* Introduction to 8085 Microprocessor :-

769000 operation/sec

1- 8 bit M.P
size of data ~~bits~~ bus

2- Address bus => 16 bits

3- The 8085 micro processor operates on single +5 volts power supply

4- It can operate with clock freq. of 3MHz

5- It provides 8 bit I/P & O/P addresses that means 256 I/O addresses devices

6- The address of 8085 is devided into two parts

(i) Hyer order bus $A_{15} - A_8$

(ii) Lower order bus $AD_7 - AD_0$

7- $AD_7 - AD_0$ is multiplex for datalines and address lines in order to reduce the no. of ~~pins~~ ext. pin in the 8085 MP

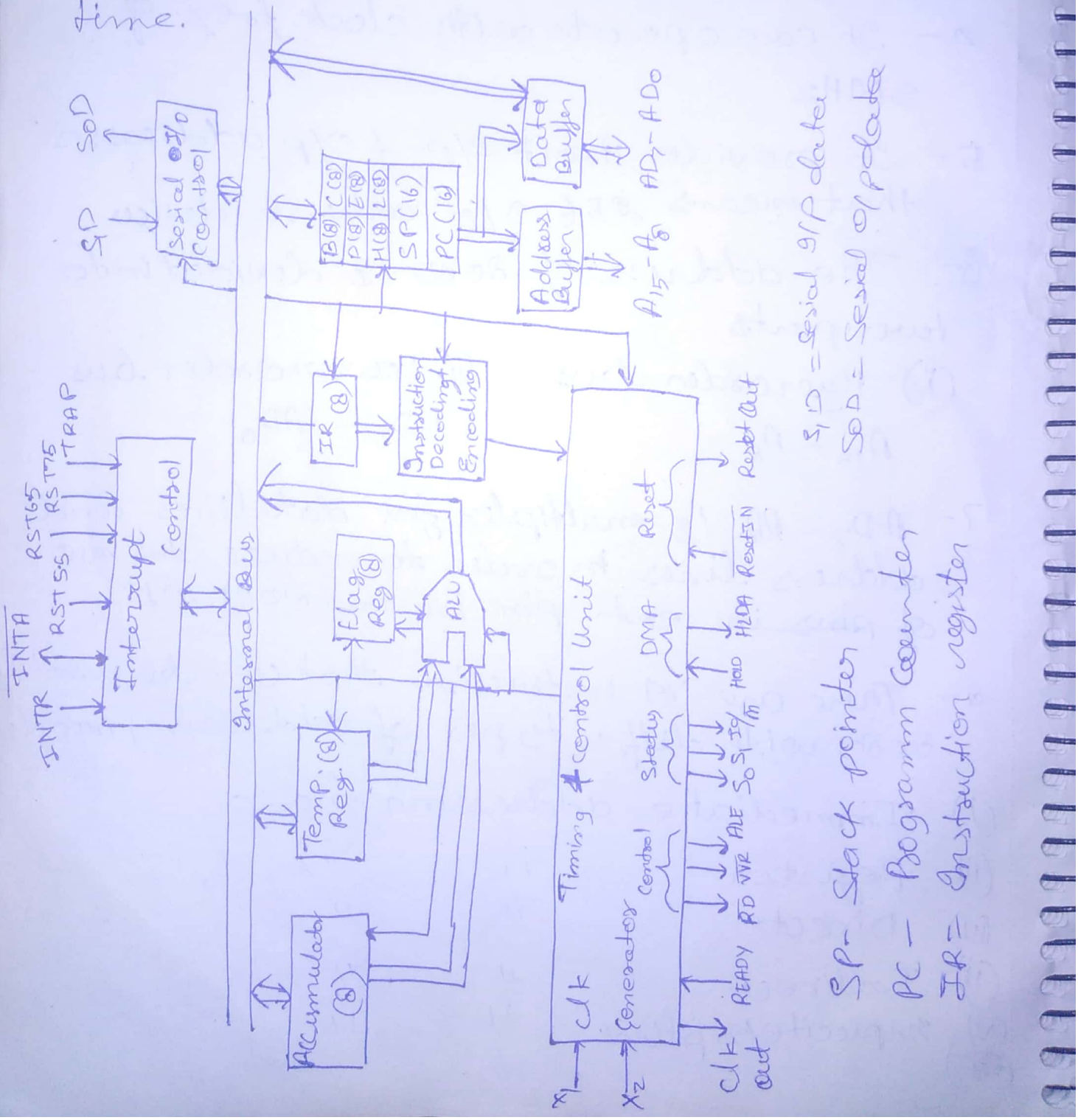
8- There are 74 instruction that can be used by 8085 with diff. types of addressing modes

(i) Immediate addressing mode
(ii) Register " "
(iii) Direct " "
(iv) Indirect " "
(v) Implecite/Implied " "
(or)

Implicit Mode :- Addressing Modes :-

(i) Provides the flexibility to the programmer in which we can write sequence of the inst. as per our choise in order to perform certain kind of operation

(ii) Different addressing modes have different lengths. so programmer can use shorter inst. in order to reduce the execution time.



Major Block of 8085

1- ALU
2- Special Purpose Register (A, T, F, IR)
3- General Purpose registers (B, C, D, E, H, L)
4- Timing & Control register
5- Interrupt Control
6- Special I/O Control

ALU

1- It is used to perform arithmatic & logical operation, Add., sub., compare, increment decrement
2- Logical => AND, OR, XOR, comp and ~~other~~
3- other branching inst

2- Special purpose Register:-

Accumulator:-

8 bit register
Primarily used to stored the first data inst whenever we are performing arithmetic or logical operation
- Mostly the o/p of these operation is also stored in accumulator

IR:-

8 bit Register which is used to stored opcode of any instruction
Max no. of opcodes = 256

General purpose Registers :-

B, C, D, E, H, L are general purpose 8 bit registers which are used to stored temp data while performing arithmetic and logical operation

* In order to operate 16 bit data these registers can be used in register pairs (BC, DE, HL)

HL :- It is the most important register pair as it can also be used as a default memory pointer

1st bit stored in 1st register and 2nd as in second register

* 16 Bit Register :- PC & SP

PC :- It is a 16 bit register which is used to hold the memory address the microprocessor uses program counter to sequence the execution of instruction

The funct of PC is to point the memory location from which the next Inst. is to be fetched when a bit is being fetched the program counter is incremented by one to point it the next memory location

SP :- This 16 bit register to hold memory address of a stack. It points to a memory location in ~~read~~ read write memory which is the starting address of a stack

*Flag Register :- It is 8 bit register special purpose. which is used to hold the status of any arithmetic or logical operation after their execution

There are five different status flag in flag register

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	X	A_C	X	P	X	CY

Sign flag :- After the executing of arithmetic or logical operations. If bit D_7 is the result of 1 then sign flag is set and if D_7 is 0 then sign flag is reset

Set sign flag represent -ve no./result
Reset " " " +ve no/result

Zero flag :- It is being set when the result of any arithmetic or logical operation is all zero

E=1 All zero result
E=0 Many zero result

Auxilary Carry result:- when a carry is being generated from bit D_3 to D_4 or lower level to higher level

Auxilary carry flag is not accessable to programmar and this kind of flag is respected to ~~DCD~~ BCD no.

Parity flag :-

Set on even no. of 1
Reset on odd no. of 1

Carry flag:- whenever the carry generated after 6th bit D_7 after performing the operation. Then this flag is set otherwise it is reset

Ex (1) Perform these following operation in binary and find the states of each flag and flag register.

1- $(27)_{10} - (54)_{10}$ 2's comp.

2- $(35)_{10} + (69)_{10}$

Sol^n 2:-

~~0 0 1 1 0 1 0 1~~

0 0 1 1 0 1 0 1
0 1 1 0 1 0 0 1
1 0 0 1 1 1 1 0

0 0 1 0 0 0 1 1
0 1 0 0 0 1 0 1
0 1 1 0 1 0 0 0

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	X	0	X	0	X	0

1- 00011011 (27)

00111110 (54)

11001010 (-54)

```
  00011011  (27)
+ 11001010  (-54)
------------
  11100101
```

```
  00011010
+       1
------------
  00011011
```

(X)

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	X	1	X	0	X	0

3- $(27)_{10} + (54)_{10}$

```
  00011011
  00110110
------------
  01010001
```

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	X	1	X	0		

Control & Status Symbols:-

1- ALE => Address Latch Enable

(i) This is control signals. This +ve going pulse generated every time 8085 begins and operations

~~2 RD~~

~~3 WR~~

(ii) It indicates that bit AD_7- AD_0 are address bits not data bits

(iii) This signal is primarily used to latch the lower order multiplexed address bus to

that $A_0 - A_{15}$ can be combinedly used as an address bus

2- $\overline{RD}$ ⇒ Read bar :-

This is the loco level control signal. which indicates that selected I/O or memory is being ~~red~~ read and data is available on data bus

3- $\overline{WR}$ ⇒ write bar :-

This the loco level control signal. which Indicats that data on the data bus is to be written either in memory or I/O

Status signals :-

(i) IO/$\overline{M}$:-

This signal is used to select or indicate the data is being stored or read either from the memory or IO device. If it is 1 then IO is used if it is zero then it is a memory operation

-ve cycle	IO/$\overline{M}$	S_1	S_0	control system
Opcod fetch	0	1	1	$\overline{RD}$ =0
Memory Read	0	1	0	$\overline{RD}$ =0
Memory write	0	0	1	$\overline{WR}$ =0
I/O read	1	1	0	$\overline{RD}$ =0
I/O write	1	0	1	$\overline{WR}$ =0
Interupt Ack	1	1	1	$\overline{INTA}$ =0
HLT	Z	0	0	High impedance state
HOLD	Z	X	X	
RESET	Z	X	X	

* Interrupt:-

(i) These signals are used to make microprocessor respond to high priority externally initiated signals

(ii) when an interrupt signal is detected by processor it suspend the execution of current program (as per priority) and execute the program corresponding to an interrupt signal known as Service routine

There are five interrupt signals:-

1- INTR ⇒ This the general purpose interrupt which is an active high I/P signal This interrupt is having the lowest priority out of all 8 interrupt

2- Restart Interrupt:- All RTS's : These interrupt are vectored interrupts. The transfer the programm control to specific memory location. These interupts have higher priority are compared to INTR

RST 7.5
RST 6.5
RST 5.5

3- TRAP ⇒ This is non mackable interrupt with highest priority. This is vectored. This interrupt cannot be stop by any command by the microprocessor

$\overline{INTA}$ ⇒ This a low level o/p signal which is an interrupt acknowledgement generated from the processor which is reply to INTR interrupt request.

Reset Signals:-

$\overline{RESET\ IN}$ ⇒ low level signal

RESET OUT ⇒ High level signal

READY ⇒ It is a signal that serves to delay the processor read/write signals until a slow peripheral is ready to send or receive the data.

If this signal goes low then processor is allow to wait for integral no. of clock cycles until ready signal becomes high

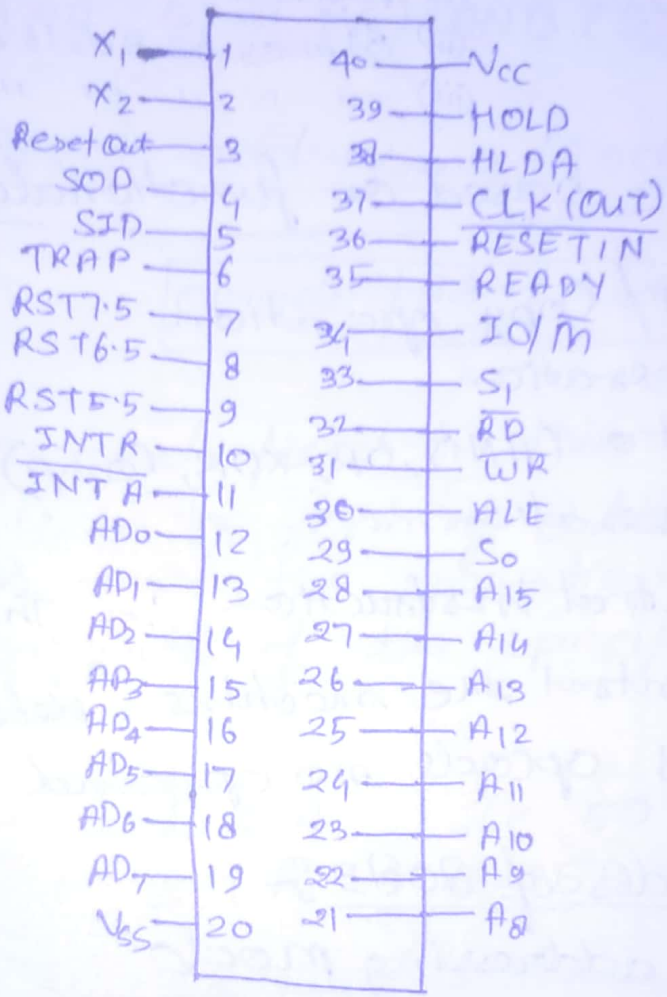
DMA Signals :- Direct memory Access

HOLD ⇒ It is Active high i/p signal from external periferal (DMA Controller) which is requesting to use address and data bus. After receiving the hold request microprocessor releases address bus & data bus in subsequent cycle.

HLDA ⇒ Hold Acknowledgement
This is an active high o/p signal which indicates that processor is ready to release address bus and data bus

Processor regain the control of buses again when hold signal goes low

Que Draw the pin diagram of 8085 micro processor and briefly explain serial control signal



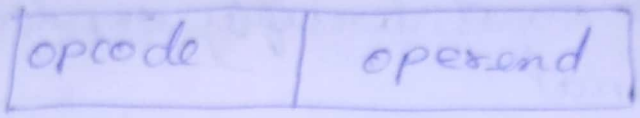
8085A

SID ⇒ Data line for serial input

SOD ⇒ Data line for serial output

* Assembly language programming of 8085 :-

1- Instruction format :-



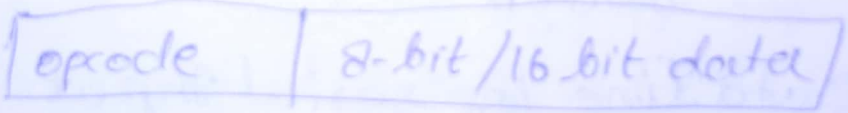
(i) Register
(ii) It may be 8 or 16 data
(iii) " " " " " address

2- classification based on functionality :-

(i) Data transfer / Copy operation
(ii) Arithmetic operation
(iii) Logical operation (AND, OR, XOR, Comp)
(iv) Branching operation
(v) Mechine control instruction :- Those instruction control the mechine, Related to mechine, only opcode no opersend

Addressing Modes of 8085 :-

(i) Immediate addressing mode



Ex (i) MVI B, 45H ⇒ B ← 45
(ii) ADI 05H ⇒ A + 05 ⇒ A
(iii) ANI 15 H ⇒ (A) AND (15) ⇒ A

In the immediate mode inst. the operand specified is ~~in~~ ~~8 bit~~ always in the form of 8 bit & 16 bit data

(ii) Register addressing Mode :-

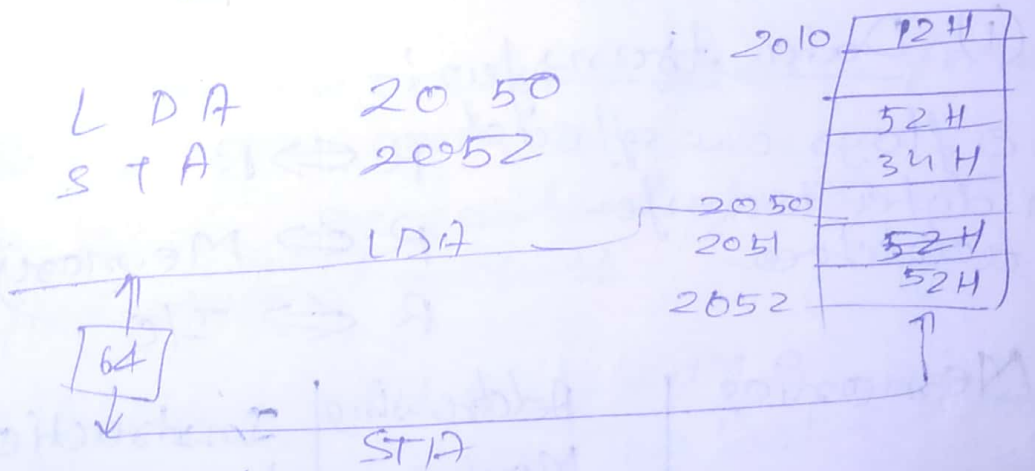
Opcode	Register

(i) ADD B ⇒ A + B ⇒ A
(ii) MOV B, D ⇒ B ← D
(iii) ANA D ⇒ (A) AND (D) ⇒ A

(iii) Direct addressing Mode :-

opcode	8 bit / 16 bit address

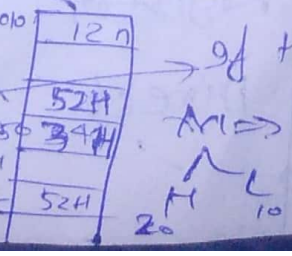
The operand field in this type of addressing mode is in the form of 8 bit or 16 bit address and the operation is perform on the contents of the specified memory location



(iv) Indirect addressing Mode :-

opcode	memory pointer

(i) ~~ADD~~ MOV B, M
(ii) ADD M ⇒ A + [34] ⇒ A
(iii) Sub M



If HL = 2050
M ⇒ Memory pointer
H L
20 10

(2) Implicit or Implied addressing Mode :-

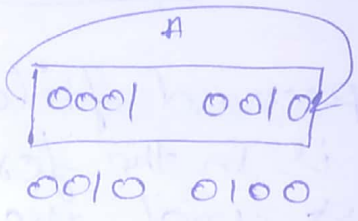
In this type of add. mode the operand are specified implicitly in ~~the inst~~ the opcode of the inst.

opcode

Ex CMA ⇒ Compliment contents of acc.

CMC ⇒ Compliment carry bit in flag register

RLC } Rotate
RRC }



Classification based on functionality :-

(1) Data transfer :-

no flags are affected in data transfer addressing

R ⇔ R
R ⇔ Memory
R ⇔ I/O

	Memonics	Addressing Mode	Instruction Length	Example
	MVI, R, 8 bit	Immediate	2-byte	MVI C, 30H
(2)	LXI Rp, 16 bit Load extended immediate	Immediate	3-byte	LXI B, 2100 21 → B, 00 → C
(3)	MOV RD, RS	Register	1-byte	MOV B H
(4)	LDA, 16 bit Load acc. contents from specified memory location	Direct	3-byte	2500 H

STA 16bit Store acc. contents at specified mem. loc.	Direct	3-byte	STA 2700H
LHLD, 16bit	Direct	3-byte	LHLD, 1525H
SHLD 16bit	Direct	3-byte	SHLD, 1600H
MOV Rd, M	Indirect	1-byte	MOV C, M
MOV M, Rs	"	"	MOV M, B Data of source (Rs) register will be store at the memory location pointed by HL register pair
IN 8bit	Direct	2 byte	IN 40H
OUT 8-bit	"	"	OUT 01H

This inst will load the contents of acc. from the specified peripheral address

This inst store the content of acc. at the specified peripheral address

LDAX Rp	Indirect	1 bit HL cannot be used in beacuse as these already used in MOV Rd, M	LDAX B This Inst. will load the acc. with the contents of memory location pointed by a Rp (only BC, DE)
STAX, Rp	Indirect	1-byte	STAX D Content of acc. will be stored at memory location pointed by a Rp

XCHG (Exchange the contents of HL & DE)	Implicit	1-byte	H ↕ D, L ↕ E
SPHL Load the contents of HL pair in stack pointer	implicit	1 byte	SP ⟷ HL
XTHL Exchange the contents of HL with top of the stack	"	"	
Push Rp	~~Indirect~~ ~~Register~~ Register	1 byte	Push H (Decreament)
Pop Rp	"	"	Pop H (increament)

Q write an assembly language program using data transfer inst. to perform the following operations.

(i) Load 20H and 36H in HL Register
(ii) Load acc. register with 42H
(iii) Store the contents of L Register at the memory location 2091
(iv) Load DAE registers from the memory locations pointed by HL registers.

Sol^n (i) ⇒ (a) MVI H, 20H (b) LXI H, 2036
MVI L, 36H

(ii)(b) Arithmetic Instructions :-

All inst. except increament & decrement :-

(i) These inst. assume implicitly that one of the operend is accumulator

(ii) These ~~in~~ inst. placed the result in the accumulator

(iii) They will modify all the flags as per the status of the result

(IV) These inst. do not affect the contents of operend register

for increment & decrement operations :-

(i) These operation can be perform on register, register pair, memory

(ii) These inst affect the contents of specified register.

(iii) These inst. affect all the flag except carry flag

Mnemonics	Addressing Mode	Instruction length	Ex	function
ADD R	Register	1 byte	ADD	A+C ⇒ A
ADI 8bit	Immediate	2 byte	ADI 05H	A+8bit data ⇒ A
SUB R	Register	1 byte	SUB D	A-D ⇒ A
SUI 8bit	Immediate	2 byte	SUI 07 H	A-07 ⇒ A
ADC R	Register	1 byte	ADC H	A+R+CY ⇒ A
SBB R	Register	1 byte	SBB C	A-R-CY ⇒ A

ADD M	Indirect	1 byte		ADD M	A+M[H2] ⇒ A
SUB M	"	"		SUB M	A − M[] = A
INR R	Register	1 byte	all flag except carry	INR L	R+1 = R
DCR R	Register	1 byte		DCR B	R−1 = R
INX RP (BC, DE, HL, SP)	Register	1 byte		INX B	BC+1 ⇒ BC
DCX RP (BC, DE, HL, SP)	Register	1 byte		DCX B	DE−1 = DE
ACI 8bit	Immediate	2 byte		ACI 43 H	A+43+CY = A
SEI 8bit	Immediate	2-byte		SBI 24H	A−24−CY = A
ADC M	Indirect	1-byte			A+M+CY = A
SBB M	Indirect	1-byte			A−M−CY = A
DCR M	"	1 byte			~~M+1 = HL~~
INR M	"	1-byte			M+1 = HL
DAD RP (BC, DE, HL, SP)	Register	1 byte			HL+BC = HL
DAA	Implicit	1-byte			

ex write the inst. to load 16-bit no. 2500 in H and L registers using MVI and LXI opcodes

sol MVI H, 25 H LXI H 2500
MVI L, 00 H

Qus The memory location 2050 hold the data byte F7 H. write an inst to transfer this data byte into accumulator. using following different opcodes

(1) LDA (2) MOV (3) LDAX

(1) LDA 2050

(2) LXI H, 2050
MOV A, M

(3) LXI B, 2050
LDAX B

A	F
B 32H	C
D	E
H	L

Que Register B contains 8 bit data & 32H. Illustrate MOV and ~~LDAX~~ STAX inst to copy the contents of register B into memory location 8000H. Using indirecting addressing mode.

Sol

MVI B, 32H
LXI H, 8000
MOV M, B

DAD Rp :- This inst is used to add the contents of HL register pair with specified register pair. and the result will also be stored in HL register pair

we can perform 16 bit addition in single line

Thise CY flag is altered to reflect the result of 16 bit add. no other flags are affected.

~~The~~ DAA :- This is used to ~~result~~ the convert the result in to valid +ve BCD no.. All flags are affected

```
 19          14
+15        +  6
 2E  -->     20
+ 6        - 16
 34           4
```

(c) Logical Instruction

(i) AND, OR, XOR can be used either with 8-bit data or with register or with memory

(ii) These inst implicitly assume that accumulator is one of the operand

(iii) While executing these inst. carry flag is reset all other flags are affected on the basis of status of the result

(iv) The result will also be placed in accumulator and second operand register will remain same

Mnemonic	Addressing Mode	Instruction	Ex
ANI 8-bit	Immediate	2 byte	ANI 45H → A(AND)45 = A
ORI 8-bit	"	"	ORI F2H → A(OR)F2 = A
XRI 8-bit	"	"	XRI A6H → A(XOR)A6 = A
ANA R	Register	1 byte	ANA B
ORA R	"	"	ORA C
XRA R	"	"	XRA D
ANA M	Indirect	1 byte	
ORA M	"	"	
XRA M	"	"	

Ex Register B & C are loaded with 8-bit data A9H & 36H respectively add the contents of B & C registers and display the store of the result at memory location 2999H. Also update the final contents of flag register

Soln

```
MVI B, A9H
MVI C, 36H
MOV A, B
ADD, C
~~HLT~~
~~LXI H 2999~~
~~MOV M, A~~
HLT
```

```
LXI H 2999
MOV M, A
```

Ex 1 A = B6 H
B = 32 H

```
ANA B
ORA B
XRA B
```

Update accumulator contents, flag status

Soln

B6 = 10110110
32 = 00110010

(i) (A) AND (B)

A = 32

(ii) (OR)

A = B6

(iii) XOR

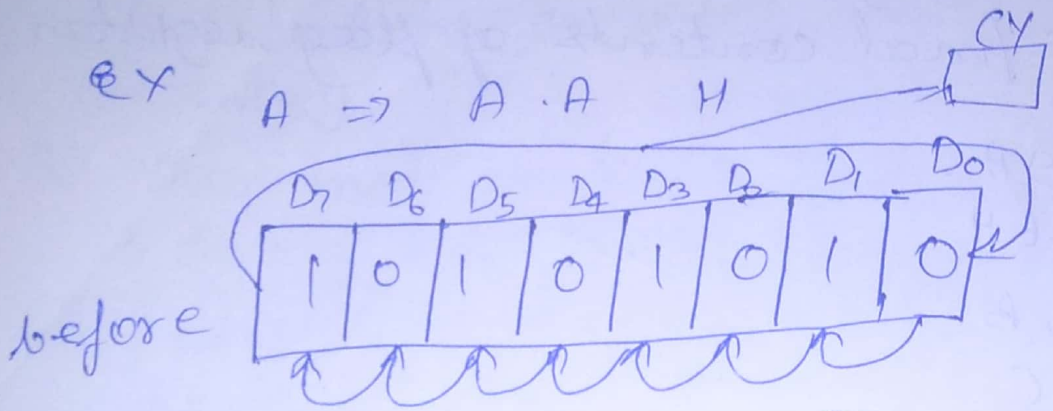
A = 84

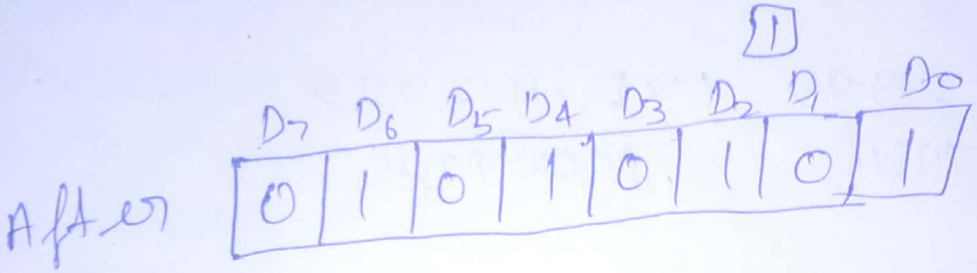
	(i)	(ii)	(iii)
S	0	01	1
Z	0	0	0
CY	0	0	0
P	0	0	1

(2) Rotate Instruction:-

(a) RLC ⇒ Rotate accumulator left with carry

Ex A ⇒ A·A H

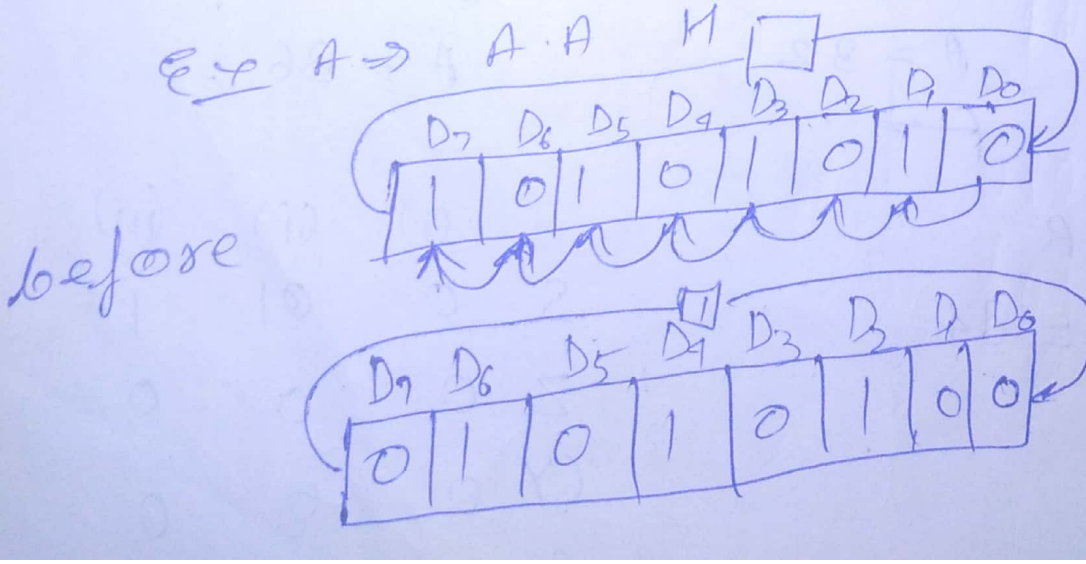




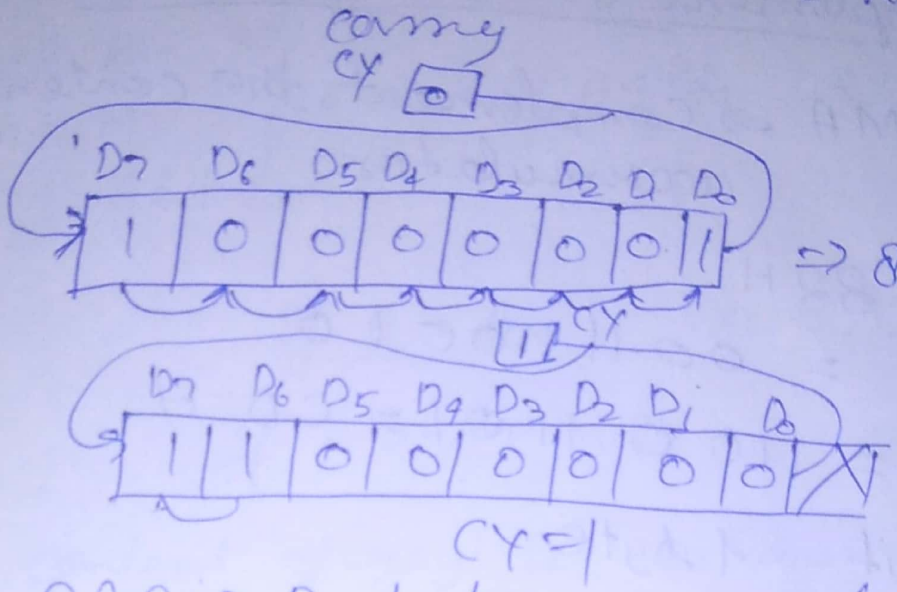
only carry flag will be update after this inst. no other flag is updated

(b) RAL ⇒ Rotate accumulator left through carry

Ex A ⇒ A·A H

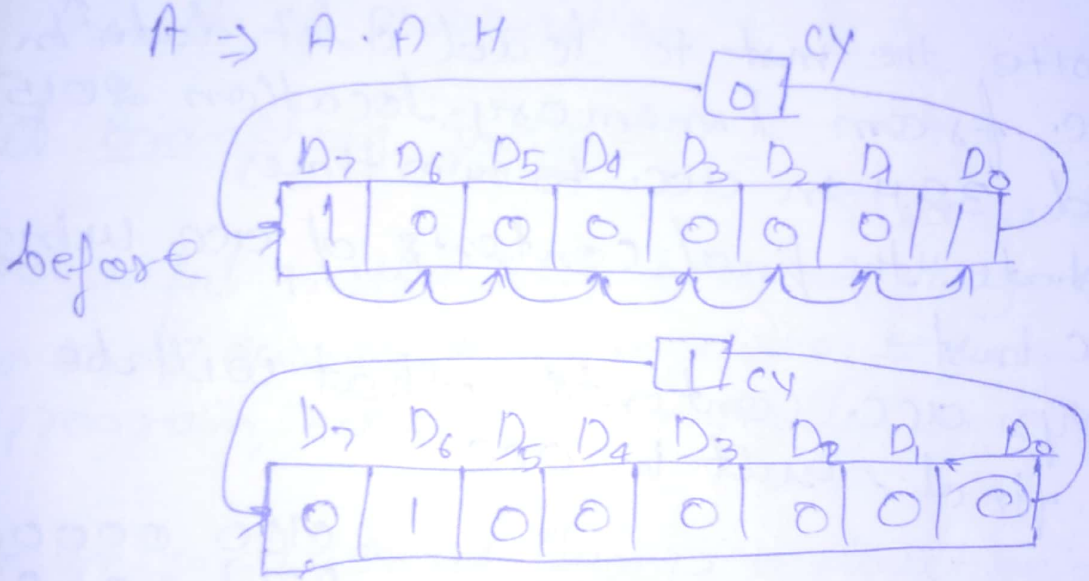


(c) RRC ⇒ Rotate accumulator Right with carry



(d) RAR ⇒ Rotate accumulator right through carry

A ⇒ A · A H



Case A = 04 Right shift ⇒ Devide
left shift ⇒ multiply

Rotate inst used in various app.

(i) Devide by 2 4 multiply by 2
(ii) To check no. is even or odd
(iii) To check no. is +ve or -ve

(3) Complement (NOT) instruction:-

(a) CMA ⇒ complement the contents of accumulator

Ex A = 32H

= 0011 0010

A = 1100 1101 = CD H

Implecit 1 byte

No flag affected.

40H

Ccc write the inst to load 8 bit data in acc. from memory location 2015

(ii) Add 32H in acc. ~~to~~ two times

(iii) Rotate the final contents of acc. using RLC inst

(iv) Comp. acc. contents. what will be the final result in acc.

```
LDA  2015
MVI  B,32
ADD  B
ADD  B
RLC
CMA
HLT
```

```
MOV A,C
ADI 32
```

```
0100 0000
0011 0010
---------
0111 0010
0011 0010
---------
1010 0100
```

RLC 01001001

CMA 10110110

B 6

(4) Compare Instruction :-

CMP R
CMP M
CPI 8bit

A - R ⇒ If A < R/M ; CY = 0; Z = 0
(↓ 05, −06)

If A = R/M ; CY = 0; Z = 1

If A > R/M ; CY = 0 ; Z = 0

(i) The content of register or memory will not be altered.

(ii) P, AC, S will be update on the basis of result of subtraction.

(5) Branching Instruction :-

These are most powerful instruction because they allow to microprocessor to change the sequence of the program either conditionally or unconditionally

1- Unconditional Jump instruction :-

Ex JMP 16 bit ⇒ JMP 2500 (Address)

2- Conditional Jump :- (CY, P, Z, S)

Ex (a) JC 16 bit ⇒ Jump if CY = 1
(b) JNC 16 bit ⇒ Jump if CY = 0
(c) JZ 16 bit ⇒ Z = 1
(d) JNZ 16 bit ⇒ Z = 0
(e) JM 16 bit ⇒ S = 1
(f) JP 16 bit ⇒ S = 0

(g) JPO 16 bit ⇒ P = 0

(h) JPE 16 bit ⇒ P = 1

Que Load the numbers 9B & A7 in registered D & E. Add both the numbers if result is greater than ff. Display 01 at o/p port 1

Sol^n

```
2000  MVI  D, 9B      D = 9B
2002  MVI  E, A7      E = A7
2004  MOV  A, D       A = 9B
2005  ADD  E          A+E = Sum (A)
2006  JNC  200B
2009  MVI  A, 01H
200B  OUT  01H        Port = A.7 ← 9B
```

Que W.a.p. to add two 8 bit no. store the results as well as carry 00 at memory location 2070, 2071 respectively

```
      MVI  D, 9B      D = 9B
      MVI  E, A7      E = A7
      MVI  A, 0
      ADD  E          A = 9B
      JC   SKIP
      STA  2070
      MVI  A, 00H
      STA  2071
      HLT
SKIP: STA  2070
      MVI  A, 01H
      STA  2071
      HLT
```

Qu write an A.LP to exchange the contents of two memory location 8001, 8002

(i) Using direct addressing (LDA)
(ii) Using indirect addressing

8001 52H
8002 35H

(i)

~~MOV~~ ~~MVI B 52H~~
~~MVI C, 35H~~

ACC. → another register → another data location → MOV

```
(i)   LDA 8001      A = 52
      MOV B, A      B = 52
      STA 8001      A = 35
      LDA 8002  → STA 8001   @ 8001 = 35
      MOV A, B      A = 52
      STA 8002      8002 = 52
      HLT
```

(ii) ~~Prob~~

```
      LXI B, 8001H   B 80  C 01
      LXI D, 8002H   D 80  E 02
      LDAX B         A = 52
      MOV H, A       H = 52
      LDAX D         A = 35
      STAX B         8001 ← 35
      MOV A, H       A = 52
      STAX D         8002 ← 52
      HLT
```

Qu w.a.p. to add two 16 bit no. stored in memory location 2051, 2052, 2053, 2054 store the result at 2055 and 2056. Use

(i) ~~data~~ ADD, ADC inst.
(ii) DAD, Rp

(i)
```
LDA 2051        A = 2051
MOV B,A         B = 2051
LDA 2053
ADD B
STA 2055
LDA 2052
MOV C,A
LDA ,2054
ADD C
STA 2056
HLT
```

(ii)
```
LHLD 2051
XCHG
LHLD 2053
DAD D
SHLD 2055
HLT
```

~~LD~~ 2051
LHLD
L = 34
H = 12
HL
⇓
DE
LHLD 2053
L = 67
H = 45

* Instruction cycle:- The time required to execute an inst. is called instruction cycle, it may consist of 1-6 ~~T states~~ machine cycles

(i) Machine cycle :- The time required to access the memory or I/O devices is called machine cycle

The time required to execute one inst. in on cycle is called machine cycle

It One machine cycle may consists of 3-6 T states

One clock cycle is equivalent to one T state

Machine cycles of 8085 :-

opcode fetch cycle (4T)
Memory read cycle (3T)
Memory write cycle (3T)
I/O read cycle (3T)
I/O write cycle (3T)

Ex STA 526A

2000 32
2001 6A
2002 5A