# 8080A/8080A-1/8080A-2
# 8-BIT N-CHANNEL MICROPROCESSOR

- **TTL Drive Capability**
- **2 μs (−1:1.3 μs, −2:1.5 μs) Instruction Cycle**
- **Powerful Problem Solving Instruction Set**
- **6 General Purpose Registers and an Accumulator**
- **16-Bit Program Counter for Directly Addressing up to 64K Bytes of Memory**
- **16-Bit Stack Pointer and Stack Manipulation Instructions for Rapid Switching of the Program Environment**
- **Decimal, Binary, and Double Precision Arithmetic**
- **Ability to Provide Priority Vectored Interrupts**
- **512 Directly Addressed I/O Ports**
- **Available in EXPRESS**
  **— Standard Temperature Range**
- **Available in 40-Lead Cerdip and Plastic Packages**
  (See Packaging Spec. Order #231369)



The Intel 8080A is a complete 8-bit parallel central processing unit (CPU). It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process. This offers the user a high performance solution to control and processing applications.

The 8080A contains 6 8-bit general purpose working registers and an accumulator. The 6 general purpose registers may be addressed individually or in pairs providing both single and double precision operators. Arithmetic and logical instructions set or reset 4 testable flags. A fifth flag provides decimal arithmetic operation.

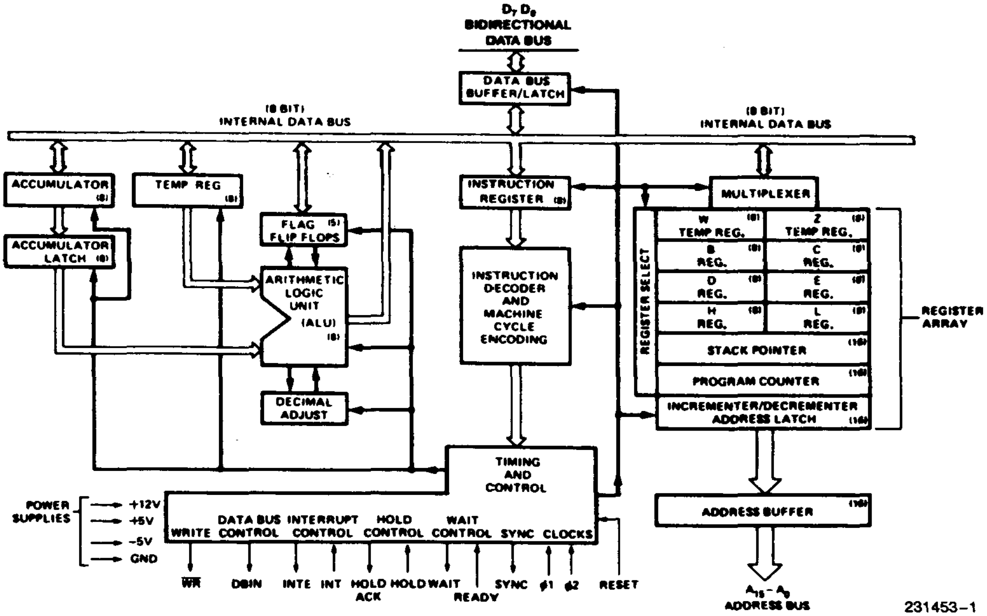
The 8080A has an external stack feature wherein any portion of memory may be used as a last in/first out stack to store/retrieve the contents of the accumulator, flags, program counter, and all of the 6 general purpose registers. The 16-bit stack pointer controls the addressing of this external stack. This stack gives the 8080A the ability to easily handle multiple level priority interrupts by rapidly storing and restoring processor status. It also provides almost unlimited subroutine nesting.

This microprocessor has been designed to simplify systems design. Separate 16-line address and 8-line bidirectional data busses are used to facilitate easy interface to memory and I/O. Signals to control the interface to memory and I/O are provided directly by the 8080A. Ultimate control of the address and data busses resides with the HOLD signal. It provides the ability to suspend processor operation and force the address and data busses into a high impedance state. This permits OR-tying these busses with other controlling devices for (DMA) direct memory access or multi-processor operation.

**NOTE:**
The 8080A is functionally and electrically compatible with the Intel 8080.

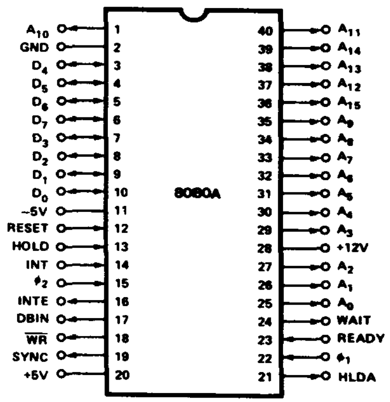


November 1986
Order Number: 231453-001




231453–1

**Figure 1. Block Diagram**




231453–2

**Figure 2. Pin Configuration**

Table 1. Pin Description

| Symbol | Type | Name and Function |
|---|---|---|
| $A_{15}-A_0$ | O | **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. |
| $D_7-D_0$ | I/O | **DATA BUS:** The data bus provides bi-directional communication between the CPU, memory, and I/O devices for instructions and data transfers. Also, during the first clock cycle of each machine cycle, the 8080A outputs a status word on the data bus that describes the current machine cycle. $D_0$ is the least significant bit. |
| SYNC | O | **SYNCHRONIZING SIGNAL:** The SYNC pin provides a signal to indicate the beginning of each machine cycle. |
| DBIN | O | **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 8080A data bus from memory or I/O. |
| READY | I | **READY:** The READY signal indicates to the 8080A that valid memory or input data is available on the 8080A data bus. This signal is used to synchronize the CPU with slower memory or I/O devices. If after sending an address out the 8080A does not receive a READY input, the 8080A will enter a WAIT state for as long as the READY line is low. READY can also be used to single step the CPU. |
| WAIT | O | **WAIT:** The WAIT signal acknowledges that the CPU is in a WAIT state. |
| $\overline{WR}$ | O | **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 low ($\overline{WR}$ = 0). |
| HOLD | I | **HOLD:** The HOLD signal requests the CPU to enter the HOLD state. The HOLD state allows an external device to gain control of the 8080A address and data bus as soon as the 8080A has completed its use of these busses for the current machine cycle. It is recognized under the following conditions:<br>• the CPU is in the HALT state.<br>• 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. |
| HLDA | O | **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:<br>• T3 for READ memory or input.<br>• The Clock Period following T3 for WRITE memory or OUTPUT operation.<br>In either case, the HLDA signal appears after the rising edge of $\phi_2$. |
| INTE | O | **INTERRUPT ENABLE:** Indicates the content of the internal interrupt enable flip/flop. This flip/flop may be set or reset by the Enable and Disable Interrupt instructions and inhibits interrupts from being accepted by the CPU when 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 | I | **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. |
| RESET[1] | I | **RESET:** While the RESET signal is activated, the content of the program counter is cleared. 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, stack pointer, and registers are not cleared. |
| $V_{SS}$ | | **GROUND:** Reference. |
| $V_{DD}$ | | **POWER:** $+12 \pm 5\%$ V. |
| $V_{CC}$ | | **POWER:** $+5 \pm 5\%$ V. |
| $V_{BB}$ | | **POWER:** $-5 \pm 5\%$ V. |
| $\phi_1, \phi_2$ | | **CLOCK PHASES:** 2 externally supplied clock phases. (non TTL compatible) |

**NOTE:**
1. The RESET signal must be active for a minimum of 3 clock cycles.

## 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 $V_{BB}$ ........... −0.3V to +20V
$V_{CC}$, $V_{DD}$ and $V_{SS}$
with Respect to $V_{BB}$ ........... −0.3V to +20V
Power Dissipation .......................... 1.5W

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

******WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

## D.C. CHARACTERISTICS

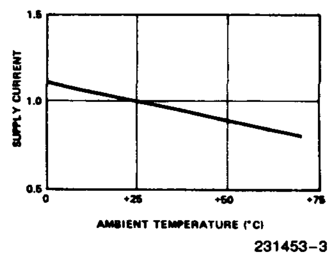
$T_A$ = 0°C to 70°C, $V_{DD}$ = +12V ±5%, $V_{CC}$ = +5V ±5%, $V_{BB}$ = −5V ±5%, $V_{SS}$ = 0V; unless otherwise noted

| Symbol | Parameter | Min | Typ | Max | Unit | Test Condition |
|---|---|---|---|---|---|---|
| $V_{ILC}$ | Clock Input Low Voltage | $V_{SS}$ − 1 | | $V_{SS}$ + 0.8 | V | |
| $V_{IHC}$ | Clock Input High Voltage | 9.0 | | $V_{DD}$ + 1 | V | |
| $V_{IL}$ | Input Low Voltage | $V_{SS}$ − 1 | | $V_{SS}$ + 0.8 | V | |
| $V_{IH}$ | Input High Voltage | 3.3 | | $V_{CC}$ + 1 | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.9 mA on All Outputs, |
| $V_{OH}$ | Output High Voltage | 3.7 | | | V | $I_{OH}$ = −150 μA. |
| $I_{DD\ (AV)}$ | Avg. Power Supply Current ($V_{DD}$) | | 40 | 70 | mA | Operation |
| $I_{CC\ (AV)}$ | Avg. Power Supply Current ($V_{CC}$) | | 60 | 80 | mA | $T_{CY}$ = 0.48 μs |
| $I_{BB\ (AV)}$ | Avg. Power Supply Current ($V_{BB}$) | | 0.01 | 1 | mA | |
| $I_{IL}$ | Input Leakage | | | ±10 | μA | $V_{SS} \le V_{IN} \le V_{CC}$ |
| $I_{CL}$ | Clock Leakage | | | ±10 | μA | $V_{SS} \le V_{CLOCK} \le V_{DD}$ |
| $I_{DL}$ | Data Bus Leakage in Input Mode | | | −100<br>−2.0 | μA<br>mA | $V_{SS} \le V_{IN} \le V_{SS} + 0.8V$<br>$V_{SS} + 0.8V \le V_{IN} \le V_{CC}$ |
| $I_{FL}$ | Address and Data Bus Leakage During HOLD | | | +10<br>−100 | μA | $V_{ADDR/DATA} = V_{CC}$<br>$V_{ADDR/DATA} = V_{SS} + 0.45V$ |

## CAPACITANCE

$T_A$ = 25°C, $V_{CC}$ = $V_{DD}$ = $V_{SS}$ = 0V, $V_{BB}$ = −5V

| Symbol | Parameter | Typ | Max | Unit | Test Condition |
|---|---|---|---|---|---|
| $C_\phi$ | Clock Capacitance | 17 | 25 | pF | $f_c$ = 1 MHz |
| $C_{IN}$ | Input Capacitance | 6 | 10 | pF | Unmeasured Pins |
| $C_{OUT}$ | Output Capacitance | 10 | 20 | pF | Returned to $V_{SS}$ |

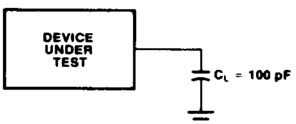



**Typical Supply Current vs Temperature, Normalized**
**$\Delta I\ Supply/\Delta T_A$ = −0.45%/°C**

## A.C. CHARACTERISTICS (8080A) $T_A$ = 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 | Max | −1 Min | −1 Max | −2 Min | −2 Max | Unit | Test Condition |
|---|---|---|---|---|---|---|---|---|---|
| $t_{CY}$(3) | Clock Period | 0.48 | 2.0 | 0.32 | 2.0 | 0.38 | 2.0 | μs | |
| $t_r$, $t_f$ | Clock Rise and Fall Time | 0 | 50 | 0 | 25 | 0 | 50 | ns | |
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | 60 | | 50 | | 60 | | ns | |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | 220 | | 145 | | 175 | | ns | |
| $t_{D1}$ | Delay $\phi_1$ to $\phi_2$ | 0 | | 0 | | 0 | | ns | |
| $t_{D2}$ | Delay $\phi_1$ to $\phi_2$ | 70 | | 60 | | 70 | | ns | |
| $t_{D3}$ | Delay $\phi_1$ to $\phi_2$ Leading Edges | 80 | | 60 | | 70 | ns | | |
| $t_{DA}$ | Address Output Delay From $\phi_2$ | | 200 | | 150 | | 175 | ns | $C_L$ = 100 pF |
| $t_{DD}$ | Data Output Delay From $\phi_2$ | | 200 | | 180 | | 200 | ns | |
| $t_{DC}$ | Signal Output Delay From $\phi_1$ or $\phi_2$ (SYNC, WR, WAIT, HLDA) | | 120 | | 110 | | 120 | ns | $C_L$ = 50 pF |
| $t_{DF}$ | DBIN Delay From $\phi_2$ | 25 | 140 | 25 | 130 | 25 | 140 | ns | |
| $t_{DI}$(1) | Delay for Input Bus to Enter Input Mode | | $t_{DF}$ | | $t_{DF}$ | | $t_{DF}$ | ns | |
| $t_{DS1}$ | Data Setup Time During $\phi_1$ and DBIN | 30 | | 10 | | 20 | | ns | |
| $t_{DS2}$ | Data Setup Time to $\phi_2$ During DBIN | 150 | | 120 | | 130 | | ns | |
| $t_{DH}$(1) | Data Hold Time From $\phi_2$ and DBIN | (1) | | (1) | | (1) | | ns | |
| $t_{IE}$ | INTE Output Delay From $\phi_2$ | | 200 | | 200 | | 200 | ns | $C_L$ = 50 pF |
| $t_{RS}$ | READY Setup Time During $\phi_2$ | 120 | | 90 | | 90 | | ns | |
| $t_{HS}$ | HOLD Setup Time During $\phi_2$ | 140 | | 120 | | 120 | | ns | |
| $t_{IS}$ | INT Setup Time During $\phi_2$ | 120 | | 100 | | 100 | | ns | |
| $t_H$ | Hold Time From $\phi_2$ (READY, INT, HOLD) | 0 | | 0 | | 0 | | ns | |
| $t_{FD}$ | Delay to Float During Hold (Address and Data Bus) | | 120 | | 120 | | 120 | ns | |
| $t_{AW}$ | Address Stable Prior to WR | (5) | | (5) | | (5) | | ns | |
| $t_{DW}$ | Output Data Stable Prior to WR | (6) | | (6) | | (6) | | ns | |
| $t_{WD}$ | Output Data Stable From WR | (7) | | (7) | | (7) | | ns | |
| $t_{WA}$ | Address Stable From WR | (7) | | (7) | | (7) | | ns | |
| $t_{HF}$ | HLDA to Float Delay | (8) | | (8) | | (8) | | ns | |
| $t_{WF}$ | WR to Float Delay | (9) | | (9) | | (9) | | ns | |
| $t_{AH}$ | Address Hold Time After DBIN During HLDA | −20 | | −20 | | −20 | | ns | |

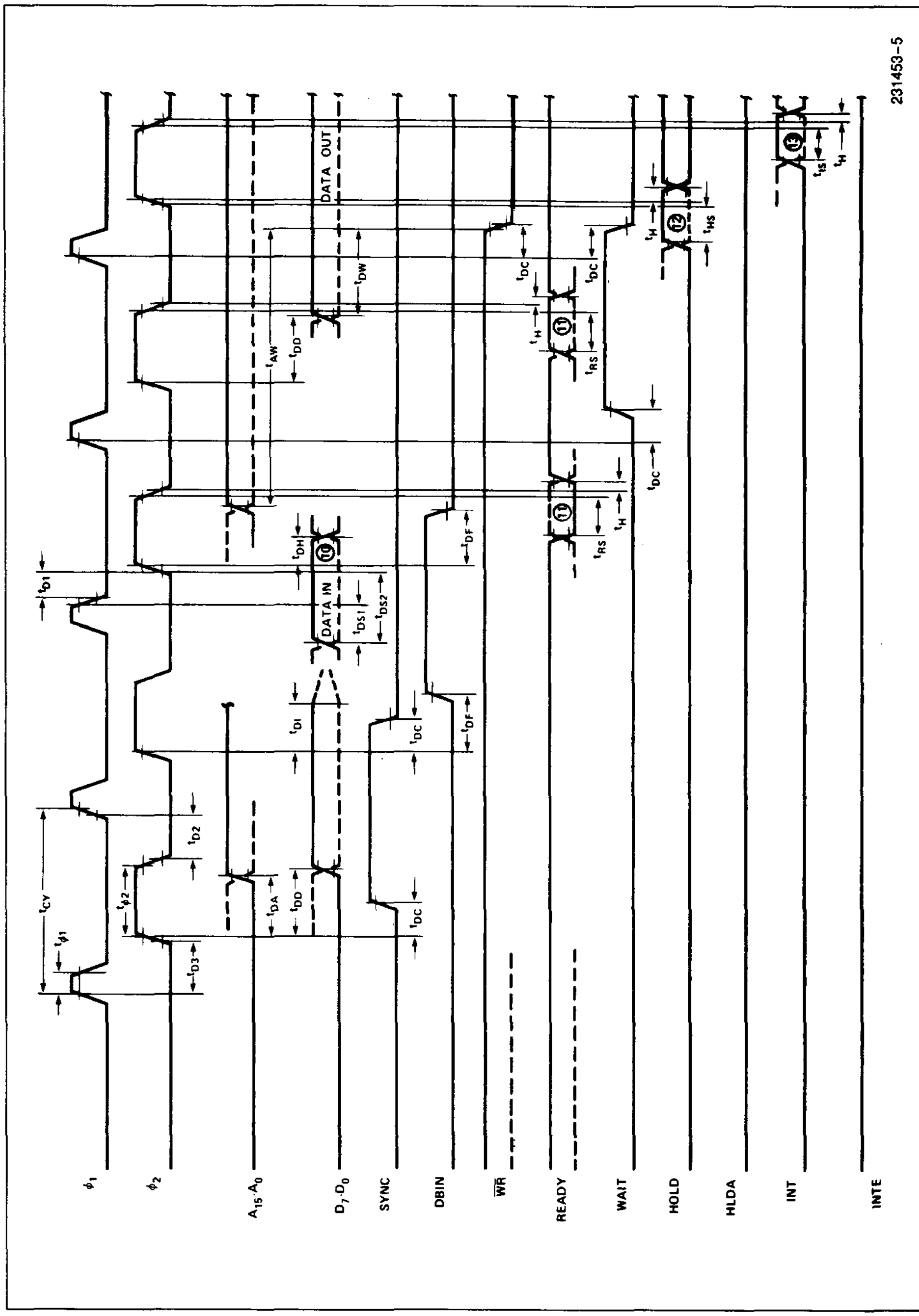
## A.C. TESTING LOAD CIRCUIT




231453-4

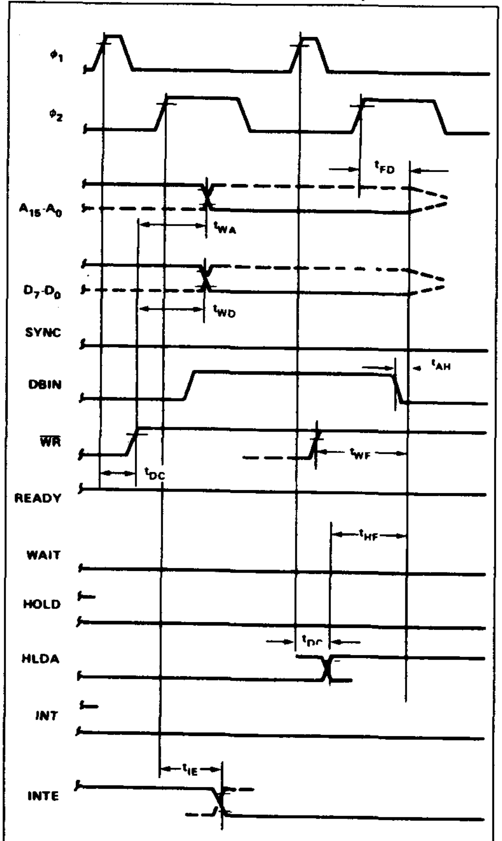
$C_L$ = 100 pF
$C_L$ Includes Jig Capacitance

## WAVEFORMS



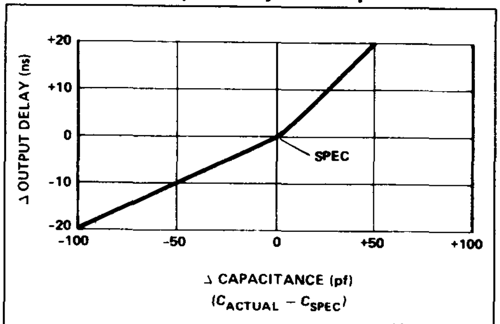

**NOTE:**
Timing measurements are made at the following reference voltages: CLOCK "1" = 8.0V, "0" = 1.0V; INPUTS "1" = 3.3V, "0" = 0.8V; OUTPUTS "1" = 2.0V, "0" = 0.8V

## WAVEFORMS (Continued)




231453-6

### Typical Δ Output Delay vs Δ Capacitance




231453-7

**NOTES:**

(Parenthesis gives −1, −2 specifications, respectively.)

1. Data input should be enabled with DBIN status. No bus conflict can then occur and data hold time is assured.

$t_{DH}$ = 50 ns or $t_{DF}$, whichever is less.

2. $t_{CY} = t_{D3} + t_{r\phi2} + t_{\phi2} + t_{f\phi2} + t_{D2} + t_{r\phi1} \geq$ 480 ns (−1:320 ns, − 2:380 ns).

3. The following are relevant when interfacing the 8080A to devices having $V_{IH}$ = 3.3V:
   a) Maximum output rise time from 0.8V to 3.3V = 100 ns @ $C_L$ = SPEC.
   b) Output delay when measured to 3.0V = SPEC +60 ns @ $C_L$ = SPEC.
   c) If $C_L$ = SPEC, add 0.6 ns/pF if $C_L > C_{SPEC}$, subtract 0.3 ns/pF (from modified delay) if $C_L < C_{SPEC}$.

4. $t_{AW} = 2\ t_{CY} - t_{D3} - t_{r\phi2} - 140$ ns (−1:110 ns, − 2:130 ns).

5. $t_{DW} = t_{CY} - t_{D3} - t_{r\phi2} - 170$ ns (−1:150 ns, − 2:170 ns).

6. If not HLDA, $t_{WD} = t_{WA} = t_{D3} + t_{r\phi2} + 10$ ns. If HLDA, $t_{WD} = t_{WA} = t_{WF}$.

7. $t_{HF} = t_{D3} + t_{r\phi2} - 50$ ns.

8. $t_{WF} = t_{D3} + t_{r\phi2} - 10$ ns.

9. Data in must be stable for this period during DBIN $T_3$. Both $t_{DS1}$ and $t_{DS2}$ must be satisfied.

10. Ready signal must be stable for this period during $T_2$ or $T_W$. (Must be externally synchronized.)

11. Hold signal must be stable for this period during $T_2$ or $T_W$ when entering hold mode, and during $T_3$, $T_4$, $T_5$ and $T_{WH}$ when in hold mode. (External synchronization is not required.)

12. Interrupt signal must be stable during this period of the last clock cycle of any instruction in order to be recognized on the following instruction. (External synchronization is not required.)

13. This timing diagram shows timing relationships only; it does not represent any specific machine cycle.

## INSTRUCTION SET

The accumulator group instructions include arithmetic and logical operators with direct, indirect, and immediate addressing modes.

Move, load, and store instruction groups provide the ability to move either 8 or 16 bits of data between memory, the six working registers and the accumulator using direct, indirect, and immediate addressing modes.

The ability to branch to different portions of the program is provided with jump, jump conditional, and computed jumps. Also the ability to call to and return from subroutines is provided both conditionally and unconditionally. The RESTART (or single byte call instruction) is useful for interrupt vector operation.

Double precision operators such as stack manipulation and double add instructions extend both the arithmetic and interrupt handling capability of the 8080A. The ability to increment and decrement memory, the six general registers and the accumulator is provided as well as extended increment and decrement instructions to operate on the register pairs and stack pointer. Further capability is provided by the ability to rotate the accumulator left or right through or around the carry bit.

Input and output may be accomplished using memory addresses as I/O ports or the directly addressed I/O provided for in the 8080A instruction set.

The following special instruction group completes the 8080A instruction set: the NOP instruction, HALT to stop processor execution and the DAA instructions provide decimal arithmetic capability. STC allows the carry flag to be directly set, and the CMC instruction allows it to be complemented. CMA complements the contents of the accumulator and XCHG exchanges the contents of two 16-bit register pairs directly.

### Data and Instruction Formats

Data in the 8080A is stored in the form of 8-bit binary integers. All data transfers to they 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.

| Format | | TYPICAL INSTRUCTIONS |
|---|---|---|
| **One Byte Instructions** | | |
| $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | Register to register, memory reference, arithmetic or logical, rotate, return, push, pop, enable or disable Interrupt instructions |
| **Two Byte Instructions** | | |
| $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | Immediate mode or I/O instructions |
| $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OPERAND | |
| **Three Byte Instructions** | | |
| $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | Jump, call or direct load and store instructions |
| $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | 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 | |

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

**Table 2. Instruction Set Summary**

| Mnemonic* | Instruction Code (1) $D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0$ | Operations Description | Clock Cycles (2) |
|---|---|---|---|
| **MOVE, LOAD, AND STORE** | | | |
| MOVr1,r2 | 0 1 D D D S S S | Move register to register | 5 |
| MOV M,r | 0 1 1 1 0 S S S | Move register to memory | 7 |
| MOV r,M | 0 1 D D D 1 1 0 | Move memory to register | 7 |
| MVI r | 0 0 D D D 1 1 0 | Move immediate register | 7 |
| MVI M | 0 0 1 1 0 1 1 0 | Move immediate memory | 10 |
| LXI B | 0 0 0 0 0 0 0 1 | Load immediate register Pair B & C | 10 |
| LXI D | 0 0 0 1 0 0 0 1 | Load immediate register Pair D & E | 10 |
| LXI H | 0 0 1 0 0 0 0 1 | Load immediate register Pair H & L | 10 |
| STAX B | 0 0 0 0 0 0 1 0 | Store A indirect | 7 |
| STAX D | 0 0 0 1 0 0 1 0 | Store A indirect | 7 |
| LDAX B | 0 0 0 0 1 0 1 0 | Load A indirect | 7 |
| LDAX D | 0 0 0 1 1 0 1 0 | Load A indirect | 7 |
| STA | 0 0 1 1 0 0 1 0 | Store A direct | 13 |
| LDA | 0 0 1 1 1 0 1 0 | Load A direct | 13 |
| SHLD | 0 0 1 0 0 0 1 0 | Store H & L direct | 16 |
| LHLD | 0 0 1 0 1 0 1 0 | Load H & L direct | 16 |
| XCHG | 1 1 1 0 1 0 1 1 | Exchange D & E, H & L Registers | 4 |
| **STACK OPS** | | | |
| PUSH B | 1 1 0 0 0 1 0 1 | Push register Pair B & C on stack | 11 |
| PUSH D | 1 1 0 1 0 1 0 1 | Push register Pair D & E on stack | 11 |
| PUSH H | 1 1 1 0 0 1 0 1 | Push register Pair H & L on stack | 11 |
| PUSH PSW | 1 1 1 1 0 1 0 1 | Push A and Flags on stack | 11 |
| POP B | 1 1 0 0 0 0 0 1 | Pop register Pair B & C off stack | 10 |
| POP D | 1 1 0 1 0 0 0 1 | Pop register Pair D & E off stack | 10 |
| POP H | 1 1 1 0 0 0 0 1 | Pop register Pair H & L off stack | 10 |
| POP PSW | 1 1 1 1 0 0 0 1 | Pop A and Flags off stack | 10 |
| XTHL | 1 1 1 0 0 0 1 1 | Exchange top of stack, H & L | 18 |
| SPHL | 1 1 1 1 1 0 0 1 | H & L to stack pointer | 5 |
| LXI SP | 0 0 1 1 0 0 0 1 | Load immediate stack pointer | 10 |
| INX SP | 0 0 1 1 0 0 1 1 | Increment stack pointer | 5 |
| DCX SP | 0 0 1 1 1 0 1 1 | Decrement stack pointer | 5 |
| **JUMP** | | | |
| JMP | 1 1 0 0 0 0 1 1 | Jump unconditional | 10 |
| JC | 1 1 0 1 1 0 1 0 | Jump on carry | 10 |
| JNC | 1 1 0 1 0 0 1 0 | Jump on no carry | 10 |
| JZ | 1 1 0 0 1 0 1 0 | Jump on zero | 10 |
| JNZ | 1 1 0 0 0 0 1 0 | Jump on no zero | 10 |
| JP | 1 1 1 1 0 0 1 0 | Jump on positive | 10 |

| Mnemonic* | Instruction Code (1) $D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0$ | Operations Description | Clock Cycles (2) |
|---|---|---|---|
| JM | 1 1 1 1 1 0 1 0 | Jump on minus | 10 |
| JPE | 1 1 1 0 1 0 1 0 | Jump on parity even | 10 |
| JPO | 1 1 1 0 0 0 1 0 | Jump on parity odd | 10 |
| PCHL | 1 1 1 0 1 0 0 1 | H & L to program counter | 5 |
| **CALL** | | | |
| CALL | 1 1 0 0 1 1 0 1 | Call unconditional | 17 |
| CC | 1 1 0 1 1 1 0 0 | Call on carry | 11/17 |
| CNC | 1 1 0 1 0 1 0 0 | Call on no carry | 11/17 |
| CZ | 1 1 0 0 1 1 0 0 | Call on zero | 11/17 |
| CNZ | 1 1 0 0 0 1 0 0 | Call on no zero | 11/17 |
| CP | 1 1 1 1 0 1 0 0 | Call on positive | 11/17 |
| CM | 1 1 1 1 1 1 0 0 | Call on minus | 11/17 |
| CPE | 1 1 1 0 1 1 0 0 | Call on parity even | 11/17 |
| CPO | 1 1 1 0 0 1 0 0 | Call on parity odd | 11/17 |
| **RETURN** | | | |
| RET | 1 1 0 0 1 0 0 1 | Return | 10 |
| RC | 1 1 0 1 1 0 0 0 | Return on carry | 5/11 |
| RNC | 1 1 0 1 0 0 0 0 | Return on no carry | 5/11 |
| RZ | 1 1 0 0 1 0 0 0 | Return on zero | 5/11 |
| RNZ | 1 1 0 0 0 0 0 0 | Return on no zero | 5/11 |
| RP | 1 1 1 1 0 0 0 0 | Return on positive | 5/11 |
| RM | 1 1 1 1 1 0 0 0 | Return on minus | 5/11 |
| RPE | 1 1 1 0 1 0 0 0 | Return on parity even | 5/11 |
| RPO | 1 1 1 0 0 0 0 0 | Return on parity odd | 5/11 |
| **RESTART** | | | |
| RST | 1 1 A A A 1 1 1 | Restart | 11 |
| **INCREMENT AND DECREMENT** | | | |
| INR r | 0 0 D D D 1 0 0 | Increment register | 5 |
| DCR r | 0 0 D D D 1 0 1 | Decrement register | 5 |
| INR M | 0 0 1 1 0 1 0 0 | Increment memory | 10 |
| DCR M | 0 0 1 1 0 1 0 1 | Decrement memory | 10 |
| INX B | 0 0 0 0 0 0 1 1 | Increment B & C registers | 5 |
| INX D | 0 0 0 1 0 0 1 1 | Increment D & E registers | 5 |
| INX H | 0 0 1 0 0 0 1 1 | Increment H & L registers | 5 |
| DCX B | 0 0 0 0 1 0 1 1 | Decrement B & C | 5 |
| DCX D | 0 0 0 1 1 0 1 1 | Decrement D & E | 5 |
| DCX H | 0 0 1 0 1 0 1 1 | Decrement H & L | 5 |
| **ADD** | | | |
| ADD r | 1 0 0 0 0 S S S | Add register to A | 4 |
| ADC r | 1 0 0 0 1 S S S | Add register to A with carry | 4 |
| ADD M | 1 0 0 0 0 1 1 0 | Add memory to A | 7 |
| ADC M | 1 0 0 0 1 1 1 0 | Add memory to A with carry | 7 |
| ADI | 1 1 0 0 0 1 1 0 | Add immediate to A | 7 |
| ACI | 1 1 0 0 1 1 1 0 | Add immediate to A with carry | 7 |
| DAD B | 0 0 0 0 1 0 0 1 | Add B & C to H & L | 10 |
| DAD D | 0 0 0 1 1 0 0 1 | Add D & E to H & L | 10 |
| DAD H | 0 0 1 0 1 0 0 1 | Add H & L to H & L | 10 |
| DAD SP | 0 0 1 1 1 0 0 1 | Add stack pointer to H & L | 10 |

**Table 2. Instruction Set Summary** (Continued)

| Mnemonic* | Instruction Code (1) $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | Operations Description | Clock Cycles (2) |
|---|---|---|---|
| **SUBTRACT** | | | |
| SUB r | 1 0 0 1 0 S S S | Subtract register from A | 4 |
| SBB r | 1 0 0 1 1 S S S | Subtract register from A with borrow | 4 |
| SUB M | 1 0 0 1 0 1 1 0 | Subtract memory from A | 7 |
| SBB M | 1 0 0 1 1 1 1 0 | Subtract memory from A with borrow | 7 |
| SUI | 1 1 0 1 0 1 1 0 | Subtract immediate from A | 7 |
| SBI | 1 1 0 1 1 1 1 0 | Subtract immediate from A with borrow | 7 |
| **LOGICAL** | | | |
| ANA r | 1 0 1 0 0 S S S | And register with A | 4 |
| XRA r | 1 0 1 0 1 S S S | Exclusive or register with A | 4 |
| ORA r | 1 0 1 1 0 S S S | Or register with A | 4 |
| CMP r | 1 0 1 1 1 S S S | Compare register with A | 4 |
| ANA M | 1 0 1 0 0 1 1 0 | And memory with A | 7 |
| XRA M | 1 0 1 0 1 1 1 0 | Exclusive Or memory with A | 7 |
| ORA M | 1 0 1 1 0 1 1 0 | Or memory with A | 7 |
| CMP M | 1 0 1 1 1 1 1 0 | Compare memory with A | 7 |
| ANI | 1 1 1 0 0 1 1 0 | And immediate with A | 7 |
| XRI | 1 1 1 0 1 1 1 0 | Exclusive Or immediate with A | 7 |
| ORI | 1 1 1 1 0 1 1 0 | Or immediate with A | 7 |
| CPI | 1 1 1 1 1 1 1 0 | Compare immediate with A | 7 |

| Mnemonic* | Instruction Code (1) $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | Operations Description | Clock Cycles (2) |
|---|---|---|---|
| **ROTATE** | | | |
| RLC | 0 0 0 0 0 1 1 1 | Rotate A left | 4 |
| RRC | 0 0 0 0 1 1 1 1 | Rotate A right | 4 |
| RAL | 0 0 0 1 0 1 1 1 | Rotate A left through carry | 4 |
| RAR | 0 0 0 1 1 1 1 1 | Rotate A right through carry | 4 |
| **SPECIALS** | | | |
| CMA | 0 0 1 0 1 1 1 1 | Complement A | 4 |
| STC | 0 0 1 1 0 1 1 1 | Set carry | 4 |
| CMC | 0 0 1 1 1 1 1 1 | Complement carry | 4 |
| DAA | 0 0 1 0 0 1 1 1 | Decimal adjust A | 4 |
| **INPUT/OUTPUT** | | | |
| IN | 1 1 0 1 1 0 1 1 | Input | 10 |
| OUT | 1 1 0 1 0 0 1 1 | Output | 10 |
| **CONTROL** | | | |
| EI | 1 1 1 1 1 0 1 1 | Enable Interrupts | 4 |
| DI | 1 1 1 1 0 0 1 1 | Disable Interrupt | 4 |
| NOP | 0 0 0 0 0 0 0 0 | No-operation | 4 |
| HLT | 0 1 1 1 0 1 1 0 | Halt | 7 |

**NOTES:**
1. DDD or SSS: B = 000, C = 001, D = 010, E = 011, H = 100, L = 101, Memory = 110, A = 111.
2. Two possible cycle times (6/12) indicate instruction cycles dependent on condition flags.
*All mnemonics copyright © Intel Corporation 1977

intel

# 8080 Microcomputer Systems User's Manual

September 1975




98-153C

intel
8080
Microcomputer
Systems
User's Manual
September 1975

In December 1973 Intel shipped the first 8-bit, N-channel microprocessor, the 8080. Since then it has become the most widely used microprocessor in the industry. Applications of the 8080 span from large, intelligent systems terminals to decompression computers for deep sea divers.

This 8080 Microcomputer Systems User's Manual presents all of the 8080 system components. Over twenty-five devices are described in detail. These new devices further enhance the 8080 system:

8080A — 8-Bit Central Processor Unit
- Functionally and Electrically Compatible with the 8080.
- TTL Drive Capability.
- Enhanced Timing.

8224 — Clock Generator for 8080A.
- Single 16 Pin (DIP) Package.
- Auxiliary Timing Functions.
- Power-On Reset.

8228 — System Controller for 8080A.
- Single 28 Pin (DIP) Package.
- Single Interrupt Vector (RST 7).
- Multi-Byte Interrupt Instruction Capability (e.g. CALL).
- Direct Data and Control Bus Connect to all 8080 System I/O and Memory Components.

8251 — Programmable Communication Interface.
- ASYNC or SYNC (including IBM bi-SYNC).
- Single 28 Pin Package.
- Single +5 Volt Power Supply.

8255 — Programmable Peripheral Interface.
- Three 8-Bit Ports.
- Bit Set/Reset Capability.
- Interrupt Generation.
- Single 40 Pin Package.
- Single +5 Volt Power Supply.

In addition, new memory components include: 8708, 8K Erasable PROM; 8316A, High Density Mask ROM; and 5101, Low Power CMOS RAM.

8080 Microcomputer Systems User's Manual

September 1975

# CONTENTS

# INTRODUCTION

Since their inception, digital computers have continuously become more efficient, expanding into new applications with each major technological improvement. The advent of minicomputers enabled the inclusion of digital computers as a permanent part of various process control systems. Unfortunately, the size and cost of minicomputers in "dedicated" applications has limited their use. Another approach has been the use of custom built systems made up of "random logic" (i.e., logic gates, flip-flops, counters, etc.). However, the huge expense and development time involved in the design and debugging of these systems has restricted their use to large volume applications where the development costs could be spread over a large number of machines.

Today, Intel offers the systems designer a new alternative . . . **the microcomputer**. Utilizing the technologies and experience gained in becoming the world's largest supplier of LSI memory components, Intel has made the power of the digital computer available at the integrated circuit level. Using the n-channel silicon gate MOS process, Intel engineers have implemented the fast (2 μs. cycle) and powerful (72 basic instructions) 8080 microprocessor on a single LSI chip. When this processor is combined with memory and I/O circuits, the computer is complete. Intel offers a variety of random-access memory (RAM), read-only memory (ROM) and shift register circuits, that combine with the 8080 processor to form the MCS-80™ microcomputer system, a system that can directly address and retrieve as many as 65,536 bytes stored in the memory devices.

The 8080 processor is packaged in a 40-pin dual in-line package (DIP) that allows for remarkably easy interfacing. The 8080 has a 16-bit address bus, a 8-bit bidirectional data bus and fully decoded, TTL-compatible control outputs. In addition to supporting up to 64K bytes of mixed RAM and ROM memory, the 8080 can address up to 256 input ports and 256 output ports; thus allowing for virtually unlimited system expansion. *The 8080 instruction set includes conditional branching, decimal as well as binary arithmetic, logical, register-to-register, stack control and memory reference instructions.* In fact, the 8080 instruction set is powerful enough to rival the performance of many of the much higher priced minicomputers, yet the 8080 is upward software compatible with Intel's earlier 8008 microprocessor (i.e., programs written for the 8008 can be assembled and executed on the 8080).

In addition to an extensive instruction set oriented to problem solving, the 8080 has another significant feature—SPEED. In contrast to random logic designs which tend to work in parallel, the microcomputer works by sequentially executing its program. As a result of this sequential execution, the number of tasks a microcomputer can undertake in a given period of time is directly proportional to the execution speed of the microcomputer. The speed of execution is the limiting factor of the realm of applications of the microcomputer. The 8080, with instruction times as short as 2 μsec., is an order of magnitude faster than earlier generations of microcomputers, and therefore has an expanded field of potential applications.

The architecture of the 8080 also shows a significant improvement over earlier microcomputer designs. The 8080 contains a 16-bit stack pointer that controls the addressing of an external stack located in memory. The pointer can be initialized via the proper instructions such that any portion of external memory can be used as a last in/first out stack; thus enabling almost unlimited subroutine nesting. The stack pointer allows the contents of the program counter, the accumulator, the condition flags or any of the data registers to be stored in or retrieved from the external stack. In addition, multi-level interrupt processing is possible using the 8080's stack control instructions. The status of the processor can be "pushed" onto the stack when an interrupt is accepted, then "popped" off the stack after the interrupt has been serviced. This ability to save the contents of the processor's registers *is possible even if an interrupt service* routine, itself, is interrupted.

| | CONVENTIONAL SYSTEM | PROGRAMMED LOGIC |
|---|---|---|
| Product definition | | Simplified because of ease of incorporating features |
| System and logic design | Done with logic diagrams | Can be programmed with design aids (compilers, assemblers, editors) |
| Debug | Done with conventional Lab Instrumentation | Software and hardware aids reduce time |
| PC card layout | | Fewer cards to layout |
| Documentation | | Less hardware to document |
| Cooling and packaging | | Reduced system size and power consumption eases job |
| Power distribution | | Less power to distribute |
| Engineering changes | Done with yellow wire | Change program |

**Table 0-1. The Advantages of Using Microprocessors**

## ADVANTAGES OF DESIGNING WITH MICROCOMPUTERS

Microcomputers simplify almost every phase of product development. The first step, as in any product development program, is to identify the various functions that the end system is expected to perform. Instead of realizing these functions with networks of gates and flip-flops, the functions are implemented by encoding suitable sequences of instructions (programs) in the memory elements. Data and certain types of programs are stored in RAM, while the basic program can be stored in ROM. The microprocessor performs all of the system's functions by fetching the instructions in memory, executing them and communicating the results via the microcomputer's I/O ports. An 8080 microprocessor, executing the programmed logic stored in a single 2048-byte ROM element, can perform the same logical functions that might have previously required up to 1000 logic gates.

The benefits of designing a microcomputer into your system go far beyond the advantages of merely simplifying product development. You will also appreciate the profit-making advantages of using a microcomputer in place of custom-designed random logic. The most apparent advantage is the significant savings in hardware costs. A microcomputer chip set replaces dozens of random logic elements, thus reducing the cost as well as the size of your system. In addition, production costs drop as the number of individual components to be handled decreases, and the number of complex printed circuit boards (which are difficult to layout, test and correct) is greatly reduced. Probably the most profitable advantage of a microcomputer is its flexibility for change. To modify your system, you merely re-program the memory elements; you don't have to redesign the entire system. You can imagine the savings in time and money when you want to upgrade your product. Reliability is another reason to choose the microcomputer over random logic. As the number of components decreases, the probability of a malfunctioning element likewise decreases. All of the logical control functions formerly performed by numerous hardware components can now be implemented in a few ROM circuits which are non-volatile; that is, the contents of ROM will never be lost, even in the event of a power failure. Table 0-1 summarizes many of the advantages of using microcomputers.

## MICROCOMPUTER DESIGN AIDS

If you're used to logic design and the idea of designing with programmed logic seems like too radical a change, regardless of advantages, there's no need to worry because Intel has already done most of the groundwork for you. The INTELLEC® 8 Development Systems provide flexible, inexpensive and simplified methods for OEM product development. The INTELLEC® 8 provides RAM program storage making program loading and modification easier, a display and control console for system monitoring and debugging, a standard TTY interface, a PROM programming capability and a standard software package (System Monitor, Assembler and Test Editor). In addition to the standard software package available with the INTELLEC® 8, Intel offers a PL/M compiler, a cross-assembler and a simulator written in FORTRAN IV and designed to run on any large scale computer. These programs may be procured directly from Intel or from a number of nationwide computer time-sharing services. Intel's Microcomputer Systems Group is always available to provide assistance in every phase of your product development.

Intel also provides complete documentation on all their hardware and software products. In addition to this User's Manual, there are the:

- PL/M Language Reference Manual
- 8080 Assembly Language Programming Manual
- INTELLEC®8/MOD 80 Operator's Manual
- INTELLEC®8/MOD 80 Hardware Reference Manual
- 8080 User's Program Library

## APPLICATIONS EXAMPLE

The 8080 can be used as the basis for a wide variety of calculation and control systems. The system configurations for particular applications will differ in the nature of the peripheral devices used and in the amount and the type of memory required. The applications and solutions described in this section are presented primarily to show how *microcomputers can be used to solve design problems.* The 8080 should not be considered limited either in scope or performance to those applications listed here.

Consider an 8080 microcomputer used within an automatic computing scale for a supermarket. The basic machine has two input devices: the weighing unit and a keyboard, used for function selection and to enter the price per unit of weight. The only output device is a display showing the total price, although a ticket printer might be added as an optional output device.

The control unit must accept weight information from the weighing unit, function and data inputs from the keyboard, and generate the display. The only arithmetic function to be performed is a simple multiplication of weight times rate.

The control unit could probably be realized with standard TTL logic. State diagrams for the various portions could be drawn and a multiplier unit designed. The whole design could then be tied together, and eventually reduced to a selection of packages and a printed circuit board layout. In effect, when designing with a logic family such as TTL, the designs are "customized" by the choice of packages and the wiring of the logic.

If, however, an 8080 microcomputer is used to realize the control unit (as shown in Figure 0-1), the only "custom" logic will be that of the interface circuits. These circuits are usually quite simple, providing electrical buffering for the input and output signals.

Instead of drawing state diagrams leading to logic, the system designer now prepares a flow chart, indicating which input signals must be read, what processing and computa*tions are needed, and what output signals must be produced.* A program is written from the flow chart. The program is then assembled into bit patterns which are loaded into the program memory. Thus, this system is customized primarily by the contents of program memory.

For this automatic scale, the program would probably reside in read-only memory (ROM), since the microcomputer would always execute the same program, the one which implements the scale functions. The processor would constantly monitor the keyboard and weighing unit, and update the display whenever necessary. The unit would require very little data memory; it would only be needed for rate storage, intermediate results, and for storing a copy of the display.

When the control portion of a product is implemented with a microcomputer chip set, functions can be changed and features added merely by altering the program in memory. With a TTL based system, however, alterations may require extensive rewiring, alteration of PC boards, etc.

The number of applications for microcomputers is limited only by the depth of the designer's imagination. We have listed a few potential applications in Table 0-2, along with the types of peripheral devices usually associated with each product.

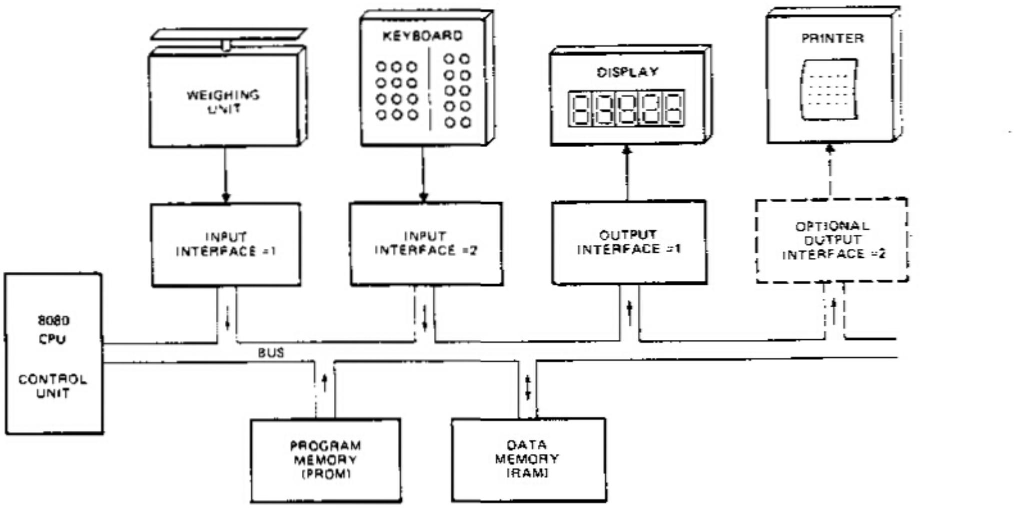



**Figure 0-1. Microcomputer Application — Automatic Scale**

| APPLICATION | PERIPHERAL DEVICES ENCOUNTERED |
|---|---|
| Intelligent Terminals | Cathode Ray Tube Display<br>Printing Units<br>Synchronous and Asynchronous data lines<br>Cassette Tape Unit<br>Keyboards |
| Gaming Machines | Keyboards, pushbuttons and switches<br>Various display devices<br>Coin acceptors<br>Coin dispensers |
| Cash Registers | Keyboard or Input Switch Array<br>Change Dispenser<br>Digital Display<br>Ticket Printer<br>Magnetic Card reader<br>Communication interface |
| Accounting and Billing Machines | Keyboard<br>Printer Unit<br>Cassette or other magnetic tape unit<br>"Floppy" disks |
| Telephone Switching Control | Telephone Line Scanner<br>Analog Switching Network<br>Dial Registers<br>Class of Service Parcel |
| Numerically Controlled Machines | Magnetic or Paper Tape Reader<br>Stepper Motors<br>Optical Shaft Encoders |
| Process Control | Analog-to-Digital Converters<br>Digital-to-Analog Converters<br>Control Switches<br>Displays |

Table 0-2. Microprocessor Applications

# CHAPTER 1 THE FUNCTIONS OF A COMPUTER

This chapter introduces certain basic computer concepts. It provides background information and definitions which will be useful in later chapters of this manual. Those already familiar with computers may skip this material, at their option.

## A TYPICAL COMPUTER SYSTEM

A typical digital computer consists of:

a) A central processor unit (CPU)
b) A memory
c) Input/output (I/O) ports

The memory serves as a place to store **Instructions**, the coded pieces of information that direct the activities of the CPU, and **Data**, the coded pieces of information that are processed by the CPU. A group of logically related instructions stored in memory is referred to as a **Program**. The CPU "reads" each instruction from memory in a logically determined sequence, and uses it to initiate processing actions. If the program sequence is coherent and logical, processing the program will produce intelligible and useful results.

The memory is also used to store the data to be manipulated, as well as the instructions that direct that manipulation. The program must be organized such that the CPU does not read a non-instruction word when it expects to see an instruction. The CPU can rapidly access any data stored in memory; but often the memory is not large enough to store the entire data bank required for a particular application. The problem can be resolved by providing the computer with one or more **Input Ports**. The CPU can address these ports and input the data contained there. The addition of input ports enables the computer to receive information from external equipment (such as a paper tape reader or floppy disk) at high rates of speed and in large volumes.

A computer also requires one or more **Output Ports** that permit the CPU to communicate the result of its processing to the outside world. The output may go to a display, for use by a human operator, to a peripheral device that produces "hard-copy," such as a line-printer, to a peripheral storage device, such as a floppy disk unit, or the output may constitute process control signals that direct the operations of another system, such as an automated assembly line. Like input ports, output ports are addressable. The input and output ports together permit the processor to communicate with the outside world.

The CPU unifies the system. It controls the functions performed by the other components. The CPU must be able to fetch instructions from memory, decode their binary contents and execute them. It must also be able to reference memory and I/O ports as necessary in the execution of instructions. In addition, the CPU should be able to recognize and respond to certain external control signals, such as INTERRUPT and WAIT requests. The functional units within a CPU that enable it to perform these functions are described below.

## THE ARCHITECTURE OF A CPU

A typical central processor unit (CPU) consists of the following interconnected functional units:

- Registers
- Arithmetic/Logic Unit (ALU)
- Control Circuitry

Registers are temporary storage units within the CPU. Some registers, such as the program counter and instruction register, have dedicated uses. Other registers, such as the accumulator, are for more general purpose use.

### Accumulator:

The accumulator usually stores one of the operands to be manipulated by the ALU. A typical instruction might direct the ALU to add the contents of some other register to the contents of the accumulator and store the result in the accumulator itself. In general, the accumulator is both a source (operand) and a destination (result) register.

Often a CPU will include a number of additional general purpose registers that can be used to store operands or intermediate data. The availability of general purpose registers eliminates the need to "shuffle" intermediate results back and forth between memory and the accumulator, thus improving processing speed and efficiency.

### Program Counter (Jumps, Subroutines and the Stack):

The instructions that make up a program are stored in the system's memory. The central processor references the contents of memory, in order to determine what action is appropriate. This means that the processor must know which location contains the next instruction.

Each of the locations in memory is numbered, to distinguish it from all other locations in memory. The number which identifies a memory location is called its **Address**.

The processor maintains a counter which contains the address of the next program instruction. This register is called the **Program Counter**. The processor updates the program counter by adding "1" to the counter each time it fetches an instruction, so that the program counter is always current (pointing to the next instruction).

The programmer therefore stores his instructions in numerically adjacent addresses, so that the lower addresses contain the first instructions to be executed and the higher addresses contain later instructions. The only time the programmer may violate this sequential rule is when an instruction in one section of memory is a **Jump** instruction to another section of memory.

A jump instruction contains the address of the instruction which is to follow it. The next instruction may be stored in any memory location, as long as the programmed jump specifies the correct address. During the execution of a jump instruction, the processor replaces the contents of its program counter with the address embodied in the Jump. Thus, the logical continuity of the program is maintained.

A special kind of program jump occurs when the stored program "**Calls**" a subroutine. In this kind of jump, the processor is required to "remember" the contents of the program counter at the time that the jump occurs. This enables the processor to resume execution of the main program when it is finished with the last instruction of the subroutine.

A **Subroutine** is a program within a program. Usually it is a general-purpose set of instructions that must be executed repeatedly in the course of a main program. Routines which calculate the square, the sine, or the logarithm of a program variable are good examples of functions often written as subroutines. Other examples might be programs designed for inputting or outputting data to a particular peripheral device.

The processor has a special way of handling subroutines, in order to insure an orderly return to the main program. When the processor receives a Call instruction, it increments the Program Counter and stores the counter's contents in a reserved memory area known as the **Stack**. The Stack thus saves the address of the instruction to be executed after the subroutine is completed. Then the processor loads the address specified in the Call into its Program Counter. The next instruction fetched will therefore be the first step of the subroutine.

The last instruction in any subroutine is a **Return**. Such an instruction need specify no address. When the processor fetches a Return instruction, it simply replaces the current *contents of the Program Counter with the address on the* top of the stack. This causes the processor to resume execution of the calling program at the point immediately following the original Call Instruction.

Subroutines are often **Nested**; that is, one subroutine will sometimes call a second subroutine. The second may call a third, and so on. This is perfectly acceptable, as long as the processor has enough capacity to store the necessary return addresses, and the logical provision for doing so. In other words, the maximum depth of nesting is determined by the depth of the stack itself. If the stack has space for storing three return addresses, then three levels of subroutines may be accommodated.

Processors have different ways of maintaining stacks. Some have facilities for the storage of return addresses built into the processor itself. Other processors use a reserved area of external memory as the stack and simply maintain a **Pointer** register which contains the address of the most recent stack entry. The external stack allows virtually unlimited subroutine nesting. In addition, if the processor provides instructions that cause the contents of the accumulator and other general purpose registers to be "pushed" onto the stack or "popped" off the stack via the address stored in the stack pointer, multi-level interrupt processing (described later in this chapter) is possible. The status of the processor (i.e., the contents of all the registers) can be saved in the stack when an interrupt is accepted and then restored after the interrupt has been serviced. This ability to save the processor's status at any given time is possible even if an interrupt service routine, itself, is interrupted.

### Instruction Register and Decoder:

Every computer has a **Word Length** that is characteristic of that machine. A computer's word length is usually determined by the size of its internal storage elements and interconnecting paths (referred to as **Busses**); for example, a computer whose registers and busses can store and transfer 8 bits of information has a characteristic word length of 8-bits and is referred to as an 8-bit parallel processor. An eight-bit parallel processor generally finds it most efficient to deal with eight-bit binary fields, and the memory associated with such a processor is therefore organized to store eight bits in each addressable memory location. Data and instructions are stored in memory as eight-bit binary numbers, or as numbers that are integral multiples of eight bits: 16 bits, 24 bits, and so on. This characteristic eight-bit field is often referred to as a **Byte**.

Each operation that the processor can perform is identified by a unique byte of data known as an **Instruction**

**Code** or **Operation Code**. An eight-bit word used as an instruction code can distinguish between 256 alternative actions, more than adequate for most processors.

The processor fetches an instruction in two distinct operations. First, the processor transmits the address in its Program Counter to the memory. Then the memory returns the addressed byte to the processor. The CPU stores this instruction byte in a register known as the **Instruction Register**, and uses it to direct activities during the remainder of the instruction execution.

The mechanism by which the processor translates an instruction code into specific processing actions requires more elaboration than we can here afford. The concept, however, should be intuitively clear to any logic designer. The eight bits stored in the instruction register can be decoded and used to selectively activate one of a number of output lines, in this case up to 256 lines. Each line represents a set of activities associated with execution of a particular instruction code. The enabled line can be combined with selected timing pulses, to develop electrical signals that can then be used to initiate specific actions. This translation of code into action is performed by the **Instruction Decoder** and by the associated control circuitry.

An eight-bit instruction code is often sufficient to specify a particular processing action. There are times, however, when execution of the instruction requires more information than eight bits can convey.

One example of this is when the instruction references a memory location. The basic instruction code identifies the operation to be performed, but cannot specify the object address as well. In a case like this, a two- or three-byte instruction must be used. Successive instruction bytes are stored in sequentially adjacent memory locations, and the processor performs two or three fetches in succession to obtain the full instruction. The first byte retrieved from memory is placed in the processor's instruction register, and subsequent bytes are placed in temporary storage; the processor then proceeds with the execution phase. Such an instruction is referred to as **Variable Length**.

### Address Register(s):

A CPU may use a register or register-pair to hold the address of a memory location that is to be accessed for data. If the address register is **Programmable**, (i.e., if there are instructions that allow the programmer to alter the contents of the register) the program can "build" an address in the address register prior to executing a **Memory Reference** instruction (i.e., an instruction that reads data from memory, writes data to memory or operates on data stored in memory).

### Arithmetic/Logic Unit (ALU):

All processors contain an arithmetic/logic unit, which is often referred to simply as the **ALU**. The ALU, as its name implies, is that portion of the CPU hardware which performs the arithmetic and logical operations on the binary data.

The ALU must contain an **Adder** which is capable of combining the contents of two registers in accordance with the logic of binary arithmetic. This provision permits the processor to perform arithmetic manipulations on the data it obtains from memory and from its other inputs.

Using only the basic adder a capable programmer can write routines which will subtract, multiply and divide, giving the machine complete arithmetic capabilities. In practice, however, most ALUs provide other built-in functions, including hardware subtraction, boolean logic operations, and shift capabilities.

The ALU contains **Flag Bits** which specify certain conditions that arise in the course of arithmetic and logical manipulations. Flags typically include **Carry**, **Zero**, **Sign**, and **Parity**. It is possible to program jumps which are conditionally dependent on the status of one or more flags. Thus, for example, the program may be designed to jump to a special routine if the carry bit is set following an addition instruction.

### Control Circuitry:

The control circuitry is the primary functional unit within a CPU. Using clock inputs, the control circuitry maintains the proper sequence of events required for any processing task. After an instruction is fetched and decoded, the control circuitry issues the appropriate signals (to units both internal and external to the CPU) for initiating the proper processing action. Often the control circuitry will be capable of responding to external signals, such as an interrupt or wait request. An **Interrupt** request will cause the control circuitry to temporarily interrupt main program execution, jump to a special routine to service the interrupting device, then automatically return to the main program. A **Wait** request is often issued by a memory or I/O element that operates slower than the CPU. The control circuitry will idle the CPU until the memory or I/O port is ready with the data.

## COMPUTER OPERATIONS

There are certain operations that are basic to almost any computer. A sound understanding of these basic operations is a necessary prerequisite to examining the specific operations of a particular computer.

### Timing:

The activities of the central processor are cyclical. The processor fetches an instruction, performs the operations required, fetches the next instruction, and so on. This orderly sequence of events requires precise timing, and the CPU therefore requires a free running oscillator clock which furnishes the reference for all processor actions. The combined fetch and execution of a single instruction is referred to as an **Instruction Cycle**. The portion of a cycle identified with a clearly defined activity is called a **State**. And the interval between pulses of the timing oscillator is referred to as a **Clock Period**. As a general rule, one or more clock periods are necessary for the completion of a state, and there are several states in a cycle.

## Instruction Fetch:

The first state(s) of any instruction cycle will be dedicated to fetching the next instruction. The CPU issues a read signal and the contents of the program counter are sent to memory, which responds by returning the next instruction word. The first byte of the instruction is placed in the instruction register. If the instruction consists of more than one byte, additional states are required to fetch each byte of the instruction. When the entire instruction is present in the CPU, the program counter is incremented (in preparation for the next instruction fetch) and the instruction is decoded. The operation specified in the instruction will be executed in the remaining states of the instruction cycle. The instruction may call for a memory read or write, an input or output and/or an internal CPU operation, such as a register-to-register transfer or an add-registers operation.

## Memory Read:

An instruction **fetch** is merely a special memory read operation that brings the instruction to the CPU's instruction register. The instruction fetched may then call for data to be read from memory into the CPU. The CPU again issues a read signal and sends the proper memory address; memory responds by returning the requested word. The data received is placed in the accumulator or one of the other general purpose registers (not the instruction register).

## Memory Write:

A memory write operation is similar to a read except for the direction of data flow. The CPU issues a write signal, sends the proper memory address, then sends the data word to be written into the addressed memory location.

## Wait (memory synchronization):

As previously stated, the activities of the processor are timed by a master clock oscillator. The clock period determines the timing of all processing activity.

The speed of the processing cycle, however, is limited by the memory's **Access Time**. Once the processor has sent a read address to memory, it cannot proceed until the memory has had time to respond. Most memories are capable of responding much faster than the processing cycle requires. A few, however, cannot supply the addressed byte within the minimum time established by the processor's clock.

Therefore a processor should contain a synchronization provision, which permits the memory to request a **Wait state**. When the memory receives a read or write enable signal, it places a request signal on the processor's READY line, causing the CPU to idle temporarily. After the memory has had time to respond, it frees the processor's READY line, and the instruction cycle proceeds.

## Input/Output:

Input and Output operations are similar to memory read and write operations with the exception that a peripheral I/O device *is addressed instead of a memory location*. The CPU issues the appropriate input or output control signal, sends the proper device address and either receives the data being input or sends the data to be output.

Data can be input/output in either parallel or serial form. All data within a digital computer is represented in binary coded form. A binary data word consists of a group of bits; each bit is either a one or a zero. **Parallel** I/O consists of transferring all bits in the word at the same time, one bit per line. **Serial** I/O consists of transferring one bit at a time on a single line. Naturally serial I/O is much slower, but it requires considerably less hardware than does parallel I/O.

## Interrupts:

**Interrupt** provisions are included on many central processors, as a means of improving the processor's efficiency. Consider the case of a computer that is processing a large volume of data, portions of which are to be output to a printer. The CPU can output a byte of data within a single machine cycle but it may take the printer the equivalent of many machine cycles to actually print the character specified by the data byte. The CPU could then remain idle waiting until the printer can accept the next data byte. If an interrupt capability is implemented on the computer, the CPU can output a data byte then return to data processing. When the printer is ready to accept the next data byte, it can request an interrupt. When the CPU acknowledges the interrupt, it suspends main program execution and automatically branches to a routine that will output the next data byte. After the byte is output, the CPU continues with main program execution. Note that this is, in principle, quite similar to a subroutine call, except that the jump is initiated externally rather than by the program.

More complex interrupt structures are possible, in which several interrupting devices share the same processor but have different priority levels. Interruptive processing is an important feature that enables maximum untilization of a processor's capacity for high system throughput.

## Hold:

Another important feature that improves the throughput of a processor is the **Hold**. The hold provision enables **Direct Memory Access** (DMA) operations.

In ordinary input and output operations, the processor itself supervises the entire data transfer. Information to be placed in memory is transferred from the input device to the processor, and then from the processor to the designated memory location. In similar fashion, information that goes from memory to output devices goes by way of the processor.

Some peripheral devices, however, are capable of transferring information to and from memory much faster than the processor itself can accomplish the transfer. If any appreciable quantity of data must be transferred to or from such a device, then **system throughput** will be increased by having the device accomplish the transfer directly. The processor must temporarily suspend its operation during such a transfer, to prevent conflicts that would arise if processor and peripheral device attempted to access memory simultaneously. It is for this reason that a **hold** provision is included on some processors.

# CHAPTER 2
# THE 8080 CENTRAL PROCESSOR UNIT

The 8080 is a complete 8-bit parallel, central processor unit (CPU) for use in general purpose digital computer systems. It is fabricated on a single LSI chip (see Figure 3-1), using Intel's n-channel silicon gate MOS process. The 8080 transfers data and internal state information via an 8-bit, bidirectional 3-state Data Bus ($D_0$-$D_7$). Memory and peripheral device addresses are transmitted over a separate 16-bit 3-state Address Bus ($A_0$-$A_{15}$). Six timing and control outputs (SYNC, DBIN, WAIT, $\overline{WR}$, HLDA and INTE) emanate from the 8080, while four control inputs (READY, HOLD, INT and RESET), four power inputs (+12v, +5v, -5v, and GND) and two clock inputs ($\phi_1$ and $\phi_2$) are accepted by the 8080.

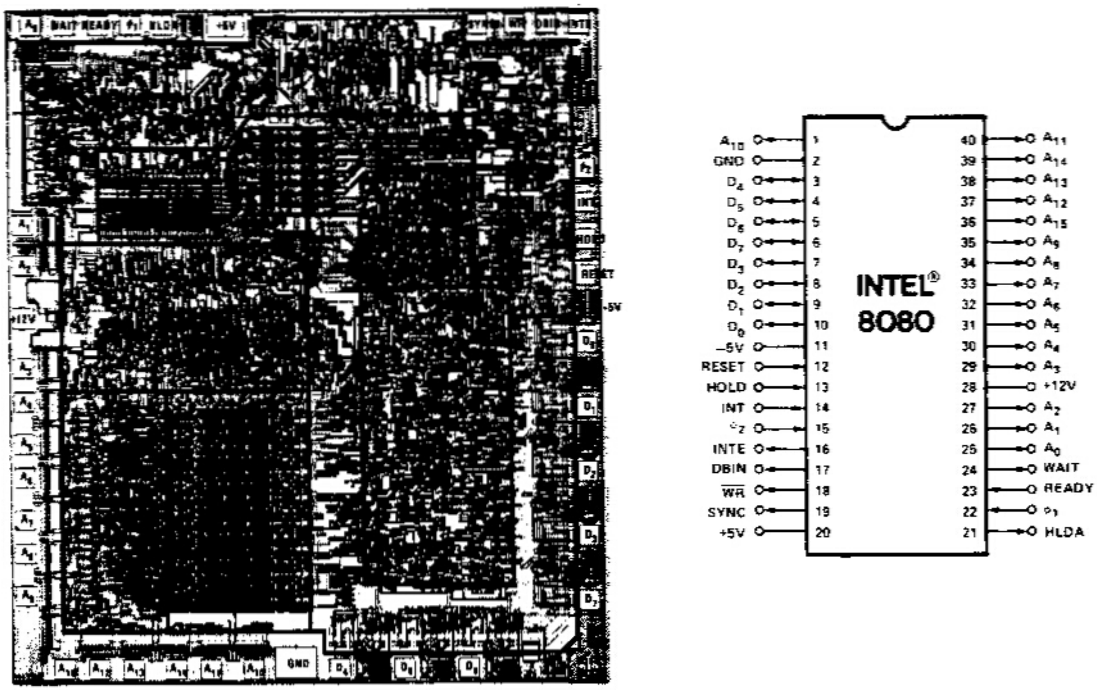



Figure 2-1. 8080 Photomicrograph With Pin Designations

## ARCHITECTURE OF THE 8080 CPU

The 8080 CPU consists of the following functional units:

- Register array and address logic
- Arithmetic and logic unit (ALU)
- Instruction register and control section
- Bi-directional, 3-state data bus buffer

Figure 2-2 illustrates the functional blocks within the 8080 CPU.

### Registers:

The register section consists of a static RAM array organized into six 16-bit registers:

- Program counter (PC)
- Stack pointer (SP)
- Six 8-bit general purpose registers arranged in pairs, referred to as B,C; D,E; and H,L
- A temporary register pair called W,Z

The program counter maintains the memory address of the current program instruction and is incremented automatically during every instruction fetch. The stack pointer maintains the address of the next available stack location in memory. The stack pointer can be initialized to use any portion of read-write memory as a stack. The stack pointer is decremented when data is "pushed" onto the stack and incremented when data is "popped" off the stack (i.e., the stack grows "downward").

The six general purpose registers can be used either as single registers (8-bit) or as register pairs (16-bit). The temporary register pair, W,Z, is not program addressable and is only used for the internal execution of instructions.

Eight-bit data bytes can be transferred between the internal bus and the register array via the register-select multiplexer. Sixteen-bit transfers can proceed between the register array and the address latch or the incrementer/decrementer circuit. The address latch receives data from any of the three register pairs and drives the 16 address output buffers ($A_0$-$A_{15}$), as well as the incrementer/decrementer circuit. The incrementer/decrementer circuit receives data from the address latch and sends it to the register array. The 16-bit data can be incremented or decremented or simply transferred between registers.

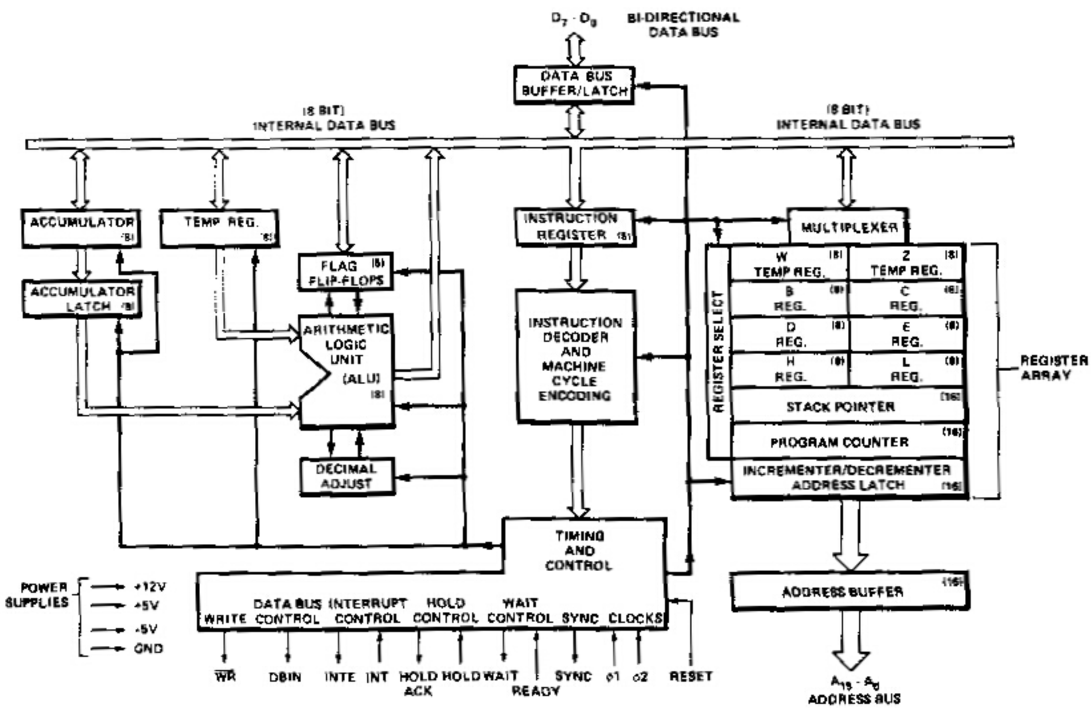



**Figure 2-2. 8080 CPU Functional Block Diagram**

## Arithmetic and Logic Unit (ALU):

The ALU contains the following registers:

- An 8-bit accumulator
- An 8-bit temporary accumulator (ACT)
- A 5-bit flag register: zero, carry, sign, parity and auxiliary carry
- An 8-bit temporary register (TMP)

Arithmetic, logical and rotate operations are performed in the ALU. The ALU is fed by the temporary register (TMP) and the temporary accumulator (ACT) and carry flip-flop. The result of the operation can be transferred to the internal bus or to the accumulator; the ALU also feeds the flag register.

The temporary register (TMP) receives information from the internal bus and can send all or portions of it to the ALU, the flag register and the internal bus.

The accumulator (ACC) can be loaded from the ALU and the internal bus and can transfer data to the temporary accumulator (ACT) and the internal bus. The contents of the accumulator (ACC) and the auxiliary carry flip-flop can be tested for decimal correction during the execution of the DAA instruction (see Chapter 4).

## Instruction Register and Control:

During an instruction fetch, the first byte of an instruction (containing the OP code) is transferred from the internal bus to the 8-bit instruction register.

The contents of the instruction register are, in turn, available to the instruction decoder. The output of the decoder, combined with various timing signals, provides the control signals for the register array, ALU and data buffer blocks. In addition, the outputs from the instruction decoder and external control signals feed the timing and state control section which generates the state and cycle timing signals.

## Data Bus Buffer:

This 8-bit bidirectional 3-state buffer is used to isolate the CPU's internal bus from the external data bus ($D_0$ through $D_7$). In the output mode, the internal bus content is loaded into an 8-bit latch that, in turn, drives the data bus output buffers. The output buffers are switched off during input or non-transfer operations.

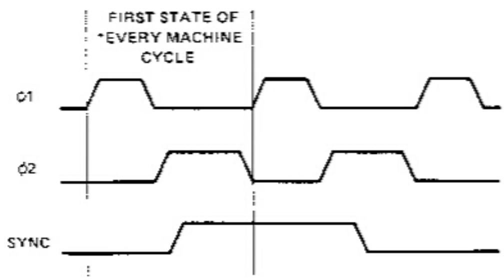
During the input mode, data from the external data bus is transferred to the internal bus. The internal bus is precharged at the beginning of each internal state, except for the transfer state ($T_3$—described later in this chapter).

## THE PROCESSOR CYCLE

An **instruction cycle** is defined as the time required to fetch and execute an instruction. During the fetch, a selected instruction (one, two or three bytes) is extracted from memory and deposited in the CPU's instruction register. During the execution phase, the instruction is decoded and translated into specific processing activities.

Every instruction cycle consists of one, two, three, four or five machine cycles. A **machine cycle** is required each time the CPU accesses memory or an I/O port. The fetch portion of an instruction cycle requires one machine cycle for each byte to be fetched. The duration of the execution portion of the instruction cycle depends on the kind of instruction that has been fetched. Some instructions do not require any machine cycles other than those necessary to fetch the instruction; other instructions, however, require additional machine cycles to write or read data to/from memory or I/O devices. The DAD instruction is an exception in that it requires two additional machine cycles to complete an internal register-pair add (see Chapter 4).

Each machine cycle consists of three, four or five states. A state is the smallest unit of processing activity and is defined as the interval between two successive positive-going transitions of the $\phi_1$ driven clock pulse. The 8080 is driven by a two-phase clock oscillator. All processing activities are referred to the period of this clock. The two non-overlapping clock pulses, labeled $\phi_1$ and $\phi_2$, are furnished by external circuitry. It is the $\phi_1$ clock pulse which divides each machine cycle into states. Timing logic within the 8080 uses the clock inputs to produce a SYNC pulse, which identifies the beginning of every machine cycle. The SYNC pulse is triggered by the low-to-high transition of $\phi_2$, as shown in Figure 2-3.




*SYNC DOES NOT OCCUR IN THE SECOND AND THIRD MACHINE CYCLES OF A DAD INSTRUCTION SINCE THESE MACHINE CYCLES ARE USED FOR AN INTERNAL REGISTER-PAIR ADD.

**Figure 2-3. $\phi_1$, $\phi_2$ And SYNC Timing**

There are three exceptions to the defined duration of a state. They are the WAIT state, the hold (HLDA) state and the halt (HLTA) state, described later in this chapter. Because the WAIT, the HLDA, and the HLTA states depend upon external events, they are by their nature of indeterminate length. Even these exceptional states, however, must be synchronized with the pulses of the driving clock. Thus, the duration of all states are integral multiples of the clock period.

To summarize then, each **clock period** marks a **state**; three to five states constitute a machine cycle; and one to five **machine cycles** comprise an **instruction cycle**. A full instruction cycle requires anywhere from four to eighteen states for its completion, depending on the kind of instruction involved.

## Machine Cycle Identification:

With the exception of the DAD instruction, there is just one consideration that determines how many machine cycles are required in any given instruction cycle: the number of times that the processor must reference a memory address or an addressable peripheral device, in order to fetch and execute the instruction. Like many processors, the 8080 is so constructed that it can transmit only one address per machine cycle. Thus, if the fetch and execution of an instruction requires two memory references, then the instruction cycle associated with that instruction consists of two machine cycles. If five such references are called for, then the instruction cycle contains five machine cycles.

Every instruction cycle has at least one reference to memory, during which the instruction is fetched. An instruction cycle must always have a fetch, even if the execution of the instruction requires no further references to memory. The first machine cycle in every instruction cycle is therefore a FETCH. Beyond that, there are no fast rules. It depends on the kind of instruction that is fetched.

Consider some examples. The add-register (ADD r) instruction is an instruction that requires only a single machine cycle (FETCH) for its completion. In this one-byte instruction, the contents of one of the CPU's six general purpose registers is added to the existing contents of the accumulator. Since all the information necessary to execute the command is contained in the eight bits of the instruction code, only one memory reference is necessary. Three states are used to extract the instruction from memory, and one additional state is used to accomplish the desired addition. The entire instruction cycle thus requires only one machine cycle that consists of four states, or four periods of the external clock.

Suppose now, however, that we wish to add the contents of a specific memory location to the existing contents of the accumulator (ADD M). Although this is quite similar in principle to the example just cited, several additional steps will be used. An extra machine cycle will be used, in order to address the desired memory location.

The actual sequence is as follows. First the processor extracts from memory the one-byte instruction word addressed by its program counter. This takes three states. The eight-bit instruction word obtained during the FETCH machine cycle is deposited in the CPU's instruction register and used to direct activities during the remainder of the instruction cycle. Next, the processor sends out, as an address, the contents of its H and L registers. The eight-bit data word returned during this MEMORY READ machine cycle is placed in a temporary register inside the 8080 CPU. By now three more clock periods (states) have elapsed. In the seventh and final state, the contents of the temporary register are added to those of the accumulator. Two machine cycles, consisting of seven states in all, complete the "ADD M" instruction cycle.

At the opposite extreme is the save H and L registers (SHLD) instruction, which requires five machine cycles. During an "SHLD" instruction cycle, the contents of the processor's H and L registers are deposited in two sequentially adjacent memory locations; the destination is indicated by two address bytes which are stored in the two memory locations immediately following the operation code byte. The following sequence of events occurs:

(1) A FETCH machine cycle, consisting of four states. During the first three states of this machine cycle, the processor fetches the instruction indicated by its program counter. The program counter is then incremented. The fourth state is used for internal instruction decoding.

(2) A MEMORY READ machine cycle, consisting of three states. During this machine cycle, the byte indicated by the program counter is read from memory and placed in the processor's Z register. The program counter is incremented again.

(3) Another MEMORY READ machine cycle, consisting of three states, in which the byte indicated by the processor's program counter is read from memory and placed in the W register. The program counter is incremented, in anticipation of the next instruction fetch.

(4) A MEMORY WRITE machine cycle, of three states, in which the contents of the L register are transferred to the memory location pointed to by the present contents of the W and Z registers. The state following the transfer is used to increment the W,Z register pair so that it indicates the next memory location to receive data.

(5) A MEMORY WRITE machine cycle, of three states, in which the contents of the H register are transferred to the new memory location pointed to by the W,Z register pair.

In summary, the "SHLD" instruction cycle contains five machine cycles and takes 16 states to execute.

Most instructions fall somewhere between the extremes typified by the "ADD r" and the "SHLD" instructions. The input (INP) and the output (OUT) instructions, for example, require three machine cycles: a FETCH, to obtain the instruction; a MEMORY READ, to obtain the address of the object peripheral; and an INPUT or an OUTPUT machine cycle, to complete the transfer.

While no one instruction cycle will consist of more then five machine cycles, the following ten different types of machine cycles may occur within an instruction cycle:

(1) FETCH (M1)

(2) MEMORY READ

(3) MEMORY WRITE

(4) STACK READ

(5) STACK WRITE

(6) INPUT

(7) OUTPUT

(8) INTERRUPT

(9) HALT

(10) *HALT • INTERRUPT*

The machine cycles that actually do occur in a particular instruction cycle depend upon the kind of instruction, with the overriding stipulation that the first machine cycle in any instruction cycle is always a FETCH.

The processor identifies the machine cycle in progress by transmitting an eight-bit status word during the first state of every machine cycle. Updated status information is presented on the 8080's data lines ($D_0$-$D_7$), during the SYNC interval. This data should be saved in latches, and used to develop control signals for external circuitry. Table 2-1 shows how the positive-true status information is distributed on the processor's data bus.

Status signals are provided principally for the control of external circuitry. Simplicity of interface, rather than machine cycle identification, dictates the logical definition of individual status bits. You will therefore observe that certain processor machine cycles are uniquely identified by a single status bit, but that others are not. The $M_1$ status bit ($D_6$), for example, unambiguously identifies a FETCH machine cycle. A STACK READ, on the other hand, is indicated by the coincidence of STACK and MEMR signals. Machine cycle identification data is also valuable in the test and de-bugging phases of system development. Table 2-1 lists the status bit outputs for each type of machine cycle.

## State Transition Sequence:

Every machine cycle within an instruction cycle consists of three to five active states (referred to as $T_1$, $T_2$, $T_3$, $T_4$, $T_5$ or $T_W$). The actual number of states depends upon the instruction being executed, and on the particular machine cycle within the greater instruction cycle. The state transition diagram in Figure 2-4 shows how the 8080 proceeds from state to state in the course of a machine cycle. The diagram also shows how the READY, HOLD, and INTERRUPT lines are sampled during the machine cycle, and how the conditions on these lines may modify the basic transition sequence. In the present discussion, we are concerned only with the basic sequence and with the READY function. The HOLD and INTERRUPT functions will be discussed later.

The 8080 CPU does not directly indicate its internal state by transmitting a "state control" output during each state; instead, the 8080 supplies direct control output (INTE, HLDA, DBIN, $\overline{WR}$ and WAIT) for use by external circuitry.

Recall that the 8080 passes through at least three states in every machine cycle, with each state defined by successive low-to-high transitions of the $\phi_1$ clock. Figure 2-5 shows the timing relationships in a typical FETCH machine cycle. Events that occur in each state are referenced to transitions of the $\phi_1$ and $\phi_2$ clock pulses.

The SYNC signal identifies the first state ($T_1$) in every machine cycle. As shown in Figure 2-5, the SYNC signal is related to the leading edge of the $\phi_2$ clock. There is a delay ($t_{DC}$) between the low-to-high transition of $\phi_2$ and the positive-going edge of the SYNC pulse. There also is a corresponding delay (also $t_{DC}$) between the next $\phi_2$ pulse and the falling edge of the SYNC signal. Status information is displayed on $D_0$-$D_7$ during the same $\phi_2$ to $\phi_2$ interval. Switching of the status signals is likewise controlled by $\phi_2$.

The rising edge of $\phi_2$ during $T_1$ also loads the processor's address lines ($A_0$-$A_{15}$). These lines become stable within a brief delay ($t_{DA}$) of the $\phi_2$ clocking pulse, and they remain stable until the first $\phi_2$ pulse after state $T_3$. This gives the processor ample time to read the data returned from memory.

Once the processor has sent an address to memory, there is an opportunity for the memory to request a WAIT. This it does by pulling the processor's READY line low, prior to the "Ready set-up" interval ($t_{RS}$) which occurs during the $\phi_2$ pulse within state $T_2$ or $T_W$. As long as the READY line remains low, the processor will idle, giving the memory time to respond to the addressed data request. Refer to Figure 2-5.

The processor responds to a wait request by entering an alternative state ($T_W$) at the end of $T_2$, rather than proceeding directly to the $T_3$ state. Entry into the $T_W$ state is indicated by a WAIT signal from the processor, acknowledging the memory's request. A low-to-high transition on the WAIT line is triggered by the rising edge of the $\phi_1$ clock and occurs within a brief delay ($t_{DC}$) of the actual entry into the $T_W$ state.

A wait period may be of indefinite duration. The processor remains in the waiting condition until its READY line again goes high. A READY indication must precede the falling edge of the $\phi_2$ clock by a specified interval ($t_{RS}$), in order to guarantee an exit from the $T_W$ state. The cycle may then proceed, beginning with the rising edge of the next $\phi_1$ clock. A WAIT interval will therefore consist of an integral number of $T_W$ states and will always be a multiple of the clock period.

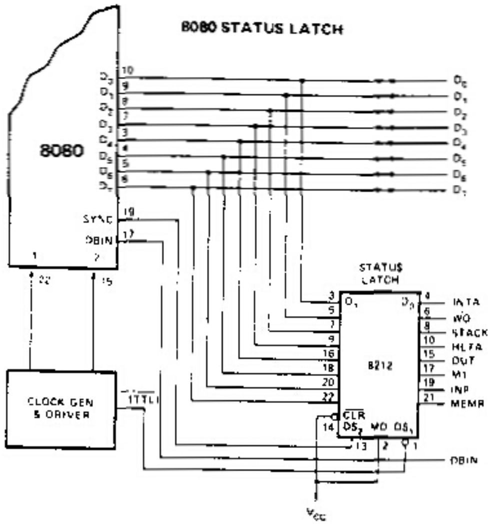
Instructions for the 8080 require from one to five machine cycles for complete execution. The 8080 sends out 8 bit of status information on the data bus at the beginning of each machine cycle (during SYNC time). The following table defines the status information.

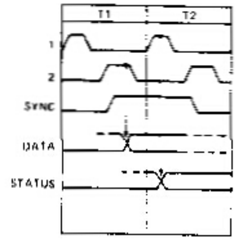
### STATUS INFORMATION DEFINITION

| Symbols | Data Bus Bit | Definition |
|---|---|---|
| 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. |
| $\overline{WO}$ | $D_1$ | 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. |
| STACK | $D_2$ | Indicates that the address bus holds the pushdown stack address from the Stack Pointer. |
| HLTA | $D_3$ | Acknowledge signal for HALT instruction. |
| 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. |
| $M_1$ | $D_5$ | Provides a signal to indicate that the CPU is in the fetch cycle for the first byte of an instruction. |
| 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. |
| MEMR* | $D_7$ | Designates that the data bus will be used for memory read data. |

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

8080 STATUS LATCH




STATUS WORD CHART

| Data Bus Bit | Status Information | ① Instruction Fetch | ② Memory Read | ③ Memory Write | ④ Stack Read | ⑤ Stack Write | ⑥ Input Read | ⑦ Output Write | ⑧ Interrupt Acknowledge | ⑨ Halt Acknowledge | ⑩ Interrupt Acknowledge While Halt |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $D_0$ | INTA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| $D_1$ | $\overline{WO}$ | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| $D_2$ | STACK | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| $D_3$ | HLTA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $D_4$ | OUT | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $D_5$ | $M_1$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| $D_6$ | INP | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $D_7$ | MEMR | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |

TYPE OF MACHINE CYCLE: columns ① – ⑩. Ⓝ STATUS WORD

**Table 2-1. 8080 Status Bit Definitions**

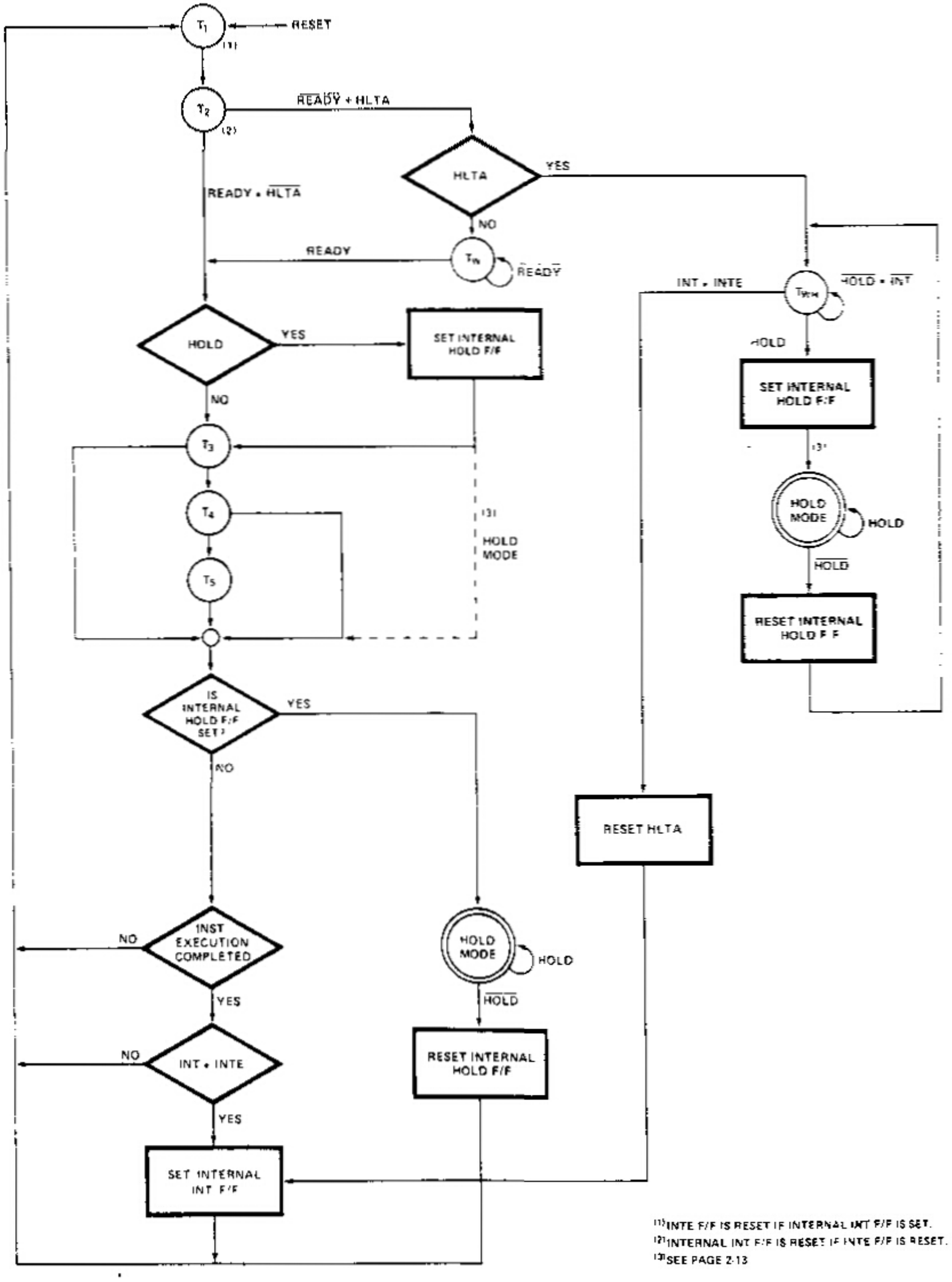



Figure 2-4. CPU State Transition Diagram

The events that take place during the $T_3$ state are determined by the kind of machine cycle in progress. In a FETCH machine cycle, the processor interprets the data on its data bus as an instruction. During a MEMORY READ or a STACK READ, data on this bus is interpreted as a data word. The processor outputs data on this bus during a MEMORY WRITE machine cycle. During I/O operations, the processor may either transmit or receive data, depending on whether an OUTPUT or an INPUT operation is involved.

Figure 2-6 illustrates the timing that is characteristic of a data input operation. As shown, the low-to-high transition of $\phi_2$ during $T_2$ clears status information from the processor's data lines, preparing these lines for the receipt of incoming data. The data presented to the processor must have stabilized prior to both the "$\phi_1$–data set-up" interval ($t_{DS1}$), that precedes the falling edge of the $\phi_1$ pulse defining state $T_3$, and the "$\phi_2$–data set-up" interval ($t_{DS2}$), that precedes the rising edge of $\phi_2$ in state $T_3$. This same data must remain stable during the "data hold" interval ($t_{DH}$) that occurs following the rising edge of the $\phi_2$ pulse. Data placed on these lines by memory or by other external devices will be sampled during $T_3$.

During the input of data to the processor, the 8080 generates a DBIN signal which should be used externally to enable the transfer. Machine cycles in which DBIN is available include: FETCH, MEMORY READ, STACK READ, and INTERRUPT. DBIN is initiated by the rising edge of $\phi_2$ during state T2 and terminated by the corresponding edge of $\phi_2$ during $T_3$. Any $T_W$ phases intervening between $T_2$ and $T_3$ will therefore extend DBIN by one or more clock periods.

Figure 2-7 shows the timing of a machine cycle in which the processor outputs data. Output data may be destined either for memory or for peripherals. The rising edge of $\phi_2$ within state $T_2$ clears status information from the CPU's data lines, and loads in the data which is to be output to external devices. This substitution takes place within the

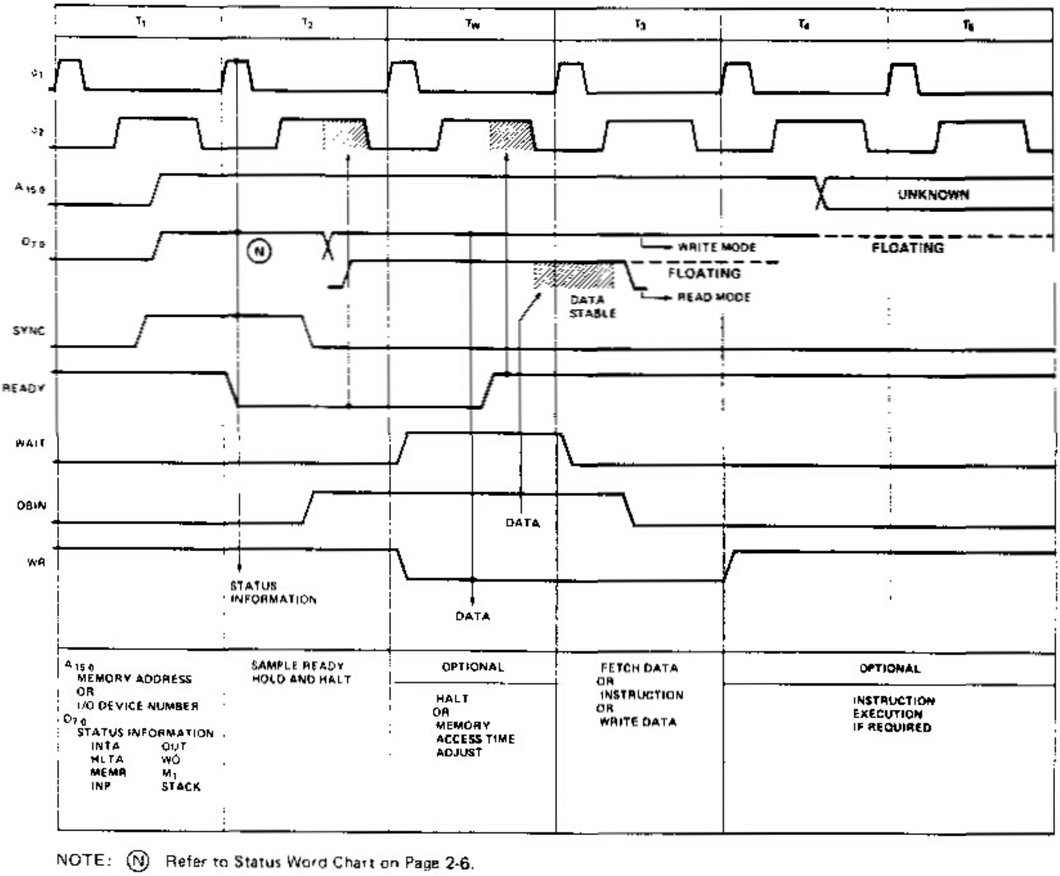
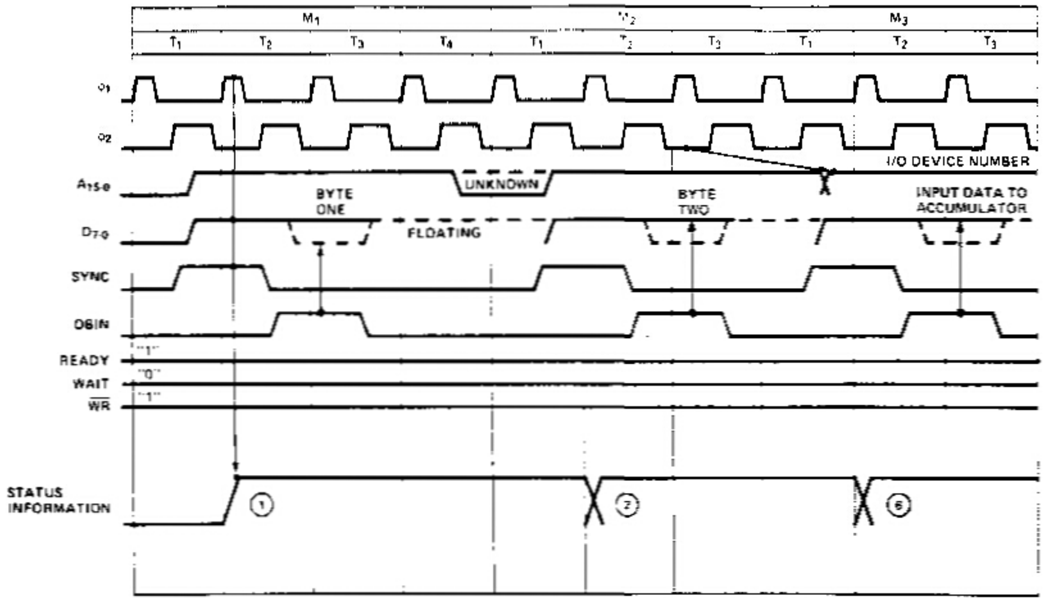
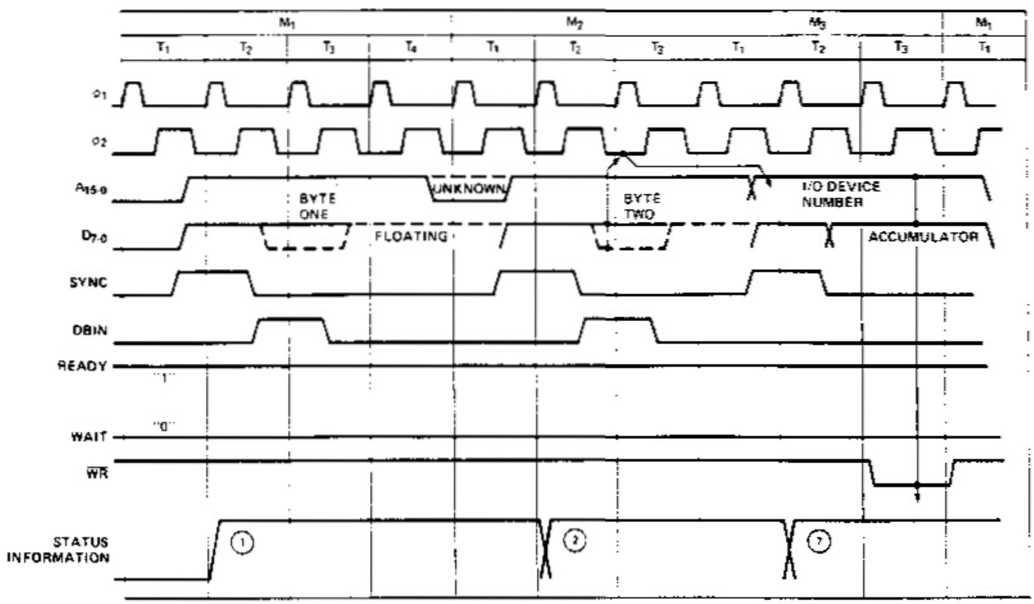



**Figure 2-5. Basic 8080 Instruction Cycle**




NOTE: (N) Refer to Status Word Chart on Page 2-6.

**Figure 2-6. Input Instruction Cycle**




NOTE. (N) Refer to Status Word Chart on Page 2-6.

**Figure 2-7. Output Instruction Cycle**

"data output delay" interval ($t_{DD}$) following the $\phi_2$ clock's leading edge. Data on the bus remains stable throughout the remainder of the machine cycle, until replaced by updated status information in the subsequent $T_1$ state. Observe that a READY signal is necessary for completion of an OUTPUT machine cycle. Unless such an indication is present, the processor enters the $T_W$ state, following the $T_2$ state. Data on the output lines remains stable in the interim, and the processing cycle will not proceed until the READY line again goes high.

The 8080 CPU generates a $\overline{WR}$ output for the synchronization of external transfers, during those machine cycles in which the processor outputs data. These include MEMORY WRITE, STACK WRITE, and OUTPUT. The negative-going leading edge of $\overline{WR}$ is referenced to the rising edge of the first $\phi_1$ clock pulse following $T_2$, and occurs within a brief delay ($t_{DC}$) of that event. $\overline{WR}$ remains low until re-triggered by the leading edge of $\phi_1$ during the state following $T_3$. Note that any $T_W$ states intervening between $T_2$ and $T_3$ of the output machine cycle will necessarily extend $\overline{WR}$, in much the same way that DBIN is affected during data input operations.

All processor machine cycles consist of at least three states: $T_1$, $T_2$, and $T_3$ as just described. If the processor has to wait for a response from the peripheral or memory with which it is communicating, then the machine cycle may also contain one or more $T_W$ states. During the three basic states, data is transferred to or from the processor.

After the $T_3$ state, however, it becomes difficult to generalize. $T_4$ and $T_5$ states are available, if the execution of a particular instruction requires them. But not all machine cycles make use of these states. It depends upon the kind of instruction being executed, and on the particular machine cycle within the instruction cycle. The processor will terminate any machine cycle as soon as its processing activities are completed, rather than proceeding through the $T_4$ and $T_5$ states every time. Thus the 8080 may exit a machine cycle following the $T_3$, the $T_4$, or the $T_5$ state and proceed directly to the $T_1$ state of the next machine cycle.

| STATE | ASSOCIATED ACTIVITIES |
|---|---|
| $T_1$ | A memory address or I/O device number is placed on the Address Bus ($A_{15\text{-}0}$); status information is placed on Data Bus ($D_{7\text{-}0}$). |
| $T_2$ | The CPU samples the READY and HOLD inputs and checks for halt instruction. |
| TW (optional) | Processor enters wait state if READY is low or if HALT instruction has been executed. |
| T3 | An instruction byte (FETCH machine cycle), data byte (MEMORY READ, STACK READ) or interrupt instruction (INTERRUPT machine cycle) is input to the CPU from the Data Bus; or a data byte (MEMORY WRITE, STACK WRITE or OUTPUT machine cycle) is output onto the data bus. |
| T4 T5 (optional) | States $T_4$ and $T_5$ are available if the execution of a particular instruction requires them; if not, the CPU may skip one or both of them. $T_4$ and $T_5$ are only used for internal processor operations. |

**Table 2-2. State Definitions**

## INTERRUPT SEQUENCES

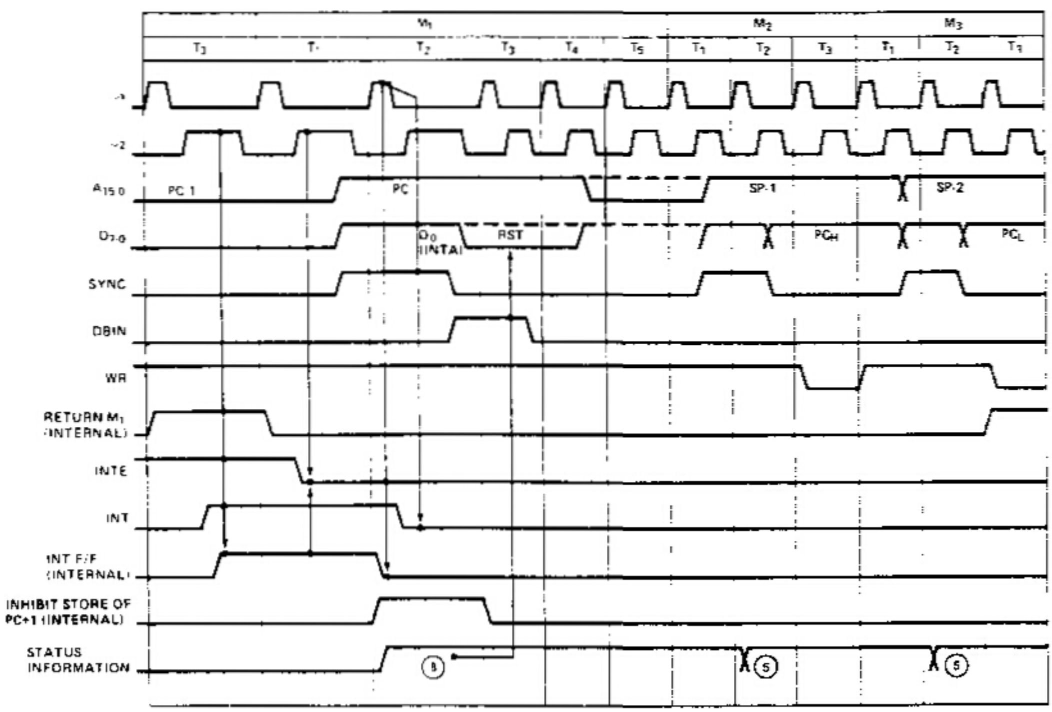
The 8080 has the built-in capacity to handle external interrupt requests. A peripheral device can initiate an interrupt simply by driving the processor's interrupt (INT) line high.

The interrupt (INT) input is asynchronous, and a request may therefore originate at any time during any instruction cycle. Internal logic re-clocks the external request, so that a proper correspondence with the driving clock is established. As Figure 2-8 shows, an interrupt request (INT) arriving during the time that the interrupt enable line (INTE) is high, acts in coincidence with the $\phi_2$ clock to set the internal interrupt latch. This event takes place during the last state of the instruction cycle in which the request occurs, thus ensuring that any instruction in progress is completed before the interrupt can be processed.

The INTERRUPT machine cycle which follows the arrival of an enabled interrupt request resembles an ordinary FETCH machine cycle in most respects. The $M_1$ status bit is transmitted as usual during the SYNC interval. It is accompanied, however, by an INTA status bit ($D_0$) which acknowledges the external request. The contents of the program counter are latched onto the CPU's address lines during $T_1$, but the counter itself is not incremented during the INTERRUPT machine cycle, as it otherwise would be. In this way, the pre-interrupt status of the program counter is preserved, so that data in the counter may be restored by the interrupted program after the interrupt request has been processed.

The interrupt cycle is otherwise indistinguishable from an ordinary FETCH machine cycle. The processor itself takes no further special action. It is the responsibility of the peripheral logic to see that an eight-bit interrupt instruction is "jammed" onto the processor's data bus during state $T_3$. In a typical system, this means that the data-in bus from memory must be temporarily disconnected from the processor's main data bus, so that the interrupting device can command the main bus without interference.

The 8080's instruction set provides a special one-byte call which facilitates the processing of interrupts (the ordinary program Call takes three bytes). This is the RESTART instruction (RST). A variable three-bit field embedded in the eight-bit field of the RST enables the interrupting device to direct a Call to one of eight fixed memory locations. The decimal addresses of these dedicated locations are: 0, 8, 16, 24, 32, 40, 48, and 56. Any of these addresses may be used to store the first instruction(s) of a routine designed to service the requirements of an interrupting device. Since the (RST) is a call, completion of the instruction also stores the old program counter contents on the STACK.




NOTE: (N) Refer to Status Word Chart on Page 2-6.

**Figure 2-8. Interrupt Timing**

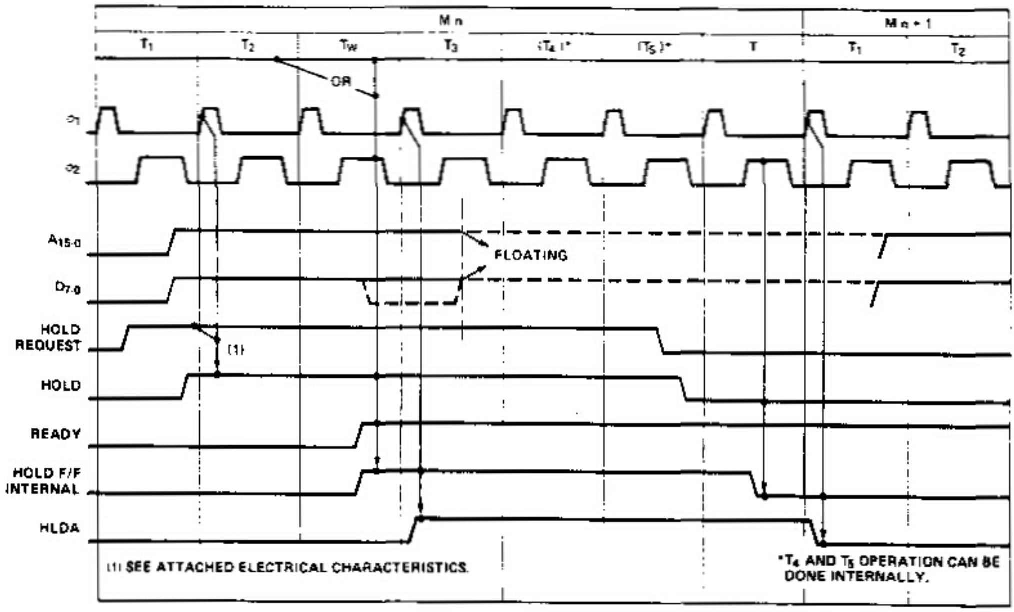



**Figure 2-9. HOLD Operation (Read Mode)**

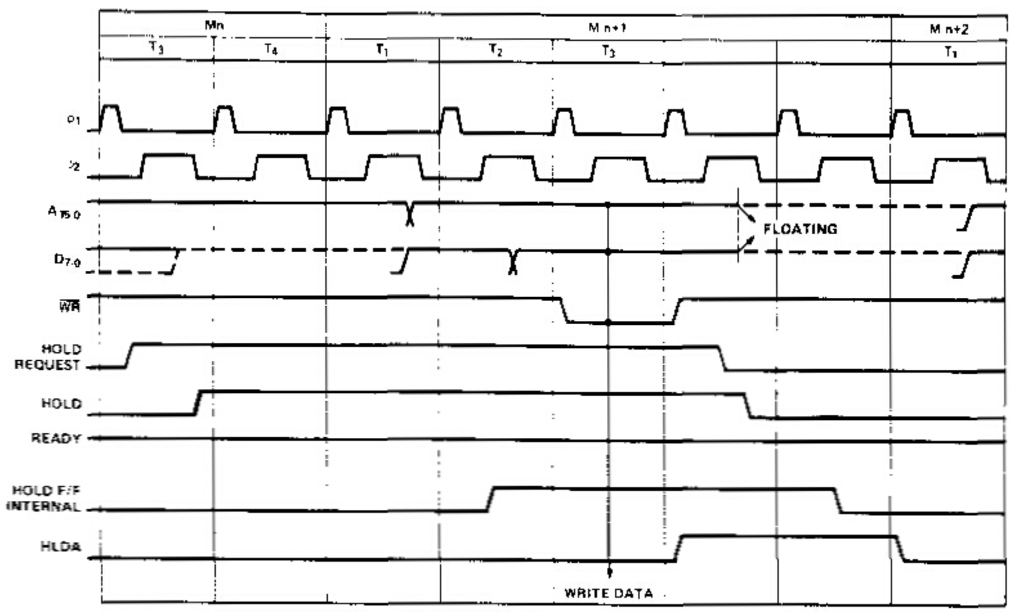



**Figure 2-10. HOLD Operation (Write Mode)**

## HOLD SEQUENCES

The 8080A CPU contains provisions for Direct Memory Access (DMA) operations. By applying a HOLD to the appropriate control pin on the processor, an external device can cause the CPU to suspend its normal operations and relinquish control of the address and data busses. The processor responds to a request of this kind by floating its address to other devices sharing the busses. At the same time, the processor acknowledges the HOLD by placing a high on its HLDA outpin pin. During an acknowledged HOLD, the address and data busses are under control of the peripheral which originated the request, enabling it to conduct memory transfers without processor intervention.

Like the interrupt, the HOLD input is synchronized internally. A HOLD signal must be stable prior to the "Hold set-up" interval ($t_{HS}$), that precedes the rising edge of $\phi_2$.

Figures 2-9 and 2-10 illustrate the timing involved in HOLD operations. Note the delay between the asynchronous HOLD REQUEST and the re-clocked HOLD. As shown in the diagram, a coincidence of the READY, the HOLD, and the $\phi_2$ clocks sets the internal hold latch. Setting the latch enables the subsequent rising edge of the $\phi_1$ clock pulse to trigger the HLDA output.

Acknowledgement of the HOLD REQUEST precedes slightly the actual floating of the processor's address and data lines. The processor acknowledges a HOLD at the beginning of $T_3$, if a read or an input machine cycle is in progress (see Figure 2-9). Otherwise, acknowledgement is deferred until the beginning of the state following $T_3$ (see Figure 2-10). In both cases, however, the HLDA goes high within a specified delay ($t_{DC}$) of the rising edge of the selected $\phi_1$ clock pulse. Address and data lines are floated within a brief delay after the rising edge of the next $\phi_2$ clock pulse. This relationship is also shown in the diagrams.

To all outward appearances, the processor has suspended its operations once the address and data busses are floated. Internally, however, certain functions may continue. If a HOLD REQUEST is acknowledged at $T_3$, and if the processor is in the middle of a machine cycle which requires four or more states to complete, the CPU proceeds through $T_4$ and $T_5$ before coming to a rest. Not until the end of the machine cycle is reached will processing activities cease. Internal processing is thus permitted to overlap the external DMA transfer, improving both the efficiency and the speed of the entire system.

The processor exits the holding state through a sequence similar to that by which it entered. A HOLD REQUEST is terminated asynchronously when the external device has completed its data transfer. The HLDA output returns to a low level following the leading edge of the next $\phi 1$ clock pulse. Normal processing resumes with the machine cycle following the last cycle that was executed.

## HALT SEQUENCES

When a halt instruction (HLT) is executed, the CPU enters the halt state ($T_{WH}$) after state $T_2$ of the next machine cycle, as shown in Figure 2-11. There are only three ways in which the 8080 can exit the halt state:

- A high on the RESET line will always reset the 8080 to state $T_1$; RESET also clears the program counter.
- A HOLD input will cause the 8080 to enter the hold state, as previously described. When the HOLD line goes low, the 8080 re-enters the halt state on the rising edge of the next $\phi_1$ clock pulse.
- An interrupt (i.e., INT goes high while INTE is enabled) will cause the 8080 to exit the Halt state and enter state $T_1$ on the rising edge of the next $\phi_1$ clock pulse. NOTE: The interrupt enable (INTE) flag **must** be set when the halt state is entered; otherwise, the 8080 will only be able to exit via a RESET signal.

Figure 2-12 illustrates halt sequencing in flow chart form.

## START-UP OF THE 8080 CPU

When power is applied initially to the 8080, the processor begins operating immediately. The contents of its program counter, stack pointer, and the other working registers are naturally subject to random factors and cannot be specified. For this reason, it will be necessary to begin the power-up sequence with RESET.

An external RESET signal of three clock period duration (minimum) restores the processor's internal program counter to zero. Program execution thus begins with memory location zero, following a RESET. Systems which require the processor to wait for an explicit start-up signal will store a halt instruction (EI, HLT) in the first two locations. A manual or an automatic INTERRUPT will be used for starting. In other systems, the processor may begin executing its stored program immediately. Note, however, that the RESET has no effect on status flags, or on any of the processor's working registers (accumulator, registers, or stack pointer). The contents of these registers remain indeterminate, until initialized explicitly by the program.

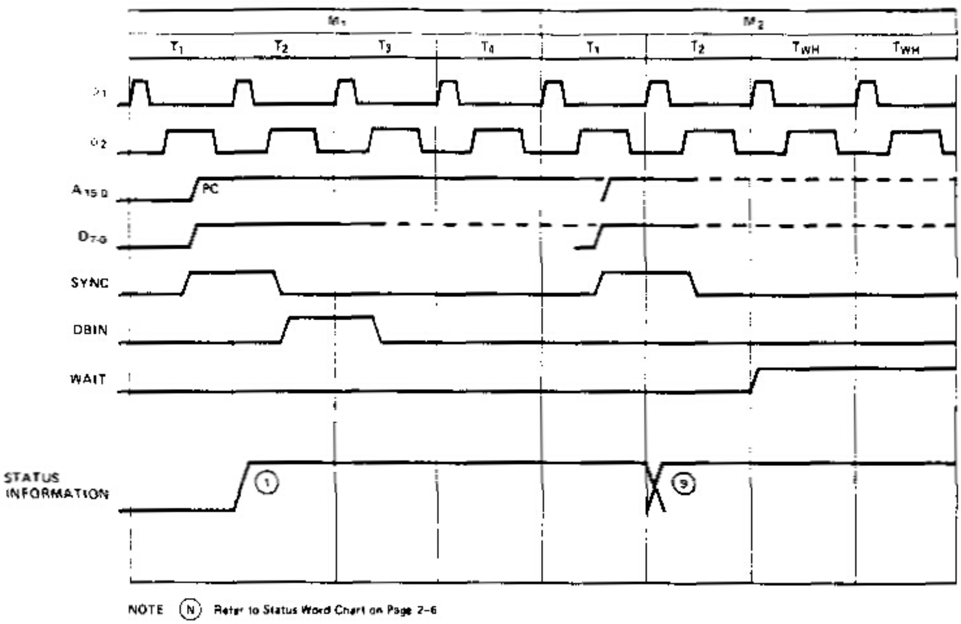



Figure 2-11. HALT Timing

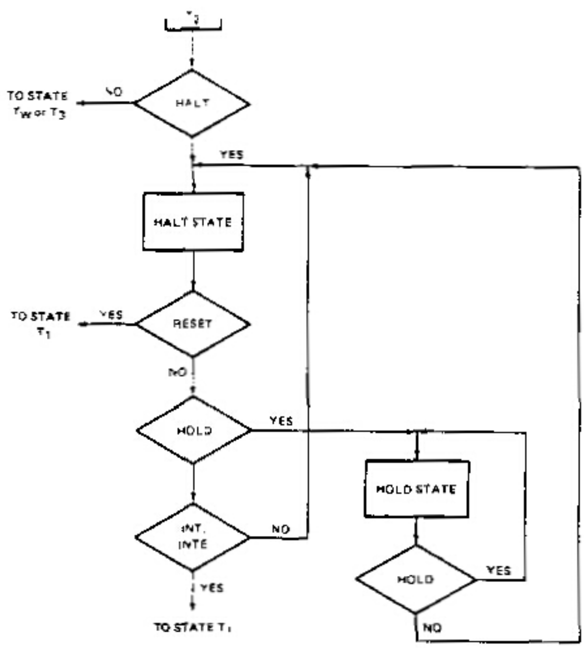
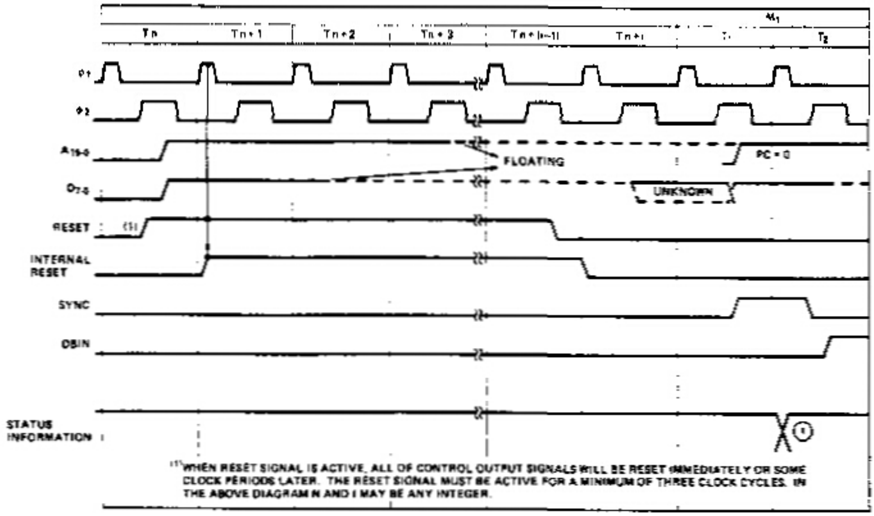
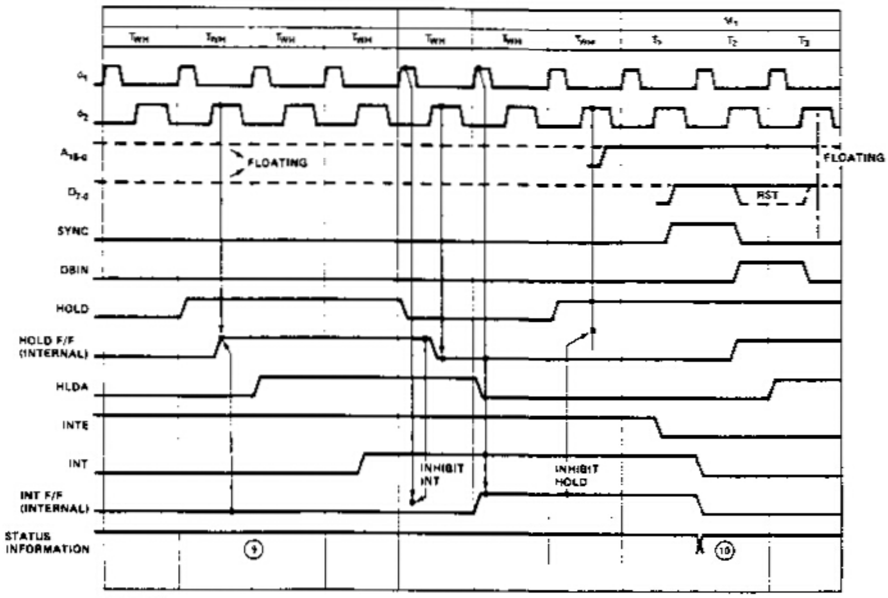



Figure 2-12. HALT Sequence Flow Chart.




NOTE: (N) Refer to Status Word Chart on Page 2-6

**Figure 2-13. Reset.**




NOTE: (N) Refer to Status Word Chart on Page 2-6

**Figure 2-14. Relation between HOLD and INT in the HALT State.**

| MNEMONIC | OP CODE | | M1[1] | | | | | M2 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | D7 D6 D5 D4 | D3 D2 D1 D0 | T1 | T2[2] | T3 | T4 | T5 | T1 | T2[2] | T3 |
| MOV r1, r2 | 0 1 D D | D S S S | PC OUT STATUS | PC = PC + 1 | INST→TMP/IR | (SSS)→TMP | (TMP)→DDD | | | |
| MOV r, M | 0 1 D D | D 1 1 0 | | | | X[3] | | HL OUT STATUS[6] | | DATA → DDD |
| MOV M, r | 0 1 1 1 | 0 S S S | | | | (SSS)→TMP | | HL OUT STATUS[7] | | (TMP) → DATA BUS |
| SPHL | 1 1 1 1 | 1 0 0 1 | | | | (HL) → | SP | | | |
| MVI r, data | 0 0 D D | D 1 1 0 | | | | X | | PC OUT STATUS[6] | | B2 → DDD |
| MVI M, data | 0 0 1 1 | 0 1 1 0 | | | | X | | | | B2 → TMP |
| LXI rp, data | 0 0 R P | 0 0 0 1 | | | | X | | | PC = PC + 1 | B2 → r1 |
| LDA addr | 0 0 1 1 | 1 0 1 0 | | | | X | | | PC = PC + 1 | B2 → Z |
| STA addr | 0 0 1 1 | 0 0 1 0 | | | | X | | | PC = PC + 1 | B2 → Z |
| LHLD addr | 0 0 1 0 | 1 0 1 0 | | | | X | | | PC = PC + 1 | B2 → Z |
| SHLD addr | 0 0 1 0 | 0 0 1 0 | | | | X | | PC OUT STATUS[6] | PC = PC + 1 | B2 → Z |
| LDAX rp[4] | 0 0 R P | 1 0 1 0 | | | | X | | rp OUT STATUS[6] | | DATA → A |
| STAX rp[4] | 0 0 R P | 0 0 1 0 | | | | X | | rp OUT STATUS[7] | | (A) → DATA BUS |
| XCHG | 1 1 1 0 | 1 0 1 1 | | | | (HL)↔(DE) | | | | |
| ADD r | 1 0 0 0 | 0 S S S | | | | (SSS)→TMP (A)→ACT | | [9] | (ACT)+(TMP)→A | |
| ADD M | 1 0 0 0 | 0 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | | DATA → TMP |
| ADI data | 1 1 0 0 | 0 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 | B2 → TMP |
| ADC r | 1 0 0 0 | 1 S S S | | | | (SSS)→TMP (A)→ACT | | [9] | (ACT)+(TMP)+CY→A | |
| ADC M | 1 0 0 0 | 1 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | | DATA → TMP |
| ACI data | 1 1 0 0 | 1 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 | B2 → TMP |
| SUB r | 1 0 0 1 | 0 S S S | | | | (SSS)→TMP (A)→ACT | | [9] | (ACT)−(TMP)→A | |
| SUB M | 1 0 0 1 | 0 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | | DATA → TMP |
| SUI data | 1 1 0 1 | 0 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 | B2 → TMP |
| SBB r | 1 0 0 1 | 1 S S S | | | | (SSS)→TMP (A)→ACT | | [9] | (ACT)−(TMP)−CY→A | |
| SBB M | 1 0 0 1 | 1 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | | DATA → TMP |
| SBI data | 1 1 0 1 | 1 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 | B2 → TMP |
| INR r | 0 0 D D | D 1 0 0 | | | | (DDD)→TMP (TMP) + 1→ALU | ALU→DDD | | | |
| INR M | 0 0 1 1 | 0 1 0 0 | | | | X | | HL OUT STATUS[6] | | DATA → TMP (TMP)+1 → ALU |
| DCR r | 0 0 D D | D 1 0 1 | | | | (DDD)→TMP (TMP)+1→ALU | ALU→DDD | | | |
| DCR M | 0 0 1 1 | 0 1 0 1 | | | | X | | HL OUT STATUS[8] | | DATA → TMP (TMP)−1 → ALU |
| INX rp | 0 0 R P | 0 0 1 1 | | | | (RP) + 1 → | RP | | | |
| DCX rp | 0 0 R P | 1 0 1 1 | | | | (RP) − 1 → | RP | | | |
| DAD rp[8] | 0 0 R P | 1 0 0 1 | | | | X | | (ri)→ACT | (L)→TMP, (ACT)+(TMP)→ALU | ALU→L, CY |
| DAA | 0 0 1 0 | 0 1 1 1 | | | | DAA→A, FLAGS[10] | | | | |
| ANA r | 1 0 1 0 | 0 S S S | | | | (SSS)→TMP (A)→ACT | | [9] | (ACT)+(TMP)→A | |
| ANA M | 1 0 1 0 | 0 1 1 0 | PC OUT STATUS | PC = PC + 1 | INST→TMP/IR | (A)→ACT | | HL OUT STATUS[6] | | DATA → TMP |

| M3 | | | M4 | | | M5 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| T1 | T2[2] | T3 | T1 | T2[2] | T3 | T1 | T2[2] | T3 | T4 | T5 |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| HL OUT STATUS[7] | (TMP) → | DATA BUS | | | | | | | | |
| PC OUT STATUS[6] | PC = PC + 1 B3 → | rh | | | | | | | | |
| ↑ | PC = PC + 1 B3 → | W | WZ OUT STATUS[6] | DATA → | A | | | | | |
| │ | PC = PC + 1 B3 → | W | WZ OUT STATUS[7] | (A) → | DATA BUS | | | | | |
| ↓ | PC = PC + 1 B3 → | W | WZ OUT STATUS[6] | DATA → WZ = WZ + 1 | L | WZ OUT STATUS[6] | DATA → | H | | |
| PC OUT STATUS[6] | PC = PC + 1 B3 → | W | WZ OUT STATUS[7] | (L) → WZ = WZ + 1 | DATA BUS | WZ OUT STATUS[7] | (H) → | DATA BUS | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | |
| | | | | | | | | | | |
| [9] | (ACT)+(TMP)+CY→A | | | | | | | | | |
| [9] | (ACT)+(TMP)+CY→A | | | | | | | | | |
| | | | | | | | | | | |
| [9] | (ACT)-(TMP)→A | | | | | | | | | |
| [9] | (ACT)-(TMP)→A | | | | | | | | | |
| | | | | | | | | | | |
| [9] | (ACT)-(TMP)-CY→A | | | | | | | | | |
| [9] | (ACT)-(TMP)-CY→A | | | | | | | | | |
| | | | | | | | | | | |
| HL OUT STATUS[7] | ALU → | DATA BUS | | | | | | | | |
| | | | | | | | | | | |
| HL OUT STATUS[7] | ALU → | DATA BUS | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| (rh)→ACT | (H)→TMP (ACT)+(TMP)+CY→ALU | ALU→H, CY | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | |

| MNEMONIC | OP CODE | | M1[1] | | | | | M2 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $D_7 D_6 D_5 D_4$ | $D_3 D_2 D_1 D_0$ | T1 | T2[2] | T3 | T4 | T5 | T1 | T2[2] | T3 |
| ANI data | 1 1 1 0 | 0 1 1 0 | PC OUT STATUS | PC = PC + 1 | INST→TMP/IR | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 B2 | →TMP |
| XRA r | 1 0 1 0 | 1 S S S | ↓ | ↓ | ↓ | (A)→ACT (SSS)→TMP | | [9] | (ACT)+(TMP)→A | |
| XRA M | 1 0 1 0 | 1 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | DATA | →TMP |
| XRI data | 1 1 1 0 | 1 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 B2 | →TMP |
| ORA r | 1 0 1 1 | 0 S S S | | | | (A)→ACT (SSS)→TMP | | [9] | (ACT)+(TMP)→A | |
| ORA M | 1 0 1 1 | 0 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | DATA | →TMP |
| ORI data | 1 1 1 1 | 0 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 B2 | →TMP |
| CMP r | 1 0 1 1 | 1 S S S | | | | (A)→ACT (SSS)→TMP | | [9] | (ACT)−(TMP), FLAGS | |
| CMP M | 1 0 1 1 | 1 1 1 0 | | | | (A)→ACT | | HL OUT STATUS[6] | DATA | →TMP |
| CPI data | 1 1 1 1 | 1 1 1 0 | | | | (A)→ACT | | PC OUT STATUS[6] | PC = PC + 1 B2 | →TMP |
| RLC | 0 0 0 0 | 0 1 1 1 | | | | (A)→ALU ROTATE | | [9] | ALU→A, CY | |
| RRC | 0 0 0 0 | 1 1 1 1 | | | | (A)→ALU ROTATE | | [9] | ALU→A, CY | |
| RAL | 0 0 0 1 | 0 1 1 1 | | | | (A), CY→ALU ROTATE | | [9] | ALU→A, CY | |
| RAR | 0 0 0 1 | 1 1 1 1 | | | | (A), CY→ALU ROTATE | | [9] | ALU→A, CY | |
| CMA | 0 0 1 0 | 1 1 1 1 | | | | $\overline{(A)}$→A | | | | |
| CMC | 0 0 1 1 | 1 1 1 1 | | | | $\overline{CY}$→CY | | | | |
| STC | 0 0 1 1 | 0 1 1 1 | | | | 1→CY | | | | |
| JMP addr | 1 1 0 0 | 0 0 1 1 | | | | X | | PC OUT STATUS[6] | PC = PC + 1 B2 | →Z |
| J cond addr[17] | 1 1 C C | C 0 1 0 | | | | JUDGE CONDITION | | PC OUT STATUS[6] | PC = PC + 1 B2 | →Z |
| CALL addr | 1 1 0 0 | 1 1 0 1 | | | | SP = SP − 1 | | PC OUT STATUS[6] | PC = PC + 1 B2 | →Z |
| C cond addr[17] | 1 1 C C | C 1 0 0 | | | | JUDGE CONDITION IF TRUE, SP = SP − 1 | | PC OUT STATUS[6] | PC = PC + 1 B2 | →Z |
| RET | 1 1 0 0 | 1 0 0 1 | | | ↓ | X | | SP OUT STATUS[15] | SP = SP + 1 DATA | →Z |
| R cond addr[17] | 1 1 C C | C 0 0 0 | | | INST→TMP/IR | JUDGE CONDITION[14] | | SP OUT STATUS[15] | SP = SP + 1 DATA | →Z |
| RST n | 1 1 N N | N 1 1 1 | | | φ→W INST→TMP/IR | SP = SP − 1 | | SP OUT STATUS[16] | SP = SP − 1 (PCH) | →DATA BUS |
| PCHL | 1 1 1 0 | 1 0 0 1 | | | INST→TMP/IR | (HL) → PC | | | | |
| PUSH rp | 1 1 R P | 0 1 0 1 | | | ↓ | SP = SP − 1 | | SP OUT STATUS[16] | SP = SP − 1 (rh) | →DATA BUS |
| PUSH PSW | 1 1 1 1 | 0 1 0 1 | | | | SP = SP − 1 | | SP OUT STATUS[16] | SP = SP − 1 (A) | →DATA BUS |
| POP rp | 1 1 R P | 0 0 0 1 | | | | X | | SP OUT STATUS[15] | SP = SP + 1 DATA | →rl |
| POP PSW | 1 1 1 1 | 0 0 0 1 | | | | X | | SP OUT STATUS[15] | SP = SP + 1 DATA | →FLAGS |
| XTHL | 1 1 1 0 | 0 0 1 1 | | | | X | | SP OUT STATUS[15] | SP = SP + 1 DATA | →Z |
| IN port | 1 1 0 1 | 1 0 1 1 | | | | X | | PC OUT STATUS[6] | PC = PC + 1 B2 | →Z, W |
| OUT port | 1 1 0 1 | 0 0 1 1 | | | | X | | PC OUT STATUS[6] | PC = PC + 1 B2 | →Z, W |
| EI | 1 1 1 1 | 1 0 1 1 | | | | SET INTE F/F | | | | |
| DI | 1 1 1 1 | 0 0 1 1 | | | | RESET INTE F/F | | | | |
| HLT | 0 1 1 1 | 0 1 1 0 | ↓ | ↓ | ↓ | X | | PC OUT STATUS | HALT MODE[20] | |
| NOP | 0 0 0 0 | 0 0 0 0 | PC OUT STATUS | PC = PC + 1 | INST→TMP/IR | X | | | | |

| M3 | | | M4 | | | M5 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T1 | T2[2] | T3 | T1 | T2[2] | T3 | T1 | T2[2] | T3 | T4 | T5 | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | | | |
| | | | | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | | | |
| | | | | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | | | |
| [9] | (ACT)+(TMP)→A | | | | | | | | | | | |
| | | | | | | | | | | | | |
| [9] | (ACT)-(TMP); FLAGS | | | | | | | | | | | |
| [9] | (ACT)-(TMP); FLAGS | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| PC OUT STATUS[6] | PC = PC + 1 B3 → W | | | | | | | | | | WZ OUT STATUS[11] | (WZ) + 1 → PC |
| PC OUT STATUS[6] | PC = PC + 1 B3 → W | | | | | | | | | | WZ OUT STATUS[11,12] | (WZ) + 1 → PC |
| PC OUT STATUS[6] | PC = PC + 1 B3 → W | | SP OUT STATUS[16] | (PCH) → DATA BUS SP = SP - 1 | | SP OUT STATUS[16] | (PCL) → DATA BUS | | | | WZ OUT STATUS[11] | (WZ) + 1 → PC |
| PC OUT STATUS[6] | PC = PC + 1 B3 → W[13] | | SP OUT STATUS[16] | (PCH) → DATA BUS SP = SP - 1 | | SP OUT STATUS[16] | (PCL) → DATA BUS | | | | WZ OUT STATUS[11,12] | (WZ) + 1 → PC |
| SP OUT STATUS[15] | SP = SP + 1 DATA → W | | | | | | | | | | WZ OUT STATUS[11] | (WZ) + 1 → PC |
| SP OUT STATUS[15] | SP = SP + 1 DATA → W | | | | | | | | | | WZ OUT STATUS[11,12] | (WZ) + 1 → PC |
| SP OUT STATUS[16] | (TMP = 00NNN000) → Z (PCL) → LATA BUS | | | | | | | | | | WZ OUT STATUS[11] | (WZ) + 1 → PC |
| | | | | | | | | | | | | |
| SP OUT STATUS[16] | (rl) → DATA BUS | | | | | | | | | | | |
| SP OUT STATUS[16] | FLAGS → DATA BUS | | | | | | | | | | | |
| SP OUT STATUS[15] | SP = SP + 1 DATA → rh | | | | | | | | | | | |
| SP OUT STATUS[15] | SP = SP + 1 DATA → A | | | | | | | | | | | |
| SP OUT STATUS[15] | DATA → W | | SP OUT STATUS[16] | (H) → | DATA BUS | SP OUT STATUS[16] | (L) → | DATA BUS | (WZ) → HL | | | |
| WZ OUT STATUS[18] | DATA → A | | | | | | | | | | | |
| WZ OUT STATUS[18] | (A) → DATA BUS | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |

NOTES:

1. The first memory cycle (M1) is always an instruction fetch; the first (or only) byte, containing the op code, is fetched during this cycle.

2. If the READY input from memory is not high during T2 of each memory cycle, the processor will enter a wait state (TW) until READY is sampled as high.

3. States T4 and T5 are present, as required, for operations which are completely internal to the CPU. The contents of the internal bus during T4 and T5 are available at the data bus; this is designed for testing purposes only. An "X" denotes that the state is present, but is only used for such internal operations as instruction decoding.

4. Only register pairs rp = B (registers B and C) or rp = D (registers D and E) may be specified.

5. These states are skipped.

6. Memory read sub-cycles; an instruction or data word will be read.

7. Memory write sub-cycle.

8. The READY signal is not required during the second and third sub-cycles (M2 and M3). The HOLD signal is accepted during M2 and M3. The SYNC signal is not generated during M2 and M3. During the execution of DAD, M2 and M3 are required for an internal register-pair add; memory is not referenced.

9. The results of these arithmetic, logical or rotate instructions are not moved into the accumulator (A) until state T2 of the next instruction cycle. That is, A is loaded while the next instruction is being fetched; this overlapping of operations allows for faster processing.

10. If the value of the least significant 4-bits of the accumulator is greater than 9 or if the auxiliary carry bit is set, 6 is added to the accumulator. If the value of the most significant 4-bits of the accumulator is now greater than 9, or if the carry bit is set, 6 is added to the most significant 4-bits of the accumulator.

11. This represents the first sub-cycle (the instruction fetch) of the next instruction cycle.

12. If the condition was met, the contents of the register pair WZ are output on the address lines ($A_{0-15}$) instead of the contents of the program counter (PC).

13. If the condition was not met, sub-cycles M4 and M5 are skipped; the processor instead proceeds immediately to the instruction fetch (M1) of the next instruction cycle.

14. If the condition was not met, sub-cycles M2 and M3 are skipped; the processor instead proceeds immediately to the instruction fetch (M1) of the next instruction cycle.

15. Stack read sub-cycle.

16. Stack write sub-cycle.

17. CONDITION

| CONDITION | CCC |
|---|---|
| NZ — not zero (Z = 0) | 000 |
| Z — zero (Z = 1) | 001 |
| NC — no carry (CY = 0) | 010 |
| C — carry (CY = 1) | 011 |
| PO — parity odd (P = 0) | 100 |
| PE — parity even (P = 1) | 101 |
| P — plus (S = 0) | 110 |
| M — minus (S = 1) | 111 |

18. I/O sub-cycle: the I/O port's 8-bit select code is duplicated on address lines 0-7 ($A_{0-7}$) and 8-15 ($A_{8-15}$).

19. Output sub-cycle.

20. The processor will remain idle in the halt state until an interrupt, a reset or a hold is accepted. When a hold request is accepted, the CPU enters the hold mode; after the hold mode is terminated, the processor returns to the halt state. After a reset is accepted, the processor begins execution at memory location zero. After an interrupt is accepted, the processor executes the instruction forced onto the data bus (usually a restart instruction).

| SSS or DDD | Value | rp | Value |
|---|---|---|---|
| A | 111 | B | 00 |
| B | 000 | D | 01 |
| C | 001 | H | 10 |
| D | 010 | SP | 11 |
| E | 011 | | |
| H | 100 | | |
| L | 101 | | |

# CHAPTER 3
# INTERFACING THE 8080

This chapter will illustrate, in detail, how to interface the 8080 CPU with Memory and I/O. It will also show the benefits and tradeoffs encountered when using a variety of system architectures to achieve higher throughput, decreased component count or minimization of memory size.

8080 Microcomputer system design lends itself to a simple, modular approach. Such an approach will yield the designer a reliable, high performance system that contains a minimum component count and is easy to manufacture and maintain.

The overall system can be thought of as a simple block diagram. The three (3) blocks in the diagram represent the functions common to any computer system.

**CPU Module*** Contains the Central Processing Unit, system timing and interface circuitry to Memory and I/O devices.

**Memory** Contains Read Only Memory (ROM) and Read/Write Memory (RAM) for program and data storage.

**I/O** Contains circuitry that allows the computer system to communicate with devices or structures existing outside of the CPU or Memory array.

for example: Keyboards, Floppy Disks, Paper Tape, etc.

There are three busses that interconnect these blocks:

**Data Bus†** A bi-directional path on which data can flow between the CPU and Memory or I/O.

**Address Bus** A uni-directional group of lines that identify a particular Memory location or I/O device.

*"Module" refers to a functional block, it does not reference a printed circuit board manufactured by INTEL.

†"Bus" refers to a set of signals grouped together because of the similarity of their functions.

**Control Bus** A uni-directional set of signals that indicate the type of activity in current process.

Type of activities:
1. Memory Read
2. Memory Write
3. I/O Read
4. I/O Write
5. Interrupt Acknowledge

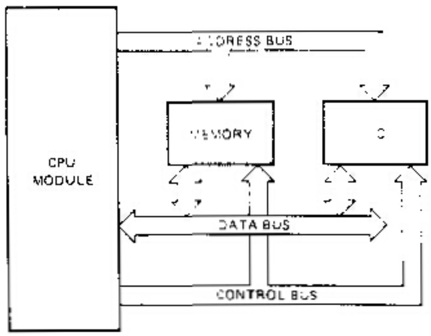



**Figure 3-1. Typical Computer System Block Diagram**

## Basic System Operation

1. The CPU Module issues an activity command on the Control Bus.
2. The CPU Module issues a binary code on the Address Bus to identify which particular Memory location or I/O device will be involved in the current process activity.
3. The CPU Module receives or transmits data with the selected Memory location or I/O device.
4. The CPU Module returns to ① and issues the next activity command.

It is easy to see at this point that the CPU module is the central element in any computer system.

The following pages will cover the detailed design of the CPU Module with the 8080. The three Busses (Data, Address and Control) will be developed and the interconnection to Memory and I/O will be shown.

Design philosophies and system architectures presented in this manual are consistent with product development programs underway at *INTEL for the MCS™-80. Thus,* the designer who uses this manual as a guide for his total system engineering is assured that all new developments in components and software for MCS-80 from INTEL will be compatible with his design approach.

## CPU Module Design

The CPU Module contains three major areas:

1. The 8080 Central Processing Unit
2. A Clock Generator and High Level Driver
3. A bi-directional Data Bus Driver and System Control Logic

The following will discuss the design of the three major areas contained in the CPU Module. This design is presented as an alternative to the Intel® 8224 Clock Generator and Intel 8228 System Controller. By studying the alternative approach, the designer can more clearly see the considerations involved in the specification and engineering of the 8224 and 8228. Standard TTL components and Intel general purpose peripheral devices are used to implement the design and to achieve operational characteristics that are as close as possible to those of the 8224 and 8228. Many auxiliary timing functions and features of the 8224 and 8228 are too complex to practically implement in standard components, so only the basic functions of the 8224 and 8228 are generated. Since significant benefits in system timing and component count reduction can be realized by using the 8224 and 8228, this is the preferred method of implementation.

**1. 8080 CPU**

The operation of the 8080 CPU was covered in previous chapters of this manual, so little reference will be made to it in the design of the Module.

**2. Clock Generator and High Level Driver**

The 8080 is a dynamic device, meaning that its internal storage elements and logic circuitry require a timing reference (Clock), supplied by external circuitry, to refresh and provide timing control signals.

The 8080 requires two (2) such Clocks. Their waveforms must be non-overlapping, and comply with the timing and levels specified in the 8080 A.C. and D.C. Characteristics, page 5-15.

**Clock Generator Design**

The Clock Generator consists of a crystal controlled,

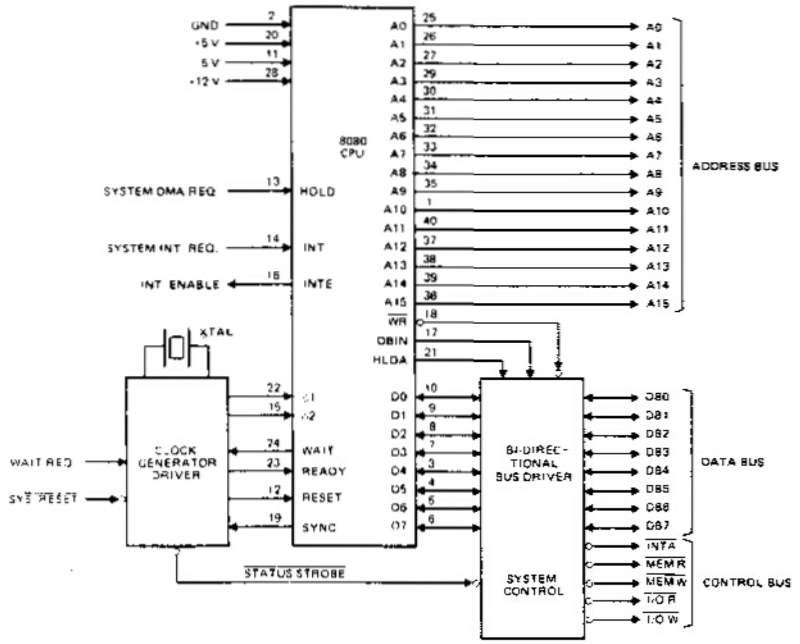



**Figure 3-2. 8080 CPU Interface**

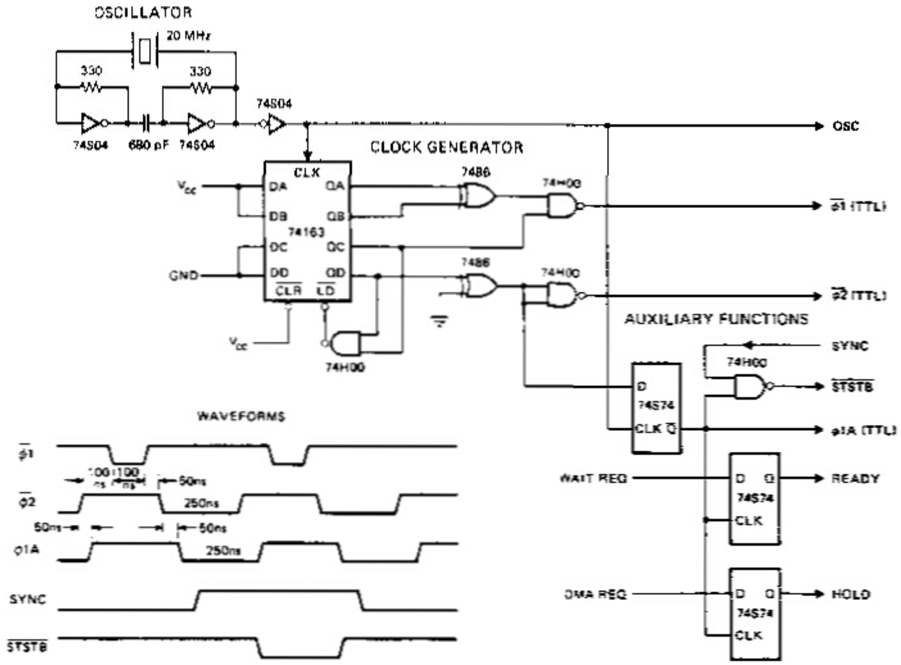



**Figure 3-3. 8080 Clock Generator**

20 MHZ oscillator, a four bit counter, and gating circuits.

The oscillator provides a 20 MHZ signal to the input of a four (4) bit, presettable, synchronous, binary counter. By presetting the counter as shown in figure 3-3 and clocking it with the 20 MHZ signal, a simple decoding of the counters outputs using standard TTL gates, provides proper timing for the two (2) 8080 clock inputs.

Note that the timing must actually be measured at the output of the High Level Driver to take into account the added delays and waveform distortions within such a device.

### High Level Driver Design

The voltage level of the clocks for the 8080 is not TTL compatible like the other signals that input to the 8080. The voltage swing is from .6 volts ($V_{ILC}$) to 11 volts ($V_{IHC}$) with risetimes and falltimes under 50 ns. The Capacitive Drive is 20 pf (max.). Thus, a High Level Driver is required to interface the outputs of the Clock Generator (TTL) to the 8080.

The two (2) outputs of the Clock Generator are capacitivity coupled to a dual- High Level clock driver. The driver must be capable of complying with the 8080 clock input specifications, page 5-15. A driver of this type usually has little problem supplying the positive transition when biased from the 8080 $V_{DD}$ supply (12V) but to achieve the low voltage specification ($V_{ILC}$) .8 volts Max. the driver is biased to the 8080 $V_{BB}$ supply (-5V). This allows the driver to swing from GND to $V_{DD}$ with the aid of a simple resistor divider.

A low resistance series network is added between the driver and the 8080 to eliminate any overshoot of the pulsed waveforms. Now a circuit is apparent that can easily comply with the 8080 specifications. In fact rise and falltimes of this design are typically less than 10 ns.

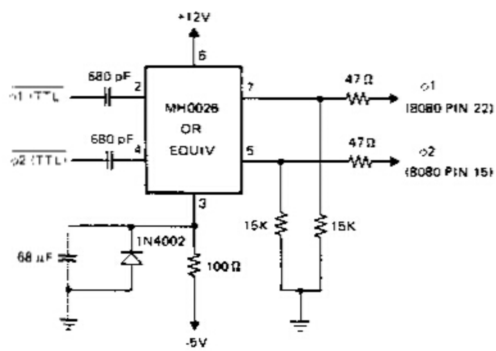



**Figure 3-4. High Level Driver**

### Auxiliary Timing Signals and Functions

The Clock Generator can also be used to provide other signals that the designer can use to simplify large system timing or the interface to dynamic memories.

Functions such as power-on reset, synchronization of external requests (HOLD, READY, etc.) and single step, could easily be added to the Clock Generator to further enhance its capabilities.

For instance, the 20 MHZ signal from the oscillator can be buffered so that it could provide the basis for communication baud rate generation.

The Clock Generator diagram also shows how to generate an advanced timing signal ($\phi$1A) that is handy to use in clocking "D" type flipflops to synchronize external requests. It can also be used to generate a strobe ($\overline{STSTB}$) that is the latching signal for the status information which is available on the Data Bus at the beginning of each machine cycle. A simple gating of the SYNC signal from the 8080 and the advanced ($\phi$1A) will do the job. See Figure 3-3.

### 3. Bi-Directional Bus Driver and System Control Logic

The system Memory and I/O devices communicate with the CPU over the bi-directional Data Bus. The system Control Bus is used to gate data on and off the Data Bus within the proper timing sequences as dictated by the operation of the 8080 CPU. The data lines of the 8080 CPU, Memory and I/O devices are 3-state in nature, that is, their output drivers have the ability to be forced into a high-impedance mode and are, effectively, removed from the circuit. This 3-state bus technique allows the designer to construct a system around a single, eight (8) bit parallel, bi-directional Data Bus and simply gate the information on or off this bus by selecting or deselecting (3-stating) Memory and I/O devices with signals from the Control Bus.

#### Bi-Directional Data Bus Driver Design

The 8080 Data Bus (D7-D0) has two (2) major areas of concern for the designer:

1. Input Voltage level ($V_{IH}$) 3.3 volts minimum.
2. Output Drive Capability ($I_{OL}$) 1.7 mA maximum.

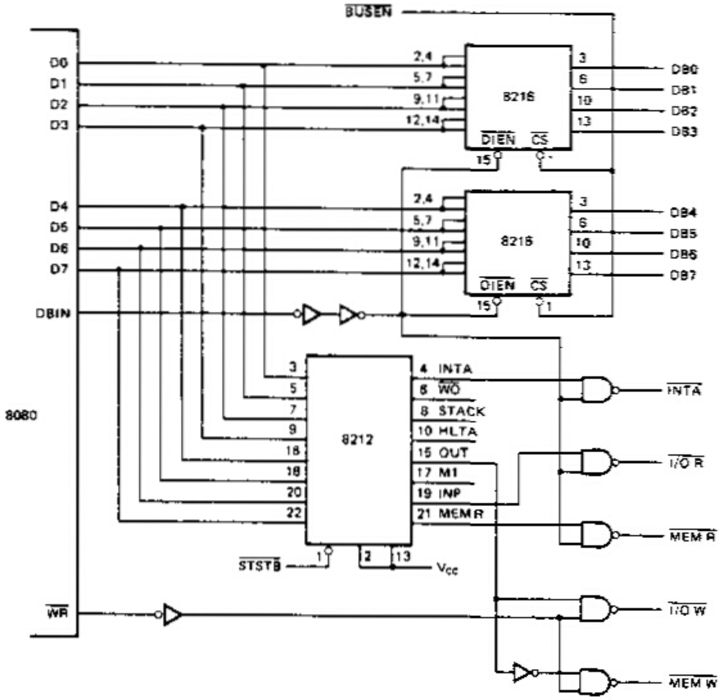



Figure 3-5. 8080 System Control

The input level specification implies that any semiconductor memory or I/O device connected to the 8080 Data Bus must be able to provide a minimum of 3.3 volts in its high state. Most semiconductor memories and standard TTL I/O devices have an output capability of between 2.0 and 2.8 volts, obviously a direct connection onto the 8080 Data Bus would require pullup resistors, whose value should not affect the bus speed or stress the drive capability of the memory or I/O components.

The 8080A output drive capability ($I_{OL}$) 1.9mA max. is sufficient for small systems where Memory size and I/O requirements are minimal and the entire system is contained on a single printed circuit board. Most systems however, take advantage of the high-performance computing power of the 8080 CPU and thus a more typical system would require some form of buffering on the 8080 Data Bus to support a larger array of Memory and I/O devices which are likely to be on separate boards.

A device specifically designed to do this buffering function is the INTEL® 8216, a (4) four bit bi-directional bus driver whose input voltage level is compatible with standard TTL devices and semiconductor memory components, and has output drive capability of 50 mA. At the 8080 side, the 8216 has a "high" output of 3.65 volts that not only meets the 8080 input spec but provides the designer with a worse case 350 mV noise margin.

A pair of 8216's are connected directly to the 8080 Data Bus (D7-D0) as shown in figure 3-5. Note that the DBIN signal from the 8080 is connected to the direction control input ($\overline{DIEN}$) so the correct flow of data on the bus is maintained. The chip select ($\overline{CS}$) of the 8216 is connected to BUS ENABLE ($\overline{BUSEN}$) to allow for DMA activities by deselecting the Data Bus Buffer and forcing the outputs of the 8216's into their high impedance (3-state) mode. This allows other devices to gain access to the data bus (DMA).

**System Control Logic Design**

The Control Bus maintains discipline of the bi-directional Data Bus, that is, it determines what type of device will have access to the bus (Memory or I/O) and generates signals to assure that these devices transfer Data with the 8080 CPU within the proper timing "windows" as dictated by the CPU operational characteristics.

As described previously, the 8080 issues Status information at the beginning of each Machine Cycle on its Data Bus to indicate what operation will take place during that cycle. A simple (8) bit latch, like an INTEL® 8212, connected directly to the 8080 Data Bus (D7-D0) as shown in figure 3-5 will store the Status information. The signal that loads the data into the Status Latch comes from the Clock Generator, it is Status Strobe ($\overline{STSTB}$) and occurs at the start of each Machine Cycle.

Note that the Status Latch is connected onto the 8080 Data Bus (D7-D0) before the Bus Buffer. This is to maintain the integrity of the Data Bus and simplify Control Bus timing in DMA dependent environments.

As shown in the diagram, a simple gating of the outputs of the Status Latch with the DBIN and $\overline{WR}$ signals from the 8080 generate the (4) four Control signals that make up the basic Control Bus.

These four signals: 1. Memory Read ($\overline{MEM\ R}$)

2. Memory Write ($\overline{MEM\ W}$)

3. I/O Read ($\overline{I/O\ R}$)

4. I/O Write ($\overline{I/O\ W}$)

connect directly to the MCS-80 component "family" of ROMs, RAMs and I/O devices.

A fifth signal, Interrupt Acknowledge ($\overline{INTA}$) is added to the Control Bus by gating data off the Status Latch with the DBIN signal from the 8080 CPU. This signal is used to enable the Interrupt Instruction Port which holds the RST instruction onto the Data Bus.

Other signals that are part of the Control Bus such as $\overline{WO}$, Stack and M1 are present to aid in the testing of the System and also to simplify interfacing the CPU to dynamic memories or very large systems that require several levels of bus buffering.

**Address Buffer Design**

The Address Bus (A15-A0) of the 8080, like the Data Bus, is sufficient to support a small system that has a moderate size Memory and I/O structure, confined to a single card. To expand the size of the system that the Address Bus can support a simple buffer can be added, as shown in figure 3-6. The INTEL® 8212 or 8216 is an excellent device for this function. They provide low input loading (.25 mA), high output drive and insert a minimal delay in the System Timing.

Note that BUS ENABLE ($\overline{BUSEN}$) is connected to the buffers so that they are forced into their high-impedance (3-state) mode during DMA activities so that other devices can gain access to the Address Bus.

## INTERFACING THE 8080 CPU TO MEMORY AND I/O DEVICES

The 8080 interfaces with standard semiconductor Memory components and I/O devices. In the previous text the proper control signals and buffering were developed which will produce a simple bus system similar to the basic system example shown at the beginning of this chapter.

In Figure 3-6 a simple, but exact 8080 typical system is shown that can be used as a guide for any 8080 system, regardless of size or complexity. It is a "three bus" architecture, using the signals developed in the CPU module.

Note that Memory and I/O devices interface in the same manner and that their isolation is only a function of the definition of the Read-Write signals on the Control Bus. This allows the 8080 system to be configured so that Memory and I/O are treated as a single array (memory mapped I/O) for small systems that require high thruput and have less than 32K memory size. This approach will be brought out later in the chapter.

## ROM INTERFACE

A ROM is a device that stores data in the form of Program or other information such as "look-up tables" and is only read from, thus the term Read Only Memory. This type of memory is generally non-volatile, meaning that when the power is removed the information is retained. This feature eliminates the need for extra equipment like tape readers and disks to load programs initially, an important aspect in small system design.

Interfacing standard ROMs, such as the devices shown in the diagram is simple and direct. The output Data lines are connected to the bi-directional Data Bus, the Address inputs tie to the Address bus with possible decoding of the most significant bits as "chip selects" and the $\overline{\text{MEMR}}$ signal from the Control Bus connected to a "chip select" or data buffer. Basically, the CPU issues an address during the first portion of an instruction or data fetch (T1 & T2). This value on the Address Bus selects a specific location within the ROM, then depending on the ROM's delay (access time) the data stored at the addressed location is present at the Data output lines. At this time (T3) the CPU Data Bus is in the "input Mode" and the control logic issues a Memory Read command ($\overline{\text{MEMR}}$) that gates the addressed data on to the Data Bus.

## RAM INTERFACE

A RAM is a device that stores data. This data can be program, active "look-up tables," temporary values or external stacks. The difference between RAM and ROM is that data can be written into such devices and are in essence, Read/Write storage elements. RAMs do not hold their data when power is removed so in the case where Program or "look-up tables" data is stored a method to load

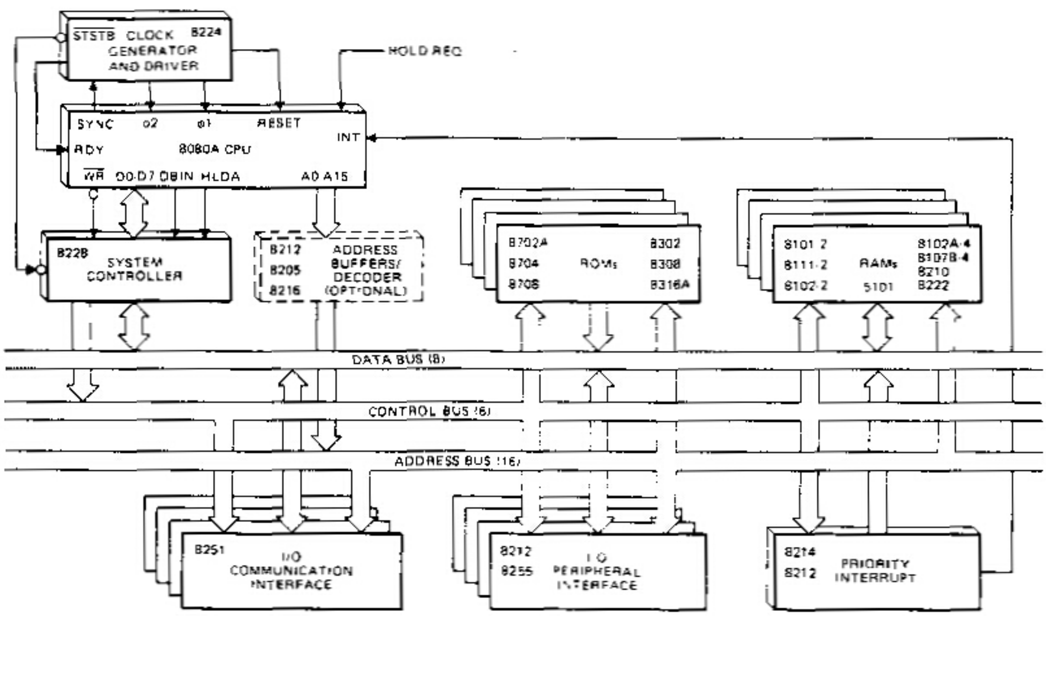



Figure 3-6. Microcomputer System

RAM memory must be provided, such as: Floppy Disk, Paper Tape, etc.

The CPU treats RAM in exactly the same manner as ROM for addressing data to be read. Writing data is very similar; the RAM is issued an address during the first portion of the Memory Write cycle (T1 & T2) in T3 when the data that is to be written is output by the CPU and is stable on the bus an $\overline{MEMW}$ command is generated. The $\overline{MEMW}$ signal is connected to the R/W input of the RAM and strobes the data into the addressed location.

In Figure 3-7 a typical Memory system is illustrated to show how standard semiconductor components interface to the 8080 bus. The memory array shown has 8K bytes (8 bits/byte) of ROM storage, using four Intel® 8216As and 512 bytes of RAM storage, using Intel 8111 static RAMs. The basic interface to the bus structure detailed here is common to almost any size memory. The only addition that might have to be made for larger systems is more buffers (8216/8212) and decoders (8205) for generating "chip selects."

The memories chosen for this example have an access time of 850 nS (max) to illustrate that slower, economical devices can be easily interfaced to the 8080 with little effect on performance. When the 8080 is operated from a clock generator with a tCY of 500 nS the required memory access time is Approx. 450-550 nS. See detailed timing specification Pg. 5-16. Using memory devices of this speed such as Intel® 8308, 8102A, 8107A, etc. the READY input to the 8080 CPU can remain "high" because no "wait" states are required. Note that the bus interface to memory shown in Figure 3-7 remains the same. However, if slower memories are to be used, such as the devices illustrated (8316A, 8111) that have access times slower than the minimum requirement a simple logic control of the READY input to the 8080 CPU will insert an extra "wait state" that is equal to one or more clock periods as an access time "adjustment" delay to compensate. The effect of the extra "wait" state is naturally a slower execution time for the instruction. A single "wait" changes the basic instruction cycle to 2.5 microSeconds.

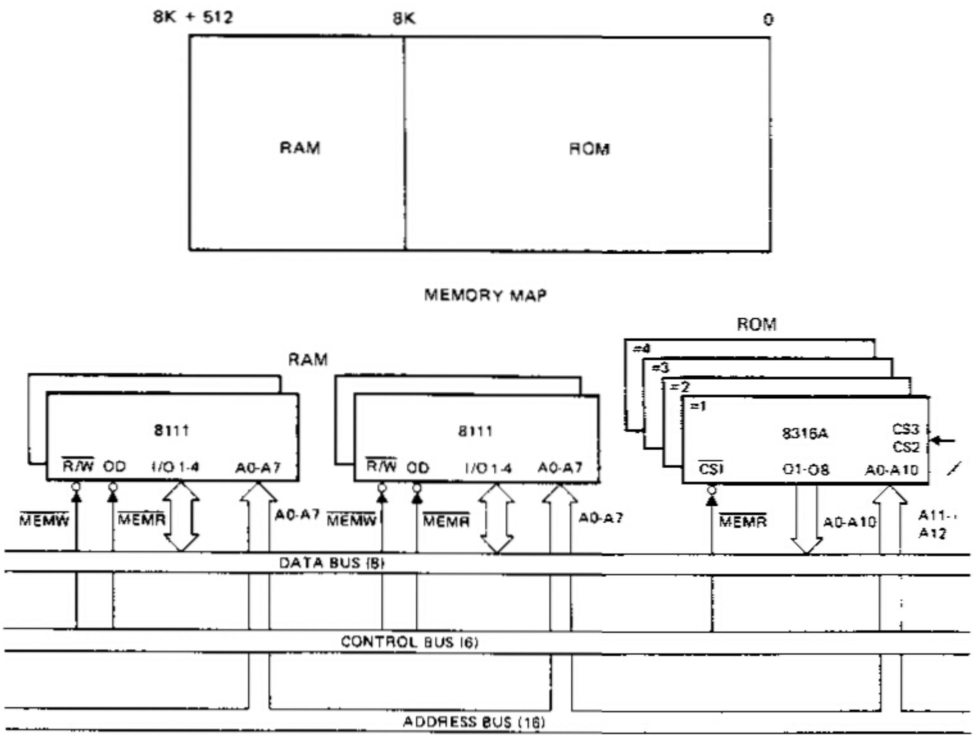



**Figure 3-7. Typical Memory Interface**

# I/O INTERFACE

## General Theory

As in any computer based system, the 8080 CPU must be able to communicate with devices or structures that exist outside its normal memory array. Devices like keyboards, paper tape, floppy disks, printers, displays and other control structures are used to input information into the 8080 CPU and display or store the results of the computational activity.

Probably the most important and strongest feature of the 8080 Microcomputer System is the flexibility and power of its I/O structure and the components that support it. There are many ways to structure the I/O array so that it will "fit" the total system environment to maximize efficiency and minimize component count.

The basic operation of the I/O structure can best be viewed as an array of single byte memory locations that can be Read from or Written into. The 8080 CPU has special instructions devoted to managing such transfers (IN, OUT). These instructions generally isolate memory and I/O arrays so that memory address space is not effected by the I/O structure and the general concept is that of a simple transfer to or from the Accumulator with an addressed "PORT". Another method of I/O architecture is to treat the I/O structure as part of the Memory array. This is generally referred to as "Memory Mapped I/O" and provides the designer with a powerful new "instruction set" devoted to I/O manipulation.

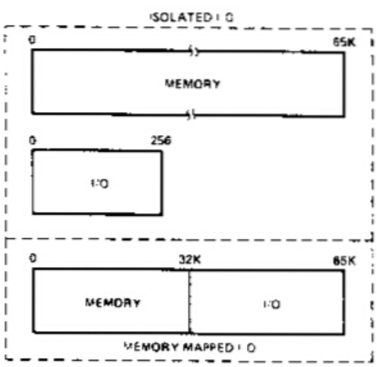



**Figure 3-8. Memory/I/O Mapping.**

## Isolated I/O

In Figure 3-9 the system control signals, previously detailed in this chapter, are shown. This type of I/O architecture separates the memory address space from the I/O address space and uses a conceptually simple transfer to or from Accumulator technique. Such an architecture is easy to understand because I/O communicates only with the Accumulator using the IN or OUT instructions. Also because of the isolation of memory and I/O, the full address space (65K) is uneffected by I/O addressing.

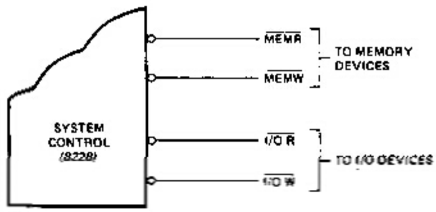



**Figure 3-9. Isolated I/O.**

## Memory Mapped I/O

By assigning an area of memory address space as I/O a powerful architecture can be developed that can manipulate I/O using the same instructions that are used to manipulate memory locations. Thus, a "new" instruction set is created that is devoted to I/O handling.

As shown in Figure 3-10, new control signals are generated by gating the $\overline{MEMR}$ and $\overline{MEMW}$ signals with $A_{15}$, the most significant address bit. The new I/O control signals connect in exactly the same manner as Isolated I/O, thus the system bus characteristics are unchanged.

By assigning $A_{15}$ as the I/O "flag", a simple method of I/O discipline is maintained:

If $A_{15}$ is a "zero" then Memory is active.
If $A_{15}$ is a "one" then I/O is active.

Other address bits can also be used for this function. $A_{15}$ was chosen because it is the most significant address bit so it is easier to control with software and because it still allows memory addressing of 32K.

I/O devices are still considered addressed "ports" but instead of the Accumulator as the only transfer medium any of the internal registers can be used. All instructions that could be used to operate on memory locations can be used in I/O.

Examples:

| | |
|---|---|
| MOVr, M | (Input Port to any Register) |
| MOV M, r | (Output any Register to Port) |
| MVI M | (Output immediate data to Port) |
| LDA | (Input to ACC) |
| STA | (Output from ACC to Port) |
| LHLD | (16 Bit Input) |
| SHLD | (16 Bit Output) |
| ADD M | (Add Port to ACC) |
| ANA M | ("AND" Port with ACC) |

It is easy to see that from the list of possible "new" instructions that this type of I/O architecture could have a drastic effect on increased system throughput. It is conceptually more difficult to understand than Isolated I/O and it does limit memory address space, but Memory Mapped I/O can mean a significant increase in overall speed and at the same time reducing required program memory area.

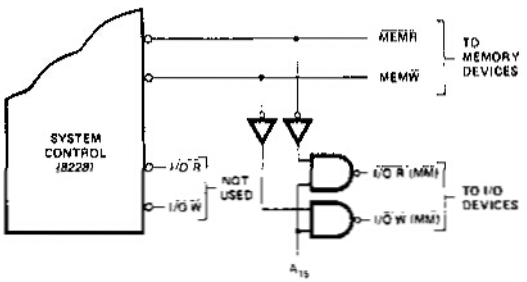



**Figure 3-10. Memory Mapped I/O.**

## I/O Addressing

With both systems of I/O structure the addressing of each device can be configured to optimize efficiency and reduce component count. One method, the most common, is to decode the address bus into exclusive "chip selects" that enable the addressed I/O device, similar to generating chip-selects in memory arrays.

Another method is called "linear select". In this method, instead of decoding the Address Bus, a singular bit from the bus is assigned as the exclusive enable for a specific I/O device. This method, of course, limits the number of I/O devices that can be addressed but eliminates the need for extra decoders, an important consideration in small system design.

A simple example illustrates the power of such a flexible I/O structure. The first example illustrates the format of the second byte of the IN or OUT instruction using the Isolated I/O technique. The devices used are Intel®8255 Programmable Peripheral Interface units and are linear selected. Each device has three ports and from the format it can be seen that six devices can be addressed without additional decoders.

**EXAMPLE #1**

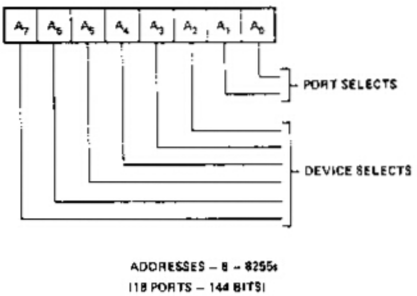



**Figure 3-11. Isolated I/O – (Linear Select) (8255)**

The second example uses Memory Mapped I/O and linear select to show how thirteen devices (8255) can be addressed without the use of extra decoders. The format shown could be the second and third bytes of the LDA or STA instructions or any other instructions used to manipulate I/O using the Memory Mapped technique.

It is easy to see that such a flexible I/O structure, that *can be "tailored" to the overall system environment, provides* the designer with a powerful tool to optimize efficiency and minimize component count.

**EXAMPLE #2**

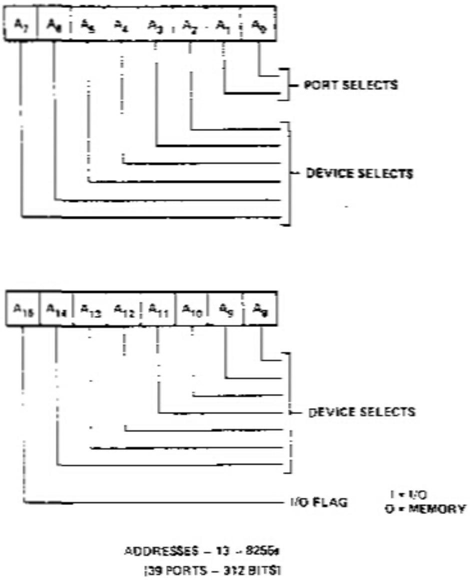



**Figure 3-12. Memory Mapped I/O – (Linear Select (8255)**

## I/O Interface Example

In Figure 3-16 a typical I/O system is shown that uses a variety of devices (8212, 8251 and 8255). It could be used to interface the peripherals around an intelligent CRT terminals; keyboards, display, and communication interface. Another application could be in a process controller to interface sensors, relays, and motor controls. The limitation of the application area for such a circuit is solely that of the designers imagination.

The I/O structure shown interfaces to the 8080 CPU using the bus architecture developed previously in this chapter. Either Isolated or Memory Mapped techniques can be used, depending on the system I/O environment.

The 8251 provides a serial data communication interface so that the system can transmit and receive data over communication links such as telephone lines.

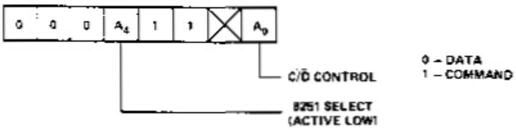



**Figure 3-13. 8251 Format.**

The two (2) 8255s provide twenty four bits each of programmable I/O data and control so that keyboards, sensors, paper tape, etc., can be interfaced to the system.

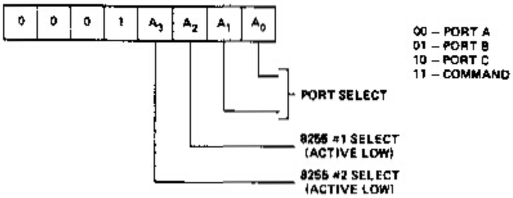



**Figure 3-14. 8255 Format.**

The three 8212s can be used to drive long lines or LED indicators due to their high drive capability. (15mA)

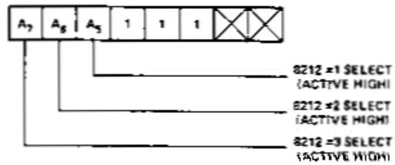



**Figure 3-15. 8212 Format.**

Addressing the structure is described in the formats illustrated in Figures 3-13, 3-14, 3-15. Linear Select is used so that no decoders are required thus, each device has an exclusive "enable bit".

The example shows how a powerful yet flexible I/O structure can be created using a minimum component count with devices that are all members of the 8080 Microcomputer System.

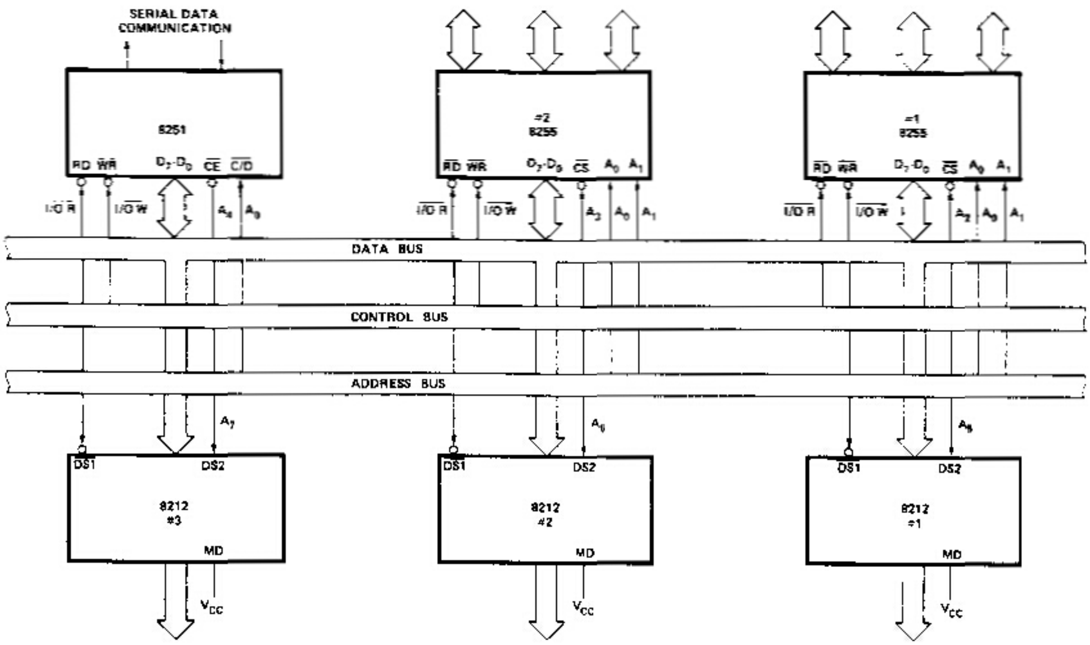



**Figure 3-16. Typical I/O Interface.**

# CHAPTER 4
# INSTRUCTION SET

A computer, no matter how sophisticated, can only do what it is "told" to do. One "tells" the computer what to do via a series of coded instructions referred to as a **Program**. The realm of the programmer is referred to as **Software**, in contrast to the **Hardware** that comprises the actual computer equipment. A computer's software refers to all of the programs that have been written for that computer.

When a computer is designed, the engineers provide the Central Processing Unit (CPU) with the ability to perform a particular set of operations. The CPU is designed such that a specific operation is performed when the CPU control logic decodes a particular instruction. Consequently, the operations that can be performed by a CPU define the computer's **Instruction Set**.

Each computer instruction allows the programmer to initiate the performance of a specific operation. All computers implement certain arithmetic operations in their instruction set, such as an instruction to add the contents of two registers. Often logical operations (e.g., OR the contents of two registers) and register operate instructions (e.g., increment a register) are included in the instruction set. A computer's instruction set will also have instructions that move data between registers, between a register and memory, and between a register and an I/O device. Most instruction sets also provide **Conditional Instructions.** A conditional instruction specifies an operation to be performed only if certain conditions have been met; for example, jump to a particular instruction if the result of the last operation was zero. Conditional instructions provide a program with a decision-making capability.

By logically organizing a sequence of instructions into a coherent program, the programmer can "tell" the computer to perform a very specific and useful function.

The computer, however, can only execute programs whose instructions are in a binary coded form (i.e., a series of 1's and 0's), that is called **Machine Code**. Because it would be extremely cumbersome to program in machine code, programming languages have been developed. There are programs available which convert the programming language instructions into machine code that can be interpreted by the processor.

One type of programming language is **Assembly Language.** A unique assembly language mnemonic is assigned to each of the computer's instructions. The programmer can write a program (called the **Source Program**) using these mnemonics and certain operands; the source program is then converted into machine instructions (called the **Object Code**). Each assembly language instruction is converted into one machine code instruction (1 or more bytes) by an **Assembler** program. Assembly languages are usually machine dependent (i.e., they are usually able to run on only one type of computer).

## THE 8080 INSTRUCTION SET

The 8080 instruction set includes five different types of instructions:

- **Data Transfer Group**—move data between registers or between memory and registers
- **Arithmetic Group** — add, subtract, increment or decrement data in registers or in memory
- **Logical Group** — AND, OR, EXCLUSIVE-OR, compare, rotate or complement data in registers or in memory
- **Branch Group** — conditional and unconditional jump instructions, subroutine call instructions and return instructions
- **Stack, I/O and Machine Control Group** — includes I/O instructions, as well as instructions for maintaining the stack and internal control flags.

### Instruction and Data Formats:

Memory for the 8080 is organized into 8-bit quantities, called Bytes. Each byte has a unique 16-bit binary address corresponding to its sequential position in memory.

The 8080 can directly address up to 65,536 bytes of memory, which may consist of both read-only memory (ROM) elements and random-access memory (RAM) elements (read/write memory).

Data in the 8080 is stored in the form of 8-bit binary integers:

DATA WORD

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| MSB | | | | | | | LSB |

When a register or data word contains a binary number, it is necessary to establish the order in which the bits of the number are written. In the Intel 8080, BIT 0 is referred to as the **Least Significant Bit (LSB)**, and BIT 7 (of an 8 bit number) is referred to as the **Most Significant Bit (MSB)**.

The 8080 program instructions may be one, two or three bytes in length. Multiple byte instructions must be stored in successive memory locations; the address of the first byte is always used as the address of the instructions. The exact instruction format will depend on the particular operation to be executed.

Single Byte Instructions

| | | | |
|---|---|---|---|
| | $D_7$ | $D_0$ | Op Code |

Two-Byte Instructions

| | | | |
|---|---|---|---|
| Byte One | $D_7$ | $D_0$ | Op Code |
| Byte Two | $D_7$ | $D_0$ | Data or Address |

Three-Byte Instructions

| | | | |
|---|---|---|---|
| Byte One | $D_7$ | $D_0$ | Op Code |
| Byte Two | $D_7$ | $D_0$ | Data or Address |
| Byte Three | $D_7$ | $D_0$ | |

## Addressing Modes:

Often the data that is to be operated on is stored in memory. When multi-byte numeric data is used, the data, like instructions, is stored in successive memory locations, with the least significant byte first, followed by increasingly significant bytes. The 8080 has four different modes for addressing data stored in memory or in registers:

- Direct – Bytes 2 and 3 of the instruction contain the exact memory address of the data item (the low-order bits of the address are in byte 2, the high-order bits in byte 3).
- Register – The instruction specifies the register or register-pair in which the data is located.
- Register Indirect – The instruction specifies a register-pair which contains the memory address where the data is located (the high-order bits of the address are in the first register of the pair, the low-order bits in the second).
- Immediate – The instruction contains the data itself. This is either an 8-bit quantity or a 16-bit quantity (least significant byte first, most significant byte second).

Unless directed by an interrupt or branch instruction, the execution of instructions proceeds through consecutively increasing memory locations. A branch instruction can specify the address of the next instruction to be executed in one of two ways:

- Direct – The branch instruction contains the address of the next instruction to be executed. (Except for the 'RST' instruction, byte 2 contains the low-order address and byte 3 the high-order address.)
- Register indirect – The branch instruction indicates a register-pair which contains the address of the next instruction to be executed. (The high-order bits of the address are in the first register of the pair, the low-order bits in the second.)

The RST instruction is a special one-byte call instruction (usually used during interrupt sequences). RST includes a three-bit field; program control is transferred to the instruction whose address is eight times the contents of this three-bit field.

## Condition Flags:

There are five condition flags associated with the execution of instructions on the 8080. They are Zero, Sign, Parity, Carry, and Auxiliary Carry, and are each represented by a 1-bit register in the CPU. A flag is "set" by forcing the bit to 1; "reset" by forcing the bit to 0.

Unless indicated otherwise, when an instruction affects a flag, it affects it in the following manner:

Zero: If the result of an instruction has the value 0, this flag is set; otherwise it is reset.

Sign: If the most significant bit of the result of the operation has the value 1, this flag is set; otherwise it is reset.

Parity: If the modulo 2 sum of the bits of the result of the operation is 0, (i.e., if the result has even parity), this flag is set; otherwise it is reset (i.e., if the result has odd parity).

Carry: If the instruction resulted in a carry (from addition), or a borrow (from subtraction or a comparison) out of the high-order bit, this flag is set; otherwise it is reset.

Auxiliary Carry: If the instruction caused a carry out of bit 3 and into bit 4 of the resulting value, the auxiliary carry is set; otherwise it is reset. This flag is affected by single precision additions, subtractions, increments, decrements, comparisons, and logical operations, but is principally used *with additions and increments preceding* a DAA (Decimal Adjust Accumulator) instruction.

## Symbols and Abbreviations:

The following symbols and abbreviations are used in the subsequent description of the 8080 instructions:

| SYMBOLS | MEANING |
|---|---|
| accumulator | Register A |
| addr | 16-bit address quantity |
| data | 8-bit data quantity |
| data 16 | 16-bit data quantity |
| byte 2 | The second byte of the instruction |
| byte 3 | The third byte of the instruction |
| port | 8-bit address of an I/O device |
| r,r1,r2 | One of the registers A,B,C,D,E,H,L |
| 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 |

rp — One of the register pairs:

B represents the B,C pair with B as the high-order register and C as the low-order register;

D represents the D,E pair with D as the high-order register and E as the low-order register;

H represents the H,L pair with H as the high-order register and L as the low-order register;

SP represents the 16-bit stack pointer register.

RP — The bit pattern designating one of the register pairs B,D,H,SP:

| RP | REGISTER PAIR |
|---|---|
| 00 | B-C |
| 01 | D-E |
| 10 | H-L |
| 11 | SP |

| | |
|---|---|
| rh | The first (high-order) register of a designated register pair. |
| rl | The second (low-order) register of a designated register pair. |
| PC | 16-bit program counter register (PCH and PCL are used to refer to the high-order and low-order 8 bits respectively). |
| SP | 16-bit stack pointer register (SPH and SPL are used to refer to the high-order and low-order 8 bits respectively). |
| $r_m$ | Bit m of the register r (bits are number 7 through 0 from left to right). |
| Z,S,P,CY,AC | The condition flags: Zero, Sign, Parity, Carry, and Auxiliary Carry, respectively. |
| ( ) | The contents of the memory location or registers enclosed in the parentheses. |
| $\leftarrow$ | "Is transferred to" |
| $\wedge$ | Logical AND |
| $\forall$ | Exclusive OR |
| $\vee$ | Inclusive OR |
| + | Addition |
| − | Two's complement subtraction |
| * | Multiplication |
| $\leftrightarrow$ | "Is exchanged with" |
| — | The one's complement (e.g., $(\overline{A})$) |
| n | The restart number 0 through 7 |
| NNN | The binary representation 000 through 111 for restart number 0 through 7 respectively. |

## Description Format:

The following pages provide a detailed description of the instruction set of the 8080. Each instruction is described in the following manner:

1. The MAC 80 assembler format, consisting of the instruction mnemonic and operand fields, is printed in **BOLDFACE** on the left side of the first line.
2. The name of the instruction is enclosed in parenthesis on the right side of the first line.
3. The next line(s) contain a symbolic description of the operation of the instruction.
4. This is followed by a narative description of the operation of the instruction.
5. The following line(s) contain the binary fields and patterns that comprise the machine instruction.

6. The last four lines contain incidental information about the execution of the instruction. The number of machine cycles and states required to execute the instruction are listed first. If the instruction has two possible execution times, as in a Conditional Jump, both times will be listed, separated by a slash. Next, any significant data addressing modes (see Page 4-2) are listed. The last line lists any of the five Flags that are affected by the execution of the instruction.

## Data Transfer Group:

This group of instructions transfers data to and from registers and memory. **Condition flags are not affected** by any instruction in this group.

**MOV r1, r2** (Move Register)

(r1) ← (r2)

The content of register r2 is moved to register r1.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | D | D | D | S | S | S |

Cycles: 1
States: 5
Addressing: register
Flags: none

**MOV r, M** (Move from memory)

(r) ← ((H) (L))

The content of the memory location, whose address is in registers H and L, is moved to register r.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | D | D | D | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

**MOV M, r** (Move to memory)

((H) (L)) ← (r)

The content of register r is moved to the memory location whose address is in registers H and L.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | S | S | S |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

**MVI r, data** (Move Immediate)

(r) ← (byte 2)

The content of byte 2 of the instruction is moved to register r.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | D | D | D | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: none

**MVI M, data** (Move to memory immediate)

((H) (L)) ← (byte 2)

The content of byte 2 of the instruction is moved to the memory location whose address is in registers H and L.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 3
States: 10
Addressing: immed./reg. indirect
Flags: none

**LXI rp, data 16** (Load register pair immediate)

(rh) ← (byte 3),
(rl) ← (byte 2)

Byte 3 of the instruction is moved into the high-order register (rh) of the register pair rp. Byte 2 of the instruction is moved into the low-order register (rl) of the register pair rp.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | R | P | 0 | 0 | 0 | 1 |
| low-order data | | | | | | | |
| high-order data | | | | | | | |

Cycles: 3
States: 10
Addressing: immediate
Flags: none

**LDA addr** (Load Accumulator direct)

(A) ← ((byte 3)(byte 2))

The content of the memory location, whose address is specified in byte 2 and byte 3 of the instruction, is moved to register A.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| *low-order addr* | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 4
States: 13
Addressing: direct
Flags: none

**STA addr** (Store Accumulator direct)

((byte 3)(byte 2)) ← (A)

The content of the accumulator is moved to the memory location whose address is specified in byte 2 and byte 3 of the instruction.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 4
States: 13
Addressing: direct
Flags: none

**LHLD addr** (Load H and L direct)

(L) ← ((byte 3)(byte 2))
(H) ← ((byte 3)(byte 2) + 1)

The content of the memory location, whose address is specified in byte 2 and byte 3 of the instruction, is moved to register L. The content of the memory location at the succeeding address is moved to register H.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 5
States: 16
Addressing: direct
Flags: none

**SHLD addr** (Store H and L direct)

((byte 3)(byte 2)) ← (L)
((byte 3)(byte 2) + 1) ← (H)

The content of register L is moved to the memory location whose address is specified in byte 2 and byte 3. The content of register H is moved to the succeeding memory location.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 5
States: 16
Addressing: direct
Flags: none

**LDAX rp** (Load accumulator indirect)

(A) ← ((rp))

The content of the memory location, whose address is in the register pair rp, is moved to register A. Note: only register pairs rp=B (registers B and C) or rp=D (registers D and E) may be specified.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | R | P | 1 | 0 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

**STAX rp** (Store accumulator indirect)

((rp)) ← (A)

The content of register A is moved to the memory location whose address is in the register pair rp. Note: only register pairs rp=B (registers B and C) or rp=D (registers D and E) may be specified.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | R | P | 0 | 0 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

**XCHG** (Exchange H and L with D and E)

(H) ↔ (D)
(L) ↔ (E)

The contents of registers H and L are exchanged with the contents of registers D and E.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |

Cycles: 1
States: 4
Addressing: register
Flags: none

## Arithmetic Group:

This group of instructions performs arithmetic operations on data in registers and memory.

**Unless indicated otherwise, all instructions in this group affect the Zero, Sign, Parity, Carry, and Auxiliary Carry flags according to the standard rules.**

All subtraction operations are performed via two's complement arithmetic and set the carry flag to one to indicate a borrow and clear it to indicate no borrow.

**ADD r** (Add Register)

(A) ← (A) + (r)

The content of register r is added to the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**ADD M** (Add memory)

(A) ← (A) + ((H) (L))

The content of the memory location whose address is contained in the H and L registers is added to the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**ADI data** (Add immediate)

(A) ← (A) + (byte 2)

The content of the second byte of the instruction is added to the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**ADC r** (Add Register with carry)

(A) ← (A) + (r) + (CY)

The content of register r and the content of the carry bit are added to the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**ADC M** (Add memory with carry)

(A) ← (A) + ((H) (L)) + (CY)

The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are added to the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**ACI data** (Add immediate with carry)

(A) ← (A) + (byte 2) + (CY)

The content of the second byte of the instruction and the content of the CY flag are added to the contents of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**SUB r** (Subtract Register)

(A) ← (A) – (r)

The content of register r is subtracted from the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**SUB M** (Subtract memory)

(A) ← (A) − ((H) (L))

The content of the memory location whose address is contained in the H and L registers is subtracted from the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**SUI data** (Subtract immediate)

(A) ← (A) − (byte 2)

The content of the second byte of the instruction is subtracted from the content of the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**SBB r** (Subtract Register with borrow)

(A) ← (A) − (r) − (CY)

The content of register r and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**SBB M** (Subtract memory with borrow)

(A) ← (A) − ((H) (L)) − (CY)

The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**SBI data** (Subtract immediate with borrow)

(A) ← (A) − (byte 2) − (CY)

The contents of the second byte of the instruction and the contents of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**INR r** (Increment Register)

(r) ← (r) + 1

The content of register r is incremented by one. Note: All condition flags **except CY** are affected.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | D | D | D | 1 | 0 | 0 |

Cycles: 1
States: 5
Addressing: register
Flags: Z,S,P,AC

**INR M** (Increment memory)

((H) (L)) ← ((H) (L)) + 1

The content of the memory location whose address is contained in the H and L registers is incremented by one. Note: All condition flags **except CY** are affected.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,AC

**DCR r** (Decrement Register)

(r) ← (r) − 1

The content of register r is decremented by one. Note: All condition flags **except CY** are affected.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | D | D | D | 1 | 0 | 1 |

Cycles: 1
States: 5
Addressing: register
Flags: Z,S,P,AC

**DCR M** (Decrement memory)

((H) (L)) ← ((H) (L)) − 1

The content of the memory location whose address is contained in the H and L registers is decremented by one. Note: All condition flags **except CY** are affected.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,AC

**INX rp** (Increment register pair)

(rh) (rl) ← (rh) (rl) + 1

The content of the register pair rp is incremented by one. Note: **No condition flags are affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | R | P | 0 | 0 | 1 | 1 |

Cycles: 1
States: 5
Addressing: register
Flags: none

**DCX rp** (Decrement register pair)

(rh) (rl) ← (rh) (rl) − 1

The content of the register pair rp is decremented by one. Note: **No condition flags are affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | R | P | 1 | 0 | 1 | 1 |

Cycles: 1
States: 5
Addressing: register
Flags: none

**DAD rp** (Add register pair to H and L)

(H) (L) ← (H) (L) + (rh) (rl)

The content of the register pair rp is added to the content of the register pair H and L. The result is placed in the register pair H and L. Note: **Only the CY flag is affected.** It is set if there is a carry out of the double precision add; otherwise it is reset.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | R | P | 1 | 0 | 0 | 1 |

Cycles: 3
States: 10
Addressing: register
Flags: CY

**DAA** (Decimal Adjust Accumulator)

The eight-bit number in the accumulator is adjusted to form two four-bit Binary-Coded-Decimal digits by the following process:

1. If the value of the least significant 4 bits of the accumulator is greater than 9 **or** if the AC flag is set, 6 is added to the accumulator.
2. If the value of the most significant 4 bits of the accumulator is now greater than 9, **or** if the CY flag is set, 6 is added to the most significant 4 bits of the accumulator.

NOTE: All flags are affected.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: Z,S,P,CY,AC

## Logical Group:

This group of instructions performs logical (Boolean) operations on data in registers and memory and on condition flags.

Unless indicated otherwise, all instructions in this group affect the Zero, Sign, Parity, Auxiliary Carry, and Carry flags according to the standard rules.

**ANA r** (AND Register)

(A) ← (A) ∧ (r)

The content of register r is logically anded with the content of the accumulator. The result is placed in the accumulator. **The CY flag is cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**ANA M** (AND memory)

(A) ← (A) ∧ ((H) (L))

The contents of the memory location whose address is contained in the H and L registers is logically anded with the content of the accumulator. The result is placed in the accumulator. **The CY flag is cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**ANI data** (AND immediate)

$(A) \leftarrow (A) \wedge (\text{byte 2})$

The content of the second byte of the instruction is logically anded with the contents of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**XRA r** (Exclusive OR Register)

$(A) \leftarrow (A) \forall (r)$

The content of register r is exclusive-or'd with the content of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**XRA M** (Exclusive OR Memory)

$(A) \leftarrow (A) \forall ((H)\ (L))$

The content of the memory location whose address is contained in the H and L registers is exclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**XRI data** (Exclusive OR immediate)

$(A) \leftarrow (A) \forall (\text{byte 2})$

The content of the second byte of the instruction is exclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**ORA r** (OR Register)

$(A) \leftarrow (A) \vee (r)$

The content of register r is inclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**ORA M** (OR memory)

$(A) \leftarrow (A) \vee ((H)\ (L))$

The content of the memory location whose address is contained in the H and L registers is inclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**ORI data** (OR Immediate)

$(A) \leftarrow (A) \vee (\text{byte 2})$

The content of the second byte of the instruction is inclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. **The CY and AC flags are cleared.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**CMP r** (Compare Register)

$(A) - (r)$

The content of register r is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. **The Z flag is set to 1 if $(A) = (r)$. The CY flag is set to 1 if $(A) < (r)$.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 | S | S | S |

Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

**CMP M** (Compare memory)

(A) – ((H) (L))

The content of the memory location whose address is contained in the H and L registers is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 if (A) = ((H) (L)). The CY flag is set to 1 if (A) < ((H) (L)).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |

Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**CPI data** (Compare immediate)

(A) – (byte 2)

The content of the second byte of the instruction is subtracted from the accumulator. The condition flags are set by the result of the subtraction. The Z flag is set to 1 if (A) = (byte 2). The CY flag is set to 1 if (A) < (byte 2).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| data | | | | | | | |

Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

**RLC** (Rotate left)

$(A_{n+1}) \leftarrow (A_n)$ ; $(A_0) \leftarrow (A_7)$
$(CY) \leftarrow (A_7)$

The content of the accumulator is rotated left one position. The low order bit and the CY flag are both set to the value shifted out of the high order bit position. **Only the CY flag is affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: CY

**RRC** (Rotate right)

$(A_n) \leftarrow (A_{n-1})$ ; $(A_7) \leftarrow (A_0)$
$(CY) \leftarrow (A_0)$

The content of the accumulator is rotated right one position. The high order bit and the CY flag are both set to the value shifted out of the low order bit position. **Only the CY flag is affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: CY

**RAL** (Rotate left through carry)

$(A_{n+1}) \leftarrow (A_n)$ ; $(CY) \leftarrow (A_7)$
$(A_0) \leftarrow (CY)$

The content of the accumulator is rotated left one position through the CY flag. The low order bit is set equal to the CY flag and the CY flag is set to the value shifted out of the high order bit. **Only the CY flag is affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: CY

**RAR** (Rotate right through carry)

$(A_n) \leftarrow (A_{n+1})$ ; $(CY) \leftarrow (A_0)$
$(A_7) \leftarrow (CY)$

The content of the accumulator is rotated right one position through the CY flag. The high order bit is set to the CY flag and the CY flag is set to the value shifted out of the low order bit. **Only the CY flag is affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: CY

**CMA** (Complement accumulator)

$(A) \leftarrow (\overline{A})$

The contents of the accumulator are complemented (zero bits become 1, one bits become 0). **No flags are affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: none

**CMC** (Complement carry)

(CY) ← $(\overline{CY})$

The CY flag is complemented. **No other flags are affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: CY

**STC** (Set carry)

(CY) ← 1

The CY flag is set to 1. **No other flags are affected.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |

Cycles: 1
States: 4
Flags: CY

## Branch Group:

This group of instructions alter normal sequential program flow.

**Condition flags are not affected** by any instruction in this group.

The two types of branch instructions are unconditional and conditional. Unconditional transfers simply perform the specified operation on register PC (the program counter). Conditional transfers examine the status of one of the four processor flags to determine if the specified branch is to be executed. The conditions that may be specified are as follows:

| CONDITION | | CCC |
|---|---|---|
| NZ | – not zero (Z = 0) | 000 |
| Z | – zero (Z = 1) | 001 |
| NC | – no carry (CY = 0) | 010 |
| C | – carry (CY = 1) | 011 |
| PO | – parity odd (P = 0) | 100 |
| PE | – parity even (P = 1) | 101 |
| P | – plus (S = 0) | 110 |
| M | – minus (S = 1) | 111 |

**JMP addr** (Jump)

(PC) ← (byte 3) (byte 2)

Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 3
States: 10
Addressing: immediate
Flags: none

**Jcondition addr** (Conditional jump)

If (CCC),

(PC) ← (byte 3) (byte 2)

If the specified condition is true, control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction; otherwise, control continues sequentially.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | C | C | C | 0 | 1 | 0 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 3
States: 10
Addressing: immediate
Flags: none

**CALL addr** (Call)

((SP) – 1) ← (PCH)

((SP) – 2) ← (PCL)

(SP) ← (SP) – 2

(PC) ← (byte 3) (byte 2)

The high-order eight bits of the next instruction address are moved to the memory location whose address is one less than the content of register SP. The low-order eight bits of the next instruction address are moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by 2. Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 5
States: 17
Addressing: immediate/reg. indirect
Flags: none

**Ccondition addr** (Condition call)

If (CCC),

((SP) – 1) ← (PCH)

((SP) – 2) ← (PCL)

(SP) ← (SP) – 2

(PC) ← (byte 3) (byte 2)

If the specified condition is true, the actions specified *in the CALL instruction (see above) are performed;* otherwise, control continues sequentially.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | C | C | C | 1 | 0 | 0 |
| low-order addr | | | | | | | |
| high-order addr | | | | | | | |

Cycles: 3/5
States: 11/17
Addressing: immediate/reg. indirect
Flags: none

**RET** (Return)

(PCL) ← ((SP));

(PCH) ← ((SP) + 1);

(SP) ← (SP) + 2;

The content of the memory location whose address is specified in register SP is moved to the low-order eight bits of register PC. The content of the memory location whose address is one more than the content of register SP is moved to the high-order eight bits of register PC. The content of register SP is incremented by 2.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: none

**Rcondition** (Conditional return)

If (CCC),

(PCL) ← ((SP))

(PCH) ← ((SP) + 1)

(SP) ← (SP) + 2

If the specified condition is true, the actions specified in the RET instruction (see above) are performed; otherwise, control continues sequentially.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | C | C | C | 0 | 0 | 0 |

Cycles: 1/3
States: 5/11
Addressing: reg. indirect
Flags: none

**RST n** (Restart)

((SP) – 1) ← (PCH)

((SP) – 2) ← (PCL)

(SP) ← (SP) – 2

(PC) ← 8 * (NNN)

The high-order eight bits of the next instruction address are moved to the memory location whose *address is one less than the content of register SP.* The low-order eight bits of the next instruction address are moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by two. Control is transferred to the instruction whose address is eight times the content of NNN.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | N | N | N | 1 | 1 | 1 |

Cycles: 3
States: 11
Addressing: reg. indirect
Flags: none

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | N | N | N | 0 | 0 | 0 |

Program Counter After Restart

**PCHL** (Jump H and L indirect – move H and L to PC)

(PCH) ← (H)

(PCL) ← (L)

The content of register H is moved to the high-order eight bits of register PC. The content of register L is moved to the low-order eight bits of register PC.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |

Cycles: 1
States: 5
Addressing: register
Flags: none

## Stack, I/O, and Machine Control Group:

This group of instructions performs I/O, manipulates the Stack, and alters internal control flags.

Unless otherwise specified, **condition flags are not affected by any instructions in this group.**

FLAG WORD

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| S | Z | 0 | AC | 0 | P | 1 | CY |

**PUSH rp** (Push)

((SP) − 1) ← (rh)
((SP) − 2) ← (rl)
(SP) ← (SP) − 2

The content of the high-order register of register pair rp is moved to the memory location whose address is one less than the content of register SP. The content of the low-order register of register pair rp is moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by 2. **Note: Register pair rp = SP may not be specified.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | R | P | 0 | 1 | 0 | 1 |

Cycles: 3
States: 11
Addressing: reg. indirect
Flags: none

**PUSH PSW** (Push processor status word)

((SP) − 1) ← (A)
$((SP) - 2)_0$ ← (CY), $((SP) - 2)_1$ ← 1
$((SP) - 2)_2$ ← (P), $((SP) - 2)_3$ ← 0
$((SP) - 2)_4$ ← (AC), $((SP) - 2)_5$ ← 0
$((SP) - 2)_6$ ← (Z), $((SP) - 2)_7$ ← (S)
(SP) ← (SP) − 2

The content of register A is moved to the memory location whose address is one less than register SP. The contents of the condition flags are assembled into a processor status word and the word is moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by two.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |

Cycles: 3
States: 11
Addressing: reg. indirect
Flags: none

**POP rp** (Pop)

(rl) ← ((SP))
(rh) ← ((SP) + 1)
(SP) ← (SP) + 2

The content of the memory location, whose address is specified by the content of register SP, is moved to the low-order register of register pair rp. The content of the memory location, whose address is one more than the content of register SP, is moved to the high-order register of register pair rp. The content of register SP is incremented by 2. **Note: Register pair rp = SP may not be specified.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | R | P | 0 | 0 | 0 | 1 |

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: none

**POP PSW** (Pop processor status word)

(CY) ← $((SP))_0$
(P) ← $((SP))_2$
(AC) ← $((SP))_4$
(Z) ← $((SP))_6$
(S) ← $((SP))_7$
(A) ← ((SP) + 1)
(SP) ← (SP) + 2

The content of the memory location whose address is specified by the content of register SP is used to restore the condition flags. The content of the memory location whose address is one more than the content of register SP is moved to register A. The content of register SP is incremented by 2.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

**XTHL** (Exchange stack top with H and L)

(L) ↔ ((SP))

(H) ↔ ((SP) + 1)

The content of the L register is exchanged with the content of the memory location whose address is specified by the content of register SP. The content of the H register is exchanged with the content of the memory location whose address is one more than the content of register SP.

| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

Cycles: 5
States: 18
Addressing: reg. indirect
Flags: none

**SPHL** (Move HL to SP)

(SP) ← (H) (L)

The contents of registers H and L (16 bits) are moved to register SP.

| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|

Cycles: 1
States: 5
Addressing: register
Flags: none

**IN port** (Input)

(A) ← (data)

The data placed on the eight bit bi-directional data bus by the specified port is moved to register A.

| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|
| port | | | | | | | |

Cycles: 3
States: 10
Addressing: direct
Flags: none

**OUT port** (Output)

(data) ← (A)

The content of register A is placed on the eight bit bi-directional data bus for transmission to the specified port.

| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|
| port | | | | | | | |

Cycles: 3
States: 10
Addressing: direct
Flags: none

**EI** (Enable interrupts)

The interrupt system is enabled following the execution of the next instruction.

| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

Cycles: 1
States: 4
Flags: none

**DI** (Disable interrupts)

The interrupt system is disabled immediately following the execution of the DI instruction.

| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

Cycles: 1
States: 4
Flags: none

**HLT** (Halt)

The processor is stopped. The registers and flags are unaffected.

| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
|---|---|---|---|---|---|---|---|

Cycles: 1
States: 7
Flags: none

**NOP** (No op)

No operation is performed. The registers and flags are unaffected.

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|

Cycles: 1
States: 4
Flags: none

## INSTRUCTION SET

### Summary of Processor Instructions

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| MOV $r_1,r_2$ | Move register to register | 0 | 1 | D | D | D | S | S | S | 5 |
| MOV M,r | Move register to memory | 0 | 1 | 1 | 1 | 0 | S | S | S | 7 |
| MOV r,M | Move memory to register | 0 | 1 | D | D | D | 1 | 1 | 0 | 7 |
| HLT | Halt | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| MVI r | Move immediate register | 0 | 0 | D | D | D | 1 | 1 | 0 | 7 |
| MVI M | Move immediate memory | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 10 |
| INR r | Increment register | 0 | 0 | D | D | D | 1 | 0 | 0 | 5 |
| DCR r | Decrement register | 0 | 0 | D | D | D | 1 | 0 | 1 | 5 |
| INR M | Increment memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 10 |
| DCR M | Decrement memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 10 |
| ADD r | Add register to A | 1 | 0 | 0 | 0 | 0 | S | S | S | 4 |
| ADC r | Add register to A with carry | 1 | 0 | 0 | 0 | 1 | S | S | S | 4 |
| SUB r | Subtract register from A | 1 | 0 | 0 | 1 | 0 | S | S | S | 4 |
| SBB r | Subtract register from A with borrow | 1 | 0 | 0 | 1 | 1 | S | S | S | 4 |
| ANA r | And register with A | 1 | 0 | 1 | 0 | 0 | S | S | S | 4 |
| XRA r | Exclusive Or register with A | 1 | 0 | 1 | 0 | 1 | S | S | S | 4 |
| ORA r | Or register with A | 1 | 0 | 1 | 1 | 0 | S | S | S | 4 |
| CMP r | Compare register with A | 1 | 0 | 1 | 1 | 1 | S | S | S | 4 |
| ADD M | Add memory to A | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ADC M | Add memory to A with carry | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUB M | Subtract memory from A | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBB M | Subtract memory from A with borrow | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANA M | And memory with A | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRA M | Exclusive Or memory with A | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORA M | Or memory with A | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CMP M | Compare memory with A | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| ADI | Add immediate to A | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ACI | Add immediate to A with carry | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUI | Subtract immediate from A | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBI | Subtract immediate from A with borrow | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANI | And immediate with A | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRI | Exclusive Or immediate with A | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORI | Or immediate with A | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CPI | Compare immediate with A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| RLC | Rotate A left | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 |
| RRC | Rotate A right | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 4 |
| RAL | Rotate A left through carry | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| RAR | Rotate A right through carry | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 |
| JMP | Jump unconditional | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 10 |
| JC | Jump on carry | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 10 |
| JNC | Jump on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 10 |
| JZ | Jump on zero | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 10 |
| JNZ | Jump on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 10 |
| JP | Jump on positive | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 10 |
| JM | Jump on minus | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 10 |
| JPE | Jump on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 10 |
| JPO | Jump on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 10 |
| CALL | Call unconditional | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 17 |
| CC | Call on carry | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CNC | Call on no carry | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CZ | Call on zero | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CNZ | Call on no zero | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| CP | Call on positive | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CM | Call on minus | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CPE | Call on parity even | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CPO | Call on parity odd | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| RET | Return | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| RC | Return on carry | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RNC | Return on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 5/11 |

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| RZ | Return on zero | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RNZ | Return on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RP | Return on positive | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RM | Return on minus | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RPE | Return on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RPO | Return on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RST | Restart | 1 | 1 | A | A | A | 1 | 1 | 1 | 11 |
| IN | Input | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 10 |
| OUT | Output | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 10 |
| LXI B | Load immediate register Pair B & C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI D | Load immediate register Pair D & E | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| LXI H | Load immediate register Pair H & L | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI SP | Load immediate stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| PUSH B | Push register Pair B & C on stack | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH D | Push register Pair D & E on stack | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 11 |
| PUSH H | Push register Pair H & L on stack | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH PSW | Push A and Flags on stack | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 11 |
| POP B | Pop register pair B & C off stack | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP D | Pop register pair D & E off stack | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| POP H | Pop register pair H & L off stack | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP PSW | Pop A and Flags off stack | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| STA | Store A direct | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 13 |
| LDA | Load A direct | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 13 |
| XCHG | Exchange D & E, H & L Registers | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 4 |
| XTHL | Exchange top of stack, H & L | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 18 |
| SPHL | H & L to stack pointer | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 5 |
| PCHL | H & L to program counter | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 5 |
| DAD B | Add B & C to H & L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD D | Add D & E to H & L | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 10 |
| DAD H | Add H & L to H & L | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD SP | Add stack pointer to H & L | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 10 |
| STAX B | Store A indirect | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7 |
| STAX D | Store A indirect | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 7 |
| LDAX B | Load A indirect | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 |
| LDAX D | Load A indirect | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 7 |
| INX B | Increment B & C registers | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX D | Increment D & E registers | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 5 |
| INX H | Increment H & L registers | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX SP | Increment stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 5 |
| DCX B | Decrement B & C | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX D | Decrement D & E | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 5 |
| DCX H | Decrement H & L | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX SP | Decrement stack pointer | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 5 |
| CMA | Complement A | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 4 |
| STC | Set carry | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 4 |
| CMC | Complement carry | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| DAA | Decimal adjust A | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 4 |
| SHLD | Store H & L direct | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 16 |
| LHLD | Load H & L direct | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 16 |
| EI | Enable Interrupts | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 |
| DI | Disable interrupt | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| NOP | No-operation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |

NOTES: 1. DDD or SSS - 000 B - 001 C - 010 D - 011 E - 100 H - 101 L - 110 Memory - 111 A.

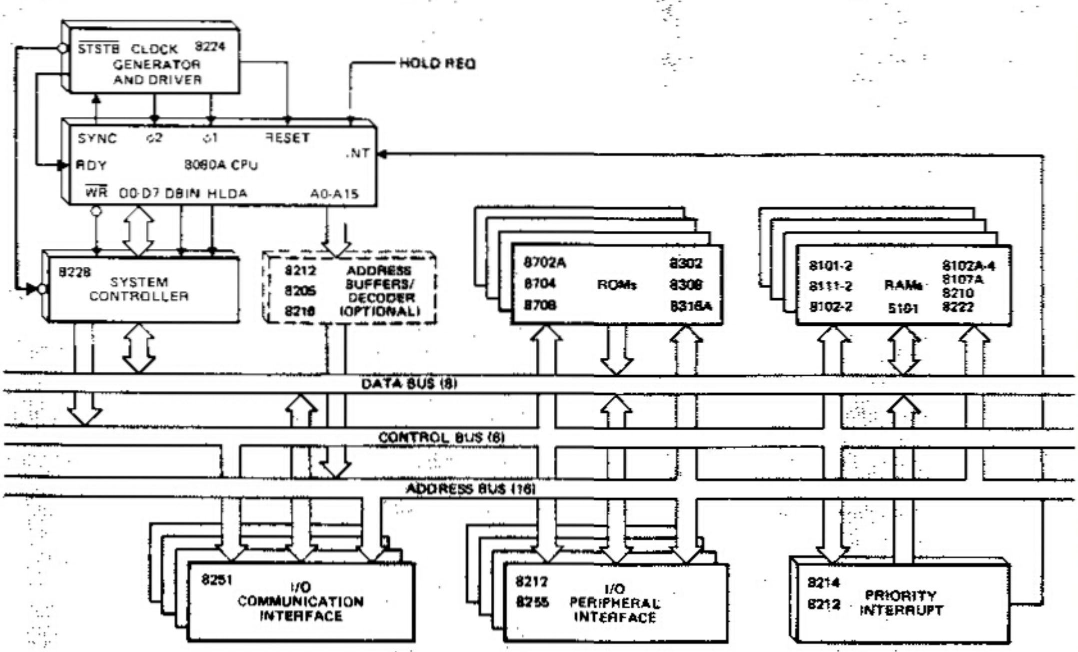
2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.

# CHAPTER 5
# MCS-80 COMPONENT FAMILY

intel® Microcomputer Systems

# CPU Group

## 8224 8080A-1
## 8228 8080A-2
## 8080A M8080-A



# Schottky Bipolar 8224

## CLOCK GENERATOR AND DRIVER FOR 8080A CPU

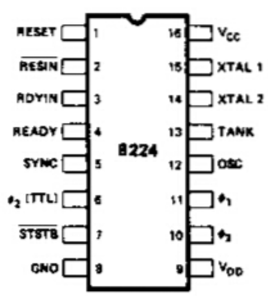
- Single Chip Clock Generator/Driver for 8080A CPU
- Power-Up Reset for CPU
- Ready Synchronizing Flip-Flop
- Advanced Status Strobe
- Oscillator Output for External System Timing
- Crystal Controlled for Stable System Operation
- Reduces System Package Count

The 8224 is a single chip clock generator/driver for the 8080A CPU. It is controlled by a crystal, selected by the designer, to meet a variety of system speed requirements.

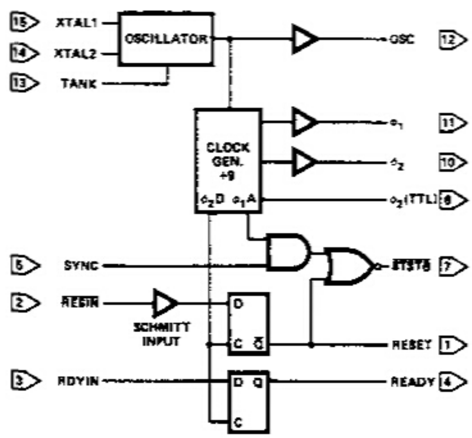
Also included are circuits to provide power-up reset, advance status strobe and synchronization of ready.

The 8224 provides the designer with a significant reduction of packages used to generate clocks and timing for 8080A.

### PIN CONFIGURATION




### BLOCK DIAGRAM




### PIN NAMES

| | |
|---|---|
| $\overline{RESIN}$ | RESET INPUT |
| RESET | RESET OUTPUT |
| RDYIN | READY INPUT |
| READY | READY OUTPUT |
| SYNC | SYNC INPUT |
| $\overline{STSTB}$ | STATUS STB (ACTIVE LOW) |
| $\phi_1$ | 8080 |
| $\phi_2$ | CLOCKS |

| | |
|---|---|
| XTAL 1 | CONNECTIONS |
| XTAL 2 | FOR CRYSTAL |
| TANK | USED WITH OVERTONE XTAL |
| OSC | OSCILLATOR OUTPUT |
| $\phi_2$ (TTL) | $\phi_2$ CLK (TTL LEVEL) |
| $V_{CC}$ | +5V |
| $V_{DD}$ | +12V |
| GND | 0V |

## FUNCTIONAL DESCRIPTION

### General

The 8224 is a single chip Clock Generator/Driver for the 8080A CPU. It contains a crystal-controlled oscillator, a "divide by nine" counter, two high-level drivers and several auxiliary logic functions.

### Oscillator

The oscillator circuit derives its basic operating frequency from an external, series resonant, fundamental mode crystal. Two inputs are provided for the crystal connections (XTAL1, XTAL2).

The selection of the external crystal frequency depends mainly on the speed at which the 8080A is to be run at. Basically, the oscillator operates at 9 times the desired processor speed.

A simple formula to guide the crystal selection is:

$$\text{Crystal Frequency} = \frac{1}{t_{CY}} \text{ times } 9$$

Example 1: ($500ns\ t_{CY}$)
2mHz times 9 = 18mHz*

Example 2: ($800ns\ t_{CY}$)
1.25mHz times 9 = 11.25mHz

Another input to the oscillator is TANK. This input allows the use overtone mode crystals. This type of crystal generally has much lower "gain" than the fundamental type so an external LC network is necessary to provide the additional "gain" for proper oscillator operation. The external LC network is connected to the TANK input and is AC coupled to ground. See Figure 4.

The formula for the LC network is:

$$F = \frac{1}{2\pi\sqrt{LC}}$$

The output of the oscillator is buffered and brought out on OSC (pin 12) so that other system timing signals can be derived from this stable, crystal-controlled source.

*When using crystals above 10mHz a small amount of frequency "trimming" may be necessary to produce the exact desired frequency. The addition of a small selected capacitance (3pF - 10pF) in series with the crystal will accomplish this function.

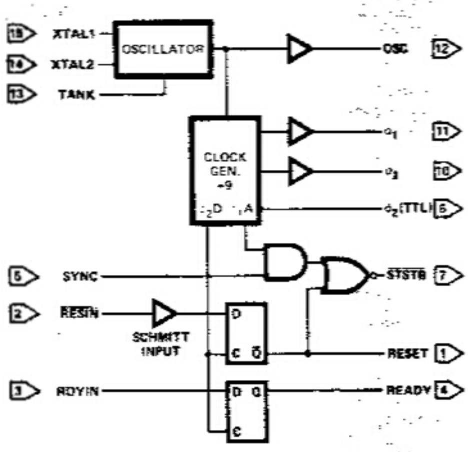
### Clock Generator

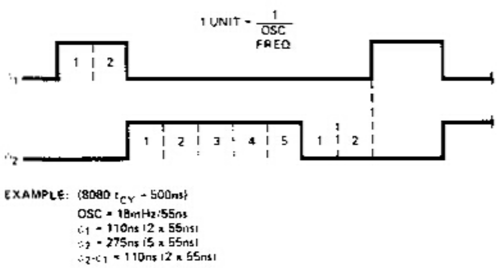
The Clock Generator consists of a synchronous "divide by nine" counter and the associated decode gating to create the waveforms of the two 8080A clocks and auxiliary timing signals.

The waveforms generated by the decode gating follow a simple 2-5-2 digital pattern. See Figure 2. The clocks generated; phase 1 and phase 2, can best be thought of as consisting of "units" based on the oscillator frequency. Assume that one "unit" equals the period of the oscillator frequency. By multiplying the number of "units" that are contained in a pulse width or delay, times the period of the oscillator frequency, the approximate time in nanoseconds can be derived.

The outputs of the clock generator are connected to two high level drivers for direct interface to the 8080A CPU. A TTL level phase 2 is also brought out $\phi_2$ (TTL) for external timing purposes. It is especially useful in DMA dependant activities. This signal is used to gate the requesting device onto the bus once the 8080A CPU issues the Hold Acknowledgement (HLDA).

Several other signals are also generated internally so that optimum timing of the auxiliary flip-flops and status strobe ($\overline{STSTB}$) is achieved.



### STSTB (Status Strobe)

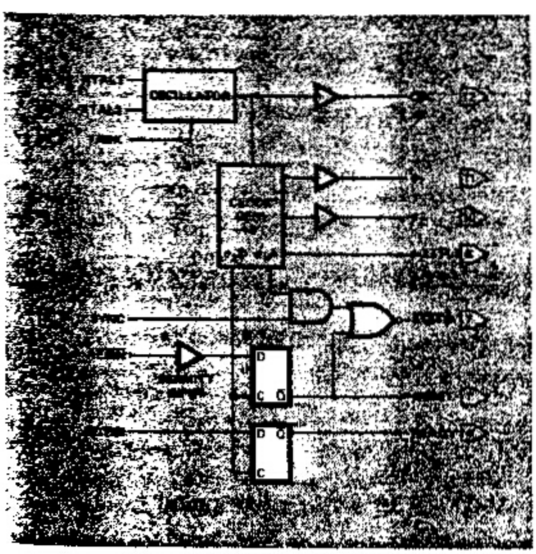
At the beginning of each machine cycle the 8080A CPU issues status information on its data bus. This information tells what type of action will take place during that machine cycle. By bringing in the SYNC signal from the CPU, and gating it with an internal timing signal (ϕ1A), an active low strobe can be derived that occurs at the start of each machine cycle at the earliest possible moment that status data is stable on the bus. The $\overline{STSTB}$ signal connects directly to the 8228 System Controller.

The power-on Reset also generates $\overline{STSTB}$, but of course, for a longer period of time. This feature allows the 8228 to be automatically reset without additional pins devoted for this function.

### Power-On Reset and Ready Flip-Flops

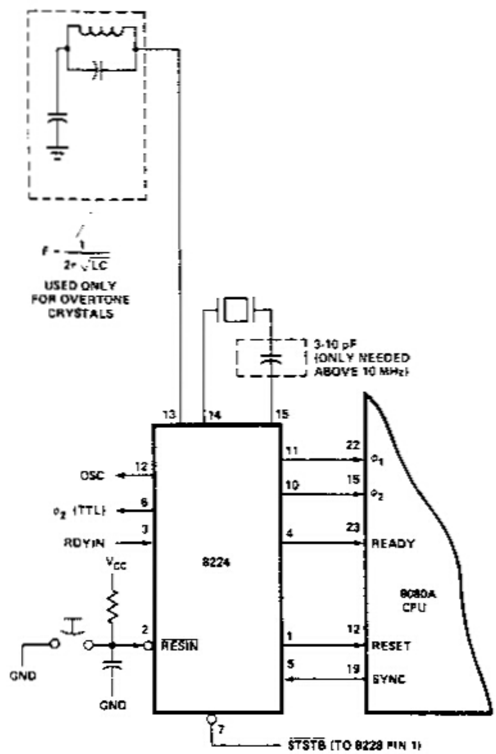
A common function in 8080A Microcomputer systems is the generation of an automatic system reset and start-up upon initial power-on. The 8224 has a built in feature to accomplish this feature.

An external RC network is connected to the $\overline{RESIN}$ input. The slow transition of the power supply rise is sensed by an internal Schmitt Trigger. This circuit converts the slow transition into a clean, fast edge when its input level reaches a predetermined value. The output of the Schmitt Trigger is connected to a "D" type flip-flop that is clocked with ϕ2D (an internal timing signal). The flip-flop is synchronously reset and an active high level that complies with the 8080A input spec is generated. For manual switch type system Reset circuits, an active low switch closing can be connected to the $\overline{RESIN}$ input in addition to the power-on RC network.



The READY input to the 8080A CPU has certain timing specifications such as "set-up and hold" thus, an external synchronizing flip-flop is required. The 8224 has this feature built-in. The RDYIN input presents the asynchronous "wait request" to the "D" type flip-flop. By clocking the flip-flop with ϕ2D, a synchronized READY signal at the correct input level, can be connected directly to the 8080A.

The reason for requiring an external flip-flop to synchronize the "wait request" rather than internally in the 8080 CPU is that due to the relatively long delays of MOS logic such an implementation would "rob" the designer of about 200ns during the time his logic is determining if a "wait" is necessary. An external bipolar circuit built into the clock generator eliminates most of this delay and has no effect on component count.



## D.C. Characteristics

$T_A$ = 0°C to 70°C; $V_{CC}$ = +5.0V ±5%; $V_{DD}$ = +12V ±5%.

| Symbol | Parameter | Limits Min. | Typ. | Max. | Units | Test Conditions |
|---|---|---|---|---|---|---|
| $I_F$ | Input Current Loading | | | -.25 | mA | $V_F$ = .45V |
| $I_R$ | Input Leakage Current | | | 10 | µA | $V_R$ = 5.25V |
| $V_C$ | Input Forward Clamp Voltage | | | 1.0 | V | $I_C$ = -5mA |
| $V_{IL}$ | Input "Low" Voltage | | | .8 | V | $V_{CC}$ = 5.0V |
| $V_{IH}$ | Input "High" Voltage | 2.6<br>2.0 | | | V | Reset Input<br>All Other Inputs |
| $V_{IH}$-$V_{IL}$ | REDIN Input Hysteresis | .25 | | | mV | $V_{CC}$ = 5.0V |
| $V_{OL}$ | Output "Low" Voltage | | | .45<br>.45 | V<br>V | ($\phi_1$,$\phi_2$), Ready, Reset, $\overline{STSTB}$ $I_{OL}$ =2.5mA<br>All Other Outputs $I_{OL}$ = 15mA |
| $V_{OH}$ | Output "High" Voltage<br>$\phi_1$ , $\phi_2$<br>READY, RESET<br>All Other Outputs | <br>9.4<br>3.6<br>2.4 | | | <br>V<br>V<br>V | <br>$I_{OH}$ = -100µA<br>$I_{OH}$ = -100µA<br>$I_{OH}$ = -1mA |
| $I_{SC}$[1] | Output Short Circuit Current (All Low Voltage Outputs Only) | -10 | | -60 | mA | $V_O$ = 0V<br>$V_{CC}$ = 5.0V |
| $I_{CC}$ | Power Supply Current | | | 115 | mA | |
| $I_{DD}$ | Power Supply Current | | | 12 | mA | |

Note: 1. Caution, $\phi_1$ and $\phi_2$ output drivers do not have short circuit protection

## CRYSTAL REQUIREMENTS

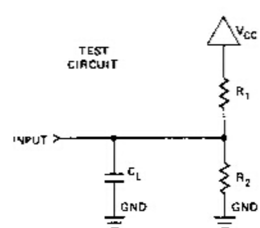
Tolerance: .005% at 0°C -70°C
Resonance: Series (Fundamental)*
Load Capacitance: 20-35pF
Equivalent Resistance: 75-20 ohms
Power Dissipation (Min): 4mW

*With tank circuit use 3rd overtone mode.

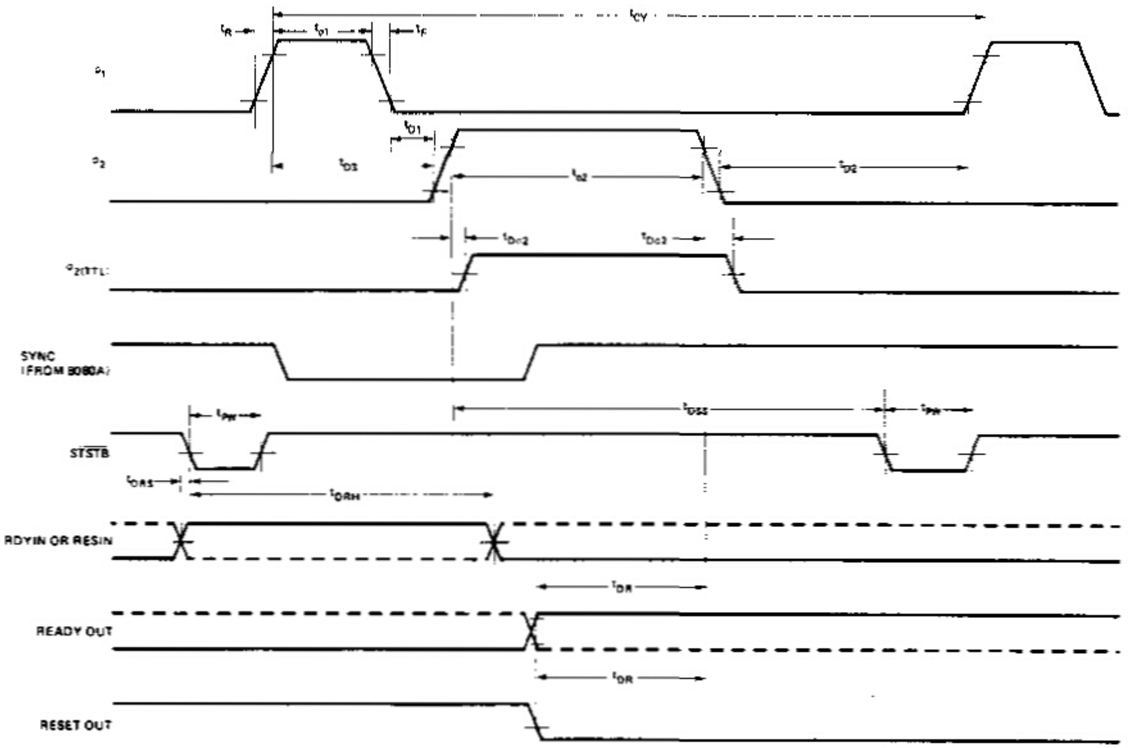
## A.C. Characteristics

$V_{CC}$ = +5.0V ± 5%; $V_{DD}$ = +12.0V ± 5%; $T_A$ = 0°C to 70°C

| Symbol | Parameter | Limits Min. | Limits Typ. | Limits Max. | Units | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | $\frac{2tcy}{9}$ - 20ns | | | ns | $C_L$ = 20pF to 50pF |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | $\frac{5tcy}{9}$ - 35ns | | | | |
| $t_{D1}$ | $\phi_1$ to $\phi_2$ Delay | 0 | | | | |
| $t_{D2}$ | $\phi_2$ to $\phi_1$ Delay | $\frac{2tcy}{9}$ - 14ns | | | | |
| $t_{D3}$ | $\phi_1$ to $\phi_2$ Delay | $\frac{2tcy}{9}$ | | $\frac{2tcy}{9}$ + 20ns | | |
| $t_R$ | $\phi_1$ and $\phi_2$ Rise Time | | | 20 | | |
| $t_F$ | $\phi_1$ and $\phi_2$ Fall Time | | | 20 | | |
| $t_{D\phi 2}$ | $\phi_2$ to $\phi_2$ (TTL) Delay | -5 | | +15 | ns | $\phi_2$TTL,CL=30 $R_1$=300Ω $R_2$=600Ω |
| $t_{DSS}$ | $\phi_2$ to $\overline{STSTB}$ Delay | $\frac{6tcy}{9}$ - 30ns | | $\frac{6tcy}{9}$ | | $\overline{STSTB}$, CL=15pF $R_1$ = 2K $R_2$ = 4K |
| $t_{PW}$ | $\overline{STSTB}$ Pulse Width | $\frac{tcy}{9}$ - 15ns | | | | |
| $t_{DRS}$ | RDYIN Setup Time to Status Strobe | 50ns - $\frac{4tcy}{9}$ | | | | |
| $t_{DRH}$ | RDYIN Hold Time After $\overline{STSTB}$ | $\frac{4tcy}{9}$ | | | | |
| $t_{DR}$ | RDYIN or RESIN to $\phi_2$ Delay | $\frac{4tcy}{9}$ - 25ns | | | | Ready & Reset CL=10pF $R_1$=2K $R_2$=4K |
| $t_{CLK}$ | CLK Period | | $\frac{tcy}{9}$ | | | |
| $f_{max}$ | Maximum Oscillating Frequency | 27 | | | MHz | |
| $C_{in}$ | Input Capacitance | | | 8 | pF | $V_{CC}$=+5.0V $V_{DD}$=+12V $V_{BIAS}$=2.5V f=1MHz |



WAVEFORMS




VOLTAGE MEASUREMENT POINTS: $\phi_1$, $\phi_2$ Logic "0" = 1.0V, Logic "1" = 8.0V. All other signals measured at 1.5V.

## EXAMPLE:

### A.C. Characteristics (For $t_{CY}$ = 488.28 ns)

$T_A$ = 0°C to 70°C; $V_{DD}$ = +5V ±5%; $V_{DD}$ = +12V ±5%.

| Symbol | Parameter | Limits Min. | Limits Typ. | Limits Max. | Units | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | 89 | | | ns | $t_{CY}$=488.28ns |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | 236 | | | ns | |
| $t_{D1}$ | Delay $\phi_1$ to $\phi_2$ | 0 | | | ns | |
| $t_{D2}$ | Delay $\phi_2$ to $\phi_1$ | 95 | | | ns | $\phi_1$ & $\phi_2$ Loaded to $C_L$ = 20 to 50pF |
| $t_{D3}$ | Delay $\phi_1$ to $\phi_2$ Leading Edges | 109 | | 129 | ns | |
| $t_r$ | Output Rise Time | | | 20 | ns | |
| $t_f$ | Output Fall Time | | | 20 | ns | |
| $t_{DSS}$ | $\phi_2$ to $\overline{STSTB}$ Delay | 296 | | 326 | ns | |
| $t_{D\phi 2}$ | $\phi_2$ to $\phi_2$ (TTL) Delay | -5 | | +15 | ns | |
| $t_{PW}$ | Status Strobe Pulse Width | 40 | | | ns | Ready & Reset Loaded to 2mA/10pF |
| $t_{DRS}$ | RDYIN Setup Time to $\overline{STSTB}$ | -167 | | | ns | |
| $t_{DRH}$ | RDYIN Hold Time after $\overline{STSTB}$ | 217 | | | ns | All measurements referenced to 1.5V unless specified otherwise. |
| $t_{DR}$ | READY or RESET to $\phi_2$ Delay | 192 | | | ns | |
| $f_{MAX}$ | Oscillator Frequency | | | 18.432 | MHz | |

# Schottky Bipolar 8228

## SYSTEM CONTROLLER AND BUS DRIVER FOR 8080A CPU

- **Single Chip System Control for MCS-80 Systems**
- **Built-in Bi-Directional Bus Driver for Data Bus Isolation**
- **Allows the use of Multiple Byte Instructions (e.g. CALL) for Interrupt Acknowledge**
- **User Selected Single Level Interrupt Vector (RST 7)**
- **28 Pin Dual In-Line Package**
- **Reduces System Package Count**

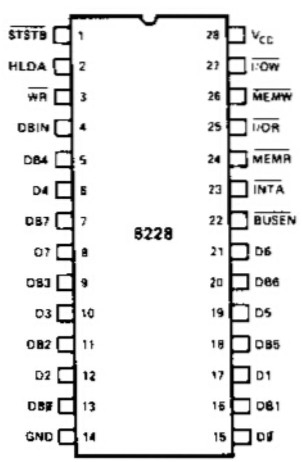
The 8228 is a single chip system controller and bus driver for MCS-80. It generates all signals required to directly interface MCS-80 family RAM, ROM, and I/O components.

A bi-directional bus driver is included to provide high system TTL fan-out. It also provides isolation of the 8080 data bus from memory and I/O. This allows for the optimization of control signals, enabling the systems deisgner to use slower memory and I/O. The isolation of the bus driver also provides for enhanced system noise immunity.

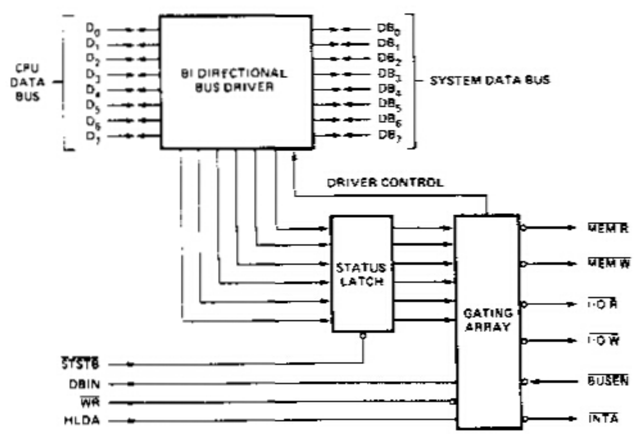
A user selected single level interrupt vector (RST 7) is provided to simplify real time, interrupt driven, small system requirements. The 8228 also generates the correct control signals to allow the use of multiple byte instructions (e.g., CALL) in response to an INTERRUPT ACKNOWLEDGE by the 8080A. This feature permits large, interrupt driven systems to have an unlimited number of interrupt levels.

The 8228 is designed to support a wide variety of system bus structures and also reduce system package count for cost effective, reliable, design of the MCS-80 systems.

### PIN CONFIGURATION




### 8228 BLOCK DIAGRAM




### PIN NAMES

| | | | |
|---|---|---|---|
| D7-D0 | DATA BUS (8080 SIDE) | INTA | INTERRUPT ACKNOWLEDGE |
| DB7-DB0 | DATA BUS (SYSTEM SIDE) | HLDA | HLDA (FROM 8080) |
| I/OR | I/O READ | WR | WR (FROM 8080) |
| I/OW | I/O WRITE | BUSEN | BUS ENABLE INPUT |
| MEMR | MEMORY READ | STSTB | STATUS STROBE (FROM 8224) |
| MEMW | MEMORY WRITE | Vcc | +5V |
| DBIN | DBIN (FROM 8080) | GND | 0 VOLTS |

## FUNCTIONAL DESCRIPTION

### General

The 8228 is a single chip System Controller and Data Bus driver for the 8080 Microcomputer System. It generates all control signals required to directly interface MCS-80™ family RAM, ROM, and I/O components.

Schottky Bipolar technology is used to maintain low delay times and provide high output drive capability to support small to medium systems.

### Bi-Directional Bus Driver

An eight bit, bi-directional bus driver is provided to buffer the 8080 data bus from Memory and I/O devices. The 8080A data bus has an input requirement of 3.3 volts (min) and can drive (sink) a maximum current of 1.9mA. The 8228 data bus driver assures that these input requirements will be not only met but exceeded for enhanced noise immunity. Also, on the system side of the driver adequate drive current is available (10mA Typ.) so that a large number of Memory and I/O devices can be directly connected to the bus.

The Bi-Directional Bus Driver is controlled by signals from the Gating Array so that proper bus flow is maintained and its outputs can be forced into their high impedance state (3-state) for DMA activities.

### Status Latch

At the beginning of each machine cycle the 8080 CPU issues "status" information on its data bus that indicates the type of activity that will occur during the cycle. The 8228 stores this information in the Status Latch when the STSTB input goes "low". The output of the Status Latch is connected to the Gating Array and is part of the Control Signal generation.

### Gating Array

The Gating Array generates control signals (MEM R, MEM W, I/O R, I/O W and INTA) by gating the outputs of the Status Latch with signals from the 8080 CPU (DBIN, WR, and HLDA).

The "read" control signals (MEM R, I/O R and INTA) are derived from the logical combination of the appropriate Status Bit (or bits) and the DBIN input from the 8080 CPU.

The "write" control signals (MEM W, I/O W) are derived from the logical combination of the appropriate Status Bit (or bits) and the WR input from the 8080 CPU.

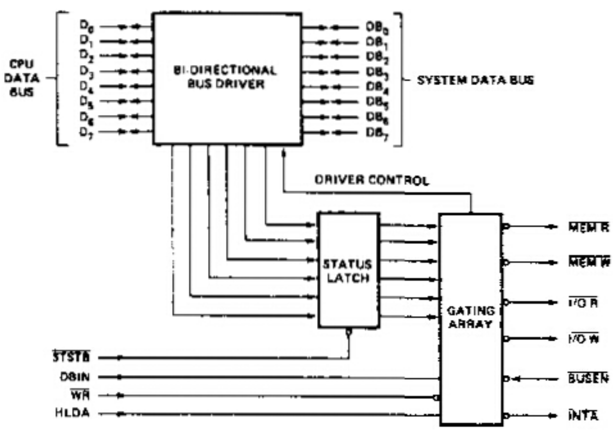
All Control Signals are "active low" and directly interface to MCS-80 family RAM, ROM and I/O components.

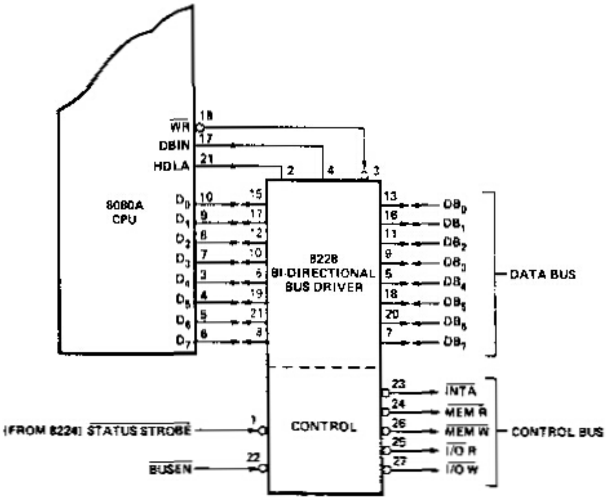
The INTA control signal is normally used to gate the "interrupt instruction port" onto the bus. It also provides a special feature in the 8228. If only one basic vector is needed in the interrupt structure, such as in small systems, the 8228 can automatically insert a RST 7 instruction onto the bus at the proper time. To use this option, simply connect the INTA output of the 8228 (pin 23) to the +12 volt supply through a series resistor (1K ohms). The voltage is sensed internally by the 8228 and logic is "set-up" so that when the DBIN input is active a RST 7 instruction is gated on to the bus when an interrupt is acknowledged. This feature provides a single interrupt vector with no additional components, such as an interrupt instruction port.

**When using CALL as an Interrupt instruction the 8228 will generate an INTA pulse for each of the three bytes.**

The BUSEN (Bus Enable) input to the Gating Array is an asynchronous input that forces the data bus output buffers and control signal buffers into their high-impedance state if it is a "one". If BUSEN is a "zero" normal operation of the data buffer and control signals take place.

**8228 BLOCK DIAGRAM**



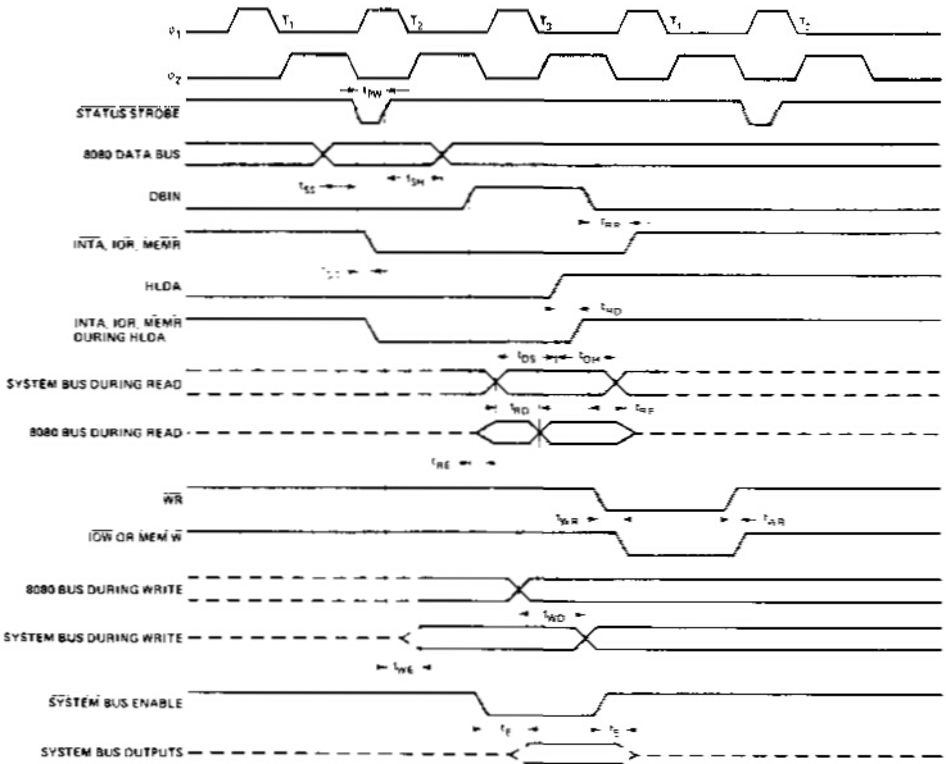



STATUS WORD CHART

TYPE OF MACHINE CYCLE (Ⓝ STATUS WORD)

| DATA BUS BIT | STATUS INFORMATION | ① INSTRUCTION FETCH | ② MEMORY READ | ③ MEMORY WRITE | ④ STACK READ | ⑤ STACK WRITE | ⑥ INPUT READ | ⑦ OUTPUT WRITE | ⑧ INTERRUPT ACKNOWLEDGE | ⑨ HALT ACKNOWLEDGE | ⑩ INTERRUPT ACKNOWLEDGE WHILE HALT |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $D_0$ | INTA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| $D_1$ | $\overline{WO}$ | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| $D_2$ | STACK | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| $D_3$ | HLTA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $D_4$ | OUT | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $D_5$ | $M_1$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| $D_6$ | INP | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $D_7$ | MEMR | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| CONTROL SIGNALS | | $\overline{MEM R}$ | $\overline{MEM R}$ | $\overline{MEM W}$ | $\overline{MEM R}$ | $\overline{MEM W}$ | $\overline{I/O R}$ | $\overline{I/O W}$ | $\overline{INTA}$ | (NONE) | $\overline{INTA}$ |

## WAVEFORMS




VOLTAGE MEASUREMENT POINTS: $D_0$-$D_7$ (when outputs) Logic "0" = 0.8V, Logic "1" = 3.0V. All other signals measured at 1.5V.

## A.C. Characteristics $T_A = 0°C$ to $70°C$; $V_{CC} = 5V \pm 5\%$.

| Symbol | Parameter | Limits Min. | Limits Max. | Units | Condition |
|---|---|---|---|---|---|
| $t_{PW}$ | Width of Status Strobe | 22 | | ns | |
| $t_{SS}$ | Setup Time, Status Inputs $D_0$-$D_7$ | 8 | | ns | |
| $t_{SH}$ | Hold Time, Status Inputs $D_0$-$D_7$ | 5 | | ns | |
| $t_{DC}$ | Delay from STSTB to any Control Signal | 20 | 60 | ns | $C_L$ = 100pF |
| $t_{RR}$ | Delay from DBIN to Control Outputs | | 30 | ns | $C_L$ = 100pF |
| $t_{RE}$ | Delay from DBIN to Enable/Disable 8080 Bus | | 45 | ns | $C_L$ = 25pF |
| $t_{RD}$ | Delay from System Bus to 8080 Bus during Read | | 30 | ns | $C_L$ = 25pF |
| $t_{WR}$ | Delay from WR to Control Outputs | 5 | 45 | ns | $C_L$ = 100pF |
| $t_{WE}$ | Delay to Enable System Bus $DB_0$-$DB_7$ after STSTB | | 30 | ns | $C_L$ = 100pF |
| $t_{WD}$ | Delay from 8080 Bus $D_0$-$D_7$ to System Bus $DB_0$-$DB_7$ during Write | 5 | 40 | ns | $C_L$ = 100pF |
| $t_E$ | Delay from System Bus Enable to System Bus $DB_0$-$DB_7$ | | 30 | ns | $C_L$ = 100pF |
| $t_{HD}$ | HLDA to Read Status Outputs | | 25 | ns | |
| $t_{DS}$ | Setup Time, System Bus Inputs to HLDA | 10 | | ns | |
| $t_{DH}$ | Hold Time, System Bus Inputs to HLDA | 20 | | ns | $C_L$ = 100pF |

## D.C. Characteristics $T_A = 0°C$ to $70°C$; $V_{CC} = 5V \pm 5\%$.

| Symbol | Parameter | Limits Min. | Limits Typ.[1] | Limits Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $V_C$ | Input Clamp Voltage, All Inputs | | .75 | -1.0 | V | $V_{CC}$=4.75V; $I_C$=-5mA |
| $I_F$ | Input Load Current, $\overline{STSTB}$ | | | 500 | µA | $V_{CC}$=5.25V |
| | $D_2$ & $D_6$ | | | 750 | µA | $V_F$=0.45V |
| | $D_0$, $D_1$, $D_4$, $D_5$, & $D_7$ | | | 250 | µA | |
| | All Other Inputs | | | 250 | µA | |
| $I_R$ | Input Leakage Current $\overline{STSTB}$ | | | 100 | µA | $V_{CC}$=5.25V |
| | $DB_0$-$DB_7$ | | | 20 | µA | $V_R$=5.25V |
| | All Other Inputs | | | 100 | µA | |
| $V_{TH}$ | Input Threshold Voltage, All Inputs | 0.8 | | 2.0 | V | $V_{CC}$=5V |
| $I_{CC}$ | Power Supply Current | | 140 | 190 | mA | $V_{CC}$=5.25V |
| $V_{OL}$ | Output Low Voltage, $D_0$-$D_7$ | | | .45 | V | $V_{CC}$=4.75V; $I_{OL}$=2mA |
| | All Other Outputs | | | .45 | V | $I_{OL}$ = 10mA |
| $V_{OH}$ | Output High Voltage, $D_0$-$D_7$ | 3.6 | 3.8 | | V | $V_{CC}$=4.75V; $I_{OH}$=-10µA |
| | All Other Outputs | 2.4 | | | V | $I_{OH}$ = -1mA |
| $I_{OS}$ | Short Circuit Current, All Outputs | 15 | | 90 | mA | $V_{CC}$=5V |
| $I_{O(off)}$ | Off State Output Current, All Control Outputs | | | 100 | µA | $V_{CC}$=5.25V; $V_O$=5.25 |
| | | | | -100 | µA | $V_O$=.45V |
| $I_{INT}$ | INTA Current | | | 5 | mA | (See Figure below) |

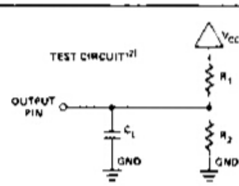
Note 1: Typical values are for $T_A$ = 25°C and nominal supply voltages.

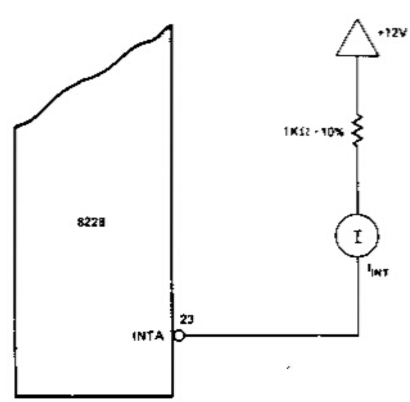
## Capacitance
This parameter is periodically sampled and not 100% tested.

| Symbol | Parameter | Limits Min. | Limits Typ.[1] | Limits Max. | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | | 8 | 12 | pF |
| $C_{OUT}$ | Output Capacitance Control Signals | | 7 | 15 | pF |
| I/O | I/O Capacitance (D or DB) | | 8 | 15 | pF |

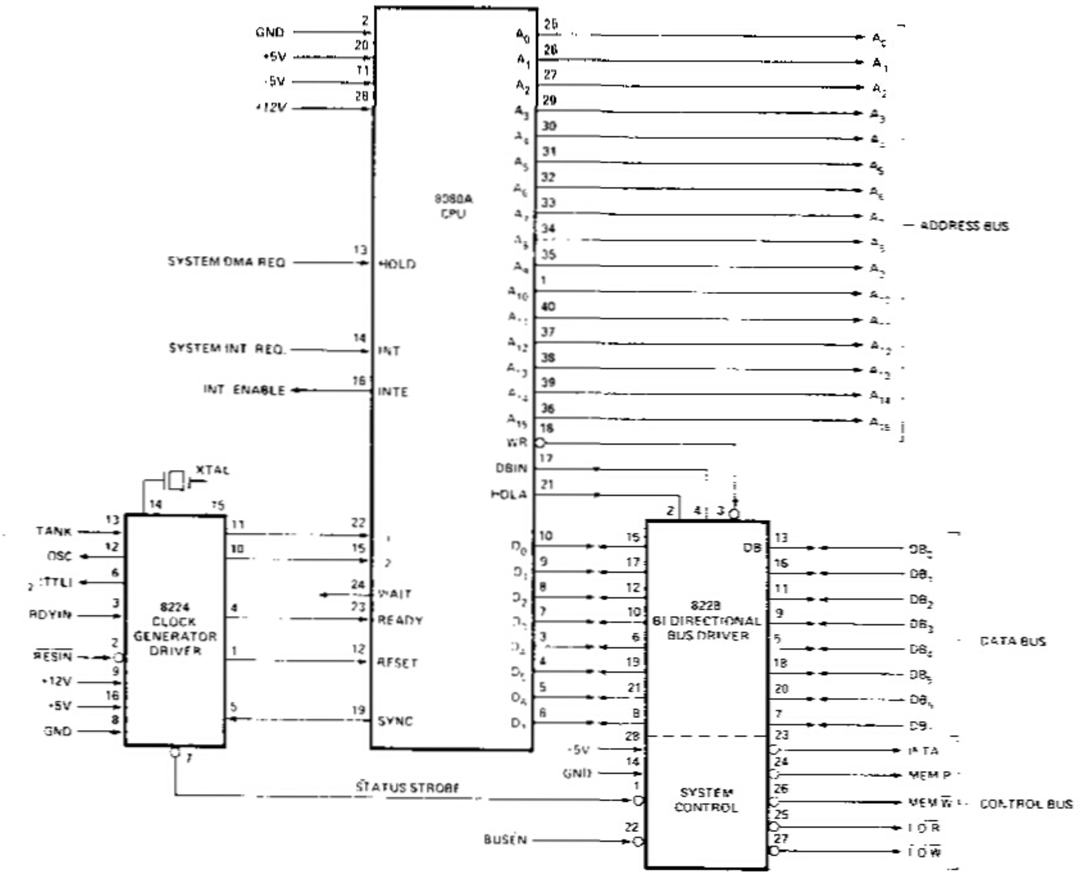
TEST CONDITIONS: $V_{BIAS}$ = 2.5V, $V_{CC}$=5.0V, $T_A$ = 25°C, f = 1MHz.

Note 2: For $D_0$-$D_7$: $R_1$ = 4KΩ, $R_2$ = ∞Ω, $C_L$ = 25pF. For all other outputs: $R_1$ = 500Ω, $R_2$ = 1KΩ, $C_L$ = 100pF.




INTA Test Circuit (for RST 7)




8080A CPU Standard Interface

# intel® Silicon Gate MOS 8080A

## SINGLE CHIP 8-BIT N-CHANNEL MICROPROCESSOR

*The 8080A is functionally and electrically compatible with the Intel® 8080.*

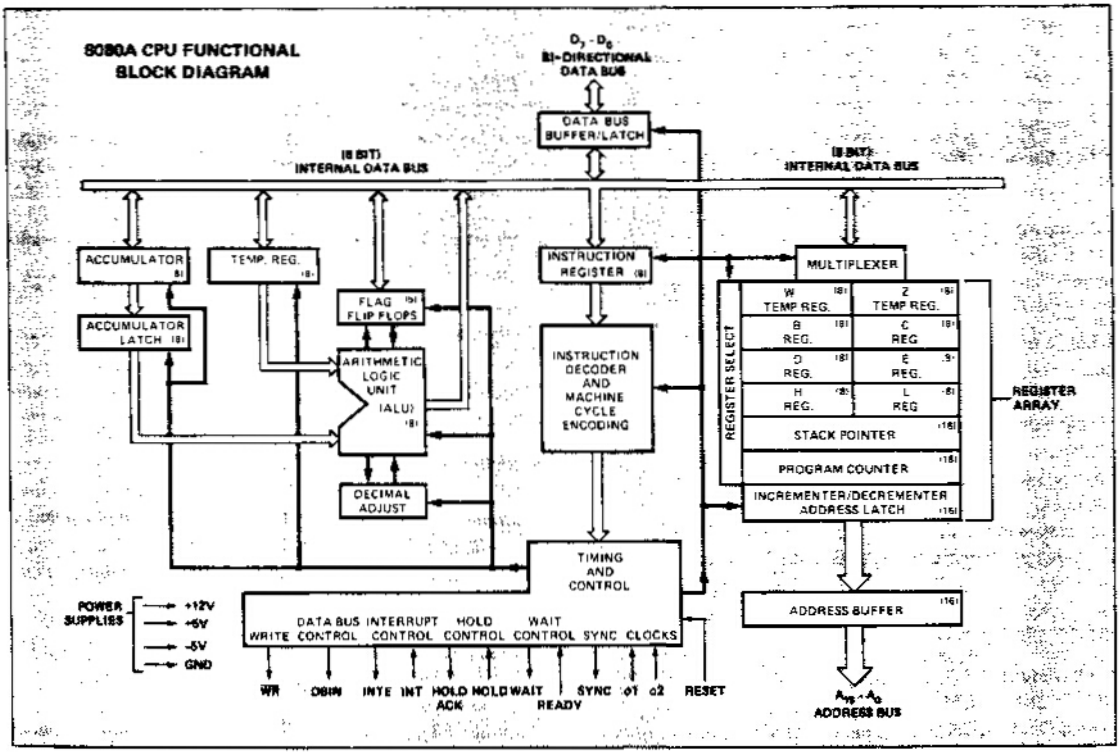
- **TTL Drive Capability**
- ***2 μs Instruction Cycle***
- **Powerful Problem Solving Instruction Set**
- **Six General Purpose Registers and an Accumulator**
- **Sixteen Bit Program Counter for Directly Addressing up to 64K Bytes of Memory**
- **Sixteen Bit Stack Pointer and Stack Manipulation Instructions for Rapid Switching of the Program Environment**
- **Decimal, Binary and Double Precision Arithmetic**
- **Ability to Provide Priority Vectored Interrupts**
- **512 Directly Addressed I/O Ports**

The Intel® 8080A is a complete 8-bit parallel central processing unit (CPU). It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process. This offers the user a high performance solution to control and processing applications.

The 8080A contains six 8-bit general purpose working registers and an accumulator. The six general purpose registers may be addressed individually or in pairs providing both single and double precision operators. Arithmetic and logical instructions set or reset four testable flags. A fifth flag provides decimal arithmetic operation.

The 8080A has an external stack feature wherein any portion of memory may be used as a last in/first out stack to store/retrieve the contents of the accumulator, flags, program counter and all of the six general purpose registers. The sixteen bit stack pointer controls the addressing of this external stack. This stack gives the 8080A the ability to easily handle multiple level priority interrupts by rapidly storing and restoring processor status. It also provides almost unlimited subroutine nesting.

This microprocessor has been designed to simplify systems design. Separate 16-line address and 8-line bi-directional data busses are used to facilitate easy interface to memory and I/O. Signals to control the interface to memory and I/O are provided directly by the 8080A. Ultimate control of the address and data busses resides with the HOLD signal. It provides the ability to suspend processor operation and force the address and data busses into a high impedance state. This permits OR-tying these busses with other controlling devices for (DMA) direct memory access or multi-processor operation.




8080A CPU FUNCTIONAL BLOCK DIAGRAM

## 8080A FUNCTIONAL PIN DEFINITION

The following describes the function of all of the 8080A I/O pins. Several of the descriptions refer to internal timing periods.

**$A_{15}$-$A_0$ (output three-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 three-state)**

DATA BUS; the data bus provides bi-directional communication between the CPU, memory, and I/O devices for instructions and data transfers. Also, during the first clock cycle of each machine cycle, the 8080A outputs a status word on the data bus that describes the current machine cycle. $D_0$ is the least significant bit.

**SYNC (output)**

SYNCHRONIZING SIGNAL; the SYNC pin provides a signal to indicate the beginning of each machine cycle.

**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 8080A data bus from memory or I/O.

**READY (input)**

READY; the READY signal indicates to the 8080A that valid memory or input data is available on the 8080A data bus. This signal is used to synchronize the CPU with slower memory or I/O devices. If after sending an address out the 8080A does not receive a READY input, the 8080A will enter a WAIT state for as long as the READY line is low. READY can also be used to single step the CPU.

**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 low ($\overline{WR}$ = 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 8080A address and data bus as soon as the 8080A 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.

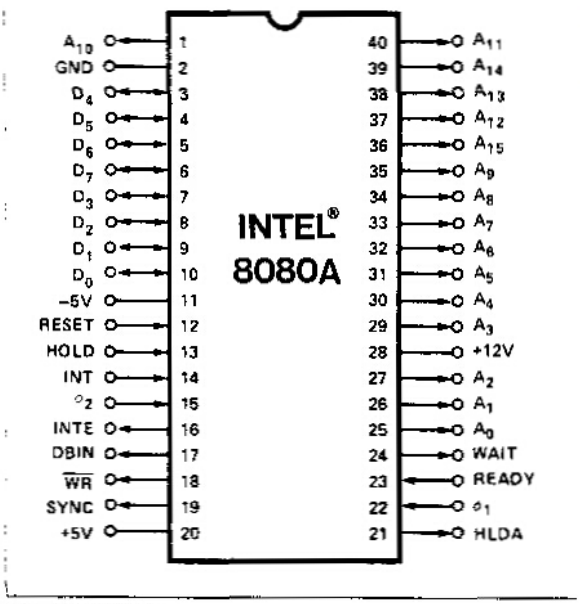
As a result of entering the HOLD state the CPU ADDRESS BUS ($A_{15}$-$A_0$) and DATA BUS ($D_7$-$D_0$) will be in their high impedance state. The CPU acknowledges its state with the HOLD ACKNOWLEDGE (HLDA) pin.

**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 $\phi$ and high impedance occurs after the rising edge of $\phi_2$.



Pin Configuration

**INTE (output)**

INTERRUPT ENABLE; indicates the content of the internal inte rupt enable flip/flop. This flip/flop may be set or reset by the Er able and Disable Interrupt instructions and inhibits interrupt from being accepted by the CPU when it is reset. It is autc matically reset (disabling further interrupts) at time T1 of the ir struction fetch cycle (M1) when an interrupt is accepted and also reset by the RESET signal.

**INT (input)**

INTERRUPT REQUEST; the CPU recognizes an interrupt re quest on this line at the end of the current instruction or whil halted. If the CPU is in the HOLD state or if the Interrupt Enabl flip/flop is reset it will not honor the request.

**RESET (input)(1)**

RESET; while the RESET signal is activated, the content of th program counter is cleared. After RESET, the program will star at location 0 in memory. The INTE and HLDA flip/flops are alsc reset. Note that the flags, accumulator, stack pointer, and register are not cleared.

| | |
|---|---|
| $V_{SS}$ | Ground Reference. |
| $V_{DD}$ | +12 ± 5% Volts. |
| $V_{CC}$ | +5 ± 5% Volts. |
| $V_{BB}$ | -5 ±5% Volts (substrate bias). |
| $\phi_1, \phi_2$ | 2 externally supplied clock phases. (non TTL compatible |

## 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 $V_{BB}$ | -0.3V to +20V |
| $V_{CC}$, $V_{DD}$ and $V_{SS}$ With Respect to $V_{BB}$ | -0.3V to +20V |
| *Power Dissipation* | 1.5W |

***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 Condition |
|---|---|---|---|---|---|---|
| $V_{ILC}$ | Clock Input Low Voltage | $V_{SS}-1$ | | $V_{SS}+0.8$ | V | |
| $V_{IHC}$ | Clock Input High Voltage | 9.0 | | $V_{DD}+1$ | V | |
| $V_{IL}$ | Input Low Voltage | $V_{SS}-1$ | | $V_{SS}+0.8$ | V | |
| $V_{IH}$ | Input High Voltage | 3.3 | | $V_{CC}+1$ | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.9mA on all outputs, |
| $V_{OH}$ | Output High Voltage | 3.7 | | | V | $I_{OH}$ = −150μA. |
| $I_{DD\,(AV)}$ | Avg. Power Supply Current ($V_{DD}$) | | 40 | 70 | mA | Operation |
| $I_{CC\,(AV)}$ | Avg. Power Supply Current ($V_{CC}$) | | 60 | 80 | mA | $T_{CY}$ = .48 μsec |
| $I_{BB\,(AV)}$ | Avg. Power Supply Current ($V_{BB}$) | | .01 | 1 | mA | |
| $I_{IL}$ | Input Leakage | | | ±10 | μA | $V_{SS} \leq V_{IN} \leq V_{CC}$ |
| $I_{CL}$ | Clock Leakage | | | ±10 | μA | $V_{SS} \leq V_{CLOCK} \leq V_{DD}$ |
| $I_{DL}$ [2] | Data Bus Leakage in Input Mode | | | -100<br>-2.0 | μA<br>mA | $V_{SS} \leq V_{IN} \leq V_{SS}+0.8V$<br>$V_{SS}+0.8V \leq V_{IN} \leq V_{CC}$ |
| $I_{FL}$ | Address and Data Bus Leakage During HOLD | | | +10<br>-100 | μA | $V_{ADDR/DATA} = V_{CC}$<br>$V_{ADDR/DATA} = V_{SS} + 0.45V$ |

## CAPACITANCE

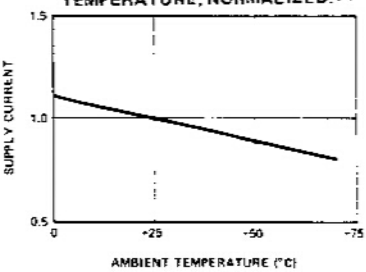
$T_A$ = 25°C  $V_{CC} = V_{DD} = V_{SS}$ = 0V, $V_{BB}$ = -5V

| Symbol | Parameter | Typ. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $C_\phi$ | Clock Capacitance | 17 | 25 | pf | $f_c$ = 1 MHz |
| $C_{IN}$ | Input Capacitance | 6 | 10 | pf | Unmeasured Pins |
| $C_{OUT}$ | Output Capacitance | 10 | 20 | pf | Returned to $V_{SS}$ |

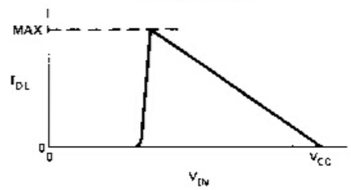
NOTES:

1. The RESET signal must be active for a minimum of 3 clock cycles.
2. When DBIN is high and $V_{IN} > V_{IH}$ an internal active pull up will be switched onto the Data Bus.
3. $\Delta I$ supply / $\Delta T_A$ = -0.45%/°C.

TYPICAL SUPPLY CURRENT VS. TEMPERATURE, NORMALIZED.[3]




DATA BUS CHARACTERISTIC DURING DBIN



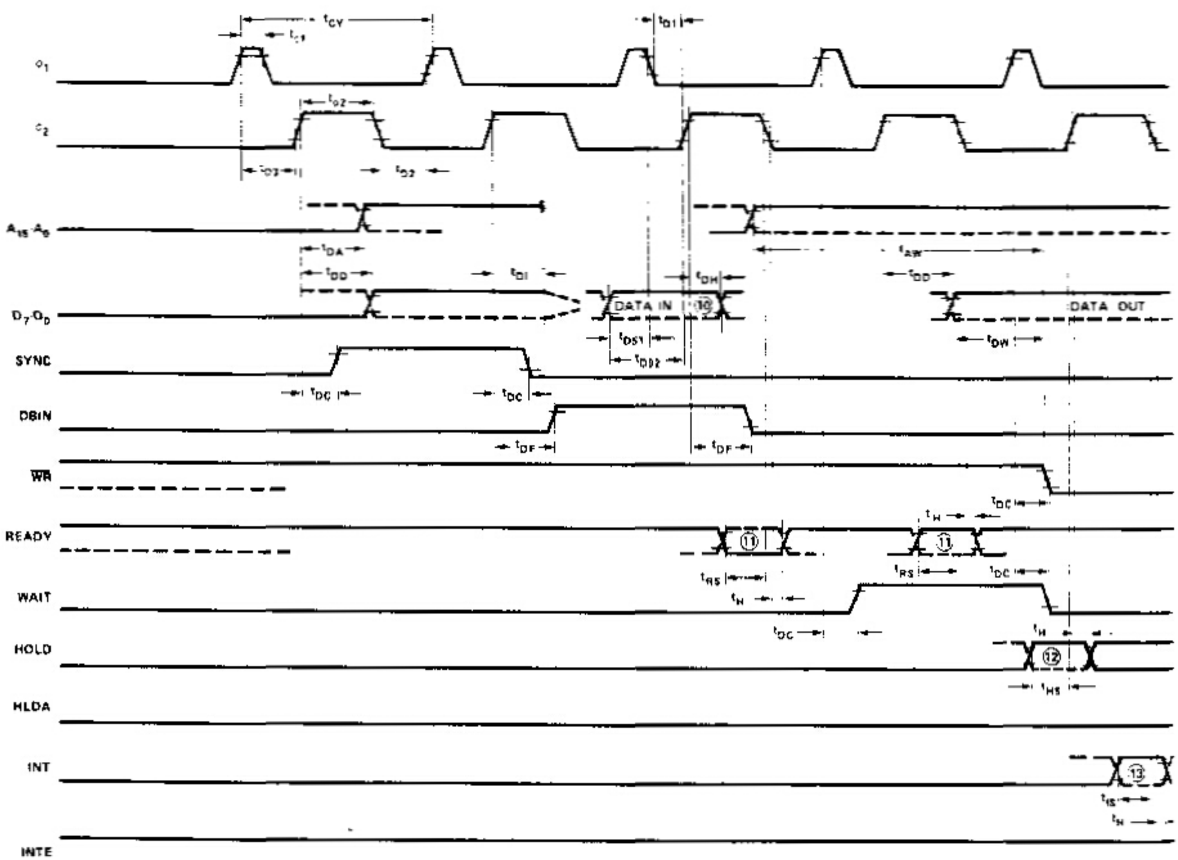
## A.C. CHARACTERISTICS

$T_A$ = 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. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{CY}$ [3] | Clock Period | 0.48 | 2.0 | μsec | |
| $t_r$, $t_f$ | Clock Rise and Fall Time | 0 | 50 | nsec | |
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | 60 | | nsec | |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | 220 | | nsec | |
| $t_{D1}$ | Delay $\phi_1$ to $\phi_2$ | 0 | | nsec | |
| $t_{D2}$ | Delay $\phi_2$ to $\phi_1$ | 70 | | nsec | |
| $t_{D3}$ | Delay $\phi_1$ to $\phi_2$ Leading Edges | 80 | | nsec | |
| $t_{DA}$ [2] | Address Output Delay From $\phi_2$ | | 200 | nsec | $C_L$ = 100pf |
| $t_{DD}$ [2] | Data Output Delay From $\phi_2$ | | 220 | nsec | $C_L$ = 100pf |
| $t_{DC}$ [2] | Signal Output Delay From $\phi_1$, or $\phi_2$ (SYNC, $\overline{WR}$,WAIT, HLDA) | | 120 | nsec | $C_L$ = 50pf |
| $t_{DF}$ [2] | DBIN Delay From $\phi_2$ | 25 | 140 | nsec | $C_L$ = 50pf |
| $t_{DI}$ [1] | Delay for Input Bus to Enter Input Mode | | $t_{DF}$ | nsec | |
| $t_{DS1}$ | Data Setup Time During $\phi_1$ and DBIN | 30 | | nsec | |

## TIMING WAVEFORMS [14]

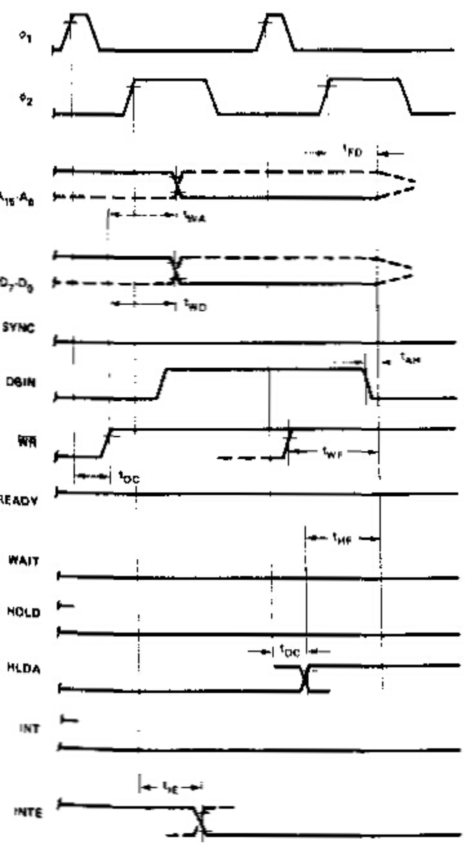
(Note: Timing measurements are made at the following reference voltages: CLOCK "1" = 8.0V "0" = 1.0V; INPUTS "1" = 3.3V, "0" = 0.8V; OUTPUTS "1" = 2.0V, "0" = 0.8V.)



### A.C. CHARACTERISTICS (Continued)

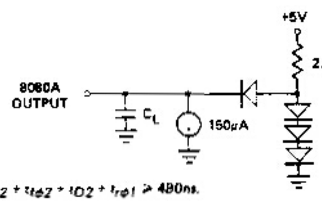
$T_A$ = 0°C to 70°C, $V_{DD}$ = +12V ± 5%, $V_{CC}$ = +5V ± 5%, $V_{BB}$ = -5V ± 5%, $V_{SS}$ = 0V, Unless Otherwise Noted

| Symbol | Parameter | Min. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{DS2}$ | Data Setup Time to $\phi_2$ During DBIN | 150 | | nsec | $C_L$ = 50pf |
| $t_{DH}$ [1] | Data Hold Time From $\phi_2$ During DBIN | [1] | | nsec | |
| $t_{IE}$ [2] | INTE Output Delay From $\phi_2$ | | 200 | nsec | |
| $t_{RS}$ | READY Setup Time During $\phi_2$ | 120 | | nsec | |
| $t_{HS}$ | HOLD Setup Time to $\phi_2$ | 140 | | nsec | |
| $t_{IS}$ | INT Setup Time During $\phi_2$ (During $\phi_1$ in Halt Mode) | 120 | | nsec | |
| $t_H$ | Hold Time From $\phi_2$ (READY, INT, HOLD) | 0 | | nsec | |
| $t_{FD}$ | Delay to Float During Hold (Address and Data Bus) | | 120 | nsec | |
| $t_{AW}$ [2] | Address Stable Prior to $\overline{WR}$ | [5] | | nsec | $C_L$ = 100pf: Address, Data<br>$C_L$ = 50pf: $\overline{WR}$, HLDA, DBIN |
| $t_{DW}$ [2] | Output Data Stable Prior to $\overline{WR}$ | [6] | | nsec | |
| $t_{WD}$ [2] | Output Data Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{WA}$ [2] | Address Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{HF}$ [2] | HLDA to Float Delay | [8] | | nsec | |
| $t_{WF}$ [2] | $\overline{WR}$ to Float Delay | [9] | | nsec | |
| $t_{AH}$ [2] | Address Hold Time After DBIN During HLDA | -20 | | nsec | |



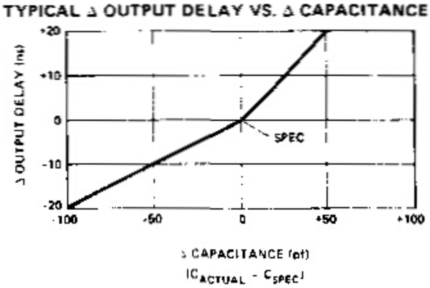

NOTES.

1. Data input should be enabled with DBIN status. No bus conflict can then occur and data hold time is assured. $t_{DH}$ = 50 ns or $t_{DF}$, whichever is less
2. Load Circuit.




3. $t_{CY} = t_{D3} + t_{r\phi2} + t_{\phi2} + t_{f\phi2} + t_{D2} + t_{r\phi1} \geq 480ns$.

TYPICAL Δ OUTPUT DELAY VS. Δ CAPACITANCE




4. The following are relevant when interfacing the 8080A to devices having $V_{IH}$ = 3.3V:
   a) Maximum output rise time from .8V to 3.3V = 100ns @ $C_L$ = SPEC.
   b) Output delay when measured to 3.0V = SPEC +60ns @ $C_L$ = SPEC.
   c) If $C_L \neq$ SPEC, add .6ns/pF if $C_L > C_{SPEC}$, subtract .3ns/pF (from modified delay) if $C_L < C_{SPEC}$.
5. $t_{AW} = 2\,t_{CY} - t_{D3} - t_{r\phi2} - 140nsec$
6. $t_{DW} = t_{CY} - t_{D3} - t_{r\phi2} - 170nsec$.
7. If not HLDA, $t_{WD} = t_{WA} = t_{D3} + t_{r\phi2} + 10ns$. If HLDA, $t_{WD} = t_{WA} = t_{WF}$.
8. $t_{HF} = t_{D3} + t_{r\phi2} - 50ns$.
9. $t_{WF} = t_{D3} + t_{r\phi2} - 10ns$
10. Data in must be stable for this period during DBIN $\cdot T_3$. Both $t_{DS1}$ and $t_{DS2}$ must be satisfied.
11. Ready signal must be stable for this period during $T_2$ or $T_W$. (Must be externally synchronized.)
12. Hold signal must be stable for this period during $T_2$ or $T_W$ when entering hold mode, and during $T_3$, $T_4$, $T_5$ and $T_{WH}$ when in hold mode. (External synchronization is not required.)
13. Interrupt signal must be stable during this period of the last clock cycle of any instruction in order to be recognized on the following instruction. (External synchronization is not required.)
14. This timing diagram shows timing relationships only; it does not represent any specific machine cycle.

## INSTRUCTION SET

The accumulator group instructions include arithmetic and logical operators with direct, indirect, and immediate addressing modes.

Move, load, and store instruction groups provide the ability to move either 8 or 16 bits of data between memory, the six working registers and the accumulator using direct, indirect, and immediate addressing modes.

The ability to branch to different portions of the program is provided with jump, jump conditional, and computed jumps. Also the ability to call to and return from subroutines is provided both conditionally and unconditionally. The RESTART (or single byte call instruction) is useful for interrupt vector operation.

Double precision operators such as stack manipulation and double add instructions extend both the arithmetic and interrupt handling capability of the 8080A. The ability to increment and decrement memory, the six general registers and the accumulator is provided as well as extended increment and decrement instructions to operate on the register pairs and stack pointer. Further capability is provided by the ability to rotate the accumulator left or right through or around the carry bit.

Input and output may be accomplished using memory addresses as I/O ports or the directly addressed I/O provided for in the 8080A instruction set.

The following special instruction group completes the 8080A instruction set: the NOP instruction, HALT to stop processor execution and the DAA instructions provide decimal arithmetic capability. STC allows the carry flag to be directly set, and the CMC instruction allows it to be complemented. CMA complements the contents of the accumulator and XCHG exchanges the contents of two 16-bit register pairs directly.

### Data and Instruction Formats

Data in the 8080A 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.

| Format | Byte | | TYPICAL INSTRUCTIONS |
|---|---|---|---|
| One Byte Instructions | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | Register to register, memory reference, arithmetic or logical, rotate, return, push, pop, enable or disable Interrupt instructions |
| Two Byte Instructions | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | |
| | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | 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 | Jump, call or direct load and store instructions |
| | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | 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 | |

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

## INSTRUCTION SET

### Summary of Processor Instructions

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| MOV $_{r1,r2}$ | Move register to register | 0 | 1 | D | D | D | S | S | S | 5 |
| MOV M, r | Move register to memory | 0 | 1 | 1 | 1 | 0 | S | S | S | 7 |
| MOV r, M | Move memory to register | 0 | 1 | D | D | D | 1 | 1 | 0 | 7 |
| HLT | Halt | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| MVI r | Move immediate register | 0 | 0 | D | D | D | 1 | 1 | 0 | 7 |
| MVI M | Move immediate memory | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 10 |
| INR r | Increment register | 0 | 0 | D | D | D | 1 | 0 | 0 | 5 |
| DCR r | Decrement register | 0 | 0 | D | D | D | 1 | 0 | 1 | 5 |
| INR M | Increment memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 10 |
| DCR M | Decrement memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 10 |
| ADD r | Add register to A | 1 | 0 | 0 | 0 | 0 | S | S | S | 4 |
| ADC r | Add register to A with carry | 1 | 0 | 0 | 0 | 1 | S | S | S | 4 |
| SUB r | Subtract register from A | 1 | 0 | 0 | 1 | 0 | S | S | S | 4 |
| SBB r | Subtract register from A with borrow | 1 | 0 | 0 | 1 | 1 | S | S | S | 4 |
| ANA r | And register with A | 1 | 0 | 1 | 0 | 0 | S | S | S | 4 |
| XRA r | Exclusive Or register with A | 1 | 0 | 1 | 0 | 1 | S | S | S | 4 |
| ORA r | Or register with A | 1 | 0 | 1 | 1 | 0 | S | S | S | 4 |
| CMP r | Compare register with A | 1 | 0 | 1 | 1 | 1 | S | S | S | 4 |
| ADD M | Add memory to A | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ADC M | Add memory to A with carry | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUB M | Subtract memory from A | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBB M | Subtract memory from A with borrow | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANA M | And memory with A | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRA M | Exclusive Or memory with A | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORA M | Or memory with A | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CMP M | Compare memory with A | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| ADI | Add immediate to A | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ACI | Add immediate to A with carry | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUI | Subtract immediate from A | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBI | Subtract immediate from A with borrow | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANI | And immediate with A | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRI | Exclusive Or immediate with A | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORI | Or immediate with A | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CPI | Compare immediate with A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| RLC | Rotate A left | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 |
| RRC | Rotate A right | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 4 |
| RAL | Rotate A left through carry | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| RAR | Rotate A right through carry | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 |
| JMP | Jump unconditional | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 10 |
| JC | Jump on carry | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 10 |
| JNC | Jump on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 10 |
| JZ | Jump on zero | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 10 |
| JNZ | Jump on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 10 |
| JP | Jump on positive | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 10 |
| JM | Jump on minus | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 10 |
| JPE | Jump on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 10 |
| JPO | Jump on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 10 |
| CALL | Call unconditional | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 17 |
| CC | Call on carry | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CNC | Call on no carry | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CZ | Call on zero | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CNZ | Call on no zero | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| CP | Call on positive | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CM | Call on minus | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CPE | Call on parity even | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CPO | Call on parity odd | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| RET | Return | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| RC | Return on carry | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RNC | Return on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 5/11 |

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| RZ | Return on zero | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RNZ | Return on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RP | Return on positive | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RM | Return on minus | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RPE | Return on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RPO | Return on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RST | Restart | 1 | 1 | A | A | A | 1 | 1 | 1 | 11 |
| IN | Input | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 10 |
| OUT | Output | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 10 |
| LXI B | Load immediate register Pair B & C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI D | Load immediate register Pair D & E | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| LXI H | Load immediate register Pair H & L | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI SP | Load immediate stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| PUSH B | Push register Pair B & C on stack | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH D | Push register Pair D & E on stack | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 11 |
| PUSH H | Push register Pair H & L on stack | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH PSW | Push A and Flags on stack | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 11 |
| POP B | Pop register pair B & C off stack | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP D | Pop register pair D & E off stack | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| POP H | Pop register pair H & L off stack | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP PSW | Pop A and Flags off stack | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| STA | Store A direct | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 13 |
| LDA | Load A direct | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 13 |
| XCHG | Exchange D & E, H & L Registers | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 4 |
| XTHL | Exchange top of stack, H & L | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 18 |
| SPHL | H & L to stack pointer | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 5 |
| PCHL | H & L to program counter | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 5 |
| DAD B | Add B & C to H & L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD D | Add D & E to H & L | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 10 |
| DAD H | Add H & L to H & L | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD SP | Add stack pointer to H & L | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 10 |
| STAX B | Store A indirect | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7 |
| STAX D | Store A indirect | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 7 |
| LDAX B | Load A indirect | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 |
| LDAX D | Load A indirect | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 7 |
| INX B | Increment B & C registers | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX D | Increment D & E registers | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 5 |
| INX H | Increment H & L registers | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX SP | Increment stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 5 |
| DCX B | Decrement B & C | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX D | Decrement D & E | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 5 |
| DCX H | Decrement H & L | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX SP | Decrement stack pointer | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 5 |
| CMA | Complement A | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 4 |
| STC | Set carry | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 4 |
| CMC | Complement carry | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| DAA | Decimal adjust A | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 4 |
| SHLD | Store H & L direct | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 16 |
| LHLD | Load H & L direct | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 16 |
| EI | Enable Interrupts | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 |
| DI | Disable interrupt | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| NOP | No-operation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |

NOTES: 1. DDD or SSS – 000 B – 001 C – 010 D – 011 E – 100 H – 101 L – 110 Memory – 111 A.

2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.

# Silicon Gate MOS 8080A-1

## SINGLE CHIP 8-BIT N-CHANNEL MICROPROCESSOR

*The 8080A is functionally and electrically compatible with the Intel® 8080.*

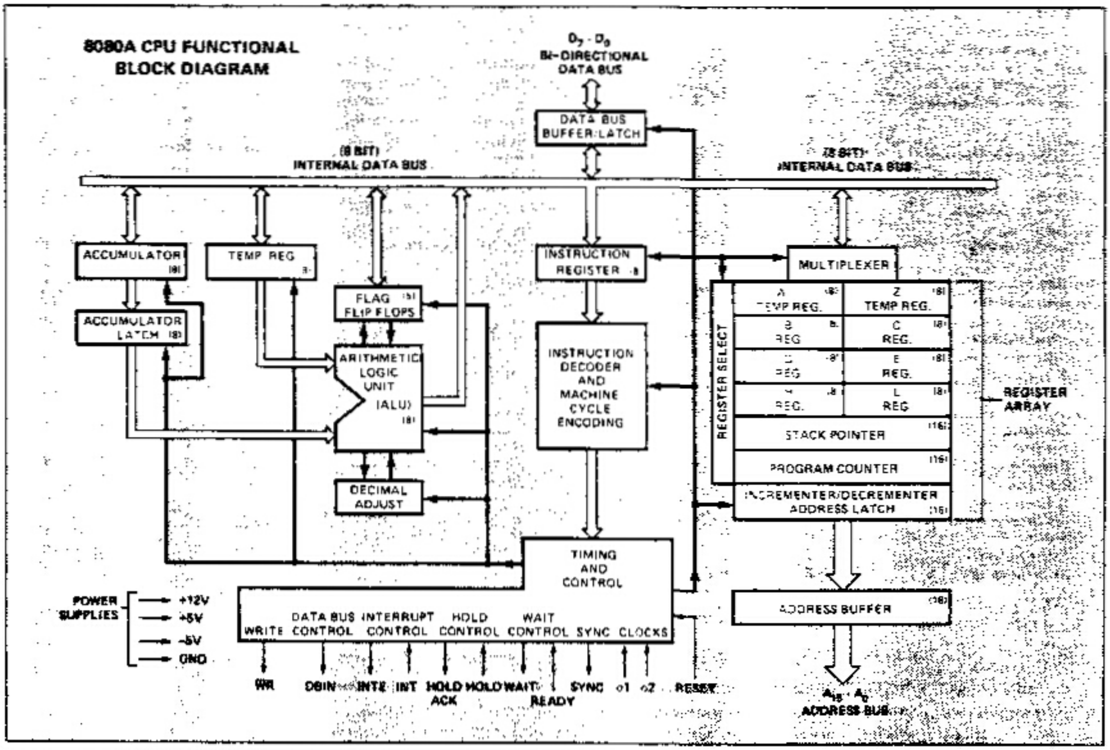
- **TTL Drive Capability**
- **1.3 μs Instruction Cycle**
- **Powerful Problem Solving Instruction Set**
- **Six General Purpose Registers and an Accumulator**
- **Sixteen Bit Program Counter for Directly Addressing up to 64K Bytes of Memory**
- **Sixteen Bit Stack Pointer and Stack Manipulation Instructions for Rapid Switching of the Program Environment**
- **Decimal, Binary and Double Precision Arithmetic**
- **Ability to Provide Priority Vectored Interrupts**
- **512 Directly Addressed I/O Ports**

The Intel® 8080A is a complete 8-bit parallel central processing unit (CPU). It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process. This offers the user a high performance solution to control and processing applications.

The 8080A contains six 8-bit general purpose working registers and an accumulator. The six general purpose registers may be addressed individually or in pairs providing both single and double precision operators. Arithmetic and logical instructions set or reset four testable flags. A fifth flag provides decimal arithmetic operation.

The 8080A has an external stack feature wherein any portion of memory may be used as a last in/first out stack to store/retrieve the contents of the accumulator, flags, program counter and all of the six general purpose registers. The sixteen bit stack pointer controls the addressing of this external stack. This stack gives the 8080A the ability to easily handle multiple level priority interrupts by rapidly storing and restoring processor status. It also provides almost unlimited subroutine nesting.

This microprocessor has been designed to simplify systems design. Separate 16-line address and 8-line bi-directional data busses are used to facilitate easy interface to memory and I/O. Signals to control the interface to memory and I/O are provided directly by the 8080A. Ultimate control of the address and data busses resides with the HOLD signal. It provides the ability to suspend processor operation and force the address and data busses into a high impedance state. This permits OR-tying these busses with other controlling devices for (DMA) direct memory access or multi-processor operation.



## 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 $V_{BB}$ | -0.3V to +20V |
| $V_{CC}$, $V_{DD}$ and $V_{SS}$ With Respect to $V_{BB}$ | -0.3V to +20V |
| Power Dissipation | 1.5W |

**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 Condition |
|---|---|---|---|---|---|---|
| $V_{ILC}$ | Clock Input Low Voltage | $V_{SS}-1$ | | $V_{SS}$+0.8 | V | |
| $V_{IHC}$ | Clock Input High Voltage | 9.0 | | $V_{DD}$+1 | V | |
| $V_{IL}$ | Input Low Voltage | $V_{SS}-1$ | | $V_{SS}$+0.8 | V | |
| $V_{IH}$ | Input High Voltage | 3.3 | | $V_{CC}$+1 | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.9mA on all outputs, |
| $V_{OH}$ | Output High Voltage | 3.7 | | | V | $I_{OH}$ = 150μA. |
| $I_{DD(AV)}$ | Avg. Power Supply Current ($V_{DD}$) | | 40 | 70 | mA | Operation |
| $I_{CC(AV)}$ | Avg. Power Supply Current ($V_{CC}$) | | 60 | 80 | mA | $T_{CY}$ = .32μsec |
| $I_{BB(AV)}$ | Avg. Power Supply Current ($V_{BB}$) | | .01 | 1 | mA | |
| $I_{IL}$ | Input Leakage | | | ±10 | μA | $V_{SS} \leq V_{IN} \leq V_{CC}$ |
| $I_{CL}$ | Clock Leakage | | | ±10 | μA | $V_{SS} \leq V_{CLOCK} \leq V_{DD}$ |
| $I_{DL}$ [2] | Data Bus Leakage in Input Mode | | | -100<br>-2.0 | μA<br>mA | $V_{SS} \leq V_{IN} \leq V_{SS}+0.8V$<br>$V_{SS}+0.8V \leq V_{IN} \leq V_{CC}$ |
| $I_{FL}$ | Address and Data Bus Leakage During HOLD | | | +10<br>-100 | μA | $V_{ADDR/DATA} = V_{CC}$<br>$V_{ADDR/DATA} = V_{SS} + 0.45V$ |

## CAPACITANCE

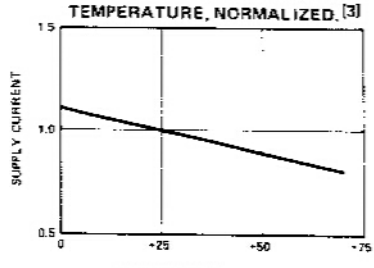
$T_A$ = 25°C $V_{CC} = V_{DD} = V_{SS}$ = 0V, $V_{BB}$ = -5V

| Symbol | Parameter | Typ. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $C_\phi$ | Clock Capacitance | 17 | 25 | pf | $f_c$ = 1 MHz |
| $C_{IN}$ | Input Capacitance | 6 | 10 | pf | Unmeasured Pins |
| $C_{OUT}$ | Output Capacitance | 10 | 20 | pf | Returned to $V_{SS}$ |

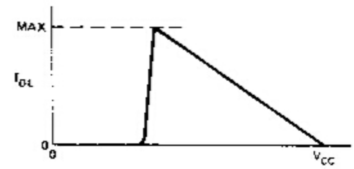
NOTES:

1. The RESET signal must be active for a minimum of 3 clock cycles.
2. When DBIN is high and $V_{IN} > V_{IH}$ an internal active pull up will be switched onto the Data Bus.
3. $\Delta I$ supply / $\Delta T_A$ = -0.45%/°C.

TYPICAL SUPPLY CURRENT VS. TEMPERATURE, NORMALIZED.[3]




DATA BUS CHARACTERISTIC DURING DBIN



## A.C. CHARACTERISTICS

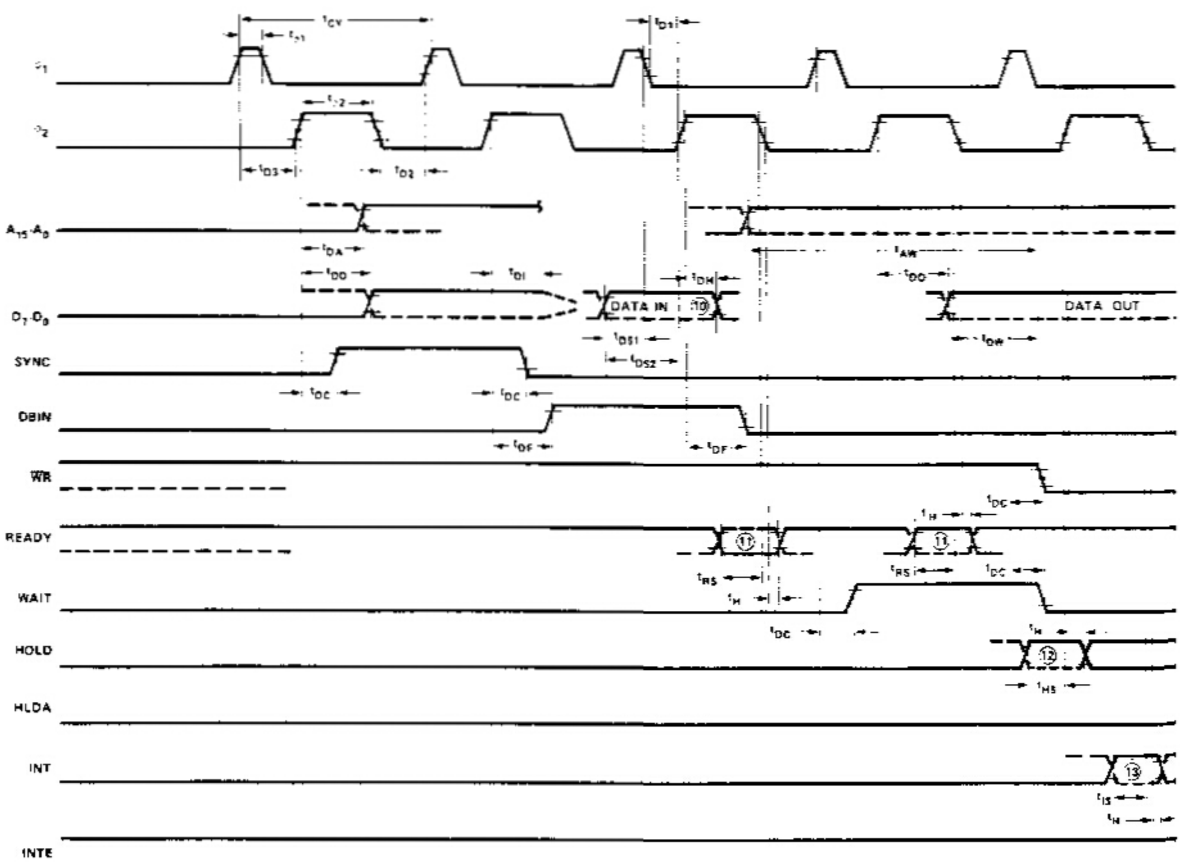
*CAUTION: When operating the 8080A-1 at or near full speed, care must be taken to assure precise timing compatibility between 8080A-1, 8224 and 8*

$T_A$ = 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. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{CY}$ [3] | Clock Period | .32 | 2.0 | μsec | |
| $t_r$, $t_f$ | Clock Rise and Fall Time | 0 | 25 | nsec | |
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | 50 | | nsec | |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | 145 | | nsec | |
| $t_{D1}$ | Delay $\phi_1$ to $\phi_2$ | 0 | | nsec | |
| $t_{D2}$ | Delay $\phi_2$ to $\phi_1$ | 60 | | nsec | |
| $t_{D3}$ | Delay $\phi_1$ to $\phi_2$ Leading Edges | 60 | | nsec | |
| $t_{DA}$ [2] | Address Output Delay From $\phi_2$ | | 150 | nsec | $C_L$ = 50pf |
| $t_{DD}$ [2] | Data Output Delay From $\phi_2$ | | 180 | nsec | $C_L$ = 50pf |
| $t_{DC}$ [2] | Signal Output Delay From $\phi_1$, or $\phi_2$ (SYNC, $\overline{WR}$, WAIT, HLDA) | | 110 | nsec | $C_L$ = 50pf |
| $t_{DF}$ [2] | DBIN Delay From $\phi_2$ | 25 | 130 | nsec | $C_L$ = 50pf |
| $t_{DI}$ [1] | Delay for Input Bus to Enter Input Mode | | $t_{DF}$ | nsec | |
| $t_{DS1}$ | Data Setup Time During $\phi_1$ and DBIN | 10 | | nsec | |

## TIMING WAVEFORMS [14]

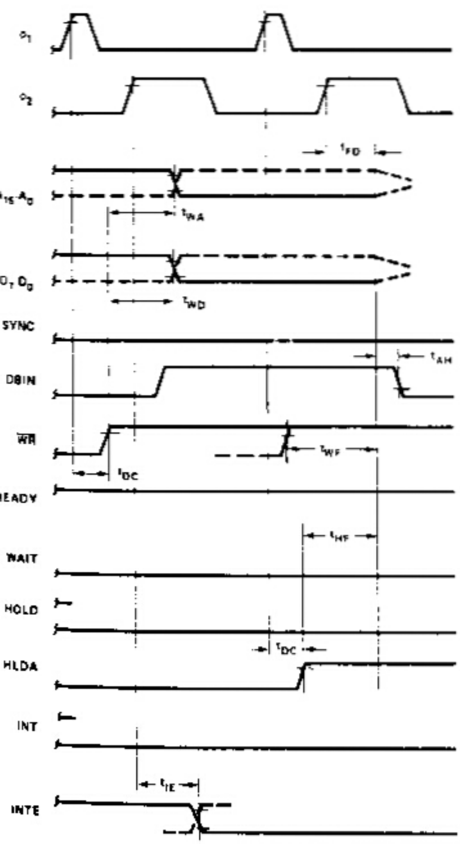
(Note: Timing measurements are made at the following reference voltages: CLOCK "1" = 8.0V "0" = 1.0V; INPUTS "1" = 3.3V, "0" = 0.8V; OUTPUTS "1" = 2.0V, "0" = 0.8V.)



## A.C. CHARACTERISTICS (Continued)

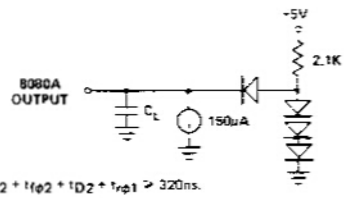
$T_A = 0°C$ to $70°C$, $V_{DD} = +12V \pm 5\%$, $V_{CC} = +5V \pm 5\%$, $V_{BB} = -5V \pm 5\%$, $V_{SS} = 0V$, Unless Otherwise Noted

| Symbol | Parameter | Min. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{DS2}$ | Data Setup Time to $\phi_2$ During DBIN | 120 | | nsec | $C_L = 50pf$ |
| $t_{DH}$ [1] | Data Hold Time From $\phi_2$ During DBIN | [1] | | nsec | |
| $t_{IE}$ [2] | INTE Output Delay From $\phi_2$ | | 200 | nsec | |
| $t_{RS}$ | READY Setup Time During $\phi_2$ | 90 | | nsec | |
| $t_{HS}$ | HOLD Setup Time to $\phi_2$ | 120 | | nsec | |
| $t_{IS}$ | INT Setup Time During $\phi_2$ (During $\phi_1$ in Halt Mode) | 100 | | nsec | |
| $t_H$ | Hold Time From $\phi_2$ (READY, INT, HOLD) | 0 | | nsec | |
| $t_{FD}$ | Delay to Float During Hold (Address and Data Bus) | | 120 | nsec | |
| $t_{AW}$ [2] | Address Stable Prior to $\overline{WR}$ | [5] | | nsec | $C_L = 50pf$: Address, Data; $C_L = 50pf$: $\overline{WR}$, HLDA, DBIN |
| $t_{DW}$ [2] | Output Data Stable Prior to $\overline{WR}$ | [6] | | nsec | |
| $t_{WD}$ [2] | Output Data Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{WA}$ [2] | Address Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{HF}$ [2] | HLDA to Float Delay | [8] | | nsec | |
| $t_{WF}$ [2] | $\overline{WR}$ to Float Delay | [9] | | nsec | |
| $t_{AH}$ [2] | Address Hold Time After DBIN During HLDA | -20 | | nsec | |



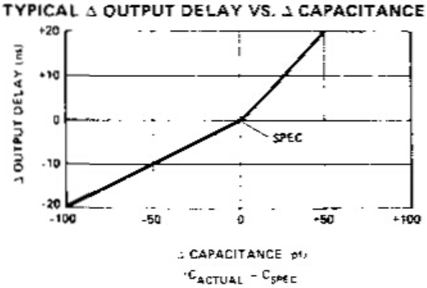

NOTES:

1. Data input should be enabled with DBIN status. No bus conflict can then occur and data hold time is assured. $t_{DH}$ = 50 ns or $t_{DF}$, whichever is less.
2. Load Circuit.




3. $t_{CY} = t_{D3} + t_{r\phi2} + t_{\phi2} + t_{f\phi2} + t_{D2} + t_{r\phi1} \geq 320ns$.

TYPICAL Δ OUTPUT DELAY VS. Δ CAPACITANCE




4. The following are relevant when interfacing the 8080A to devices having $V_{IH} = 3.3V$:
   a) Maximum output rise time from .8V to 3.3V = 100ns @ $C_L$ = SPEC.
   b) Output delay when measured to 3.0V = SPEC +60ns @ $C_L$ = SPEC.
   c) If $C_L \neq$ SPEC, add .6ns/pF if $C_L > C_{SPEC}$, subtract .3ns/pF (from modified delay) if $C_L < C_{SPEC}$.
5. $t_{AW} = 2\,t_{CY} - t_{D3} - t_{r\phi2} - 110nsec$.
6. $t_{DW} = t_{CY} - t_{D3} - t_{r\phi2} - 150nsec$.
7. If not HLDA, $t_{WD} = t_{WA} = t_{D3} + t_{r\phi2} + 10ns$. If HLDA, $t_{WD} = t_{WA} = t_{WF}$.
8. $t_{HF} = t_{D3} + t_{r\phi2} - 50ns$.
9. $t_{WF} = t_{D3} + t_{r\phi2} - 10ns$
10. Data in must be stable for this period during DBIN $\cdot T_3$. Both $t_{DS1}$ and $t_{DS2}$ must be satisfied.
11. Ready signal must be stable for this period during $T_2$ or $T_W$. (Must be externally synchronized.)
12. Hold signal must be stable for this period during $T_2$ or $T_W$ when entering hold mode, and during $T_3$, $T_4$, $T_5$ and $T_{WH}$ when in hold mode. (External synchronization is not required.)
13. Interrupt signal must be stable during this period of the last clock cycle of any instruction in order to be recognized on the following instruction. (External synchronization is not required.)
14. This timing diagram shows timing relationships only; it does not represent any specific machine cycle.

# Silicon Gate MOS 8080A-2

## SINGLE CHIP 8-BIT N-CHANNEL MICROPROCESSOR

*The 8080A is functionally and electrically compatible with the Intel® 8080.*

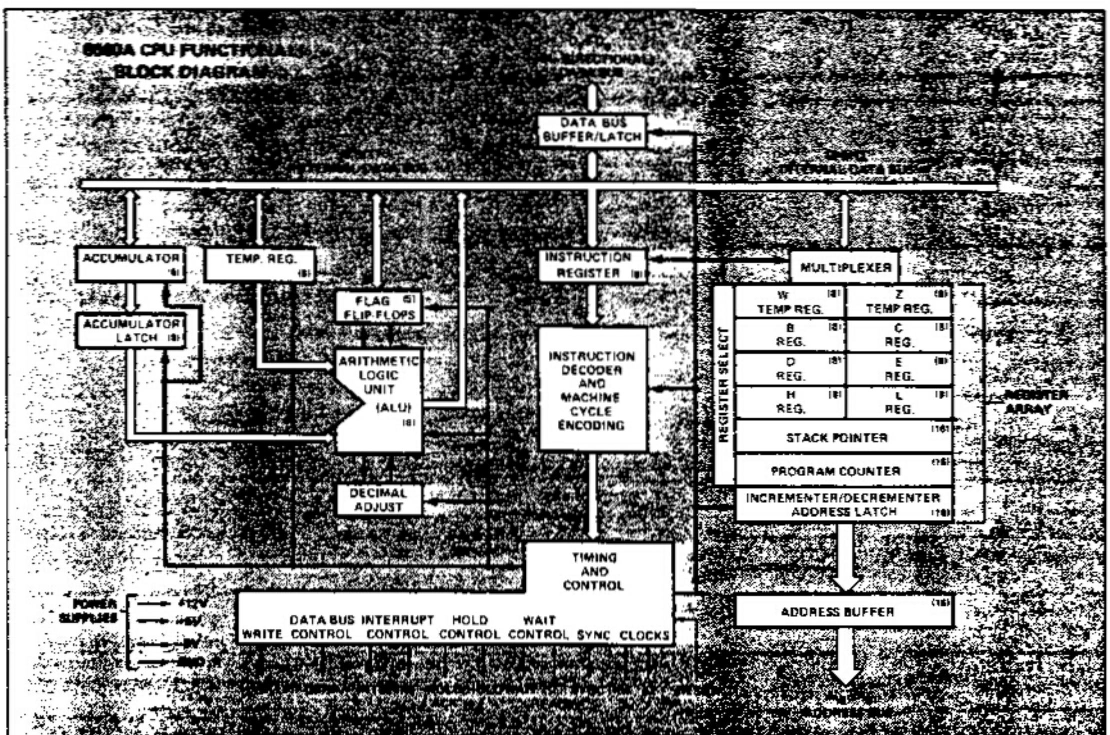
- **TTL Drive Capability**
- **1.5 μs Instruction Cycle**
- **Powerful Problem Solving Instruction Set**
- **Six General Purpose Registers and an Accumulator**
- **Sixteen Bit Program Counter for Directly Addressing up to 64K Bytes of Memory**
- **Sixteen Bit Stack Pointer and Stack Manipulation Instructions for Rapid Switching of the Program Environment**
- **Decimal, Binary and Double Precision Arithmetic**
- **Ability to Provide Priority Vectored Interrupts**
- **512 Directly Addressed I/O Ports**

The Intel® 8080A is a complete 8-bit parallel central processing unit (CPU). It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process. This offers the user a high performance solution to control and processing applications.

The 8080A contains six 8-bit general purpose working registers and an accumulator. The six general purpose registers may be addressed individually or in pairs providing both single and double precision operators. Arithmetic and logical instructions set or reset four testable flags. A fifth flag provides decimal arithmetic operation.

The 8080A has an external stack feature wherein any portion of memory may be used as a last in/first out stack to store/retrieve the contents of the accumulator, flags, program counter and all of the six general purpose registers. The sixteen bit stack pointer controls the addressing of this external stack. This stack gives the 8080A the ability to easily handle multiple level priority interrupts by rapidly storing and restoring processor status. It also provides almost unlimited subroutine nesting.

This microprocessor has been designed to simplify systems design. Separate 16-line address and 8-line bi-directional data busses are used to facilitate easy interface to memory and I/O. Signals to control the interface to memory and I/O are provided directly by the 8080A. Ultimate control of the address and data busses resides with the HOLD signal. It provides the ability to suspend processor operation and force the address and data busses into a high impedance state. This permits OR-tying these busses with other controlling devices for (DMA) direct memory access or multi-processor operation.



## 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 $V_{BB}$ . . . . . . . . . . . . . . . -0.3V to +20V
$V_{CC}$, $V_{DD}$ and $V_{SS}$ With Respect to $V_{BB}$ -0.3V to +20V
Power Dissipation . . . . . . . . . . . . . . . . . . . . . . . . 1.5W

**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 Condition |
|---|---|---|---|---|---|---|
| $V_{ILC}$ | Clock Input Low Voltage | $V_{SS}-1$ | | $V_{SS}+0.8$ | V | |
| $V_{IHC}$ | Clock Input High Voltage | 9.0 | | $V_{DD}+1$ | V | |
| $V_{IL}$ | Input Low Voltage | $V_{SS}-1$ | | $V_{SS}+0.8$ | V | |
| $V_{IH}$ | Input High Voltage | 3.3 | | $V_{CC}+1$ | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.9mA on all outputs, |
| $V_{OH}$ | Output High Voltage | 3.7 | | | V | $I_{OH}$ = 150μA. |
| $I_{DD\,(AV)}$ | Avg. Power Supply Current ($V_{DD}$) | | 40 | 70 | mA | Operation |
| $I_{CC\,(AV)}$ | Avg. Power Supply Current ($V_{CC}$) | | 60 | 80 | mA | $T_{CY}$ = .38μsec |
| $I_{BB\,(AV)}$ | Avg. Power Supply Current ($V_{BB}$) | | .01 | 1 | mA | |
| $I_{IL}$ | Input Leakage | | | ±10 | μA | $V_{SS} \leq V_{IN} \leq V_{CC}$ |
| $I_{CL}$ | Clock Leakage | | | ±10 | μA | $V_{SS} \leq V_{CLOCK} \leq V_{DD}$ |
| $I_{DL}$ [2] | Data Bus Leakage in Input Mode | | | -100<br>-2.0 | μA<br>mA | $V_{SS} \leq V_{IN} \leq V_{SS}+0.8V$<br>$V_{SS}+0.8V \leq V_{IN} \leq V_{CC}$ |
| $I_{FL}$ | Address and Data Bus Leakage During HOLD | | | +10<br>-100 | μA | $V_{ADDR/DATA} = V_{CC}$<br>$V_{ADDR/DATA} = V_{SS} + 0.45V$ |

## CAPACITANCE

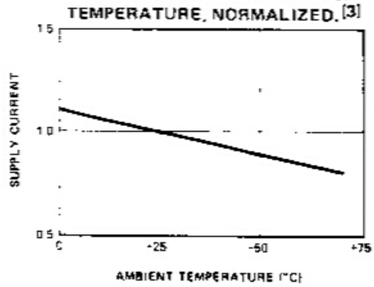
$T_A$ = 25°C $V_{CC} = V_{DD} = V_{SS}$ = 0V, $V_{BB}$ = -5V

| Symbol | Parameter | Typ. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $C_\phi$ | Clock Capacitance | 17 | 25 | pf | $f_c$ = 1 MHz |
| $C_{IN}$ | Input Capacitance | 6 | 10 | pf | Unmeasured Pins |
| $C_{OUT}$ | Output Capacitance | 10 | 20 | pf | Returned to $V_{SS}$ |

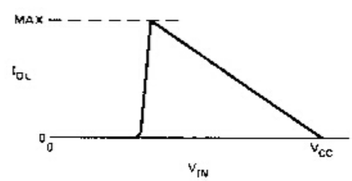
NOTES:

1. The RESET signal must be active for a minimum of 3 clock cycles.
2. When DBIN is high and $V_{IN} > V_{IH}$ an internal active pull up will be switched onto the Data Bus.
3. $\Delta I$ supply / $\Delta T_A$ = -0.45%/°C.

TYPICAL SUPPLY CURRENT VS. TEMPERATURE, NORMALIZED.[3]




DATA BUS CHARACTERISTIC DURING DBIN



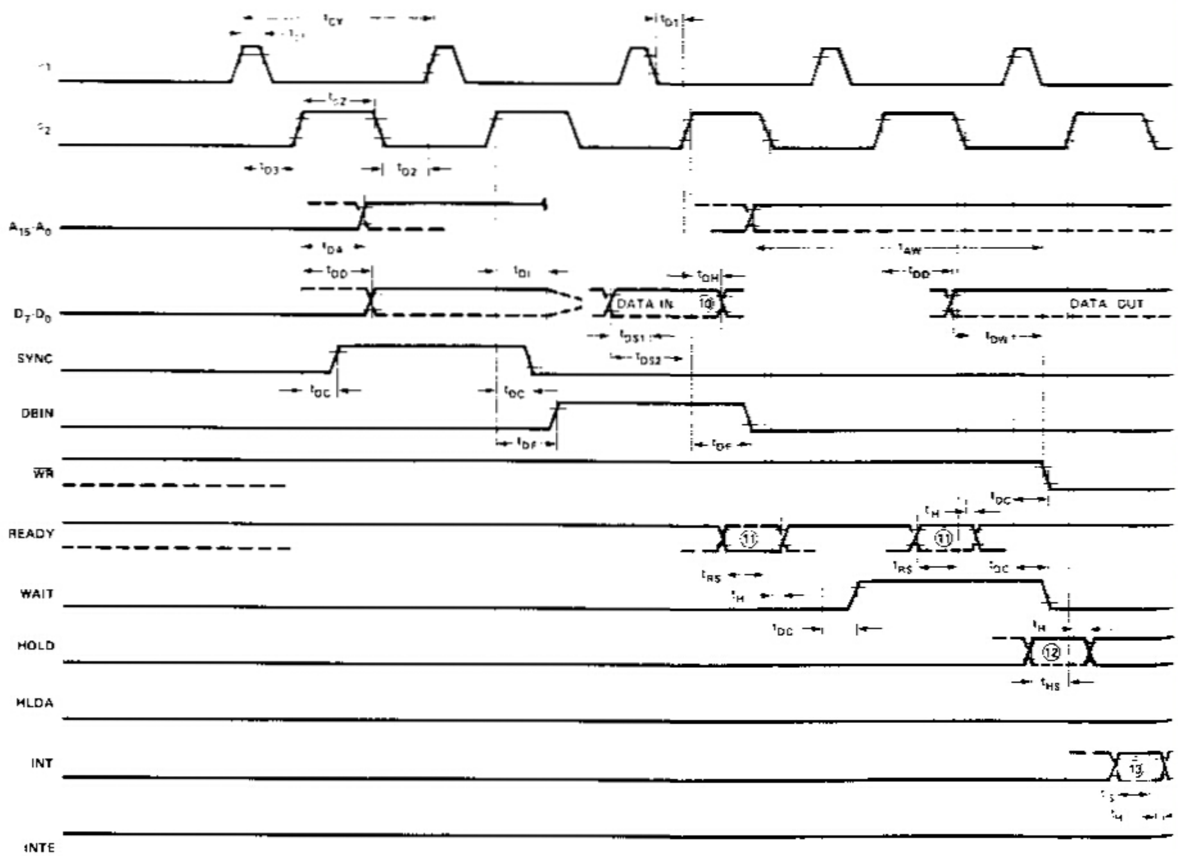
## A.C. CHARACTERISTICS

$T_A = 0°C$ to $70°C$, $V_{DD} = +12V \pm 5\%$, $V_{CC} = +5V \pm 5\%$, $V_{BB} = -5V \pm 5\%$, $V_{SS} = 0V$, Unless Otherwise Noted

| Symbol | Parameter | Min. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{CY}$ [3] | Clock Period | .38 | 2.0 | μsec | |
| $t_r, t_f$ | Clock Rise and Fall Time | 0 | 50 | nsec | |
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | 60 | | nsec | |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | 175 | | nsec | |
| $t_{D1}$ | Delay $\phi_1$ to $\phi_2$ | 0 | | nsec | |
| $t_{D2}$ | Delay $\phi_2$ to $\phi_1$ | 70 | | nsec | |
| $t_{D3}$ | Delay $\phi_1$ to $\phi_2$ Leading Edges | 70 | | nsec | |
| $t_{DA}$ [2] | Address Output Delay From $\phi_2$ | | 175 | nsec | $C_L = 100pf$ |
| $t_{DD}$ [2] | Data Output Delay From $\phi_2$ | | 200 | nsec | $C_L = 100pf$ |
| $t_{DC}$ [2] | Signal Output Delay From $\phi_1$, or $\phi_2$ (SYNC, $\overline{WR}$, WAIT, HLDA) | | 120 | nsec | $C_L = 50pf$ |
| $t_{DF}$ [2] | DBIN Delay From $\phi_2$ | 25 | 140 | nsec | $C_L = 50pf$ |
| $t_{DI}$ [1] | Delay for Input Bus to Enter Input Mode | | $t_{DF}$ | nsec | |
| $t_{DS1}$ | Data Setup Time During $\phi_1$ and DBIN | 20 | | nsec | |

## TIMING WAVEFORMS [14]

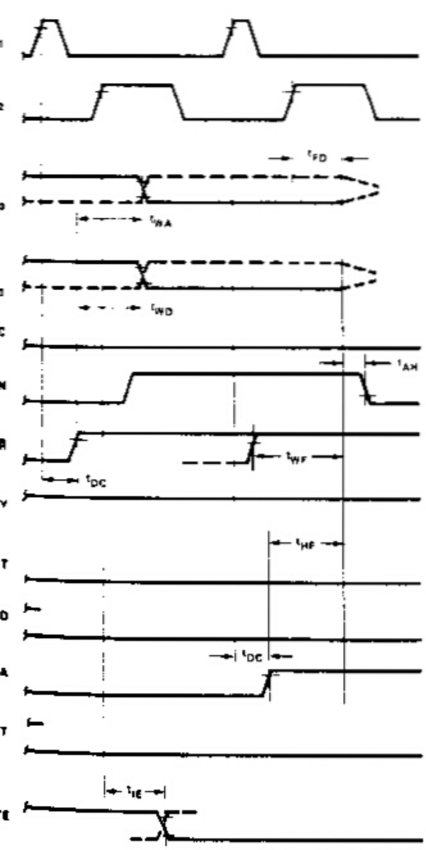
(Note: Timing measurements are made at the following reference voltages: CLOCK "1" = 8.0V "0" = 1.0V; INPUTS "1" = 3.3V, "0" = 0.8V; OUTPUTS "1" = 2.0V, "0" = 0.8V.)



## A.C. CHARACTERISTICS (Continued)

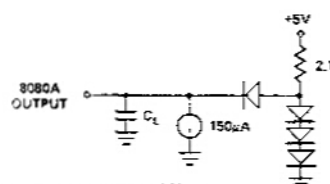
$T_A$ = 0°C to 70°C, $V_{DD}$ = +12V ± 5%, $V_{CC}$ = +5V ± 5%, $V_{BB}$ = -5V ± 5%, $V_{SS}$ = 0V, Unless Otherwise Noted

| Symbol | Parameter | Min. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{DS2}$ | Data Setup Time to $\phi_2$ During DBIN | 130 | | nsec | |
| $t_{DH}$ [1] | Data Hold Time From $\phi_2$ During DBIN | [1] | | nsec | |
| $t_{IE}$ [2] | INTE Output Delay From $\phi_2$ | | 200 | nsec | $C_L$ = 50pf |
| $t_{RS}$ | READY Setup Time During $\phi_2$ | 90 | | nsec | |
| $t_{HS}$ | HOLD Setup Time to $\phi_2$ | 120 | | nsec | |
| $t_{IS}$ | INT Setup Time During $\phi_2$ (During $\phi_1$ in Halt Mode) | 100 | | nsec | |
| $t_H$ | Hold Time From $\phi_2$ (READY, INT, HOLD) | 0 | | nsec | |
| $t_{FD}$ | Delay to Float During Hold (Address and Data Bus) | | 120 | nsec | |
| $t_{AW}$ [2] | Address Stable Prior to $\overline{WR}$ | [5] | | nsec | |
| $t_{DW}$ [2] | Output Data Stable Prior to $\overline{WR}$ | [6] | | nsec | |
| $t_{WD}$ [2] | Output Data Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{WA}$ [2] | Address Stable From $\overline{WR}$ | [7] | | nsec | $C_L$ = 100pf: Address, Data<br>$C_L$ = 50pf: $\overline{WR}$, HLDA, DBIN |
| $t_{HF}$ [2] | HLDA to Float Delay | [8] | | nsec | |
| $t_{WF}$ [2] | $\overline{WR}$ to Float Delay | [9] | | nsec | |
| $t_{AH}$ [2] | Address Hold Time After DBIN During HLDA | -20 | | nsec | |



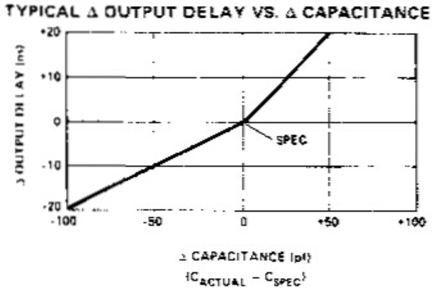

NOTES:

1. Data input should be enabled with DBIN status. No bus conflict can then occur and data hold time is assured. $t_{DH}$ = 50ns or $t_{DF}$, whichever is less.
2. Load Circuit




3. $t_{CY} = t_{D3} + t_{r\phi2} + t_{\phi2} + t_{f\phi2} + t_{D2} + t_{r\phi1} > 380ns$.

TYPICAL Δ OUTPUT DELAY VS. Δ CAPACITANCE




4. The following are relevent when interfacing the 8080A to devices having $V_{IH}$ = 3.3V:
   a) Maximum output rise time from .8V to 3.3V = 100ns @ $C_L$ = SPEC.
   b) Output delay when measured to 3.0V = SPEC +60ns @ $C_L$ = SPEC.
   c) If $C_L \neq$ SPEC, add .6ns/pF if $C_L > C_{SPEC}$, subtract .3ns/pF (from modified delay) if $C_L < C_{SPEC}$.
5. $t_{AW} = 2\, t_{CY} - t_{D3} - t_{r\phi2} - 130nsec$.
6. $t_{DW} = t_{CY} - t_{D3} - t_{r\phi2} - 170nsec$.
7. If not HLDA, $t_{WD} = t_{WA} = t_{D3} + t_{r\phi2} + 10ns$. If HLDA, $t_{WD} = t_{WA} = t_{WF}$.
8. $t_{HF} = t_{D3} + t_{r\phi2} - 50ns$.
9. $t_{WF} = t_{D3} + t_{r\phi2} - 10ns$
10. Data in must be stable for this period during DBIN $\cdot T_3$. Both $t_{DS1}$ and $t_{DS2}$ must be satisfied.
11. Ready signal must be stable for this period during $T_2$ or $T_W$. (Must be externally synchronized.)
12. Hold signal must be stable for this period during $T_2$ or $T_W$ when entering hold mode, and during $T_3$, $T_4$, $T_5$ and $T_{WH}$ when in hold mode. (External synchronization is not required.)
13. Interrupt signal must be stable during this period of the last clock cycle of any instruction in order to be recognized on the following instruction. (External synchronization is not required.)
14. This timing diagram shows timing relationships only; it does not represent any specific machine cycle.

# Silicon Gate MOS M8080A

## SINGLE CHIP 8-BIT N-CHANNEL MICROPROCESSOR

*The M8080A is functionally compatible with the Intel® 8080.*

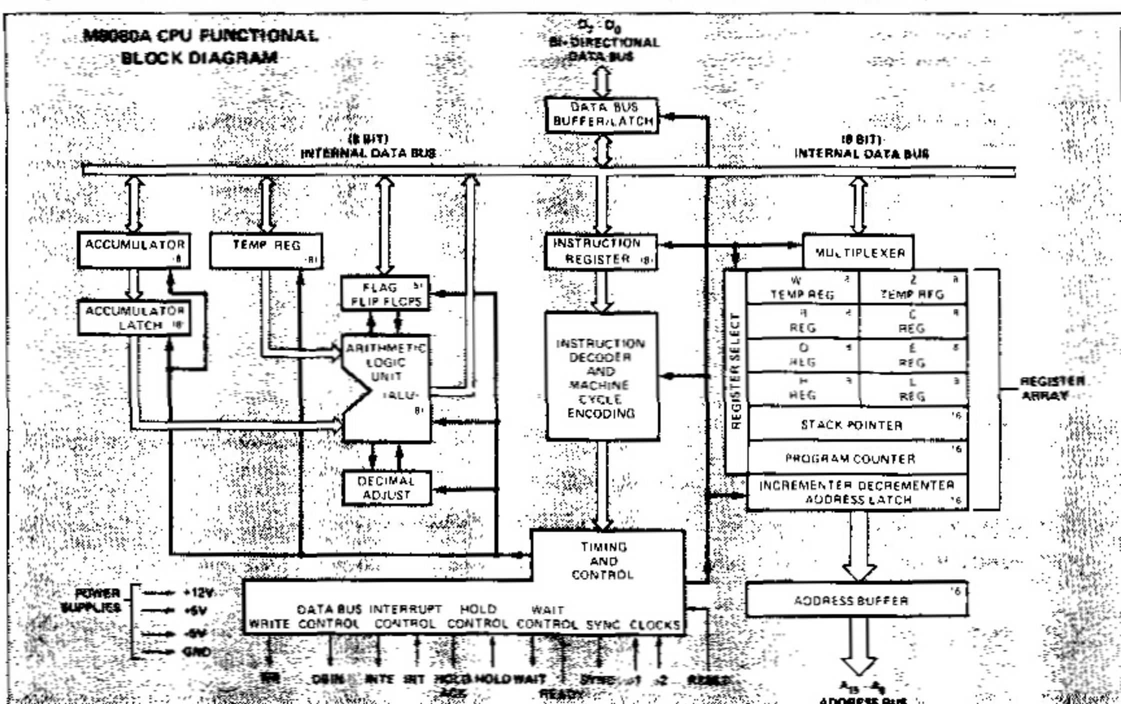
- **Full Military Temperature Range −55°C to +125°C**
- **±10% Power Supply Tolerance**
- **2 μs Instruction Cycle**
- **Powerful Problem Solving Instruction Set**
- **Six General Purpose Registers and an Accumulator**
- **Sixteen Bit Program Counter for Directly Addressing up to 64K Bytes of Memory**
- **Sixteen Bit Stack Pointer and Stack Manipulation Instructions for Rapid Switching of the Program Environment**
- **Decimal, Binary and Double Precision Arithmetic**
- **Ability to Provide Priority Vectored Interrupts**
- **512 Directly Addressed I/O Ports**
- **TTL Drive Capability**

The Intel® M8080A is a complete 8-bit parallel central processing unit (CPU). It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process. This offers the user a high performance solution to control and processing applications.

The M8080A contains six 8-bit general purpose working registers and an accumulator. The six general purpose registers may be addressed individually or in pairs providing both single and double precision operators. Arithmetic and logical instructions set or reset four testable flags. A fifth flag provides decimal arithmetic operation.

The M8080A has an external stack feature wherein any portion of memory may be used as a last in/first out stack to store/retrieve the contents of the accumulator, flags, program counter and all of the six general purpose registers. The sixteen bit stack pointer controls the addressing of this external stack. This stack gives the M8080A the ability to easily handle multiple level priority interrupts by rapidly storing and restoring processor status. It also provides almost unlimited subroutine nesting.

This microprocessor has been designed to simplify systems design. Separate 16-line address and 8-line bi-directional data busses are used to facilitate easy interface to memory and I/O. Signals to control the interface to memory and I/O are provided directly by the M8080A. Ultimate control of the address and data busses resides with the HOLD signal. It provides the ability to suspend processor operation and force the address and data busses into a high impedance state. This permits OR-tying these busses with other controlling devices for (DMA) direct memory access or multi-processor operation.



## INSTRUCTION SET

The accumulator group instructions include arithmetic and logical operators with direct, indirect, and immediate addressing modes.

Move, load, and store instruction groups provide the ability to move either 8 or 16 bits of data between memory, the six working registers and the accumulator using direct, indirect, and immediate addressing modes.

The ability to branch to different portions of the program is provided with jump, jump conditional, and computed jumps. Also the ability to call to and return from subroutines is provided both conditionally and unconditionally. The RESTART (or single byte call instruction) is useful for interrupt vector operation.

Double precision operators such as stack manipulation and double add instructions extend both the arithmetic and interrupt handling capability of the M8080A. The ability to increment and decrement memory, the six general registers and the accumulator is provided as well as extended increment and decrement instructions to operate on the register pairs and stack pointer. Further capability is provided by the ability to rotate the accumulator left or right through or around the carry bit.

Input and output may be accomplished using memory addresses as I/O ports or the directly addressed I/O provided for in the M8080A instruction set.

The following special instruction group completes the M8080A instruction set: the NOP instruction, HALT to stop processor execution and the DAA instructions provide decimal arithmetic capability. STC allows the carry flag to be directly set, and the CMC instruction allows it to be complemented. CMA complements the contents of the accumulator and XCHG exchanges the contents of two 16-bit register pairs directly.

### Data and Instruction Formats

Data in the M8080A 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.

| Format | Byte | | TYPICAL INSTRUCTIONS |
|---|---|---|---|
| One Byte Instructions | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | Register to register, memory reference, arithmetic or logical, rotate, return, push, pop, enable or disable Interrupt instructions |
| Two Byte Instructions | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | OP CODE | |
| | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | 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 | Jump, call or direct load and store instructions |
| | $D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$ | 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 | |

For the M8080A a logic "1" is defined as a high level and a logic "0" is defined as a low level.

## INSTRUCTION SET

### Summary of Processor Instructions

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| MOV $r_1,r_2$ | Move register to register | 0 | 1 | D | D | D | S | S | S | 5 |
| MOV M, r | Move register to memory | 0 | 1 | 1 | 1 | 0 | S | S | S | 7 |
| MOV r, M | Move memory to register | 0 | 1 | D | D | D | 1 | 1 | 0 | 7 |
| HLT | Halt | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| MVI r | Move immediate register | 0 | 0 | D | D | D | 1 | 1 | 0 | 7 |
| MVI M | Move immediate memory | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 10 |
| INR r | Increment register | 0 | 0 | D | D | D | 1 | 0 | 0 | 5 |
| DCR r | Decrement register | 0 | 0 | D | D | D | 1 | 0 | 1 | 5 |
| INR M | Increment memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 10 |
| DCR M | Decrement memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 10 |
| ADD r | Add register to A | 1 | 0 | 0 | 0 | 0 | S | S | S | 4 |
| ADC r | Add register to A with carry | 1 | 0 | 0 | 0 | 1 | S | S | S | 4 |
| SUB r | Subtract register from A | 1 | 0 | 0 | 1 | 0 | S | S | S | 4 |
| SBB r | Subtract register from A with borrow | 1 | 0 | 0 | 1 | 1 | S | S | S | 4 |
| ANA r | And register with A | 1 | 0 | 1 | 0 | 0 | S | S | S | 4 |
| XRA r | Exclusive Or register with A | 1 | 0 | 1 | 0 | 1 | S | S | S | 4 |
| ORA r | Or register with A | 1 | 0 | 1 | 1 | 0 | S | S | S | 4 |
| CMP r | Compare register with A | 1 | 0 | 1 | 1 | 1 | S | S | S | 4 |
| ADD M | Add memory to A | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ADC M | Add memory to A with carry | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUB M | Subtract memory from A | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBB M | Subtract memory from A with borrow | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANA M | And memory with A | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRA M | Exclusive Or memory with A | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORA M | Or memory with A | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CMP M | Compare memory with A | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| ADI | Add immediate to A | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ACI | Add immediate to A with carry | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUI | Subtract immediate from A | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBI | Subtract immediate from A with borrow | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANI | And immediate with A | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRI | Exclusive Or immediate with A | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORI | Or immediate with A | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CPI | Compare immediate with A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| RLC | Rotate A left | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 |
| RRC | Rotate A right | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 4 |
| RAL | Rotate A left through carry | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| RAR | Rotate A right through carry | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 |
| JMP | Jump unconditional | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 10 |
| JC | Jump on carry | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 10 |
| JNC | Jump on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 10 |
| JZ | Jump on zero | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 10 |
| JNZ | Jump on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 10 |
| JP | Jump on positive | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 10 |
| JM | Jump on minus | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 10 |
| JPE | Jump on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 10 |
| JPO | Jump on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 10 |
| CALL | Call unconditional | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 17 |
| CC | Call on carry | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CNC | Call on no carry | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CZ | Call on zero | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CNZ | Call on no zero | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| CP | Call on positive | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CM | Call on minus | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CPE | Call on parity even | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CPO | Call on parity odd | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| RET | Return | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| RC | Return on carry | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RNC | Return on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RZ | Return on zero | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RNZ | Return on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RP | Return on positive | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RM | Return on minus | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RPE | Return on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RPO | Return on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RST | Restart | 1 | 1 | A | A | A | 1 | 1 | 1 | 11 |
| IN | Input | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 10 |
| OUT | Output | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 10 |
| LXI B | Load immediate register Pair B & C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI D | Load immediate register Pair D & E | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| LXI H | Load immediate register Pair H & L | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI SP | Load immediate stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| PUSH B | Push register Pair B & C on stack | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH D | Push register Pair D & E on stack | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 11 |
| PUSH H | Push register Pair H & L on stack | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH PSW | Push A and Flags on stack | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 11 |
| POP B | Pop register pair B & C off stack | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP D | Pop register pair D & E off stack | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| POP H | Pop register pair H & L off stack | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP PSW | Pop A and Flags off stack | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| STA | Store A direct | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 13 |
| LDA | Load A direct | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 13 |
| XCHG | Exchange D & E, H & L Registers | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 4 |
| XTHL | Exchange top of stack, H & L | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 18 |
| SPHL | H & L to stack pointer | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 5 |
| PCHL | H & L to program counter | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 5 |
| DAD B | Add B & C to H & L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD D | Add D & E to H & L | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 10 |
| DAD H | Add H & L to H & L | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD SP | Add stack pointer to H & L | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 10 |
| STAX B | Store A indirect | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7 |
| STAX D | Store A indirect | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 7 |
| LDAX B | Load A indirect | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 |
| LDAX D | Load A indirect | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 7 |
| INX B | Increment B & C registers | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX D | Increment D & E registers | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 5 |
| INX H | Increment H & L registers | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX SP | Increment stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 5 |
| DCX B | Decrement B & C | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX D | Decrement D & E | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 5 |
| DCX H | Decrement H & L | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX SP | Decrement stack pointer | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 5 |
| CMA | Complement A | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 4 |
| STC | Set carry | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 4 |
| CMC | Complement carry | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| DAA | Decimal adjust A | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 4 |
| SHLD | Store H & L direct | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 16 |
| LHLD | Load H & L direct | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 16 |
| EI | Enable Interrupts | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 |
| DI | Disable interrupt | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| NOP | No-operation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |

NOTES: 1. DDD or SSS – 000 B – 001 C – 010 D – 011 E – 100 H – 101 L – 110 Memory – 111 A.
2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.

## M8080A FUNCTIONAL PIN DEFINITION

The following describes the function of all of the M8080A I/O pins. Several of the descriptions refer to internal timing periods.

**$A_{15}$-$A_0$ (output three-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 three-state)**

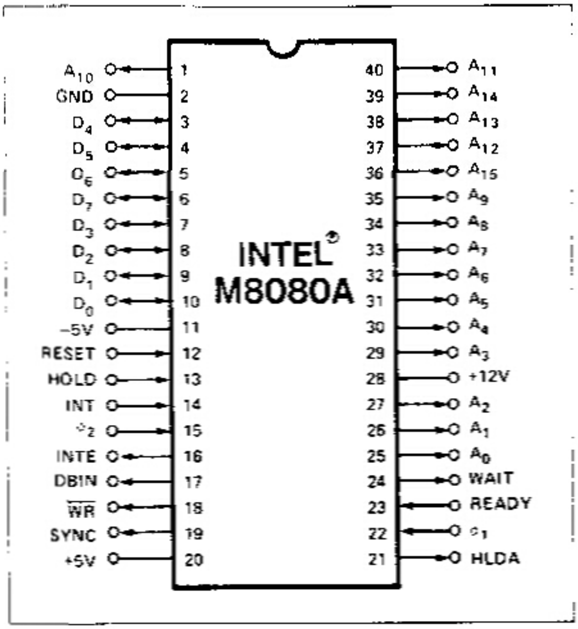
DATA BUS; the data bus provides bi-directional communication between the CPU, memory, and I/O devices for instructions and data transfers. Also, during the first clock cycle of each machine cycle, the M8080A outputs a status word on the data bus that describes the current machine cycle. $D_0$ is the least significant bit.

**SYNC (output)**

SYNCHRONIZING SIGNAL; the SYNC pin provides a signal to indicate the beginning of each machine cycle.

**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 M8080A data bus from memory or I/O.




Pin Configuration

**READY (input)**

READY; the READY signal indicates to the M8080A that valid memory or input data is available on the M8080A data bus. This signal is used to synchronize the CPU with slower memory or I/O devices. If after sending an address out the M8080A does not receive a READY input, the M8080A will enter a WAIT state for as long as the READY line is low. READY can also be used to single step the CPU.

**WAIT (output)**

WAIT; the WAIT signal acknowledges that the CPU is in a WAIT state.

**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 low ($\overline{WR}$ = 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 M8080A address and data bus as soon as the M8080A 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.

**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 $\phi_1$ and high impedance occurs after the rising edge of $\phi_2$.

**INTE (output)**

INTERRUPT ENABLE; indicates the content of the internal interrupt enable flip/flop. This flip/flop may be set or reset by the Enable and Disable Interrupt instructions and inhibits interrupts from being accepted by the CPU when 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.

**RESET (input)**[1]

RESET; while the RESET signal is activated, the content of the program counter is cleared. 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, stack pointer, and registers are not cleared.

| | |
|---|---|
| $V_{SS}$ | Ground Reference |
| $V_{DD}$ | +12 Volts ±10%. |
| $V_{CC}$ | +5 Volts ±10%. |
| $V_{BB}$ | -5 Volts ±10%. |
| $\phi_1$, $\phi_2$ | 2 externally supplied clock phases. (non TTL compatible) |

## ABSOLUTE MAXIMUM RATINGS*

| | |
|---|---|
| Temperature Under Bias | -55°C to +125°C |
| Storage Temperature | -65°C to +150°C |
| All Input or Output Voltages With Respect to $V_{BB}$ | -0.3V to +20V |
| $V_{CC}$, $V_{DD}$ and $V_{SS}$ With Respect to $V_{BB}$ | -0.3V to +20V |
| Power Dissipation | 1.7W |

***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

$T_A$ = -55°C to +125°C, $V_{DD}$ = +12V ±10%, $V_{CC}$ = +5V ±10%, $V_{BB}$ = -5V ±10%, $V_{SS}$ = 0V, Unless Otherwise Noted.

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|---|
| $V_{ILC}$ | Clock Input Low Voltage | $V_{SS}-1$ | | $V_{SS}+0.8$ | V | |
| $V_{IHC}$ | Clock Input High Voltage | 8.5 | | $V_{DD}+1$ | V | |
| $V_{IL}$ | Input Low Voltage | $V_{SS}-1$ | | $V_{SS}+0.8$ | V | |
| $V_{IH}$ | Input High Voltage | 3.0 | | $V_{CC}+1$ | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.9mA on all outputs, |
| $V_{OH}$ | Output High Voltage | 3.7 | | | V | $I_{OH}$ = 150μA. |
| $I_{DD\,(AV)}$ | Avg. Power Supply Current ($V_{DD}$) | | 50 | 80 | mA | Operation |
| $I_{CC\,(AV)}$ | Avg. Power Supply Current ($V_{CC}$) | | 60 | 100 | mA | $T_{CY}$ = .48 μsec |
| $I_{BB\,(AV)}$ | Avg. Power Supply Current ($V_{BB}$) | | .01 | 1 | mA | |
| $I_{IL}$ | Input Leakage | | | ±10 | μA | $V_{SS} \leq V_{IN} \leq V_{CC}$ |
| $I_{CL}$ | Clock Leakage | | | ±10 | μA | $V_{SS} \leq V_{CLOCK} \leq V_{DD}$ |
| $I_{DL}$ [2] | Data Bus Leakage in Input Mode | | | -100<br>-2.0 | μA<br>mA | $V_{SS} \leq V_{IN} \leq V_{SS}+0.8V$<br>$V_{SS}+0.8V \leq V_{IN} \leq V_{CC}$ |
| $I_{FL}$ | Address and Data Bus Leakage During HOLD | | | +10<br>-100 | μA | $V_{ADDR/DATA} = V_{CC}$<br>$V_{ADDR/DATA} = V_{SS} + 0.45V$ |

## CAPACITANCE

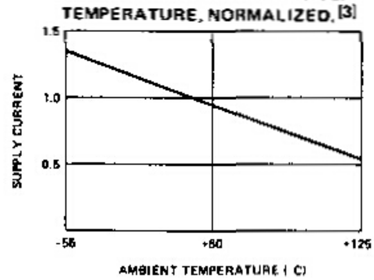
$T_A$ = 25°C $V_{CC} = V_{DD} = V_{SS}$ = 0V, $V_{BB}$ = -5V

| Symbol | Parameter | Typ. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $C_\phi$ | Clock Capacitance | 17 | 25 | pf | $f_c$ = 1 MHz |
| $C_{IN}$ | Input Capacitance | 6 | 10 | pf | Unmeasured Pins |
| $C_{OUT}$ | Output Capacitance | 10 | 20 | pf | Returned to $V_{SS}$ |

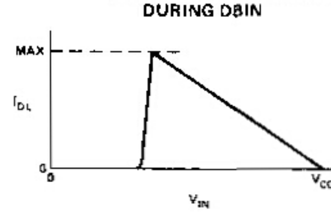
NOTES:

1. The RESET signal must be active for a minimum of 3 clock cycles.
2. When DBIN is high and $V_{IN} > V_{IH}$ an internal active pull up will be switched onto the Data Bus.
3. $\Delta I$ supply / $\Delta T_A$ = -0.45%/°C.

TYPICAL SUPPLY CURRENT VS. TEMPERATURE, NORMALIZED.[3]




DATA BUS CHARACTERISTIC DURING DBIN



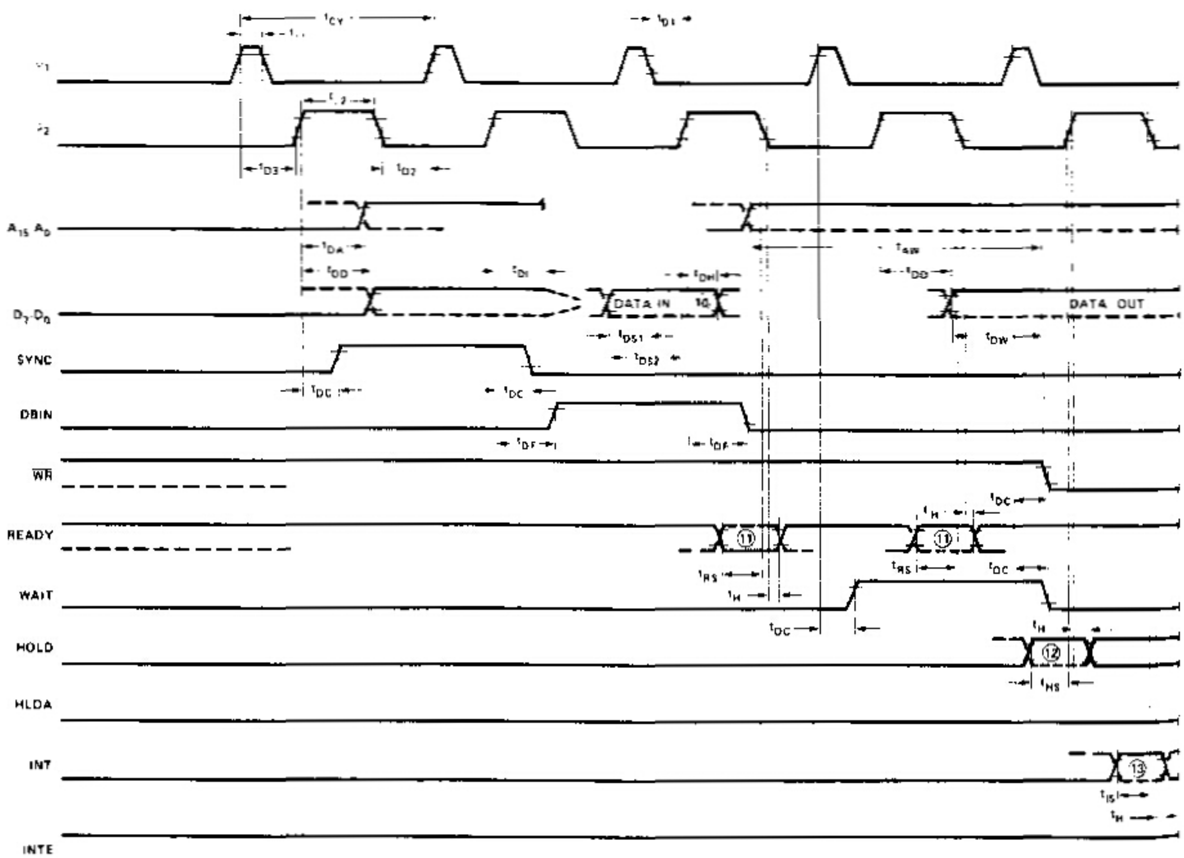
## A.C. CHARACTERISTICS

$T_A$ = -55°C to +125°C, $V_{DD}$ = +12V ±10%, $V_{CC}$ = +5V ±10%, $V_{BB}$ = -5V ±10%, $V_{SS}$ = 0V, Unless Otherwise Noted.

| Symbol | Parameter | Min. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{CY}$ [3] | Clock Period | 0.48 | 2.0 | μsec | |
| $t_r$, $t_f$ | Clock Rise and Fall Time | 0 | 50 | nsec | |
| $t_{\phi 1}$ | $\phi_1$ Pulse Width | 60 | | nsec | |
| $t_{\phi 2}$ | $\phi_2$ Pulse Width | 220 | | nsec | |
| $t_{D1}$ | Delay $\phi_1$ to $\phi_2$ | 0 | | nsec | |
| $t_{D2}$ | Delay $\phi_2$ to $\phi_1$ | 80 | | nsec | |
| $t_{D3}$ | Delay $\phi_1$ to $\phi_2$ Leading Edges | 80 | | nsec | |
| $t_{DA}$ [2] | Address Output Delay From $\phi_2$ | | 200 | nsec | $C_L$ = 50pf |
| $t_{DD}$ [2] | Data Output Delay From $\phi_2$ | | 220 | nsec | $C_L$ = 50pf |
| $t_{DC}$ [2] | Signal Output Delay From $\phi_1$, or $\phi_2$ (SYNC, $\overline{WR}$, WAIT, HLDA) | | 140 | nsec | $C_L$ = 50pf |
| $t_{DF}$ [2] | DBIN Delay From $\phi_2$ | 25 | 150 | nsec | $C_L$ = 50pf |
| $t_{DI}$ [1] | Delay for Input Bus to Enter Input Mode | | $t_{DF}$ | nsec | |
| $t_{DS1}$ | Data Setup Time During $\phi_1$ and DBIN | 30 | | nsec | |

## TIMING WAVEFORMS [14]

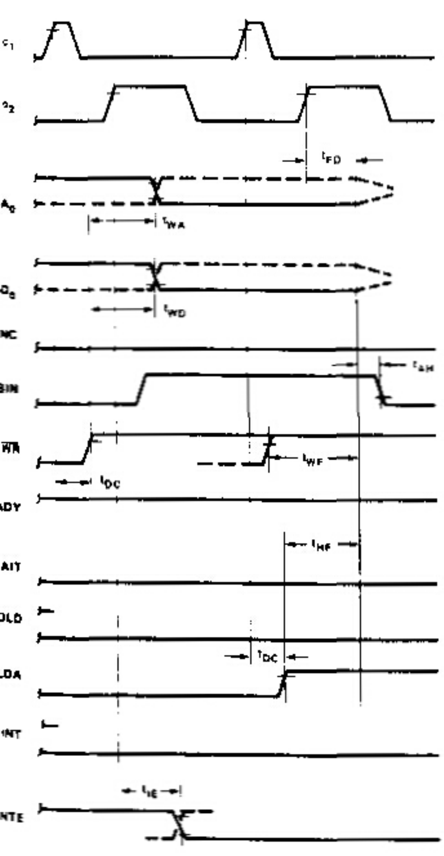
(Note: Timing measurements are made at the following reference voltages: CLOCK "1" = 7.0V, "0" = 1.0V; INPUTS "1" = 3.0V, "0" = 0.8V; OUTPUTS "1" = 2.0V, "0" = 0.8V.)



## A.C. CHARACTERISTICS (Continued)

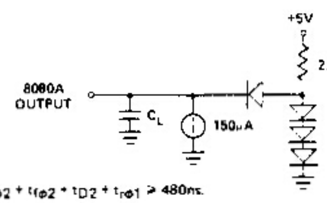
$T_A$ = -55°C to +125°C, $V_{DD}$ = +12V ±10%, $V_{CC}$ = +5V ±10%, $V_{BB}$ = -5V ±10%, $V_{SS}$ = 0V, Unless Otherwise Noted.

| Symbol | Parameter | Min. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|
| $t_{DS2}$ | Data Setup Time to $\phi_2$ During DBIN | 130 | | nsec | $C_L$ = 50pf |
| $t_{DH}$ [1] | Data Hold Time From $\phi_2$ During DBIN | 50 | | nsec | |
| $t_{IE}$ [2] | INTE Output Delay From $\phi_2$ | | 200 | nsec | |
| $t_{RS}$ | READY Setup Time During $\phi_2$ | 120 | | nsec | |
| $t_{HS}$ | HOLD Setup Time to $\phi_2$ | 140 | | nsec | |
| $t_{IS}$ | INT Setup Time During $\phi_2$ (During $\phi_1$ in Halt Mode) | 120 | | nsec | |
| $t_H$ | Hold Time From $\phi_2$ (READY, INT, HOLD) | 0 | | nsec | |
| $t_{FD}$ | Delay to Float During Hold (Address and Data Bus) | | 130 | nsec | |
| $t_{AW}$ [2] | Address Stable Prior to $\overline{WR}$ | [5] | | nsec | $C_L$ = 50pf |
| $t_{DW}$ [2] | Output Data Stable Prior to $\overline{WR}$ | [6] | | nsec | |
| $t_{WD}$ [2] | Output Data Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{WA}$ [2] | Address Stable From $\overline{WR}$ | [7] | | nsec | |
| $t_{HF}$ [2] | HLDA to Float Delay | [8] | | nsec | |
| $t_{WF}$ [2] | $\overline{WR}$ to Float Delay | [9] | | nsec | |
| $t_{AH}$ [2] | Address Hold Time After DBIN During HLDA | -20 | | nsec | |



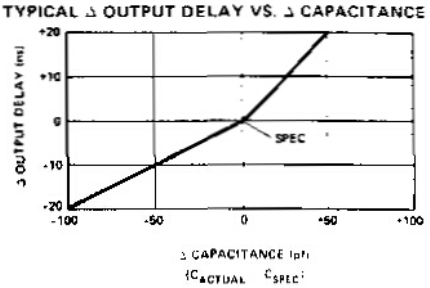

NOTES:

1. Data input should be enabled with DBIN status. No bus conflict can then occur and data hold time is assured. $t_{DH}$ = 50 ns or $t_{DF}$, whichever is less.
2. Load Circuit.




3. $t_{CY} = t_{D3} + t_{r\phi2} + t_{\phi2} + t_{f\phi2} + t_{D2} + t_{r\phi1} \geq 480ns$.

TYPICAL Δ OUTPUT DELAY VS. Δ CAPACITANCE

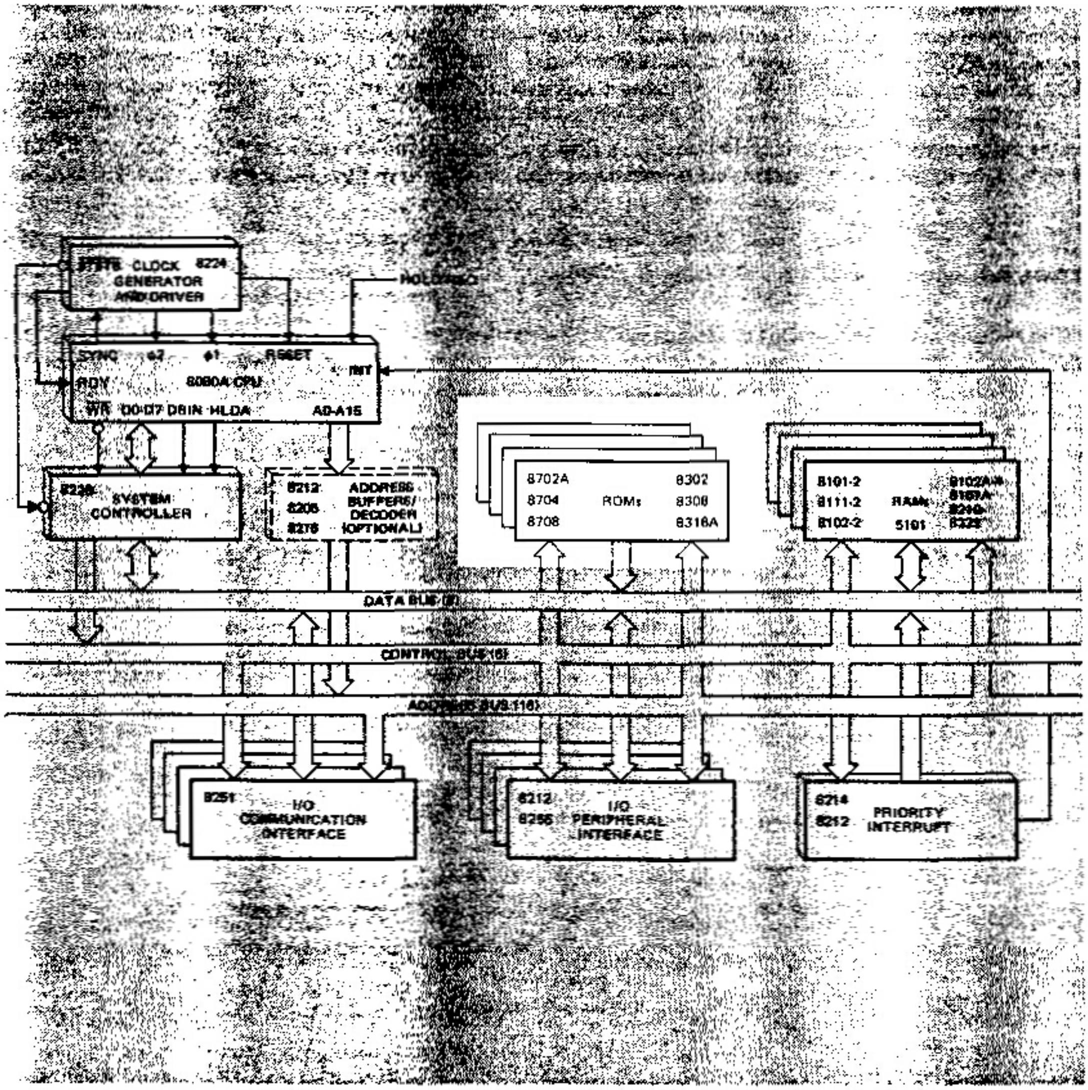



4. The following are relevent when interfacing the M8080A to devices having $V_{IH}$ = 3.3V
   a) Maximum output rise time from .8V to 3.3V = 100ns @ $C_L$ = SPEC
   b) Output delay when measured to 3.0V = SPEC +60ns @ $C_L$ = SPEC.
   c) If $C_L \neq$ SPEC, add .6ns/pF if $C_L > C_{SPEC}$, subtract .3ns/pF (from modified delay) if $C_L < C_{SPEC}$
5. $t_{AW} = 2\,t_{CY} - t_{D3} - t_{r\phi2} - 140nsec$.
6. $t_{DW} = t_{CY} - t_{D3} - t_{r\phi2} - 170nsec$
7. If not HLDA, $t_{WD} = t_{WA} = t_{D3} + t_{r\phi2} + 10ns$. If HLDA, $t_{WD} = t_{WA} = t_{WF}$
8. $t_{HF} = t_{D3} + t_{r\phi2} - 50ns$.
9. $t_{WF} = t_{D3} + t_{r\phi2} - 10ns$
10. Data in must be stable for this period during DBIN $\cdot T_3$. Both $t_{DS1}$ and $t_{DS2}$ must be satisfied.
11. Ready signal must be stable for this period during $T_2$ or $T_W$. (Must be externally synchronized.)
12. Hold signal must be stable for this period during $T_2$ or $T_W$ when entering hold mode, and during $T_3$, $T_4$, $T_5$ and $T_{WH}$ when in hold mode. (External synchronization is not required.)
13. Interrupt signal must be stable during this period of the last clock cycle of any instruction in order to be recognized on the following instruction. (External synchronization is not required.)
14. This timing diagram shows timing relationships only; it does not represent any specific machine cycle

intel® Microcomputer Systems

# ROMs

## 8702A 8704 8708 8302 8308 8316A



# Silicon Gate MOS 8702A

## 2048 BIT ERASABLE AND ELECTRICALLY REPROGRAMMABLE READ ONLY MEMORY

- **Access Time — 1.3 μsec Max.**
- **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.

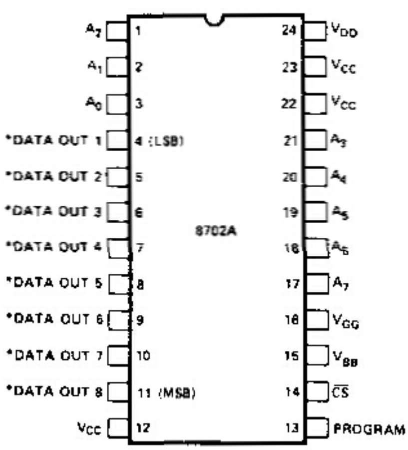
The 8702A is packaged in a 24 pin dual-in line package with a transparent quartz lid. The transparent quartz lid allows the user to expose the chip to ultraviolet light to erase the bit pattern. A new pattern can then be written into the device. This procedure can be repeated as many times as required.

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.

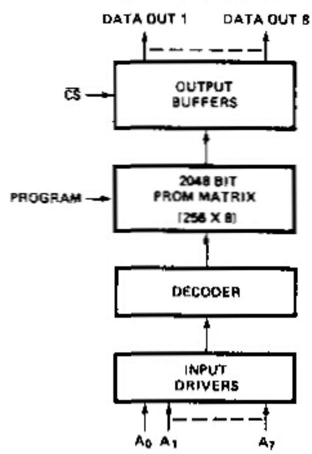
The 8702A is fabricated with silicon gate technology. This low threshold technology allows the design and production of higher performance MOS circuits and provides a higher functional density on a monolithic chip than conventional MOS technologies.

### PIN CONFIGURATION




*THIS PIN IS THE DATA INPUT LEAD DURING PROGRAMMING.

### BLOCK DIAGRAM




### PIN NAMES

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

## PIN CONNECTIONS

The external lead connections to the 8702A differ, depending on whether the device is being programmed[1] or used in read mode. (See following table.)

| MODE \ PIN | 12 ($V_{CC}$) | 13 (Program) | 14 ($\overline{CS}$) | 15 ($V_{BB}$) | 16 ($V_{GG}$) | 22 ($V_{CC}$) | 23 ($V_{CC}$) |
|---|---|---|---|---|---|---|---|
| Read | $V_{CC}$ | $V_{CC}$ | GND | $V_{CC}$ | $V_{GG}$ | $V_{CC}$ | $V_{CC}$ |
| Programming | GND | Program Pulse | GND | $V_{BB}$ | Pulsed $V_{GG}$ ($V_{IL4P}$) | GND | GND |

## ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias . . . . . . . . 0°C to +70°C
Storage Temperature . . . . . . . . . . . . . . −65°C to +125°C
Soldering Temperature of Leads (10 sec) . . . . . . . . +300°C
Power Dissipation . . . . . . . . . . . . . . . . . . . . . . . 2 Watts
Read Operation: Input Voltages and Supply
Voltages with respect to $V_{CC}$ . . . . . . . . . . .+0.5V to −20V
Program Operation: Input Voltages and Supply
Voltages with respect to $V_{CC}$ . . . . . . . . . . . . . . . . −48V

*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 at any other condition 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.

# READ OPERATION

## D.C. AND OPERATING CHARACTERISTICS

$T_A$ = 0°C to 70°C, $V_{CC}$ = +5V±5%, $V_{DD}$ = −9V±5%, $V_{GG}$[2] = −9V±5%, unless otherwise noted.

| SYMBOL | TEST | MIN. | TYP.[3] | MAX. | UNIT | CONDITIONS | |
|---|---|---|---|---|---|---|---|
| $I_{LI}$ | Address and Chip Select Input Load Current | | | 10 | μA | $V_{IN}$ = 0.0V | |
| $I_{LO}$ | Output Leakage Current | | | 10 | μA | $V_{OUT}$ = 0.0V, $\overline{CS}$ = $V_{CC}$ −2 | |
| $I_{DD0}$ | Power Supply Current | | 5 | 10 | mA | $V_{GG}$=$V_{CC}$, $\overline{CS}$=$V_{CC}$−2, $I_{OL}$ = 0.0mA, $T_A$ = 25°C | |
| $I_{DD1}$ | Power Supply Current | | 35 | 50 | mA | $\overline{CS}$=$V_{CC}$ −2, $I_{OL}$ =0.0mA, $T_A$ = 25°C | Continuous Operation |
| $I_{DD2}$ | Power Supply Current | | 32 | 46 | mA | $\overline{CS}$=0.0, $I_{OL}$ =0.0mA, $T_A$ = 25°C | Continuous Operation |
| $I_{DD3}$ | Power Supply Current | | 38.5 | 60 | mA | $\overline{CS}$=$V_{CC}$ −2, $I_{OL}$ =0.0mA, $T_A$ = 0°C | Continuous Operation |
| $I_{CF1}$ | Output Clamp Current | | 8 | 14 | mA | $V_{OUT}$ = −1.0V, $T_A$ = 0°C | Continuous Operation |
| $I_{CF2}$ | Output Clamp Current | | | 13 | mA | $V_{OUT}$ = −1.0V, $T_A$ = 25°C | Continuous Operation |
| $I_{GG}$ | Gate Supply Current | | | 10 | μA | | |
| $V_{IL1}$ | Input Low Voltage for TTL Interface | −1.0 | | 0.65 | V | | |
| $V_{IL2}$ | Input Low Voltage for MOS Interface | $V_{DD}$ | | $V_{CC}$ −6 | V | | |
| $V_{IH}$ | Address and Chip Select Input High Voltage | $V_{CC}$−2 | | $V_{CC}$ +0.3 | V | | |
| $I_{OL}$ | Output Sink Current | 1.6 | 4 | | mA | $V_{OUT}$ = 0.45V | |
| $V_{OL}$ | Output Low Voltage | | [illegible] | 0.45 | V | $I_{OL}$ = 1.6mA | |
| $V_{OH}$ | Output High Voltage | 3.5 | | | V | $I_{OH}$ = −200 μA | |

Note 1: In the programming mode, the data inputs 1–8 are pins 4–11 respectively. $\overline{CS}$ = GND.
Note 2: $V_{GG}$ may be clocked to reduce power dissipation. In this mode average $I_{DD}$ increases in proportion to $V_{GG}$ duty cycle. (See p. 5)
Note 3: Typical values are at nominal voltages and $T_A$ = 25°C.

## A.C. CHARACTERISTICS

$T_A$ = 0°C to +70°C, $V_{CC}$ = +5V ±5%, $V_{DD}$ = −9V ±5%, $V_{GG}$ = −9V ±5% unless otherwise noted

| SYMBOL | TEST | MINIMUM | TYPICAL | MAXIMUM | UNIT |
|---|---|---|---|---|---|
| Freq. | Repetition Rate | | | 1 | MHz |
| $t_{OH}$ | Previous read data valid | | | 100 | ns |
| $t_{ACC}$ | Address to output delay | | | 1.3 | µs |
| $t_{DVGG}$ | Clocked $V_{GG}$ set up | 1.0 | | | µs |
| $t_{CS}$ | Chip select delay | | | 400 | ns |
| $t_{CO}$ | Output delay from $\overline{CS}$ | | | 900 | ns |
| $t_{OD}$ | Output deselect | | | 400 | ns |
| $t_{OHC}$ | Data out hold in clocked $V_{GG}$ mode (Note 1) | | | 5 | µs |

Note 1. The output will remain valid for $t_{OHC}$ as long as clocked $V_{GG}$ is at $V_{CC}$. An address change may occur as soon as the output is sensed (clocked $V_{GG}$ may still be at $V_{CC}$). Data becomes invalid for the old address when clocked $V_{GG}$ is returned to $V_{GG}$.

## CAPACITANCE* $T_A$ = 25°C

| SYMBOL | TEST | MINIMUM | TYPICAL | MAXIMUM | UNIT | CONDITIONS |
|---|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | | 8 | 15 | pF | $V_{IN} = V_{CC}$; $\overline{CS} = V_{CC}$; $V_{OUT} = V_{CC}$; $V_{GG} = V_{CC}$ — All unused pins are at A.C. ground |
| $C_{OUT}$ | Output Capacitance | | 10 | 15 | pF | |
| $C_{V_{GG}}$ | $V_{GG}$ Capacitance (Clocked $V_{GG}$ Mode) | | | 30 | pF | |

*This parameter is periodically sampled and is not 100% tested.

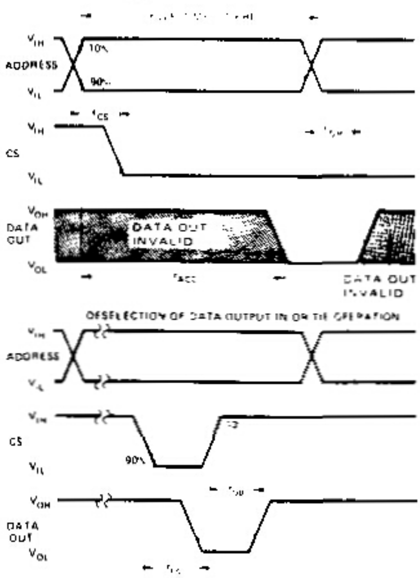
## SWITCHING CHARACTERISTICS

Conditions of Test:

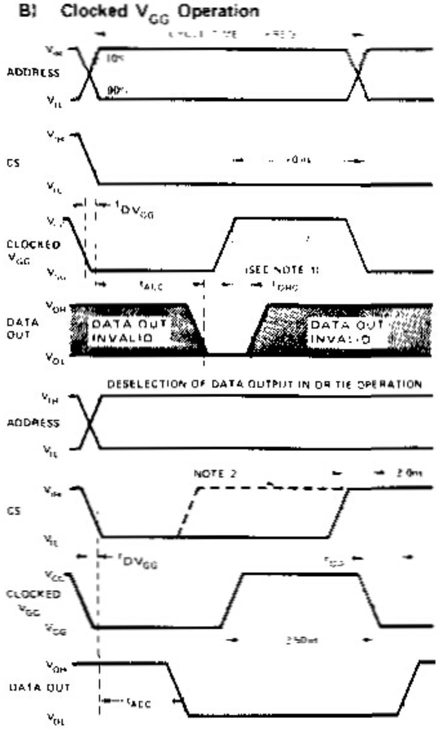
Input pulse amplitudes: 0 to 4V, $t_R$, $t_F$ ≤ 50 ns

Output load is 1 TTL gate, measurements made at output of TTL gate ($t_{PD}$ ≤ 15 ns)

A) Constant $V_{GG}$ Operation




B) Clocked $V_{CC}$ Operation

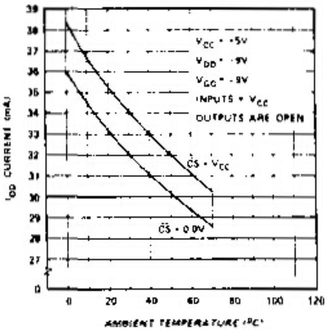



NOTE 1. The output will remain valid for $t_{OHC}$ as long as clocked $V_{GG}$ is at $V_{CC}$. An address change may occur as soon as the output is sensed (clocked $V_{GG}$ may still be at $V_{CC}$). Data becomes invalid for the old address when clocked $V_{GG}$ is returned to $V_{GG}$.

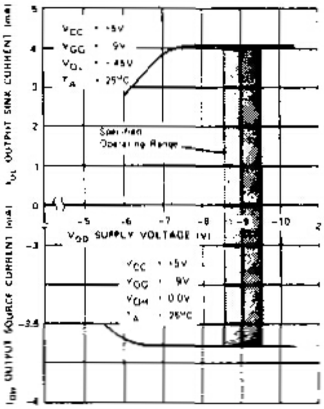
NOTE 2. If $\overline{CS}$ makes a transition from $V_{IL}$ to $V_{IH}$ while clocked $V_{GG}$ is at $V_{GG}$, then deselection of output occurs at $t_{OD}$ as shown in static operation with constant $V_{GG}$.

## TYPICAL CHARACTERISTICS

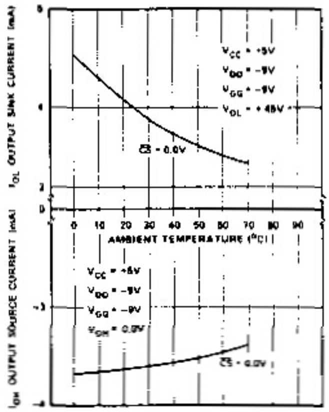
$I_{DD}$ CURRENT VS. TEMPERATURE



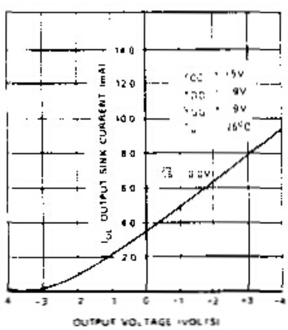

OUTPUT CURRENT VS. $V_{DD}$ SUPPLY VOLTAGE



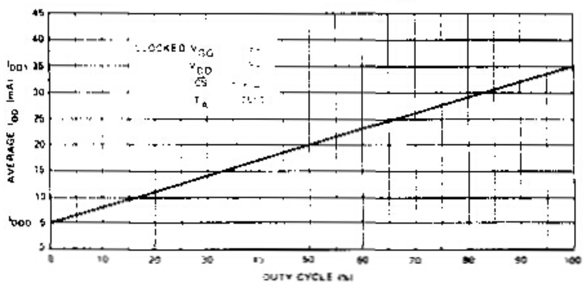

OUTPUT CURRENT VS. TEMPERATURE



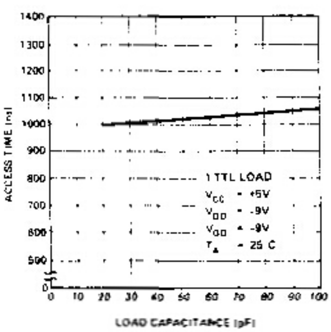

OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



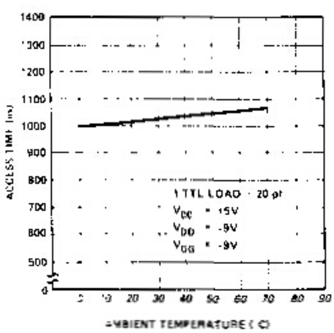

AVERAGE CURRENT VS. DUTY CYCLE FOR CLOCKED $V_{GG}$




ACCESS TIME VS. LOAD CAPACITANCE




ACCESS TIME VS. TEMPERATURE



## PROGRAMMING OPERATION

### D.C. AND OPERATING CHARACTERISTICS FOR PROGRAMMING OPERATION

$T_A = 25°C$, $V_{CC} = 0V$, $V_{BB} = +12V \pm 10\%$, $\overline{CS} = 0V$ unless otherwise noted

| SYMBOL | TEST | MIN. | TYP. | MAX. | UNIT | CONDITIONS |
|---|---|---|---|---|---|---|
| $I_{LI1P}$ | Address and Data Input Load Current | | | 10 | mA | $V_{IN} = -48V$ |
| $I_{LI2P}$ | Program and $V_{GG}$ Load Current | | | 10 | mA | $V_{IN} = -48V$ |
| $I_{BB}$ | $V_{BB}$ Supply Load Current | | .05 | | mA | |
| $I_{DDP}$ (1) | Peak $I_{DD}$ Supply Load Current | | 200 | | mA | $V_{DD} = V_{prog} = -48V$<br>$V_{GG} = -35V$ |
| $V_{IHP}$ | Input High Voltage | | | 0.3 | V | |
| $V_{IL1P}$ | Pulsed Data Input Low Voltage | −46 | | −48 | V | |
| $V_{IL2P}$ | Address Input Low Voltage | −40 | | −48 | V | |
| $V_{IL3P}$ | Pulsed Input Low $V_{DD}$ and Program Voltage | −46 | | −48 | V | |
| $V_{IL4P}$ | Pulsed Input Low $V_{GG}$ Voltage | −35 | | −40 | V | |

Note 1: $I_{DDP}$ flows only during $V_{DD}$, $V_{GG}$ on time. $I_{DDP}$ should not be allowed to exceed 300 mA for greater than 100 μsec. Average power supply current $I_{DDP}$ is typically 40 mA at 20% duty cycle.

### A.C. CHARACTERISTICS FOR PROGRAMMING OPERATION

$T_{AMBIENT} = 25°C$, $V_{CC} = 0V$, $V_{BB} = +12V \pm 10\%$, $\overline{CS} = 0V$ unless otherwise noted

| SYMBOL | TEST | MIN. | TYP. | MAX. | UNIT | CONDITIONS |
|---|---|---|---|---|---|---|
| | Duty Cycle ($V_{DD}$, $V_{GG}$) | | | 20 | % | |
| $t_{\phi PW}$ | Program Pulse Width | | | 3 | ms | $V_{GG} = -35V$, $V_{DD} = V_{prog} = -48V$ |
| $t_{DW}$ | Data Set Up Time | 25 | | | μs | |
| $t_{DH}$ | Data Hold Time | 10 | | | μs | |
| $t_{VW}$ | $V_{DD}$, $V_{GG}$ Set Up | 100 | | | μs | |
| $t_{VD}$ | $V_{DD}$, $V_{GG}$ Hold | 10 | | 100 | μs | |
| $t_{ACW}$ (2) | Address Complement Set Up | 25 | | | μs | |
| $t_{ACH}$ (2) | Address Complement Hold | 25 | | | μs | |
| $t_{ATW}$ | Address True Set Up | 10 | | | μs | |
| $t_{ATH}$ | Address True Hold | 10 | | | μs | |

Note 2. All 8 address bits must be in the complement state when pulsed $V_{DD}$ and $V_{GG}$ move to their negative levels. The addresses (0 through 255) must be programmed as shown in the timing diagram for a minimum of 32 times.

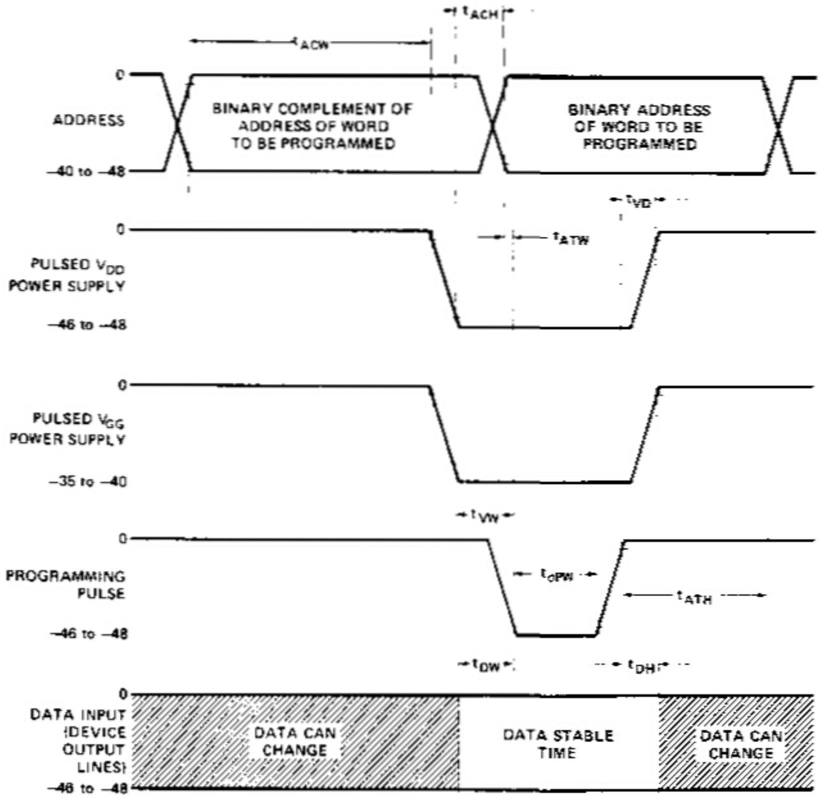
## SWITCHING CHARACTERISTICS FOR PROGRAMMING OPERATION

PROGRAM OPERATION

Conditions of Test:

Input pulse rise and fall times $\leq 1\mu sec$
$\overline{CS}$ = 0V

PROGRAM WAVEFORMS




## PROGRAMMING OPERATION OF THE 8702A

| When the Data Input for the Program Mode is: | Then the Data Output during the Read Mode is: |
|---|---|
| $V_{ILIP}$ = ~−48V pulsed | Logic 1 = $V_{OH}$ = 'P' on tape |
| $V_{IHP}$ = ~ 0V | Logic 0 = $V_{OL}$ = 'N' on tape |

| WORD | ADDRESS $A_7$ | $A_6$ | $A_5$ | $A_4$ | $A_3$ | $A_2$ | $A_1$ | $A_0$ |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| \| | \| | \| | \| | \| | \| | \| | \| | \| |
| \| | \| | \| | \| | \| | \| | \| | \| | \| |
| 255 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Address Logic Level During Read Mode:** Logic 0 = $V_{IL}$ (~ .3V) Logic 1 = $V_{IH}$ (~ 3V)

**Address Logic Level During Program Mode:** Logic 0 = $V_{IL2P}$ (~ −40V) Logic 1 = $V_{IHP}$ (~ 0V)

## PROGRAMMING INSTRUCTIONS FOR THE 8702A

### I. Operation of the 8702A in Program Mode

Initially, all 2048 bits of the ROM are in the "0" state (output low). Information is introduced by selectively programming "1"s (output high) in the proper bit locations.

Word address selection is done by the same decoding circuitry used in the READ mode (see table on page 6 for logic levels). All 8 address bits must be in the binary complement state when pulsed $V_{DD}$ and $V_{GG}$ move to their negative levels. The addresses must be held in their binary complement state for a minimum of 25 μsec after $V_{DD}$ and $V_{GG}$ have moved to their negative levels. The addresses must then make the transition to their true state a minimum of 10 μsec before the program pulse is applied. The addresses should be programmed in the sequence 0 through 255 for a minimum of 32 times. The eight output terminals are used as data inputs to determine the information pattern in the eight bits of each word. A low data input level (−48V) will program a "1" and a high data input level (ground) will leave a "0" (see table on page 6). All eight bits of one word are programmed simultaneously by setting the desired bit information patterns on the data input terminals.

During the programming, $V_{GG}$, $V_{DD}$ and the Program Pulse are pulsed signals.

### II. Programming of the 8702A Using Intel® Microcomputers

Intel provides low cost program development systems which may be used to program its electrically programmable ROMs. Note that the programming specifications that apply to the 8702A are identical to those for Intel's 1702A.

A. Intellec®

The Intellec series of program development systems, the Intellec 8/Mod 8 and Intellec 8/Mod 80, are used as program development tools for the 8008 and 8080 microprocessors respectively. As such, they are equipped with a PROM programmer card and may be used to program Intel's electrically programmable and ultraviolet erasable ROMs.

An ASR-33 teletype terminal is used as the input device. Through use of the Intellec software system monitor, programs to be loaded into PROM may be typed in directly or loaded through the paper tape reader. The system monitor allows the program to be reviewed or altered at will prior to actually programming the PROM. For more complete information on these program development systems, refer to the Intel Microcomputer Catalog or the Intellec Specifications.

B. Users of the SIM8 microcomputer programming systems may also program the 8702A using the MP7-03 programmer card and the appropriate control ROMs:
SIM8 system—Control ROMs A0860, A0861 and A0863.

### III. 8702A Erasing Procedure

The 8702A may be erased by exposure to high intensity short-wave ultraviolet light at a wavelength of 2537A. The recommended integrated dose (i.e., UV intensity x exposure time) is 6W-sec/cm². Examples of ultraviolet sources which can erase the 8702A in 10 to 20 minutes are the Model UVS-54 and Model S-52 short-wave ultraviolet lamps manufactured by Ultra-Violet Products, Inc. (5114 Walnut Grove Avenue, San Gabriel, California). The lamps should be used without short-wave filters, and the 8702A to be erased should be placed about one inch away from the lamp tubes.

# intel® Silicon Gate MOS 8708/8704

## 8192/4096 BIT ERASABLE AND ELECTRICALLY REPROGRAMMABLE READ ONLY MEMORY

- **8708 1024x8 Organization**
- **8704 512x8 Organization**

- **Fast Programming — Typ. 100 sec. For All 8K Bits**
- **Low Power During Programming**
- **Access Time — 450 ns**
- **Standard Power Supplies — +12V, ±5V**
- **Static — No Clocks Required**
- **Inputs and Outputs TTL Compatible During Both Read and Program Modes**
- **Three-State Output — OR-Tie Capability**

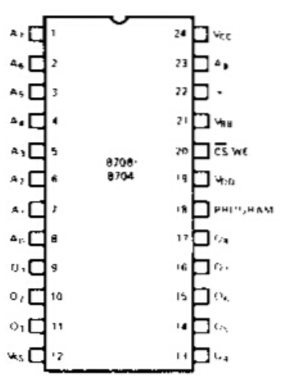
The Intel® 8708/8704 are high speed 8192/4096 bit erasable and electrically reprogrammable ROM's (EPROM) ideally suited where fast turn around and pattern experimentation are important requirements.

The 8708/8704 are packaged in a 24 pin dual-in-line package with transparent 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 devices.

A pin for pin mask programmed ROM, the Intel® 8308, is available for large volume production runs of systems initially using the 8708.

The 8708/8704 is fabricated with the time proven N-channel silicon gate technology.

### PIN CONFIGURATIONS

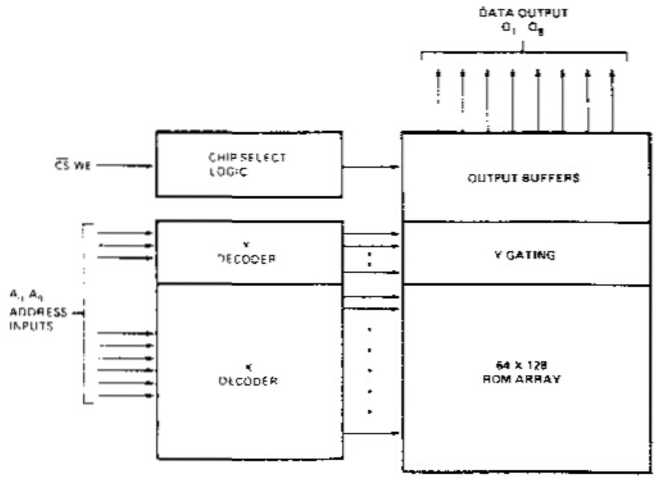



*8704 = $V_{CC}$
8708 = $A_9$

### PIN NAMES

| | |
|---|---|
| $A_0$-$A_9$ | ADDRESS INPUTS |
| $O_1$-$O_8$ | DATA OUTPUTS |
| $\overline{CS}$/WE | CHIP SELECT/WRITE ENABLE INPUT |

### BLOCK DIAGRAM



## Absolute Maximum Ratings*

Temperature Under Bias . . . . . . . . . . -25°C to +85°C
Storage Temperature . . . . . . . . . . -65°C to +125°C
All Input or Output Voltages with Respect to $V_{BB}$
*(except Program)* . . . . . . . . . . +15V to -0.3V
Program Input to $V_{BB}$ . . . . . . . . . . +35V to -0.3V
Supply Voltages $V_{CC}$ and $V_{SS}$ with Respect to $V_{BB}$ . . . . . . . +15V to -0.3V
$V_{DD}$ with Respect to $V_{BB}$ . . . . . . . . . . +20V to -0.3V
Power Dissipation . . . . . . . . . . 1.5W

*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.

# READ OPERATION

## D.C. and Operating Characteristics

$T_A$ = 0°C to 70°C, $V_{CC}$ = +5V ±5%, $V_{DD}$ = +12V ±5%, $V_{BB}$ = -5V ±5%, $V_{SS}$ = 0V, Unless Otherwise Noted.

| Symbol | Parameter | Min. | Typ.[1] | Max. | Unit | Conditions |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Address and Chip Select Input Load Current | | | 10 | µA | $V_{IN}$ = 5.25V |
| $I_{LO}$ | Output Leakage Current | | | 10 | µA | $V_{OUT}$ = 5.25V, $\overline{CS}$/WE = 5V |
| $I_{DD}$ | $V_{DD}$ Supply Current | | 50 | 65 | mA | Worst Case Supply Currents: |
| $I_{CC}$ | $V_{CC}$ Supply Current | | 6 | 10 | mA | All Inputs High |
| $I_{BB}$ | $V_{BB}$ Supply Current | | 30 | 45 | mA | $\overline{CS}$/WE = 5V; $T_A$ = 0°C |
| $V_{IL}$ | Input Low Voltage | $V_{SS}$ | | 0.65 | V | |
| $V_{IH}$ | Input High Voltage | 3.0 | | $V_{CC}$+1 | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.6mA |
| $V_{OH1}$ | Output High Voltage | 3.7 | | | V | $I_{OH}$ = -100µA |
| $V_{OH2}$ | Output High Voltage | 2.4 | | | V | $I_{OH}$ = -1mA |
| $P_D$ | Power Dissipation | | | 800 | mW | $T_A$ = 70°C |

NOTES: 1. Typical values are for $T_A$ = 25°C and nominal supply voltages.
2. The program input (Pin 18) may be tied to $V_{SS}$ or $V_{CC}$ during the read mode.

## A.C. Characteristics

$T_A$ = 0°C to 70°C, $V_{CC}$ = +5V ±5%, $V_{DD}$ = −12V ±5%, $V_{BB}$ = −5V ±5%, $V_{SS}$ = 0V, Unless Otherwise Noted.

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $t_{ACC}$ | Address to Output Delay | | 280 | 450 | ns |
| $t_{\overline{CO}}$ | Chip Select to Output Delay | | | 120 | ns |
| $t_{DF}$ | Chip De-Select to Output Float | 0 | | 120 | ns |
| $t_{OH}$ | Address to Output Hold | 0 | | | ns |

### Capacitance[1] $T_A$ = 25°C, f = 1MHz

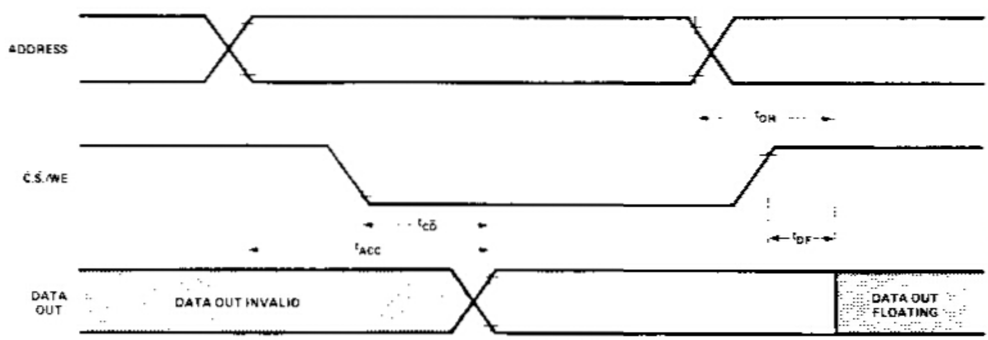
| Symbol | Parameter | Typ. | Max. | Unit | Conditions |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | 4 | 6 | pF | $V_{IN}$=0V |
| $C_{OUT}$ | Output Capacitance | 8 | 12 | pF | $V_{OUT}$=0V |

Note 1. This parameter is periodically sampled and not 100% tested.

### A.C. Test Conditions:

Output Load: 1 TTL gate and $C_L$ = 100pF
Input Rise and Fall Times: ≤20ns
Timing Measurement Reference Levels: 0.8V and 2.8V for inputs; 0.8V and 2.4V for outputs
Input Pulse Levels: 0.65V to 3.0V

### Waveforms



# PROGRAMMING OPERATION

## Description

Initially, and after each erasure, all bits of the 8708/8704 are in the "1" state (Output High). Information is introduced by selectively programming "0" into the desired bit locations.

The circuit is set up for programming operation by raising the $\overline{CS}$/WE input (Pin 20) to +12V. The word address is selected in the same manner as in the read mode. Data to be programmed are presented, 8-bits in parallel, to the data output lines ($O_1$-$O_8$). Logic levels for address and data lines and the supply voltages are the same as for the read mode. After address and data set up one program pulse ($V_P$) per address is applied to the program input (Pin 18). One pass through all addresses to be programmed is defined as a program loop. The number of loops (N) required is a function of the program pulse width ($t_{PW}$) according to $N \times t_{PW} \geqslant 100$ ms.

For program verification, program loops and read loops may be alternated as shown in waveform B.

## Program Characteristics

$T_A = 25°C$, $V_{CC} = +5V \pm 5\%$, $V_{DD} = +12V \pm 5\%$, $V_{BB} = -5V \pm 5\%$, $V_{SS} = 0V$, $\overline{CS}$/WE = +12V, Unless Otherwise Noted.

| Symbol | Parameter | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{AS}$ | Address Setup Time | 10 | | | µs |
| $t_{CSS}$ | $\overline{CS}$/WE Setup Time | 10 | | | µs |
| $t_{DS}$ | Data Setup Time | 10 | | | µs |
| $t_{AH}$ | Address Hold Time | 1 | | | µs |
| $t_{CH}$ | $\overline{CS}$/WE Hold Time | .5 | | | µs |
| $t_{DH}$ | Data Hold Time | 1 | | | µs |
| $t_{DF}$ | Chip Deselect to Output Float Delay | 0 | | 120 | ns |
| $t_{DPR}$ | Program To Read Delay | | | 10 | µs |
| $t_{PW}$ | Program Pulse Width | .1 | | 1.0 | ms |
| $t_{PR}$ | Program Pulse Rise Time | .5 | | 2.0 | µs |
| $t_{PF}$ | Program Pulse Fall Time | .5 | | 2.0 | µs |
| $I_P$ | Programming Current | | 10 | 20 | mA |
| $V_P$ | Program Pulse Amplitude | 25 | | 27 | V |

NOTE: Intel's standard product warranty applies only to devices programmed to specifications described herein.

## Erasing Procedure

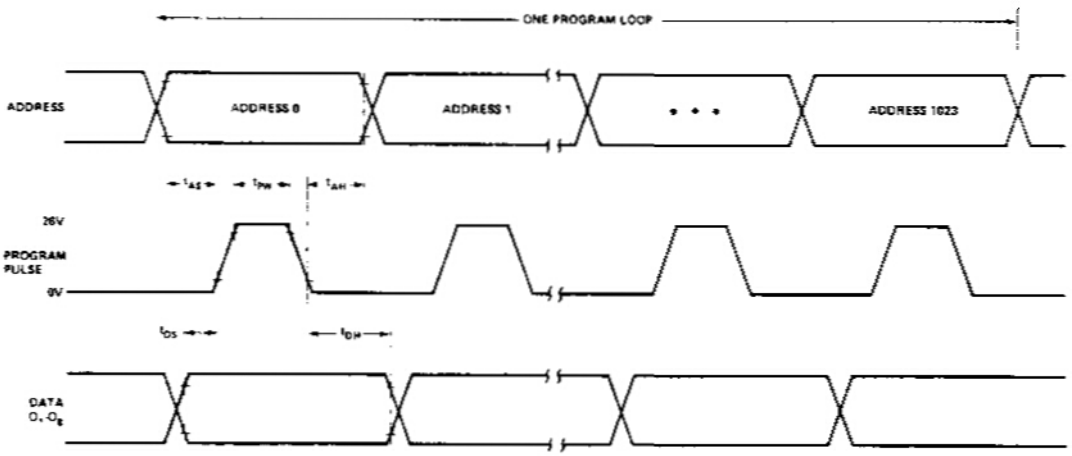
The 8708/8704 may be erased by exposure to high intensity short-wave ultraviolet light at a wavelength of 2537Å. The recommended integrated dose, (i.e., UV intensity x exposure time) is 10W-sec/cm$^2$. Examples of ultraviolet sources which can erase the 8708/8704 in 20 to 30 minutes are the Model UVS-54 and Model S-52 short-wave ultraviolet lamps manufactured by Ultra-Violet Products, Inc. (5114 Walnut Grove Avenue, San Gabriel, California). The lamps should be used without short-wave filters, and the 8708/8704 to be erased should be placed about one inch away from the lamp tubes.

## Waveforms

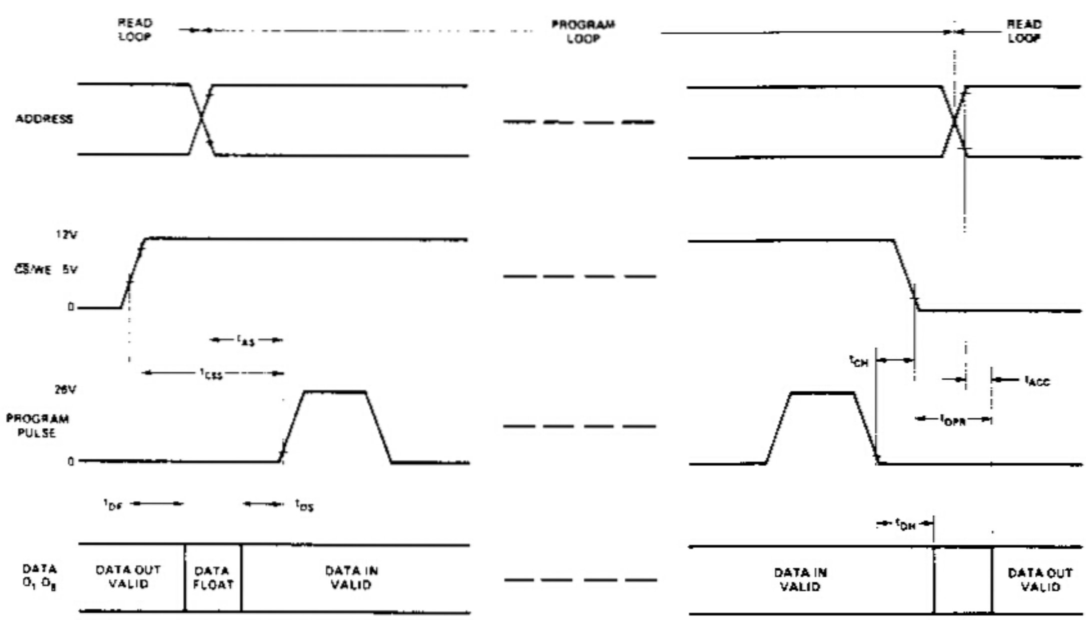
(Logic levels and timing reference levels same as in the Read Mode unless noted otherwise.)

A) Program Mode

$\overline{CS}$/WE = +12V

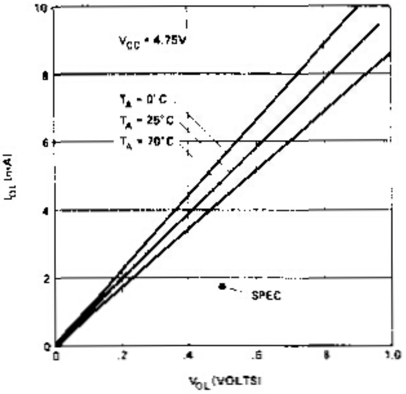



B) Read/Program/Read Transitions

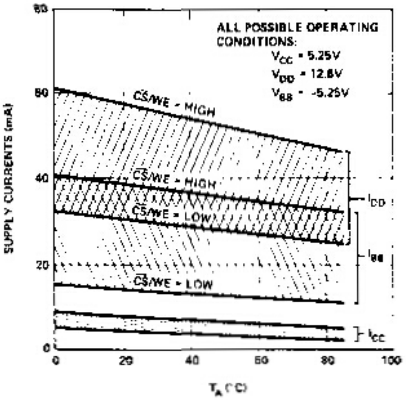


## Typical Characteristics (Nominal supply voltages unless otherwise noted):

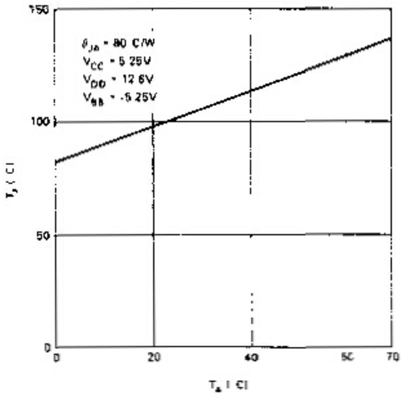
OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



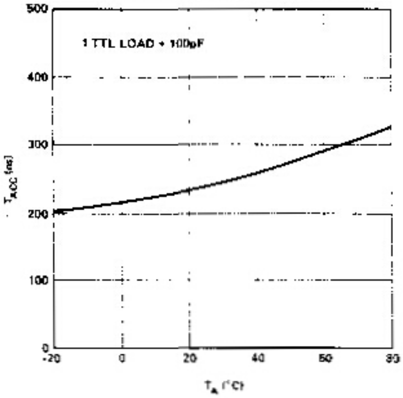

RANGE OF SUPPLY CURRENTS VS. TEMPERATURE



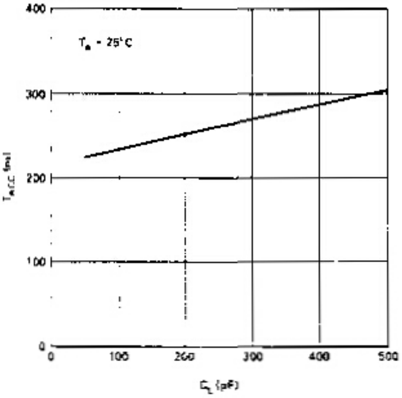

MAXIMUM JUNCTION TEMPERATURE VS. AMBIENT TEMPERATURE




ACCESS TIME VS. TEMPERATURE




ACCESS TIME VS. LOAD CAPACITANCE



intel® Silicon Gate MOS 8302

# 2048 BIT MASK PROGRAMMABLE READ ONLY MEMORY

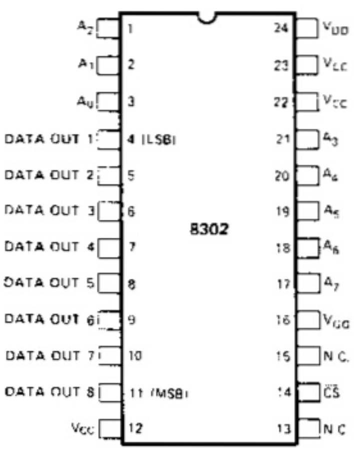
- **Access Time — 1 μsec Max.**
- **Fully Decoded, 256 x 8 Organization**
- **Inputs and Outputs TTL Compatible**
- **Three-State Output — OR-Tie Capability**
- **Static MOS — No Clocks Required**
- **Simple Memory Expansion — Chip Select Input Lead**
- **24-Pin Dual-In-Line Hermetically Sealed Ceramic 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.

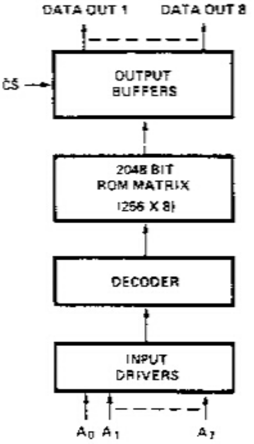
The 8302 is entirely static — no clocks are required. Inputs and outputs of the 8302 are TTL compatible. The output is three-state for OR-tie capability. A separate chip select input allows easy memory expansion. The 8302 is packaged in a 24 pin dual-in-line hermetically sealed ceramic package.

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 |

## Absolute Maximum Ratings*

Ambient Temperature Under Bias . . . . . . . . . 0°C to +70°C
Storage Temperature . . . . . . . . . . . . . . −65°C to +125°C
*Soldering Temperature of Leads (10 sec)* . . . . . . . . . +300°C
Power Dissipation . . . . . . . . . . . . . . . . . . . . . 2 Watts
Input Voltages and Supply
Voltages with respect to $V_{CC}$ . . . . . . . . . . . +0.5V to −20V

*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 at any other condition 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.

# READ OPERATION

## D.C. and Operating Characteristics

$T_A$ = 0°C to 70°C, $V_{CC}$ = +5V±5%, $V_{DD}$ = −9V±5%, $V_{GG}^{(1)}$ = −9V±5%, unless otherwise noted.

| SYMBOL | TEST | MIN. | TYP.[2] | MAX. | UNIT | CONDITIONS | |
|---|---|---|---|---|---|---|---|
| $I_{LI}$ | Address and Chip Select Input Load Current | | | 1 | μA | $V_{IN}$ = 0.0V | |
| $I_{LO}$ | Output Leakage Current | | | 1 | μA | $V_{OUT}$ = 0.0V, $\overline{CS}$ = $V_{CC}$ −2 | |
| $I_{DD0}$ | Power Supply Current | | 5 | 10 | mA | $V_{GG}$=$V_{CC}$, $\overline{CS}$=$V_{CC}$−2, $I_{OL}$ = 0.0mA, $T_A$ = 25°C | |
| $I_{DD1}$ | Power Supply Current | | 35 | 50 | mA | $\overline{CS}$=$V_{CC}$ −2, $I_{OL}$ =0.0mA, $T_A$ = 25°C | Continuous Operation |
| $I_{DD2}$ | Power Supply Current | | 32 | 46 | mA | $\overline{CS}$=0.0, $I_{OL}$ =0.0mA, $T_A$ = 25°C | Continuous Operation |
| $I_{DD3}$ | Power Supply Current | | 38.5 | 60 | mA | $\overline{CS}$=$V_{CC}$ −2, $I_{OL}$ =0.0mA, $T_A$ = 0°C | Continuous Operation |
| $I_{CF1}$ | Output Clamp Current | | 8 | 14 | mA | $V_{OUT}$ = −1.0V, $T_A$ = 0°C | Continuous Operation |
| $I_{CF2}$ | Output Clamp Current | | | 13 | mA | $V_{OUT}$ = −1.0V, $T_A$ = 25°C | Continuous Operation |
| $I_{GG}$ | Gate Supply Current | | | 1 | μA | | |
| $V_{IL1}$ | Input Low Voltage for TTL Interface | −1.0 | | 0.65 | V | | |
| $V_{IL2}$ | Input Low Voltage for MOS Interface | $V_{DD}$ | | $V_{CC}$ −6 | V | | |
| $V_{IH}$ | Address and Chip Select Input High Voltage | $V_{CC}$ −2 | | $V_{CC}$ +0.3 | V | | |
| $I_{OL}$ | Output Sink Current | 1.6 | 4 | | mA | $V_{OUT}$ = 0.45V | |
| $I_{OH}$ | Output Source Current | −2.0 | | | mA | $V_{OUT}$ = 0.0V | |
| $V_{OL}$ | Output Low Voltage | | −.7 | 0.45 | V | $I_{OL}$ = 1.6mA | |
| $V_{OH}$ | Output High Voltage | 3.5 | 4.5 | | V | $I_{OH}$ = −100 μA | |

Note 1. $V_{GG}$ may be clocked to reduce power dissipation. In this mode average $I_{DD}$ increases in proportion to $V_{GG}$ duty cycle.
Note 2. Typical values are at nominal voltages and $T_A$ = 25°C.

## A.C. Characteristics

$T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$, $V_{DD} = -9V \pm 5\%$, $V_{GG} = -9V \pm 5\%$ unless otherwise noted

| SYMBOL | TEST | MINIMUM | TYPICAL | MAXIMUM | UNIT |
|---|---|---|---|---|---|
| Freq. | Repetition Rate | | | 1 | MHz |
| $t_{OH}$ | Previous read data valid | | | 100 | ns |
| $t_{ACC}$ | Address to output delay | | .700 | 1 | µs |
| $t_{DVGG}$ | Clocked $V_{GG}$ set up | 1 | | | µs |
| $t_{CS}$ | Chip select delay | | | 200 | ns |
| $t_{CO}$ | Output delay from $\overline{CS}$ | | | 500 | ns |
| $t_{OD}$ | Output deselect | | | 300 | ns |
| $t_{OHC}$ | Data out hold in clocked $V_{GG}$ mode (Note 1) | | | 5 | µs |

Note 1. The output will remain valid for $t_{OHC}$ as long as clocked $V_{GG}$ is at $V_{CC}$. An address change may occur as soon as the output is sensed (clocked $V_{GG}$ may still be at $V_{CC}$). Data becomes invalid for the old address when clocked $V_{GG}$ is returned to $V_{GG}$.

## Capacitance* $T_A = 25°C$

| SYMBOL | TEST | MINIMUM | TYPICAL | MAXIMUM | UNIT | CONDITIONS |
|---|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | | 5 | 10 | pF | $V_{IN} = V_{CC}$, $\overline{CS} = V_{CC}$, $V_{OUT} = V_{CC}$, $V_{GG} = V_{CC}$ — All unused pins are at A.C. ground |
| $C_{OUT}$ | Output Capacitance | | 5 | 10 | pF | |
| $C_{V_{GG}}$ | $V_{GG}$ Capacitance (Clocked $V_{GG}$ Mode) | | | 30 | pF | |

*This parameter is periodically sampled and is not 100% tested.

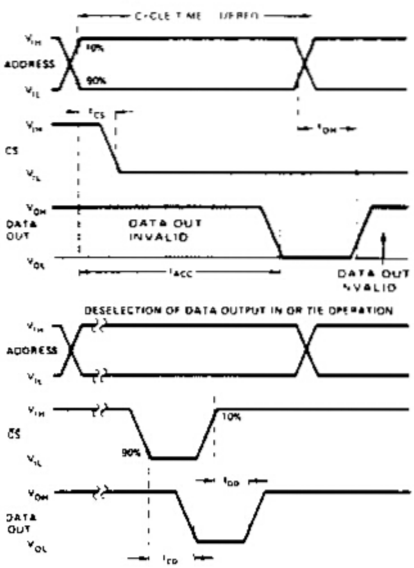
## Switching Characteristics

**Conditions of Test:**

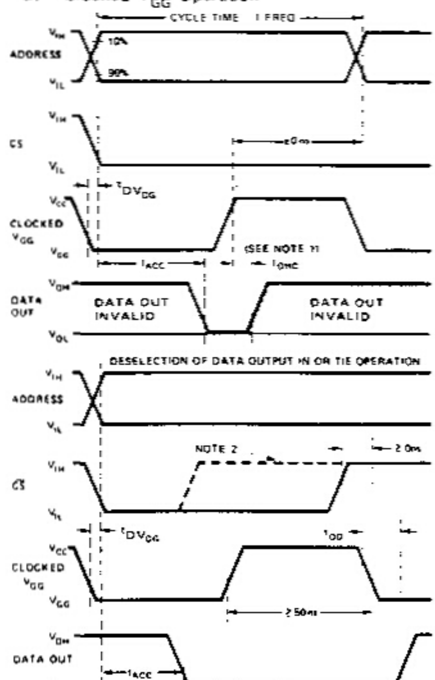
Input pulse amplitudes: 0 to 4V; $t_R$, $t_F \leq 50$ ns

Output load is 1 TTL gate; measurements made at output of TTL gate ($t_{PD} \leq 15$ ns)

**A) Constant $V_{GG}$ Operation**




**B) Clocked $V_{GG}$ Operation**

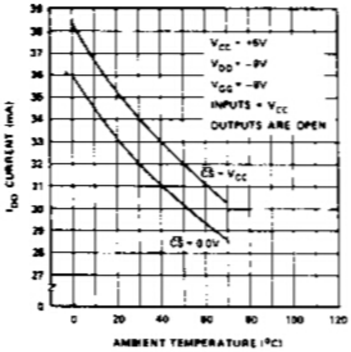



NOTE 1: The output will remain valid for $t_{OHC}$ as long as clocked $V_{GG}$ is at $V_{CC}$. An address change may occur as soon as the output is sensed (clocked $V_{GG}$ may still be at $V_{CC}$). Data becomes invalid for the old address when clocked $V_{GG}$ is returned to $V_{GG}$.

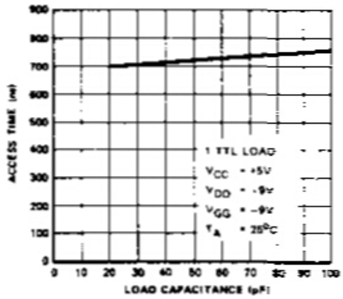
NOTE 2: If $\overline{CS}$ makes a transition from $V_{IL}$ to $V_{IH}$ while clocked $V_{GG}$ is at $V_{GG}$, then deselection of output occurs at $t_{OD}$ as shown in static operation with constant $V_{GG}$.

## Typical Characteristics

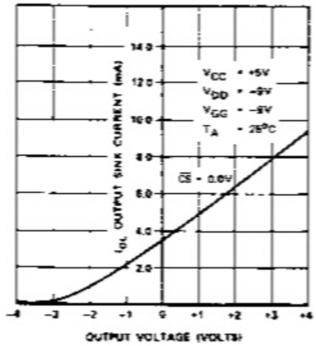
$I_{DD}$ CURRENT VS. TEMPERATURE



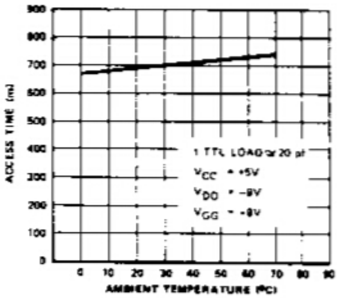

ACCESS TIME VS. LOAD CAPACITANCE



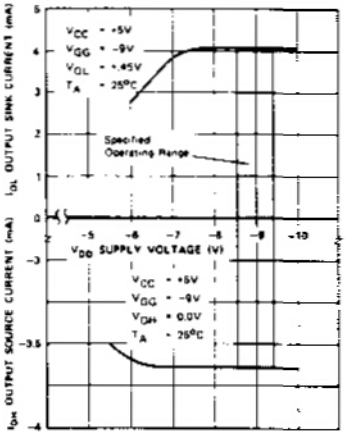

OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



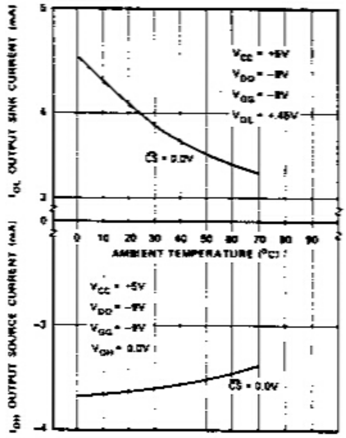

ACCESS TIME VS. TEMPERATURE



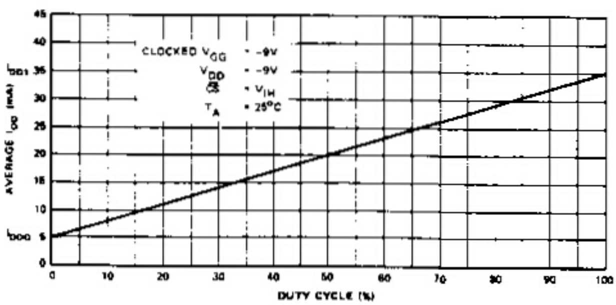

OUTPUT CURRENT VS. $V_{DD}$ SUPPLY VOLTAGE




OUTPUT CURRENT VS. TEMPERATURE




AVERAGE CURRENT VS. DUTY CYCLE FOR CLOCKED $V_{GG}$



# Silicon Gate MOS 8308

## 8192 BIT STATIC MOS READ ONLY MEMORY
### Organization -- 1024 Words x 8 Bits

- Fast Access – 450 ns
- 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
- Standard Power Supplies +12V DC, ±5V DC

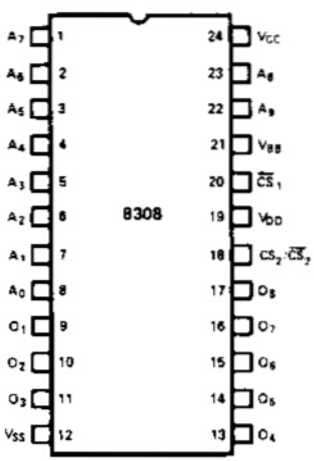
The Intel® 8308 is an 8,192 bit static MOS mask programmable Read Only Memory organized as 1024 words by 8-bits. This ROM is designed for 8080 microcomputer system applications where high performance, large bit storage, and simple interfacing are important design objectives. The inputs and outputs are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

A pin for pin compatible electrically programmed erasable ROM, the Intel® 8708, is available for system development and small quantity production use.

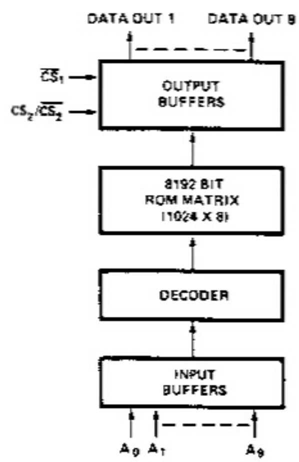
Two Chip Selects are provided – $\overline{CS}_1$ which is negative true, and $CS_2/\overline{CS}_2$ which may be programmed either negative or positive true at the mask level.

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 |
| $\overline{CS}_1$, $CS_2$ | CHIP SELECT INPUTS |

## Absolute Maximum Ratings*

Ambient Temperature Under Bias . . . . . . . -25°C to +85°C
Storage Temperature . . . . . . . . . . . . . . -65°C to +150°C
Voltage On Any Pin With Respect
To $V_{BB}$ . . . . . . . . . . . . . . . . . . . . . -0.3V to 20V
Power Dissipation . . . . . . . . . . . . . . . . . . . . 1.0 Watt

*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.

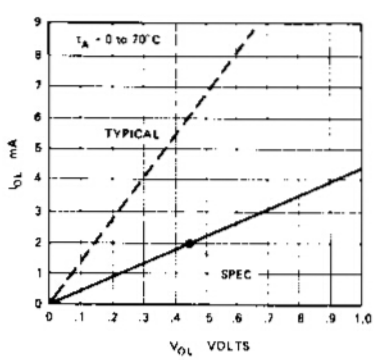
## D.C. and Operating Characteristics

$T_A$ = 0°C to +70°C, $V_{CC}$ = 5V ±5%; $V_{DD}$ = 12V ±5%, $V_{BB}$ = -5V ±5%, $V_{SS}$ = 0V Unless Otherwise Specified.

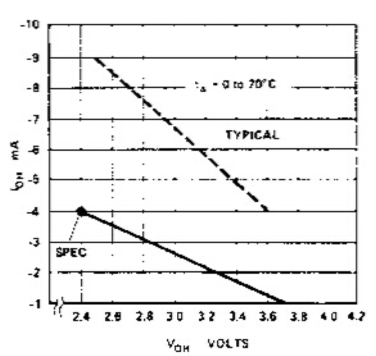
| Symbol | Parameter | Limits | | | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| | | Min. | Typ.(1) | Max. | | |
| $I_{LI}$ | Input Load Current (All Input Pins Except $\overline{CS}_1$) | | | 10 | µA | $V_{IN}$ = 0 to 5.25V |
| $I_{LCL}$ | Input Load Current on $\overline{CS}_1$ | | | 1.6 | mA | $V_{IN}$ = 0.45V |
| $I_{LPC}$ | Input Peak Load Current on $\overline{CS}_1$ | | | 4 | mA | $V_{IN}$ = 0.8V to 3.3V |
| $I_{LKC}$ | Input Leakage Current on $\overline{CS}_1$ | | | 10 | µA | $V_{IN}$ = 3.3V to 5.25V |
| $I_{LO}$ | Output Leakage Current | | | 10 | µA | Chip Deselected |
| $V_{IL}$ | Input "Low" Voltage | $V_{SS}$-1 | | 0.8V | V | |
| $V_{IH}$ | Input "High" Voltage | 3.3 | | $V_{CC}$+1.0 | V | |
| $V_{OL}$ | Output "Low" Voltage | | | 0.45 | V | $I_{OL}$ = 2mA |
| $V_{OH1}$ | Output "High" Voltage | 2.4 | | | V | $I_{OH}$ = -4mA |
| $V_{OH2}$ | Output "High" Voltage | 3.7 | | | V | $I_{OH}$ = -1mA |
| $I_{CC}$ | Power Supply Current $V_{CC}$ | | .8 | 2 | mA | |
| $I_{DD}$ | Power Supply Current $V_{DD}$ | | 32 | 60 | mA | |
| $I_{BB}$ | Power Supply Current $V_{BB}$ | | 10µA | 1 | mA | |
| $P_D$ | Power Dissipation | | | 775 | mW | |

NOTE 1: Typical values for $T_A$ = 25°C and nominal supply voltage

D.C. OUTPUT CHARACTERISTICS




D.C. OUTPUT CHARACTERISTICS



## A.C. Characteristics

$T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$; $V_{DD} = +12V \pm 5\%$, $V_{BB} = -5V \pm 5\%$, $V_{SS} = 0V$, Unless Otherwise Specified.

| Symbol | Parameter | Limits[2] | | | Unit |
|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | |
| $t_{ACC}$ | Address to Output Delay Time | | 200 | 450 | ns |
| $t_{CO_1}$ | Chip Select 1 to Output Delay Time | | 85 | 160 | ns |
| $t_{CO_2}$ | Chip Select 2 to Output Delay Time | | 125 | 220 | ns |
| $t_{DF}$ | Chip Deselect to Output Data Float Time | | 125 | 220 | ns |

NOTE 2: Refer to conditions of Test for A.C. Characteristics. Add 50 nanoseconds (worst case) to specified values at $V_{OH} = 3.7V$ @ $I_{OH} = -1mA$, $C_L = 100pF$.

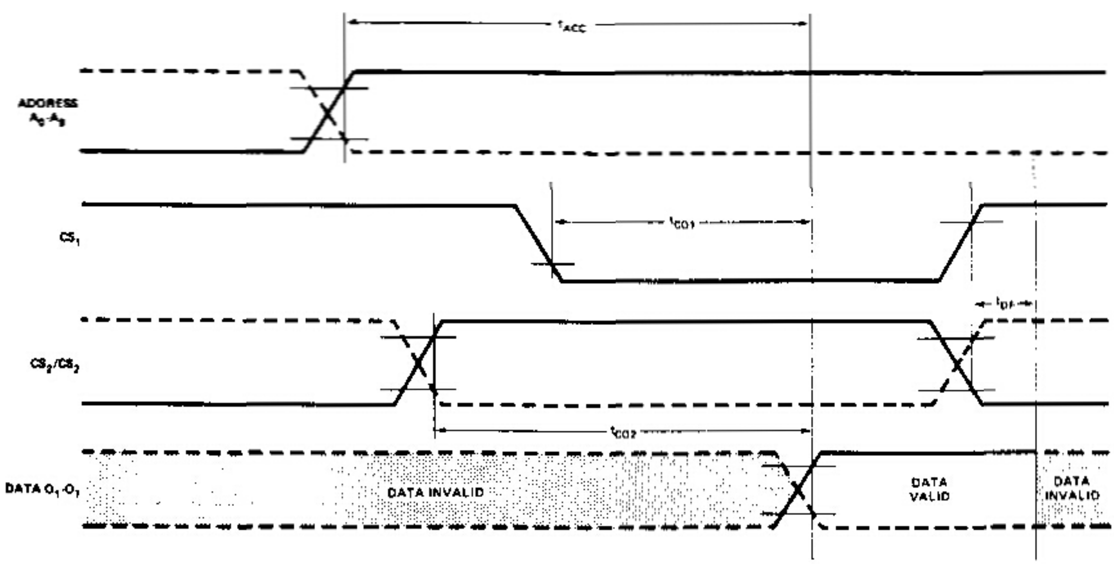
### CONDITIONS OF TEST FOR A.C. CHARACTERISTICS

Output Load . . . . . . . . 1 TTL Gate, and $C_{LOAD} = 100pF$
Input Pulse Levels . . . . . . . . . . . . . . . . . . .65V to 3.3V
Input Pulse Rise and Fall Times . . . . . . . . . . . . 20 nsec
Timing Measurement Reference Level
. . . . . . . . . . . . . . . . . . . 2.4V $V_{IH}$, $V_{OH}$; 0.8V $V_{IL}$, $V_{OL}$

### CAPACITANCE

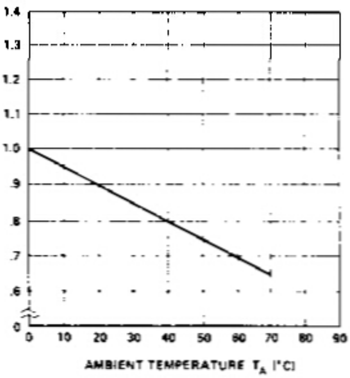
$T_A = 25°C$, f = 1 MHz, $V_{BB} = -5V$, $V_{DD}$, $V_{CC}$ and all other pins tied to $V_{SS}$.

| Symbol | Test | Limits | |
|---|---|---|---|
| | | Typ. | Max. |
| $C_{IN}$ | Input Capacitance | | 6pF |
| $C_{OUT}$ | Output Capacitance | | 12pF |

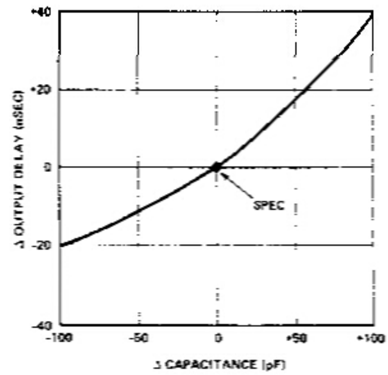


## Typical Characteristics (Nominal supply voltages unless otherwise noted.)

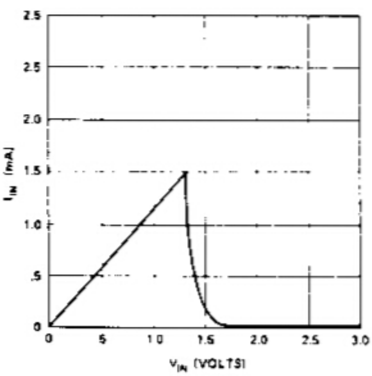
$I_{DD}$ VS. TEMPERATURE (NORMALIZED)



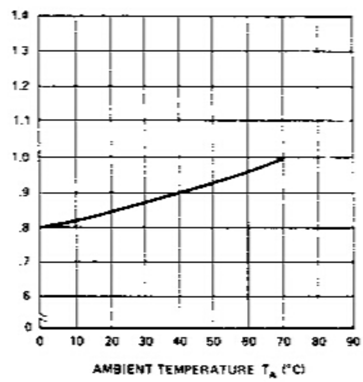

Δ OUTPUT CAPACITANCE VS. Δ OUTPUT DELAY




$\overline{CS_1}$ INPUT CHARACTERISTICS




$T_{ACC}$ VS. TEMPERATURE (NORMALIZED)



# MCS™ CUSTOM ROM ORDER FORM

# 8308 ROM

CUSTOMER ____________________

P.O. NUMBER ____________________

DATE ____________________

For Intel use only

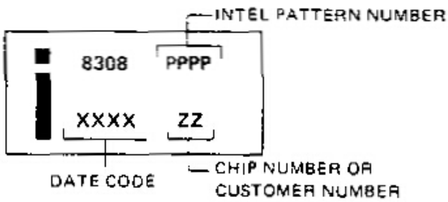
| | |
|---|---|
| S# ____________ | PPPP ____________ |
| STD ____________ | ZZ ____________ |
| ____________ | DD ____________ |
| APP ____________ | DATE ____________ |

All custom 8308 ROM orders must be submitted on this form. Programming information should be sent in the form of computer punched card: or punched paper tape per the formats designated on this order form. Additional forms are available from Intel.

## MARKING

The marking as shown at the right must contain the Intel logo, the product type (P8308), the 4-digit Intel pattern number (PPPP), a date code (XXXX), and the 2-digit chip number (DD). An optional customer identification number may be substituted for the chip number (ZZ). Optional Customer Number (maximum 9 characters or spaces).

CUSTOMER NUMBER ____________________




## MASK OPTION SPECIFICATIONS

**A. CHIP NUMBER (CHIP SELECT OPTION)**

Must be specified 0 or 1.

The chip number will be coded in terms of positive logic where a logic "1" is high level input.

**Chip Select Truth Table**

| Chip Number | $\overline{CS1}$ | CS2 | Selected |
|---|---|---|---|
| 0 | 0 | 0 | Yes |
| 1 | 0 | 1 | Yes |
| 0 | 1 | 0 | No |
| 1 | 1 | 1 | No |

Chip Number ____________

B. ROM Truth Table Format

Programming information should be sent in the form of computer punched cards or punched paper tape. In either case, a printout of the truth table should be accompanied with the order.

The following general format is applicable to the programming information sent to Intel:

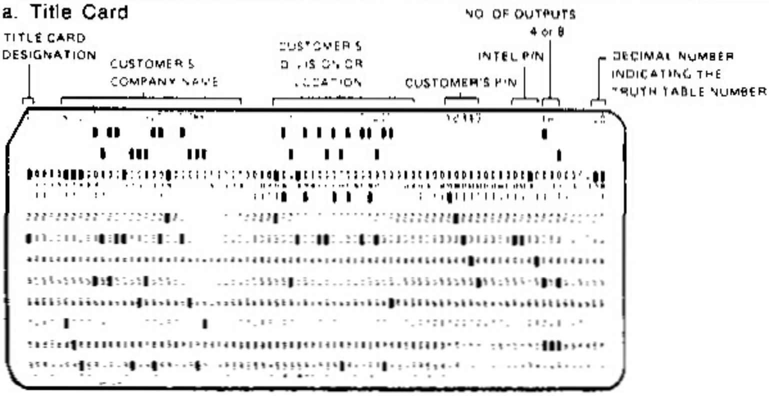
- Data fields should be ordered beginning with the least significant address (0000) and ending with the most significant address (1023).
- A data field should start with the most significant bit and end with the least significant bit.
- The data field should consist of P's and N's. A P is to indicate a high level output (most positive) and an N a low level output (most negative). In terms of positive logic, a P is defined as a logic "1" and an N is defined as a logic "0". If the programming information is sent on a punched paper tape, then a start character, B, and an end character, F, must be used in the data field. See paragraph 2.

1. Punched Card Format

An 80-column Hollerith card (preferably interpreted) punched by an IBM 026 or 029 keypunch should be submitted. The first card will be a title card; the format is as follows:

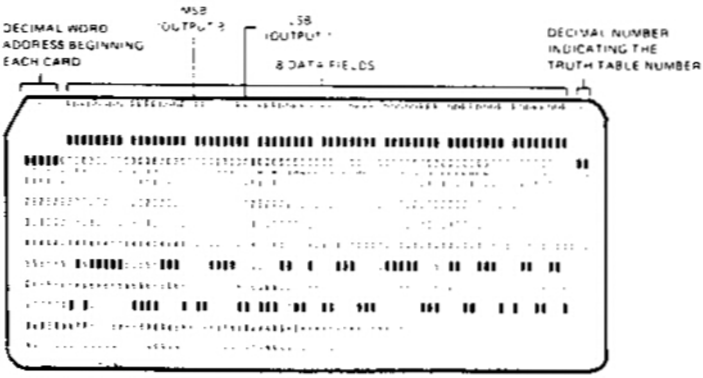
# MCS™ CUSTOM ROM ORDER FORM 8308

a. Title Card




| Column | Data |
|---|---|
| 1 | Punch a T |
| 2–5 | Blank |
| 6–30 | Customer Company Name |
| 31–34 | Blank |
| 35–54 | Customer's Company Division or location |
| 55–57 | Blank |
| 58–66 | Customer Part Number |
| 67 | Blank |
| 68–75 | Punch the Intel 4-digit basic part number and in ( ) the number of output bits e.g., 8308(8). |
| 76–78 | Blank |
| 79–80 | Punch a 2-digit decimal number to identify the truth table number (mask programmed chip select number). |

b. For a 1024 word X 8-bit organization only, cards 2 and the following cards should be punched as shown.




| Column | Data |
|---|---|
| 1–5 | Punch the 5-digit decimal equivalent of the binary coded location which begins each card. The address is right justified, i.e., 00000, 00008, 00016 etc. |
| 6 | Blank |
| 7–14 | Data Field |
| 15 | Blank |
| 16–23 | Data Field |
| 33 | Blank |
| 34–41 | Data Field |
| 42 | Blank |
| 43–50 | Data Field |
| 51 | Blank |
| 52–59 | Data Field |
| 60 | Blank |
| 61–68 | Data Field |
| 69 | Blank |
| 70–77 | Data Field |
| 78 | Blank |
| 79–80 | Punch same 2-digit decimal number as in title card. |

2. Paper Tape Format

1" wide paper tape using 7- or 8-bit ASCII code, such as a model 33 ASR teletype produces, or the 11/16" wide paper tape using a 5-bit Baudot code, such as a Telex produces.

The format requirements are as follows:

a. All word fields are to be punched in consecutive order, starting with word field 0 (all addresses low). There must be exactly 1024 word fields for the 1024 X 8 ROM organization.

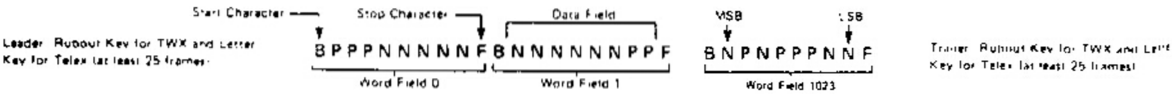
b. Each word field must begin with the start character B and end with the stop character F. There must be exactly 8 data characters between the B and F. NO OTHER CHARACTERS, SUCH AS RUBOUTS, ARE ALLOWED ANYWHERE IN A WORD FIELD. If in preparing a tape an error is made, the entire word field, including the B and F, must be rubbed out. Within the word field, a P results in a high level output and an N results in a low level output.

c. Preceding the first word field and following the last word field, there must be a leader/trailer length of *at least 25 characters. This should consist of rubout or null punches (letter key for Telex tapes).*

d. Between word fields, comments not containing B's or F's may be inserted. Carriage return and line feed characters should be inserted as a "comment") just before each word field (or at least between every four word fields). When these carriage returns, etc., are inserted, the tape may be easily listed on the teletype for purposes of error checking. The customer may also find it helpful to insert the word number (as a comment) at least every four word fields.

e. Included in the tape before the leader should be the customer's complete Telex or TWX number and, if more than one pattern is being transmitted, the ROM pattern number.

f. MSB and LSB are the most and least significant bit of the device outputs. Refer to the data sheet for the pin numbers.



# Silicon Gate MOS ROM 8316A

## 16,384 BIT STATIC MOS READ ONLY MEMORY
## Organization—2048 Words x 8 Bits
## Access Time-850 ns max

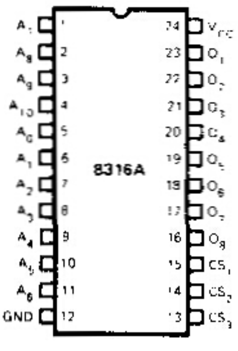
- **Single +5 Volts Power Supply Voltage**
- **Directly TTL Compatible — All Inputs and Outputs**
- **Low Power Dissipation of 31.4 μW/Bit Maximum**
- **Three Programmable Chip Select Inputs for Easy Memory Expansion**
- **Three-State Output — OR-Tie Capability**
- **Fully Decoded — On Chip Address Decode**
- **Inputs Protected — All Inputs Have Protection Against Static Charge**

The Intel® 8316A 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.

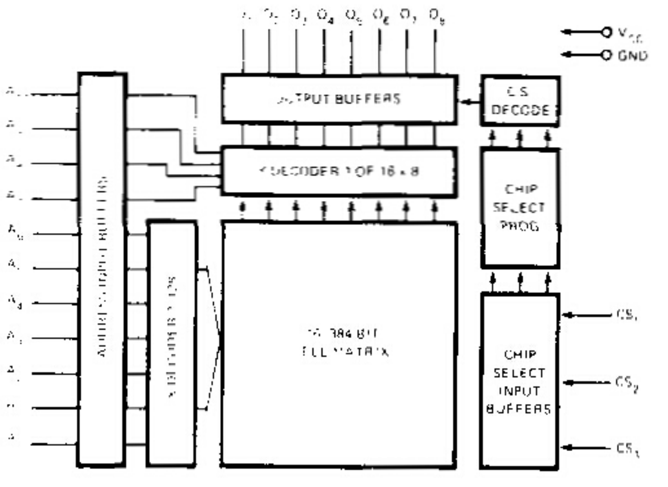
The inputs and outputs are fully TTL compatible. This device operates with a single +5V power supply. The three chip select inputs are programmable. Any combination of active high or low level chip select inputs can be defined and the desired chip select code is fixed during the masking process. These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

The 8316A 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 |

## ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias . . . . . . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . . . . . . . . . . -65°C to +150°C
Voltage On Any Pin With Respect
To Ground . . . . . . . . . . . . . . . . . . . . . . -0.5V to +7V
Power Dissipation . . . . . . . . . . . . . . . . . . . . . 1.0 Watt

*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. AND OPERATING CHARACTERISTICS

$T_A = 0°C$ to $+70°C$, $V_{CC} = 5V \pm 5\%$ unless otherwise specified

| SYMBOL | PARAMETER | LIMITS MIN. | TYP.[1] | MAX. | UNIT | TEST CONDITIONS |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Load Current (All Input Pins) | | | 10 | µA | $V_{IN} = 0$ to 5.25V |
| $I_{LOH}$ | Output Leakage Current | | | 10 | µA | CS = 2.2V, $V_{OUT} = 4.0V$ |
| $I_{LOL}$ | Output Leakage Current | | | -20 | µA | CS = 2.2V, $V_{OUT} = 0.45V$ |
| $I_{CC}$ | Power Supply Current | | 40 | 98 | mA | All inputs 5.25V Data Out Open |
| $V_{IL}$ | Input "Low" Voltage | -0.5 | | 0.8 | V | |
| $V_{IH}$ | Input "High" Voltage | 2.0 | | $V_{CC}$+1.0V | V | |
| $V_{OL}$ | Output "Low" Voltage | | | 0.45 | V | $I_{OL} = 2.0$ mA |
| $V_{OH}$ | Output "High" Voltage | 2.2 | | | V | $I_{OH} = -100$ µA |

(1) Typical values for $T_A = 25°C$ and nominal supply voltage.

## A.C. CHARACTERISTICS

$T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$ unless otherwise specified

| SYMBOL | PARAMETER | LIMITS MIN. | TYP.[1] | MAX. | UNIT |
|---|---|---|---|---|---|
| $t_A$ | Address to Output Delay Time | | 400 | 850 | nS |
| $t_{CO}$ | Chip Select to Output Enable Delay Time | | | 300 | nS |
| $t_{DF}$ | Chip Deselect to Output Data Float Delay Time | 0 | | 300 | nS |

## CONDITIONS OF TEST FOR A.C. CHARACTERISTICS

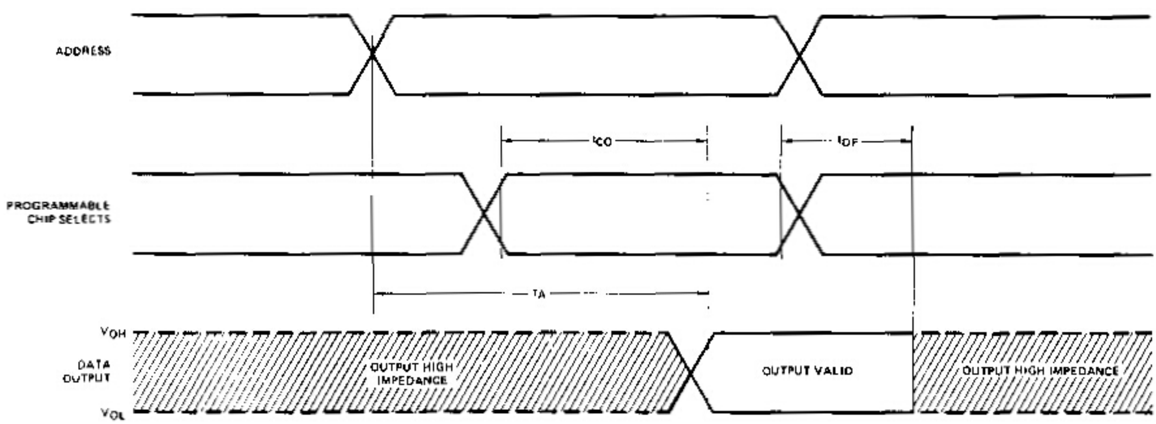
Output Load . . . 1 TTL Gate, and $C_{LOAD} = 100$ pF
Input Pulse Levels . . . . . . . . . . . . . . . . 0.8 to 2.0V
Input Pulse Rise and Fall Times . (10% to 90%) 20 nS
Timing Measurement Reference Level
Input . . . . . . . . . . . . . . . . . . . . . . . . . . 1.5V
Output . . . . . . . . . . . . . . . . . 0.45V to 2.2V

## CAPACITANCE(2) $T_A = 25°C$, f = 1 MHz

| SYMBOL | TEST | LIMITS TYP. | MAX. |
|---|---|---|---|
| $C_{IN}$ | All Pins Except Pin Under Test Tied to AC Ground | 4 pF | 10 pF |
| $C_{OUT}$ | All Pins Except Pin Under Test Tied to AC Ground | 8 pF | 15 pF |

(2) This parameter is periodically sampled and is not 100% tested.

## WAVEFORMS

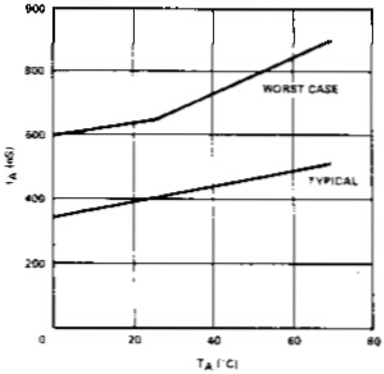



## 16K ROM PROTOTYPING

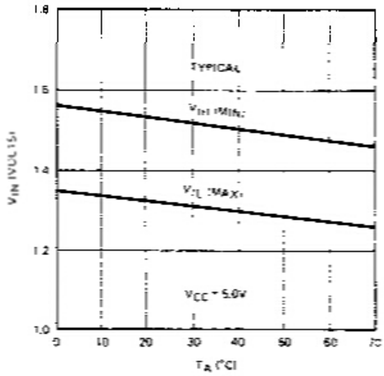
ROM systems may be developed and programs may be verified using Intel's 1702A or 2708 PROMs.

## TYPICAL D.C. CHARACTERISTICS

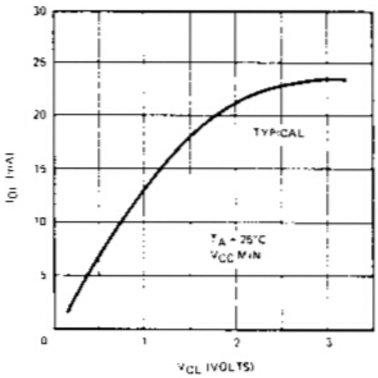
ACCESS TIME VS. AMBIENT TEMPERATURE



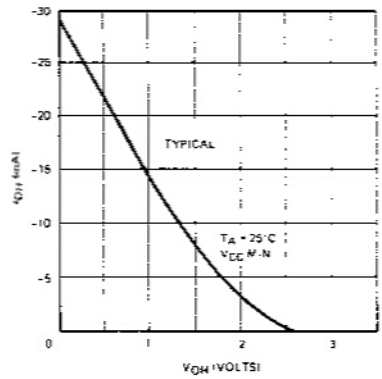

$V_{IN}$ LIMITS VS. TEMPERATURE



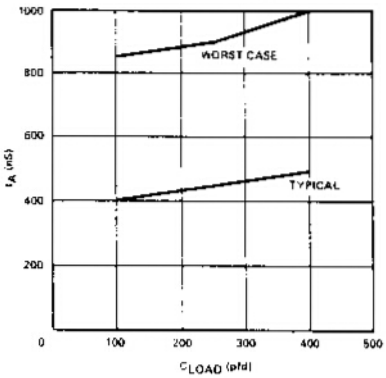

OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



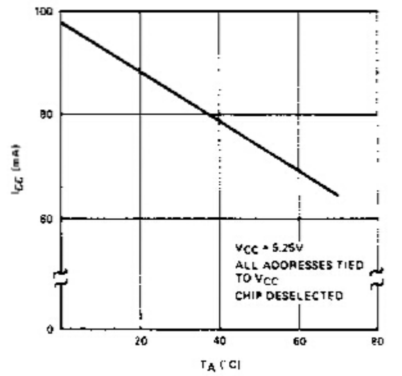

OUTPUT SOURCE CURRENT VS. OUTPUT VOLTAGE




ACCESS TIME VS. LOAD CAPACITANCE




STATIC $I_{CC}$ VS. AMBIENT TEMPERATURE WORST CASE



intel®

# MCS™ CUSTOM ROM ORDER FORM

# 8316A ROM

CUSTOMER ______________________

P.O. NUMBER ______________________

DATE ______________________

For Intel use only

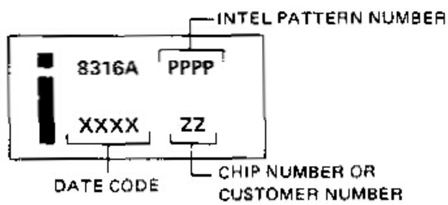
| | |
|---|---|
| S# ________ | PPPP ________ |
| STD ________ | ZZ ________ |
| ________ | DD ________ |
| APP ________ | DATE ________ |

All custom 8316A ROM orders must be submitted on this form. Programming information should be sent in the form of computer punched cards or punched paper tape per the formats designated on this order form. Additional forms are available from Intel.

## MARKING

The marking as shown at the right must contain the Intel® logo, the product type (P8316A), the 4-digit Intel pattern number (PPPP), a date code (XXXX), and the 2-digit chip number (DD). An optional customer identification number may be substituted for the chip number (ZZ). Optional Customer Number (maximum 9 characters or spaces).

CUSTOMER NUMBER ______________________




## MASK OPTION SPECIFICATIONS

**A. CHIP NUMBER** ________ (Must be specified—any number from 0 through 7—DD).

The chip number will be coded in terms of positive logic where a logic "1" is a high level input.

| Chip Number | CS3 | CS2 | CS1 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 2 | 0 | 1 | 0 |
| 3 | 0 | 1 | 1 |
| 4 | 1 | 0 | 0 |
| 5 | 1 | 0 | 1 |
| 6 | 1 | 1 | 0 |
| 7 | 1 | 1 | 1 |

B. ROM Truth Table Format

Programming information should be sent in the form of computer punched cards or punched paper tape. In either case, a printout of the truth table should be accompanied with the order.

The following general format is applicable to the programming information sent to Intel:

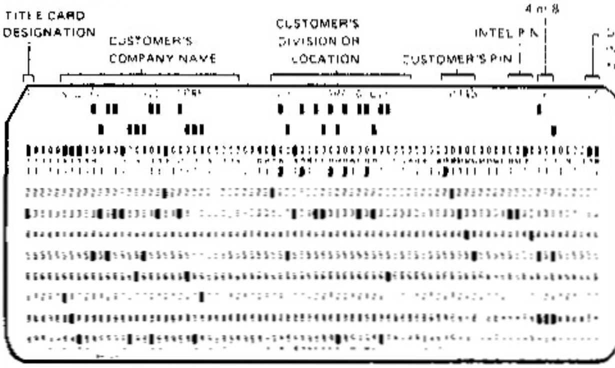
- Data fields should be ordered beginning with the least significant address (0000) and ending with the most significant address (2047).
- A data field should start with the most significant bit and end with the least significant bit.
- The data field should consist of P's and N's. A P is to indicate a high level output (most positive) and an N a low level output (most negative). In terms of positive logic, a P is defined as a logic "1" and an N is defined as a logic "0". If the programming information is sent on a punched paper tape, then a start character, B, and an end character, F, must be used in the data field.

1. Punched Card Format

An 80-column Hollerith card (preferably interpreted) punched by an IBM 026 or 029 keypunch should be submitted. The first card will be a title card; the format is as follows:

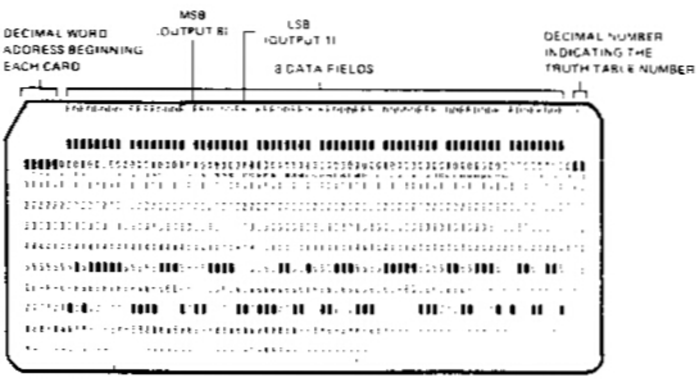
# MCS™ CUSTOM ROM ORDER FORM

a. Title Card




| Column | Data |
|---|---|
| 1 | Punch a T |
| 2–5 | Blank |
| 6–30 | Customer Company Name |
| 31–34 | Blank |
| 35–54 | Customer's Company Division or location |
| 55–57 | Blank |
| 58–66 | Customer Part Number |
| 67 | Blank |
| 68–75 | Punch the Intel 4-digit basic part number and in ( ) the number of output bits, e.g., 8316A(8). |
| 76–78 | Blank |
| 79–80 | Punch a 2-digit decimal number to identify the truth table number (mask programmed chip select number). |

b. For a 2048 word X 8-bit organization only, cards 2 and the following cards should be punched as shown.




| Column | Data |
|---|---|
| 1–5 | Punch the 5-digit decimal equivalent of the binary coded location which begins each card. The address is right justified, i.e., 00000, 00008, 00016, etc. |
| 6 | Blank |
| 7–14 | Data Field |
| 15 | Blank |
| 16–23 | Data Field |
| 33 | Blank |
| 34–41 | Data Field |
| 42 | Blank |
| 43–50 | Data Field |
| 51 | Blank |
| 52–59 | Data Field |
| 60 | Blank |
| 61–68 | Data Field |
| 69 | Blank |
| 70–77 | Data Field |
| 78 | Blank |
| 79–80 | Punch same 2-digit decimal number as in title card. |

2. Paper Tape Format

1" wide paper tape using 7- or 8-bit ASCII code, such as a model 33 ASR teletype produces, or the 11/16" wide paper tape using a 5-bit Baudot code, such as a Telex produces.

The format requirements are as follows:

a. All word fields are to be punched in consecutive order, starting with word field 0 (all addresses low). There must be exactly 2048 word fields for the 2048 X 8 ROM organization.

b. Each word field must begin with the start character B and end with the stop character F. There must be exactly 8 data characters between the B and F. NO OTHER CHARACTERS, SUCH AS RUBOUTS, ARE ALLOWED ANYWHERE IN A WORD FIELD. If in preparing a tape an error is made, the entire word field, including the B and F, must be rubbed out. Within the word field, a P results in a high level output and an N results in a low level output.

c. Preceding the first word field and following the last word field, there must be a leader/trailer length of *at least 25 characters. This should consist of rubout or null punches (letter key for Telex tapes).*

d. Between word fields, comments not containing B's or F's may be inserted. Carriage return and line feed characters should be inserted as a "comment") just before each word field (or at least between every four word fields). When these carriage returns, etc., are inserted, the tape may be easily listed on the teletype for purposes of error checking. The customer may also find it helpful to insert the word number (as a comment) at least every four word fields.

e. Included in the tape before the leader should be the customer's complete Telex or TWX number and, if more than one pattern is being transmitted, the ROM pattern number.

f. MSB and LSB are the most and least significant bit of the device outputs. Refer to the data sheet for the pin numbers.

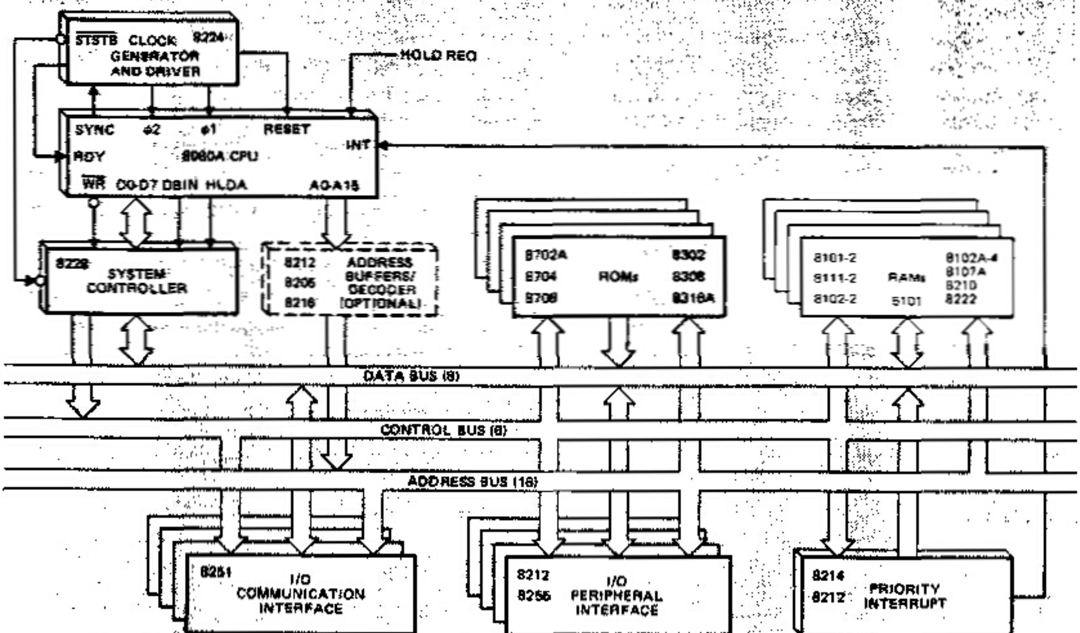
intel® Microcomputer Systems

# RAMs

8101-2
8111-2

8102-2
8102A-4
8107B-4

5101
8210
8222



# Silicon Gate MOS 8101-2

## 1024 BIT (256 x 4) STATIC MOS RAM WITH SEPARATE I/O

- **256 x 4 Organization to Meet Needs *for Small System Memories***
- **Access Time — 850 nsec 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**
- **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**
- **Output Disable Provided for Ease of Use in Common Data Bus Systems**

The Intel® 8101-2 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.

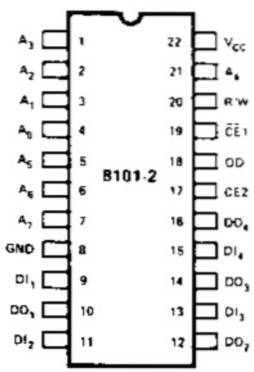
The 8101-2 is designed for memory applications where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, outputs, and a single +5V supply. Two chip-enables allow easy selection of an individual package when outputs are OR-tied. An output disable is provided so that data inputs and outputs can be tied for common I/O systems. Output disable is then used to eliminate any bidirectional logic.

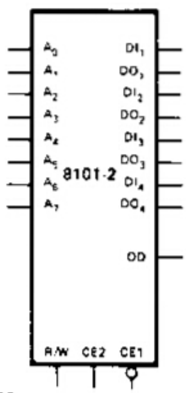
The Intel® 8101-2 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

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

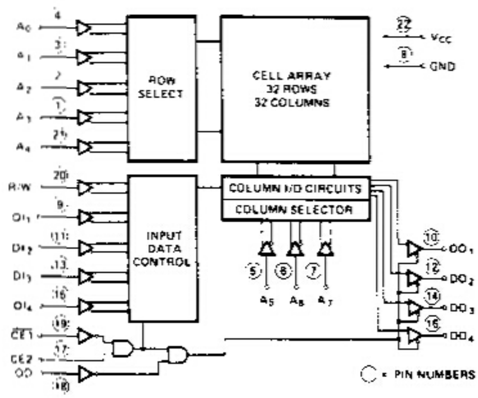
**PIN CONFIGURATION**




**LOGIC SYMBOL**




**BLOCK DIAGRAM**




**PIN NAMES**

| | | | |
|---|---|---|---|
| $D_{IN}$ | DATA INPUT | OD | OUTPUT DISABLE |
| $A_0 - A_7$ | ADDRESS INPUTS | $D_{OUT}$ | DATA OUTPUT |
| R/W | READ/WRITE INPUT | $V_{CC}$ | POWER (+5V) |
| $\overline{CE1}$, CE2 | CHIP ENABLE | | |

## Absolute Maximum Ratings*

Ambient Temperature Under Bias . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . . . . . -65°C to +150°C
Voltage On Any Pin
With Respect to Ground . . . . . . . . . . -0.5V to +7V
Power Dissipation . . . . . . . . . . . . . . . . . . . . 1 Watt

*COMMENT:

*Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition 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.*

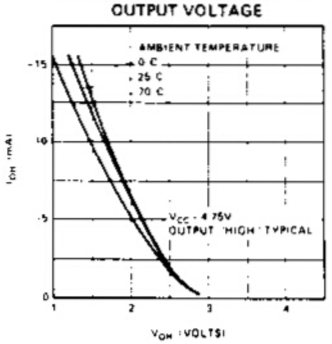
## D.C. and Operating Characteristics

$T_A$ = 0°C to 70°C, $V_{CC}$ = 5V ±5% unless otherwise specified.

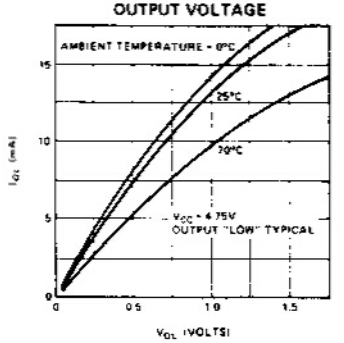
| Symbol | Parameter | Min. | Typ.[1] | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Current | | | 10 | µA | $V_{IN}$ = 0 to 5.25V |
| $I_{LOH}$ | I/O Leakage Current[2] | | | 15 | µA | $\overline{CE}$ = 2.2V, $V_{OUT}$ = 4.0V |
| $I_{LOL}$ | I/O Leakage Current[2] | | | -50 | µA | $\overline{CE}$ = 2.2V, $V_{OUT}$ = 0.45V |
| $I_{CC1}$ | Power Supply Current | | 30 | 60 | mA | $V_{IN}$ = 5.25V, $I_O$ = 0mA $T_A$ = 25°C |
| $I_{CC2}$ | Power Supply Current | | | 70 | mA | $V_{IN}$ = 5.25V, $I_O$ = 0mA $T_A$ = 0°C |
| $V_{IL}$ | Input "Low" Voltage | -0.5 | | +0.65 | V | |
| $V_{IH}$ | Input "High" Voltage | 2.2 | | $V_{CC}$ | V | |
| $V_{OL}$ | Output "Low" Voltage | | | +0.45 | V | $I_{OL}$ = 2.0mA |
| $V_{OH}$ | Output "High" Voltage | 2.2 | | | V | $I_{OH}$ = -150 µA |

NOTE: 1. Typical values are for $T_A$ = 25°C and nominal supply voltage.
2. Input and Output tied together.

OUTPUT SOURCE CURRENT VS. OUTPUT VOLTAGE




OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



## A.C. Characteristics

READ CYCLE $T_A = 0°C$ to $70°C$, $V_{CC} = 5V \pm 5\%$, unless otherwise specified.

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{RCY}$ | Read Cycle | 850 | | | ns | (See below) |
| $t_A$ | Access Time | | | 850 | ns | |
| $t_{CO}$ | Chip Enable To Output | | | 650 | ns | |
| $t_{OD}$ | Output Disable To Output | | | 550 | ns | |
| $t_{DF}$ [1] | Data Output to High Z State | 0 | | 200 | ns | |
| $t_{OH}$ | Previous Data Read Valid after change of Address | 0 | | | ns | |

WRITE CYCLE

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{WCY}$ | Write Cycle | 850 | | | ns | (See below) |
| $t_{AW}$ | Write Delay | 150 | | | ns | |
| $t_{CW}$ | Chip Enable To Write | 750 | | | ns | |
| $t_{DW}$ | Data Setup | 500 | | | ns | |
| $t_{DH}$ | Data Hold | 100 | | | ns | |
| $t_{WP}$ | Write Pulse | 630 | | | ns | |
| $t_{WR}$ | Write Recovery | 50 | | | ns | |

### A. C. CONDITIONS OF TEST

Input Pulse Levels: +0.65 Volt to 2.2 Volt

Input Pulse Rise and Fall Times: 20nsec

Timing Measurement Reference Level: 1.5 Volt

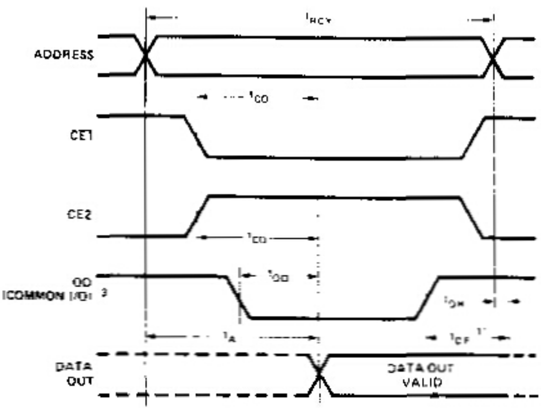
Output Load: 1 TTL Gate and $C_L$ = 100pF

## Capacitance $T_A = 25°C$, f = 1MHz

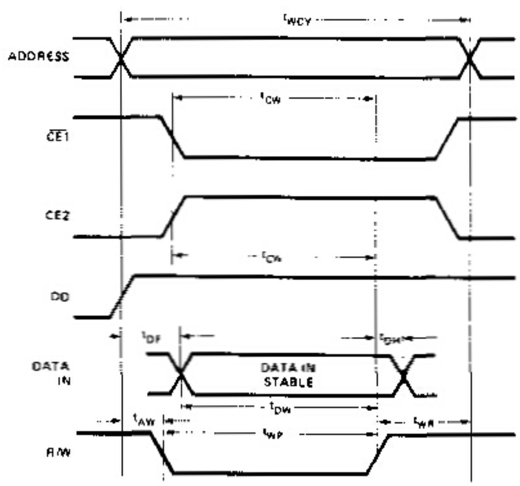
| Symbol | Test | Limits (pF) | |
|---|---|---|---|
| | | Typ. | Max. |
| $C_{IN}$ | Input Capacitance (All Input Pins) $V_{IN} = 0V$ | 4 | 8 |
| $C_{OUT}$ | Output Capacitance $V_{OUT} = 0V$ | 8 | 12 |

## Waveforms

READ CYCLE




WRITE CYCLE [2]




NOTES: 1. $t_{DF}$ is with respect to the trailing edge of $\overline{CE1}$, CE2, or OD, whichever occurs first.

2. During the write cycle, OD is a logical 1 for common I/O and "don't care" for separate I/O operation.

3. OD should be tied low for separate I/O operation.

# Silicon Gate MOS 8111-2

## 1024 BIT (256 x 4) STATIC MOS RAM WITH COMMON I/O AND OUTPUT DISABLE

- **Organization 256 Words by 4 Bits**
- **Access Time — 850 nsec Max.**
- **Common Data Input and Output**
- **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 — 18 Pin Plastic Dual-In-Line Configuration**
- **Low Power — Typically 150 mW**
- **Three-State Output — OR-Tie Capability**

The Intel® 8111-2 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.

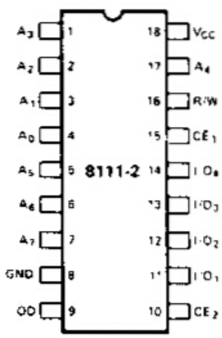
The 8111-2 is designed for memory applications in small systems where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

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

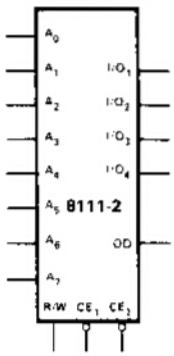
The Intel® 8111-2 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

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

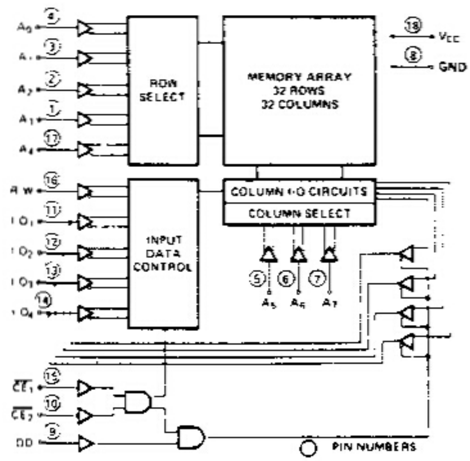
PIN CONFIGURATION




LOGIC SYMBOL




BLOCK DIAGRAM




PIN NAMES

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

## Absolute Maximum Ratings*

Ambient Temperature Under Bias . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . . . . . -65°C to +150°C
Voltage On Any Pin
With Respect to Ground . . . . . . . . . -0.5V to +7V
Power Dissipation . . . . . . . . . . . . . . . . . . . . 1 Watt

*COMMENT:

*Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition 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.*

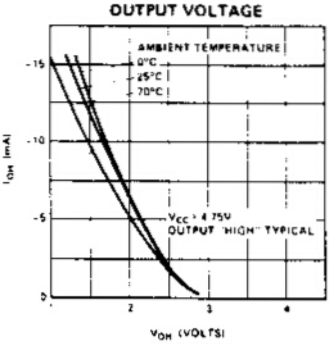
## D.C. and Operating Characteristics

$T_A$ = 0°C to 70°C, $V_{CC}$ = 5V ±5% , unless otherwise specified.

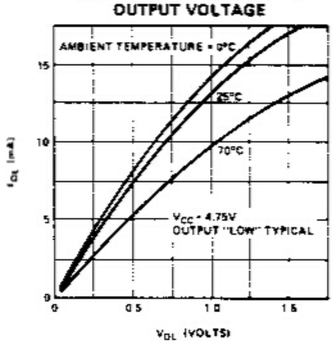
| Symbol | Parameter | Min. | Typ.[1] | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Load Current | | | 10 | μA | $V_{IN}$ = 0 to 5.25V |
| $I_{LOH}$ | I/O Leakage Current | | | 15 | μA | $\overline{CE}$ = 2.2V, $V_{I/O}$ = 4.0V |
| $I_{LOL}$ | I/O Leakage Current | | | -50 | μA | $\overline{CE}$ = 2.2V, $V_{I/O}$ = 0.45V |
| $I_{CC1}$ | Power Supply Current | | 30 | 60 | mA | $V_{IN}$ = 5.25V<br>$I_{I/O}$ = 0mA, $T_A$ = 25°C |
| $I_{CC2}$ | Power Supply Current | | | 70 | mA | $V_{IN}$ = 5.25V<br>$I_{I/O}$ = 0mA, $T_A$ = 0°C |
| $V_{IL}$ | Input Low Voltage | -0.5 | | +0.65 | V | |
| $V_{IH}$ | Input High Voltage | 2.2 | | $V_{CC}$ | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 2.0mA |
| $V_{OH}$ | Output High Voltage | 2.2 | | | V | $I_{OH}$ = -150 μA |

NOTES. 1. Typical values are for $T_A$ = 25°C and nominal supply voltage.

OUTPUT SOURCE CURRENT VS. OUTPUT VOLTAGE




OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



## A.C. Characteristics

**READ CYCLE** $T_A = 0°C$ to $70°C$, $V_{CC} = 5V \pm 5\%$, unless otherwise specified.

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{RCY}$ | Read Cycle | 850 | | | ns | (See below) |
| $t_A$ | Access Time | | | 850 | ns | |
| $t_{CO}$ | Chip Enable To Output | | | 650 | ns | |
| $t_{OD}$ | Output Disable To Output | | | 550 | ns | |
| $t_{DF}$ [1] | Data Output to High Z State | 0 | | 200 | ns | |
| $t_{OH}$ | Previous Data Read Valid after change of Address | 0 | | | ns | |

**WRITE CYCLE**

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{WCY}$ | Write Cycle | 850 | | | ns | (See below) |
| $t_{AW}$ | Write Delay | 150 | | | ns | |
| $t_{CW}$ | Chip Enable To Write | 750 | | | ns | |
| $t_{DW}$ | Data Setup | 500 | | | ns | |
| $t_{DH}$ | Data Hold | 100 | | | ns | |
| $t_{WP}$ | Write Pulse | 630 | | | ns | |
| $t_{WR}$ | Write Recovery | 50 | | | ns | |

### A. C. CONDITIONS OF TEST

Input Pulse Levels: +0.65 Volt to 2.2 Volt

Input Pulse Rise and Fall Times: 20nsec

Timing Measurement Reference Level: 1.5 Volt

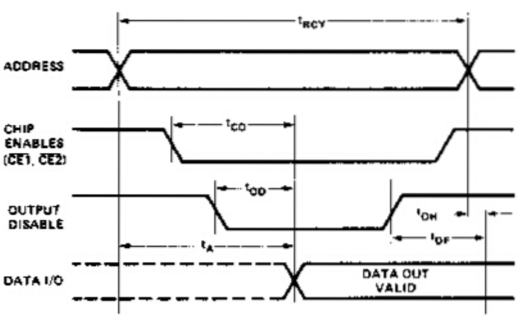
Output Load: 1 TTL Gate and $C_L = 100pF$

## Capacitance $T_A = 25°C$, $f = 1 MHz$

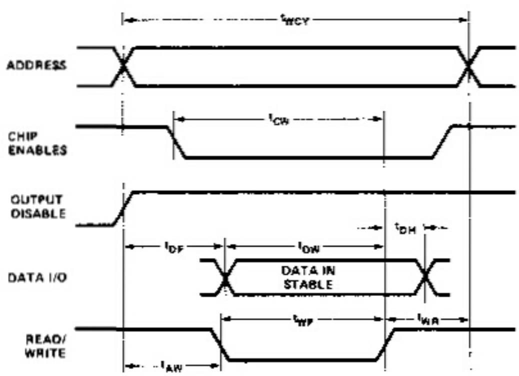
| Symbol | Test | Limits (pF) | |
|---|---|---|---|
| | | Typ. | Max. |
| $C_{IN}$ | Input Capacitance (All Input Pins) $V_{IN} = 0V$ | 4 | 8 |
| $C_{OUT}$ | Output Capacitance $V_{OUT} = 0V$ | 10 | 15 |

## Waveforms

**READ CYCLE**




**WRITE CYCLE**




NOTE: 1. $t_{DF}$ is with respect to the trailing edge of $\overline{CE1}$, $\overline{CE2}$, or OD, whichever occurs first.

# intel® Silicon Gate MOS 8101-2

## 1024 BIT (256 x 4) STATIC MOS RAM WITH SEPARATE I/O

- ***256 x 4 Organization to Meet Needs for Small System Memories***
- **Access Time — 850 nsec 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**
- ***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**
- **Output Disable Provided for Ease of Use in Common Data Bus Systems**

The Intel 8101-2® 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.

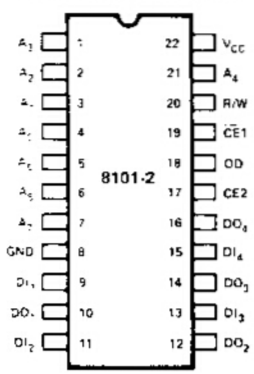
The 8101-2 is designed for memory applications where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, outputs, and a single +5V supply. Two chip-enables allow easy selection of an individual package when outputs are OR-tied. An output disable is provided so that data inputs and outputs can be tied for common I/O systems. Output disable is then used to eliminate any bidirectional logic.

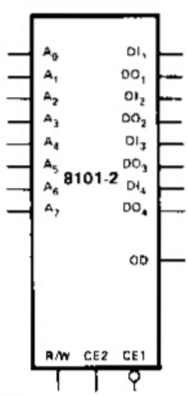
The Intel 8101-2 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

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

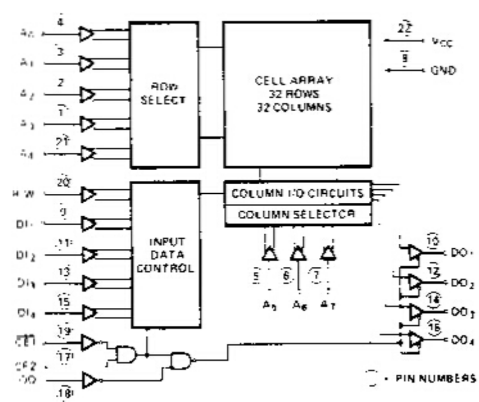
**PIN CONFIGURATION**




**LOGIC SYMBOL**




**BLOCK DIAGRAM**




**PIN NAMES**

| | | | |
|---|---|---|---|
| $D_{IN}$ | DATA INPUT | OD | OUTPUT DISABLE |
| $A_0 - A_7$ | ADDRESS INPUTS | $D_{OUT}$ | DATA OUTPUT |
| R/W | READ/WRITE INPUT | $V_{CC}$ | POWER (+5V) |
| $\overline{CE1}$, CE2 | CHIP ENABLE | | |

## Absolute Maximum Ratings*

Ambient Temperature Under Bias . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . . . . . -65°C to +150°C
Voltage On Any Pin
With Respect to Ground . . . . . . . . . . -0.5V to +7V
Power Dissipation . . . . . . . . . . . . . . . . . . . . . 1 Watt

**COMMENT:*

*Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition 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.*

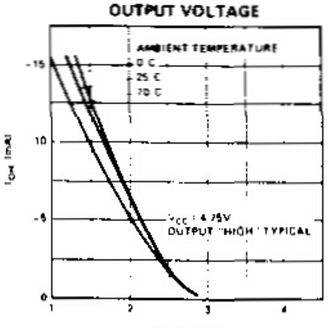
## D.C. and Operating Characteristics

$T_A$ = 0°C to 70°C, $V_{CC}$ = 5V ±5% unless otherwise specified.

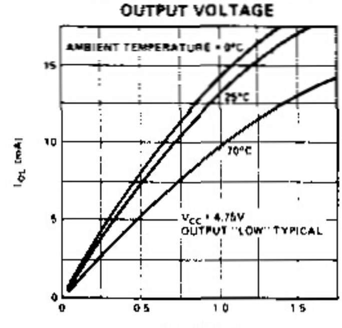
| Symbol | Parameter | Min. | Typ.[1] | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Current | | | 10 | μA | $V_{IN}$ = 0 to 5.25V |
| $I_{LOH}$ | I/O Leakage Current[2] | | | 15 | μA | $\overline{CE}$ = 2.2V, $V_{OUT}$ = 4.0V |
| $I_{LOL}$ | I/O Leakage Current[2] | | | -50 | μA | $\overline{CE}$ = 2.2V, $V_{OUT}$ = 0.45V |
| $I_{CC1}$ | Power Supply Current | | 30 | 60 | mA | $V_{IN}$ = 5.25V, $I_O$ = 0mA, $T_A$ = 25°C |
| $I_{CC2}$ | Power Supply Current | | | 70 | mA | $V_{IN}$ = 5.25V, $I_O$ = 0mA, $T_A$ = 0°C |
| $V_{IL}$ | Input "Low" Voltage | -0.5 | | +0.65 | V | |
| $V_{IH}$ | Input "High" Voltage | 2.2 | | $V_{CC}$ | V | |
| $V_{OL}$ | Output "Low" Voltage | | | +0.45 | V | $I_{OL}$ = 2.0mA |
| $V_{OH}$ | Output "High" Voltage | 2.2 | | | V | $I_{OH}$ = -150 μA |

NOTE: 1. Typical values are for $T_A$ = 25°C and nominal supply voltage.
2. Input and Output tied together.

OUTPUT SOURCE CURRENT VS. OUTPUT VOLTAGE




OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



## A.C. CHARACTERISTICS $T_A = 0°C$ to $70°C$, $V_{CC} = 5V \pm 5\%$ unless otherwise specified

| SYMBOL | PARAMETER | LIMITS | | | UNIT |
|---|---|---|---|---|---|
| | | MIN. | TYP.[1] | MAX. | |
| **READ CYCLE** | | | | | |
| $t_{RC}$ | READ CYCLE | 850 | | | ns |
| $t_A$ | ACCESS TIME | | 500 | 850 | ns |
| $t_{CO}$ | CHIP ENABLE TO OUTPUT TIME | | | 500 | ns |
| $t_{OH1}$ | PREVIOUS READ DATA VALID WITH RESPECT TO ADDRESS | 50 | | | ns |
| $t_{OH2}$ | PREVIOUS READ DATA VALID WITH RESPECT TO CHIP ENABLE | 0 | | | ns |
| **WRITE CYCLE** | | | | | |
| $t_{WC}$ | WRITE CYCLE | 850 | | | ns |
| $t_{AW}$ | ADDRESS TO WRITE SETUP TIME | 200 | | | ns |
| $t_{WP}$ | WRITE PULSE WIDTH | 600 | | | ns |
| $t_{WR}$ | WRITE RECOVERY TIME | 50 | | | ns |
| $t_{DW}$ | DATA SETUP TIME | 650 | | | ns |
| $t_{DH}$ | DATA HOLD TIME | 100 | | | ns |
| $t_{CW}$ | CHIP ENABLE TO WRITE SETUP TIME | 750 | | | ns |

(1) Typical values are for $T_A = 25°C$ and nominal supply voltage.

### A.C. CONDITIONS OF TEST

Input Pulse Levels: +0.65 Volt to 2.2 Volt

Input Pulse Rise and Fall Times: 20nsec

Timing Measurement Reference Level: 1.5 Volt

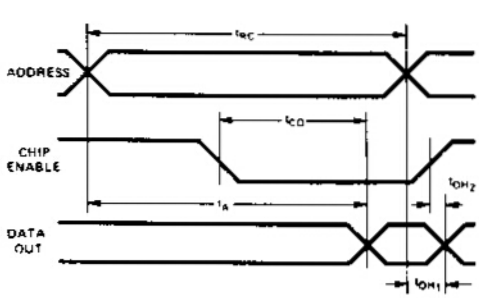
Output Load: 1 TTL Gate and $C_L$ = 100 pF

### CAPACITANCE $T_A = 25°C$, f = 1 MHz

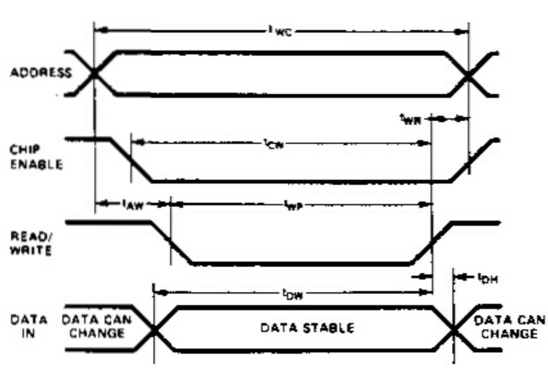
| SYMBOL | TEST | LIMITS (pF) | |
|---|---|---|---|
| | | TYP. | MAX. |
| $C_{IN}$ | INPUT CAPACITANCE (ALL INPUT PINS) $V_{IN}$ = 0V | 3 | 5 |
| $C_{OUT}$ | OUTPUT CAPACITANCE $V_{OUT}$ = 0V | 7 | 10 |

## WAVEFORMS

### READ CYCLE

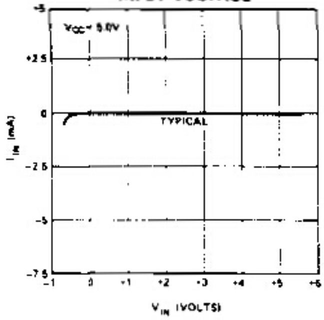



### WRITE CYCLE

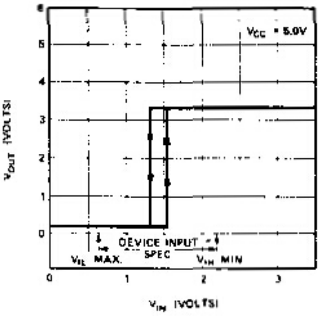


## TYPICAL D.C. CHARACTERISTICS

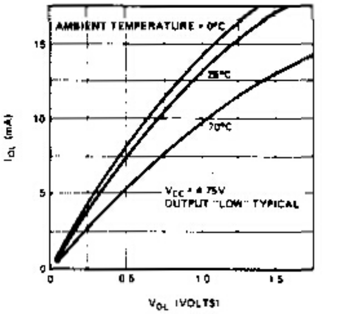
INPUT CURRENT VS. INPUT VOLTAGE



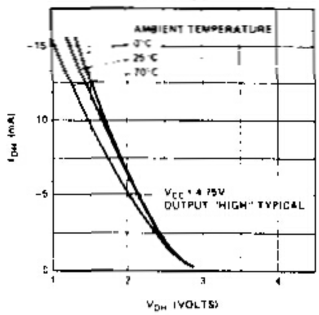

EFFECTIVE INPUT CHARACTERISTIC



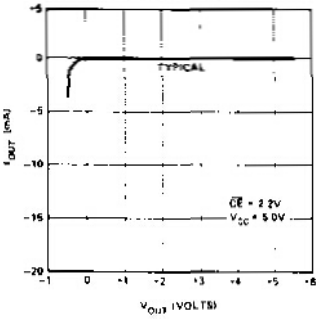

OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



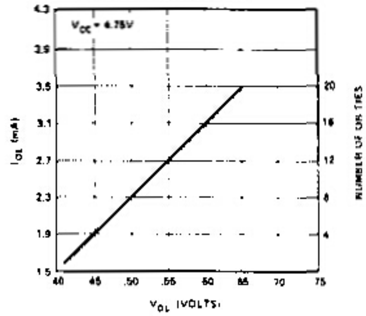

OUTPUT SOURCE CURRENT VS. OUTPUT VOLTAGE




OUTPUT CURRENT VS. OUTPUT VOLTAGE WITH CHIP DISABLED

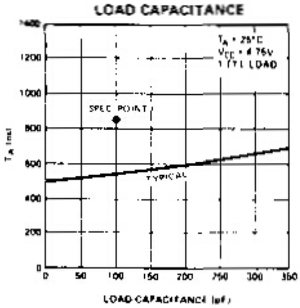



RELATIONSHIP BETWEEN OUTPUT SINK CURRENT, NUMBER OF OR-TIES, AND OUTPUT VOLTAGE

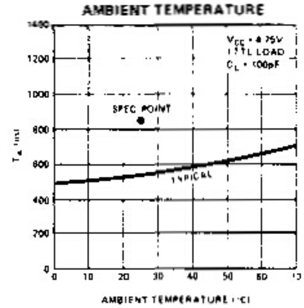



## TYPICAL A.C. CHARACTERISTICS

ACCESS TIME VS. LOAD CAPACITANCE




ACCESS TIME VS. AMBIENT TEMPERATURE



# Silicon Gate MOS 8102A-4

## 1024 BIT FULLY DECODED STATIC MOS RANDOM ACCESS MEMORY

- *Access Time — 450 ns Max.*
- **Single +5 Volts Supply Voltage**
- **Directly TTL Compatible — All Inputs and Output**
- **Static MOS — No Clocks or Refreshing Required**
- **Low Power — Typically 150 mW**
- **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® 8102A-4 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.

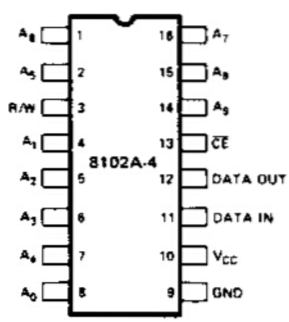
The 8102A-4 is designed for microcomputer memory applications where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

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.

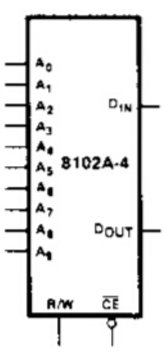
The Intel® 8102A-4 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

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

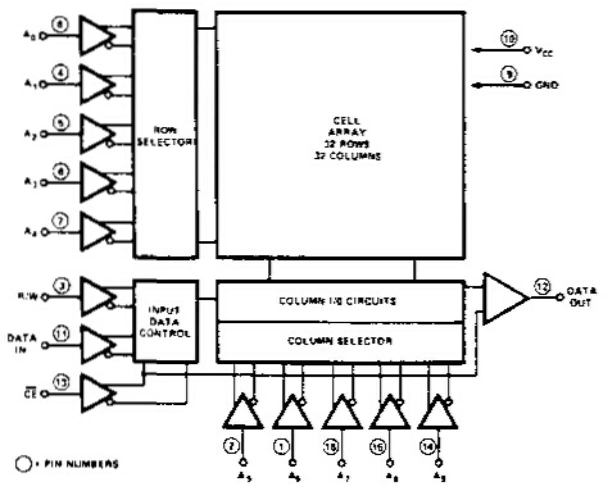
PIN CONFIGURATION




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) |

## ABSOLUTE MAXIMUM RATINGS*

| | |
|---|---|
| Ambient Temperature Under Bias | 0°C to 70°C |
| Storage Temperature | −65°C to +150°C |
| Voltage On Any Pin With Respect To Ground | −0.5V to +7V |
| Power Dissipation | 1 Watt |

*COMMENT:

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition 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. AND OPERATING CHARACTERISTICS

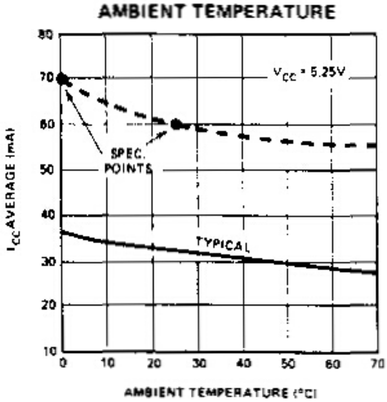
$T_A$ = 0°C to +70°C, $V_{CC}$ = 5V ±5% unless otherwise specified

| SYMBOL | PARAMETER | LIMITS | | | UNIT | TEST CONDITIONS |
|---|---|---|---|---|---|---|
| | | MIN. | TYP.[1] | MAX. | | |
| $I_{LI}$ | INPUT LOAD CURRENT (ALL INPUT PINS) | | | 10 | µA | $V_{IN}$ = 0 to 5.25V |
| $I_{LOH}$ | OUTPUT LEAKAGE CURRENT | | | 5 | µA | $\overline{CE}$ = 2.0V, $V_{OUT}$ = 2.4 to $V_{CC}$ |
| $I_{LOL}$ | OUTPUT LEAKAGE CURRENT | | | -10 | µA | $\overline{CE}$ = 2.0V, $V_{OUT}$ = 0.4V |
| $I_{CC1}$ | POWER SUPPLY CURRENT | | 30 | 60 | mA | ALL INPUTS = 5.25V DATA OUT OPEN $T_A$ = 25°C |
| $I_{CC2}$ | POWER SUPPLY CURRENT | | | 70 | mA | ALL INPUTS = 5.25V DATA OUT OPEN $T_A$ = 0°C |
| $V_{IL}$ | INPUT "LOW" VOLTAGE | −0.5 | | 0.8 | V | |
| $V_{IH}$ | INPUT "HIGH" VOLTAGE | 2.0 | | $V_{CC}$ | V | |
| $V_{OL}$ | OUTPUT "LOW" VOLTAGE | | | 0.4 | V | $I_{OL}$ = 2.1mA |
| $V_{OH}$ | OUTPUT "HIGH" VOLTAGE | 2.4 | | | V | $I_{OH}$ = −100µA |

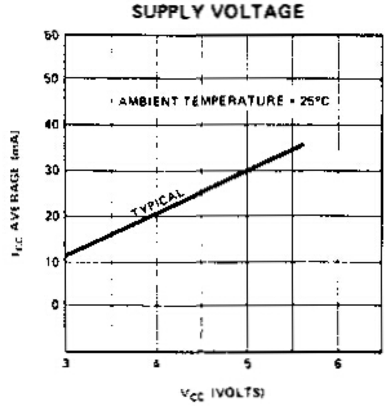
(1) Typical values are for $T_A$ = 25°C and nominal supply voltage.

## TYPICAL D.C. CHARACTERISTICS

POWER SUPPLY CURRENT VS. AMBIENT TEMPERATURE




POWER SUPPLY CURRENT VS. SUPPLY VOLTAGE



## A. C. Characteristics $T_A$ = 0°C to 70°C, $V_{CC}$ = 5V ±5% unless otherwise specified

| Symbol | Parameter | Limits Min. | Limits Typ.[1] | Limits Max. | Unit |
|---|---|---|---|---|---|
| **READ CYCLE** | | | | | |
| $t_{RC}$ | Read Cycle | 450 | | | ns |
| $t_A$ | Access Time | | | 450 | ns |
| $t_{CO}$ | Chip Enable to Output Time | | | 230 | ns |
| $t_{OH1}$ | Previous Read Data Valid with Respect to Address | 40 | | | ns |
| $t_{OH2}$ | Previous Read Data Valid with Respect to Chip Enable | 0 | | | ns |
| **WRITE CYCLE** | | | | | |
| $t_{WC}$ | Write Cycle | 450 | | | ns |
| $t_{AW}$ | Address to Write Setup Time | 20 | | | ns |
| $t_{WP}$ | Write Pulse Width | 300 | | | ns |
| $t_{WR}$ | Write Recovery Time | 0 | | | ns |
| $t_{DW}$ | Data Setup Time | 300 | | | ns |
| $t_{DH}$ | Data Hold Time | 0 | | | ns |
| $t_{CW}$ | Chip Enable to Write Setup Time | 300 | | | ns |

NOTE: 1. Typical values are for $T_A$ = 25°C and nominal supply voltage.

**A. C. CONDITIONS OF TEST**

| | | |
|---|---|---|
| Input Pulse Levels | | 0.8 Volt to 2.0 Volt |
| Input Rise and Fall Times: | | 10nsec |
| Timing Measurement Reference Levels | Inputs: | 1.5 Volts |
| | Output: | 0.8 and 2.0 Volts |
| Output Load | | 1 TTL Gate and $C_L$ = 100 pF |

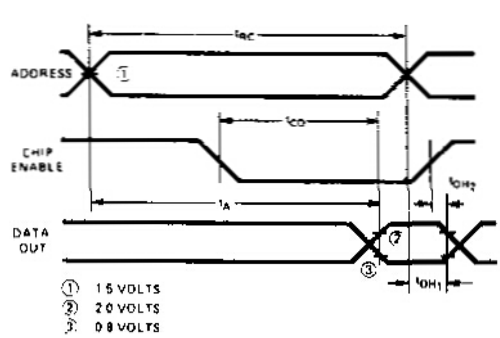
## Capacitance[2] $T_A$ = 25°C, f = 1 MHz

| SYMBOL | TEST | LIMITS (pF) TYP.[1] | LIMITS (pF) MAX. |
|---|---|---|---|
| $C_{IN}$ | INPUT CAPACITANCE (ALL INPUT PINS) $V_{IN}$ = 0V | 3 | 5 |
| $C_{OUT}$ | OUTPUT CAPACITANCE $V_{OUT}$ = 0V | 7 | 10 |

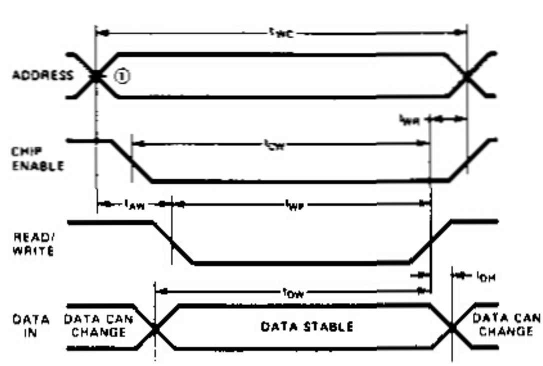
NOTE: 2. This parameter is periodically sampled and is not 100% tested.

## Waveforms

**READ CYCLE**

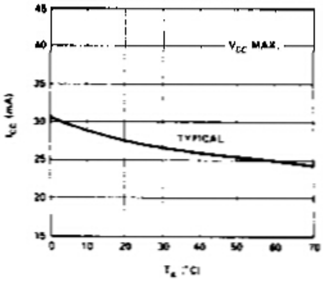



**WRITE CYCLE**

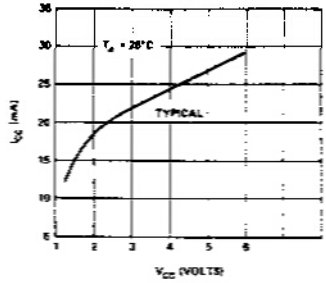


## Typical D. C. and A. C. Characteristics

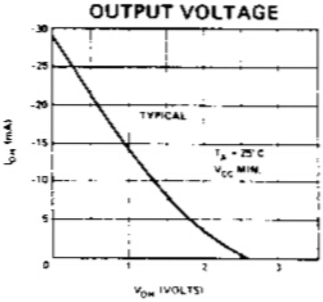
POWER SUPPLY CURRENT VS. AMBIENT TEMPERATURE



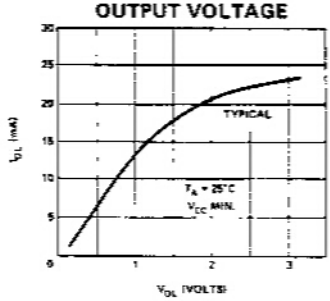

POWER SUPPLY CURRENT VS. SUPPLY VOLTAGE



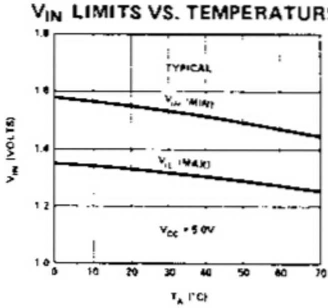

OUTPUT SOURCE CURRENT VS. OUTPUT VOLTAGE



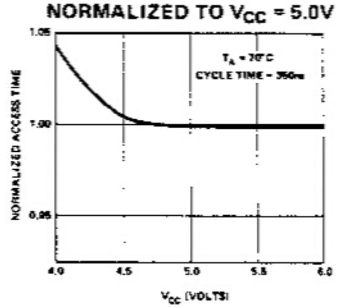

OUTPUT SINK CURRENT VS. OUTPUT VOLTAGE



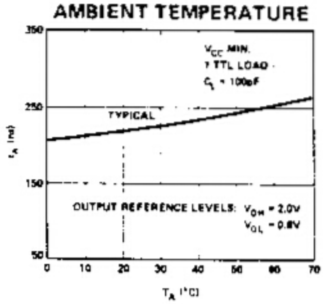

$V_{IN}$ LIMITS VS. TEMPERATURE



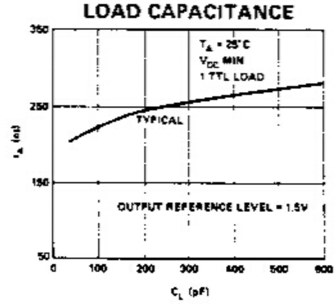

ACCESS TIME VS. $V_{CC}$ NORMALIZED TO $V_{CC}$ = 5.0V




ACCESS TIME VS. AMBIENT TEMPERATURE




ACCESS TIME VS. LOAD CAPACITANCE



intel®

# Silicon Gate MOS **8107B-4**

# FULLY DECODED RANDOM ACCESS 4096 BIT DYNAMIC MEMORY

## * Access Time -- 270 ns max.
## * Read, Write Cycle Times – 470 ns max.
## * Refresh Period -- 2 ms

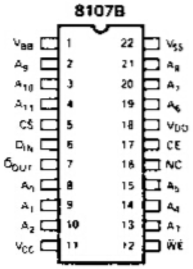
- **Low Cost Per Bit**
- **Low Standby Power**
- **Easy System Interface**
- **Only One High Voltage Input Signal – Chip Enable**
- **TTL Compatible -- All Address, Data, Write Enable, Chip Select Inputs**
- **Read-Modify-Write Cycle Time -- 590 ns**
- **Address Registers Incorporated on the Chip**
- **Simple Memory Expansion – Chip Select Input Lead**
- **Fully Decoded – On Chip Address Decode**
- **Output is Three State and TTL Compatible**
- **Industry Standard 22-Pin Configuration**

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

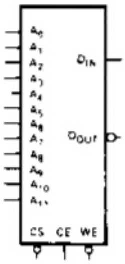
Reading information from the memory is non-destructive. Refreshing is most easily accomplished by performing one read cycle on each of the 64 row addresses. Each row address must be refreshed every two milliseconds. The memory is refreshed whether Chip Select is a logic one or a logic zero.

The 8107B 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 other MOS technologies. The 8107B uses a single transistor cell to achieve high speed and low cost. It is a replacement for the 8107B.

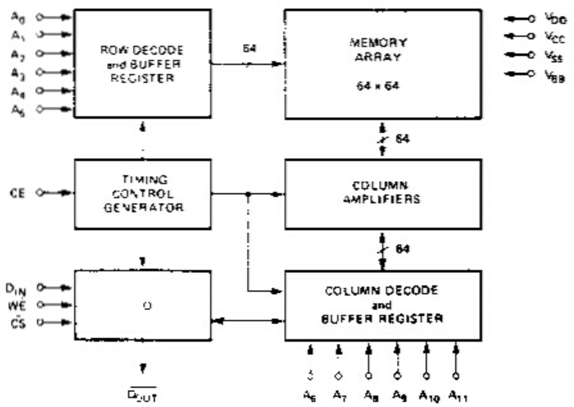
**PIN CONFIGURATION**




**LOGIC SYMBOL**




**BLOCK DIAGRAM**




**PIN NAMES**

| | | | |
|---|---|---|---|
| $A_0$-$A_{11}$ | ADDRESS INPUTS* | $V_{BB}$ | POWER (-5V) |
| CE | CHIP ENABLE | $V_{CC}$ | POWER (+5V) |
| $\overline{CS}$ | CHIP SELECT | $V_{DD}$ | POWER (+12V) |
| $D_{IN}$ | DATA INPUT | $V_{SS}$ | GROUND |
| $\overline{D_{OUT}}$ | DATA OUTPUT | $\overline{WE}$ | WRITE ENABLE |
| NC | NOT CONNECTED | | |

*Refresh Address $A_0$-$A_5$.

## 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 $V_{DD}$, $V_{CC}$, and $V_{SS}$ with Respect to $V_{BB}$ | +20V to −0.3V |
| Power Dissipation | 1.25W |

**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.*

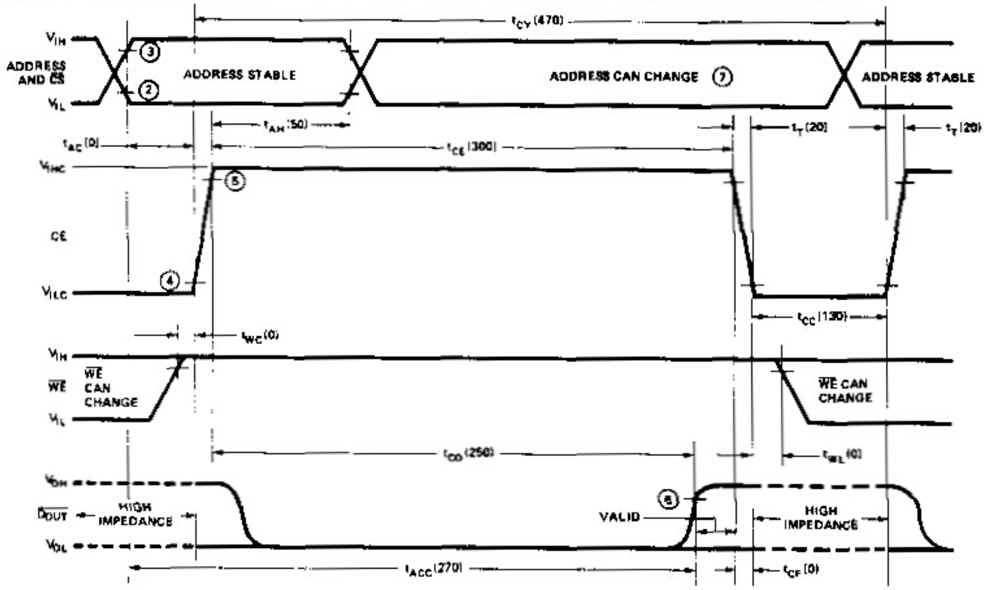
## D.C. and Operating Characteristics

$T_A$ = 0°C to 70°C, $V_{DD}$ = +12V ± 5%, $V_{CC}$ = +5V ± 5%, $V_{BB}$[1] = -5V ± 5%, $V_{SS}$ = 0V, unless otherwise noted.

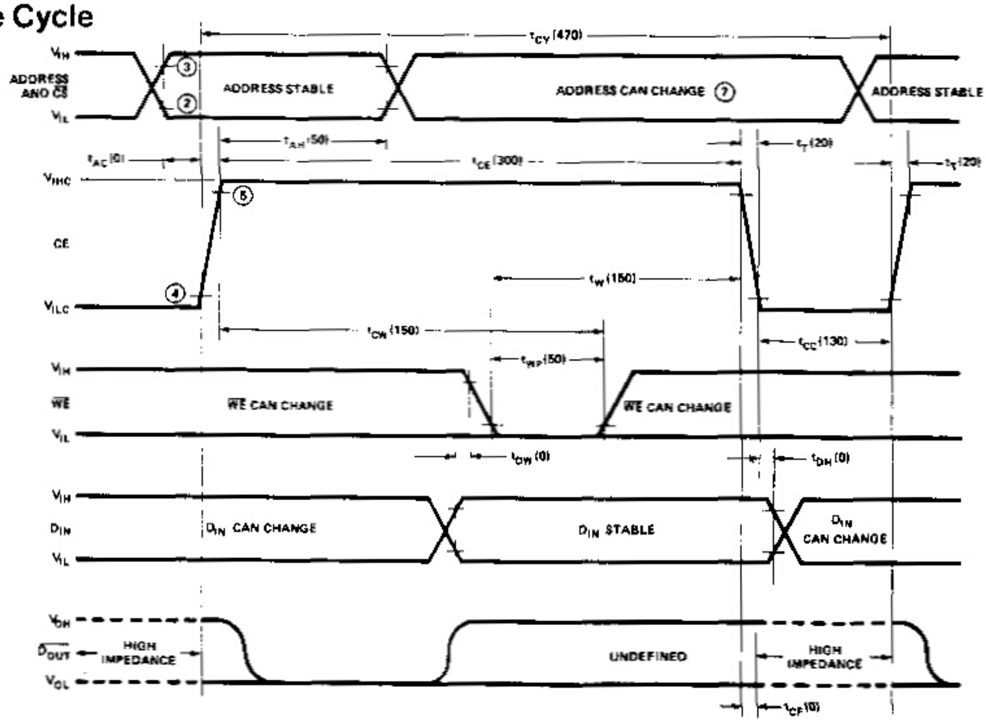
| Symbol | Parameter | Limits Min. | Typ.[2] | Max. | Unit | Conditions |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Load Current (all inputs except CE) | | .01 | 10 | µA | $V_{IN} = V_{IL\ MIN}$ to $V_{IH\ MAX}$ |
| $I_{LC}$ | Input Load Current | | .01 | 10 | µA | $V_{IN} = V_{IL\ MIN}$ to $V_{IH\ MAX}$ |
| $\|I_{LO}\|$ | Output Leakage Current for high impedance state | | .01 | 10 | µA | CE = $V_{ILC}$ or $\overline{CS} = V_{IH}$; $V_O$ = 0V to 5.25V |
| $I_{DD1}$ | $V_{DD}$ Supply Current during CE off[3] | | 110 | 200 | µA | CE = −1V to +.6V |
| $I_{DD2}$ | $V_{DD}$ Supply Current during CE on | | 80 | 100 | mA | CE = $V_{IHC}$, $T_A$ = 25°C |
| $I_{DD\ AV1}$ | Average $V_{DD}$ Current | | 55 | 80 | mA | Cycle time=470ns, $t_{CE}$ = 300ns; $T_A$ = 25°C |
| $I_{DD\ AV2}$ | Average $V_{DD}$ Current | | 27 | 40 | mA | Cycle time=1000ns, $t_{CE}$ = 300ns; $T_A$ = 25°C |
| $I_{CC1}$[4] | $V_{CC}$ Supply Current during CE off | | .01 | 10 | µA | CE = $V_{ILC}$ or $\overline{CS} = V_{IH}$ |
| $I_{BB}$ | $V_{BB}$ Supply Current | | 5 | 100 | µA | |
| $V_{IL}$ | Input Low Voltage | -1.0 | | 0.6 | V | $t_T$ = 20ns — See Figure 4 |
| $V_{IH}$ | Input High Voltage | 2.4 | | $V_{CC}$+1 | V | |
| $V_{ILC}$ | CE Input Low Voltage | -1.0 | | +1.0 | V | |
| $V_{IHC}$ | CE Input High Voltage | $V_{DD}$-1 | | $V_{DD}$+1 | V | |
| $V_{OL}$ | Output Low Voltage | 0.0 | | 0.45 | V | $I_{OL}$ = 2.0mA |
| $V_{OH}$ | Output High Voltage | 2.4 | | $V_{CC}$ | V | $I_{OH}$ = -2.0mA |

NOTES:
1. The only requirement for the sequence of applying voltage to the device is that $V_{DD}$, $V_{CC}$, and $V_{SS}$ should never be .3V more negative than $V_{BB}$.
2. Typical values are for $T_A$ = 25°C and nominal power supply voltages.
3. The $I_{DD}$ and $I_{CC}$ currents flow to $V_{SS}$. The $I_{BB}$ current is the sum of all leakage currents.
4. During CE on $V_{CC}$ supply current is dependent on output loading, $V_{CC}$ is connected to output buffer only.

## Read and Refresh Cycle [1] (Numbers in parentheses are for minimum cycle timing in ns)




## Write Cycle




NOTES:
1. For Refresh cycle row and column addresses must be stable before $t_{AC}$ and remain stable for entire $t_{AH}$ period.
2. $V_{IL}$ MAX is the reference level for measuring timing of the addresses, $\overline{CS}$, $\overline{WE}$, and $D_{IN}$.
3. $V_{IH}$ MIN is the reference level for measuring timing of the addresses, $\overline{CS}$, $\overline{WE}$, and $D_{IN}$.
4. $V_{SS}$ +2.0V is the reference level for measuring timing of CE.
5. $V_{DD}$ -2V is the reference level for measuring timing of CE.
6. $V_{SS}$ +2.0V is the reference level for measuring the timing of $\overline{D_{OUT}}$.
7. During CE high typically 0.5mA will be drawn from any address pin which is switched from low to high.

## A.C. Characteristics $T_A = 0°C$ to $70°C$, $V_{DD} = 12V \pm 5\%$, $V_{CC} = 5V \pm 10\%$, $V_{BB} = -5V \pm 5\%$,

READ, WRITE, AND READ MODIFY/WRITE CYCLE $V_{SS} = 0V$, unless otherwise noted.

| Symbol | Parameter | Min. | Max. | Unit | Conditions |
|---|---|---|---|---|---|
| $t_{REF}$ | Time Between Refresh | | 2 | ms | |
| $t_{AC}$ | Address to CE Set Up Time | 0 | | ns | $t_{AC}$ is measured from end of address transition |
| $t_{AH}$ | Address Hold Time | 100 | | ns | |
| $t_{CC}$ | CE Off Time | 130 | | ns | |
| $t_T$ | CE Transition Time | 10 | 40 | ns | |
| $t_{CF}$ | CE Off to Output High Impedance State | 0 | | ns | |

### READ CYCLE

| Symbol | Parameter | Min. | Max. | Unit | Conditions |
|---|---|---|---|---|---|
| $t_{CY}$ | Cycle Time | 470 | | ns | $t_T = 20ns$ |
| $t_{CE}$ | CE On Time | 300 | 4000 | ns | |
| $t_{CO}$ | CE Output Delay | | 250 | ns | $C_{load} = 50pF$, Load = One TTL Gate, |
| $t_{ACC}$ | Address to Output Access | | 270 | ns | Ref = 2.0V. |
| $t_{WL}$ | CE to $\overline{WE}$ | 0 | | ns | $t_{ACC} = t_{AC} + t_{CO} + 1t_T$ |
| $t_{WC}$ | $\overline{WE}$ to CE on | 0 | | ns | |

### WRITE CYCLE

| Symbol | Parameter | Min. | Max. | Unit | Conditions |
|---|---|---|---|---|---|
| $t_{CY}$ | Cycle Time | 470 | | ns | $t_T = 20ns$ |
| $t_{CE}$ | CE On Time | 300 | 4000 | ns | |
| $t_W$ | $\overline{WE}$ to CE Off | 150 | | ns | |
| $t_{CW}$ | CE to $\overline{WE}$ | 150 | | ns | |
| $t_{DW}$ [2] | $D_{IN}$ to $\overline{WE}$ Set Up | 0 | | ns | |
| $t_{DH}$ | $D_{IN}$ Hold Time | 0 | | ns | |
| $t_{WP}$ | $\overline{WE}$ Pulse Width | 50 | | ns | |

## Read Modify Write Cycle

| Symbol | Parameter | Min. | Max. | Unit | Conditions |
|---|---|---|---|---|---|
| $t_{RWC}$ | Read Modify Write (RMW) Cycle Time | 590 | | ns | $t_T = 20ns$ |
| $t_{CRW}$ | CE Width During RMW | 420 | 4000 | ns | |
| $t_{WC}$ | $\overline{WE}$ to CE on | 0 | | ns | |
| $t_W$ | $\overline{WE}$ to CE off | 150 | | ns | $C_{load} = 50pF$, Load = One TTL Gate, |
| $t_{WP}$ | $\overline{WE}$ Pulse Width | 50 | | ns | Ref = 2.0V |
| $t_{DW}$ | $D_{IN}$ to $\overline{WE}$ Set Up | 0 | | ns | |
| $t_{DH}$ | $D_{IN}$ Hold Time | 0 | | ns | |
| $t_{CO}$ | CE to Output Delay | | 250 | ns | |
| $t_{ACC}$ | Access Time | | 270 | ns | $t_{ACC} = t_{AC} + t_{CO} + 1t_T$ |

## Typical Characteristics

Fig. 1. $I_{DD}$ AV VS. TEMPERATURE

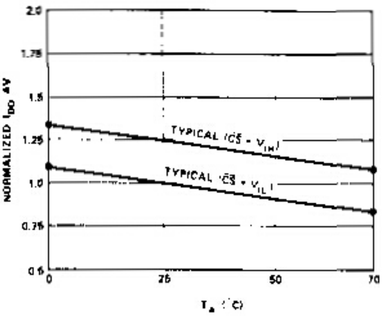



Fig. 2. TYPICAL $I_{DD}$ AVERAGE VS. CYCLE TIME

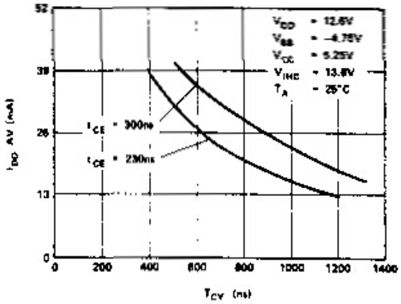



Fig. 3. $I_{DD2}$ VS. TEMPERATURE

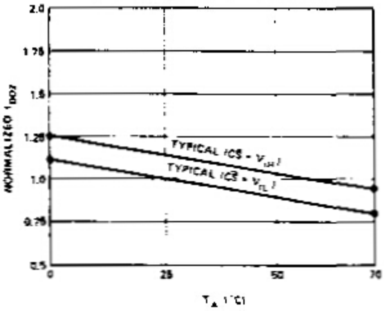



Fig. 4. TYPICAL $V_{IL}$ MAX VS. CE RISE TIME

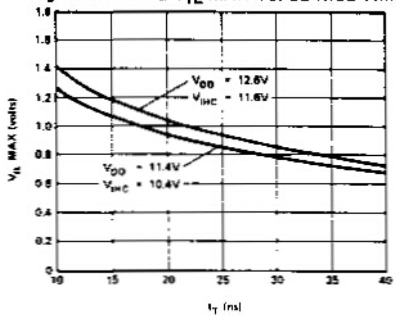



Fig. 5. TYPICAL $I_{OH}$ VS. $V_{OH}$

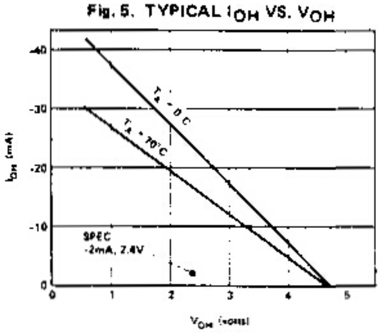



Fig. 6. TYPICAL $I_{OL}$ VS. $V_{OL}$

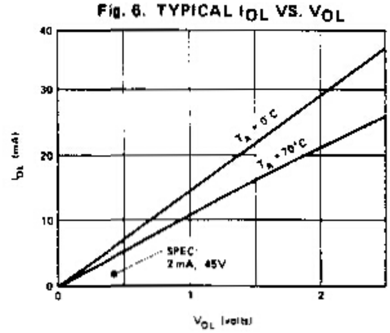



Fig. 7. TYPICAL REFRESH VS. TEMPERATURE

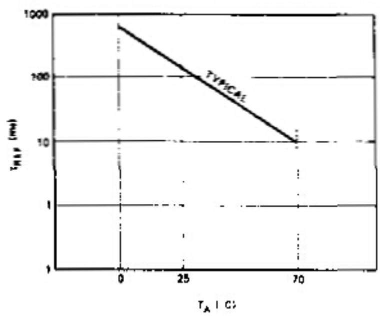
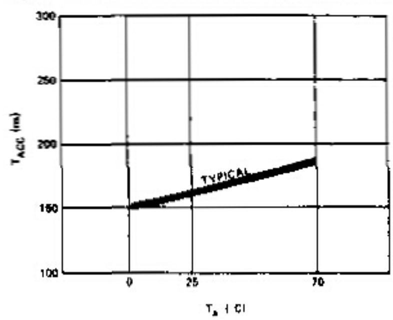


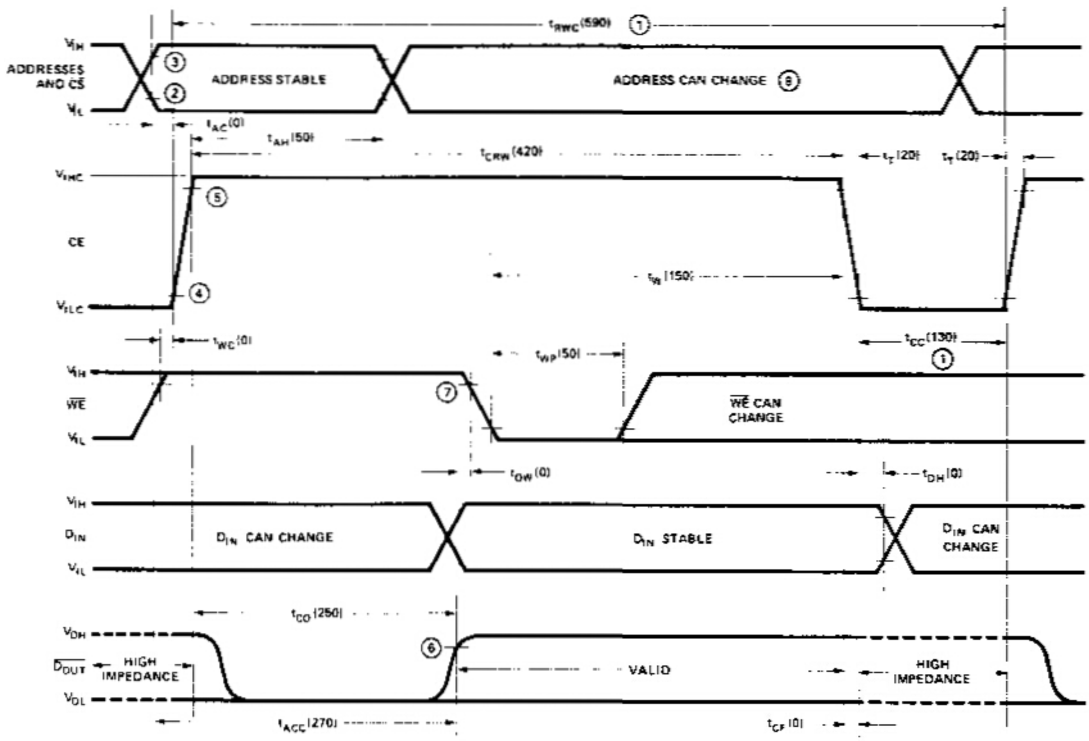

Fig. 8. TYPICAL ACCESS TIME VS. TEMPERATURE



## Read Modify Write Cycle[1]

| Symbol | Parameter | Min. | Max. | Unit | Conditions |
|---|---|---|---|---|---|
| $t_{RWC}$ | Read Modify Write (RMW) Cycle Time | 590 | | ns | $t_T$ = 20ns |
| $t_{CRW}$ | CE Width During RMW | 420 | 3000 | ns | |
| $t_{WC}$ | $\overline{WE}$ to CE on | 0 | | ns | |
| $t_W$ | $\overline{WE}$ to CE off | 150 | | ns | $C_{load}$ = 50pF, Load = One TTL Gate, Ref = 2.0V |
| $t_{WP}$ | $\overline{WE}$ Pulse Width | 50 | | ns | |
| $t_{DW}$ | $D_{IN}$ to $\overline{WE}$ Set Up | 0 | | ns | |
| $t_{DH}$ | $D_{IN}$ Hold Time | 0 | | ns | |
| $t_{CO}$ | CE to Output Delay | | 250 | ns | |
| $t_{ACC}$ | Access Time | | 270 | ns | $t_{ACC} = t_{AC} + t_{CO} + 1t_T$ |

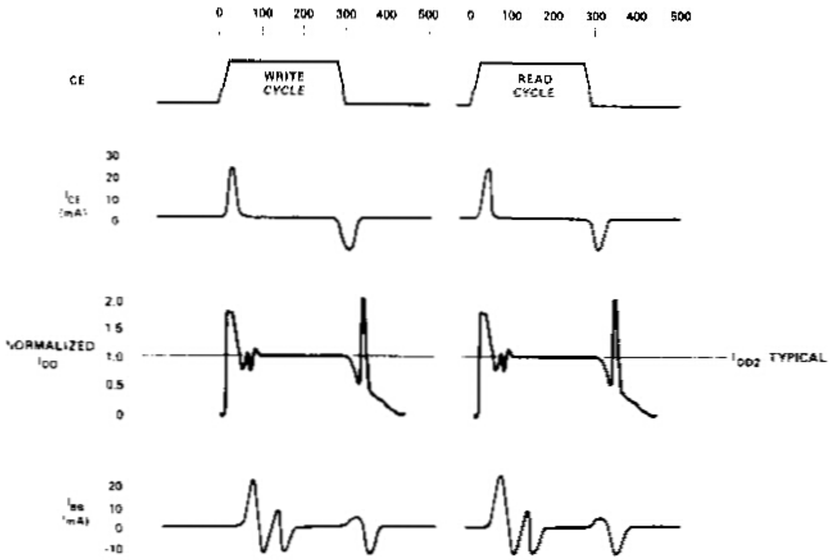
(Numbers in parentheses are for minimum cycle timing in ns.)




NOTES:

1. A.C. characteristics are guaranteed only if cumulative CE on time during $t_{REF}$ is ≤65% of $t_{REF}$. For continuous Read-Modify-Write operation, $t_{CC}$ and $t_{RWC}$ should be increased to at least 185ns and 645ns, respectively.
2. $V_{IL}$ MAX is the reference level for measuring timing of the addresses, $\overline{CS}$, $\overline{WE}$, and $D_{IN}$.
3. $V_{IH}$ MIN is the reference level for measuring timing of the addresses, $\overline{CS}$, $\overline{WE}$, and $D_{IN}$.
4. $V_{SS}$ +2.0V is the reference level for measuring timing of CE.
5. $V_{DD}$ −2V is the reference level for measuring timing of CE.
6. $V_{SS}$ +2.0V is the reference level for measuring the timing of $\overline{D_{OUT}}$.
7. $\overline{WE}$ must be at $V_{IH}$ until end of $t_{CO}$.
8. During CE high typically 0.5mA will be drawn from any address pin which is switched from low to high.

## Typical Current Transients vs. Time




## Applications

### Refresh

The 8107B-4 is refreshed by either a read cycle, write cycle, or read-modify write cycle. Only the selected row of memory array is refreshed. The row address is selected by the input signals $A_0$ thru $A_5$. Each individual row address must receive at least one refresh cycle within any two milliseconds time period.

If a read cycle is used for refreshing, then the chip select input, $\overline{CS}$, can be a logic high or a logic low. If a write cycle or read-modify write cycle is used to refresh the device, then $\overline{CS}$ must be a logic high. This will prevent writing into the memory during refresh.

### Power Dissipation

The operating power dissipation of a selected device is the sum of $V_{DD}$ x $I_{DDAV}$ and $V_{BB}$ x $I_{BB}$. For a cycle of 400ns and $t_{CE}$ of 230ns typical power dissipation is 456mW.

### Standby Power

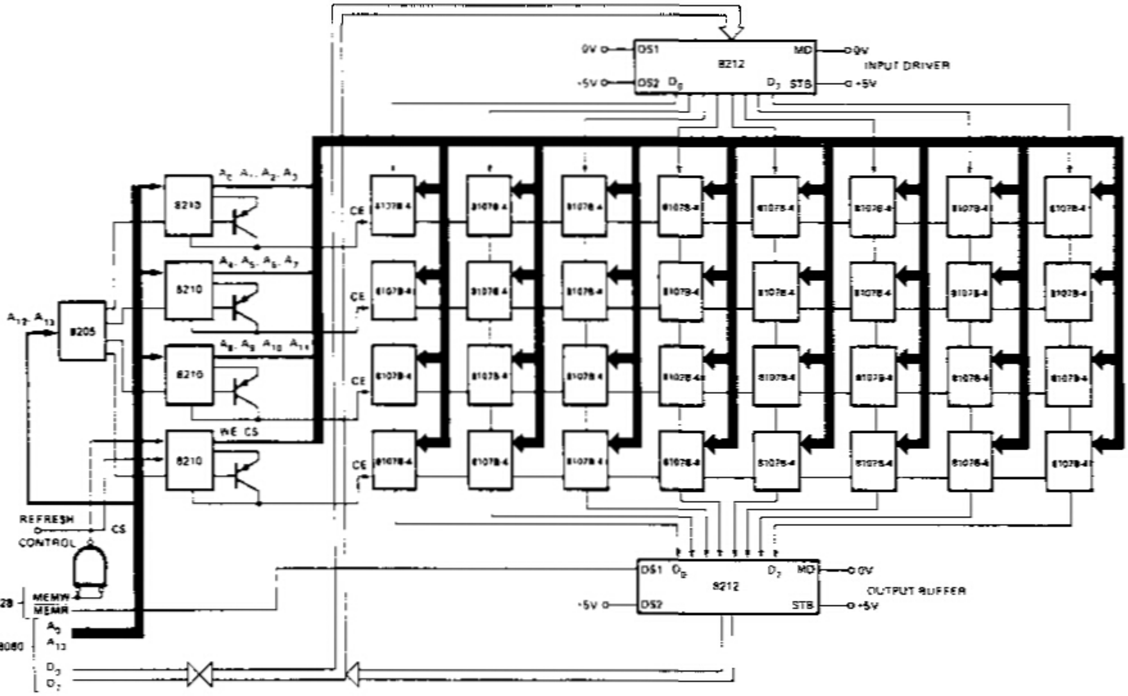
The 8107B-4 is a dynamic RAM therefore when $V_{CE} = V_{ILC}$ very little power is dissipated. In a typical system most devices are in standby with $V_{CE}$ at $V_{ILC}$. During this time only leakage currents flow (i.e., $I_{DD1}$, $I_{CC1}$, $I_{BB}$, $I_{LO}$, $I_{L1}$). The power dissipated during this inactive period is typically 1.4mW. The typical power dissipation required to perform refresh during standby is the refresh duty cycle, 1.3%, multiplied by the operating power dissipation, or 5.9mW. The total power dissipation during standby is then 7.3mW typical.

### System Interfaces and Filtering

On the following page is an example of a 16K x 8 bit memory system. Device decoding is done with the CE input. All devices are unselected during refresh with CS. It is recommended that 1$\mu$F high frequency, low inductance capacitors be used on double sided boards. $V_{CC}$ to $V_{SS}$ decoupling is required only on the devices located around the periphery of the array. For each 36 devices a 100$\mu$F tantalum or equivalent capacitor should be placed from $V_{DD}$ to $V_{SS}$ close to the array.

### Typical System

Below is an example of a 16K × 8 bit memory circuit. Device decoding is done with the CE input. All devices are unselected during refresh with CS input. The 8210, 8205 and 8212 are standard Intel products.



# intel® Silicon Gate CMOS 5101, 5101-3, 5101L, 5101L-3

# 1024 BIT (256 x 4) STATIC CMOS RAM

## *Ultra Low Standby Current: 15 nA/Bit for the 5101

- **Fast Access Time — 650 ns**
- **Single +5 V Power Supply**
- **$CE_2$ Controls Unconditional Standby Mode**
- **Directly TTL Compatible — All Inputs and Outputs**
- **Three-State Output**

The Intel® 5101 and 5101-3 are ultra-low power 1024 bit (256 words x 4-bits) static RAMs fabricated with an advanced ion-implanted silicon gate CMOS technology. The devices have two chip enable inputs. When $CE_2$ is at a low level, the minimum standby current is drawn by these devices, regardless of any other input transitions on the addresses and other control inputs. Also, when $\overline{CE_1}$ is at a high level and address and other control transitions are inhibited, the minimum standby current is drawn by these devices. When in standby the 5101 and 5101-3 draw from the single 5 volt supply only 15 microamps and 200 microamps, respectively. These devices are ideally suited for low power applications where battery operation or battery backup for non-volatility are required.

The 5101 and 5101-3 use fully DC stable (static) circuitry; it is not necessary to pulse chip select for each address transition. The data is read out non-destructively and has the same polarity as the input data. All inputs and outputs are directly TTL compatible. The 5101 and 5101-3 have separate data input and data output terminals. An output disable function is provided so that the data inputs and outputs may be wire OR-ed for use in common data I/O systems.

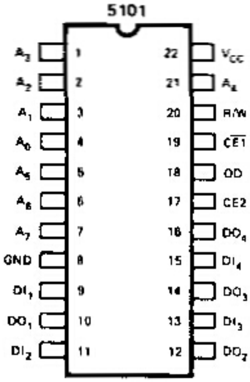
*The 5101L and 5101L-3 are identical to the 5101 and 5101-3, respectively, with the additional feature of guaranteed data retention at a power supply voltage as low as 2.0 volts.*

A pin compatible N-channel static RAM, the Intel 2101, is also available for low cost applications where a 256 x 4 organization is needed.

The Intel ion-implanted, silicon gate, complementary MOS (CMOS) allows the design and production of ultra-low power, high performance memories.

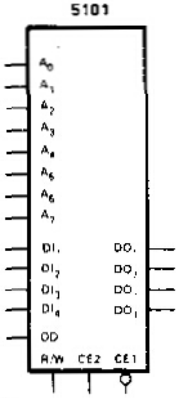
**PIN CONFIGURATION**

5101

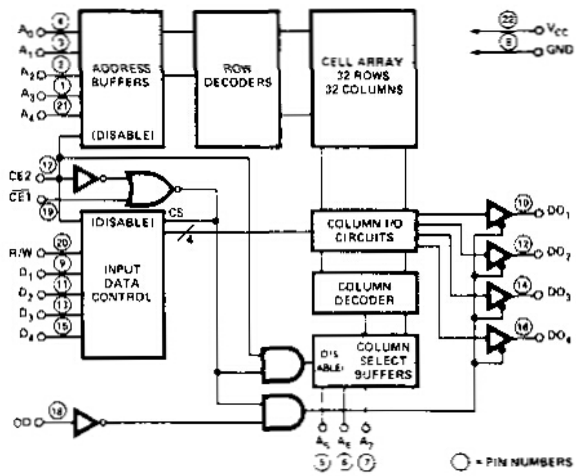



**LOGIC SYMBOL**

5101




**BLOCK DIAGRAM**




**PIN NAMES**

| | | | |
|---|---|---|---|
| $DI_1$–$DI_4$ | DATA INPUT | OD | OUTPUT DISABLE |
| $A_0$–$A_7$ | ADDRESS INPUTS | $DO_1$–$DO_4$ | DATA OUTPUT |
| R/W | READ/WRITE INPUT | $V_{CC}$ | POWER (+5V) |
| $\overline{CE1}$, CE2 | CHIP ENABLE | | |

## Absolute Maximum Ratings *

Ambient Temperature Under Bias . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . . . . -65°C to +150°C
Voltage On Any Pin
With Respect to Ground . . . . -0.3V to $V_{CC}$ +0.3V
Maximum Power Supply Voltage . . . . . . . . . +7.0V
Power Dissipation . . . . . . . . . . . . . . . . . . 1 Watt

*COMMENT:

*Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition 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. and Operating Characteristics for 5101, 5101-3, 5101L, 5101L-3

$T_A$ = 0°C to 70°C, $V_{CC}$ = 5V ±5% unless otherwise specified.

| Symbol | Parameter | Min. | Typ.[1] | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $I_{LI}$ [2] | Input Current | | 5 | | nA | $V_{IN}$ = 0 to 5.25V |
| $I_{LOH}$ [2] | Output High Leakage | | | 1 | µA | $\overline{CE1}$ = 2.2V, $V_{OUT}$ = $V_{CC}$ |
| $I_{LOL}$ [2] | Output Low Leakage | | | 1 | µA | $\overline{CE1}$ = 2.2V, $V_{OUT}$ = 0.0V |
| $I_{CC1}$ | Operating Current | | 9 | 22 | mA | $V_{IN}$ = $V_{CC}$ Except $\overline{CE1}$ ≤ 0.01V Outputs Open |
| $I_{CC2}$ | Operating Current | | 13 | 27 | mA | $V_{IN}$ = 2.2V Except $\overline{CE1}$ ≤ 0.65V Outputs Open |
| 5101 $I_{CCL}$ [2] | Standby Current | | | 15 | µA | $V_{IN}$ = 0 to $V_{CC}$, Except CE2 ≤ 0.2V |
| 5101-3 $I_{CCL}$ [2] | Standby Current | | | 200 | µA | $V_{IN}$ = 0 to $V_{CC}$, Except CE2 ≤ 0.2V |
| $V_{IL}$ | Input "Low" Voltage | -0.3 | | 0.65 | V | |
| $V_{IH}$ | Input "High" Voltage | 2.2 | | $V_{CC}$ | V | |
| $V_{OL}$ | Output "Low" Voltage | | | 0.4 | V | $I_{OL}$ = 2.0mA |
| $V_{OH}$ | Output "High" Voltage | 2.4 | | | V | $I_{OH}$ = 1.0mA |

**Low $V_{CC}$ Data Retention Characteristics (For 5101L and 5101L-3)** $T_A$ = 0°C to 70°C

| Symbol | Parameter | Min. | Typ.[1] | Max. | Unit | Test Conditions | |
|---|---|---|---|---|---|---|---|
| $V_{DR}$ | $V_{CC}$ for Data Retention | 2.0 | | | V | CE2 ≤ 0.2V | |
| 5101L $I_{CCDR}$ | Data Retention Current | | | 15 | µA | | $V_{DR}$ = 2.0V |
| 5101L-3 $I_{CCDR}$ | Data Retention Current | | | 200 | µA | | $V_{DR}$ = 2.0V |
| $t_{CDR}$ | Chip Deselect to Data Retention Time | 0 | | | ns | | |
| $t_R$ | Operation Recovery Time | $t_{RC}$ [3] | | | ns | | |

NOTES: 1. Typical values are $T_A$ = 25°C and nominal supply voltage. 2. Current through all inputs and outputs included in $I_{CCL}$ measurement. 3. $t_{RC}$ = Read Cycle Time.

# A.C. Characteristics for 5101, 5101-3, 5101L, 5101L-3

**READ CYCLE** $T_A = 0°C$ to $70°C$, $V_{CC} = 5V \pm 5\%$, unless otherwise specified.

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{RC}$ | Read Cycle | 650 | | | ns | (See below) |
| $t_A$ | Access Time | | | 650 | ns | |
| $t_{CO1}$ | Chip Enable ($\overline{CE1}$) to Output | | | 600 | ns | |
| $t_{CO2}$ | Chip Enable (CE2) to Output | | | 700 | ns | |
| $t_{OD}$ | Output Disable To Output | | | 350 | ns | |
| $t_{DF}$ | Data Output to High Z State | 0 | | 150 | ns | |
| $t_{OH1}$ | Previous Read Data Valid with Respect to Address Change | 0 | | | ns | |
| $t_{OH2}$ | Previous Read Data Valid with Respect to Chip Enable | 0 | | | ns | |

**WRITE CYCLE**

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{WC}$ | Write Cycle | 650 | | | ns | (See below) |
| $t_{AW}$ | Write Delay | 150 | | | ns | |
| $t_{CW1}$ | Chip Enable ($\overline{CE1}$) To Write | 550 | | | ns | |
| $t_{CW2}$ | Chip Enable (CE2) To Write | 550 | | | ns | |
| $t_{DW}$ | Data Setup | 400 | | | ns | |
| $t_{DH}$ | Data Hold | 100 | | | ns | |
| $t_{WP}$ | Write Pulse | 400 | | | ns | |
| $t_{WR}$ | Write Recovery | 50 | | | ns | |
| $t_{DS}$ | Output Disable Setup | 150 | | | ns | |

## A. C. CONDITIONS OF TEST

Input Pulse Levels: +0.65 Volt to 2.2 Volt

Input Pulse Rise and Fall Times: 20nsec

Timing Measurement Reference Level: 1.5 Volt

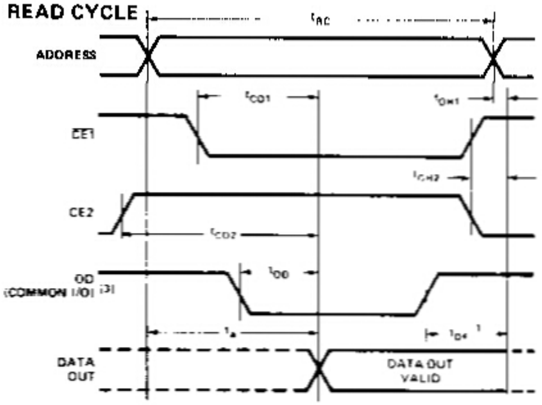
Output Load: 1 TTL Gate and $C_L$ = 100pF

## Capacitance[2] $T_A = 25°C$, f = 1MHz

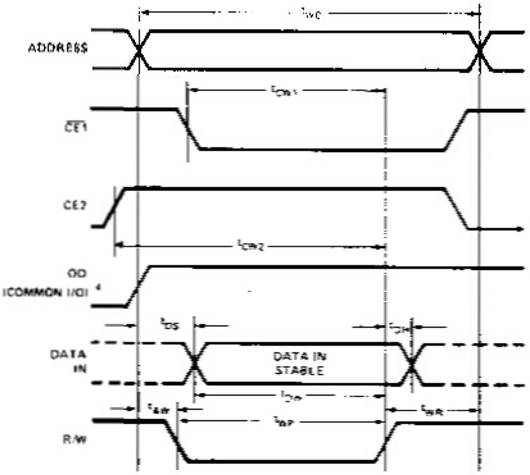
| Symbol | Test | Limits (pF) | |
|---|---|---|---|
| | | Typ. | Max. |
| $C_{IN}$ | Input Capacitance (All Input Pins) $V_{IN}$ = 0V | 4 | 8 |
| $C_{OUT}$ | Output Capacitance $V_{OUT}$ = 0V | 8 | 12 |

# Waveforms

READ CYCLE

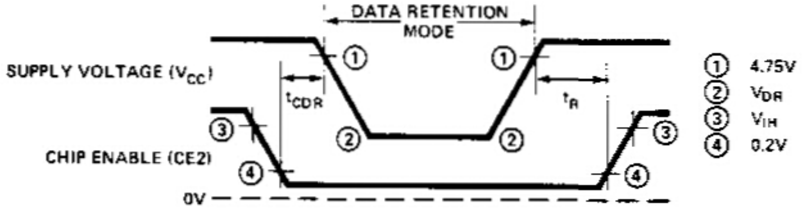



WRITE CYCLE




NOTES: 1. Typical values are for $T_A = 25°C$ and nominal supply voltage.
2. This parameter is periodically sampled and is not 100% tested.
3. OD may be tied low for separate I/O operation.
4. During the write cycle, OD is "high" for common I/O and "don't care" for separate I/O operation.

## Low $V_{CC}$ Data Retention



# Schottky Bipolar 8210

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

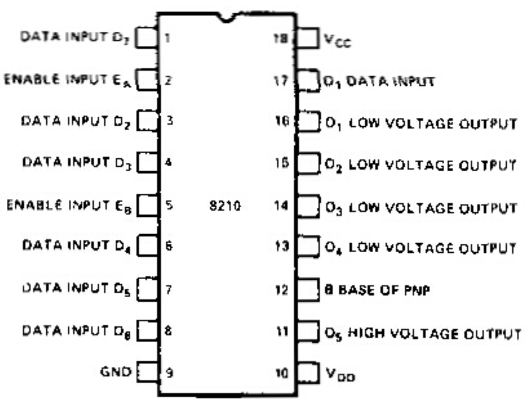
- **Four Low Voltage Drivers**
- **One High Voltage Driver**
- **TTL and DTL Compatible Inputs**
- **Outputs Compatible with 8107A MOS Memories**
- **Operates from Standard Bipolar and MOS Power Supplies**
- **Maximum MOS Device Protection — Output Clamp Diodes**

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.

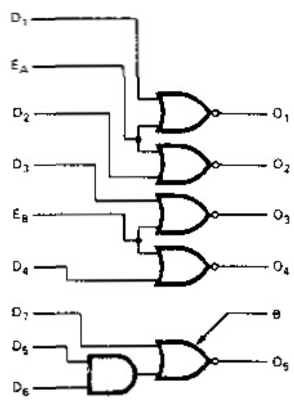
The four low voltage drivers feature two common enable inputs per pair of drivers which permits address or data decoding. The high voltage driver swings the 12 volts required to drive the chip enable (clock) input for the 8107A.

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 2N5057 is recommended.

PIN CONFIGURATION




LOGIC SYMBOL



## A.C. Characteristics $T_A = 0°C$ to $70°C$, $V_{CC} = 5.0V \pm 5\%$, $V_{DD} = 12V \pm 5\%$

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $t_{Ld+}$ | Delay Plus Rise Time for Low Voltage Drivers | 5 | 13 | 20 | ns |
| $t_{Ld-}$ | Delay Plus Fall Time for Low Voltage Drivers | 5 | 13 | 20 | ns |
| $t_{Hd+}$ | Delay Plus Rise Time for High Voltage Driver | 10 | 30 | 40 | ns |
| $t_{Hd-}$ | Delay Plus Fall Time for High Voltage Driver | 10 | 30 | 40 | ns |

## Capacitance* $T_A = 25°C$

| Symbol | Test | Typ. | Max. |
|---|---|---|---|
| $C_{IN}$ | Input Capacitance | 6pF | 12pF |

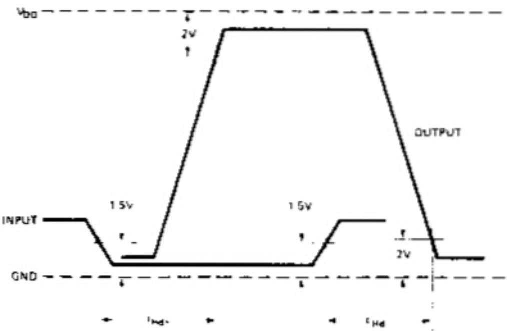
*This parameter is periodically sampled and is not 100% tested. Condition of measurement is f = 1 MHz, $V_{bias} = 2V$, $V_{CC} = 0V$, and $T_A = 25°C$.

### A.C. CONDITIONS OF TEST

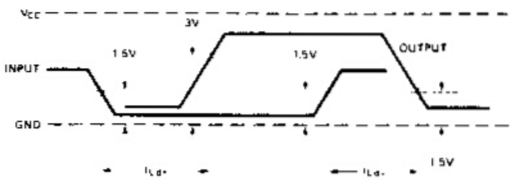
Test Load: $C_L = 200pF$ for Low Voltage Drivers,
$C_L = 350pF$ for High Voltage Drivers
Input Pulse Amplitudes: 3.0V
Input Pulse Rise and Fall Times: 5 ns between 1 volt and 2 volts
Measurement Points: See Waveforms

## Waveforms

HIGH VOLTAGE DRIVER

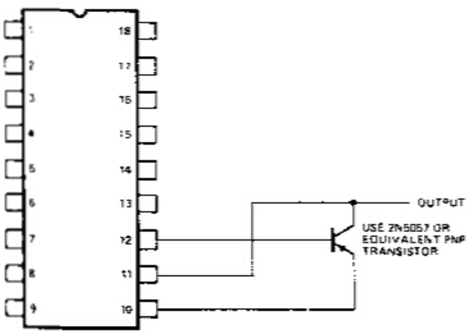



LOW VOLTAGE DRIVER




## Application

HIGH VOLTAGE OUTPUT CONNECTIONS



## Absolute Maximum Ratings*

Temperature Under Bias . . . . . . . . . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . . . . . . −65°C to +150°C
Supply Voltage, $V_{CC}$ . . . . . . . . . . . . . . . . −0.5 to +7V
Supply Voltage, $V_{DD}$ . . . . . . . . . . . . . . . . −0.5 to +13V
All Input Voltages . . . . . . . . . . . . . . . . −1.0 to +5.5V
Outputs for Low Voltage Drivers . . . . . . . . −0.5 to +7V
Outputs for Clock Driver . . . . . . . . . . . . −1.0 to +13V
Power Dissipation at 25°C . . . . . . . . . . . . . . . . . 2W

*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, $V_{CC}$ = 5.0V ± 5%, $V_{DD}$ = 12V ± 5%

| Symbol | Parameter | Min. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|
| $I_{FD}$ | Data Input Load Current | | −0.25 | mA | $V_F$ = 0.45V |
| $I_{FE}$ | Enable Input Load Current | | −0.50 | mA | $V_F$ = 0.45V |
| $I_{RD}$ | Data Input Leakage Current | | 10 | µA | $V_R$ = 12.6V |
| $I_{RE}$ | Enable Input Leakage Current | | 20 | µA | $V_R$ = 12.6V |
| $V_{OL}$ | Output Low Voltage for all Drivers | | 0.45 | V | $I_{OL}$ = 3mA, $V_{IH}$ = 2V |
| | | −1.0 | | V | $I_{OL}$ = −5mA |
| $V_{OH1}$ | Output High Voltage for Low Voltage Drivers | $V_{CC}$ −1.0 | | V | $I_{OH}$ = −1mA, $V_{IL}$ = 0.8V |
| | | | $V_{CC}$ +1.0 | V | $I_{OH}$ = 5mA |
| $V_{OH2}$ | Output High Voltage for High Voltage Driver | $V_{DD}$ −0.75 | | V | $I_{OH}$ = −1mA, $V_{IL}$ = 0.8V |
| | | | $V_{DD}$ + 0.5 | V | $I_{OH}$ = 5mA |
| $I_{O1}$ | Pulsed Output Sink Current for Low Voltage Drivers | 75 | | mA | $V_O$ = 2V, $V_{IH}$ = 2V |
| $I_{O2}$ | Pulsed Output Sink Current for High Voltage Driver | 100 | | mA | $V_O$ = 3V, $V_{IH}$ = 2V |
| $I_{O3}$ | Pulsed Output Source Current for Low Voltage Drivers | −75 | | mA | $V_O$ = $V_{CC}$ −1.5V, $V_{IL}$ = 0.8V |
| $I_{O4}$ | Pulsed Output Source Current for High Voltage Driver | −100 | | mA | $V_O$ = $V_{DD}$ −3V, $V_{IL}$ = 0.8V |
| $V_{IL}$ | Input Low Voltage, All Inputs | | 0.8 | V | |
| $V_{IH}$ | Input High Voltage, All Inputs | 2 | | V | |

### POWER SUPPLY CURRENT DRAIN AND POWER DISSIPATION

All driver outputs are in the state indicated

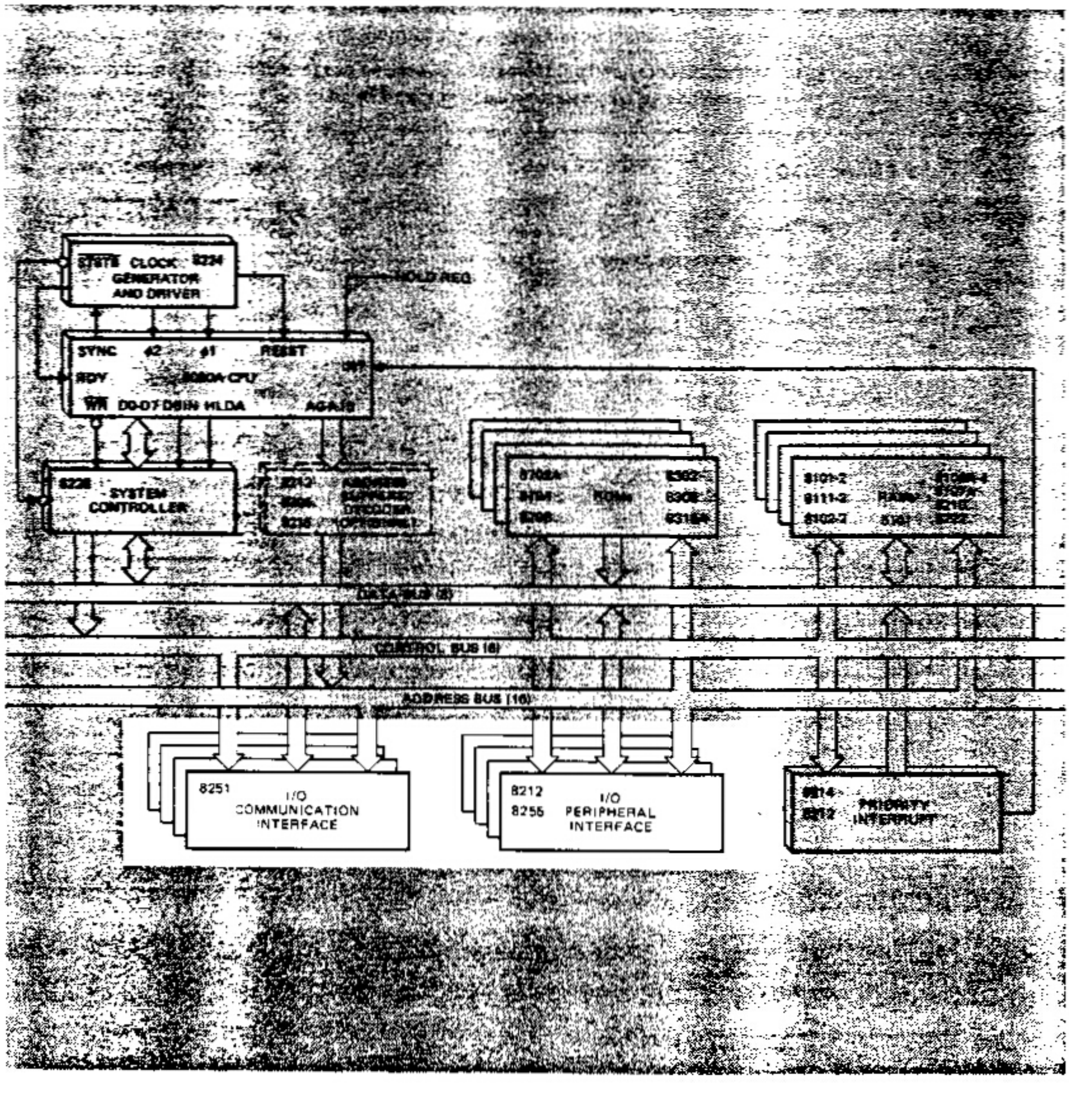
| Symbol | Parameter | Typ.[1] | Max. | Unit | Test Conditions — Input states to ensure the following output states: All Low Voltage Outputs | High Voltage Output | Additional Test Conditions |
|---|---|---|---|---|---|---|---|
| $I_{CC1}$ | Current from $V_{CC}$ | 26 | 35 | mA | Low | Low | $V_{CC}$ = 5.25V, $V_{DD}$ = 12.6V |
| $I_{DD1}$ | Current from $V_{DD}$ | 12 | 16 | mA | Low | Low | |
| $P_{D1}$ | Power Dissipation | 290 | 390 | mW | Low | Low | |
| $I_{CC2}$ | Current from $V_{CC}$ | 21 | 28 | mA | Low | High | |
| $I_{DD2}$ | Current from $V_{DD}$ | 26 | 35 | mA | Low | High | |
| $P_{D2}$ | Power Dissipation | 450 | 600 | mW | Low | High | |
| $I_{CC3}$ | Current from $V_{CC}$ | 19 | 25 | mA | High | Low | |
| $I_{DD3}$ | Current from $V_{DD}$ | 12 | 16 | mA | High | Low | |
| $P_{D3}$ | Power Dissipation | 260 | 340 | mW | High | Low | |
| $I_{CC4}$ | Current from $V_{CC}$ | 14 | 18 | mA | High | High | |
| $I_{DD4}$ | Current from $V_{DD}$ | 26 | 35 | mA | High | High | |
| $P_{D4}$ | Power Dissipation | 410 | 550 | mW | High | High | |

[1] This parameter is periodically sampled and is not 100% tested. Condition of measurement is $T_A$ = 25°C, $V_{CC}$ = 5V, $V_{DD}$ = 12V.

intel® Microcomputer Systems

# I/O

# 8212
# 8255
# 8251



intel®

# Schottky Bipolar 8212

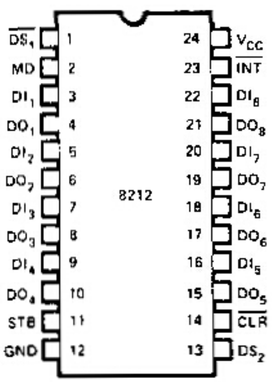
## EIGHT-BIT INPUT/OUTPUT PORT

- **Fully Parallel 8-Bit Data Register and Buffer**
- **Service Request Flip-Flop for Interrupt Generation**
- **Low Input Load Current — .25 mA Max.**
- **Three State Outputs**
- **Outputs Sink 15 mA**
- **3.65V Output High Voltage for Direct Interface to 8080 CPU or 8008 CPU**
- **Asynchronous Register Clear**
- **Replaces Buffers, Latches and Multiplexers in Microcomputer Systems**
- **Reduces System Package Count**

The 8212 input/output port consists of an 8-bit latch with 3-state output buffers along with control and device selection logic. Also included is a service request flip-flop for the generation and control of interrupts to the microprocessor.

The device is multimode in nature. It can be used to implement latches, gated buffers or multiplexers. Thus, all of the principal peripheral and input/output functions of a microcomputer system can be implemented with this device.

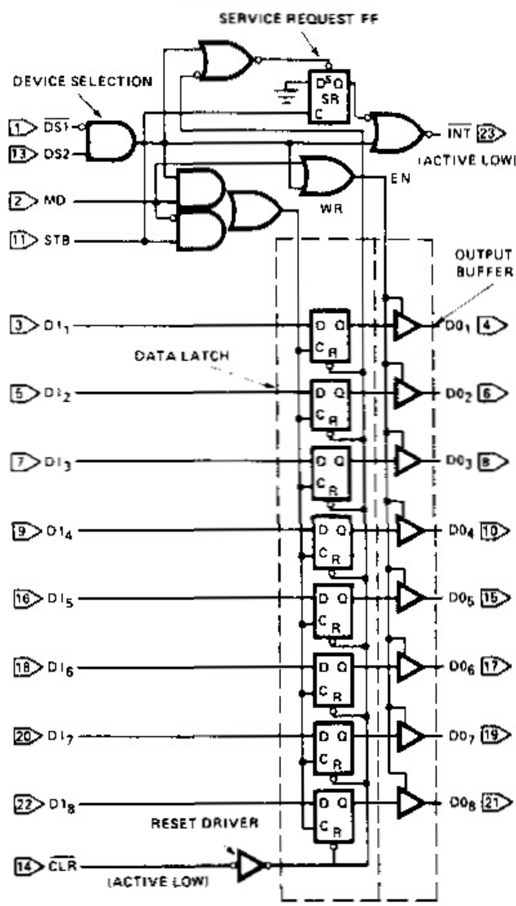
### PIN CONFIGURATION




### PIN NAMES

| | |
|---|---|
| $DI_1$-$DI_8$ | DATA IN |
| $DO_1$-$DO_8$ | DATA OUT |
| $\overline{DS_1}$-$DS_2$ | DEVICE SELECT |
| MD | MODE |
| STB | STROBE |
| $\overline{INT}$ | INTERRUPT (ACTIVE LOW) |
| $\overline{CLR}$ | CLEAR (ACTIVE LOW) |

### LOGIC DIAGRAM



## Functional Description

### Data Latch

The 8 flip-flops that make up the data latch are of a "D" type design. The output (Q) of the flip-flop will follow the data input (D) while the clock input (C) is high. Latching will occur when the clock (C) returns low.

The data latch is cleared by an asynchronous reset input ($\overline{CLR}$). (Note: Clock (C) Overides Reset ($\overline{CLR}$).)

### Output Buffer

The outputs of the data latch (Q) are connected to 3-state, non-inverting output buffers. These buffers have a common control line (EN); this control line either enables the buffer to transmit the data from the outputs of the data latch (Q) or disables the buffer, forcing the output into a high impedance state. (3-state)

This high-impedance state allows the designer to connect the 8212 directly onto the microprocessor bi-directional data bus.

### Control Logic

The 8212 has control inputs $\overline{DS1}$, DS2, MD and STB. These inputs are used to control device selection, data latching, output buffer state and service request flip-flop.

### $\overline{DS1}$, DS2 (Device Select)

These 2 inputs are used for device selection. When $\overline{DS1}$ is low and DS2 is high ($\overline{DS1} \cdot DS2$) the device is selected. In the selected state the output buffer is enabled and the service request flip-flop (SR) is asynchronously set.

### MD (Mode)

This input is used to control the state of the output buffer and to determine the source of the clock input (C) to the data latch.

When MD is high (output mode) the output buffers are enabled and the source of clock (C) to the data latch is from the device selection logic ($\overline{DS1} \cdot DS2$).

When MD is low (input mode) the output buffer state is determined by the device selection logic ($\overline{DS1} \cdot DS2$) and the source of clock (C) to the data latch is the STB (Strobe) input.

### STB (Strobe)

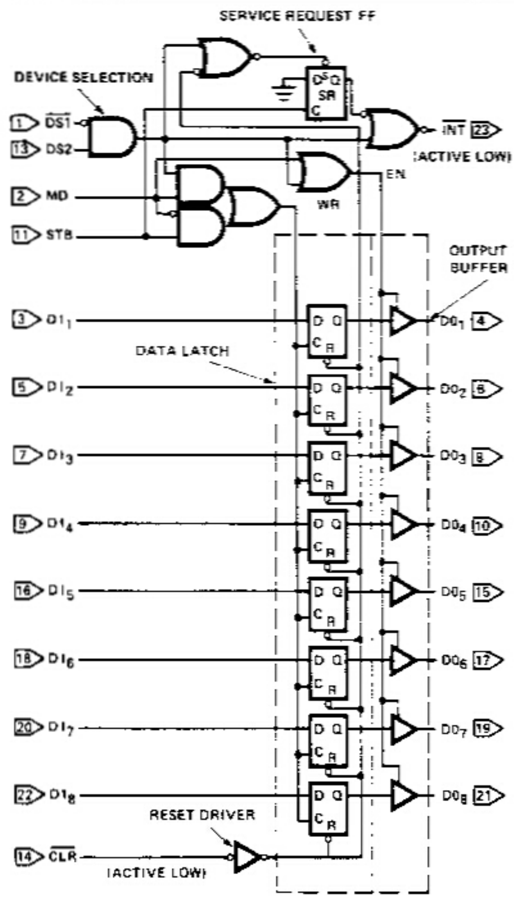
This input is used as the clock (C) to the data latch for the input mode MD = 0) and to synchronously reset the service request flip-flop (SR).

Note that the SR flip-flop is negative edge triggered.

### Service Request Flip-Flop

The (SR) flip-flop is used to generate and control interrupts in microcomputer systems. It is asynchronously set by the $\overline{CLR}$ input (active low). When *the (SR) flip-flop is set it is in the non-interrupting* state.

The output of the (SR) flip-flop (Q) is connected to an inverting input of a "NOR" gate. The other input to the "NOR" gate is non-inverting and is connected to the device selection logic ($\overline{DS1} \cdot DS2$). The output of the "NOR" gate ($\overline{INT}$) is active low (interrupting state) for connection to active low input priority generating circuits.




| STB | MD | $(\overline{DS_1} \cdot DS_2)$ | DATA OUT EQUALS |
|---|---|---|---|
| 0 | 0 | 0 | 3-STATE |
| 1 | 0 | 0 | 3-STATE |
| 0 | 1 | 0 | DATA LATCH |
| 1 | 1 | 0 | DATA LATCH |
| 0 | 0 | 1 | DATA LATCH |
| 1 | 0 | 1 | DATA IN |
| 0 | 1 | 1 | DATA IN |
| 1 | 1 | 1 | DATA IN |

$\overline{CLR}$ – RESETS DATA LATCH
SETS SR FLIP-FLOP
(NO EFFECT ON OUTPUT BUFFER)

| CLR | $(\overline{DS_1} \cdot DS_2)$ | STB | *SR | INT |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | ↓ | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | ↓ | 1 | 0 |

*INTERNAL SR FLIP-FLOP

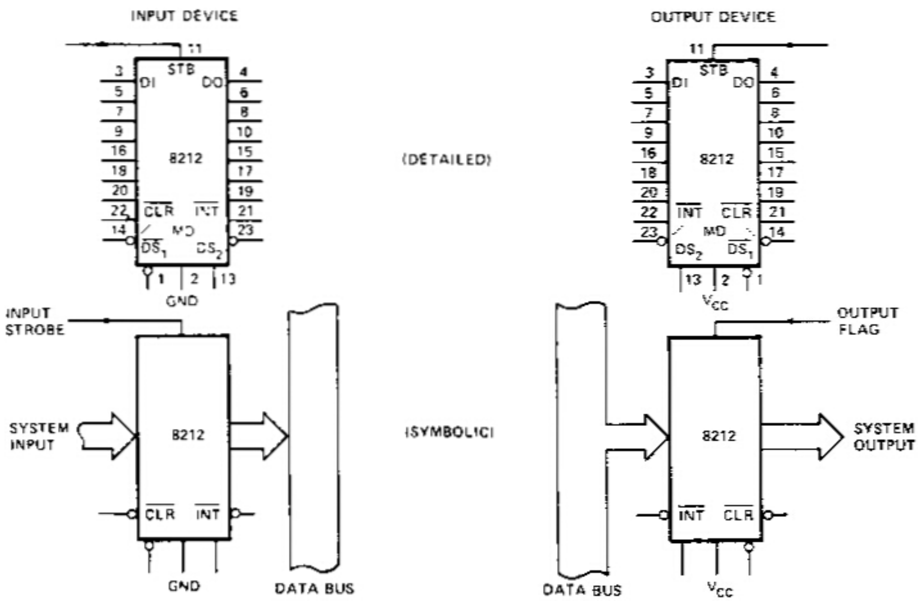
# Applications Of The 8212 -- For Microcomputer Systems



## I. Basic Schematic Symbols

Two examples of ways to draw the 8212 on system schematics—(1) the top being the detailed view showing pin numbers, and (2) the bottom being the symbolic view showing the system input or output as a system bus (bus containing 8 parallel lines). The output to the data bus is symbolic in referencing 8 parallel lines.

BASIC SCHEMATIC SYMBOLS



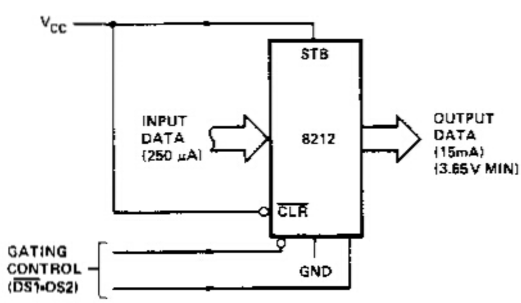

## II. Gated Buffer (3 - STATE)

The simplest use of the 8212 is that of a gated buffer. By tying the mode signal low and the strobe input high, the data latch is acting as a straight through gate. The output buffers are then enabled from the device selection logic $\overline{DS1}$ and DS2.

When the device selection logic is false, the outputs are 3-state.

When the device selection logic is true, the input data from the system is directly transferred to the output. The input data load is 250 micro amps. The output data can sink 15 milli amps. The minimum high output is 3.65 volts.

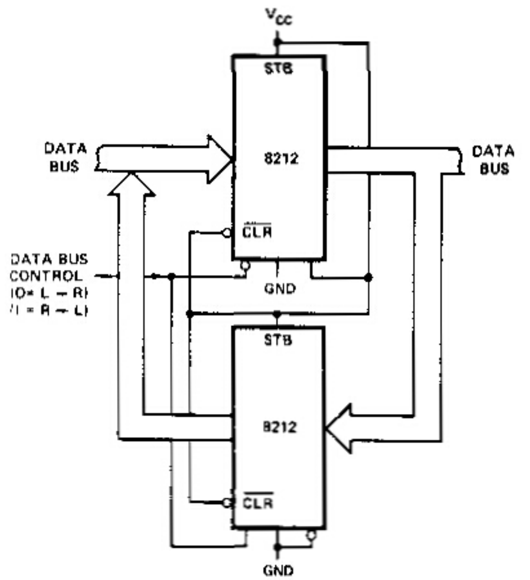
GATED BUFFER
3-STATE



## III. Bi-Directional Bus Driver

A pair of 8212's wired (back-to-back) can be used as a symmetrical drive, bi-directional bus driver. The devices are controlled by the data bus input control which is connected to $\overline{DS1}$ on the first 8212 and to DS2 on the second. One device is active, and acting as a straight through buffer the other is in 3-state mode. This is a very useful circuit in small system design.

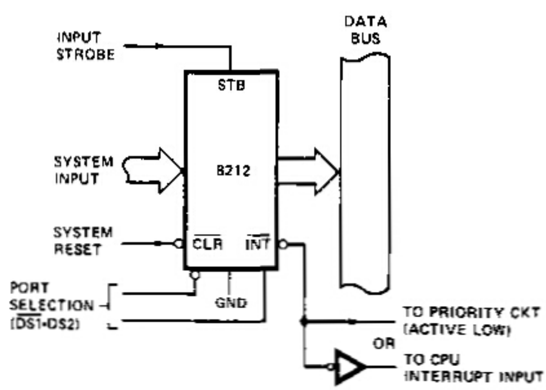
BI-DIRECTIONAL BUS DRIVER




## IV. Interrupting Input Port

This use of an 8212 is that of a system input port that accepts a strobe from the system input source, which in turn clears the service request flip-flop and interrupts the processor. The processor then goes through a service routine, identifies the port, and causes the device selection logic to go true — enabling the system input data onto the data bus.

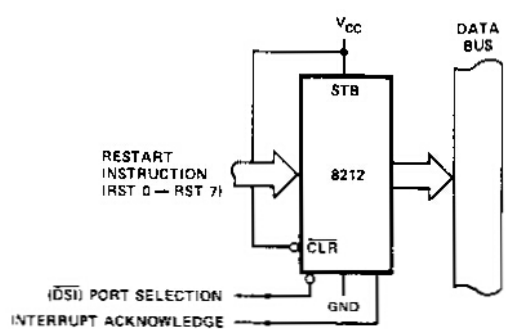
INTERRUPTING INPUT PORT




## V. Interrupt Instruction Port

The 8212 can be used to gate the interrupt instruction, normally RESTART instructions, onto the data bus. The device is enabled from the interrupt acknowledge signal from the microprocessor and from a port selection signal. This signal is normally tied to ground. ($\overline{DS1}$ could be used to multiplex a variety of interrupt instruction ports onto a common bus).

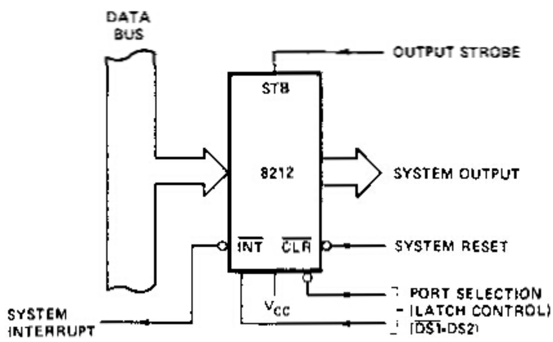
INTERRUPT INSTRUCTION PORT



## VI. Output Port (With Hand-Shaking)

The 8212 can be used to transmit data from the data bus to a system output. The output strobe could be a hand-shaking signal such as "reception of data" from the device that the system is outputting to. It in turn, can interrupt the system signifying the reception of data. The selection of the port comes from the device selection logic. ($\overline{DS1} \cdot DS2$)

OUTPUT PORT (WITH HAND-SHAKING)



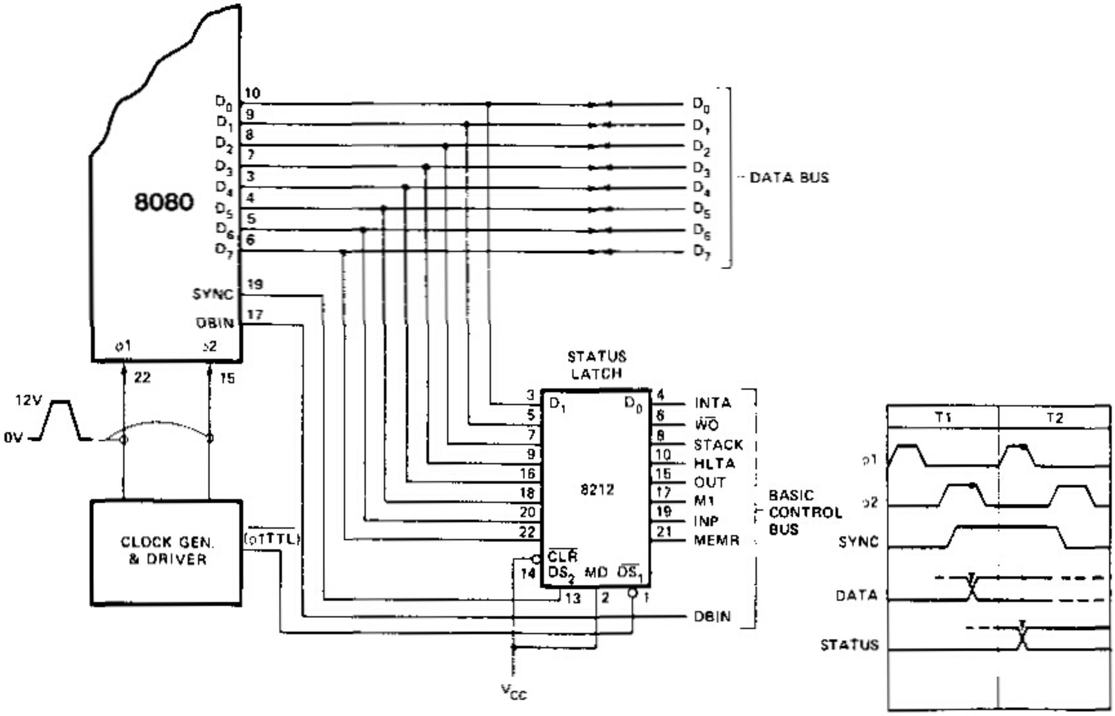

## VII. 8080 Status Latch

Here the 8212 is used as the status latch for an 8080 microcomputer system. The input to the 8212 latch is directly from the 8080 data bus. Timing shows that when the SYNC signal is true, which is connected to the DS2 input and the phase 1 signal is true, which is a TTL level coming from the clock generator; then, the status data will be latched into the 8212.

Note: The mode signal is tied high so that the output on the latch is active and enabled all the time.

It is shown that the two areas of concern are the bidirectional data bus of the microprocessor and the control bus.

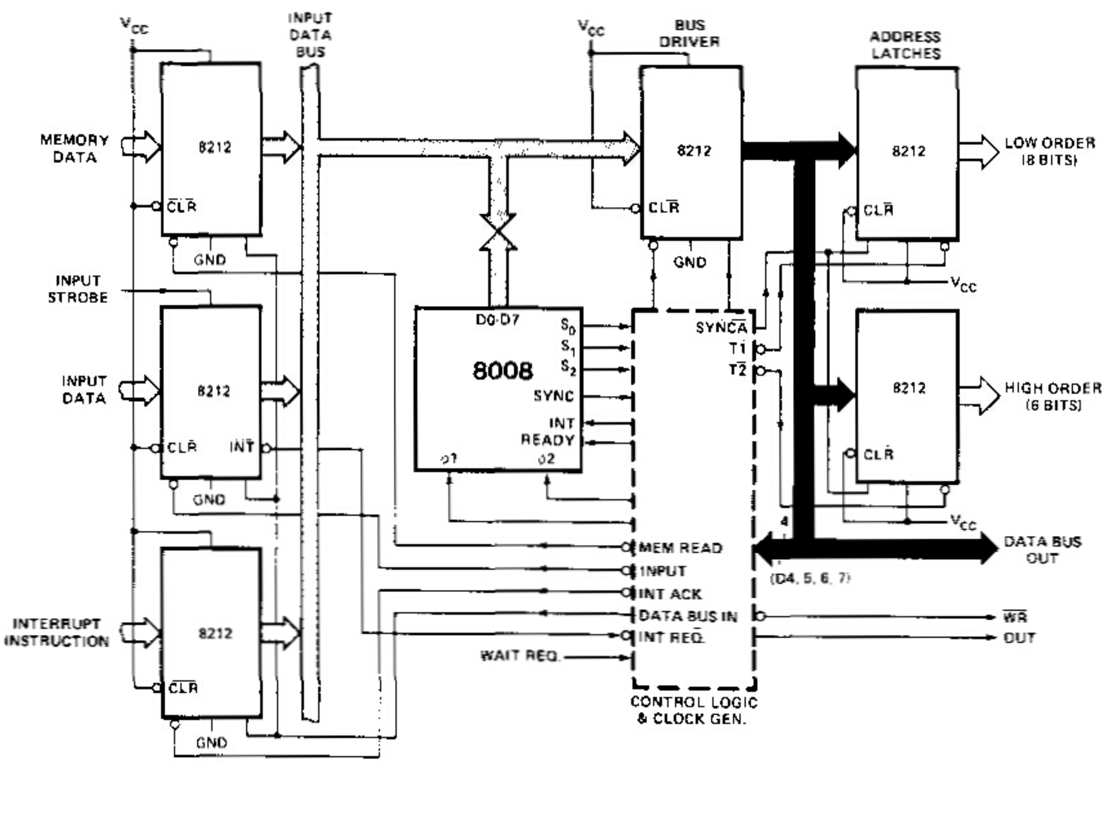
8080 STATUS LATCH



## VIII. 8008 System

This shows the 8212 used in an 8008 microcomputer system. They are used to multiplex the data from three different sources onto the 8008 input data bus. The three sources of data are: memory data, input data, and the interrupt instruction. The 8212 is also used as the uni-directional bus driver to provide a proper drive to the address latches (both low order and high order are also 8212's) and to provide adequate drive to the output data bus. The control of these six 8212's in the 8008 system is provided by the control logic and clock generator circuits. These circuits consist of flip-flops, decoders, and gates to generate the control functions necessary for 8008 microcomputer systems. Also note that the input data port has a strobe input. This allows the processor to be interrupted from the input port directly. The control of the input bus consists of the data bus input signal, control logic, and the appropriate status signal for bus discipline whether memory read, input, or interrupt acknowledge. The combination of these four signals determines which one of these three devices will have access to the input data bus. The bus driver, which is implemented in an 8212, is also controlled by the control logic and clock generator so it can be 3-stated when necessary and also as a control transmission device to the address latches. Note: The address latches can be 3-stated for DMA purposes and they provide 15 milli amps drive, sufficient for large bus systems.

8008 SYSTEM



## IX. 8080 System

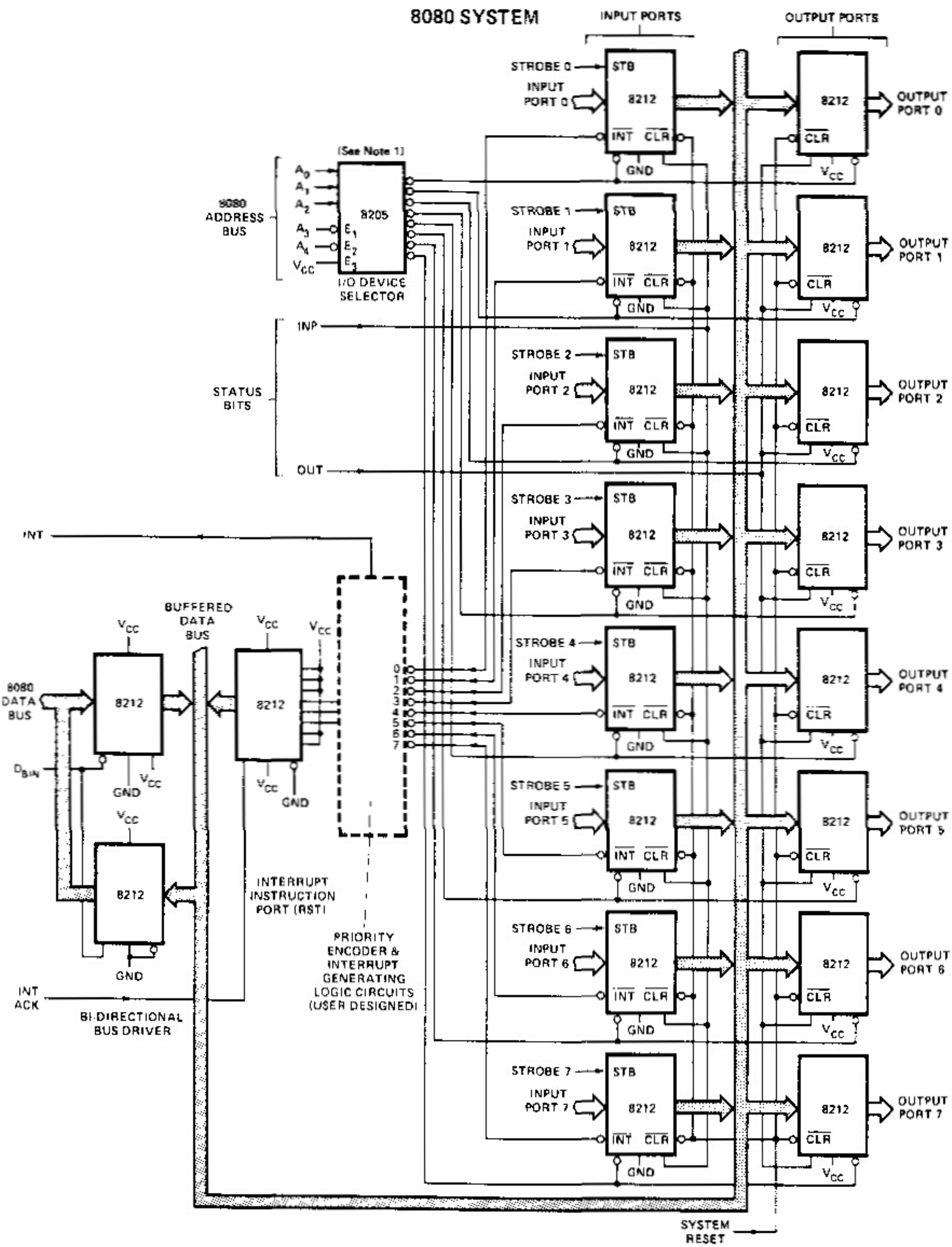
This drawing shows the 8212 used in the I/O section of an 8080 microcomputer system. The system consists of 8 input ports, 8 output ports, 8 level priority systems, and a bidirectional bus driver. (The data bus within the system is darkened for emphasis).

Basically, the operation would be as follows: The 8 ports, for example, could be connected to 8 keyboards, each keyboard having its own priority level. The keyboard could provide a strobe input of its own which would clear the service request flip-flop. The $\overline{INT}$ signals are connected to an 8 level priority encoding circuit. This circuit provides a positive true level to the central processor (INT) along with a three-bit code to the interrupt instruction port for the generation of RESTART instructions. Once the processor has been interrupted and it acknowledges the reception of the interrupt, the Interrupt Acknowledge signal is generated. This signal transfers data in the form of a RESTART instruction onto the buffered data bus. When the DBIN signal is true this RESTART instruction is gated into the microcomputer, in this case, the 8080 CPU. The 8080 then performs a software controlled interrupt service routine, saving the status of its current operation in the push-down stack and performing an INPUT instruction. The INPUT instruction thus sets the INP status bit, which is common to all input ports.

Also present is the address of the device on the 8080 address bus which in this system is connected to an 8205, one out of eight decoder with active low outputs. These active low outputs will enable one of the input ports, the one that interrupted the processor, to put its data onto the buffered data bus to be transmitted to the CPU when the data bus input signal is true. The processor can also output data from the 8080 data bus to the buffered data bus when the data bus input signal is false. Using the same address selection technique from the 8205 decoder and the output status bit, we can select with this system one of eight output ports to transmit the data to the system's output device structure.

Note: This basic I/O configuration for the 8080 can be expanded to 256 input devices and 256 output devices all using 8212 and, of course, the appropriate decoding.

Note that the 8080 is a 3.3-volt minimum high input requirement and that the 8212 has a 3.65-volt minimum high output providing the designer with a 350 milli volt noise margin worst case for 8080 systems when using the 8212.




Note 1. This basic I/O configuration for the 8080 can be expanded to 256 input devices and 256 output devices all using 8212 and the appropriate decoding.

## Absolute Maximum Ratings*

Temperature Under Bias Plastic . . −65°C to +75°C
Storage Temperature . . . . . . . . . −65°C to +160°C
All Output or Supply Voltages . . . . −0.5 to +7 Volts
All Input Voltages . . . . . . . . . . . . −1.0 to 5.5 Volts
Output Currents . . . . . . . . . . . . . . . . . . . 125 mA

*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 at any other condition above those indicated in the operational sections of this specification is not implied.

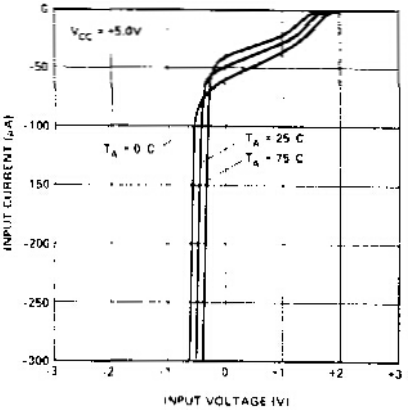
## D.C. Characteristics

$T_A$ = 0°C to +75°C $V_{CC}$ = +5V ±5%

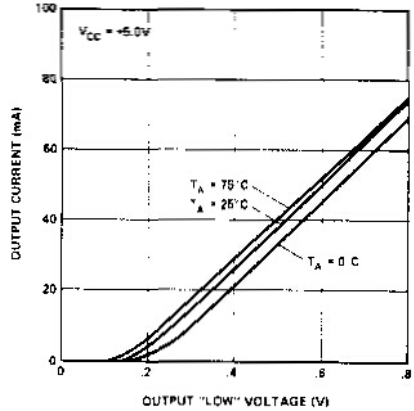
| Symbol | Parameter | Limits Min. | Limits Typ. | Limits Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $I_F$ | Input Load Current ACK, $DS_2$, CR, $DI_1$-$DI_8$ Inputs | | | −.25 | mA | $V_F$ = .45V |
| $I_F$ | Input Load Current MD Input | | | −.75 | mA | $V_F$ = .45V |
| $I_F$ | Input Load Current $DS_1$ Input | | | −1.0 | mA | $V_F$ = .45V |
| $I_R$ | Input Leakage Current ACK, DS, CR, $DI_1$-$DI_8$ Inputs | | | 10 | μA | $V_R$ = 5.25V |
| $I_R$ | Input Leakage Current MO Input | | | 30 | μA | $V_R$ = 5.25V |
| $I_R$ | Input Leakage Current $DS_1$ Input | | | 40 | μA | $V_R$ = 5.25V |
| $V_C$ | Input Forward Voltage Clamp | | | −1 | V | $I_C$ = −5 mA |
| $V_{IL}$ | Input "Low" Voltage | | | .85 | V | |
| $V_{IH}$ | Input "High" Voltage | 2.0 | | | V | |
| $V_{OL}$ | Output "Low" Voltage | | | .45 | V | $I_{OL}$ = 15 mA |
| $V_{OH}$ | Output "High" Voltage | 3.65 | 4.0 | | V | $I_{OH}$ = −1 mA |
| $I_{SC}$ | Short Circuit Output Current | -15 | | -75 | mA | $V_O$ = 0 V |
| $\lvert I_O \rvert$ | Output Leakage Current High Impedance State | | | 20 | μA | $V_O$ = .45V/5.25V |
| $I_{CC}$ | Power Supply Current | | 90 | 130 | mA | |

## Typical Characteristics

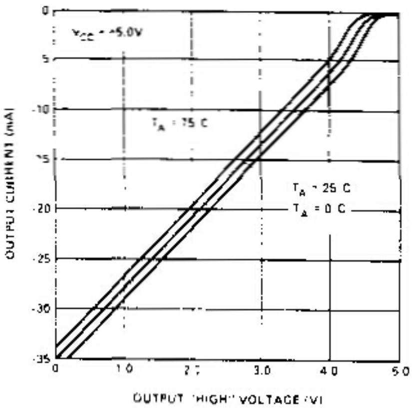
**INPUT CURRENT VS. INPUT VOLTAGE**



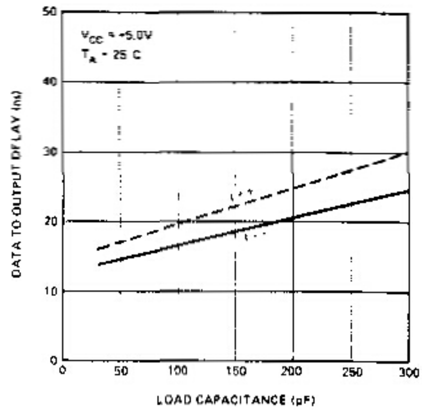

**OUTPUT CURRENT VS. OUTPUT "LOW" VOLTAGE**



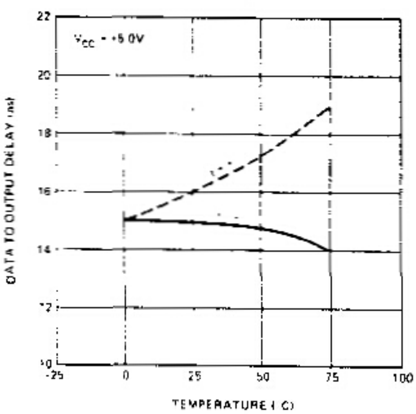

**OUTPUT CURRENT VS. OUTPUT "HIGH" VOLTAGE**



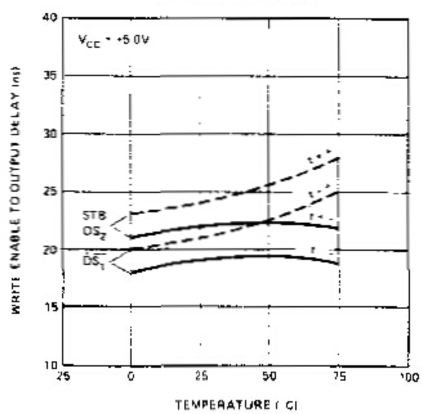

**DATA TO OUTPUT DELAY VS. LOAD CAPACITANCE**



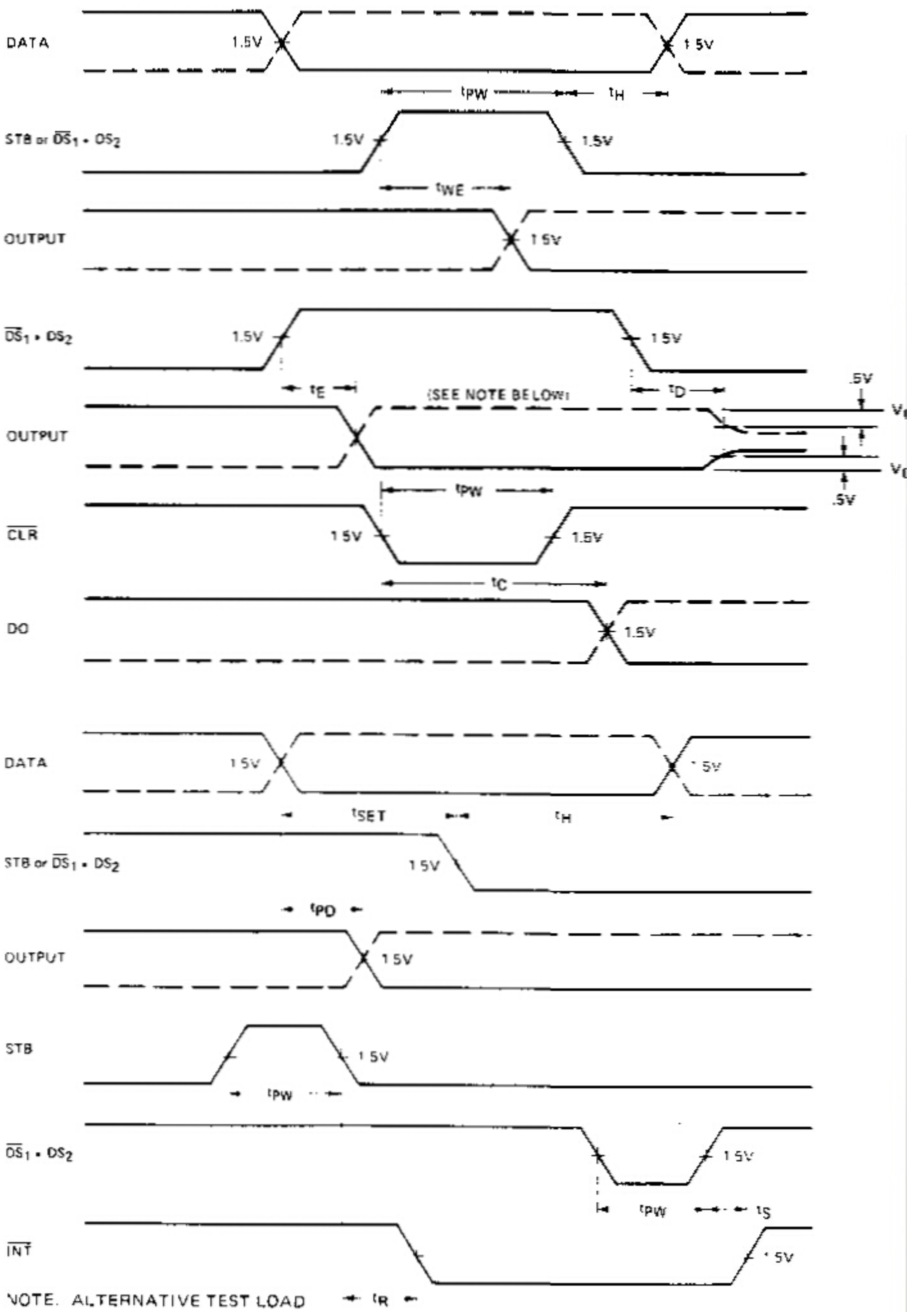

**DATA TO OUTPUT DELAY VS. TEMPERATURE**



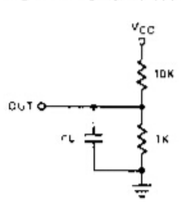

**WRITE ENABLE TO OUTPUT DELAY VS. TEMPERATURE**



## Timing Diagram




NOTE: ALTERNATIVE TEST LOAD



## A.C. Characteristics

$T_A = 0°C$ to $+75°C$ $V_{CC} = +5V \pm 5\%$

| Symbol | Parameter | Limits Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{pw}$ | Pulse Width | 30 | | | ns | |
| $t_{pd}$ | Data To Output Delay | | | 30 | ns | |
| $t_{we}$ | Write Enable To Output Delay | | | 40 | ns | |
| $t_{set}$ | Data Setup Time | 15 | | | ns | |
| $t_h$ | Data Hold Time | 20 | | | ns | |
| $t_r$ | Reset To Output Delay | | | 40 | ns | |
| $t_s$ | Set To Output Delay | | | 30 | ns | |
| $t_e$ | Output Enable/Disable Time | | | 45 | ns | |
| $t_c$ | Clear To Output Delay | | | 55 | ns | |

CAPACITANCE* $F = 1$ MHz $V_{BIAS} = 2.5V$ $V_{CC} = +5V$ $T_A = 25°C$

| Symbol | Test | LIMITS Typ. | Max. |
|---|---|---|---|
| $C_{IN}$ | $DS_1$ MD Input Capacitance | 9 pF | 12 pF |
| $C_{IN}$ | $DS_2$, CK, ACK, $DI_1$-$DI_8$ Input Capacitance | 5 pF | 9 pF |
| $C_{OUT}$ | $DO_1$-$DO_8$ Output Capacitance | 8 pF | 12 pF |

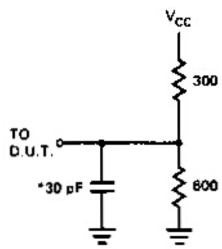
*This parameter is sampled and not 100% tested.

## Switching Characteristics

CONDITIONS OF TEST

Input Pulse Amplitude = 2.5 V
Input Rise and Fall Times 5 ns
Between 1V and 2V Measurements made at 1.5V
with 15 mA & 30 pF Test Load

TEST LOAD
15 mA & 30 pF




* INCLUDING JIG & PROBE CAPACITANCE

intel® Silicon Gate MOS 8255

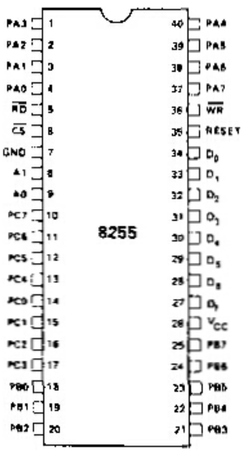
# PROGRAMMABLE PERIPHERAL INTERFACE

- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with MCS™-8 and MCS™-80 Microprocessor Families
- Direct Bit Set/Reset Capability Easing Control Application Interface
- 40 Pin Dual In-Line Package
- Reduces System Package Count

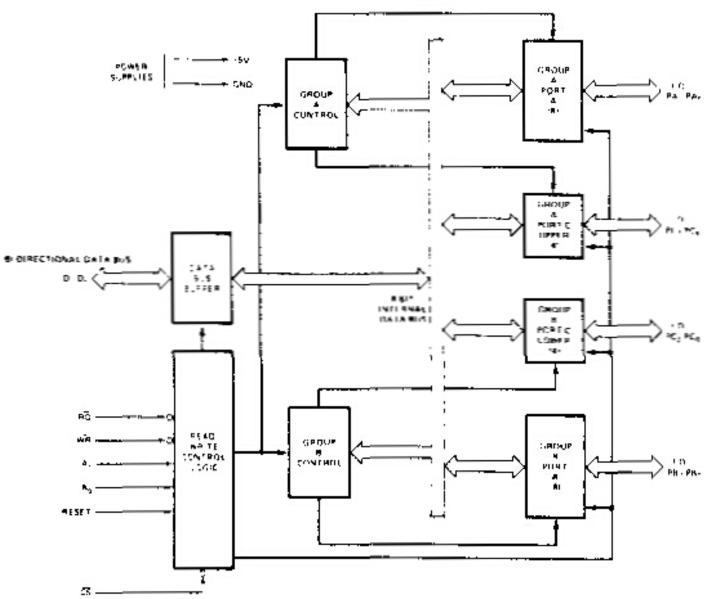
The 8255 is a general purpose programmable I/O device designed for use with both the 8008 and 8080 microprocessors. It has 24 I/O pins which may be individually programmed in two groups of twelve and used in three major modes of operation. In the first mode (Mode 0), each group of twelve I/O pins may be programmed in sets of 4 to be input or output. In Mode 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining four pins three are used for handshaking and interrupt control signals. The third mode of operation (Mode 2) is a Bidirectional Bus mode which uses 8 lines for a bidirectional bus, and five lines, borrowing one from the other group, for handshaking.

Other features of the 8255 include bit set and reset capability and the ability to source 1mA of current at 1.5 volts. This allows darlington transistors to be directly driven for applications such as printers and high voltage displays.

PIN CONFIGURATION




8255 BLOCK DIAGRAM




PIN NAMES

| | |
|---|---|
| $D_7$–$D_0$ | DATA BUS (BI-DIRECTIONAL) |
| RESET | RESET INPUT |
| $\overline{CS}$ | CHIP SELECT |
| $\overline{RD}$ | READ INPUT |
| $\overline{WR}$ | WRITE INPUT |
| A0, A1 | PORT ADDRESS |
| PA7-PA0 | PORT A (BIT) |
| PB7-PB0 | PORT B (BIT) |
| PC7-PC0 | PORT C (BIT) |
| Vcc | +5 VOLTS |
| GND | 0 VOLTS |

## 8255 BASIC FUNCTIONAL DESCRIPTION

### General

The 8255 is a Programmable Peripheral Interface (PPI) device designed for use in 8080 Microcomputer Systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the 8080 system bus. The functional configuration of the 8255 is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

### Data Bus Buffer

This 3-state, bi-directional, eight bit buffer is used to interface the 8255 to the 8080 system data bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions by the 8080 CPU. Control Words and Status information are also transferred through the Data Bus buffer.

### Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the 8080 CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

### ($\overline{CS}$)

**Chip Select:** A "low" on this input pin enables the communication between the 8255 and the 8080 CPU.

### ($\overline{RD}$)

**Read:** A "low" on this input pin enables the 8255 to send the Data or Status information to the 8080 CPU on the Data Bus. In essence, it allows the 8080 CPU to "read from" the 8255.

### ($\overline{WR}$)

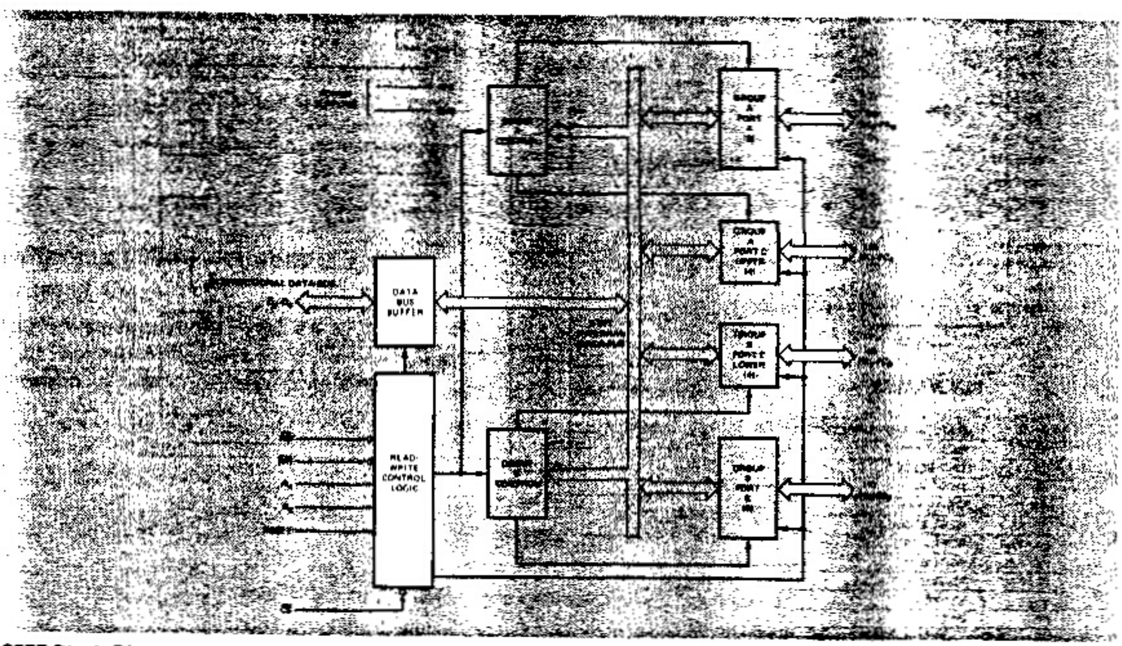
**Write:** A "low" on this input pin enables the 8080 CPU to write Data or Control words into the 8255.

### ($A_0$ and $A_1$)

**Port Select 0 and Port Select 1:** These input signals, in conjunction with the $\overline{RD}$ and $\overline{WR}$ inputs, control the selection of one of the three ports or the Control Word Register. They are normally connected to the least significant bits of the Address Bus ($A_0$ and $A_1$).

## 8255 BASIC OPERATION

| $A_1$ | $A_0$ | $\overline{RD}$ | $\overline{WR}$ | $\overline{CS}$ | INPUT OPERATION (READ) |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | PORT A → DATA BUS |
| 0 | 1 | 0 | 1 | 0 | PORT B → DATA BUS |
| 1 | 0 | 0 | 1 | 0 | PORT C → DATA BUS |
| | | | | | **OUTPUT OPERATION (WRITE)** |
| 0 | 0 | 1 | 0 | 0 | DATA BUS → PORT A |
| 0 | 1 | 1 | 0 | 0 | DATA BUS → PORT B |
| 1 | 0 | 1 | 0 | 0 | DATA BUS → PORT C |
| 1 | 1 | 1 | 0 | 0 | DATA BUS → CONTROL |
| | | | | | **DISABLE FUNCTION** |
| X | X | X | X | 1 | DATA BUS → 3-STATE |
| 1 | 1 | 0 | 1 | 0 | ILLEGAL CONDITION |




**8255 Block Diagram**

### (RESET)

**Reset:** A "high" on this input clears all internal registers including the Control Register and all ports (A, B, C) are set to the input mode.

## Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the 8080 CPU "outputs" a control word to the 8255. The control word contains information such as "mode", "bit set", "bit reset" etc. that initializes the functional configuration of the 8255.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A — Port A and Port C upper (C7-C4)
Control Group B — Port B and Port C lower (C3-C0)

The Control Word Register can Only be written into. No Read operation of the Control Word Register is allowed.

## Ports A, B, and C

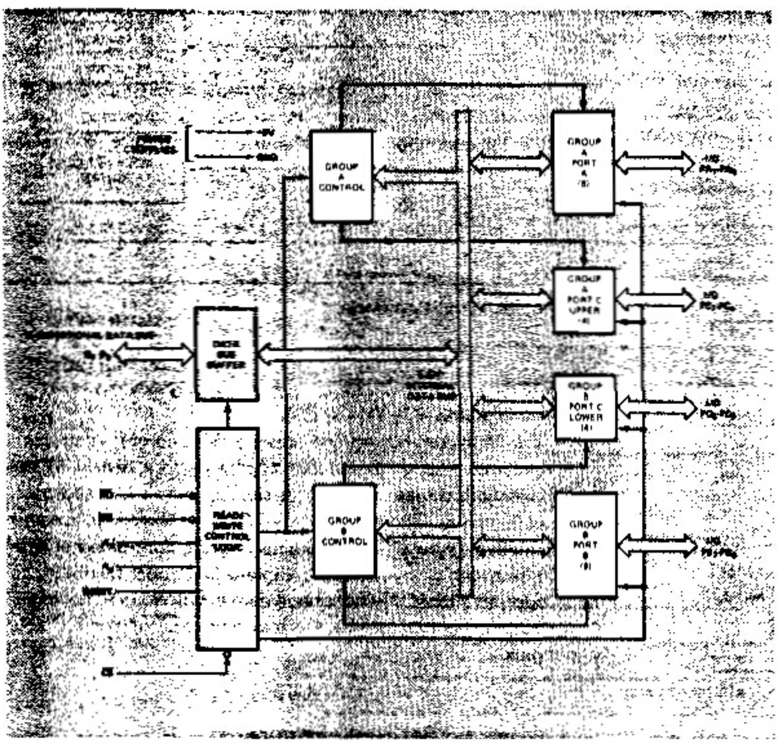
The 8255 contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255.

**Port A:** One 8-bit data output latch/buffer and one 8-bit data input latch.

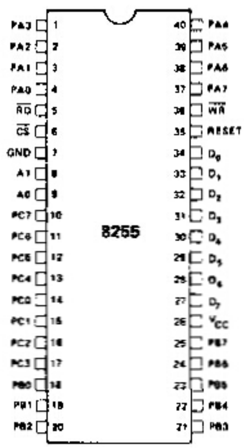
**Port B:** One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

**Port C:** One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with Ports A and B.

8255 BLOCK DIAGRAM




PIN CONFIGURATION




PIN NAMES

| | |
|---|---|
| $D_7$–$D_0$ | DATA BUS (BI-DIRECTIONAL) |
| RESET | RESET INPUT |
| CS | CHIP SELECT |
| RD | READ INPUT |
| WR | WRITE INPUT |
| A0, A1 | PORT ADDRESS |
| PA7-PA0 | PORT A (BIT) |
| PB7-PB0 | PORT B (BIT) |
| PC7-PC0 | PORT C (BIT) |
| Vcc | +5 VOLTS |
| GND | 0 VOLTS |

## 8255 DETAILED OPERATIONAL DESCRIPTION

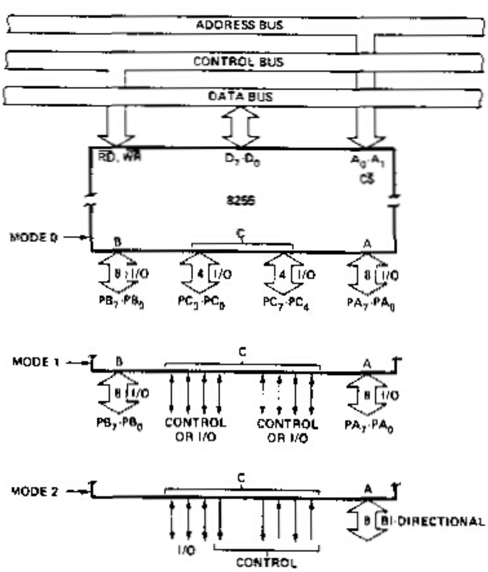
### Mode Selection

There are three basic modes of operation that can be selected by the system software:

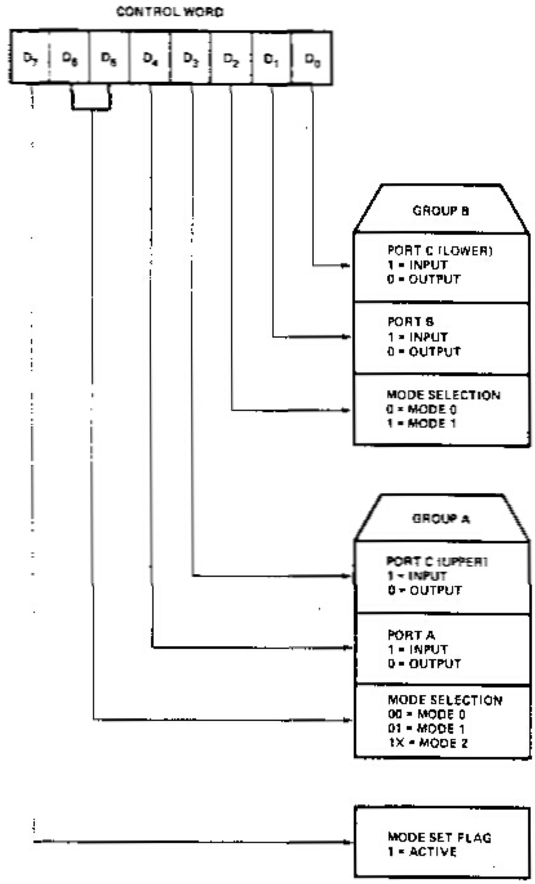
Mode 0 – Basic Input/Output
Mode 1 – Strobed Input/Output
Mode 2 – Bi-Directional Bus

When the RESET input goes "high" all ports will be set to the Input mode (i.e., all 24 lines will be in the high impedance state). After the RESET is removed the 8255 can remain in the Input mode with no additional initialization required. During the execution of the system program any of the other modes may be selected using a single OUTput instruction. This allows a single 8255 to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.




**Basic Mode Definitions and Bus Interface**

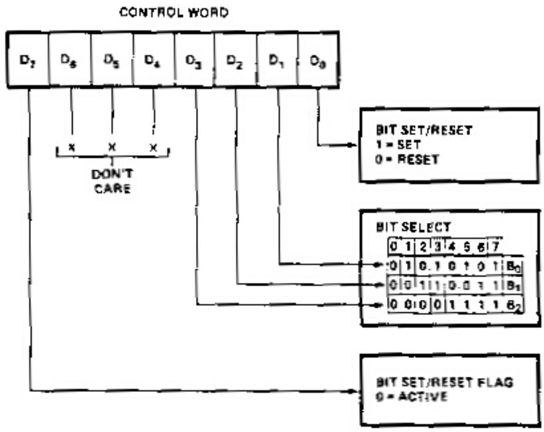



**Mode Definition Format**

The Mode definitions and possible Mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255 has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

### Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.




Bit Set/Reset Format

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

### Interrupt Control Functions

When the 8255 is programmed to operate in Mode 1 or Mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from Port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the Bit set/reset function of Port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without effecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) – INTE is SET – Interrupt enable
(BIT-RESET) – INTE is RESET – Interrupt disable

Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

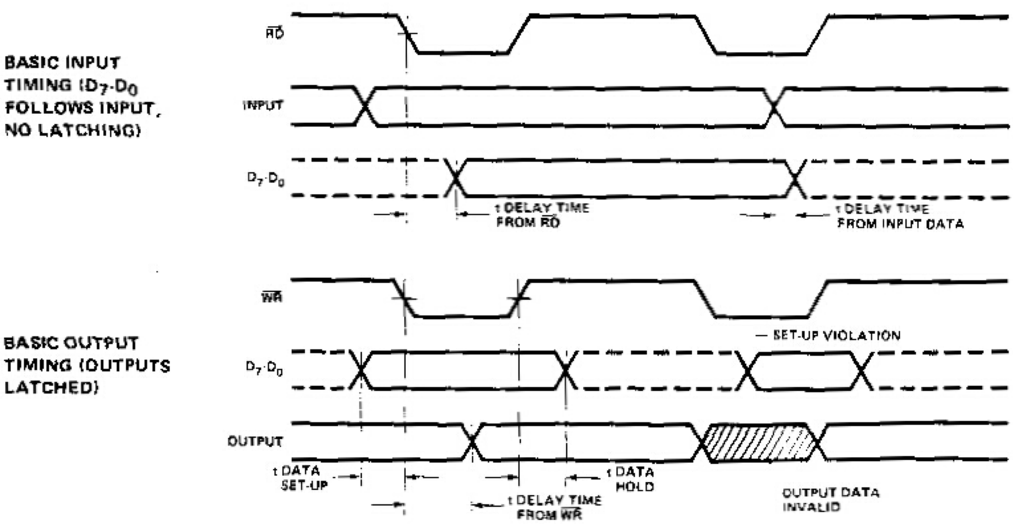
## Operating Modes

### Mode 0 (Basic Input/Output)

This functional configuration provides simple Input and Output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.



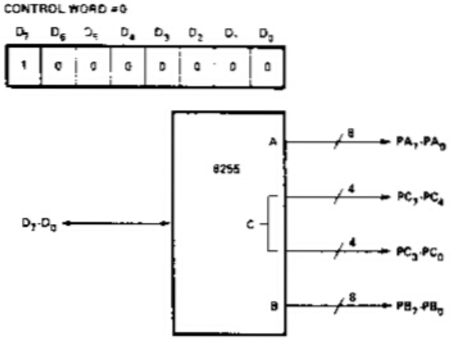

Mode 0 Timing

## MODE 0 PORT DEFINITION CHART

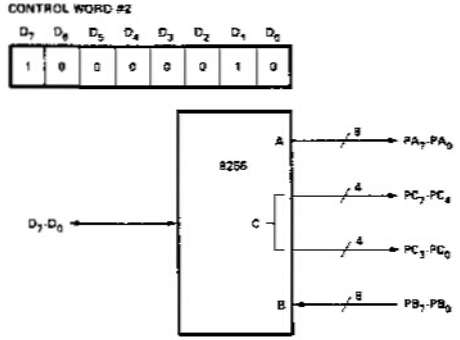
| A | | B | | GROUP A | | | GROUP B | |
|---|---|---|---|---|---|---|---|---|
| $D_4$ | $D_3$ | $D_1$ | $D_0$ | PORT A | PORT C (UPPER) | # | PORT B | PORT C (LOWER) |
| 0 | 0 | 0 | 0 | OUTPUT | OUTPUT | 0 | OUTPUT | OUTPUT |
| 0 | 0 | 0 | 1 | OUTPUT | OUTPUT | 1 | OUTPUT | INPUT |
| 0 | 0 | 1 | 0 | OUTPUT | OUTPUT | 2 | INPUT | OUTPUT |
| 0 | 0 | 1 | 1 | OUTPUT | OUTPUT | 3 | INPUT | INPUT |
| 0 | 1 | 0 | 0 | OUTPUT | INPUT | 4 | OUTPUT | OUTPUT |
| 0 | 1 | 0 | 1 | OUTPUT | INPUT | 5 | OUTPUT | INPUT |
| 0 | 1 | 1 | 0 | OUTPUT | INPUT | 6 | INPUT | OUTPUT |
| 0 | 1 | 1 | 1 | OUTPUT | INPUT | 7 | INPUT | INPUT |
| 1 | 0 | 0 | 0 | INPUT | OUTPUT | 8 | OUTPUT | OUTPUT |
| 1 | 0 | 0 | 1 | INPUT | OUTPUT | 9 | OUTPUT | INPUT |
| 1 | 0 | 1 | 0 | INPUT | OUTPUT | 10 | INPUT | OUTPUT |
| 1 | 0 | 1 | 1 | INPUT | OUTPUT | 11 | INPUT | INPUT |
| 1 | 1 | 0 | 0 | INPUT | INPUT | 12 | OUTPUT | OUTPUT |
| 1 | 1 | 0 | 1 | INPUT | INPUT | 13 | OUTPUT | INPUT |
| 1 | 1 | 1 | 0 | INPUT | INPUT | 14 | INPUT | OUTPUT |
| 1 | 1 | 1 | 1 | INPUT | INPUT | 15 | INPUT | INPUT |

## MODE 0 CONFIGURATIONS

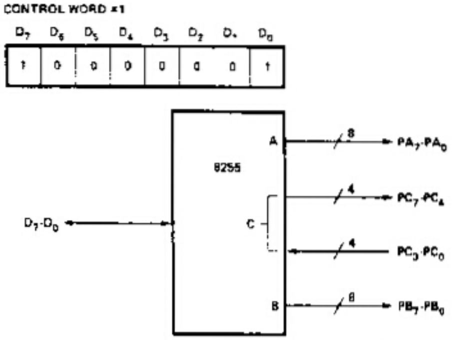
CONTROL WORD #0



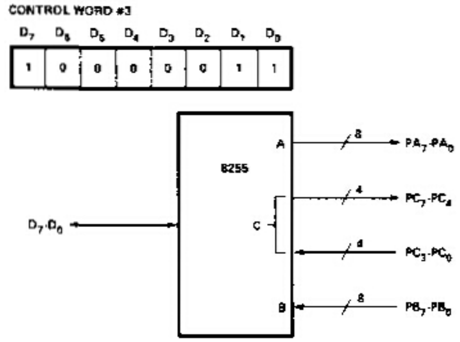

CONTROL WORD #2




CONTROL WORD #1

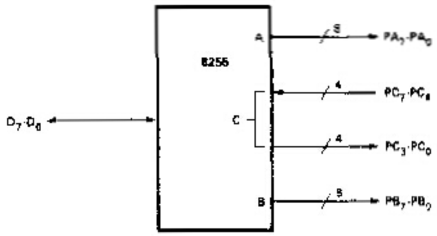



CONTROL WORD #3



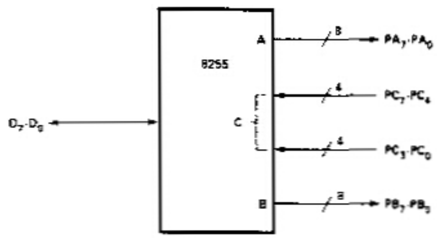
CONTROL WORD #4

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |



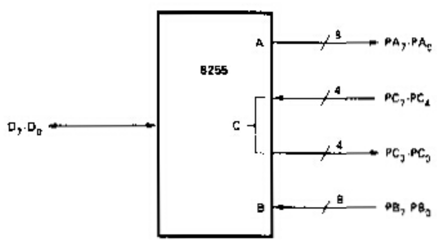

CONTROL WORD #5

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |



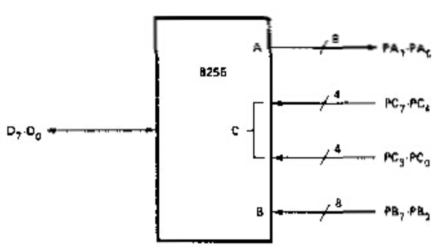

CONTROL WORD #6

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |



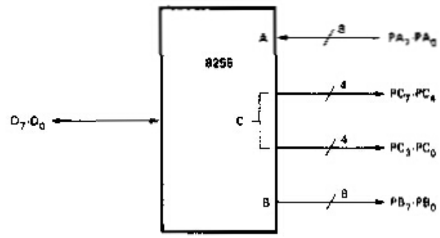

CONTROL WORD #7

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |



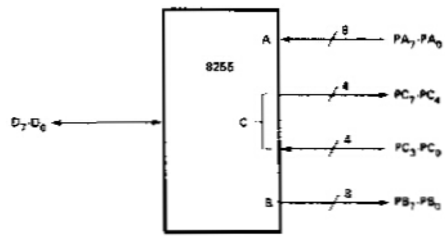

CONTROL WORD #8

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |



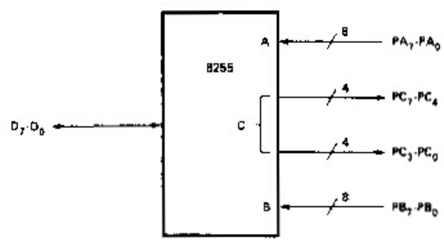

CONTROL WORD #9

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |



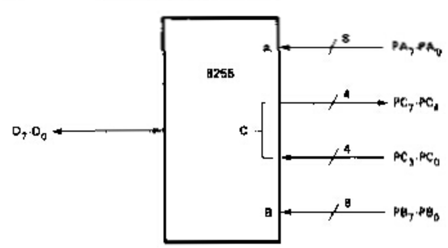

CONTROL WORD #10

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |



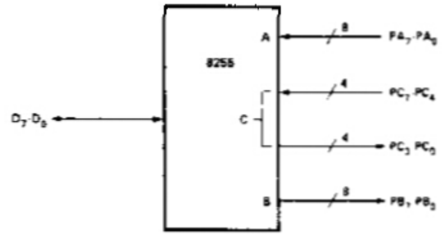

CONTROL WORD #11

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |



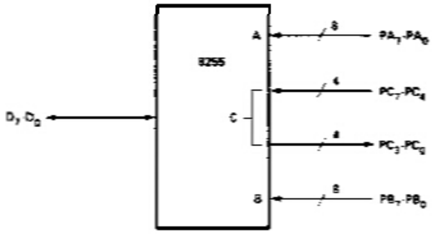
CONTROL WORD #12

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |



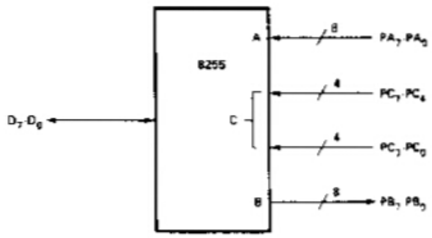

CONTROL WORD #14

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |



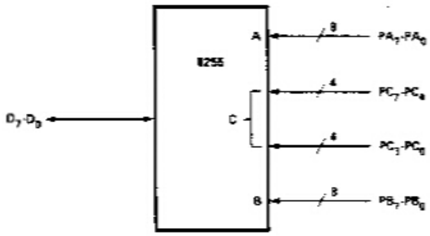

CONTROL WORD #13

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |




CONTROL WORD #15

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |




## Operating Modes

### Mode 1 (Strobed Input/Output)

This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In Mode 1, Port A and Port B use the lines on Port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

## Input Control Signal Definition

**$\overline{STB}$ (Strobe Input)**

A "low" on this input loads data into the input latch.

### IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by the falling edge of the STB input and is reset by the rising edge of the RD input.

### INTR (Interrupt Request)

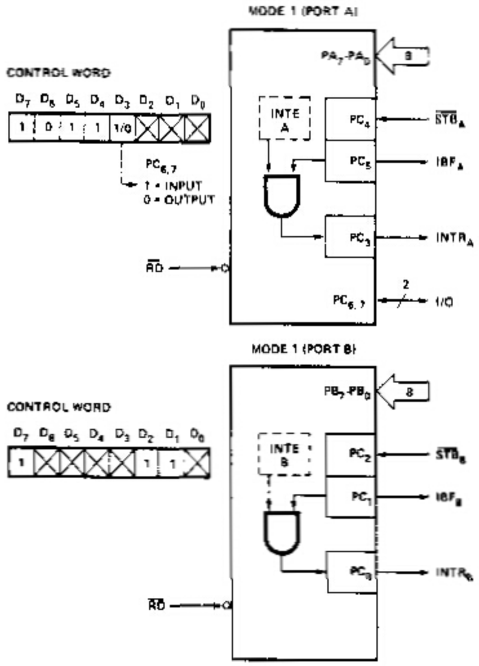
A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the rising edge of $\overline{STB}$ if IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

**INTE A**

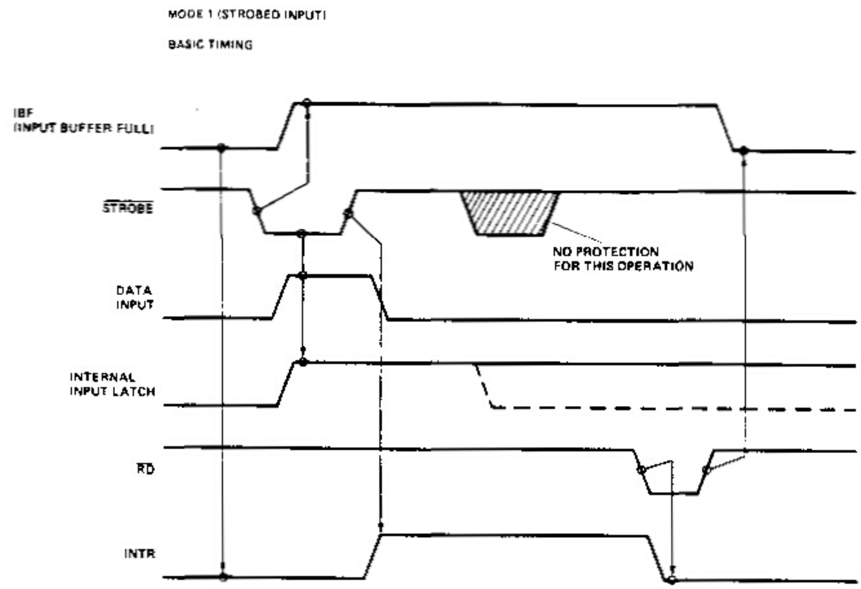
Controlled by bit set/reset of $PC_4$.

**INTE B**

Controlled by bit set/reset of $PC_2$.




**Mode 1 Input**




**Basic Timing Input**

## Output Control Signal Definition

**$\overline{OBF}$ (Output Buffer Full F/F)**

The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to the specified port. The OBF F/F will be set by the rising edge of the WR input and reset by the falling edge of the $\overline{ACK}$ input signal.

**$\overline{ACK}$ (Acknowledge Input)**

A "low" on this input informs the 8255 that the data from Port A or Port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

**INTR (Interrupt Request)**

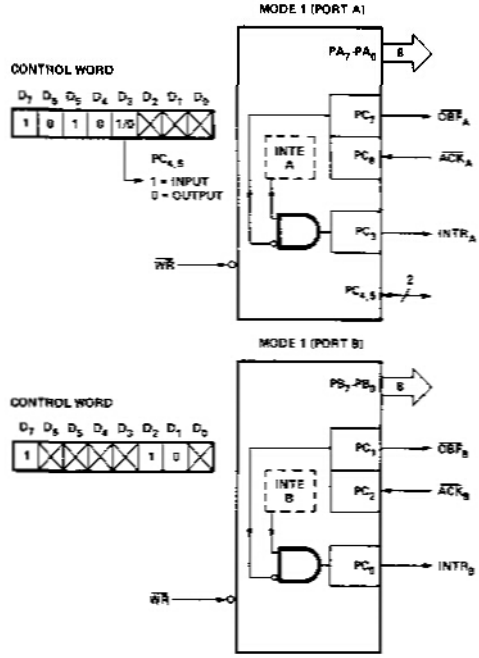
A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set by the rising edge of $\overline{ACK}$ if $\overline{OBF}$ is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{WR}$.

**INTE A**

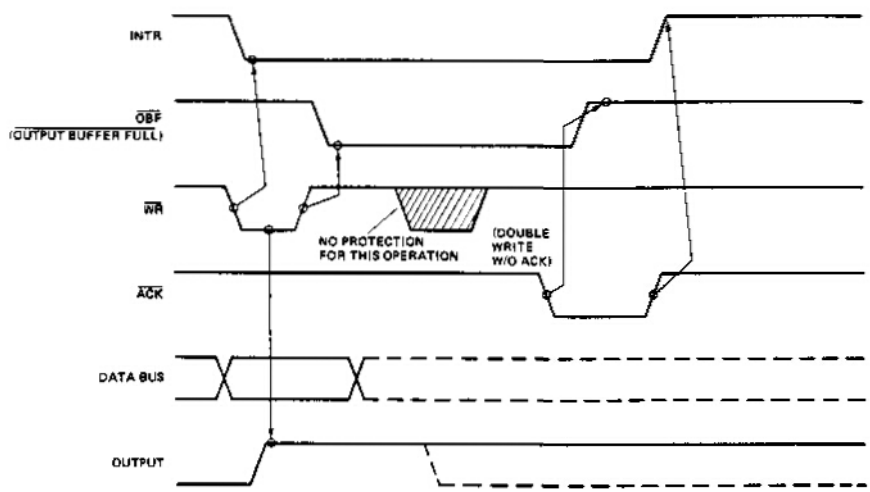
Controlled by bit set/reset of $PC_6$.

**INTE B**

Controlled by bit set/reset of $PC_2$.



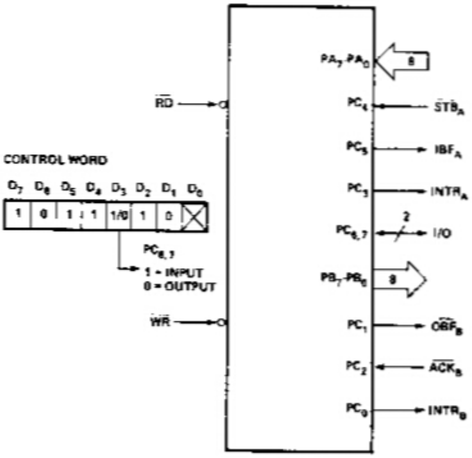

Mode 1 Output



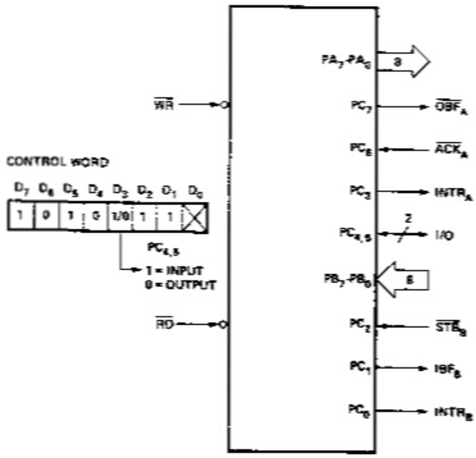

Basic Timing Output

### Combinations of Mode 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.




PORT A – (STROBED INPUT)
PORT B – (STROBED OUTPUT)




PORT A – (STROBED OUTPUT)
PORT B – (STROBED INPUT)

## Operating Modes

**Mode 2 (Strobed Bi-Directional Bus I/O)**

This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bi-directional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to Mode 1. Interrupt generation and enable/disable functions are also available.

Mode 2 Basic Functional Definitions:

- Used in Group A only.
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

### Bi-Directional Bus I/O Control Signal Definition

**INTR (Interrupt Request)**

A high on this output can be used to interrupt the CPU for both input or output operations.

### Output Operations

**$\overline{\text{OBF}}$ (Output Buffer Full)**

The $\overline{\text{OBF}}$ output will go "low" to indicate that the CPU has written data out to Port A.

**$\overline{\text{ACK}}$ (Acknowledge)**

A "low" on this input enables the tri-state output buffer of Port A to send out the data. Otherwise, the output buffer will be in the high-impedance state.

**INTE 1 (The INTE Flip-Flop associated with $\overline{\text{OBF}}$)**

Controlled by bit set/reset of $PC_6$.

### Input Operations

**$\overline{\text{STB}}$ (Strobe Input)**

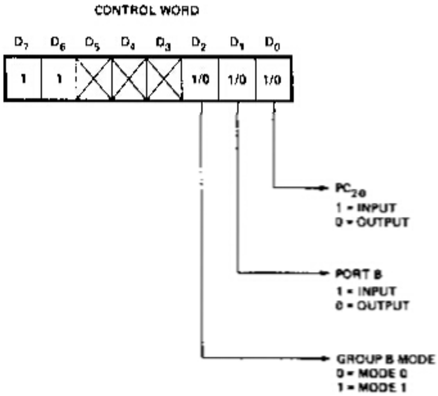
A "low" on this input loads data into the input latch.

**IBF (Input Buffer Full F/F)**

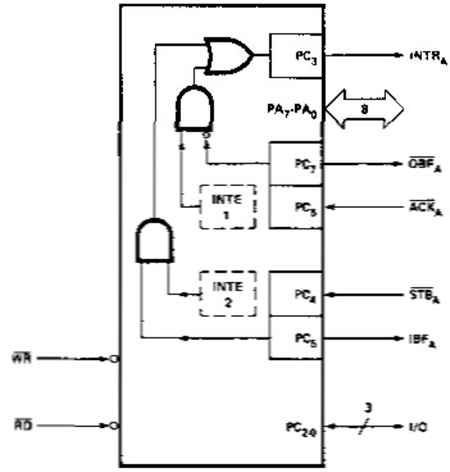
A "high" on this output indicates that data has been loaded into the input latch.

**INTE 2 (The INTE Flip-Flop associated with IBF)**

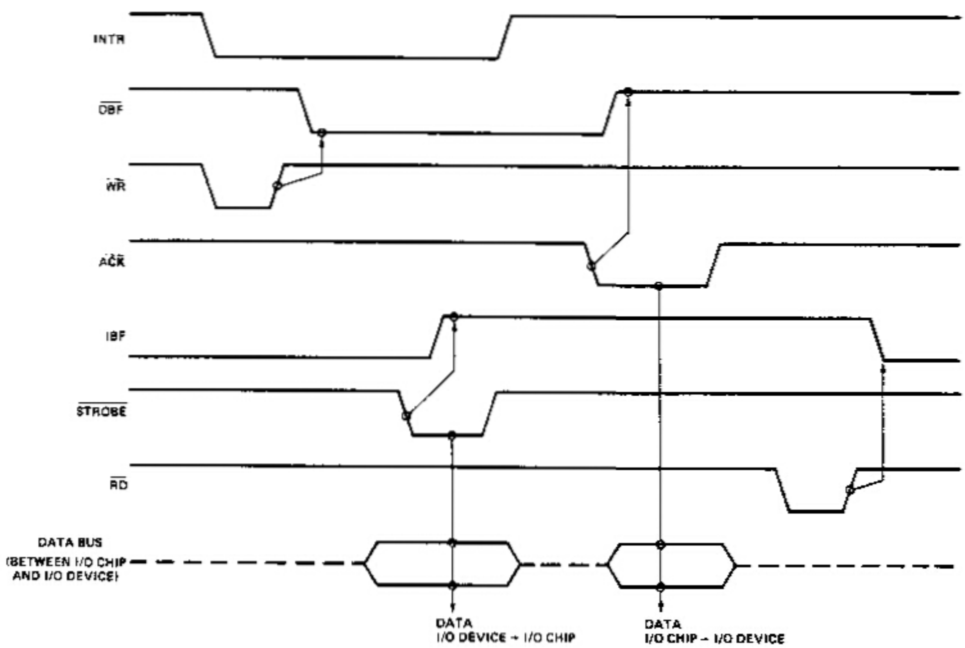
Controlled by bit set/reset of $PC_4$.




Mode 2 Control Word

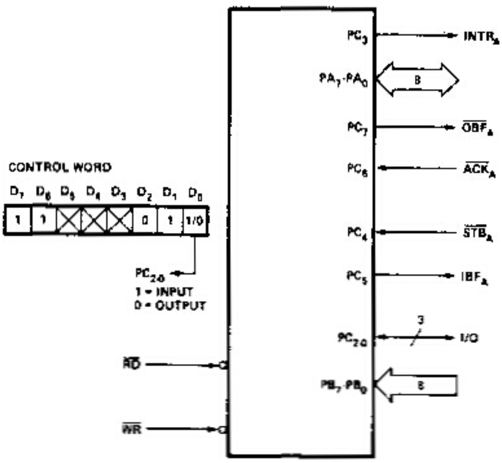



Mode 2

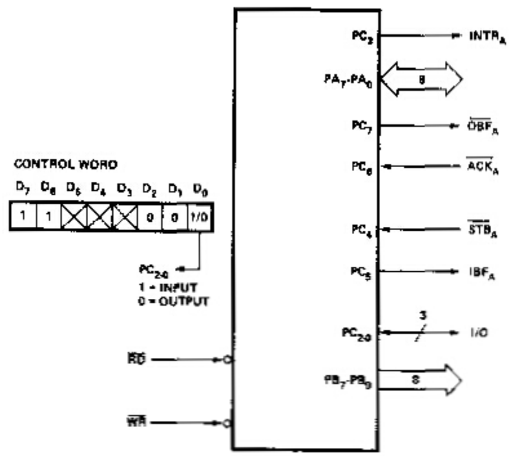



Mode 2 (Bi-directional) Timing

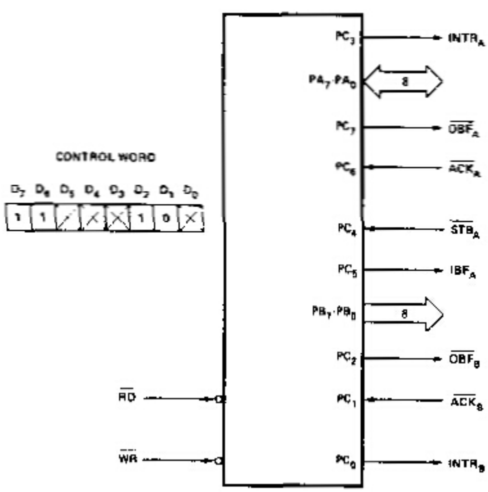
MODE 2 AND MODE 0 (INPUT)



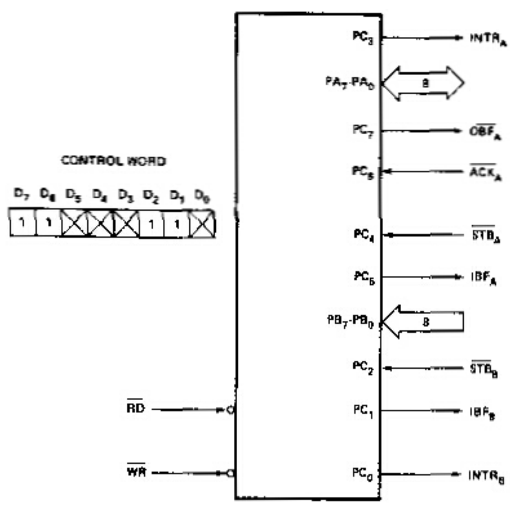

MODE 2 AND MODE 0 (OUTPUT)




MODE 2 AND MODE 1 (OUTPUT)




MODE 2 AND MODE 1 (INPUT)




Mode 2 Combinations

## MODE DEFINITION SUMMARY TABLE

| | MODE 0 | | MODE 1 | | MODE 2 |
|---|---|---|---|---|---|
| | IN | OUT | IN | OUT | GROUP A ONLY |
| $PA_0$ | IN | OUT | IN | OUT | ↔ |
| $PA_1$ | IN | OUT | IN | OUT | ↔ |
| $PA_2$ | IN | OUT | IN | OUT | ↔ |
| $PA_3$ | IN | OUT | IN | OUT | ↔ |
| $PA_4$ | IN | OUT | IN | OUT | ↔ |
| $PA_5$ | IN | OUT | IN | OUT | ↔ |
| $PA_6$ | IN | OUT | IN | OUT | ↔ |
| $PA_7$ | IN | OUT | IN | OUT | ↔ |
| $PB_0$ | IN | OUT | IN | OUT | — (MODE 0 OR MODE 1 ONLY) |
| $PB_1$ | IN | OUT | IN | OUT | — |
| $PB_2$ | IN | OUT | IN | OUT | — |
| $PB_3$ | IN | OUT | IN | OUT | — |
| $PB_4$ | IN | OUT | IN | OUT | — |
| $PB_5$ | IN | OUT | IN | OUT | — |
| $PB_6$ | IN | OUT | IN | OUT | — |
| $PB_7$ | IN | OUT | IN | OUT | — |
| $PC_0$ | IN | OUT | $INTR_B$ | $INTR_B$ | I/O |
| $PC_1$ | IN | OUT | $IBF_B$ | $\overline{OBF}_B$ | I/O |
| $PC_2$ | IN | OUT | $\overline{STB}_B$ | $\overline{ACK}_B$ | I/O |
| $PC_3$ | IN | OUT | $INTR_A$ | $INTR_A$ | $INTR_A$ |
| $PC_4$ | IN | OUT | $\overline{STB}_A$ | I/O | $\overline{STB}_A$ |
| $PC_5$ | IN | OUT | $IBF_A$ | I/O | $IBF_A$ |
| $PC_6$ | IN | OUT | I/O | $\overline{ACK}_A$ | $\overline{ACK}_A$ |
| $PC_7$ | IN | OUT | I/O | $\overline{OBF}_A$ | $\overline{OBF}_A$ |

### Special Mode Combination Considerations

There are several combinations of modes when not all of the bits in Port C are used for control or status. The remaining bits can be used as follows:

If Programmed as Inputs –
All input lines can be accessed during a normal Port C read.

If Programmed as Outputs –
Bits in C upper ($PC_7$-$PC_4$) must be individually accessed using the bit set/reset function.

Bits in C lower ($PC_3$-$PC_0$) can be accessed using the bit set/reset function or accessed as a threesome by writing into Port C.

### Source Current Capability on Port B and Port C

Any set of eight output buffers, selected randomly from Ports B and C can source 1mA at 1.5 volts. This feature allows the 8255 to directly drive Darlington type drivers and high-voltage displays that require such source current.

### Reading Port C Status

In Mode 0, Port C transfers data to or from the peripheral device. When the 8255 is programmed to function in Modes 1 or 2, Port C generates or accepts "hand-shaking" signals with the peripheral device. Reading the contents of Port C allows the programmer to test or verify the "status" of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.

INPUT CONFIGURATION

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| I/O | I/O | $IBF_A$ | $INTE_A$ | $INTR_A$ | $INTE_B$ | $IBF_B$ | $INTR_B$ |
| GROUP A | | | | | GROUP B | | |

OUTPUT CONFIGURATION

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| $\overline{OBF}_A$ | $INTE_A$ | I/O | I/O | $INTR_A$ | $INTE_B$ | $\overline{OBF}_B$ | $INTR_B$ |
| GROUP A | | | | | GROUP B | | |

**Mode 1 Status Word Format**

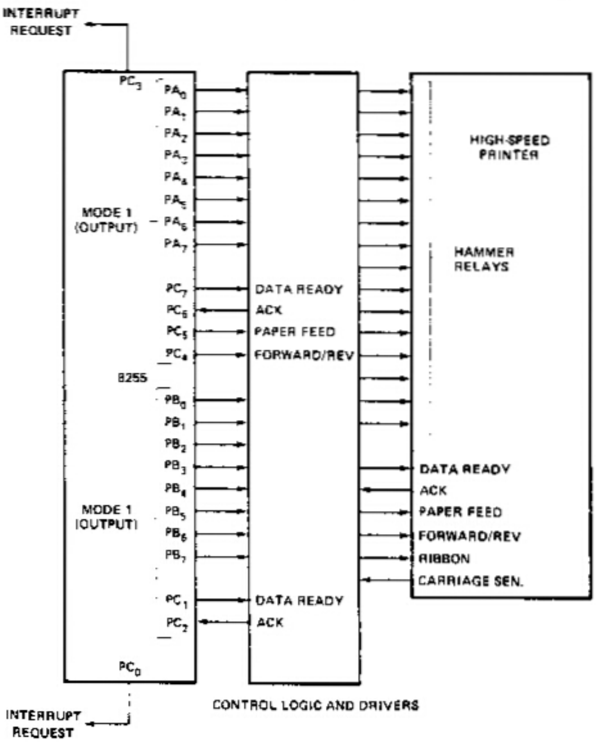
| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| $\overline{OBF}_A$ | $INTE_1$ | $IBF_A$ | $INTE_2$ | $INTR_A$ | X | X | X |
| GROUP A | | | | | GROUP B (DEFINED BY MODE 0 OR MODE 1 SELECTION) | | |

**Mode 2 Status Word Format**

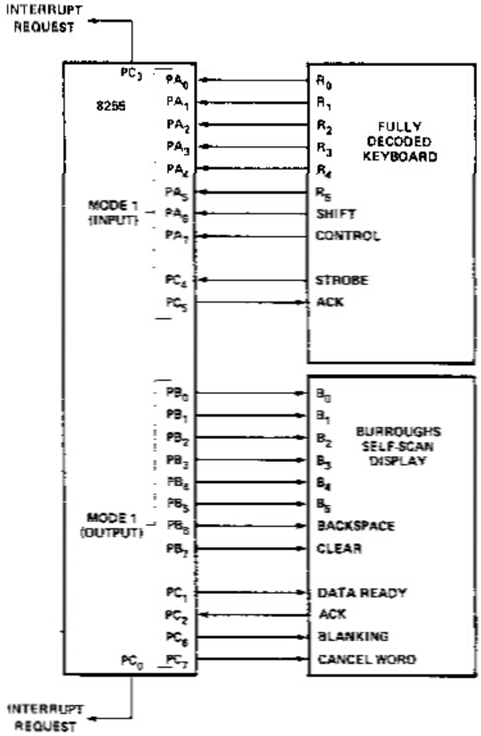
## APPLICATIONS OF THE 8255

The 8255 is a very powerful tool for interfacing peripheral equipment to the 8080 microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

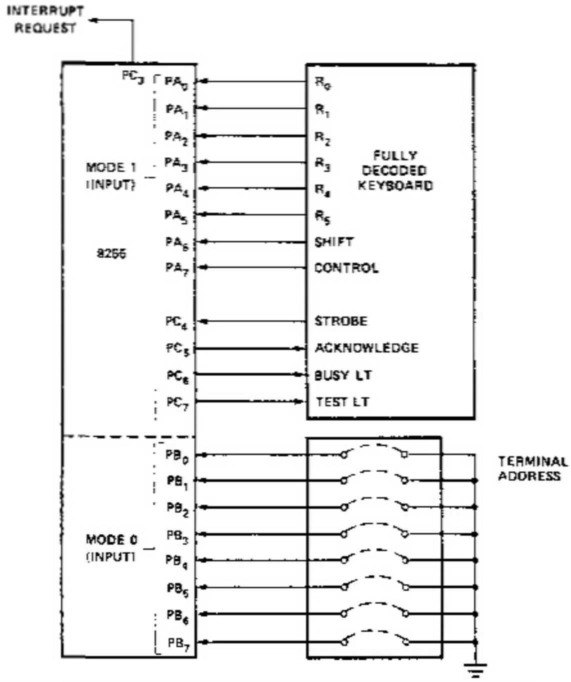
Each peripheral device in a Microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255 is programmed by the I/O service routine and becomes an extension of the systems software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the Detailed Operational Description, a control word can easily be developed to initialize the 8255 to exactly "fit" the application. Here are a few examples of typical applications of the 8255.



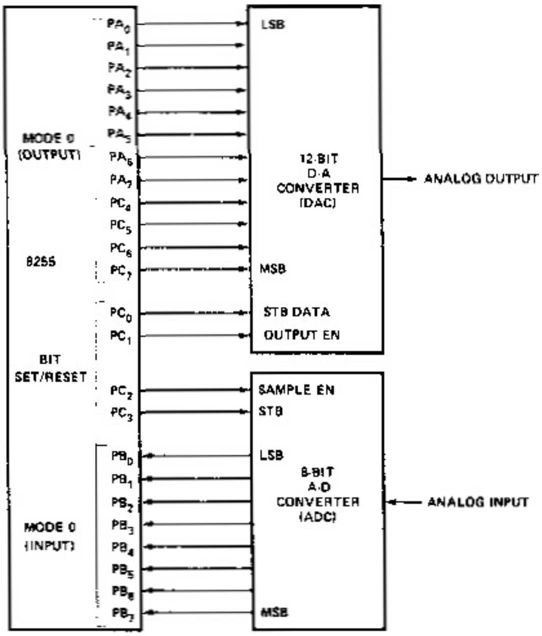

Printer Interface



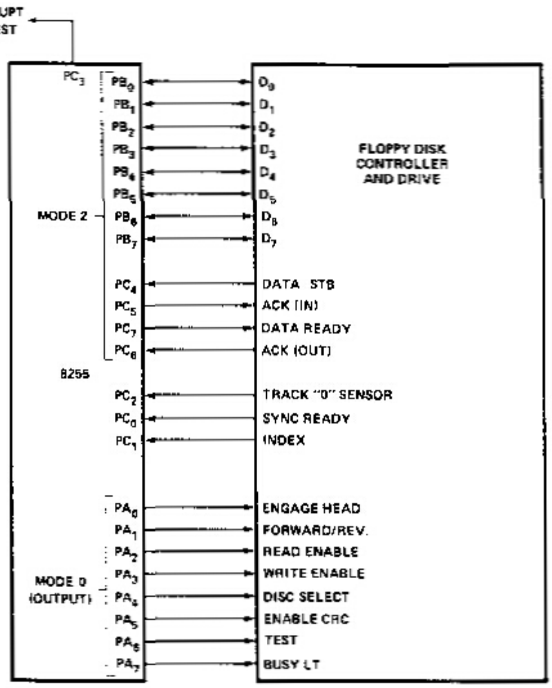

Keyboard and Display Interface



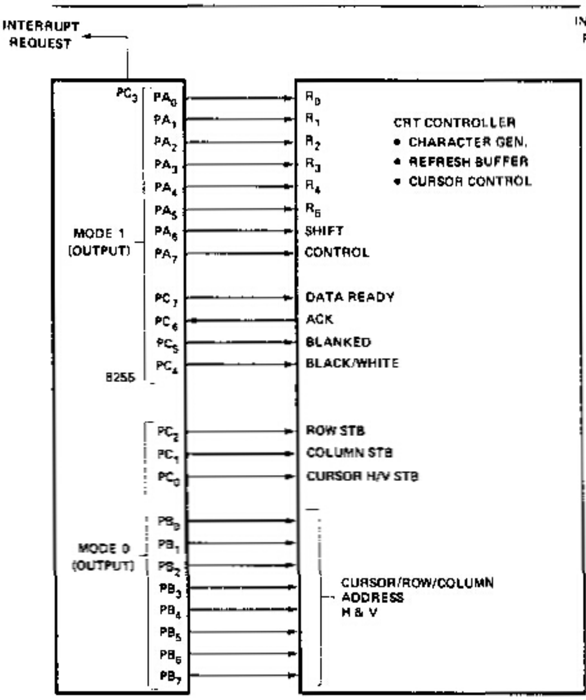

Keyboard and Terminal Address Interface



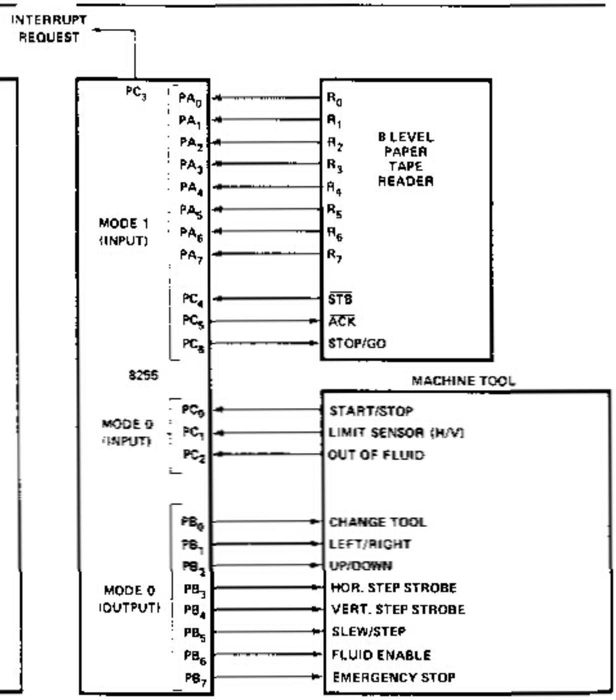

Digital to Analog, Analog to Digital



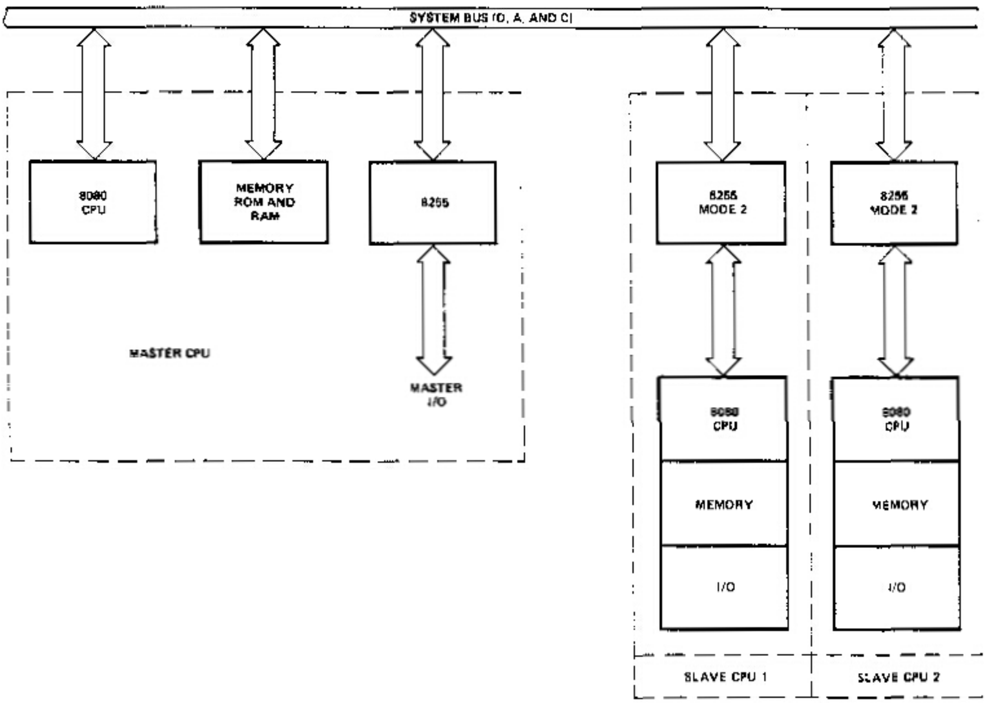

Basic Floppy Disc Interface




Basic CRT Controller Interface




Machine Tool Controller Interface




**Distributed Intelligence Multi-Processor Interface**

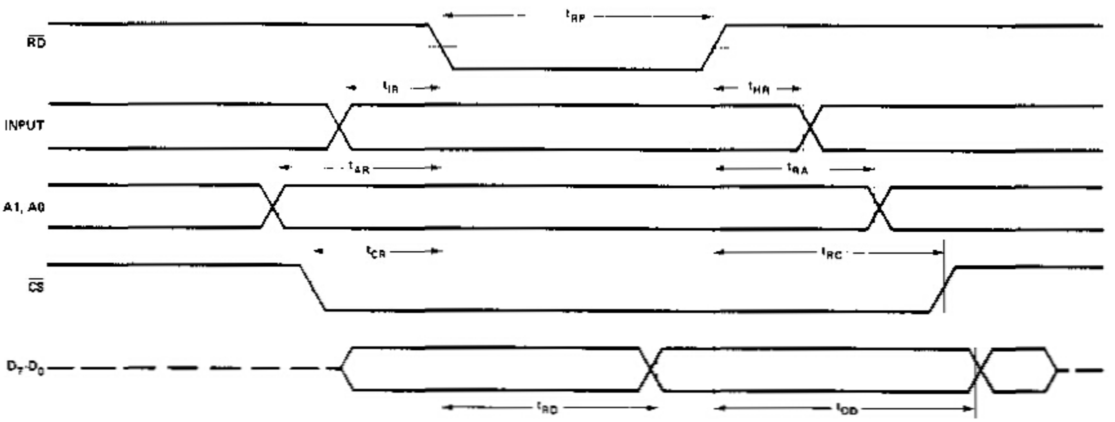
## D.C. CHARACTERISTICS $T_A = 0°C$ to $70°C$; $V_{CC} = +5V \pm 5\%$; $V_{SS} = 0V$

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $V_{IL}$ | Input Low Voltage | | | .8 | V | |
| $V_{IH}$ | Input High Voltage | 2.0 | | | V | |
| $V_{OL}$ | Output Low Voltage | | | .4 | V | $I_{OL} = 1.6mA$ |
| $V_{OH}$ | Output High Voltage | 2.4 | | | V | $I_{OH} = -50\mu A$ ($-100\mu A$ for D.B. Port) |
| $I_{OH}$[1] | Darlington Drive Current | | 2.0 | | mA | $V_{OH} = 1.5V$, $R_{EXT} = 390\Omega$ |
| $I_{CC}$ | Power Supply Current | | 40 | | mA | |

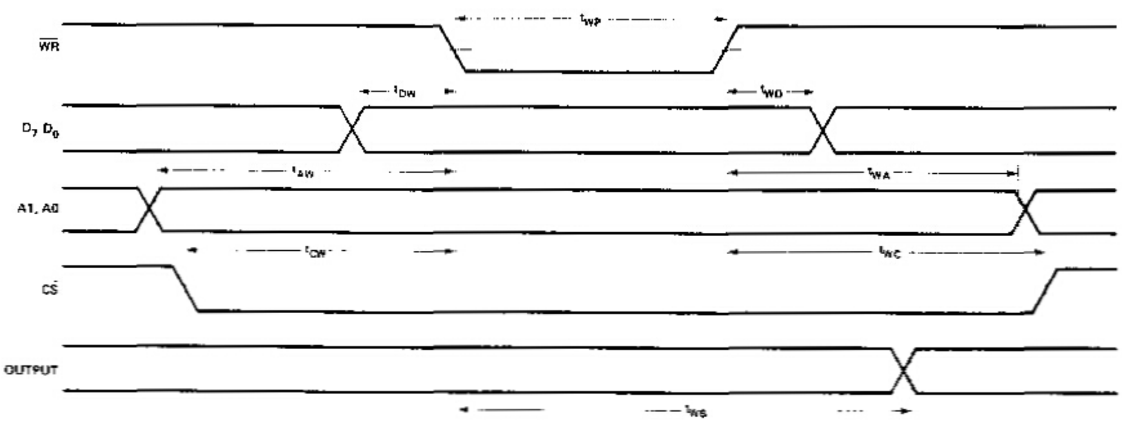
NOTE:
1. Available on 8 pins only.

## A.C. CHARACTERISTICS $T_A = 0°C$ to $70°C$; $V_{CC} = +5V \pm 5\%$; $V_{SS} = 0V$

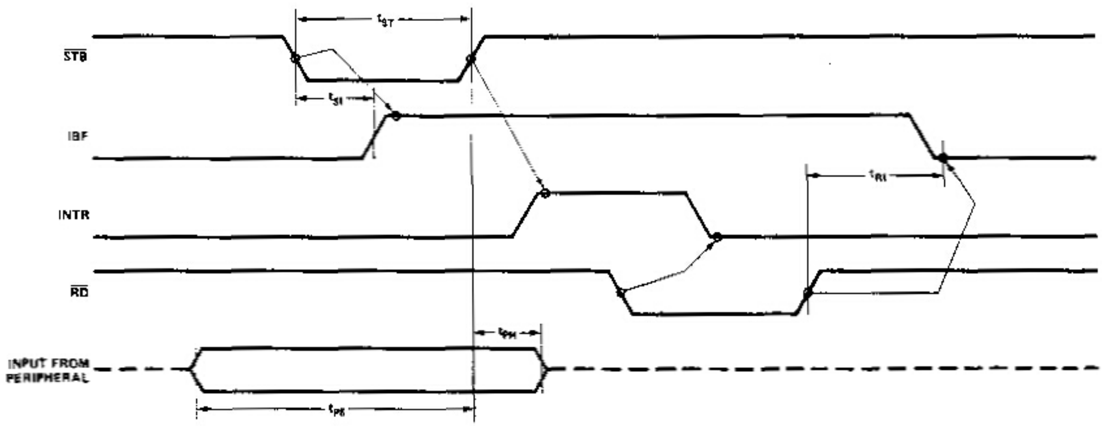
| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Condition |
|---|---|---|---|---|---|---|
| $t_{WP}$ | Pulse Width of $\overline{WR}$ | | | 430 | ns | |
| $t_{DW}$ | Time D.B. Stable Before $\overline{WR}$ | 10 | | | ns | |
| $t_{WD}$ | Time D.B. Stable After $\overline{WR}$ | 65 | | | ns | |
| $t_{AW}$ | Time Address Stable Before $\overline{WR}$ | 20 | | | ns | |
| $t_{WA}$ | Time Address Stable After $\overline{WR}$ | 35 | | | ns | |
| $t_{CW}$ | Time CS Stable Before $\overline{WR}$ | 20 | | | ns | |
| $t_{WC}$ | Time CS Stable After $\overline{WR}$ | 35 | | | ns | |
| $t_{WB}$ | Delay From $\overline{WR}$ To Output | | | 500 | ns | |
| $t_{RP}$ | Pulse Width of $\overline{RD}$ | 430 | | | ns | |
| $t_{IR}$ | $\overline{RD}$ Set-Up Time | 50 | | | ns | |
| $t_{HR}$ | Input Hold Time | 50 | | | ns | |
| $t_{RD}$ | Delay From $\overline{RD}$ = 0 To System Bus | 350 | | | ns | |
| $t_{OD}$ | Delay From $\overline{RD}$ = 1 To System Bus | 150 | | | ns | |
| $t_{AR}$ | Time Address Stable Before $\overline{RD}$ | 50 | | | ns | |
| $t_{CR}$ | Time $\overline{CS}$ Stable Before $\overline{RD}$ | 50 | | | ns | |
| $t_{AK}$ | Width Of $\overline{ACK}$ Pulse | 500 | | | ns | |
| $t_{ST}$ | Width Of $\overline{STB}$ Pulse | 350 | | | ns | |
| $t_{PS}$ | Set-Up Time For Peripheral | 150 | | | ns | |
| $t_{PH}$ | Hold Time For Peripheral | 150 | | | ns | |
| $t_{RA}$ | Hold Time for $A_1$, $A_0$ After $\overline{RD}$ = 1 | 379 | | | ns | |
| $t_{RC}$ | Hold Time For CS After $\overline{RD}$ = 1 | 5 | | | ns | |
| $t_{AD}$ | Time From $\overline{ACK}$ = 0 To Output (Mode 2) | | | 500 | ns | |
| $t_{KD}$ | Time From $\overline{ACK}$ = 1 To Output Floating | | | 300 | ns | |
| $t_{WO}$ | Time From $\overline{WR}$ = 1 To $\overline{OBF}$ = 0 | | | 300 | ns | |
| $t_{AO}$ | Time From $\overline{ACK}$ = 0 To $\overline{OBF}$ = 1 | | | 500 | ns | |
| $t_{SI}$ | Time From $\overline{STB}$ = 0 To IBF | | | 600 | ns | |
| $t_{RI}$ | Time From $\overline{RD}$ = 1 To IBF = 0 | | | 300 | ns | |



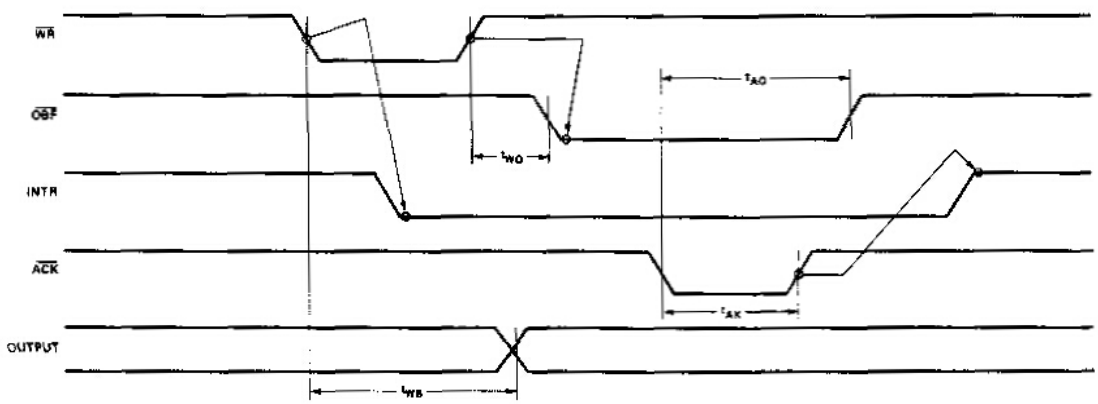

Mode 0 (Basic Input)



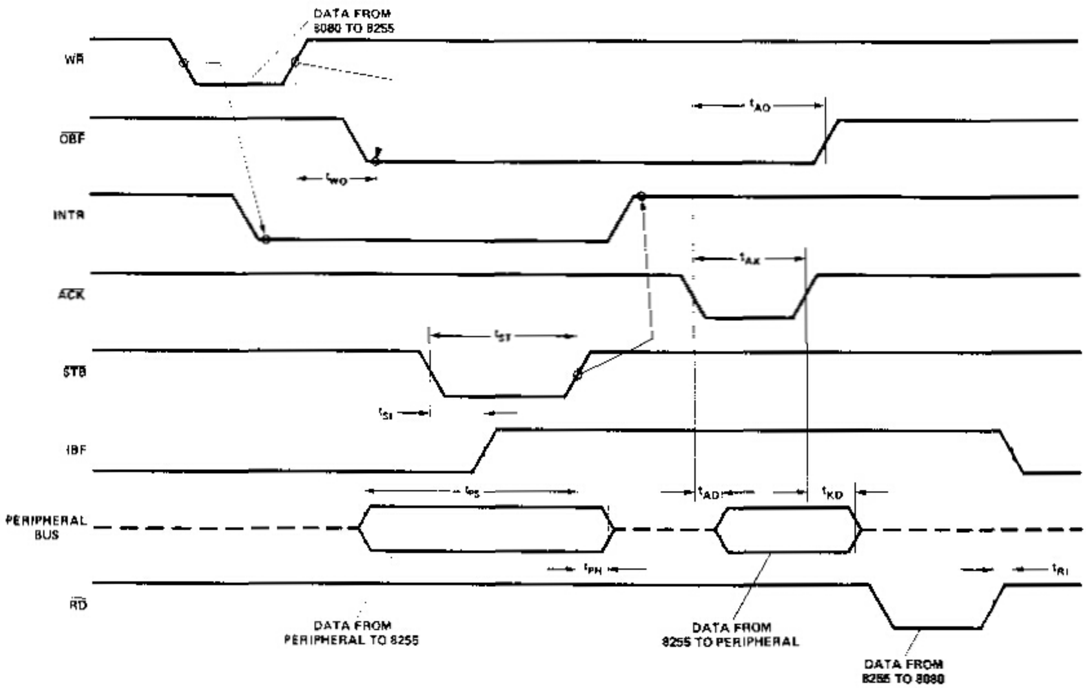

Mode 0 (Basic Output)




Mode 1 (Strobed Input)




Mode 1 (Strobed Output)




Mode 2 (Bi-directional)

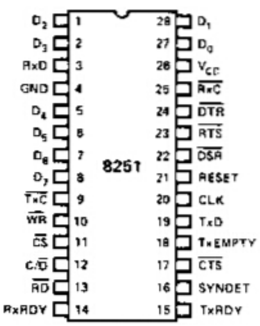
intel® **Silicon Gate MOS 8251**

# PROGRAMMABLE COMMUNICATION INTERFACE

- ***Synchronous and Asynchronous* Operation**
  - **Synchronous:**
    **5-8 Bit Characters**
    **Internal or External Character Synchronization**
    **Automatic Sync Insertion**
  - **Asynchronous:**
    **5-8 Bit Characters**
    **Clock Rate — 1, 16 or 64 Times Baud Rate**
    **Break Character Generation**
    **1, 1½, or 2 Stop Bits**
    **False Start Bit Detection**
- ***Baud Rate* — *DC to 56k Baud* (Sync Mode)**
  **DC to 9.6k Baud (Async Mode)**
- **Full Duplex, Double Buffered, Transmitter and Receiver**
- **Error Detection — Parity, Overrun, 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 8251 is a Universal Synchronous/Asynchronous Receiver / Transmitter (USART) Chip designed for data communications in microcomputer systems. The USART is used as a peripheral device and is programmed by the CPU to operate using virtually any serial data transmission technique presently in use (including IBM Bi-Sync). The USART accepts data characters from the CPU 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 CPU. The USART will signal the CPU whenever it can accept a new character for transmission or whenever it has received a character for the CPU. The CPU can read the complete status of the USART at any time. These include data transmission errors and control signals such as SYNDET, TxEMPT. The chip is constructed using N-channel silicon gate technology.

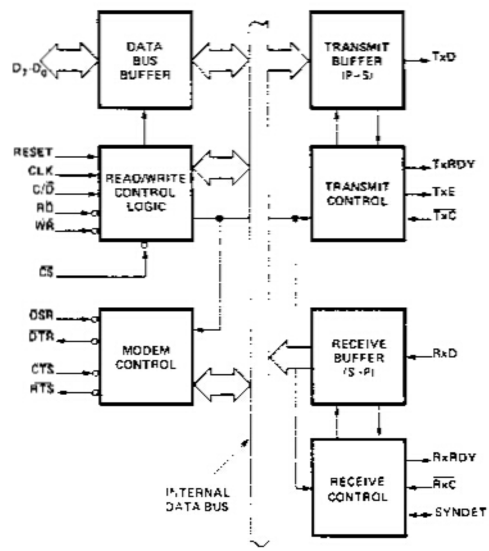
**PIN CONFIGURATION**




| Pin Name | Pin Function |
|---|---|
| $D_7$-$D_0$ | Data Bus (8 bits) |
| C/D̅ | Control or Data is to be Written or Read |
| R̅D̅ | Read Data Command |
| W̅R̅ | Write Data or Control Command |
| C̅S̅ | Chip Enable |
| CLK | Clock Pulse (TTL) |
| RESET | Reset |
| T̅x̅C̅ | Transmitter Clock |
| TxD | Transmitter Data |
| R̅x̅C̅ | Receiver Clock |
| RxD | Receiver Data |
| RxRDY | Receiver Ready (has character for 8080) |
| TxRDY | Transmitter Ready (ready for char. from 8080) |

| Pin Name | Pin Function |
|---|---|
| D̅S̅R̅ | Data Set Ready |
| D̅T̅R̅ | Data Terminal Ready |
| SYNDET | Sync Detect |
| R̅T̅S̅ | Request to Send Data |
| C̅T̅S̅ | Clear to Send Data |
| TxE | Transmitter Empty |
| Vcc | +5 Volt Supply |
| GND | Ground |

**BLOCK DIAGRAM**



## 8251 BASIC FUNCTIONAL DESCRIPTION

### General

The 8251 is a Universal Synchronous/Asynchronous Receiver/Transmitter designed specifically for the 8080 Microcomputer System. Like other I/O devices in the 8080 Microcomputer System its functional configuration is programmed by the systems software for maximum flexibility. The 8251 can support virtually any serial data technique currently in use (including IBM "bi-sync").

In a communication environment an interface device must convert parallel format system data into serial format for transmission and convert incoming serial format data into parallel system data for reception. The interface device must also delete or insert bits or characters that are functionally unique to the communication technique. In essence, the interface should appear "transparent" to the CPU, a simple input or output of byte-oriented system data.

### Data Bus Buffer

This 3-state, bi-directional, 8-bit buffer is used to interface the 8251 to the 8080 system Data Bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions of the 8080 CPU. Control words, Command words and Status information are also transferred through the Data Bus Buffer.

### Read/Write Control Logic

This functional block accepts inputs from the 8080 Control bus and generates control signals for overall device operation. It contains the Control Word Register and Command Word Register that store the various control formats for device functional definition.

### RESET (Reset)

A "high" on this input forces the 8251 into an "Idle" mode. The device will remain at "Idle" until a new set of control words is written into the 8251 to program its functional definition.

### CLK (Clock)

The CLK input is used to generate internal device timing and is normally connected to the Phase 2 (TTL) output of the 8224 Clock Generator. No external inputs or outputs are referenced to CLK but the frequency of CLK must be greater than 30 times the Receiver or Transmitter clock inputs for synchronous mode (4.5 times for asynchronous mode).

### $\overline{WR}$ (Write)

A "low" on this input informs the 8251 that the CPU is outputting data or control words, in essence, the CPU is writing out to the 8251.

### $\overline{RD}$ (Read)

A "low" on this input informs the 8251 that the CPU is inputting data or status information, in essence, the CPU is reading from the 8251.

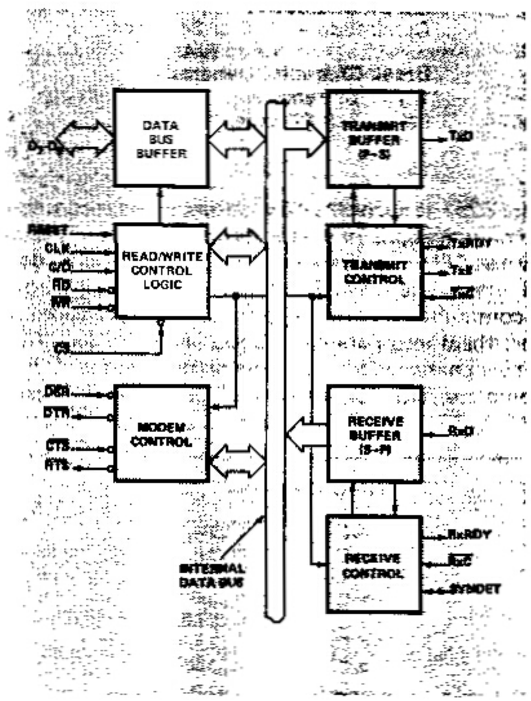
### C/$\overline{D}$ (Control/Data)

This input, in conjunction with the $\overline{WR}$ and $\overline{RD}$ inputs informs the 8251 that the word on the Data Bus is either a data character, control word or status information.

1 = CONTROL  0 = DATA

### $\overline{CS}$ (Chip Select)

A "low" on this input enables the 8251. No reading or writing will occur unless the device is selected.




| C/$\overline{D}$ | $\overline{RD}$ | $\overline{WR}$ | $\overline{CS}$ | |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 8251 → DATA BUS |
| 0 | 1 | 0 | 0 | DATA BUS → 8251 |
| 1 | 0 | 1 | 0 | STATUS → DATA BUS |
| 1 | 1 | 0 | 0 | DATA BUS → CONTROL |
| X | X | X | 1 | DATA BUS → 3-STATE |

### Modem Control

The 8251 has a set of control inputs and outputs that can be used to simplify the interface to almost any Modem. The modem control signals are general purpose in nature and can be used for functions other than Modem control, if necessary.

### $\overline{\text{DSR}}$ (Data Set Ready)

The $\overline{\text{DSR}}$ input signal is general purpose in nature. Its condition can be tested by the CPU using a Status Read operation. The $\overline{\text{DSR}}$ input is normally used to test Modem conditions such as Data Set Ready.

### $\overline{\text{DTR}}$ (Data Terminal Ready)

The $\overline{\text{DTR}}$ output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{\text{DTR}}$ output signal is normally used for Modem control such as Data Terminal Ready or Rate Select.

### $\overline{\text{RTS}}$ (Request to Send)

The $\overline{\text{RTS}}$ output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{\text{RTS}}$ output signal is normally used for Modem control such as Request to Send.

### $\overline{\text{CTS}}$ (Clear to Send)

A "low" on this input enables the 8251 to transmit data (serial) if the Tx EN bit in the Command byte is set to a "one."

### Transmitter Buffer

The Transmitter Buffer accepts parallel data from the Data Bus Buffer, converts it to a serial bit stream, inserts the appropriate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the TxD output pin.

### Transmitter Control

The Transmitter Control manages all activities associated with the transmission of serial data. It accepts and issues signals both externally and internally to accomplish this function.

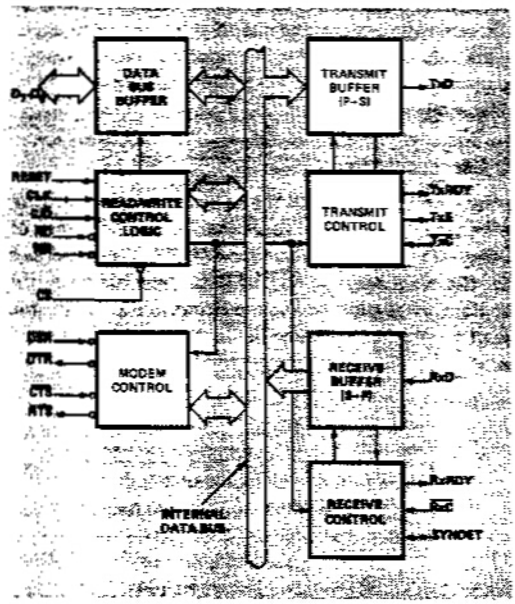
### TxRDY (Transmitter Ready)

This output signals the CPU that the transmitter is ready to accept a data character. It can be used as an interrupt to the system or for the Polled operation the CPU can check TxRDY using a status read operation. TxRDY is automatically reset when a character is loaded from the CPU.

### TxE (Transmitter Empty)

When the 8251 has no characters to transmit, the TxE output will go "high". It resets automatically upon receiving a character from the CPU. TxE can be used to indicate the end of a transmission mode, so that the CPU "knows" when to "turn the line around" in the half-duplexed operational mode.

In SYNChronous mode, a "high" on this output indicates that a character has not been loaded and the SYNC character or characters are about to be transmitted automatically as "fillers".




### $\overline{\text{TxC}}$ (Transmitter Clock)

The Transmitter Clock controls the rate at which the character is to be transmitted. In the Synchronous transmission mode, the frequency of $\overline{\text{TxC}}$ is equal to the actual Baud Rate (1X). In Asynchronous transmission mode, the frequency of $\overline{\text{TxC}}$ is a multiple of the actual Baud Rate. A portion of the mode instruction selects the value of the multiplier; it can be 1x, 16x or 64x the Baud Rate.

For Example:

If Baud Rate equals 110 Baud,
$\overline{\text{TxC}}$ equals 110 Hz (1x)
$\overline{\text{TxC}}$ equals 1.76 kHz (16x)
$\overline{\text{TxC}}$ equals 7.04 kHz (64x).
If Baud Rate equals 9600 Baud,
$\overline{\text{TxC}}$ equals 614.4 kHz (64x).

The falling edge of $\overline{\text{TxC}}$ shifts the serial data out of the 8251.

## Mode Instruction Definition

The 8251 can be used for either Asynchronous or Synchronous data communication. To understand how the Mode Instruction defines the functional operation of the 8251 the designer can best view the device as two separate components sharing the same package. One Asynchronous the other Synchronous. The format definition can be changed "on the fly" but for explanation purposes the two formats will be isolated.

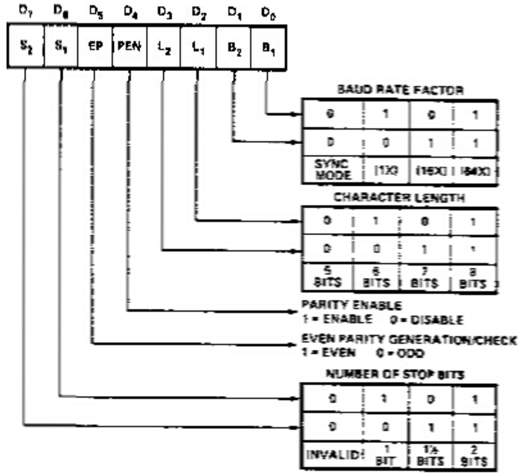
## Asynchronous Mode (Transmission)

Whenever a data character is sent by the CPU the 8251 automatically adds a Start bit (low level) and the programmed number of Stop bits to each character. Also, an even or odd Parity bit is inserted prior to the Stop bit(s), as defined by the Mode Instruction. The character is then transmitted as a serial data stream on the TxD output. The serial data is shifted out on the falling edge of $\overline{TxC}$ at a rate equal to 1, 1/16, or 1/64 that of the $\overline{TxC}$, as defined by the Mode Instruction. BREAK characters can be continuously sent to the TxD if commanded to do so.

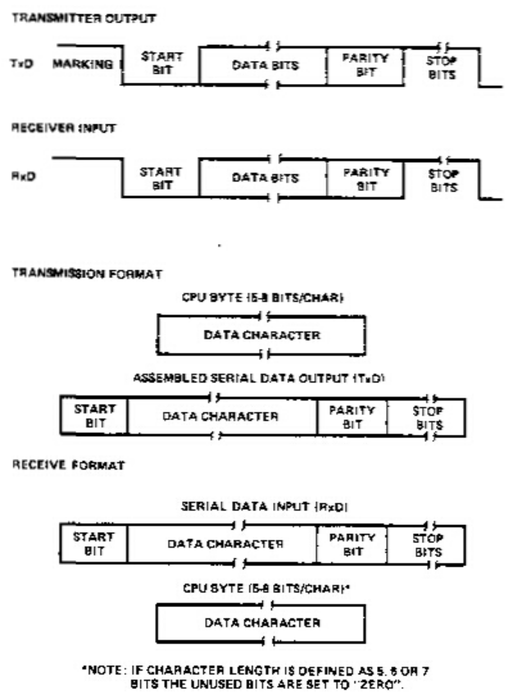
When no data characters have loaded into the 8251 the TxD output remains "high" (marking) unless a Break (continuously low) has been programmed.

## Asynchronous Mode (Receive)

The RxD line is normally high. A falling edge on this line triggers the beginning of a START bit. The validity of this START bit is checked by again strobing this bit at its nominal center. If a low is detected again, it is a valid START bit, and the bit counter will start counting. The bit counter locates the center of the data bits, the parity bit (if it exists) and the stop bits. If parity error occurs, the parity error flag is set. Data and parity bits are sampled on the RxD pin with the rising edge of $\overline{RxC}$. If a low level is detected as the STOP bit, the Framing Error flag will be set. The STOP bit signals the end of a character. This character is then loaded into the parallel I/O buffer of the 8251. The RxRDY pin is raised to signal the CPU that a character is ready to be fetched. If a previous character has not been fetched by the CPU, the present character replaces it in the I/O buffer, and the OVERRUN flag is raised (thus the previous character is lost). All of the error flags can be reset by a command instruction. The occurrence of any of these errors will not stop the operation of the 8251.




Mode Instruction Format, Asynchronous Mode




Asynchronous Mode

## Synchronous Mode (Transmission)

The TxD output is continuously high until the CPU sends its first character to the 8251 which usually is a SYNC character. When the $\overline{CTS}$ line goes low, the first character is serially transmitted out. All characters are shifted out on the falling edge of $\overline{TxC}$. Data is shifted out at the same rate as the $\overline{TxC}$.

Once transmission has started, the data stream at TxD output must continue at the $\overline{TxC}$ rate. If the CPU does not provide the 8251 with a character before the 8251 becomes empty, the SYNC characters (or character if in single SYNC word mode) will be automatically inserted in the TxD data stream. In this case, the TxEMPTY pin is raised high to signal that the 8251 is empty and SYNC characters are being sent out. The TxEMPTY pin is internally reset by the next character being written into the 8251.

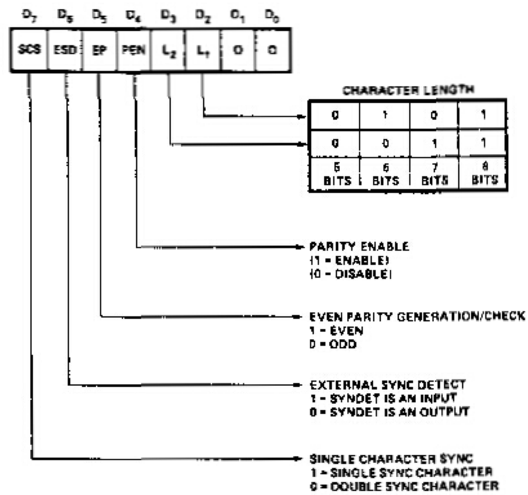
## Synchronous Mode (Receive)

In this mode, character synchronization can be internally or externally achieved. If the internal SYNC mode has been programmed, the receiver starts in a HUNT mode. Data on the RxD pin is then sampled in on the rising edge of $\overline{RxC}$. The content of the Rx buffer is continuously compared with the first SYNC character until a match occurs. If the 8251 has been programmed for two SYNC characters, the subsequent received character is also compared; when both SYNC characters have been detected, the USART ends the HUNT mode and is in character synchronization. The SYNDET pin is then set high, and is reset automatically by a STATUS READ.

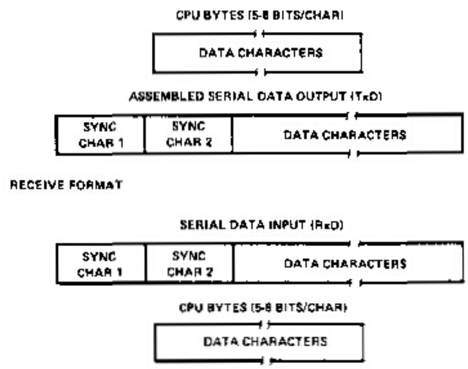
In the external SYNC mode, synchronization is achieved by applying a high level on the SYNDET pin. The high level can be removed after one $\overline{RxC}$ cycle.

Parity error and overrun error are both checked in the same way as in the Asynchronous Rx mode.

The CPU can command the receiver to enter the HUNT mode if synchronization is lost.




Mode Instruction Format, Synchronous Mode

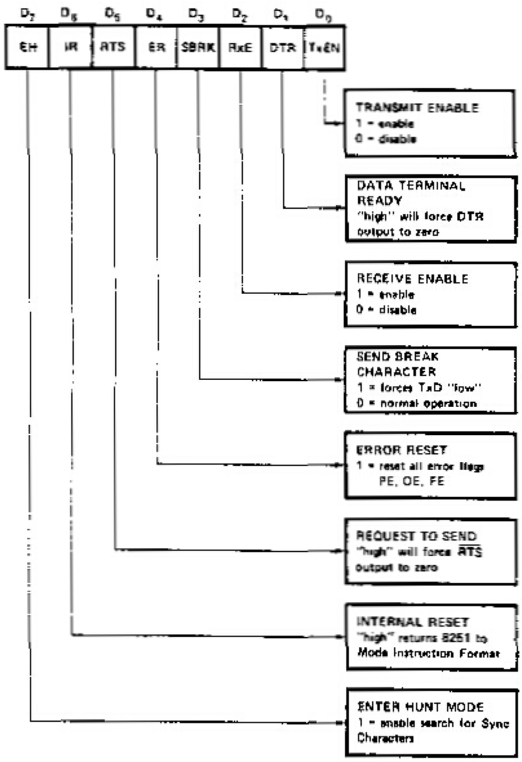



Synchronous Mode, Transmission Format

## COMMAND INSTRUCTION DEFINITION

Once the functional definition of the 8251 has been programmed by the Mode Instruction and the Sync Characters are loaded (if in Sync Mode) then the device is ready to be used for data communication. The Command Instruction controls the actual operation of the selected format. Functions such as: Enable Transmit/Receive, Error Reset and Modem Controls are provided by the Command Instruction.

Once the Mode Instruction has been written into the 8251 and Sync characters inserted, if necessary, then all further "control writes" ($C/\overline{D}$ = 1) will load the Command Instruction. A Reset operation (internal or external) will return the 8251 to the Mode Instruction Format.



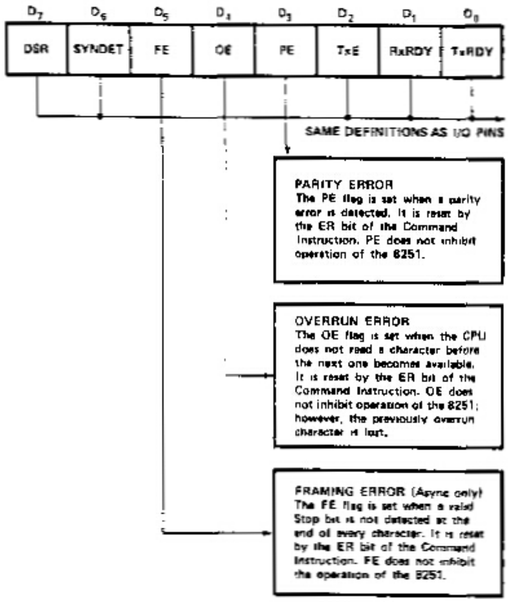

Command Instruction Format

## STATUS READ DEFINITION

In data communication systems it is often necessary to examine the "status" of the active device to ascertain if errors have occurred or other conditions that require the processor's attention. The 8251 has facilities that allow the programmer to "read" the status of the device at any time during the functional operation.

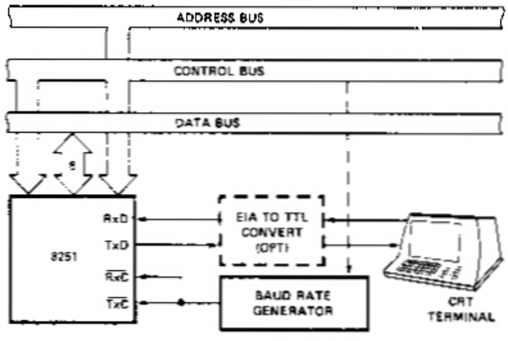
A normal "read" command is issued by the CPU with the C/D input at one to accomplish this function.

Some of the bits in the Status Read Format have identical meanings to external output pins so that the 8251 can be used in a completely Polled environment or in an interrupt driven environment.

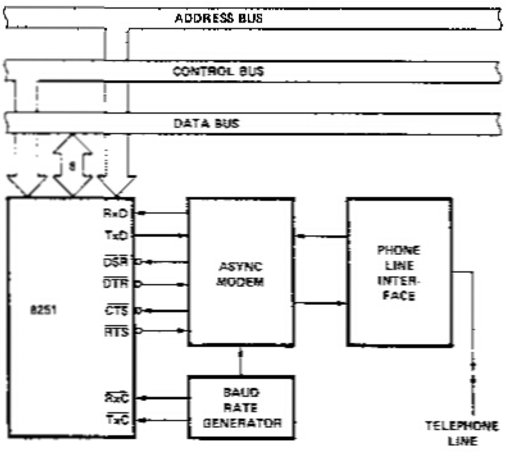



Status Read Format

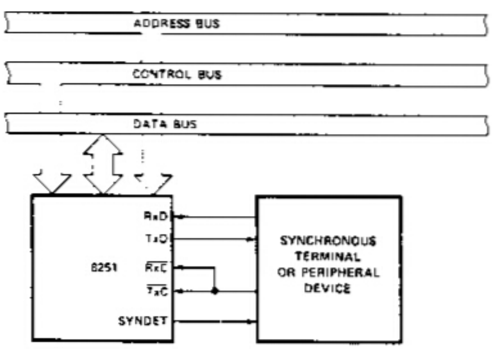
## APPLICATIONS OF THE 8251



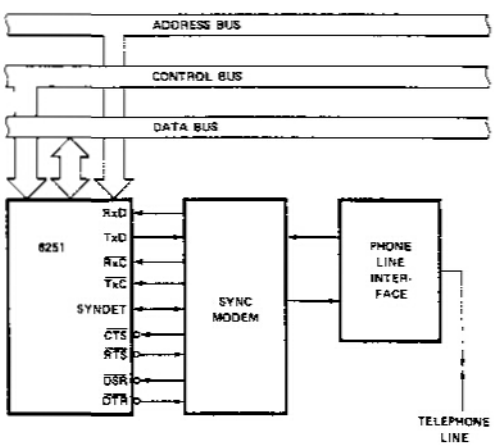

**Asynchronous Serial Interface to CRT Terminal, DC-9600 Baud**




**Asynchronous Interface to Telephone Lines**




**Synchronous Interface to Terminal or Peripheral Device**




**Synchronous Interface to Telephone Lines**

### D.C. Characteristics:

$T_A$ = 0°C to 70°C; $V_{CC}$ = 5.0V ± 5%; $V_{SS}$ = 0V

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $V_{IL}$ | Input Low Voltage | $V_{SS}$-.5 | | 0.8 | V | |
| $V_{IH}$ | Input High Voltage | 2.0 | | $V_{CC}$ | V | |
| $V_{OL}$ | Output Low Voltage | | | 0.45 | V | $I_{OL}$ = 1.6mA |
| $V_{OH}$ | Output High Voltage | 2.2 | | | V | $I_{OH}$ = -100μA ($DB_{0-7}$)<br>$I_{OH}$ = -100μA (Others) |
| $I_{DL}$ | Data Bus Leakage | | | 50 | μA | $V_{OUT}$ = 4.5V |
| $I_{LI}$ | Input Load Current | | | 10 | μA | @ 5.5V |
| $I_{CC}$ | Power Supply Current | | 45 | 80 | | |

### Capacitance

$T_A$ = 25°C; $V_{CC}$ = $V_{SS}$ = 0V

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | | | 10 | pF | fc = 1MHz |
| $C_{I/O}$ | I/O Capacitance | | | 20 | pF | Unmeasured pins returned to $V_{SS}$. |

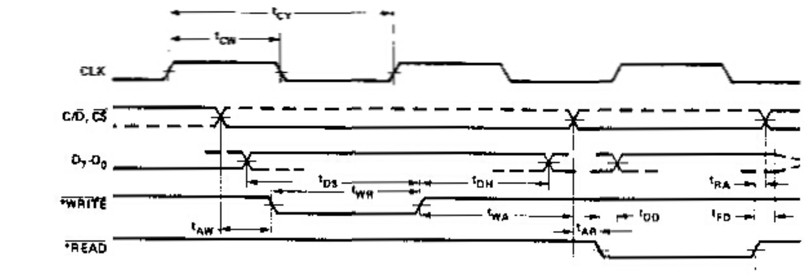
## A.C. Characteristics:

$T_A = 0°C$ to $70°C$; $V_{CC} = 5.0V \pm 5\%$; $V_{SS} = 0V$

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| $t_{CY}$ | Clock Period | .420 | | 1.35 | μs | |
| $t_{\phi W}$ | Clock Pulse Width | 220 | | 300 | ns | |
| $t_R, t_F$ | Clock Rise and Fall Time | 0 | | 50 | ns | |
| $t_{WR}$ | $\overline{\text{WRITE}}$ Pulse Width | 430 | | | ns | |
| $t_{DS}$ | Data Set-Up Time for $\overline{\text{WRITE}}$ | 0 | | | ns | |
| $t_{DH}$ | Data Hold Time for $\overline{\text{WRITE}}$ | 65 | | | ns | |
| $t_{AW}$ | Address Stable before $\overline{\text{WRITE}}$ | 20 | | | ns | |
| $t_{WA}$ | Address Hold Time for $\overline{\text{WRITE}}$ | 35 | | | ns | |
| $t_{RD}$ | READ Pulse Width | 430 | | | ns | |
| $t_{DD}$ | Data Delay from $\overline{\text{READ}}$ | 350 | | | ns | $C_L$=100pF |
| $t_{DF}$ | $\overline{\text{READ}}$ to Data Floating | 150 | | | ns | $C_L$=100pF |
| $t_{AR1}$ | Address Stable before $\overline{\text{READ}}$, CE (C/D) | 50 | | | ns | |
| $t_{RA1}$ | Address Hold Time for $\overline{\text{READ}}$, CE | 5 | | | ns | |
| $t_{RA2}$ | Address Hold Time for $\overline{\text{READ}}$, C/D | 370 | | | ns | |
| $t_{DTx}$ | TxD Delay from Falling Edge of TxC | 1 | | | μs | $C_L$=100pF |
| $t_{SRx}$ | Rx Data Set-Up Time to Sampling Pulse | 2 | | | μs | $C_L$=100pF |
| $t_{HRx}$ | Rx Data Hold Time to Sampling Pulse | 2 | | | μs | $C_L$=100pF |
| $f_{Tx}$ | Transmitter Input Clock Frequency | | | | | |
| | 1X Baud Rate | DC | | 56 | KHz | |
| | 16X and 64X Baud Rate | DC | | 615 | KHz | |
| $f_{Rx}$ | Receiver Input Clock Frequency | | | | | |
| | 1X Baud Rate | DC | | 56 | KHz | |
| | 16X and 64X Baud Rate | DC | | 615 | KHz | |
| $t_{Tx}$ | TxRDY Delay from Center of Data Bit | | | 16 | CLK Period | $C_L$=50pF |
| $t_{Rx}$ | RxRDY Delay from Center of Data Bit | 15 | | 20 | CLK Period | |
| $t_{IS}$ | Internal Syndet Delay from Center of Data Bit | 20 | | 25 | CLK Period | |
| $t_{ES}$ | External Syndet Set-Up Time before Falling Edge of RxC | | | 15 | CLK Period | |

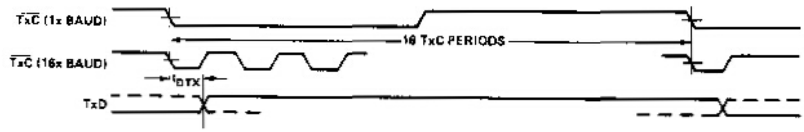
Note: The TxC and RxC frequencies have the following limitation with respect to CLK.
For ASYNC Mode, $t_{Tx}$ or $t_{Rx} \geqslant 4.5\ t_{CY}$
For SYNC Mode, $t_{Tx}$ or $t_{Rx} \geqslant 30\ t_{CY}$

### READ AND WRITE TIMING

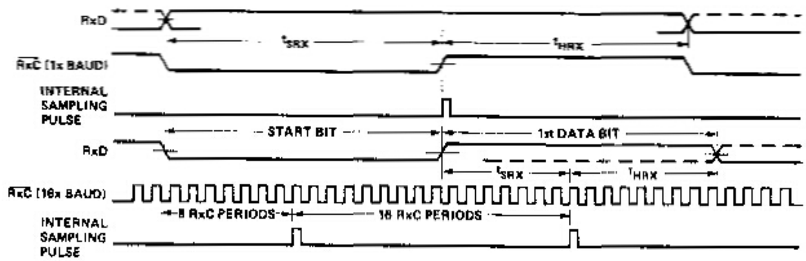



*WRITE AND READ PULSES HAVE NO TIMING LIMITATION WITH RESPECT TO CLK

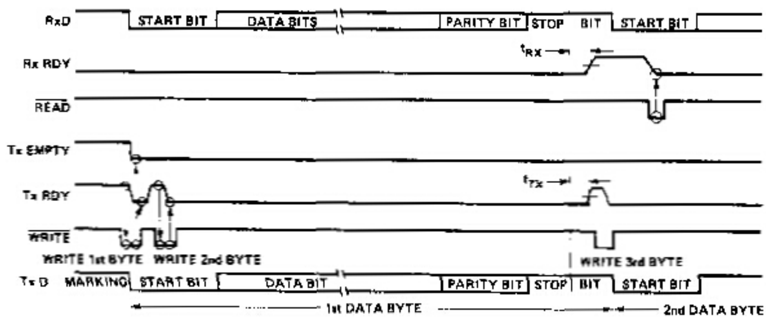
### TRANSMITTER CLOCK AND DATA



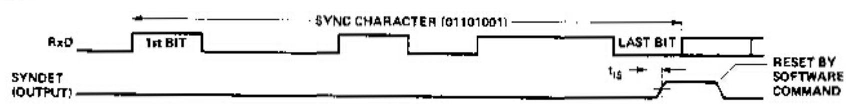

### RECEIVER CLOCK AND DATA



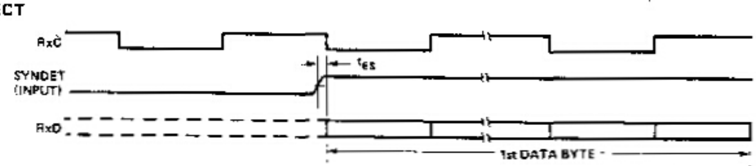

### Tx RDY AND Rx RDY TIMING (ASYNC MODE)




### INTERNAL SYNC DETECT



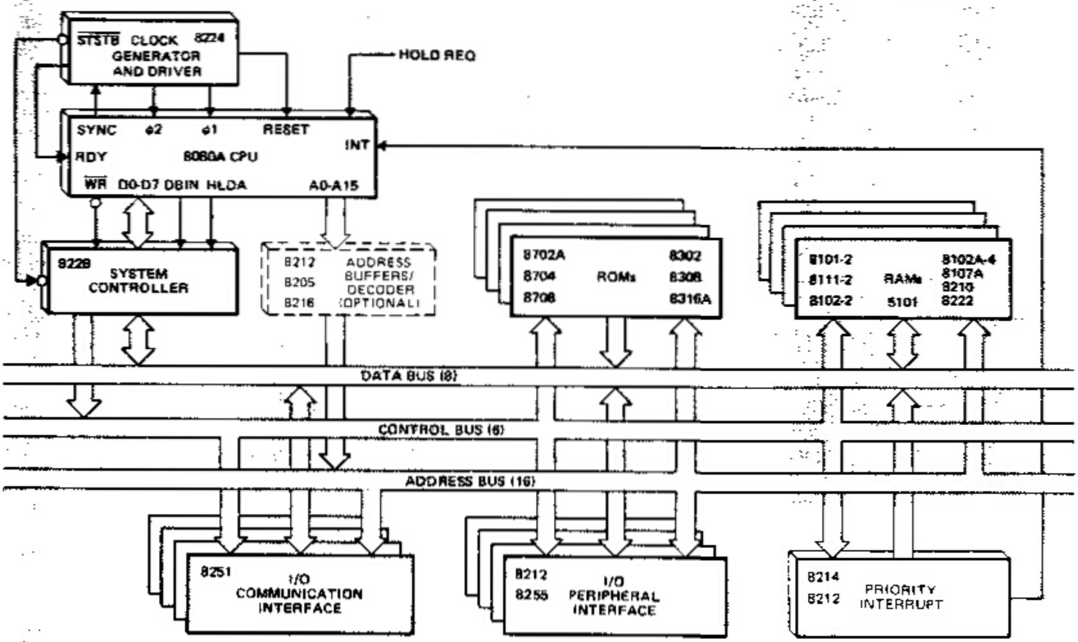

### EXTERNAL SYNC DETECT



intel® Microcomputer Systems

# Peripherals

# 8205
# 8214
# 8216/8226



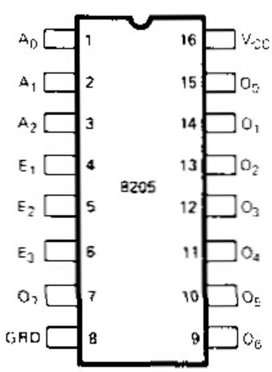
# HIGH SPEED 1 OUT OF 8 BINARY DECODER

- **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**

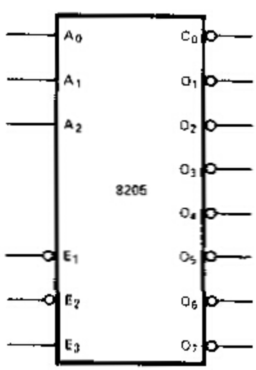
The 8205 decoder can be used for expansion of systems which utilize input ports, output ports, and memory components with active low chip select input. When the 8205 is enabled, one of its eight outputs goes "low", thus a single row of a memory system is selected. The 3 chip enable inputs on the 8205 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 |
| $\overline{E_1}$-$E_3$ | 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 |

## FUNCTIONAL DESCRIPTION

### Decoder

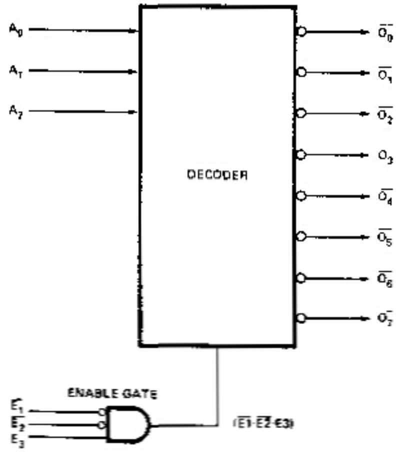
The 8205 contains a one out of eight binary decoder. It accepts a three bit binary code and by gating this input, creates an exclusive output that represents the value of the input code.

For example, if a binary code of 101 was present on the A0, A1 and A2 address input lines, and the device was enabled, an active low signal would appear on the $\overline{05}$ output line. Note that all of the other output pins are sitting at a logic high, thus the decoded output is said to be exclusive. The decoders outputs will follow the truth table shown below in the same manner for all other input variations.

### Enable Gate

When using a decoder it is often necessary to gate the outputs with timing or enabling signals so that the exclusive output of the decoded value is synchronous with the overall system.

The 8205 has a built-in function for such gating. The three enable inputs ($\overline{E1}$, $\overline{E2}$, E3) are ANDed together and create a single enable signal for the decoder. The combination of both active "high" and active "low" device enable inputs provides the designer with a powerfully flexible gating function to help reduce package count in his system.




| 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 |

## APPLICATIONS OF THE 8205

The 8205 can be used in a wide variety of applications in microcomputer systems. I/O ports can be decoded from the address bus, chip select signals can be generated to select memory devices and the type of machine state such as in 8008 systems can be derived from a simple decoding of the state lines (S0, S1, S2) of the 8008 CPU.

### I/O Port Decoder

Shown in the figure below is a typical application of the 8205. Address input lines are decoded by a group of 8205s (3). Each input has a binary weight. For example, A0 is assigned a value of 1 and is the LSB; A4 is assigned a value of 16 and is the MSB. By connecting them to the decoders as shown, an active low signal that is exclusive in nature and represents the value of the input address lines, is available at the outputs of the 8205s.

This circuit can be used to generate enable signals for I/O ports or any other decoder related application.

Note that no external gating is required to decode up to 24 exclusive devices and that a simple addition of an inverter or two will allow expansion to even larger decoder networks.

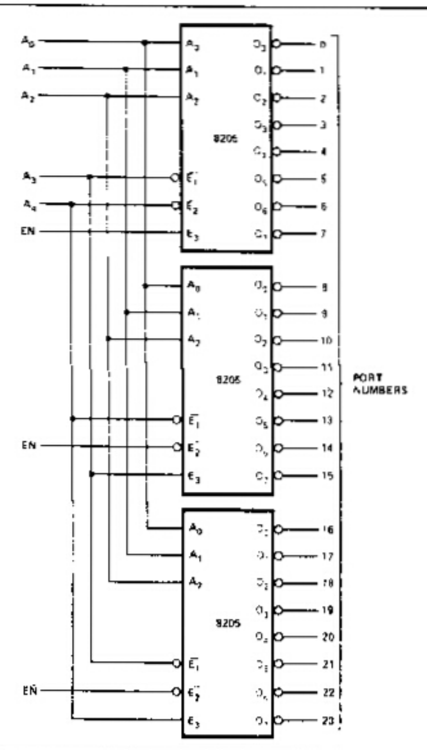
### Chip Select Decoder

Using a very similar circuit to the I/O port decoder, an array of 8205s can be used to create a simple interface to a 24K memory system.

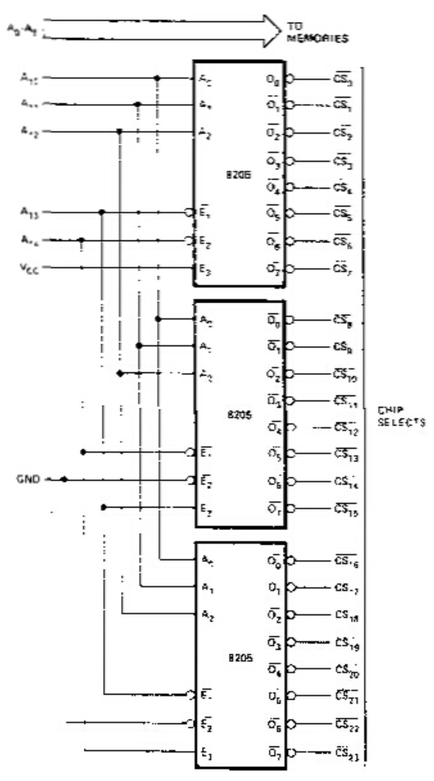
The memory devices used can be either ROM or RAM and are 1K in storage capacity. 8308s and 8102s are the devices typically used for this application. This type of memory device has ten (10) address inputs and an active "low" chip select ($\overline{CS}$). The lower order address bits A0-A9 which come from the microprocessor are "bussed" to all memory elements and the chip select to enable a specific device or group of devices comes from the array of 8205s. The output of the 8205 is active low so it is directly compatible with the memory components.

Basic operation is that the CPU issues an address to identify a specific memory location in which it wishes to "write" or "read" data. The most significant address bits A10-A14 are decoded by the array of 8205s and an exclusive, active low, chip select is generated that enables a specific memory device. The least significant address bits A0-A9 identify a specific location within the selected device. Thus, all addresses throughout the entire memory array are exclusive in nature and are non-redundant.

This technique can be expanded almost indefinitely to support even larger systems with the addition of a few inverters and an extra decoder (8205).




I/O Port Decoder




24K Memory Interface

## Logic Element Example

Probably the most overlooked application of the 8205 is that of a general purpose logic element. Using the "on-chip" enabling gate, the 8205 can be configured to gate its decoded outputs with system timing signals and generate strobes that can be directly connected to latches, flip-flops and one-shots that are used throughout the system.

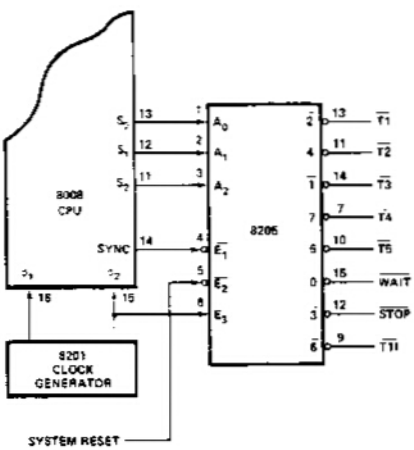
An excellent example of such an application is the "state decoder" in an 8008 CPU based system. The 8008 CPU issues three bits of information (S0, S1, S2) that indicate the nature of the data on the Data Bus during each machine state. Decoding of these signals is vital to generate strobes that can load the address latches, control bus discipline and general machine functions.

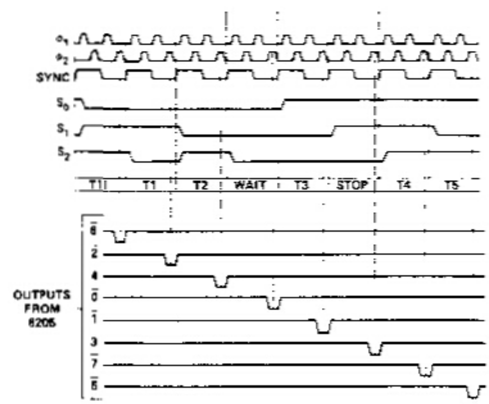
In the figure below a circuit is shown using the 8205 as the "state decoder" for an 8008 CPU that not only decodes the S0, S1, S2 outputs but gates these signals with the clock (phase 2) and the SYNC output of the 8008 CPU. The $\overline{T1}$ and $\overline{T2}$ decoded strobes can connect directly to devices like 8212s for latching the address information. The other decoded strobes can be used to generate signals to control the system data bus, memory timing functions and interrupt structure. RESET is connected to the enable gate so that strobes are not generated during system reset, eliminating accidental loading.

The power of such a circuit becomes evident when a single decoded strobe is logically broken down. Consider $\overline{T1}$ output, the boolean equation for it would be:

$$\overline{T1} = (\overline{S0}\cdot S1\cdot \overline{S2})\cdot(\overline{SYNC}\cdot \text{Phase 2}\cdot \overline{\text{Reset}})$$

A six input NAND gate plus a few inverters would be needed to implement this function. The seven remaining outputs would need a similar circuit to duplicate their function, obviously a substantial savings in components can be achieved when using such a technique.




State Control Coding

| $S_0$ | $S_1$ | $S_2$ | STATE |
|---|---|---|---|
| 0 | 1 | 0 | T1 |
| 0 | 1 | 1 | T1I |
| 0 | 0 | 1 | T2 |
| 0 | 0 | 0 | WAIT |
| 1 | 0 | 0 | T3 |
| 1 | 1 | 0 | STOP |
| 1 | 1 | 1 | T4 |
| 1 | 0 | 1 | T5 |

## ABSOLUTE MAXIMUM RATINGS*

| | | |
|---|---|---|
| Temperature Under Bias: | Ceramic | −65°C to +125°C |
| | Plastic | −65°C to +75°C |
| Storage Temperature | | −65°C to +160°C |
| All Output or Supply Voltages | | −0.5 to +7 Volts |
| All Input Voltages | | −1.0 to +5.5 Volts |
| Output Currents | | 125 mA |

*COMMENT

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition 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.

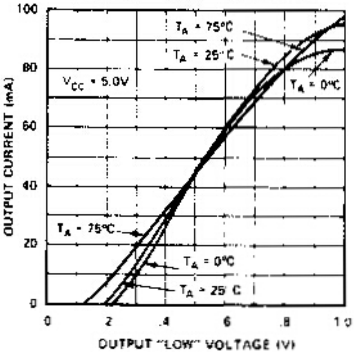
## D.C. CHARACTERISTICS $T_A$ = 0°C to +75°C, $V_{CC}$ = 5.0V ±5%

### 8205

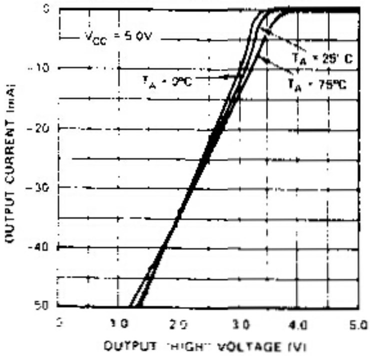
| SYMBOL | PARAMETER | LIMIT | | UNIT | TEST CONDITIONS |
|---|---|---|---|---|---|
| | | MIN. | MAX. | | |
| $I_F$ | INPUT LOAD CURRENT | | −0.25 | mA | $V_{CC}$ = 5.25V, $V_F$ = 0.45V |
| $I_R$ | INPUT LEAKAGE CURRENT | | 10 | μA | $V_{CC}$ = 5.25V, $V_R$ = 5.25V |
| $V_C$ | INPUT FORWARD CLAMP VOLTAGE | | −1.0 | V | $V_{CC}$ = 4.75V, $I_C$ = −5.0 mA |
| $V_{OL}$ | OUTPUT "LOW" VOLTAGE | | 0.45 | V | $V_{CC}$ = 4.75V, $I_{OL}$ = 10.0 mA |
| $V_{OH}$ | OUTPUT HIGH VOLTAGE | 2.4 | | V | $V_{CC}$ = 4.75V, $I_{OH}$ = −1.5 mA |
| $V_{IL}$ | INPUT "LOW" VOLTAGE | | 0.85 | V | $V_{CC}$ = 5.0V |
| $V_{IH}$ | INPUT "HIGH" VOLTAGE | 2.0 | | V | $V_{CC}$ = 5.0V |
| $I_{SC}$ | OUTPUT HIGH SHORT CIRCUIT CURRENT | −40 | −120 | mA | $V_{CC}$ = 5.0V, $V_{OUT}$ = 0V |
| $V_{CX}$ | OUTPUT "LOW" VOLTAGE @ HIGH CURRENT | | 0.8 | V | $V_{CC}$ = 5.0V, $I_{OX}$ = 40 mA |
| $I_{CC}$ | POWER SUPPLY CURRENT | | 70 | mA | $V_{CC}$ = 5.25V |

## TYPICAL CHARACTERISTICS

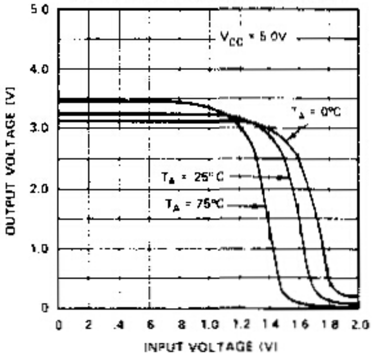
OUTPUT CURRENT VS. OUTPUT "LOW" VOLTAGE




OUTPUT CURRENT VS. OUTPUT "HIGH" VOLTAGE




DATA TRANSFER FUNCTION



## 8205 SWITCHING CHARACTERISTICS

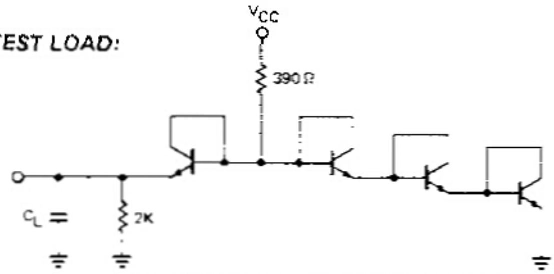
### CONDITIONS OF TEST:

Input pulse amplitudes: 2.5V

Input rise and fall times: 5 nsec between 1V and 2V

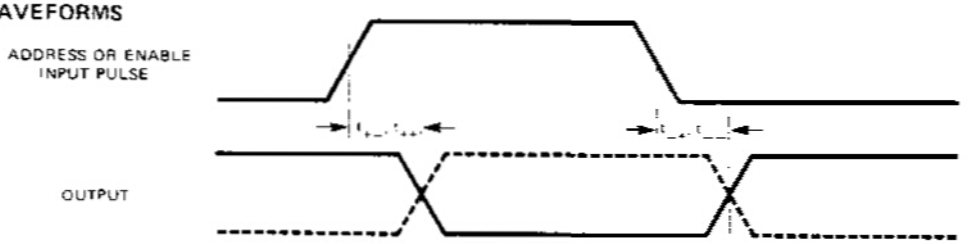
Measurements are made at 1.5V

### TEST LOAD:




All Transistors 2N2369 or Equivalent. $C_L$ = 30 pF

### TEST WAVEFORMS



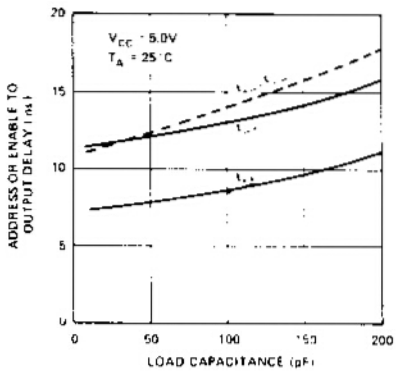

### A.C. CHARACTERISTICS $T_A$ = 0°C to +75°C, $V_{CC}$ = 5.0V ±5% unless otherwise specified.

| SYMBOL | PARAMETER | | MAX. LIMIT | UNIT | TEST CONDITIONS |
|---|---|---|---|---|---|
| $t_{++}$ | ADDRESS OR ENABLE TO OUTPUT DELAY | | 18 | ns | |
| $t_{-+}$ | | | 18 | ns | |
| $t_{+-}$ | | | 18 | ns | |
| $t_{--}$ | | | 18 | ns | |
| $C_{IN}$ (1) | INPUT CAPACITANCE | P8205 | 4(typ.) | pF | f = 1 MHz, $V_{CC}$ = 0V |
| | | C8205 | 5(typ.) | pF | $V_{BIAS}$ = 2.0V, $T_A$ = 25°C |

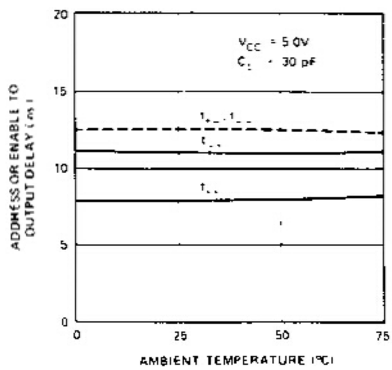
1. This parameter is periodically sampled and is not 100% tested.

### TYPICAL CHARACTERISTICS

ADDRESS OR ENABLE TO OUTPUT DELAY VS. LOAD CAPACITANCE




ADDRESS OR ENABLE TO OUTPUT DELAY VS. AMBIENT TEMPERATURE



# Schottky Bipolar 8214

## PRIORITY INTERRUPT CONTROL UNIT

- **Eight Priority Levels**
- **Current Status Register**
- **Priority Comparator**
- **Fully Expandable**
- **High Performance (50ns)**
- **24-Pin Dual In-Line Package**

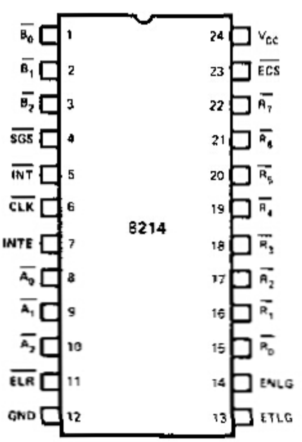
The 8214 is an eight level priority interrupt control unit designed to simplify interrupt driven microcomputer systems.

The PICU can accept eight requesting levels; determine the highest priority, compare this priority to a software controlled current status register and issue an interrupt to the system along with vector information to identify the service routine.

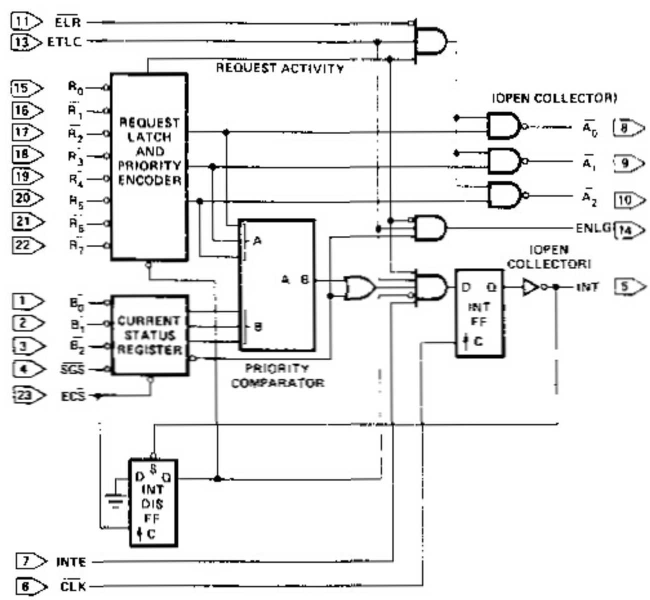
The 8214 is fully expandable by the use of open collector interrupt output and vector information. Control signals are also provided to simplify this function.

The PICU is designed to support a wide variety of vectored interrupt structures and reduce package count in interrupt driven microcomputer systems.

### PIN CONFIGURATION




### LOGIC DIAGRAM




### PIN NAMES

| | |
|---|---|
| **INPUTS** | |
| $\overline{R_0}$-$\overline{R_7}$ | REQUEST LEVELS ($R_7$ HIGHEST PRIORITY) |
| $\overline{B_0}$-$\overline{B_2}$ | CURRENT STATUS |
| $\overline{SGS}$ | STATUS GROUP SELECT |
| $\overline{ECS}$ | ENABLE CURRENT STATUS |
| INTE | INTERRUPT ENABLE |
| $\overline{CLK}$ | CLOCK (INT F-F) |
| $\overline{ELR}$ | ENABLE LEVEL READ |
| ETLG | ENABLE THIS LEVEL GROUP |
| **OUTPUTS:** | |
| $\overline{A_0}$-$\overline{A_2}$ | REQUEST LEVELS — OPEN COLLECTOR |
| $\overline{INT}$ | INTERRUPT (ACT. LOW) — OPEN COLLECTOR |
| ENLG | ENABLE NEXT LEVEL GROUP |

## INTERRUPTS IN MICROCOMPUTER SYSTEMS

Microcomputer system design requires that I/O devices such as keyboards, displays, sensors and other components receive servicing in an efficient method so that large amounts of the total systems tasks can be assumed by the microcomputer with little or no effect on throughput.

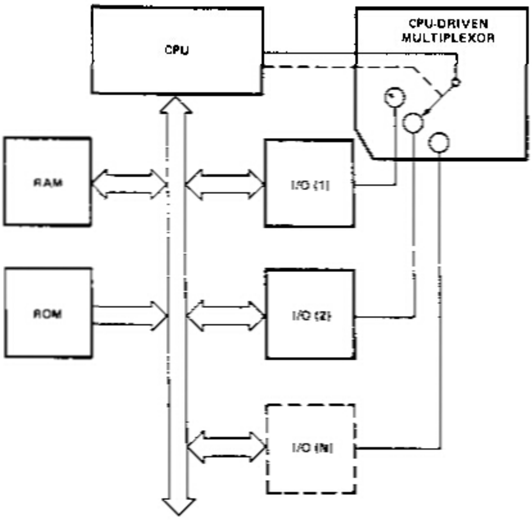
The most common method of servicing such devices is the **Polled** approach. This is where the processor must test each device in sequence and in effect "ask" each one if it needs servicing. It is easy to see that a large portion of the main program is looping through this continuence polling cycle and that such a method would have a serious, detrimental effect on system throughput thus limiting the tasks that could be assumed by the microcomputer and reducing the cost effectiveness of using such devices.

A more desireable method would be one that would allow the microprocessor to be executing its main program and only stop to service peripheral devices when it is told to do so by the device itself. In effect, the method would provide an external asynchronous input that would inform the processor that it should complete whatever instruction that is currently being executed and fetch a new routine that will service the requesting device. Once this servicing is complete however the processor would resume exactly where it left off.

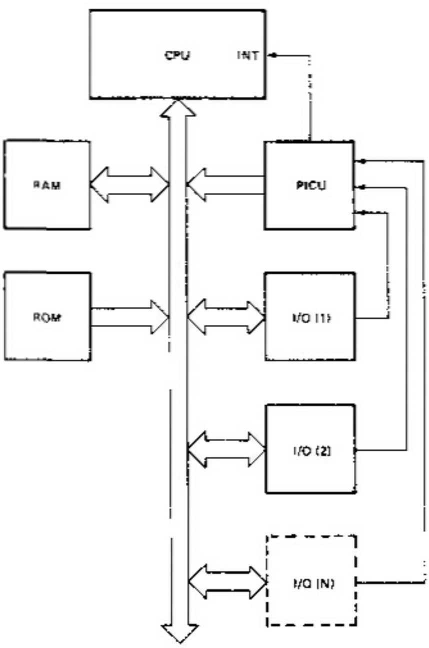
This method is called **Interrupt**. It is easy to see that system throughput would drastically increase, and thus more tasks could be assumed by the microcomputer to further enhance its cost effectiveness.

The Priority Interrupt Control Unit (PICU) functions as an overall manager in an Interrupt-Driven system environment. It accepts requests from the peripheral equipment, determines which of the incoming requests is of the highest importance (priority), ascertains whether the incoming request has a higher priority value than the level currently being serviced and issues an Interrupt to the CPU based on this determination.

Each peripheral device or structure usually has a special program or "routine" that is associated with its specific functional or operational requirements; this is referred to as a "service routine". The PICU, after issuing an Interrupt to the CPU, must somehow input information into the CPU that can "point" the Program Counter to the service routine associated with the requesting device. The PICU encodes the requesting level into such information for use as a "vector" to the correct Interrupt Service Routine.




**Polled Method**




**Interrupt Method**

## FUNCTIONAL DESCRIPTION

### General

The 8214 is a device specifically designed for use in real time, interrupt driven, microcomputer systems. Basically it is an eight (8) level priority control unit that can accept eight different interrupt requests, determine which has the highest priority, compare that level to a software maintained current status register and issue an interrupt to the system based on this comparison along with vector information to indicate the location of the service routine.

### Priority Encoder

The eight requests inputs, which are active low, come into the Priority Encoder. This circuit determines which request input is the most important (highest priority) as preassigned by the designer. ($\overline{R7}$) is the highest priority input to the 8214 and ($\overline{R0}$) is the lowest. The logic of the Priority Encoder is such that if two or more input levels arrive at the same time then the input having the highest priority will take presidence and a three bit output, corresponding to the active level (modulo 8) will be sent out. The Priority Encoder also contains a latch to store the request input. This latch is controlled by the Interrupt Disable Flip-flop so that once an interrupt has been issued by the 8214 the request latch is no longer open. (Note that the latch does not store inactive requests. In order for a request to be monitored by the 8214 it must remain present until it has been serviced.)

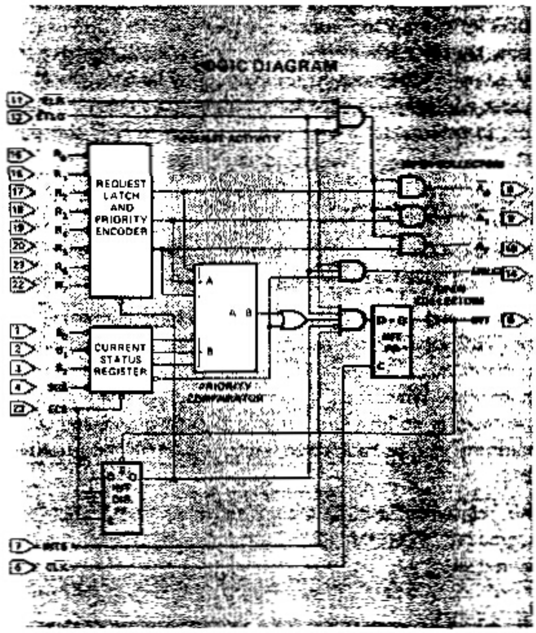
### Current Status Register

In an interrupt driven microcomputer system it is important to not only prioritize incoming requests but to ascertain whether such a request is a higher priority than the interrupt currently being serviced.

The Current Status Register is a simple 4-bit latch that is treated as an addressable outport port by the microcomputer system. It is loaded when the $\overline{ECS}$ input goes low.

Maintenance of the Current Status Register is performed as a portion of the service routine. Basically, when an interrupt is issued to the system the programmer outputs a binary code (modulo 8) that is the compliment of the interrupt level. This value is stored in the Current Status Register and is compared to all further prioritized incoming requests by the Priority Comparator. In essence, a copy of the current interrupt level is written into the 8214 to be used as a reference for comparison. There is no restriction to this maintenance, other level values can be written into this register as references so that groups of interrupt requests may be disallowed under complete control of the programmer.

Note that the fourth bit in the register is $\overline{SGS}$. This input is part of the value written out by the programmer and performs a special function. The Priority Comparator will only issue an output that indicates the request level is greater than the Current Status Register. If both comparator inputs are equal to zero no output will be present. The $\overline{SGS}$ input allows the programmer to, in effect, disable this comparison and allow the 8214 to issue an interrupt to the system that is based only on the logic of the priority encoder.



### Control Signals

The 8214 also has several inputs that enable the designer to synchronize the interrupt issued to the microprocessor and to allow or disallow such an issuance. Also, signals are provided that permit simple expansion to other 8214s so that more than eight levels can be controlled.

### INTE, $\overline{\text{CLK}}$

The INTE (Interrupt Enable) input allows the designer to "shutoff" the interrupt system under control of external logic or possibly under software maintenance. A "zero" on this line will not allow interrupts to be issued to the microcomputer system.

The $\overline{\text{CLK}}$ (Clock) input is actually the trigger that strobes the Interrupt Flip-Flop. It can be connected to one of the clocks of the microprocessor so that the interrupt issued meets the CPU set-up time specification. Note that due to the gating of the input to the Interrupt Flip-Flop the $\overline{\text{INT}}$ output will only be active for the time of a single clock period, so external latching may be required to hold this signal.

### $\overline{\text{ELR}}$, ETLG, ENGL

These three signals allow 8214s to be cascaded so that more than eight levels of interrupt requests can be controlled.

Basically, the ENLG output of one 8214 is connected to the ETLG input of the next and so on, with the first 8214 having its ETLG input pulled "high" and assigned the highest priority. When the ENLG output is "high" it indicates that there is no interrupt pending on that device and that interrupts can be monitored on the next lower priority 8214.

This "cascading" can be expanded almost indefinitely to accomodate even the largest of interrupt driven system architectures.

### $\overline{\text{A0}}$, $\overline{\text{A1}}$, $\overline{\text{A2}}$

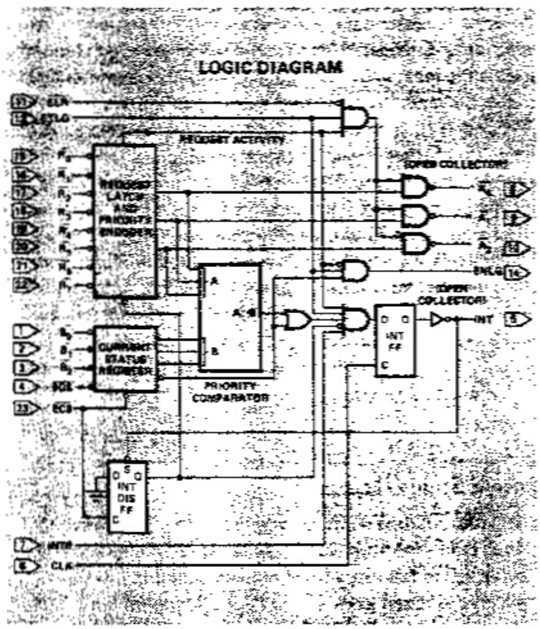
In order to identify which device has interrupted the processor so that the service routine associated with it can be addressed, a pointer or "vector" must accompany the interrupt issued to the microcomputer system.

The $\overline{\text{A0}}$, $\overline{\text{A1}}$ and $\overline{\text{A2}}$ outputs represent the complement of the active interrupt level (modulo 8). By using these signals to encode the special instruction, RST, the program counter of the microprocessor, can point to the location of the service routine. Note that these three outputs are gated by the $\overline{\text{ELR}}$ input and are open collector so that expansion is simplified.

### $\overline{\text{INT}}$

The $\overline{\text{INT}}$ output of the 8214 is the signal that is issued to the microprocessor to initiate the interrupt sequence. As soon a $\overline{\text{INT}}$ *is active the INT DIS FF is set, inhibiting further* requests from entering the Request Latch. Only the writing out of the current status information by strobing the $\overline{\text{ECS}}$ input will clear the INT DIS FF and allow requests to enter the latch.

Note that $\overline{\text{INT}}$ is also open collector so that when cascaded to other 8214s an interrupt in any of the active devices will set all INT DIS FFs in the entire array.



## APPLICATIONS OF THE 8214

### 8 Level Controller (8080)

The most common of applications of the 8214 is that of an eight level priority structure for 8080 or 8008 microcomputer systems.

Shown in the figure below is a detailed logic schematic of a simple circuit that will accept eight input requests, maintain current status, issue the interrupt signal to the 8080 and encode the proper RST instruction to gate onto the data bus.

The eight requests are connected to the 8214 by the designer in whatever order of priority is to be preassigned. For example, eight keyboards could be monitored and each assigned a degree of importance (level of priority) so that faster processor attention or access can be assigned to the critical or time dependent tasks.

The inputs to the Current Status Register are connected to the Data Bus so that data can be written out into this "port".

An 8212 is used to encode the RST instruction and also to act as a 3-state gate to place the proper RST instruction when the 8080 Data Bus is in the input mode. Note that the $\overline{INT}$ signal from the 8214 is latched in the SR flip-flop of the 8212 so that proper timing is maintained. The 8212 is selected (enabled) when the INTA signal from the 8080 status latch and the DBIN from the 8080 are active, this assures that the RST instruction will be placed on the Data Bus at the proper time. Note that the $\overline{INT}$ output from the 8212 is inverted and pulled up before it is connected to the 8080. This is to generate an $\overline{INT}$ signal to the 8080 that has the correct polarity and meets the input voltage requirement (3.3V).

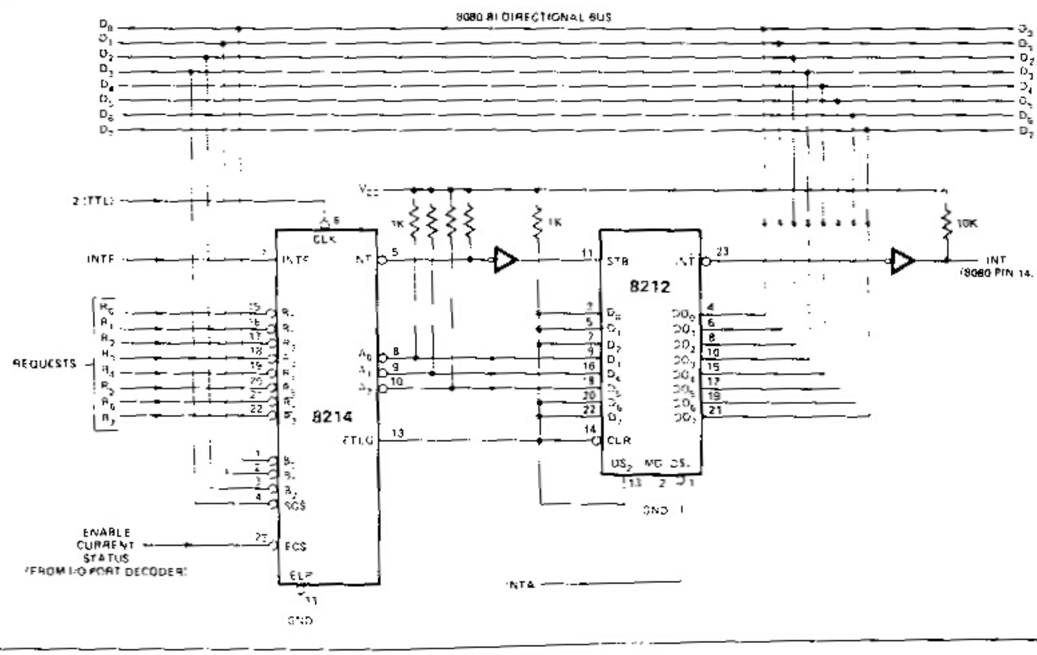
### Basic Operation

When the initial interrupt request is presented to the 8214 it will issue an interrupt to the 8080 if the structure is enabled. The 8214 will encode the request into 3 bits (modulo 8) and output them to the 8212. After the acknowledgement of the interrupt has been issued by the 8080 the encoded RST instruction is gated onto the Data Bus by the 8212. The processor executes the instruction and points the program counter to the desired serviced routine. In this routine the programmer will probably save the status of the register array and flags within a series of PUSH instructions (4). Then a copy of the current interrupt level (modulo 8) can be "built" in the Accumulator and output to the Current Status Register of the 8214 for use as a comparison reference for all further incoming requests to the system.

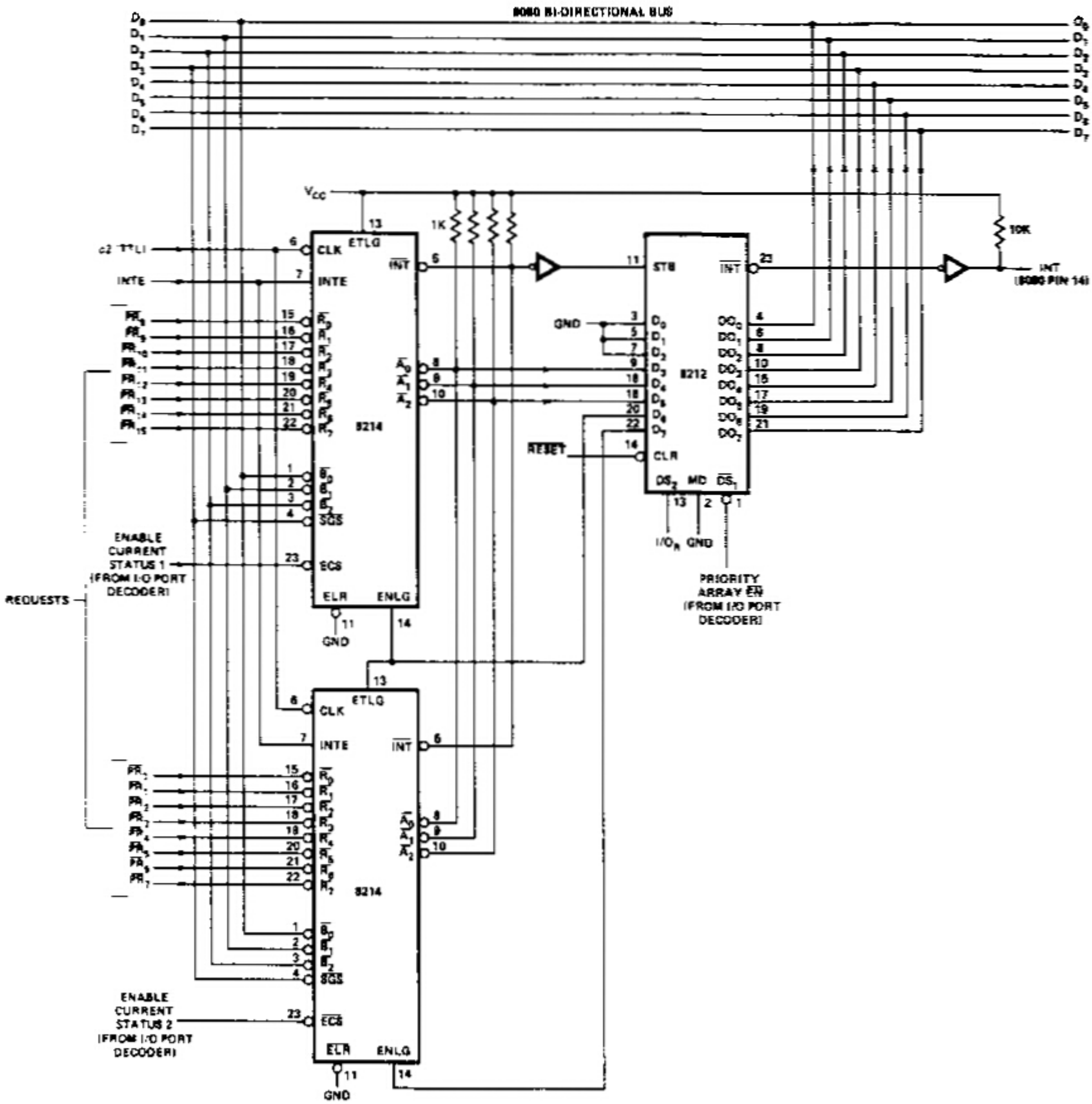
This Vectored Eight Level Priority Interrupt Structure for 8080 microcomputer systems is a powerful yet flexible circuit that is high performance and has a minimal component count.

| | | | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|---|---|---|
| | PRIORITY REQUEST | RST | 1 | 1 | $A_2$ | $A_1$ | $A_0$ | 1 | 1 | 1 |
| LOWEST | 0 | 7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 1 | 6 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| | 2 | 5 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| | 3 | 4 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| | 4 | 3 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| | 5 | 2 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| | 6 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| HIGHEST | 7 | *0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

*RST 0 WILL VECTOR PROGRAM COUNTER TO LOCATION 0 (ZERO) AND INVOKE THE SAME ROUTINE AS "RESET" INPUT TO 8080
THIS COULD RE-INITIALIZE THE SYSTEM BASED ON THE ROUTINE INVOKED
(A CAUTION TO SYSTEM PROGRAMMERS.)




8 Level Controller




16 Level Controller

## APPLICATIONS OF THE 8214

### Cascading the 8214

When greater than eight levels of interrupts must be prioritized and serviced, the 8214 can be cascaded with other 8214s to support such an architecture.

On the previous page a simple circuit is shown that can control 16 levels of interrupt and is easily expandable to support up to 40 levels of interrupt by just cascading more 8214s.

As described previously, there are signals provided in the 8214 for cascading ($\overline{ELR}$, ETLG, ENLG) and in effect the ENLG output of the first 8214 "ripples" down to the next and so on. The entire array of 8214s regardless of size, can be thought of as a single priority control unit, with the first having the highest priority and the next 8214 having a lower priority and so on.

In this application, the manner in which software handles the servicing of the interrupt will change. Since more than eight vectors must be generated a method other than the common RST instruction must be implemented. Basically, the priority control array must somehow modify the contents of the 8080 Program Counter so that it can point ("vector") to one of 16 (or how many levels are to be serviced) and fetch the proper service routine. A simple approach is to treat the priority control array as a single input port that can input a value into the Accumulator and use this value as an offset to modify the Program Counter (Indirect Jump).

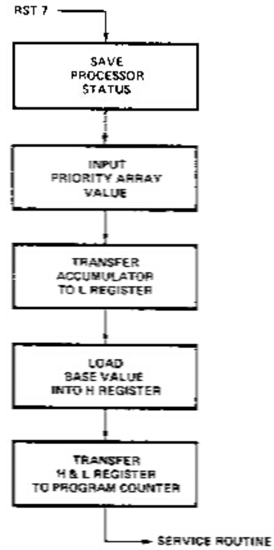
An initial CALL is needed to invoke this Indirect Jump routine so the circuitry is configured to insert an RST 7 (FFh) for all interrupts, thus the Indirect Jump Routine starts at location (56d).

The Assembly Code for the flow chart is as follows:

```
PUSH  PSW  ┐
PUSH  B    │  (save processor status)
PUSH  D    │  (22 microseconds)
PUSH  H    ┘
IN    (n)     (input Priority Array Value)
MOV   L,A     (transfer Accumulator to L register)
MVI   H,(n)   (load Base Address into H register)
PCHL          (transfer H&L to Program Counter)
```

(The execution time for the total routine is 35.5 microseconds based on an 8080 clock period of 500ns.)

Following is a basic flowchart of the priority array Indirect Jump routine. Note that the last step in the routine will vector the processor to fetch the proper service routine as dictated by the interrupting level.




| | | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|---|---|
| REQUEST (PR) PRIORITIES | | 0-7 $\overline{EN}$ | 8-15 $\overline{EN}$ | $\overline{A_2}$ | $\overline{A_1}$ | $\overline{A_0}$ | 0 | 0 | 0 |
| LOWEST | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 2 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| | 3 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 4 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| | 5 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| | 6 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| | 7 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 8 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| | 9 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 10 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| | 11 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 12 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| | 13 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| | 14 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| HIGHEST | 15 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Shown in the figure above is a chart of the 16 different array values that are used to offset the Program Counter and vector to the proper service routine. These values are the ones that are loaded into the "L" register; the value loaded into the "H" register with an "immediate instruction" is used to identify the major area of memory where the service routines are stored, similar to a "course setting" and the value in the "L" register is used to identify a specific location, similar to a "fine setting".

Note that D0, D1, and D2 are always set to "zero", this provides the programmer eight (8) memory locations between the start of each service routine so that maintenance of the associated Current Status Register and a JUMP or CALL instruction can be implemented.

This method of interrupt control can be almost indefinitely expanded and provides the system designer with a powerful tool to enhance total system throughput.

## D.C. AND OPERATING CHARACTERISTICS

### ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias . . . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . -65°C to +150°C
All Output and Supply Voltages . . . . . . . . -0.5V to +7V
All Input Voltages . . . . . . . . -1.0V to +5.5V
Output Currents . . . . . . . . 100 mA

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

$T_A$ = 0°C to +70°C, $V_{CC}$ = 5V ±5%.

| Symbol | Parameter | | Limits | | | Unit | Conditions |
|---|---|---|---|---|---|---|---|
| | | | Min. | Typ.[1] | Max. | | |
| $V_C$ | Input Clamp Voltage (all inputs) | | | | -1.0 | V | $I_C$=-5mA |
| $I_F$ | Input Forward Current: | ETLG input | | -.15 | -0.5 | mA | $V_F$=0.45V |
| | | all other inputs | | -.08 | -0.25 | mA | |
| $I_R$ | Input Reverse Current: | ETLG input | | | 80 | μA | $V_R$=5.25V |
| | | all other inputs | | | 40 | μA | |
| $V_{IL}$ | Input LOW Voltage: | all inputs | | | 0.8 | V | $V_{CC}$=5.0V |
| $V_{IH}$ | Input HIGH Voltage: | all inputs | 2.0 | | | V | $V_{CC}$=5.0V |
| $I_{CC}$ | Power Supply Current | | | 90 | 130 | mA | See Note 2. |
| $V_{OL}$ | Output LOW Voltage: | all outputs | | .3 | .45 | V | $I_{OL}$=15mA |
| $V_{OH}$ | Output HIGH Voltage: | ENLG output | 2.4 | 3.0 | | V | $I_{OH}$=-1mA |
| $I_{OS}$ | Short Circuit Output Current: ENLG output | | -20 | -35 | -55 | mA | $V_{OS}$=0V, $V_{CC}$=5.0V |
| $I_{CEX}$ | Output Leakage Current: $\overline{INT}$ and $\overline{A_0}$-$\overline{A_2}$ | | | | 100 | μA | $V_{CEX}$=5.25V |

NOTES:
1. Typical values are for $T_A$ = 25°C, $V_{CC}$ = 5.0V.
2. $B_0$-$B_2$, $\overline{SGS}$, CLK, $\overline{R_0}$-$\overline{R_4}$ grounded, all other inputs and all outputs open.

## A.C. CHARACTERISTICS AND WAVEFORMS $T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$

| Symbol | Parameter | Limits Min. | Limits Typ.[1] | Limits Max. | Unit |
|---|---|---|---|---|---|
| $t_{CY}$ | $\overline{CLK}$ Cycle Time | 80 | 50 | | ns |
| $t_{PW}$ | $\overline{CLK}$, $\overline{ECS}$, $\overline{INT}$ Pulse Width | 25 | 15 | | ns |
| $t_{ISS}$ | INTE Setup Time to $\overline{CLK}$ | 16 | 12 | | ns |
| $t_{ISH}$ | INTE Hold Time after $\overline{CLK}$ | 20 | 10 | | ns |
| $t_{ETCS}$[2] | ETLG Setup Time to $\overline{CLK}$ | 25 | 12 | | ns |
| $t_{ETCH}$[2] | ETLG Hold Time After $\overline{CLK}$ | 20 | 10 | | ns |
| $t_{ECCS}$[2] | $\overline{ECS}$ Setup Time to CLK | 80 | 50 | | ns |
| $t_{ECCH}$[3] | $\overline{ECS}$ Hold Time After CLK | 0 | | | ns |
| $t_{ECRS}$[3] | $\overline{ECS}$ Setup Time to $\overline{CLK}$ | 110 | 70 | | ns |
| $t_{ECRH}$[3] | $\overline{ECS}$ Hold Time After $\overline{CLK}$ | 0 | | | |
| $t_{ECSS}$[2] | $\overline{ECS}$ Setup Time to $\overline{CLK}$ | 75 | 70 | | ns |
| $t_{ECSH}$[2] | $\overline{ECS}$ Hold Time After $\overline{CLK}$ | 0 | | | ns |
| $t_{DCS}$[2] | $\overline{SGS}$ and $\overline{B_0}$-$\overline{B_2}$ Setup Time to $\overline{CLK}$ | 70 | 50 | | ns |
| $t_{DCH}$[2] | $\overline{SGS}$ and $\overline{B_0}$-$\overline{B_2}$ Hold Time After $\overline{CLK}$ | 0 | | | ns |
| $t_{RCS}$[3] | $\overline{R_0}$-$\overline{R_7}$ Setup Time to $\overline{CLK}$ | 90 | 55 | | ns |
| $t_{RCH}$[3] | $\overline{R_0}$-$\overline{R_7}$ Hold Time After $\overline{CLK}$ | 0 | | | ns |
| $t_{ICS}$ | $\overline{INT}$ Setup Time to CLK | 55 | 35 | | ns |
| $t_{CI}$ | $\overline{CLK}$ to $\overline{INT}$ Propagation Delay | | 15 | 25 | ns |
| $t_{RIS}$[4] | $\overline{R_0}$-$\overline{R_7}$ Setup Time to $\overline{INT}$ | 10 | 0 | | ns |
| $t_{RIH}$[4] | $\overline{R_0}$-$\overline{R_7}$ Hold Time After $\overline{INT}$ | 35 | 20 | | ns |
| $t_{RA}$ | $\overline{R_0}$-$\overline{R_7}$ to $\overline{A_0}$-$\overline{A_2}$ Propagation Delay | | 80 | 100 | ns |
| $t_{ELA}$ | $\overline{ELR}$ to $\overline{A_0}$-$\overline{A_2}$ Propagation Delay | | 40 | 55 | ns |
| $t_{ECA}$ | $\overline{ECS}$ to $\overline{A_0}$-$\overline{A_2}$ Propagation Delay | | 100 | 120 | ns |
| $t_{ETA}$ | ETLG to $\overline{A_0}$-$\overline{A_2}$ Propagation Delay | | 35 | 70 | ns |
| $t_{DECS}$[4] | $\overline{SGS}$ and $\overline{B_0}$-$\overline{B_2}$ Setup Time to $\overline{ECS}$ | 15 | 10 | | ns |
| $t_{DECH}$[4] | $\overline{SGS}$ and $\overline{B_0}$-$\overline{B_2}$ Hold Time After $\overline{ECS}$ | 15 | 10 | | ns |
| $t_{REN}$ | $\overline{R_0}$-$\overline{R_7}$ to ENLG Propagation Delay | | 45 | 70 | ns |
| $t_{ETEN}$ | ETLG to ENLG Propagation Delay | | 20 | 25 | ns |
| $t_{ECRN}$ | $\overline{ECS}$ to ENLG Propagation Delay | | 85 | 90 | ns |
| $t_{ECSN}$ | $\overline{ECS}$ to ENLG Propagation Delay | | 35 | 55 | ns |

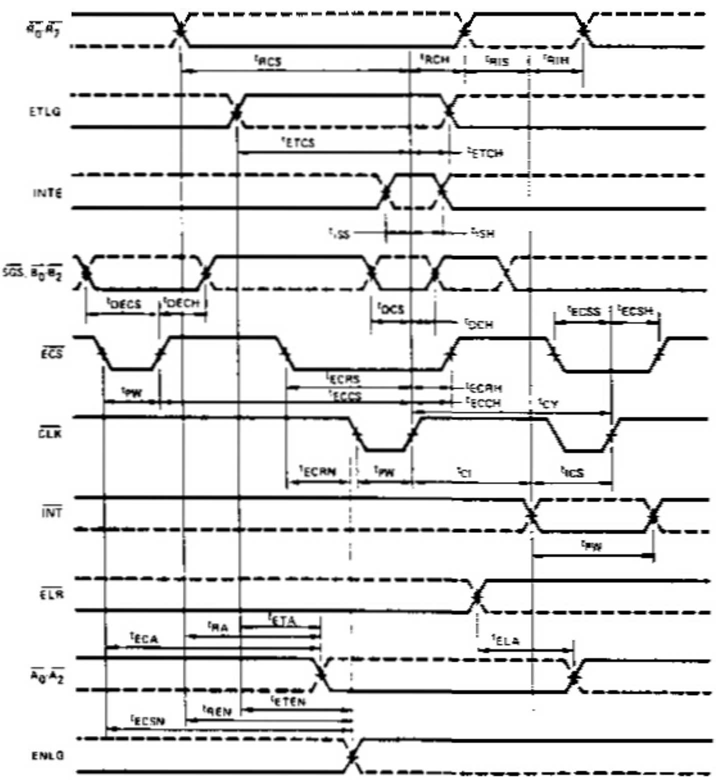
### CAPACITANCE (5)

| Symbol | Parameter | Limits Min. | Limits Typ.[1] | Limits Max | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | | 5 | 10 | pF |
| $C_{OUT}$ | Output Capacitance | | 7 | 12 | pF |

**TEST CONDITIONS:** $V_{BIAS} = 2.5V$, $V_{CC} = 5V$, $T_A = 25°C$, f = 1 MHz

NOTE 5. This parameter is periodically sampled and not 100% tested.

## WAVEFORMS



NOTES:

(1) Typical values are for $T_A = 25°C$, $V_{CC} = 5.0V$.

(2) Required for proper operation if ISE is enabled during next clock pulse.

(3) These times are not required for proper operation but for desired change in interrupt flip-flop.

(4) Required for new request or status to be properly loaded.

**TEST CONDITIONS:**

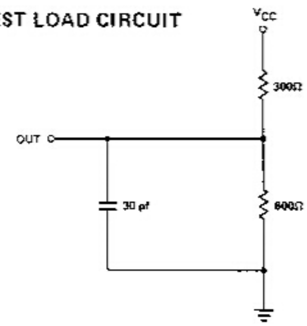
Input pulse amplitude: 2.5 volts.

Input rise and fall times: 5 ns between 1 and 2 volts.

Output loading of 15 mA and 30 pf.

Speed measurements taken at the 1.5V levels.

**TEST LOAD CIRCUIT**



# Schottky Bipolar 8216/8226

## 4 BIT PARALLEL BIDIRECTIONAL BUS DRIVER

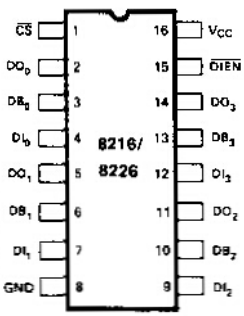
- **Data Bus Buffer Driver for 8080 CPU**
- **Low Input Load Current — .25 mA Maximum**
- **High Output Drive Capability for Driving System Data Bus**
- **3.65V Output High Voltage for Direct Interface to 8080 CPU**
- **Three State Outputs**
- **Reduces System Package Count**

The 8216/8226 is a 4-bit bi-directional bus driver/receiver.

All inputs are low power TTL compatible. For driving MOS, the DO outputs provide a high 3.65V $V_{OH}$, and for high capacitance terminated bus structures, the DB outputs provide a high 50mA $I_{OL}$ capability.

A non-inverting (8216) and an inverting (8226) are available to meet a wide variety of applications for buffering in microcomputer systems.

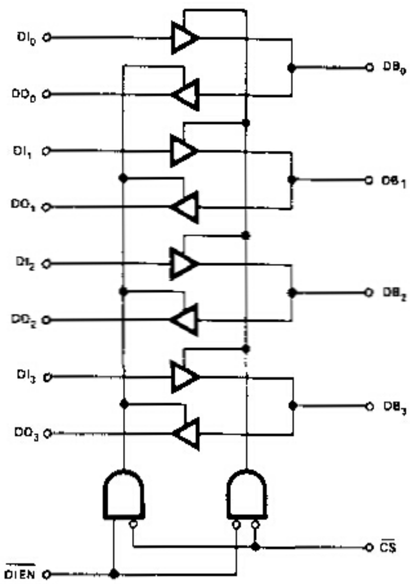
**PIN CONFIGURATION**



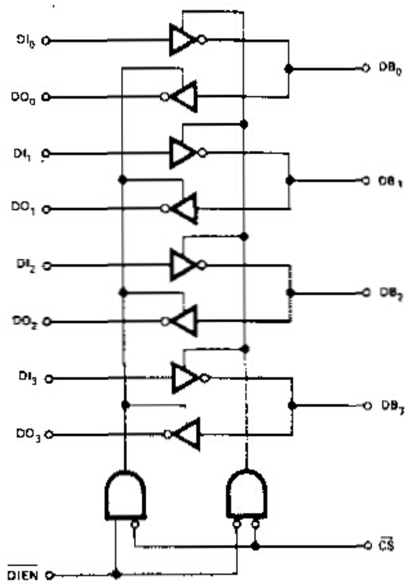

**PIN NAMES**

| | |
|---|---|
| DB0-DB3 | DATA BUS BI-DIRECTIONAL |
| DI0-DI3 | DATA INPUT |
| DO0-DO3 | DATA OUTPUT |
| DIEN | DATA IN ENABLE DIRECTION CONTROL |
| CS | CHIP SELECT |

**LOGIC DIAGRAM 8216**




**LOGIC DIAGRAM 8226**



## FUNCTIONAL DESCRIPTION

Microprocessors like the 8080 are MOS devices and are generally capable of driving a single TTL load. The same is true for MOS memory devices. While this type of drive is sufficient in small systems with few components, quite often it is necessary to buffer the microprocessor and memories when adding components or expanding to a multi-board system.

The 8216/8226 is a four bit bi-directional bus driver specifically designed to buffer microcomputer system components.

### Bi-Directional Driver

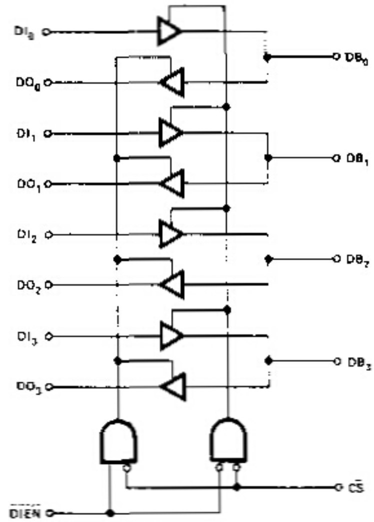
Each buffered line of the four bit driver consists of two separate buffers that are tri-state in nature to achieve direct bus interface and bi-directional capability. On one side of the driver the output of one buffer and the input of another are tied together (DB), this side is used to interface to the system side components such as memories, I/O, etc., because its interface is direct TTL compatible and it has high drive (50mA). On the other side of the driver the inputs and outputs are separated to provide maximum flexibility. Of course, they can be tied together so that the driver can be used to buffer a true bi-directional bus such as the 8080 Data Bus. The DO outputs on this side of the driver have a special high voltage output drive capability (3.65V) so that direct interface to the 8080 and 8008 CPUs is achieved with an adequate amount of noise immunity (350mV worst case).

### Control Gating $\overline{\text{DIEN}}$, $\overline{\text{CS}}$

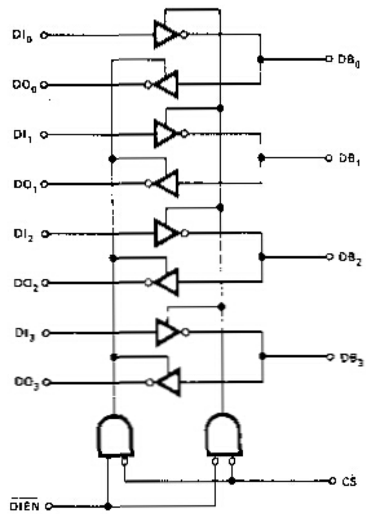
The $\overline{\text{CS}}$ input is actually a device select. When it is "high" the output drivers are all forced to their high-impedance state. When it is at "zero" the device is selected (enabled) and the direction of the data flow is determined by the $\overline{\text{DIEN}}$ input.

The $\overline{\text{DIEN}}$ input controls the direction of data flow (see Figure 1) for complete truth table. This direction control is accomplished by forcing one of the pair of buffers into its high impedance state and allowing the other to transmit its data. A simple two gate circuit is used for this function.

The 8216/8226 is a device that will reduce component count in microcomputer systems and at the same time enhance noise immunity to assure reliable, high performance operation.




(a) 8216




(b) 8226

| $\overline{\text{DIEN}}$ | $\overline{\text{CS}}$ | |
|---|---|---|
| 0 | 0 | DI → DB |
| 1 | 0 | DB → DO |
| 0 | 1 | HIGH IMPEDANCE |
| 1 | 1 | |

Figure 1. 8216/8226 Logic Diagrams

# SCHOTTKY BIPOLAR 8216/8226

## APPLICATIONS OF 8216/8226

### 8080 Data Bus Buffer

The 8080 CPU Data Bus is capable of driving a single TTL load and is more than adequate for small, single board systems. When expanding such a system to more than one board to increase I/O or Memory size, it is necessary to provide a buffer. The 8216/8226 is a device that is exactly fitted to this application.

Shown in Figure 2 are a pair of 8216/8226 connected directly to the 8080 Data Bus and associated control signals. The buffer is bi-directional in nature and serves to isolate the CPU data bus.

On the system side, the DB lines interface with standard semiconductor I/O and Memory components and are completely TTL compatible. The DB lines also provide a high drive capability (50mA) so that an extremely large system can be dirven along with possible bus termination networks.

On the 8080 side the DI and DO lines are tied together and are directly connected to the 8080 Data Bus for bi-directional operation. The DO outputs of the 8216/8226 have a high voltage output capability of 3.65 volts which allows direct connection to the 8080 whose minimum input voltage is 3.3 volts. It also gives a very adequate noise margin of 350mV (worst case).

The $\overline{\text{DIEN}}$ inputs to 8216/8226 is connected directly to the 8080. $\overline{\text{DIEN}}$ is tied to DBIN so that proper bus flow is maintained, and $\overline{\text{CS}}$ is tied to $\overline{\text{BUSEN}}$ so that the system side Data Bus will be 3-stated when a Hold request has been acknowledged during a DMA activity.

### Memory and I/O Interface to a Bi-directional Bus

In large microcomputer systems it is often necessary to provide Memory and I/O with their own buffers and at the same time maintain a direct, common interface to a bi-directional Data Bus. The 8216/8226 has separated data in and data out lines on one side and a common bi-directional set on the other to accomodate such a function.

Shown in Figure 3 is an example of how the 8216/8226 is used in this type of application.

The interface to Memory is simple and direct. The memories used are typically Intel® 8102, 8102A, 8101 or 8107B-4 and have separate data inputs and outputs. The DI and DO lines of the 8216/8226 tie to them directly and under control of the $\overline{\text{MEMR}}$ signal, which is connected to the $\overline{\text{DIEN}}$ input, an interface to the bi-directional Data Bus is maintained.

The interface to I/O is similar to Memory. The I/O devices used are typically Intel® 8255s, and can be used for both input and output ports. The $\overline{\text{I/O R}}$ signal is connected directly to the $\overline{\text{DIEN}}$ input so that proper data flow from the I/O device to the Data Bus is maintained.

The 8216/8226 can be used in a wide variety of other buffering functions in microcomputer systems such as Address Bus Drivers, Drivers to peripheral devices such as printers, and as Drivers for long length cables to other peripherals or systems.

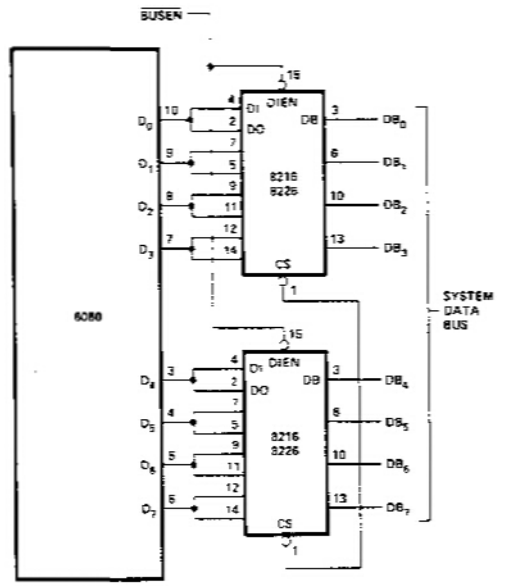



Figure 2. 8080 Data Bus Buffer.

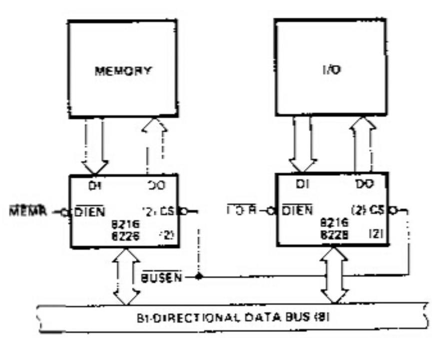



Figure 3. Memory and I/O Interface to a Bi-Directional Bus.

## D.C. AND OPERATING CHARACTERISTICS

### ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias . . . . . . . . 0°C to 70°C
Storage Temperature . . . . . . . . -65°C to +150°C
All Output and Supply Voltages . . . . . . . . -0.5V to +7V
All Input Voltages . . . . . . . . -1.0V to +5.5V
Output Currents . . . . . . . . 125 mA

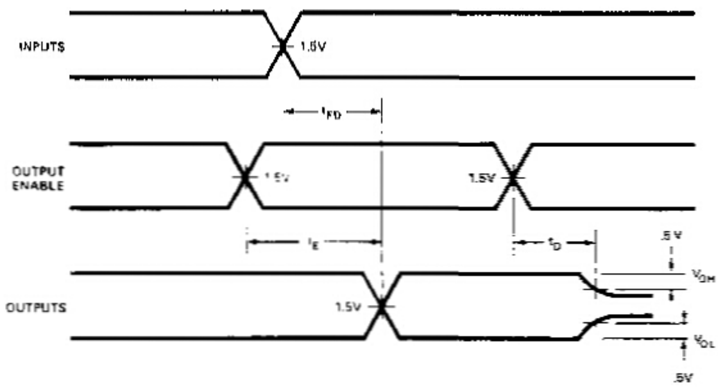
*COMMENT: Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied.

$T_A$ = 0°C to +70°C, $V_{CC}$=+5V ±5%

| Symbol | Parameter | | Limits Min. | Typ. | Max. | Unit | Conditions |
|---|---|---|---|---|---|---|---|
| $I_{F1}$ | Input Load Current $\overline{DIEN}$, $\overline{CS}$ | | | -0.15 | -.5 | mA | $V_F$ = 0.45 |
| $I_{F2}$ | Input Load Current All Other Inputs | | | -0.08 | -.25 | mA | $V_F$ = 0.45 |
| $I_{R1}$ | Input Leakage Current $\overline{DIEN}$, $\overline{CS}$ | | | | 20 | µA | $V_R$ = 5.25V |
| $I_{R2}$ | Input Leakage Current DI Inputs | | | | 10 | µA | $V_R$ = 5.25V |
| $V_C$ | Input Forward Voltage Clamp | | | | -1 | V | $I_C$ = -5mA |
| $V_{IL}$ | Input "Low" Voltage | | | | .95 | V | |
| $V_{IH}$ | Input "High" Voltage | | 2.0 | | | V | |
| $\|I_O\|$ | Output Leakage Current (3-State) | DO<br>DB | | | 20<br>100 | µA | $V_O$ = 0.45V/5.25V |
| $I_{CC}$ | Power Supply Current | 8216 | | 95 | 130 | mA | |
| | | 8226 | | 85 | 120 | mA | |
| $V_{OL1}$ | Output "Low" Voltage | | | 0.3 | .45 | V | DO Outputs $I_{OL}$=15mA<br>DB Outputs $I_{OL}$=25mA |
| $V_{OL2}$ | Output "Low" Voltage | 8216 | | 0.5 | .6 | V | DB Outputs $I_{OL}$=55mA |
| | | 8226 | | 0.5 | .6 | V | DB Outputs $I_{OL}$=50mA |
| $V_{OH1}$ | Output "High" Voltage | | 3.65 | 4.0 | | V | DO Outputs $I_{OH}$ = -1mA |
| $V_{OH2}$ | Output "High" Voltage | | 2.4 | 3.0 | | V | DB Outputs $I_{OH}$ = -10mA |
| $I_{OS}$ | Output Short Circuit Current | | -15<br>-30 | -35<br>-75 | -65<br>-120 | mA<br>mA | DO Outputs $V_O \cong$ 0V,<br>DB Outputs $V_{CC}$=5.0V |

NOTE: Typical values are for $T_A$ = 25°C, $V_{CC}$ = 5.0V.

## WAVEFORMS




## A.C. CHARACTERISTICS

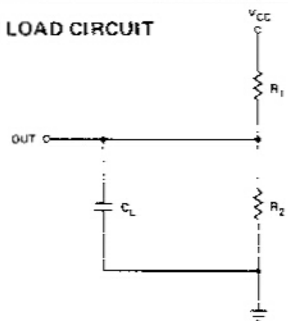
$T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$

| Symbol | Parameter | | Limits Min. | Typ.[1] | Max. | Unit | Conditions |
|---|---|---|---|---|---|---|---|
| $T_{PD1}$ | Input to Output Delay DO Outputs | | | 15 | 25 | ns | $C_L$=30pF, $R_1$=300Ω $R_2$=600Ω |
| $T_{PD2}$ | Input to Output Delay DB Outputs | 8216 | | 20 | 30 | ns | $C_L$=300pF, $R_1$=90Ω |
| | | 8226 | | 16 | 25 | ns | $R_2$ = 180Ω |
| $T_E$ | Output Enable Time | 8216 | | 45 | 65 | ns | (Note 2) |
| | | 8226 | | 35 | 54 | ns | (Note 3) |
| $T_D$ | Output Disable Time | | | 20 | 35 | ns | (Note 4) |

### TEST CONDITIONS:

Input pulse amplitude of 2.5V.
Input rise and fall times of 5 ns between 1 and 2 volts.
Output loading is 5 mA and 10 pF.
Speed measurements are made at 1.5 volt levels.

### TEST LOAD CIRCUIT




### Capacitance[5]

| Symbol | Parameter | Limits Min. | Typ.[1] | Max. | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | | 4 | 8 | pF |
| $C_{OUT1}$ | Output Capacitance | | 6 | 10 | pF |
| $C_{OUT2}$ | Output Capacitance | | 13 | 18 | pF |

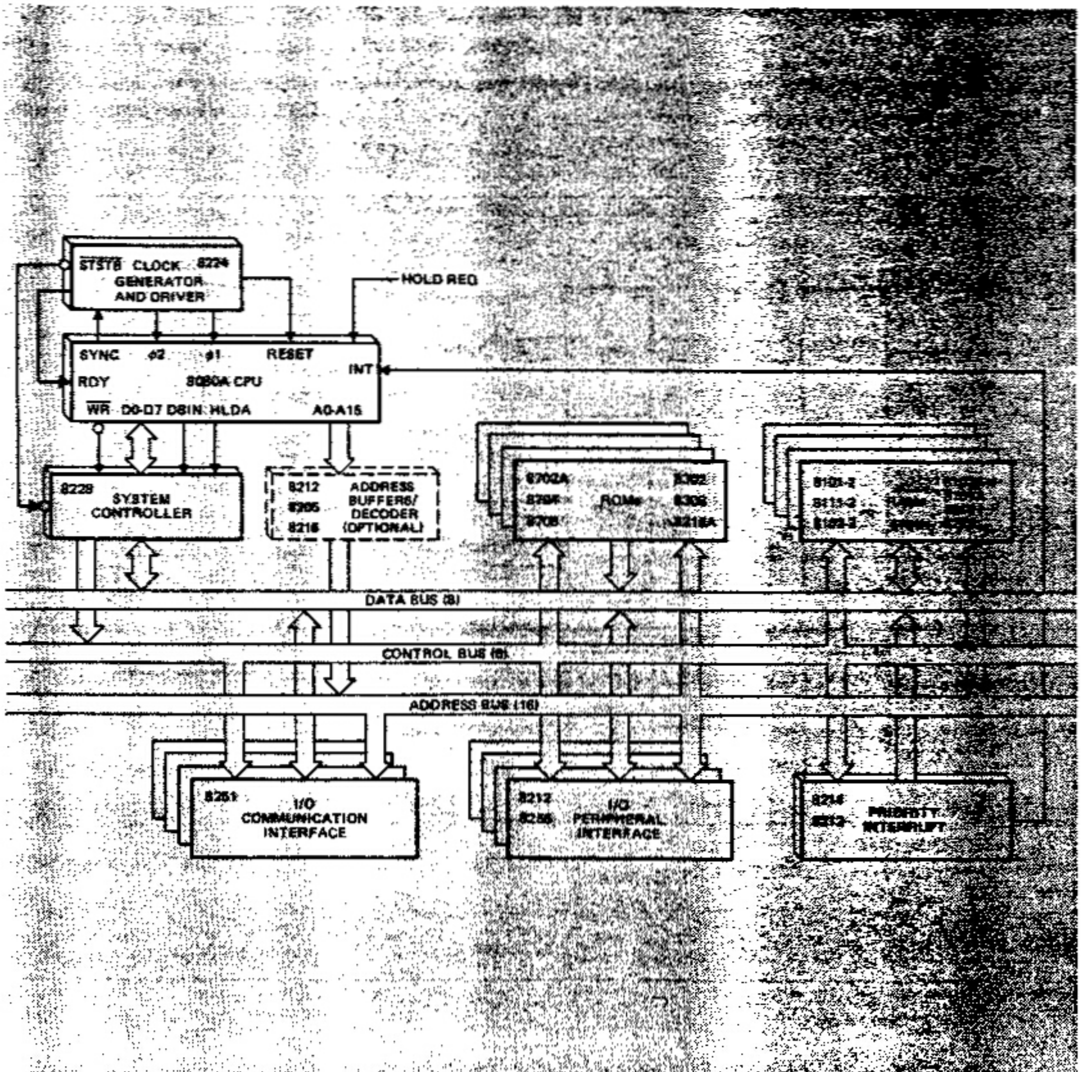
**TEST CONDITIONS:** $V_{BIAS}$ = 2.5V, $V_{CC}$ = 5.0V, $T_A$ = 25°C, f = 1 MHz.

NOTES:
1. Typical values are for $T_A$ = 25°C, $V_{CC}$ = 5.0V.
2. DO Outputs, $C_L$ = 30pF, $R_1$ = 300/10 KΩ, $R_2$ = 180/1KΩ; DB Outputs, $C_L$ = 300pF, $R_1$ = 90/10 KΩ, $R_2$ = 180/1 KΩ.
3. DO Outputs, $C_L$ = 30pF, $R_1$ = 300/10 KΩ, $R_2$ = 600/1K; DB Outputs, $C_L$ = 300pF, $R_1$ = 90/10 KΩ, $R_2$ = 180/1 KΩ.
4. DO Outputs, $C_L$ = 5pF, $R_1$ = 300/10 KΩ, $R_2$ = 600/1 KΩ; DB Outputs, $C_L$ = 5pF, $R_1$ = 90/10 KΩ, $R_2$ = 180/1 KΩ.
5. This parameter is periodically sampled and not 100% tested.

intel® Microcomputer Systems

# Coming Soon

8253
8257
8259



# Silicon Gate MOS 8253

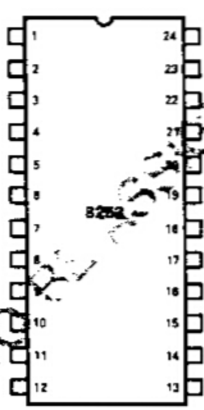
## PROGRAMMABLE INTERVAL TIMER

- 3 Independent 16-Bit Counters
- DC to 3 MHz
- Programmable Counter Modes
- Count Binary or BCD
- Single +5V Supply
- 24 Pin Dual-in-line Package

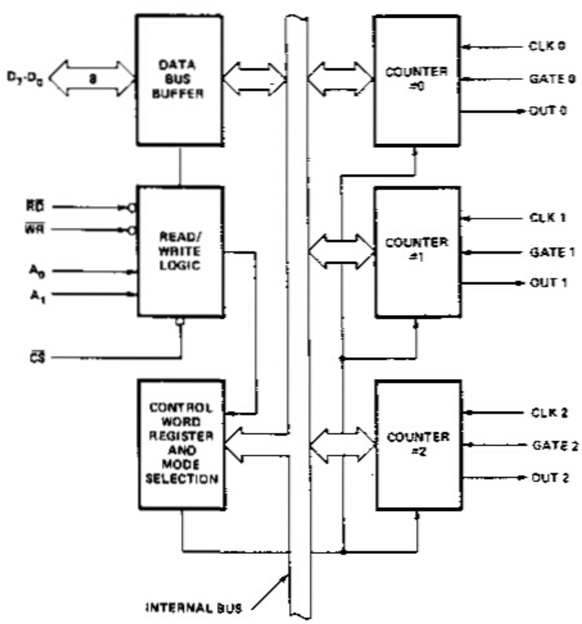
The 8253 is a programmable counter/timer chip designed for use as an 8080 (or 8008) peripheral. It uses nMOS technology with a single +5V supply and is packaged in a 24-pin plastic DIP.

It is organized as three independent 16-bit counters, each with a count rate from 0Hz to 3MHz. All modes of operation are software programmable by the 8080.

PIN CONFIGURATION




BLOCK DIAGRAM



## 8253 PRELIMINARY FUNCTIONAL DESCRIPTION

In Microcomputer-based systems the most common interface is to a mechanical device such as a printer head or stepper motor. All such devices have inherent delays that must be accounted for if accurate and reliable performance is to be achieved. The systems software allows for such delays by programmed timing loops. This type of programming requires significant overhead and maintenance of multiple loops gets extremely complicated.

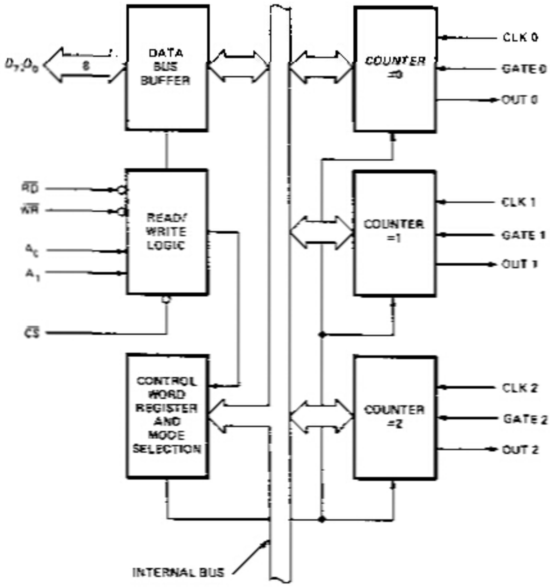
The 8253 Programmable Interval Timer is a single chip solution to system timing problems. In essence, it is a group of three 16-bit counters that are independent in nature but driven commonly as I/O peripheral ports. Instead of setting up timing loops in the system software, the programmer configures the 8253 to match his requirements. The programmer initializes one of the three counters of the 8253 with the quantity and mode desired then, upon command, the 8253 will count out the delay and interrupt the microcomputer when it has finished its task. It is easy to see that the software overhead is minimal and that multiple delays can be easily maintained by assigned interrupt levels to different counters. Other functions that are non-delay in nature and require counters can also be implemented with the 8253.

- Programmable Baud Rate Generator
- Event Counter
- Binary Rate Multiplier
- Real Time Clock

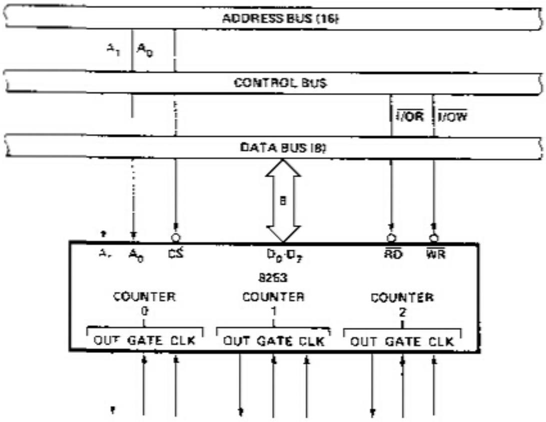
### System Interface

The 8253 is a component of the MCS-80 system and interfaces in the same manner as all other peripherals of the family. It is treated by the systems software as an array of I/O ports; three are counters and the fourth is a control register for programming. The OUT lines of each counter would normally be tied to the interrupt request inputs of the 8259.

The 8253 represents a significant improvement for solving one of the most common problems in system design and reducing software overhead.




8253 Block Diagram.




8253 System Interface.

intel®

# Silicon Gate MOS 8257

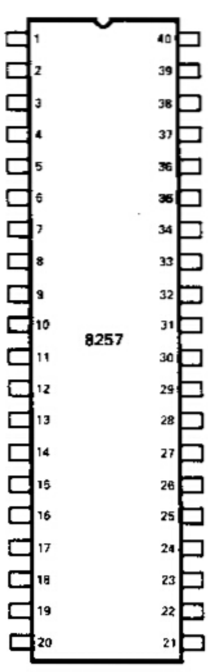
## PROGRAMMABLE DMA CONTROLLER

- **Four Channel DMA Controller**
- **Priority DMA Request Logic**
- **Channel Inhibit Logic**
- **Terminal and Modulo 256/128 Outputs**
- **Auto Load Mode**
- **Single TTL Clock (ϕ2/TTL)**
- **Single +5V Supply**
- **Expandable**
- **40 Pin Dual-in-Line Package**

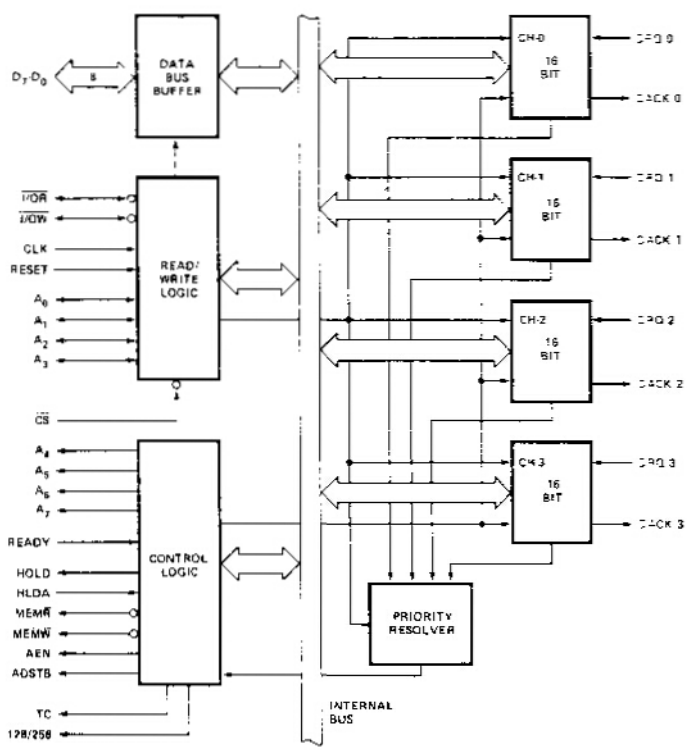
The 8257 is a Direct Memory Access (DMA) Chip which has four channels for use in 8080 microcomputer systems. Its primary function is to generate, upon a peripheral request, a sequential memory address which will allow the peripheral to access or deposit data directly from or to memory. It uses the Hold feature of the 8080 to acquire the system bus. It also keeps count of the number of DMA cycles for each channel and notifies the peripheral when a programmable terminal count has been reached. Other features that it has are two mode priority logic to resolve the request among the four channels, programmable channel inhibit logic, an early write pulse option, a modulo 256/128 Mark output for sectored data transfers, an automatic load mode, a terminal count status register, and control signal timing generation during DMA cycles. There are three types of DMA cycles: Read DMA Cycle, Write DMA Cycle and Verify DMA Cycle.

The 8257 is a 40-pin, N-channel MOS chip which uses a single +5V supply and the ϕ2 (TTL) clock of the 8080 system. It is designed to work in conjunction with a single 8212 8-bit, three-state latch chip. Multiple DMA chips can be used to expand the number of channels with the aid of the 8214 Priority Interrupt Chip.

### PIN CONFIGURATION




### BLOCK DIAGRAM



## 8257 PRELIMINARY FUNCTIONAL DESCRIPTION

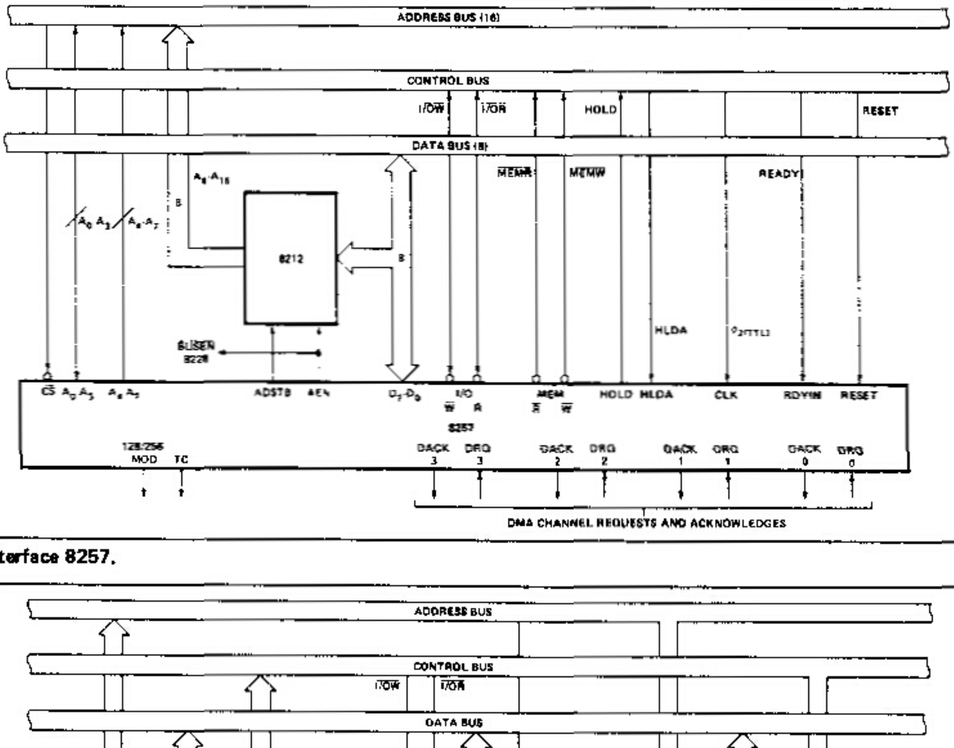
The transfer of data between a mass storage device such as a floppy disk or mag cassette and system RAM memory is often limited by the speed of the microprocessor. Removing the processor during such a transfer and letting an auxillary device manage the transfer in a more efficient manner would greatly improve the speed and make mass storage devices more attractive, even to the small system designer.

The transfer technique is called DMA (Direct Memory Access); in essence the CPU is idled so that it no longer has control of the system bus and a DMA controller takes over to manage the transfer.

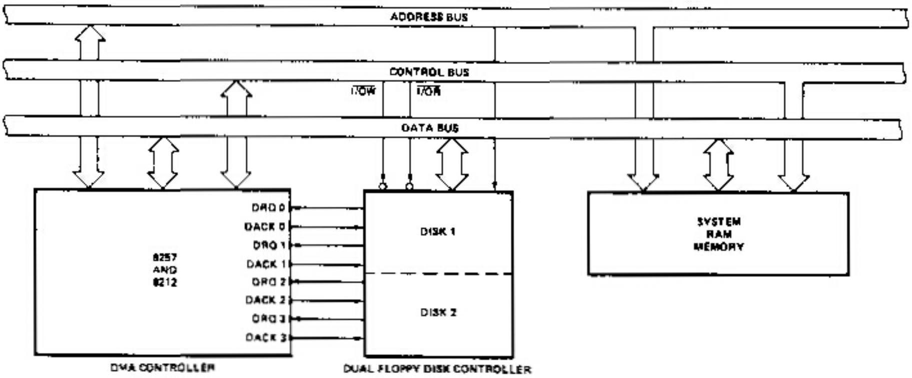
The 8257 Programmable DMA Controller is a single chip, four channel device that can efficiently manage DMA activities. Each channel is assigned a priority level so that if multi-DMA activities are required each mass storage device can be serviced, based on its importance in the system. In operation, a request is made from a peripheral device for access to the system bus. After its priority is accepted a HOLD command is ussued to the CPU, the CPU issues a HLDA and that DMA channel has complete control of the system bus. Transfers can be made in blocks, suspending the processors operation during the entire transfer or, the transfer can be made a few bytes at a time, hidden in the execution states of each instruction cycle, (cycle-stealing).

The modes and priority resolving are maintained by the system software as well as initializing each channel as to the starting address and length of transfer.

The system interface is similar to the other peripherals of the MCS-80 but an additional 8212 is necessary to control the entire address bus. A special control signal $\overline{\text{BUSEN}}$ is connected directly to the 8228 so that the data bus and control bus will be released at the proper time.




**System Interface 8257.**




**System Application of 8257.**

# Silicon Gate MOS 8259

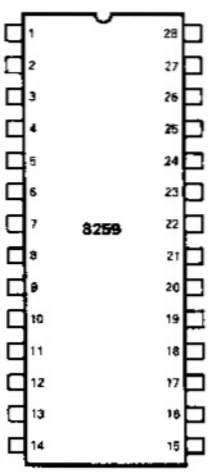
## PROGRAMMABLE INTERRUPT CONTROLLER

- **Eight Level Priority Controller**
- **Expandable to 64 Levels**
- **Programmable Interrupt Modes (Algorithms)**
- **Individual Request Mask Capability**
- **Single +5V Supply (No Clocks)**
- **28 Pin Dual-in-Line Package**

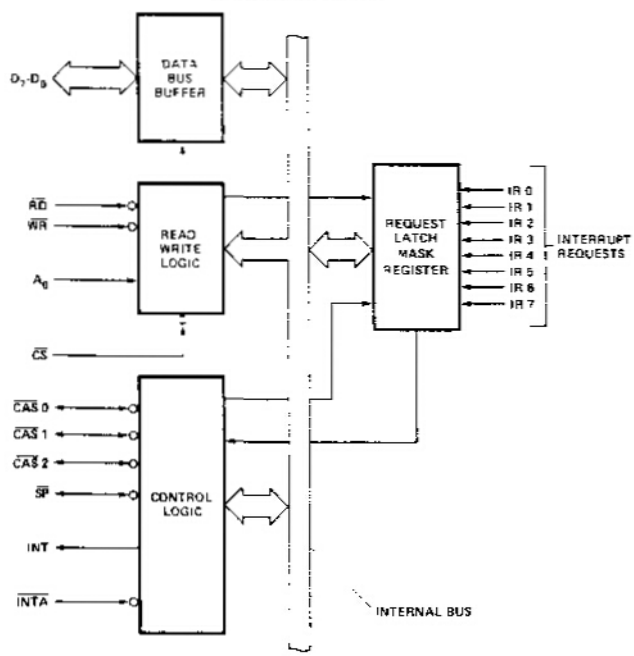
The 8259 handles up to eight vectored priority interrupts for the 8080A CPU. It is cascadable for up to 64 vectored priority interrupts, without additional circuitry. It will be packaged in a 28-pin plastic DIP, uses nMOS technology and requires a single +5V supply. Circuitry is static, requiring no clock input.

The 8259 is designed to minimize the software and real time overhead in handling multi-level priority interrupts. It has several modes, permitting optimization for a variety of system requirements.

PIN CONFIGURATION




BLOCK DIAGRAM

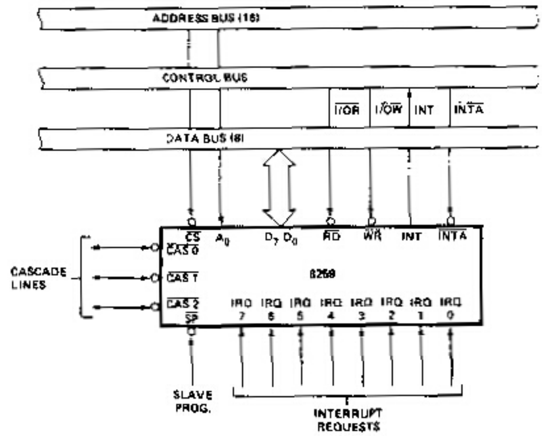


## 8259 PRELIMINARY FUNCTIONAL DESCRIPTION

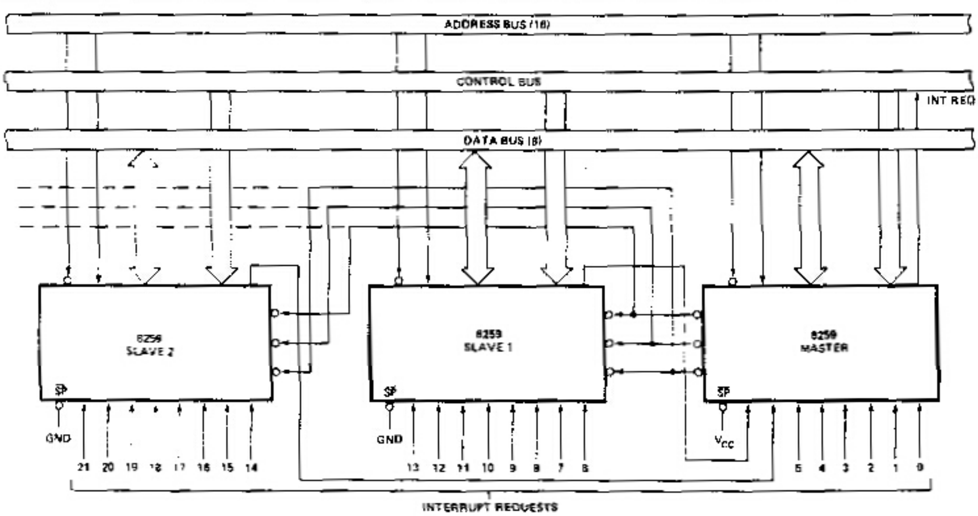
In microcomputer systems, the rate at which a peripheral device or devices can be serviced determines the total amount of system tasks that can be assigned to the control of the microprocessor. The higher the throughput the more jobs the microcomputer can do and the more cost effective it becomes. Interrupts have long been accepted as a key to improving system throughput by servicing a peripheral device only when the device has requested it to do so. Efficient managing of the interrupt requests to the CPU will have a significant effect on the overall cost effectiveness of the microcomputer system.

The 8259 Programmable Interrupt Controller is a single-chip device that can manage eight levels of requests and has built-in features for expandability to other 8259s (up to 64 levels). It is programmed by the systems software as an I/O peripheral. A selection of priority algorithms is available to the programmer so that the manner in which the requests are processed by the 8259 can be configured to match his system requirements. The priority assignments and algorithms can be changed or reconfigured dynamically at any time during the main program. This means that the complete interrupt structure can be defined as required, based on the total system environment.

The system interface is the same as other peripheral devices in the MCS-80. A special input is provided ($\overline{SP}$) to program the 8259 as a slave or master device when expanding to more than eight levels. Basically the master accepts INT inputs from the slaves and issues a composite request to the 8080A; when it receives the $\overline{INTA}$ from the 8228 it puts the first byte on the CALL on the bus. On subsequent $\overline{INTA}$s the interrupting slave puts out the address of the vector.




8259 System Interface.




Cascading the 8259 22 Level Controller (Expandable to 64 levels).

# CHAPTER 6
# PACKAGING INFORMATION

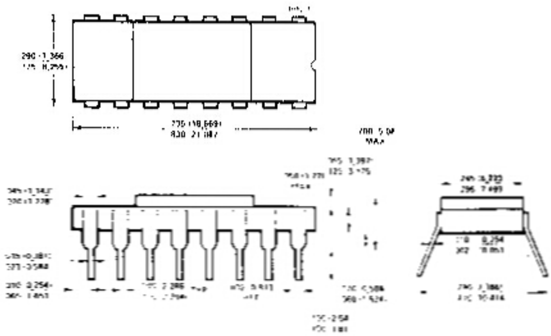
| | Intel Product Number | Standard Package Type | | | Number Of Pins | Comments |
|---|---|---|---|---|---|---|
| CPU GROUP | 8224 | C | D | P | 16 | |
| | 8228 | C | D | P | 28 | |
| | 8080A | C | | | 40 | Including 8080A-1, 8080A-2 and M8080A |
| ROMs | 8702A | C | | | 24 | |
| | 8708/4 | C | | | 24 | |
| | 8302 | C | | P | 24 | |
| | 8308 | C | | P | 24 | |
| | 8316A | C | D | P | 24 | |
| RAMs | 8101-2 | C | D | P | 22 | |
| | 8111-2 | C | D | P | 18 | |
| | 8102-2 | C | D | P | 16 | |
| | 8102A-4 | C | D | P | 16 | |
| | 8107B-4 | C | D | P | 22 | |
| | 5101 | C | D | P | 22 | |
| | 8210 | | D | P | 18 | |
| | 8222 | | D | | 22 | New Product |
| I/O | 8212 | | D | P | 24 | |
| | 8255 | C | | | 40 | |
| | 8251 | C | D | P | 28 | |
| ERIPHERAL | 8205 | C | D | P | 16 | |
| | 8214 | C | D | P | 24 | |
| | 8216/26 | | D | P | 16 | |
| COMING SOON | 8253 | | | | 24 | Coming Soon |
| | 8257 | | | | 40 | Coming Soon |
| | 8259 | | | | 28 | Coming Soon |

C = Ceramic   D = Cerdip   P = Plastic

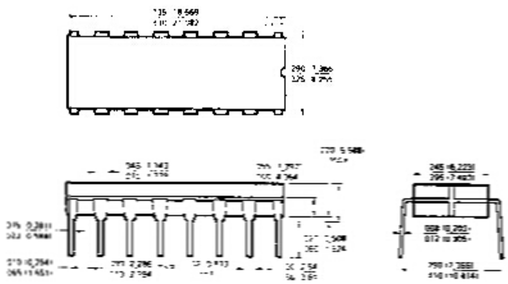
# PACKAGING INFORMATION

Dimensions in inches and (millimeters).

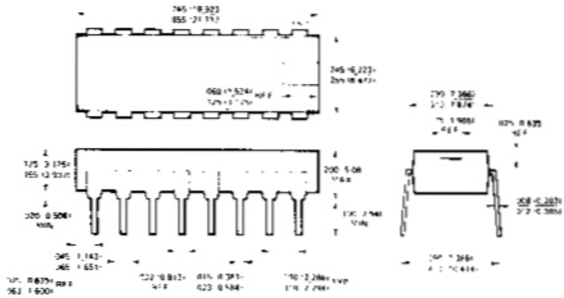
## 16-LEAD CERAMIC DUAL IN-LINE PACKAGE (C)



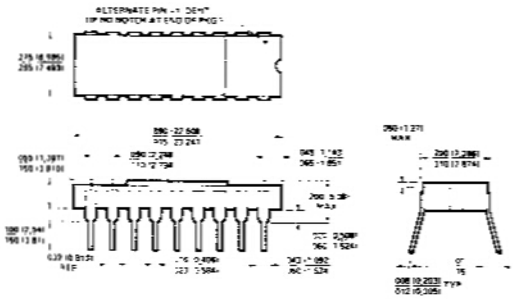
## 16-LEAD CerDIP DUAL IN-LINE PACKAGE (D)



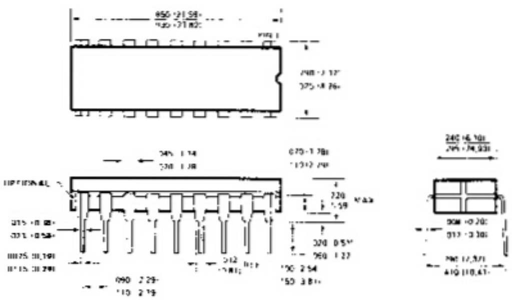
## 16-LEAD PLASTIC DUAL IN-LINE PACKAGE (P)



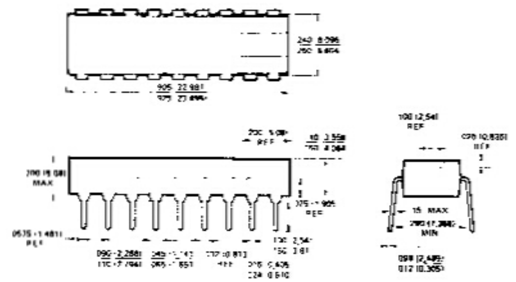
## 18-LEAD CERAMIC DUAL IN-LINE PACKAGE (C)



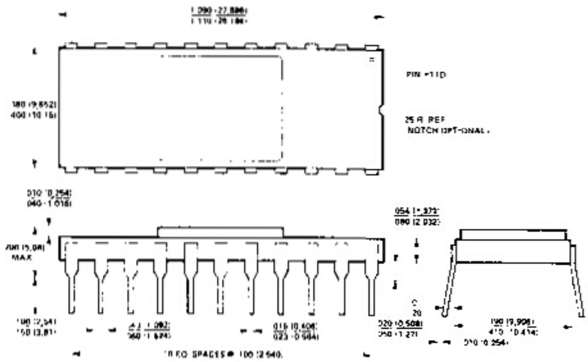
## 18-LEAD CerDIP DUAL IN-LINE PACKAGE (D)



## 18-LEAD PLASTIC DUAL IN-LINE PACKAGE (P)



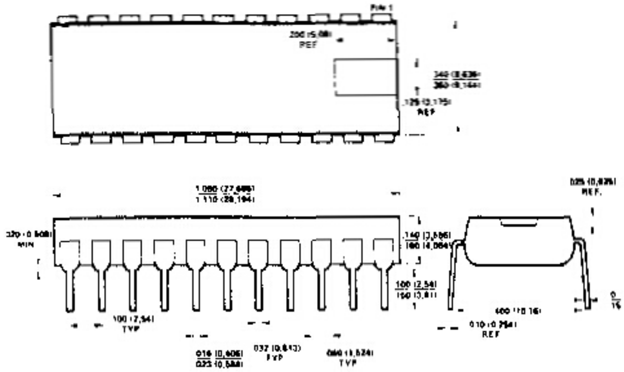
## 22-LEAD CERAMIC DUAL IN-LINE PACKAGE (C)



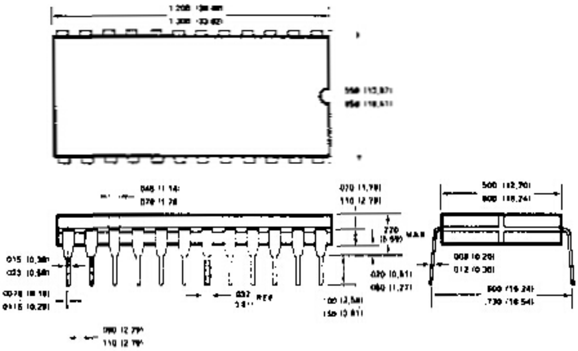
# PACKAGING INFORMATION

Dimensions in inches and (millimeters).

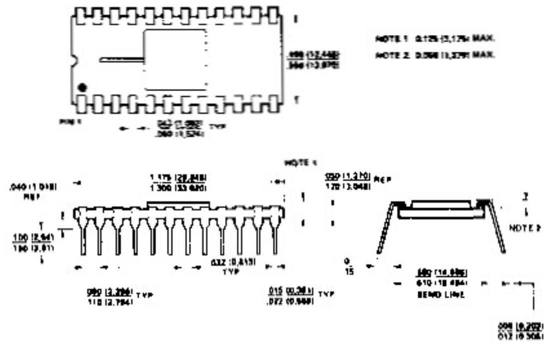
## 22-LEAD PLASTIC DUAL IN-LINE PACKAGE (P)



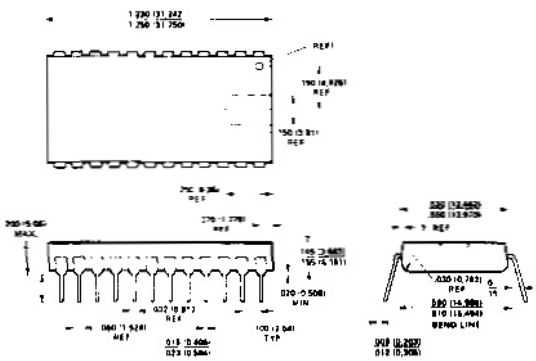
## 24-LEAD CerDIP DUAL IN-LINE PACKAGE (D)



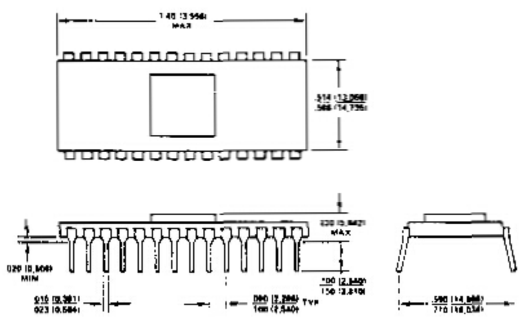
## 24-LEAD CERAMIC DUAL IN-LINE PACKAGE (C)



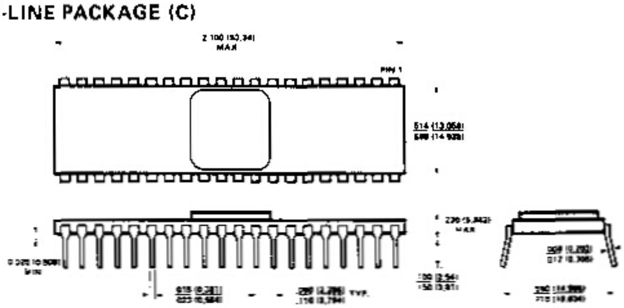
## 24-LEAD PLASTIC DUAL IN-LINE PACKAGE (P)



## 28-LEAD CERAMIC DUAL IN-LINE PACKAGE (C)



## 40-LEAD CERAMIC DUAL IN-LINE PACKAGE (C)



intel

3065 Bowers Avenue
Santa Clara, California 95051
Tel (408) 246-7501
TWX 910-338-0026
TELEX 34-6372

# SALES AND MARKETING OFFICES

## U.S. AND CANADIAN SALES OFFICES

**ALABAMA**
Barnhill and Associates
7844 Horseshoe Trail
Huntsville 35802
Tel (205) 883-9394

**ARIZONA**
Sales Engineering Inc
7155 E Thomas Road, No. 6
Scottsdale 85252
Tel (602) 945-5781
TWX 910-950-1288

**CALIFORNIA**
Intel Corp.*
990 E. Arques Ave
Suite 112
Sunnyvale 94086
Tel (408) 738-3870
TWX 910-339-9279

Mac-I
P.O. Box 1420
Cupertino 95014
Tel (408) 257-9880

Earle Associates, Inc
4433 Convoy Street
Suite A
San Diego 92111
Tel (714) 278-5441
TWX 910-335-1585

Intel Corp.*
1651 East 4th Street
Suite 228
Santa Ana 92701
Tel (714) 835-9642
TWX 910-595-1114

**COLORADO**
Intel Corp.
12075 East 45th Avenue
Suite 310
Denver 80239
Tel (303) 373-4920
TWX 910-932-0322

**FLORIDA**
Intel Corp.
1090 NE 27th Terrace
Pompano Beach 33062
Tel (305) 781-7450
TWX 510-956-9407

**FLORIDA (cont.)**
Intel Corp
5151 Adanson Street, Suite 200-3
Orlando 32804
Tel (305) 628-2393
TWX 810-853-9219

**ILLINOIS**
Intel Corp.*
900 Jorie Boulevard
Suite 138
Oakbrook 60521
Tel (312) 325-9510
TWX 910-651-5881

**IOWA**
Technical Representatives, Inc.
1703 Hillside Drive
Cedar Rapids
Tel (319) 396-5662

**KANSAS**
Technical Representatives, Inc
801 Clairborne
Olathe 66061
Tel (913) 782-1177
TWX 910-749-6412

**MARYLAND**
Barnhill and Associates
57 West Timonium Road
Timonium 21093
Tel (301) 252-7742

Intel Corp.*
57 West Timonium Road
Suite 307
Timonium 21093
Tel (301) 252-7742
TWX 710-232-1807

**MASSACHUSETTS**
Datcom
55 Moody Street
Waltham 02154
Tel (617) 891-4600
TELEX 92-3462

Intel Corp.*
187 Billerica Road Suite 14A
Chelmsford 01824
Tel (617) 861-1136
TWX 710-343-6333

**MICHIGAN**
Intel Corp
725 South Adams Road
Suite 288
Birmingham 48011
Tel (313) 642-7018
TWX 910-420-1212
TELEX 2 31143

**MINNESOTA**
Intel Corp
675 Southgate Office Plaza
5001 West 80th Street
Bloomington 55437
Tel (612) 835-6722
TWX 910-576-2867

**MISSOURI**
Technical Representatives, Inc.
Trade Center Bldg
300 Brookes Drive Suite 108
Hazelwood 63042
Tel (314) 731-5200
TWX 910-762-0618

**NEW JERSEY**
Intel Corp.
2 Kilmer Road
Edison 08817
Tel (201) 985-9100
TWX 710-480-6238

**NEW YORK**
Intel Corp.*
6901 Jericho Turnpike
Syosset 11791
Tel (516) 364-9860
TWX 510-221-2198

Intel Corp.
474 Thurston Road
Rochester, N.Y. 14619
Tel (716) 328-7340
TWX 510-253-3841

T-Squared
3522 James Street
Syracuse 13206
Tel (315) 483-8592
TWX 710-541-0554

**NEW YORK (cont.)**
T-Squared
640 Craig Road
P.O. Box W
Pittsford 14534
Tel (716) 381-2551
TELEX 97-8289

Intel Corp
55 Market Street
Poughkeepsie, New York 12601
Tel (914) 473-2303
TWX 510-248-0060

**NORTH CAROLINA**
Barnhill and Associates
113 Plateau Lane
Raleigh 27609
Tel (919) 876-5617

**OHIO**
Intel Corp.*
8312 North Main Street
Dayton 45415
Tel (513) 890-5350
TELEX 288-004

Intel Corp.*
27801 Euclid Ave.
Suite 450
Euclid 44132
Tel (216) 289-0101

**PENNSYLVANIA**
Vantage Sales Company
21 Bala Avenue
Bala Cynwyd 19004
Tel (215) 567-0999
TWX 510-662-5846

Intel Corp.*
1777 Walton Rd.
Suite 328A
Blue Bell 19422
Tel (215) 542-9444
TWX 510-661-0709

**TENNESSEE**
Barnhill and Associates
206 Chickasaw Drive
Johnson City 37601
Tel (615) 928-0184

**TEXAS**
Evans & McDowell Associates
13777 N. Central Expressway
Suite 405
Dallas 75231
Tel (214) 238-7157
TWX 910-867-4763

Evans & McDowell Associates
6610 Harwin Avenue, Suite 125
Houston 77036
Tel (713) 783-2900

Intel Corp.*
6350 L.B.J. Freeway
Suite 178
Dallas 75240
Tel (214) 661-8829
TWX 910-860-5487

**VIRGINIA**
Barnhill and Associates
P.O. Box 1104
Lynchburg 24505
Tel (804) 846-4624

**WASHINGTON**
E.S./Chase Co.
P.O. Box 80903
Seattle 98108
Tel (206) 762-4824
TWX 910-444-2298

**CANADA**
Multitek, Inc
4 Barren Street
Ottawa, Ontario K2J 1G2
Tel (613) 825-4695
TELEX 053-4585

## EUROPEAN MARKETING OFFICES

**BELGIUM**
Intel International*
Rue du Moulin a Papier
51-Boite 1
B-1160 Brussels
Tel (02) 660 30 10
TELEX 24814

**FRANCE**
Intel Corporation, S.A.R.L.*
74, Rue D'Arcueil
Silic 223
94528 Rungis Cedex
Tel (01) 687 22 21
TELEX 270475

**SCANDINAVIA**
Intel Scandinavia A/S*
Lyngbyvej 32 2nd Floor
DK-2100 Copenhagen East
Denmark
Tel (01) 18 20 00
TELEX 19567

Intel Sweden AB*
Box 86
S-16212 Vallingby 1
Sweden
Tel (08) 37 53 70
TELEX 13164 (ABCENT)

**ENGLAND**
Intel Corporation (U.K.) Ltd.*
Broadfield House
4 Between Towns Road
Cowley, Oxford OX4 3NB
Tel (0865) 77 14 31
TELEX 837203

Intel Corporation (U.K.) Ltd.
46-50 Beam Street
Nantwich, Cheshire CW5 5LJ
Tel (0270) 62 65 60
TELEX 36620

**GERMANY**
Intel Semiconductor GmbH*
Wolfratshauserstrasse 169
D8 Munich 71
Tel (089) 79 89 23
TELEX 5-212870

Intel Semiconductor GmbH
D-6272 Niedernhausen
Wiesenweg 26
Tel (06127) 2314
TELEX 04186183

Intel Semiconductor GmbH
D-7000 Stuttgart 80
Ernsthaldenstrasse 17
Tel (0711) 7351506
TELEX 7255346

## ORIENT MARKETING OFFICES

**JAPAN**
Intel Japan Corporation*
Flower Hill-Shinmachi East Bldg
1-23-9, Shinmachi, Setagaya-ku
Tokyo 154
Tel (03) 426-9261
TELEX 781-28426

**HONG KONG**
Q1 (Far East) Ltd
Tak Yan Commercial Bldg 6th floor
30-32 D'Aguilar Street, Central
Hong Kong
Tel 5-260311
TELEX 83138 JADE HX

**TAIWAN**
Taiwan Automation Co.*
8th Floor, 140 Section 1
Chung Hsiao E. Road
Taipei
Tel 393-1115
TELEX 11942 TA-AUTO

**TAIWAN (cont.)**
Asonics-Taiwan, Inc.
205 Pa-Teh Road, Section 4
Taipei
Tel 75 55 82
TELEX 22158 Asonics

## INTERNATIONAL DISTRIBUTORS

**AUSTRALIA**
A. J. Ferguson (Adelaide) PTY. Ltd
44 Prospect Rd
Prospect 5082
South Australia
Tel 269 1244
TELEX 82635

**AUSTRIA**
Bacher Elektronische Geräte GmbH
Meidlinger Hauptstrasse 78
A 1120 Vienna
Tel (0222) 83 63 96
TELEX (01) 1532

**BELGIUM**
Inelco Belgium S.A.
Avenue Val Duchesse, 3
B-1160 Brussels
Tel (02) 660 00 12
TELEX 25441

**DENMARK**
Scandinavian Semiconductor
Supply A/S
Nannasgade 18
DK-2200 Copenhagen N
Tel (01) 93 50 90
TELEX 19037

**FINLAND**
Oy Fintronic AB
Harjuankatu 2C
SF 00520
Helsinki 52
Tel (90) 664 451
TELEX 12426

**FRANCE**
Tekelec Airtronic
Cite des Bruyeres
Rue Carle Vernet
92310 Sevres
Tel (1) 027 75 35
TELEX 250997

**GERMANY**
Alfred Neye Enatechnik GmbH
Schillerstrasse 14
D-2085 Quickborn-Hamburg
Tel (04106) 6121
TELEX 02 13590

Electronic 2000 Vertriebs GmbH
Neumarkterstrasse 75
D-8000 Muenchen 80
Tel (089) 434061
TELEX 484426

Jermyn GmbH
Postfach 1146
D-6277 Kamberg
Tel (06434) 6005
TELEX 484426

**HONG KONG**
ASTEC International
Keystone House, 2nd Floor
Hankow Road, Kowloon
Tel 3-687760
TELEX 74899 ASCOM

**ISRAEL**
Telsys Ltd
54, Jabotinsky Road
IL-Ramat - Gan 52 464
Tel (3) 73 98 65
TELEX 32392

Eastronics Ltd
11 Rozanis Street
P.O. Box 39300
Tel-Aviv
Tel 475151
TELEX 33638

**ITALY**
Eledra 3S S.P.A.
Viale Elvezia, 18
20154 Milan
Tel (02) 3493041
TELEX 39332

Eledra 3S S.P.A.
Via Giuseppe Valmarana, 63
00139 Rome, Italy
Tel (06) 81 27 290 - 81 27 324

**JAPAN**
Pan Electron
No. 1 Higashikata-Machi
Midori-Ku, Yokohama 226
Tel (045) 471-8811
TELEX 781-4773

**NETHERLANDS**
Inelco Nederland
AFD Elektronic
Joan Muyskenweg 22
NL-1006 Amsterdam
Tel (020) 934824
TELEX 14622

**NORWAY**
Nordisk Elektronik (Norge) A/S
Mustads Vei 1
N-Oslo 2
Tel (02) 55 38 93
TELEX 16963

**SOUTH AFRICA**
Electronic Building Elements
P.O. Box 4609
Pretoria
Tel 78 92 21
TELEX 30181

**SPAIN**
Interface
Ronda General Mitre #7
E-Barcelona 17
Tel (93) 203-53-30
TELEX 52838

**SWEDEN**
Nordisk Electronik AB
Fack
S-10380 Stockholm 7
Tel (08) 248340
TELEX 10547

**SWITZERLAND**
Industrade AG
Gemsenstrasse 2
Postcheck 80 - 21190
CH-8021 Zurich
Tel (01) 60 22 30
TELEX 56788

**UNITED KINGDOM**
Rapid Recall Ltd.
11-15 Betterton Street
Drury Lane
London WC2H 9BS
Tel (01) 379-6741
TELEX 28752

G.E.C. Semiconductors Ltd
East Lane
Wembley HA9 7PP
Middlesex
Tel (01) 904-9303
TELEX 923429

Jermyn Industries
Vestry Sevenoaks Road
Sevenoaks Kent
Tel (0732) 51174
TELEX 95143

***Field Application Location**

intel

3065 Bowers Avenue
Santa Clara, California 95051
Tel: (408) 246-7501
TWX: 910-338-0026
TELEX: 34-6372

# U.S. AND CANADIAN DISTRIBUTORS

**ALABAMA**
†Hamilton/Avnet Electronics
805 Oser Drive NW
Huntsville 35805
Tel: (205) 533-1170

**ARIZONA**
Cramer/Arizona
2643 East University Drive
Phoenix 85034
Tel: (602) 263-1112

Hamilton/Avnet Electronics
2615 South 21st Street
Phoenix 85034
Tel: (602) 275-7851

Liberty/Arizona
3130 N. 27th Avenue
Phoenix 85107
Tel: (602) 257-1272
TELEX: 910-951-4282

**CALIFORNIA**
†Hamilton/Avnet Electronics
575 E. Middlefield Road
Mountain View 94040
Tel: (415) 961-7000

†Hamilton/Avnet Electronics
8917 Complex Drive
San Diego 92123
Tel: (714) 279-2421

†Hamilton Electro Sales
10912 W. Washington Boulevard
Culver City 90230
Tel: (213) 558-2121

†Cramer/San Francisco
720 Palomar Avenue
Sunnyvale 94086
Tel: (408) 739-3011

†Cramer/Los Angeles
1720 Daimler Street
Irvine 92705
Tel: (714) 979-3000

Cramer/San Diego
8975 Complex Drive
San Diego 92123
Tel: (714) 565-1881

†Liberty Electronics
124 Maryland Street
El Segundo 90245
Tel. (213) 322-8100
Tel. (714) 638-7601
TWX: 910-348-7140

Liberty/San Diego
8248 Mercury Court
San Diego 92111
Tel: (714) 565-9171
TELEX: 910-335-1590

Elmar Electronics
2288 Charleston Road
Mountain View 94040
Tel: (415) 961-3611
TELEX: 910-379-6437

**COLORADO**
Cramer/Denver
5465 E. Evans Pl. at Hudson
Denver 80222
Tel: (303) 758-2100

Elmar/Denver
6777 E. 50th Avenue
Commerce City 80022
Tel: (303) 287-9611
TWX: 910-936-0770

†Hamilton/Avnet Electronics
5921 No. Broadway
Denver 80216
Tel: (303) 534-1212

**CONNECTICUT**
Cramer/Connecticut
35 Dodge Avenue
North Haven 06473
Tel: (203) 239-5641

Components Plus
361 W. State
Westport 06880
Tel: (203) 226-4731

Hamilton/Avnet Electronics
643 Danbury Road
Georgetown 06829
Tel: (203) 762-0361

**FLORIDA**
Cramer/E.W. Hollywood
4035 No. 29th Avenue
Hollywood 33020
Tel: (305) 923-8181

†Hamilton/Avnet Electronics
4020 No. 29th Ave.
Hollywood 33021
Tel: (305) 925-5401

†Cramer/EW Orlando
345 No. Graham Ave.
Orlando 32814
Tel: (305) 894-1511

**GEORGIA**
Cramer/EW Atlanta
3923 Oakcliff Industrial Center
Atlanta 30340
Tel: (404) 448-9050

*Hamilton/Avnet Electronics*
6700 I 85. Access Road Suite 2B
Norcross 30071
Tel: (404) 448-0800

**ILLINOIS**
†Cramer/Chicago
1911 So. Busse Rd.
Mt. Prospect 60056
Tel. (312) 593-8230

†Hamilton/Avnet Electronics
3901 No. 25th Ave.
Schiller Park 60176
Tel. (312) 678-6310

**INDIANA**
Pioneer/Indiana
6408 Castleplace Drive
Indianapolis 46250
Tel: (317) 547-7777

Sheridan Sales Co.
5790 Purdue Road
Indianapolis 46268
Tel. (317) 297-3146

**KANSAS**
Hamilton/Avnet Electronics
37 Lenexa Industrial Center
9900 Pflumm Road
Lenexa 66215
Tel: (913) 888-8900

**MARYLAND**
Cramer/EW Baltimore
7235 Standard Drive
Hanover 21076
Tel: (301) 796-5790

†Cramer/EW Washington
16021 Industrial Drive
Gaithersburg 20760
Tel: (301) 948-0110

†Hamilton/Avnet Electronics
7235 Standard Drive
Hanover 21076
Tel: (301) 796-5000

**MASSACHUSETTS**
†Cramer Electronics Inc.
85 Wells Avenue
Newton 02159
Tel: (617) 969-7700

†Hamilton/Avnet Electronics
100 E. Commerce Way
Woburn 01801
Tel: (617) 273-2120

**MICHIGAN**
Sheridan Sales Co.
24543 Indoplex Drive
Farmington Hills 48024
Tel: (313) 477-3800

†Pioneer/Michigan
13485 Stamford
Livonia 48150
Tel. (313) 729-8500

†Hamilton/Avnet Electronics
12870 Farmington Road
Livonia 48150
Tel: (313) 522-4700
TWX: 810-242-8775

**MINNESOTA**
†Industrial Components
5280 West 74th Street
Minneapolis 55435
Tel: (612) 831-2666

Cramer/Bonn
7275 Bush Lake Road
Edina 55435
Tel: (612) 835-7811

†Hamilton/Avnet Electronics
7683 Washington Avenue So.
Edina 55435
Tel: (612) 941-3801

**MISSOURI**
†Hamilton/Avnet Electronics
364 Brookes Lane
Hazelwood 63042
Tel: (314) 731-1144

**NEW JERSEY**
Cramer/Pennsylvania, Inc.
12 Springdale Road
Cherry Hill Industrial Center
Cherry Hill 08003
Tel: (609) 424-5993
TWX: 710-896-0908

Components Plus
310 Railroad Avenue
Hackensack 07601
Tel: (201) 487-0565

**NEW JERSEY (cont.)**
†Hamilton/Avnet Electronics
218 Little Falls Road
Cedar Grove 07009
Tel: (201) 239-0800
*TWX: 710-994-5787*

Cramer/New Jersey
No. 1 Barrett Avenue
Moonachie 07074
Tel. (201) 935-5600

†Hamilton/Avnet Electronics
113 Gaither Drive
East Gate Industrial Park
Mt. Laurel 08057
Tel: (609) 234-2133
TWX: 710-897-1405

**NEW MEXICO**
Hamilton/Avnet Electronics
2450 Baylor Drive, S.E.
Albuquerque 87119
Tel: (505) 765-1500

Cramer/New Mexico
137 Vermont, N.E.
Albuquerque 87108
Tel: (505) 265-5767

**NEW YORK**
Cramer/Rochester
3000 Winton Road South
Rochester 14623
Tel: (716) 275-0300

Components Plus
40 Oser Avenue
Hauppauge 11787
Tel: (516) 231-9200

†Hamilton/Avnet Electronics
167 Clay Road
Rochester 14623
Tel: (716) 442-7820

†Cramer/Syracuse
6716 Joy Road
East Syracuse 13057
Tel. (315) 437-6671

†Hamilton/Avnet Electronics
6500 Joy Road
E. Syracuse 13057
Tel: (315) 437-2642

†Cramer/Long Island
29 Oser Avenue
Hauppauge, L.I. 11787
Tel: (516) 231-5600
TWX: 510-227-9863

†Hamilton/Avnet Electronics
70 State Street
Westbury, L.I. 11590
Tel: (516) 333-5800
TWX: 510-222-8237

**NORTH CAROLINA**
Cramer Electronics
938 Burke Street
Winston-Salem 27102
Tel: (919) 725-8711

**OHIO**
†Hamilton/Avnet Electronics
118 Westpark Road
Dayton 45459
Tel: (513) 433-0610
TWX: 810-450-2531

†Pioneer/Dayton
1900 Troy Street
Dayton 45404
Tel: (513) 236-9900

†Sheridan Sales Co.
10 Knollcrest Drive
Cincinnati 45222
Tel: (513) 761-5432
TWX: 810-461-2670

†Pioneer/Cleveland
4800 E. 131st Street
Cleveland 44105
Tel: (216) 587-3600

†Hamilton/Avnet Electronics
761 Beta Drive
Cleveland 44143
Tel: (216) 461-1400

Sheridan Sales Co.
23224 Commerce Park Road
Beachwood 44122
Tel: (216) 831-0130

Sheridan Sales Co.
Shiloh Building, Suite 250
5045 North Main Street
Dayton 45405
Tel: (513) 277-8911

**OKLAHOMA**
Components Specialties, Inc.
7920 E. 40th Street
Tulsa 74145
Tel: (918) 664-2820

**OREGON**
Almac/Stroum Electronics
4475 S.W. Scholls Ferry Rd.
Portland 97225
Tel. (503) 292-3534

**PENNSYLVANIA**
Sheridan Sales Co.
1717 Penn Avenue, Suite 5009
Pittsburgh 15221
Tel: (412) 244-1640

Pioneer Pittsburgh
560 Alpha Drive
Pittsburgh 15238
Tel: (412) 782-2300

**TEXAS**
Cramer Electronics
13740 Midway Road
Dallas 75240
Tel: (214) 661-9300

†Hamilton/Avnet Electronics
4445 Sigma Road
Dallas 75240
Tel. (214) 661-8661

†Hamilton/Avnet Electronics
1216 W. Clay
Houston 77019
Tel. (713) 526-4661

Component Specialties, Inc.
10907 Shady Trail, Suite 101
Dallas 75220
Tel: (214) 357-4576

†Component Specialties, Inc.
7313 Ashcroft Street
Houston 77036
Tel: (713) 771-7237

**UTAH**
Cramer/Utah
391 W. 2500 South
Salt Lake City 84115
Tel: (801) 487-4131

Hamilton/Avnet Electronics
647 W. Billinis Road
Salt Lake City 84119
Tel: (801) 262-8451

**WASHINGTON**
†Hamilton/Avnet Electronics
13407 Northrup Way
Bellevue 98005
Tel: (206) 746-8750

†Almac/Stroum Electronics
5811 Sixth Ave. South
Seattle 98108
Tel: (206) 763-2300

Cramer/Seattle
1059 Andover Park East
Tukwila 98188
Tel: (206) 575-0907

## CANADA

**ALBERTA**
L. A. Varah Ltd.
4742 14th Street N.E.
Calgary T2E 6LT
Tel: (403) 276-8818
Telex: 13 825 89 77

**BRITISH COLUMBIA**
†L.A. Varah Ltd.
2077 Alberta Street
Vancouver V5Y 1C4
Tel: (604) 873-3211
TWX: 610-929-1068
Telex: 04 53167

**ONTARIO**
Cramer/Canada
920 Alness Avenue, Unit No. 9
Downsview
Toronto 392 M3J 2H7
Tel: (416) 661-9222
TWX: 610-492-6210

Hamilton/Avnet Electronics
6291-16 Dorman Road
Mississauga L4V 1H2
Tel: (416) 677-7432
TWX: 610-492-8867

Hamilton/Avnet Electronics
1735 Courtwood Crest
Ottawa K2C 2B4
Tel: (613) 226-1700
TWX: 610 562-1906

**QUEBEC**
†Hamilton/Avnet Electronics
2670 Paulus
St. Laurent H4S 1G2
Tel: (514) 331-6443
TWX: 610-421-3731

**MANITOBA**
L.A. Varah Ltd.
153 Dorbell Drive
Winnipeg R2Y 1V4
Tel: (204) 989-9607

†MDS Centers

## INSTRUCTION SET

### Summary of Processor Instructions

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| MOV $r_1,r_2$ | Move register to register | 0 | 1 | D | D | D | S | S | S | 5 |
| MOV M, r | Move register to memory | 0 | 1 | 1 | 1 | 0 | S | S | S | 7 |
| MOV r, M | Move memory to register | 0 | 1 | D | D | D | 1 | 1 | 0 | 7 |
| HLT | Halt | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| MVI r | Move immediate register | 0 | 0 | D | D | D | 1 | 1 | 0 | 7 |
| MVI M | Move immediate memory | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 10 |
| INR r | Increment register | 0 | 0 | D | D | D | 1 | 0 | 0 | 5 |
| DCR r | Decrement register | 0 | 0 | D | D | D | 1 | 0 | 1 | 5 |
| INR M | Increment memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 10 |
| DCR M | Decrement memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 10 |
| ADD r | Add register to A | 1 | 0 | 0 | 0 | 0 | S | S | S | 4 |
| ADC r | Add register to A with carry | 1 | 0 | 0 | 0 | 1 | S | S | S | 4 |
| SUB r | Subtract register from A | 1 | 0 | 0 | 1 | 0 | S | S | S | 4 |
| SBB r | Subtract register from A with borrow | 1 | 0 | 0 | 1 | 1 | S | S | S | 4 |
| ANA r | And register with A | 1 | 0 | 1 | 0 | 0 | S | S | S | 4 |
| XRA r | Exclusive Or register with A | 1 | 0 | 1 | 0 | 1 | S | S | S | 4 |
| ORA r | Or register with A | 1 | 0 | 1 | 1 | 0 | S | S | S | 4 |
| CMP r | Compare register with A | 1 | 0 | 1 | 1 | 1 | S | S | S | 4 |
| ADD M | Add memory to A | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ADC M | Add memory to A with carry | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUB M | Subtract memory from A | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBB M | Subtract memory from A with borrow | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANA M | And memory with A | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRA M | Exclusive Or memory with A | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORA M | Or memory with A | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CMP M | Compare memory with A | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| ADI | Add immediate to A | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ACI | Add immediate to A with carry | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| SUI | Subtract immediate from A | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBI | Subtract immediate from A with borrow | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANI | And immediate with A | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRI | Exclusive Or immediate with A | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORI | Or immediate with A | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CPI | Compare immediate with A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| RLC | Rotate A left | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 |
| RRC | Rotate A right | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 4 |
| RAL | Rotate A left through carry | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| RAR | Rotate A right through carry | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 |
| JMP | Jump unconditional | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 10 |
| JC | Jump on carry | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 10 |
| JNC | Jump on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 10 |
| JZ | Jump on zero | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 10 |
| JNZ | Jump on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 10 |
| JP | Jump on positive | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 10 |
| JM | Jump on minus | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 10 |
| JPE | Jump on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 10 |
| JPO | Jump on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 10 |
| CALL | Call unconditional | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 17 |
| CC | Call on carry | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CNC | Call on no carry | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CZ | Call on zero | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CNZ | Call on no zero | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| CP | Call on positive | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CM | Call on minus | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CPE | Call on parity even | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CPO | Call on parity odd | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| RET | Return | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| RC | Return on carry | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RNC | Return on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 5/11 |

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code[1] | | | | | | | | |
| RZ | Return on zero | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RNZ | Return on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RP | Return on positive | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RM | Return on minus | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RPE | Return on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RPO | Return on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RST | Restart | 1 | 1 | A | A | A | 1 | 1 | 1 | 11 |
| IN | Input | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 10 |
| OUT | Output | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 10 |
| LXI B | Load immediate register Pair B & C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI D | Load immediate register Pair D & E | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| LXI H | Load immediate register Pair H & L | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI SP | Load immediate stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| PUSH B | Push register Pair B & C on stack | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH D | Push register Pair D & E on stack | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 11 |
| PUSH H | Push register Pair H & L on stack | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH PSW | Push A and Flags on stack | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 11 |
| POP B | Pop register pair B & C off stack | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP D | Pop register pair D & E off stack | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| POP H | Pop register pair H & L off stack | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP PSW | Pop A and Flags off stack | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| STA | Store A direct | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 13 |
| LDA | Load A direct | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 13 |
| XCHG | Exchange D & E, H & L Registers | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 4 |
| XTHL | Exchange top of stack, H & L | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 18 |
| SPHL | H & L to stack pointer | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 5 |
| PCHL | H & L to program counter | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 5 |
| DAD B | Add B & C to H & L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD D | Add D & E to H & L | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 10 |
| DAD H | Add H & L to H & L | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD SP | Add stack pointer to H & L | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 10 |
| STAX B | Store A indirect | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7 |
| STAX D | Store A indirect | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 7 |
| LDAX B | Load A indirect | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 |
| LDAX D | Load A indirect | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 7 |
| INX B | Increment B & C registers | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX D | Increment D & E registers | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 5 |
| INX H | Increment H & L registers | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX SP | Increment stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 5 |
| DCX B | Decrement B & C | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX D | Decrement D & E | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 5 |
| DCX H | Decrement H & L | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX SP | Decrement stack pointer | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 5 |
| CMA | Complement A | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 4 |
| STC | Set carry | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 4 |
| CMC | Complement carry | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| DAA | Decimal adjust A | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 4 |
| SHLD | Store H & L direct | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 16 |
| LHLD | Load H & L direct | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 16 |
| EI | Enable Interrupts | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 |
| DI | Disable interrupt | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| NOP | No-operation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |

NOTES: 1. DDD or SSS – 000 B – 001 C – 010 D – 011 E – 100 H – 101 L – 110 Memory – 111 A.

2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.

# INSTRUCTION SET

## Summary of Processor Instructions
### By Alphabetical Order

| Mnemonic | Description | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock [2] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Instruction Code [1] | | | | | | | | |
| ACI | Add immediate to A with carry | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| ADC M | Add memory to A with carry | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| ADC r | Add register to A with carry | 1 | 0 | 0 | 0 | 1 | S | S | S | 4 |
| ADD M | Add memory to A | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ADD r | Add register to A | 1 | 0 | 0 | 0 | 0 | S | S | S | 4 |
| ADI | Add immediate to A | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ANA M | And memory with A | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| ANA r | And register with A | 1 | 0 | 1 | 0 | 0 | S | S | S | 4 |
| ANI | And immediate with A | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| CALL | Call unconditional | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 17 |
| CC | Call on carry | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CM | Call on minus | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 11/17 |
| CMA | Complement A | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 4 |
| CMC | Complement carry | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| CMP M | Compare memory with A | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| CMP r | Compare register with A | 1 | 0 | 1 | 1 | 1 | S | S | S | 4 |
| CNC | Call on no carry | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CNZ | Call on no zero | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| CP | Call on positive | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 11/17 |
| CPE | Call on parity even | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| CPI | Compare immediate with A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| CPO | Call on parity odd | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 11/17 |
| CZ | Call on zero | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 11/17 |
| DAA | Decimal adjust A | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 4 |
| DAD B | Add B & C to H & L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD D | Add D & E to H & L | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 10 |
| DAD H | Add H & L to H & L | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD SP | Add stack pointer to H & L | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 10 |
| DCR M | Decrement memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 10 |
| DCR r | Decrement register | 0 | 0 | D | D | D | 1 | 0 | 1 | 5 |
| DCX B | Decrement B & C | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX D | Decrement D & E | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 5 |
| DCX H | Decrement H & L | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 5 |
| DCX SP | Decrement stack pointer | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 5 |
| DI | Disable Interrupt | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| EI | Enable Interrupts | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 |
| HLT | Halt | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| IN | Input | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 10 |
| INR M | Increment memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 10 |
| INR r | Increment register | 0 | 0 | D | D | D | 1 | 0 | 0 | 5 |
| INX B | Increment B & C registers | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX D | Increment D & E registers | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 5 |
| INX H | Increment H & L registers | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 |
| INX SP | Increment stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 5 |
| JC | Jump on carry | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 10 |
| JM | Jump on minus | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 10 |
| JMP | Jump unconditional | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 10 |
| JNC | Jump on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 10 |
| JNZ | Jump on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 10 |
| JP | Jump on positive | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 10 |
| JPE | Jump on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 10 |
| JPO | Jump on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 10 |
| JZ | Jump on zero | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 10 |
| LDA | Load A direct | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 13 |
| LDAX B | Load A indirect | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 |
| LDAX D | Load A indirect | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 7 |
| LHLD | Load H & L direct | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 16 |
| LXI B | Load immediate register Pair B & C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI D | Load immediate register Pair D & E | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| LXI H | Load immediate register Pair H & L | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI SP | Load immediate stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| MVI M | Move immediate memory | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 10 |
| MVI r | Move immediate register | 0 | 0 | D | D | D | 1 | 1 | 0 | 7 |
| MOV M, r | Move register to memory | 0 | 1 | 1 | 1 | 0 | S | S | S | 7 |
| MOV r, M | Move memory to register | 0 | 1 | D | D | D | 1 | 1 | 0 | 7 |
| MOV $r_1, r_2$ | Move register to register | 0 | 1 | D | D | D | S | S | S | 5 |
| NOP | No-operation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| ORA M | Or memory with A | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| ORA r | Or register with A | 1 | 0 | 1 | 1 | 0 | S | S | S | 4 |
| ORI | Or immediate with A | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| OUT | Output | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 10 |
| PCHL | H & L to program counter | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 5 |
| POP B | Pop register pair B & C off stack | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP D | Pop register pair D & E off stack | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| POP H | Pop register pair H & L off stack | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP PSW | Pop A and Flags off stack | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| PUSH B | Push register Pair B & C on stack | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH D | Push register Pair D & E on stack | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 11 |
| PUSH H | Push register Pair H & L on stack | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 11 |
| PUSH PSW | Push A and Flags on stack | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 11 |
| RAL | Rotate A left through carry | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| RAR | Rotate A right through carry | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 |
| RC | Return on carry | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RET | Return | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| RLC | Rotate A left | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 |
| RM | Return on minus | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5/11 |
| RNC | Return on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RNZ | Return on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RP | Return on positive | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 5/11 |
| RPE | Return on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| RPO | Return on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5/11 |
| RRC | Rotate A right | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 4 |
| RST | Restart | 1 | 1 | A | A | A | 1 | 1 | 1 | 11 |
| RZ | Return on zero | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 5/11 |
| SBB M | Subtract memory from A with borrow | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| SBB r | Subtract register from A with borrow | 1 | 0 | 0 | 1 | 1 | S | S | S | 4 |
| SBI | Subtract immediate from A with borrow | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| SHLD | Store H & L direct | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 16 |
| SPHL | H & L to stack pointer | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 5 |
| STA | Store A direct | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 13 |
| STAX B | Store A indirect | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7 |
| STAX D | Store A indirect | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 7 |
| STC | Set carry | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 4 |
| SUB M | Subtract memory from A | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SUB r | Subtract register from A | 1 | 0 | 0 | 1 | 0 | S | S | S | 4 |
| SUI | Subtract immediate from A | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| XCHG | Exchange D & E, H & L Registers | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 4 |
| XRA M | Exclusive Or memory with A | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| XRA r | Exclusive Or register with A | 1 | 0 | 1 | 0 | 1 | S | S | S | 4 |
| XRI | Exclusive Or immediate with A | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| XTHL | Exchange top of stack, H & L | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 18 |

NOTES: 1. DDD or SSS – 000 B – 001 C – 010 D – 011 E – 100 H – 101 L – 110 Memory – 111 A.
2. Two possible cycle times, (5/11) indicate instruction cycles dependent on condition flags.