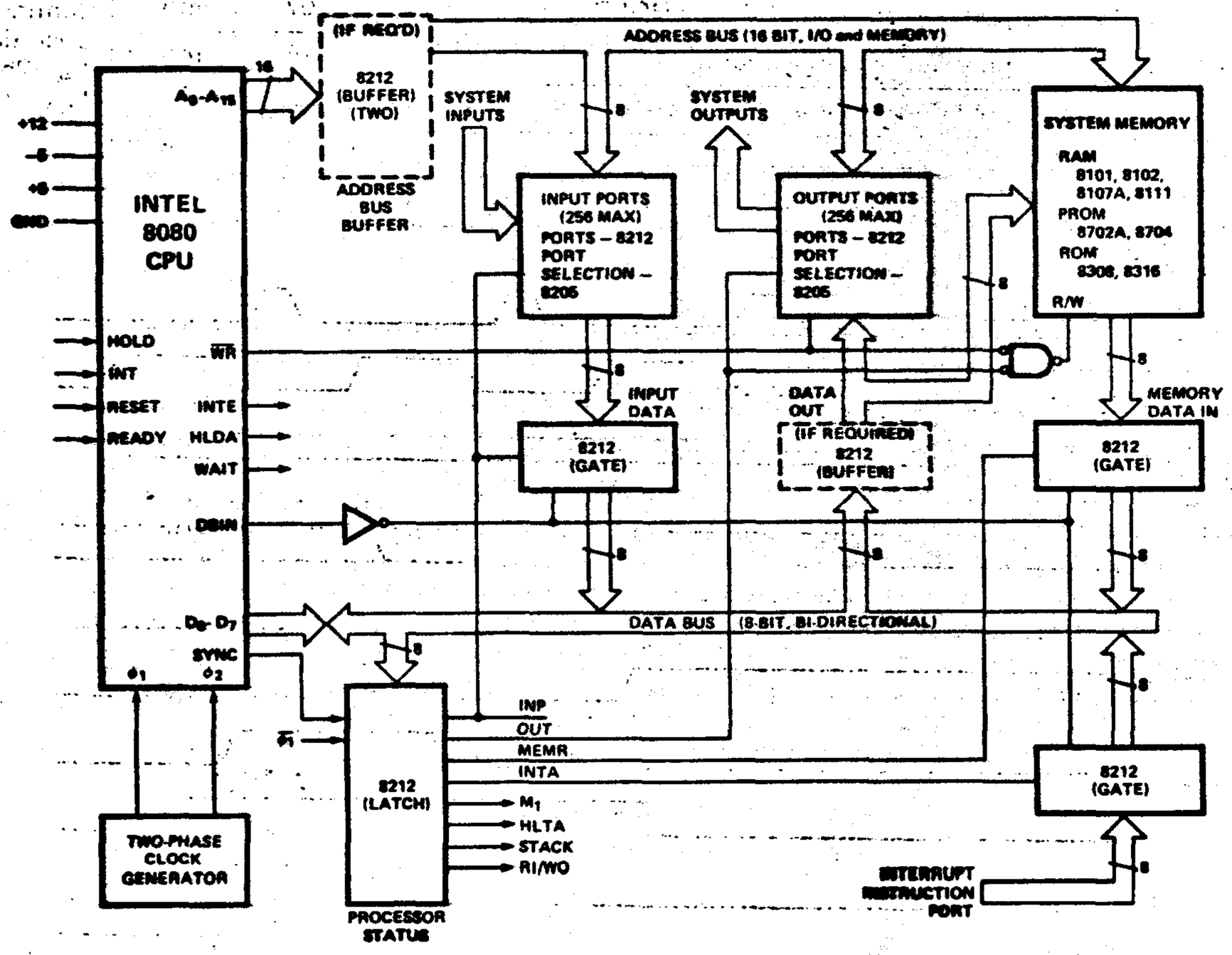
j. RESET



(1) WHEN RESET SIGNAL IS ACTIVE, ALL OF CONTROL OUTPUT SIGNALS WILL BE RESET IMMEDIATELY OR SOME CLOCK PERIODS LATER. THE RESET SIGNAL MUST BE ACTIVE FOR A MINIMUM OF THREE CLOCK CYCLES. IN THE ABOVE DIAGRAM N AND I MAY BE ANY INTEGER.

APPENDIX IV
BASIC SYSTEM BLOCK DIAGRAM





Typical 8080 System Block Diagram

APPENDIX V
ELECTRICAL SPECIFICATIONS

NOTE: This electrical and timing specification is only preliminary. No assurance can be given at this time that some changes will not occur during the engineering testing and characterization of this product. The final specification will be released in late May, 1974.

Absolute Maximum Ratings*

Temperature Under Bias . . . . . 0°C to 70°C
Storage Temperature . . . . . −65°C to +150°C
All Input or Output Voltages with respect to the most negative supply voltage, V_{BB} . . . . . +25V to −0.3V
Supply Voltages V_{DD} and V_{SS} with respect to V_{BB} . . . . . +20V to −0.3V
Power Dissipation 1.0W

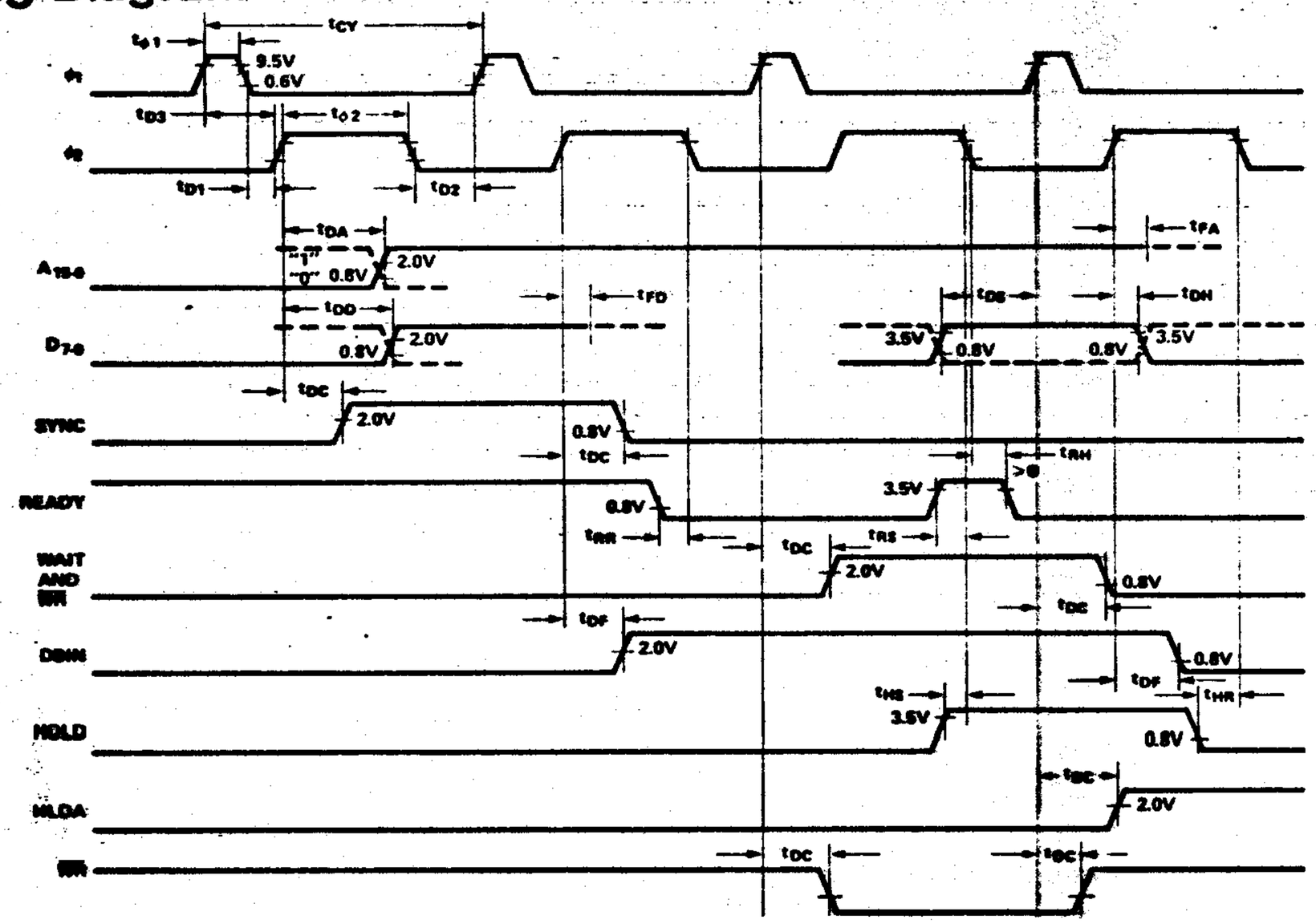
*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. Characteristics

T_A = 0°C to +70°C, V_{DD} = +12V ± 5%, V_{CC} = +5V ± 5%, V_{BB} = −5V ± 5%, V_{SS} = 0V, unless otherwise noted.

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
V_{ILC}	Clock Input Low Voltage	V_{SS}−1.0		0.6	V	
V_{IHC}	Clock Input High Voltage	10.4		V_{DD}	V	
V_{IL}	Input Low Voltage	V_{SS}		0.8	V	I_{OL} = 1.7mA On Data Bus
V_{IH}	Input High Voltage	3.3		V_{CC}	V	I_{OL} = 0.75mA On All Others
V_{OL}	Output Low Voltage			0.45	V	
V_{OH}	Output High Voltage	2.4			V	I_{OH} = 100μA
I_{DD1}	Power Supply Current (V_{DD}) during $\overline{HOLD}$			60	mA	Continuous Operation T_A = 25°C T_{CY} = 480ns
I_{DD2}	Power Supply Current (V_{DD}) during HOLD			67	mA	
I_{CC1}	Power Supply Current (V_{CC}) during $\overline{HOLD}$			75	mA	
I_{CC2}	Power Supply Current (V_{CC}) during HOLD			73	mA	
I_{BB}	Power Supply Current			1	mA	

Timing Diagram



A.C. Characteristics

T_A = 0°C to 70°C, V_{DD} = +12V ± 5%, V_{CC} = +5V ± 5%, V_{BB} = −5V ± 5%, V_{SS} = 0V, unless otherwise noted.

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
t_{CY}	Clock Period	.48		2.0	μs	
t_R,t_F	Clock Rise and Fall Times			50	ns	
$t_{\phi 1}$	Pulse Width of ϕ_1	60			ns	
$t_{\phi 2}$	Pulse Width of ϕ_2	220			ns	
t_{D1}	Clock Delay between ϕ_1 and ϕ_2	0			ns	
t_{D2}	Clock Delay between ϕ_2 and ϕ_1	70			ns	
t_{DA}	Address Output Delay from ϕ_2			200	ns	1 T^2L and CL=50pF
t_{DD}	Data Output Delay from ϕ_2			220	ns	
t_{DC}	Control Signal Output Delay from ϕ_1 or ϕ_2 (SYNC, $\overline{WR}$, WAIT and HLDA)			120	ns	
t_{DF}	DBIN Output Delay from ϕ_2	25		140	ns	
t_{D3}	Clock Delay ϕ_1 to ϕ_2	130			ns	
t_{DS}	Data Setup Time to ϕ_1 during DBIN	20			ns	
t_{DH}	Data Hold Time from ϕ_2 during DBIN	t_{DF}			ns	
t_{RR}	Ready Reset Time during ϕ_2	120			ns	
t_{RS}	Ready Setup Time during ϕ_2	110			ns	
t_{HR}	Hold Reset Time during ϕ_2	110			ns	
t_{HS}	Hold Setup Time during ϕ_2	70			ns	
t_{FA}	Address Delay to Enter Hold State			150	ns	
t_{FD}	Data Delay to Enter Hold State			150	ns	

Capacitance T_A = 25°C; Unmeasured Pins Grounded

Symbol	Test	Limit (pF)	
		Typ.	Max.
$C_{\phi 1}$	Clock 1 Capacitance	10	20
$C_{\phi 2}$	Clock 2 Capacitance	10	20
C_{IN}	Input Capacitance	4	8
C_{OUT}	Output Capacitance (Address In High Impedance State)	5	10

PRELIMINARY

APPENDIX VI
MEMORY AND PERIPHERAL DEVICES FOR THE MCS-80 MICROPROCESSOR FAMILY

intel®

Silicon Gate MOS 8101

1024 BIT FULLY DECODED STATIC MOS RANDOM ACCESS MEMORY

- 256 x 4 Organization to Meet Needs for Small System Memories
- Access Time — 850ns Max.
- Single +5V Supply Voltage
- Directly TTL Compatible — All Inputs and Output
- Static MOS — No Clocks or Refreshing Required
- Simple Memory Expansion — Chip Enable Input
- Fully Decoded — on Chip Address Decode
- Inputs Protected — All Inputs Have Protection Against Static Charge
- Low Cost Packaging — 22 Pin Plastic Dual-In-Line Configuration
- Low Power — Typically 150 mW
- Three-State Output — OR-Tie Capability

The Intel 8101 is a 256 word by 4 bit static random access memory element using normally off N-channel MOS devices integrated on a monolithic array. It uses fully DC stable (static) circuitry and therefore requires no clocks or refreshing to operate. The data is read out nondestructively and has the same polarity as the input data.

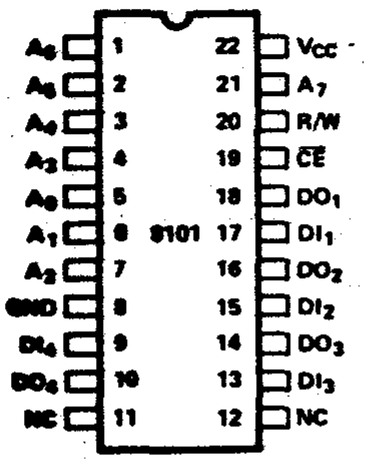
The 8101 is designed for memory applications where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, outputs, and a single +5V supply. A separate chip enable ($\overline{CE}$) lead allows easy selection of an individual package when outputs are OR-tied.

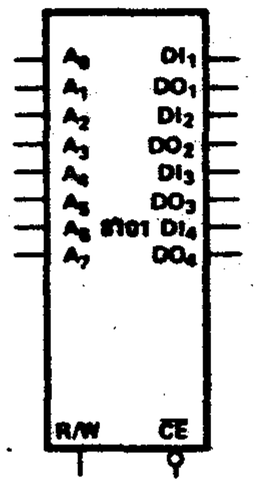
The Intel 8101 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

Intel's silicon gate technology also provides excellent protection against contamination. This permits the use of low cost silicone packaging.

PIN CONFIGURATION



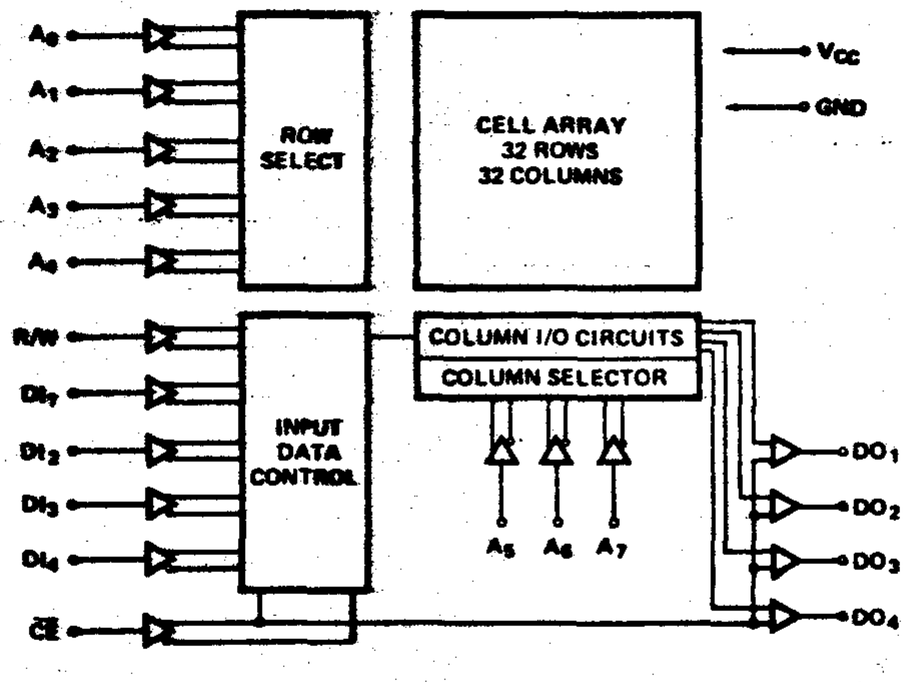
LOGIC SYMBOL



PIN NAMES

DI_1- DI_4	DATA INPUTS	$\overline{CE}$	CHIP ENABLE
A_0- A_7	ADDRESS INPUTS	DO_1- DO_4	DATA OUTPUTS
R/W	READ/WRITE INPUT	V_{CC}	POWER (+5V)

BLOCK DIAGRAM



Silicon Gate MOS 8111

PRELIMINARY

1024 BIT FULLY DECODED STATIC MOS RANDOM ACCESS MEMORY WITH COMMON DATA I/O

- **Organization 256 Words by 4 Bits**
- **Common Data Input and Output**
- **Single +5V Supply Voltage**
- **Directly TTL Compatible — All Inputs and Output**
- **Static MOS — No Clocks or Refreshing Required**
- **Access Time — 850ns Max.**
- **Simple Memory Expansion — Chip Enable Input**
- **Fully Decoded — On Chip Address Decode**
- **Inputs Protected — All Inputs Have Protection Against Static Charge**
- **Low Cost Packaging — 18 Pin Plastic Dual-In-Line Configuration**
- **Low Power — Typically 150 mW**
- **Three-State Output — OR-Tie Capability**

The Intel 8111 is a 256 word by 4 bit static random access memory element using normally off N-channel MOS devices integrated on a monolithic array. It uses fully DC stable (static) circuitry and therefore requires no clocks or refreshing to operate. The data is read out nondestructively and has the same polarity as the input data. Common input/output pins are provided.

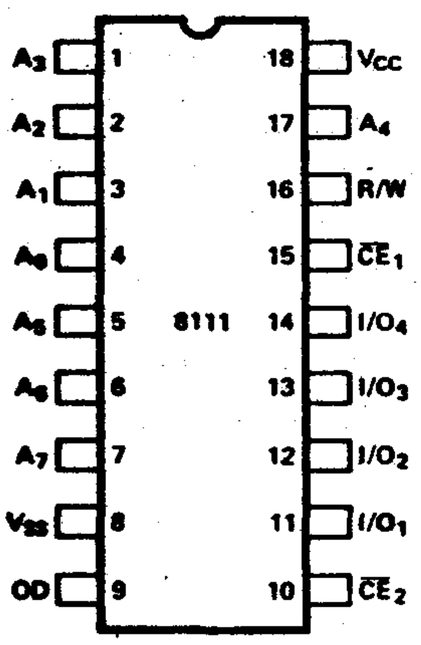
The 8111 is designed for microcomputer memory applications in small systems where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, outputs, and a single +5V supply. A separate chip enable ($\overline{CE}$) lead allows easy selection of an individual package when outputs are OR-tied.

The Intel 8111 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

Intel's silicon gate technology also provides excellent protection against contamination. This permits the use of low cost silicone packaging.

PIN CONFIGURATION



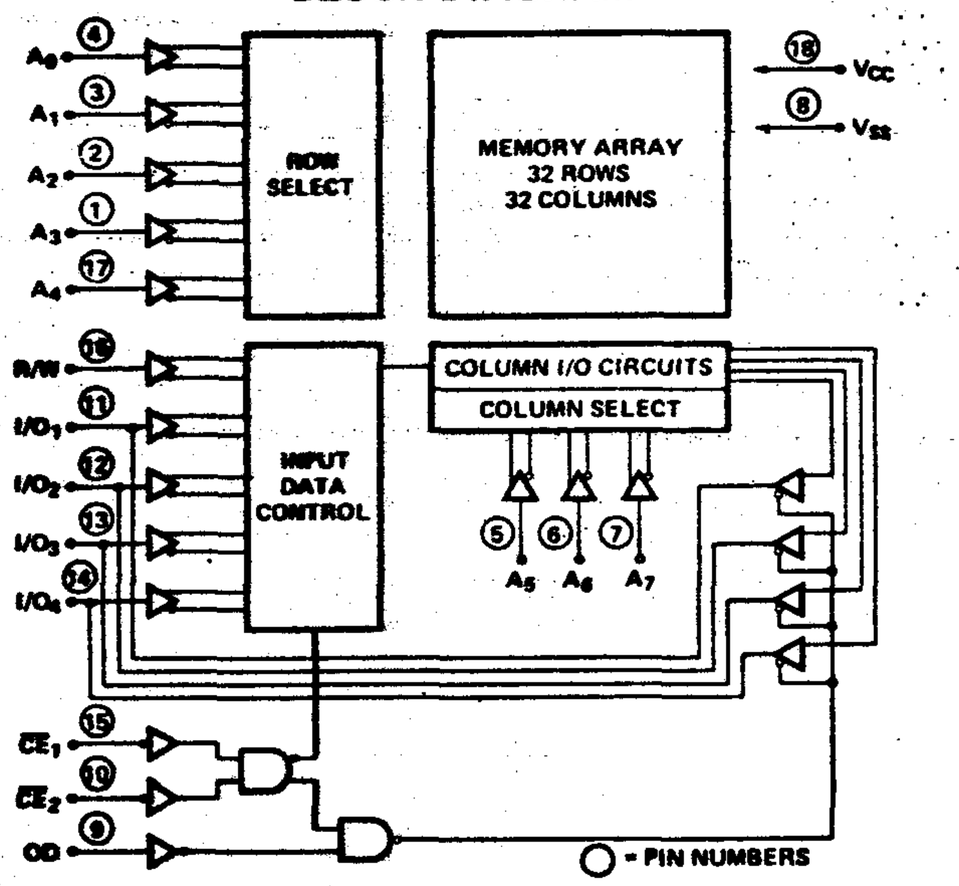
PIN NAMES

A_0-A_7	ADDRESS INPUTS
OD	OUTPUT DISABLE
R/W	READ/WRITE INPUT
$\overline{CE}_1$	CHIP ENABLE 1
$\overline{CE}_2$	CHIP ENABLE 2
I/O_1- I/O_4	INPUT/OUTPUT

ENABLE

OUTPUT:	$OD \cdot \overline{CE}_1 \cdot \overline{CE}_2$
WRITE:	$W \cdot \overline{CE}_1 \cdot \overline{CE}_2$

BLOCK DIAGRAM



Silicon Gate MOS 8102

1024 BIT FULLY DECODED STATIC MOS RANDOM ACCESS MEMORY

- **Single +5 Volts Supply Voltage**
- **Directly TTL Compatible — All Inputs and Output**
- **Static MOS — No Clocks or Refreshing Required**
- **Low Power — Typically 150 mW**
- **Access Time — 850ns Max.**
- **Three-State Output — OR-Tie Capability**
- **Simple Memory Expansion — Chip Enable Input**
- **Fully Decoded — On Chip Address Decode**
- **Inputs Protected — All Inputs Have Protection Against Static Charge**
- **Low Cost Packaging — 16 Pin Plastic Dual-In-Line Configuration**

The Intel 8102 is a 1024 word by one bit static random access memory element using normally off N-channel MOS devices integrated on a monolithic array. It uses fully DC stable (static) circuitry and therefore requires no clocks or refreshing to operate. The data is read out nondestructively and has the same polarity as the input data.

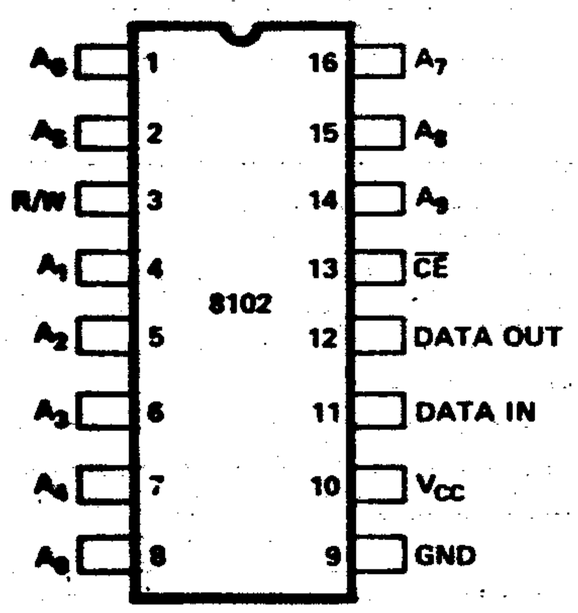
The 8102 is designed for microcomputer memory applications where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, output, and a single +5 volt supply. A separate chip enable ($\overline{CE}$) lead allows easy selection of an individual package when outputs are OR-tied.

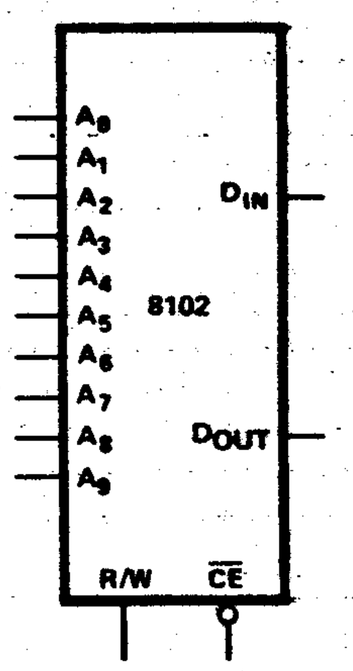
The Intel 8102 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

Intel's silicon gate technology also provides excellent protection against contamination. This permits the use of low cost silicone packaging.

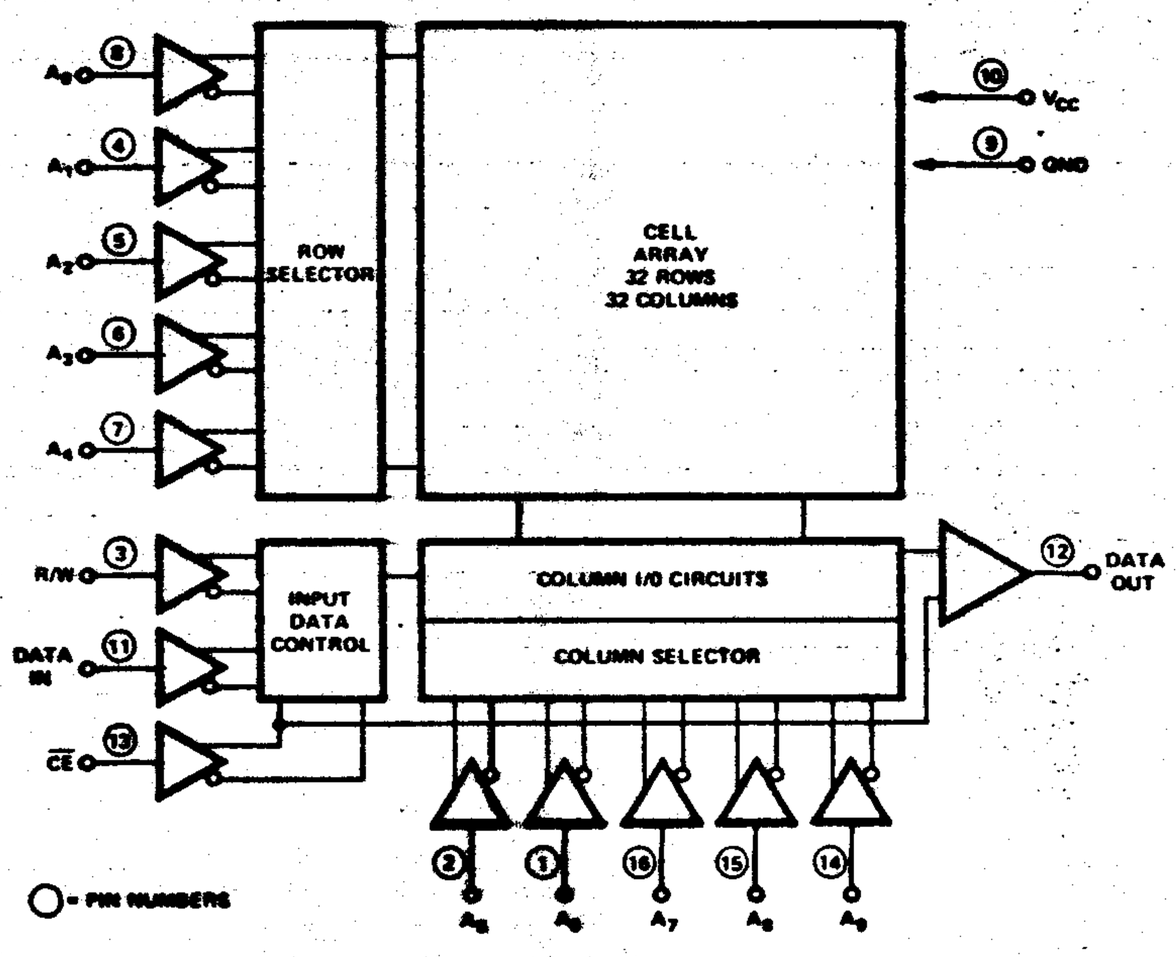
PIN CONFIGURATION



LOGIC SYMBOL



BLOCK DIAGRAM



PIN NAMES

D_{IN}	DATA INPUT	$\overline{CE}$	CHIP ENABLE
A_0-A_9	ADDRESS INPUTS	D_{OUT}	DATA OUTPUT
R/W	READ/WRITE INPUT	V_{CC}	POWER (+5V)

Silicon Gate MOS 8107A

FULLY DECODED RANDOM ACCESS 4096 BIT DYNAMIC MEMORY

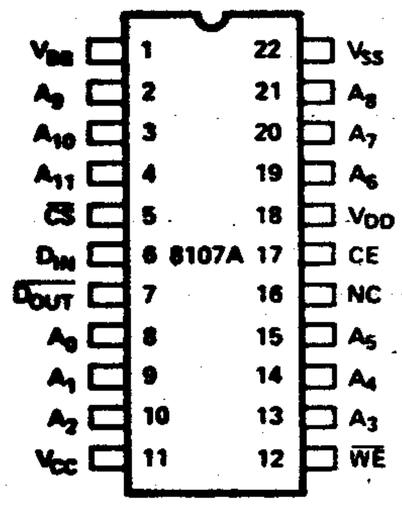
- Low Cost Per Bit
- Low Standby Power — Typical 7 μW/Bit
- Easy System Interface
- Only One High Voltage Input Signal — Chip Enable
- All Other Inputs are TTL Compatible
- Address Registers Incorporated on the Chip
- Simple Memory Expansion — Chip Select Input Lead
- Fully Decoded — On Chip Address Decode
- Output Is Three State and Compatible with Low Power TTL Gates
- Ceramic 22-Pin DIP

The 8107A is a 4096 word by 1 bit dynamic RAM. It was designed for microcomputer memory applications where very low cost and large bit storage are important design objectives. The 8107A uses dynamic circuitry which reduces the operating and standby power dissipation.

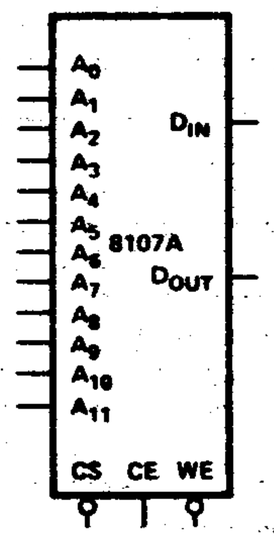
Reading information from the memory is non-destructive. Refreshing is accomplished by performing one read cycle on each of the 64 row addresses. Each row address must be refreshed every one millisecond. The memory is refreshed whether Chip Select is a logic one or a logic zero.

The 8107A is fabricated with N-channel silicon gate technology. This technology allows the design and manufacture of devices using minimum size transistors that have the same performance as devices using much larger transistors.

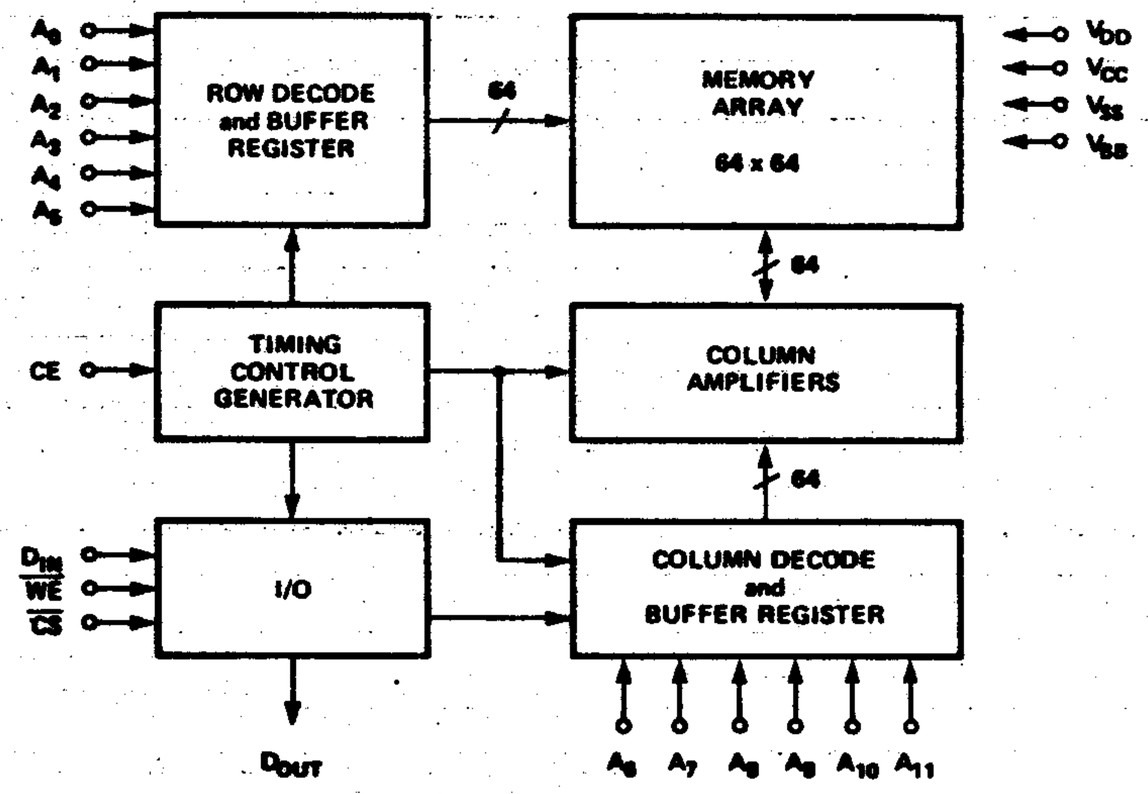
PIN CONFIGURATION



LOGIC SYMBOL



BLOCK DIAGRAM



PIN NAMES

D_{IN}	DATA INPUT	CE	CHIP ENABLE
A_0-A_{11}	ADDRESS INPUTS	D_{OUT}	DATA OUTPUT
$\overline{WE}$	WRITE ENABLE	V_{CC}	POWER (+5V)
$\overline{CS}$	CHIP SELECT	NC	NOT CONNECTED

Silicon Gate MOS 8302

2048 BIT MASK PROGRAMMABLE READ ONLY MEMORY

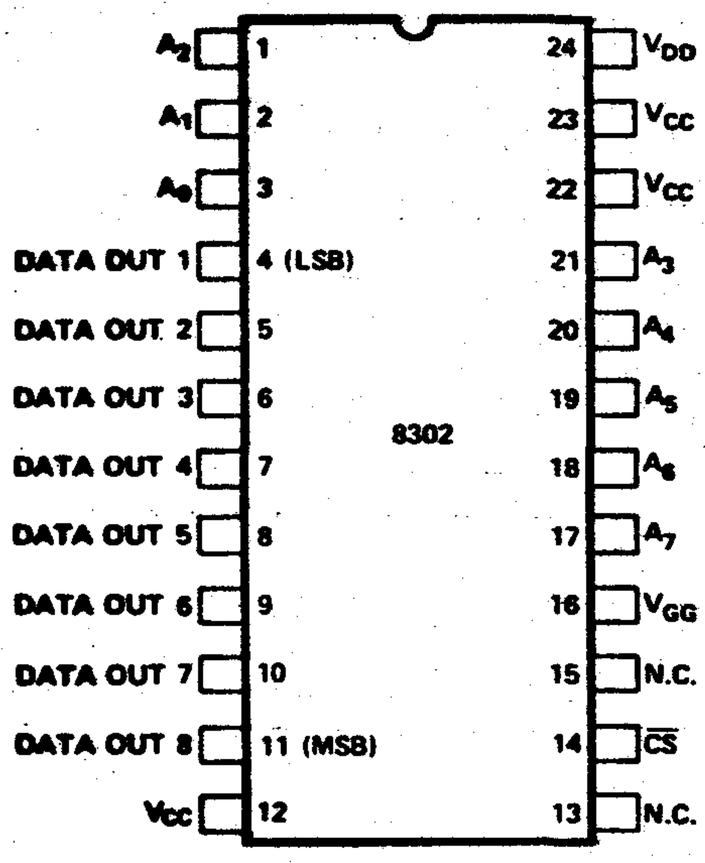
- **Fully Decoded, 256 x 8 Organization**
- **Inputs and Outputs TTL Compatible**
- **Three-State Output — OR-Tie Capability**
- **Static MOS — No Clocks Required**
- **Simple Memory Expansion — Chip Select Input Lead**
- **24-Pin Dual-In-Line Hermetically Sealed Ceramic Package**

The Intel 8302 is a fully decoded 256 word by 8 bit metal mask ROM. It is ideal for large volume production runs of microcomputer systems initially using the 8702A erasable and electrically programmable ROM. **The 8302** has the same pinning as the 8702A.

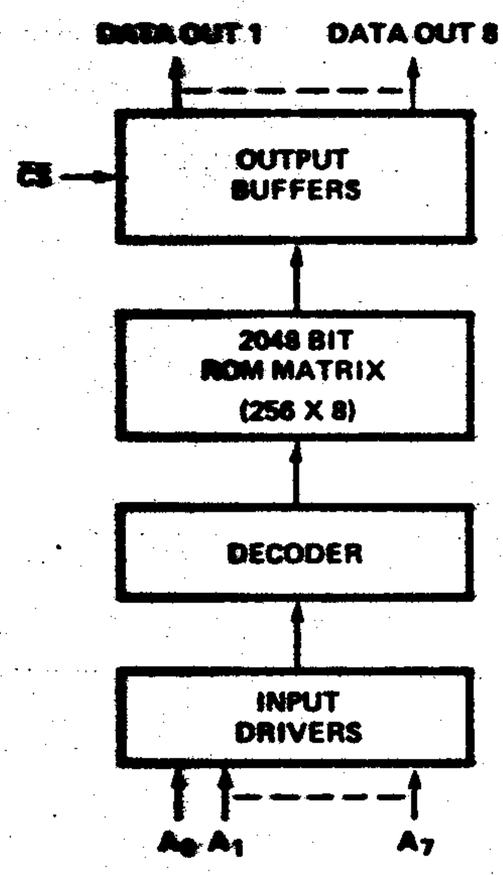
The 8302 is entirely static — no clocks are required. Inputs and outputs of the 8302 are TTL compatible. The output is three-state for OR-tie capability. A separate chip select input allows easy memory expansion. **The 8302** is packaged in a 24 pin dual-in-line hermetically sealed ceramic package.

The 8302 is fabricated with p-channel silicon gate technology. This low threshold allows the design and production of higher performance MOS circuits and provides a higher functional density on a monolithic chip than conventional MOS technologies.

PIN CONFIGURATION



BLOCK DIAGRAM



PIN NAMES

A_0-A_7	ADDRESS INPUTS
$\overline{CS}$	CHIP SELECT INPUT
DO_1-DO_8	DATA OUTPUTS

Silicon Gate MOS 8308

PRELIMINARY

8192 BIT STATIC MOS READ ONLY MEMORY

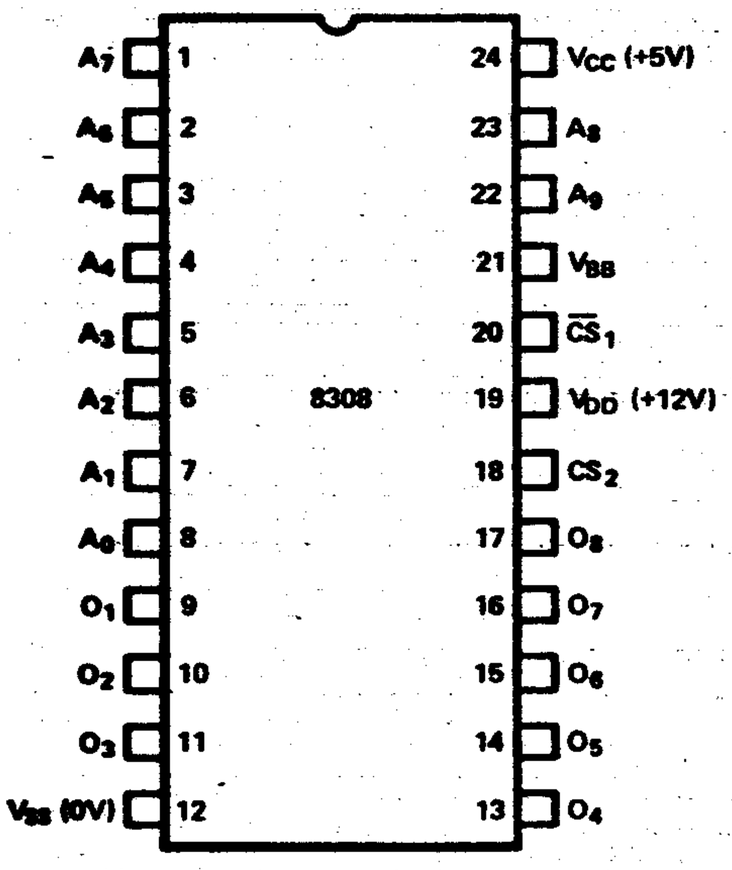
Organization -- 1024 Words x 8 Bits

- **Fast Access — 450 ns Maximum**
- **Directly Compatible with 8080 CPU at Maximum Processor Speed**
- **Two Chip Select Inputs for Easy Memory Expansion**
- **Directly TTL Compatible — All Inputs and Outputs**
- **Three State Output — OR-Tie Capability**
- **Fully Decoded — On Chip Decode**
- **Inputs Protected — All Have Protection Against Static Charge**

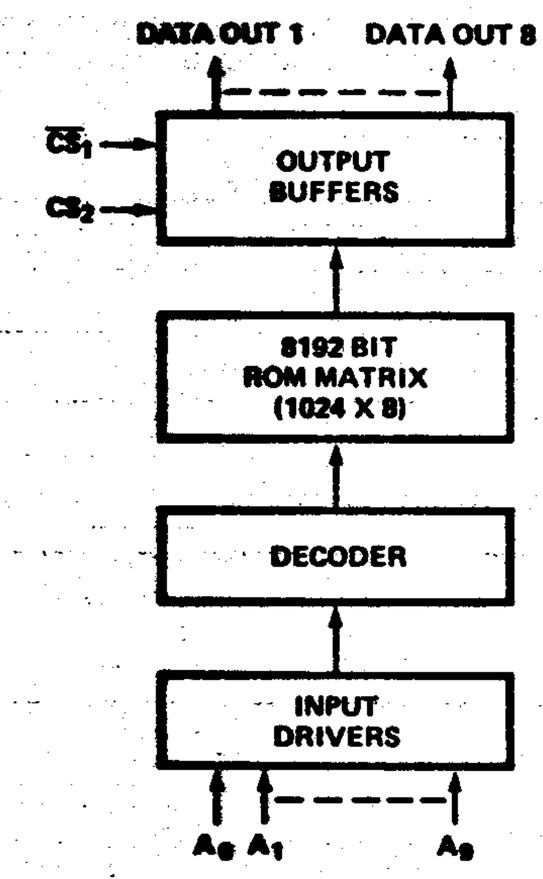
The Intel 8308 is an 8,192 bit static MOS Read Only Memory organized as **1024** words by 8 bits. This ROM **is designed** for 8080 microcomputer system applications where high performance, large bit storage, and simple interfacing are important design objectives. The inputs and outputs are fully TTL compatible.

The 8308 read only memory is fabricated with N-channel silicon gate technology. This technology provides **the designer** with high performance, easy-to-use MOS circuits.

PIN CONFIGURATION



BLOCK DIAGRAM



PIN NAMES

A_0- A_9	ADDRESS INPUTS
O_1- O_8	DATA OUTPUTS
CS_1, CS_2	CHIP SELECT INPUTS

Silicon Gate MOS ROM **8316**

16,384 BIT STATIC MOS READ ONLY MEMORY
Organization – 2048 Words x 8 Bits

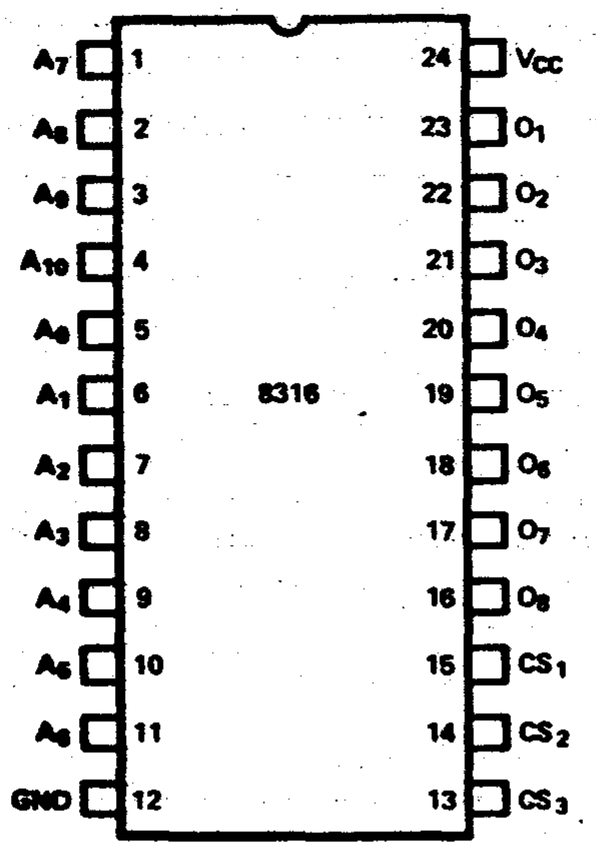
- **Single +5 Volts Power Supply Voltage**
- **Directly TTL Compatible — All Inputs and Outputs**
- **Low Power Dissipation of 10.7 μW/Bit Maximum**
- **Three Programmable Chip Select Inputs for Easy Memory Expansion**
- **Three-State Output — OR-Tie Capability**
- **Fully Decoded — On Chip Address Decode**
- **Inputs Protected — All Inputs Have Protection Against Static Charge**

The Intel 8316 is a 16,384 bit static MOS read only memory organized as **2048** words by 8 bits. This ROM **is designed** for microcomputer memory applications where **high performance,** large bit storage, and simple **interfacing** are important design objectives.

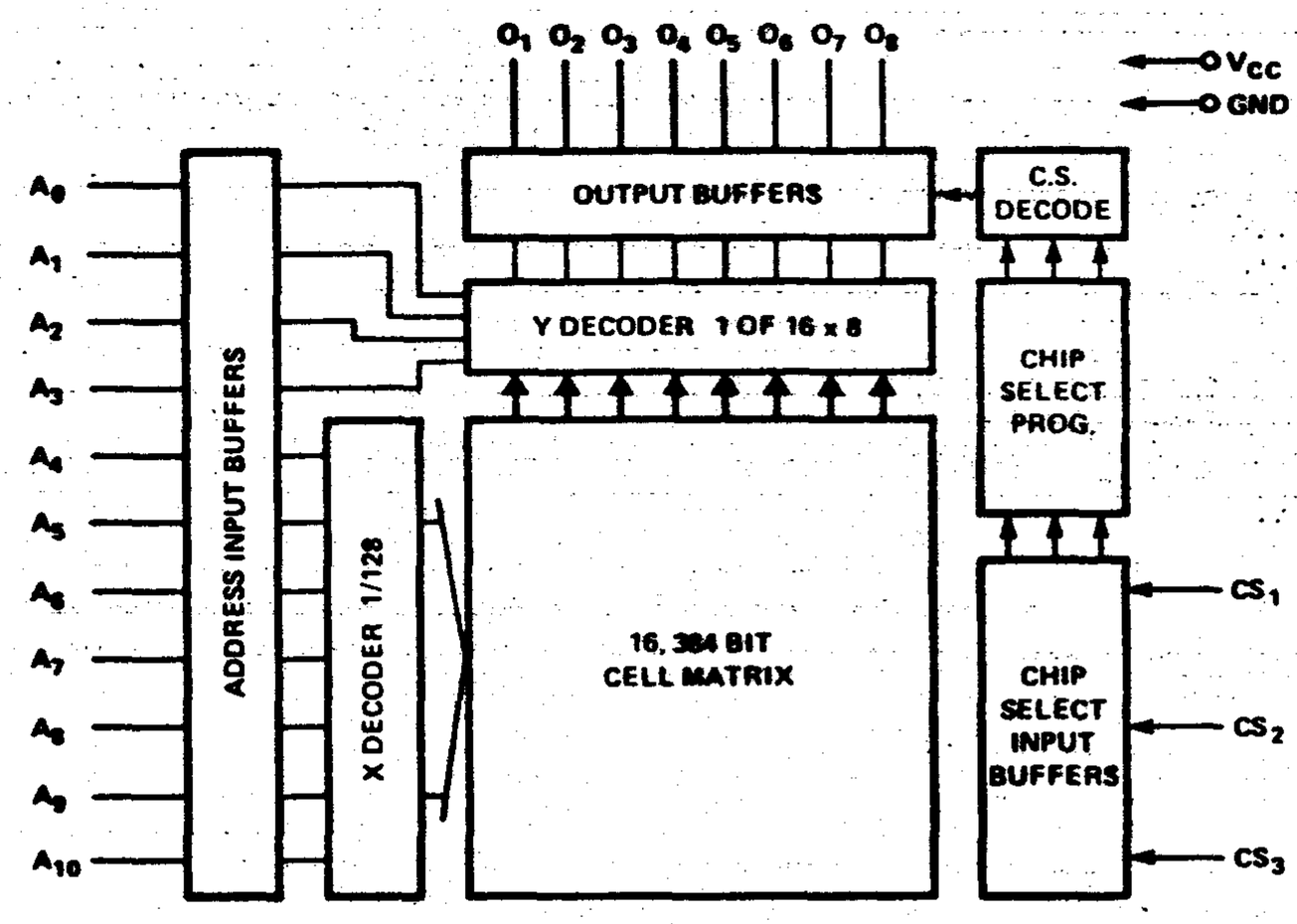
The inputs and outputs are fully TTL compatible. This device operates **with a single** +5V power supply. The **three** chip select inputs are programmable. Any combination of active **high** or low level chip select inputs **can be** defined and the desired chip select code is fixed during the masking process. These three programmable chip select inputs, as well as OR-tie compatibility on the outputs, facilitate easy memory expansion.

The 8316 read only memory is fabricated with N-channel silicon gate technology. This technology provides **the** designer with high performance, easy-to-use MOS circuits. Only **a single** +5V power supply is needed **and all** devices are directly TTL compatible.

PIN CONFIGURATION



BLOCK DIAGRAM



PIN NAMES

A_0- A_{10}	ADDRESS INPUTS
O_1- O_8	DATA OUTPUTS
CS_1- CS_3	PROGRAMMABLE CHIP SELECT INPUTS

Silicon Gate MOS 8702A

2048 BIT ERASABLE AND ELECTRICALLY REPROGRAMMABLE READ ONLY MEMORY

- **Fast Programming — 2 Minutes for All 2048 Bits**
- **Fully Decoded, 256 x 8 Organization**
- **Static MOS — No Clocks Required**
- **Inputs and Outputs TTL Compatible**
- **Three-State Output — OR-Tie Capability**
- **Simple Memory Expansion Chip Select Input Lead**

The 8702A is a 256 word by 8 bit electrically programmable ROM ideally suited for microcomputer system development where fast turn-around and pattern experimentation are important. The 8702A undergoes complete programming and functional testing on each bit position prior to shipment, thus insuring 100% programmability.

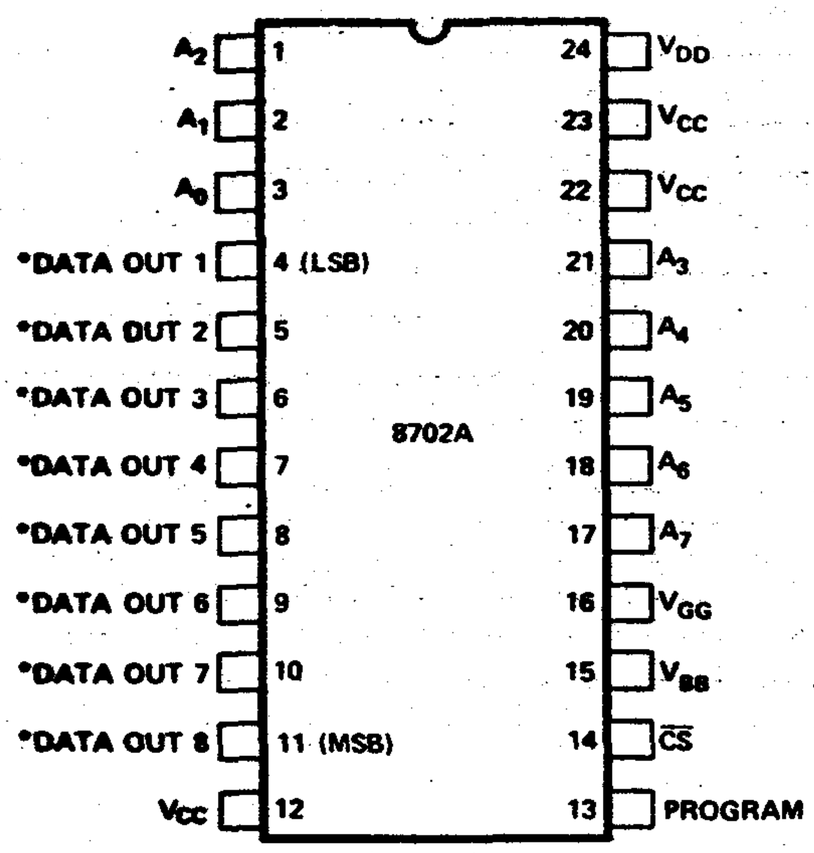
The 8702A is packaged in a 24 pin dual-in line package with a transparent quartz lid. The transparent quartz lid allows the user to expose the chip to ultraviolet light to erase the bit pattern. A new pattern can then be written into the device. This procedure can be repeated as many times as required.

The circuitry of the 8702A is entirely static; no clocks are required.

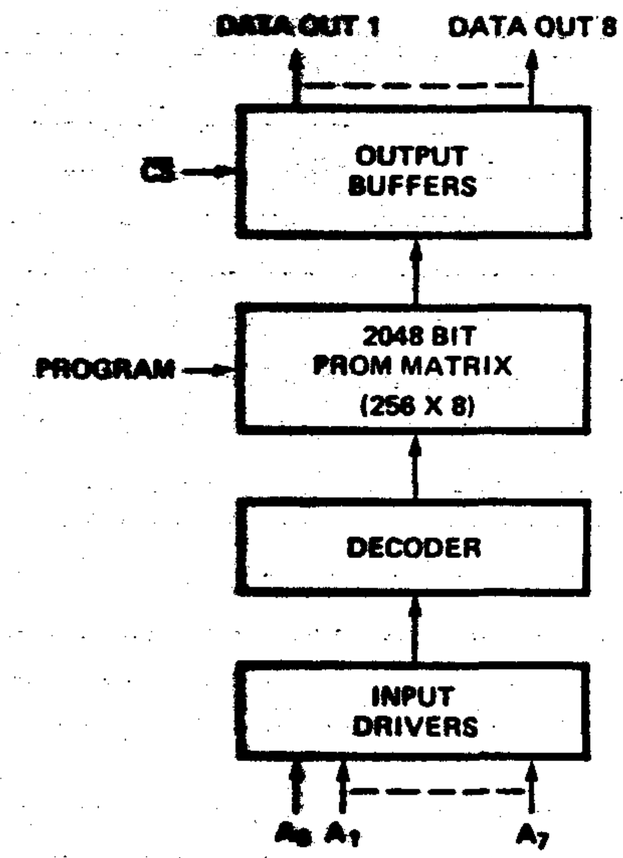
A pin-for-pin metal mask programmed ROM, the Intel 8302, is ideal for large volume production runs of systems initially using the 8702A.

The 8702A is fabricated with silicon gate technology. This low threshold technology allows the design and production of higher performance MOS circuits and provides a higher functional density on a monolithic chip than conventional MOS technologies.

PIN CONFIGURATION



BLOCK DIAGRAM



*THIS PIN IS THE DATA INPUT LEAD DURING PROGRAMMING.

PIN NAMES

A_0-A_7	ADDRESS INPUTS
$\overline{CS}$	CHIP SELECT INPUT
DO_1- DO_2	DATA OUTPUTS

intel®

Schottky Bipolar 8604

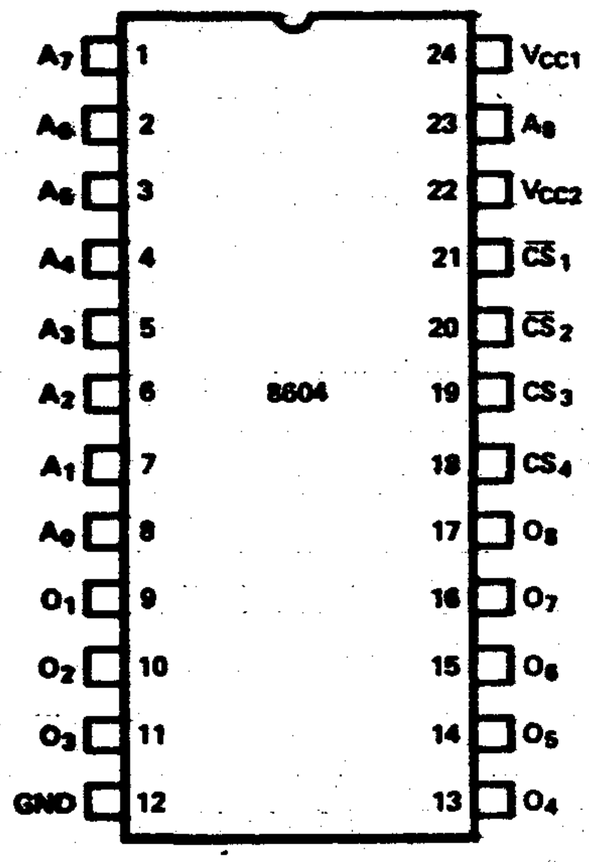
PRELIMINARY

HIGH SPEED ELECTRICALLY PROGRAMMABLE 4096 BIT READ ONLY MEMORY

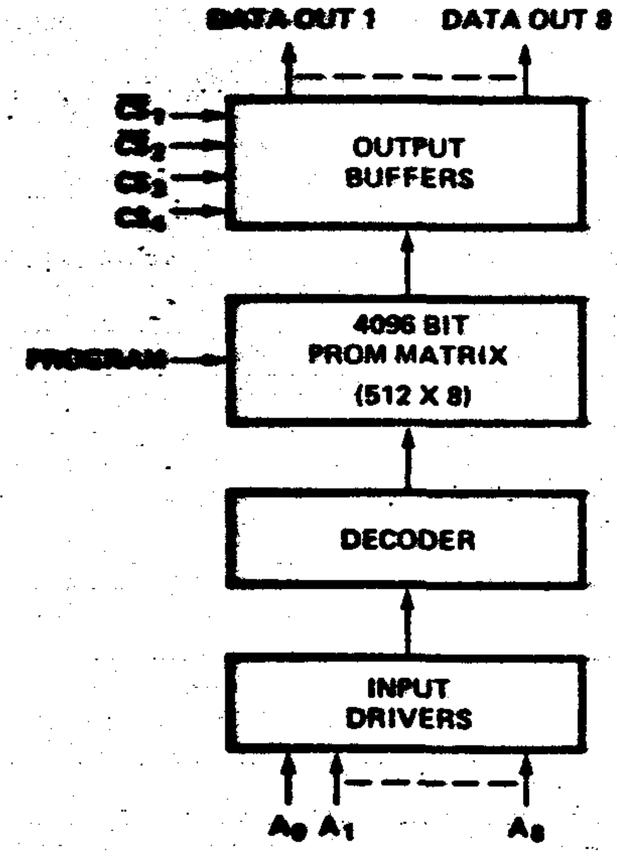
- 512 x 8 Organization for Microcomputer System Program Storage
- Fast Access Time — 100 ns
- Fully Decoded — On Chip Address Decode and Buffer

The 8604 is a 512 x 8 electrically programmable ROM ideally suited for high performance microcomputer systems where fast turnaround is important for system program development and for small volumes of identical programs in production systems.

PIN CONFIGURATION



BLOCK DIAGRAM



PIN NAMES

A_0-A_8	ADDRESS INPUTS
$\overline{CS}_1$-$\overline{CS}_2$, CS_3-CS_4	CHIP SELECT INPUTS
O_1-O_8	DATA OUTPUTS

intel Silicon Gate MOS 8704

PRELIMINARY

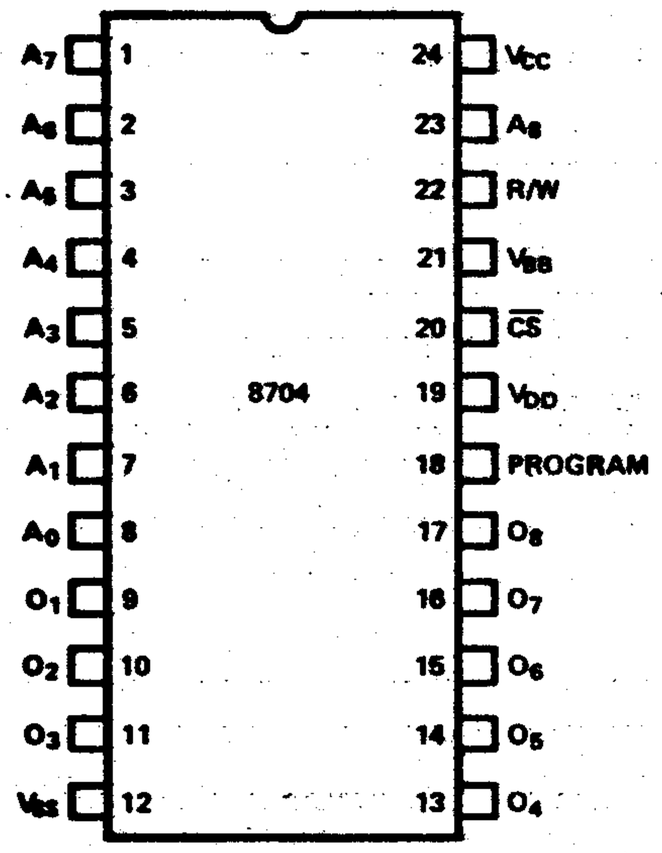
4096 BIT ERASABLE AND ELECTRICALLY REPROGRAMMABLE READ ONLY MEMORY

- **Fast Programming with only One High Voltage Pulse per Bit**
- **Low Power During Programming**
- **Fully Decoded, 512 x 8 Organization**
- **Access Time — 500 ns**
- **Static — No Clocks Required**
- **Inputs and Ouptuts TTL Compatible During Both Read and Program Modes**
- **Three-State Output — OR-Tie Capability**

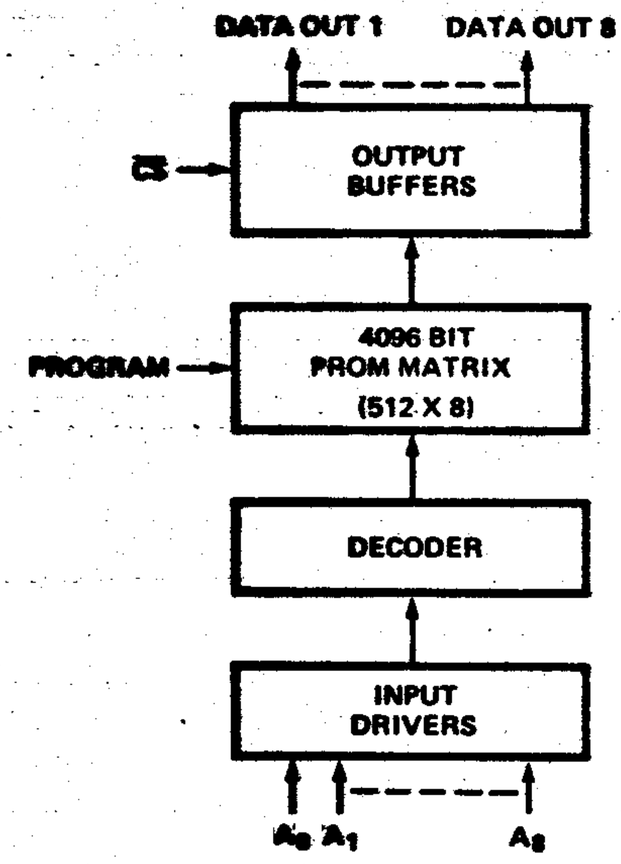
The 8704 is a high speed 512 word by 8 bit electrically programmable ROM ideally suited for microcomputer system development where fast access and low power are required.

The 8704 is packaged in a 24 pin dual-in-line package with transparent quartz lid. The transparent lid allows the user to expose the chip to ultraviolet light to erase the bit pattern. A new pattern can then be written into the device.

PIN CONFIGURATION



BLOCK DIAGRAM



PIN NAMES

A_0-A_8	ADDRESS INPUTS
O_1-O_8	DATA OUTPUTS
R/W	READ/WRITE
$\overline{CS}$	CHIP SELECT INPUTS

Schottky Bipolar 8205

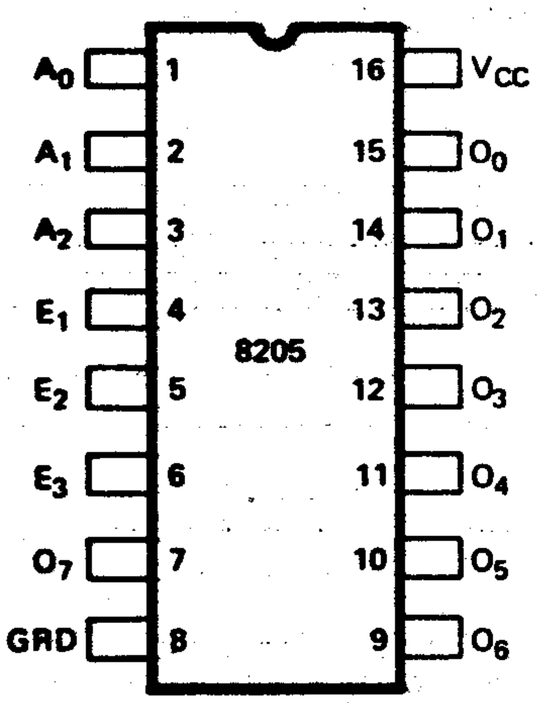
HIGH SPEED 1 OUT OF 8 BINARY DECODER

- I/O Port or Memory Selector
- Simple Expansion — Enable Inputs
- High Speed Schottky Bipolar Technology — 18ns Max. Delay
- Directly Compatible with TTL Logic Circuits
- Low Input Load Current — .25 mA max., 1/6 Standard TTL Input Load
- Minimum Line Reflection — Low Voltage Diode Input Clamp
- Outputs Sink 10 mA min.
- 16-Pin Dual-In-Line Ceramic or Plastic Package

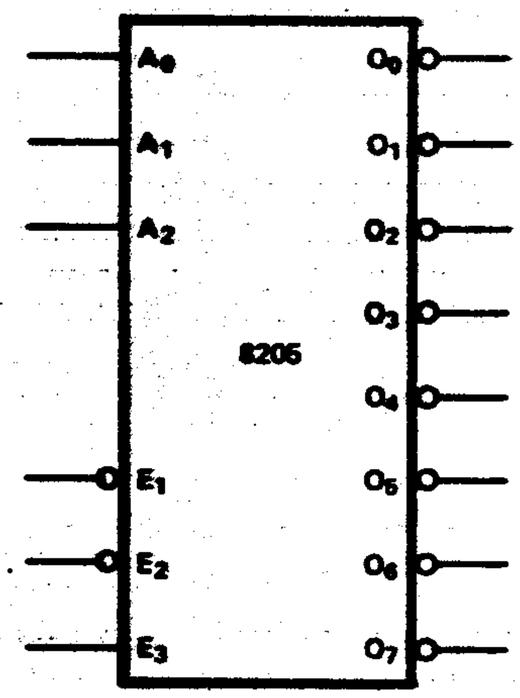
The 8205 decoder can be used for expansion of systems which utilize input ports, output ports, and memory components with active low chip select input. When the 8205 is enabled, one of its eight outputs goes "low", thus a single row of a memory system is selected. The 3 chip enable inputs on the 8205 allow easy system expansion. For very large systems, 8205 decoders can be cascaded such that each decoder can drive eight other decoders for arbitrary memory expansions.

The Intel 8205 is packaged in a standard 16 pin dual-in-line package; and its performance is specified over the temperature range of 0°C to +75°C, ambient. The use of Schottky barrier diode clamped transistors to obtain fast switching speeds results in higher performance than equivalent devices made with a gold diffusion process.

PIN CONFIGURATION



LOGIC SYMBOL



PIN NAMES

A_0-A_2	ADDRESS INPUTS
$\overline{E_1}$-$\overline{E_3}$	ENABLE INPUTS
$\overline{O_0}$-$\overline{O_7}$	DECODED OUTPUTS

ADDRESS			ENABLE			OUTPUTS							
A_0	A_1	A_2	E_1	E_2	E_3	0	1	2	3	4	5	6	7
L	L	L	L	L	H	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
L	H	L	L	L	H	H	H	L	H	H	H	H	H
H	H	L	L	L	H	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
H	L	H	L	L	H	H	H	H	H	H	L	H	H
L	H	H	L	L	H	H	H	H	H	H	H	L	H
H	H	H	L	L	H	H	H	H	H	H	H	H	L
X	X	X	L	L	L	H	H	H	H	H	H	H	H
X	X	X	H	L	L	H	H	H	H	H	H	H	H
X	X	X	L	H	L	H	H	H	H	H	H	H	H
X	X	X	H	H	L	H	H	H	H	H	H	H	H
X	X	X	H	L	H	H	H	H	H	H	H	H	H
X	X	X	L	H	H	H	H	H	H	H	H	H	H
X	X	X	H	H	H	H	H	H	H	H	H	H	H

Schottky Bipolar 8210

PRELIMINARY

TTL-TO-MOS LEVEL SHIFTER AND HIGH VOLTAGE CLOCK DRIVER

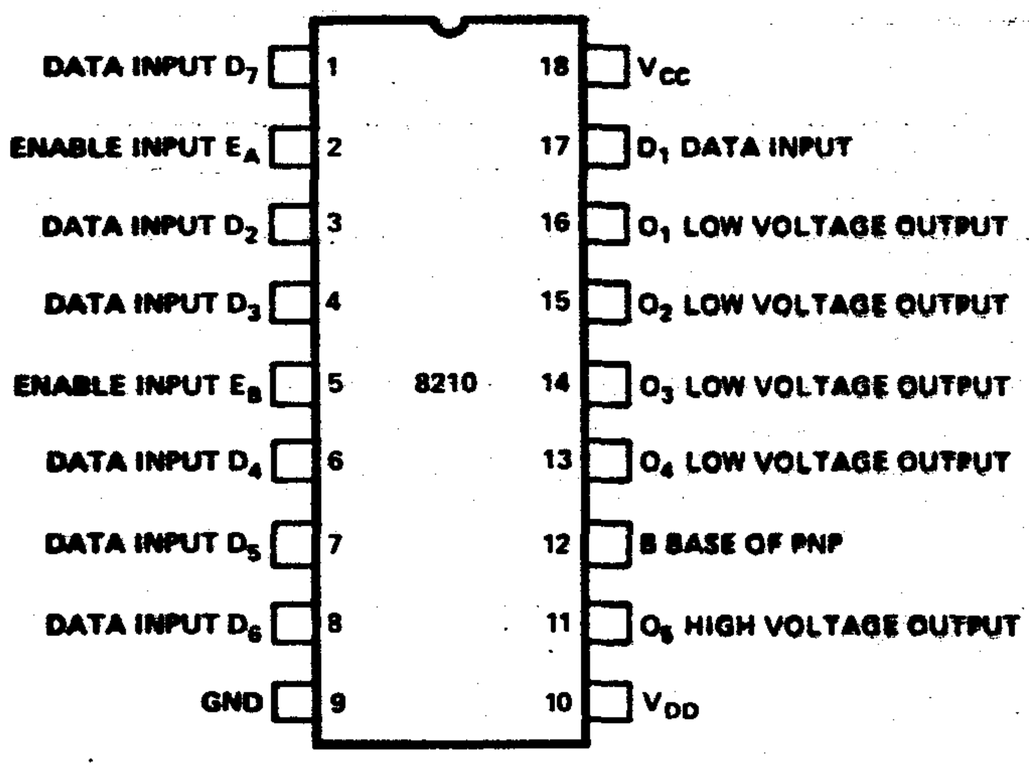
- Four Low Voltage Drivers
- One High Voltage Driver
- TTL and DTL Compatible Inputs
- Outputs Compatible with 8107A MOS Memories
- Operates from Standard Bipolar and MOS Power Supplies
- Maximum MOS Device Protection — Output Clamp Diodes

The Intel 8210 is a Bipolar-to-MOS level shifter and high voltage driver which accepts TTL and DTL inputs. It contains four (4) low voltage drivers and one high voltage driver, each with current driving capabilities suitable for driving N-channel MOS memory devices. The 8210 is particularly suitable for driving the 8107A N-channel MOS memory chips. The 8210 operates from the 5 volt and 12 volt power supplies used to bias the memory devices.

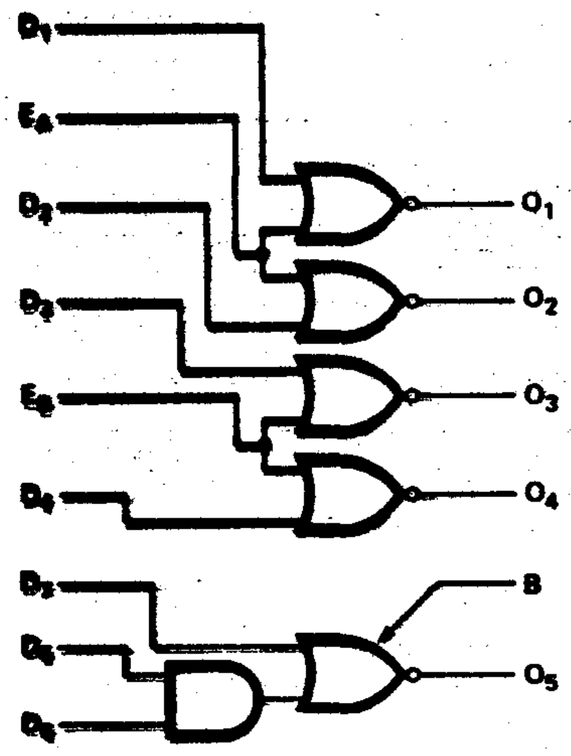
The four low voltage drivers feature two common enable inputs per pair of drivers which permits address or data decoding. The high voltage driver swings the 12 volts required to drive the chip enable (clock) input for the 8107A.

The 8210 high voltage driver requires an externally connected PNP transistor. The PNP base is connected to pin 12, the collector to pin 11, and the emitter to pin 10 or V_{DD}. The use of a fast switching, high voltage, high current gain PNP, like the 2N5707 is recommended.

PIN CONFIGURATION



LOGIC SYMBOL



Schottky Bipolar 8212

PRELIMINARY

8 BIT INPUT/OUTPUT PORT

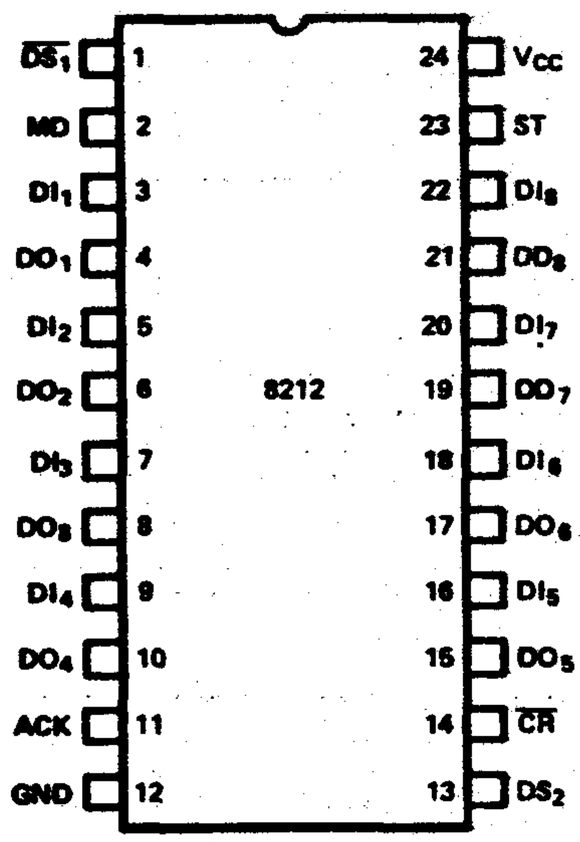
- **Fully Parallel 8-Bit Data Register or Buffer**
- **Low Input Load Current — .25 mA Max.**
- **Three State Outputs**
- **Outputs Sink 15 mA**
- **3.5V Output High Voltage for Direct Interface to 8080 CPU**
- **Replaces Buffers, Latches and Multiplexers in Microcomputer Systems**
- **Reduces System Package Count**
- **Asynchronous Register Clear**

The 8212 is a multi-mode latch/buffer device designed for use in microcomputer systems.

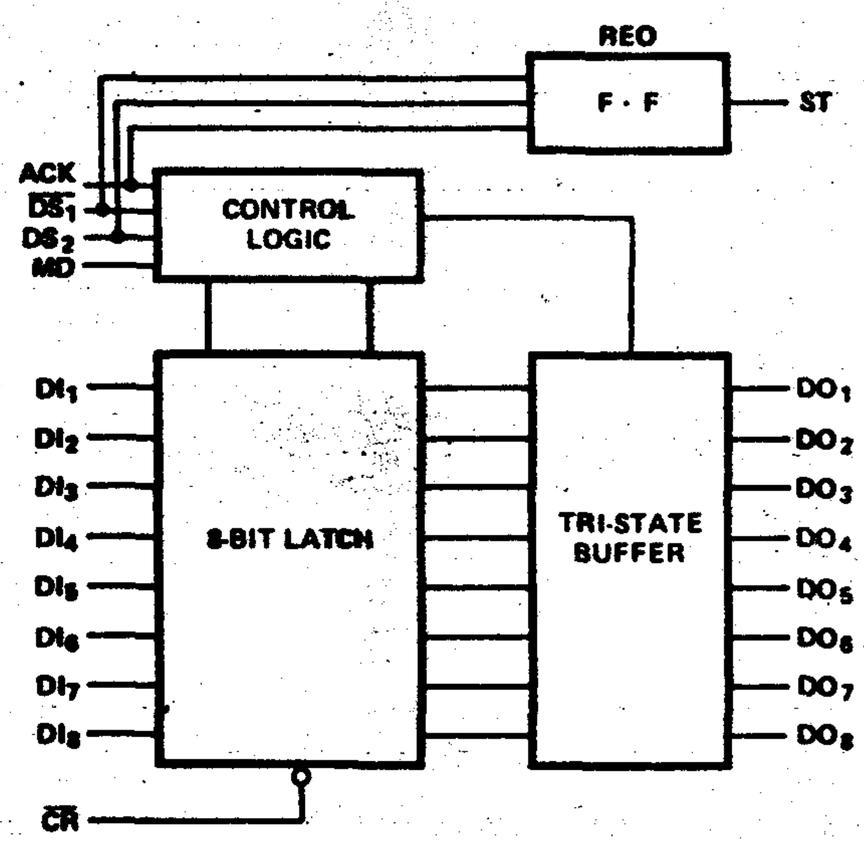
The device consists of an 8-bit latch with tri-state output buffers, along with control logic, and a service request flip-flop.

All of the principal peripheral and input/output functions of a microcomputer system can be implemented **with this** device.

PIN CONFIGURATION



BLOCK DIAGRAM



PIN NAMES

DI_1- DI_8	DATA IN
DO_1- DO_8	DATA OUT
$\overline{DS_1}$- DS_2	DEVICE SELECT
MD	MODE
ACK	ACKNOWLEDGE (CLK)
ST	STATUS F-F
$\overline{CR}$	CLEAR (ACTIVE LOW)

FUNCTIONAL LOGIC DESCRIPTION

WR (Write Enable)	$\overline{DS_1} \cdot DS_2 \cdot MD + ACK \cdot \overline{MD}$ (Data is latched after WR is removed)
EN (Output Enable)	$\overline{DS_1} \cdot DS_2 \cdot \overline{MD} + MD$
S (SR Flip-Flop Set)	$\overline{DS_1} \cdot DS_2$ (Asynchronous, Set overrides Reset)
R (SR Flip-Flop Reset)	ACK (Synchronous, negative edge triggered)
ST (Status Out)	$O_{SR} \cdot \overline{S}$ (Output inhibited during Set, when set by DS_1 and DS_2 only)
$\overline{CR}$ (Clear)	Resets all latches to "0" and sets the S.R. Flip-Flop to "1"

Schottky Bipolar 8216

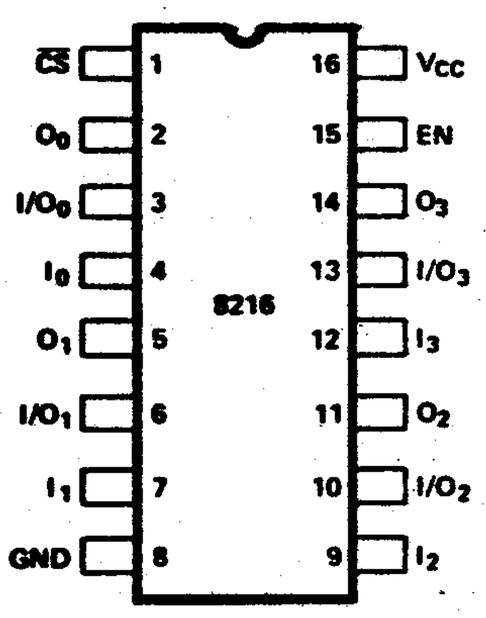
PRELIMINARY

4 BIT PARALLEL BIDIRECTIONAL BUS DRIVER

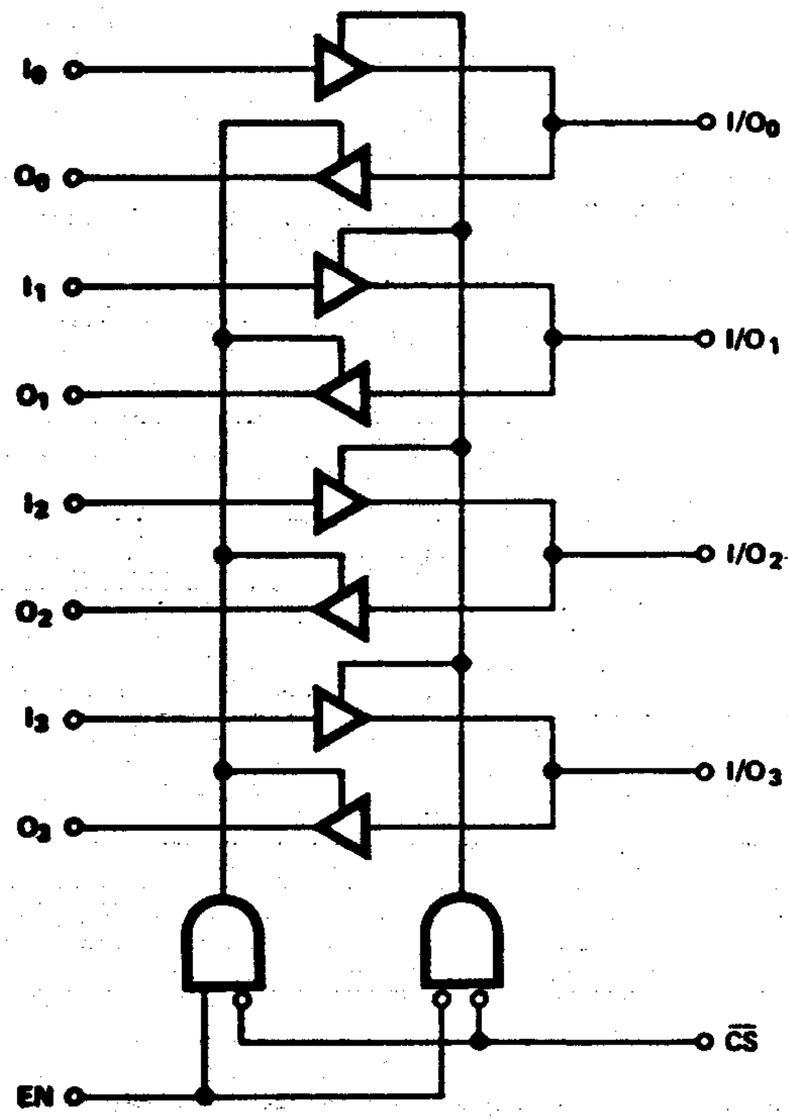
- Data Bus Buffer Driver for 8080 CPU
- Low Input Load Current — .25 mA Maximum
- High Output Drive Capability for Driving System Data Bus
- 3.5V Output High Voltage for Direct Interface to 8080 CPU
- Three State Outputs
- Reduces System Package Count

The 8216 is a 4-bit bi-directional bus driver/receiver. All inputs are low power TTL compatible. For driving MOS, the O outputs provide V_{OH} (3.5V), and for high capacitance terminated bus structures, the I/O outputs provide a higher I_{OL} (25 mA) capability.

PIN CONFIGURATION



LOGIC DIAGRAM



intel

Silicon Gate MOS 8201

PRELIMINARY

UNIVERSAL COMMUNICATION INTERFACE

- **Synchronous and Asynchronous Operation**
  - **Synchronous:**
    - 5-8 Bit Characters
    - Internal or External Character Synchronization
    - Automatic Sync Insertion/Deletion
  - **Asynchronous:**
    - 5-8 Bit Characters
    - Clock Rate — 16, 32 or 64 Times Baud Rate
    - Break Character Generation
    - 1, 1½, or 2 Stop Bits
    - False Start Bit Detection
- **Baud Rate — DC to 50 k Baud**
- **Full Duplex, Double Buffered, Transmitter and Receiver**
- **Error Detection — Parity, Overrun, Overflow, and Framing**
- **Fully Compatible with 8080 CPU**
- **28-Pin DIP Package**
- **All Inputs and Outputs Are TTL Compatible**
- **Single 5 Volt Supply**
- **Single TTL Clock**

The 8201 is a universal communication interface device used for data communication in 8080 microprocessor systems. This device is used as an 8080 peripheral device and it can be programmed by the 8080 to operate using virtually any serial data transmission technique presently in use. It accepts data characters from the 8080 in parallel format and then converts them into a continuous serial data stream for transmission. Simultaneously, it can receive serial data streams and convert them into parallel data characters for the 8080. This communication interface will interrupt the 8080 wherever it can accept a new character for transmission, or whenever it has received a character for the 8080. In addition, it will also interrupt the 8080 if any data transmission error occurs (i.e., parity error, framing error, overrun error or underrun error).

Pin Name	Pin Function
D	Data Bus (8 bits)
C/D	Control or Data is to be Written or Read
READ	Read Data Command
WRITE	Write Data or Control Command
CE	Chip Enable
ϕ_1	Clock Pulse (TTL)
RST	Reset
TxC	Transmitter Clock
TxD	Transmitter Data
RxC	Receiver Clock
RxD	Receiver Data
RxRDY	Receiver Ready (has character for 8080)
TxRDY	Transmitter Ready (ready for char. from 8080)
PE	Parity Error
OE	Overrun Error
FE/SYNC	Framing Error (Asyn Mode) Sync Detect (Sync Mode)
RTS	Request to Send Data
CTS	Clear to Send Data
TxE	Transmitter Empty
V_{CC}	+5 Volt Supply
V_{SS}	Ground

APPENDIX VII
MCS-80 SOFTWARE LIBRARY

PL/M Compiler — A High Level Systems Language

It's easy to program the MCS-80 Microcomputer using PL/M, a new high level language concept developed to meet the special needs of microcomputer systems programming. Programmers can now utilize a true high level language to efficiently program microcomputers. PL/M is an assembly language replacement that can fully command the 8080 CPU and future processors to produce efficient run-time object code. PL/M was designed to provide additional developmental software support for the MCS-80 microcomputer system, permitting the programmer to concentrate more on his problem and less on the actual task of programming than is possible with assembly language.

Programming time and costs are drastically reduced, and training, documentation and program maintenance are simplified. User application programs and standard systems programs may be transferred to future computer systems that support PL/M with little or no reprogramming. These are advantages of high-level language programming that have been proven in the large computer field and are now available to the microcomputer user.

PL/M is derived from IBM's PL/I, a very extensive and sophisticated language which promises to become the most widely known and used language in the near future. PL/M is a subset of PL/I with emphasis on those features that accurately reflect the nature of systems programming requirements.

PL/M IS AN EFFICIENT LANGUAGE

Tests on sample programs indicate that a PL/M program can be written in less than 10% of the time it takes to write the same program in assembly language with little efficiency loss. The main reason for this savings in time is the fact that PL/M allows the programmer to define his problem in terms natural to him, not in the computer's terms. Consider the following sample program which selects the largest of two numbers. In PL/M, the programmer might write: If A > B, then C = A; else C = B; Meaning:

> "If variable A is greater than Variable B, then assign A to Variable C; otherwise, assign B to C."

A corresponding program in assembly language is twelve separate machine instructions, and conveys little of original intent of the program.

Because of the ease and conciseness with which programs can be written and the error free translation into machine language achieved by the compiler, the time to program a given system is reduced substantially over assembly language.

Debug and checkout time of a PL/M program is also much less than that of an assembly language program, partly because of the inherent clarity of PL/M, but also because writing a program in PL/M encourages good programming techniques. Furthermore, the structure of the PL/M language enables the PL/M compiler to detect error conditions that would slip by an assembler. The PL/M compiler is written in Standard FORTRAN IV and will execute on most large machines with little alteration.

MCS-80 Cross Assembler Software Package

The MCS-80 cross assembler translates a symbolic representation of the instructions and data into a form which can be loaded and executed by the MCS-80. By cross assembler, we mean an assembler executing on a machine other than the MCS-80 which generates code for the MCS-80. Initial development time can be significantly reduced by taking advantage of a large scale computer's processing, editing and high speed peripheral capability. Programs are written in the assembly language using mnemonic symbols both for 8080 instruction and for special assembler operations. Symbolic addresses can be used in the source program; however, the assembled program will use absolute address.

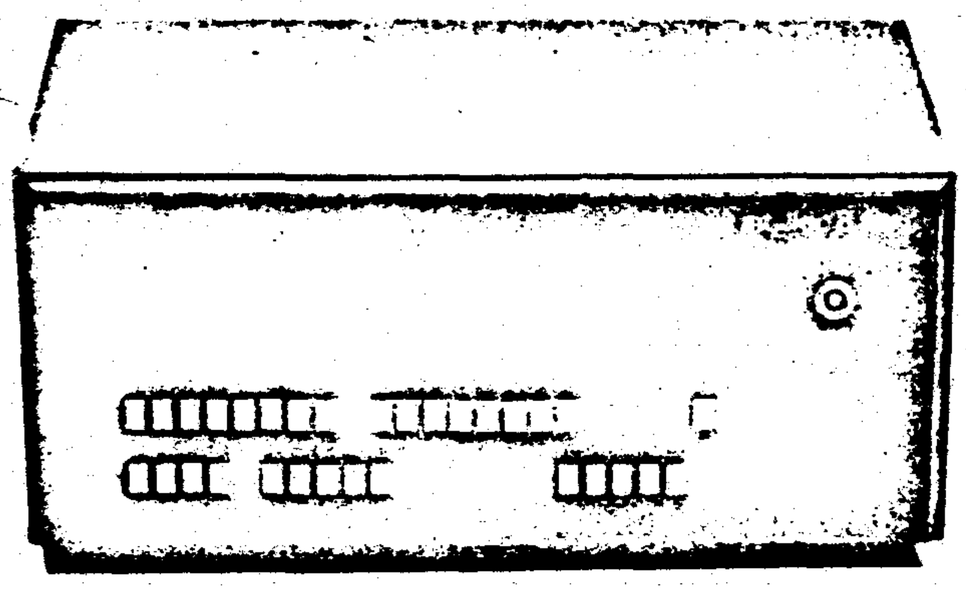
The Assembler, designed to operate interactively from a terminal, is written in standard FORTRAN IV and can be modified to run on most large scale machines.

MCS-80 Simulator Software Package

The MCS-80 Simulator is a computer program written in FORTRAN IV language and called INTERP/80 This program provides a software simulation of the Intel 8080 CPU, along with execution monitoring commands to aid program development for the MCS-80.

INTERP/80 accepts machine code produced by the 8080 Assembler, along with execution commands from a time sharing terminal, card reader, or disk file. The execution commands allow manipulation of the simulated MCS-80 memory and the 8080 CPU registers. In addition, operand and instruction breakpoints may be set to stop execution at crucial points in the program. Tracing features are also available which allow the CPU operation to be monitored. INTERP/80 also accepts symbol tables from either the PL/M compiler or MCS-80 cross assembler to allow debugging, tracing and braking, and displaying of program using symbolic names.

The PL/M compiler, MCS-80 assembler and MCS-80 simulator software packages may be procured from Intel on magnetic tape, or alternatively, from nationwide computer time sharing services. Contact Intel for details.



APPENDIX VIII
MCS-80 DEVELOPMENT SYSTEMS

Intellec 8 With 8080 CPU

FEATURES

- Ideal for developing MCS-80 systems.
- The Intellec 8 microcomputer system has 10K bytes of memory (expandable to 16K, I/O, TTY Interface, standard software, control panel, power supplies, and a compact finished cabinet (less than 0.8 ft.^{3}).
- The heart of the Intellec 8 is Intel's eight-bit "computer-on-a-chip," the 8080. This is an 8-bit parallel CPU with a repertoire of 78 instructions, seven working registers, stack architecture, interrupt capability, and it directly addresses 64K bytes of memory.
- Direct access to memory via control console.
- Standard software provided with the Intellec 8 includes a system monitor (loader, hex memory dump, instruction editor), a resident assembler, and a text editor.
- With this system, all program development may be done in RAM memory.
- A complete PROM programmer is provided. After the program is firm, it may be committed to non-volatile storage in Intel's 1702A programmable and erasable Read-Only-Memory.
- Complete system control and hardware debugging aids are provided via the control panel.
- Crystal clocks are used for system stability.
- System is expandable to 16 microcomputer modules in a single chassis.

SPECIFICATIONS

Word Size:	Data: 8 bits Instruction: 8, 16, or 24 bits
Memory Size:	8K RAM, 2K ROM bytes expandable to 16K bytes
Instruction Set:	78, including: conditional branching, decimal binary arithmetic, logical, register-to-register and memory reference operations
Machine Cycle Time:	2μs - 3μs
System Clock:	Crystal controlled
I/O Channels:	4 expandable to 8 input ports 4 expandable to 24 output ports (TTL Compatible)
Interrupt:	Single level
Direct Memory Access:	Standard via the control panel
Operating Temperature:	0°C to 55°C
Power Supplies:	+5V ±5% —9V ±5% 12 amps* 1.8 amps* *Larger power supplies may be required for expanded systems
Physical Size:	Intellec 8: 7"x17⅛"x12¼" (table top only)
Weight:	30 lb.
Standard Software:	System Monitor Resident Assembler Text Editor
Support Software:	PL/M Compiler Cross Assembler Simulator (written in FORTRAN IV)

STANDARD SYSTEMS AND OPTIONAL MODULES

INTELLEC 8 (imm8-84). Standard System includes the following modules:

- Central Processor Module with 8080 CPU
- Input/Output Module
- PROM Memory Module
- Two RAM Memory Modules
- PROM Programmer Module
- Chassis with Mother Board
- Power Supplies
- Control and Display Panel
- Finished Cabinet
- Standard Software
  - System Monitor
  - Resident Assembler
  - Text Editor

OPTIONAL MODULES available for the Intellec 8:

- Additional I/O or Output Modules
- Additional RAM Memory Modules
- Universal Prototype Module
- Module Extender
- Drawer Slides and Extenders for Rack Mounting

Software

STANDARD. All peripheral interface to Intellec 8 standard software is via TTY, model ASR33. The standard software includes a System Monitor, Resident Assembler and Text Editor.

A. System Monitor

1. Contained in eight 1702A PROMs located on the PROM memory module.
2. Program assigned to upper 2K bytes of memory.
3. Remaining 14K of memory may then be used for either program or data storage.
4. Intellec 8 modular computer systems have a control program called a Resident Monitor in PROM so that no "bootstrap" operation need ever be performed. The monitor functions are as follows:
   a. Load RAM memory from paper tape, either in BNPF format or hexadecimal format.
   b. Display the contents of RAM memory on a printer.
   c. Modify individual bytes of RAM memory, move blocks of RAM memory, fill blocks of RAM memory with constant data.
   d. Write contents of RAM memory to paper tape in either BNPF or hexadecimal format.

B. Resident Assembler

1. Translates the mnemonic code to binary machine code.
2. Loaded into system RAM memory via paper tape.
3. 8K of memory storage is required for both the resident assembler and the symbol table.
4. This three pass assembler generates program tape which is reloaded via the monitor.

C. Text Editor

1. Loaded to system via paper tape.
2. Edits the source program during program development.

DEVELOPMENT SUPPORT: PL/M Compiler, Assembler and Simulator. In addition to the standard software available with the Intellec 8, Intel offers a PL/M compiler, cross assembler, and simulator written in FORTRAN IV and designed to run on a large scale computer. These routines may be procured directly from Intel, or alternatively, designers may contact nation-wide computer time-sharing services.

PL/M COMPILER. PL/M is a high level procedure-oriented systems language for programming the Intel MCS-80 microcomputer. The language contains the features of a high-level language, without sacrificing the efficiencies of assembly language.

A significant advantage of this language is that PL/M programs can be compiled for either the Intel 8008 or 8080.

ASSEMBLER. The MCS-80 Assembler generates object codes from symbolic assembly language instructions.

It is designed to operate on a large scale computer with input by paper tape, directly from a terminal keyboard, or system file.

SIMULATOR. The MCS-80 Simulator, called INTERP/8, provides a software simulation of the Intel 8080 CPU, along with execution monitoring commands to aid program development for the MCS-80.

Microcomputer Module Description

Modules may be ordered individually. All modules are 8" wide, 6.18" high and use standard 100-pin connectors.

imm8-83 Central Processor Module

- Intel's 8080 eight-bit parallel single chip CPU—n-channel silicon gate MOS.
- Accumulator and six 8-bit working registers.
- Unlimited subroutine nesting.
- Interface to 64K 8-bit bytes of PROM, ROM, or RAM via the PROM Memory Module and RAM Memory Module.
- Interface for expansion to 256 8-bit input ports and 256 8-bit output ports, via the I/O and Output Modules.
- Interrupt capability.
- Two phase crystal clock.
- All module interfaces are TTL compatible.

imm6-26 PROM Memory Module

- Provides sockets for up to sixteen 1702A electrically programmable and erasable PROMs for a system's fixed program memory (maximum 4K bytes/module).
- For volume requirements, Intel mask programmed 1302 ROMs may be substituted in the same module.

imm6-28 RAM Memory Module

- A 4K x 8 n-channel MOS memory system using Intel's 1024 bit static RAM (2102).
- Address latching, data latching, and module select decoding are provided on the card.
- Provides both program storage and data storage.

imm8-61 Input/Output Module

- Four 8-bit input ports (32 lines).
- Four 8-bit data latching output ports (32 lines).
- One pair of ports for TTY communication.
- All input and output ports are TTL compatible.

imm8-63 Output Module

- Eight 8-bit data latching output ports (64 lines).
- All output ports are TTL compatible.

imm6-70 Universal Prototype Module

- Accommodates 14, 16, 24, or 40 pin wire wrap sockets (maximum of 52 16-pin sockets)
- Provides breadboard capability for developing custom and specialized interface circuits.

imm6-72 Module Extender

- Extends Intellec modules out of card chassis for ease in test and system debugging.

imm6-76 PROM Programmer Module

- Provides all timing and level shifting circuitry for programming Intel's electrically programmable and erasable 1702A PROMs.

Control and Display Panel

- Provides complete operator control for Intellec 8 and displays system status.
  - Address and Data Entry switches.
  - Status, instruction code, data and address displays.
- Complete program development tool.
  - ADDRESS, PROGRAM SEQUENCE, and MODE CONTROL switches permit easy alteration and examination of the program during the debugging phase of program development.
- Control and socket for 1702A PROM programming is also provided.

Chassis

- Capacity for up to sixteen microcomputer modules. (16 sockets with standard system).
- PC Mother Board eliminates back plane wiring—all cards plug into common bus.
- Standard 100 pin connectors (125 Mil centers) are used for all boards in system.
- Space is provided for additional memory and I/O modules and unique customer developed system interface modules.
- A fan is provided.

imm8-88 Conversion Kit

- Allows imm8-80 (Intellec 8 with 8008 CPU) to be used as 8080 development system.