imp Definations

i) Address Bus

- The **address bus** is a set of parallel wires used to transmit memory addresses. It is unidirectional, meaning data flows only in one direction (from CPU to memory or I/O devices). The number of lines in the address bus determines the addressable memory (e.g., a 16-bit address bus can address up to 65,536 memory locations).

ii) ALE (Address Latch Enable)

- **ALE** is a control signal used in microprocessors to separate the multiplexed address/data bus. In a multiplexed bus system, the lower address bits and data bits are shared. The ALE signal helps to latch the lower address bits into a register, enabling the processor to work with the full 16-bit address.

iii) Stack Pointer

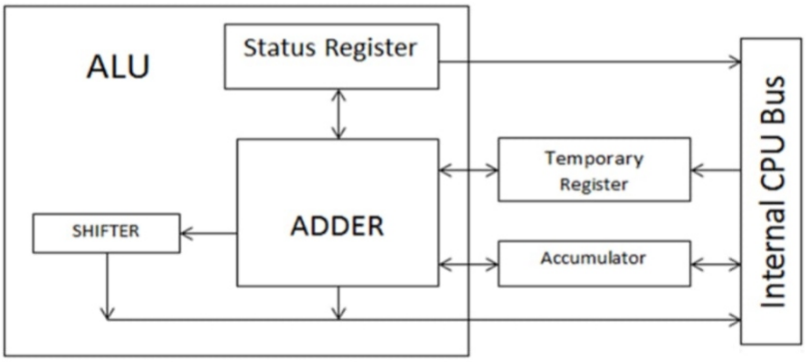
- The **stack pointer (SP)** is a special-purpose 16-bit register that holds the address of the top of the stack in memory. The stack is used to store temporary data such as return addresses, parameters, and local variables during function calls. The stack grows downwards in memory.

Q. Organization of ALU with a simple block diagram

The **Arithmetic Logic Unit (ALU)** is responsible for performing arithmetic and logical operations in the CPU. Here’s a breakdown of its functions:

1. **Input Registers**: ALU receives data from these registers.
2. **Arithmetic Unit**: Performs arithmetic operations like addition, subtraction, etc.
3. **Logic Unit**: Performs logical operations like AND, OR, NOT, etc.
4. **Multiplexer**: Selects the desired operation based on control signals.
5. **Accumulator**: Stores the result of operations.
6. **Flags**: The ALU updates the status flags (Carry, Zero, Sign, Parity, etc.) after every operation.

Block Diagram:



Q. Addressing Modes of 8085

Addressing modes of Intel 8085

1. **Immediate Addressing Mode**
    - In this mode, the operand (data) is given directly in the instruction.
    - Example: `MVI A, 45H`
    - The value `45H` is moved directly into the accumulator.
2. **Direct Addressing Mode**
    - The operand is located at a memory address specified directly in the instruction.
    - Example: `LDA 2050H`
    - The contents of memory location `2050H` are loaded into the accumulator.
3. **Register Addressing Mode**
    - The operand is stored in one of the processor's registers.
    - Example: `MOV A, B`
    - The contents of register B are moved into register A.
4. **Indirect Addressing Mode**
    - The operand is at the address pointed to by a register pair.

    - Example: `MOV A, M`
    - The contents of the memory location addressed by the HL pair are moved into the accumulator.

5. **Implied Addressing Mode**
    - In this mode, the operand is implied and not explicitly specified in the instruction.
    - Example: `NOP` (No operation), `HLT` (Halt).

Q. Difference between SUB and CMP

While both instructions perform subtraction internally, they serve different purposes:

Feature	SUB (Subtract)	CMP (Compare)
Operation	$A = A - \text{Reg/Mem}$	$A - \text{Reg/Mem}$ (Internal)
Result	Result is stored in the Accumulator.	Result is discarded; Accumulator remains unchanged.
Flags	All flags are affected based on the result.	Flags are set to show the relationship ($<, >, =$).
Usage	Used when you need the actual difference.	Used for decision making (loops/conditional jumps).

Q. Interrupt System of 8085

An interrupt is a signal sent by an external device to the CPU to request service.

- **Hardware Interrupts:** 8085 has five pins for hardware interrupts:
    1. **TRAP:** Highest priority, Edge and Level triggered, **Non-maskable** (cannot be disabled).
    2. **RST 7.5:** Second highest, Edge triggered, Maskable.
    3. **RST 6.5:** Third highest, Level triggered, Maskable.
    4. **RST 5.5:** Fourth highest, Level triggered, Maskable.

5. **INTR:** Lowest priority, Level triggered, Maskable.

- **Software Interrupts:** Instructions like `RST 0` to `RST 7` used to transfer control to specific vector locations.

Q. What is Protocol? Explain TCP/IP

Protocol: A protocol is a formal set of rules and conventions that govern how data is exchanged between devices over a network. Without protocols, devices might send data, but the receiver would not know how to interpret it.

TCP/IP (Transmission Control Protocol / Internet Protocol):

This is the standard communication suite of the Internet. It consists of two main parts:

1. **TCP (Transmission Control Protocol):**
   - Breaks data into small **packets**.
   - Ensures reliable delivery by checking for errors and re-sending lost packets.
   - It is **connection-oriented**.
2. **IP (Internet Protocol):**
   - Responsible for **addressing** and **routing**.
   - Ensures packets are sent to the correct destination IP address.
   - It is **connectionless**.

Q Sort 25 Numbers in Ascending Order (Logic)

To sort 25 numbers stored from `C000H`, we use the **Bubble Sort** algorithm.

Code snippet

```
      LXI H, C000H    ; Point to start of data
      MVI D, 19H      ; Counter for Outer Loop (25 numbers = 24 comparisons)
OUTER:LXI H, C000H    ; Reset pointer for each pass
      MOV C, D        ; Counter for Inner Loop
INNER:MOV A, M        ; Get first number
```

```
      INX H           ; Point to next number
      CMP M           ; Compare A with next number
      JC SKIP         ; If A < M, no swap (already ascending)
      MOV B, M        ; --- SWAP LOGIC ---
      MOV M, A        ; Move larger to next position
      DCX H           ; Move back to original position
      MOV M, B        ; Move smaller to original position
      INX H           ; Restore pointer
SKIP: DCR C           ; Decrement inner counter
      JNZ INNER       ; Repeat inner loop
      DCR D           ; Decrement outer counter
      JNZ OUTER       ; Repeat until fully sorted
      HLT
```

Q. Fill Memory with 00 to 99 (Hex 63H)

This program fills memory locations starting from `AB01H` with a sequence of numbers from 0 to 99.

Code snippet

```
      MVI A, 00H      ; Start value = 0
      LXI H, AB01H    ; Starting memory address
      MVI B, 64H      ; Counter for 100 decimal (0 to 99)
LOOP: MOV M, A        ; Store current value in memory
      INX H           ; Increment memory pointer
      INR A           ; Increment the value to be stored
      DCR B           ; Decrement the counter
      JNZ LOOP        ; Jump back if counter not zero
      HLT
```

Short note on Modem

```
```

- **Modem** (Modulator-Demodulator) is a device that modulates digital signals from a computer into analog signals for transmission over phone lines, and demodulates the received analog signals back into digital signals.

- It allows computers to communicate over long distances, particularly in dial-up Internet connections.
- **Key Functions**:
  - **Modulation**: Converts digital signals to analog signals for transmission.
  - **Demodulation**: Converts incoming analog signals back into digital signals.

[redacted]

Short note on Co-axial Cable

[redacted]

- **Co-axial cable** (or coaxial cable) is used for transmitting high-frequency signals in communication systems like television, internet, and LAN.
- It consists of:
  - **Inner conductor**: A solid wire (usually copper) carrying the signal.
  - **Insulating layer**: A dielectric material that isolates the inner conductor.
  - **Outer conductor**: A braided shield that provides grounding and reduces electromagnetic interference (EMI).
  - **Outer insulating layer**: Protects the inner layers.
- **Advantages**:
  - Higher bandwidth compared to twisted pair cables.
  - Less susceptible to external interference (EMI).
  - Suitable for both analog and digital signal transmission.

[redacted]

the function of:

[redacted]

i) AD0 – AD7

- These are **multiplexed address/data lines** used for transferring both the address and data in Intel 8085. During the first part of a cycle, these lines carry the address, and during the second part, they carry the data.

ii) S0, S1 and IO/M

- **S0 and S1** are status signals used to define the type of operation being performed (whether it is reading/writing memory or I/O devices).
- **IO/M** indicates whether the operation is related to memory or I/O devices. If IO/M = 1, the operation is I/O; if IO/M = 0, it is a memory operation.

iii) HOLD and HLDA

- **HOLD** is an input signal used by external devices to request the bus.
- **HLDA** is the output signal that acknowledges the hold request. It indicates to the external device that the CPU is releasing the bus.

Intel 8085 instructions, covering their function, addressing modes, and impact on the processor.

[redacted]

[redacted]

1) ADD C (Add Register C to Accumulator)

[redacted]

This is an arithmetic instruction used to perform 8-bit addition.

[redacted]

- **Function:** It adds the contents of register **C** to the contents of the **Accumulator (A)**. The result is then stored back in the Accumulator.
- **Mathematical Expression:** $A \leftarrow A + C$
- **Addressing Mode:** Register Addressing Mode (as the operand is in a register).
- **Size:** 1 Byte.
- **Flags:** All status flags (Carry, Zero, Sign, Parity, Auxiliary Carry) are affected based on the result.
- Example:

  If $A = 10H$ and $C = 20H$:

  After ADD C, the Accumulator will contain $30H$.

[redacted]

2) NOP (No Operation)

[redacted]

As the name suggests, this instruction tells the CPU to do nothing for a specific period.

[redacted]

- **Function:** No operation is performed. The instruction is fetched and decoded, but no registers or memory contents are changed.
- **Purpose:**
  - To create small **time delays** in a program.
  - To "blank out" or delete unwanted instructions during debugging without shifting the rest of the code.
- **Addressing Mode:** None.
- **Size:** 1 Byte.
- **Machine Cycles:** It takes 1 Machine Cycle (4 T-states) to execute.
- **Flags:** No flags are affected.

---

3) LDAX D (Load Accumulator Indirect from DE Pair)

This is a data transfer instruction that uses "indirect" addressing.

- **Function:** It copies the data byte from a memory location into the **Accumulator**. The address of that memory location is contained in the **DE register pair**.
- **Instruction breakdown:** **L**oad **DA**ta to **X** (extended/register pair) **D** (DE pair).
- **Mathematical Expression:** $A \leftarrow [[D][E]]$
- **Addressing Mode:** Register Indirect Addressing Mode (the address is not in the instruction, but in a register pair).
- **Size:** 1 Byte.

- **Flags:** No flags are affected (data transfer instructions do not change flags).
- Example:

  Suppose Register $D = 20H$ and Register $E = 50H$.

  The DE pair points to memory address 2050H.

  If memory location 2050H contains the value F2H:

  After LDAX D, the Accumulator will contain F2H.

Microprocessor vs Microcontroller

Feature	Microprocessor (e.g., 8085)	Microcontroller (e.g., 8051)
Core Components	Contains only the CPU.	Contains CPU, RAM, ROM, I/O, and Timers on one chip.
External Parts	Requires external RAM, ROM, and I/O timers to function.	Minimal external parts needed for a working system.
Application	General-purpose (PCs, Laptops).	Application-specific (Washing machines, Microwaves).
Cost & Power	Higher cost and power consumption.	Lower cost and power consumption.

prominent features Pentium processor

was a major step forward in microprocessor technology when it was introduced by Intel in 1993. It marked a significant improvement over previous Intel processors like the 80486, particularly in performance and functionality. Below are **four prominent features** of the Pentium processor, which are particularly relevant in the context of **computer science D9** (likely referring to topics like computer architecture and processor design):

1. Superscalar Architecture

- **Explanation**: The Pentium processor introduced a **superscalar architecture**, which means it can execute more than one instruction per clock cycle. The processor has multiple execution units (ALU, floating-point unit, etc.), allowing it to execute several instructions in parallel.
- **Significance**: This feature increased the overall performance of the processor because it enabled better utilization of the processor's resources, reducing idle times and increasing throughput.

2. Pipelining

- **Explanation**: The **Pentium processor** utilizes **pipelining**, a technique where instruction execution is divided into different stages (fetch, decode, execute, and write-back). Each stage can process a different instruction at the same time, creating an "assembly line" effect.
- **Significance**: Pipelining helps improve the speed and efficiency of the processor by allowing multiple instructions to be processed simultaneously at different stages. It reduces the time needed to execute a sequence of instructions and optimizes performance.

3. MMX Technology (Multimedia Extensions)

- **Explanation**: **MMX** (Multimedia Extensions) is a set of instructions introduced in the Pentium processor to accelerate multimedia tasks such as audio, video, and graphics processing. MMX instructions work on **SIMD** (Single Instruction, Multiple Data) principles, enabling the processor to handle multiple data points in a single instruction cycle.
- **Significance**: MMX significantly boosted the performance of multimedia applications, making the Pentium processor more suitable for tasks like video encoding/decoding, 3D rendering, and audio compression.

4. Dual Processing Support (Symmetric Multiprocessing)

- **Explanation**: Some versions of the Pentium processor supported **dual-processing** configurations, where two Pentium processors could be used in parallel within a system, sharing the processing load.
- **Significance**: This feature was particularly useful in servers and high-performance workstations, where multiple processors could work together to improve multitasking and handle more complex computational tasks efficiently.

1. CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

This method is primarily used in wired **Ethernet** networks. It functions like a group of people having a conversation where everyone listens before speaking.

- **Carrier Sense:** A computer "listens" to the network cable to see if any other device is currently transmitting data.
- **Multiple Access:** Many computers are connected to the same cable and have equal right to use it.
- **Collision Detection:** If two computers sense that the line is free and start transmitting at the exact same moment, their signals will overlap and cause a **collision**.
  - When a collision is detected, both computers stop transmitting immediately.
  - They send out a "jam signal" to warn other devices

They send out a "jam signal" to warn other devices.

   - Each computer then waits for a **random amount of time** (backoff period) before trying to listen and transmit again. This randomness prevents them from colliding a second time.

2. Token Passing

This is a "controlled access" method commonly used in Token Ring and FDDI networks. It is much more orderly than CSMA/CD and prevents collisions entirely.

- **The Token:** A special small data packet called a "token" circulates continuously around the network (usually in a logical ring).
- **The Rule:** A computer is only allowed to transmit data if it is currently "holding" the token.
- **The Process:**
  1. The token moves from one computer to the next.
  2. If a computer has data to send, it "seizes" the token, attaches its data to it, and sends it out.
  3. The data travels around the ring until it reaches the destination.
  4. Once the destination receives the data, it sends an acknowledgment.
  5. The original sender then releases the token back into the ring for the next computer to use.

Comparison Table

Feature	CSMA/CD	Token Passing
Typical Network	Ethernet	Token Ring / FDDI
Collisions	Possible (and detected)	Eliminated (impossible)
Performance	Fast under light loads; slows down with many devices	Consistent; better under heavy loads
Access Type	Contention-based (competition)	Deterministic (turn-based)