# EXPERIMENT – 3

**Program -1**

**Aim:** Write a program to perform selective set operation on data stored at 4000H with the data stored at 4001H and store the result at 4002H. Verify the result and write bite wise operation of this program. (OR)

## Code:

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AX, 0400H ; 4000 is base address stored in AX MOV DS, AX

MOV DI, 0000H ; offset is set to zero

MOV [DI], 0001B ; binary of one is moved in DI MOV BX, [DI] ; value is moved to BX

MOV DI, 0001H

MOV [DI], 0010B ; binary value of two is moved to DI MOV CX, [DI]

OR BX, CX ; Destination(BX) will store the result of OR operation

MOV DI, 0002H MOV [DI], BX

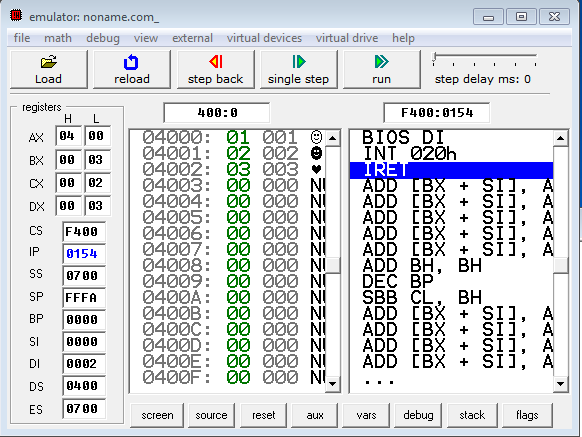
MOV DX, [DI] ; result is stored in DX

ret

## Input / Output:

Input: BX: 0001B , CX: 0010B

Output: DX: 3



# EXPERIMENT – 3

**Program -2**

**Aim:** Write a program to perform selective compliment operation on data stored at

4000H corresponding to the data stored at 4001H and store the result at 4002H. Verify the result and write bite wise operation of this program. (XOR)

## Code:

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AX, 0400H ; 4000 is base address stored in AX MOV DS, AX

MOV DI, 0000H ; offset is set to zero

MOV [DI], 0101B ; binary of five is moved in DI MOV BX, [DI] ; value is moved to BX

MOV DI, 0001H

MOV [DI], 0110B ; binary value of six is moved to DI MOV CX, [DI]

XOR BX, CX ; Destination(BX) will store the result of XOR operation

MOV DI, 0002H

MOV [DI], BX

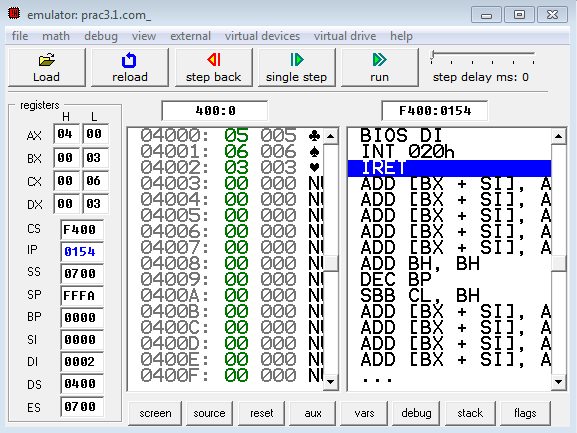
MOV DX, [DI] ; result is stored in DX

ret

## Input / Output:

Input: BX: 0101B , CX: 0110B

Output: DX: 3



# EXPERIMENT – 3

**Program -3**

**Aim:** Write a program to perform selective clear operation on data stored at 4000H corresponding to the data stored at 4001H and store the result at 4002H.Verify the result and write bite wise operation of this program. ( A AND B’)

## Code:

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AX, 0400H ; 4000 is base address stored in AX MOV DS, AX

MOV DI, 0000H ; offset is set to zero

MOV [DI], 0101B ; binary of five is moved in DI MOV BX, [DI] ; value is moved to BX

MOV DI, 0001H

MOV [DI], 0110B ; binary value of six is moved to DI MOV CX, [DI]

NOT CX ; not of CX is performed

AND BX, CX ; Destination(BX) will store the result of AND operation

MOV DI, 0002H MOV [DI], BX

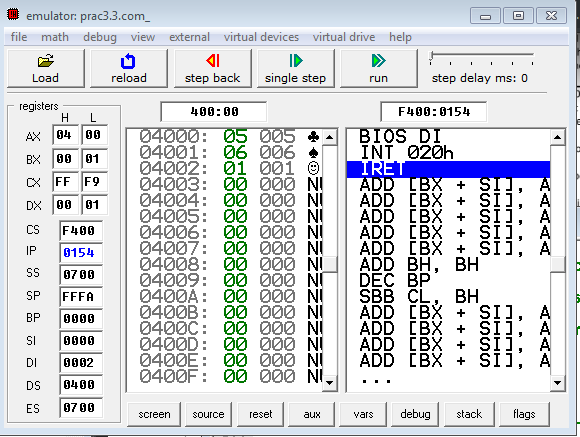
MOV DX, [DI] ; result is stored in DX

ret

## Input / Output:

Input: BX: 0101B , CX: 0110B

Output: DX: 1



# EXPERIMENT – 3

**Program -5**

**Aim:** Write a program to multiply & divide the number stored at 4000H by 2 and store the result at 4001H & 4002H .(Use Shift instructions).

**Code:**

; You may customise this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

MOV BX,0400h

MOV DS,BX

MOV [00h],19h ; 4000H memory location will have 19h

MOV AL,[00h] ; AL will have 19h

SAL AL,1 ; Shift Arithmetic left will be applied on Al by 1 (multiply by 2)

MOV [01h],AL ; now result i.e. 36 is stored at 4001H memory location

MOV AL,00h

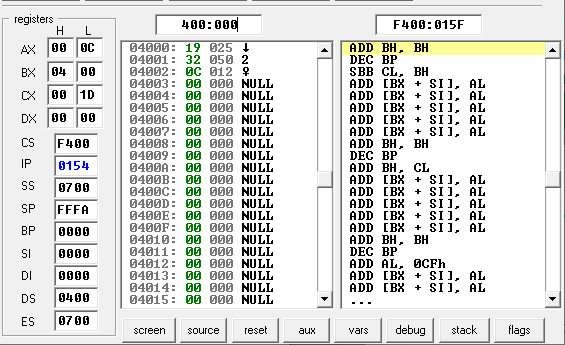
MOV AL,[00h]

SAR AL,1 ; Shift arithmetic right will divide 36 by 2

MOV [02h],AL ; memory location 4002H will have result of division stored in it

ret

**Input / Output:** Input: AL:19h Output: [4002h]: 0C



# EXPERIMENT – 3

**Program -6**

**Aim:** Write a Program to subtract the contents of memory location 4001H from the memory location 4002H and place the result in memory location 4003H without SUB instruction.

**Code:**

; You may customise this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt org 100h

; COMPLEMENTING THE NUMBER AND THEN ADDING WILL GIVE RESULT SAME AS SUBTRACTION

MOV AX, 0400H ; base address is stored here

MOV DS, AX MOV DI, 0001H MOV [DI], 9H MOV AX, [DI]

NOT AX ; PERFORM 2'S COMPLEMENT OF 9H INC AX

MOV DI,0002H MOV [DI], 15H MOV BX, [DI]

ADD AX,BX ; PERFORMS SUBTRACTION INDIRECTLY

MOV DI, 0003H MOV [DI], AX

MOV DX, [DI] ; RESULT OF SUBTRACTION IS STORED IN DX

**Input / Output:**

Input: AX: 9H, BX: 15H

Output: DX: 0C

# EXPERIMENT – 3

**Program -7**

**Aim:** Implement a program to mask the lower four bits of content of the memory location.

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

; Masking of four lower bits

MOV AX, 400H

MOV DS, AX

MOV DI, 000H

MOV [DI], 26H

MOV BL, [DI]

AND BL, 0FH ; here to make last four lower bits as the same digit as the last digit of the content we AND it with 0F

; 0010 0110

; AND 0000 1111

ret

**Input/Output:**

Input: AX: 26h , mask value(0F)

Output: BL: 06h

****

# EXPERIMENT – 3

**Program -8**

**Aim:** Implement a program to set higher four bits of content of the memory location to 1.

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AX, 400H

MOV DS, AX

MOV DI, 000H

MOV [DI], 26H

MOV BL, [DI]

AND BL, 0FH ; TO MASK LOWER BIT AS THE SAME OF LASTDIGIT OF THE CONTENT

OR BL, 10H ; TO GET ONE ON HIGHER FOUR BYTE

; 0010 0110

;AND 0000 1111

;OR 0001 0000

ret

**Input/Output:**

Input: AX: 26h

Output: BL:16h

****

# EXPERIMENT – 3

**Program -9**

**Aim:** Calculate the sum of series of numbers (Data set-1) from the memory location listed

below & store the result at 400AH location.

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AL, 0

MOV BX, 0

MOV CX, 5 ; COUNTER VALUE OR LENGTH OF AN ARRAY

NEXT:

ADD AL, ARRAY[BX] ; ADDS EACH ELEMENT OF ARRAY

INC BX ; INCREMENTS BX

LOOP NEXT

MOV SUM, AL

MOV DX, 0400H

MOV DS, DX

MOV DI, 00AH

MOV [DI], AL

MOV DX, [DI] ; STORES RESULT IN 4000A LOCATION

array db 1,2,3,4,5 ; DEFINES ARRAY WITH 5 ELEMENTS

SUM db 0 ; INITAL VALUE OF SUM IS ZERO

ret

**Input/Output:**

Input: array: 1,2,3,4,5

Output: sum: 0F





# EXPERIMENT – 3

**Program -10**

**Aim:** Modify above the program such a way that it halts the execution if carry generated

& stores the intermediate result at 400AH location. (Data set-2) (Note: Student need to

implement FOR loop in this program: initialization, Compare, Decrement/Increment; also need

to use JMP, JMx instructions.)

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AL, 0

MOV BX, 0

MOV CX, 5 ; COUNTER VALUE OR LENGTH OF AN ARRAY

NEXT:

ADD AL, ARRAY[BX] ; ADDS EACH ELEMENT OF ARRAY

JC CARRY

INC BX ; INCREMENTS BX

LOOP NEXT

CARRY:

MOV SUM,AL

MOV DX, 0400H

MOV DS, DX

MOV DI, 00AH

MOV [DI], AL

MOV DX, [DI] ; STORES RESULT IN 4000A LOCATION

array db 71,72,73,74,75 ; DEFINES ARRAY WITH 5 ELEMENTS

SUM db 0 ; INITAL VALUE OF SUM IS ZERO

ret

**Input/Output:**

Input: array: 71,72,73,74,75

Output: sum:22h





# EXPERIMENT – 3

**Program -11**

**Aim:** Multiply two 8-bit numbers stored in memory locations 4001H and 4006H by

repetitive addition and store the result at 400AH location.(Use Data Set -3) (Note: Student need

to implement FOR loop in this program: initialization, Compare, Decrement/Increment; also

need to use JMP, JMx instructions.)

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AL, 0

MOV BX, 0 ; INITALISING I=0

MOV DX, 0400H

MOV DS, DX

MOV DI, 000H

MOV [DI], 02H ; VALUE 2 WILL BE STORED IN DL

MOV DL, [DI]

MOV DI, 006H ; 4006 WILL STORE CX VALUE OR COUNTER VALUE

MOV [DI], 04

MOV CX, [DI] ; INITIALISING TH COUNTER

NEXT:

ADD AL, DL

INC BX

LOOP NEXT

MOV SUM,AL

SUM DB 0

MOV DI, 00AH

MOV [DI], AL

MOV DX, [DI]

ret

**Input/Output:**

Input: DL: 02h , CX: 04h

Output: AL: 08h

****

****

# EXPERIMENT – 3

**Program -12**

**Aim:** Program to find average of n numbers

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

MOV AL, 0

MOV BX, 0

MOV CX, 5 ; COUNTER VALUE OR LENGTH OF AN ARRAY

NEXT:

ADD AL, ARRAY[BX] ; ADDS EACH ELEMENT OF ARRAY

INC BX ; INCREMENTS BX

LOOP NEXT

MOV SUM , AL

MOV CL, 05

DIV CL

MOV AVG, AL

array db 1,2,3,4,5 ; DEFINES ARRAY WITH 5 ELEMENTS

SUM db 0 ; INITAL VALUE OF SUM IS ZERO

AVG db 0 ; stores average value

ret

**Input/Output:**

Input: array: 1,2,3,4,5

Output: sum: 0Fh, avg: 03h

****

# EXPERIMENT – 3

**Program -13**

**Aim:** Write an assembly language program to find the no. of odd numbers and even

numbers, given an array of n numbers.

**Code:**

; You may customize this and other start-up templates;

; The location of this template is c:\emu8086\inc\0\_com\_template.txt

org 100h

; add your code here

VECTOR DB 1,2,3,4,5,6,7,8,9,10

    COUNTE DB 0 ; counter for even number

    COUNTO DB 0 ; counter for odd number

    MOV AL,0

    MOV BX,0

    MOV CX,10

    MOV DL,02H

    NEXT:

    MOV AH,0

    MOV AL,VECTOR[BX]

    DIV DL ; check for even and odd

    INC BX

    CMP AH,0 ; if zero or not

    JZ EVEN ; jump on zero if even

    JNZ ODD ; jump on non zero if odd

    LOOP NEXT

    EVEN: ; for even no

    INC COUNTE

    DEC CX

    JNZ NEXT

    HLT

    ODD: for odd number

    INC COUNTO

    DEC CX

    JNZ NEXT

**Input/Output:**

Input: vector: 1,2,3,4,5,6,7,8,9,10

Output: counte: 5h, counto 5h

