Microprocessor-Based Systems Design (UCS617)

Lab Assignment-2 (8086)

Submitted by:

Aditya Sayal 102103559

Manjot Singh Selopal 102103560

Mannat Sadana 102103561

Samarjeet Singh 102103562

Arshiya Kishore 102103565

Submitted to

Dr. Shivani Sharma

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY,

(A DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB

INDIA

Jan-May 2024

INDEX

Experiment No.-16

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

A) Fix-time code

```
MOV BX,1234H

MOV CX,5678H

ADD BX,CX

 HLT          ; halt!
```

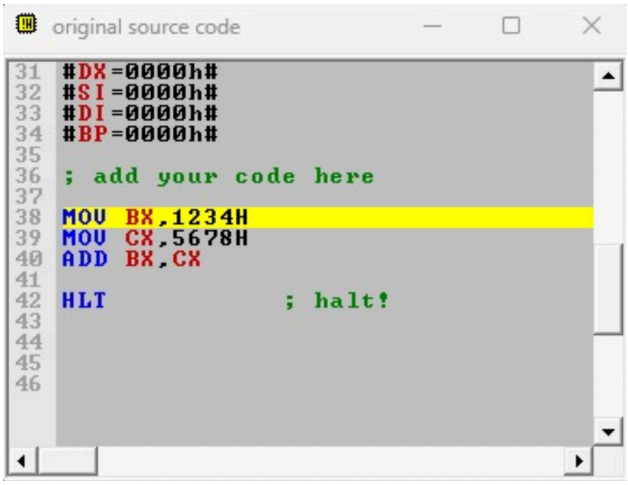


Figure 16.1

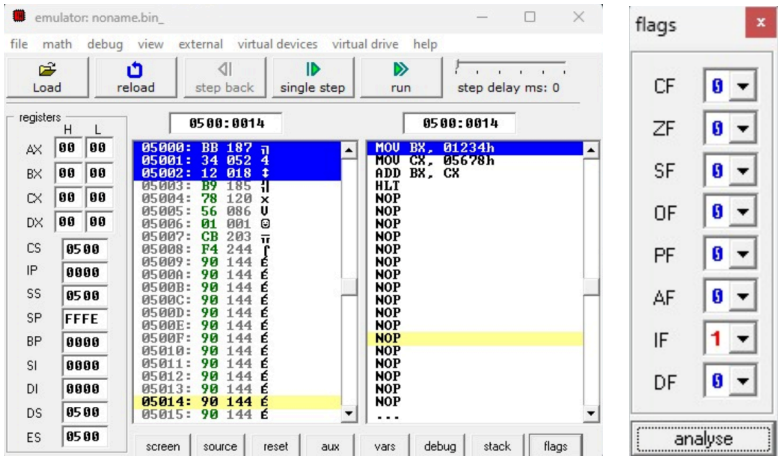


Figure 16.2

B) Run-time code

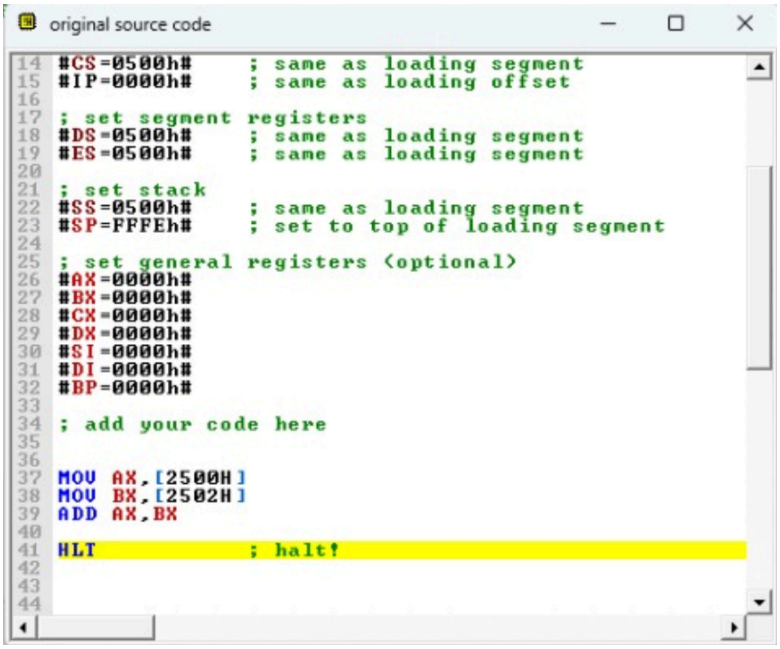


Figure 16.3

Experiment No.-17

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

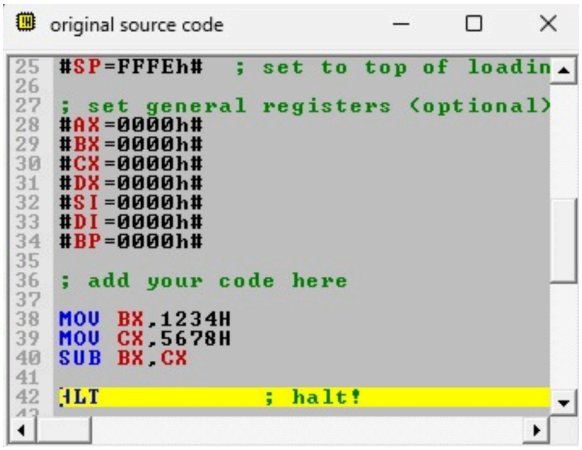
A)Fix-time code

```
MOV BX,1234H

MOV CX,5678H

SUB  BX,CX

HLT          ; halt!
```



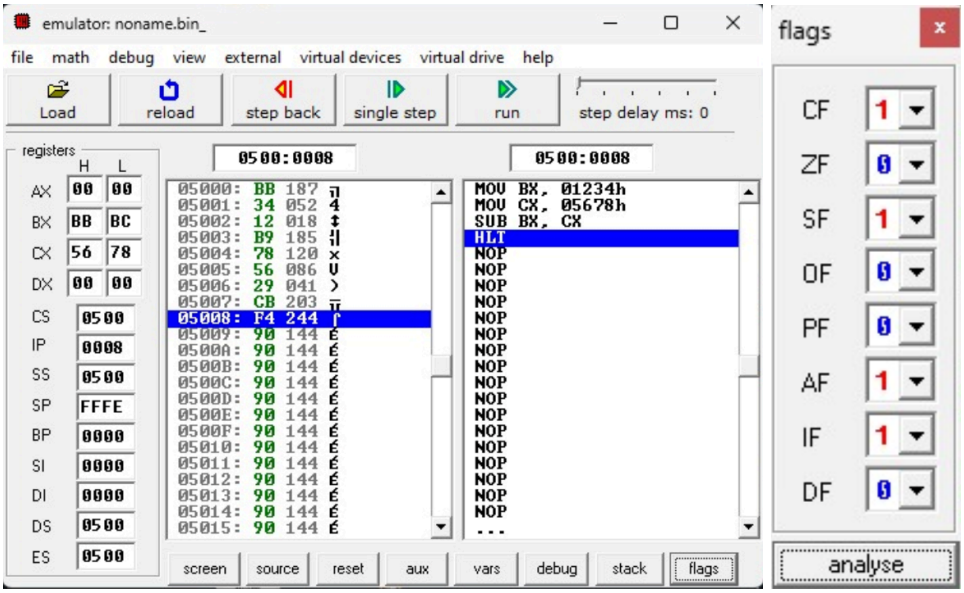

emulator: noname.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
H L
AX 00 00
BX BB BC
CX 56 78
DX 00 00
CS 0500
IP 0008
SS 0500
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0500
ES 0500
0500:0008
0500:0008
05000: BB 187
05001: 34 052 4
05002: 12 018
05003: B9 185
05004: 78 120 x
05005: 56 086 V
05006: 29 041)
05007: CB 203
05008: F4 244
05009: 90 144 É
0500A: 90 144 É
0500B: 90 144 É
0500C: 90 144 É
0500D: 90 144 É
0500E: 90 144 É
0500F: 90 144 É
05010: 90 144 É
05011: 90 144 É
05012: 90 144 É
05013: 90 144 É
05014: 90 144 É
05015: 90 144 É
MOV BX, 01234h
MOV CX, 05678h
SUB BX, CX
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags
flags
CF 1
ZF 0
SF 1
OF 0
PF 0
AF 1
IF 1
DF 0
analyse

Figure 17.1

Experiment No.-18

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

Static

MOV AX, 1234H

MOV BX, 5678H

MUL BX

HLT

OUTPUT: AX-0060, DX-0626

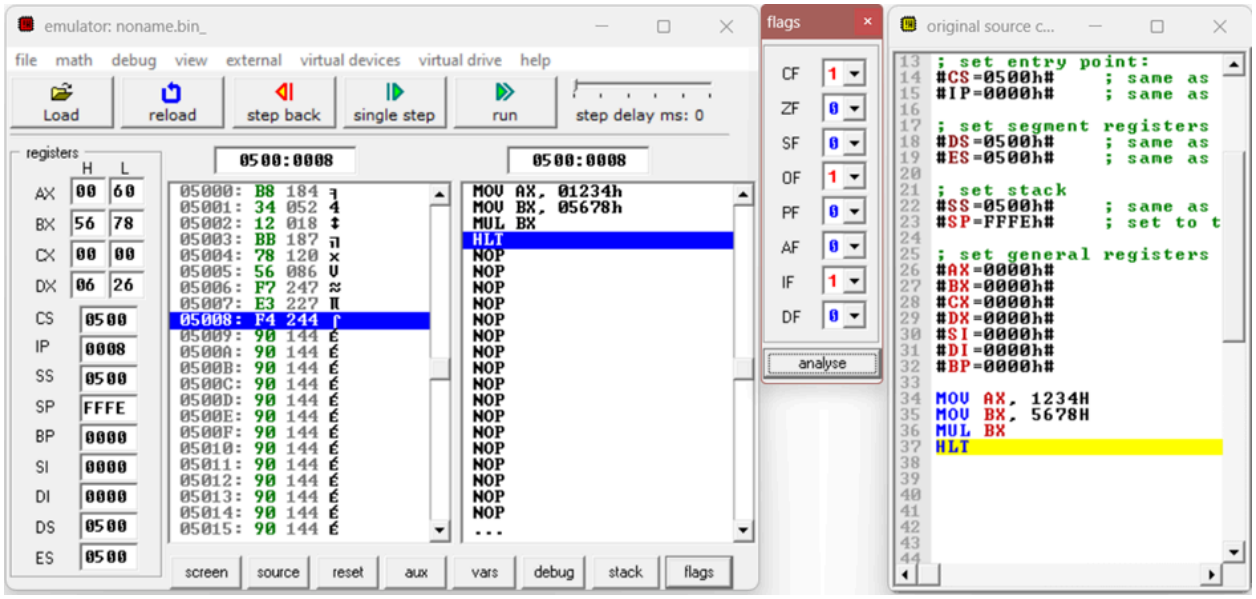


Figure 18.1

Dynamic

MOV AX, [1234H]

MOV CX, [1236H]

MUL CX

MOV [1238H], AX

MOV [123AH], DX

HLT

INPUT: [1234]-11, [1235]-22, [1236]-33 [1237]-44

OUTPUT: [1238]-63, [1239]-4D, [123A]-13, [123B]-09

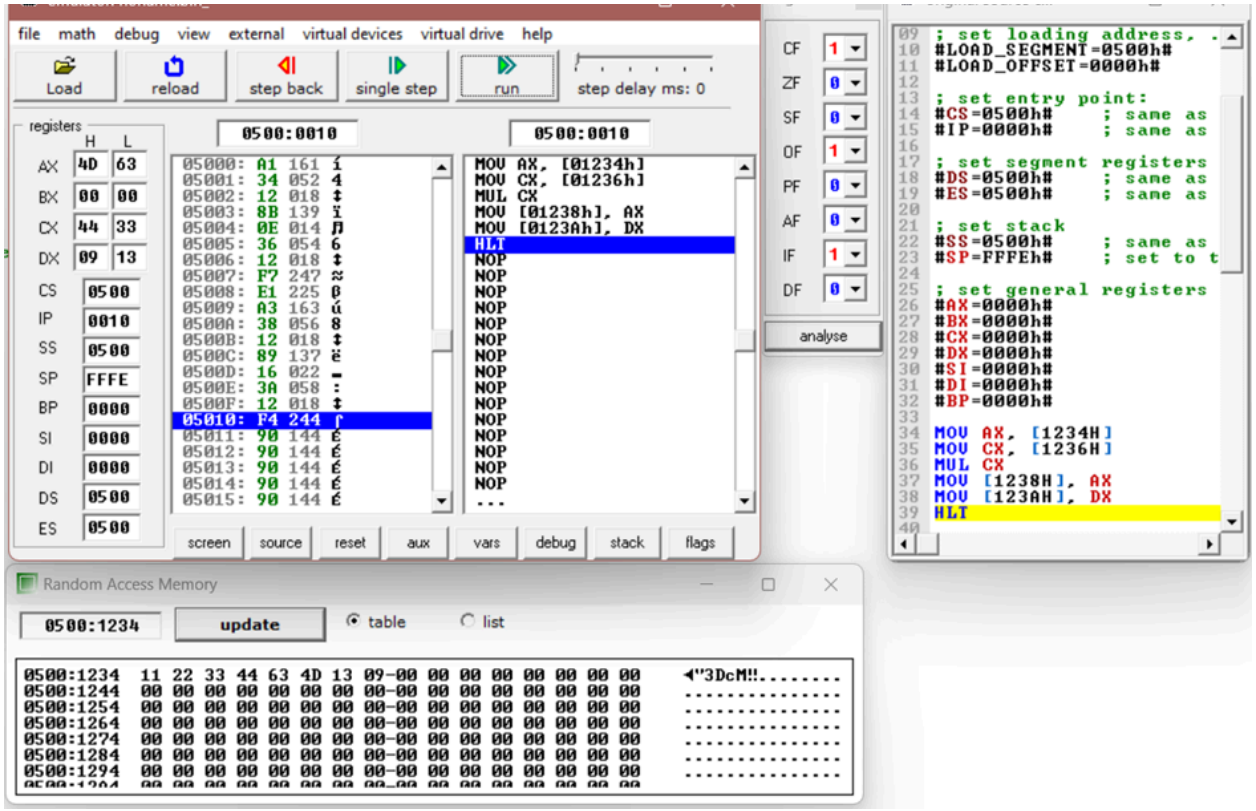


Figure 18.2

Experiment No.-19

Aim: Write an assembly language program to divide two 16-bit numbers in 8086

Static

MOV AX, 5678H

MOV BX, 1234H

DIV BX

HLT

OUTPUT: AX-0004, DX-0DA8

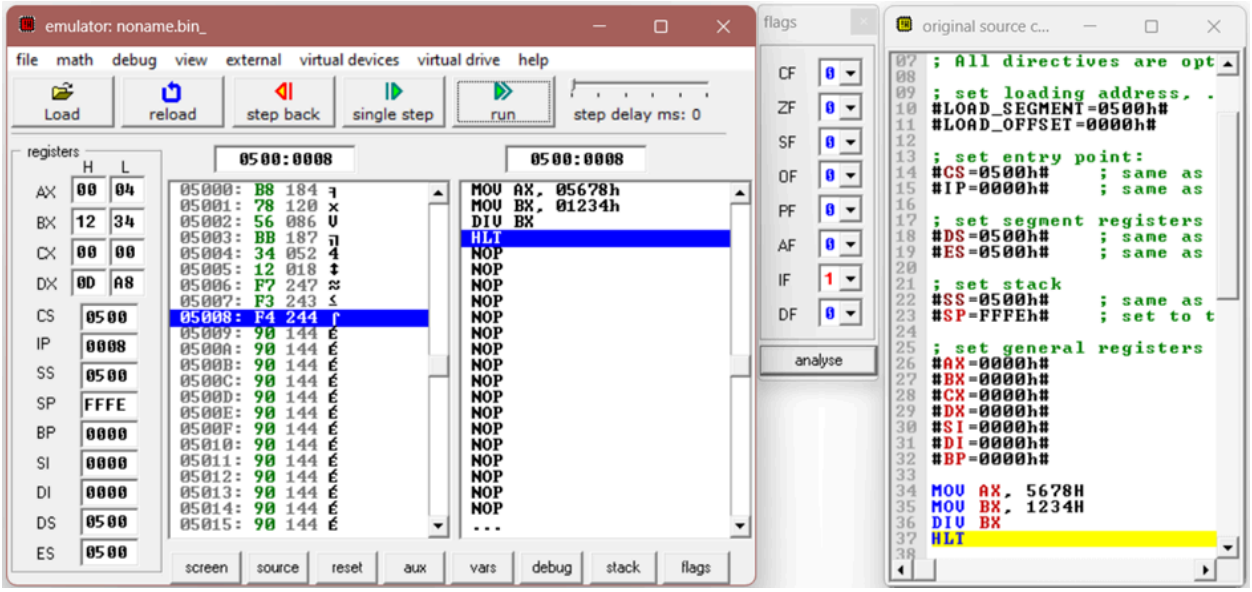


Figure 19.1

Dynamic

MOV AX, [1234H]

MOV BX, [1236H]

DIV BX

MOV [1238H], AX

MOV [123AH], DX

HLT

INPUT: [1234]-11, [1235]-22, [1236]-33 [1237]-44

OUTPUT: [1238]-00, [1239]-00, [123A]-11, [123B]-22

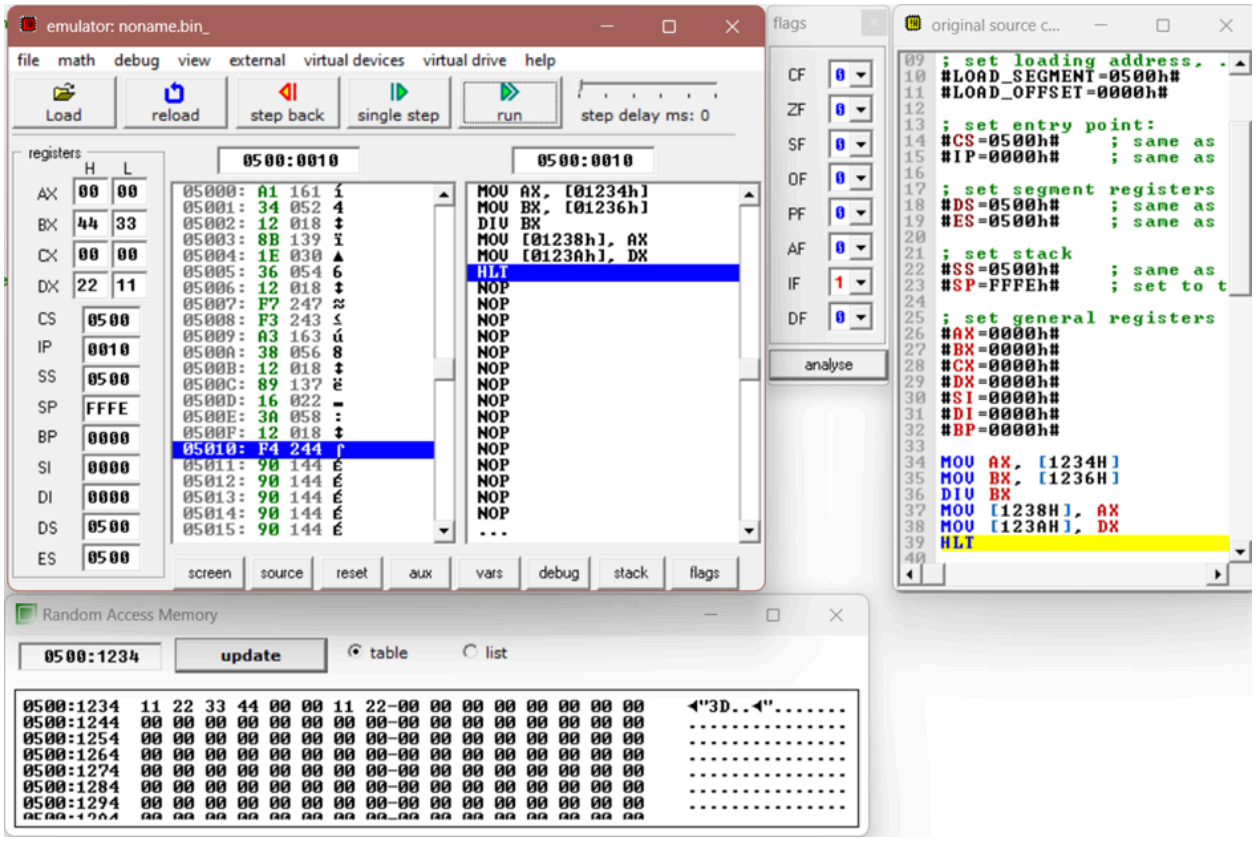


Figure 19.2

Experiment No.-20

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

1. AAA

MOV AL, ‘5’

MOV BL, ‘6’

ADD BL, AL

AAA

HLT

OUTPUT: AL-05

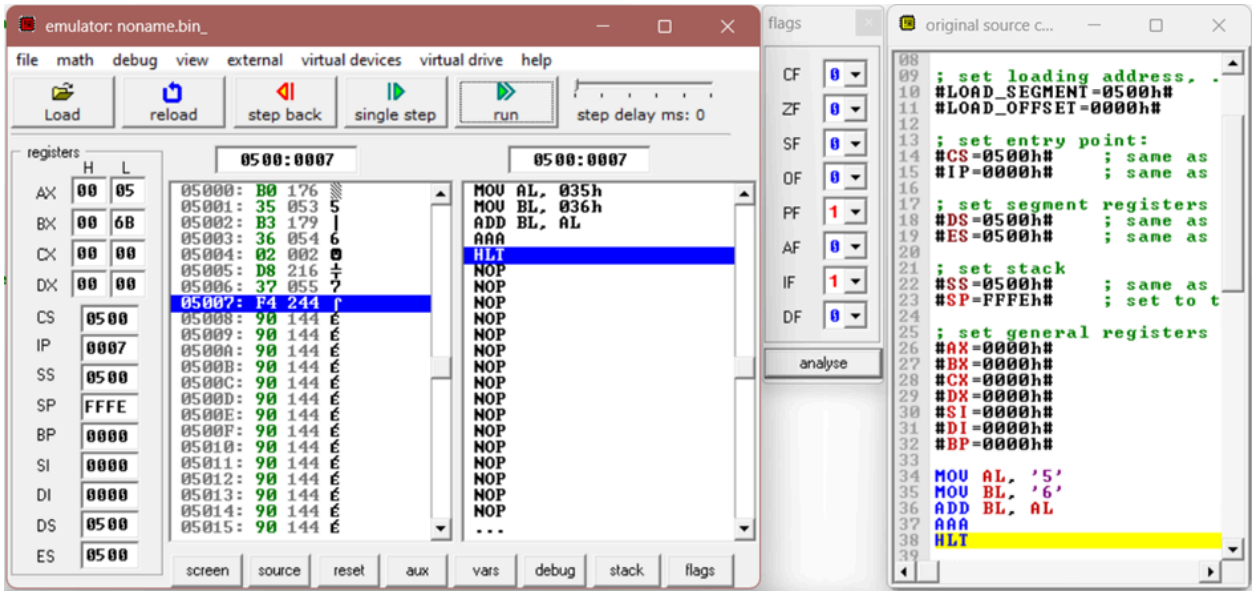


Figure 20.1

2. AAS

MOV AL, ‘3’

SUB AL, ‘9’

AAS

HLT

OUTPUT: AX-FF04

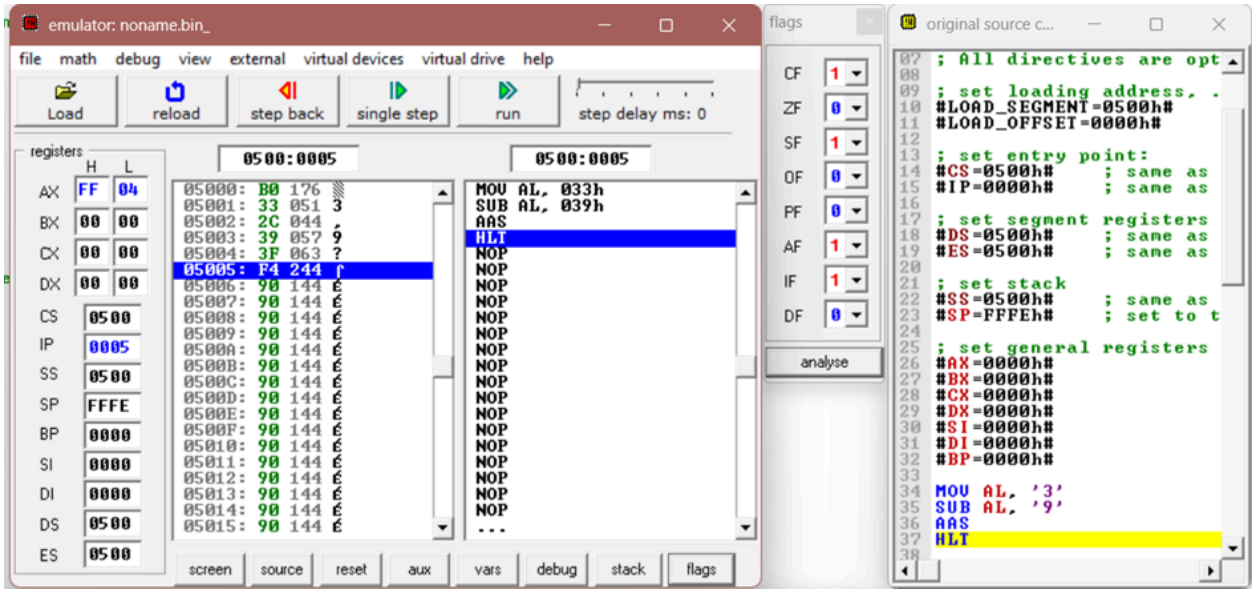


Figure 20.2

3. AAM

MOV AL, ‘5’

MOV BL, ‘7’

MUL BL

AAM

HLT

OUTPUT: AX-0909

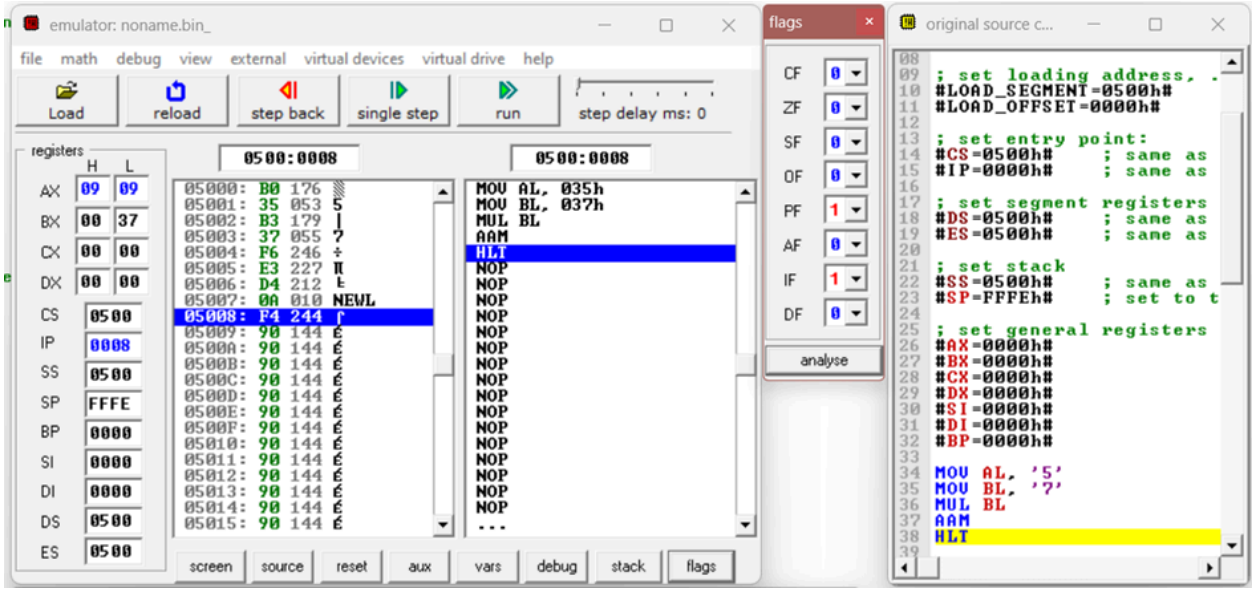


Figure 20.3

4. AAD

MOV AX, 0205H

AAD

HLT

OUTPUT: AL-09

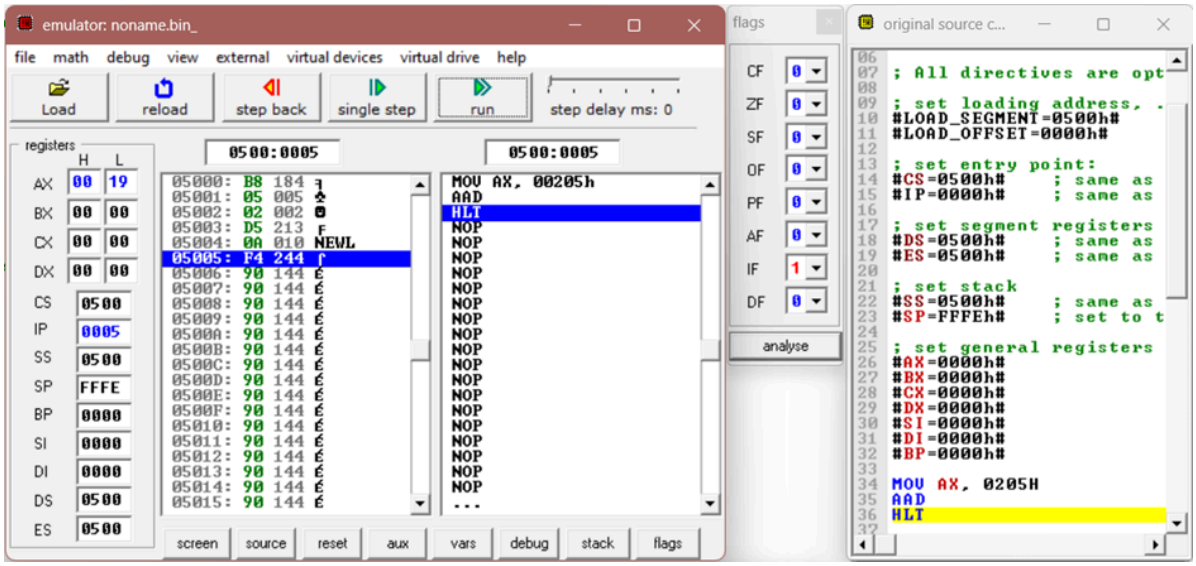


Figure 20.4

5. DAA

MOV AL, 71H

ADD AL, 43H

DAA

HLT

OUTPUT: AL-14

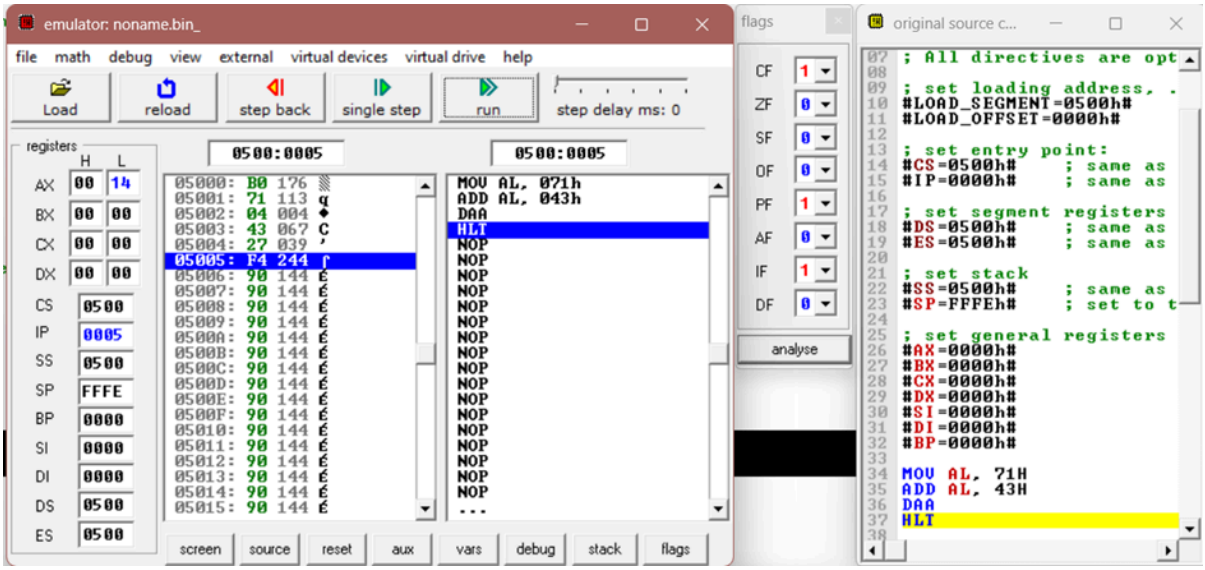


Figure 20.5

6. DAS

MOV AL, 71H

SUB AL, 43H

DAS

HLT

OUTPUT: AL-28

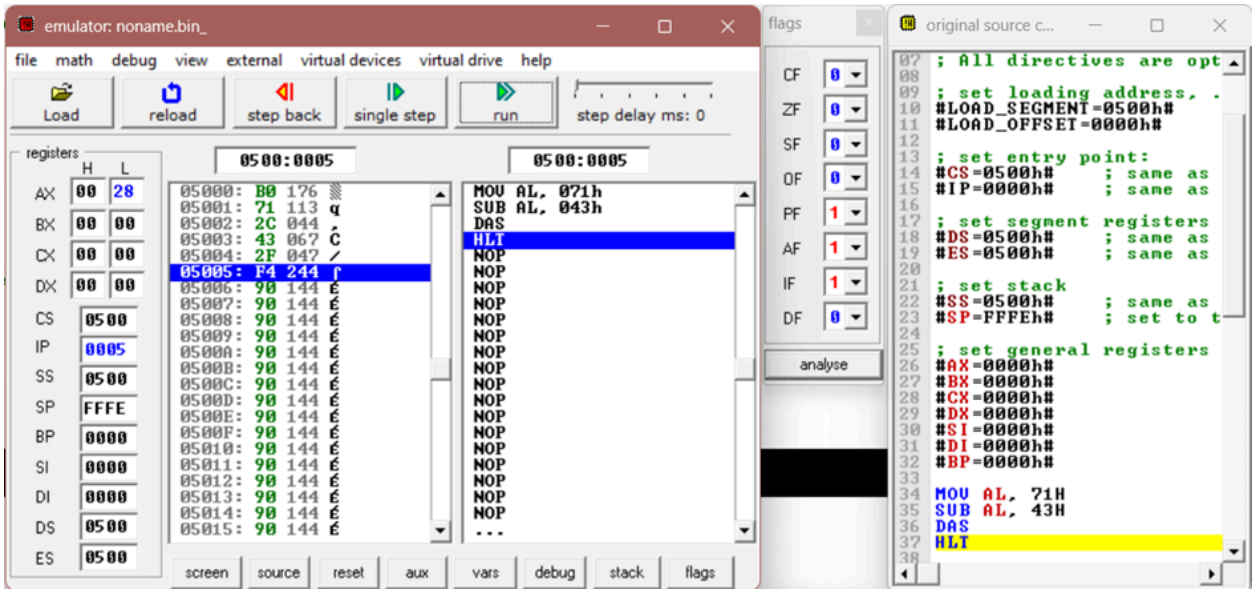

emulator: noname.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
H L
AX 00 28
BX 00 00
CX 00 00
DX 00 00
CS 0500
IP 0005
SS 0500
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0500
ES 0500
0500:0005
0500:0005
05000: B0 176
05001: 71 113 q
05002: 2C 044 ,
05003: 43 067 C
05004: 2F 047 /
05005: F4 244
05006: 90 144 É
05007: 90 144 É
05008: 90 144 É
05009: 90 144 É
0500A: 90 144 É
0500B: 90 144 É
0500C: 90 144 É
0500D: 90 144 É
0500E: 90 144 É
0500F: 90 144 É
05010: 90 144 É
05011: 90 144 É
05012: 90 144 É
05013: 90 144 É
05014: 90 144 É
05015: 90 144 É
MOV AL, 071h
SUB AL, 043h
DAS
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags
flags
CF 0
ZF 0
SF 0
OF 0
PF 1
AF 1
IF 1
DF 0
analyse
original source c...
; All directives are opt
; set loading address, .
#LOAD_SEGMENT=0500h#
#LOAD_OFFSET=0000h#
; set entry point:
#CS=0500h# ; same as
#IP=0000h# ; same as
; set segment registers
#DS=0500h# ; same as
#ES=0500h# ; same as
; set stack
#SS=0500h# ; same as
#SP=FFFEh# ; set to t
; set general registers
#AX=0000h#
#BX=0000h#
#CX=0000h#
#DX=0000h#
#SI=0000h#
#DI=0000h#
#BP=0000h#
MOV AL, 71H
SUB AL, 43H
DAS
HLT

Figure 20.6

Experiment No.-21

Aim: Write an assembly language program to find the count of positive 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

DEC CL

JNZ L1

MOV [100AH], BL

MOV [100BH], DL

HLT ; halt!
```

INPUT:[1000H]-01,[1001H]-02,[1002H]-03,[1003H]-04,[1004H]-80,[1005H]-81 , [1006H]-82, [1007H]-83,[1008H]-84,[1009H]-85

OUTPUT: [100AH]-04,[1020H]-06

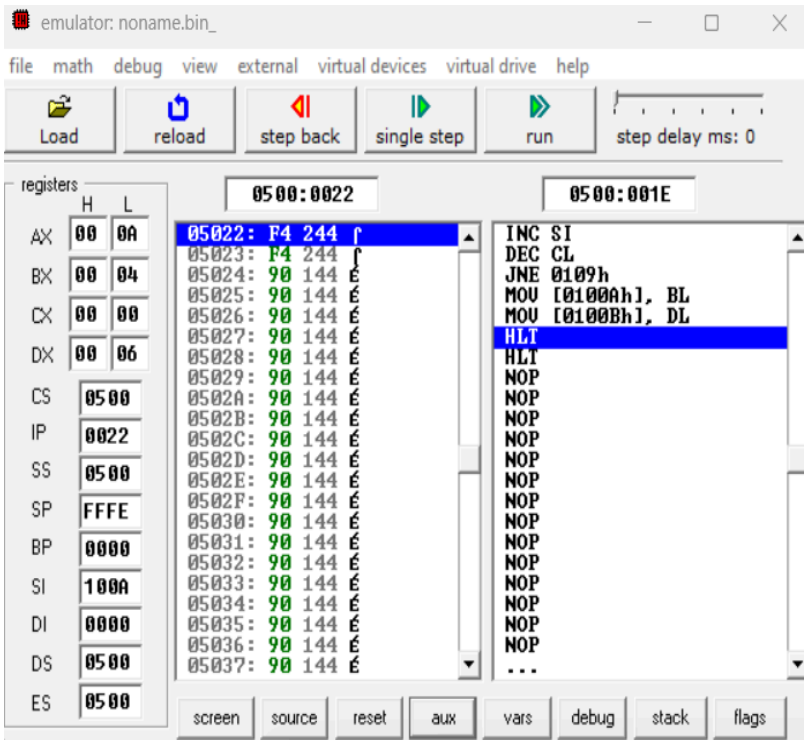


Figure 20.1

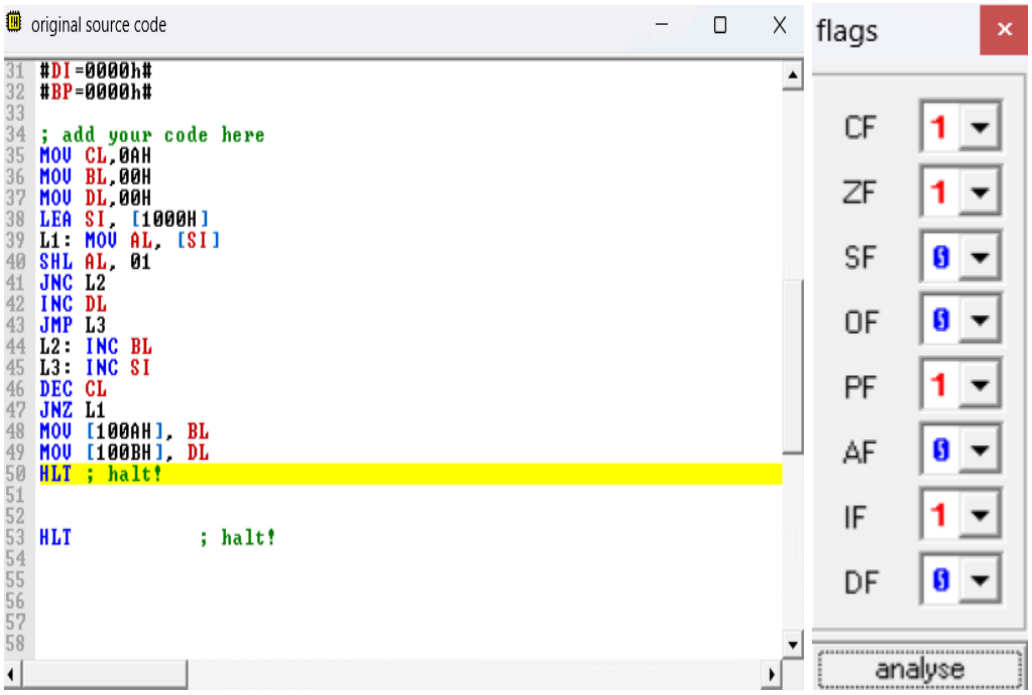


Figure 20.2

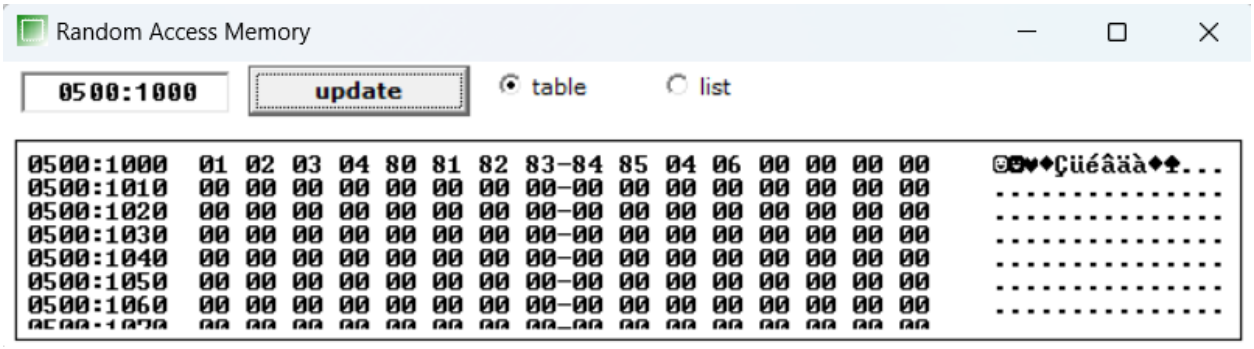


Figure 20.3

Experiment No.-22

Aim: Write an assembly language program to find the largest number from a list of numbers (unsorted) in 8086.

For 8-bit number :

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
HLT
```

INPUT: [1000H]-11, [1001H]-22, [1003H]-33, [1004H]-44, [1005H]-55, [1006H]-66, [1007H]-77, [1008H]-88, [1009H]-99, [100AH]-12
OUTPUT: AL-99

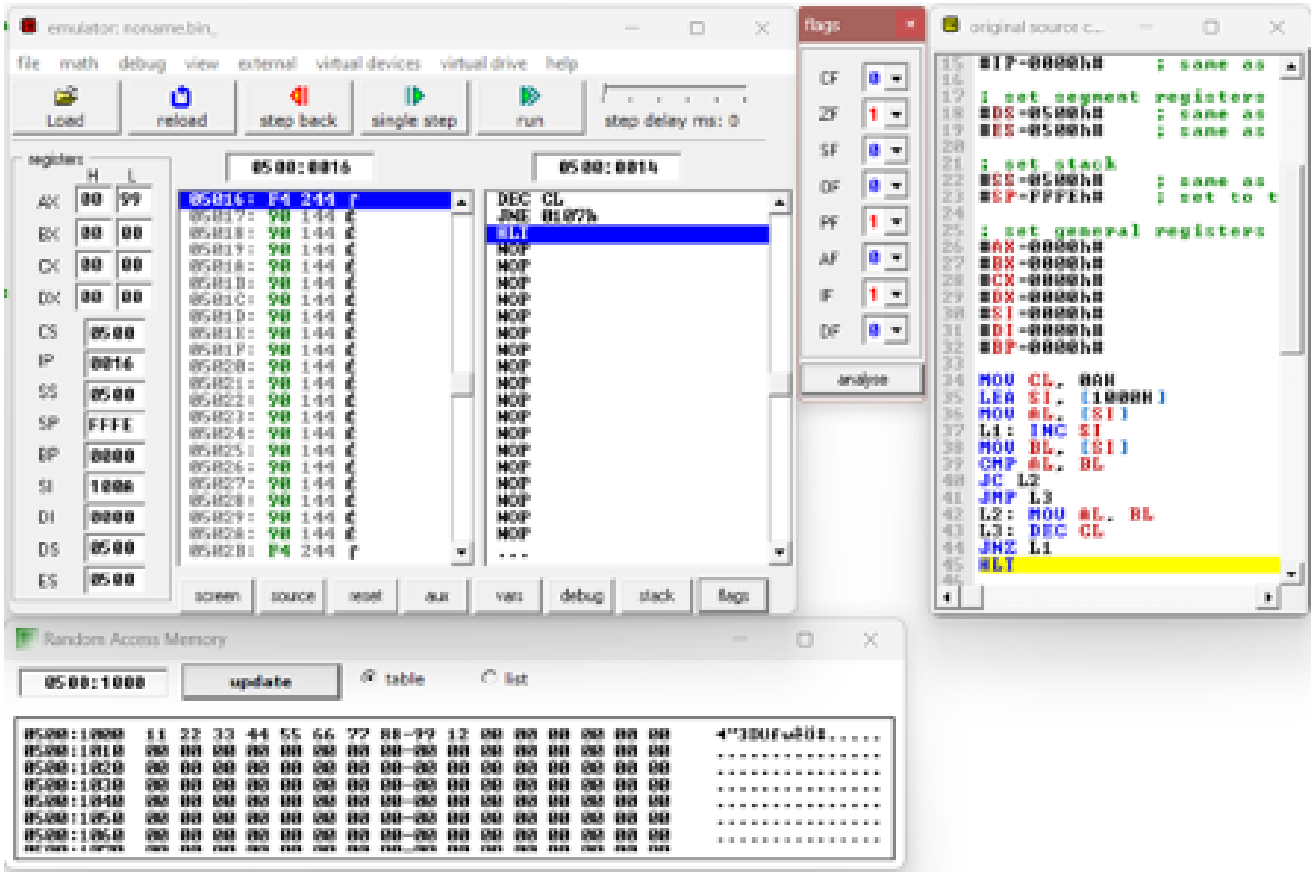


Figure: 22.1

For 16-bit Number :

Code:-

MOV CL, 05H

LEA SI, [1000H]

MOV AL, [SI]

L1: INC SI

MOV BX, [SI]

CMP AX, BX

JC L2

JMP L3

L2: MOV AX, BX

L3: DEC CL

JNZ L1

HLT

INPUT: [1000H]-11, [1001H]-22, [1003H]-33, [1004H]-44, [1005H]-55, [1006H]-66, [1007H]-77, [1008H]-88, [1009H]-99, [100AH]-12

OUTPUT: AX-9988

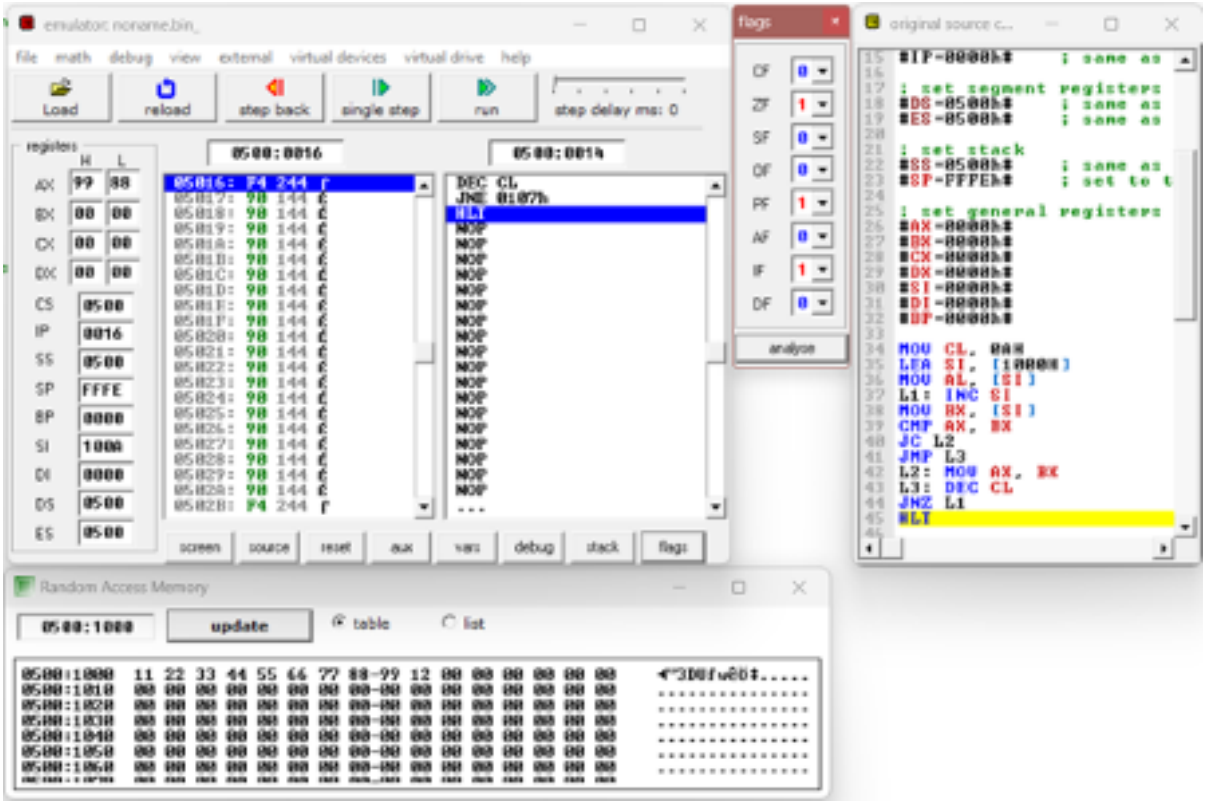


Figure:-22.2

Experiment No.-23

Aim: Write an assembly language program to perform division 15/6 using ASCII Codes in 8086

Code-:

```
MOV AX,'15'

MOV BL,'6'

SUB AX,3030H

SUB BL,30H

AAD

DIV BL

HLT          ; halt!
```

Output-AX-0302 BL-06

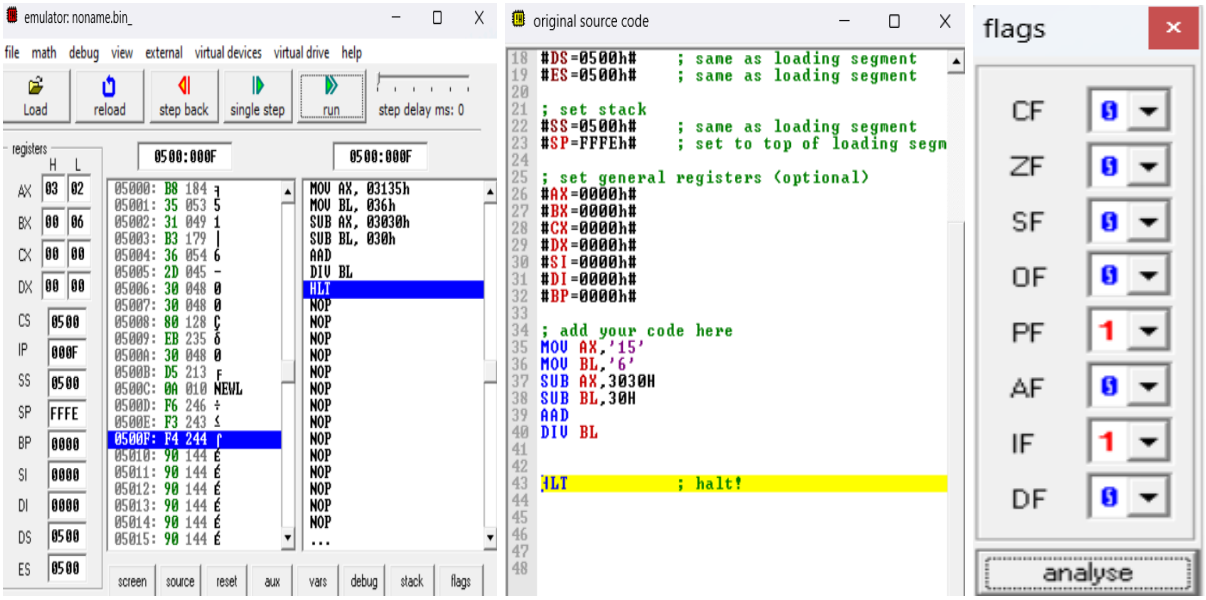


Figure 23.1

Experiment No.-24

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

Code:-

```
MOV CL,0AH

LEA SI, [1001H]

L1: MOV BL,[SI]

MOV DL,[SI-1]

ADD BL,DL

MOV [SI+1],BL

INC SI

DEC CL

JNZ L1
```

Output- BX-59

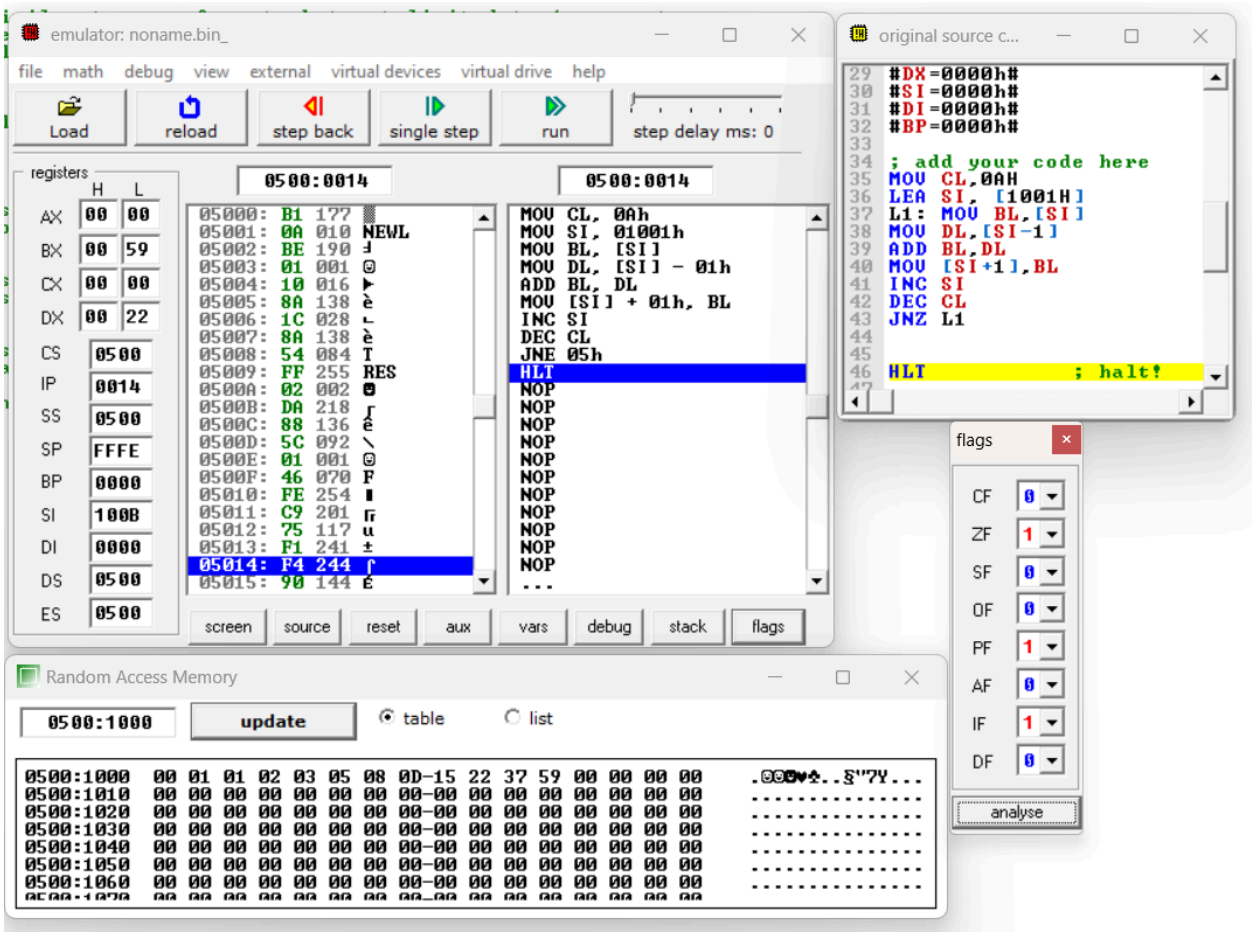

emulator: noname.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
H L
AX 00 00
BX 00 59
CX 00 00
DX 00 22
CS 0500
IP 0014
SS 0500
SP FFFE
BP 0000
SI 100B
DI 0000
DS 0500
ES 0500
0500:0014
0500:0014
05000: B1 177
05001: 0A 010 NEWL
05002: BE 190
05003: 01 001
05004: 10 016
05005: 8A 138
05006: 1C 028
05007: 8A 138
05008: 54 084 T
05009: FF 255 RES
0500A: 02 002
0500B: DA 218
0500C: 88 136
0500D: 5C 092
0500E: 01 001
0500F: 46 070 F
05010: FE 254
05011: C9 201
05012: 75 117 u
05013: F1 241 ±
05014: F4 244
05015: 90 144 É
MOV CL, 0Ah
MOV SI, 01001h
MOV BL, [SI]
MOV DL, [SI] - 01h
ADD BL, DL
MOV [SI] + 01h, BL
INC SI
DEC CL
JNE 05h
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen source reset aux vars debug stack flags
original source c...
29 #DX=0000h#
30 #SI=0000h#
31 #DI=0000h#
32 #BP=0000h#
33
34 ; add your code here
35 MOV CL,0AH
36 LEA SI, [1001H]
37 L1: MOV BL,[SI]
38 MOV DL,[SI-1]
39 ADD BL,DL
40 MOV [SI+1],BL
41 INC SI
42 DEC CL
43 JNZ L1
44
45
46 HLT ; halt!
flags
CF 0
ZF 1
SF 0
OF 0
PF 1
AF 0
IF 1
DF 0
analyse
Random Access Memory
0500:1000
update
table
list
0500:1000 00 01 01 02 03 05 08 0D-15 22 37 59 00 00 00 00
0500:1010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00

Figure 24.1