Date 04/04/24

Page No.

(Unit:3)

Control Design

OPCODE	OPERAND

→ 3 ADDRESS INSTRUCTION

→ 2 ADDRESS INSTRUCTION

→ 1 ADDRESS INSTRUCTION.

→ 0 ADDRESS INSTRUCTION

$X = (a+b) \times c$

★ **3 ADDRESS INSTRUCTION :**

ADD R1 A,B.	$R1 \leftarrow M[A] + M[B]$
MUL X R1,C	$X \leftarrow R1 * M[C]$

★ **2 ADDRESS INSTRUCTION :**

MOV R1, A	$R1 \leftarrow M[A]$.
ADD R1, B	$R1 \leftarrow R1 + M[B]$
MUL R1, C	$R1 \leftarrow R1 * M[C]$

Use X instead of R1.

→ **One Address Instruction:**

① LOAD	A	$AC \leftarrow M[A]$
② ADD	B	$AC \leftarrow AC + M[B]$
③ MUL	C	$AC \leftarrow AC * M[C]$
④ STORE	X	$X \leftarrow AC$.

Zero Address Instruction: (ab+)*c

⇒ ab+c*

X = ab+c*

ADD ;	TOS → $\boxed{a+b}$
	TOS → $\boxed{c}$
	MUL
	TOS → $\boxed{(a+b)*c}$
	POP → Ⓧ operation.
	$\boxed{X=(a+b)*c}$

X = (a-b)÷[(c*d)+e]

$X = \frac{(a-b)}{(c*d)+e}$

Three Address Instruction:

① SUB R1 a,b	R1 ← M[A]-M[B]
② MUL R2 c,d	R2 ← M[C]*M[D]
③ ADD R3,R2,e	R3 ← R2+M[E]
④ DIV X R1,R3	X ← R1÷R3.

Two-Address Instruction:

① MOV R1, A	R1 ← M[A]
② SUB R1, B	R1 ← R1-M[B]
③ MOV R2, C	R2 ← M[C]
④ MUL R2, D	R2 ← R2*D
⑤ ADD R2, E	R2 ← R2+E
⑥ MOV X, R1	X ← R1
⑦ DIV X, R2	X ← X÷R2.

→ One Address Instruction:

① Load A	AC ← M[A].
② SUB B	AC ← AC - M[B].
③ MOV X1	X1 ← AC.
④ LOAD C	AC ← M[C]
⑤ MUL D	AC ← AC *M[D].
⑥ ADD E	AC ← AC + M[E].
⑦ MOV X2	X2 ← AC.
⑧ LOAD X1	AC ← X1
⑨ DIV X2	AC ← AC ÷X2
⑩ STORE X	X ← AC

Date 08 04 24

→ Zero Address Instruction:

X= (a-b) ÷ [(c*d) +e]

X = ab - (cd* +e) %.

SUB; TOS → $\boxed{a-b}$

PUSH; TOS → c

PUSH; TOS → d.

POP; TOS → $\boxed{cd*}$

PUSH; TOS → e.

ADD; TOS → (c*d) +e.

POP; TOS → (a-b) ÷ ((c*d)+e).

$\boxed{X = TOS}$

Control Unit

HARDWIRED

- HARD WARD
- FAST
- SIMPLE CKT
- NO INTERFACE REQUIRED
- INSTRUCTIONS ARE PROESSED DIRECTLY BY PROCESSOR
- NOT SCALABLE
- SHORTER CONTROL UNIT
- SPECIFIC

MICROPROGRAMMING CONTROL UNIT

- SOFTWARD
- SLOW
- COMPLEX

HORIZONTAL (SUPPORT LARGER CODEWORD)

VERTICAL (SUPPORT SMALLER CODE WORD)

- INTERFACE REQUIRED TO CONVERT INSTRUCTION TO MACHINE LANGUAGE
- SCALABLE
- LARGER CONTROL UNIT
- GENERALIZED

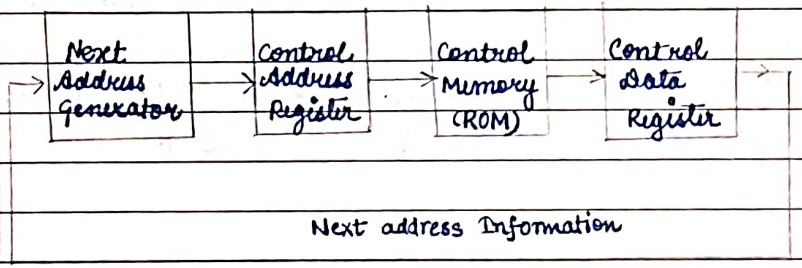
Horizontal [COMPUTER]	Vertical [MOBILE]
* Degree of Parallelism is larger.	* Supports lower degree of Parallelism
* Supports longer control word.	* Supports shorter control word.
* Additional Hardware in form of decoder is used to generate signal.	* No Additional Hardware is required.
* Faster than Vertical Micropro-grammed Control Unit.	* Slower than Horizontal Microprogrammed Control Unit.
* It is more flexible than Vertical control Unit.	* It is less flexible than horizontal Control Unit but more flexible than Hardwired control unit.
* It is used in horizontal micro instruction where every bit in control field attaches to control line.	* A code is used for each instruction to be performed, decoder, translator this code to individual control signal

* It makes less use of ROM encoding than Vertical.	* It makes more use of ROM encoding than horizontal

Date 19/04/24

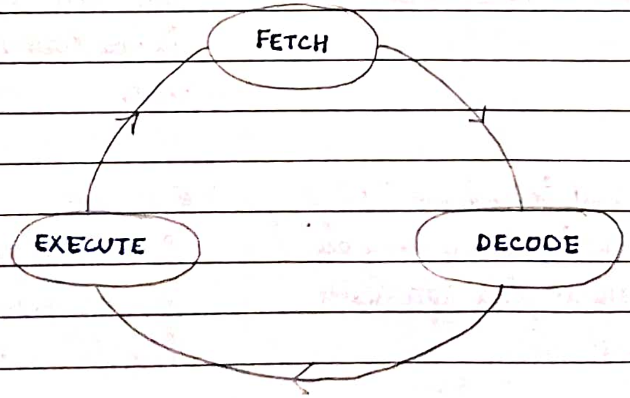
★ It makes less use of ROM encoding than Vertical.	★ It makes more use of ROM encoding than horizontal

Microprogram Control Unit:



Microprogram Sequencer

Instruction Cycle:



1) Fetch: Instructions are fetch from the memory to the Instructio Register.

{PC → IR (PC passes its instruction to IR)} following intermediate steps are followed.

★ PC → MAR

★ MAR → Memory

★ Memory → MDR/MBR

★ MBR → IR

2) Decode: IR → Instruction Decoder

3) Execute: "STORAGE"

★ CPU → MAR

★ {MAR → Memory} this is executed by the steps given below:

★ MAR → Address Bus

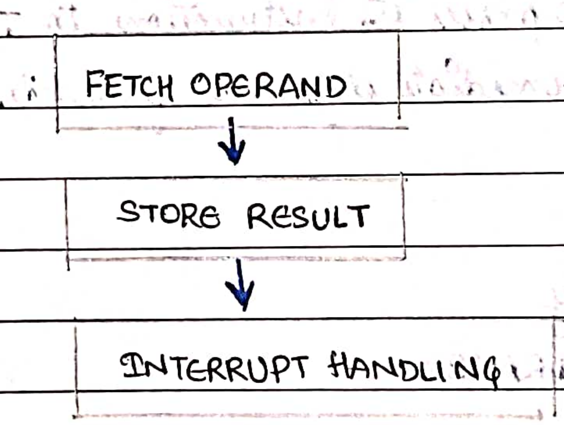
★ MDR → Data Bus

★ MDR → Memory

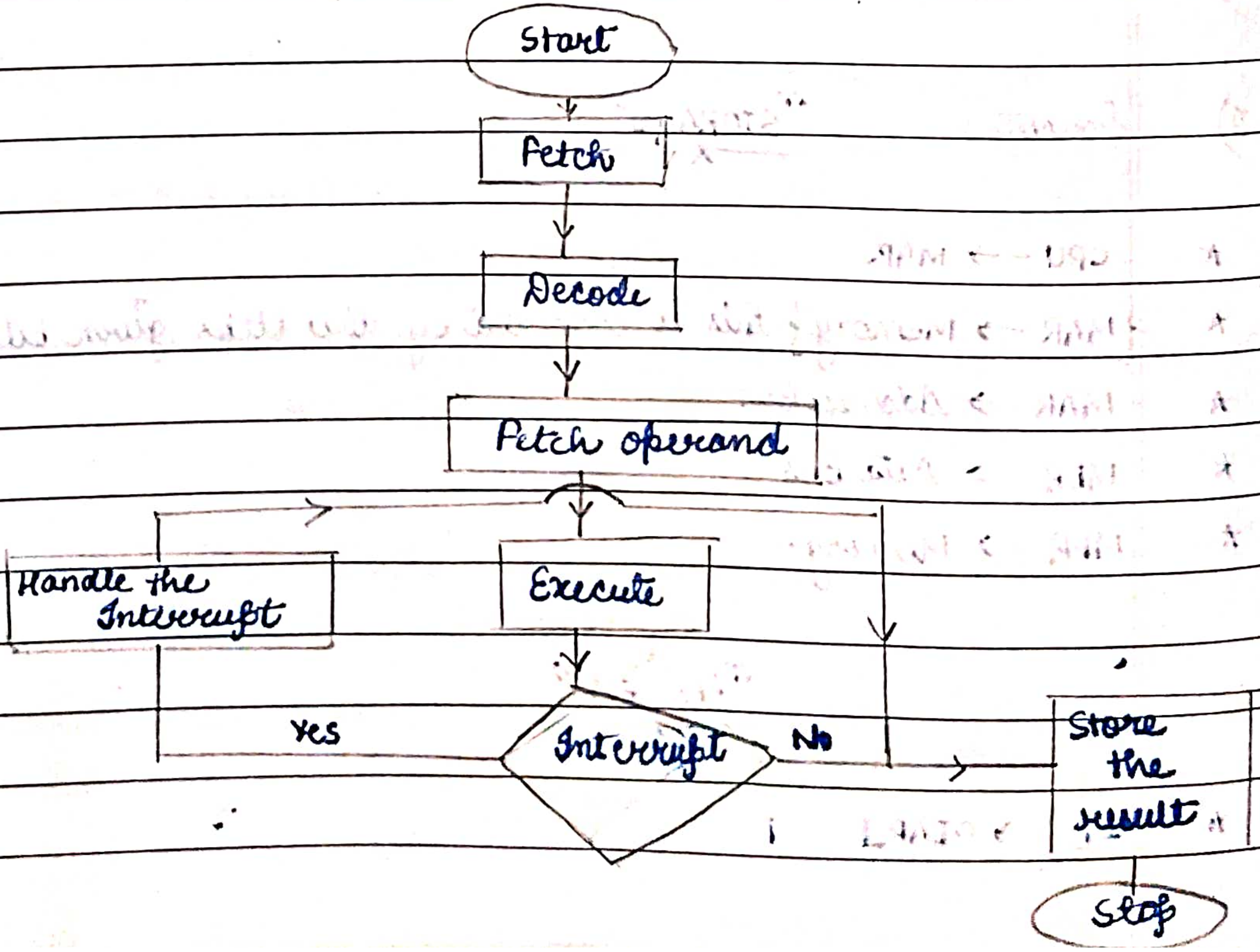
"PROCESS"

★ ID → C[AR]

- ★ C[AR] → Control Memory
- ★ CM → C[DR]
- ★ C[DR] → AC



Instruction Subcycle


Start
Fetch
Decode
Fetch operand
Handle the Interrupt
Execute
Yes
Interrupt
No
Store the result
Stop