**QUESTION 1(Illustrate with a diagram Flynn’s Single Instruction Stream Single Data Stream (SISD) )**

1-4

Data stream

(1,4)

Single Instruction (-)

(-)

(1,4)

**Program Counter (PC)**: The system has a single **program counter** that keeps track of the current instruction being executed. It points to the location in memory from which the next instruction will be fetched.

**Instruction Stream**: Only one instruction stream is being processed, meaning the system executes one instruction at a time, sequentially.

**Data Stream**: Only one piece of data is processed at a time, meaning each instruction operates on a single data element, often from memory.

**QUESTION 2 (Explain the following distributed Models)**

**Minicomputer Model:**

A distributed system where a central **minicomputer** connects to multiple **terminals** or **workstations**. The minicomputer manages resources and handles most of the processing, while the terminals interact with the system for input and output. Example :ARPAnet characteristics are

**Extension of Time-sharing**: Users log into their home minicomputer first, then can access remote machines using tools like **telnet**.

**Resource Sharing**: Users can access and share resources like databases and high-performance devices across different systems.

**Centralized Control**: The home minicomputer manages user access and resources, while remote machines provide additional services and computing power

**Workstation-Server Model:**

This is model **workstations** connect to a **central server**. The server provides resources like storage or processing power, and workstations handle local processing while relying on the server for shared tasks. characteristics are

**Client Workstations**: These are computers that run interactive applications locally but rely on servers for tasks like file storage, printing, and computation.

**Server Minicomputers**: These are powerful systems dedicated to providing services like file storage, printing, and computation requests from clients.

**Communication**: Clients send requests (using protocols like RPC or RMI) to servers, which handle the requested tasks. There's no process migration; each machine handles its own tasks.