**EXPERIMENT – 5**

**Program – 1**

**Aim:** Write a program which sets the parity bit.

**Code:**

.model prog-5.1

.data

var db 0100b ; setting the flags so PF is at the last third position so it sets Parity Flag

.code

mov ax, @data

mov ds, ax

mov ah, var

sahf; load from register to flag

; lahf- loads flag to register

end

**Input/Output:**

Input: var – 0100b

Output: PF: 1

**EXPERIMENT – 5**

**Program – 2**

**Aim:** Write a program which transfers content of Flags to Register

**Code:**

.model prog-5.2

.data

var db 0111b

.code

mov ax, @data

mov ds, ax

mov ah,var ; 7 is initially stored in variable

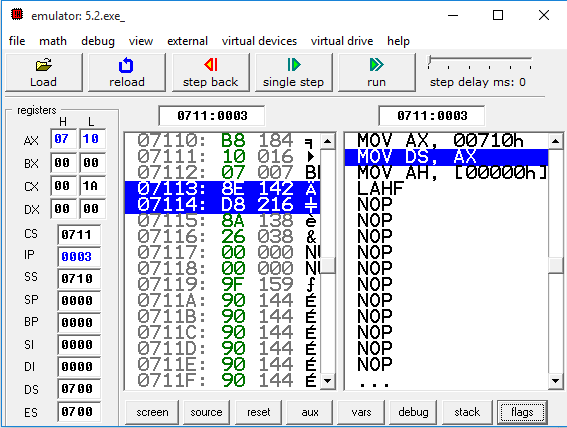
lahf; loads flags value in AH

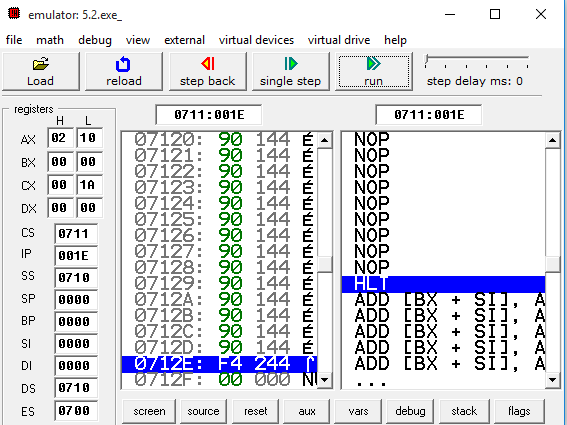
end

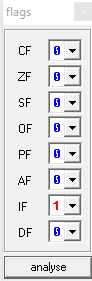
**Input/Output:**

Input: var – 0111b , ah - 7

Output: ah – 02h



****

****

**EXPERIMENT – 5**

**Program – 3**

**Aim:** Write a program to add the two Hex Numbers 7AH and 46H and to store the sum at

memory location 2098 and flags status at 2097 location.

**Code:**

.model prog-5.3

.data

var db 7Ah

var1 db 46h

sum db 1 DUB(?)

flag db 1 DUB(?)

.code

mov ax, @data

mov ds, ax

mov al, var

add al, var1 ; adds two variables

mov sum, al ; result is transfered in sum variable

lahf ; flag is stored ah register

mov flag, ah

mov bx, 200h

mov ds, bx

mov di, 0098h

mov [di], al ; sum result stored in 2098h

mov di, 0097h

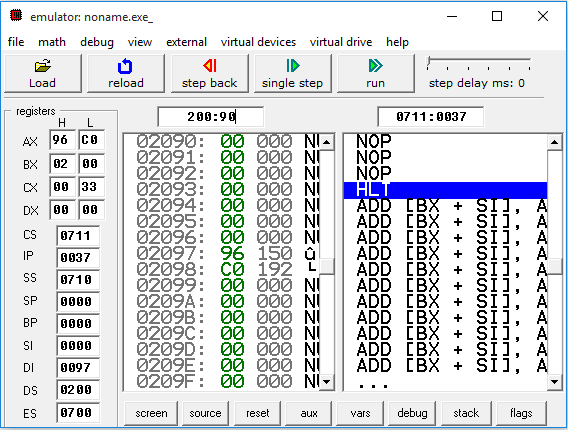
mov [di], ah ; flag is stored in 2097h

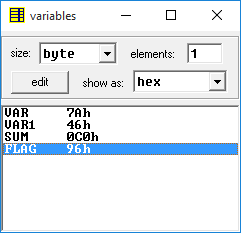
end

**Input/Output:**

Input: var - 7Ah , var1 – 46h

Output: sum – 0C0h, flag – 96h

****

** **

**EXPERIMENT – 5**

**Program – 5**

**Aim:** Using a Subroutine, write a program which adds two hex number 10H and F0H and

store result at 2040H location in memory. At the end of subroutine, clear the flag Z without

affecting other flags and return to main program.

**Code:**

.model prog-5.5

.data

var db 10h

var1 dw 00F0h

result dw 1 DUP (?)

.code

mov ax, @data

mov ds, ax

call sum ; clals procedure

sum proc ; declare a procedure

mov al, var

add ax, var1 ; addition of two variables

mov result, ax

mov bx, 200h

mov ds, bx

mov di, 0040h ; stores result at 2040h

mov [di], ax

lahf

and ah, 10111111b; clears the zero flag without affecting other flag

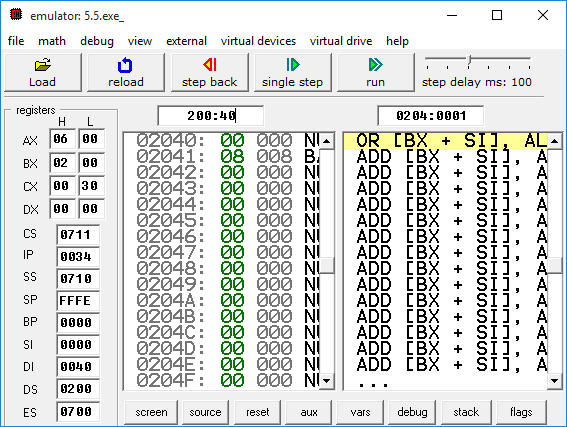
sum endp ; end a procedure

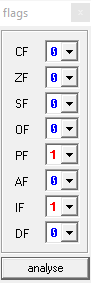
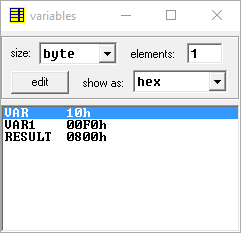
end

**Input/Output:**

Input: var – 10h , var1 – 00F0h

Output: result – 0800h

****

****

**EXPERIMENT – 5**

**Program – 6**

**Aim:** Write a program which set and resets zero flag at next iteration. (take number of

iteration equal to 5)

**Code:**

.model prog-5.6

.code

mov bx, 0

mov cx, 5

mov ax, 0111111b

sahf ; loads from register to flag

next:

and ah, ax ; initial value of ah and ax

loop next

end

**Input/Output:**

Input:

Output:

**EXPERIMENT – 5**

**Program – 8**

**Aim:** Implement a program to reverse a string using stack operations and stored in same

memory area.

**Code:**

.model prac5-8

.data

var dw 'C','S','P','I','T'

.code

mov ax,@data

mov ds,ax

mov cx,10

mov bx,0

Next:

mov ax,var[bx]

push ax ;pushing single elements into stack

inc bx

loop Next

mov cx,10

mov bx,0

Rev:

pop ax

mov var[bx],ax ;poping the elements from the stack and

inc bx ;placing them into same variable

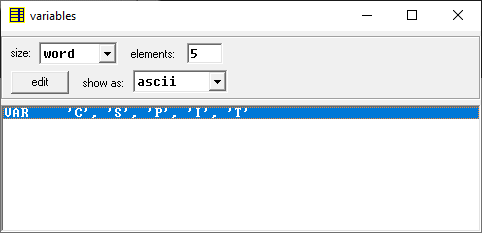
loop Rev

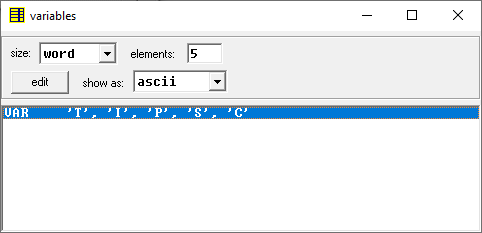
end

**Input/Output:**

Input: var-CSPIT

Output: var-TIPSC





**EXPERIMENT – 5**

**Program – 9**

**Aim:** Calculate the sum of series of even numbers from the list of numbers. The length of

the list is in memory location 2200H and the series itself begins from memory location 2201H.

Assume the sum to be 8 bit number so you can ignore carries and store the sum at memory

location 2210H.

**Code:**

.model prac5.9

.code

mov bx,200h

mov ds,bx

mov bx,5

mov [200h],bx

mov di,201h

mov bx,0

mov [222h],bx

mov bx,3;

mov cx,5

next:

mov [di],bx

inc bx

inc di

loop next

mov di,201h;

mov cx,5 ; counter set to 5

check:

mov bl,2 ; check for even numbers

mov al,[di]

mov dh,[di]

div bl

cmp ah,[222h]

jz sum

jnz end

sum:

add [210h],dh ; for addition

jmp end

end:

inc di

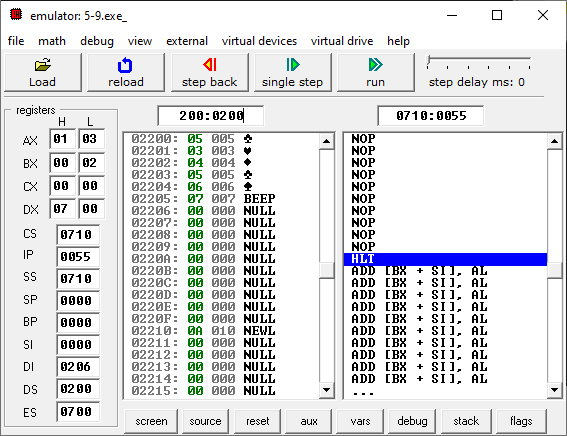
loop check

end

**Input/Output:**

Input: cx- 1,2,3,4,5

Output: al-3, bl-2



**EXPERIMENT – 5**

**Program – 10**

**Aim:** Write an assembly language program to arrange an array of 10 data in ascending

order. The length of the list is in memory location 2200H and the series itself begins from memory location 2201H

**Code:**

model prac5-10

.code

mov bx,200h

mov ds,bx

mov di,201h

mov bx,20

mov cx,5

giving:

mov [di],bx

dec bx

inc di

loop giving

mov di,201h

mov bx,201h

mov cx,5

outerloop:

innerloop:

mov ah,[di]

mov dh,[bx] ;compare and if greater then go

cmp dh,ah ; for change

jc change

jnc exit

change:

mov dh,[bx]

mov [bx],ah ;swaping elements

mov [di],dh

exit:

mov si,205h

cmp si,bx ;storing the value

jz abc

inc bx

jmp innerloop

abc:

inc di

mov bx,di

inc bx

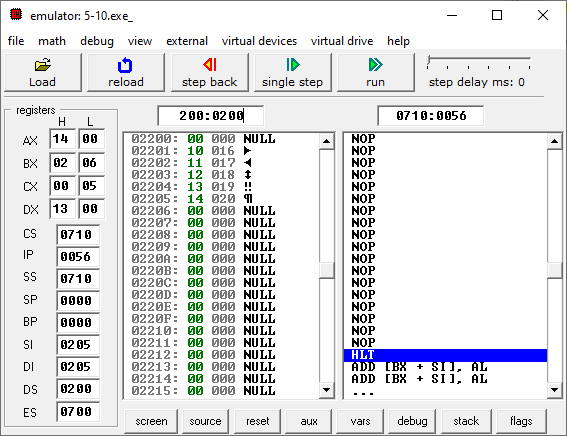
mov si,205h

cmp si,di

jnz outerloop

end

**Input/Output:**



**EXPERIMENT – 5**

**Program – 11**

**Aim:** Write an assembly language program to fill the memory locations starting from

3000h, with n Fibonacci numbers.

**Code:**

.model prac5-11

.code

mov bx,300h

mov ds,bx

mov di,00h

mov cx,100

mov [di],0

mov [di+1],1

giving:

mov ax,[di]

mov bx,[di+1]

adc ax,bx ;adding the last and next number

mov [di+2],ax ;storing the value at next location

inc di

loop giving

end

**Input/Output:**

