

## EXP. NO. (12)

# LOGICAL INSTRUCTIONS OF THE 8085 MICROPROCESSOR

### OBJECT:

To study the logical capabilities of the 8085 microprocessor.

### THEORY

The logical instructions, as the name implies perform logical operations on the data stored in the memory or the register. These operations are supported by the three different modes of addressing which are; the implied, immediate and direct modes.

All the addressing modes use the accumulator as the second operand to perform one of the following instructions, and generated a result again in the accumulator. The corresponding flags are set according to the result after each instruction. The logical instructions supported by these operations are:

1. AND Operation (ANA, ANI) :- These instructions ANDs the accumulator with the required 8-bit data, and put the result again in the accumulator. These instructions are symbol as follows:

$$A=A.X$$

2. OR Operation (ORA, ORI) :- These instructions ORs the accumulator with the required 8-bit data, and put the results again in the accumulator. These instructions are symbol as follows:

$$A=A+X$$

3. EX-OR Operation (XRA, XRI) :- These instructions Exclusive-ORs the accumulator with the required 8-bit data, and put the results again in the accumulator. These instructions are symbol as follows:

$$A=A\oplus X$$

4. Rotate Operations (RLC, RRC, RAL and RAR) :- These instructions address the accumulator only. They perform a shift left or shift right of the accumulator contents.

5. Complement Operation (CMA) :- This instruction also addresses the accumulator only. It performs the ONEs complement of the accumulator contents.

$$A=\overline{A}$$

6. Also falling under this category are certain instructions that do not affect the contents of the accumulator or any other register, yet it affects the appropriate flags. These instructions are:

Compare Operations (CMP, CPI) :- These instructions compare the accumulator with the required 8-bit data, setting the appropriate flag and leaving the accumulator untouched.

No	Instruction	Type	No. of Bytes	Function	Effect
1.	ANA r	Logical	1	A=A and r	All
2.	ANI byte	Logical	2	A=A and byte	All
3.	ORA r	Logical	1	A=A or r	All
4.	ORI byte	Logical	2	A=A or byte	All
5.	XRA r	Logical	1	A=A xor r	All
6.	XRI byte	Logical	2	A=A xor byte	All
7.	CMA	Logical	1	A=A	None
8.	CMC	Logical	1	CY=CY	CY
9.	STC	Logical	1	CY=1	CY

**Notes:-**

- 1- Most of logical instructions affect the contents of an important CPU register; namely the flag register.
- 2- Except CMC and STC instructions. All Logical instructions use accumulator as the 1<sup>st</sup> operand.
- 3- You can use STC then CMC to reset the carry flag CY.

## **Review of logical operations:**

### **1. Complement**

**E.g.:**

MVI A,5Ah

CMA

HLT

A= 5Ah= 01011010

-----  
 $\overline{A}$ =0A5h= 10100101

### **2. And**

**E.g.:**

MVI A,5Ah

MVI B,1Fh

ANA B ; or directly using ANI 1FH

HLT

A= 5Ah= 01011010

B= 1Fh= 00011111

-----  
A AND B=1Ah= 00011010 ; 1=1 AND 1 ONLY, otherwise=0

### **3. Or**

**E.g.:**

MVI A,5Ah

MVI B,1Fh

ORA B ; or directly using ORI 1FH

HLT

A= 5Ah= 01011010

B= 1Fh= 00011111

-----  
A OR B=5Fh= 01011111 ; 0=0 OR 0 ONLY, otherwise=1

### **4. Xor**

**E.g.:**

MVI A,5Ah

MVI B,1Fh

XRA B ; or directly using XRI 1FH

HLT

A= 5Ah= 01011010

B= 1Fh= 00011111

-----  
A XOR B=45h= 01000101 ; 0=0 XOR 0, 0=1 XOR 1, otherwise=1

### Notes:-

- The instruction **XRA A** is used to **reset the accumulator** which is preferred than **MVI A, 0** because the 1<sup>st</sup> instruction is translated to one byte rather than the 2<sup>nd</sup> one which translated to two bytes.

#### E.g.:

Reset the Accumulator

MVI A,5bh

XRA A

HLT

```
A= 5bh=  01011011
A= 5bh=  01011011
-----
A xor A=0 = 00000000
```

- To **complement specific bits** of the accumulator, use **XRI** instruction with byte when the corresponding bits are set and the other bits are reset.

#### E.g.:

Complement bit0 and bit7 of A, when A=5b

MVI A,5bh

XRI 81

HLT

```
A= 5bh=  01011011
81=     10000001
-----
A xor 81=dah = 11011010
```

- To **set specific bits of the accumulator**, use **ORI** instruction with byte when the corresponding bits are set and the other bits are reset.

#### E.g.:

Set bit0 of A, when A=5ah

MVI A,5ah

ORI 1

HLT

```
A= 5ah=  01011010
1=      00000001
-----
A OR 1=5bh= 01011011
```

- To **reset specific bits** of the accumulator, use **ANI** instruction with byte when the corresponding bits are reset and the other bits are set.

**E.g.:**

Reset bit0 of A, when A=5bh

MVI A,5bh

ANI 0feh

HLT

A= 5bh= 01011011

0Feh= 11111110

-----

A AND 0feh=5ah = 01011010

- In other word reset bits means **mask these specific bits**. Bit masking involves isolating one or more bits in a binary quantity while hiding or masking the unwanted bits is usually done with the logical instructions. Then you can used unmask bits in decision making when using branch instructions. In more precise words this operation is called **checking status bit** which involve **ANI** instruction with byte when the corresponding bits are set and the other bits are reset. Then we can check the result if it is zero that is mean that the status bit still unready, otherwise the status bit is one and wanted state is ready.

**E.g.:**

Mask all bits of A=5bh except bit 4 (checking status bit 4)

MVI A,5bh

ANI 10

HLT

A= 5bh= 01011011

10= 00010000

-----

A AND 10=10 = 00010000

## Rotate Instructions

**RAL:** Each binary bit of the accumulator is rotated left by one position through the carry flag. Bit  $D_7$  is placed in the bit in the carry flag and the carry flag is placed in the least significant position  $D_0$ .

**RAR:** Each binary bit of the accumulator is rotated right by one position through the carry flag. Bit  $D_0$  is placed in the carry flag and the bit in the carry flag is placed in the most significant position  $D_7$ .

**RLC:** Each binary bit of the accumulator is rotated left by one position Bit  $D_7$  is placed in the position of  $D_0$  as well as in the carry flag.

**RRC:** Each binary bit of the accumulator is rotated right by one position Bit  $D_0$  is placed in the position of  $D_7$  as well as in the carry flag.

## Examples

### RAL

CY=0	CY=1
MVI A,8fh ; A=8fh RAL ; A=1eh, CY=1	STC ; CY=1 MVI A,8fh ; A=8fh RAL ; A=1fh, CY=1

### RAR

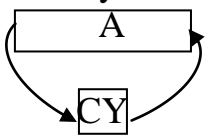
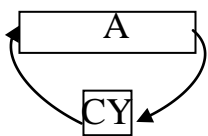
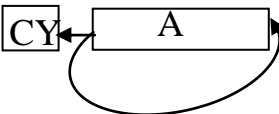
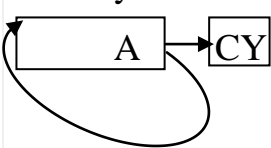
CY=0	CY=1
MVI A,1fh ; A=1fh RAR ; A=0fh, CY=1	STC ; CY=1 MVI A,1fh ; A=1fh RAR ; A=8fh, CY=1

### RLC

CY=0	CY=1
MVI A,8fh ; A=8fh RLC ; A=1fh, CY=1	STC ; CY=1 MVI A,8fh ; A=8fh RLC ; A=1fh, CY=1

### RRC

CY=0	CY=1
MVI A,1fh ; A=1fh RRC ; A=8fh, CY=1	STC ; CY=1 MVI A,1fh ; A=1fh RRC ; A=8fh, CY=1

No	Instruction	Type	No. of Bytes	Function	Benefits	Effect
1.	RAL (Rotate All Left)	Logical	1	Shift left A one pos., bit0=old cy New cy=bit7  New      old	Delay, Mull. By 2 *reset cy before using this instruction	CY only
2.	RAR (Rotate All Right)	Logical	1	Shift Right A one pos., bit7=old cy, New cy=bit0  Old      new	Delay, Div. By 2 *reset cy before using this instruction	CY only
3.	RLC (Rotate Left with Carry)	Logical	1	Shift left A one pos., bit0=bit7, New cy=bit7 	Delay, Check bit7 which saved in the cy. Use RRC then to retrieve the old byte	CY only
4.	RRC (Rotate Right with Carry)	Logical	1	Shift Right A one pos., bit7=bit0, New cy=bit0 	Delay, Check bit0 which saved in the cy. Use RLC then to retrieve the old byte	CY only

### **Notice:-**

- 1- Each of these instructions manipulates the accumulator and the (CY) flag during the rotation process. The rotate instructions are often used to check the status of individual bit. This is done by rotating the bit into the carry flag, and then using the JC, JNC instructions to jump, based on the (CY) flag value.
- 2- Rotate instructions are also used to perform binary multiplication and division. A binary value is multiplied by (2) by shifting the bits left by one bit position. Similarly, a number is divided by two by shifting it right.

## **LAB Work**

1.  $C = (A + D) \text{ XOR } (C \text{ AND } 15h)$  when  $A = 0F1h$ ,  $D = 0E3h$ ,  $C = 36h$
2.  $D = (H \text{ OR } B) \text{ NAND } (C - D)$  when  $H = 15h$ ,  $B = 2Ah$ ,  $C = 0Ah$ ,  $D = 5$

## **Home Work**

Write programs with effects

- 1-  $HL = (BC + HL) \text{ OR } DE$  (use register pair when necessary), when  $BC = 105h$ ,  $HL = 340h$ ,  $DE = 180h$
- 2- Reset bits 0,2 of A and set bits 4,6,7 when  $A = 0A7H$
- 3-  $D = (C \text{ OR } 5) - (B \text{ XOR } D)$  when  $D = 3fh$ ,  $BC = 1da5h$
- 4- What is the effect of the following instructions:  
ANA A, ORA A, CMA, XRA A when  $A = 75h$
- 5- What is the result of each instruction of the following program and its effect?

```
MVI A,56h
CMA
STC
MVI B,0FH
ANA B
ANI 1
ORA B
ORI 80h
XRA B
XRA A
XRI 32h
STC
RST1
```