CO-3 SHORT ANSWERS

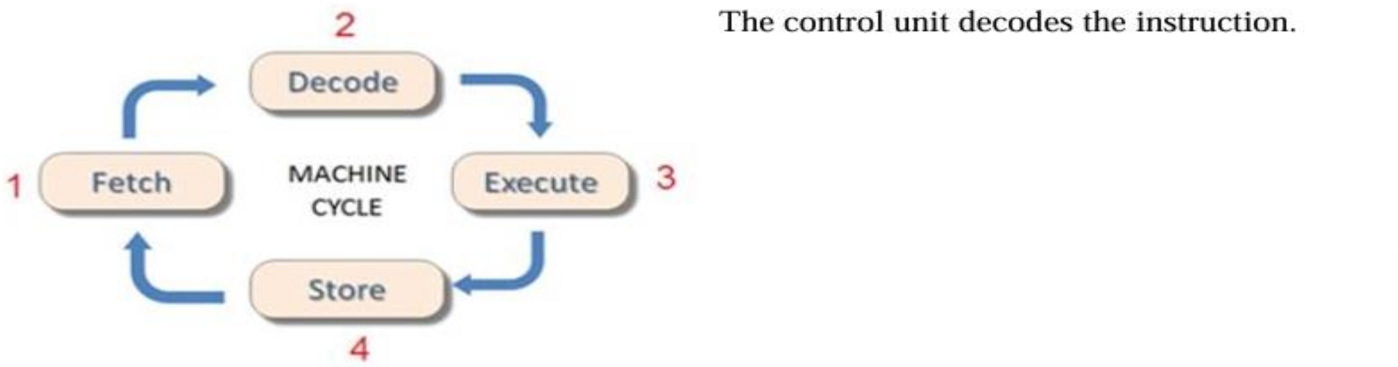
1.Explore the significance of a microprocessor in modern computing devices.

Ans: The microprocessor serves as the "brain" of modern computing devices, executing instructions and managing tasks efficiently.

2. Provide the common types of operands with an example.

Operands are values used in arithmetic or logical operations. Example: In the operation x = y + z y and z are the operands. The computer retrieves their values from memory, performs the addition, and stores the result in x.

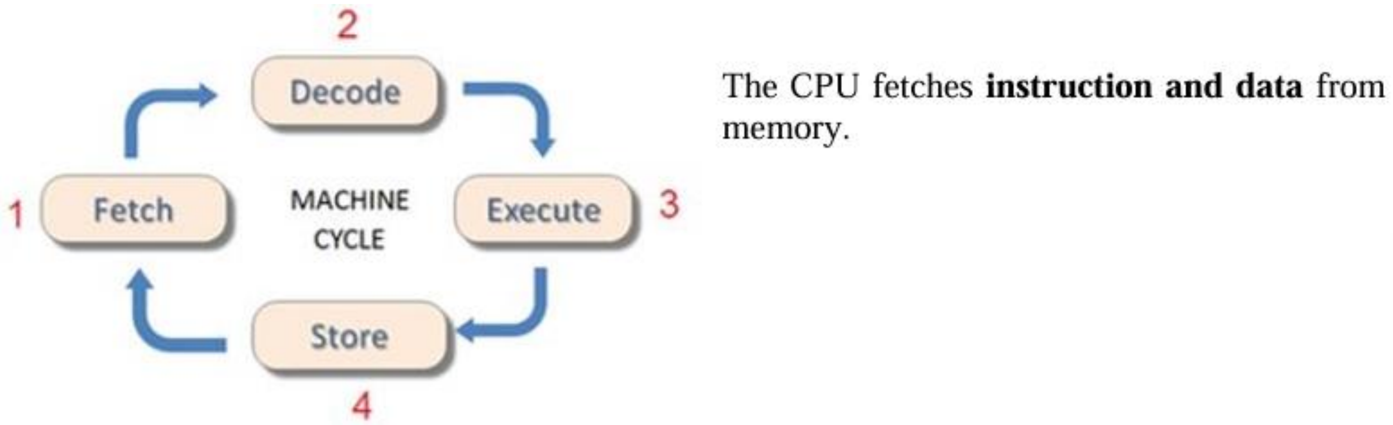
3. Explore the role of the decoding phase in a machine cycle.



The control unit decodes the instruction.

Ans: The decoding phase translates instructions into signals that control the operation of the CPU, enabling it to understand and execute commands.

4. Specify the role of the fetching phase in a machine cycle.

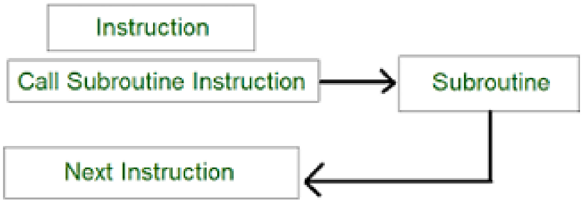


The CPU fetches **instruction and data** from memory.

Ans: The fetching phase retrieves instructions from memory and loads them into the CPU, preparing them for execution.

5. Specify the purpose of subroutine call in computer programming.

Ans:



The purpose of a subroutine call in computer programming is to execute a specific sequence of code that performs a particular task or function.

Subroutine calls allow for the modular organization of code, enabling the reuse of code segments and promoting better code maintenance and readability.

6. Provide the expansion of RISC and CISC.

Ans: RISC stands for Reduced Instruction Set Computer, focusing on simpler instructions for faster execution, while CISC stands for Complex Instruction Set Computer, emphasizing more complex instructions to handle various tasks in fewer steps.

7. List the characteristics of a multicycle implementation in processor design.

Ans: Multicycle implementation in processor design allows for the execution of instructions in multiple cycles, enabling more complex operations and increasing flexibility.

1. Reuse of Hardware Components
2. Variable Execution Time
3. Control Logic Complexity
4. Efficiency and Performance

CO-3 SHORT ANSWERS

8. Highlight the advantages of hardwired realization in the control unit design of a microprocessor.

Ans: Hardwired realization in the control unit design of a microprocessor offers faster instruction execution and simpler circuitry, leading to higher performance and efficiency.

1. Speed
2. Low latency
3. Security
4. Parallelism

9. Illustrate the cause of structural hazards in pipelining.

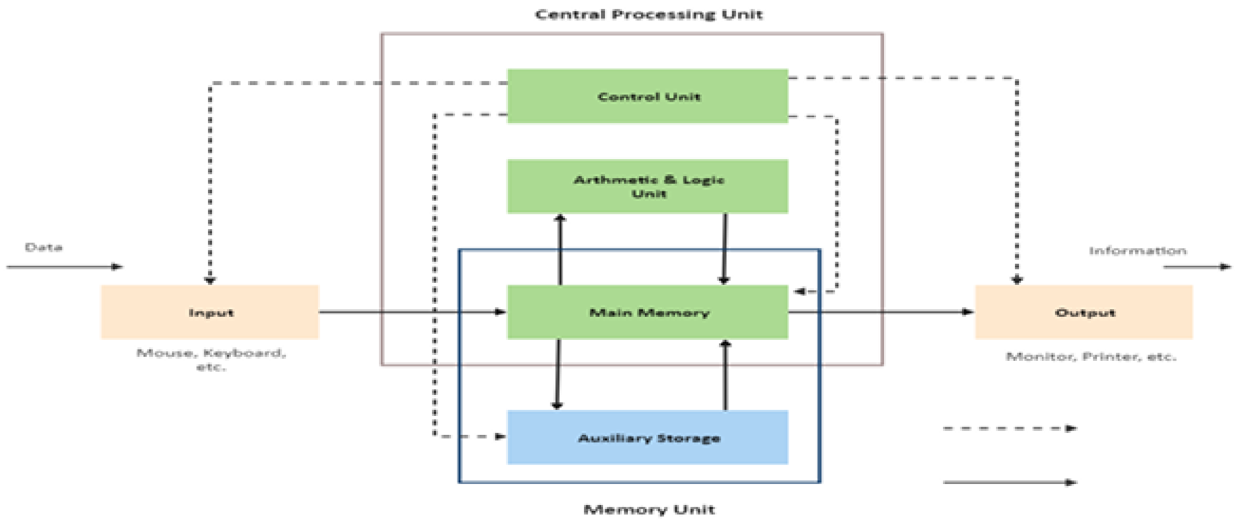
Ans: Structural hazards in pipelining occur when the hardware resources needed for instruction execution are insufficient, causing conflicts and delays in the pipeline stages.

1. Resource contention
2. Limited hardware resources
3. Instruction overlaps.
4. Memory access conflicts

CO-3 LONG ANSWERS

1.Illustrate the architecture of a CPU and its constituent blocks, elaborating on their functions.

Ans:



(i) The CPU architecture is composed of several essential blocks, including the ALU (Arithmetic Logic Unit), CU (Control Unit), Registers, and buses.

(ii) The ALU is responsible for executing arithmetic and logical operations on data received from memory or input devices.

(iii) The CU manages the execution of instructions by decoding them, generating control signals, and coordinating data movement between various components.

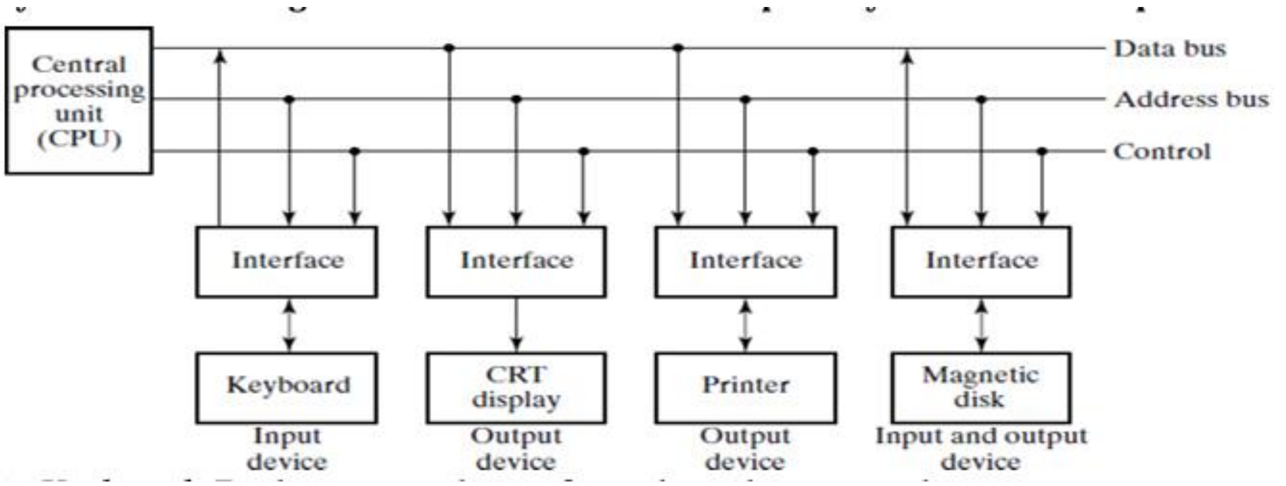
(iv)Registers are small, fast storage units located within the CPU, used to temporarily hold data, instructions, and addresses during processing.

(v)Buses are communication pathways that transfer data, addresses, and control signals between the CPU, memory, and I/O devices.

CO-3 LONG ANSWERS

2. Analyze the role and significance of I/O devices in computer systems with examples.

Ans:



(i) I/O (Input/Output) devices play a crucial role in computer systems by facilitating communication between the computer and external devices.
(ii) Input devices, such as keyboards, mice, and scanners, allow users to input data and commands into the computer.

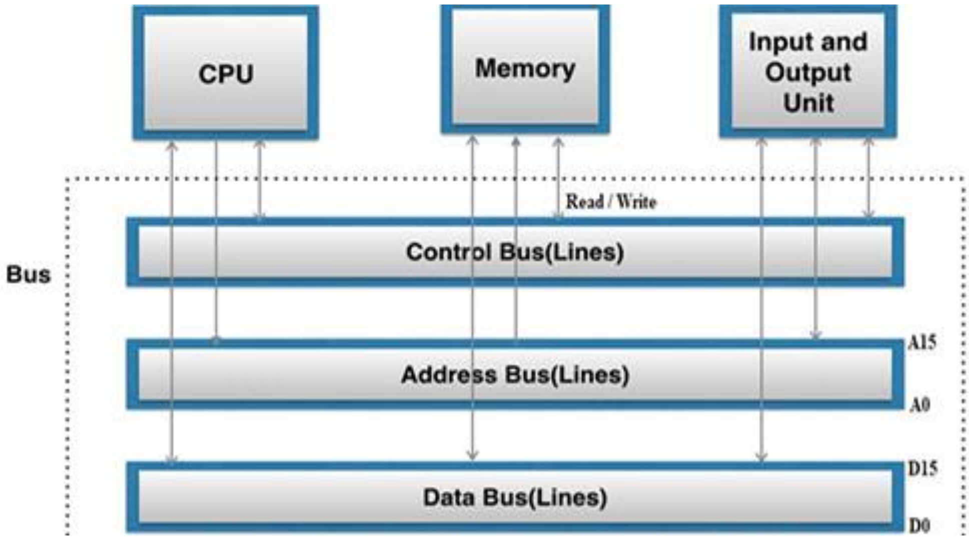
(iii) Output devices, including monitors, printers, and speakers, display or provide processed data and information to the user.

(iv)Examples of I/O devices include USB drives for data storage, network adapters for internet connectivity, and graphics cards for displaying visuals on monitors.

CO-3 LONG ANSWERS

3. Make use of various buses in microcomputer architecture to investigate on data transfer, detailing their types, functions.

Ans:



(i) Microcomputer architecture features three primary buses: the data bus, address bus, and control bus.

(ii) The data bus carries data between the CPU, memory, and I/O devices bidirectionally.

(iii) The address bus specifies memory locations for read or write operations, allowing the CPU to access specific data or instructions.

(iv)The control bus carries control signals generated by the CPU to coordinate activities such as memory read/write, I/O operations, and interrupt handling.

4. Identify the buses in microcomputer architecture, detailing their types, functions.

Ans:

(i) Microcomputer architecture features three primary buses: the data bus, address bus, and control bus.

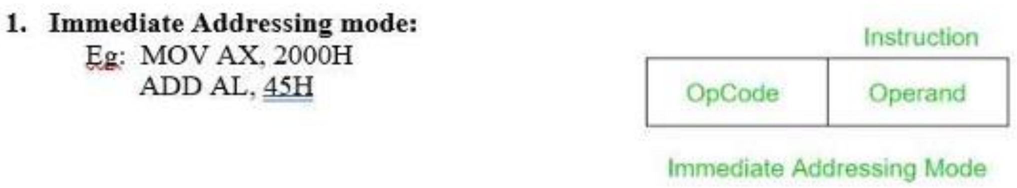
(ii) The data bus carries data between the CPU, memory, and I/O devices bidirectionally.

(iii) The address bus specifies memory locations for read or write operations, allowing the CPU to access specific data or instructions.

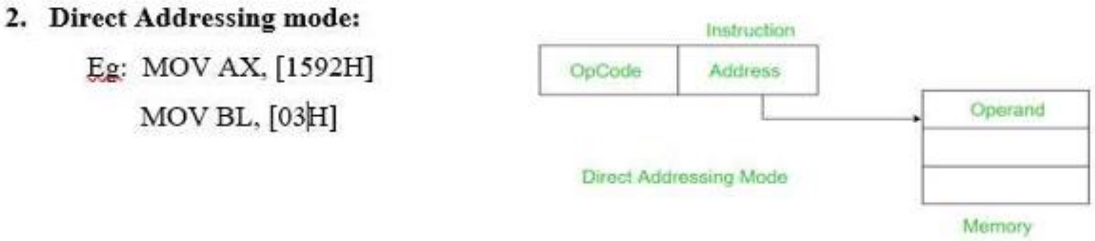
(iv)The control bus carries control signals generated by the CPU to coordinate activities such as memory read/write, I/O operations, and interrupt handling.

5. Examine the concepts of immediate, direct, and indirect addressing modes in computer architecture.

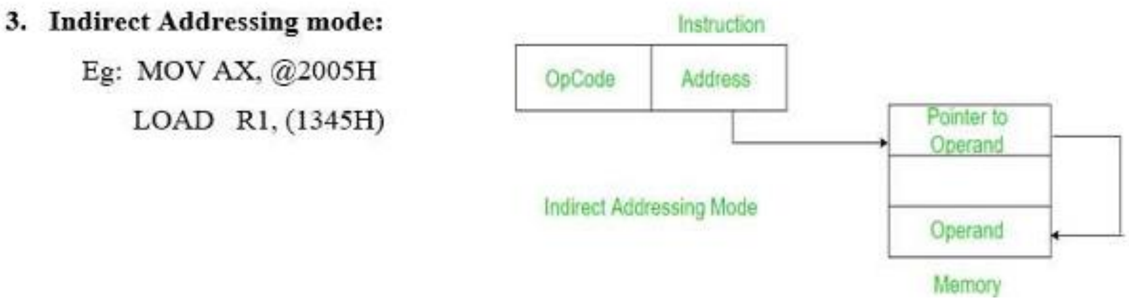
Ans:



1. **Immediate Addressing mode:**
   Eg: MOV AX, 2000H
   ADD AL, 45H



2. **Direct Addressing mode:**
   Eg: MOV AX, [1592H]
   MOV BL, [03H]



3. **Indirect Addressing mode:**
   Eg: MOV AX, @2005H
   LOAD R1, (1345H)

6. Elaborate on various instruction formats with structure and example.

Ans:

(i) Instruction formats define the structure of machine instructions, typically consisting of an opcode (operation code) and operands.

(ii) The register format specifies one or more registers as operands, such as ADD R1, R2.

(iii) Immediate format includes the opcode and a constant value as an operand, like MOV A, #10.

(iv) Direct format involves specifying a memory address directly within the instruction, such as MOV A, 500.

(v) Indirect format utilizes a memory address indirectly through a register, for instance, MOV A, [BX].

7. Examine the concepts of register, register indirect, displacement addressing modes in computer architecture.

Ans:

(i) Register addressing mode involves specifying a register directly as an operand in the instruction. For example, ADD R1, R2 adds the content of register R2 to register R1.

(ii) Register Indirect addressing mode accesses data stored at a memory location whose address is held in a register. For instance, MOV A, [BX] moves the content of the memory location pointed to by register BX into register A.

(iii) Displacement addressing mode adds an offset to a base address specified in a register to access a memory location. An example is MOV A, 100[BX], where the content of the memory location at the address (100 + content of BX) is moved into register A.

8. Interpret the concept of a machine cycle in computer architecture, outlining its phases.

Ans:

(i) A machine cycle is the basic operation performed by a CPU to execute an instruction, consisting of several phases. (

ii) Fetch: The CPU retrieves the instruction from memory.

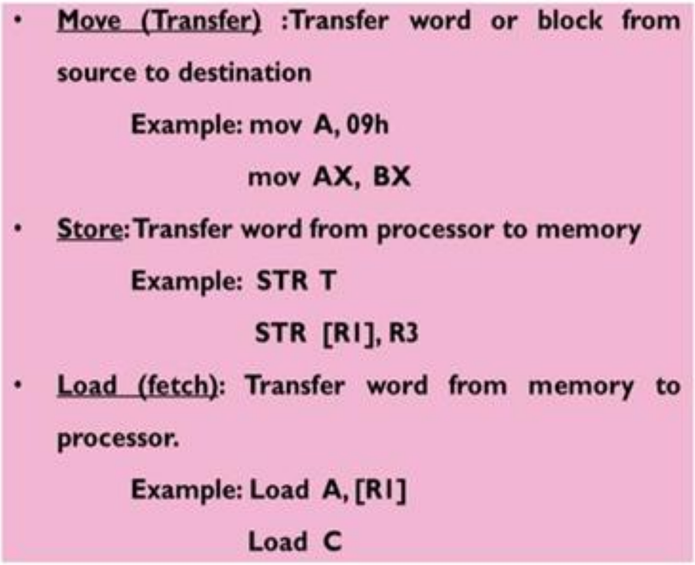
(iii) Decode: The CPU interprets the instruction and determines the action to be performed.

(iv) Execute: The CPU carries out the operation specified by the instruction.

(v) Store: The CPU writes the result back to memory or registers if necessary.

9.Discuss the data transfer and arithmetic logic instruction sets with examples.

Data Transfer Instructions:



- Move (Transfer) :Transfer word or block from source to destination

  Example: mov A, 09h

  mov AX, BX

- Store:Transfer word from processor to memory

  Example: STR T

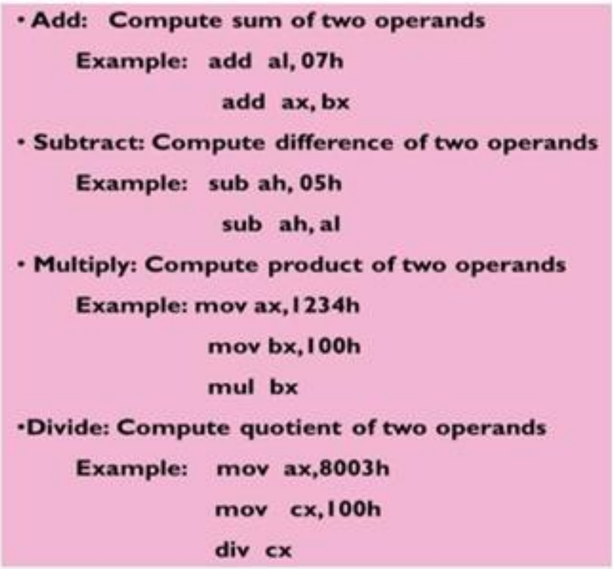
  STR [R1], R3

- Load (fetch): Transfer word from memory to processor.

  Example: Load A, [R1]

  Load C

Arithmetic l instruction:



- Add: Compute sum of two operands

  Example: add al, 07h

  add ax, bx

- Subtract: Compute difference of two operands

  Example: sub ah, 05h

  sub ah, al

- Multiply: Compute product of two operands

  Example: mov ax, 1234h

  mov bx, 100h

  mul bx

- Divide: Compute quotient of two operands

  Example: mov ax,8003h

  mov cx,100h

  div cx

10.Consider a scenario where you are developing a simple combinational circuit of basic logical operations. Make use of the related instruction sets and accomplish the task.

Let us consider the expression $F=A.B+C'.D$ to develop using the Logical instruction set.

Load input values

LOAD R0, A ; Load input A into register R0

LOAD R1, B ; Load input B into register R1

LOAD R2, C ; Load input C into register R2

LOAD R3, D ; Load input D into register R3

Perform logical operations

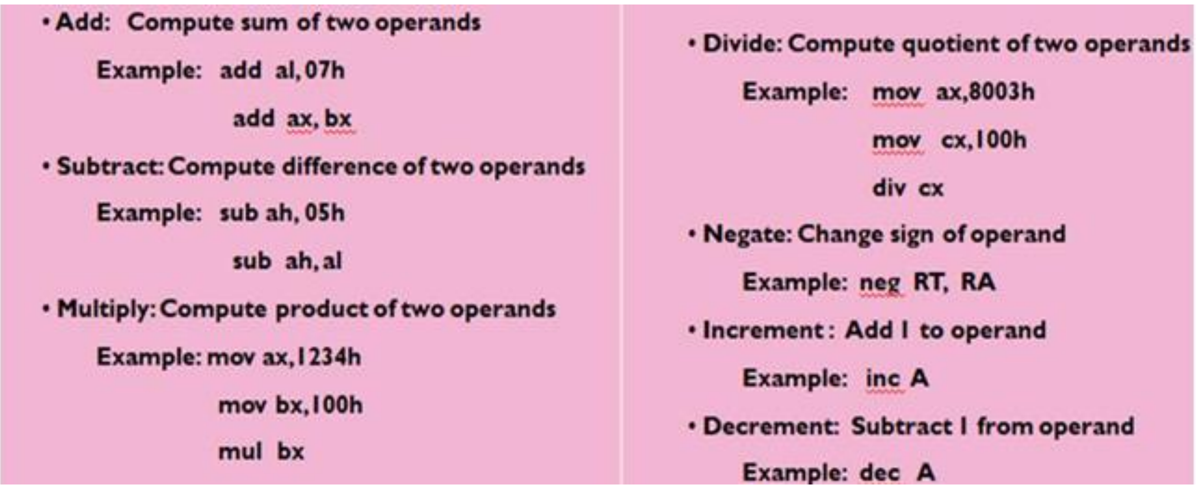
AND R5, R0, R1 ; Compute A AND B and store the result in R5

NOT R6, R2 ; Compute the complement of C and store it in R6

AND R7, R6, R3 ; Compute (NOT C) AND D and store the result in R7

OR R4, R5, R7 ; Compute (A AND B) OR ((NOT C) AND D) and store the result in R4

11. Consider a scenario where you are developing a simple calculator application for a mobile device. You want to implement basic arithmetic operations. Make use of the related instruction sets and accomplish the task.



- Add: Compute sum of two operands

  Example: add al,07h

  add ax, bx

- Subtract: Compute difference of two operands

  Example: sub ah, 05h

  sub ah, al

- Multiply: Compute product of two operands

  Example: mov ax,1234h

  mov bx,100h

  mul bx

- Divide: Compute quotient of two operands

  Example: mov ax,8003h

  mov cx,100h

  div cx

- Negate: Change sign of operand

  Example: neg RT, RA

- Increment : Add 1 to operand

  Example: inc A

- Decrement: Subtract 1 from operand

  Example: dec A

12. Explore the subroutine call and return mechanism in computer programming.

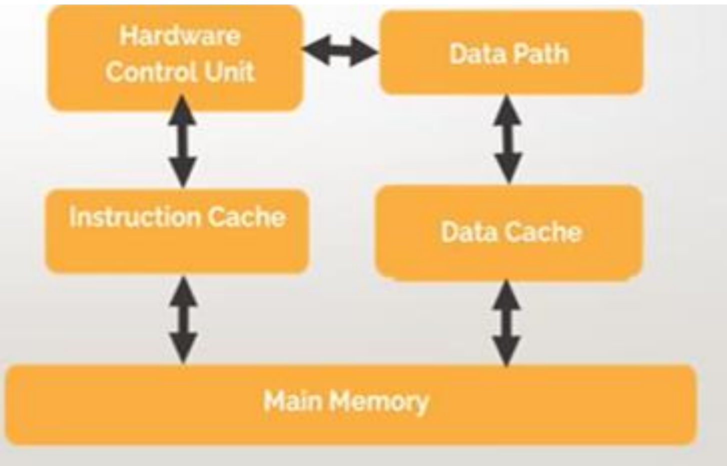
Ans:

(i) Subroutine call is used to transfer control to a specific subroutine or function within a program. It involves saving the current execution point and transferring control to the subroutine.

(ii) Return mechanism allows the subroutine to return control to the calling program after execution. It involves restoring the saved execution point and resuming execution from the instruction following the call.

13. Develop the model of RISC architecture, detailing its design philosophy, key features.

Ans:



Design Philosophy:

RISC (Reduced Instruction Set Computing) architecture focuses on simplicity and efficiency by using a smaller set of simple instructions.

Key Features:

(i) **Simple Instructions**: RISC processors have a reduced instruction set, with each instruction performing a basic operation.

(ii) **Uniform Instruction Format**: Instructions typically have a fixed length and a uniform format, simplifying decoding and execution.

(iii) **Register-Centric**: RISC architectures heavily utilize registers for operand storage, reducing memory accesses and improving performance.

(iv) **Pipelining**: RISC processors often employ pipelining to overlap instruction execution stages, enhancing throughput and efficiency.

12. Differentiate CISC and RISC architectures in computer organization, discussing their design principles.

Ans:

(i) CISC (Complex Instruction Set Computing) architectures emphasize complex instructions capable of performing multiple operations in a single instruction.

(ii) RISC architectures prioritize simplicity and efficiency by using a smaller set of simple instructions.

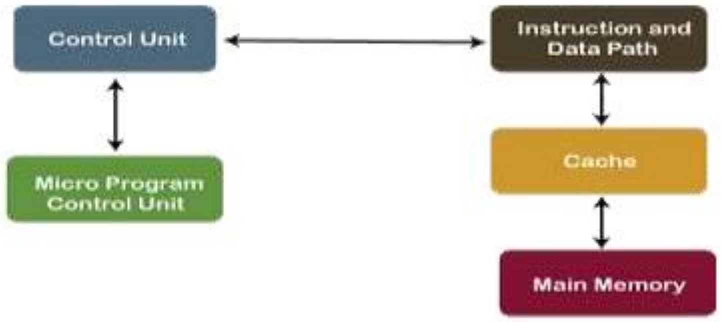
(iii) CISC processors tend to have variable-length instructions, while RISC processors usually have fixed-length instructions.

(iv) CISC architectures often include specialized instructions for common tasks, while RISC architectures rely on optimizing compiler techniques and hardware pipelining.

(v) CISC architectures may have more complex instruction decoding and execution units, leading to higher power consumption and complexity compared to RISC architectures.

13. Develop the model of CISC architecture, detailing its design philosophy, key features.

Ans:



CISC Architecture

Design Philosophy:

CISC (Complex Instruction Set Computing) architecture aims to provide a wide variety of complex instructions capable of performing multiple tasks in a single operation.

Key Features:

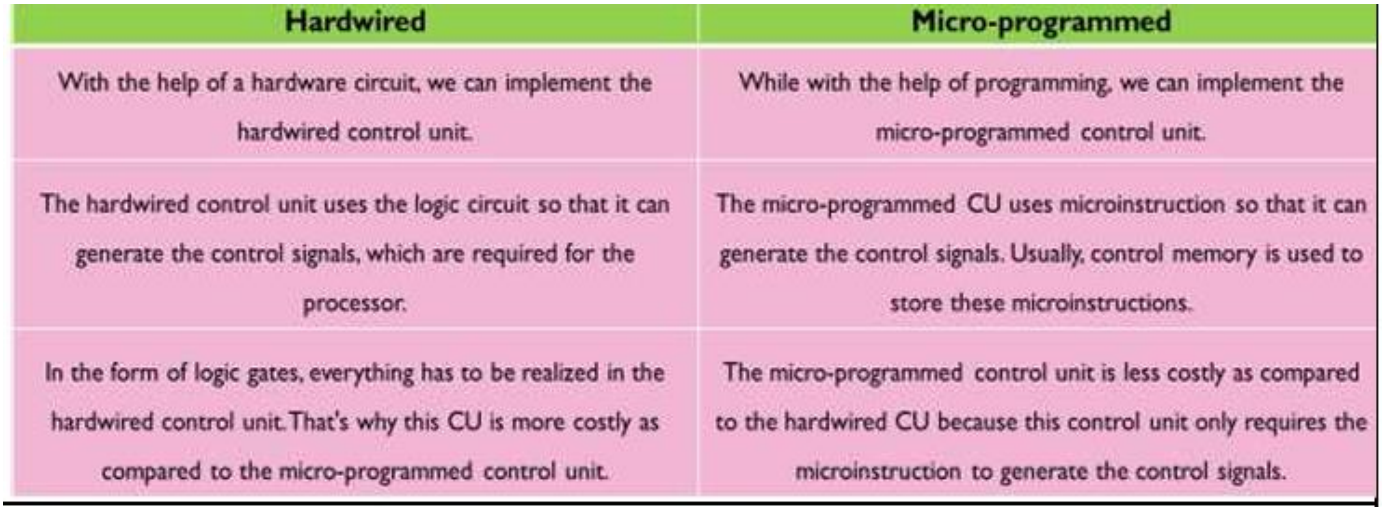
(i) **Rich Instruction Set**: CISC processors feature a large and diverse set of instructions, including complex operations such as string manipulation and f loating-point arithmetic.

(ii) **Variable-Length Instructions**: Instructions in CISC architectures may vary in length, allowing for more flexibility in encoding complex operations.

(iii**) Memory-to-Memory Operations**: CISC architectures support direct memory-to-memory operations, allowing operations to be performed directly on memory locations.

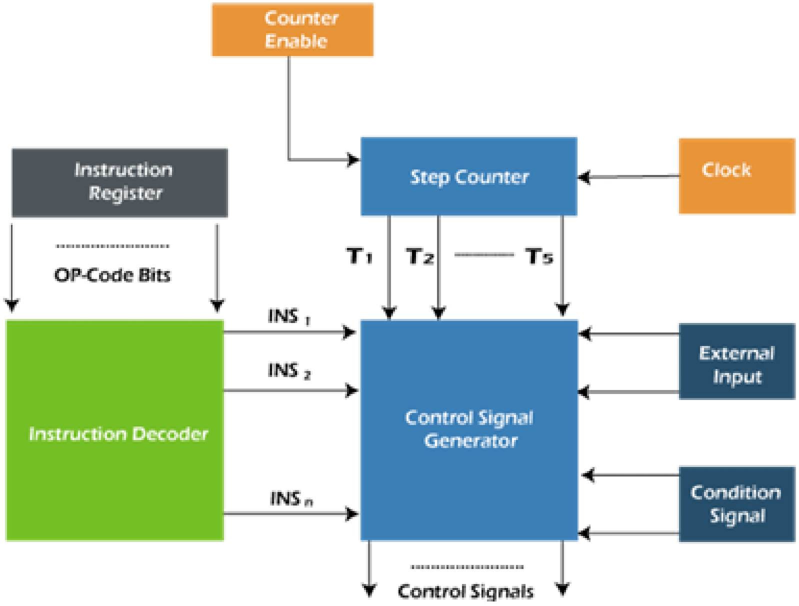
(iv) **Specialized Instructions**: CISC processors include specialized instructions for common tasks, reducing the need for complex sequences of simpler instructions.

14. Differentiate hardwired realization and micro- programmed realization in the design of control units within a CPU.



Hardwired	Micro-programmed
With the help of a hardware circuit, we can implement the hardwired control unit.	While with the help of programming, we can implement the micro-programmed control unit.
The hardwired control unit uses the logic circuit so that it can generate the control signals, which are required for the processor.	The micro-programmed CU uses microinstruction so that it can generate the control signals. Usually, control memory is used to store these microinstructions.
In the form of logic gates, everything has to be realized in the hardwired control unit. That's why this CU is more costly as compared to the micro-programmed control unit.	The micro-programmed control unit is less costly as compared to the hardwired CU because this control unit only requires the microinstruction to generate the control signals.

15. Illustrate the hardwired realization in CPU design,detailing its architecture.

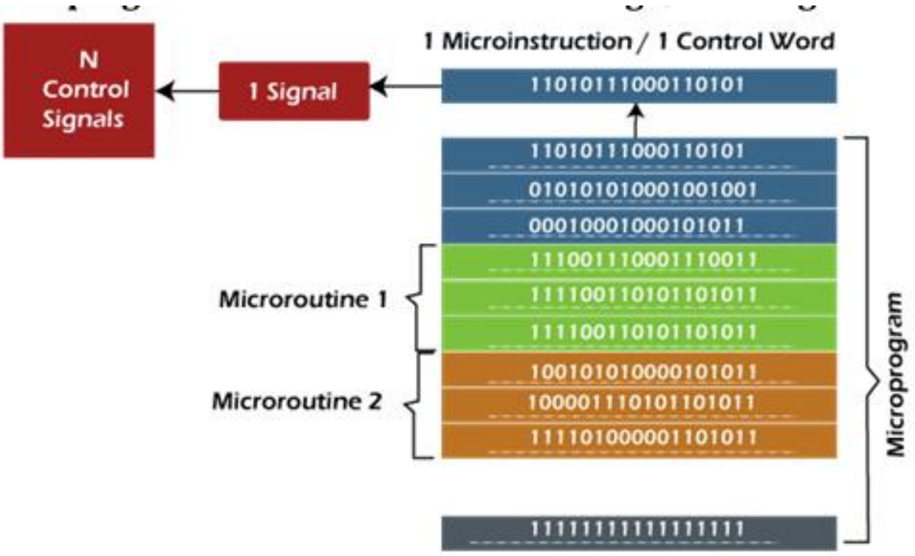


1. Hardwired control units execute instructions by generating control signals at the right time and sequence.

2. Faster than micro-programmed units, they use PLA circuit and state counter to generate control signals.

3. Hardware-based, they employ circuitry to produce control signals needed by the CPU for operation.

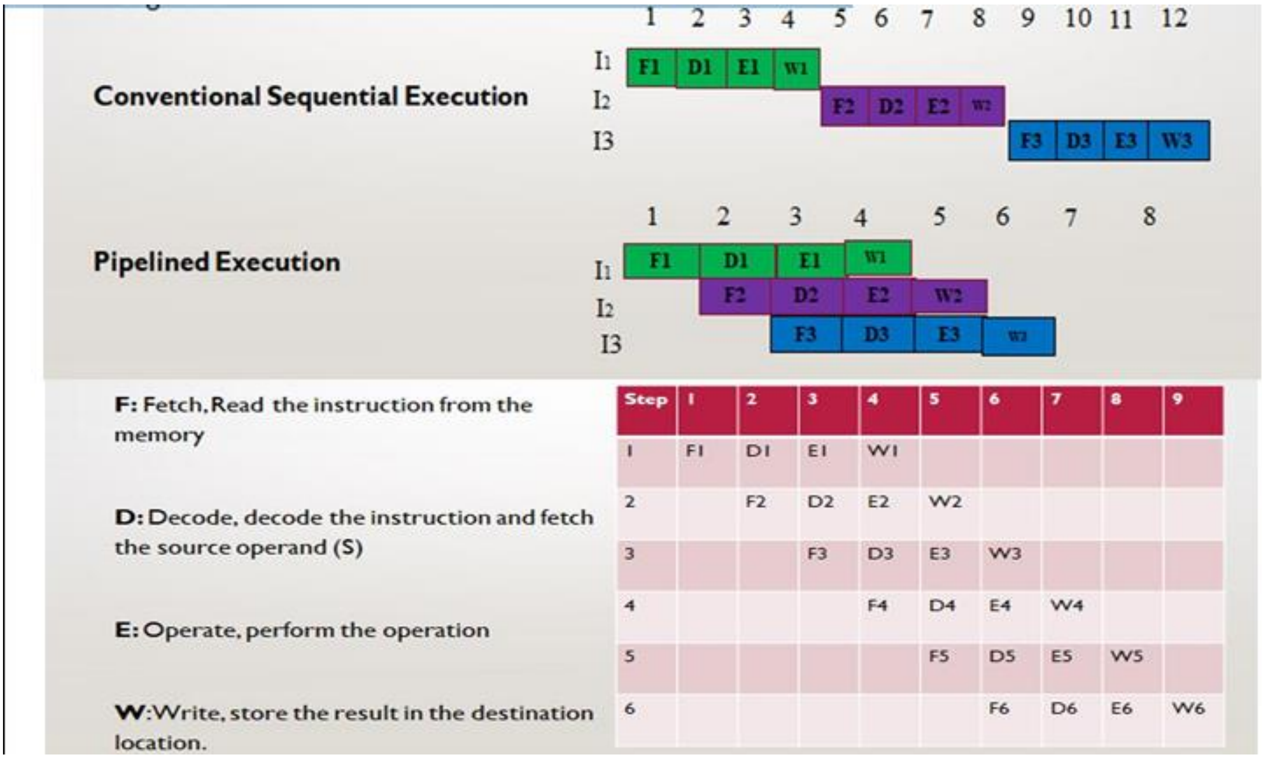
16. Illustrate the micro-programmed realization in CPU design, detailing its architecture.



1. Micro-programmed control units operate as basic logic circuits.
2. They execute instructions by generating control signals and sequencing through microinstructions.
3. Microprograms stored in fast memory, termed control store or control memory, dictate the sequence of control signals for each instruction, facilitating efficient execution.

CO-3 LONG ANSWERS

17. Design the operational flow for a pipelined processor with four stages, outlining each stage's function and how instructions progress through the pipeline.



F: Fetch, Read the instruction from the memory

D: Decode, decode the instruction and fetch the source operand (S)

E: Operate, perform the operation

W: Write, store the result in the destination location.

Step	1	2	3	4	5	6	7	8	9
1	F1	D1	E1	W1					
2		F2	D2	E2	W2				
3			F3	D3	E3	W3			
4				F4	D4	E4	W4		
5					F5	D5	E5	W5	
6						F6	D6	E6	W6

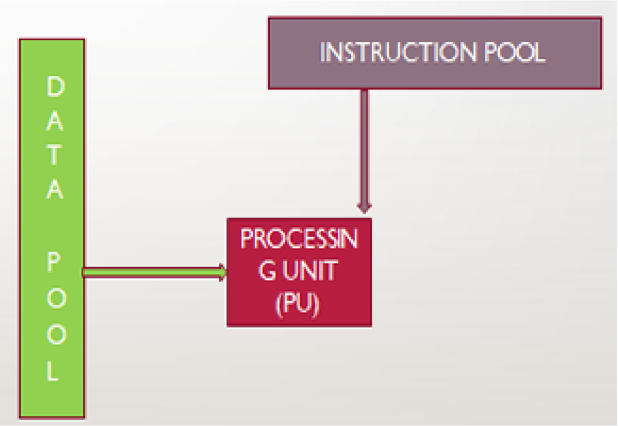
18. In order to determine the city with the most consistently warm temperatures from a large weather dataset, which parallel processing architecture (SISD, SIMD, MISD, MIMD) would be best suited for efficient analysis and why?

Ans:

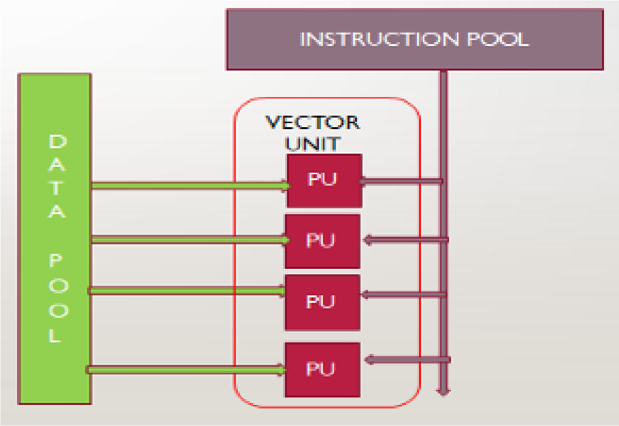
For determine the city with the most consistently warm temperatures from a large weather dataset, most appropriate architecture would be MIMD (Multiple Instruction, Multiple Data) due to following reasons.

1. Independent Processing
2. Flexibility and Scalability
3. Complexity of Analysis
4. Data Distribution and Communication

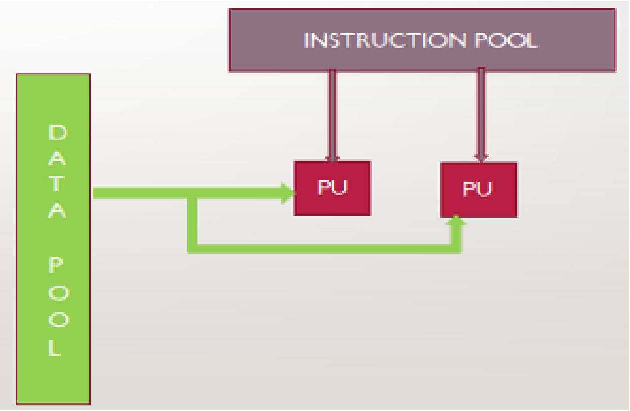
SISD



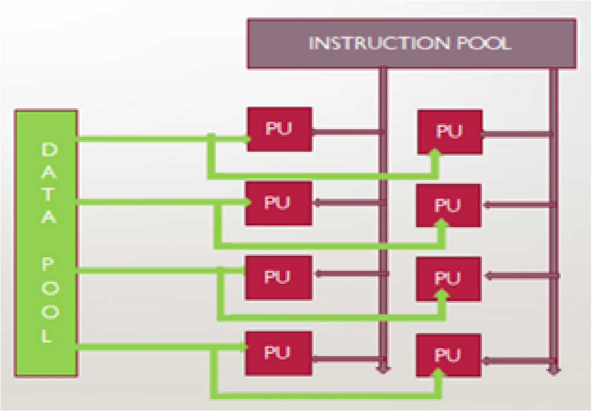
SIMD



MISD

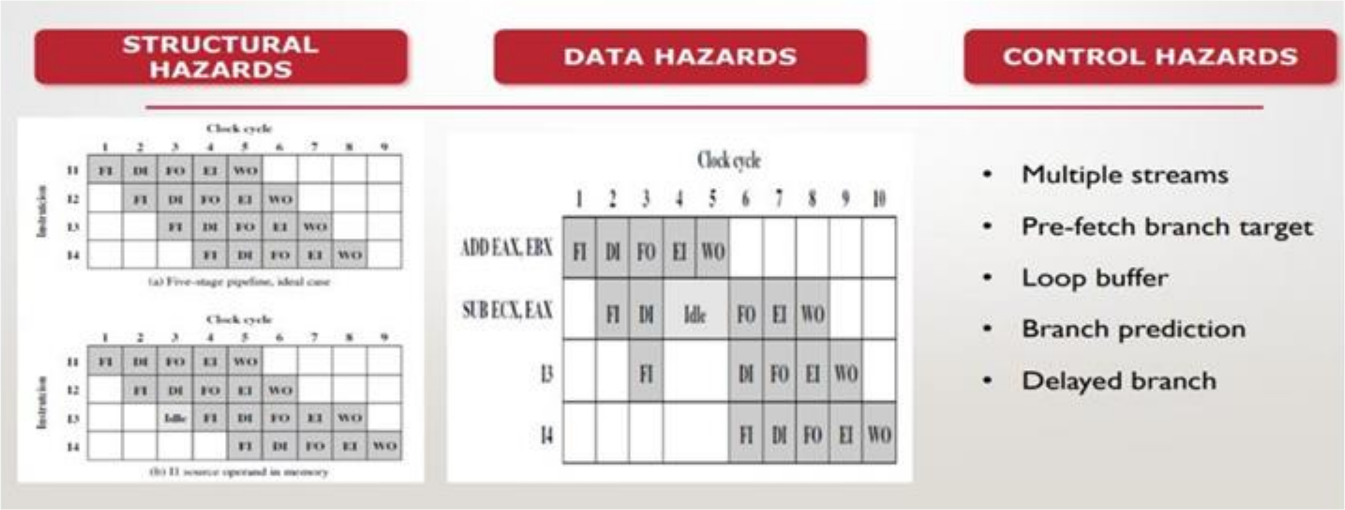


MIMD



19. Identify the various pipelining hazards in processor design, detailing the types of hazards, their causes.

Ans:



STRUCTURAL HAZARDS

Clock cycle

Instruction	1	2	3	4	5	6	7	8	9
I1	FI	DI	FO	EI	WO				
I2		FI	DI	FO	EI	WO			
I3			FI	DI	FO	EI	WO		
I4				FI	DI	FO	EI	WO	

(a) Five-stage pipeline, ideal case

Clock cycle

Instruction	1	2	3	4	5	6	7	8	9
I1	FI	DI	FO	EI	WO				
I2		FI	DI	FO	EI	WO			
I3			Idle	FI	DI	FO	EI	WO	
I4					FI	DI	FO	EI	WO

(b) I1 source operand in memory

DATA HAZARDS

Clock cycle

	1	2	3	4	5	6	7	8	9	10
ADD EAX, EBX	FI	DI	FO	EI	WO					
SUB ECX, EAX		FI	DI	Idle		FO	EI	WO		
I3			FI			DI	FO	EI	WO	
I4						FI	DI	FO	EI	WO

CONTROL HAZARDS

- Multiple streams
- Pre-fetch branch target
- Loop buffer
- Branch prediction
- Delayed branch

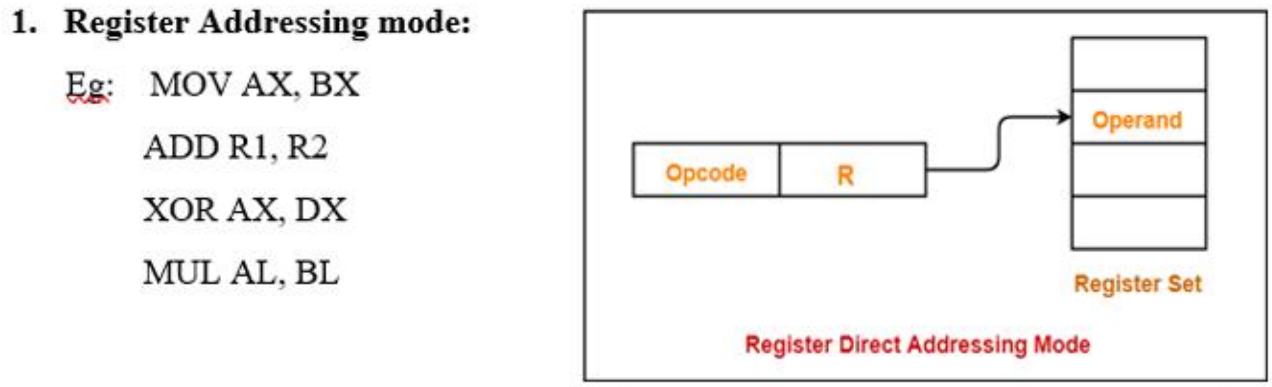
(i) Structural Hazards: Arise when hardware resources are insufficient to execute multiple instructions simultaneously. Causes include resource contention and limited hardware resources.

(ii) Data Hazards: Occur when instructions depend on the results of previous instructions, leading to data dependencies. Causes include data hazards and pipeline interlocks.

(iii) Control Hazards: Arise from changes in program flow, such as branches or jumps, leading to incorrect speculation or pipeline stalls. Causes include branch hazards and delayed branches.

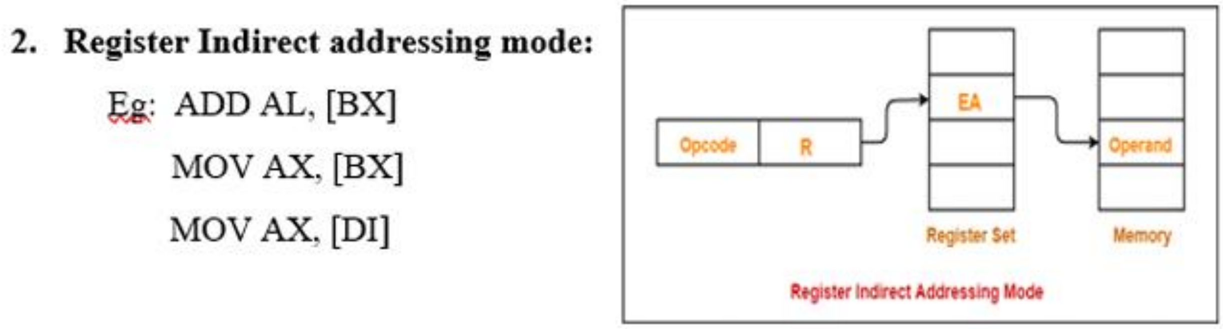
20. Examine the concepts of register, register indirect, displacement addressing modes in computer architecture.

Ans:



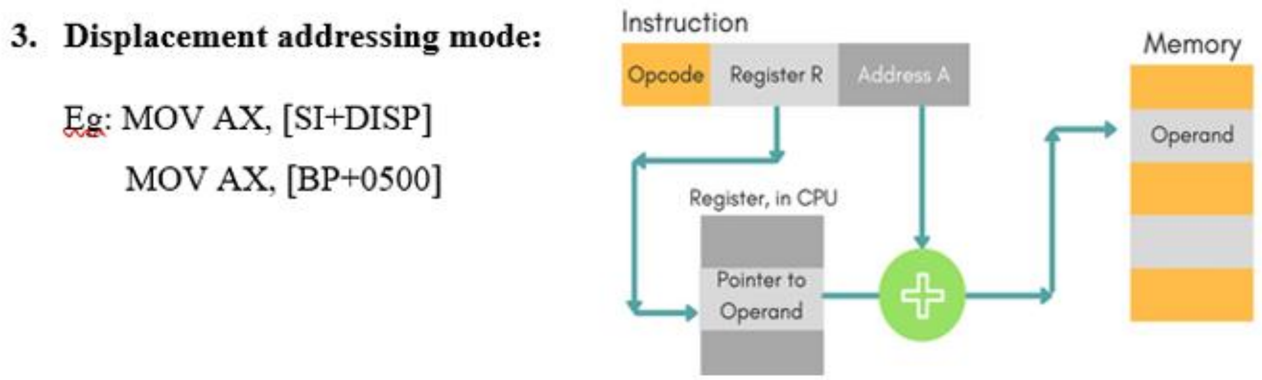
1. **Register Addressing mode:**

Eg: MOV AX, BX
ADD R1, R2
XOR AX, DX
MUL AL, BL



2. **Register Indirect addressing mode:**

Eg: ADD AL, [BX]
MOV AX, [BX]
MOV AX, [DI]



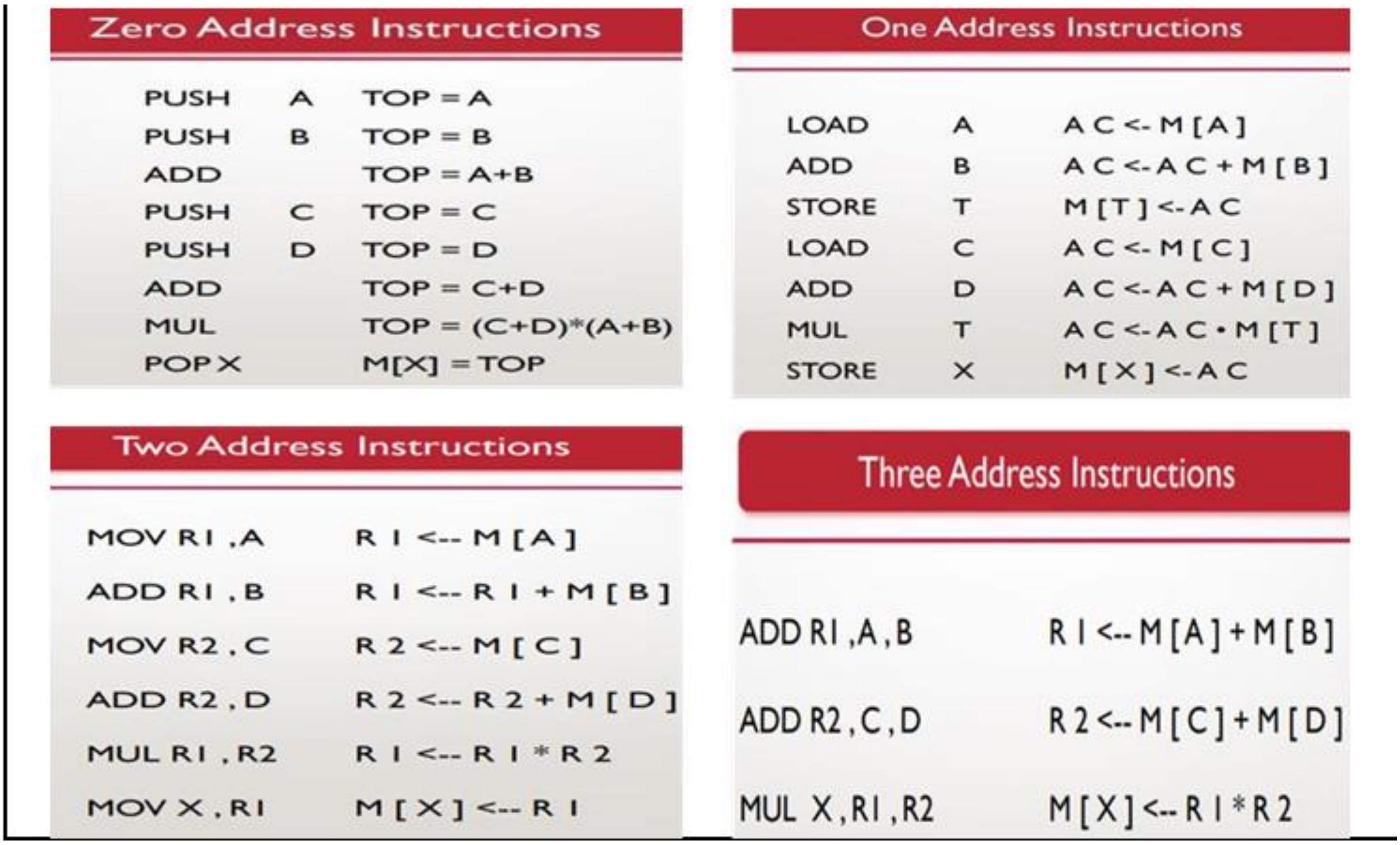
3. **Displacement addressing mode:**

Eg: MOV AX, [SI+DISP]
MOV AX, [BP+0500]

CO-3 LONG ANSWERS

21. Model the structures of different instruction formats and provide illustrative examples for each.

Ans:



Zero Address Instructions

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

One Address Instructions

LOAD	A	AC <- M[A]
ADD	B	AC <- AC + 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

Two Address Instructions

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

Three Address Instructions

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

22. Analyze any four different instruction sets used in execution.

Ans:

(i) **x86**: Widely used in PCs, featuring a variable-length instruction format and complex instructions capable of performing multiple tasks.

(ii**) ARM:** Commonly used in mobile devices and embedded systems, known for its reduced instruction set and energy efficiency.

(iii) **MIPS**: Often employed in academic and research environments, featuring a simple instruction set with fixed-length instructions.

(iv) **PowerPC**: Historically used in Apple Macintosh computers and gaming consoles, known for its high-performance architecture and RISC-based design.