```
MOV E, L

HLT
```

7th Dec

ii) using XCHG

```
LXI B, 1122H
LXI D, 3344H

MOV H, B
MOV L, C

XCHG

MOV B, H
MOV C, L

HLT
```

Note: · Hardware model ↳ DIY

· inst^n format
  - 1 byte
  - 2 byte
  - 3 byte

  } wala

· inst^n type / classification
  - data transfer
  - Arithmetic
  - Logical
  - Branching
  - miscellaneous

iii Arithmetic group inst^n

↳ implicitly assumes A is one of operand

↳ modifies flag

↳ Result stored in A.

① ADD R/M

· ADD R ⇝ Reg dir

· ADD M ⇝ Reg ind

↳ 1 byte inst^n

↳ Adds content of reg/ memory to contents of A & stores result in A.

↳ Result greater than 8 bit,
If carry flag → set

· Eg:

ADD E; A ← A+E

ADD M; A ← A+[M]

1. WAP to add reg D & data at location 4050H store result at 5000H.

```
LDA 4050H

ADD D

STA 5000H

HLT
```

ii) using HL pair

```
LXI H, 4050H
MOV A, D

ADD M

LXI H, 5000H
MOV M, A

HLT
```

(ii) ADI 8 bit data ← ADI Immediate

↳ 2 byte instⁿ

↳ Adds 8 bit data w. contents of A & stores result in A

Eg: ADD B7H

↳ A ← A + B7H

w. 2's complement

SUB R → reg dir

SUB M → reg indir

(iii) SUB R/M

↳ 1 byte subtract inst^n

↳ subtracts content of reg / memory w. content of A & stores result in A.

· Eg: SUB R ; A ← A - R

SUB M ; A ← A - [M]

(iv) SUI 8 bit data → immediate ~~Add~~

↳ 2 byte immediate sub. inst^n

↳ SUB 8 bit data w. content of A & stores result in A.

· Eg:

SUI D3H ; A ← A - D3H

(v) INR/M → Reg dir / Reg in

↳ increment

DCR R/M → R.d. / R.i.

↳ decrement

↳ 1 byte increment / decrement inst^n

↳ ↑ / ↓ content of reg / mem by 1 resp.

↳ data ka ↑ / ↓

Eg: INR A ; A = A+1 (122H → 122H+1)
INR m ; [HL] = [HL]+1

DCR B ; B = B-1
DCR m ; [HL] = [HL]-1

(vi) INX Rp (R in) DCX Rp (R in)

- 1 byte instⁿ
- ↑/↓ reg pair (by 1)
  - address ko value lai ↑/↓

BC = 2050 —INX→ 2051 (C lai matra ↑ garera hudaina at times, tara BC = 20FF +1 2100 tab change vayo)

INRC W as only C↑

- (No flags) affected
  - as data ma ↓/↑ hudaina

Eg:

INX B ; BC = BC+1

DCX D ; DC = DC-1

(vii) ADC R/M (Add w. carry)

↳ 1 byte inst^n

↳ Adds A w. content of reg/ mem & previous carry.
↳ Result stored in A.

• Eg:

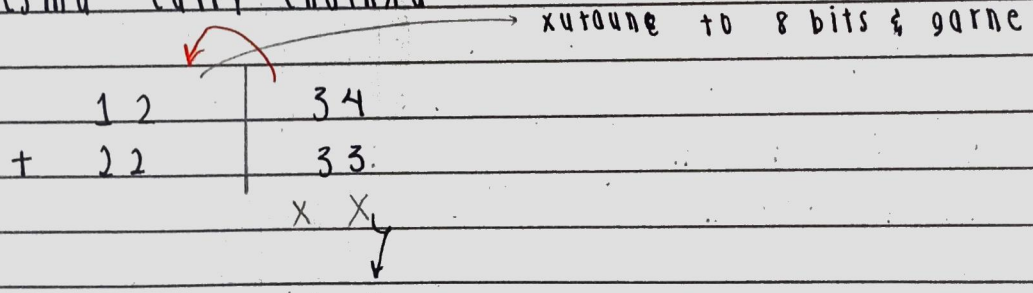
ADC B ; A ← A + B + CY

ADC M ; A ← A + [M] + CY

• carry aaune case

↳ 16 bit addition garna xa vane chainxa carry

1234 H
+ 2233 H
↓
yesma carry chainxa



(XIII) ACI 8 bit data
↳ 2 byte inst"

↳ Adds A w. 8 bit data & previous carry.
↳ Result stored in A

• Eg: ACI 70H
↓
A ← A+70H+(CY)

1 Load Rp B & D w. ABCD H & 3092 H WAP to add these two 16 bits no. & store the sum in H & L registers

B: AB	C: CD
+ D: 30	E: 92
ADC	ADD
I/P	I/P
↓	↳ L
H	

LXI B, ABCD H
LXI D, 3092 H

1st mv C & E add gerne

MOV A, C
ADD E

hunxa ki hunna not my prob ?~<

```
MOV L, A
```
yeta B+D+CY

```
MOV A, B
ADC D
```
→ Add w. carry

```
MOV H, A

HLT
```

2. ii) Store sum in 2001H & 2000H registers (↑ order data)

B+D → 2001H

C+E → 2000H

```
LXI B, ABCD H
LXI D, 3092 H
```
1st Add C+E
```
MOV A, C
ADD E

STA 2000H

MOV A, B
ADC D
```
→ D+B+CY

```
STA 2001 H

HLT
```