UNIT 2

Instructions

One binary bit pattern with specific meaning to microprocessor

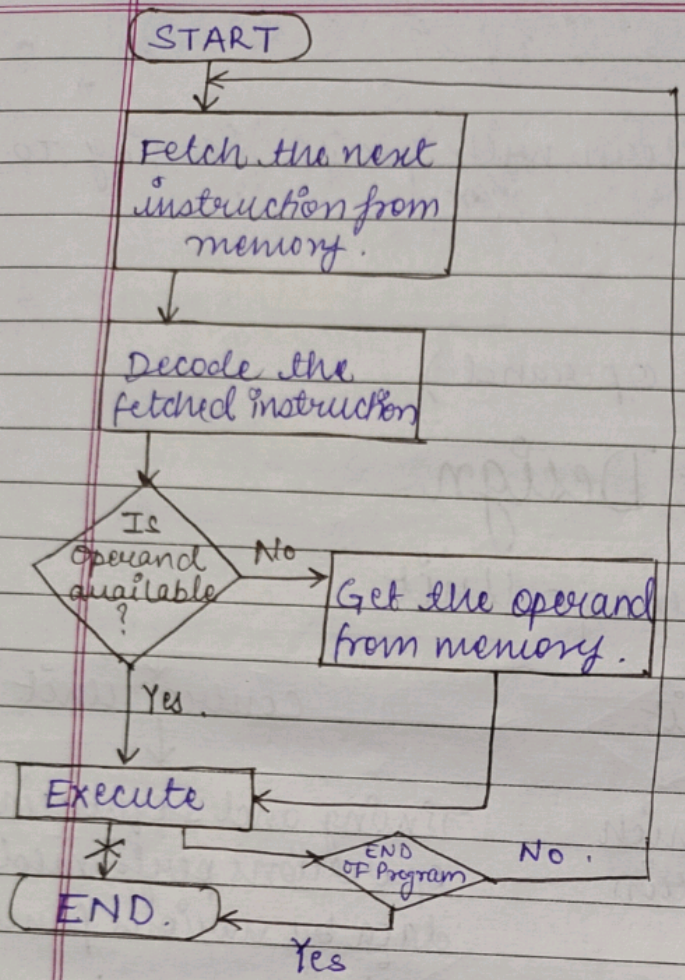
Instruction size

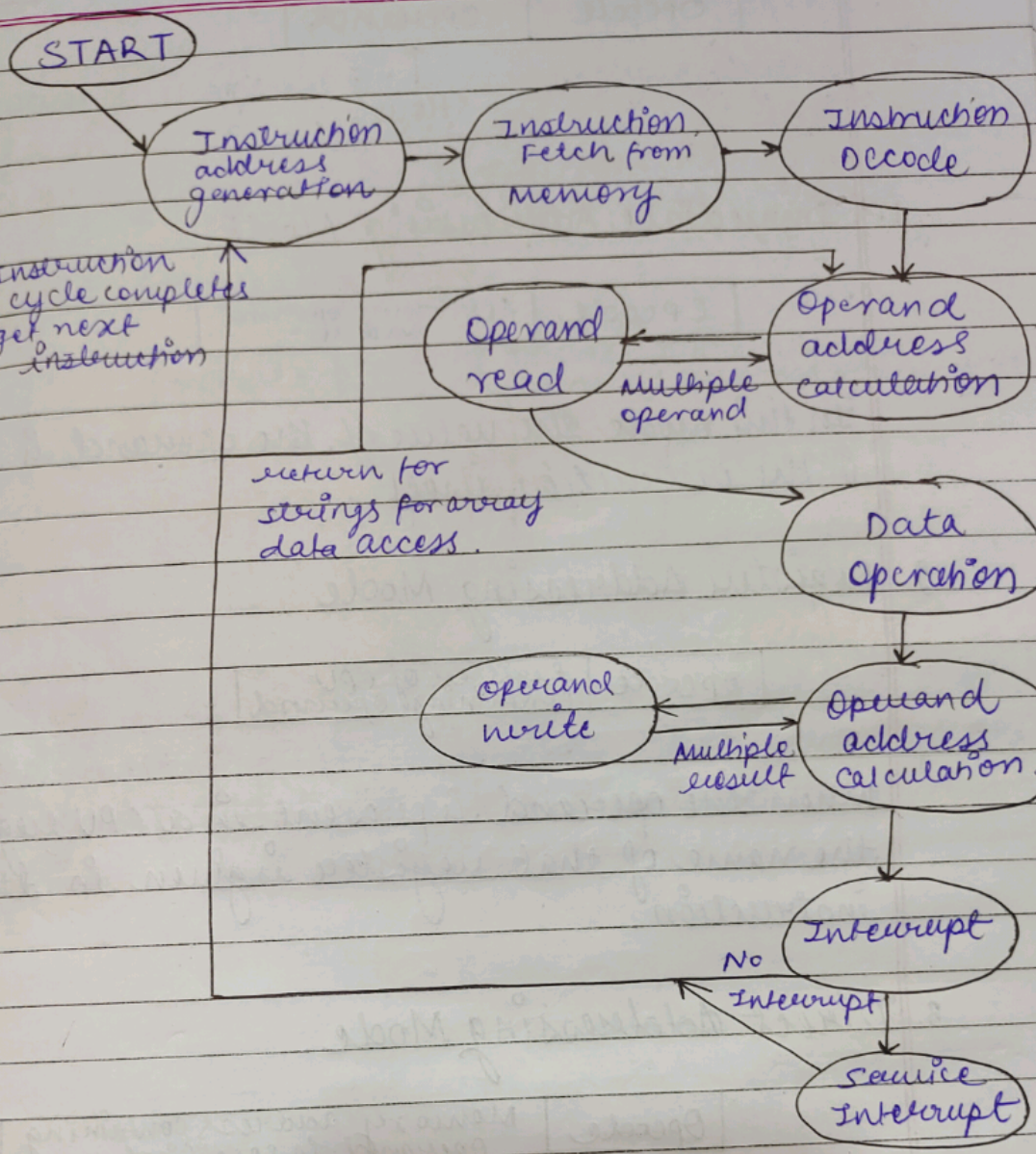
sizeof (opcode + operand).

Control Unit Design.

Processor Unit

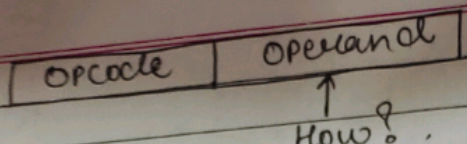
- Data Processing Unit
  - Functional units which can perform operation on Data.
    - * ALU
    - * Registers
    - * Accumulator
    - * Flags.
- Control Unit
  - Timing and sequencing of operations performed on data by various functional units
  - It decides which and when micro-operation is to be performed.


START
Fetch the next instruction from memory.
Decode the fetched instruction
Is Operand available ?
No
Get the operand from memory.
Yes.
Execute
END OF Program
No.
END.
Yes

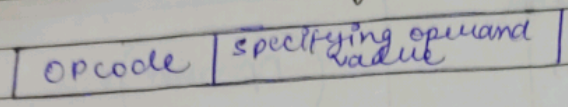


Addressing Modes.

→ The various ways of specifying operand in an instruction.

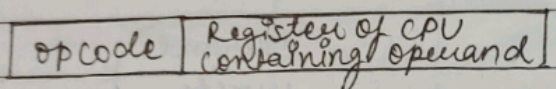


1. Immediate Addressing Mode.



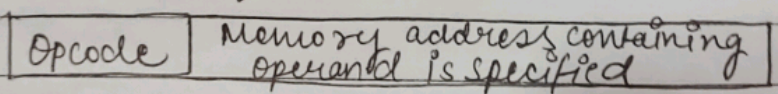
In this mode the value of the operand is specified in the instruction itself.

2. Register Addressing Mode



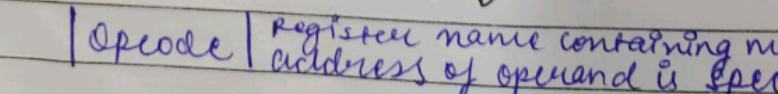
When the operand is present in a CPU register then the name of that register is given in the instruction.

3. Direct Addressing Mode.



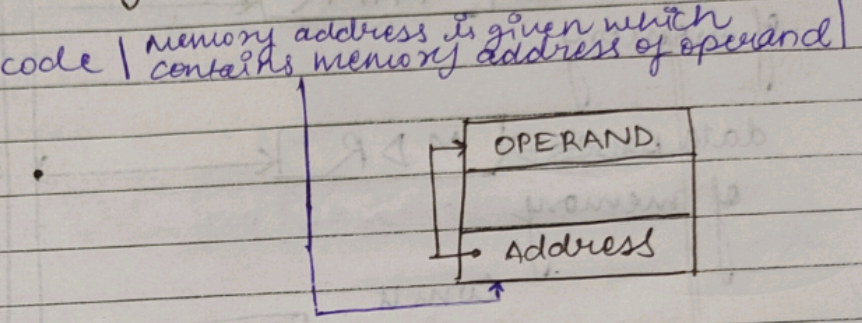
operand is present in some memory location whose address is directly specified in the instruction.

4. Register Indirect Addressing Mode.



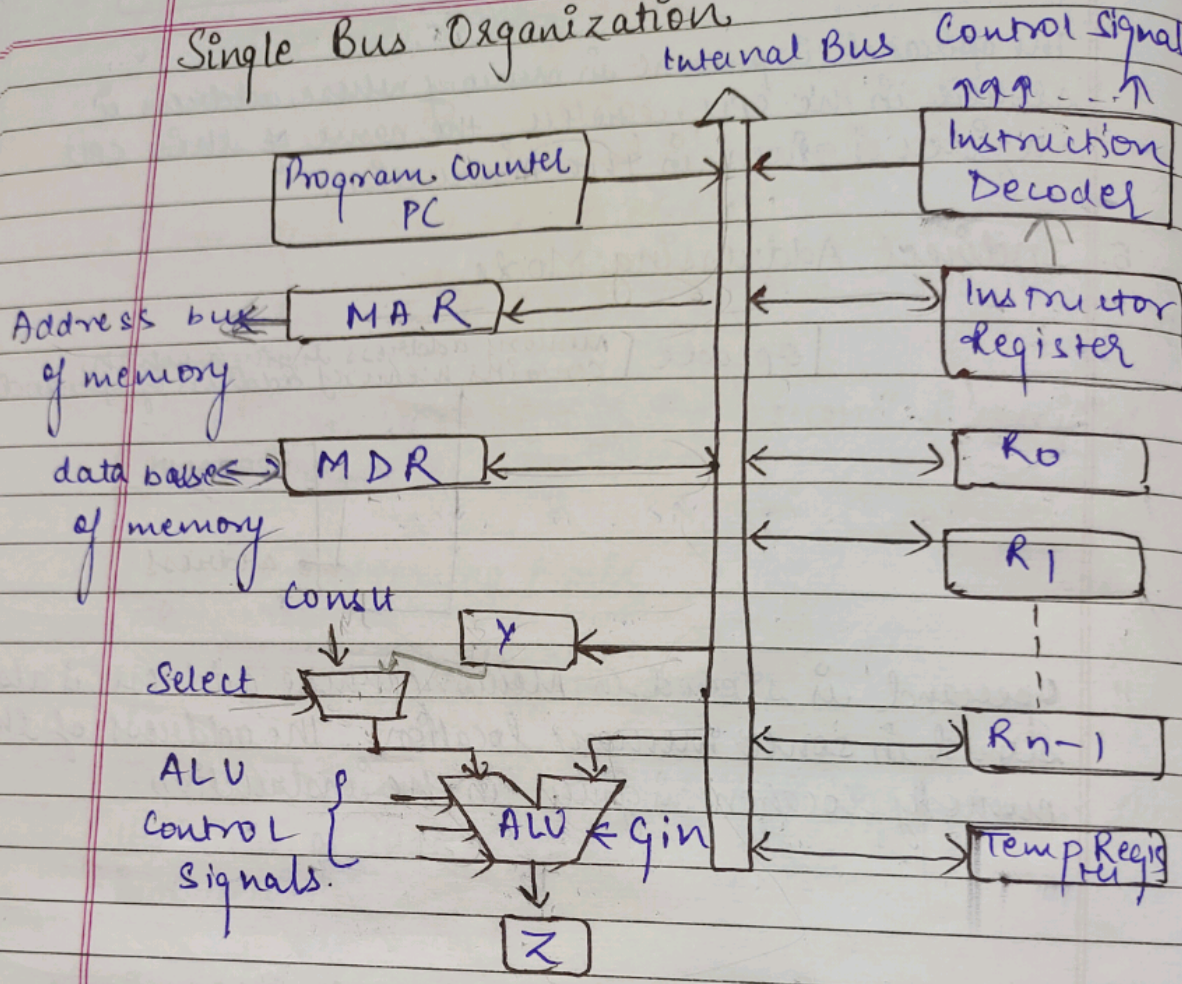
The operand is present in memory whose address is stored in the CPU register, the name of this CPU register is given in the instruction.

5. Indirect Addressing Mode.

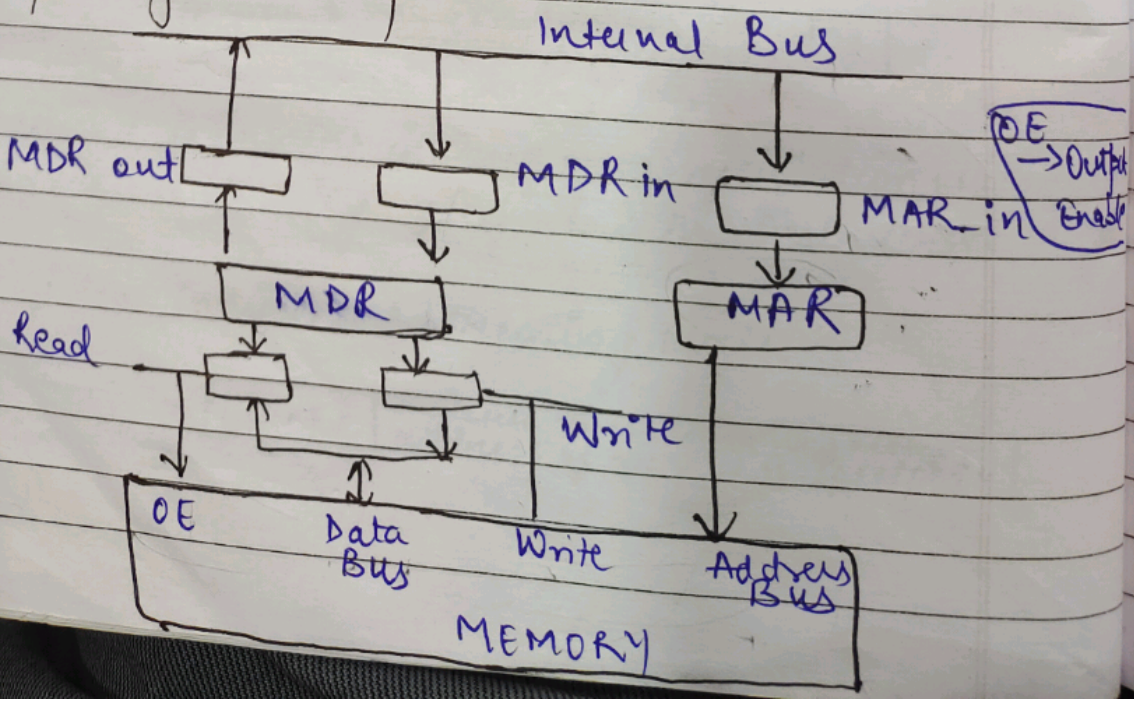


Operand is stored in Memory whose address is also stored in some memory location. The address of this memory location is given in the instruction.

Single Bus Organization



Memory Read Operation



for Operation MOVE $R_1, [R_2]$ R_2 contains address of memory location

Pseudo code.
- MAR ← R_2
- Read, Memory
- Wait for MFC
- operand → MDR
- MDR → R_1

Signals:

① R_2_out, MAR_in, Read

② Wait for MFC

③ MDR_out, RI_in

Timing Diagram:

Que MOVE $[R_2], R_1$

$MAR \leftarrow R_1$
$MDR \leftarrow R_2$
Write
Wait for MFC

① R_1_out, MAR_in
② R_2_out, MDR_in, write
③ WMFC

- Complete Instruction for ADD $R_1, [R_2]$

(PC_out, MAR_in, Read, Select C, Add, Z_in)

(Z_out, PC_in), WMFC.

(MDR_out, IR_in) Decode

(R_2_out, MAR_in, Read

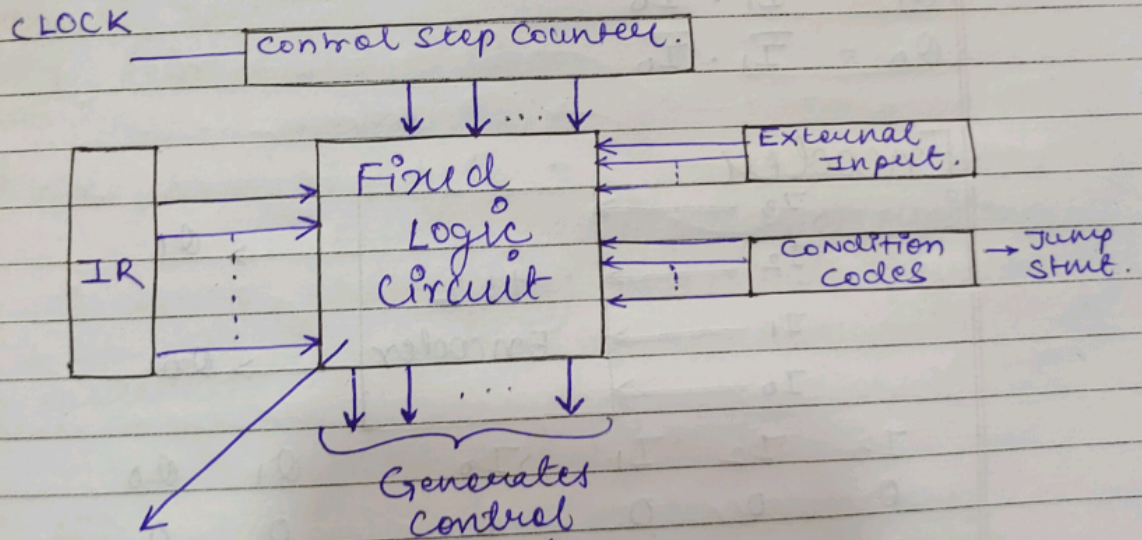
R_1_out, Y_in, WMFC

MDR_out, SelectY, Add, Z_in.

(Z_out, RI_in, END) Execute

Control Unit Design.

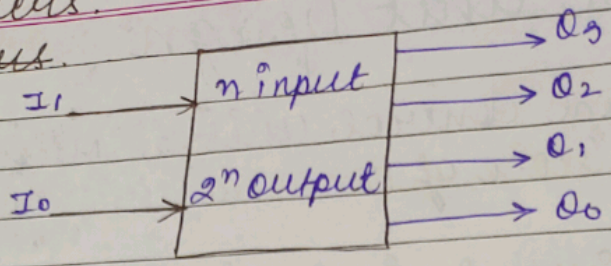
- Hardwired control unit (complete role of Hardware)
  - It uses fixed logic circuit to interpret instructions and generate control signals.
- Micro-programmed control unit
  - control signal selection and sequencing information is stored in a memory called control memory. Hence the control signal to be activated at any time are specified by micro-instruction.



Combinational circuits
- Encoders
- Decoders

Decoders.

~~Encoders.~~



I_1	I_0	O_3	O_2	O_1	O_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

$O_3 = I_1 \cdot I_0$

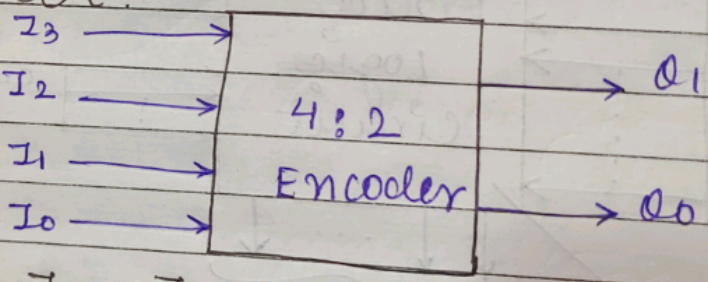
$O_2 = I_1 \cdot \overline{I_0}$

$O_1 = \overline{I_1} \cdot I_0$

$O_0 = \overline{I_1} \cdot \overline{I_0}$

Made up using AND operations

Encoders.



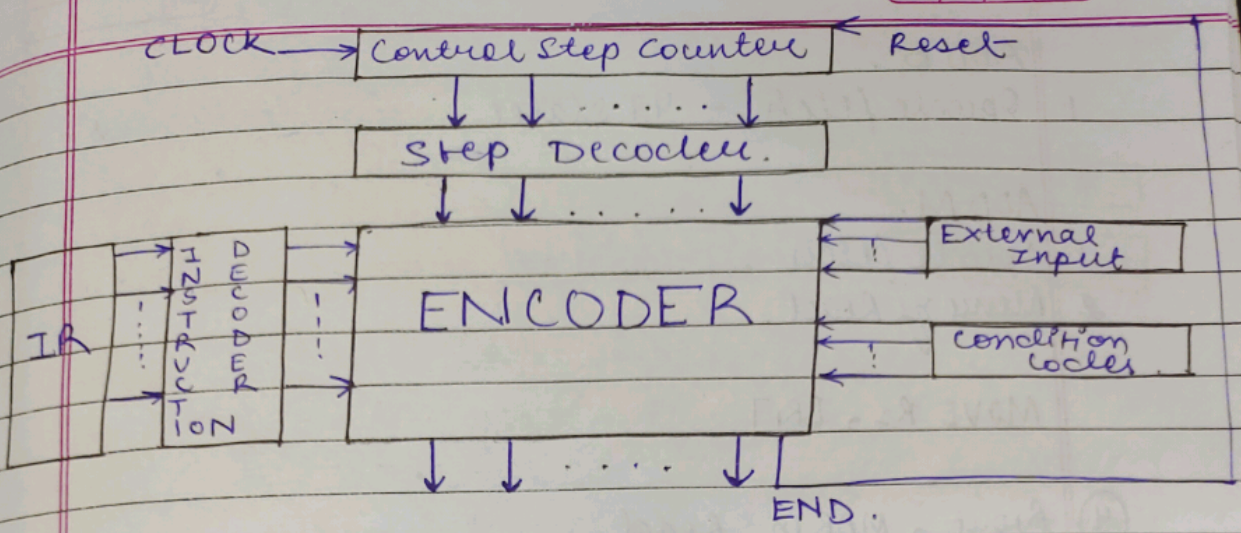
I_3	I_2	I_1	I_0
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

O_1	O_0
0	0
0	1
1	0
1	1

$O_1 = I_2 + I_3$

$O_0 = I_1 + I_3$.

Made up using OR operations



Block Diagram of Hardwired Control Unit.

1. PCout, MARin, Select C, Add, Zin, Read
2. WMFC, Zout, PCin
3. MDRout, IRin
4. R1out, R2in.

MOVE R2, R1 → One byte
operand was in processor register.
No memory Read/Write operation is required for execution.

micro-operation.

INSTRUCTION CYCLE

Machine cycle | Machine cycle | Machine cycle.

Tstates | Tstates | Tstates

Machine cycle →
- Opcode fetch
- Memory Read
- Memory Write
- I/O Read
- I/O Write.

ADD B.

1. Opcode fetch - 4T states

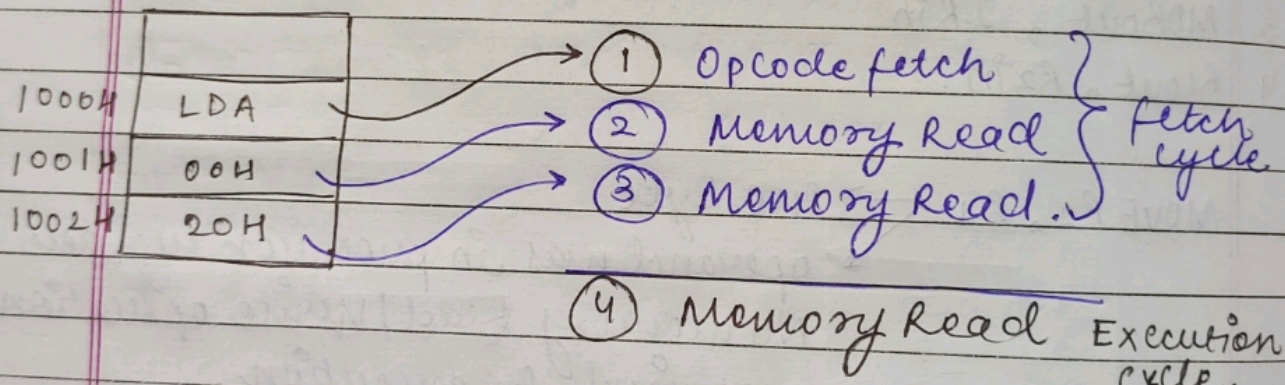
ADD M.

1. Opcode fetch
2. Memory Read.

MOVE R_2, [R_1]

(4) R_1out, MARin, Read
(5) WMFC
(6) MDRout, R_2in.

LDA 2000H.



Size of fetch cycle depends on the size of instruction.

PCout depends on the timing / clock cycle.

PCout depends on T_1 when T_1 is ON PCout is ON
MARin depends on PCout + depends on IR.

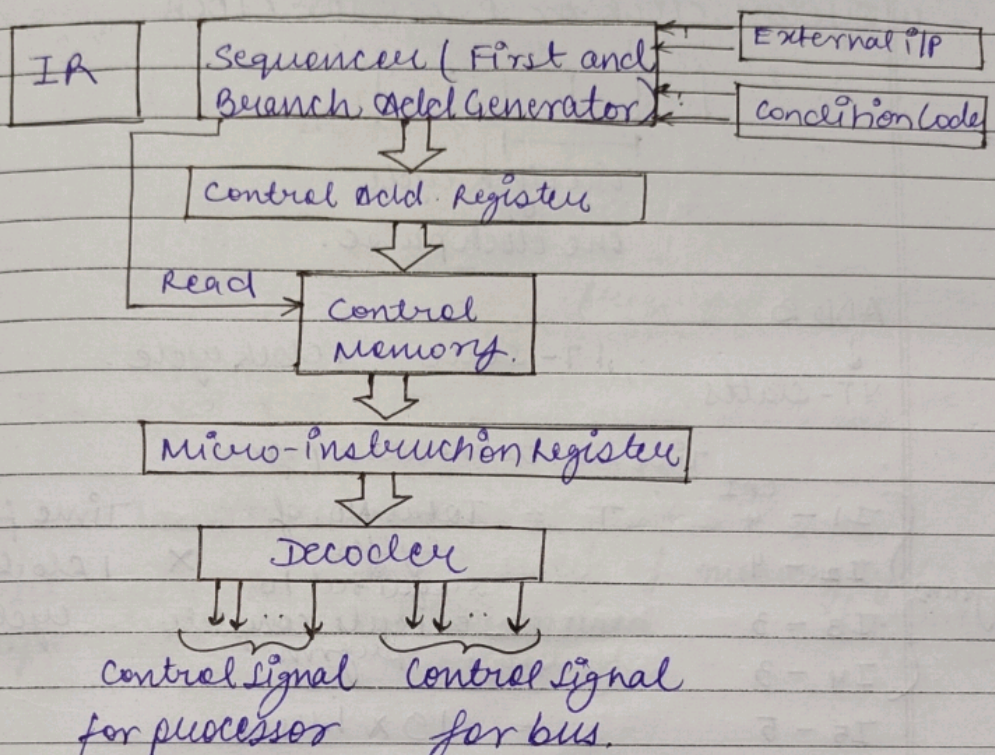
Fastest speed → Hardwired control unit. → everything is used using logic design circuit

Disadvantage:
complexity of the circuit.
size becomes large.

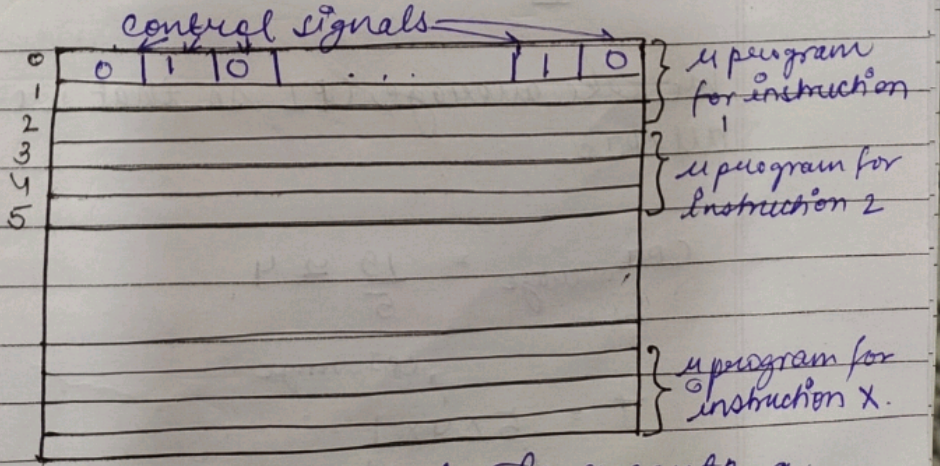
No programming req.
Execution of instr. → which control signals need to be set.

μprogram

micro instruction



Sequencer generates the address of micropeogram's in the control memory. Also known as micro peogram counter.



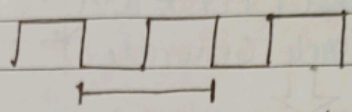
info of various control signals to execute a particular cycle of a instruction cycle.

Memory Read/Write 3 - T states

PAGE No.
DATE

Performance Assessment Of Computer

1. System CLOCK or Processor CLOCK.



one clock cycle
or
one clock pulse.

ADD B
↓
4T-states

1 T-state = 1 clock cycle.

Time req. to execute a prgrm

Program:
- I1 - 4 (CPI)
- I2 - 4
- I3 - 3
- I4 - 3
- I5 - 5

T = Total No. of cycles required to execute complete prgrm × Time for 1 Clock cycle.

= 19 × 1 μsec

= 19 μsec.

f = 1 MHz

T = 1 μsec.

CPI = 19

We take average CPI so that we don't have to add all CPIs.

$CPI_{average} = \frac{19}{5} \simeq 4.$

$T = 5 \times 4 \times 1$

5 → No. of instruction
4 → CPI average
1 → Cycle time

Manufacturer directly provides the value of average CPI

$$T = I_c \times CPI \times \tau$$

I_c → No of instructions per program

CPI → average CPI per instruction

τ → cycle time.

Basic performance equation.

$$T = \frac{I_c \times CPI}{f}$$ where $f = \frac{1}{\tau}$ → frequency of system clock

$$T = I_c \times CPI \times \tau$$

$$CPI = p + (m \times k)$$

p → cycles required by processor

m → cycles required by memory.

k → ratio of memory cycle time & processor cycle time.

memory slower than processor.

memory cycle time > processor cycle time.

$$\text{MIPS RATE} = \frac{I_c}{T \times 10^6} = \frac{I_c}{I_c \times CPI \times \tau \times 10^6}$$

MIPS → Million Instructions per second.

$$= \frac{1}{CPI \times \tau \times 10^6}$$

$$= \frac{f}{CPI \times 10^6}$$

A pugum has 2 million instruction which is to be executed on a processor with f = 400 MHz. The program consists of 4 types of instruction mixes

	CPI	Instruction mix
I_1	1	60%
I_2	2	18%
I_3	4	12%
I_4	8	10%

Calculate MIPS rate.

$0.6 + 0.36 + 0.48 + 0.8 = 2.24 = CPI$.

$= \frac{400 \times 10^6}{2.24 \times 10^6}$

$= 178$

MFLOPS rate $= \frac{f}{CPI \times 10^6}$ (same as MIPS)

CPU

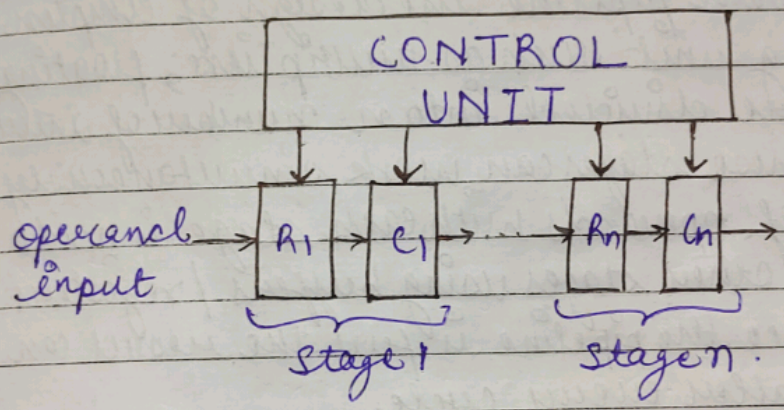
Processing unit OR Data path — Control unit.

PAGE No. DATE

Parallel Processing.

Pipelining

- Data path pipelining OR Arithmetic pipelining OR Processing Unit pipelining.
- Instruction pipeline



R_i → Registers or Buffers to store incomplete result

C_i → Computation unit stage.

Instruction Pipeline.

No Pipeline

I_1	F	D	E						6T states
I_2				F	D	E			

I_1	F	D	E		4T.
I_2		F	D	E	

- Pipeline is a technique for increasing processor throughput.
- It can be applied in processor in 2 ways:

(i) Hardware pipeline (also processing pipeline or arithmetic or data path pipeline)

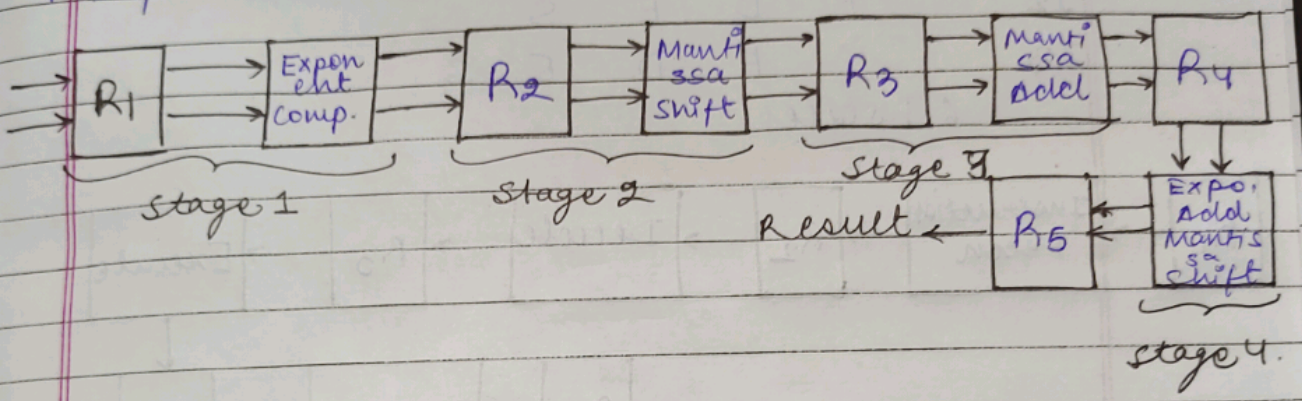
(ii) Instruction pipeline.

In hardware pipeline the design of complex processing unit such as multipliers, floating point adders are divided into n number of intermediate stages. These stages can work simultaneously on a number of operands with each stage is made independent of other stages using buffers / registers as a result once the pipeline is full the result can be obtained after every cycle.

In case of instruction pipeline fetching of next instruction can be overlapped with decoding of first instruction which means when I st Instruction is being decoded next instruction can be fetched.

Floating Point Addition

1. Exponent comparison
2. Mantissa shift
3. Mantissa add
4. Exponent add/sub, Mantissa shift.



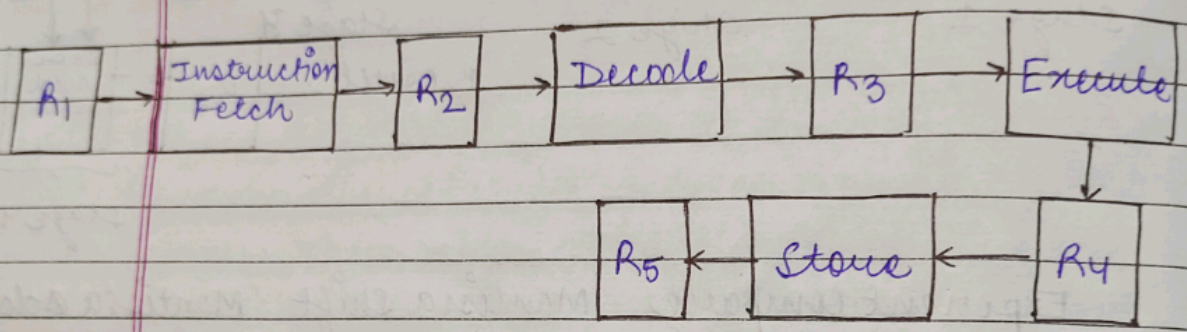
Exponent compare Mantissa shift Mantissa add

T → A, B
2T → C, D A, B
3T → E, F C, D A, B
4T → G, H E, F C, D A, B.

Assume that the instruction cycle of a processor requires Four steps i.e., instruction fetch, decode execute, store result. Draw the diagram to implement four stage pipelining. Also, indicate how pipelining increase throughput in this case.

```
I1  F  D  E  S
I2        F  D  E  S
I3              F  D  E  S
    12 T states
```

```
I1  F  D  E  S
I2     F  D  E  S
I3        F  D  E  S
    6 T states
```



on branching achieving this type of pipelining is problematic

```
    MOV A,B
    ADD B
    JNC AGAIN
    STA 3000
AGAIN HLT
```

Branch Prediction

whenever we encounter conditional statement there are two probabilities.