

~~74~~

(74)

-: HAND WRITTEN NOTES:-

OF

ELECTRONICS & COMMUNICATION

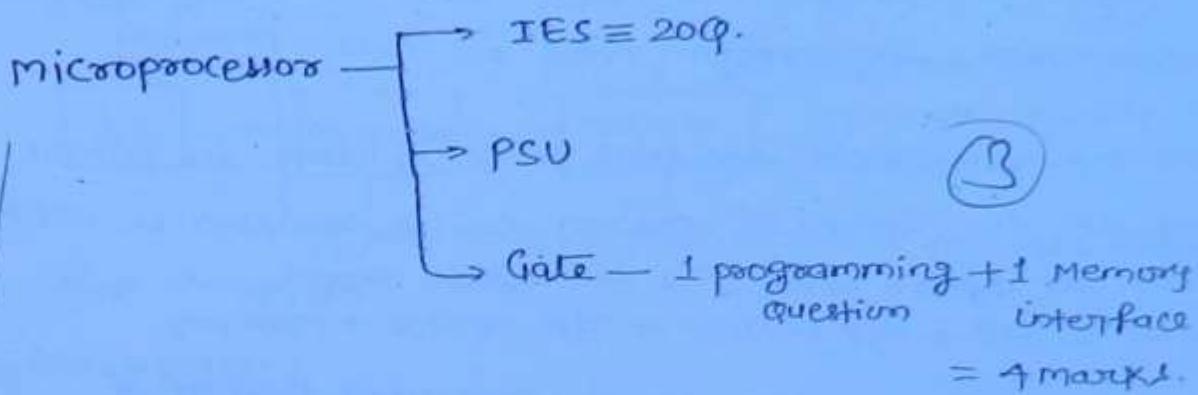
ENGINEERING

(1)

-: SUBJECT:-

 MICROPROCESSOR &  
COMPUTER ORGANIZATION

、②



Definition:- it is electronic device that fetch instruction from memory execute them & provided result.

it is integrated chip which has computing & decision making capability.

↳ Analysis of data.

<u>chip</u>	<u>bit of chip</u>	<u>Tech</u>
4004	4-bit	PMOS
8008	8-bit	NMOS
8080	8-bit	NMOS
→ 8085	8-bit	NMOS
→ 8086	16-bit	HCMOS
Improved by (IBM) 8088	8/16 bit	HCMOS

↳ H = High density channel

Note- Externally 8-bit internally 16-bit.

80186	16 bit
80286	16 bit
80386	32 bit
80486	32 bit

80586 = pentium = 64 bit

Pentium

Pentium Pro.

P-II

P-III

P-IV

Processor - RISC technology

Main memory:- RAM

→ instructions or program feed in main memory.

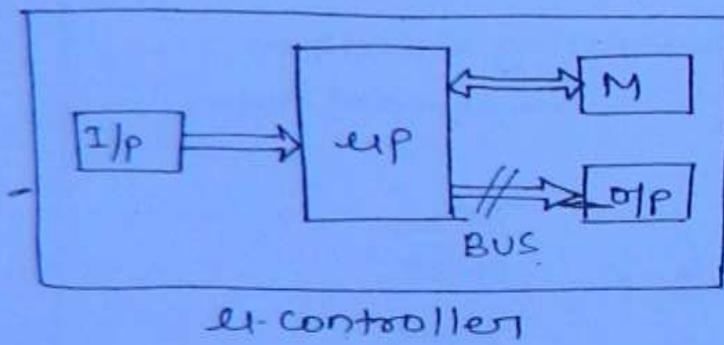
ROM-

size of processor is big & it come in picture only at the time of power switch ON.

→ in that all system related information stored.

• computer = CPU + I/O device + O/P device + Memory.

if CPU is work as CPU in a computer then this computer is known as micro-computer.



μC = Hardware + Software  
CPU

Ques ① if micro-computer on single platform then it is known as micro-controller.

② A processor cannot perform any task on its own.

ASIC = Application specific integrated chip.

↳ A programmed micro-controller

BUS :- it is group (parallel combination) of metal wire that is used for interfacing b/w different devices.

Machine language:-

→ If command (or instruction) written in binary pattern then it is called machine language.

→ It is platform dependent language.

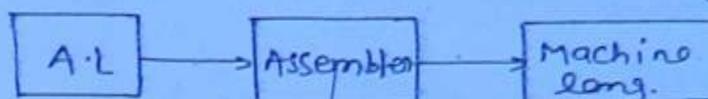
Assembly language:- If binary pattern replace by English like word that is called "Mnemonics".

→ i.e. commands in assembly language

→ Assembly language is also platform dependent.

(S)

⇒ If this task is performed manually then it is known as hand assembly



→ Assembler is nothing but a SW that converts Assembly language to machine language.

Low level language:-

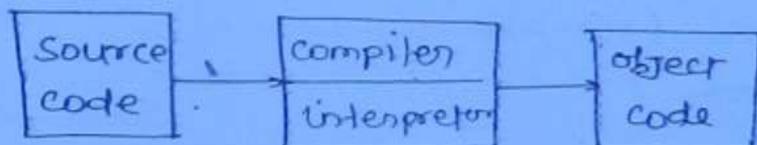
that language which are platform specific or platform dependent

Ex- M.L & A.lang.

High level language:-

All platform independent language is called HLL

Ex- C, C++, Java.



→ User understandable language

→ Machine understandable language

→ Compiler read whole program at once produce its object code which is executed by microprocessor

→ Compiler is nothing but SW. ex C, C++

→ Interpreter read one instruction at a time from source code produce its object code which is executed by CPU before reading next instruction and interpreter is a SW.

8085 UP :-

Ex BASIC (Beginner's purpose symbolic instruction code)

→ 8085A is a commonly known 8085 CPU.

→ It is based on NMOS technology.

→ 8085 CPU is 8-bit CPU.

→ Instruction set of 8080 CPU is exactly common in instruction set of 8085.

Note: Inst. set of 8080 + RIM + SIM = Inst. set of 8085

Two new inst. (RIM, SIM) are added in the instruction set of 8080.

RIM - Read interrupt mask

SIM - Set interrupt mask.

(6)

### bit of processor:-

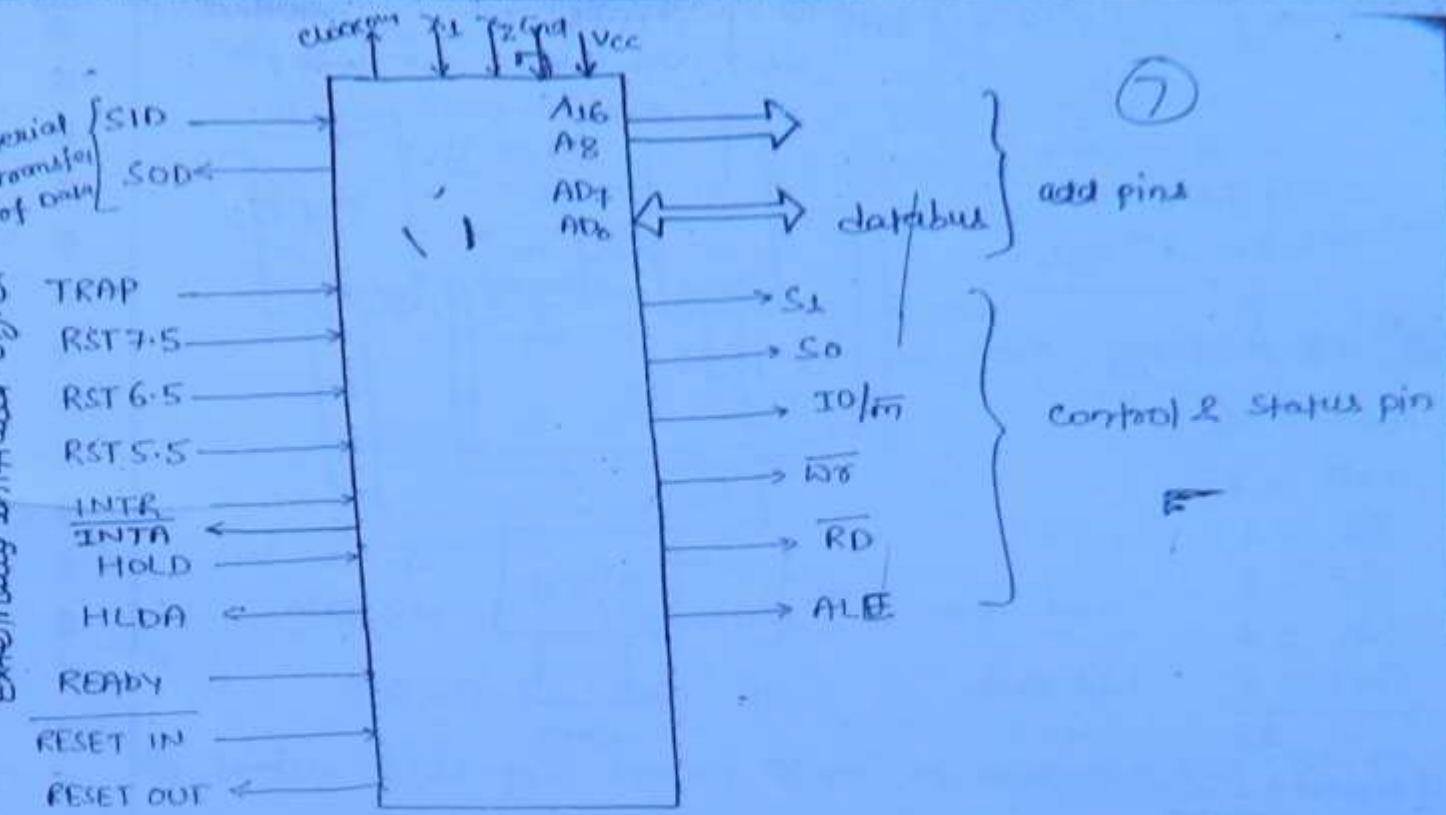
- Size of ALU is defined as bit of 4 processor.
- In one machine cycle no. of bit of data executed is known as bit of exp.
- $f = 3\text{MHz}$  (in 1971 after that upto 5MHz).
- (i) Speed  $\propto f$
- (ii) Speed  $\propto$  size of ALU i.e.  $\rightarrow$  data bus.

Arch. of 8085:- External Arch. (IES-10)  
internal Arch.

### External Arch.:-

- It is 40 pin exp.
- There is no disable enable pin 8085 exp.
- One Ground pin that is pin no 20.
- All 40 pins divide in 6 group as per there function.
  - (i) Address pin.
  - (ii) Data pin/bus.
  - (iii) Control & status pin.
  - (iv) Serial I/O pin.
  - (v) externally initiated signal pin.
  - (vi) power supply & freq. pin.

{ Bus is nothing but parallel combination of metal wire.



in 8085

\* total no. of pins direction outward = 27

inward to CPU = 21

### (i) Address pins/Bus :-

→ Add bus of 16 bit

$2^{10}$  = Kilo

→ Unidirectional / Outward

$2^{20}$  = Mega

\* → Total no. of memory location that can interface with 8085  
 $= 2^{16}$  Location

$= 2^6$  K

$= 64K$  location

→ no. of bits stored in memory location that is known as word size  
Data pins -

no. of data pins = 8

### Bi-directional

Maximum memory that can interface with 8085 CPU

$= 2^{16} \times 8$  bits

$= 2^{16}$  Byte

$= 64K$  byte =  $2^{16}$  Kbytes = 65536 bytes

$$\text{Memory} = (2^N \times d) \text{ bits}$$

$N$  - add.

$d$  - data

If  $d=8$

$$\underline{2^N \text{ byte}}$$

Lower 8 pins of add. bus  
are used as data pin  
data pin + add. pin = 16  
(8) (40)

(8)

$2^{16} \times 8$  memory. Max. no. of pins.

$$\text{Add.} = 16$$

$$\text{data} = 8$$

$$\overline{RD} = 1$$

$$\overline{WR} = 1$$

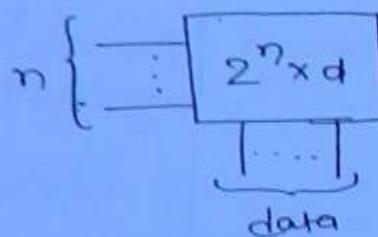
$$Vcc = 1$$

$$Gnd = 1$$

$$\underline{28}$$

chip enable + 1

$$\frac{\text{pin}}{29}$$



$n$  - add. bus  
 $d$  - data

Control & status pin :-

$S_1$  } By measuring status of these two pin we can  
 $S_0$  } find out which machine cycle going on inside the chip.

D/m :-

- [ ] 0 transfer of data from memory
- [ ] 1 transfer of data from I/O devices

$\overline{WR} = 0$  = write operation perform.

$\overline{RD} = 0$  = Read operation will perform.

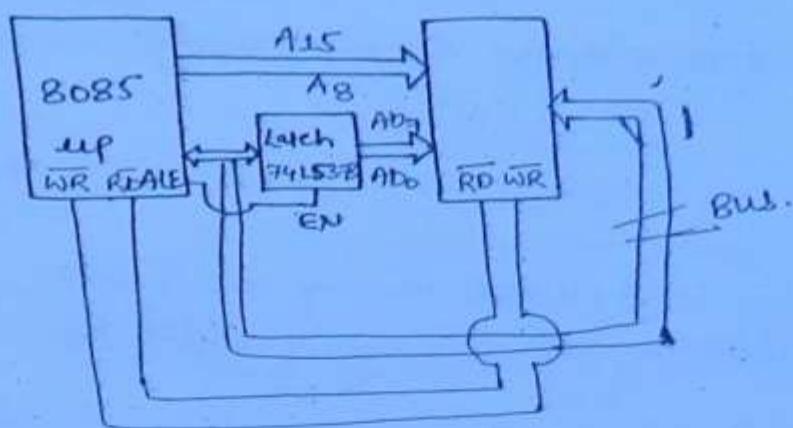
$$\overline{RD} = 0 \\ \overline{WR} = 0 \quad \times$$

$\rightarrow$  Simplex Mode

$$\overline{RD} = 1 \\ \overline{WR} = 1 \quad \checkmark$$

active low (8085)  $\rightarrow$  only five pins are active low

ALE → add latch enable.



(9)

74LS373  
→ D-flip-flop

$$ALE = 1 \quad AD_7 - AD_0 \equiv \text{add. bus}$$

$$ALE = 0 \quad AD_7 - AD_0 \equiv \text{data bus.}$$

→ By tying this pin lower 8-pin of address bus can be demultiplexed used as data pins also & multiplexing/ demultiplexing ≡ TDM

power supply & freq. pins:

$$\rightarrow V_{CC} = +5V$$

$$\rightarrow \text{pins } 20, 21, 22 = GND$$

$$\rightarrow f = 3MHz \quad (5MHz 8085 also available but not so popular)$$

Note — There is 6MHz crystal oscillator connected b/w X<sub>1</sub>, X<sub>2</sub> and a internally divided by two CLK is used to get 3MHz clock freq. (becas 3MHz crystal oscillates stability is less).

Clockout — by tying this pin up send clock freq. to other interface peripherals

for better synchronization.

serial I/O pins—

These pins are used for serial transfer of data by tying RIM & SIM low.

RIM = serial off data

SIM = serial off data

SID = 1 = serial I/O data

SOD = 0 = serial off data

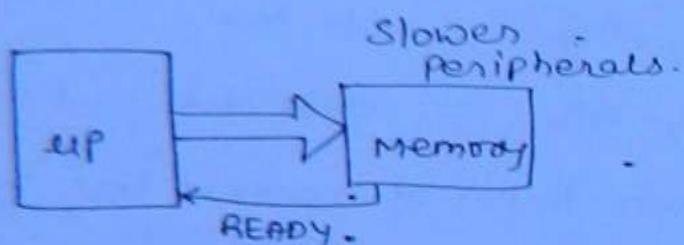
## Externally initiated signals:-

(10)

- TRAP
  - RSF7-5
  - RSF6-5
  - RSF5-5
  - INTR
- hardware interrupted (5 pins) (in 5 options are not given)  
then 6 pins.

HOLD :- By using this pin DMA (direct memory access) like peripherals make request to CPU for delinquish the vacant data pin

READY :-



By using this pin slower peripherals make interfaced with up.

RESET IN →

By using this pin reset command gives to CPU for reset.

Note → as soon as CPU receives reset command it generates reset out command for reset other interfaced peripherals.

Internal Arch.:-

ALU - Arithmetic logic unit

Accumulator

Registers

Register array

Temporary Reg.

Latch

→ All arithmetic & logic operations  
will occur in ALU

→ control timing CLK

→ interrupt control CLK

Note - Size of ALU = 8 bit  
(maximum)

## Registers

(11)

User Accessible Registers  
(General purpose registers)

→ B  
→ C  
→ D  
→ E  
→ H  
→ L

Bbit

Special Purpose Registers

user access 8-bit { → Accumulator (A) → Status reg (Flag reg.) → Stack pointer (SP) → Program Counter } 16 bit	user not access 8-bit { → temporary reg. → interrupt Reg. → Loaches → 16 bit         }
---	--

Note → in 8085, B, 8bit register and two 16 bit register (enhanced)

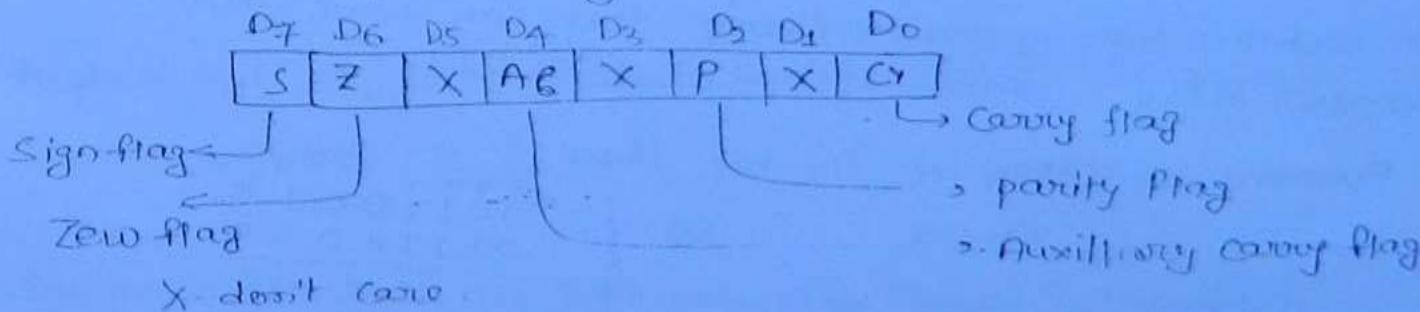
Accumulator - it is 8-bit special type of Register.

when Arithmetic & Logic operation b/w the two numbers, one number always taken from accumulator and result of arithmetic and logic operation will store in Accumulator.

Status Register (Flag reg. F)

→ it is 8-bit special type register.

→ in 8085 five flags are defined at the different bits of this register.



Note → Status of flag affect according the result of arithmetic and logic operation.

Sign Flag (S) →

- if on the result of math & logic operation D7 bit or MSB is zero { S=0 (Reset) (no. is +ve) }
- if  $D_7 = 0 \text{ (+ve)}$  S=1 (Set) (no. is +ve)

If Z is the result of Arith & Logic operation all bits zero then Z=1 (Set).

If any bit is one Z=0 (Reset)

Parity flag →

If in the result of Arith & Logic operation no. of ones are even then P=1 (Set)

If no. of 1's are odd then P=0 (Reset)

→ 8085 is odd parity based system.

Result of Arith & Logic

0 0 0 0 0 0 0 0

S=0

Z=1

P=1

Carry flag (Cy) :-

If in the Arith & Logic operation carry pass from D<sub>7</sub> or MSB then Cy=1 (Set)

Otherwise Cy=0 (Reset)

→ In subtraction operation carry flag can work also Borrow flag.

→ If Borrow is taken at D<sub>7</sub> bit then

$$\begin{array}{ll} \text{Cy} = 1 & \text{Ex: } \left\{ \begin{array}{r} 10110010 \\ 01110010 \\ \hline 100100100 \end{array} \right. \\ \text{Otherwise } \text{Cy} = 0 & \text{Cy} = 1 \text{ (Set)} \end{array}$$

Auxiliary Carry flag :-

If carry pass from lower nibble to upper nibble or  
D<sub>3</sub> to D<sub>4</sub> AC=1 (Set)      Ex: {  
Otherwise AC=0 (Reset)      

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
d <sub>1</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>

→ for programmer status of four flag available, Cy, Z, S, P.

→ programmer can check these flags available.

line AC flag used internally for the adjustment of content of Accumulator in BCD format by assuming earlier operation was BCD addition operation. (B)

When DAN (Decimal Adjustment of NBR) instruction will execute.

Stack pointer:- (SP)

→ It is 16 bit special type Reg.

SP hold the add. of top of stack that define inside the Memory.

Programm Counter:- (PC)

→ It is 16 bit special type of Register.

\*→ Programm counter hold the address of next instruction to be fetch.  
$$PC \leftarrow PC + 1$$

NT → For the storage of 16 bit data then two register can be combined in specific way.

$$B\ C \equiv B$$

$$D\ E \equiv D$$

$$H\ L \equiv H$$

→ Higher byte of 16 bit data always store in higher order register =  $\begin{matrix} B \\ D \\ H \end{matrix}$

$$\text{Order register} = \begin{matrix} B \\ D \\ H \end{matrix}$$

→ Lower order byte of data always store in

$$\begin{matrix} C \\ E \\ L \end{matrix}$$

NT PSW (Programm Status Word)

It is user defined 16 bit register in which higher 8 bits are accumulator & lower 8 bit status register

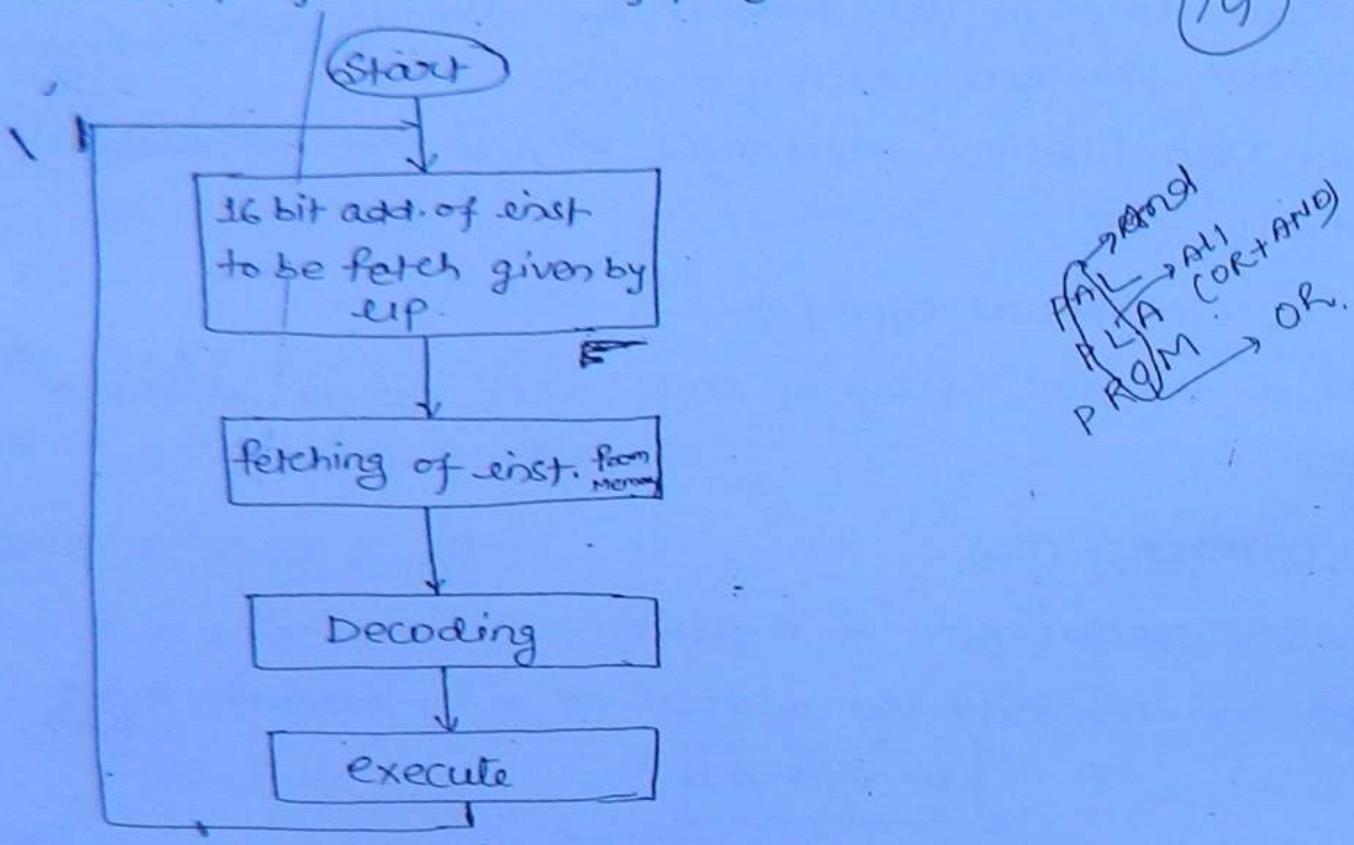
$$\boxed{PSW = AF}$$

A - Accumulator  
C - ..

## Instructions

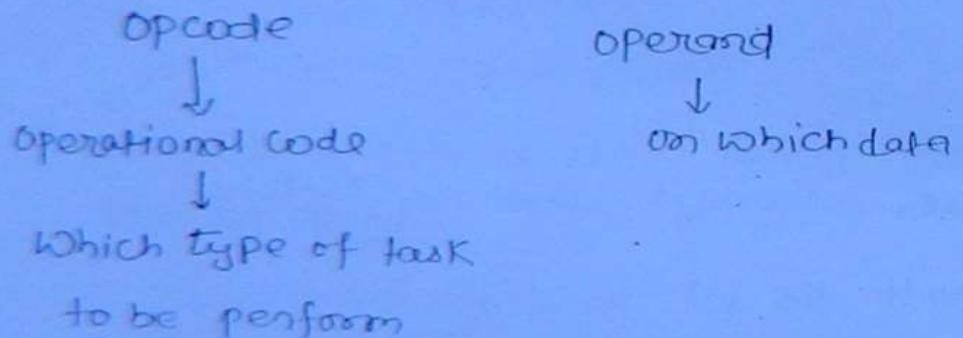
### (ii) flow chart of execution of program

74



## Instruction:-

- It is command that given to CPU to perform a task.
- Every instruction has two field.



→ Label: opcode operand ; discription.  
add of      instn |      operation (optional)

→ In 8085, 74 opcodes are define and by using them 256 inst. are possible but 246 instructions are available.

Ex -  
 MOV R, C } both are different  
 MOV A, B } inst.

Ex.

<u>MOV A C</u>	$\equiv 1\text{ byte}$
<u>MVI B, 28H</u>	$\equiv 2\text{ bytes}$
<u>LXI D, 2589H</u>	$\equiv 3\text{ bytes}$
<u>SHL</u>	

1000 : 27
1001 : F0
1002 : 28
1003 : 90
1004 : 89
1005 : 25

(iii) IWS (Inst. word size)

program always stores to continuous memory location because here  $PC = PC+1$

→ No. of memory bytes register to feed any inst. that is known as IWS (inst. word size).

On the basis of IWS, instructions can be classified in three groups.

- (i) 1 byte ( $IWS = 1\text{ byte}$ )
- (ii) 2 byte ( $IWS = 2\text{ bytes}$ )
- (iii) 3 byte ( $IWS = 3\text{ bytes}$ )

(i) If in the inst. R, Rp (or) no operand then  $IWS = 1\text{ byte}$ .

R = Register

Rp = Register pair

Ex

MOV A, B	} $IWS = 1\text{ byte}$
Nop	
LDAX R	

(ii)

If in the inst. 8 bit no. (add. or data) is present  $IWS = 2\text{ bytes}$

Ex  $MVI @ C, 25H \equiv IWS = 2\text{ bytes}$

IN 25H  $\equiv 2\text{ bytes}$

(iii) If in the inst. 16 bit no. (add. or data) is present then

$IWS = 3\text{ bytes}$

Ex

LXI SP, 2718H  $\equiv IWS = 3\text{ bytes}$

LOD 5986  $\equiv IWS = 3\text{ bytes}$

NT

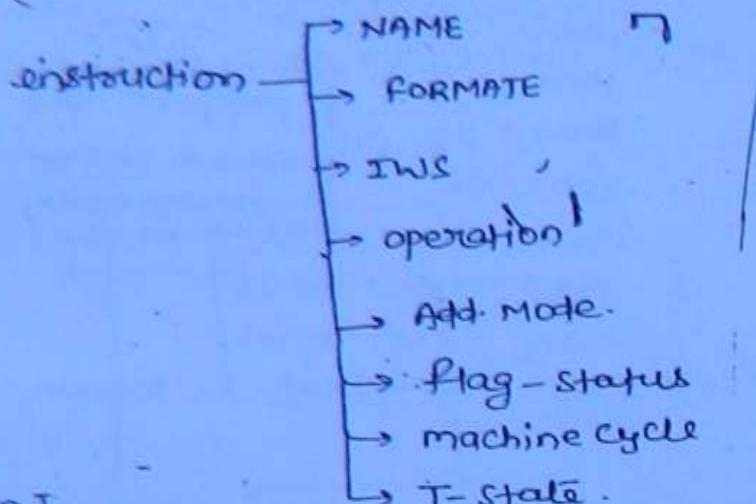
If a 16 bit no. stores in memory then higher order byte always at higher order add. (memory add.) Lower order byte always at lower order add. (memory add.)

→ In 8085 left and right shift of 8bit

∴ maximum no. of bits desired  $\approx 256 - 2^8 =$  more no. of address

(at a time) Total devices =  $256 + 256 = 512$

16



group I

8-bit data transfer instruction :-

MOVE instruction.

(i) MOV Rd, Rs

Rd = destination Register

Rs = source register

ex mov B, C

Rd } A, B, C, D, E, H, L &  
Rs } M

IWS = 1 byte

operation - Content of Rs will pass in Rd (copy of)

$$[R_d] \leftarrow [R_s]$$

[ ] ← data  
add

B = 26

C = 59H

mov B, C

B = 59H

C = 59H

(ii) MVI R, 8 bit data

(move immediately)

IWS = 2 bytes

operation -  $[R] \leftarrow$  8bit data { 8bit data will store in

R: ABCDEHL&M

MVI B, 29H

B = 29H

} if last character of  
only operand I then data  
otherwise address

M if it is 8-bit user define register - in memory. and address of M is the content of HL pair.

(17)

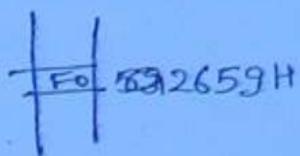
Ex

$$[H] = 26H$$

$$[L] = 59H$$

MOV A, M

$$A = F0$$



(iii)

IN 8 bit post add.

IWS = 2 byte

operation - when this inst. will execute data available at the 8-bit post address will store in Accumulator.

$[A] \leftarrow [8\text{-bit post Add.}]$ .

$\begin{array}{r} 10111110 \\ \hline 101011 \end{array}$  post add.  
A F keyboard

Ex

IN AFH

$$[\bar{A}] = BEH$$

'P' -  $\begin{array}{r} 10111110 \\ \hline 8 \quad E \end{array}$

(iv)

OUT 8 bit post add.

IWS = 2 byte

operation :- when inst. will execute content of Accumulator will available at 8-bit post add. given in the inst.

Ex:

MVI B, 92H

MOV A, B

OUT 12H

Note :- All above four inst. are data transfer inst. so status of flag will not affect

Group II -

## Machine Control Instruction :-

1) NOP no operand

\_\_\_\_\_  
\_\_\_\_\_  
NOP

IWS = 1 byte.

operation:- when this inst. will execute processor will not perform any task.

Note :- this inst. is used for creation of delay

2) HLT No Operand.

IWS = 1 byte

operation:- when this inst. will execute further increment of program counter will stop.

Note :- it is last inst. of every program.

1000H MOV BC  
1001 MVJ A 29H  
1002 IN 20H  
1003  
1004  
1005 MOV DE  
1006 HLT

PC = 1007

Note :- all above two inst. are machine control inst. and that will not affect status of flag.

### Addressing Mode :-

- Form of add. If data given in the inst. is known as add. Mode
- ① Register add. Mode :-  
 If add. of data given in the form of Register  
 \ Then Reg. add.  
 Ex: MOV B, C (19)

- ② Direct add. Mode :-  
 If add. of data given directly in the inst.  
 Ex: IN 25H F  
OUT 32H

- ③ Immediate add. Mode :-  
 If data itself given in the inst.  
 Ex: MVI D, 29H { if opcode last character I  
Then immediate add. mode  
Reverse not correct }

- ④ Indirect add. mode :- Indirect Reg. add. mode.  
 If Content add. of data given in the form of content of  
 Reg. then indirect add. mode

Ex: MOV B M.

- ⑤ Implicit add. mode :-  
 If add. of data not require it is define  
 in opcodes only

Ex: NOP, CMA.

### 8-bit Arithmetic Instructions

② ADD R

IWS = 1 byte

R → A, B, C, E, D, H, L & M

Operation → Content of R will add in [A] & final result will stored in [A]

$$[A] \leftarrow [A] + [R]$$

<u>Add. mode</u> <u><del>R ≠ M</del></u>	<u>Reg. add.</u>	<u>Indirect add.</u>	<u>Ex:</u> <u>[A] = 25H</u> <u>[R] = 3CH</u> <u>ADD R</u> <u>[A] = 61H</u>
			<u>25H</u> <u>3CH</u> <u>61H</u>

S Z AC P Cy  
1 0 1 0 0

$$[A] = 11000001 = 1CH$$

### 1) ADI 8 bit data

IWS → 2 byte

(20)

operation:- 8 bit data will get added in accumulator & result will store in accumulator.

$$[A] \leftarrow [A] + [8\text{bit data}]$$

Add. Mode immediate add. mode.

### 2) SUB R

IWS = 1 byte

R → A, B, C, D, E, H, L & M.

operation:- When inst. will execute content of R will get subtracted from content of A & result will store in A.

$$[A] \leftarrow [A] - [R]$$

Add. Mode

Register Add. mode RFM

Indirect R = M

$$[A] = 29H$$

$$[B] = 35H$$

### SUB B

$$[A] = 0010\ 1001$$

S Z AC P Cy  
1 0 ? 0 1

$$[B] = 0011\ 0101$$

$$[R] = 11110100$$

### 3) SUI 8 bit data

IWS = 2 byte

8 bit data get subtracted from [A] & result will store in [A]

[A] ← [A] - [8 bit data] add mode

Q: write down one line inst. that make content of Acc. 00H regardless of previous value.

- (i) SUB A  
(ii) MVI A 00H.

(2)

INR R

IWS = 1 byte

R = A, B, C, D, E, H, L, & M

operation - when this inst. will execute content of R will increase by 1 & result will store in R

$$[R] \leftarrow [R] + 1_{LSB}$$

Ex: B = 21H

INR B

$$\begin{array}{r} [B] = 00100001 \\ \quad + 1_{LSB} \\ \hline 00100010 \end{array}$$

[B] = 22H.

Add mode R ≠ M Reg. add.

R = M Indirect Reg. add. mode.

(vi)

DCR R

IWS = 1 byte

R = A, B, - - -

operation - when this inst. will execute content of R will decrease by one & Result will store in R.

$$[R] \leftarrow [R] - 1_{LSB}$$

Note: → ADD, ADC, SUI, SUB will affect status of all flag

→ INR & DCR will affect only four flag (S, Z, AC, P)

→ INR & DCR will not affect carry flag.

B = FFH

INR B

$$\begin{array}{r} [B] = 11111111 \\ \quad + 1 \\ \hline 00000000 \end{array}$$

S Z AC P C  
0 1 1 1 1 previous

Group IV

## 8-bit Logical operation:-

### i) AND operation:-

(i) AN A R.

22

IWS = 1 byte

R → A, B, C, D, E, H, L & M.

operation:- when this inst. will execute content of R will get AND operation with [A] bit by bit result will store in A.

Ex:

$$\begin{array}{rcl}
 [A] = & 0000\ 1100 & [A] = 1010\ 1100 - ACH \\
 & (0CH) & \\
 [B] = & 5EH & [B] = 0101\ 1110 - 5EH \\
 & & \hline
 & & [A] = \begin{matrix} ACH \\ 0000\ 01100 \end{matrix}
 \end{array}$$

### ii) ANI 8-bit data :-

IWS = 2 byte

operation:- 8-bit data will get AND operation with content of Accumulator and Result will store in [A]

Add. Mode = Immediate add.

Ex ANI 00H

### ② OR operation:-



### iii) ORFA R

IWS = 1 byte

R = ABCDEHL&M

operation:- Content of R will get OR operation with content of A bit by bit & result will store in A.

Ex: [A] = 25H = 00101001

$$\begin{array}{rcl}
 \text{MVI A } 25H & [A] = 25H = 00101001 \\
 \text{MVI C } 1EH & [C] = 1EH = 00011110 \\
 \text{...} & [A] \rightarrow \begin{matrix} 00101001 \\ 00011110 \end{matrix} = 00111111
 \end{array}$$

if  $R \neq M$  Register add. Mode  
 $R = M$  indirect add. Mode.

(iv) ORI 8 bit data

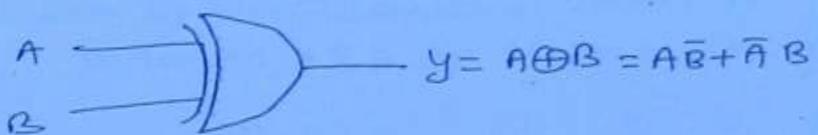
IWS - 2 byte

(23)

operation:- 8 bit data will get OR operation with content of acc. bit by bit & result will store in [A].

Add. Mode = immediate

(v) Ex-OR operation  $\rightarrow$



(vi) XRA R ..  
IWS = 1 byte.  $\therefore R \rightarrow ABCDEHL \& M.$

operation:- when this inst. will execute content of [R] will get EX-OR operation with [A] bit by bit & result will store in [A].

Ex:  $[A] = 0101\ 0011 = 53H$

$[D] = 1011\ 0010 = B2H$

XRAD  $\rightarrow 1110\ 0001 = [A] = E1H$

(vii) XRI 8 bit data :-

IWS = 2 byte

operation:- 8 bit data will get Ex-OR operation with content of [A] bit by bit & result will store in [A].

Add. Mode  $\rightarrow$  immediate add. mode.

CMA no operand mode :- IWS = 1 byte

operation:- when this inst. will execute CMA

~~content of [A]~~ will get Complemented bit by bit & result

$[A] \leftarrow [A]$

$[A] = 29H = 00101001$

CMA

$[A] = 29H = 00101001$

$[A] = D6H = 11010110$

(24)

E :- flag can be classified in two categories

- Affect according the final result of A & L operation  
= S, Z, P.
- Affect according the process of A & L operation  
= Cy, AC.

	S	Z	AC	P	Cy.
ANA	✓	✓	1	✓	0
ANI	✓	✓	1	✓	0
ORA	✓	✓	0	✓	0
ORI	✓	✓	0	✓	0
XRA	✓	✓	0	✓	0
XRI	✓	✓	0	✓	0
CMA	x	x	x	x	x

x → not affect

✓ → According the result

0 → Reset

1 → Set

→ CMA will not affect any status of any flag.

→ ~~ANI, OR, XRI~~ will always reset the ~~or~~ reset common flag.

- AND operation always set the AC flag.
- OR & EX-OR Operation will always Reset the AC flag.

Ques MVI A, 2AH

(25)

ADD A

ORI AFH

INR A

CMA

HLT

after the execution of HLT instruction, status of flag.

Soln

$$[A] = 2AH$$

$$\begin{array}{r} [A] = \phantom{0}0\ 0\ 1\ 0\ 1010 \\ \phantom{[A] = }0\ 0\ 1\ 0\ 1010 \\ \hline \end{array}$$

$$[A] = 0\ 1\ 0\ 1\ 0\ 1000$$

$$AFH = 1\ 0\ 1\ 0\ 1111$$

~~11111111~~

~~----- 1 - INR A -----~~

S	Z	AC	P	CY
0	0	1	0	0
1	0	0	1	0
0	1	-1	1	0

↳ status of flag

Ques FFO0 MVI A 23H

FF02 MVI B 32H

FF04 XRA B

FF05 ADD 88H

FF07 HLT

after execution of above program value of program counter(PC)

[B] = ? and PSW (program status word)

$$[A] = 23H = 0010\ 0011$$

$$[B] = 32H = 0011\ 0010$$

$$[A] = 00010001$$

$$88H = 10001000$$

$$= 10011001$$

Ans = 00111001 FPC = FF07

S	Z	AC	X	PSW
0	0	0	1	0
1	0	0	0	100

PSW = A-F

= 99-84 H

F

Ques:- After the Arithmetic operation status of flag Register BBH  
Then content of [A] may be

(26)

	@ DBH	@ 75H	@ 65H	@ B6H	
BB	$\begin{array}{ c } \hline S \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline Z \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline \times \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline AC \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline \times \\ \hline 1 \\ \hline \end{array}$
DB	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$
75	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$
65	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline 0 \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$
B6	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 0 \\ \hline 1 \\ \hline \end{array}$	$\begin{array}{ c } \hline 1 \\ \hline 0 \\ \hline \end{array}$

P=0 odd parity.

Z=0 S=1

- ② Auxillary (AC) and carry (Cy) are check when process is given only then these parity will find out.

### Branch operation:-

→ Loop is the Special case of branch operation.



In 8085 3 instruction is defined for branch operation.

1. JUMP

2. CALL

3. RESTART

### JUMP instruction:-

JMP 16 bit address → Vector Location (unconditional JMP)

IWS = 3 by 6

Operation:- When this instruction will execute control will jump at given vector location (address).

Add. Mode → Immediate Add. mode

Ques: 1000 MVI B 27H  
 1002 MOV 0, E  
 1003 JMP 2918H  
 1006 —————  
 1004 → 18  
 1005 → 29

(Q7)

- ★ Jump instruction is data transfer instruction so status of flag will not affect.

Instruction cycle, Machine cycle and T-state :-

Instruction cycle:-

Total time required by execution to execute the instruction is known as instruction cycle.

- ★ Instruction cycle is the combination of one or more than one machine cycle.

Instruction cycle out of 6 machine cycle one or more than one machine cycle will exist.

Machine cycle :-

In 8085 six type of Machine cycle exist

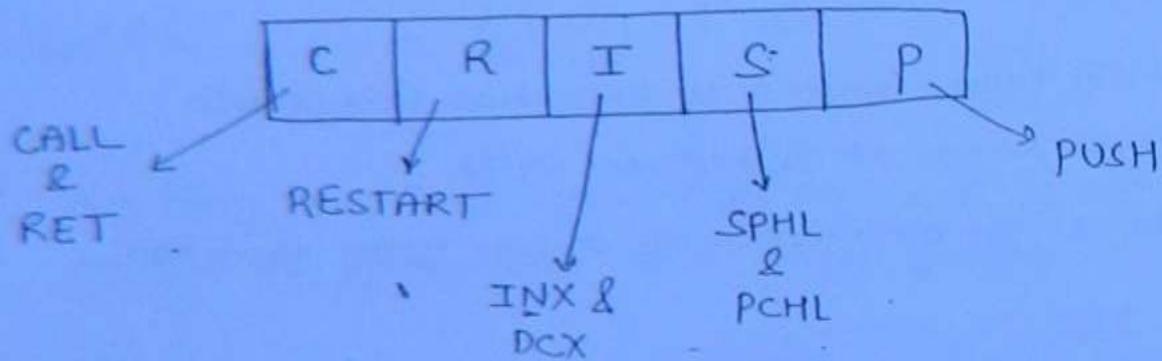
- [1] Memory Read M.cy (R)
- [2] Opcode Fetch M.cy (F/C)
  - ↳ (Machine code fetch M.cy)
- [3] Memory write M.cy (W)
- [4] Input Read M.cy (I)
- [5] Output write M.cy (O)
- [6] Bus idle M.cy (B)

Definition:- total time required by execution to execute different type of task in the execution of instruction.

### [1] - Fetch Machine cycle (F/S) :-

- It is first or only first machine cycle of every inst.
  - total time required by execution to fetch opcode (M. Cy) from memory
  - opcode fetch M. Cy is the special case of Memory Read operation.
- $F = 4T \rightarrow (238 \text{ code})$
- $S = 6T \rightarrow (8 \text{ code})$

opcode/machine code required 6T state M. Cy



### [2] - Memory Read Machine cycle (R) :-

- Time required to read 8-bit data from Memory

$$\boxed{1 R = 3T}$$

$$T \text{ state} = 1 T = \frac{1}{f}$$

### [3] - Memory Write Machine cycle (W) :-

- time required by execution to store 8-bit data in Memory

$$\boxed{1 W = 3T}$$

### [4] - Input Read Machine cycle (I) :-

- Time required by execution to read 8-bit information from A/D port

$$\boxed{1 I = 3T}$$

[5] - off write M-cycle (o):-

(29)

Time required by execution to make available 8 bit data at off.

$$1 \text{ O} = 3T$$

[6] Bus idle Machine cycle:- (B)

during the execution of some special inst. time for which Bus of CPU will be in idle condition and

$$1 \text{ B} = 3T$$

\*\* this machine cycle will exist in the execution of DAD Inst.

Inst.	M-Cy.	T-State
MOV B C	F	4T
MVI D, 29H	FR	7T
MOV B M	FR	7T
IN 25H	FRI	10T
OUT 19H	FRO	10T
ADD B	F	4T
ADD M	FR	7T
NOP	F	4T
HLT	F	4T/5T
SUB M	FR	4T
JNR D	F	10T
JNR M	FRW	
ANA M	FR	
CMA	F	
JUMP label add	FRR	10T

(30)

	<u>RD/M</u>	<u>RD</u>	<u>WR</u>	<u>S<sub>1</sub></u>	<u>S<sub>0</sub></u>
Fetch M-cy	0	0	1	1	1
Memory Read M-cy	0	0	1	1	0
Memory Write M-cy	0	1	0	0	1
IFP Read M-cy	1	1	0	1	0
IFP Write M-cy	1	1	0	0	1

S<sub>1</sub> S<sub>0</sub>

F

1 1 → fetch M-cy.

1 0 → Read operation

0 1 → Write operation.

Conditional JUMP Inst. :-

JC 16 bit add. if Cy = 1

JNC " Cy = 0

JZ " Z = 0

JNZ " Z = 0

JP " S = 0

JM " S = 1

JPE " P = 1

JPO " P = 0

IWS ≡ 3 byte

operations:-

If condition satisfy then jump instruction will execute otherwise skip it.

→ Add Mode → Immediate Add Mode same as unconditional jump.

conditional JUMP  $\rightarrow$  satisfy = FRR = 10

$\rightarrow$  not satisfy = FR = 7T

(31)

$\rightarrow$  if it is also data transfer inst. so state of flag is not affect

16 bit data TX instruction:-

(iv) LXI Rp 16 bit data :-

$R_p = BC \leftarrow DE, HL, SP.$

$\downarrow \quad \downarrow \quad \downarrow$

B

D

H

IWS  $\rightarrow$  3 byte.

16 bit data given in the instruction will store in Register pairs.

$[R_p] \leftarrow [16 \text{ bit data}]$

Add. Mode  $\rightarrow$  immediate Add. Mode.

M. cycle      FRR  
                  LXI B 2011 H

note

LXI SP 16 bit data

$[S] = 16 \text{ bit data}$ .

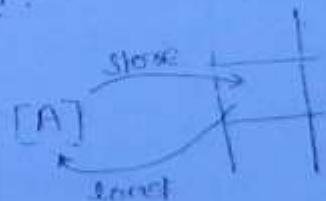
This inst. is use for initialization state in machine memory.

LXI SP 11 FFH

$[SP] = 11 FF$

(v) LDA 16 bit address :-

IWS = 3 byte



- data available at the Memory add. that is given in inst.

$[A] \leftarrow [16 \text{ bit add.}]$

Add. Mode  $\rightarrow$  Direct  
addr. mode  
M. Cy      FRRR  
                  LDA 20 FC

i) STA 16 bit add :-

IWS = 8 byte

(32)

Operation content of [A] will store at the memory location, that is given in instruction.

$$[A] \rightarrow [(16\text{ bit add})]$$

Ex:- STA 29DEH

2000 - STA

2001 - DE -

2002 - 29

Machine cycle  $\rightarrow$  FRRW

Add. Mode  $\rightarrow$  direct addressing mode.

IV) LDA X Rp :-

IWS = 1 byte

Operation :- data available at the 16 bit memory location of Rp will load in [A].

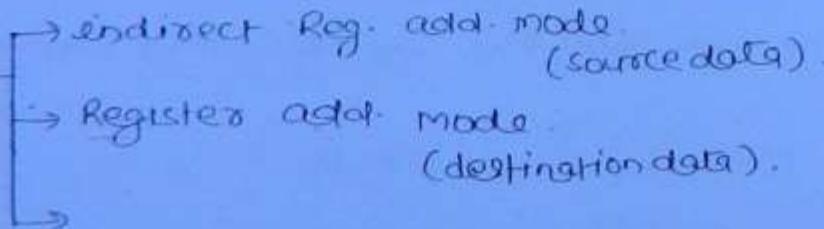
Ex:- LXI B 2919H

LOAX B

[A] = FC

Add. Mode  $\rightarrow$

LDA X Rp



Machine cycle  $\rightarrow$  FR

v) STAX Rp :-

IWS = 1 byte

Content of [A] will store at Memory location that is content of Register pair

$$[A] \rightarrow [(R)]$$

~~STAX Rp~~ → indirect add. (destination data)  
→ Register add. (source data)

(33)

Note →

[1] - for LDAX and ~~STAX~~ possible value of Rp.

$$Rp = \frac{BC}{DE}$$

[2] - All above inst. is data transfer instruction so status of carry flag will not affect.

Ques

MVI A FOH

ORA A

Loop: INR A

JNC Loop,

HLT

How many time loop will execute.

$$[A] = FO = \begin{array}{r} 1111 \\ 0000 \end{array}$$

Qy

0

$$[A] \rightarrow \begin{array}{r} 1111 \\ 0000 \\ 0 \end{array}$$

0

$$\begin{array}{r} 1111 \\ 0000 \\ 1 \end{array}$$

1

$$\begin{array}{r} 1111 \\ 0000 \\ 10 \end{array}$$

1

$$\begin{array}{r} 1111 \\ 1111 \end{array}$$

1

$$\begin{array}{r} 0000 \\ 0000 \end{array}$$

⑧ In INR will not affect  
carry flag.

Loop will execute  $\infty$  time.

MVI A FOH	FR = 7T	
ORA A	F = 4T	(34)
Loop: INR A	F = 4T	
JNZ Loop	FRR/FR = 10T/7T	
HLT	F = 4T	

If  $f = 3 \text{ MHz}$  then find total time of execution of program.

Total T-state =  $7T + 4T + 15T (14T) + 1(4T+7T) + 4T$   
 $= 236T$   
 $= 236 \times \frac{1}{3} \mu\text{sec}$   
 $= \frac{236}{3} \mu\text{sec}$

### 16 bit Arithmetic instruction:-

1) INX Rp :

IWS = 1 byte

Rp  $\rightarrow$  BC, DE, HL

Content of Reg. pair will increase by 1 and result will store in Reg. pair.

$$[Rp] \leftarrow [Rp] + 1_{LSB}$$

Add. Mode  $\rightarrow$  Reg. Add mode

M.Cy  $\rightarrow$  S (C.F)

2) DCX Rp

IWS = 1 byte

Rp  $\rightarrow$  BC, DE, HL

Content of Rp will decrease by 1 and result will store in Rp

$$[Rp] = [Rp] - 1_{LSB}$$

Note:- INX & DCX will not affect status of flag.

(iii) DAD Rp

$IWS = 1 \text{ byte}$ .

$Rp \rightarrow BC, DE, HL$

Content of Rp will get added with [HL] and result will store in [HL] pair.

→ Reg. Add. mode.

\*\*\* →  $M \cdot Cy = FBB \equiv 10T$

Note → DAD inst. will affect only one flag that is carry flag.

Ques = 1000H : MVI A A1H       $[A] = A1H$   
 1002H LXI H 1007H       $[H] = 10H ; [L] = 07H$   
 1005H SUB M  
 1006H OUT 05H  
 1008H HLT  
 after the execution of this program display the op port

$$\begin{array}{r} (A) = 1010\ 0001 \\ (M) = 00\ 00\ 0101 \\ \hline (A) = \underbrace{1001}_{B} \underbrace{1100}_{C} \end{array}$$

9CH      Ans

(ii) HL pair is add. of M and Content of HL pair will end that add. data present in M.

Ques : LXI H 2000H  
 LDA 2002H      Memory Location      Data  
 XRA M      2000H      00H  
 MOV E,A      2001H      01H  
 MVI D, 20H      2002H      02H  
 LDAX D      2003H      03H  
 OUT 05H      after the execution of this program

$$\begin{array}{l} [H] = 20H, [L] = 00H \\ [A] = [D2]H \\ [A] = 0000\ 0010 \\ [M] = 0000\ 0000 \\ \hline [A] = 0000\ 0010 = 02H \\ \hline [E] = \underline{\underline{02H}} \quad [D] = 20H \end{array}$$

## 8-bit logical Rotational instruction:-

(36)

Execution of these instruction based upon content of [A].

After the execution of these instruction content of accumulator will rotate by one bit either left or right as per instruction.

RLC No operand  $\rightarrow$  content of [A] rotate by one bit left without carry.

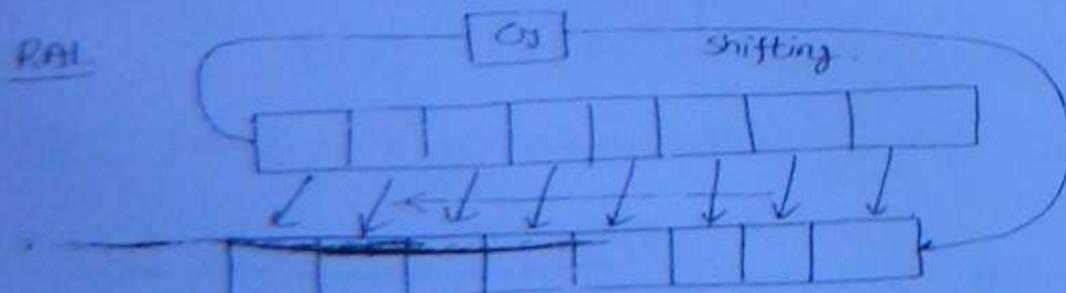
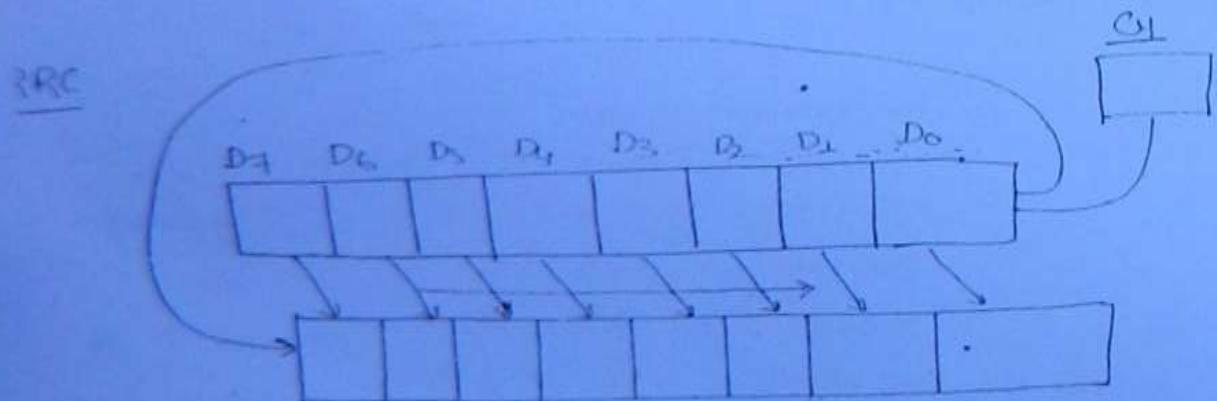
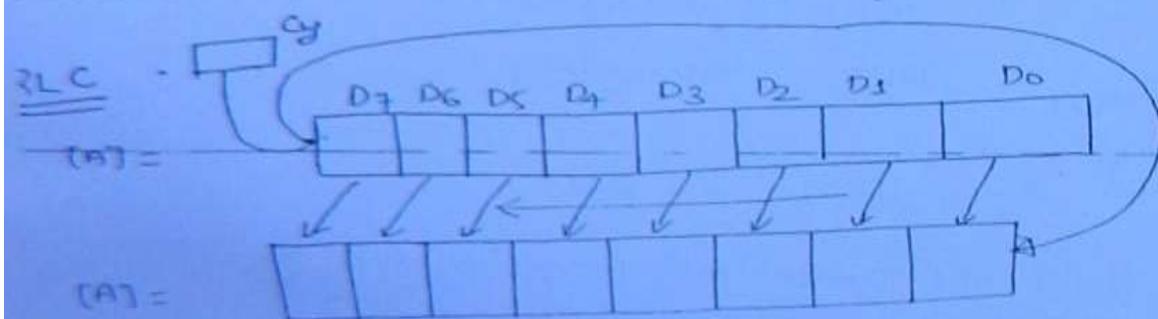
RAL No operand  $\rightarrow$  content of [A] Rotate by 1 bit left with carry.

RRC No operand  $\rightarrow$  content of [A] Rotate by 1 bit Right without carry.

RAR No operand  $\rightarrow$  content of [A] will rotate by one bit right with carry.

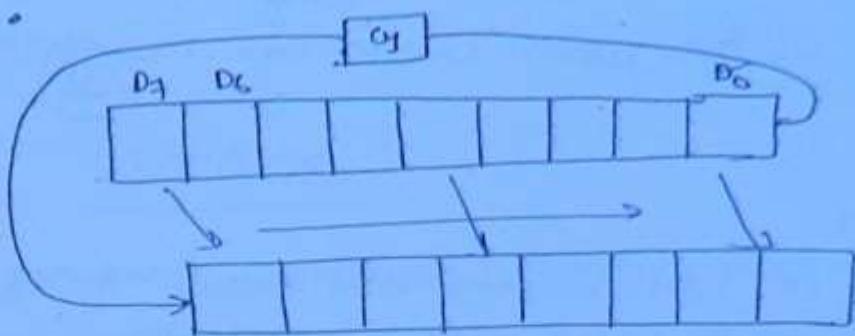
IWS = 1 byte

operation:- As per inst. content of [A] will Rotate by one bit.



RAR

(37)



Note:- Logical rotation will affect only one flag that is carry flag.

→ Implicit Add Mode.

→ M.C<sub>f</sub> = F

Ques:- MVI A - B9H

ADI A2H

RAL

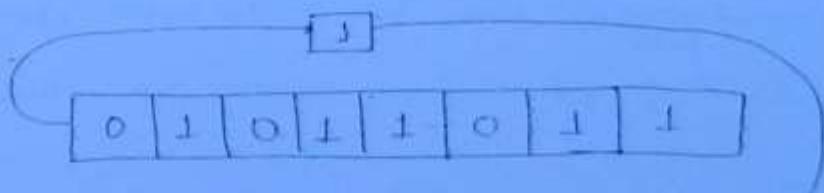
RLC

HLT

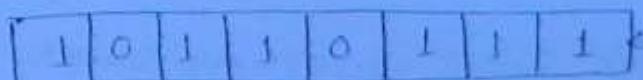
What is the content of Acc & R status of flag.

$$[A] = B9 = 10111001 \\ 10100010$$

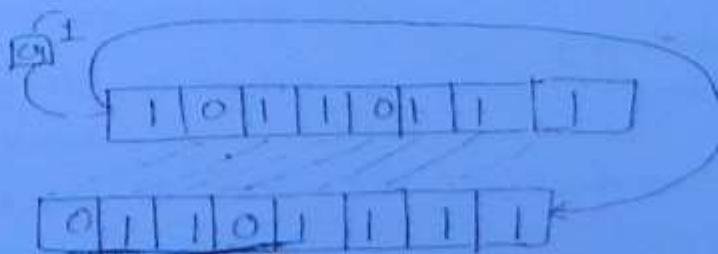
$$[A] = \underline{01011011} \quad [C_f = 1]$$



RAL



RLC



## Logical compare instruction:-

(28)

### i) CMP R

IWS = 1 byte

R = ABCDEHL & M

operation :- Content of [R] will compare with content of [A] and status of flag will affect accordingly.

Note :- Compare inst. is just equivalent to sub. inst. of (A-R) and status of flag will affect according result of (A-R) but result (A-R) will discard it means content of [A] and content of [R] will not change.

	cy	Z
[A] > [R]	0	0
[A] < [R]	1	0
[A] = [R]	0	1

→ Rest of flag affect according the result of (A-R).

$R \neq M$  Register add. mode.

$R = M$  Indirect add. mode.

M. cy

$R \neq M = F$

$R = M = FR$

### ii) CPI 8 bit data

IWS = 2 byte

8 bit data will compare with content of [A] and status of flag will affect accordingly.

→ immediate add. mode

→ no op

Note - CMP & CPI will affect status of all flag

(39)

## STACK

it is group of continuous memory location in main memory that is used for temporary storage of information during the execution of main program.

- Stack is define by instruction LXI SP 16 bit data.
- add. of top of stack always store in stack pointer.
- at a time two byte data can store or retrieve from stack.
- if two byte data stored at the top of stack then it grows upwards in numerically decrease order of its address.
- in 8085 two inst. are define for store or retrieve of two byte data at /from top of stack that is PUSH & POP.

PUSH → store

POP → Retrieve

during the execution of CALL subroutine add. of next instruction will store at the top of stack automatically.

PUSH R<sub>f</sub> :-

1 byte R<sub>f</sub> ⇒ BC, DL, HL, PW

Operation :- when this inst will execute content of R<sub>f</sub> will store at top of stack.

- after the execution of push inst content of stack pointer decrease by 2
- it is data transfer inst so status of flag will not affect
- ~~Register~~ . (Q) Most of no conditional push inst.
- ...

POP Rp

7

IWS = 1 byte.

Rp = BC, DE, HL PSW  
B D H

40

Operation :- when this inst. will execute two byte data will retrieve from top of stack & store in Rp.

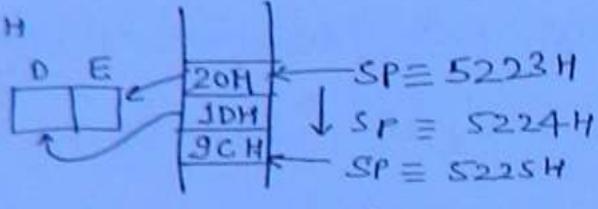
Ex:

LXI SP 5223H

POP D

D = ?

E = ?



∴ After the execution of pop inst. content of sp will ↑ by two.

Add. Mode

POP Rp → indirect add. mode (source data)  
 POP Rp → Registers add. mode (destination data)

M.Cy - FRR.

→ it is data transfer inst. so status of flag will not affect.

Ex:- but in pop psw

status of flag may get affected indirectly.

MVI A, 29H

$$[A] = 29H = 00101001$$

ADD A

$$[A] = \underline{00101001}$$

LXI SP 5986H

$$[A] = \underline{01010010}$$

LXI B CD~~14~~H

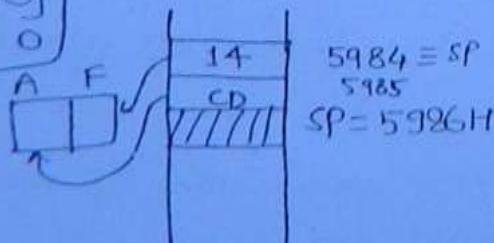
S	Z	AC	P	Cy
0	0	1	0	0

PUSH B

$$[B] = CDH$$

POP PSW

$$[C] = 14H$$



$$[A] = \underline{CDH}$$

$$\rightarrow 14H$$

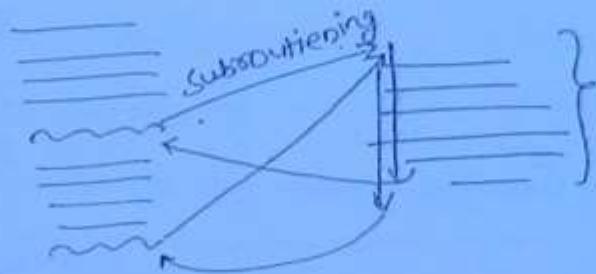
$$0001\ 0100$$

$$0001\ 0100$$

## Subroutine :-

(4)

it is set of inst. that written separately from main programs regarding the task that occur in main program it known as subroutine.



in 8085 two inst. are available for subroutine.

### ① CALL inst:-

#### uncondition call inst:-

CALL 16 bit add.

IWS = 3 byte.

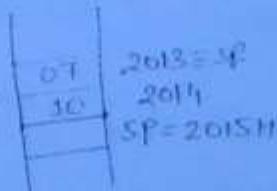
operation - when this inst. will execute control will jump at 16 bit vectors location given in the inst. but before jump add of next inst. will store at top of stack.

Ex: 1000 LXI SP 2015H  
1003 MOVI B C      jump  $\rightarrow$  5920H  
1004 CALL 5920H

1007 MVI D 18H

PUSH

CALL = PUSH + JMP



Note → After the execution of call inst. content of stack pointer will decrease by two.

Note :- immediate add. mode.

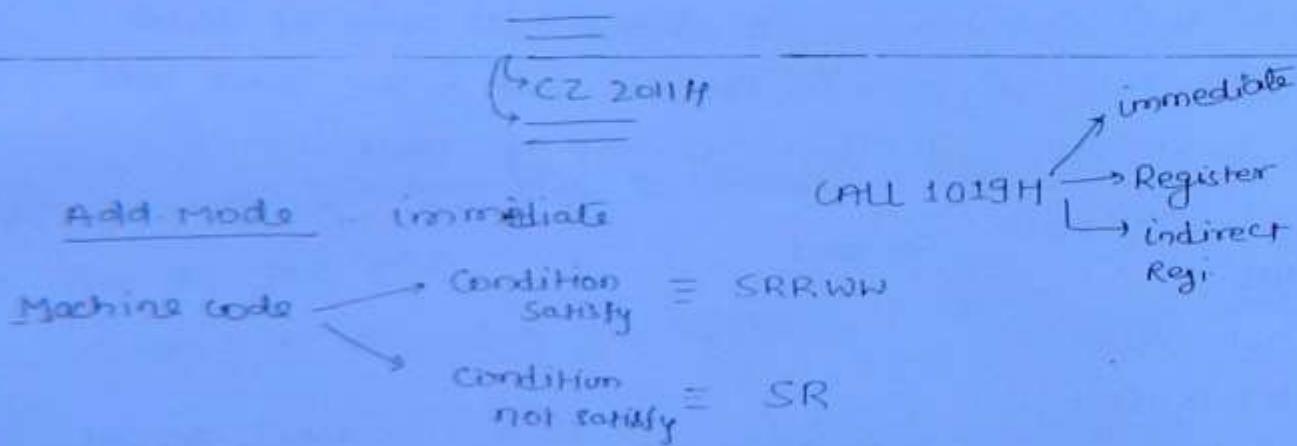
M Oy SRRWW (3BT)

## conditional CALL inst:-

CC	16 bit add.	CALL - CALL inst. will execute if	Cy = 1
CNC	"	"	Cy = 0
CZ	"	"	Z = 1
CNZ	"	"	Z = 0
CP	"	"	S = 0
CM	"	"	S = 1
CPE	"	"	P = 1
CPO	"	"	P = 0

IWS = 3 byte

~~if condition satisfy then CALL inst. will execute  
if condition not satisfy then skip it.~~



- Note
- ① CALL inst. is data transfer inst. so status of flag will not affect
  - ② CALL inst. (conditional or satisfy case (or) unconditional) is the largest of inst. of 8085 & it take 1BT.

Ques

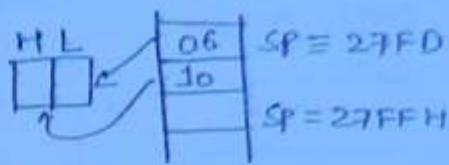
1000: LXI SP 27FFH

1003: CALL 1006H

~~pop~~

	SP	HL
①	27FF	1006
②	27FD	1006
③	27FF	1003
④	27FD	1003

17 SP = 27FF  
HL = 1006



### RET inst :-

- unconditional
- conditional

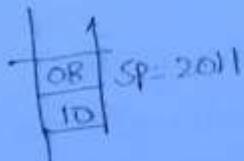
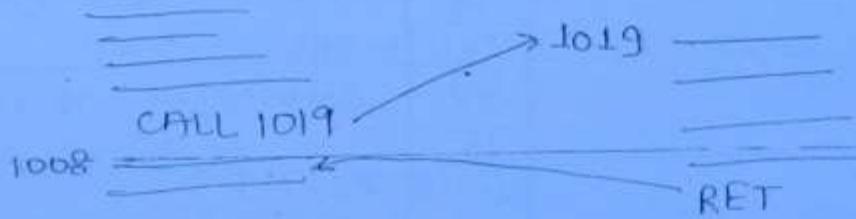
Note → it is last inst. of every subroutine.

→ RET no operand.

IWS =

operation:- when this inst. will execute

- ① two byte data will retrieve from the top of stack & store in program counter.
- & next inst. fetch from this vector location.



Add. Mode -

- implicit
- indirect ✓

M<sub>cy</sub> = SRR

RET = POP + JMP

Note:- After the execution of RET inst. content of SP will ↑ by two

Q: 1000H: LXI SP 2719H  
 1003, CALL 3000H  
 1006 \_\_\_\_\_

44

3000H: LXI H 19H

PUSH B  
 PUSH H  
 PUSH PSW  
 LXI SP 3C82H  
 POP PSW  
 POP H  
 POP B  
 RET

After the execution of program

SP = 2719

H=19, L=19

SP = 3C82H

F	← 2711
A	
value of SP = ?	
19	← 2713
[C]	← 2715
[B]	← 2717
06	
10	
	SP = 2719

PSW	{       } 3C82-
HL	{ } ← 3C84
BC	{ } ← 3C86
DC	{ } ← 3C88
	{ } ← 3C8A

main program      ↳ subroutines

(45)

```

1002 MOV B C      5985: MVI A, 00H
1003 CALL 5985H   5987: CALL SUB1
1006 MVI D 29     (598A) SUB1 : INR A.
                    ↓
RET

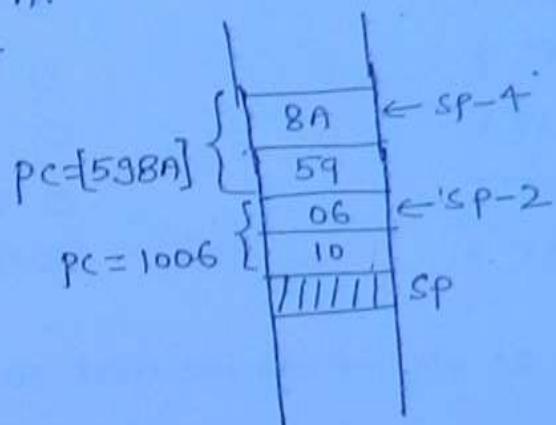
```

[A] = ?

[A] = 00H

[A] = 01

↓  
[A] = 02



### RESTART

it is just like as one byte CALL inst.

it is used when execution wraps in interrupts.

RST N no operand

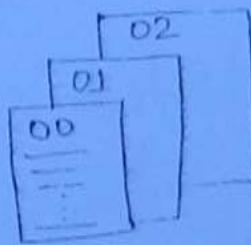
INSE 1 byte

N = 0, 1, 2, 3, 4, 5, 6, 7.

operation:- when this inst. will execute control will come at specific vector location of memory page no. 00h

page no. line no.

XY WZ H



Note:- RST is software interrupt

<u>inst.</u>	<u>vector location</u>
RSTN	00XXH
RSTO	0000H
RST1	0008H
RST2	0010H
RST3	0018H
RST4	0020H
RST5	0028H
RST6	0030H
RST7	0038H

(46)

Note :- XX is the hexa decimal converter of  $(N \times 8)$

Add. Mode - implicit :-

M.Cy = S

Advance Arithmetic inst :-

ADC R

3WS = 1 byte

R = A, B, C, D, E, H, L & M

content of R will get added in content of Accumulator  
with status of carry flag

$$[A] \leftarrow [A] + [R] + [Cy]_{LSB}$$

$$\begin{aligned} [A] &= 25H \\ [B] &= 10H \end{aligned} \quad \left\{ \text{Cy} = 1 \right.$$

ADC B

$$[A] = 0010\ 0001$$

$$[B] = 0001\ 0000$$

$$[D] =$$

Add Mode -  $R \neq M$  Register add.  $\eta$

$R = M$  indirect.

(47)

M.Cy  $R \neq M \equiv F$   
 $R = M \equiv FR.$

(ii) ACI 8 bit data.

IWS = 2 byte.

operation:-  $[A] \leftarrow [A] + 8\text{ bit data} + [Cy]_{LSB}.$

$\rightarrow$  immediate add. mode.

M.Cy  $\equiv FR.$

(iii) SBB R

IWS = 1 byte

$R = ABCDEHL\&M.$

Content of R will get subtracted from content of Accumulator with status of carry flag.

$[A] \leftarrow [A] - [R] - [Cy]_{LSB}.$

$R \neq M \rightarrow$  Registers Add. mode

$R = M \rightarrow$  indirect

Machine Cycle  $R \neq M \equiv F$

$R = M \equiv FR.$

(iv) SBI 8 bit data

IWS = 2 byte.

$[A] \leftarrow [A] - 8\text{ bit data} - [Cy]_{LSB}.$

M.Cy  $= FR$

immediate Add. mode.

inst. Related to HL pair:-

i) LHLD 16 bit add.

(48)

IWS = 3 byte.

When this inst. will execute data available at the memory location that is given in east. Will load in L and data available at the next memory location will load in H.

[L] ← [16 bit add]

[H] ← [16 bit add + 1]

LHLD 201CH

[H] = 7BH

[L] = 69H.

69H	201CH
FBH	201DH

M-cy FRRRR

1000 LHLD

1001 LC

1002 20

ii) SHLD 16 bit add

IWS = 3 byte

operation:-

A [L] → [16 bit add]

[H] → [16 bit add + 1]

M-cy (FRRWWWW)

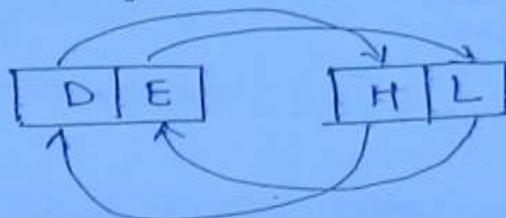
1000 SHLD 2019H

(iii) XCHG no operand.

(49)

IWS = 1 byte.

when this inst will execute content of DE region exchange with  
the content of HL register pair.



Add. mode

- implicit add. ✓
- Register add.

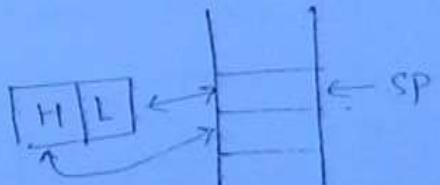
M.Cy F

(iv) XTHL no operand

IWS = 1 byte.

operation :- when this inst will execute content of HL pair  
will exchange from top of stack

XTHL = PUSH + POP



M.Cy F WWR

Add. Mode

- implicit
- Registers
- Indirect Reg.

(v) SPHL no operand

IWS = 1 byte.

when this inst will execute content of HL pair will copy

Stack pointer



Note → This inst. also used for initialising of stack indirectly.

Add mode → implicit ✓  
 → Register

(50)

M.Cy → S.

(vi) PCHL no operand.

1 byte.

When this inst. will execute content of HL pair, will copy in PC.

$$[PC] \leftarrow [HL]$$

Add Mode → implicit ✓  
 → Register

M.Cy → S

Note:- All above inst. are data transfer inst. so status of flag with nor affe

Ques)

MVI B 28H FR FT

NOP F 4T

Loop: DCR B F 4T ... ]

JNZ loop FRR/FR 10T/FT ↳

HLT F 4T Total time elapsed during  
the execution of program.

$$\text{Total } T = FT + 4T + 3(14T) + 1(1T) + 4T$$

$$= 572T$$

$$= 572 \times 0.5 \text{ microseconds}$$

MVI B, 38H	FR	7T	(5)	$(38)_H = (56)_{10}$
Loop1: MVI C, FFH	FR	7T		
Loop2: DCR B	F	4T		$(FF)_H = (255)_{10}$
JNZ loop1	FRR/FR	10T/7T		
DCR B	BPF F	4T		
JNZ loop2	FRR/FR	10T/7T		
operating freq. $\leq 2 \text{ MHz}$				
total time elapsed = ?				

total T-state req. to execute inner loop completely  
once

$$= 255(14T) + 11T = 3567T$$

total T-state for program

$$= 7T + 55[7T + 3567T + 4T + 10T]$$

↑  
 satisfies  
 Loop2      +    ↑  
 not satisfies.

$$= 200932T$$

$$T = \frac{1}{2 \text{ MHz}} = 0.5 \mu\text{s}$$

Ques:-

LXI H, 2041H	FRR	= 10T
MVI A, D6H	FR	= 7T
CMP M	FR	= 7T
MOV B A	F	= 4T
IN 20H	FRI	= 10T
SUB B	F	= 4T
OUT 52H	FRD	= 10T
INX H	S	= 6T
MOV M,A	FW	= 7T
HLT	F	= 4T

(i) How many times CPU executes memory read machine cycle of given program. = 6 times

(ii) How many times CPU performs memory read operation (as) —  
How many times CPU will read data from memory = 12 times

Q) How many times CPU performs read operation or  
how many times RD signal will active low  
during the execution of program.  $\equiv 17$  times.

(52)

N) How many T-state req. of execution of prog  $\equiv 69T$   
3)  $f = 3 \text{ MHz}$ . So total time req.  $= ? = 23 \mu\text{sec}$ .

### Some Advance Inst. :-

i) STC no operand.

I<sub>WOS</sub> = 1 byte.

When this inst. will execute carry flag will set  
regardless of previous status. Cy = 1.

→ implicit add. mode.

→ M.Cy → F.

ii) CMC no operand

I<sub>WOS</sub> = 1 byte

When this inst. will execute status of carry  
flag will get complemented

$$Cy \leftarrow \overline{Cy}$$

Add. mode  $\rightarrow$  implicit

M.Cy  $\rightarrow$  F

Note:- above two inst. affect only carry flag.

iii) DAA

Note:- BCD Numbers

→ Binary coded no. is not binary no.

→ It is binary coded sys decimal

~~DATA~~ No operand

INS - 1 byte

this inst. will execute then content of Accumulator will adjust to BCD format by assuming earlier operation was BCD addition

Add-mode → implicit

M.CY → F

Note:-

- ① if lower nibble of content of Accumulator is greater than 1001 then 0110 get added in it.
- ② if lower nibble of [A] is less than or equal to 1001 and Auxiliary carry flag set the 0110 will get added in it.
- ③ if upper nibble of [A] is greater than 1001 then 0110 is added in it.
- ④ if upper nibble of [A] is less than or equal to 1001 & Carry flag is set then 0110 will get added in it.

Note → this inst. will affect all flags.

6010H: LXI H 8A79H

6013H MOV A,L

6014H ADD H

6015H DAA

60 PCHL

$$[H] = 8A \text{H}, [L] = 79 \text{H}$$

$$[A] = 79 \text{H}, [L] = 75 \text{H}$$

$$[A] = 10111001$$

$$[H] = 10001010$$

$$[A] = 0000\ 0011$$

$$[A] = 0110\ 0110$$

$$[A] = 0110\ 1001 = 69 \text{H}$$

$$[H] = 69 \text{H}$$

S Z AC P CY

0 0 1 1 1

0 0 0 1 0

-MHE

After the execution of PCHL inst. next inst. will fetch from

(A) 6019H

[PC] = 6979H

(B) 0379H

(C) 6979H

(D) none of these

## Interrupts

In 8085, 5 hardware interrupt pins are available

(54)

TRAP → non maskable.

RST 7.5  
RST 6.5  
RST 5.5  
INTR.

} Maskable interrupt.

On the basis of different characteristic interrupts are categorized as:

i) maskable & non maskable interrupt.

Interrupt that can make disable is known as = maskable interrupt.

Interrupt that can not make disable = i. maskable notes IT

In 8085, to control interrupt process there is a flip-flop that is known as interrupt enable flip-flop by set or reset of interrupt enable flip-flop we can make enable or disable of interrupt process.

In 8085 for set/reset two inst are available to make Set or Reset of interrupt enable flip-flop

1) EI no operand

IWS = 1 byte

When this inst will execute interrupt enable FF will get set and interrupt process will enable

Add. mode → Implicit

M. cy → F

2) DI no operand

1 byte

When this inst will execute, interrupt enable FF will get reset and interrupt process will disable

Note:- EI & DI is machine control instruction so status of flag will not affect

### (ii) vectored & Non vectored interrupt:-

55

→ If interrupt is acknowledge & execution transfer of definite vector location of memory page no. 00H then such type of interrupts are known as Vector interrupt.

TRAP }  
RST 7·5 }  
RST 6·5 }  
RST 5·5 }

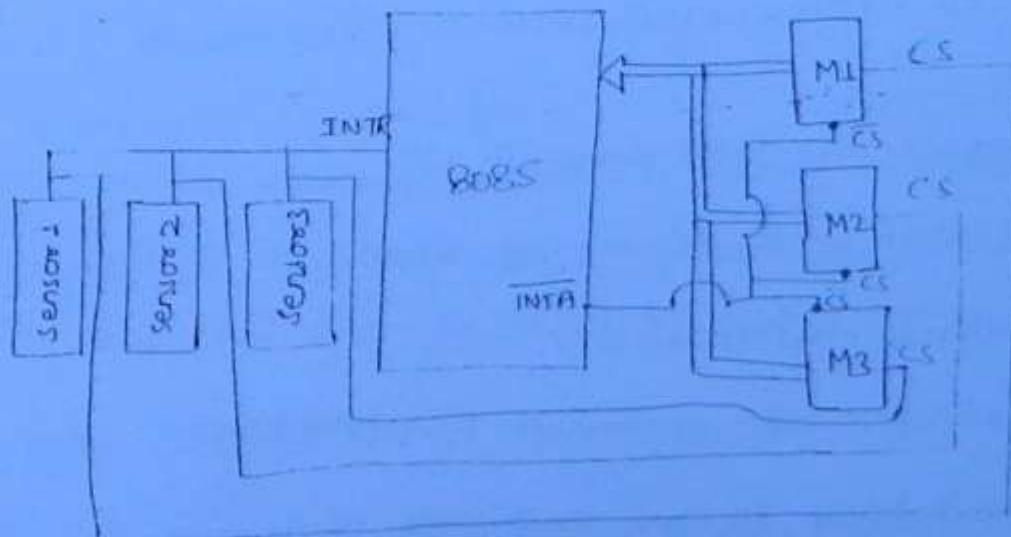
INTR → non vector

Note ① Vector location for non vectored interrupt provided by exterally.

② TRAP is also known as RST 4·5.

*** Vectored interrupt	Vector Location.
TRAP	00 2 4
RST 7·5	00 3 5
RST 6·5	00 3 4
RST 5·5	00 2 5

$$4 \cdot 5 \times 8 = 36 \quad (360)_{10} \quad (16)_{16}$$
$$7 \cdot 5 \times 8 = 60 \quad (600)_{10} \quad (24)_{16}$$
$$16 \times 8 = 128 \quad (1280)_{10} \quad (80)_{16}$$



## ① Triggering :-

TRAP } edge trigg.  
RST 7.5 }  
RST 6.5 }  
RST 5.5 } level trigg.  
INTR }

(56)

② TRAP is edge and level both trigg.

## ③ Priority:-

TRAP      ↑ higher priority.  
RST 7.5  
RST 6.5  
RST 5.5  
INTR      ↓ lowest priority

TRAP has highest priority & INTR has lowest priority.

→ HOLD signal has highest priority in externally initiated signal.

→ TRAP interrupt independent of EI & DI interrupt.

→ maskable & non maskable concept Valid only when interrupt enable bit is set.

When a interrupt is acknowledge following steps execute automatically.

Execution of current inst. will complete 1st.

Ad. of next inst. will store at top of stack.

DI inst. will execute automatically.

Execution will transfer at interrupt service routine.

→ programmer should have to write last inst. of interrupt service routine.

→ ~~minimum time duration~~ for which a interrupt should

## SIM (Set interrupt mask).

- by using this inst, interrupt can be masked.
- serial transfer of data through SOD pin.

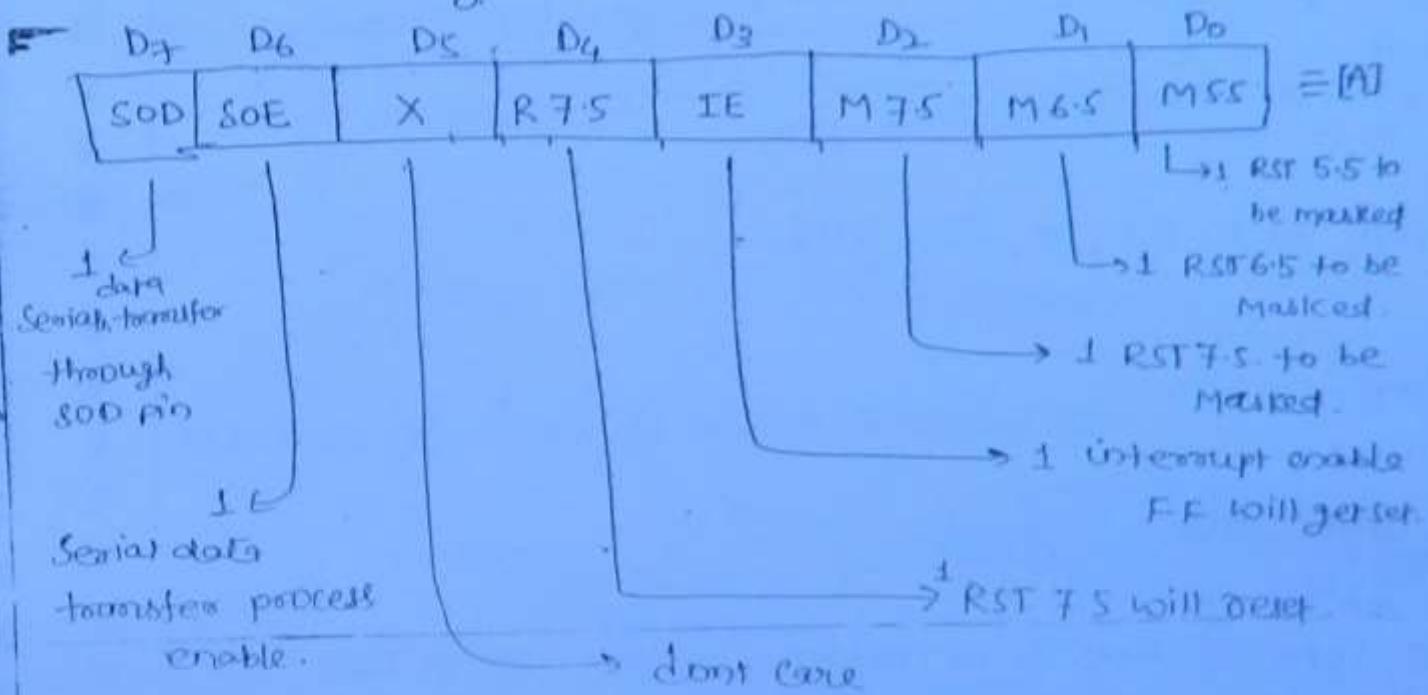
(S7)

### SIM no operand

1 byte

When this inst execute on the basis of content of ACCUM.

and according its characteristic task will perform.



implicit & F.

## RIM (Read interrupt mask)

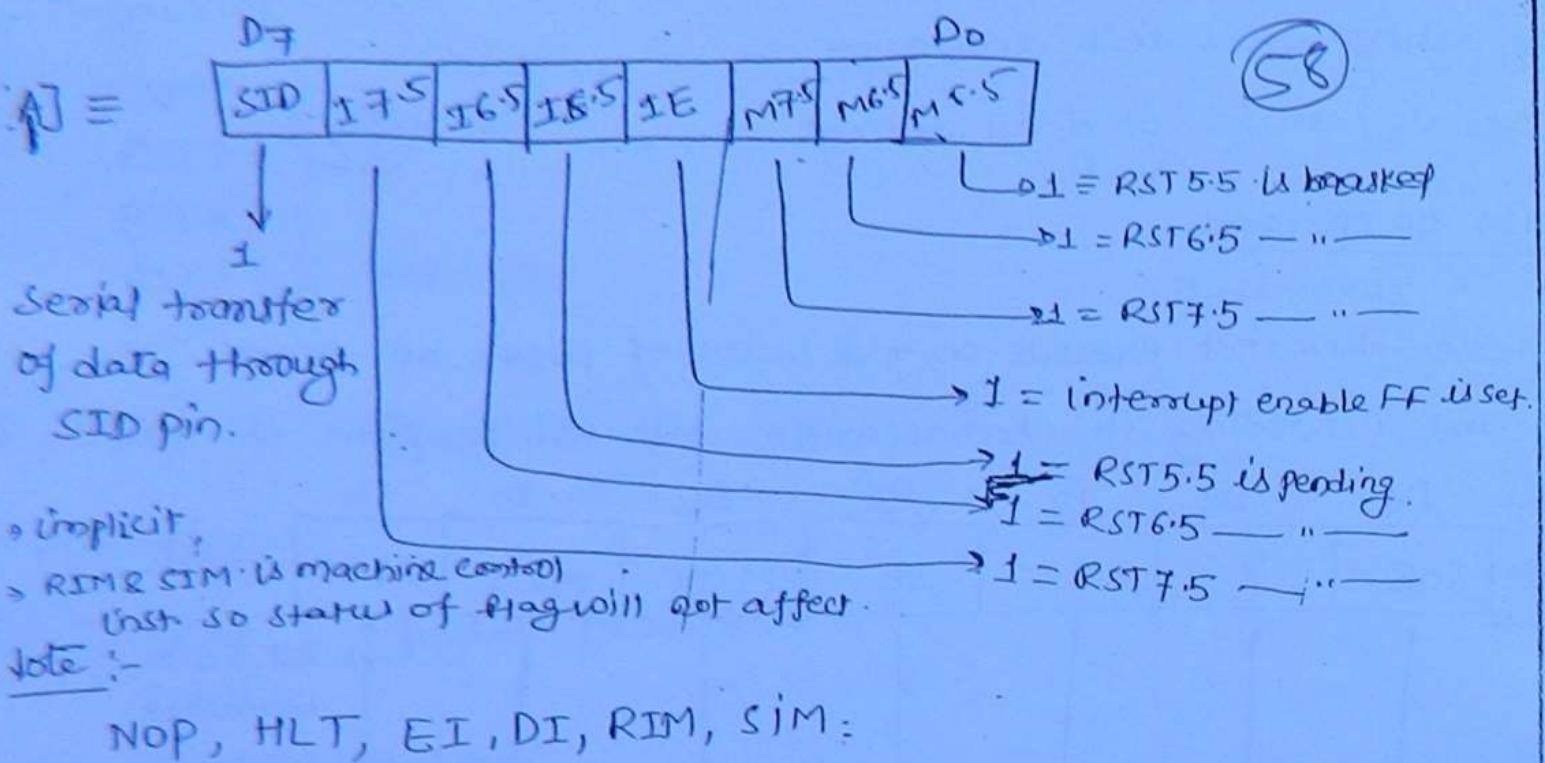
RIM inst is used for

- to find out status of pending interrupt.
- to find out status of masked interrupt.
- This inst is used for serial transfer of data through SOD pin.

### RIM no operand

1 byte

When this inst will execute ACCUM. loaded as per function of RIM inst.



→ EI  
RIM  
ANI 08H.  
SIM

What type of task to be perform by above set of instruc.

- ① Send bit out on DO pin.
- ② Accept bit in from SID pin
- ③ Accept RST 7.5 interrupt
- ④ Reset RST 7.5 interrupt

## Interfacing

(59)

Memory interfacing

I/O interfacing

→ Memory mapped I/O interfacing

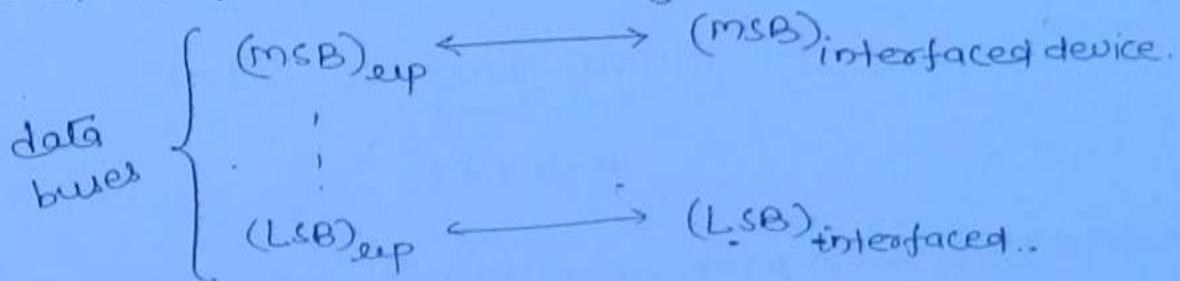
→ I/O mapped I/O interfacing ~~or~~

peripheral mapped I/O interfacing.

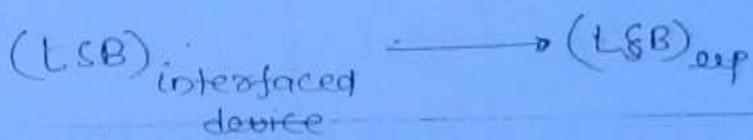
General steps of interfacing :-

Command signal connect directly b/w the interfaced device

~~Data buses~~ connected directly



add. bus of interfaced device connected at



Remaining add. bus of exp is used for development of chip select logic

→ chip selection logic developed by two way

(i) by use of logic gate or buffer OR

(ii) by use of decoders OR

(i) Memory interfacing :-

first write memory in  $2^{\text{nd}}$  format

iii) chip selection by logic gate:-

60

$$2 \times 2^{\frac{1}{2}} \times 8$$

$$2^{14} \times 8$$

001 = 11

$$d_2 a = g.$$

Find out add. range of memory interfaced with esp.

$$\underbrace{A_{15} A_{14} A_{13} A_{12} A_{11}}_{1 \quad 1 \quad 1 \quad 1 \quad 1} A_{10} A_9 A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 = A_{10}$$

min add 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 = F800H.

$$\underline{\text{Max odd:}} \quad \underbrace{1 \quad 1 \quad 1 \quad 1}_{\downarrow} = \text{FFFFH}$$

Q. CS =  $\overline{A_{15}} A_{14} A_{13}$  is used as chip selection logic of 4K byte RAM in 8085 microprocessor

{ 0110000100000000 }  
{ 6000H }  
{ FFFFH }

$$4 \text{K byte} = 2^{12} \times 8$$

$$A_5 A_{11} \cancel{A_{12}} \cancel{A_{12}} A_{11} A_{10} A_9 A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0$$

min. add.

X=0

don't  $x=0$

$0\ 1\ 1\ 0$ $\underline{0\ 0\ 0\ 0}$ $\underline{0\ 0\ 0\ 0}$ $\underline{0\ 0\ 0\ 0}$	$0\ 0\ 0\ 0\ H$  $0\ 1\ 1\ 1$ $\underline{\ 0\ 0\ 0\ 0}$ $\underline{\ 0\ 0\ 0\ 0}$
--	---

$6000H$

$7000H$

(61)

max. add

X=0

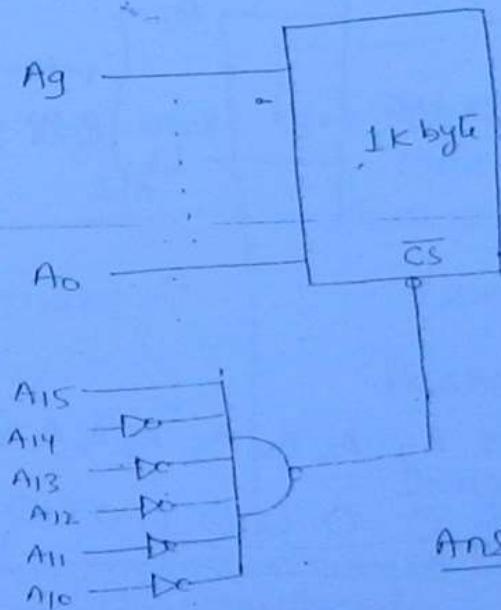
$0\ 1\ 1\ 0$ $\underline{1\ 1\ 1\ 1}$ $\underline{1\ 1\ 1\ 1}$ $\underline{1\ 1\ 1\ 1}$	$6\ F\ F\ F\ H$
--	-----------------

X=1

$0\ 1\ 1\ 1$ $\underline{1\ 1\ 1\ 1}$ $\underline{1\ 1\ 1\ 1}$ $\underline{1\ 1\ 1\ 1}$	$7\ F\ F\ F\ H$
--	-----------------

$6000H - 6FFFFH$  to  $7000H$  to  $7FFFFH$ .  $\leftarrow$

Q



consider a memory chip with 1024 byte storage capacity it is interfaced with 8085 @ 16 bit add. but as shown in fig what will be the memory interfaced range

Ans  $\rightarrow$   $8000H$  to  $83FFH$ .

$1\ 0\ 0\ 0\ 0\ 0\ .\ \underline{a\ 9\ a\ 8\ a\ 7\ a\ 6\ a\ 5\ a\ 4\ a\ 3\ a\ 2\ a\ 1\ a\ 0}$

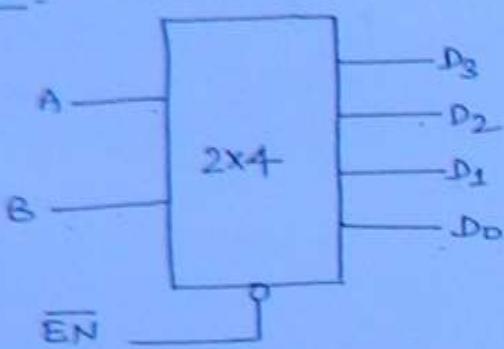
$8000H$       min.add.

$83FFH$

chip selection logic by decoders CKR :-

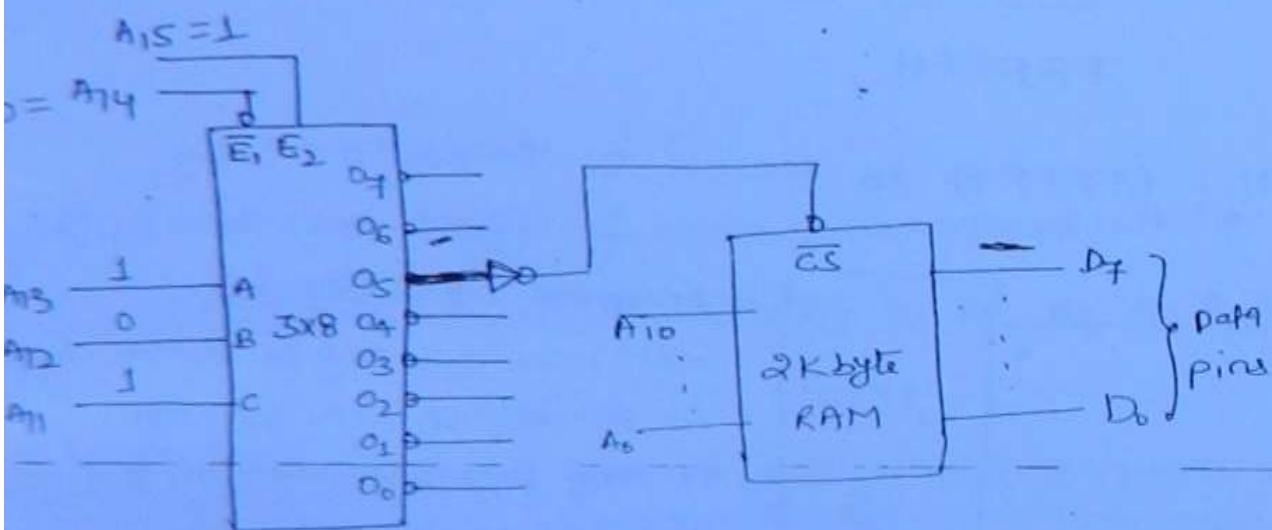
62

Roeder 3



A	B	D <sub>4</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

$$f(AB) = \overline{A}\overline{B}D_0 + \overline{A}BD_1 + A\overline{B}D_2 + AB D_3.$$



find out memory sample

A800H — min. add.

1 0 1 0    1 1 1 1,    1 1 1 1    1 1 1 1

$\Delta E_{\text{FFF}} H = \text{max. abs.}$

A 800H to AFFFH

6

$$\text{Case ①:- } \neg f(A \cdot B \cdot C) = A800 - AFFF@H$$

$$\begin{array}{l} \text{case ②} \\ \text{FCBCA) } \\ \text{101} \\ \backslash \quad | \\ A_{12}=1, A_{13}=1, \end{array}$$

## I/O interfacing →

Memory mapped I/O	I/O mapped I/O interfacing	Characteristics
M-RD   M-WR	I/O RD   I/O WR	Commonal sign
64K byte memory shared b/w internal memory & I/O devices.	256 I/p & 256 O/p maximum.	No. of devices interfaced
All memory related inst. Applicabt. ex MVI, LHLD	IN & OUT	Instruction
Any memory Reg. and I/O	b/w I/O devices & Acc only	b/w Hardw & Acc only Data trans
Hardware more	Less	Hardware
Slower	Faster	Execution
Small	Large	Application

→ All operations performed directly with any reg. but I/O mapped interfacing directly  
Some important I/O peripherals:-

8255 : programmable peripheral interface

8273 = - - - keyboard & display interface

8259 =  Interrupt control

8237 = DMA controller.

8251 = USART

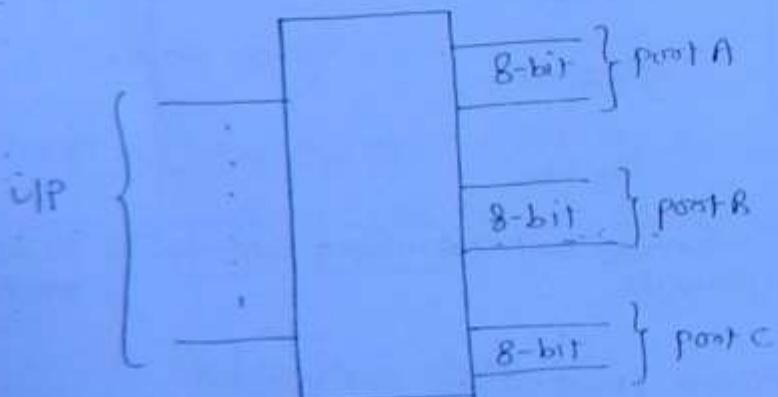
8155 - Programmable I/O port & timer (CPLD)

8254/8253 → programmable  
interval timers

- PSU - Some important digital IC.
- { 74182 → look ahead carry generator. (64)
  - 74180 → 8-bit parity generator & checker.
  - \*\* 7477 → seven segment display decoder.
  - 7493 → 4-bit binary counter
  - 7490 → decade counter.
  - { 7400 → Quad 2 I/p NAND gate.
  - 7402 → Quad 2 I/p NOR.
  - 7408 → ----- AND.
  - 7432 → ----- OR
  - 7486 → ----- EX-OR.

- it is 40 pin IC.

--- input port add is 8-bit



it has three port, port A, B, C

All three port can be used

port C can also work as port C' upper

and port C' lower having port add of 4bit (lower nibble & upper nibble)

~~port C upper & port C' lower can also generate control~~

it has two mode of operation  
↳ bit set Reset mode  
↳ I/O mode.

(65)

### I/O mode

- mode '0' General mode
- mode '1' Handshaking mode
- mode '2' Bidirectional mode.

#### mode '0' -

In this mode of operation port 'A', port 'B', port 'C' can work as I/O port individual.

#### Mode '1'

In this mode of operation port 'A' & port 'B' will work in handshaking mode.

port 'C' upper & port 'C' lower will generate control signal for port A & port B respectively.

#### mode '2' :-

In this mode of operation port 'A' works in bidirectional mode only & port C Upper generates control signal for it.

(Q22)

P-147

A<sub>7</sub> A<sub>6</sub> A<sub>5</sub> A<sub>4</sub> A<sub>3</sub> A<sub>2</sub> A<sub>1</sub> A<sub>0</sub>

1 1 1 1 1 X 0 0

F8-FFH

1 1

X=0

F8 → min add. { for 00  
FB → } for 11

X=1

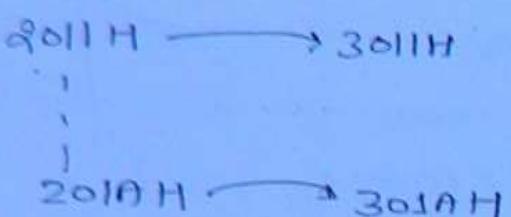
FC → { for 00

FF → { for 11

F8-FB → FC → FF = F8-FFH

(a)

Write a ALP of 8085 to transfer 1 byte data from memory location starting from 2011H to 3011H as



(66)

Solution

LXI B 2011H ;

LXI D 3011H ;

MVI H OAD ;

Loop: LDXA B .

STAX D

INX B

INX D

DCR H

JNZ Loop.

HLT

- Q: write a 8085 ALP to add a 16 bit no in locations 5000H (higher byte) and 5001C (lower byte) with another 16 bit no. Store in 5002 (higher byte) & 5003 (lower byte) & find result store in BC.

LDA 5000H

MOV H, A

LDA 5002H

MOV L, A

LDA 5002H

MOV B, A

LDA 5003H

DAD B

LXI SP 1234H

PUSH H

POP B

## introduction of 8086

### comparison b/w 8088 & 8086

#### 8085

it is 40 pin IC

$f = 3 \text{ MHz}$

$V_{CC} = +5V$

Based upon NMOS

8-bit op

Add. line - 16

data line - 8

max. memory location that can be interfaced =  $2^{16}$

size of flag register = 8 bit

no. of flag = 5

normal memory concept is used

#### 8086

40 pin IC

$f = 5 \text{ MHz}$

$V_{CC} = +5$

Based on HMOS tech

16-bit op.

Add. line 20

data line 16

max. memory location that can be interfaced =  $2^{20}$

size of flag register = 16 bit

no. of flag = 9

Memory segmentation concept is used.

## ARCHITECTURE

internal Arch. of 8086 divide in two parts

- Bus interface unit (BIU)
- Execution unit (EU)

### Bus interface unit (BIU)

This part manage all type of data / add. flow over bus. In BIU, a queue of six location ex. exist & during the execution of one inst. in execution unit next inst. can fetch & move to memory

(67)

7

Queue based upon first i/p - first o/p (FIFO). (68)  
All register that available is BIU size of 16 bit  
EU (Execution unit).

- All type of Arith. & Logic unit present in this part.
- All ~~type~~ of data execution will perform in EU.
- Reg. present in EU can also work as 8 bit & 16 bit both type.
  - AX = 16 bit reg
  - AH = 8 bit
  - AL = 8 bit
  - :
  - AH - AL = AX

Note → i/p o/p port add. is 8086 of 16 bit

- max. no. of i/p & o/p devices can be connected =  $2^{16}$ .
  - max. memory that can interfaced =  $2^{20}$  bytes  
= 1 M byte
- & 16 bit data will store in two continuous memory location.
- in 8086 CPU can read two byte data from memory either one m.cy or in two m.cy.
    - if 1st byte at even add then CPU required only one m.cy to read 16-bit data from memory.
    - if 1st byte at odd add then CPU required two m.cy to read 16-bit data from memory.
  - But in case of 8088 to read 16-bit data from memory always two m.cy required to read data from

but it process 16 bit data at a time so it is also called externally 8 bit & internally 16 bit esp.

(69)

### Memory Segmentation:

- 1 Mbyte memory divide in 4 segment size of 64 Kbyte in continuous memory range.
- Segments can be define in anywhere inside the 1Mbyte memory.
- every code segment is used for definite type of information storage.
- code segment  $\rightarrow$  (fixed) inst. set
- stack segment  $\rightarrow$  temporary storage of information during the execution of main program
- data segment } data storage
- extra segment }
- Regarding every segment there is 16 bit reg. that will hold top 16 bit bottom add of that particular segment
- Bottom add. of every segment select in such away that last four bit always zero.

Code Segment :	CS	16 bit reg.
Stack Segment :	SS	
Data Segment :	DS	
Extra Segment :	ES	

- Regarding each segment another 16 bit reg. that will hold 16 bit add. of memory range 0000H to FFFFH (64Kbyte Range) & content of this reg. is also known as effective add.

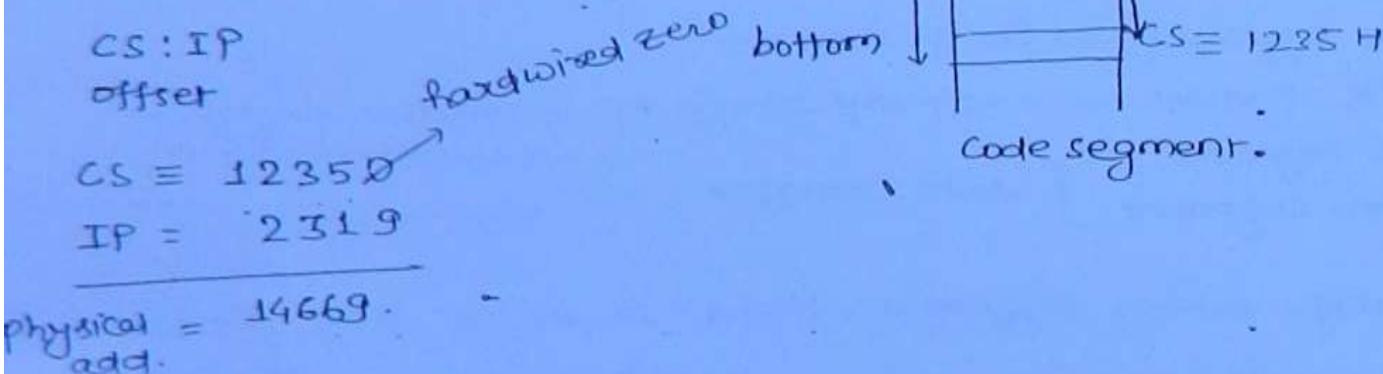
code segment  $\rightarrow$  IP (instruction pointer)  
 stack segment  $\rightarrow$  SP (stack pointer)  
 Data Segment  $\rightarrow$  SI (source index)  
 Extra segment  $\rightarrow$  DI (destination index)

(70)

Physical Add.

IP of 8086 ~~is out~~

Analogous to program counter 64Kbyte  
of 8085.



physical add. also known as, memory add or memory location add.

Flag Registers :-

Size - 16 bit

- in 8086, 9 flag are define that can be divide two groups.
- ①  $\rightarrow$  conditional flag  $\rightarrow$  carry flag, auxiliary carry flag, sign flag, parity flag, overflow flag, zero flag.

Note - All 5 flag of 8085 is exactly same in 8086

$\rightarrow$  process control flag / machine control flag = 3

Toop flag, interrupt process control flag,

Stop/direction flag.

X	X	X	X	O	D	I	T	S	Z	X	A	C	X	P	X	G
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

61

Gy - Coony flag.

P - porosity flag.

A-C - Auxiliary Carry Flag

Z - zero flag

S - sign flag

T - Trap flag.

I - interrupt process control.

D - String direction.

O - overflow.

## Addressing mode :-

- Addressing Mode :-

  - Form of effective add. given in the inst. is known as add. mode.
  - Reg. addressing mode → if effective add. of data given in the form of Register as operand.

Note - Reg. add mode is also Indirect add. mode

Ex: mov A [BX]

immediat add mode :- If data itself given in the inst.

Direct add mode :- If effective add itself given to the inst.

\* Registers Relative add mode: Effective add. of data can be produced by adding 8 bit or 16 bit displacement no.

$[BX]$     {  
 $[DI]$     } + 8 or 16 bit displacement acc  
 $[SI]$     }

Ex Base index add. :-

$$EA = [Bx] + [Dz] \quad \{$$

72

Relative Base index adol. :-

$$EA = [BX] + \underbrace{[DI]}_{\text{or } [SI]} \} + \begin{array}{l} 8 \text{ bit or } 16 \text{ bit} \\ \text{displacement no.} \end{array}$$

Gate

4

β = 87%

A = 8千M

$$A = \text{ooH}$$

$$\zeta \stackrel{?}{=} 0$$

$$\begin{array}{rcl} A & = & 0000 \quad 0000 \\ B & = & 1000 \quad 0111 \end{array}$$

$\begin{matrix} A & = & 0000 & 0000 \\ B & = & 1000 & 0111 \end{matrix}$

$$A = \overbrace{1000}^1 011$$

5 = 1

三

87-1

MVIA OOH

loop ADD B

DC R C

JNZ loop

四

0100 : LXI SP DOFF

0103 LXI H 0107

Q106 MVI A 2011

SUB M

$$HL \equiv 0.67 \equiv 90\%$$

[A] = 20H = 00100000

## AR L Inst

(23)

- ① affect all flag  $\rightarrow$  ADD, ADI, SUB, SUI, ADC, ACI, SBB, SBI, DAA, CMP, CPI, ANA, ANI, ORA, ORF, XRA, XRI.
- ② affect only 4 flag  $\rightarrow$  INR & DCR.  
cy will not affect
- ③ affect only carry flag  $\rightarrow$  STC, CMC, DAD,  
Logical rotation inst.
- ④ not affect any flag  $\rightarrow$  CMA, INA, DCX.

6<sup>th</sup>

```
ORG 7000H
7000H BEGIN LXI H, 7000H
        MOV A, L      [A] = 00H
        ADD H          [A] = 70H
        JP END         [A] = 70H = 01100000
        RSTO
END    PCHL
        HLT
```

74

The End