Basics of Computer Organization & Architecture

Computer Architecture :—

parameters that are visible to programmer or that have direct impact on logical execution of a program. Exp⇒ instruction set, size of datatype I/O mechanism, techn for addressing memory.

~~Concept~~ set of rules stating that how computer software & hardware are joined together. & interact to make comp. netw. also determine which tech the comp. is capable.

Include → ① System Design ~~②~~ – Hardware part.

② Instruction Set Architecture – CPU's func. & capabilities CPU's pro. lang. etc. (Software)

③ Microarchitecture → Defines the data processing & Storage Element or data path & how ~~they are~~ should imp. into ISA.

Computer Organization :—

Defines the way the system is structured so that all these catalogue tools can be used properly.

Refers to the operational units & their interconnection. ~~that realize the such~~

Operational Units! - ~~Units~~ a Like Input ~~Unit~~ Unit, Memory Unit Logic Unit

Computer Architech	Comp. Organization
• The way software & hardware are connected.	• Structure & behaviour seen by user.
• Interface btw Sw. & Hw.	• ~~Deals with connection of syst~~ Deals with comp. of connection.
• Help us to understand functionalities of a sys.	• Co. O. tells how exactly all the units in the system are arranged & interconnected.
• while Designing C.S it is done 1st	• on the basis of C.A.
• Deals with high level Design issues	• Deals with low level design issues.
• It involve ISA, Addressing Mode, Data type Cache Optim	• Physical Compo. & circuit Adders, Signal & peripherals

~~Model~~

Computer Model & Unit :— There are 4.

① CPU ② Input Unit ③ Output Unit

④ Memory Unit.

Comp. of Computers :—

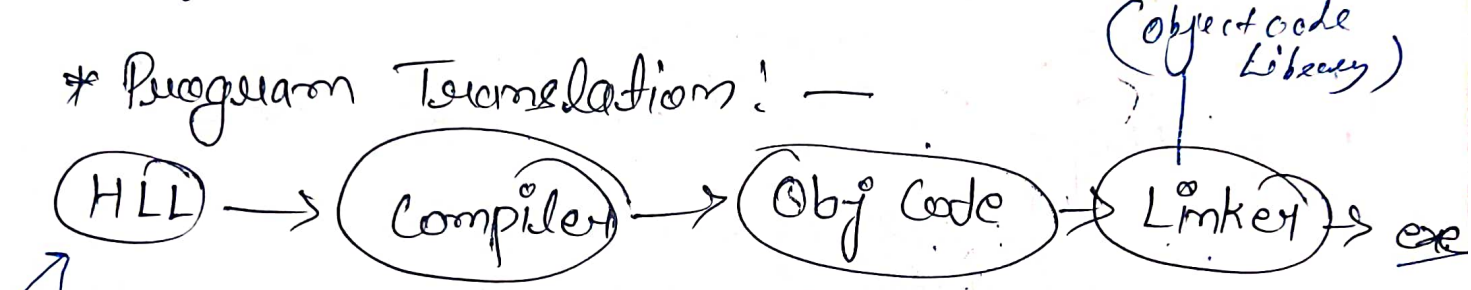
① Hardware, ② CU, MU, ALU, I/O Units

② Software — OS,

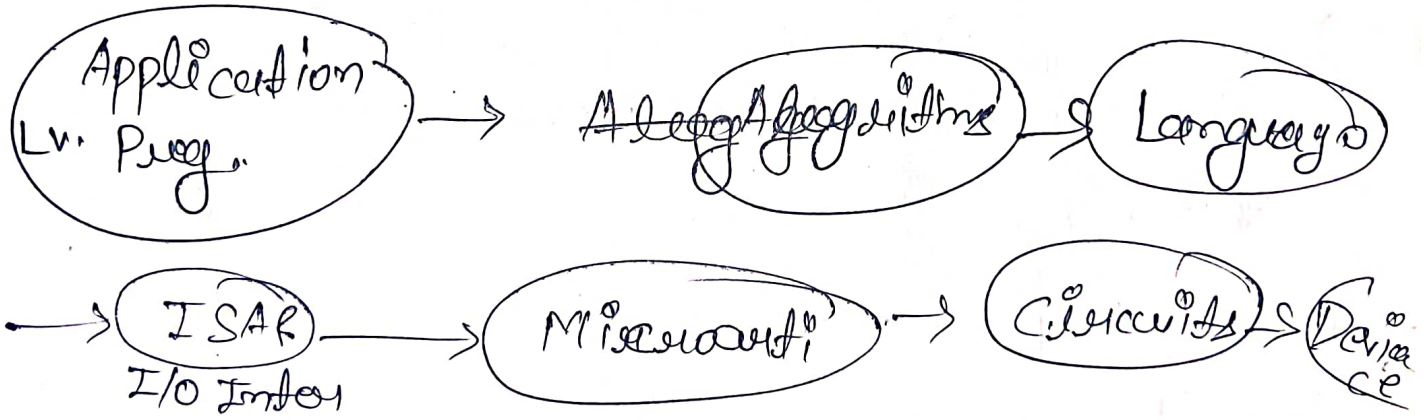
③ Machine Language ① As lang. ③ HLL

④ Sys Soft & ⑤ Application Software

* Program Translation :—

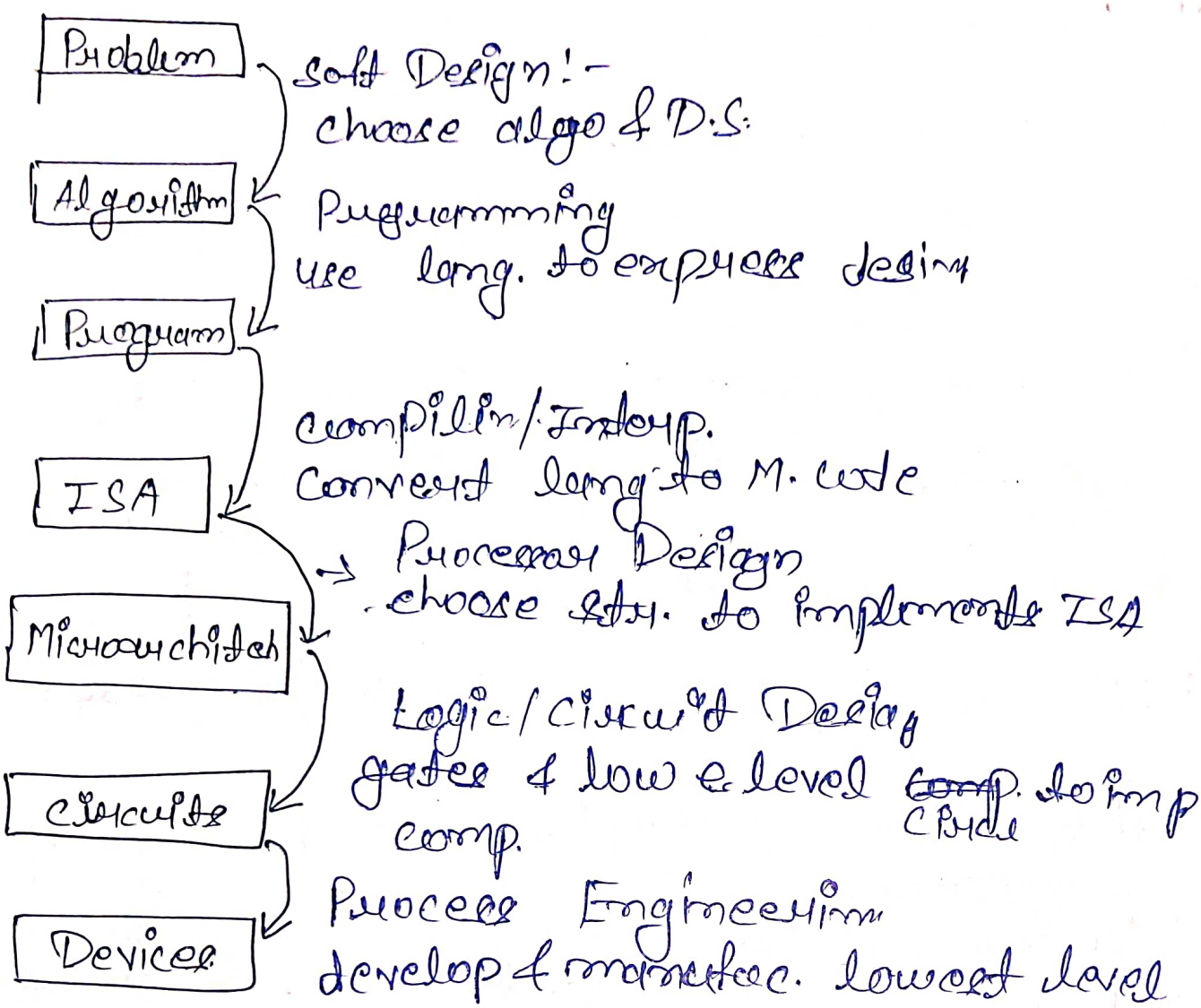


* Layer of Abstraction :—



Transformation btw Layers:—

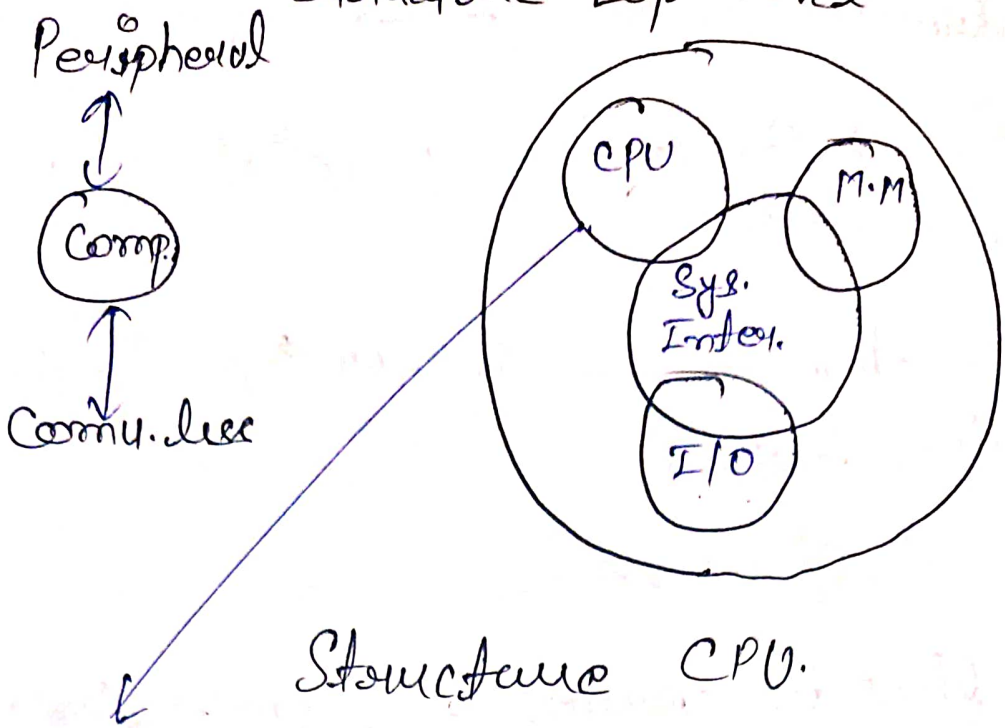
How do we solve a problem? A sys



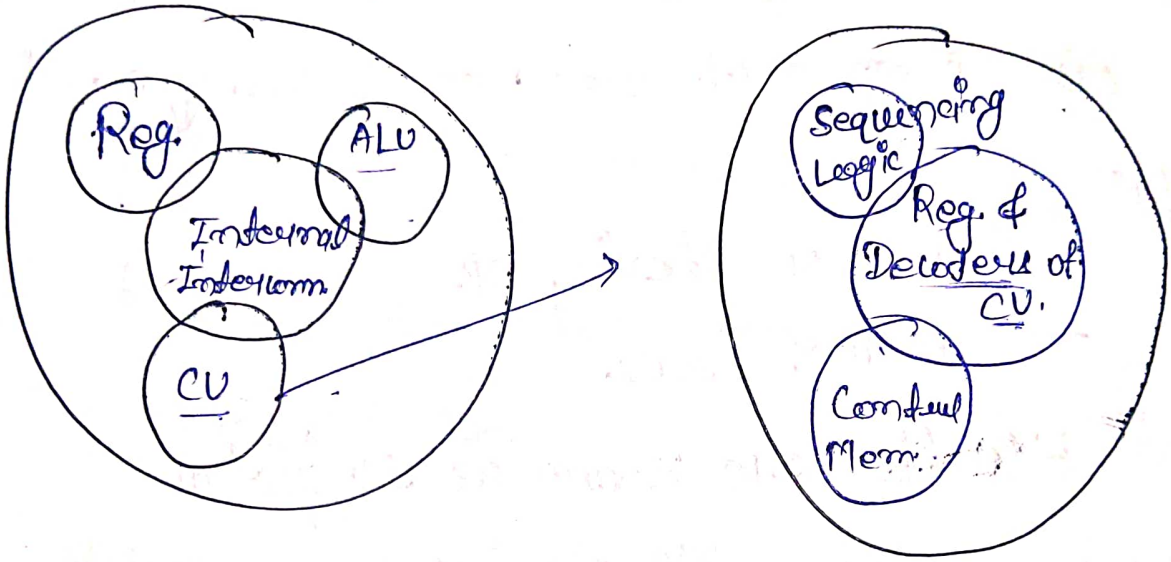
Computer Leve Hierarchy:—

Lv. 6 User — Exe. Prog.
" 5 HLL — Java, C++
" 4 As. lang. — Asc. Code
" 3 Sys sw — O.S. Lib Code
" 2 Machie — ISA.
" 1 control — Microcode of Hardware
" 0 Digital Logic — circuits, Gates.

Structure - Top Level



Structure CPU.



Basic Working Principle of a Computer:—

- For ALU → 0 in that signal → arithmetic operations
1. in that signal → Logical operations

Arithmetic		Logical	
0000	ADD	100	OR
001	SUB.	101	AND
010	MULT.	110	NAND
011	DIV.	111	~~ADD~~ → XOR

Intro. to Comp. Sys :—

(1) CU → Generates appropriate signal at right moment

(2) Decoder → decode ins in CPU & do perform the desired task.

(3) Simp. Mod. of Comp →

3 input lines to decoder & it generates 8 output.

(4) In our simp. mod. we use 3 storage units in CPU

- Two for storing op.
- One for res.

These 3 units are known as registers.

(5) CPU can work with info only av. in Main Mem

(6) To access data from Main Mem we need two special reg. — Memory Data Reg. (MDR) & Memory Address Reg. (MAR).

Main Memory Organizations:—

① Capacity of M. Mem. i.e. = no. of mem locax size of each location.

Exp ⇒ Mem. Module of capacity 16×4 means, there are 16 location & each of 4 bit.

② We need two operand to work with mem.

① Read Operation → Retrive Data & Bring it to CPU Reg.

② Write Operation → CPU Reg → Memory

③. Data BUS:- connection to transfer data b/w CPU & Mem.

④. Address Bus → Signal lines used to identify a mem. location.

⑤ Size of address bus depends upon mem. size for mem. module of capacity 2^n location we need n address lines that is an address bus of size n.

(*) Address Decoder:— Used to decode address present in address Bus.

Eg. consider mem = 16×4

Size of Add. Bus = 4 bit & Data bus = 4bit

The size of add. decoder = 4×16

Eg:—

Add Bus = 0101 & Data Bus = 1100 then & R/W=1 then 1100 will be written at location 5(Add. Bus.)

Eg:— If Add bus = 1011 & R/W = 0 then contents of location 1011 will be placed on Data bus.

Memory Instructions:— See pdf page No. 27&28

Number Sys Repre:—

① Signed-Mg Magnitude form:— for n-bit number

MSB → Sign Rest → Magnitude

Range → $-2^{n-1}-1$ to $+2^{n-1}-1$

② Diminished Radix comp.:—

For Given no. N with n digits & base is r the

$(r-1)$'s comp = $(r^n-1)-N$

③ Radix comp. ⇒ $(r-1)^{s}$ comp $+1$ = r's comp.

r's comp = (r^n-N)

✱ Representation of floating point numbers

① Fixed Fixed-Point Representation:—

② Floating-Point Representation.

③ IEEE standard to floating point ~~represent~~ representation.

1. Fixed Point Representation:—

$-2^{16}-1$ to $+2^{16}-1$

1 sign bit	16 bit before decimal	8 bit after decimal.

41.6875 →

41 → 101001 sign bit = 0.

6875 → 1011

0 | 0000000000101001 | 1011

0 ↓ sign bit — 0000000000101001 ↓ Binary rep. for before decimal part — 1011 ↓ Binary for after decimal.

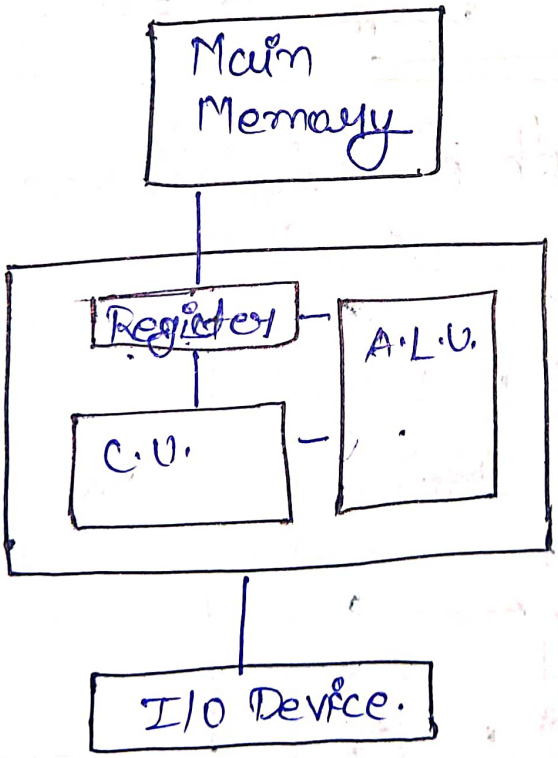
2. Floating Point Representation:—

5236 → 5.236×10^3

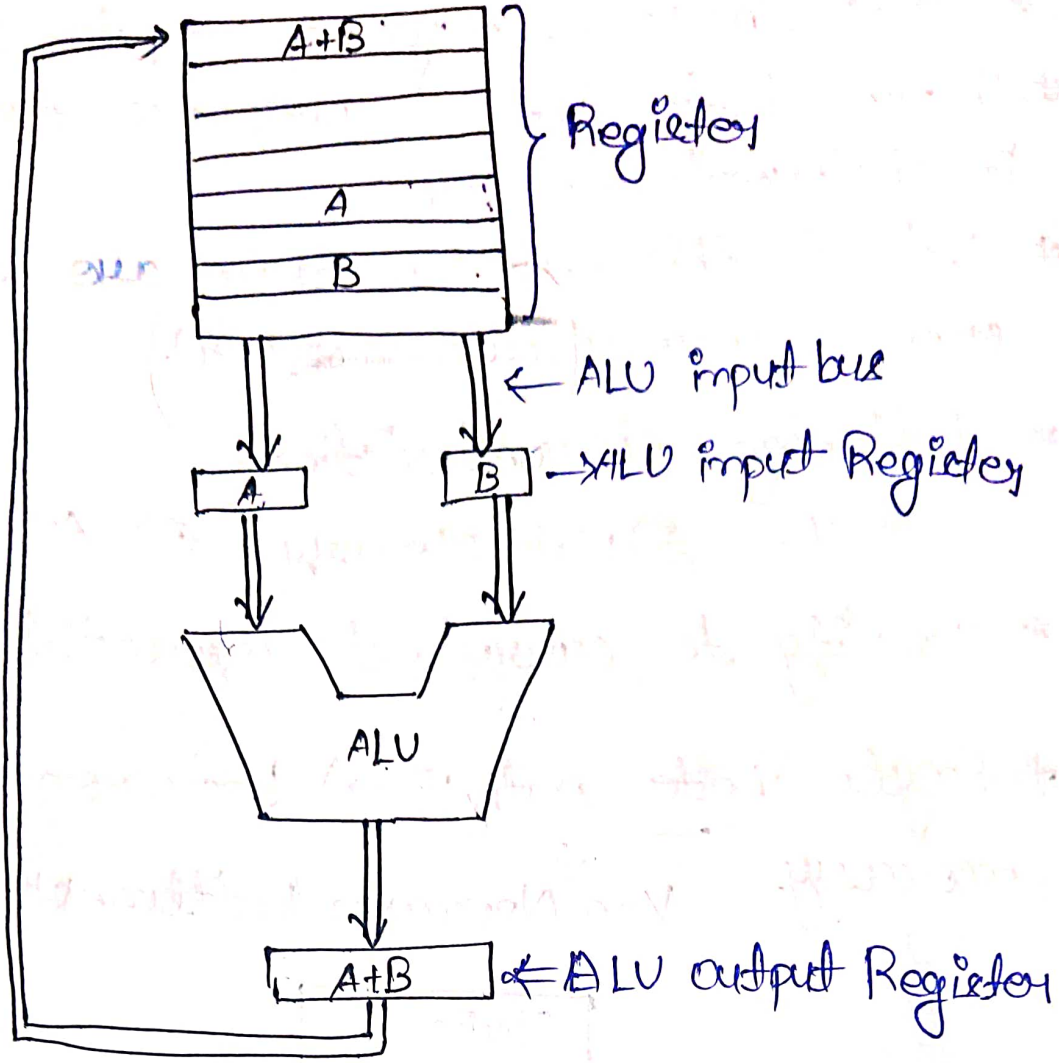
IEEE Standard:—

The von Neumann Model:-

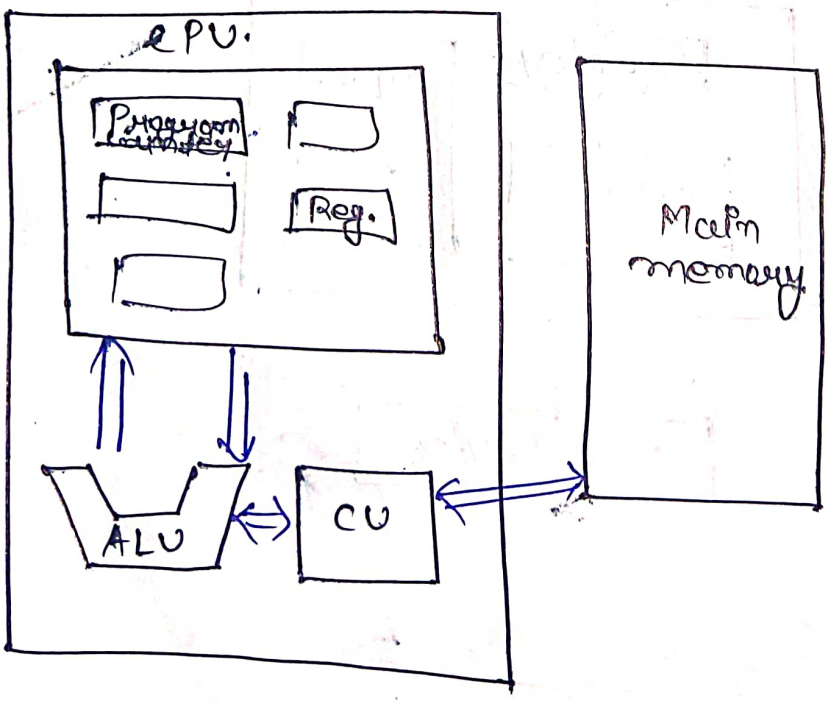
* Gives the stored program computer Model.
* Before that Data & programe ~~are~~ were stored in different memory
* But in this data & program are stored in same memory (main memory).
* Following characteristics -
  ① CPU ② Main Memory & ③ An I/O system.
* Capacity to carry out sequential ins. processing
* Single Data path (Bus) between CPU & main memory. ↓ Von Neumann Bottleneck.



CPU Organization



The data path of a typical von Machine



* Computer employ a fetch-decode-execute cycle to run program -

(1.) C.U. fetch next instruction from memory using Program counter to determine where is the Instruc. is located.

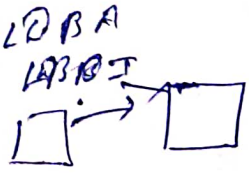
(2.) Then ins decode lang. that ALU understood

(3.) The data operand req. to execute ins. are fetched from CPU & placed into register.

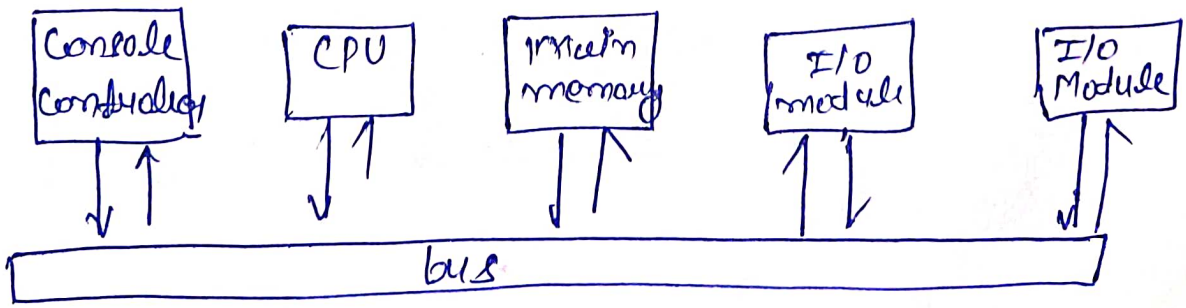
(4.) ALU executes the ins. & place results in register → memory.

* Set of Registers (Storage of CPU) →

→ Memory Buffer Register (MBR)
→ Memory Address " (MAR)
→ Instruction Register (IR)
→ " Buffer Register (IBR)
→ Program Counter (PC)
→ Accumulator (ACC)
→ Multiplier Quotient



Bus Structure:-



Chapter(2) ⇒ Arithmetical & Logical Unit

ALU:-

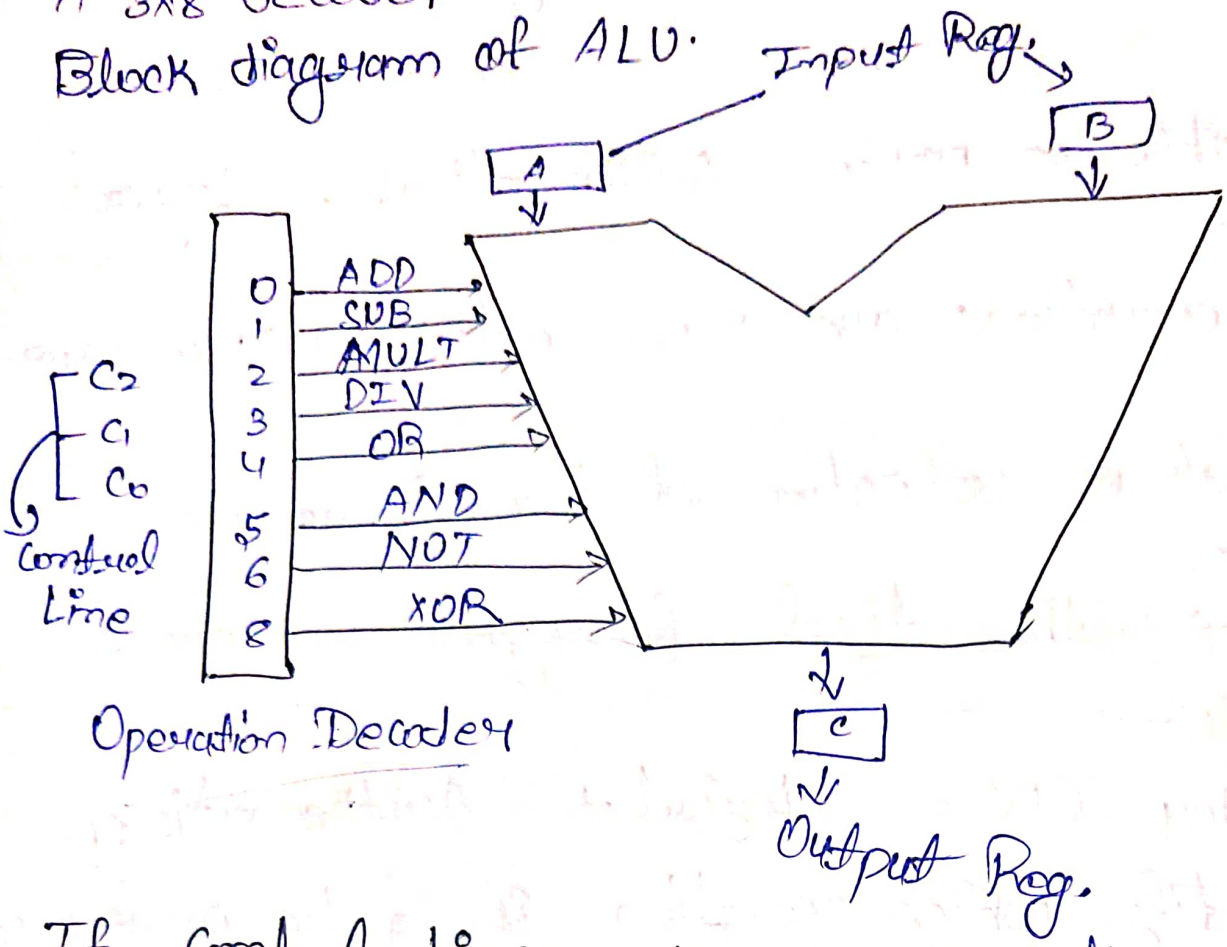
- Responsible to perform Arithmetical & Logical operation.
- Basic operations are implemented in hardware level.
- Consist of collection of two types of operation.
  → Arithmetical & logical Operations.

Example:-
Consider CPU = 4 Logical + 4 Arithmetic op.
To identify which operation is to be performe
we need 3 ~~contol~~ control lines.
One for deciding Logical or Arithmatical op
Second for deciding which Arithmetical op.
Third for " " Logical op.

C_1	C_0	Arith. $C_2=0$	Logical $C_2=1$
0	0	ADD	OR
0	1	SUB	AND
1	0	MULT	NOT
1	1	DIVI	EX-OR.

A 3x8 decoder is used to decope instruction.

Block diagram of ALU.



Operation Decoder

If Control Lines are 0 0 0 (C_2 C_1 C_0) then decoder enable the ADD operation.

* Microoperations:-

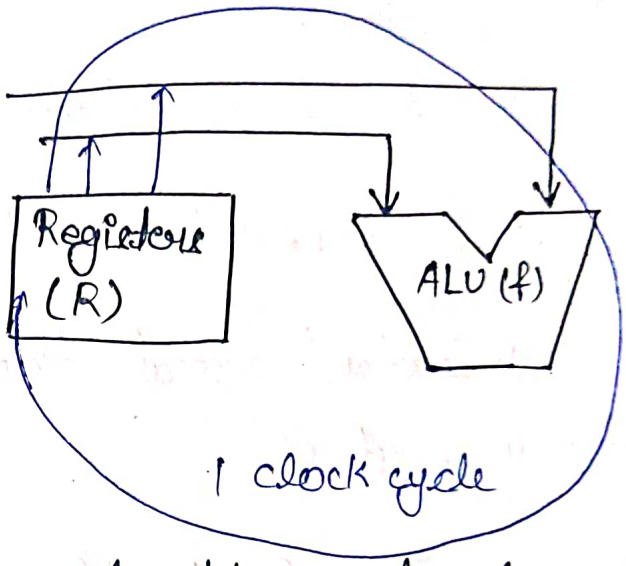
→ The operations on the data in register are called microoperations.

→ The func. built in into register are example of microoperations.

→ Shift, Load, Clear, Increment

→ An elementary operation (one of microoperation) is performed in one clock pulse.

→ These op. are performed on the data stored in register.



$R \leftarrow f(R, R)$

It means perform some function on R & store it into R.

f: shift, load, clear, increment, add, subtract, complement, and, or, xor

* Computer Organization :—

Comp Org. is the knowing, what the func. ~~com~~ component of a comp. are, how they work & how their performance is measured & optimized.

- Comp. Org. refers to the level of abstraction above the digital logic level, but below the O.S level.

It contains →

→ Set of registers & their func.

→ Microoperation set

→ Control Signal that initiate the seq. of microoperations

Register Transfer Level :—

~~A synchronous digital circuit model that Its a design abstraction that shell tell about computer at a level~~

Its a design abstraction that describe that how the data flows between registers & logical operations are performed on these signals.

→ It is used to create high level representation of a circuit.
→ Based on it low-level representation & ultimately actual wiring can be derived.

Synchronous Circuit

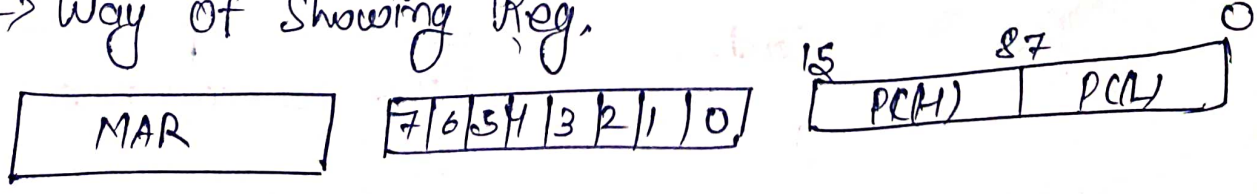
Registers (Sequential Logic) (D-flip flop)

Combinational Logic (perform all of logical op.)

*Register Transfer Language:—

→ Instead of words, specifying a digital System in some specific notations is called RT Lang.
→ This is used to describe microoperations
→ Symbolic Language
→ Used to describe internal C.O. of digital comp
→ Used for design process of digital System
→ Regis. are designated by Capital Letter follw. by no.
A, R13, IR.

MAR → Memory Address Register.
PC → Program Counter
IR → Instruction Register
→ Way of Showing Reg.



* Register Transfer :-

→ Copying content of one reg. to another reg.

→ Happens in one clock pulse

R2 ← R1

Load (copy)

→ non-destructive. content of R1 is not removed after load

* Control Function :-

(a) Ac. take place if a certain condi. is true.

(b) Similar to if statements in prog. lang.

(c) In digital ~~Signal~~ sys. it is done via control signal called control func.

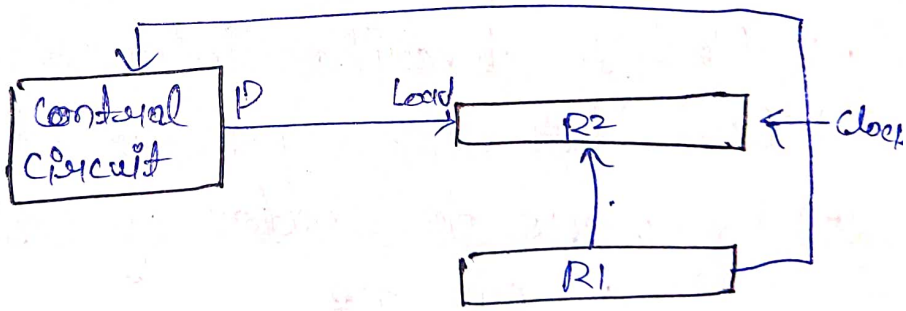
→ if signal is 1 ac. take place.

Eg:-

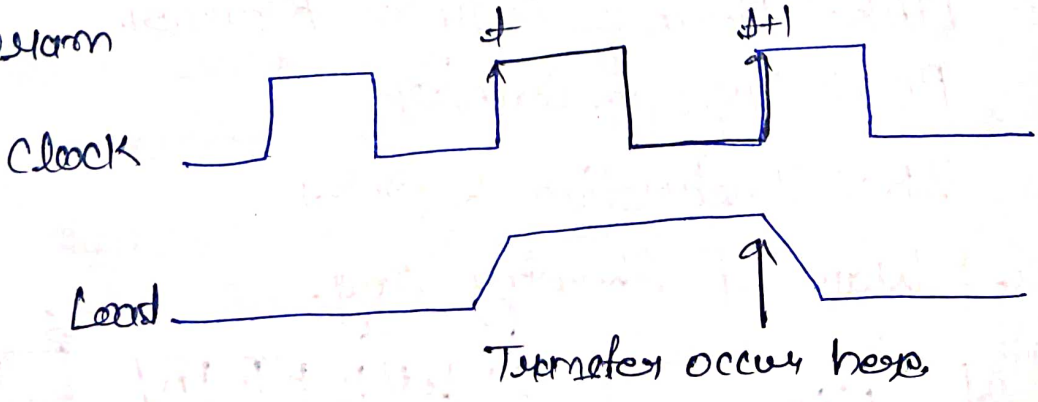
P: R2 ← R1 ⇒ If P is true then Transfer R1 to R2.

Hardware Implementation :-

Block Diagram



Timing Diagram



* Simultaneous Operations :-

P: R3 ← R5, MAR ← IR

If (P = 1) then.

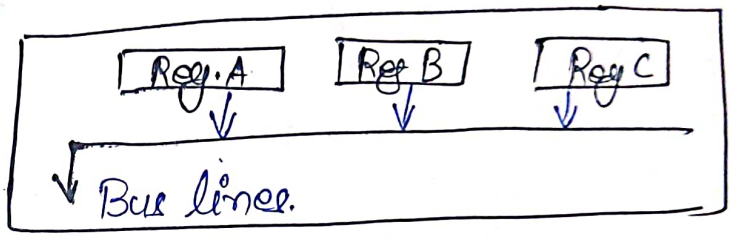
* The Bus & Bus Transfer :-

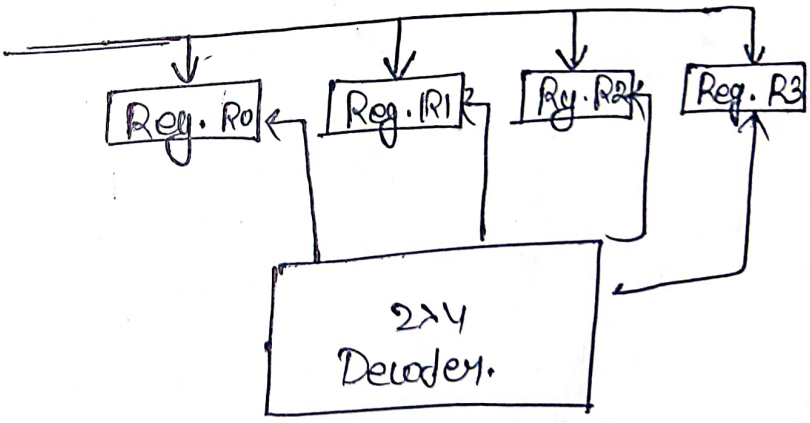
→ It is impractical to connect n registers with each other by n(n-1) connections. So bus is introduced.

→ It's a path or a group of wires over which the info. is transfered.

→ Uses control circuit to select source & destination.

Registers to Bus.





CLASSMATE
Date :
Page :

1

Chapter-6 Processor Organisation

Four important function of CPU

1. Fetch → Main memory → CPU: Instruction
2. Decode → Decode Instruction in Machine Langu.
3. Execute → Run Machine code
4. Right back → Store back the res in mem.

The component of the CPU help it to perform these four fty.

1. ALU - Arithmatic & logical perform / There can be more than one ALU
2. CU = { its a electronic circuit that uses electrical signal to instruct the whole computer. It control & coordinate computer component.
3. Register Set { It receive, decode & store res & manage execution of data. It manage flow of info. through processor

Temporary storage that works along with CPU. very fast memory.

Specific task Register
→ Program Counter, flag, stack.
Fetch next Inst.

* Computer Register Set (Register Organisation) :-
Very fast memory space, works with CPU to store result of execution & the data that is needed for execution of any instruction.

Types of Register
1. General purpose Register: Used for multiple purpose & assigned to a variety of function.

2. Special purpose Registers :- Restricted to specific fn
ex: stack pointer, stack limit, program counter, processor status, flags, Index Registers.

3. Address Register :- Dedicated to addressing mode. It holds the addresses of the memory to perform certain tasks.

* Memory Access Register : MDR, & MAR
MDR - Memory Data Register
MAR - Memory Address Register

They mutually works together to perform read & write operation.
These are not directly accessible to developers

Follow instruction to write operation
1. Data → MDR by CPU
2. Address → MAR by CPU.
3. A write signal is generated by CPU
4. Data → from MDR goes to the location specified by MAR.

Read Operation :-
1. Address → MAR by CPU
2. Read Signal is generated by CPU
3. Data at MAR → loaded MDR

* Instruction fetching Register :- IR, PC.

IR (Instruction Register) :- That holds the Instructions

PC (Program Counter) :-

PC — hold Address of next Instruction to be fetched.

IR — hold the instruction fetched by PC. After that pc updated to next Instruction.

* Condition Register :- Used to maintain status information. ~~It indicate~~ Based on this a Instruction is executed. It also Control Some operation.

Exp: Carry Flag (CF), Parity Flag (PF) Overflow flag (OF), Sign flag (SF),

* Data Path :-

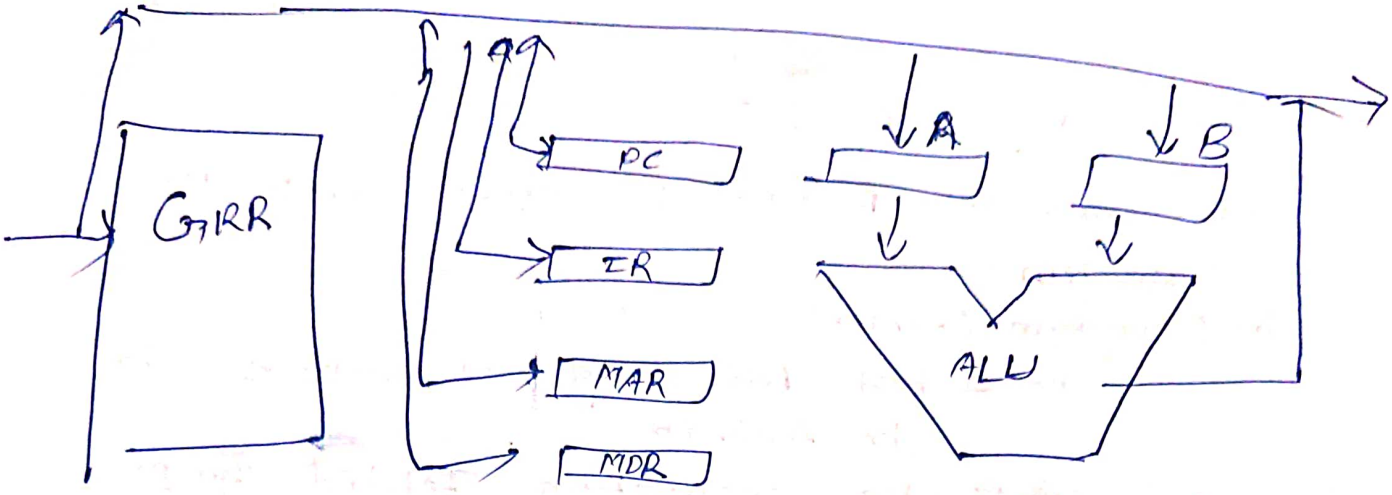
* One bus Organisation :- 1. only one bus used to move outgoing & Incoming data.

2. It can handle only single data moments at one clock cycle.

3. Two operand operation need two cycles to fetch the operand.

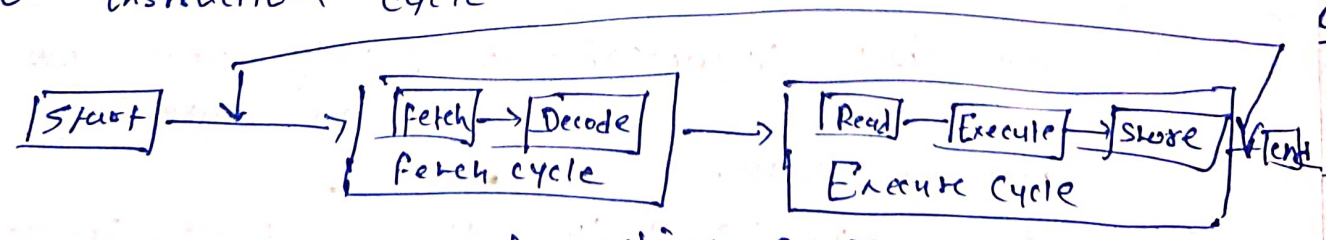
4. Cheapest & Simple but limit the ~~data~~ amount of data in one cycle 5. Slow down the overall performance.

* ISA :- ISA related to programming including native data types, instructions, Registers, Addressing mode, handling & external input/output.



* Two bus Organisation. 1. fast performance than one bus organisation.
2. In this gpR is directly connected to both bus.
3. Two two different data can be transfer to ALU in same clock cycle.
4. It uses two bus for input & one bus for output.
5. One bus can input & one bus can output at same time.

* Three bus organisation:- 1. faster than two bus.
2. Two bus use ~~by~~ for source & one bus use for destination.
3. CPU Instruction cycle



* Instruction cycle.

example:- Add R1, R2, R3. (R3 = R1 + R2)
1. Registers R3, R1, R2 are extracted from the IR
2. ~~Two cont~~ $R_1, R_2 \rightarrow$ ALU for addition
3. ALU → output → R3.

* ISA :- ISA related to programming including native data types, Instructions, Register, Addressing mode, memory architecture, exception handling & external input/output.

* Elements of Instruction :- Each Instruction must convey the information required by processors for execution.

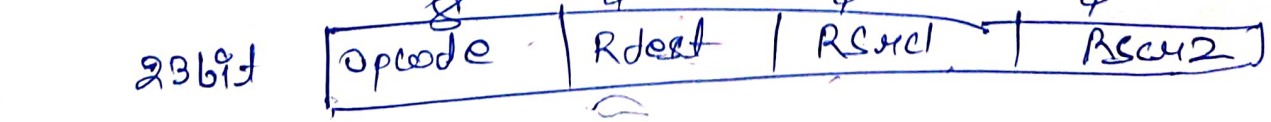
1. Operation code :- Specified the operation to be performed
2. Source operand reference : The op involve one or more operands for input.
3. Result operand Reference :- Ne
4. Next Instruction Reference

* Op code :-

ADD	Add
SUB	Substract
MUL	Multiply
DIV	Divide

* Number of Addresses :-
  * Addressing Instruction :-
    3 Add. Instr.
    2 Add. Instr
    1 —
    0 —

3 Addr. Instruction :- 23 bit Calculation



ADD DEST, SRC1, SRC2 M(DEST) = [SRC1]+[SRC2]

CLASSMATE
Date :
Page :

A = B+C*D-E+F+A

Instr.	
MUL	T, C, D
ADD	T, T, B
SUB	T, T, E
ADD	T, T, F
ADD	A, T, A

Comments

T = C*D

T = B+C*D

T = B+C*D-E

T = B+C*D-E+F

A = B+C*D-E+F+A