RAJASTHAN TECHNICAL UNIVERSITY, KOTA

Syllabus

III Year-VI Semester: B.Tech. Computer Science and Engineering

6CS4-04: Computer Architecture and Organization

Credit: 3 **Max. Marks: 150(IA:30, ETE:120)**

3L+0T+0P **End Term Exam: 3 Hours**

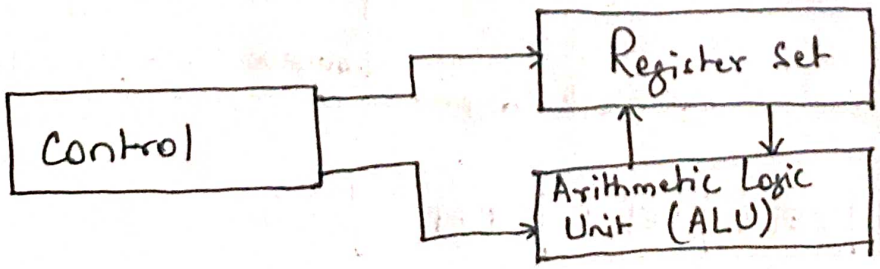
SN	Contents	Hours
1	**Introduction:** Objective, scope and outcome of the course.	01
2	**Computer Data Representation:** Basic computer data types, Complements, Fixed point representation, Register Transfer and Micro-operations: Floating point representation, Register Transfer language, Register Transfer, Bus and Memory Transfers (Tree-State Bus Buffers, Memory Transfer), Arithmetic Micro-Operations, Logic Micro-Operations, Shift Micro-Operations, Arithmetic logical shift unit. Basic Computer Organization and DesignInstruction codes, Computer registers, computer instructions, Timing and Control, Instruction cycle, Memory-Reference Instructions, Input-output and interrupt, Complete computer description, Design of Basic computer, design of Accumulator Unit.	10
3	**Programming The Basic Computer:** Introduction, Machine Language, Assembly Language, assembler, Program loops, Programming Arithmetic and logic operations, subroutines, I-O Programming. Micro programmed Control: Control Memory, Address sequencing, Micro program Example, design of control Unit	7
4	**Central Processing Unit:** Introduction, General Register Organization, Stack Organization, Instruction format, Addressing Modes, data transfer and manipulation, Program Control, Reduced Instruction Set Computer (RISC)Pipeline And Vector Processing, Flynn's taxonomy, Parallel Processing, Pipelining, Arithmetic Pipeline, Instruction, Pipeline, RISC Pipeline, Vector Processing, Array Processors	8
5	**Computer Arithmetic:** Introduction, Addition and subtraction, Multiplication Algorithms (Booth Multiplication Algorithm), Division Algorithms, Floating Point Arithmetic operations, Decimal Arithmetic Unit. Input-Output Organization, Input-Output Interface, Asynchronous Data Transfer, Modes Of Transfer, Priority Interrupt, DMA, Input-Output Processor (IOP), CPUIOP Communication, Serial communication.	8
6	**Memory Organization:** Memory Hierarchy, Main Memory, Auxiliary Memory, Associative Memory, Cache Memory, Virtual Memory. Multipreocessors: Characteristics of Multiprocessors, Interconnection Structures, Inter-processor Arbitration, Inter-processor Communication and Synchronization, Cache Coherence, Shared Memory Multiprocessors.	8
	Total	42

PAGE NO.

Central Processing Unit

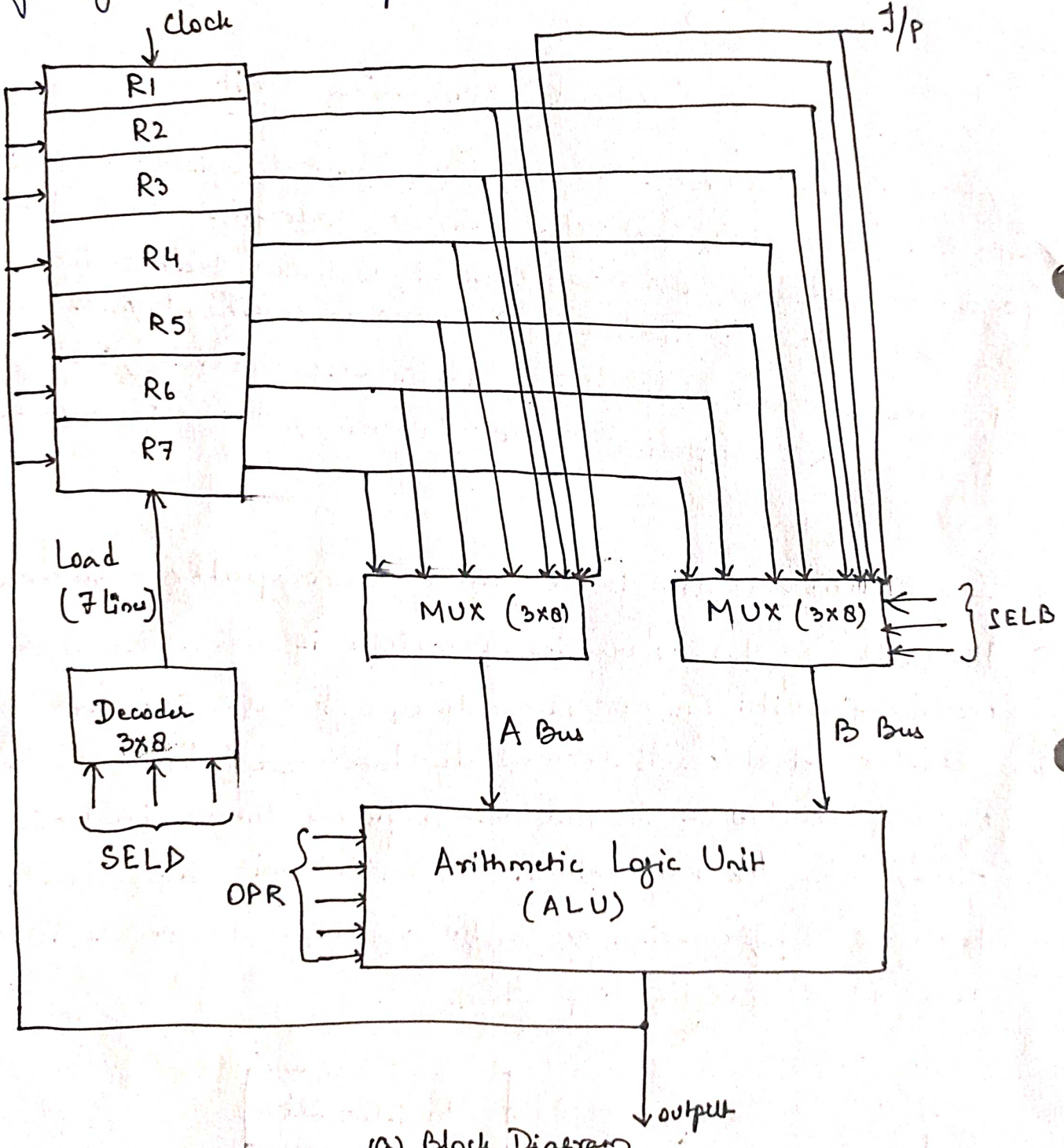
Central Processing Unit:- Introduction, General Register Organization, Stack Organization, Instruction format, Addressing modes, data transfer and manipulation, program control, Reduced Instruction set computer (RISC) Pipeline and Vector Processing, Flynn's taxonomy, Parallel processing, Pipelining, Arithmetic Pipelining, Instruction, Pipeline, RISC Pipeline, Vector Processing, Array Processors.

Introduction:- CPU is the part of the computer that Performs the bulk of data-Processing operations is called the central Processing unit. The CPU is made up of three major Parts, the register set stores intermediate data used during the execution of the instructions. The ALU unit Performs the required microoperation for executing the instructions. The control unit supervises the transfer of information among the register and instructs the ALU as to which operation to perform.



Major Components of CPU

General Register Organization:- It is more convenient and more efficient to store these intermediate values in processor registers. When a large number of registers are included in CPU, it is the most effecient to connect them through a common bus system. The registers communicate with each other not only for direct data transfers, but also while Performing various microoperations.



(a) Block Diagram

SEL A	SEL B	SEL D	OPR

(b) Control word

Register Set with Common ALU

PAGE NO. .....

The operation selected in the ALU determines the arithmetic or logic micro-operation that is to be performed. The result of the micro-operation is available for o/p data and also goes into the inputs of all the registers. The register that receives the information from the o/p bus is selected by a decoder.

The decoder activates one of the register load inputs, thus providing a transfer path b/w the data in the o/p bus and the inputs of the selected destination register.

The control unit that operates the CPU bus system directs the information flow through the registers and ALU by selecting the various components in the system. For eg: to perform the operation.

$$R1 \leftarrow R2 + R3$$

the control must provide binary selection variables to the following selector inputs:-

1. MUX A selector (SELA): to place the content of R2 into bus A.
2. MUX B selector (SELB): to place the content of R3 into bus B.
3. ALU Operation selector (OPR):- to provide the arithmetic addition A+B.
4. Decoder destination selector (SELD): to transfer the content of the output bus into R1.

Control word:- There are 14 binary selection inputs in the unit and their combined value specifies a control word.

Encoding of Register selection fields

Binary Code	SELA	SELB	SELD
000	Input	Input	None
001	R1	R1	R1
010	R2	R2	R2
011	R3	R3	R3
100	R4	R4	R4
101	R5	R5	R5
110	R6	R6	R6
111	R7	R7	R7

Encoding of ALU operations

OPR select	Operation	Symbol
00000	Transfer A	TSFA
00001	Increment A	INCA
00010	Add A+B	ADD
00101	Subtract A-B	SUB
00110	Decrement A	DECA
01000	AND A and B	AND
01010	OR A and B	OR
01100	XOR A and B	XOR
01110	Complement A	COMA
10000	Shift right A	SHRA
11000	Shift Left A	SHLA

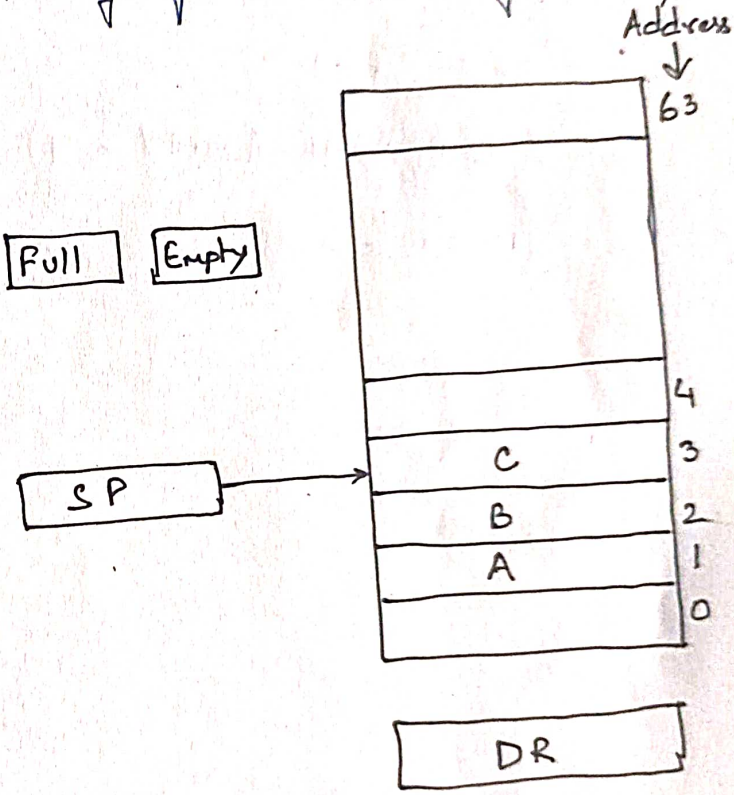
Examples of Microoperations for the CPU.

Microoperation	Symbolic Designation SELA	SELB	SELD	OPR	Control word			
R1 ← R2 − R3	R2	R3	R1	SUB	010	011	001	00101
R4 ← R4 ∨ R5	R4	R5	R4	OR	100	101	100	01010
R6 ← R6 + 1	R6	—	R6	INCA	110	000	110	00001
R7 ← R1	R1	—	R7	TSFA	001	000	111	00000
Output ← R2	R2	—	None	TSFA	010	000	000	00000
Output ← Input	Input	—	None	TSFA	000	000	000	00000
R4 ← shl R4	R4	—	R4	SHLA	100	000	100	11000
R5 ← 0	R5	R5	R5	XOR	101	000	101	01100

Stack Organization:- A stack is a storage device that stores information is such a manner that the item stored last is the first item retrived. The operation of the stack can be compared to a stack of trays. The last tray placed on the top of the stack is the first to be taken off.

The two operations of a stack are the insertion and deletion of items. The operation of insertion is called push (or push down). The operation of deletion is called pop (or pop-up) because it can be thought of as the result of removing one item so that the stack pops up.

Register stack:- A stack can be placed in a portion of a large memory or it can be organized as a collection of a finite number of memory words or registers.



Block Diagram of a 64-word stack.

Initially, SP is cleared to 0, EMPTY is set to 1, and FULL is cleared to 0, so that SP points to the word at address 0 and the stack is marked empty and not full. If the stack is not full if (FULL=0), a new item is inserted with a push operation.

SP ← SP+1	Increment stack pointer.
M[SP] ← DR	Write item on top of the stack.
If (SP=0) then (FULL←1)	Check if stack is full.
EMPTY ← 0	Mark the stack not empty.

The stack pointer is incremented so that it points to the address of the next higher word. A memory write operation inserts the word from DR into the top of the stack. The first item stored in the stack is at address 1 and last item is stored at address 0. if SP reaches 0, so FULL is set to 1.

This condition is reached if the top item prior to the last Push was in location 63 and after incrementing SP, the last item is stored in location 0.

A new item is deleted from the stack if the stack is not empty (if EMPTY = 0). The pop operation consists of the following of micro operations.

DR ← M[SP]	Read item from the top of stack
SP ← SP - 1	Decrement stack Pointer.
If (SP = 0) the (EMPTY ← 1)	Check if stack is empty.
FULL ← 0	Mark the stack not full.

PAGE NO.

Instruction formats :- A computer will usually have a variety of instruction code formats. it is the function of the control unit within the CPU to interpret each instruction code and provide the necessary control functions needed to process the instruction.

The bits of the instruction are divided into groups called fields. The most common fields found in instruction formats are:-

1. An operation code field that specifies the operation to be Performed.
2. An address field that designates a memory address or a Processor register.
3. A mode field that specifies the way the operand or the effective address is determined.

- Three types of CPU organizations:-
  1. Single accumulator organizations
  2. General register organization
  3. Stack organizations.

Ex:-

ADD X means AC ← AC + M[X].

ADD R1, R2, R3 means R1 ← R2 + R3

ADD R1, R2 means R1 ← R1 + R2

MOV R1, R2 means R1 ← R2

ADD R1, X means R1 ← R1 + M[X]

PUSH X means push the word at address X to the top of the stack. The stack pointer is updated automatically.

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

It has assumed that the computer has two processor register, R1 and R2.

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

One Address Instructions:- One address instructions use an implied accumulator (AC) register for all data manipulation.

LOAD A	AC ← M[A]
STORE T	AC ← AC + M[B]
ADD B	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

Zero address Instructions:- The PUSH and POP instructions, need an address field to specify the operand that communicates with the stack. (TOS stands top of stack).

PUSH A	TOS ← A
PUSH B	TOS ← B
ADD	TOS ← (A+B)
PUSH C	TOS ← C
PUSH D	TOS ← D
ADD	TOS ← (C+D)
MUL	TOS ← (C+D) * (A+B)
POP X	M[X] ← TOS

Addressing Modes :- The operation field of an instruction specifies the operation to be performed. This operation will be executed on some data which is stored in computer registers or the main memory. The way any operand is selected during the program execution is dependent on the addressing mode of the instruction. The purpose of using addressing modes is as follows:-

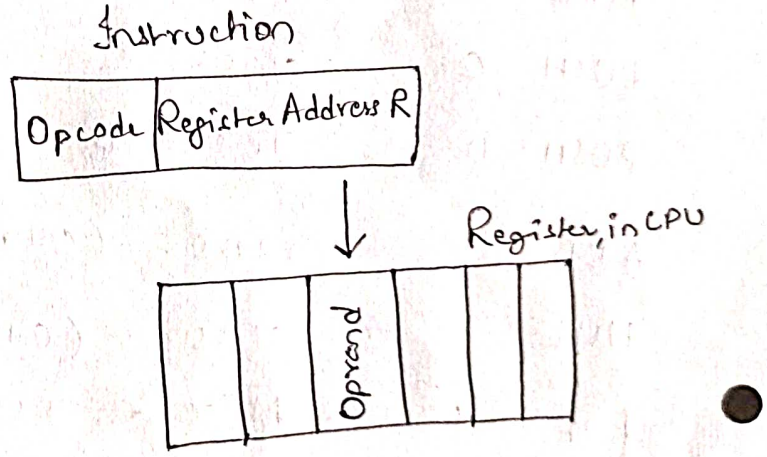
1. To give the programming versatility to the user.
2. To reduce the number of bits in addressing field of instruction.

Types of addressing Modes:-

1. **Immediate Mode:-** In this mode, the operand is specified in the instruction itself. An immediate mode instruction has an operand field rather than the address field.

   for eg:- ADD 7, which says Add 7 to contents of accumulator
   7 is the operand here.

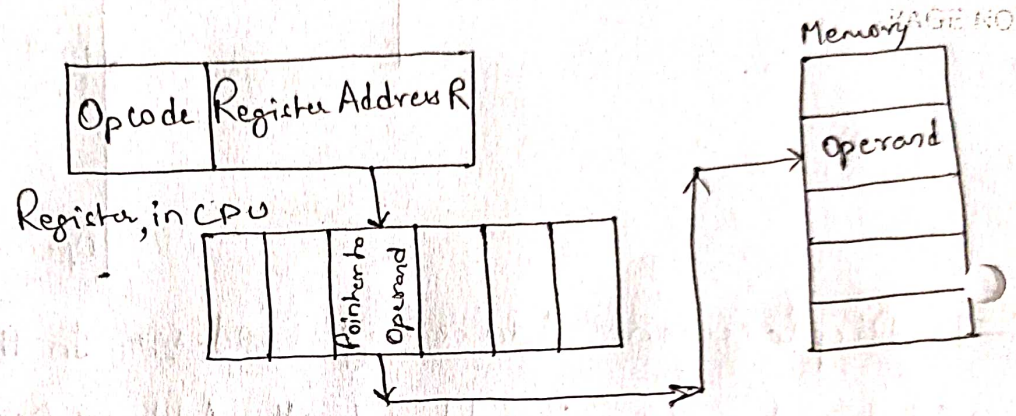
2. **Register Mode:-** In this mode the operand is stored in the register and this register is present in CPU. The instruction has the address of the register where the operand is stored.



Advantages
- Shorter instructions and faster instruction fetch.
- Faster memory access to the operand (s).

Disadvantages
- Very limited address space
- using multiple registers helps performance but it complicates the instruction.

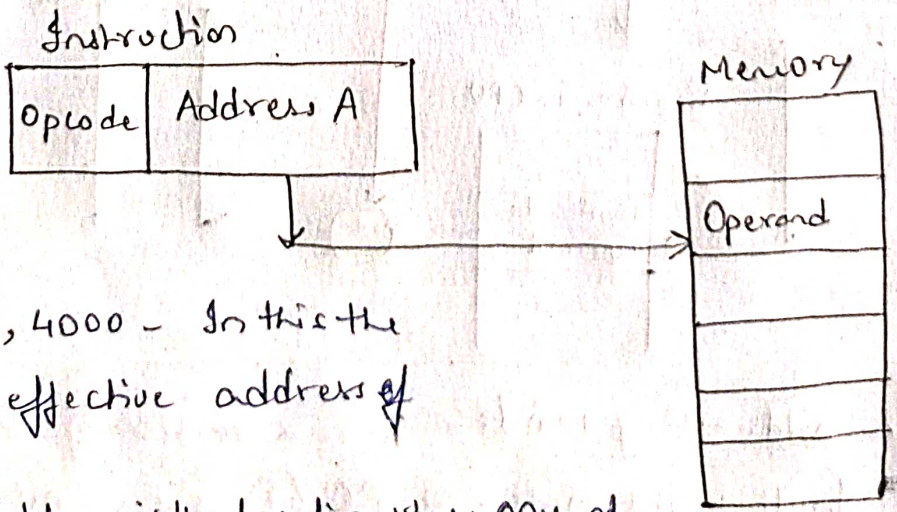
3. **Register Indirect Mode:-** In this mode, the instruction specifies the register whose contents give us the address of operand which is in memory. Thus, the register contains the address of operand rather than the operand itself.



4. Auto Increment/Decrement Mode:- In this the register is incremented or decremented after or before its value is used.

5. Direct Addressing Mode:- In this mode, effective address of operand is present in instruction itself.

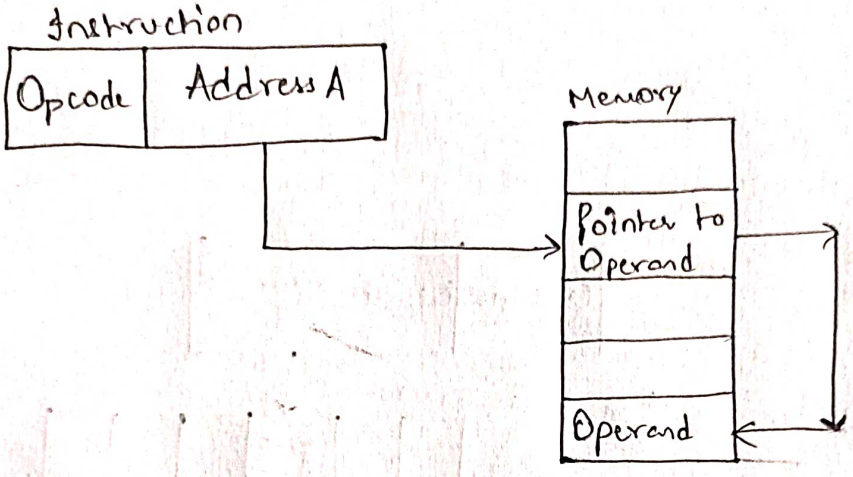
- Single memory reference to access data
- No additional calculations to find the effective address of the operand



for eg:- ADD R1, 4000 - In this the 4000 is effective address of operand.

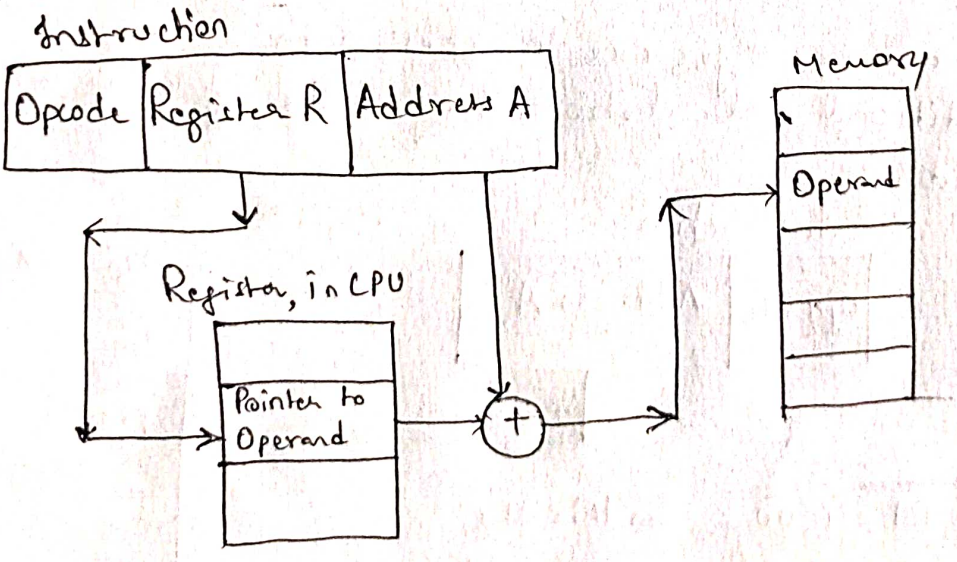
NOTE:- Effective address is the location where operand is present.

6. Indirect Addressing Mode :- In this, the address field of instruction gives the address where the effective address is stored in memory. This slows down the execution, as this includes multiple memory lookups to find the operand.



7. Displacement Addressing Modes :- In this the contents of the indexed register is added to the Address part of the instruction, to obtain the effective address of operand.

EA = A + (R), In this the address field holds two values, A (which is the base value) and R (that holds the displacement), or vice versa.



8. Relative addressing mode :- it is a version of Displacement addressing mode. In this the content of PC (Program Counter) is added to address part of instruction to obtain the effective address.
EA = A + (PC), where EA is effective address and PC is Program counter

The operand is A cells away from the current cell (the one pointed to by PC).

Base Register Addressing Mode:-

it is again a version of displacement addressing mode. This can be defined as EA = A + (R), where A is displacement and R holds pointer to base address.

Stack Addressing Mode:- In this mode, operand is at the top of the stack. for eg: ADD, this instruction will POP two items from the stack, add them, and will then PUSH the result to the top of the stack.

Data transfer and Manipulation :- Most computer instructions can be classified into three categories:-

1. Data transfer instructions
2. Data manipulation instructions
3. Program control instructions

1. Data transfer instructions:- The most common transfers are b/w memory and processor registers, between processor register and I/P or O/P and b/w processor registers themselves.

Typical Data Transfer Instructions:-

Name	Mnemonic
Load	LD
Store	ST
Move	MOV
Exchange	XCH
Input	IN
Output	OUT
Push	PUSH
Pop	POP

2. **Data Manipulation Instructions:-** it can be divided into three types:-

1. Arithmetic instructions.
2. Logical and bit manipulation instructions.
3. Shift instructions.

1. Arithmetic instructions:-

Name	Mnemonic
Increment	INC
Decrement	DEC
Add	ADD
Substract	SUB
Multiply	MUL
divide	DIV
Add with carry	ADDC
Substract with borrow	SUBB
Negative (2's complement)	NEG

2. Logical and Bit manipulation Instructions:-

Name	Mnemonic
Clear	CLR
Complement	COM
AND	AND
OR	OR
Exclusive OR	XOR
clear carry	CLRC

Name	Mnemonic
set Carry	SETC
Complement Carry	COMC
Enable interrupt	EI
Disable interrupt	DI

Data Transfer and Manipulation:- Most computer instructions can be classified into three categories:-

1. Data transfer instructions
2. Data manipulation instructions
3. Program control instructions.

Data transfer instructions cause transfer of data from one location to another without changing the binary information content.

Data manipulation instructions are those that perform arithmetic, logic and shift operations.

Program control instructions provide decision-making capabilities and change the path taken by the program when executed in the computer.

1. **Data transfer instructions:-**

Name	Mnemonic
Load	LD
Store	ST
Move	MOVE
Exchange	XCH
Input	IN
Output	OUT
Push	PUSH
Pop	POP

2. **Data Manipulation instructions:-** Data manipulation instructions Perform operation on Data and Provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types:-

1. Arithmetic instructions
2. Logical and bit manipulation
3. Shift instructions

1. Arithmetic instructions :- The four basic arithmetic operations are addition, Subtraction, multiplication and division. Most computers provide instructions for all four operations.

Typical Arithmetic Instructions :-

Name	Mnemonic	Example
Increment	INC	INC B
Decrement	DEC	DEC B
Add	ADD	ADD B
Subtract	SUB	SUB B
Multiply	MUL	MUL B
divide	DIV	DIV B
Add with Carry	ADDC	ADDC B
Subtract with Borrow	SUBB	SUBB B
Negative (2's complement)	NEG	NEG B

2. Logical and Bit Manipulation Instructions :-

Name	Mnemonic	Example
clear	CLR	
Complement	COM	
AND	AND	
OR	OR	
Exclusive-OR	XOR	
Clear carry	CLRC	
Set Carry	SETC	
Complement Carry	COMC	
Enable Interrupt	EI	
Disable Interrupt	DI	

3. Shift Instructions:-

Name	Mnemonic
Logical shift right	SHR
Logical shift left	SHL
Arithmetic shift right	SHRA
Arithmetic shift left	SHLA
Rotate Right	ROR
Rotate Left	ROL
Rotate right through Carry	RORC
Rotate Left through Carry	ROLC

3. Program Control Instructions:-

Name	Mnemonic
Branch	BR
Jump	JMP
Skip	SKP
Call	CALL
Return	RET
Compare (by subtraction)	CMP
Test (by ANDing)	TST

Subroutine call and Return:-

SP ← SP − 1	Decrement stack Pointer
M[SP] ← PC	Push content of PC onto the stack
PC ← effective address	Transfer control to the subroutine
PC ← M[SP]	Pop stack and transfer to PC
SP ← SP + 1	Increment stack Pointer.

Program Interrupt :- Program interrupt refers to the transfer of Program control from a currently running Program to another service Program as a result of an external or internal generated request.

The collection of all status bit conditions in the CPU is sometimes called a program status word or PSW.

3. Shift Instructions:-

Name	Mnemonic
Logical shift Right	SHR
Logical shift Left	SHL
Arithmetic shift Right	SHRA
Arithmetic shift Left	SHLA
Rotate right	ROR
Rotate Left	ROL
Rotate right through Binary	RORC
Rotate Left through Binary	ROLC

PAGE NO. ...

Types of Interrupts :-

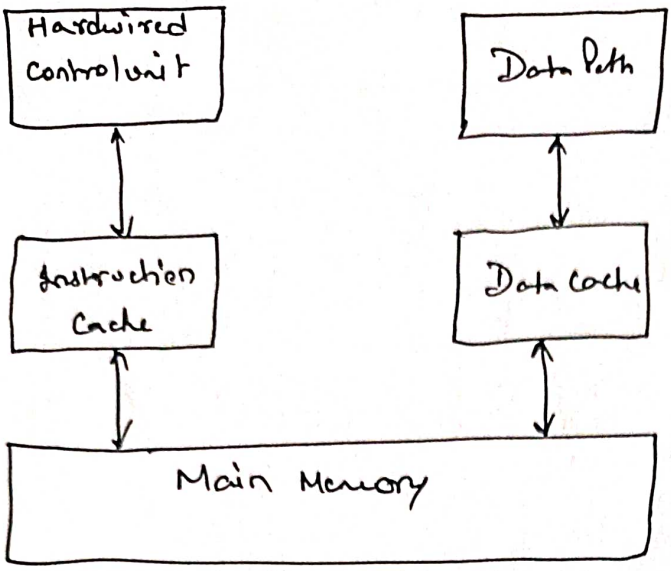
1. External Interrupts
2. Internal Interrupts
3. Software Interrupts

External Interrupts come from I/P-O/P devices, from a timing device, from a circuit monitoring the power supply, or from any other external source.

Internal Interrupts arise from illegal or erroneous use of an instruction or data. Internal interrupts are also called traps. Examples of interrupts caused by internal error conditions are register overflow, attempt to divide by zero, an invalid operation code, stack overflow and protection violation.

RISC (Reduced Instruction set computer) and CISC (complex Instruction set computer) :-

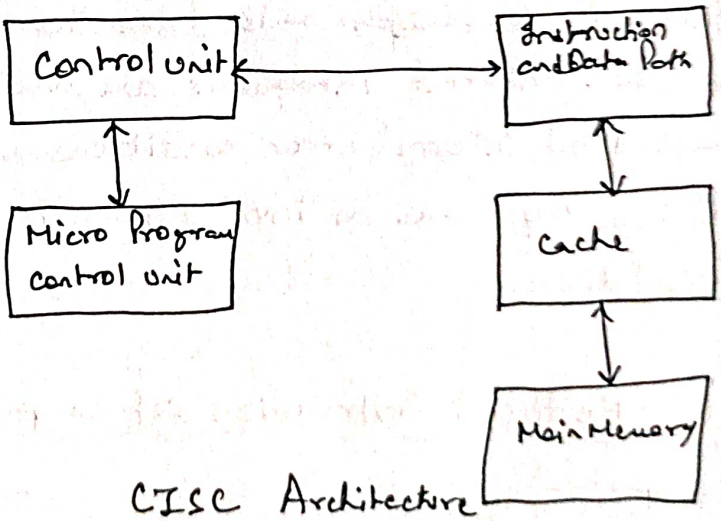
1. **RISC (Reduced Instruction set computer) :-** RISC is a microprocessor architecture with a simple collection and highly customized set of instructions. It is built to minimize the instruction execution time by optimizing or limiting the number of instructions. It means each instruction cycle requires only one clock cycle and each cycle contains three parameters :- fetch, decode, execute.



RISC architecture.

2. CISC (Complex Instruction Set Computer) :- CISC developed by intel.

it had a large collection of complex instructions that range from simple to very complex and specialized in the assembly language level, which takes a long time to execute the instructions. So, CISC approaches reducing the number of instruction on each program and ignoring the number of cycles per instruction.



CISC Architecture

Difference B/w RISC and CISC

RISC	CISC
1. it emphetizes on software to optimize the instruction set.	1. it emphetizes the hardware to optimize the instruction set.
2. it is a hard wired unit of Programming in the RISC processor.	2. Microprogramming unit in CISC Processor.
3. it requires multiple register sets to store the instructions.	3. it requires a single register set to store the instruction.

4. RISC has more transistors on memory registers
5. The execution time of RISC is very short
6. it has fixed format instruction
7. it uses LOAD and STORE that are independent instructions in the register-to-register a Programs interaction.

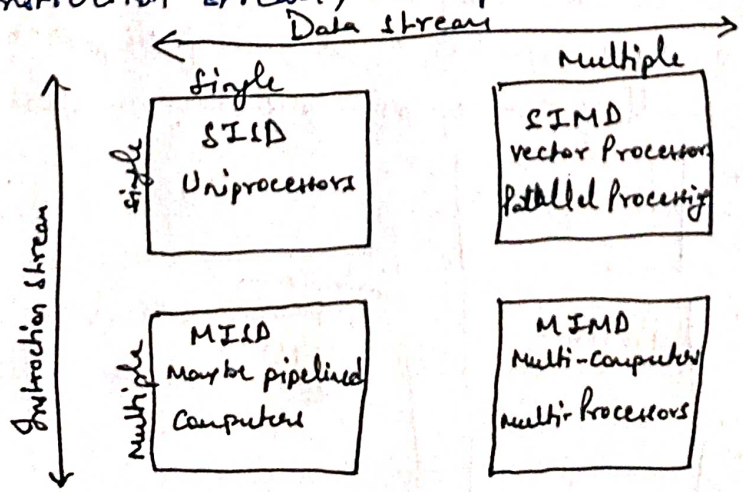
Example:- ARM, PA-RISC, Power Architecture Alpha, AVR, ARC and the SPARC

4. CISC has transistors to store complex instructions.
5. The execution time of CISC is Longer.
6. it has variable format instruction
7. it uses LOAD and STORE instruction in the memory-to-memory interaction of a Program.

Example of CISC:- VAX, Motorola 68000 family System/360, AMD and the intel x86 CPUs.

Flynn's classification:-

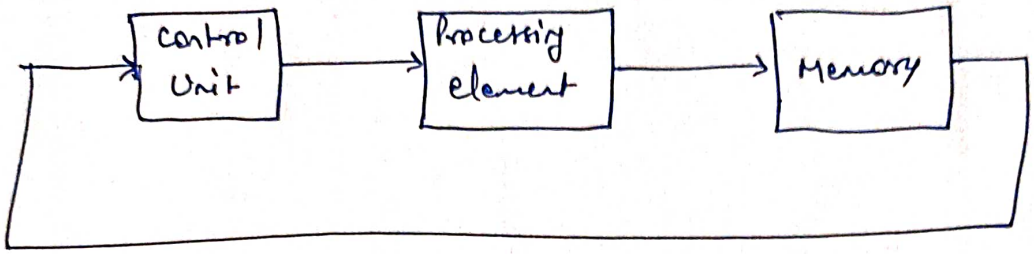
1. Single Instruction stream, Single data stream (SISD)
2. Single Instruction stream, multiple data stream (SIMD)
3. Multiple Instruction stream, Single data stream (MISD)
4. Multiple Instruction stream, Multiple data stream (MIMD).



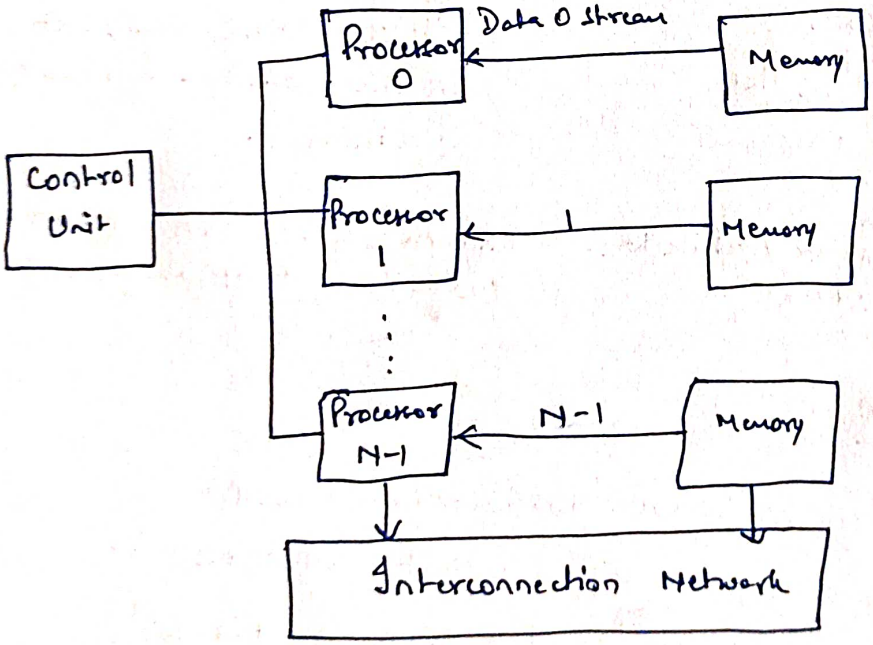
1. SISD (Single Instruction Single Data Stream) :-

Single instruction :- Only one instruction stream is being acted or executed by CPU during one clock cycle.

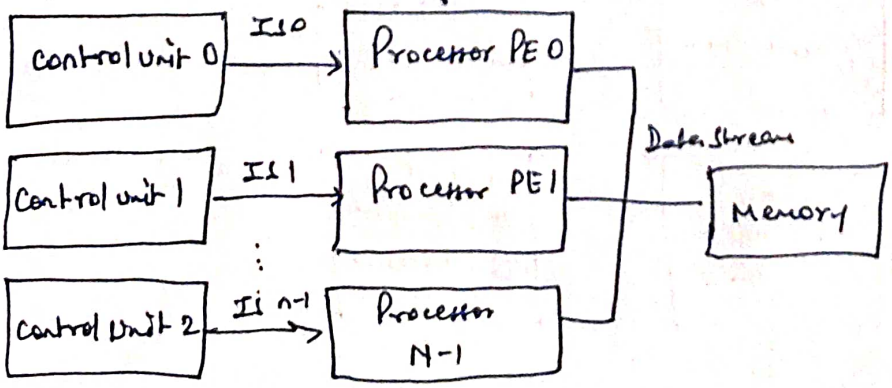
Single data stream :- Only one data stream is used as input during one clock cycle



2. SIMD (Single Instruction Multiple Data stream) :- A SIMD system is a multiprocessor machine, capable of executing the same instruction on all the CPUs but operating on the different data stream.



3. MISD (Multiple Instruction single Data Stream) :- A MISD computing is a ~~micro~~ multiprocessor machine capable of executing different instructions on Processing elements but all of them operating on the same data set.



PAGE NO. .....

4. MIMD (Multiple Instruction Multiple Data) :- A MIMD system is a multiprocessor machine that is capable of executing multiple instructions over multiple data streams. Each processing elements has a separate instruction stream and data stream.