Parshvanath Charitable Trust's

A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING DATA SCIENCE

UNIT TEST-II-Solution

Class: SE **Semester: III** **Subject: DLCA**

Max marks: 40

Q.1	Attempt any two.	**Marks**
(a)	Write a short note on Address Sequencing technique in micro-control unit. **Address Sequencing** - Microinstructions are stored in control memory in groups, with each group specifying a ***routine***. - To appreciate the address sequencing in a micro-program control unit, let us specify the steps that the control must undergo during the execution of a single computer instruction. **Step-1:** - An initial address is loaded into the control address register when power is turned on in the computer. - This address is usually the address of the first microinstruction that activates the instruction fetch routine. - The fetch routine may be sequenced by incrementing the control address register through the rest of its microinstructions. - At the end of the fetch routine, the instruction is in the instruction register of the computer. **Step-2:** - The control memory next must go through the routine that determines the effective address of the operand. - A machine instruction may have bits that specify various addressing modes, such as indirect address and index registers. - The effective address computation routine in control memory can be reached through a branch microinstruction, which is conditioned on the status of the mode bits of the instruction. - When the effective address computation routine is completed, the address of the operand is available in the memory address register. **Step-3:** - The next step is to generate the microoperations that execute the instruction fetched from memory. - The microoperation steps to be generated in processor registers depend on the operation code part of the instruction. - Each instruction has its own micro-program routine stored in a given location of control memory. - The transformation from the instruction code bits to an address in control memory where the routine is located is referred to as a ***mapping*** process. - A mapping procedure is a rule that transforms the instruction code into a control memory address.	[5]

	Step-4: - Once the required routine is reached, the microinstructions that execute the instruction may be sequenced by incrementing the control address register. - Micro-programs that employ subroutines will require an external register for storing the return address. - Return addresses cannot be stored in ROM because the unit has no writing capability. - When the execution of the instruction is completed, control must return to the fetch routine. - This is accomplished by executing an unconditional branch microinstruction to the first address of the fetch routine. **In summary, the address sequencing capabilities required in a control memory are:** 1. Incrementing of the control address register. 2. Unconditional branch or conditional branch, depending on status bit conditions. 3. A mapping process from the bits of the instruction to an address for control memory. 4. A facility for subroutine call and return.	
(b)	Write a micro-program for FETCH routine of instruction cycle. FETCH: ORG 64 U JMP NEXT PCTAR U JMP NEXT READ, INCPC U MAP DRTAR U INDRCT: READ U JMP NEXT DRTAR U RET	[5]
(c)	Explain Delay element method for designing of Hardwired Control Unit. DELAY ELEMENT METHOD → In this the behaviour of control unit is represented in the form of flowchart. → Every step of flowchart at time t_i will activate $\{C_{ij}\}$. where C_i is the ctrl s/g at time t_i for execution of instruction j	[5]

i.e at t_1 : Activate $\{C_{1,j}\}$

t_2 : Activate $\{C_{2,j}\}$

⋮

at t_n : Activate $\{C_{n,j}\}$

→ Once the flow chart is complete, the individual ckts for each $\{C_{i,j}\}$ are formed

→ It is obvious that the instruction j will be executed when all the steps of $\{C_{i,j}\}$ are performed from $C_{1,j}$, $C_{2,j}$, $C_{3,j}$ .... $C_{n,j}$.

→ But all these steps should not be performed together, instead there should be finite time gap (delay) between every 2 steps.

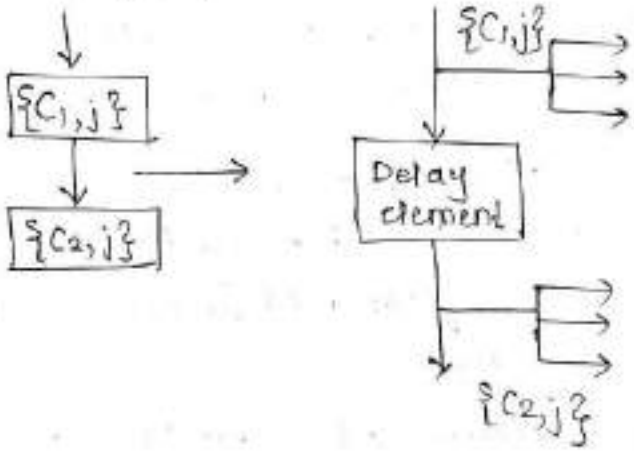
→ The delay in the ckt is introduced by a "D" Flipflop.

Delay time $= t_2 - t_1 = t_3 - t_2 =$ one clk pulse.

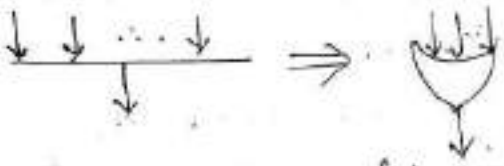
→ Thus one after another all steps are performed in the order of flowchart.

→ The different parts of a flowchart are handled as follows:

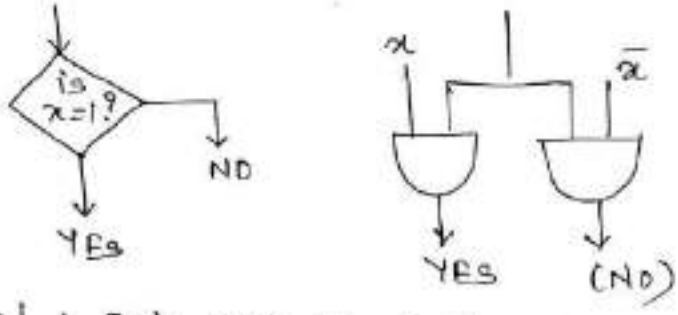
i) Between 2 successive steps simply a D flip flop is inserted



ii) Multiple i/p's are combined by an OR gate. An OR gate will produce a 1 when any of the i/p's is 1.



iii) A conditional branch (decision box) of flowchart is implemented by a pair of AND gates.

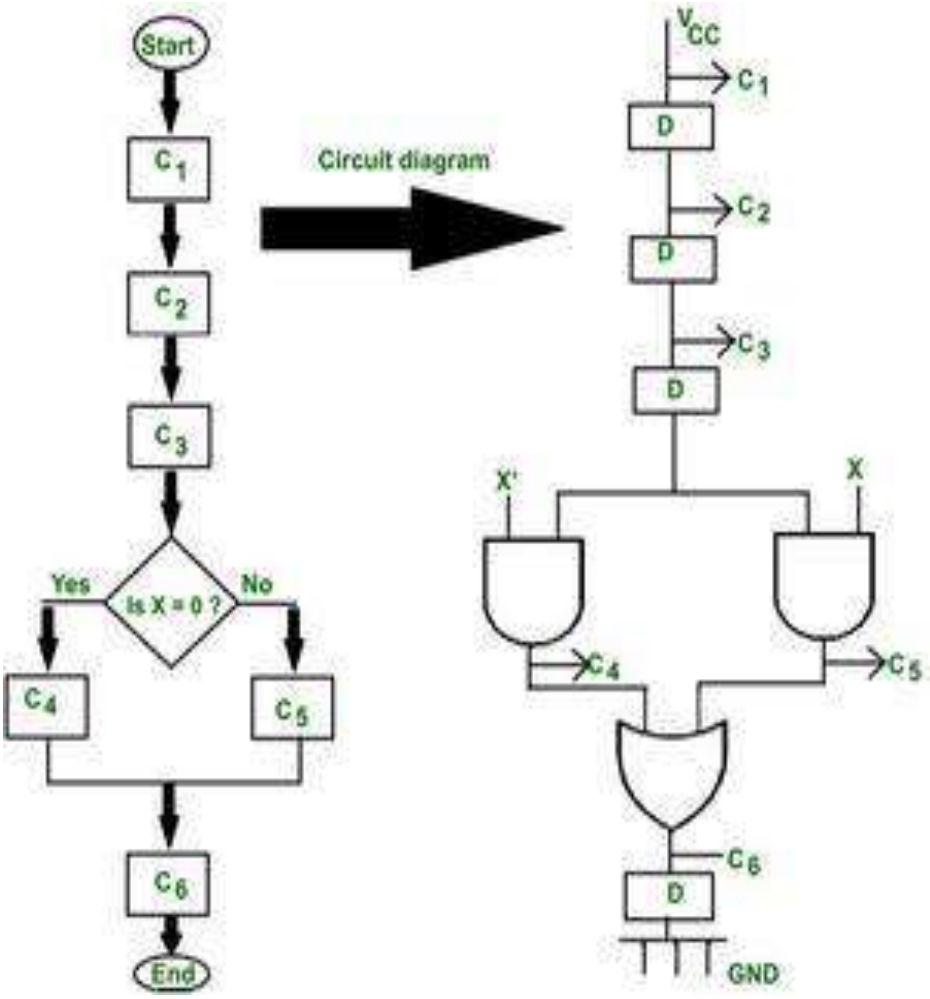


If $x = 1$: gate on the left will be active

$x = 0$: gate on right will be active.

Example

Suppose the processor has two instructions add or subtract (Therefore an opcode of 1 bit is needed in which 0 opcode for add instruction and 1 for subtract is used.



(d)	Compare Hardwired and Microprogrammed control unit.	[5]

Basis of Differentiation	Hardwired Control Unit	Microprogrammed Control Unit
Basic	It is a circuitry approach.	This control unit is implemented by programming
Design	RISC style instructions	CISC style instructions
Modification	Modification is difficult as the control unit is hardwired. Modifying it will require the	Modifications are easy in case of microprogrammed control unit as it will
Basis of Differentiation	**Hardwired Control Unit**	**Microprogrammed Control Unit**
	change in hardware.	require the in change in the code only.
Instructions	It works well for simple instructions.	It works well for complex instructions also.
Costing	Implementing hardwired structure requires a cost.	Implementing microprograms is not costly.
Control memory	No control memory is required	Control memory is required
ExecutionSpeed	Faster execution	Execution Speed

Q.2(a)	Consider a direct mapped cache of size 512 KB with block size 1 KB. There are 7 bits in the tag. Find- 1. Size of main memory 2. Tag directory size	[10]

We consider that the memory is byte addressable.

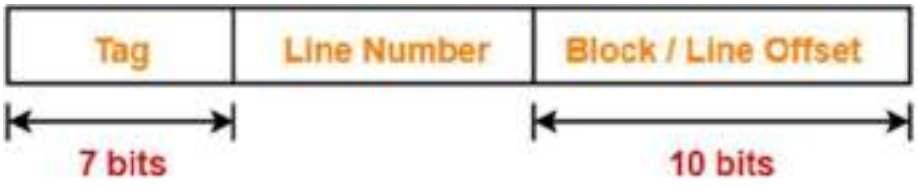
Number of Bits in Block Offset-

We have, Block size

= 1 KB

= 2^{10} bytes

Thus, Number of bits in block offset = 10 bits



Number of Bits in Line Number-

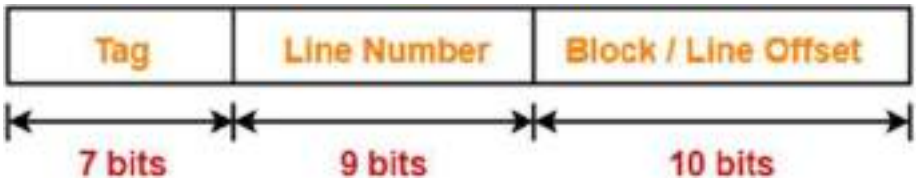
Total number of lines in cache

= Cache size / Line size

= 512 KB / 1 KB

= 2^9 lines

Thus, Number of bits in line number = 9 bits



Number of Bits in Physical Address-

Number of bits in physical address

= Number of bits in tag + Number of bits in line number + Number of bits in block offset

= 7 bits + 9 bits + 10 bits

= 26 bits

Thus, Number of bits in physical address = 26 bits

Size of Main Memory-

We have,

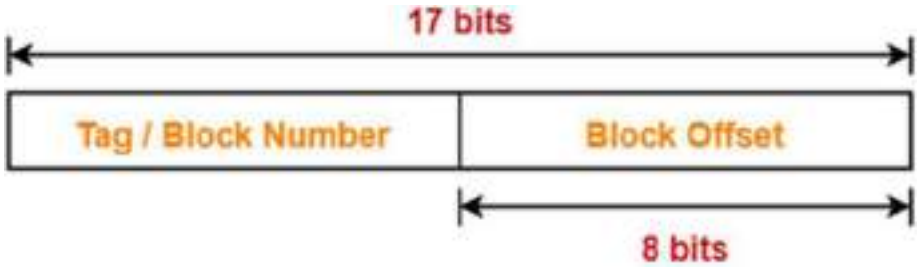
Number of bits in physical address = 26 bits

Thus, Size of main memory

= 2^{26} bytes

= 64 MB

	Tag Directory Size- Tag directory size = Number of tags x Tag size = Number of lines in cache x Number of bits in tag = 2^9 x 7 bits = 3584 bits = 448 bytes Thus, size of tag directory = 448 bytes	
	OR	
Q.2(b)	Consider a fully associative mapped cache of size 16 KB with block size 256 bytes. The size of main memory is 128 KB. Determine Number of bits in tag and tag directory size. We consider that the memory is byte addressable. Number of Bits in Physical Address- We have, Size of main memory = 128 KB = 2^{17} bytes Thus, Number of bits in physical address = 17 bits  Number of Bits in Block Offset- We have, Block size = 256 bytes = 2^8 bytes Thus, Number of bits in block offset = 8 bits	[10]



Number of Bits in Tag-

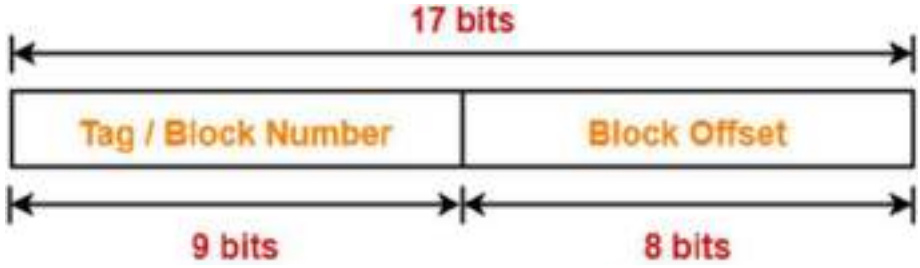
Number of bits in tag

= Number of bits in physical address – Number of bits in block offset

= 17 bits – 8 bits

= 9 bits

Thus, Number of bits in tag = 9 bits



Number of Lines in Cache-

Total number of lines in cache

= Cache size / Line size

= 16 KB / 256 bytes

= 2^{14} bytes / 2^{8} bytes

= 2^{6} lines

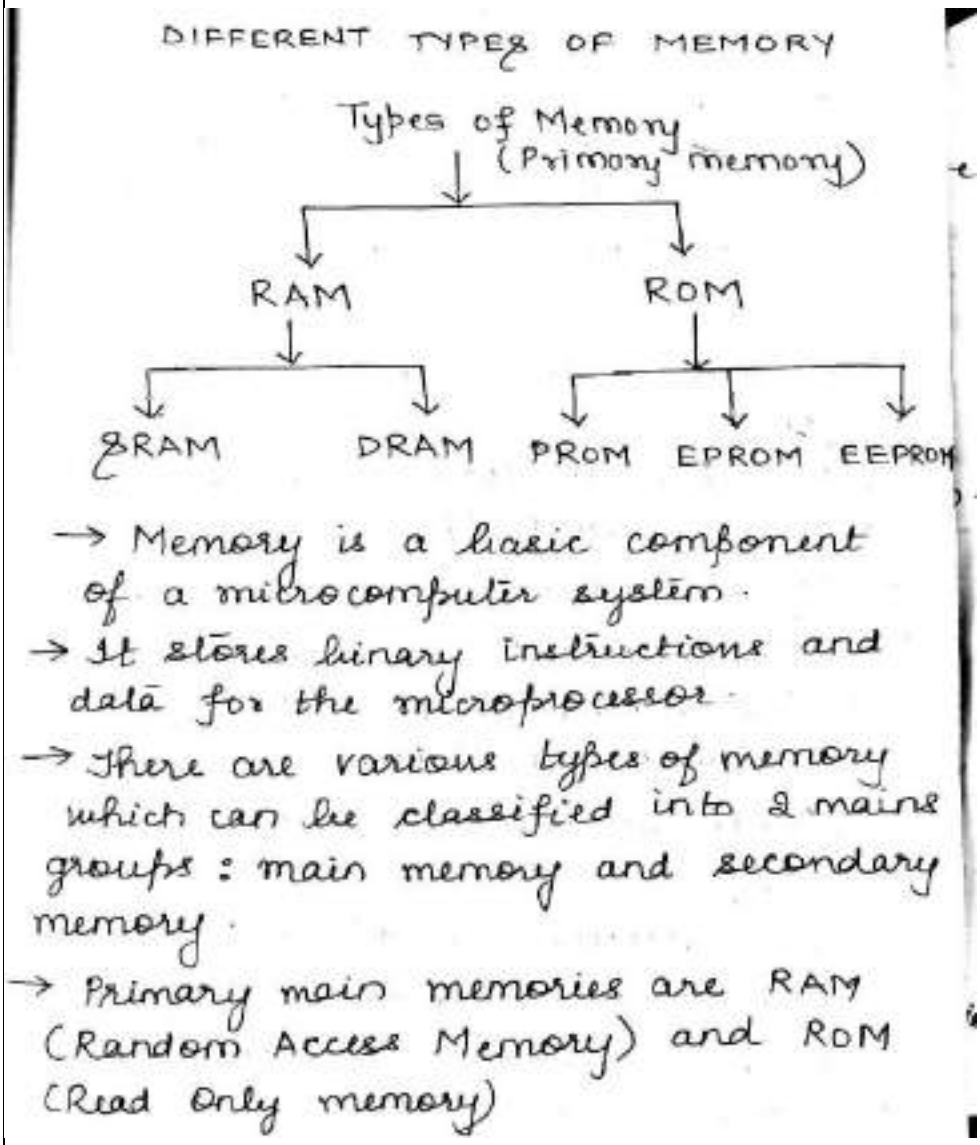
Tag Directory Size-

Tag directory size

= Number of tags x Tag size

= Number of lines in cache x Number of bits in tag

= 2^{6} x 9 bits

	= 576 bits = 72 bytes Thus, size of tag directory = 72 bytes	
Q.2(c)	Explain different types of primary memory.  	[5]

→ Secondary storage memories are magnetic tapes and magnetic disks.

→ The memory is made up of registers and each register has a group of flip flops that store bits of inform^n; these flip flops are called memory cells.

RAM stands for random access memory.

→ RAM loses its contents when the power is turned off.

→ RAM is called as volatile memory

→ RAM is used for storing the bulk of the programs and data that are subject to change.

→ A chip select, CS, enables or disables the RAM.

→ ADRS specifies the address / location to read from or write to.

→ WR selects between reading from and writing to memory.
Read — WR ⇒ set to 0
OUT will be n bit value stored at ADRS

Write to memory ⇒ WR = 1.
DATA is n bit value to save in memory

→ There are k address lines which can specify one of 2^k addresses.
→ Each address contains an n bit word.

RAM are of 2 types
STATIC RAM (SRAM)

→ It is easier to use and has shorter read and write cycles.
→ It consists of internal flip flops that store binary informⁿ.
→ The stored informⁿ remains valid as long as power is applied to the unit.
→ SRAM ~~devices~~ are low density, high power, expensive and fast in operⁿ.
→ All computer memory modules used in today's computers are of SRAM type.

DYNAMIC RAM (DRAM)

→ DRAM stores the binary inform^n in the form of electric charges that are applied to capacitors.

→ It is a refreshing type memory. As long as power is maintained on the memory modules of DRAM will hold its inform^n.

→ The content of DRAM memory disappears from memory within millisec, so to maintain the data it should be refreshed periodically.

→ This makes DRAM memory much slower than SRAM.

→ The stored charge on capacitors tend to discharge with time and the cap needs to recharged periodically by refreshing the dynamic memory.

→ DRAM offers reduced power consumption and larger storage capacity in a single memory chip.

	OR	
Q.2(d)	Explain various characteristics of memory.	[5]

CHARACTERISTICS OF MEMORY

→ The key characteristics of memory devices are as follows:

i) Location: It deals with the location of memory device in the computer system. There are 3 possible loc^n:

a) CPU: This is often in the form of CPU registers and small amount of cache.

b) Internal / Main: This is the main memory like RAM or ROM.

→ The CPU can directly access the main memory.

c) External / Secondary: It comprises of secondary storage devices like hard disks, magnetic tapes.

→ CPU doesnt have access to these devices directly.

→ It uses device controllers to access secondary devices.

ii) Capacity : It is expressed in terms of a) word size & b) no. of words.

a) Word size – Words are expressed in bytes (8 bits).

→ A word can however mean any number of bytes.

→ Commonly used word sizes are 1 byte (8 bits), 2 bytes (16 bits) and 4 bytes (32 bits)

b) Number of words – This specifies the number of words available in the particular memory device.

iii) Unit of Transfer : It is the max no. of bits that can be read or written into the memory at a time.

→ In case of main memory, it is mostly equal to word size.

→ In case of external memory, unit of transfer is not limited to word size.

iv) Access Methods : It is the fundamental characteristic of memory devices.

→ It is the sequence in which memory can be accessed.

There are 3 types :

a) Random Access : If storage locations in a particular memory device can be accessed in any order and access time is independent of memory locn being accessed, then such memory devices are said to have random access mechanism.

→ RAM IC's use this mechanism.

b) Serial Access : If memory locations can be accessed only in a certain predetermined sequence, this access method is called serial access.

Magnetic Tapes, CD-ROMs employ serial access methods.

c) Semi random access : In this each track has a read / write head thus each track can be accessed randomly but access within each track is restricted to serial access.

v) Performance: 3 parameters determine the performance of memory.

a) Access time: Time taken by memory to complete read/write oper^n from the instant that an address is sent to the memory for random access memories
→ For non random access memories, it is the time taken to position read and write head at the desired loc^n.

b) Memory cycle time: It is defined only for random access memories & it is the sum of access time and additional time reqd before second access can commence.

c) Transfer rate: Defined as rate at which data can be transferred into or out of a memory unit.

vi) Physical type: Memory devices can be either semiconductor memory (like RAM) or magnetic surface memory (like hard disks)

vii) Physical characteristics :

a) Volatile / Non-Volatile : If a memory devices continues hold data even if power is turned off, then memory device is non volatile else it is volatile.

viii) Organization :

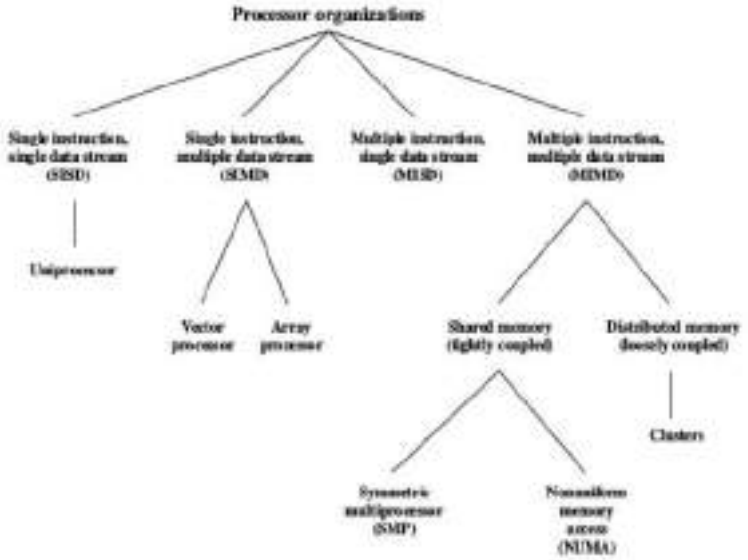
a) Erasable / Non Erasable : The memory in which data once programmed cannot be erased are called as non erasable memories.
Memories in which data can be erased - erasable memory.

Q.3 (a)	Illustrate Flynn's classification.	[10]

Flynn's Classification

Introduction

The most popular taxonomy of computer architecture was defined by Flynn in 1966. Flynn's classification scheme is based on the notion of a stream of information. Two types of information flow into a processor: instructions and data. The instruction stream is defined as the sequence of instructions performed by the processing unit. The data stream is defined as the data traffic exchanged between the memory and the processing unit.



According to Flynn's classification, either of the instruction or data streams can be single or multiple.

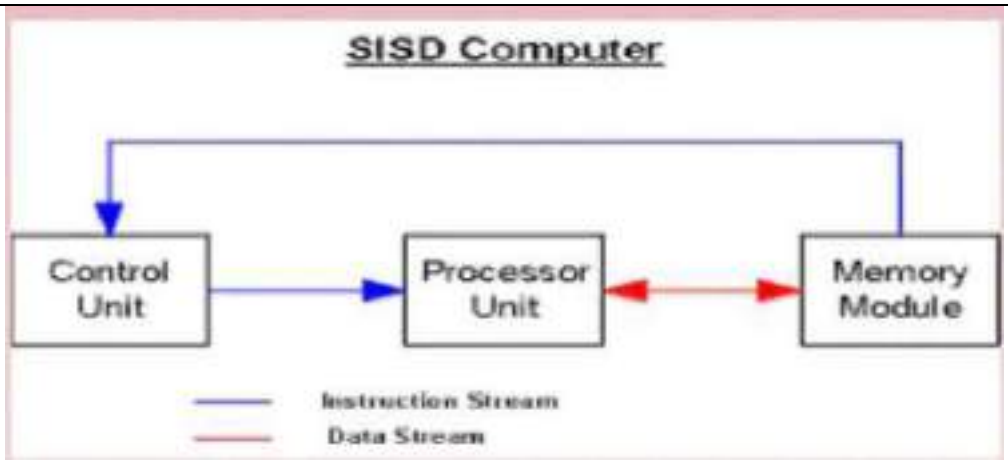
- Single-instruction single-data streams (SISD);
- Single-instruction multiple-data streams (SIMD);
- Multiple-instruction single-data streams (MISD); and
- Multiple-instruction multiple-data streams (MIMD).

Conventional single-processor von Neumann computers are classified as SISD systems. Parallel computers are either SIMD or MIMD. When there is only one control unit and all processors execute the same instruction in a synchronized fashion, the parallel machine is classified as SIMD. In a MIMD machine, each processor has its own control unit and can execute different instructions on different data.

SISD Architecture

In computing, SISD is a computer architecture in which a single uni-core processor, executes a single instruction stream, to operate on data stored in a single memory.

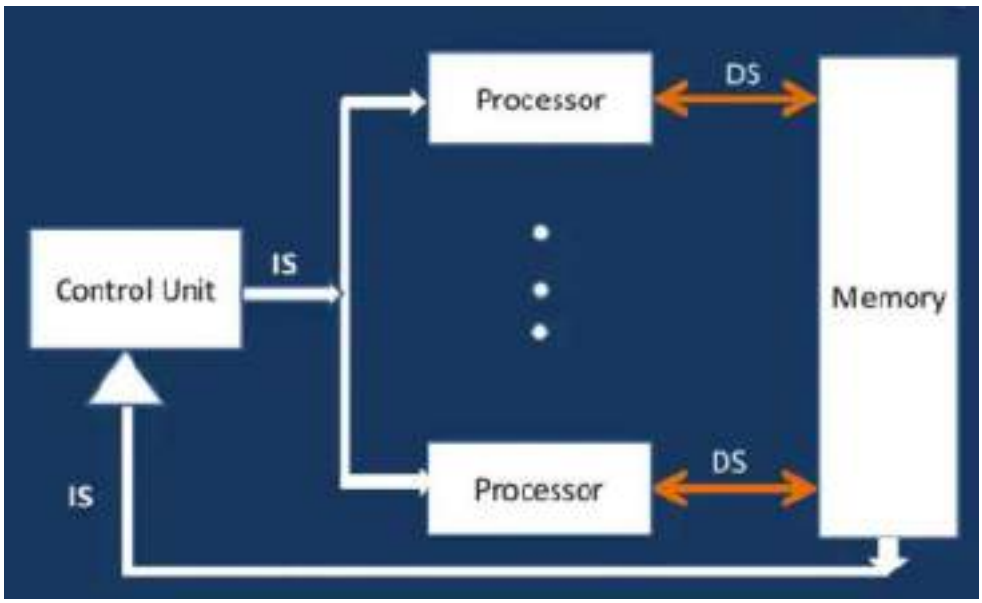
1. Single instruction: only one instruction stream is being acted on by the CPU during any one clock cycle
2. Single data: only one data stream is being used as input during any one clock cycle Deterministic execution
3. This is the oldest and until recently, the most prevalent form of computer Examples: most PCs, single CPU workstations and mainframes



SIMD Architecture

The SIMD architecture performs a single, identical action simultaneously on multiple data pieces. Here we have a single control unit (CU) and more than one processing unit (PU). For e.g. a single instruction to fetch multiple files.

- Single instruction: All processing units execute the same instruction at any given clock cycle
- Multiple data: Each processing unit can operate on a different data element
- This type of machine typically has an instruction dispatcher, a very high-bandwidth internal network, and a very large array of very small-capacity instruction units.
- Best suited for specialized problems characterized by a high degree of regularity, such as image processing.



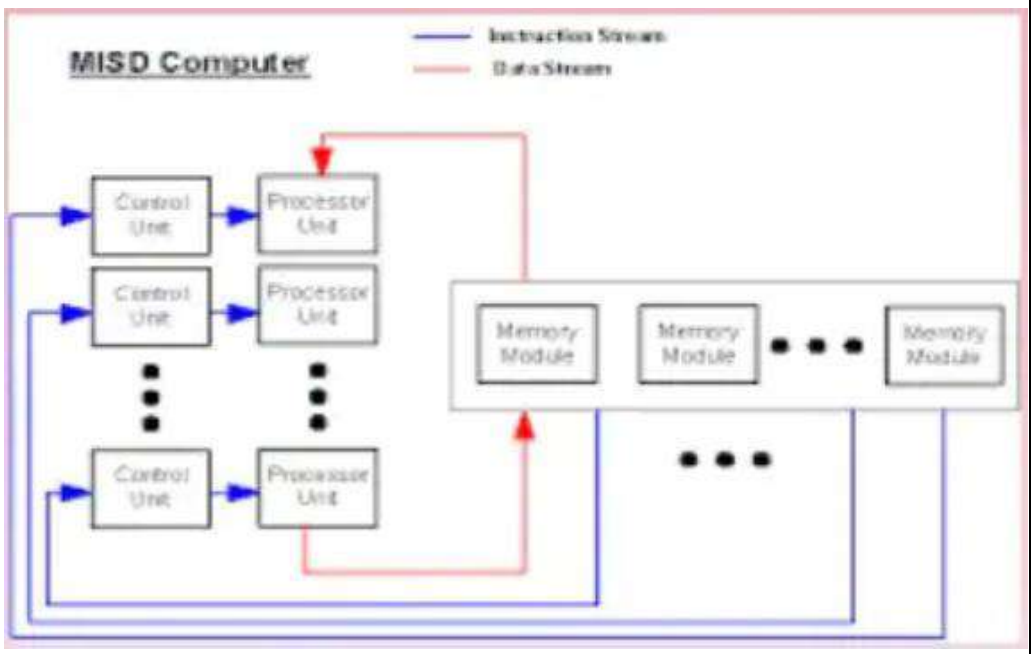
- Two varieties: Processor Arrays and Vector Pipelines Examples:
  - Processor Arrays: Connection Machine CM-2, Maspar MP-1, MP-2
  - Vector Pipelines: IBM 9000, Cray C90, Fujitsu VP, NEC SX-2, Hitachi S820

MISD Architecture

Multiple Instruction, Single Data (MISD) computers have multiple processors.Each processor uses a different algorithm but uses same shared input data. MISD computers can analyze same set of data using several different operations at same time. The number of operations depends upon number of processors.

There aren't many actual examples of MISD computers,for e.g. fault-tolerant computers executing the same nstructions redundantly in order to detect and mask errors.

- A single data stream is fed into multiple processing units.
- Each processing unit operates on the data independently via independent instruction streams.
- Few actual examples of this class of parallel computer have ever existed. One is the experimental Carnegie-Mellon C.mmp computer (1971).
- Some conceivable uses might be:
  - Multiple frequency filters operating on a single signal stream
  - Multiple cryptography algorithms attempting to crack a single coded message.



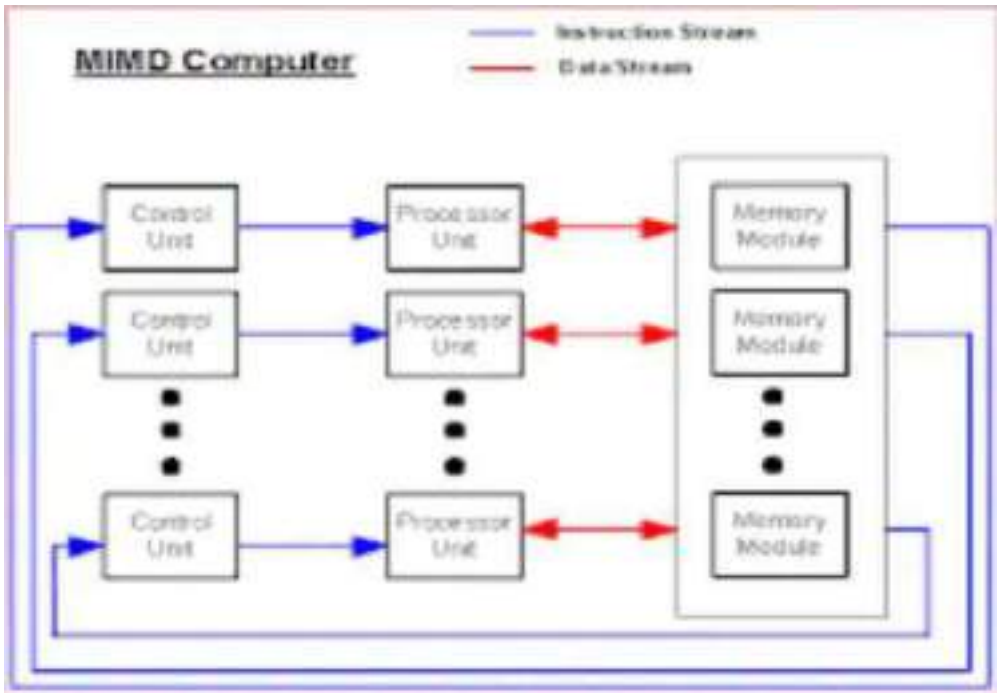
MIMD Architecture

Multiple-instruction multiple-data streams (MIMD) parallel architectures are made of multiple processors and multiple memory modules connected together via some interconnection network. It means that parallel units have separate instructions, so each of them can do something different at any given time.

While technically it's true that most modern desktop/laptops are MIMD. Using the MIMD, each processor in a multiprocessor system can execute asynchronously different set of the instructions independently on the different set of data units.

They fall into two broad categories: shared memory or message passing. Processors exchange information through their central shared memory in shared memory systems, and exchange information through their interconnection network in message passing systems.

- Currently, the most common type of parallel computer. Most modern computers fall into this category.
- Multiple Instruction: every processor may be executing a different instruction stream Multiple Data: every processor may be working with a different data stream
- Execution can be synchronous or asynchronous, deterministic or non-deterministic
- Examples: most current supercomputers, networked parallel computer "grids" and multi-processor SMP computers - including some types of PCs.
- A shared memory system typically accomplishes interprocessor coordination through a global memory shared by all processors. These are typically server systems that communicate through a bus and cache memory controller.



	OR	
Q.3 (b)	Explain the performance measures of pipelining.	[10]

Performance of Pipelined Execution-

The following parameters serve as criterion to estimate the performance of pipelined execution-

Speed Up
Efficiency
Throughput

1. Speed Up-

It gives an idea of "**how much faster**" the pipelined execution is as compared to non-pipelined execution.

It is calculated as-

$$\text{Speed Up (S)} = \frac{\text{Non-pipelined execution time}}{\text{Pipelined execution time}}$$

2. Efficiency-

The efficiency of pipelined execution is calculated as-

$$\text{Efficiency } (\eta) = \frac{\text{Speed Up}}{\text{Number of stages in Pipelined Architecture}}$$

OR

$$\text{Efficiency } (\eta) = \frac{\text{Number of boxes utilized in phase-time diagram}}{\text{Total number of boxes in phase-time diagram}}$$

3.Throughput-

Throughput is defined as number of instructions executed per unit time.

It is calculated as-

$$\text{Throughput} = \frac{\text{Number of instructions executed}}{\text{Total time taken}}$$

	Response time is the time from start to completion of a task. This also includes: • Operating system overhead. • Waiting for I/O and other processes • Accessing disk and memory • Time spent executing on the CPU or execution time. **Throughput** is the total amount of work done in a given time. **CPU execution time** is the total time a CPU spends computing on a given task. It also excludes time for I/O or running other programs. This is also referred to as simply CPU time. Performance is determined by execution time as performance is inversely proportional to execution time. **How to Improve Performance?** To improve performance you can either: • Decrease the CPI (clock cycles per instruction) by using new Hardware. • Decrease the clock time or Increase clock rate by reducing propagation delays or by use pipelining. • Decrease the number of required cycles or improve ISA or Compiler. In instruction pipelining, • A form of parallelism called as instruction level parallelism is implemented. • Multiple instructions execute simultaneously. • The efficiency of pipelined execution is more than that of non-pipelined execution.	
Q.3 (c)	Illustrate and derive the expression for Amdahls law.	[5]

AMDAHL'S LAW.

→ Amdahl's law relates the improvement of system's performance with the parts that didnt perform well.

→ This law is often used in parallel computing to predict the theoretical speedup when using multiple processors.

→ Amdahl's law is expressed mathematically as

$$\text{Speed up max} \leq \frac{1}{F + \frac{(1-F)}{P}}$$

$$S(P) \leq \frac{1}{F + \frac{(1-F)}{P}}$$

where p : no. of processors
F : sequential fraction of program
$1-F$: parallel fraction of program.

PROOF:
Consider execution of a task by a single processor as well as p no. of processors.
Let F be the sequential part of program
$\therefore 1-F$ will be the parallel part of program.

	Speed up = $\frac{\text{Time taken by single processor}}{\text{Time taken by p no of processors}}$ $= \frac{T(s)}{T(p)}$. Let $T(s) = T$ $\therefore T(p) =$ Sequential time + Parallel time $= F \cdot T + \frac{(1-F) \cdot T}{p}$. $S(p) = \frac{T}{F \cdot T + \frac{(1-F) \cdot T}{p}}$ $S(p) = \frac{1}{F + \frac{(1-F)}{p}}$ S(p) will always be less than or equal to 1 $\therefore S(p) \leq \frac{1}{F + \frac{(1-F)}{p}}$	
	OR	
Q.3 (d)	A program having 10 instructions is executed on non-pipeline and pipeline processors. All instructions are of same length and having 4 pipeline stages and time required to each stage is 1ns. 1) Calculate time required to execute the program on non-pipeline and pipeline processor. 2) Calculate speed-up Solution: Non pipeline processor: Cycle time = 4*10 = 40 nsec Pipeline Processor (4 Stages): 2. Cycle time = 10+3=13 nsec Speedup = 40/13 = 3.08	[5]