Middle Technical University
Electrical Engineering Technical College
Control and Automation Engineering Tech. Dept.

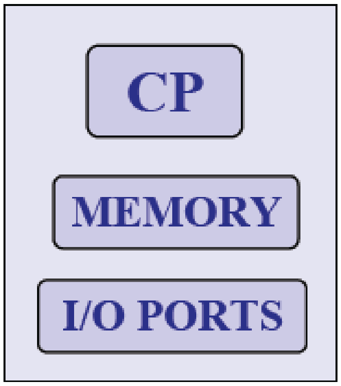
8085 Microprocessor

➢ Microprocessor:

- It is a silicon chip that contains a CPU. In the world of personal computers, the terms *microprocessor* and CPU are used interchangeably.
- A **microprocessor** (sometimes abbreviated **µP**) is a digital electronic component with miniaturized transistors on a single semiconductor integrated circuit (IC).
- One or more microprocessors typically serve as a central processing unit (CPU) in a computer system or handheld device.
- Microprocessors made possible the advent of the microcomputer.
- At the heart of all personal computers and most working stations sits a microprocessor.
- Microprocessors also control the logic of almost all digital devices, from clock radios to fuel-injection systems for automobiles.
- Three basic characteristics differentiate microprocessors:
- **Instruction set**: The set of instructions that the microprocessor can execute.
- **Bandwidth**: The number of bits processed in a single instruction.
- **Clock speed**: Given in megahertz (MHz), the clock speed determines how many instructions per second the processor can execute.
- In both cases, the higher the value, the more powerful the CPU. For example, a 32 bit microprocessor that runs at 50MHz is more powerful than a 16-bit microprocessor that runs at 25MHz.
- In addition to bandwidth and clock speed, microprocessors are classified as being either RISC (reduced instruction set computer) or CISC (complex instruction set computer).
- **Reduced Set Instruction Set Architecture (RISC)** The main idea behind is to make hardware simpler by using an instruction set composed of a few basic steps for loading, evaluating and storing operations just like an addition command will be composed of loading data, evaluating and storing.
- **Complex Instruction Set Architecture (CISC)** The main idea is to make hardware complex as a single instruction will do all loading, evaluating and storing operations just like a multiplication command will do stuff like loading data, evaluating and storing it.

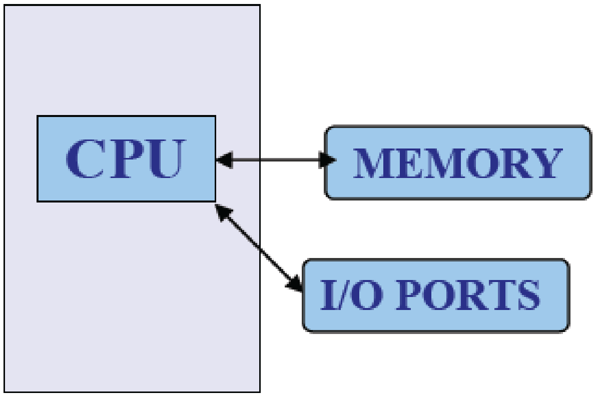
MICRO CONTROLLER

- It is a single chip
- Consists Memory,
  I/o ports



MICRO PROCESSER

- It is a CPU
- Memory, I/O Ports to be
  connected externally



Intel 8085 8-bit Microprocessor

- It is 8 bit microprocessor.
- It is manufactured with N-MOS technology.
- It has 16-bit address bus and hence can address up to 216 = 65536 bytes (64KB) memory locations through A_0-A_{15}.
- The first 8 lines of address bus and 8 lines of data bus are multiplexed AD_0 – AD_7.
- Data bus is a group of 8 lines D_0 – D_7.
- It supports external interrupt request.
- A 16 bit program counter (PC)
- A 16 bit stack pointer (SP)
- Six 8-bit general purpose register arranged in pairs: BC, DE, HL.
- It requires a signal +5V power supply and operates at 3.2 MHZ single phase clock.
- It is enclosed with 40 pins DIP (Dual in line package).

Figure (2.1) shows the block diagram of Intel 8085A. It consists of three main sections, an arithmetic and logic unit (ALU), a timing and control unit and several registers. These important sections are described as under.

ALU

The arithmetic and logic unit, ALU performs the following arithmetic and logic operations.

1. Addition
2. Subtraction
3. Logical AND
4. Logical OR
5. Logical EXCLUSIVE OR
6. Complement (logical NOT)
7. Increment (add 1)
8. Decrement (subtract 1)
9. Left shift (add input to itself)
10. Clear (result is zero)

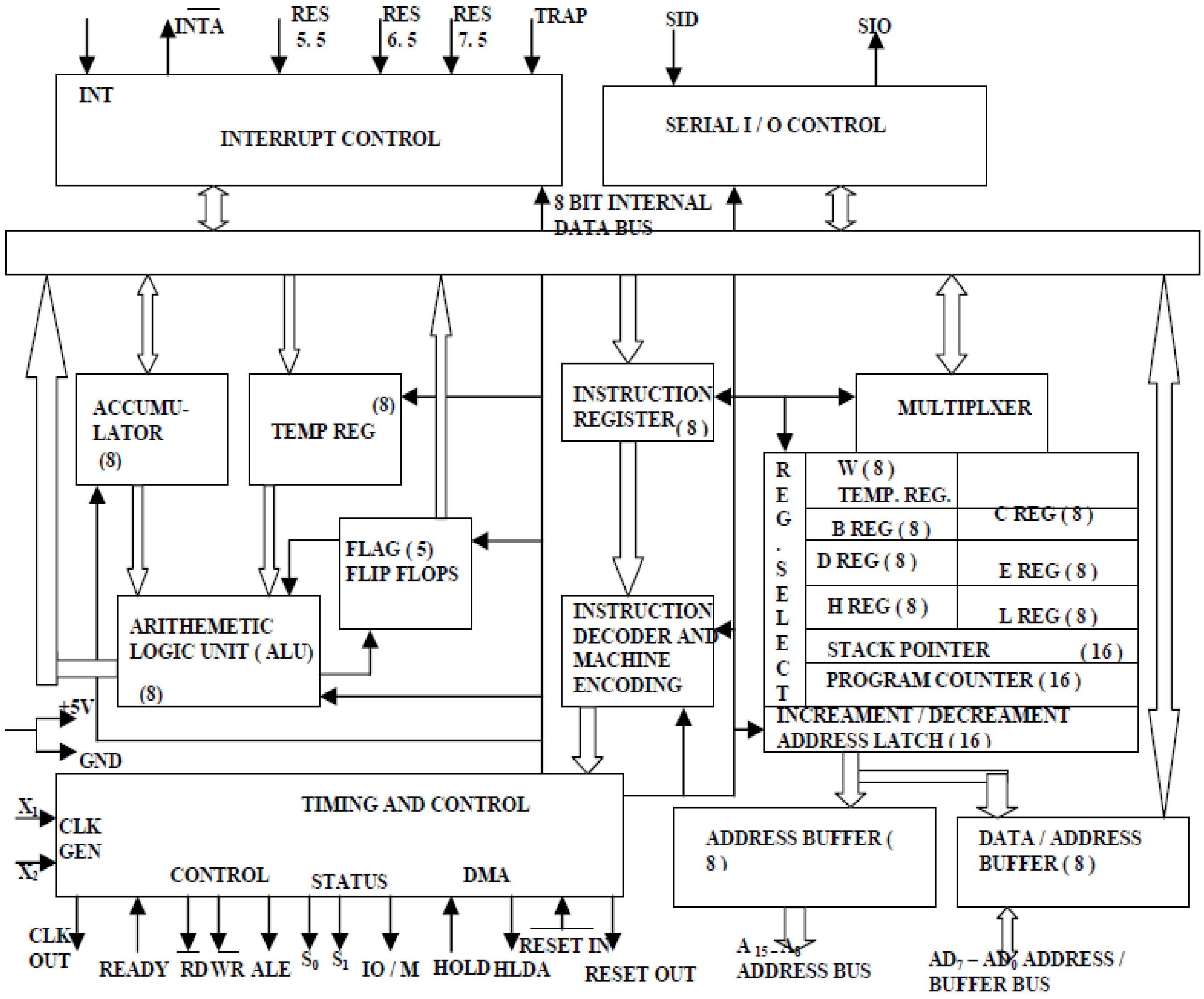


Fig (2.1): Block diagram of 8085 MP

Figure (2.2) shows the pin diagram of 8085 MP.

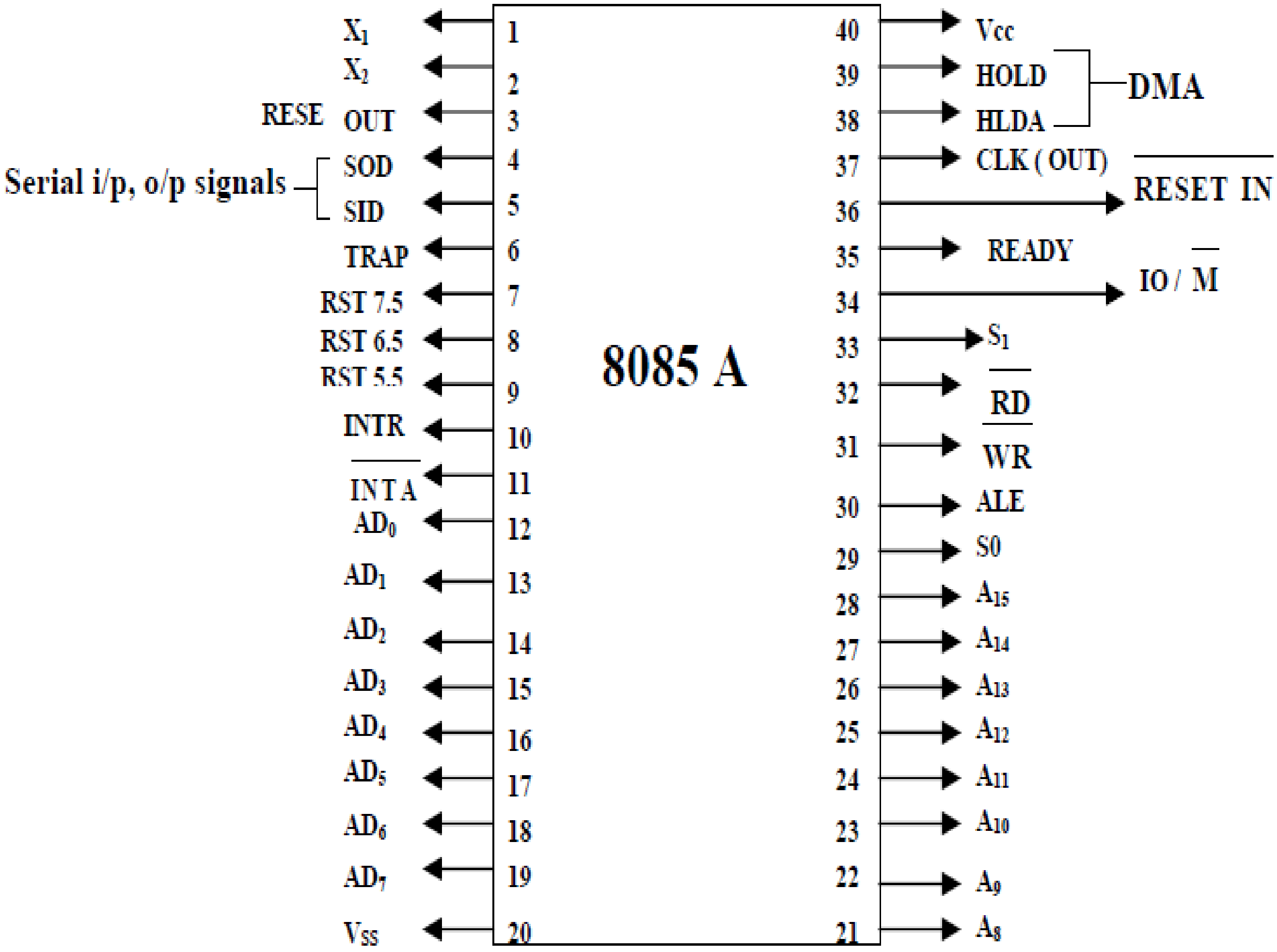


Fig (2.2): Pin diagram of 8085 MP

Timing and Control Unit

The timing and control unit generates timing signals for the execution of instruction and control of peripheral devices. The organization of a microprocessor and types of registers differ from processor to processor. The timing used for the execution of instructions and control of peripherals are different for different microprocessors. The selection of a suitable microprocessor for a particular application is a tough task for an engineer. The knowledge of the organization and timing and control system helps an engineer in the selection of a microprocessor. The design and cost of a processor also depends on the timing structure and register organization. For the execution of an instruction a microprocessor fetches the instruction from the memory and executes it. The time taken for the execution of an instruction is called instruction cycle (IC). An instruction cycle (IC). An instruction cycle consists of a fetch cycle (FC) and an execute cycle (EC). A fetch cycle is the time required for the fetch operation in which the machine code of the instruction (opcode) is fetched from the memory. This time is a fixed slot of time. An execute cycle is of variable width which depends on the instruction to be executed. The total time for the execution is given by IC = FC + EC.

Fetch Operation

In fetch operation the microprocessor gets the 1st byte of the instruction, which is operation code (opcode), from the memory. The program counter keeps the track of address of the next instruction to be executed. In the beginning of the fetch cycle the content of the program counter is sent to the memory. This takes one clock cycle. The memory first reads the opcode. This operation also takes one clock cycle. Then the memory sends the opcode to the microprocessor, which takes one clock period. The total time for fetch operation is the time required for fetching an opcode from the memory. This time is called fetch cycle.

Execute Operation

The opcode fetched from the memory goes to the data register, DR (data/address buffer in Intel 8085) and then to instruction register, IR. From the instruction register it goes to the decoder circuitry is within the microprocessor. After the instruction is decoded, execution begins. If the operand is in the general purpose registers, execution is immediately performed. The time taken in decoding and the address of the data, some read cycles are also necessary to receive the data from the memory. These read cycle are similar to opcode fetch cycle. The fetch quantities in these cycles are address or data. Figure (2.3) (a) and Figure (2.3) (b) shows an instruction and fetch cycle respectively.

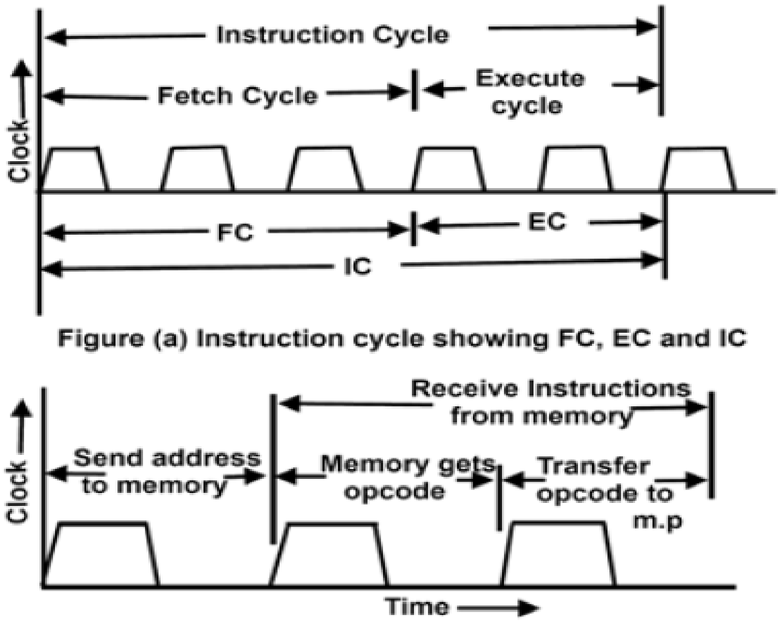


Figure (a) Instruction cycle showing FC, EC and IC

Figure (b) A Typical Fetch Cycle

Figure (2.3): Instruction cycle

There are many signals in this unit:

1. ALE: stands for Address Latch Enable signal. It is the 30th pin of 8085 which is used to enable or disable the address bus. The address bus will be enabled during the first clock cycle as the ALE pin goes high (logic 1). During the 2nd and 3rd clock cycles it goes low (logic 0) indicating that the data bus is used as shown in figure (2.4). The high order bits of the address remain on the bus for three clock periods. However, the low order bits remain for only one clock period and they would be lost if they are not saved externally. To make sure we have the entire address for the full three clock cycles, we will use an external latch to save the value of the AD0-AD7 when it is carrying the address bits. ALE operates as a pulse during T1 to the latch this will be able to latch the address in the latch. Then the ALE signal goes low, now the AD0-AD7 can be used as a data bus.

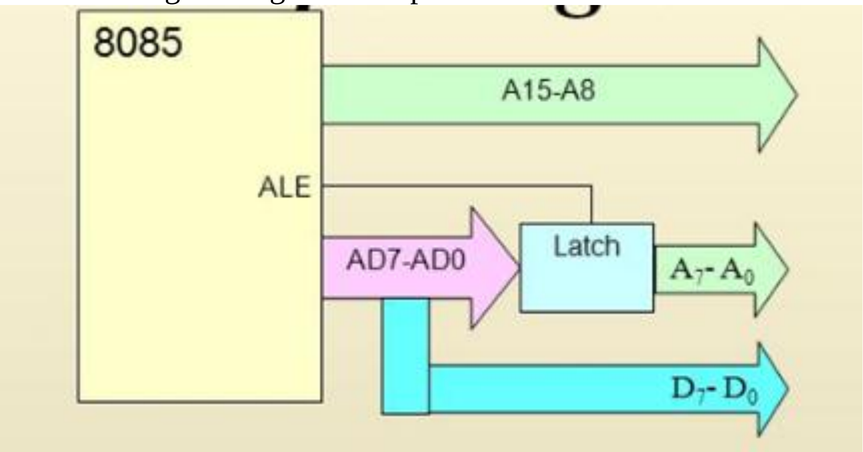


Figure (2.4): ALE signal in 8085 MP

2. IO/M’: it is a status signal which determine whether the address is for input-output or memory. When it is high (1) the address on the address bus is for input-output devices. When it is low (0) the address on the address bus is for the memory.
3. S_0, S_1: these are status signals. They distinguish the various types of operations as shown in table 1 below.

Table 1: Operations in 8085 MP

IO/M’	S_1	S_0	Data bus status
0	1	1	0pcode fetch
0	1	0	Memory read
0	0	1	Memory write
1	1	0	I/O read
1	0	1	I/O write
1	1	1	Interrupt acknowledge
0	0	0	Halt

4. RD’: it is a signal to control the read operation. It is low active signal. When it is low the selected memory or I/O device is read.
5. WR’: it is a signal to control the write operation. It is low active signal. When it goes low the data on the data bus is written to the selected memory or I/O location.
6. READY: it senses whether a peripheral is ready to transfer data or not. If the READY signal is 1 that mean the peripheral is ready. If it is 0 the MP waits till it goes 1. It is useful for interfacing low speed devices.

Registers

- **Accumulator** or A register is an 8-bit register used for arithmetic, logic, I/O and load/store operations.
- **Flag Register:** its 8 bit register, five bits used only as shown in figure (2.5).
- **Sign (S)** - set if the most significant bit of the result is set (one).
- **Zero (Z)** - The zero status flag Z is set to 1 if the result of an arithmetic or logical operation is zero. For non-zero result it is set to 0.
- **Auxiliary carry (AC)** – set to 1 if there was a carry out from bit D3 to bit D4 of the result.
- **Parity (P)** - The parity status flag is set to 1 when result of the operation contains even number of 1's. It is set to zero when there is odd number of 1's.
- **Carry (CY)** - The carry status flag holds carry out of the most significant bit resulting from the execution of an arithmetic operation. If there is a carry from addition or a borrow from subtraction or comparison, the carry flag CS is set to 1, otherwise 0.

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

Fig (2.5): Flag register

General Registers

- 8-bit B and 8-bit C registers can be used as one 16-bit BC register pair. When used as a pair the C register contains low-order byte. Some instructions may use BC register as a data pointer.
- 8-bit D and 8-bit E registers can be used as one 16-bit DE register pair. When used as a pair the E register contains low-order byte. Some instructions may use DE register as a data pointer.
- 8-bit H and 8-bit L registers can be used as one 16-bit HL register pair. When used as a pair the L register contains low-order byte. HL register usually contains a data pointer used to reference memory addresses.
- **Stack pointer** is a 16 bit register. It holds the address of the stack top. The stack is a sequence of memory locations defined by the programmer. The stack is used to save the content of a register during the execution of a program. The last memory location of the occupied 4 portion of the stack is called stack top. For example, suppose that the stack location 2000 is the stack top which is contained by the stack pointer. Now the contents of B-C pair so to be saved. This will be stored in the stack locations 1999 and 1998. The new stack top will be stored in

the stack pointer. The new stack top is the location 1998. If more data come they will be stored in the stack location 1997 onwards. Suppose the contents of H-L pair are to be pushed. They will go in 1997 and 1996. The new stack top will be the stack location 1996 and vacant locations are 1995 onward. This register is always decremented/incremented by 2 during push and pop.

- **Program counter** is a 16-bit register. It contains the address of the next instruction. The CPU fetches an instruction from the memory executes it and increments the content of the program counter. Thus in the next instruction cycle it will fetch next instruction. Instructions are executed sequentially unless an instruction changes the content of the program counter.

INSTRUCTION SET OF INTEL 8085

An Instruction is a command given to the computer to perform a specified operation on given data. The instruction set of a microprocessor is the collection of the instructions that the microprocessor is designed to execute. The instructions described here are of Intel 8085. These instructions are of Intel Corporation. They cannot be used by other microprocessor manufactures. The programmer can write a program in assembly language using these instructions. These instructions have been classified into the following groups:

1. Data Transfer Group

2. Arithmetic Group

3. Logical Group

4. Branch Control Group

5. I/O and Machine Control Group

1. Data Transfer Group

Instructions, which are used to transfer data from one register to another register, from memory to register or register to memory, come under this group. Examples are: MOV, MVI, LXI, LDA, STA etc. When an instruction of data transfer group is executed, data is transferred from the source to the destination without altering the contents of the source. For example, when MOV A, B is executed the content of the register B is copied into the register A, and the content of register B remains unaltered. Similarly, when LDA 2500 is executed the content of the memory location 2500 is loaded into the accumulator. But the content of the memory location 2500 remains unaltered.

2. Arithmetic Group

The instructions of this group perform arithmetic operations such as addition, subtraction; increment or decrement of the content of a register or memory. Examples are: ADD, SUB, INR, DAD etc.

3. Logical Group

The Instructions under this group perform logical operation such as AND, OR, compare, rotate etc. Examples are: ANA, XRA, ORA, CMP, and RAL etc.

4. Branch Control Group

This group includes the instructions for conditional and unconditional jump, subroutine call and return, and restart. Examples are: JMP, JC, JZ, CALL, CZ, RST etc.

5. I/O and Machine Control Group

This group includes the instructions for input/output ports, stack and machine control. Examples are: IN, OUT, PUSH, POP, and HLT etc.

Addressing modes of 8085 MP

- To perform any operation, we have to give the corresponding instructions to the MP. In each instruction, programmer has to specify 3 things:

1. Operation to be performed
2. source Address of data
3. destination Address of result

- The method by which the address of source of data or address of destination of the result is given in the instruction is called addressing modes. The term addressing mode refers to the way in which the operand of the instruction is specified.
- Intel 8085 uses the following addressing modes:
    1. Direct addressing mode
    2. Register addressing mode
    3. Register indirect addressing mode
    4. Immediate addressing mode
    5. Implicit addressing mode

1. Direct addressing mode

In this mode, the address of the operand is given in the instruction itself.

Ex.:

LDA 2500 H	Load the contents of memory location 2500H in accumulator.

- LDA is the operation.
- 2500 H is the source address.
- Accumulator is the destination.

2. Register addressing mode

In this mode, the operand is in general purpose register.

Ex.:

MOV A,B	Move the contents of register B to register A.

- MOV is the operation.
- B is the source of data.
- A is the destination.

3. Register indirect addressing mode

In this mode, the address of operand is specified by a register pair.

Ex.:

MOV A,M	Move the contents of memory location specified by H-L pair to accumulator.

- MOV is the operation.
- M is the memory location specified by H-L register pair.
- A is the destination.

4. Immediate addressing mode

In this mode, the operand is specified within the instruction itself.

Ex.:

MVI A, 05 H	Move 05 H to register A (accumulator).

- MVI is the operation.
- 05 H is the immediate data (source).
- A is the destination.

5. Implicit addressing mode

If the address of source of data as well as address of destination of result is fixed, then there is no need to give any operand along with the instruction.

Ex.:

CMA	Complement accumulator.

- CMA is the operation.
- A is the source.
- A is the destination.