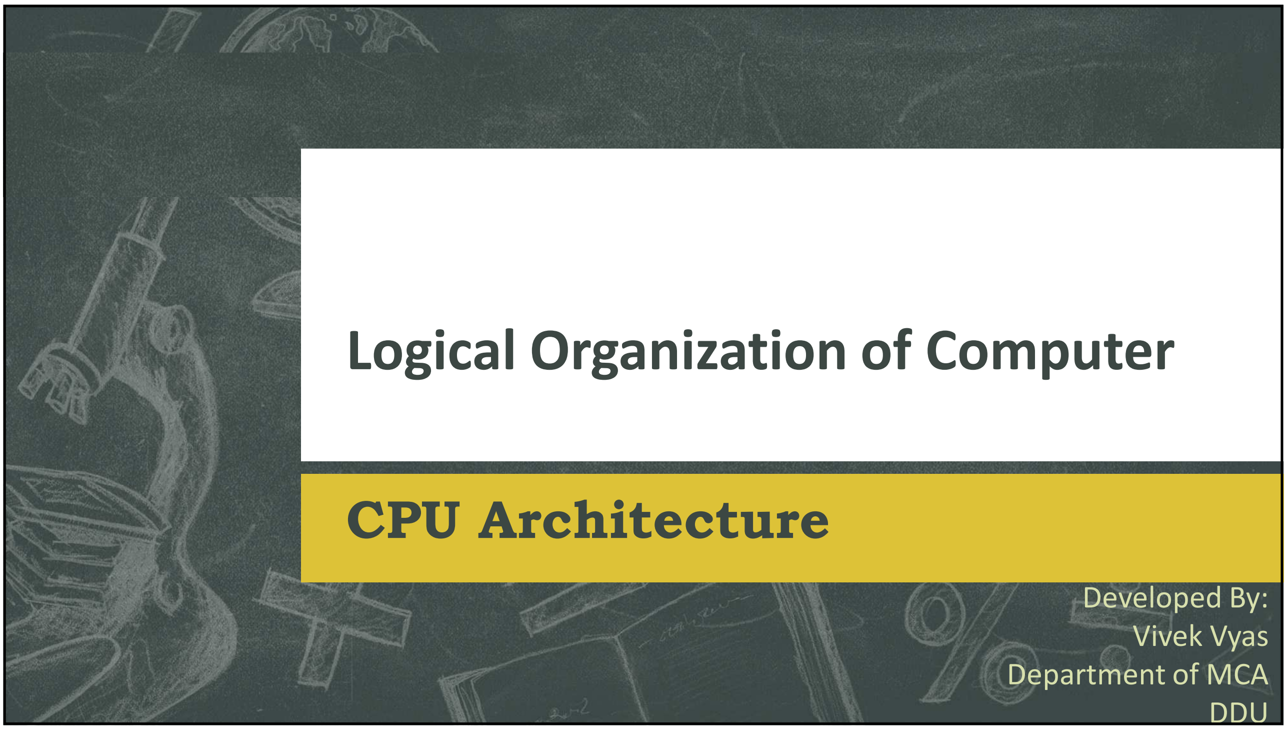
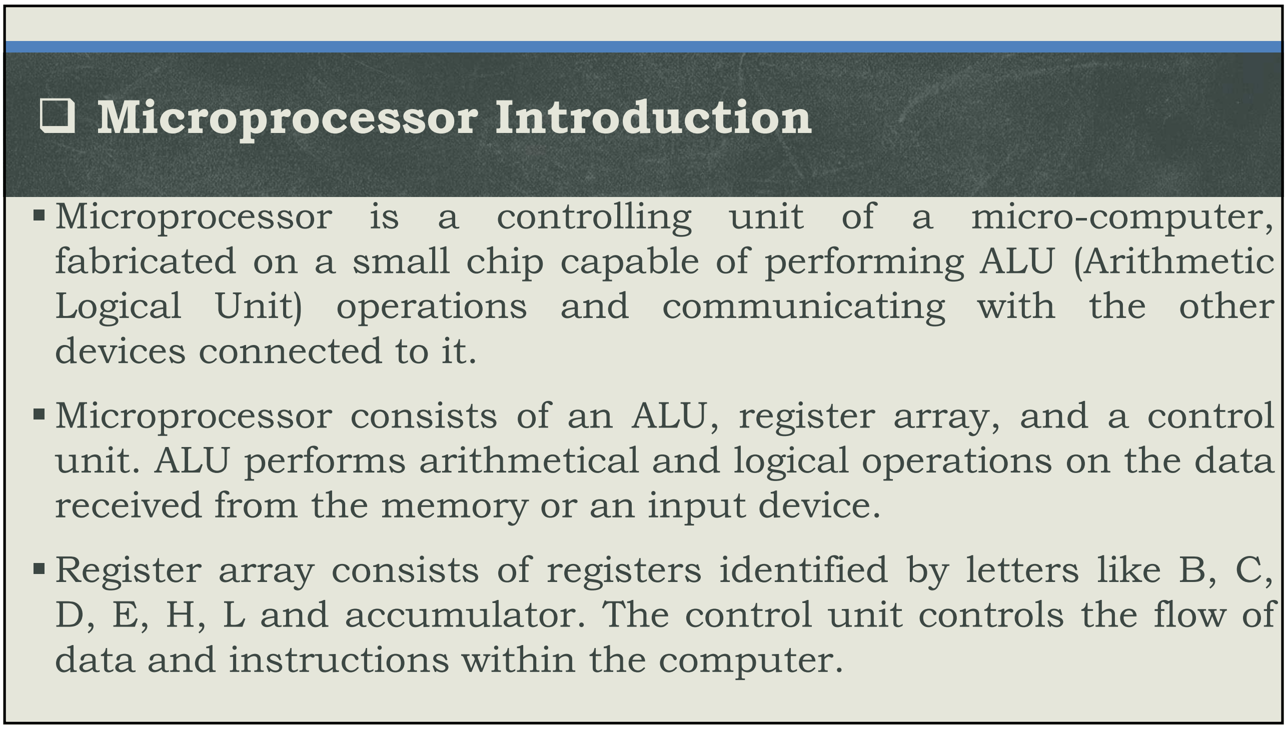


Logical Organization of Computer

CPU Architecture

Developed By:
Vivek Vyas
Department of MCA
DDU

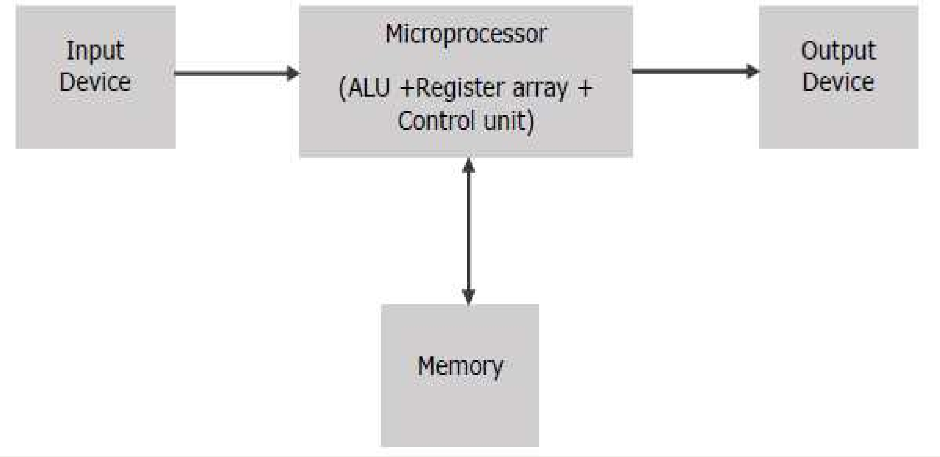


❑ Microprocessor Introduction

- Microprocessor is a controlling unit of a micro-computer, fabricated on a small chip capable of performing ALU (Arithmetic Logical Unit) operations and communicating with the other devices connected to it.
- Microprocessor consists of an ALU, register array, and a control unit. ALU performs arithmetical and logical operations on the data received from the memory or an input device.
- Register array consists of registers identified by letters like B, C, D, E, H, L and accumulator. The control unit controls the flow of data and instructions within the computer.

❑ Microprocessor Introduction

- **Block Diagram of a Basic Microcomputer**



❑ Microprocessor Introduction

- **How does a Microprocessor Work?**
- The microprocessor follows a sequence: Fetch, Decode, and then Execute.
- Initially, the instructions are stored in the memory in a sequential order. The microprocessor fetches those instructions from the memory, then decodes it and executes those instructions till STOP instruction is reached.
- Later, it sends the result in binary to the output port. Between these processes, the register stores the temporarily data and ALU performs the computing functions.

❑ Microprocessor Introduction

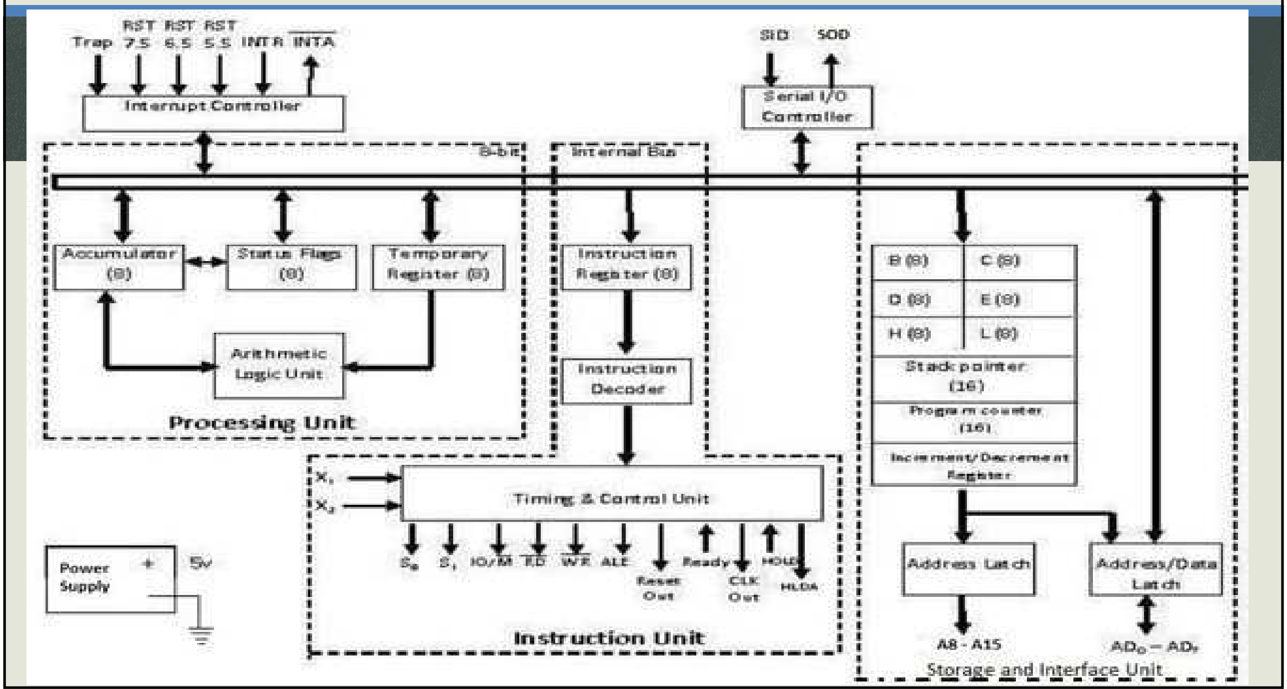
- **List of Terms Used in a Microprocessor**
- Here is a list of some of the frequently used terms in a microprocessor −
- **Instruction Set** − It is the set of instructions that the microprocessor can understand.
- **Bandwidth** − It is the number of bits processed in a single instruction.
- **Clock Speed** − It determines the number of operations per second the processor can perform. It is expressed in megahertz (MHz) or gigahertz (GHz).It is also known as Clock Rate.

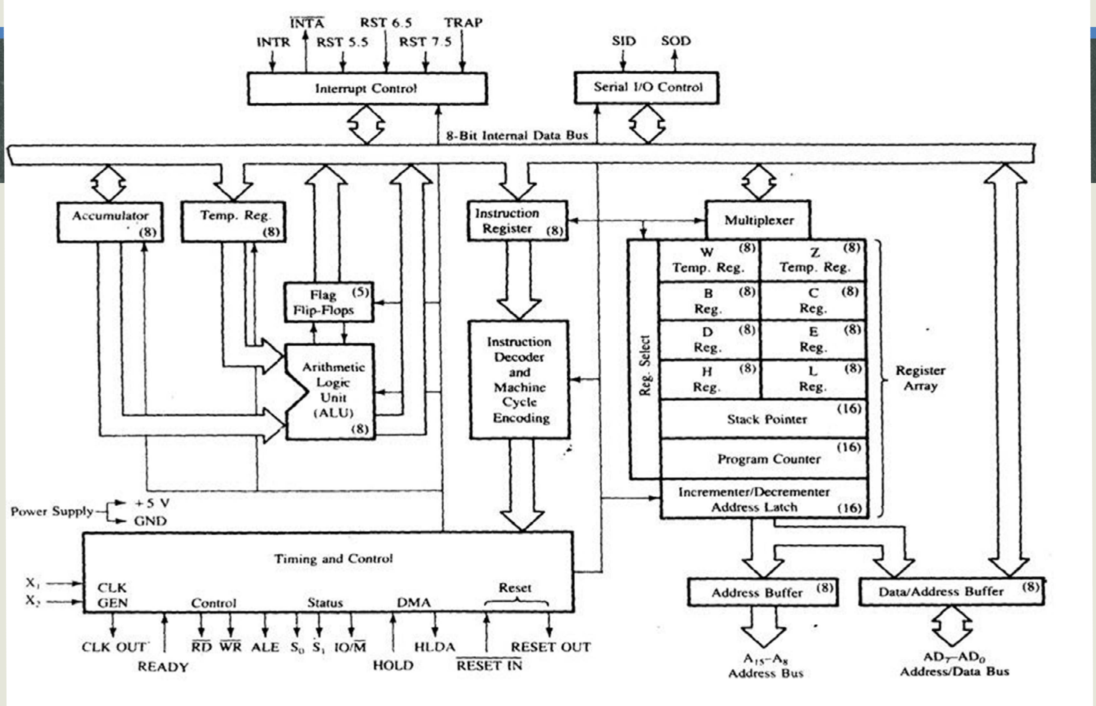
❑ Microprocessor Introduction

- **Word Length** − It depends upon the width of internal data bus, registers, ALU, etc. An 8-bit microprocessor can process 8-bit data at a time. The word length ranges from 4 bits to 64 bits depending upon the type of the microcomputer.
- **Data Types** − The microprocessor has multiple data type formats like binary, BCD, ASCII, signed and unsigned numbers.

❑ Microprocessor - 8085 Architecture

- 8085 is pronounced as "eighty-eighty-five" microprocessor. It is an 8-bit microprocessor designed by Intel in 1977 using NMOS technology.
- It has the following configuration −
  - 8-bit data bus
  - 16-bit address bus, which can address upto 64KB
  - A 16-bit program counter
  - A 16-bit stack pointer
  - Six 8-bit registers arranged in pairs: BC, DE, HL
  - Requires +5V supply to operate at 3.2 MHZ single phase clock
  - It is used in washing machines, microwave ovens, mobile phones, etc.





❑ 8085 Microprocessor – Functional Units

- **Accumulator**
- It is an 8-bit register used to perform arithmetic, logical, I/O & LOAD/STORE operations. It is connected to internal data bus & ALU.
- **Arithmetic and logic unit**
- As the name suggests, it performs arithmetic and logical operations like Addition, Subtraction, AND, OR, etc. on 8-bit data.

❑ 8085 Microprocessor – Functional Units

- **General purpose register**
- There are 6 general purpose registers in 8085 processor, i.e. B, C, D, E, H & L. Each register can hold 8-bit data.
- These registers can work in pair to hold 16-bit data and their pairing combination is like B-C, D-E & H-L.
- **Program counter**
- It is a 16-bit register used to store the memory address location of the next instruction to be executed. Microprocessor increments the program whenever an instruction is being executed, so that the program counter points to the memory address of the next instruction that is going to be executed.

❑ 8085 Microprocessor – Functional Units

- **Stack pointer**
- It is also a 16-bit register works like stack, which is always incremented/decremented by 2 during push & pop operations.
- **Temporary register**
- It is an 8-bit register, which holds the temporary data of arithmetic and logical operations.

❑ 8085 Microprocessor – Functional Units

- **Flag register**
- It is an 8-bit register having five 1-bit flip-flops, which holds either 0 or 1 depending upon the result stored in the accumulator.
- These are the set of 5 flip-flops –
  - Sign (S)
  - Zero (Z)
  - Auxiliary Carry (AC)
  - Parity (P)
  - Carry (C)

❑ 8085 Microprocessor – Functional Units

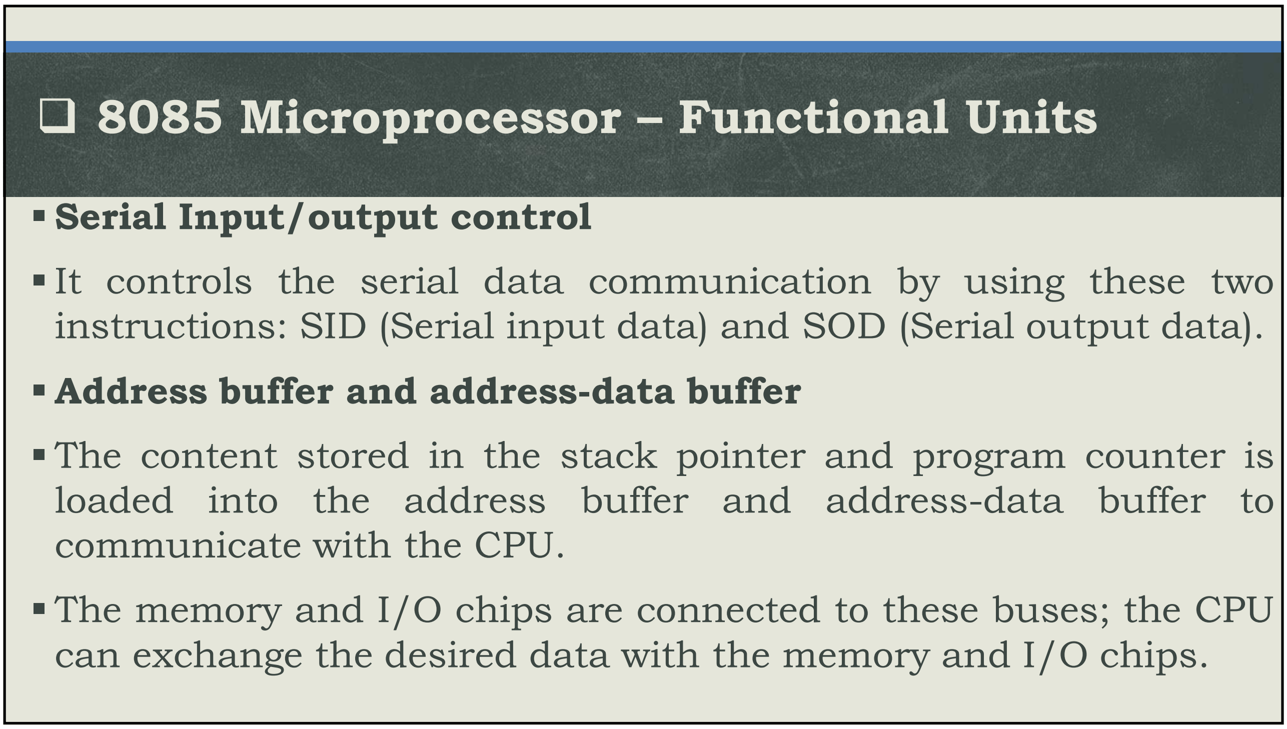
- **Instruction register and decoder**
- It is an 8-bit register. When an instruction is fetched from memory then it is stored in the Instruction register. Instruction decoder decodes the information present in the Instruction register.
- **Timing and control unit**
- It provides timing and control signal to the microprocessor to perform operations. Following are the timing and control signals, which control external and internal circuits –

❑ 8085 Microprocessor – Functional Units

- Control Signals: READY, RD’, WR’, ALE
- Status Signals: S0, S1, IO/M’
- DMA Signals: HOLD, HLDA
- RESET Signals: RESET IN, RESET OUT

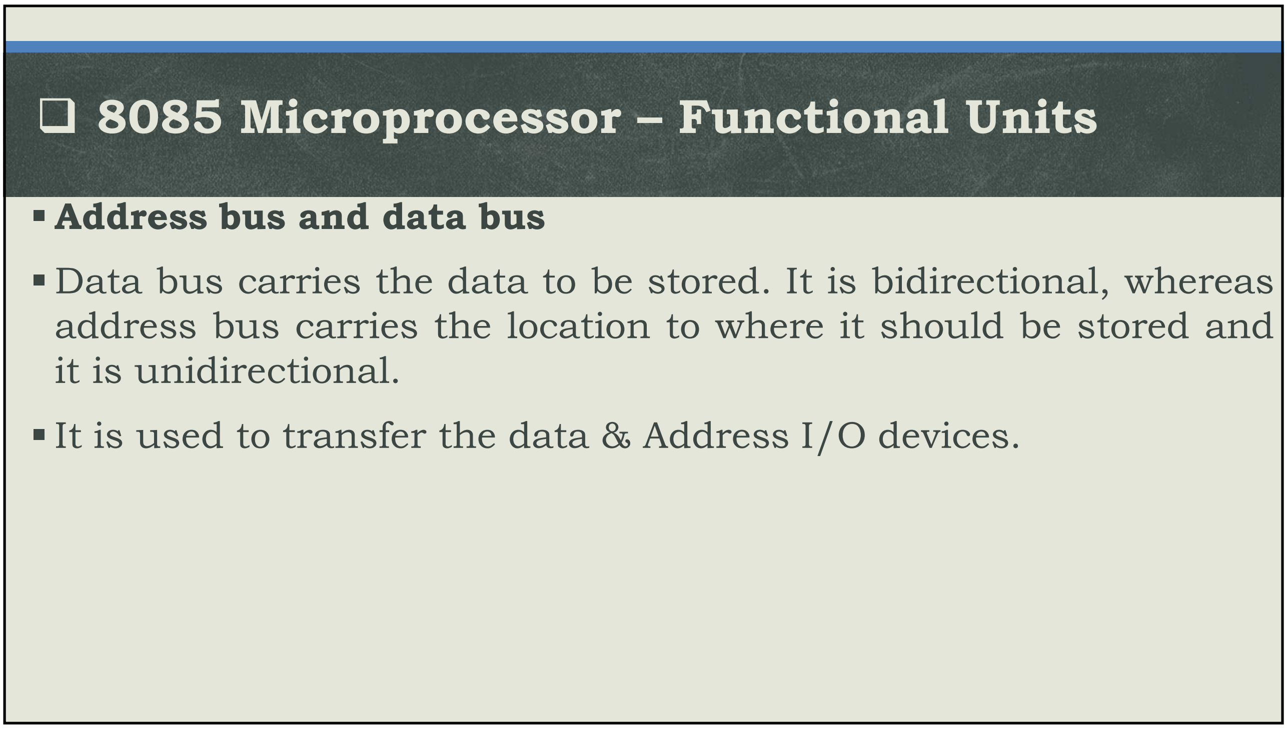
❑ 8085 Microprocessor – Functional Units

- **Interrupt control**
- As the name suggests it controls the interrupts during a process.
- When a microprocessor is executing a main program and whenever an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming request.
- After the request is completed, the control goes back to the main program.
- There are 5 interrupt signals in 8085 microprocessor: INTR, RST 7.5, RST 6.5, RST 5.5, TRAP.



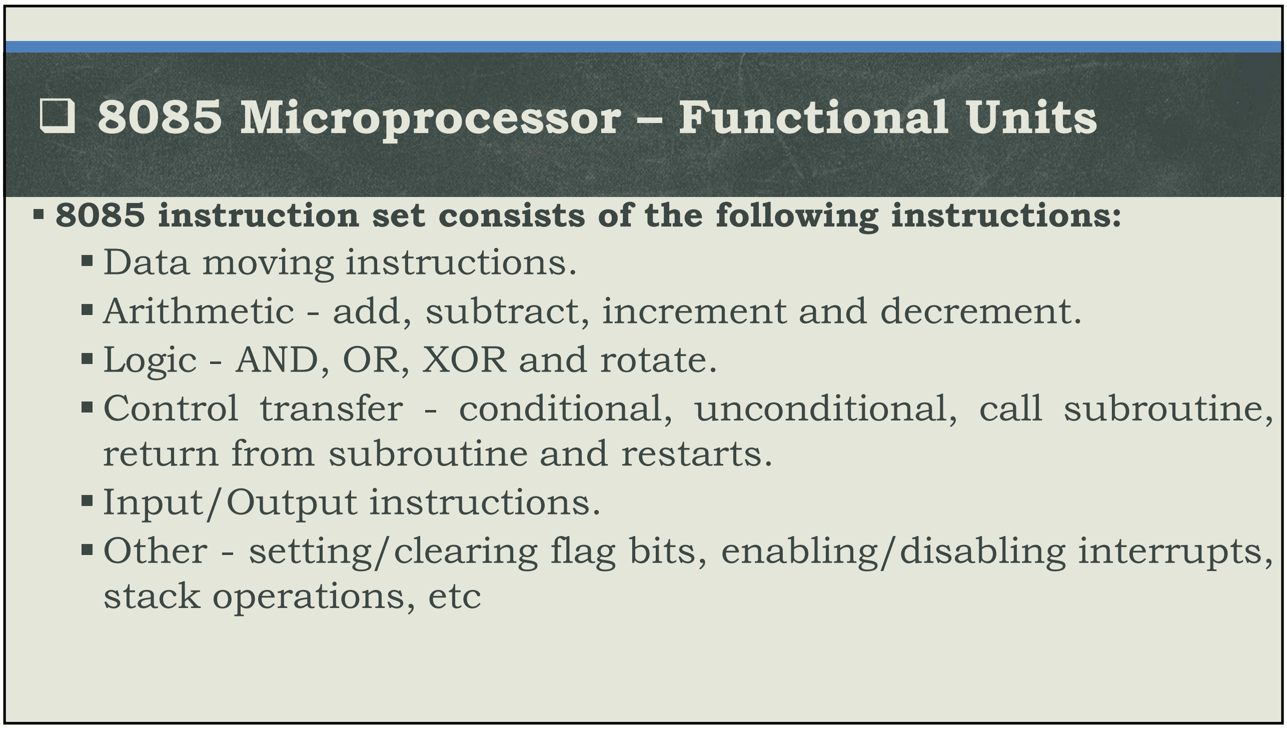
❑ 8085 Microprocessor – Functional Units

- **Serial Input/output control**
- It controls the serial data communication by using these two instructions: SID (Serial input data) and SOD (Serial output data).
- **Address buffer and address-data buffer**
- The content stored in the stack pointer and program counter is loaded into the address buffer and address-data buffer to communicate with the CPU.
- The memory and I/O chips are connected to these buses; the CPU can exchange the desired data with the memory and I/O chips.



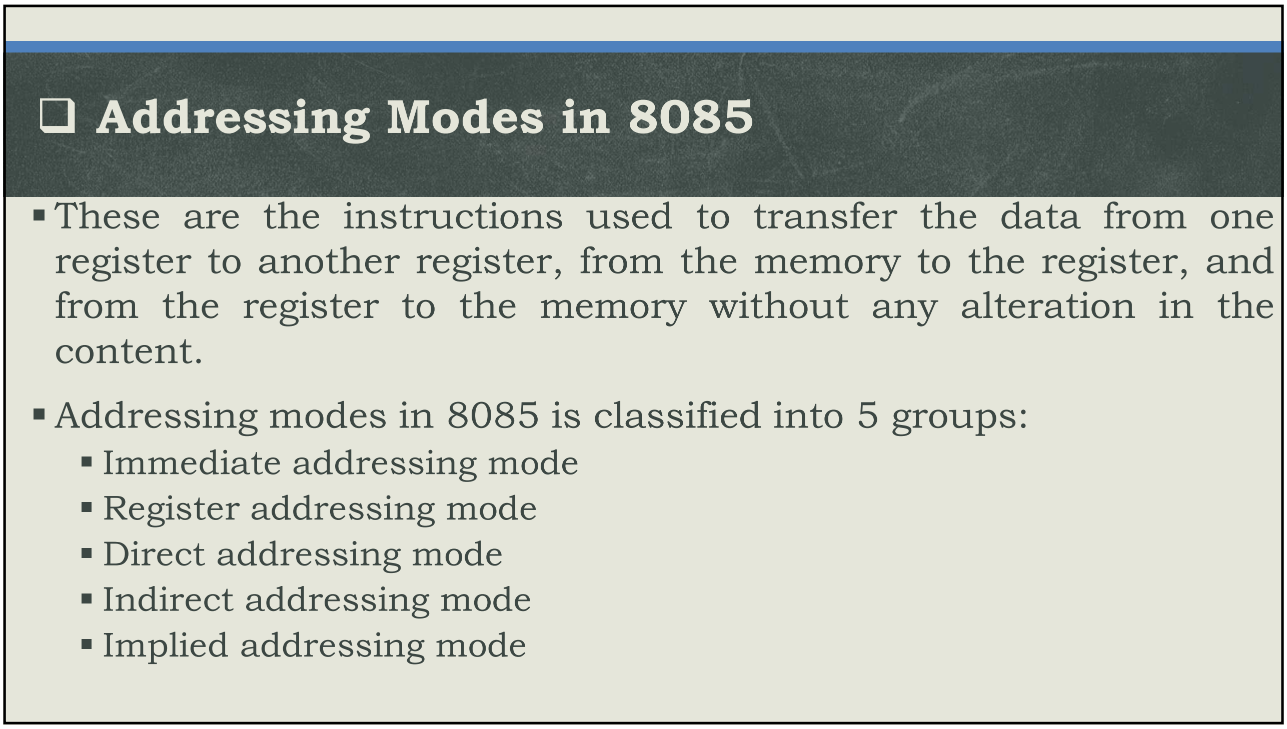
❑ 8085 Microprocessor – Functional Units

- **Address bus and data bus**
- Data bus carries the data to be stored. It is bidirectional, whereas address bus carries the location to where it should be stored and it is unidirectional.
- It is used to transfer the data & Address I/O devices.



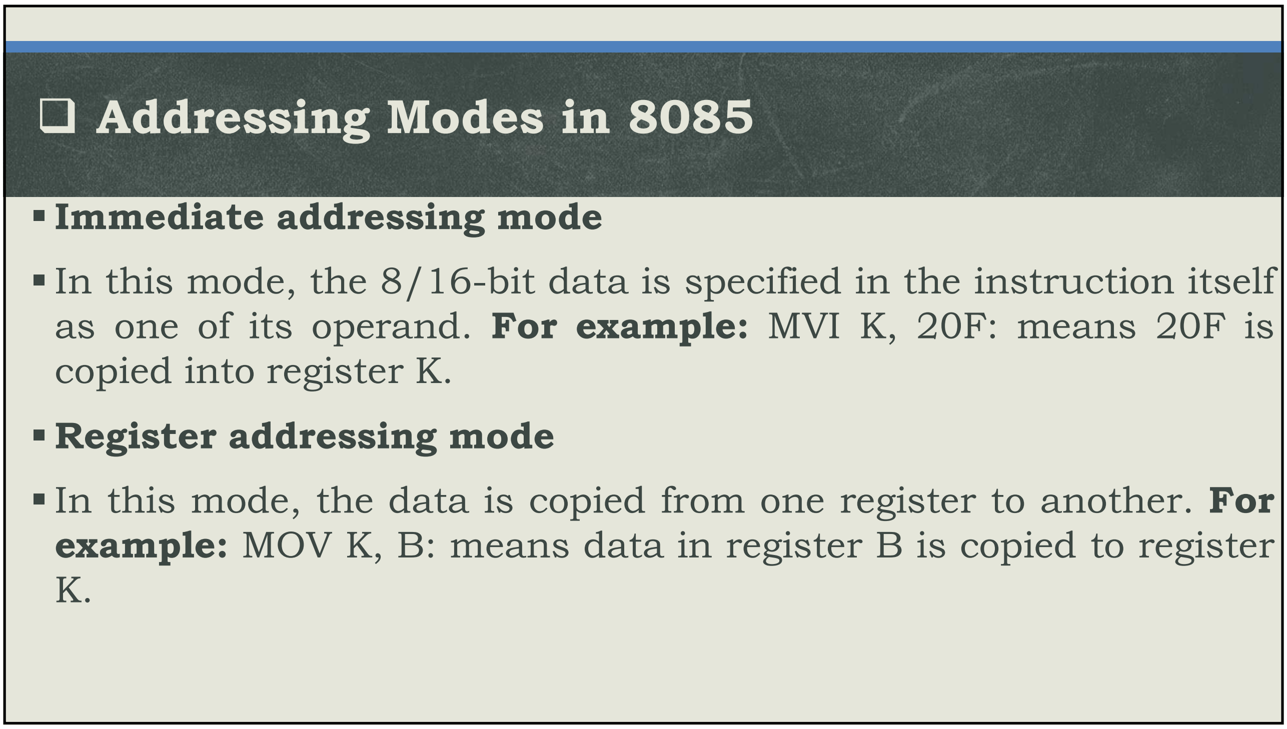
❑ 8085 Microprocessor – Functional Units

- **8085 instruction set consists of the following instructions:**
  - Data moving instructions.
  - Arithmetic - add, subtract, increment and decrement.
  - Logic - AND, OR, XOR and rotate.
  - Control transfer - conditional, unconditional, call subroutine, return from subroutine and restarts.
  - Input/Output instructions.
  - Other - setting/clearing flag bits, enabling/disabling interrupts, stack operations, etc



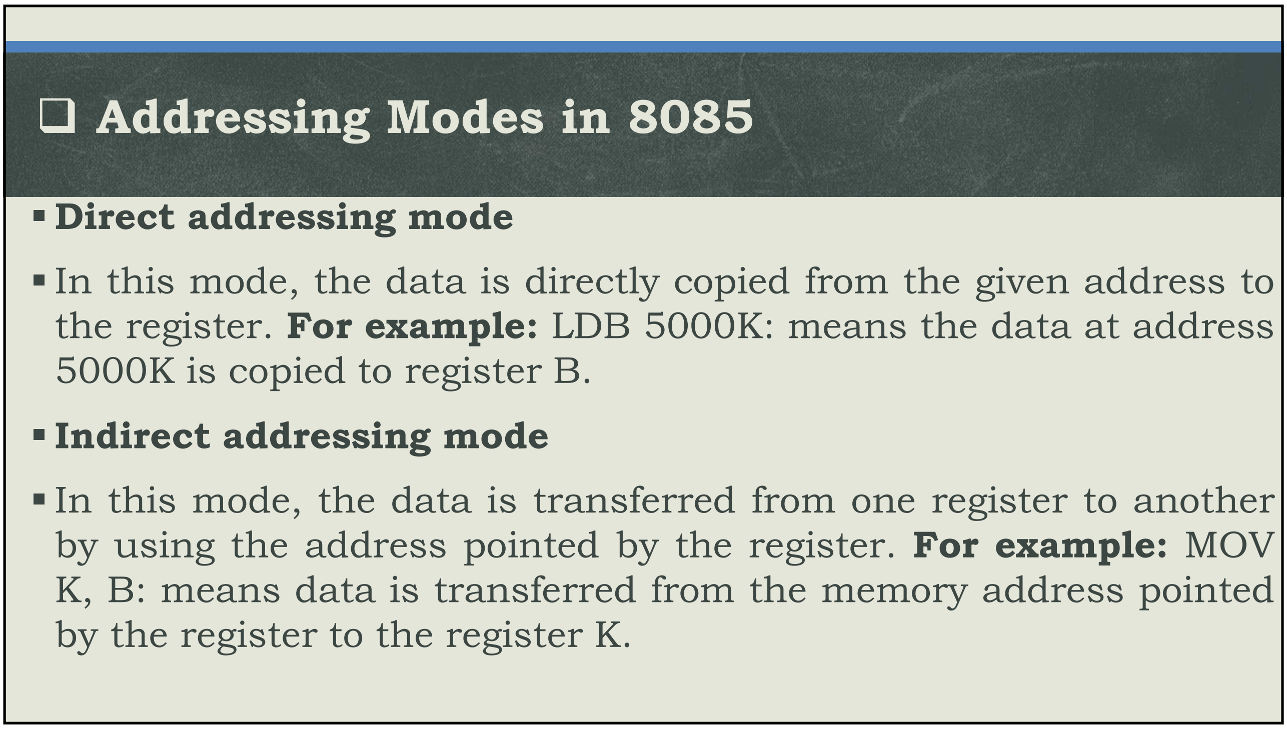
❑ Addressing Modes in 8085

- These are the instructions used to transfer the data from one register to another register, from the memory to the register, and from the register to the memory without any alteration in the content.
- Addressing modes in 8085 is classified into 5 groups:
  - Immediate addressing mode
  - Register addressing mode
  - Direct addressing mode
  - Indirect addressing mode
  - Implied addressing mode



❑ Addressing Modes in 8085

- **Immediate addressing mode**
- In this mode, the 8/16-bit data is specified in the instruction itself as one of its operand. **For example:** MVI K, 20F: means 20F is copied into register K.
- **Register addressing mode**
- In this mode, the data is copied from one register to another. **For example:** MOV K, B: means data in register B is copied to register K.



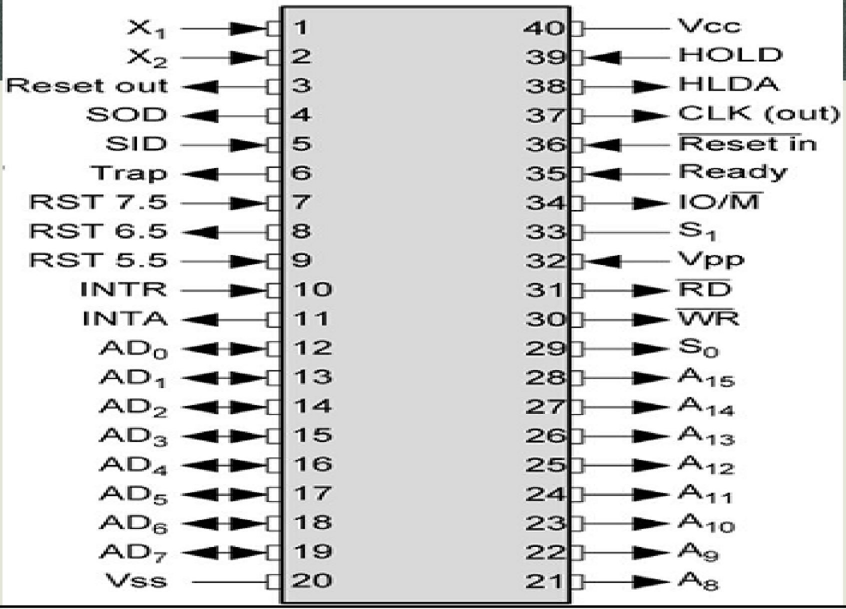
❑ Addressing Modes in 8085

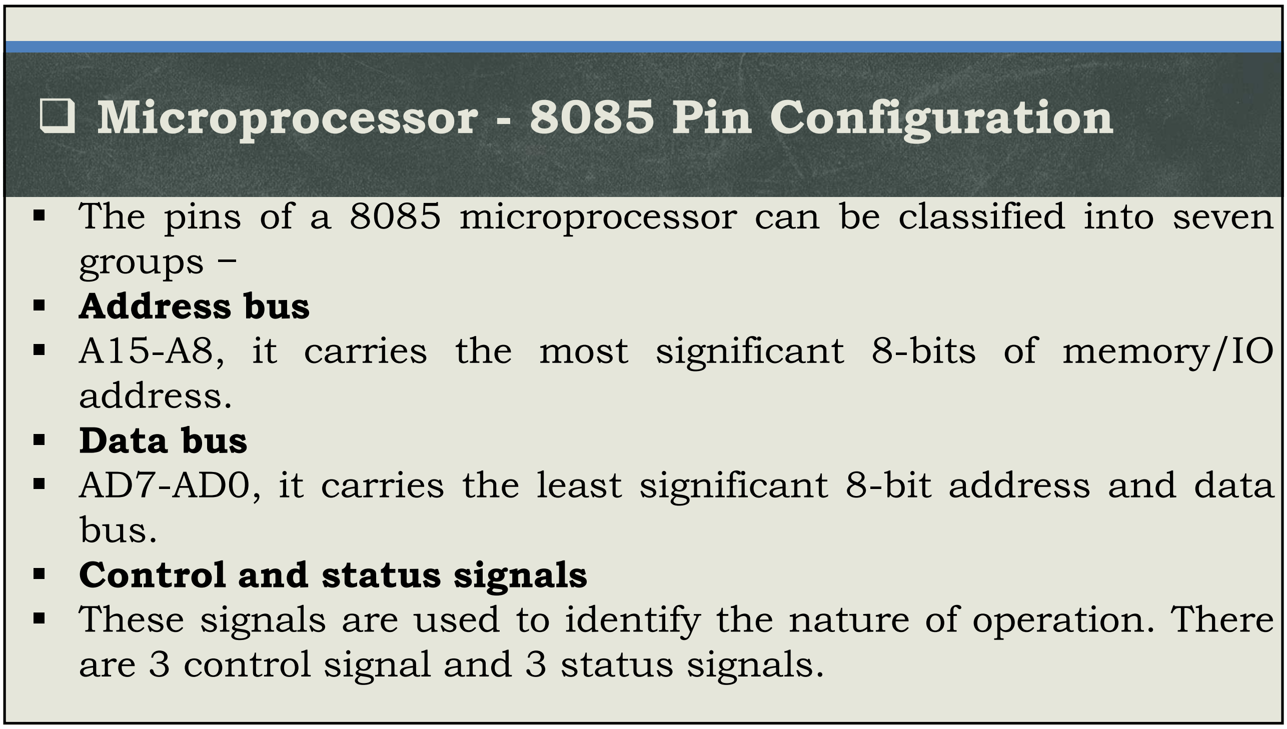
- **Direct addressing mode**
- In this mode, the data is directly copied from the given address to the register. **For example:** LDB 5000K: means the data at address 5000K is copied to register B.
- **Indirect addressing mode**
- In this mode, the data is transferred from one register to another by using the address pointed by the register. **For example:** MOV K, B: means data is transferred from the memory address pointed by the register to the register K.

❑ Addressing Modes in 8085

- **Implied addressing mode**
- This mode doesn’t require any operand; the data is specified by the opcode itself. **For example:** CMA (finds and stores the 1’s complement of the contains of accumulator A in A).

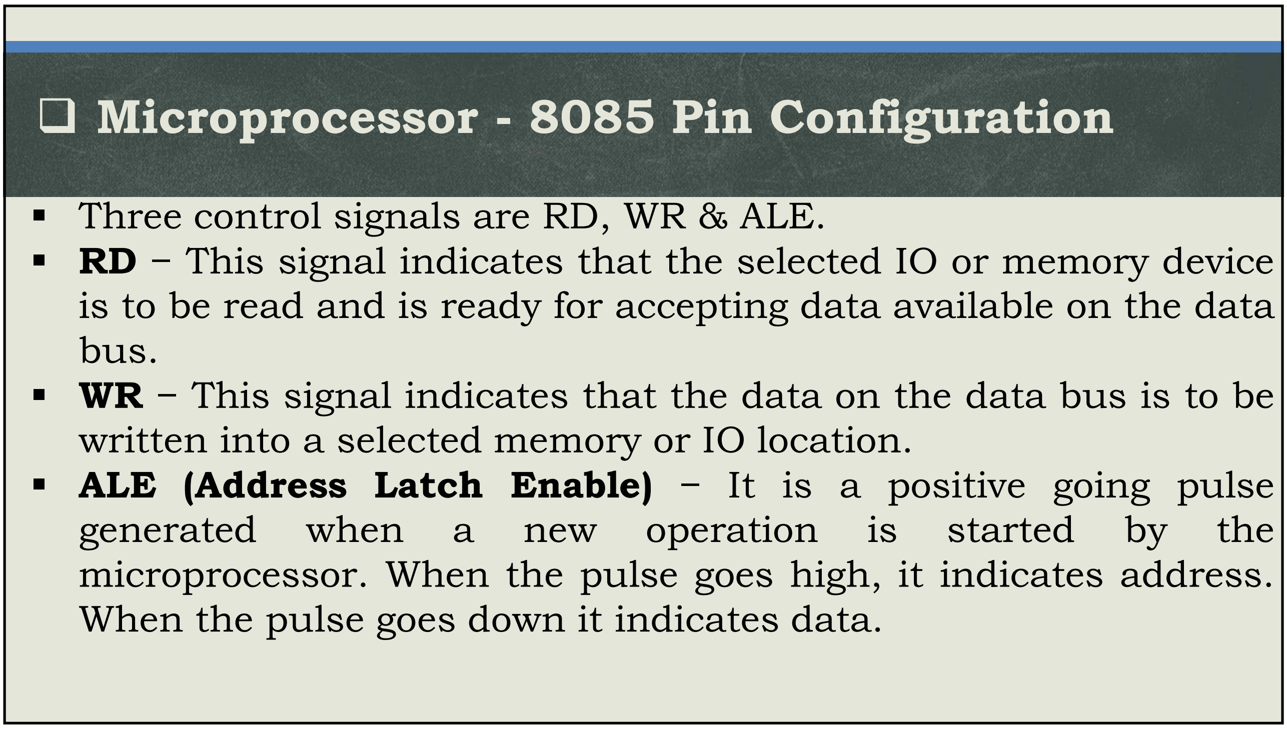
❑ Microprocessor - 8085 Pin Configuration





❑ Microprocessor - 8085 Pin Configuration

- The pins of a 8085 microprocessor can be classified into seven groups −
- **Address bus**
- A15-A8, it carries the most significant 8-bits of memory/IO address.
- **Data bus**
- AD7-AD0, it carries the least significant 8-bit address and data bus.
- **Control and status signals**
- These signals are used to identify the nature of operation. There are 3 control signal and 3 status signals.



❑ Microprocessor - 8085 Pin Configuration

- Three control signals are RD, WR & ALE.
- **RD** − This signal indicates that the selected IO or memory device is to be read and is ready for accepting data available on the data bus.
- **WR** − This signal indicates that the data on the data bus is to be written into a selected memory or IO location.
- **ALE (Address Latch Enable)** − It is a positive going pulse generated when a new operation is started by the microprocessor. When the pulse goes high, it indicates address. When the pulse goes down it indicates data.

❑ Microprocessor - 8085 Pin Configuration

- Three status signals are IO/M, S0 & S1.
- **IO/M**
- This signal is used to differentiate between IO and Memory operations, i.e. when it is high indicates IO operation and when it is low then it indicates memory operation.
- **S1 & S0**
- These signals are used to identify the type of current operation.
- **Power supply**
- There are 2 power supply signals – VCC & VSS. VCC indicates +5v power supply and VSS indicates ground signal.

❑ Microprocessor - 8085 Pin Configuration

- **Clock signals**
- There are 3 clock signals, i.e. X1, X2, CLK OUT.
- **X1, X2** – It is used to set frequency of the internal clock generator. This frequency is internally divided by 2.
- **CLK OUT** – This signal is used as the system clock for devices connected with the microprocessor.

❑ Microprocessor - 8085 Pin Configuration

- **Interrupts & externally initiated signals**
- Interrupts are the signals generated by external devices to request the microprocessor to perform a task. There are 5 interrupt signals, i.e. TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR.
- **INTA** – It is an interrupt acknowledgment signal.
- **RESET IN** – This signal is used to reset the microprocessor by setting the program counter to zero.
- **RESET OUT** – This signal is used to reset all the connected devices when the microprocessor is reset.

❑ Microprocessor - 8085 Pin Configuration

- **READY –** This signal indicates that the device is ready to send or receive data. If READY is low, then the CPU has to wait for READY to go high.

- **HOLD –** This signal indicates that another master is requesting the use of the address and data buses.

- **HLDA (HOLD Acknowledge) –** It indicates that the CPU has received the HOLD request and it will hand over the bus in the next clock cycle.

❑ Microprocessor - 8085 Pin Configuration

- **Serial I/O signals**
- There are 2 serial signals, i.e. SID and SOD and these signals are used for serial communication.
- **SOD** (Serial output data line) − The output SOD is set/reset as specified by the SIM (Set Interrupt Mask) instruction.
- **SID** (Serial input data line) − The data on this line is loaded into accumulator whenever a RIM (Read Interrupt Mask) instruction is executed.

❑ Interrupts in 8085

- Interrupts are the signals generated by the external devices to request the microprocessor to perform a task.
- There are 5 interrupt signals, i.e. TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR.
- Interrupt are classified into following groups based on their parameter:
- **Vector interrupt** − In this type of interrupt, the interrupt address is known to the processor. **For example:** RST7.5, RST6.5, RST5.5, TRAP.
- **Non-Vector interrupt** − In this type of interrupt, the interrupt address is not known to the processor so, the interrupt address needs to be sent externally by the device to perform interrupts. **For example:** INTR.

❑ Interrupts in 8085

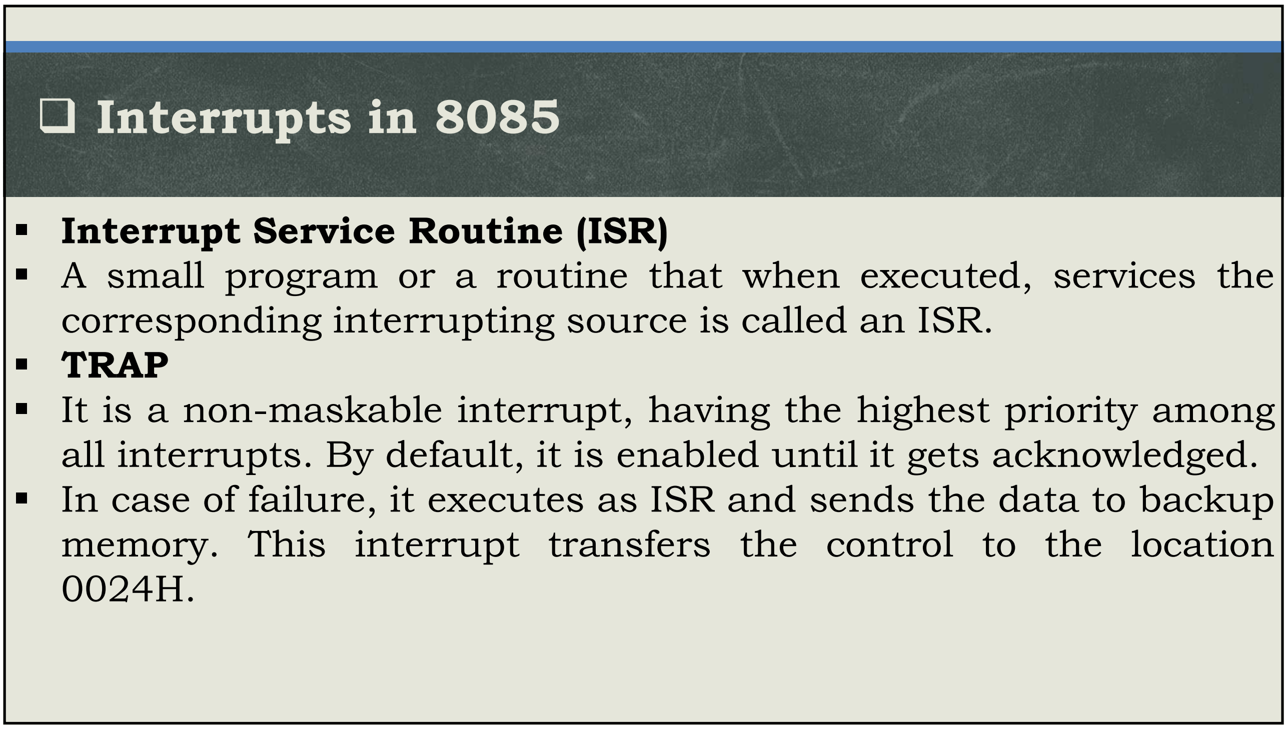
- **Maskable interrupt** – In this type of interrupt, we can disable the interrupt by writing some instructions into the program. For example: RST7.5, RST6.5, RST5.5.

- **Non-Maskable interrupt** – In this type of interrupt, we cannot disable the interrupt by writing some instructions into the program. For example: TRAP.

❑ Interrupts in 8085

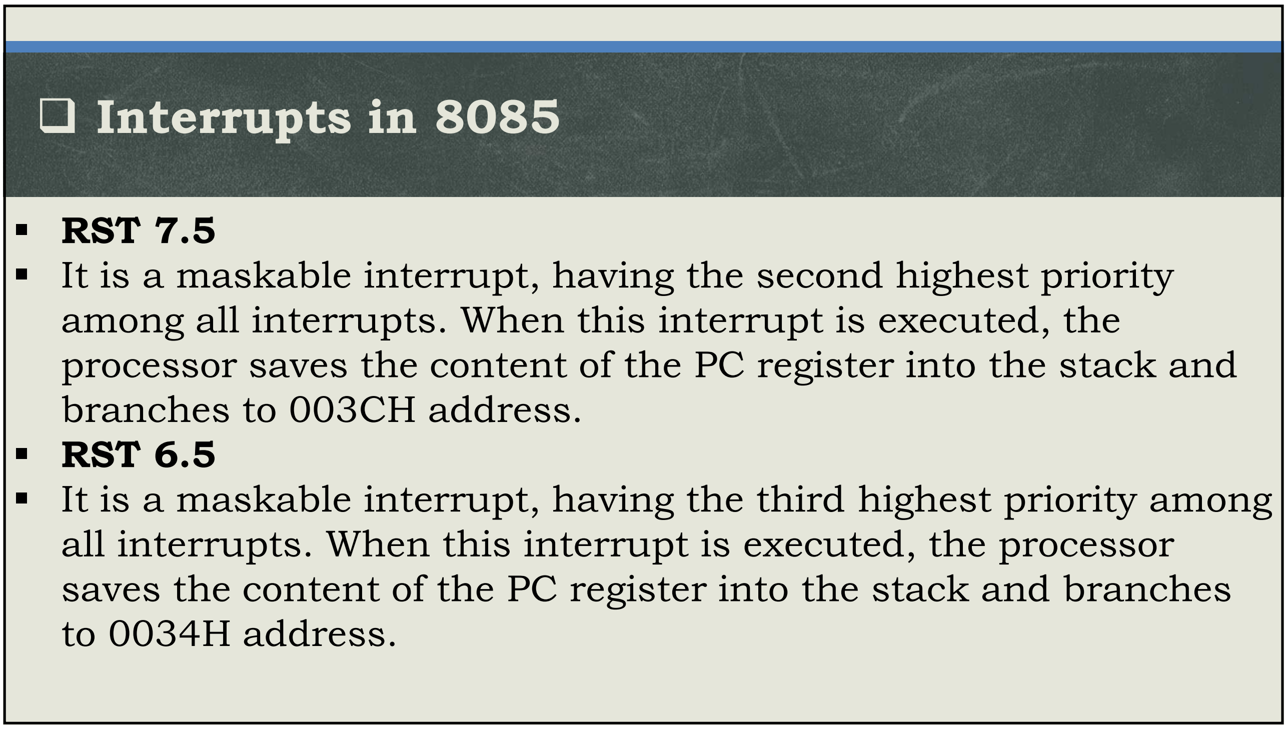
- **Software interrupt** – In this type of interrupt, the programmer has to add the instructions into the program to execute the interrupt. There are 8 software interrupts in 8085, i.e. RST0, RST1, RST2, RST3, RST4, RST5, RST6, and RST7.

- **Hardware interrupt** – There are 5 interrupt pins in 8085 used as hardware interrupts, i.e. TRAP, RST7.5, RST6.5, RST5.5, INTA.



❑ Interrupts in 8085

- **Interrupt Service Routine (ISR)**
- A small program or a routine that when executed, services the corresponding interrupting source is called an ISR.
- **TRAP**
- It is a non-maskable interrupt, having the highest priority among all interrupts. By default, it is enabled until it gets acknowledged.
- In case of failure, it executes as ISR and sends the data to backup memory. This interrupt transfers the control to the location 0024H.



❑ Interrupts in 8085

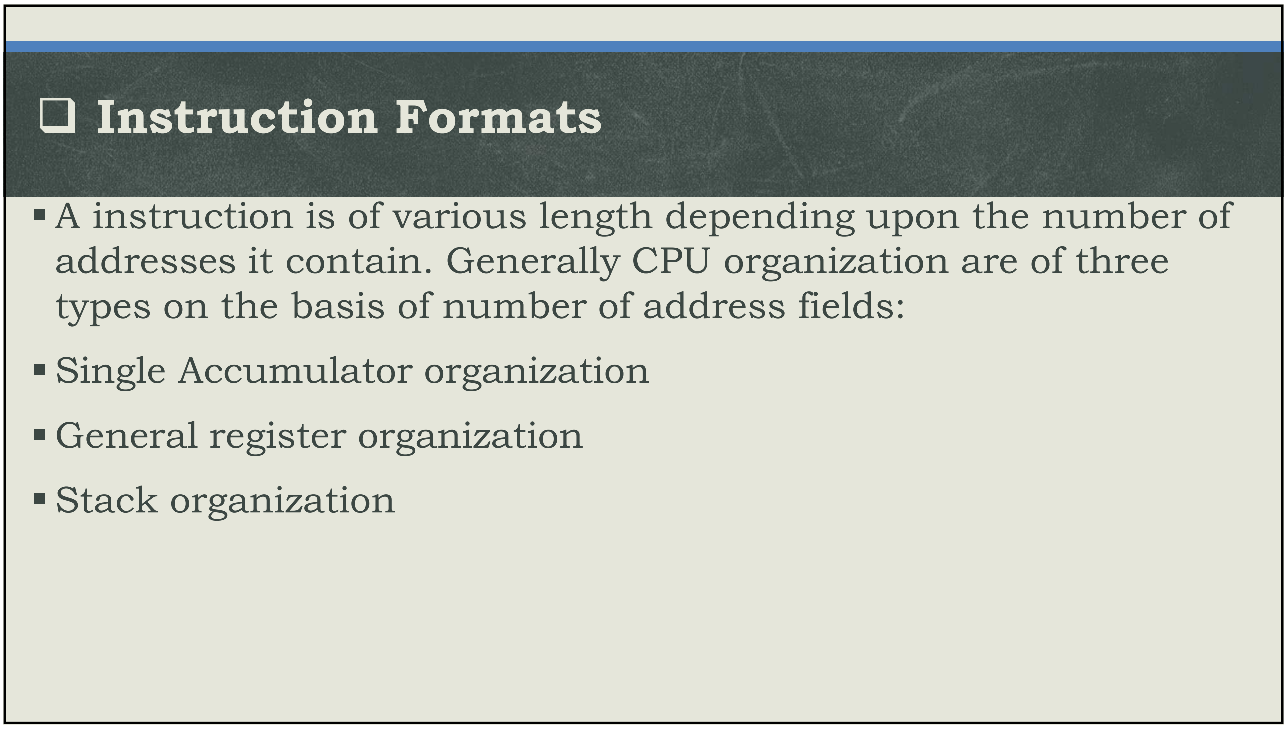
- **RST 7.5**
- It is a maskable interrupt, having the second highest priority among all interrupts. When this interrupt is executed, the processor saves the content of the PC register into the stack and branches to 003CH address.
- **RST 6.5**
- It is a maskable interrupt, having the third highest priority among all interrupts. When this interrupt is executed, the processor saves the content of the PC register into the stack and branches to 0034H address.

❑ Interrupts in 8085

- **RST 5.5**
- It is a maskable interrupt. When this interrupt is executed, the processor saves the content of the PC register into the stack and branches to 002CH address.
- **INTR**
- It is a maskable interrupt, having the lowest priority among all interrupts. It can be disabled by resetting the microprocessor.

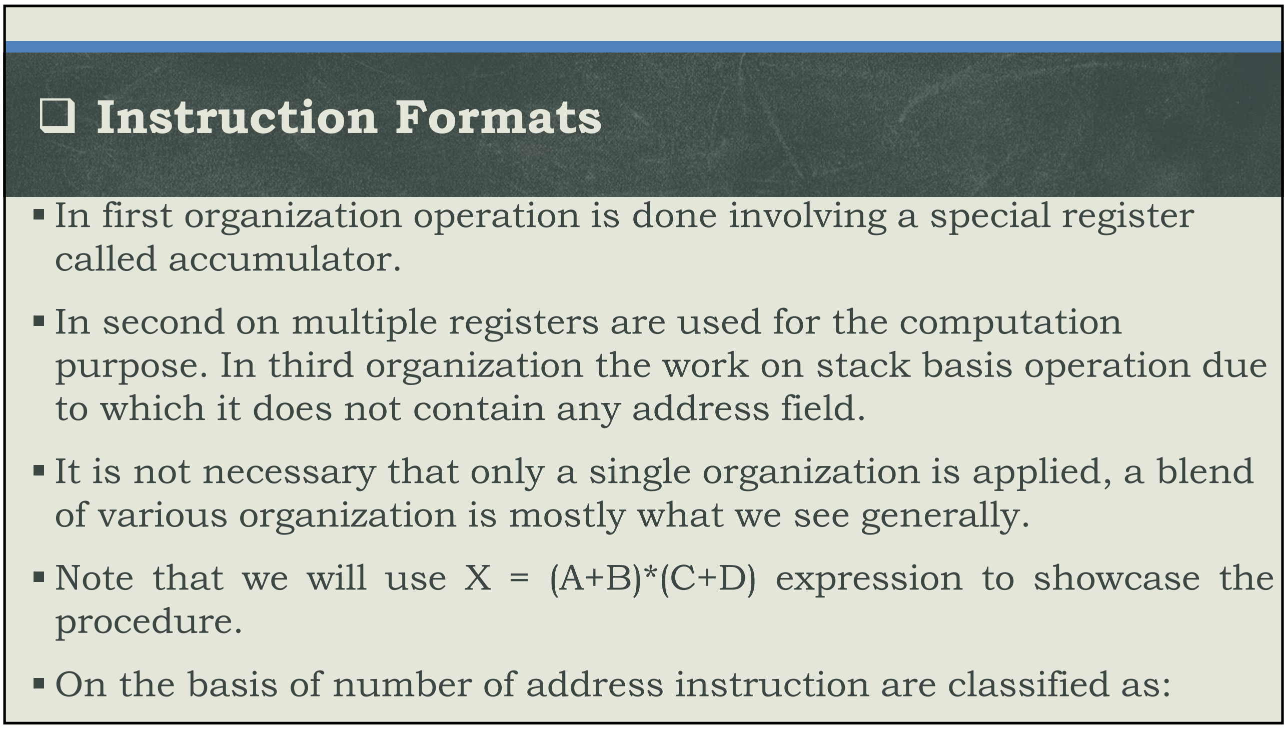
❑ Instruction Formats

- Computer perform task on the basis of instruction provided. A instruction in computer comprises of groups called fields.
- These field contains different information as for computers every thing is in 0 and 1 so each field has different significance on the basis of which a CPU decide what so perform. The most common fields are:
- **Operation field** which specifies the operation to be performed like addition.
- **Address field** which contain the location of operand, i.e., register or memory location.
- **Mode field** which specifies how operand is to be founded.



❑ Instruction Formats

- A instruction is of various length depending upon the number of addresses it contain. Generally CPU organization are of three types on the basis of number of address fields:
- Single Accumulator organization
- General register organization
- Stack organization



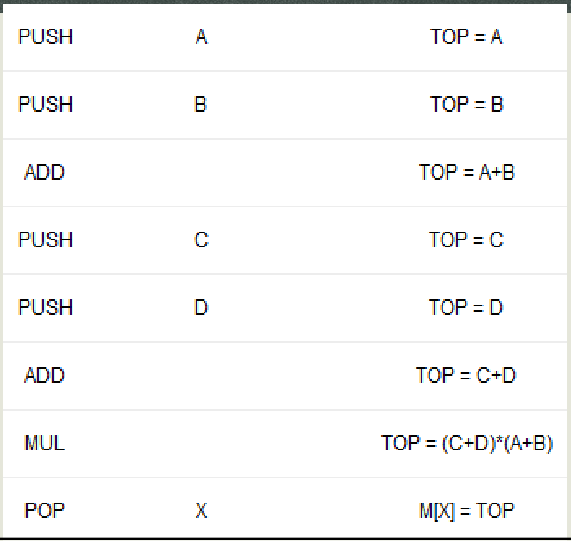
❑ Instruction Formats

- In first organization operation is done involving a special register called accumulator.
- In second on multiple registers are used for the computation purpose. In third organization the work on stack basis operation due to which it does not contain any address field.
- It is not necessary that only a single organization is applied, a blend of various organization is mostly what we see generally.
- Note that we will use X = (A+B)*(C+D) expression to showcase the procedure.
- On the basis of number of address instruction are classified as:

❑ Instruction Formats

- **1. Zero Address Instructions**
- A stack based computer do not use address field in instruction. To evaluate a expression first it is converted to revere Polish Notation i.e. Post fix Notation.
- Expression: X = (A+B)*(C+D)
- Postfixed : X = AB+CD+*
- TOP means top of stack
- M[X] is any memory location

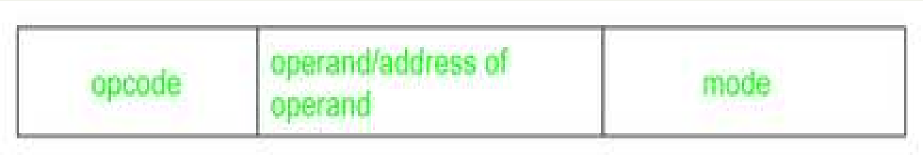
❑ Instruction Formats



PUSH	A	TOP = A
PUSH	B	TOP = B
ADD		TOP = A+B
PUSH	C	TOP = C
PUSH	D	TOP = D
ADD		TOP = C+D
MUL		TOP = (C+D)*(A+B)
POP	X	M[X] = TOP

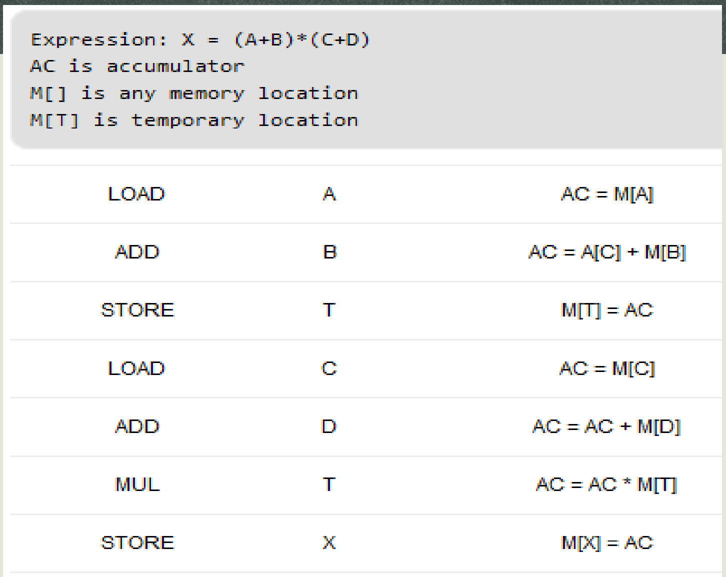
❑ Instruction Formats

- **2. One Address Instructions:**
- This use a implied ACCUMULATOR register for data manipulation.
- One operand is in accumulator and other is in register or memory location.
- Implied means that the CPU already know that one operand is in accumulator so there is no need to specify it.



opcode	operand/address of operand	mode

❑ Instruction Formats

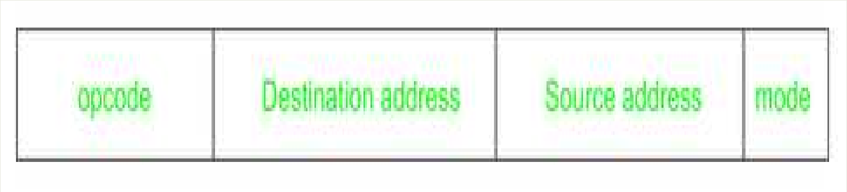


```
Expression: X = (A+B)*(C+D)
AC is accumulator
M[] is any memory location
M[T] is temporary location
```

LOAD	A	AC = M[A]
ADD	B	AC = A[C] + M[B]
STORE	T	M[T] = AC
LOAD	C	AC = M[C]
ADD	D	AC = AC + M[D]
MUL	T	AC = AC * M[T]
STORE	X	M[X] = AC

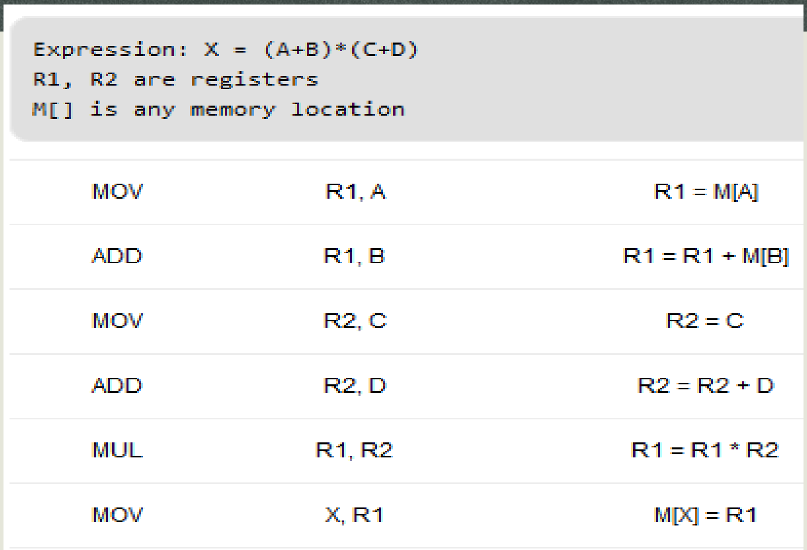
❑ Instruction Formats

- **3. Two Address Instructions:**
- This is common in commercial computers. Here two address can be specified in the instruction.
- Unlike earlier in one address instruction the result was stored in accumulator here result cab be stored at different location rather than just accumulator, but require more number of bit to represent address.



opcode	Destination address	Source address	mode

❑ Instruction Formats

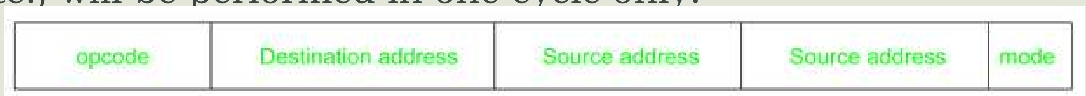


```
Expression: X = (A+B)*(C+D)
R1, R2 are registers
M[] is any memory location
```

MOV	R1, A	R1 = M[A]
ADD	R1, B	R1 = R1 + M[B]
MOV	R2, C	R2 = C
ADD	R2, D	R2 = R2 + D
MUL	R1, R2	R1 = R1 * R2
MOV	X, R1	M[X] = R1

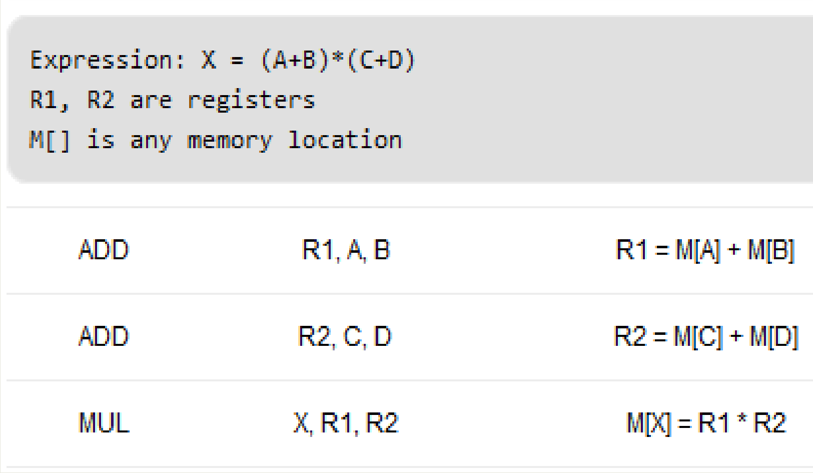
❑ Instruction Formats

- **4. Three Address Instructions:**
- This has three address field to specify a register or a memory location. Program created are much short in size but number of bits per instruction increase.
- These instructions make creation of program much easier but it does not mean that program will run much faster because now instruction only contain more information but each micro operation (changing content of register, loading address in address bus etc.) will be performed in one cycle only.



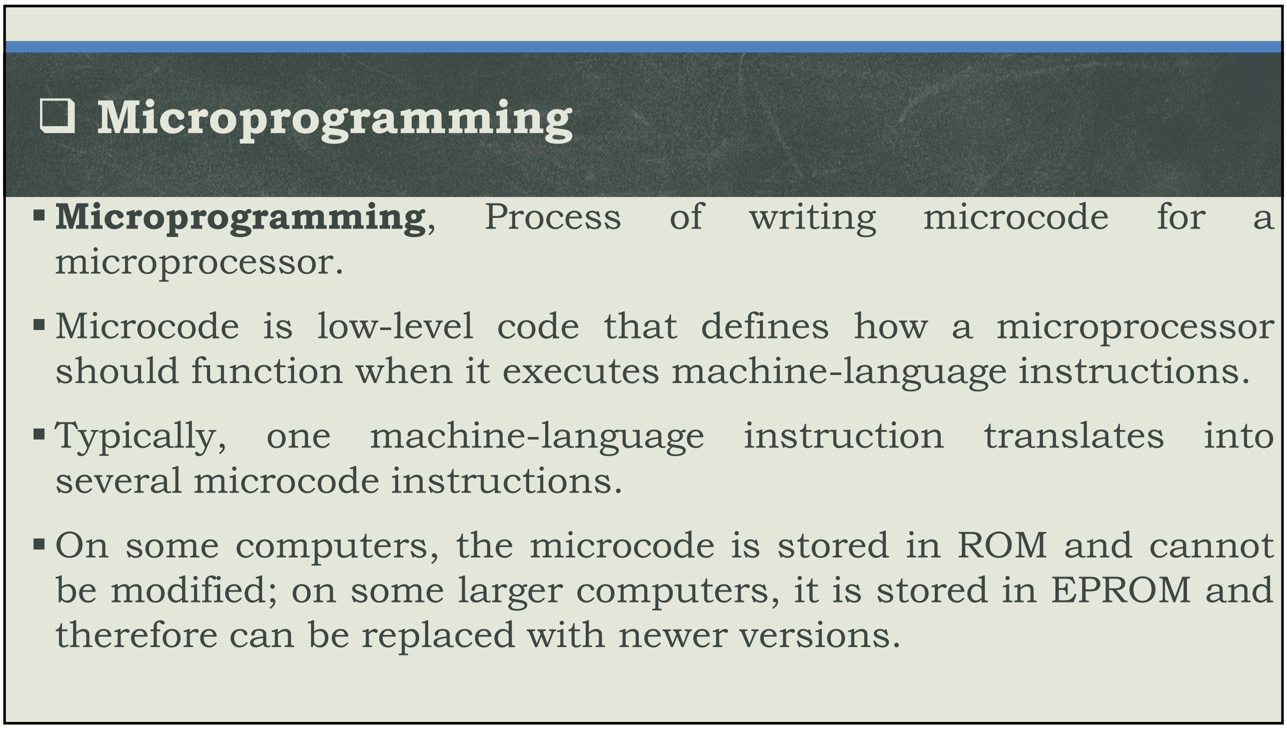
opcode	Destination address	Source address	Source address	mode

❑ Instruction Formats



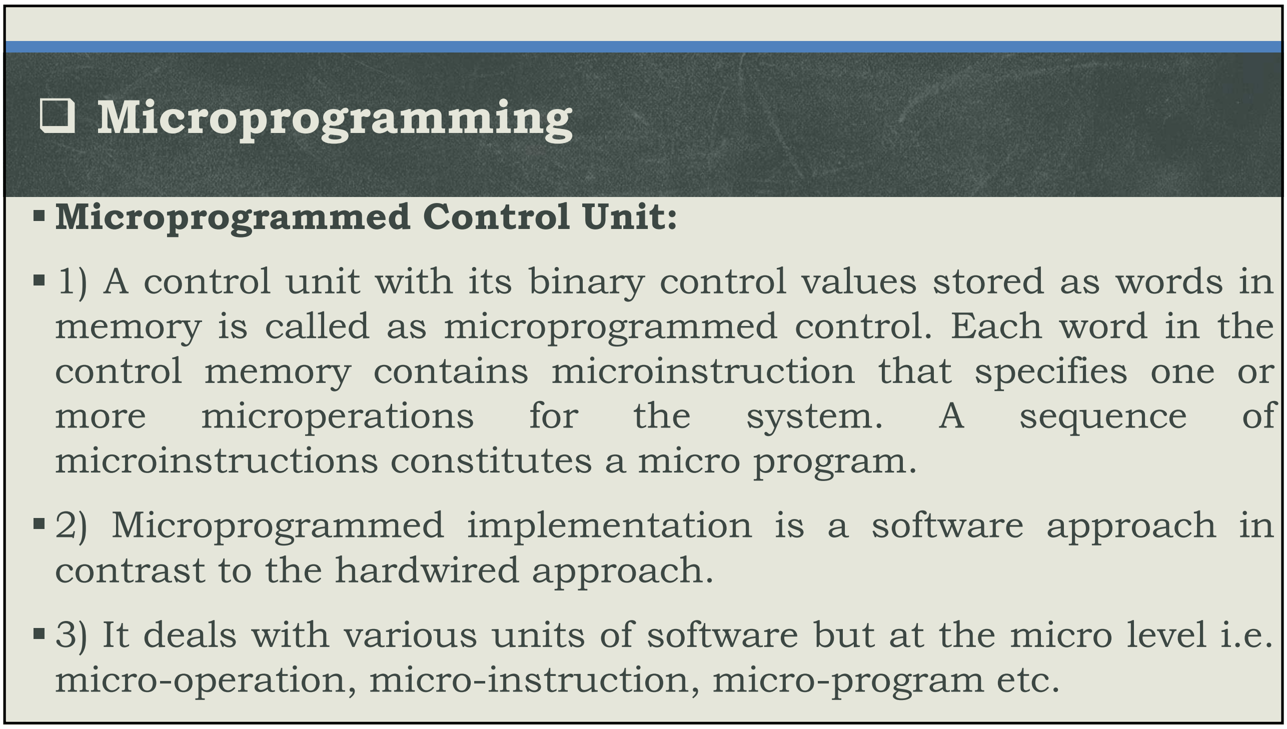
```
Expression: X = (A+B)*(C+D)
R1, R2 are registers
M[] is any memory location
```

ADD	R1, A, B	R1 = M[A] + M[B]
ADD	R2, C, D	R2 = M[C] + M[D]
MUL	X, R1, R2	M[X] = R1 * R2



❑ Microprogramming

- **Microprogramming**, Process of writing microcode for a microprocessor.
- Microcode is low-level code that defines how a microprocessor should function when it executes machine-language instructions.
- Typically, one machine-language instruction translates into several microcode instructions.
- On some computers, the microcode is stored in ROM and cannot be modified; on some larger computers, it is stored in EPROM and therefore can be replaced with newer versions.

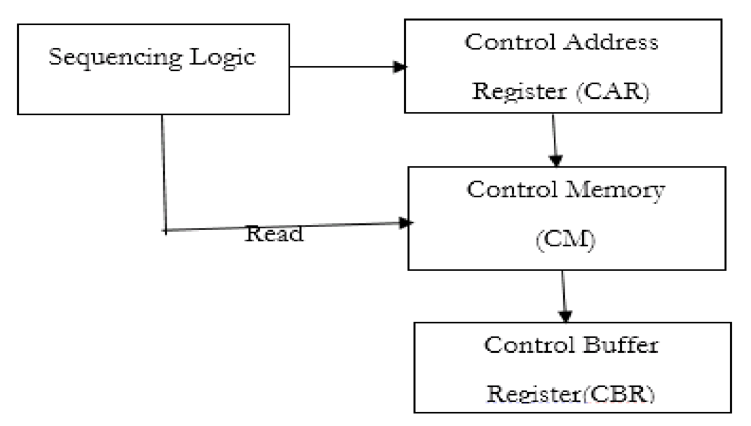


❑ Microprogramming

- **Microprogrammed Control Unit:**
- 1) A control unit with its binary control values stored as words in memory is called as microprogrammed control. Each word in the control memory contains microinstruction that specifies one or more microperations for the system. A sequence of microinstructions constitutes a micro program.
- 2) Microprogrammed implementation is a software approach in contrast to the hardwired approach.
- 3) It deals with various units of software but at the micro level i.e. micro-operation, micro-instruction, micro-program etc.

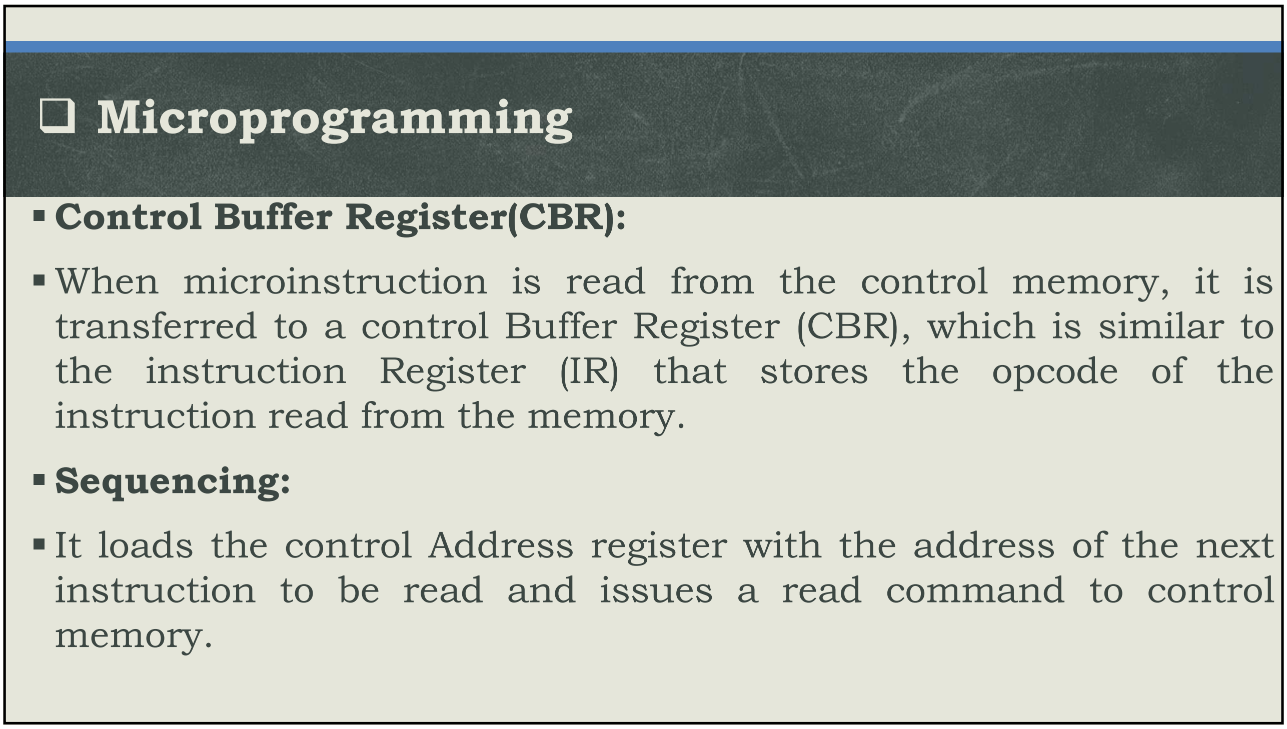
❑ Microprogramming

- 4) Different key elements used for implementation of a control unit using microprogrammed approach is shown in diagram below:



❑ Microprogramming

- **Control Memory:**
- The set of microinstruction is stored in control Memory (CM) also called as control store.
- **Control Address Register (CAR):**
- It contains the address of next microinstruction to be read. This is similar to the program counter(PC) which stores the address of the next instruction.



❑ Microprogramming

- **Control Buffer Register(CBR):**
- When microinstruction is read from the control memory, it is transferred to a control Buffer Register (CBR), which is similar to the instruction Register (IR) that stores the opcode of the instruction read from the memory.
- **Sequencing:**
- It loads the control Address register with the address of the next instruction to be read and issues a read command to control memory.