LOGICAL INSTRUCTIONS OF THE 8085 MICROPROCESSOR

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	$A = \overline{A}$	None
8.	CMC	Logical	1	$CY = \overline{CY}$	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 1st operand.

3- You can use STC then CMC to reset the carry flag$\overline{CY}$.

Review of logical operations:

1. Complement

E.g.:

```
MVI A, 5Ah
CMA
HLT
```

```
        A= 5Ah=    01011010
                   ------------
   ___
   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 1st instruction is translated to one byte rather than the 2nd 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	STC	; CY=1
RAL	; A=1eh, CY=1	MVI A,8fh	; A=8fh
		RAL	; A=1fh, CY=1

RAR

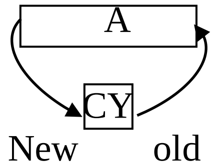
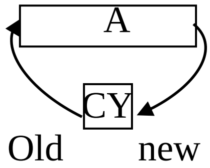
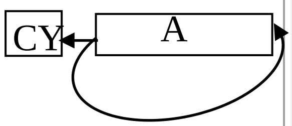
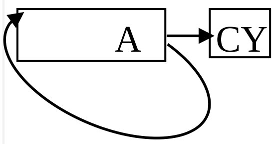
CY=0		CY=1	
MVI A,1fh	; A=1fh	STC	; CY=1
RAR	; A=0fh, CY=1	MVI A,1fh	; A=1fh
		RAR	; A=8fh, CY=1

RLC

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

RRC

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

Class Work

1) D=(15 AND B) NOR (C+1) when B=2Fh, C=D1h

Address	HexCode	Label	Opcode	Operands	Comments
2000 2001 2002	 D1 2F		LXI	B,2FD1	; B=2F,C=D1
2003			INR	C	; C=C+1=D2, S=1,Z=0,AC=0, P=1
2004			MOV	A,B	;A=B=2F
2005 2006	 15		ANI	15	; A=A AND 15=05,S=0,Z=0,AC=0, P=1, CY=0
2007			ORA	C	; A=A OR C=D7, S=1, Z=0,AC=0, P=1,CY=0
2008			CMA		; A=$\overline{A}$=28
2009			MOV	D,A	; D=A=28
200A			RST1		; End

2) E=(D AND 2*C) - (B XOR 2B) when BC=605, D=1B

Address	HexCode	Label	Opcode	Operands	Comments
2000 2001 2002	 05 06		LXI	B,605	; B=6, C=5
2003 2004	 1B		MVI	D,1B	; D=1B
2005			MOV	A,B	; A=B=6
2006 2007	 2B		XRI	2B	; A=A XOR 2B=2D, S=0, Z=0, AC=0, P=1,CY=0
2008			MOV	B,A	; D=A=2D
2009			MOV	A,C	;A=C=5
200A			RAL		; A=A*2=A,CY=0
200B			ANA	D	; A=A AND D=A, S=0, Z=0,AC=0, P=1,CY=0
200C			SUB	B	A=A-B=DD, S=1, Z=0,AC=1, P=1,CY=1
200D			MOV	E,A	; D=A=DD
200E			RST1		; End

3) HL= (BC+HL) OR DE (use register pair when necessary), when BC=105h, HL=340h, DE=180h

Address	HexCode	Label	Opcode	Operands	Comments
2000 2001 2002	01 05 01		LXI	B,105	; BC=105
2003 2004 2005	21 40 03		LXI	H,340	; HL=340
2006 2007 2008	11 80 01		LXI	D,180	; DE=180
2009	09		DAD	B	; HL=HL+BC=445, CY=0
200A	7D		MOV	A,L	; A=L=45
200B	B3		ORA	E	;A=A+E=C5, S=1,Z=0,AC=0, P=1, CY=0
200C	6F		MOV	L,A	; L=A=C5
200D	7C		MOV	A,H	; A=H=04
200E	B2		ORA	D	; A=A+D=5, S=0,Z=0,AC=0, P=1, CY=0
200F	67		MOV	H,A	; H=A=5
2010	CF		RST1		; End

Write a program to solve the following equation: (The table content: Instruction, Function, Type, No. of bytes, and Effect on flags)

Q) $H = \overline{(L - 5)}\ XOR\ (D + (H/2))$, $L = 8,\ D = 3,\ H = 4$

Instruction	Function	Type	No. of Byte	Effect on Flag
LXI H,0408	; H=4, L=8	Data transfer	3	None
MOV D,03	; D=3	Data transfer	2	None
MOV A,L	; A=L	Data transfer	1	None
SUI 5	; A=A-5	Arithmetic	2	All
CMA	; $A = \overline{A}$	Logical	1	None
MOV L,A	; L=A	Data transfer	1	None
MOV A,H	; A=H	Data transfer	1	None
STC	; CY=1	Logical	1	CY only
CMC	; $CY = \overline{CY}$	Logical	1	CY only
RAR	; A=A/2	Logical	1	CY only
ADD D	; A=A+D	Arithmetic	1	All
XRA L	; A=A xor L	Logical	1	All
MOV H,A	; H=A	Data transfer	1	None
RST1	; END	restart	1	None

Homework

1. Write programs with effects
   $E=(H\ \ XOR\ \overline{C})\ OR\ (L - 20h)$ when H=0B1h, L=30h, C=45h
2. Reset bits 1,4 of A and set bits 0,3,6 when A=9Eh
3. What is the effect of the following instructions:
   ANA A, ORA A, CMA, XRA A when A=2Bh
4. What is the result of each instruction of the following program and its effect?

```
MVI     C,63
MOV     A,C
CMA
MOV     C,A
CMC

MVI     A,3D
ANA     C
XRI     25
XRA     A
ORI     20
STC
CMC
RAL
RST1
```