

Shubham Shekhaliya

U18CO018

Assignment-3(Tutorial)(MIT)

1-> WALP to check the forth bit of a byte stored at location 3000H is 0 or 1. If 0 store 00h else store FFH at location 3002H.

Code:-

mvi a,3Ah

sta 3000h

lxi h,3000h ; load data into memory from address 3000H

mov a ,m ; move in to the accumulator

lxi h,3002h ;where we want store the ans

ani 08h ; and with 2^3 as we want to check fourth bit

jnz noz ; if zero flag value is 0 which means 1
; else set 00

mvi m,00H

jmp end

noz: mvi m,8fH

end: hlt

Output:-

GNUSim8085 - 8085 Microprocessor Simulator

File Reset Assembler Debug Help

Registers

Register	Value
A	08
BC	00 00
DE	00 00
HL	30 02
PSW	00 00
PC	42 19
SP	FF FF
Int-Reg	00

Flag

Flag	Value
S	0
Z	0
AC	1
P	0
C	0

Decimal - Hex Conversion

Decimal: 0 Hex: 0

I/O Ports

0 - + 00

Update Port Value

Memory

0 - + 00

Update Memory

Load me at

```
1 mvi a,3Ah
2 sta 3000h
3 lxi h,3000h ; load data into memory from address 3000H
4 mov a ,m ; move in to the accumulator
5 lxi h,3002h ;where we want store the ans
6 ani 08h ; and with 2^3 as we want to check fourth bit
7 jnz noz ; if zero flag value is 0 which means 1
8 ; else set 00
9 mvi m,00H
10 jmp end
11 noz: mvi m,8fH
12 end: hlt
13
```

Memory

Address (Hex)	Address	Data
3000	12288	58
3001	12289	0
3002	12290	143
3003	12291	0
3004	12292	0
3005	12293	0
3006	12294	0
3007	12295	0
3008	12296	0
3009	12297	0
300A	12298	0

Line No Assembler Message

0 Program assembled successfully

Simulator: Idle

2-> Write Assembly language program to count the number of 1s in 8-bit number stored in register B.

Code:-

;let's store data in register B

mvi b, 4Ch ;data on we want ro check

mvi c,00H ;contains ans how many one's till

mvi d,08H ;store how many bits we want to check

mov a,b ;transfer in to accumulator

loop: rar ;rotate right with carry

jnc next ; if carry 0 then jump

inr c ; if carry 1 then increment ans

next: dcr d ; decrement ans by 1

jnz loop ; do loop till counter is zero

mov a,c ; store ans in to the accumulator

hlt

Output:-

GNUSim8085 - 8085 Microprocessor Simulator

File Reset Assembler Debug Help

Registers

Register	Value
A	03
BC	4C 03
DE	00 00
HL	30 02
PSW	00 00
PC	42 12
SP	FF FF
Int-Reg	00

Flag

Flag	Value
S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Conversion

Decimal: 0 Hex: 0

To Hex To Dec

I/O Ports

0 - + 00

Update Port Value

Memory

0 - + 00

Update Memory

Load me at

```
1 mvi b, 4Ch
2 mvi c, 00H
3 mvi d, 08H
4 mov a, b
5 loop: rar
6 jnc next
7 inr c
8 next: dcr d
9 jnz loop
10 mov a, c
11 hlt
```

Start: 3000h OK

Address (Hex)	Address	Data
3000	12288	58
3001	12289	0
3002	12290	143
3003	12291	0
3004	12292	0
3005	12293	0
3006	12294	0
3007	12295	0
3008	12296	0
3009	12297	0
300A	12298	0

Line No Assembler Message

0 Program assembled successfully

Simulator: Idle

3-> There is an array of some elements. Write Assembly language program to count number of elements that are lesser than 09H.

Code:-

```
lxi h,07D0H
```

```
mvi c,00H
```

```
mov d, m ;d register contains counter
```

```
next: inx h
```

```
mov a,m
```

```
cpi 09H ;compare with 09
```

```
jnc skip ;if greater
```

```
inc c ;if smaller then increment ans
```

```
skip: dcr d
```

```
jnz next
```

```
mov a,c ;store result into the accumulator
```

```
hlt
```

Output:-

The screenshot displays the GNUSim8085 - 8085 Microprocessor Simulator interface. The main window is titled "GNUSim8085 - 8085 Microprocessor Simulator". The interface includes a menu bar (File, Reset, Assembler, Debug, Help) and a toolbar with various icons. The central area is divided into several panels:

- Registers:** A table showing the state of the 8085 registers. The PC register is highlighted in red, indicating the current instruction address.
- Flag:** A table showing the status of the 8085 flags (S, Z, AC, P, C).
- Decimal - Hex Conversion:** A panel with input fields for decimal and hex values and buttons for conversion.
- I/O Ports:** A panel with input fields for port values and buttons for updating.
- Memory:** A panel with input fields for memory address and value, and buttons for updating.

The **Assembler** panel shows the assembly program being loaded. The program is as follows:

```
1 lxi h,07D0H
2 mvi c,00H ;d register contains counter
3 mov d, m
4 next: inx h
5 mov a,m
6 cpi 09H ;compare with 09
7 jnc skip ;if greater
8 inc c ;if smaller then increment ans
9 skip: dcr d
10 jnz next
11 mov a,c ;store result into the accumulator
12 hlt
13
```

The **Memory** panel shows the memory dump starting at address 3000h. The data is as follows:

Address (Hex)	Address	Data
3000	12288	58
3001	12289	0
3002	12290	143
3003	12291	0
3004	12292	0
3005	12293	0
3006	12294	0
3007	12295	0
3008	12296	0
3009	12297	0
300A	12298	0

The **Assembler Message** panel shows the following message:

```
0 Program assembled successfully
```

The status bar at the bottom indicates "Simulator: Idle".