Unit -III

Instruction set of 8085 Microprocessor

Instruction set of 8085 Microprocessor:

Every processor(Microprocessor) has its own instruction set, so 8085 Microprocessor also has its own instruction set. The Instruction set of 8085 Microprocessor is classified into 5 main groups as follows:

I]Data Transfer group

II] Arithmetic Group

III] Logical Group

IV] Branch Control Group

V] Machine and I/O Control Group

I] Data Transfer Group of Instructions:

This group of instructions are used to transfer the data from source to destination that is from register to register, register to memory and memory to register, but not from memory to memory. In this group of instruction data is just copied from source to destination. The instructions of this group are as follows:

1) MOV r1, r2

Format :-

[r1] ← [r2]

Addressing mode :- register

Flag affected :- No flags are affected

ex. :- MOV A,B

This instruction moves the content of reg. 2 into reg. r1.

2) MOV r , M

Format :-

[r] ← [[HL]]

Addressing mode :- Indirect

Flag affected :- No flags are affected

Ex. :- MOV B,M

This instruction moves the content of memory, whose address is in HL pair into reg. r.

3) MOV M, r

Format :-

[[HL]] ← [r]

Addressing mode :- Indirect

Flag affected :- No flags are affected

Ex. :- MOV M , C

This instruction moves the content of reg. r into memory whose address is in HL pair

4) MVI r, data (8-bit)

Format :-

[r] ← [data]

Addressing mode :- Immediate

Flag affected :- No flags are affected

Ex. :- MVI D, 05 H

This instruction moves immediate data, which is given in the instruction into reg. r .

5) MVI M, 8-bit data

Format :-

[[HL]] ←[data]

Addressing mode :- Immediate / Indirect

Flag affected :- no flags are affected

Ex. :- MVI M ,05 H

This instruction moves immediate data, which is given in the instruction into memory , whose address is in HL pair.

6) LXI rp,16-bit data

Format :-

[rp] ← [16-bit data]

Addressing mode :- Immediate

Flag affected :- No flags are affected

Ex. :- LXI H, 2000H

This instruction loads immediate data which is given in the instruction into reg. pair rp.

7) LDA addr. (16-bit)

Format :-

[A] ← [addr.]

Addressing mode :- Direct

Flag affected :- No flags are affected

Ex. :- LDA 2000 H

This instruction loads directly Accumulator from address which is given in the instruction itself.

8) STA 16-bit addr.

Format :-

[addr.] ← [A]

Addressing mode	:- Direct
Flag affected	:- No flags are affected
Ex.	:- STA 2000H

This instruction stores the contents of accumulator at the address which is given in the instruction.

9) LHLD addr.

Format :-

[L] ← [addr.]

[H] ← [addr. + 1]

Addressing mode	:- Direct
Flag affected	:- No flags are affected
Ex.	:- LHLD 2000H

This instruction loads HL pair directly from the address given in the instruction. Here reg. L is loaded from the address given in the instruction and reg. H is loaded from the next memory location

10) SHLD addr.

Format :-

[addr.] ← [L]

[addr. + 1] ← [H]

Addressing mode	:- Direct
Flag affected	:- No flags are affected
Ex.	:- SHLD 2000H

This instruction loads HL pair directly from the address given in the instruction. Here reg. L is contents are stored at address given in the instruction and reg. H contents are stored at next memory location.

11) LADX rp

Format :-

[A] ← [[rp]]

Addressing mode :- Indirect

Flag affected :- No flags are affected

Ex. :- LADX B

This instruction loads accumulator indirectly from the memory location ,whose address is given in reg. pair rp.

12) STAX rp

Format :-

[[rp]] ← [A]

Addressing mode :- Indirect

Flag affected :- No flags are affected

Ex. :- STAX B

This instruction stores accumulator indirectly from the memory location , whose address is in reg. pair rp.

13) XCHG

Format :-

[DE] ← [HL]

Addressing mode :- Register

Flag affected :- no flags are affected

Ex. :- STAX B

This instruction exchange the contents of HL pair with DE pair.

II) Arithmetic group of instructions:

This group of instruction perform arithmetic operations such as addition, subtraction, increment and decrement.

1) ADD r

Format:-

[A] ← [A] +[r]

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- ADD C

This instruction adds the content of reg. r with accumulator and result is stored back into the accumulator.

2) ADD M

Format

[A] ← [A] + [[HL]]

Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction adds the contents of memory, whose address is in HL pair with accumulator and result is stored back into the accumulator.

3) ADC r

Format:

[A] ← [A] + [r] + [CY]

Addressing mode :- Register

Flag affected :- All flags are affected

This instruction adds the content of register r with accumulator, along with carry and result is stored back into accumulator .

4) ADC M

Format:

[A] ← [A] + [[HL]] + [CY]

Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction adds the content of memory, whose address is in HL pair with accumulator along with carry and result is stored back into accumulator.

5) ADI data

Format:

[A] ← [A] + [data]

Addressing mode :- Immediate

Flags affected :- All flag are affected

Ex. :- ADI 04H

This instruction adds immediate data given in the instruction with accumulator and result is stored back into accumulator.

6) ACI data

Format:

[A] ← [A] + [data] + [CY]

Addressing mode :- Immediate

Flags affected :- All flag are affected

Ex. :- ACI 04H

This instruction adds immediate data with accumulator along with carry and result is stored back into accumulator.

7) SUB r

Format:

[A] ← [A] - [r]

Addressing mode :- Register

Flags affected :- All flag are affected

Ex. :- SUB L

This instruction subtract the content of reg. r from accumulator and result is stored back into accumulator.

8) SUB M

Format:

[A] ← [A] - [[HL]]

Addressing mode :- Indirect

Flags affected :- All flag are affected

Ex. :- SUB L

This instruction subtract the content of memory whose address is in HL pair from accumulator and result is stored back into accumulator.

9) SBB r

Format:

[A] ← [A] - [r] - [CY]

Addressing mode :- Register

Flags affected :- All flag are affected

Ex. :- SBB C

This instruction subtract the content of reg. r from accumulator along with borrow (negaive carry) and result is stored back into accumulator.

10) SBB M

Format:

[A] ← [A] - [[HL]] - [CY]

Addressing mode :- Indirect

Flags affected :- All flag are affected

This instruction subtract the content of memory whose address is in HL pair from accumulator along with borrow (negative carry) and result is stored back into accumulator.

11) SUI data

Format:

[A] ← [A] - [data]

Addressing mode :- Indirect

Flags affected :- All flag are affected

Ex. :- SUI 56H

This instruction subtract the immediate data given in the instruction from accumulator and result is stored back into accumulator.

12) SBI data

Format:

[A] ← [A] - [data] - [CY]

Addressing mode :- Immediate

Flags affected :- All flags are affected

Ex. :- SBI 97H

This instruction subtract the immediate data given in the instruction from accumulator along with borrow and result is stored back into accumulator.

13) INR r

Format:

[r] ← [r] +1

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- INR C

This instruction increments the content of reg. r by one.

14) INR M

Format:

[[HL]] ← [[HL]] + 1

Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction increment the content of memory, whose address is in HL pair by one.

15) DCR r

Format:

[r] ← [r] - [1]

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- DCR B

This instruction decrement the content of reg. r by one.

16) DCR M

Format:

[[HL]] ← [[HL]] - 1

Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction decrement the content of memory, whose address is in HL pair by one.

17) INX rp

Format:

[rp] ← [rp] + 1

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- INX B

This instruction increment the content reg. pair by one

Here only high order reg. is defined i.e B for BC pair , D for DE pair and H for HL pair.

18) DCX rp

Format:

[rp] ← [rp] - 1

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- DCX B

This instruction decrement the content reg. pair by one.

19) DAA

Addressing mode :- Implicit \ Implied

Flags affected :- All flags are affected

DAA means Decimal Adjust Accumulator. It is used to convert the hexadecimal result into decimal. It is always use after the addition instruction that is ADD , ADI and ACI. If the result is between 0 to 9 then the result is kept as it is and if the result is between A and F then 6 is added in the result.

20) DAD rp

Format:

[HL] ← [HL] + [rp]

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- DAD D

This instruction adds the content of reg. pair rp with HL pair and result is stored back in to the HL pair.

III) Logical group of instructions:

This group of instruction perform logical operation such as AND, OR, NOT, EX-OR, ROTATE, SHIFT, COMPARE.

1) ANA r

Format:

[A] ← [A] ^ [r]

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- ANA C

This instruction AND's the content of reg. r with accumulator and result is stored back into accumulator.

2) ANA M

Format:

[A] ← [A] ^ [[HL]]

Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction AND's the content of memory, whose address in HL pair with accumulator and result is stored back into accumulator.

3) ANI data

Format:

[A] ← [A] ^ [data]

Addressing mode :- Immediate

Flags affected :- All flags are affected

Ex. :- ANI 65H

This instruction AND's Immediate data given in the instruction with accumulator and result is stored back into accumulator.

4) ORA r

Format:

[A] ← [A] v [r]

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- ORA C

This instruction OR's the contents of reg. r with accumulator and result is stored back into accumulator .

5) ORA M

Format:

[A] ← [A] v [[HL]]

Addressing mode :- Indirect

Flags affected :- All flags are affected

Ex. :- ORA C

This instruction OR's the content of memory, whose address is in HL pair with accumulator and the result is stored back into accumulator.

6) ORI data

Format:

[A] ← [A] v [data]

Addressing mode :- Immediate

Flags affected :- All flags are affected

Ex. :- ORA 65 H

This instruction OR's immediate data which is given in the instruction with accumulator and the result is stored back into accumulator.

7) XRA r

Format:

[A] ← [A] ∀ [r]

Addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- XRA C

This instruction XOR's the content of reg. r with accumulator and the result is stored back into accumulator.

8) XRA M

Format:

[A] ← [A] ∀ [[HL]]

Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction XOR's the contents of memory whose address is in HL pair with accumulator and the result is stored back into accumulator.

9) XRI data

Format:

[A] ← [A] ∀ [data]

Addressing mode :- Immediate

Flags affected :- All flags are affected

Ex. :- XRI 65 H

This instruction XOR's immediate data which is given in the instruction with accumulator and the result is stored back into accumulator.

10) CMA

Format:

$[A] \leftarrow [\bar{A}]$

Addressing mode :- Implicit / Implied

Flags affected :- All flags are affected

This instruction one's complement the content of accumulator. one's complement zero replaced by one and one is replaced by zero.

11) CMC

Format:

$[CY] \leftarrow [\overline{CY}]$

Flags affected :- Only carry flag

This instruction one's complement the carry flag.

12) STC

Format:

$[CY] \leftarrow 1$

Flags affected :- Only carry flag

This instruction sets carry flag i.e. CY=1

13) CPM r

Format:-

[A] - [r]

addressing mode :- Register

Flags affected :- All flags are affected

Ex. :- CMP H

This instruction compares the content of reg. r with accumulator, here the content of reg. r are subtracted from accumulator and according to the result of subtraction flags are affected but the result of subtraction is not stored .Also accumulator contents remains unaffected.

14) CMP M

Format:-

[A] - [[HL]]

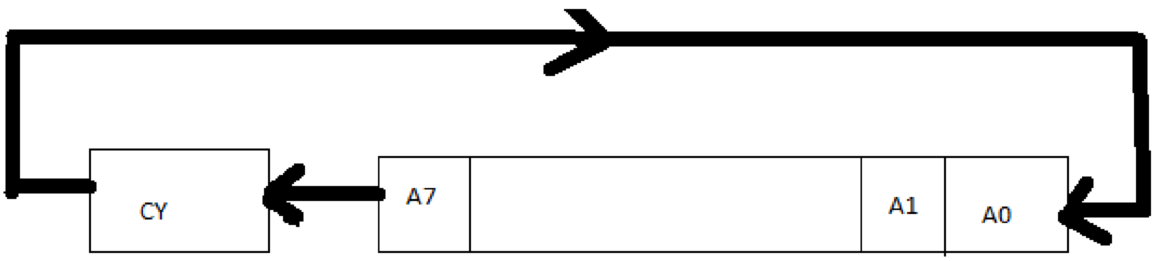
Addressing mode :- Indirect

Flags affected :- All flags are affected

This instruction compares the contents of memory whose address is in HL pair with accumulator, here the content of memory are subtracted from accumulator and according to the result of subtraction flags are affected but the result of subtraction is not stored. Also accumulator contents remains unaffected.

16) RAL

Format:-



[A1] ← [A0]

[CY] ← [A7]

[A0] ← [CY]

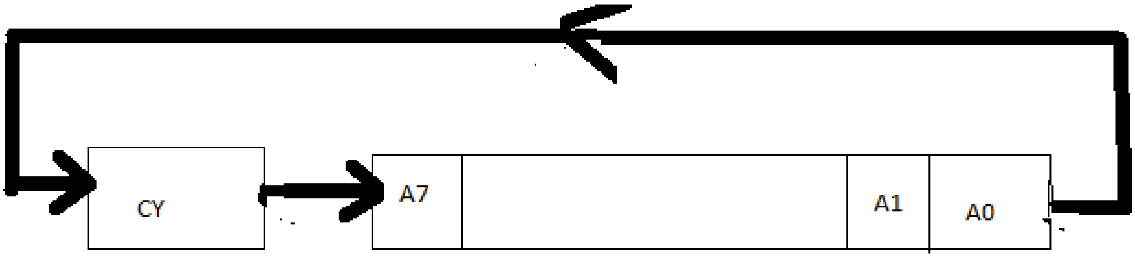
Addressing mode :- Implicit / Implied

Flags affected :- all flags are affected

This instruction rotates the content of accumulator in left direction by one bit with carry.

17) RAR

Format:-



[A0] ← [A1]

[CY] ← [A0]

[A7] ← [CY]

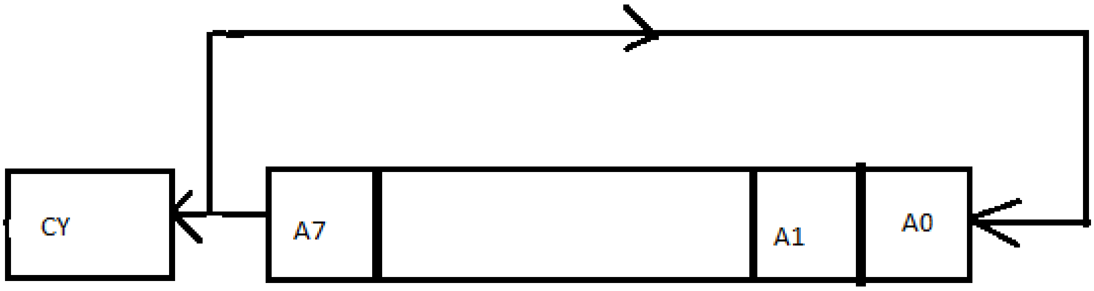
Addressing mode :- Implicit / Implied

Flags affected :- all flags are affected

This instruction rotates the content of accumulator in right direction by one bit with carry.

18) RLC

Format:-



[A1] ← [A0]

[CY] ← [A7]

[A0] ← [A7]

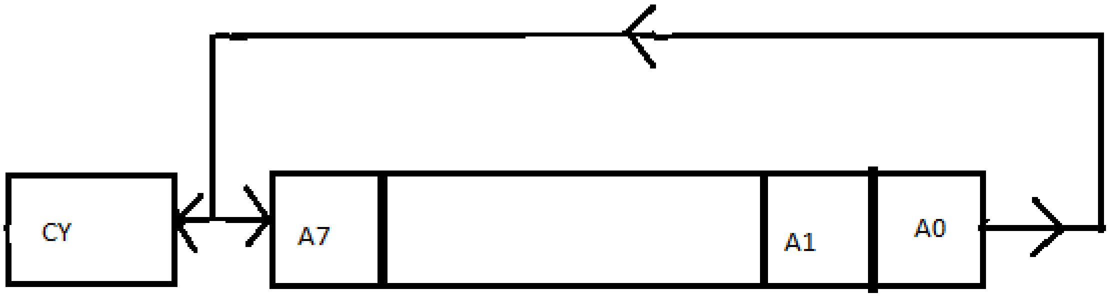
Addressing mode :- Implicit / Implied

Flags affected :- all flags are affected

This instruction rotates the contents of accumulator in left direction by one bit without carry.

19) RRC

Format:-



[A0] ← [A1]

[CY] ← [A0]

[A7] ← [A0]

Addressing mode :- Implicit / Implied

Flags affected :- All flags are affected

This instruction rotates the contents of accumulator in right direction by one bit without carry.

IV] Machine & I/O Control Group of Instructions:

This group of instruction are used to control the operation of machine (processor), I/O devices & Stack (reserved area in the RAM).

1) In Port address

Format:

$[A] \leftarrow [Port\ addr.]$

Addressing mode :- Direct

Flags affected :- None.

Ex. :- IN 01H

This instruction inputs the data to the processor accumulator from port address given in the instruction. Here processor has 3 port i.e. Port A (00), Port B (01H), Port C (l0H).

2) Out Port addr.

Format:

$[Port\ addr] \leftarrow [A]$

Addressing Mode :- Direct

Flags affected :- None

Ex. :- OUT 01H

This instruction outputs the data from processor accumulator to the port address given in the instruction.

3) Push rp

Format:

$[[sp]] \leftarrow [rp]$

$[[sp-1]] \leftarrow [rl]$

$[[sp-2]] \leftarrow [rh]$

Addressing Mode :- Indirect

Flags affected :- None

Ex. :- Push B.

This instruction push the contents of register pair rp into the stack, so stack is decremented by 2. Here rh contents are moved into [SP-1] & r contents are moved into [SP-2].

4) Push PSW.

Format:

[[SP]] ← [PSW]

[[SP-1]] ← [A]

[[SP-2]] ← [PSW]

Addressing Mode :- Indirect

Flags affected :- None

This instruction push the contents of PSW (processor status word) into the stack, so stack is decremented by 2. Here PSW is 16-bit reg. i.e. it is combination of accumulator & flag register.

5) POP rp

Format:

[rp] ← [[SP]]

[rl] ← [[SP]]

[rh] ← [[SP+1]]

Addressing Mode :- Indirect

Flags affected :- None

Ex. :- POP B.

This instruction POP's the contents of reg. pair rp from stack which were saved earlier.

6) POP PSW

Format

$$[PSW] \longleftarrow [[SP]]$$

$$[A] \longleftarrow [[SP+1]]$$

Addressing Mode :- Indirect

Flags affected :- None

This instruction POP's (takes back) the contents of PSW from stack which were save earlier.

7) HLT

Addressing Mode :- None

Flags affected :- None

HLT means halt, this instruction stops the processor execution operation.

8) NOP

Addressing Mode :- None

Flags affected :- None

NOP means no operation . When this instruction is executed then no operation is performed. This instruction can be used as delay.

9) EI.

Addressing Mode :- None

Flags affected :- None

This instruction Enables (starts) the interrupt.

10) DI

Addressing Mode :- None

Flags affected :- None

DI means Disable Interrupt .i.e. (Stop).

11) XTHL

Format

[HL] ↔ [[sp]]

Addressing Mode :- Indirect

Flags affected :- None

This instruction Exchange the contents of HL pair with stack top, whose address is in SP.

12) SPHL

Format

[SP] ← [HL]

Addressing Mode :- Register

Flags affected :- None

This instruction moves (transfer) the contents of HL pairs into SP.

V] Branch Control Group of Instructions:

This group of instruction is used to break the normal sequence of the program. There are two types of branch control instruction-

1. Unconditional

2. Conditional.

1. Unconditional

In this instruction normal sequence of the program is break without any condition.

2. Conditional

In this instruction the normal sequence of program is break only if the condition is satisfied.

1) JMP addr. [label]

Format

[pc] ← **addr [label]**

Addressing mode	:- Direct / Immediate
Flags affected	:- None.
Ex.	:- JMP 2000H.

When this instruction is executed then program jumps to the address of label given in the instruction unconditionally.

2) Conditional Imp addr. [label] Instruction.

There are 8 types of conditional jump instruction.

i) JC addr. (label)

Program jumps to the address of label, if there is carry in the result.

ii) JNC addr. (label)

Program jumps to the address of label, if there is no carry in the result.

iii) JZ addr. (label)

Program jumps to the address of label if result is zero.

iv) JNZ addr. (label)

Program jumps to the address of label if result is not Zero.

v) JP addr. (label)

Program jumps to the address of label if result is plus (+).

vi) JM addr. (label)

Program jumps to the address of label if result is minus (-).

vii) JPE addr. (label)

Program jumps to the address of label, if parity is even

viii) JPO addr. (label)

Program jumps to the address of label, if parity is odd.

3) CALL addr. (label) [unconditional]

Format:

[[SP]] ← [PC]
[[SP-1]] ← [PCH]
[[SP-2]] ← [PCL]

Addressing mode :- Indirect

Flags affected :- None.

Ex. :- CALL 5000H.

This instruction is use to call the subroutine (subroutine means a small program which is required many times in our main program.) Before calling the subroutine the contents of pc are stored in the stack, so stack is decremented by 2. This is unconditional call instruction.

4) CALL addr. (label) [conditional]

There are 8 types of conditional call instruction.

i) CC addr. (label)

Call the subroutine, if there is carry in the result.

ii) CNC addr. (label)

Call the subroutine, if there is no carry in the result.

iii) CZ addr. (label)

Call the subroutine, if there the result is zero.

iv) CNZ addr. (label)

Call the subroutine, if there the result is not zero.

v) CP addr. (label)

Call the subroutine, if the result is plus (+)

vi) CM addr. (label)

Call the subroutine, if result is minus (-)

vii) CPE addr. (label)

Call the subroutine, if parity of the result is even.

viii) CPO addr. (label)

Call the subroutine, if parity of the result is odd.

5) RET [unconditional]

Format:

[PCL] ← [[SP]]
[PCH] ← [[SP]+1]
[SP] ← [SP]+2

Addressing mode :- Indirect

Flags affected :- None.

Ex. :- RET

RET instruction is used at the end of the subroutine. Before the execution of the subroutine the address of the next instruction of the main program is saved in the stack. The execution of the RET instruction brings back the saved contents of stack into PC. The contents of stack is incremented by 2.

6) RET [unconditional]

There are 8 types of conditional return instruction-

i) RC

Return from subroutine, if there is carry in the result.

ii) RNC

Return from subroutine, if there is no carry in the result.

iii) RZ

Return from subroutine, if result is zero.

iv) RNZ

Return from subroutine, if result is not zero.

v) RP

Return from subroutine, if result is plus (+)

vi) RM

Return from subroutine, if result is minus (-).

vii) RPE

Return from subroutine, if parity is even.

viii) RPO

Return from subroutine, if parity is odd.

7) PCHL.

Format:

[PC] ← [HL]

Addressing mode :- Register / Indirect

Flags affected :- None.

The contents of HL pair are transferred into PC, So program jumps to the address given by HL pair.