**QUESTION : 18-19**

**1.** What is a von Neumann machine? Write down the major difference between Intel 8085 and 8086 microprocessor.

**Q: What is a Von Neumann Machine?**

A Von Neumann machine is a computer architecture where the program instructions and data are stored in the same memory.

* It uses a single memory for both code and data.
* Instructions are fetched and executed sequentially using the fetch–decode–execute cycle.
* Example: Most modern general-purpose computers.

**Q2: Mechanical Calculator made by Blaise Pascal**

* In **1642**, Blaise Pascal (a French mathematician) invented the first **mechanical calculator**, called the **Pascaline**.
* It could perform **addition and subtraction** directly, and multiplication/division by repeated addition/subtraction.
* The Pascaline used a system of gears and wheels to represent digits.

**Major Difference between Intel 8085 and 8086**

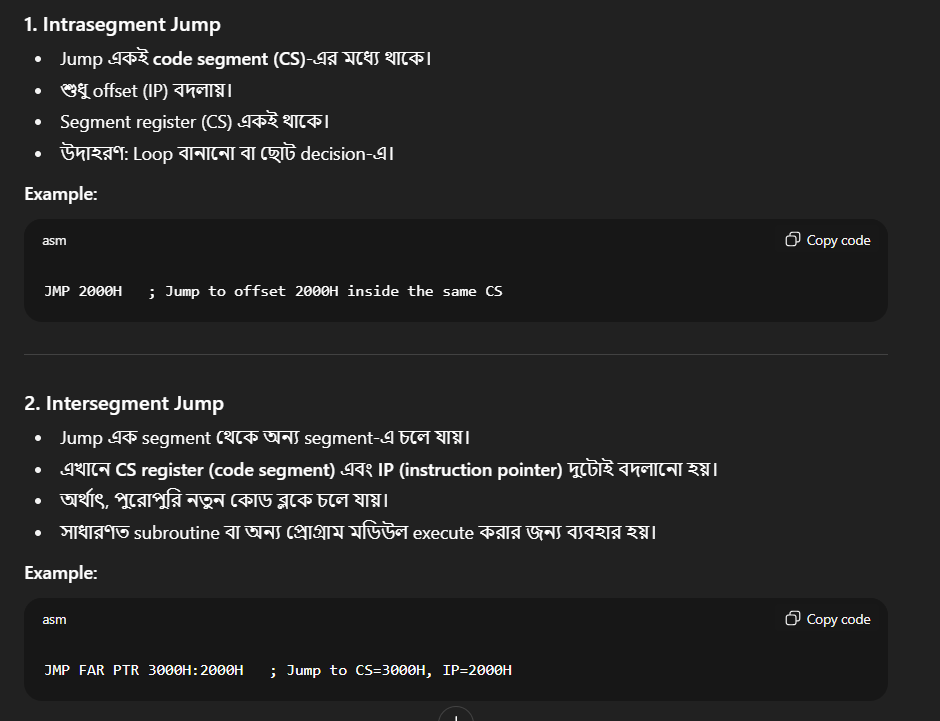
* **8085:** 8-bit microprocessor, 16-bit address bus, can access **64 KB memory**.
* **8086:** 16-bit microprocessor, 20-bit address bus, can access **1 MB memory**.
* **80286:** 16-bit microprocessor, 24-bit address bus, can access **16 MB memory**.

**2. What is the difference between intersegment and intrasegment jump? Show which JMP instruction assembles (short, near, or far) if the JMP THERE instruction is stored at memory address 10000H and the address of THERE is: i. 10020H ii. F1000H**

A black screen with white text

AI-generated content may be incorrect.





## (b) Given data and assumption

You said the JMP THERE instruction is stored at physical address **10000h**.  
A convenient and usual assumption (and the one used in class/exams) is that this means the **code segment base = 1000h**, so the instruction’s IP = 0000h (because 1000h × 10h = 10000h). We use that to decide same/different segment and compute displacements.

So:

* CS = 1000h, IP = 0000h → instruction location = physical 10000h.

Now check each target.

### i. Target = ****10020h****

* Physical target 10020h can be represented as CS:offset = 1000h:0020h because 1000h × 10h + 0020h = 10000h + 0020h = 10020h.
* **Same segment** (CS still 1000h), offset = 0020h.
* Relative displacement from IP=0000h is +0020h = 32 decimal.
* 32 lies within signed 8-bit range (−128..+127), so this can be encoded as a **short** jump (8-bit relative).  
  **Answer (i): short jump.**

### ii. Target = ****F1000h****

* Physical target F1000h can be represented naturally as F100h:0000h (because F100h × 10h = F1000h). That segment F100h ≠ current CS = 1000h.
* Since the target lies in a **different segment**, the jump must change CS → this requires a **far** jump (segment:offset).  
  **Answer (ii): far jump.**

(Even if you tried to force a near/relative encoding, you cannot because the segment would have to remain the same; here the target’s segment is different, so only a far jump is correct.)

## Final summary

* i) 10020h → **short** (intrasegment, 8-bit relative).
* ii) F1000h → **far** (intersegment, requires new CS:IP).

If you want, I can also show the exact opcode sizes/bytes for short/near/far JMP encodings on 8086.

**3)** Convert an 8B9E004CH from machine language to assembly language. If a MOV SI,[BX+12] instruction appears in a program, what is its machine language equivalent?

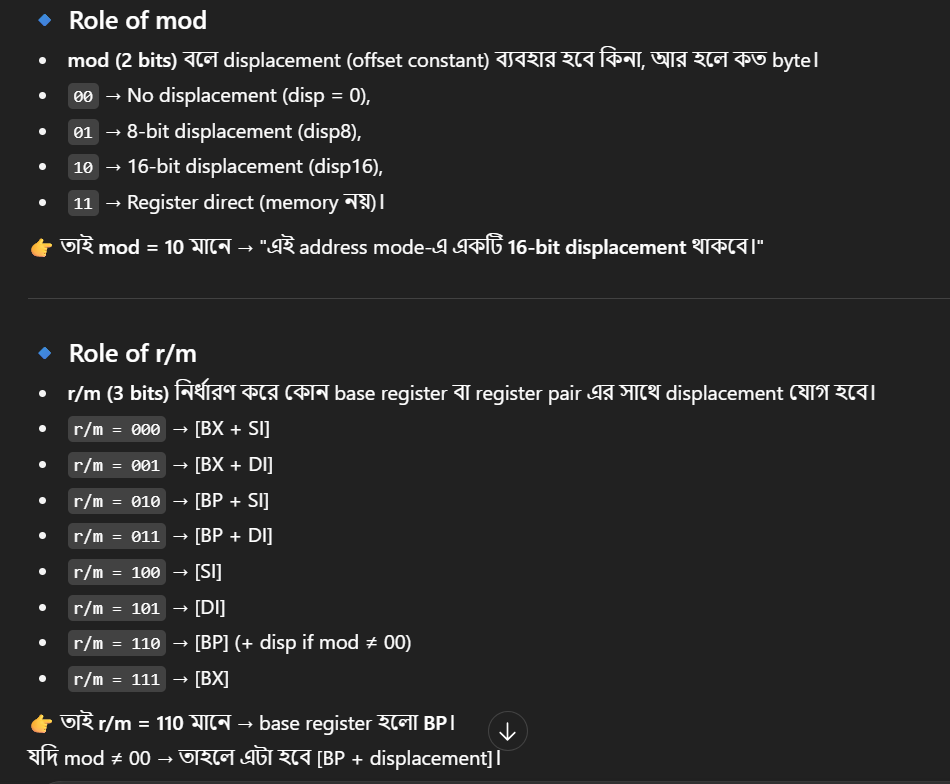
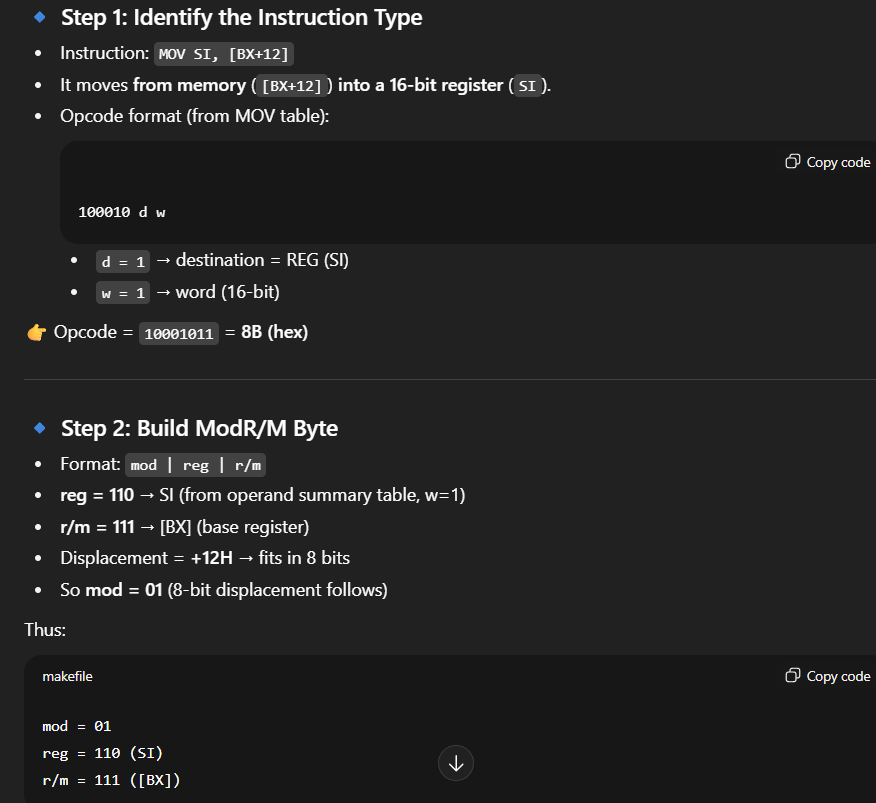
A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect. A black background with white text

AI-generated content may be incorrect.

  A screenshot of a computer

AI-generated content may be incorrect.

4. What will be the CS:IP of physical address BCDEFH where CS=FFF0? How is the local descriptor table addressed in the memory system?

## **Q4 (Part 1): What will be the CS:IP of physical address BCDEFH where CS=FFF0?**

A screenshot of a computer

AI-generated content may be incorrect.

## **Q4 (Part 2): How is the Local Descriptor Table (LDT) addressed in the memory system?**

## In **protected mode** (80286 and above), segmentation uses **descriptors** instead of segment:offset.

* The **Local Descriptor Table (LDT)** is a table in memory that holds descriptors for a process.
* The processor finds the LDT using the **LDTR (Local Descriptor Table Register)**, which stores the **base address and limit** of the LDT.
* When a program uses a **selector** that points to the LDT, the CPU uses LDTR to locate the correct descriptor in the LDT, and then maps it to the physical memory address.

✅ In short:

* **LDT is addressed via LDTR**.
* Selector → LDTR (base of LDT) + index → Descriptor → Physical address.

### ****Final Answer (Exam style):****

1. For physical address **BCDEFh** with CS=FFF0h:

CS:IP=FFF0h:0DEFhCS:IP = FFF0h : 0DEFh

1. The **Local Descriptor Table (LDT)** is addressed in memory through the **LDTR register**, which holds the base address of the LDT. The CPU uses the selector to index into the LDT and fetch the descriptor for memory access.

5. Which register locates the global descriptor table? Describe the content of the segment register at protected mode memory addressing.

## **Answer**

### (1) Which register locates the Global Descriptor Table (GDT)?

* The **GDTR (Global Descriptor Table Register)** locates the **Global Descriptor Table**.
* It stores the **base address** and **limit** of the GDT.
* The CPU uses GDTR whenever a selector refers to a descriptor in the GDT.

### (2) Content of the Segment Register in Protected Mode

In **protected mode**, a segment register (like CS, DS, ES, SS, FS, GS) does **not directly hold a physical address**.  
Instead, it contains:

1. **Selector (16-bit):**
   * **Index (13 bits):** Points to an entry in GDT or LDT.
   * **TI (Table Indicator, 1 bit):** Selects **GDT (0)** or **LDT (1)**.
   * **RPL (Request Privilege Level, 2 bits):** Privilege check.
2. **Hidden Part (Descriptor Cache):** Loaded automatically by CPU when selector is used:
   * Base address (32-bit)
   * Segment limit (20-bit or 32-bit)
   * Access rights (type, privilege level, etc.)

Thus, the segment register in protected mode = **Selector + Hidden Descriptor Cache**.

✅ **Final Short Exam Answer:**

* The **GDTR register** locates the Global Descriptor Table.
* In protected mode, each segment register contains a **16-bit selector** (Index, TI, RPL), and a **hidden part** storing the base, limit, and access rights of the segment (loaded from descriptor).

তুমি চাইলে আমি এর একটা ছোট্ট **diagram এঁকে** বুঝিয়ে দিতে পারি (selector fields + hidden descriptor)। চাও কি সেটা?

6. Explain the instruction with respect to 8086 microprocessor MOV AX,[BX].  
ঠিক আছে 👍 প্রশ্নটা 8086 এর instruction explanation চায়। আমি exam–style concise উত্তর দিলাম:

## **Answer**

### Instruction:

MOV AX, [BX]

### Explanation:

1. **Operation:**

* This instruction copies the **word (16-bit) data** from the memory location pointed to by **BX register** into **AX register**.
* Format: Destination ← Source
* So: AX ← [BX]

1. **Addressing Mode:**

* This uses **Register Indirect Addressing Mode** because BX holds the effective address of the memory operand.

1. **Data Size:**

* Since the destination is AX (16-bit register), the instruction transfers **16 bits (a word)** from memory.

1. **Effect on Flags:**

* **No flag is affected** by this instruction.

✅ **Final Short Answer (Exam-ready):**  
The instruction MOV AX, [BX] transfers a 16-bit word from the memory location addressed by BX into AX register. It uses **register indirect addressing mode**, copies data (AX ← [BX]), and does not affect any flags.

চ