1. Write an assembly language program to add two 16-bit numbers in 8086.

**CODE:**

MOV BX, [1100H]

MOV CX, [1102H]

ADD BX,CX

MOV [1106H], BX

HLT

**INPUT:**

[1100H] <- 34

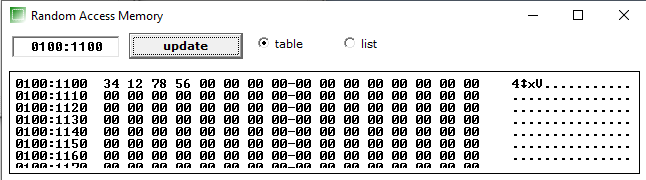
[1101H] <- 12

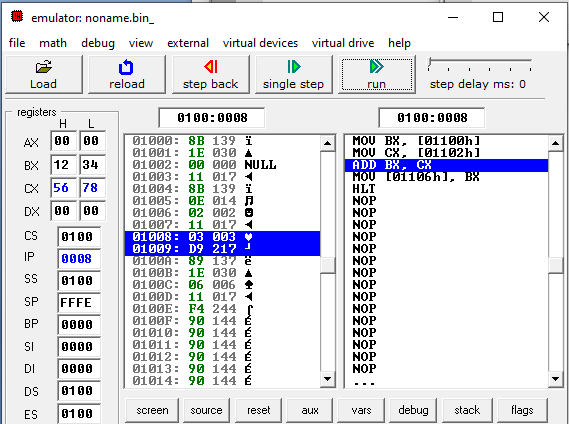
BX <- 1234

[1102H] <- 78

[1103H] <- 56

CX <- 5678



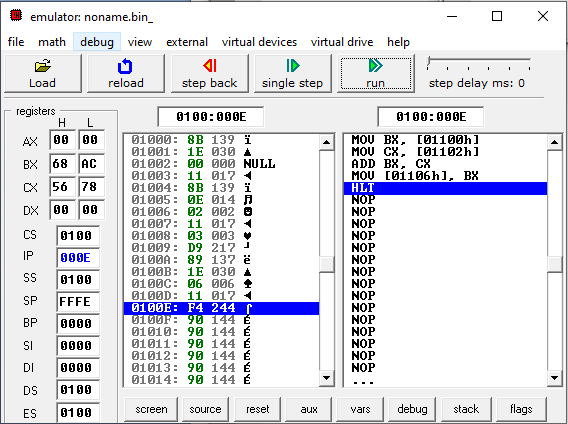


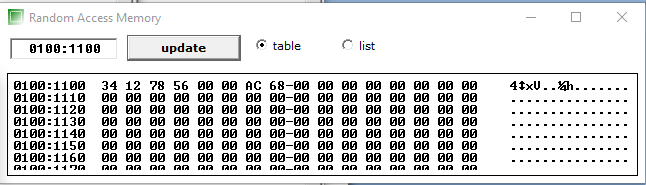
**OUTPUT:**

[1106H] <- AC

[1107H] <- 68

BX <- 68AC





2. Write an assembly language program to subtract two 16-bit numbers in 8086.

**CODE:**

MOV BX, [1100H]

MOV CX, [1102H]

SUB BX,CX

MOV [1106H], BX

HLT

**INPUT**:

[1100H] <- 34

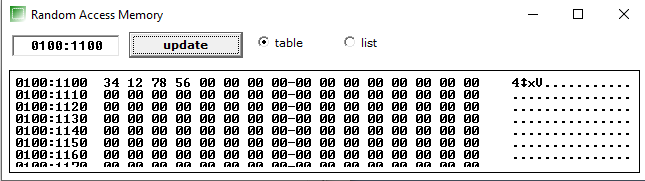
[1101H] <- 12

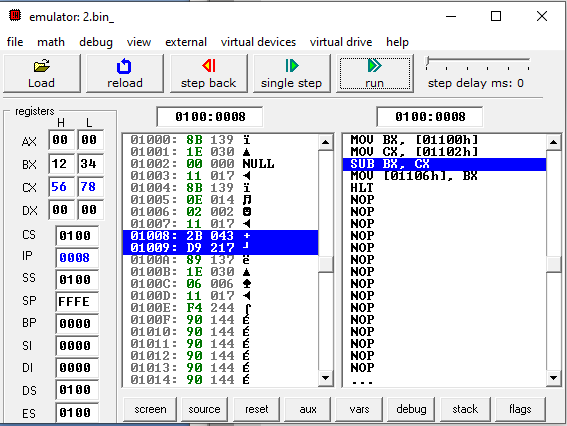
BX <- 1234

[1102H] <- 78

[1103H] <- 56

CX <- 5678



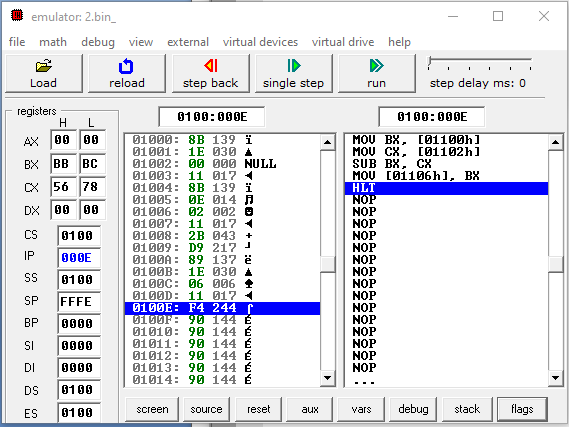


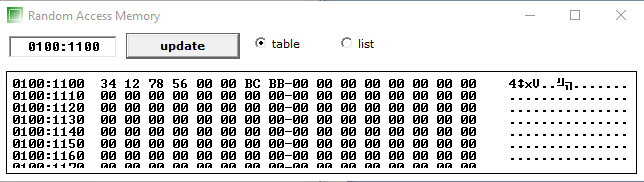
**OUTPUT**:

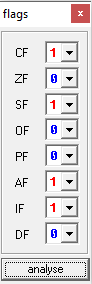
[1106H] <- BC

[1107H] <- BB

BX <- BBBC







3. Write an assembly language program to multiply two 16-bit numbers in 8086.

**CODE**:

MOV AX, [1100H]

MOV BX, [1102H]

MUL BX

MOV [1106H], AX

MOV [1108H], DX

HLT

**INPUT**:

[1100H] <- 11

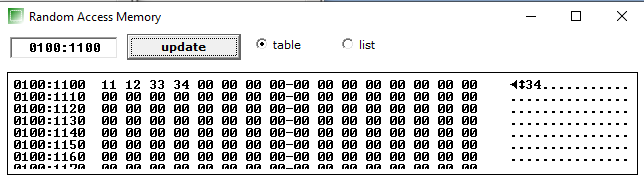
[1101H] <- 12

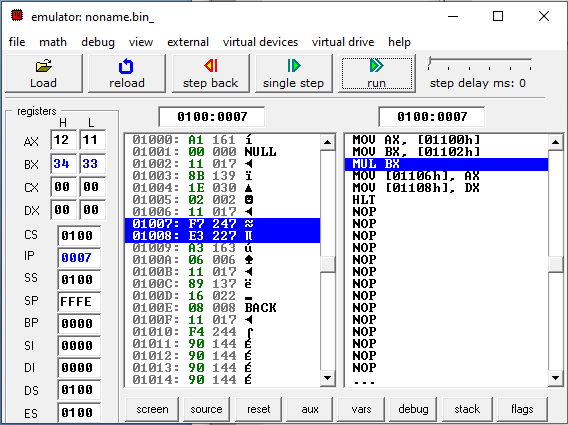
AX <- 1211

[1102H] <- 33

[1103H] <- 34

BX <- 3433





**OUTPUT**:

[1106H] <- 63

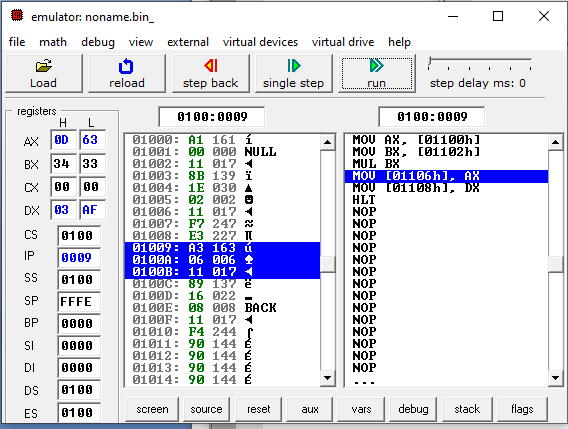
[1107H] <- 0D

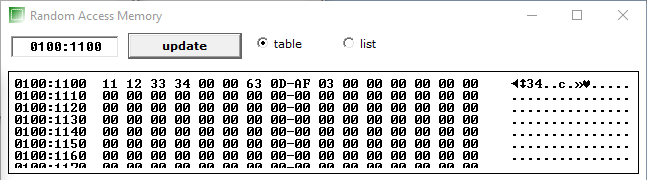
AX <- 0D63

[1108H] <- AF

[1109H] <- 03

DX <- 03AF







4. Write an assembly language program to divide two 16-bit numbers in 8086.

**CODE**:

MOV AX, [1100H]

MOV BX, [1102H]

DIV BX

MOV [1106H], AX

MOV [1108H], DX

HLT

**INPUT**:

[1100H] <- 74

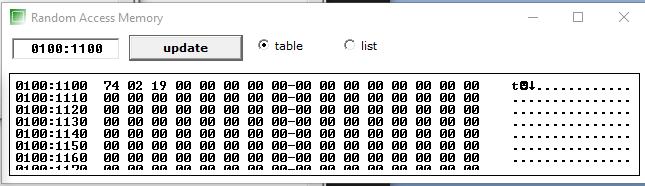
[1101H] <- 02

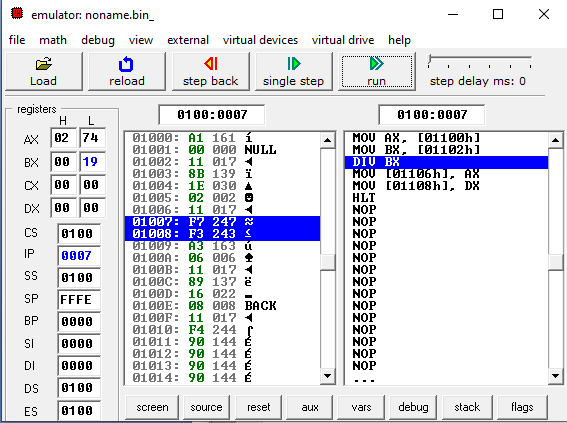
BX <- 0274 (628 in DEC)

[1102H] <- 19

[1103H] <- 00

CX <- 0019 (25 in DEC)





**OUTPUT**:

[1106H] <- 19

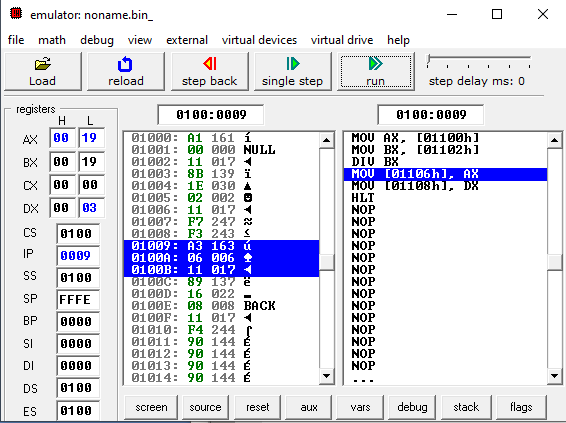
[1107H] <- 00

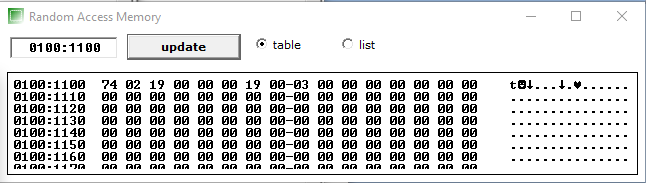
AX <- 0019

[1108H] <- 03

[1109H] <- 00

DX <- 0003





5. Write an assembly language program to demonstrate AAA, AAS, AAM, AAD, DAA and DAS in 8086.

**AAA:**

**CODE**:

mov AL,'6'

add AL,'7'

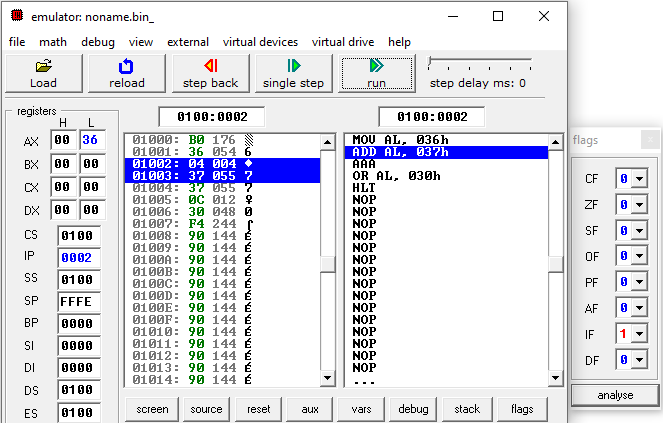
aaa

or AL,30H

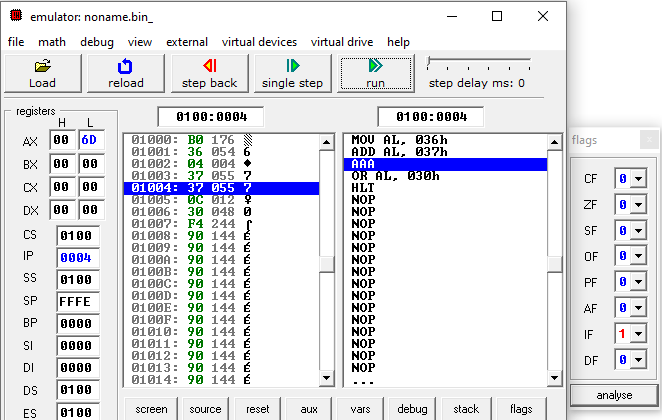
hlt

**OUTPUT**:

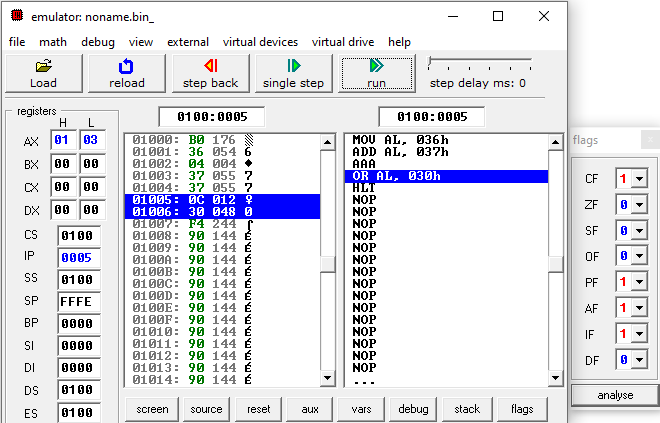
AL:=36



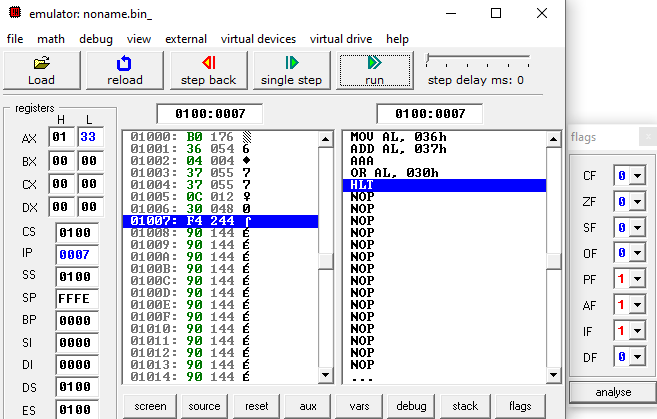
AL:=36H+37H=6DH



AAA; AX:=0103H



AL:=33H



**AAS:**

**CODE**:

mov AL,'6'

sub AL,'7'

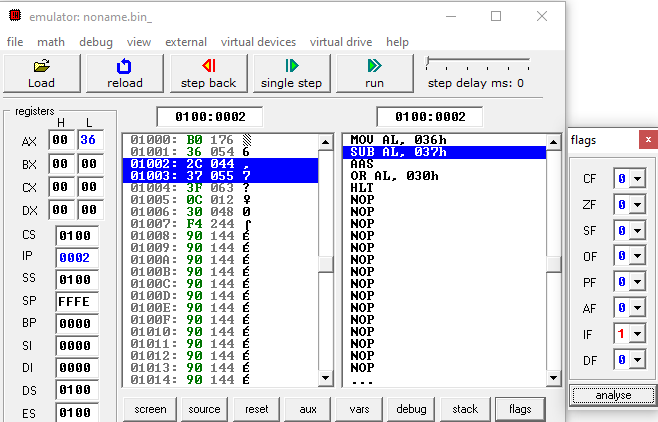
aas

or AL,30H

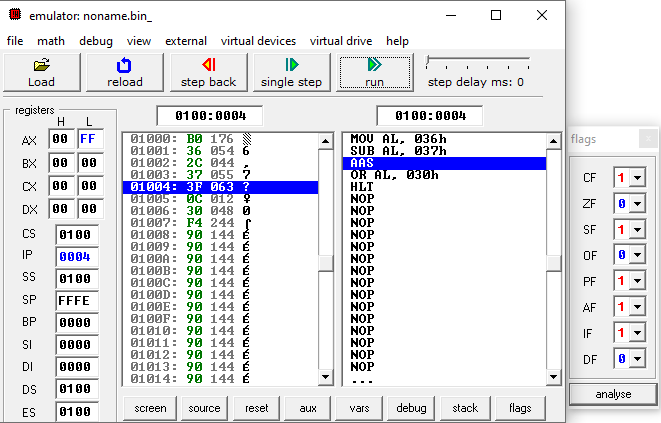
hlt

**OUTPUT**:

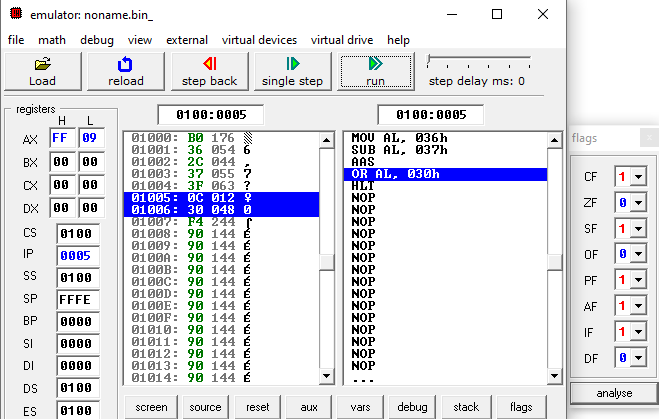
AL:=36



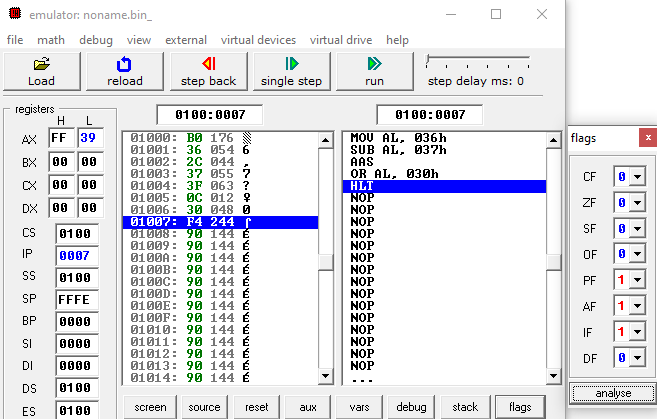
AL:=36H-37H=FFH



aas; AX:=FF09H



AX:=FF39H



**AAM:**

**CODE**:

mov AL,'3'

mov BL,'9'

and AL,0FH

and BL,0FH

mul BL

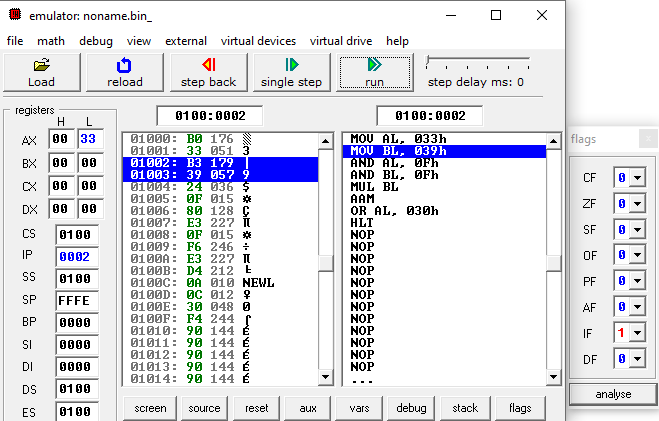
aam

or AL,30H

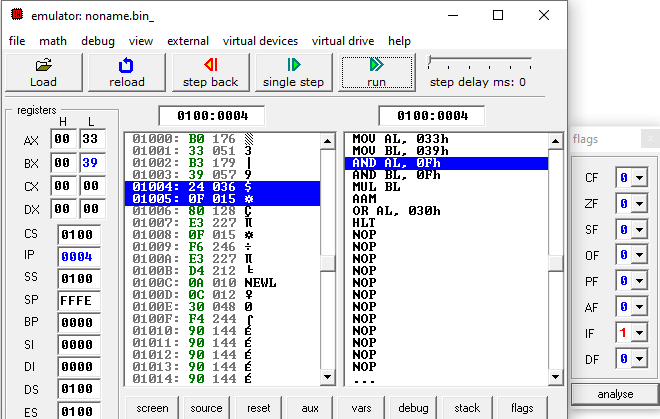
hlt

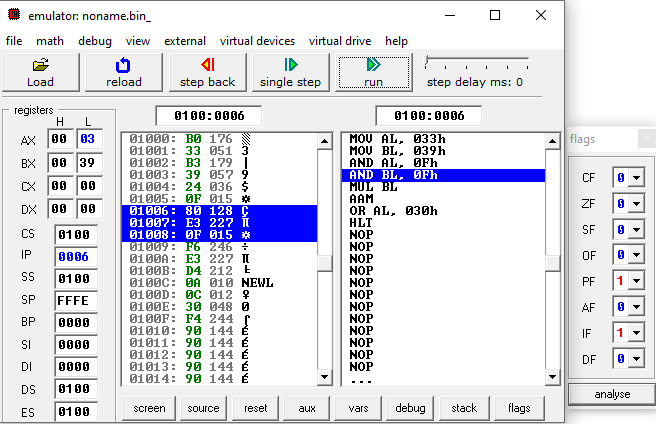
**OUTPUT**:

AL:=33H

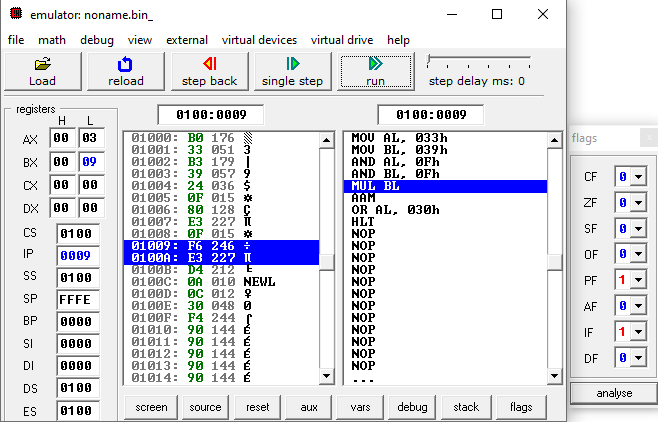


BL:=39H

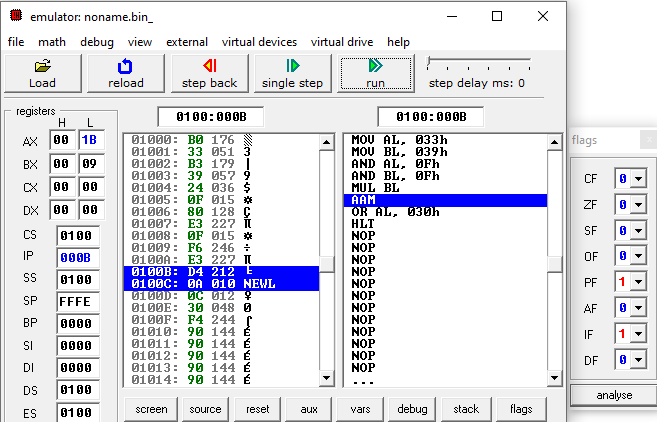


AL:=33H AND 0FH = 03H

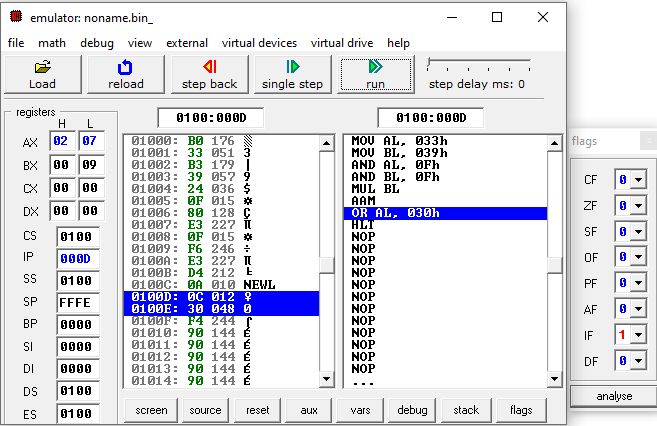
BL:=39H AND 0FH = 09H



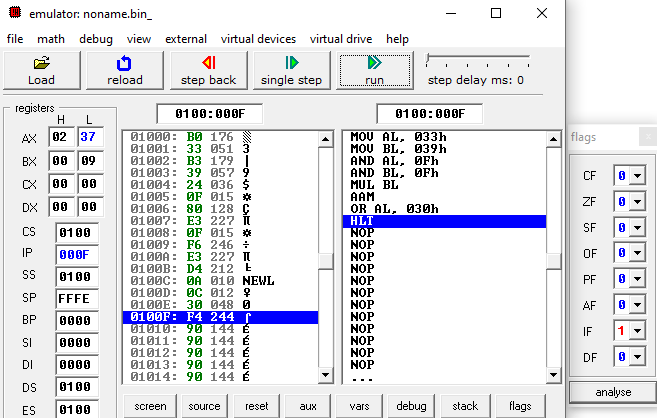
AL:=AL MUL BL = 1BH

****

aam; AX:=0207H



AL:= AL OR 30H=37H



**AAD:**

**CODE**:

mov AX,0207H

mov BL,05H

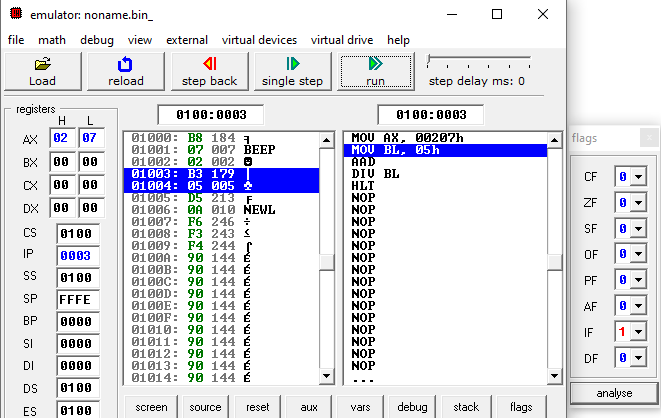
aad

div BL

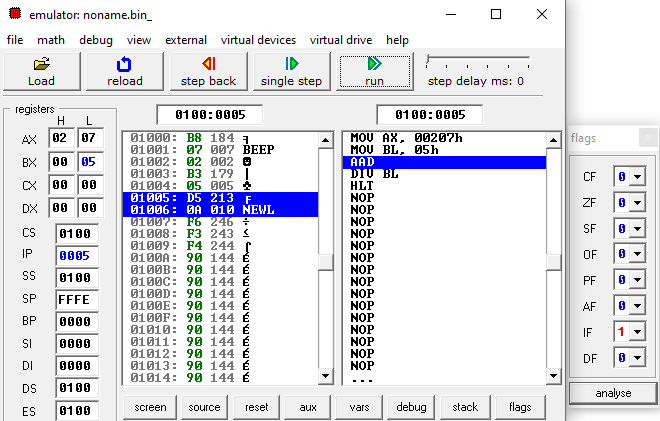
hlt

**OUTPUT**:

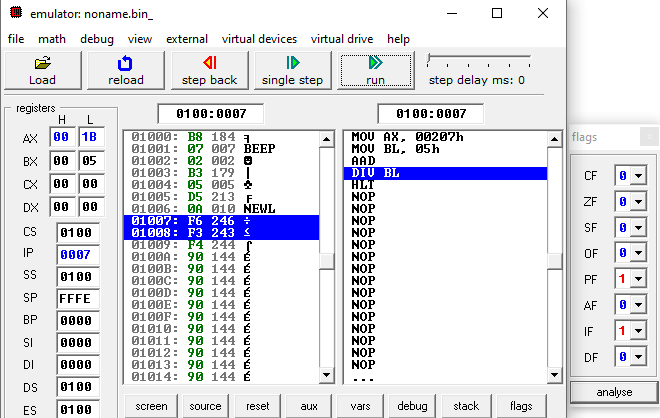
AX:=0207H



BL:=05H

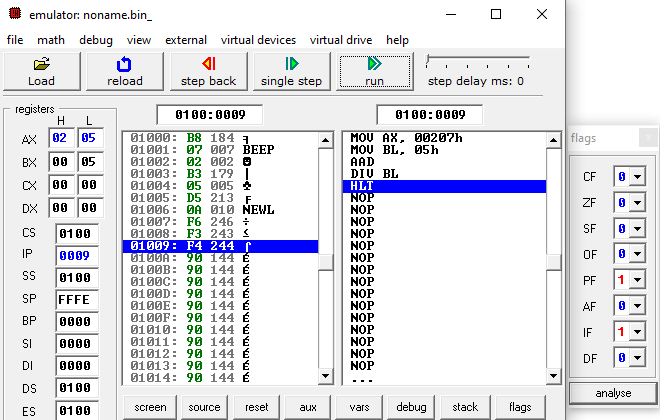


aad; AX:=001BH



AL DIV BL

AH:=REMAINDER; AL:=QUOTIENT



**DAA:**

**CODE**:

MOV AL, 71H

ADD AL, 43H

DAA

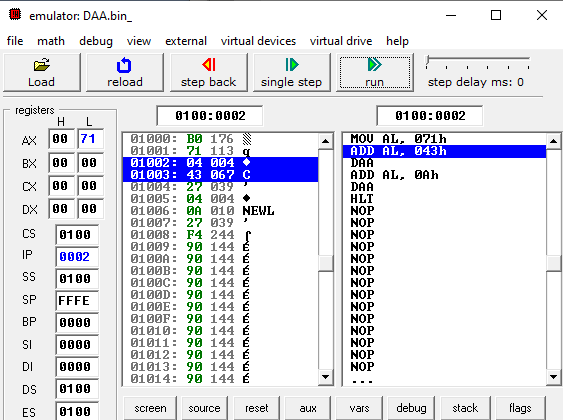
ADD AL, 0AH

DAA

HLT

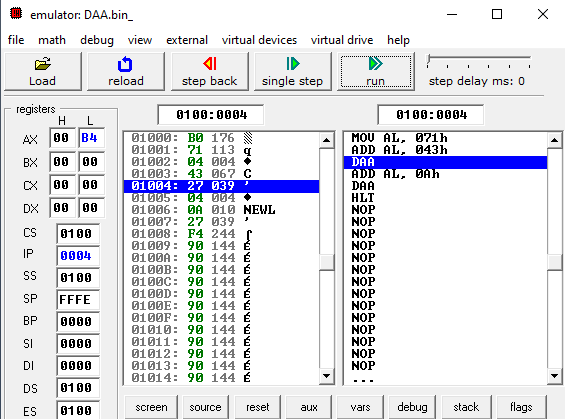
**OUTPUT**:

AL <- 71H



AL <- AL+43H

AL<- B4



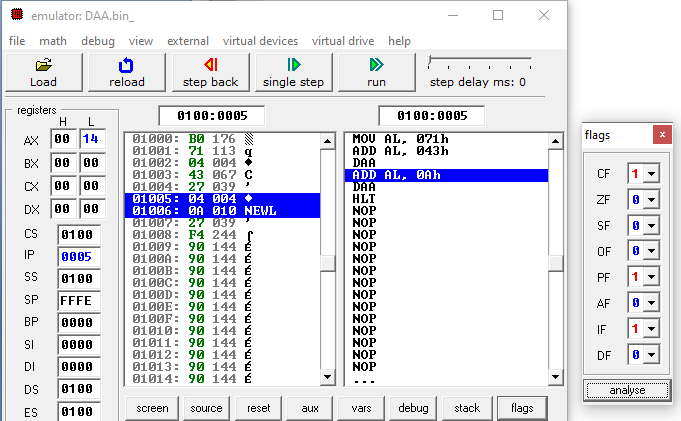
DAA

MOST SIGNIFICANT 4 bits IN AL>9,

So

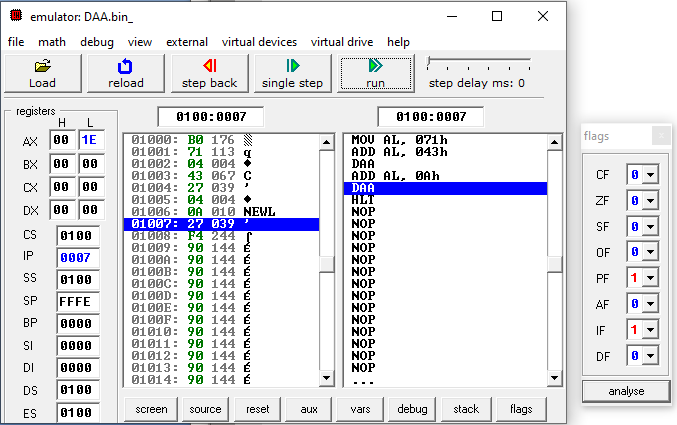
AL<- AL+60H;

AL<- 14H ; CY <- 1



AL <- AL+0AH

AL<-1E



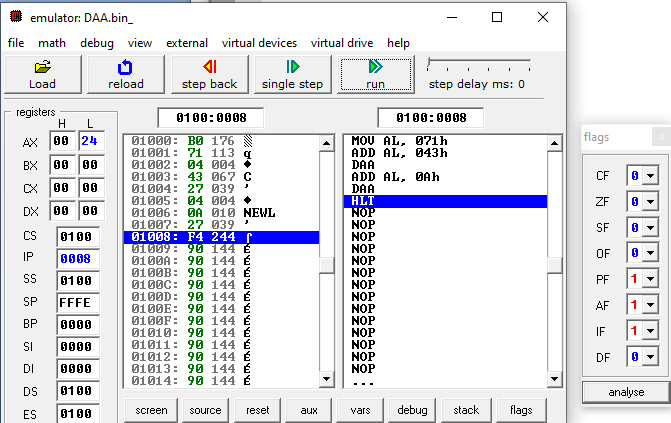
DAA

LEAST SIGNIFICANT 4 Bits IN AL>9,

So,

AL <- AL+06H

AL<- 24; AF=1



**DAS:**

**CODE**:

MOV AL, 10101011B

SUB AL, 03H

DAS

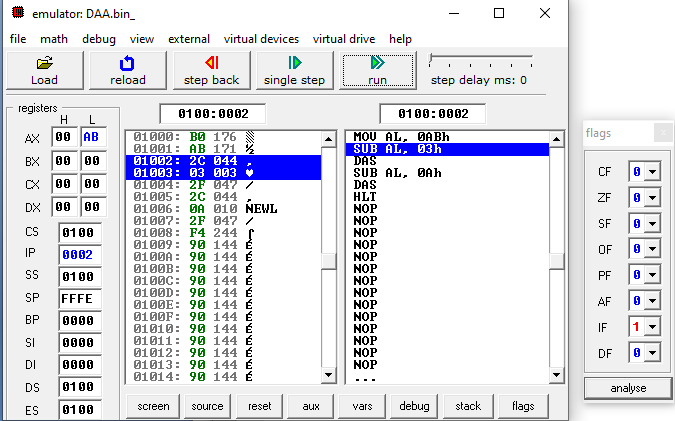
SUB AL, 0AH

DAS

HLT

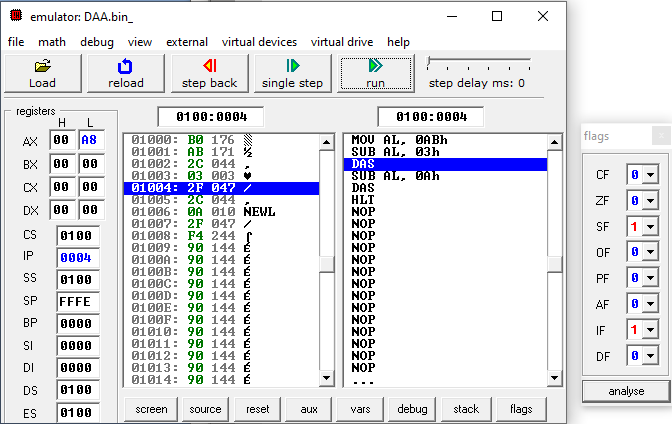
**OUTPUT**:

AL<-10101011B (EQUIVALENT TO AB IN HEX)



AL <- AL - 03H

AL<- A8



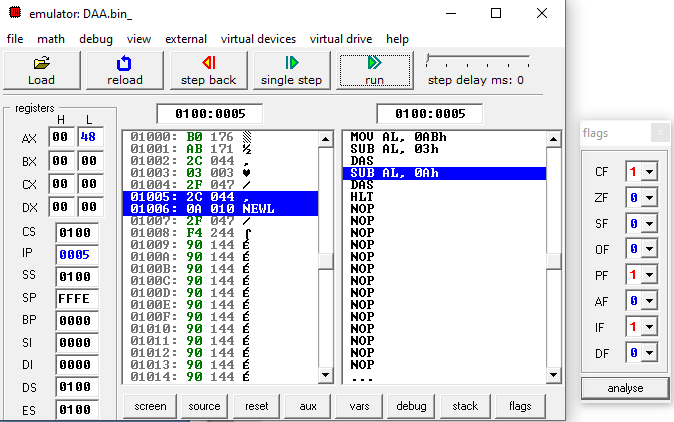
DAS

MOST SIGNIFICANT 4 bits OF AL>9,

So

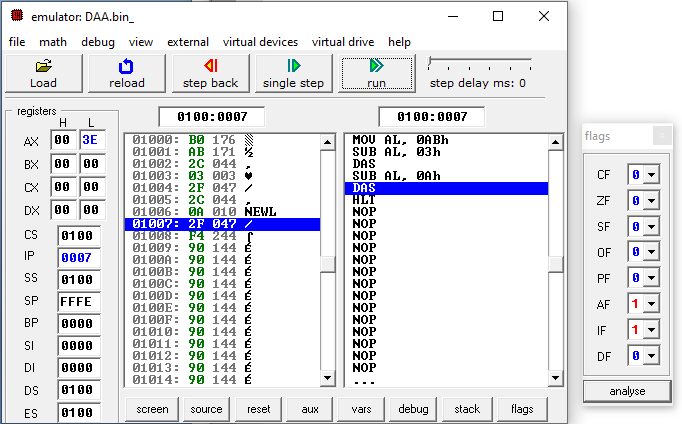
AL <- AL-60H

AL<-48; CF<-1



AL <- AL-0AH

AL <- 3EH



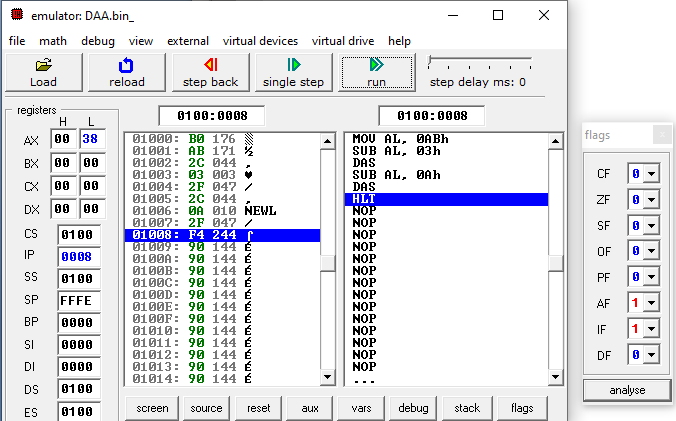
DAS

LEAST SIGNIFICANT 4 bits OF AL>9

SO

AL <- AL-06H

AL<- 38H;AF<-1



6. Write an assembly language program to find out the count of positive numbers and negative numbers

from a series of signed numbers in 8086.

**CODE**:

MOV CL, 0AH

MOV BL, 00H

MOV DL,00H

LEA SI,[1000H]

L1:MOV AL,[SI]

SHL AL,01

JNC L2

INC DL

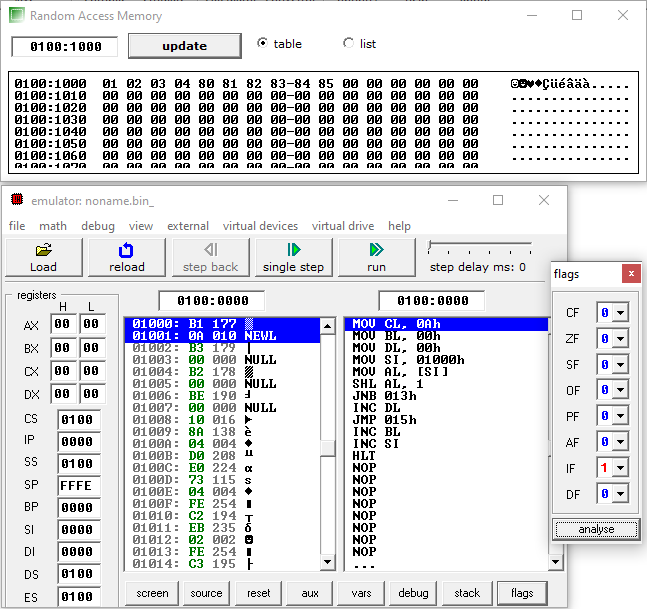
JMP L3

L2: INC BL

L3: INC SI

HLT

**OUTPUT**:



**INPUT**:

1000:=01

1001:=02

1002:=03

1003:=04

1004:=80

1005:=81

1006:=82

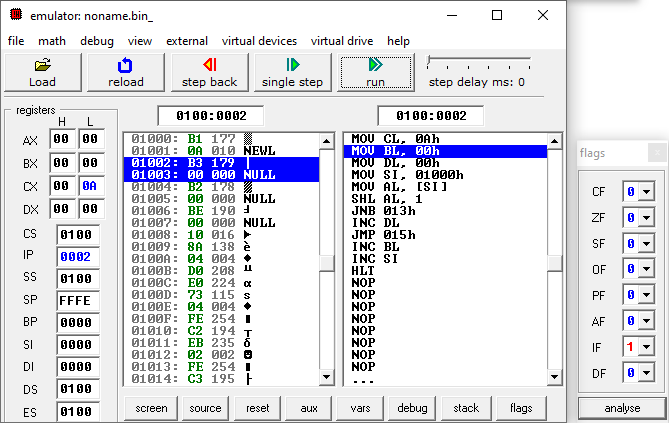
1007:=83

1008:=84

1009:=85

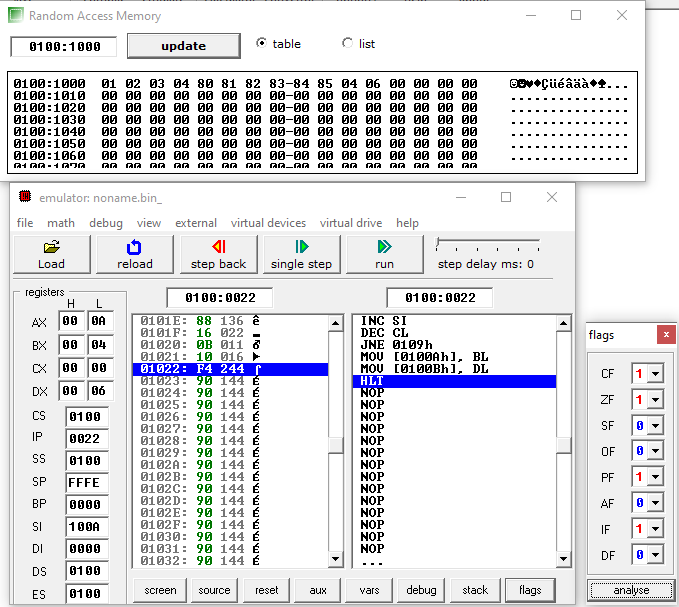
**OUTPUT**:

CX:=0AH



BL:=NUMBER OF POSITIVE NUMBERS=4

DL:=NUMBER OF NEGATIVE NUMBERS=6



7. Write an assembly language program to convert to find out the largest number from a given unordered

array of 8-bit numbers, stored in the locations starting from a known address in 8086.

**CODE**:

MOV CL, 0AH

LEA SI,[1000H]

MOV AL,[SI]

L1:INC SI

MOV BL,[SI]

CMP AL,BL

JC L2

JMP L3

L2: MOV AL,BL

L3: DEC CL

JNZ L1

MOV [100AH],AL

HLT

**INPUT**:

1000:=00H

1001:=01H

1002:=40H

1003:=50H

1004:=33H

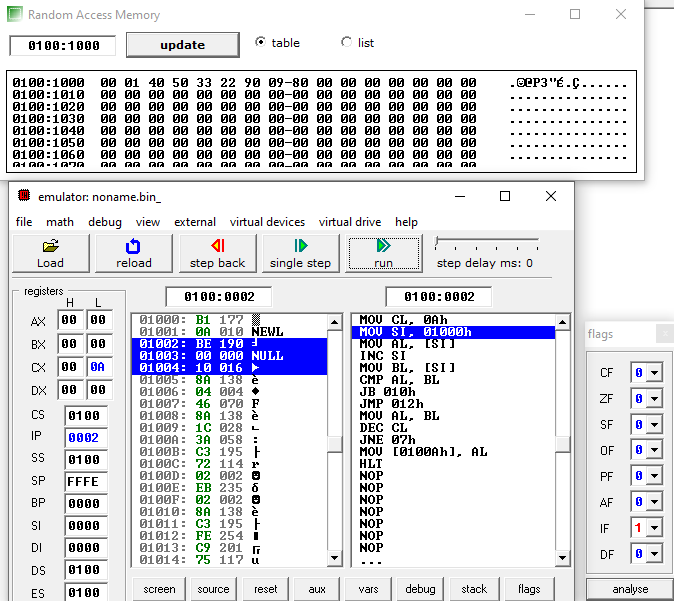
1005:=22H

1006:=90H

1007:=09H

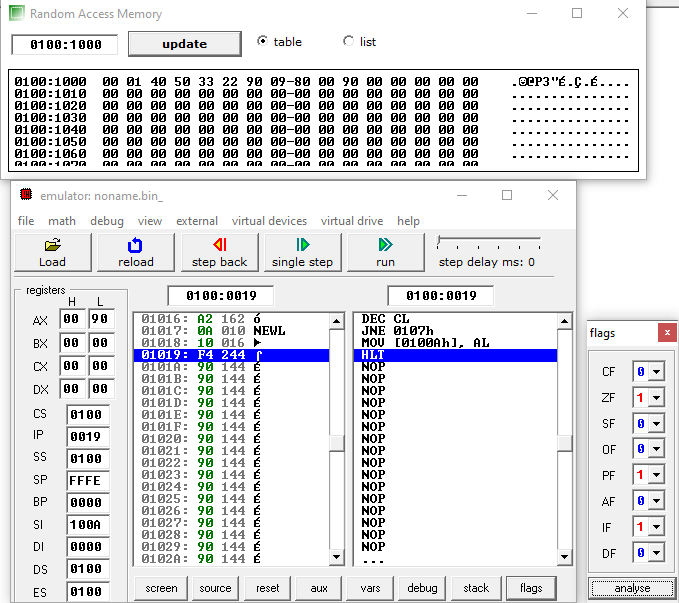
1008:=80H

1009:=00H



**OUTPUT**:

AL:=90 (LARGEST NUMBER IN THAT ARRAY)



8. Write an assembly language program to convert to find out the largest number from a given unordered

array of 16-bit numbers, stored in the locations starting from a known address in 8086.

**CODE**:

MOV BX, 1000H

MOV CL, [BX]

INC BX

MOV AX, [BX]

DEC CL

BACK: INC BX

INC BX

CMP AX,[BX]

JNC Next

MOV AX, [BX]

Next: DEC CL

JNZ Back

MOV [1020H],AX

HLT

**INPUT**:

1000:=05H; CX:=05H

1001:=20H

1002:=30H; 3020H

1003:=40H

1004:=50H; 5040H

1005:=60H

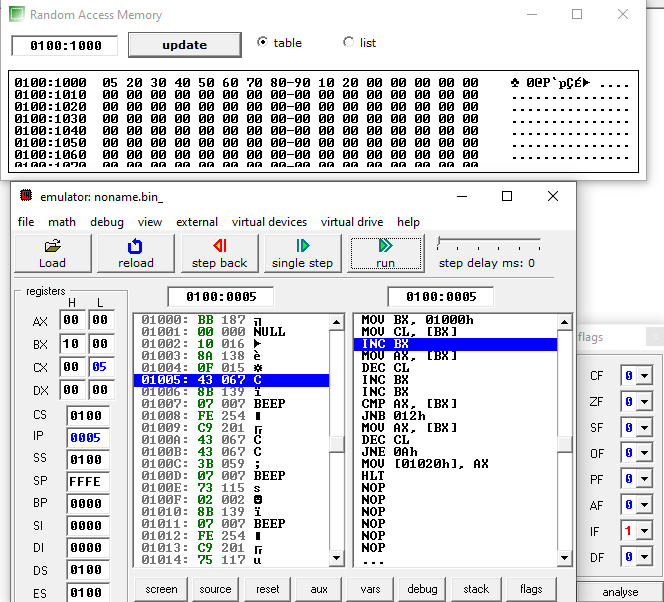
1006:=70H; 7060H

1007:=80H

1008:=90H; 9080H

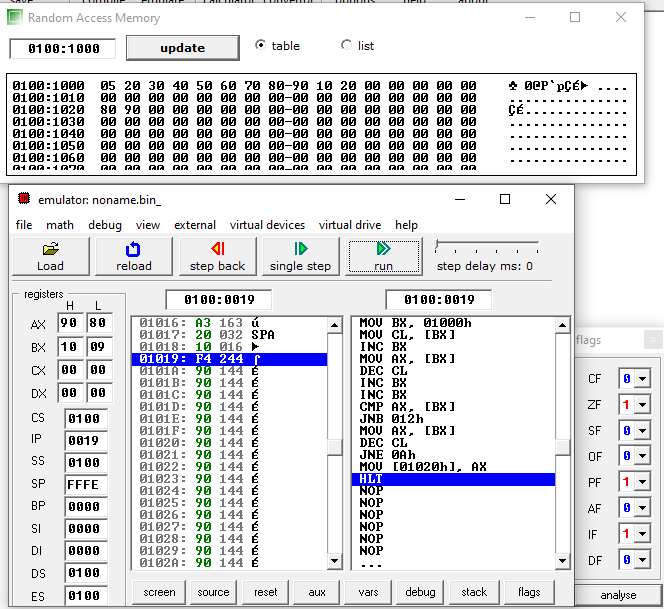
1009:=10H

100A:=20H; 2010H



**OUTPUT**:

AX:=9080H (LARGEST 16-BIT NUMBER)



9. Write an assembly language program to print Fibonacci series in 8086.

**CODE**:

MOV SI,3000H

MOV CX,0AH

XOR AL,AL

MOV [SI],0AH

INC SI

MOV [SI],00H

ADD AL,01H

INC SI

MOV [SI],AL

INC SI

MOV [SI],AL

Back: ADD AL,[SI]

INC SI

MOV [SI],AL

DEC SI

MOV AL,[SI]

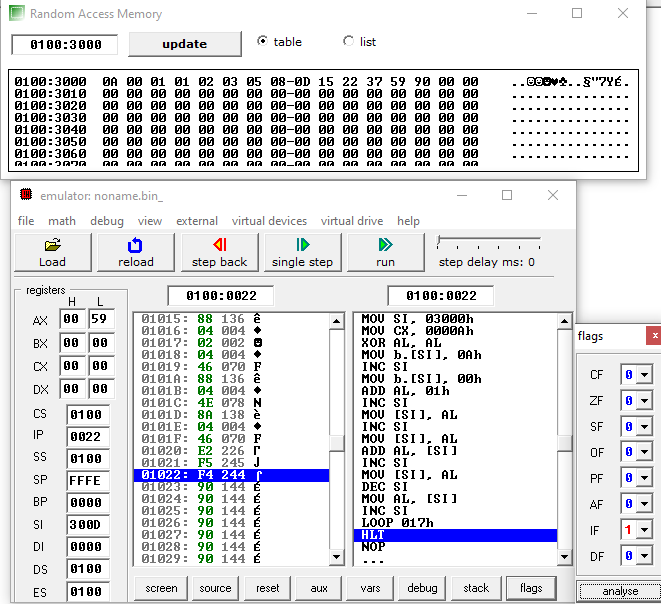
INC SI

LOOP Back

HLT

**OUTPUT**:

3000:=0AH (NUMBER OF FIBONACCI NUMBERS)



10. Write an assembly language program to perform the division 15/6 using the ASCII **CODE**s. Store the

ASCII **CODE**s of the result in register DX.

**CODE**:

MOV AX, '15'

MOV BX, '6'

SUB AX, 3030H

SUB BX, 30H

AAD

DIV BL

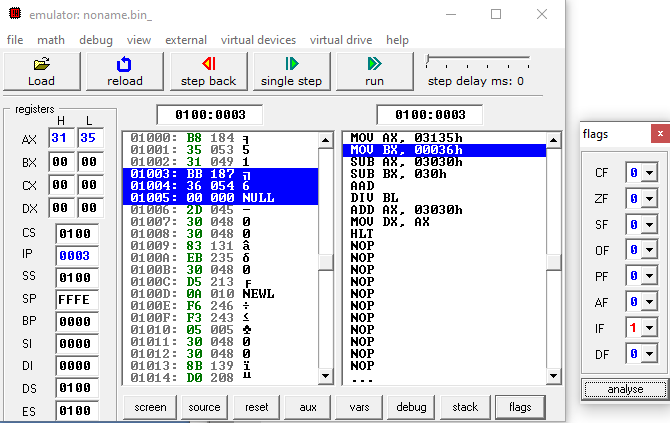
ADD AX, 3030H

MOV DX, AX

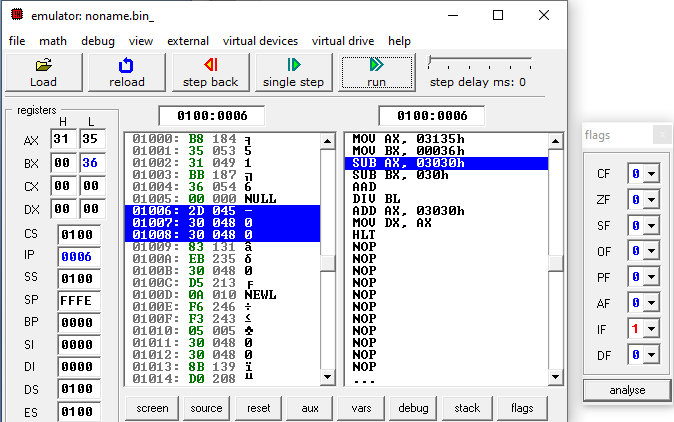
HLT

**OUTPUT**:

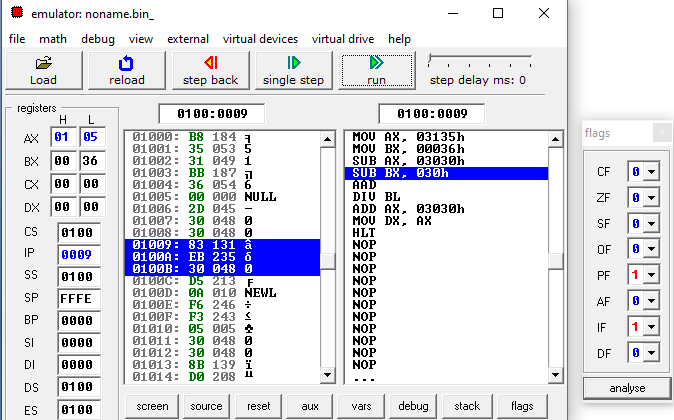
AX <- ‘15’ => AH <- 31, AL<- 35 (ASCII REPRESENTATION)



BX <- ‘6’ => BX <- 00, AL<- 36 (ASCII REPRESENTATION)

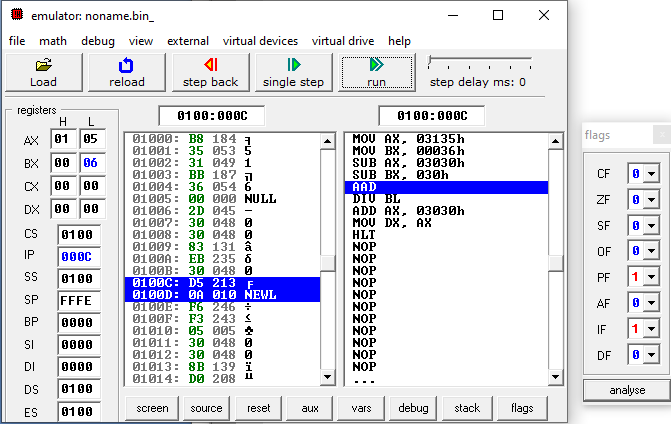


SUB AX, 3030H (TO MAKE AX 0105H)



SIMILARLY, FOR BX

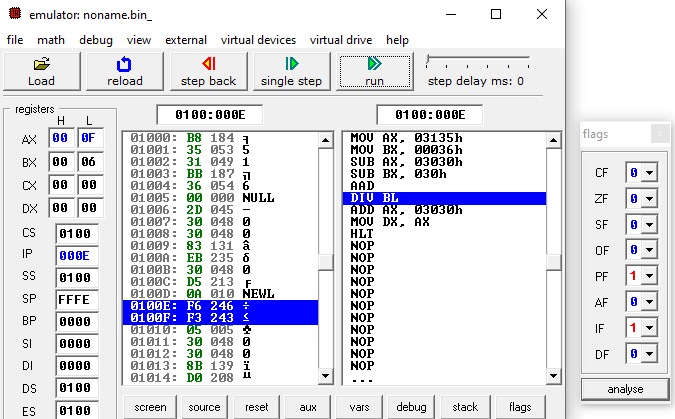
SUB BX, 30H



AAD

AL <- (AH\*10)+AL (DECIMAL) => AL <- (01\*10)+05=10+5=15=0FH

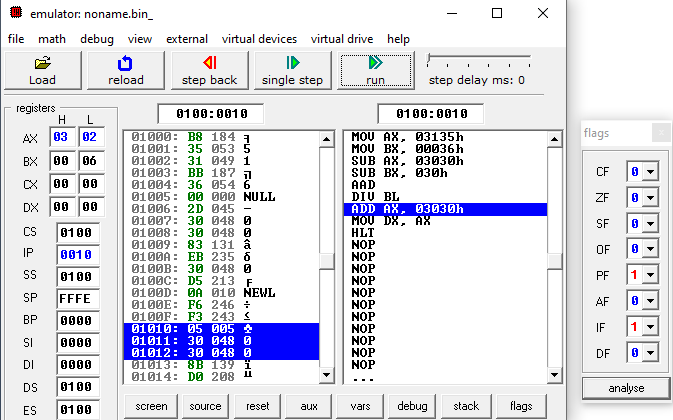
AH <- 00H



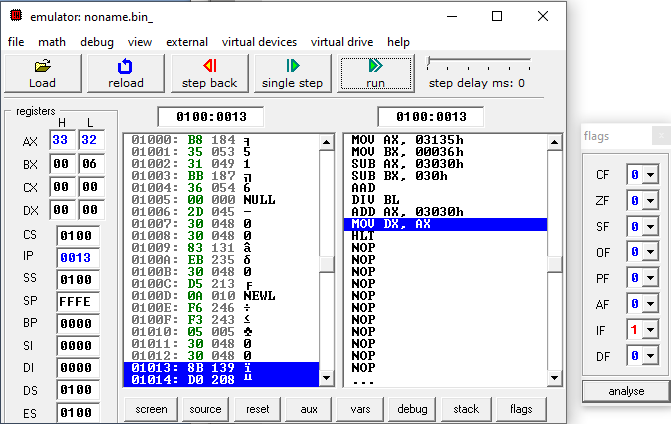
DIV BL

15/6 GIVES 2 QUOTIENT AND 3 REMAINDER IN DECIMAL.

AL <- 02, AH <- 03



ADD AX, 3030H (TO MAKE IT BACK TO ASCII)



STORE AX IN DX.

