Table of Contents

Introduction to 8086

ET8086 LCD by Excel Technology

The 8086 Microprocessor is an enhanced version of the 8085 Microprocessor that was designed by Intel in 1976. It is a 16-bit Microprocessor having 20 address lines and16 data lines that provides up to 1MB storage. It consists of a powerful instruction set, which provides operations like multiplication and division easily.

Features of 8086 Microprocessor:

1. It is a 16-bit, N-channel, HMOS (High speed metal oxide semiconductor) microprocessor.
2. Its CMOS (Complementary MOS) version, 80C86 is also available.
3. It consumes less power.
4. The 8086 draws 360 mA on 5V whereas the 80C86 draws only 10 mA.
5. 8086 is manufactured for standard temperature range 32 F to 180 F as well as extended temperature range (40 F to +225 F).
6. Its clock frequencies for its different versions are: 5, 8 and 10 MHz.
7. It was introduced in 1978.

8. It contains an electronic circuitry of 29000 transistors.
9. It is built on a single semiconductor chip and packaged on a 40-Pin IC package.
10. The type of package is DIP (Dual In-Line Package).
11. 8086 uses 20 address lines and 16 data lines.
12. It can directly address up to 2^20=1Mbytes of memory.
13. The 16-bit data word is divided into a low-order byte and a high order byte.
14. The 16 low order address lines are time multiplexed with data, and the 4 high order address lines are time multiplexed with status signals.

Assignment 16

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

Ans.

Program:

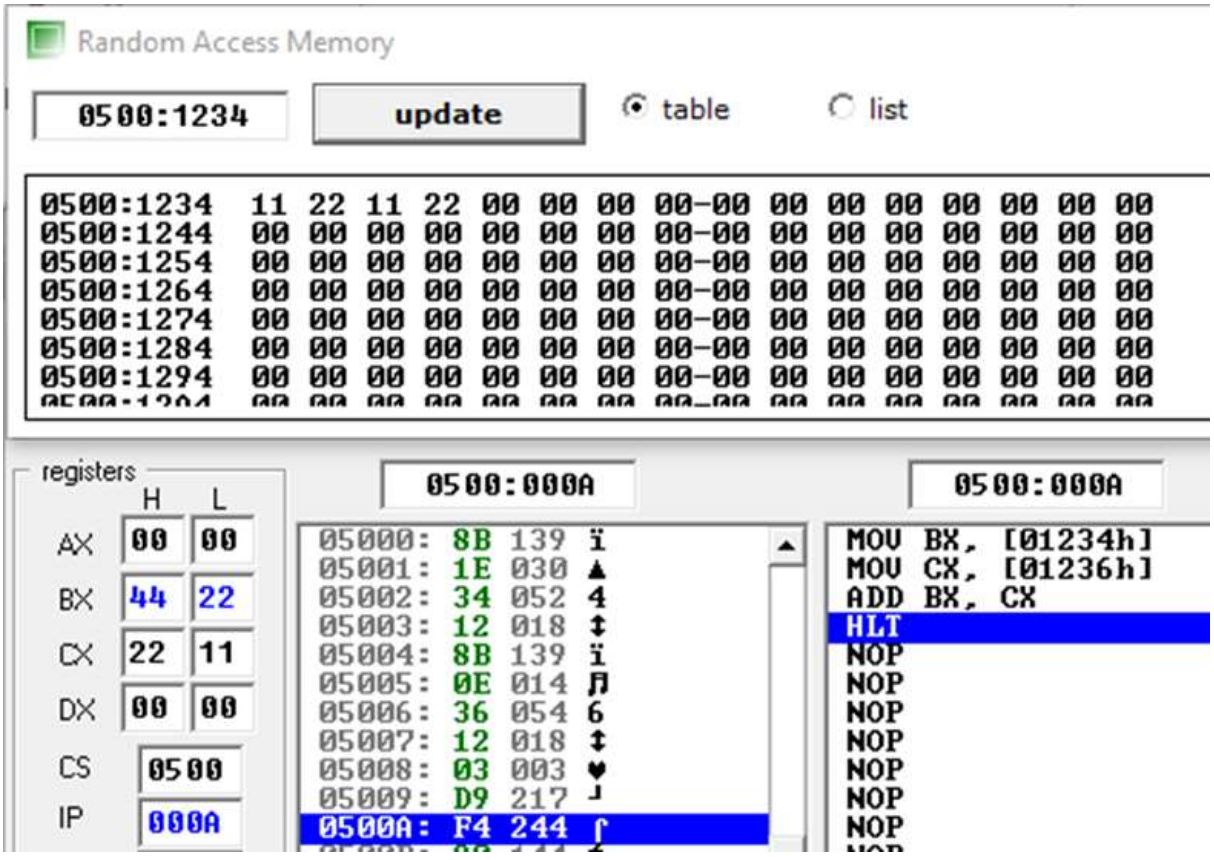
```
MOV BX, [1234H]
MOV CX, [1236H]
ADD BX, CX
HLT
```

INPUT::

[1234] - 11, [1235] - 22, [1236] - 11, [1237] - 22

OUTPUT:

BX: 4422



Assignment 17

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

Ans.

Program:

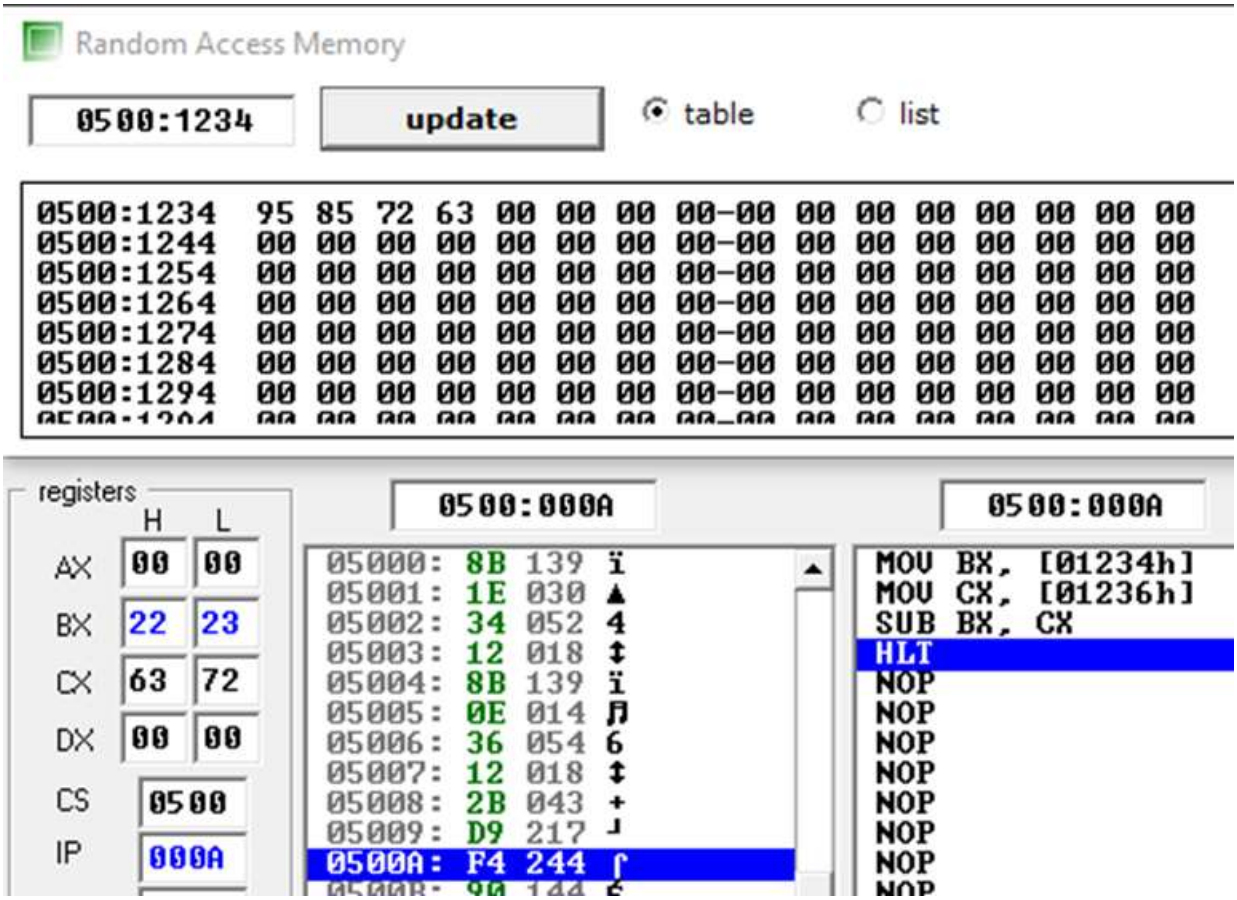
```
MOV BX, [1234H]
MOV CX, [1236H]
SUB BX, CX
HLT
```

INPUT::

[1234] - 95, [1235] - 85, [1236] - 72, [1237] - 63

OUTPUT:

BX: 2223



Assignment 18

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

Ans.

Program:

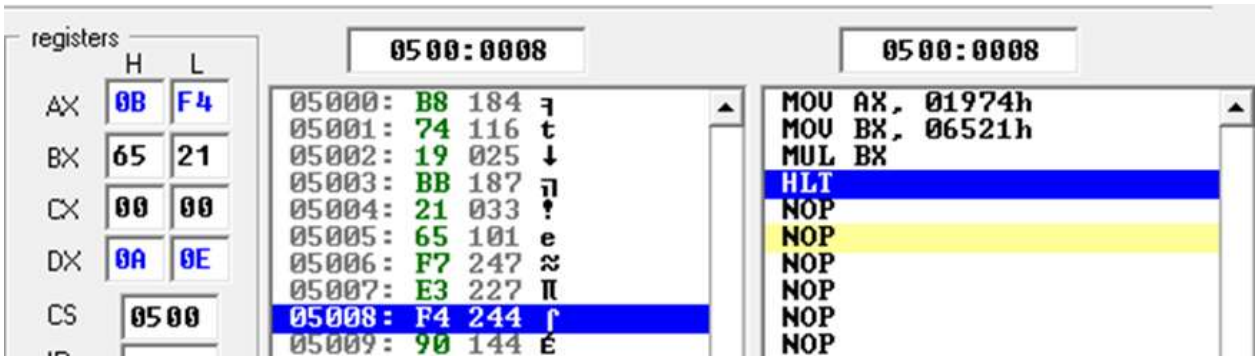
```
MOV AX, 1974H
MOV BX, 6521H
MUL BX
HLT
```

INPUT::

AX: 1974, BX: 6521

OUTPUT:

AX: 0BF4, DX: 0A0E



Assignment 19

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

Ans.

Program:

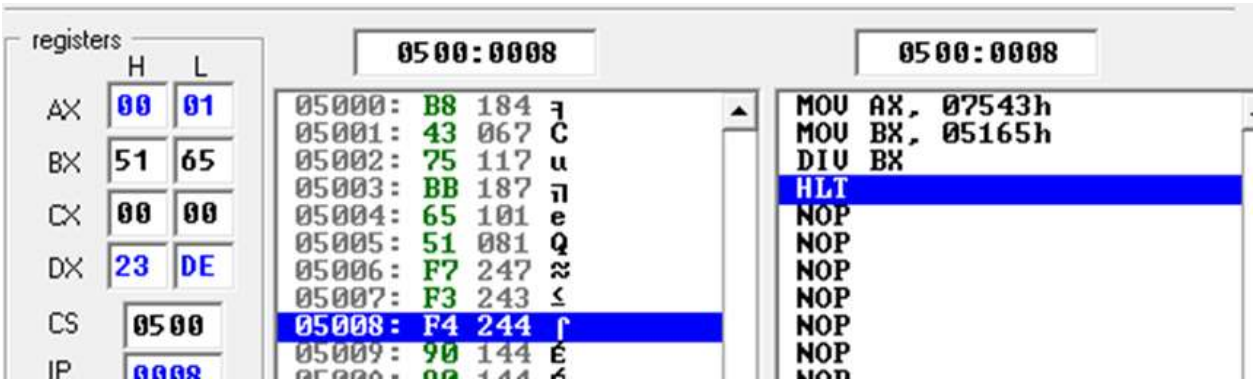
```
MOV AX, 7543H
MOV BX, 5165H
DIV BX
HLT
```

INPUT::

AX: 7543, BX: 5165

OUTPUT:

AX: 0001, DX: 23DE



Assignment 20

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

Ans.

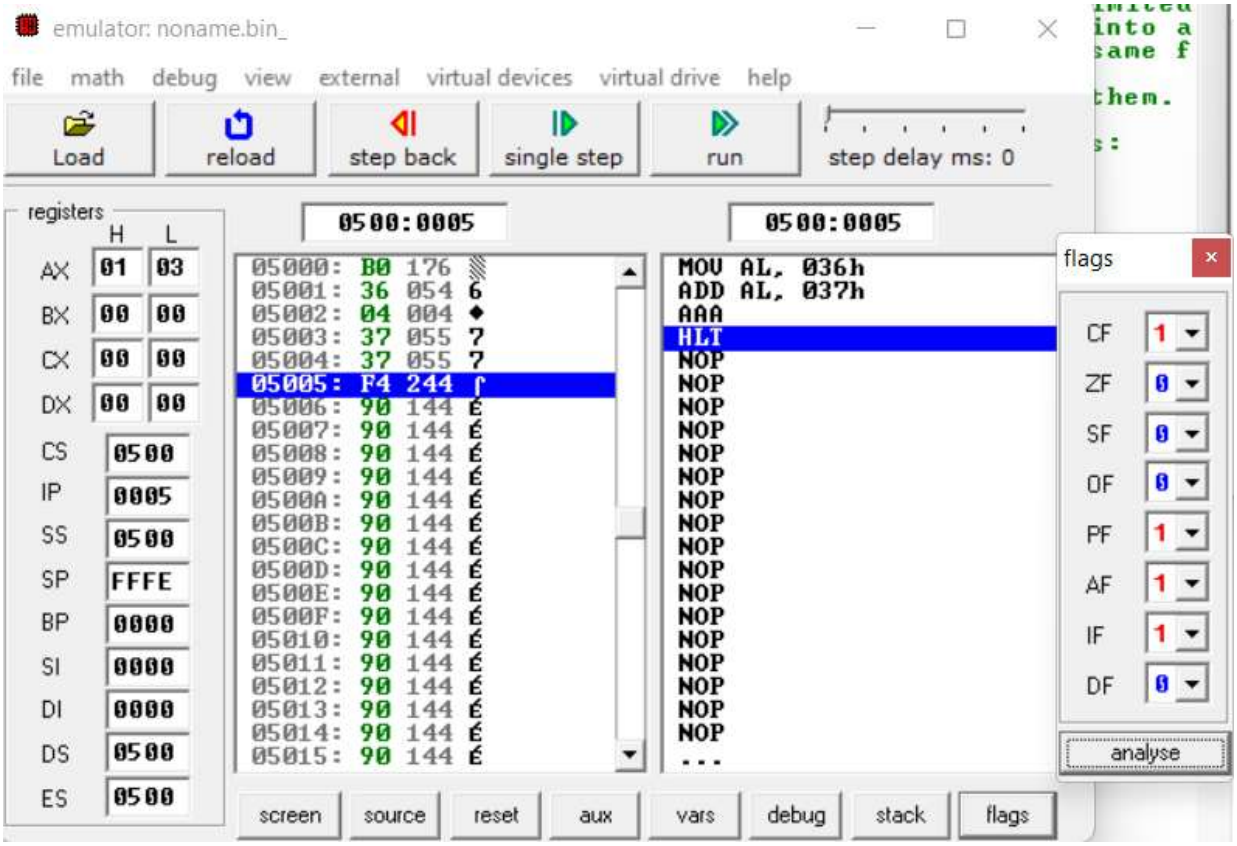
AAA (ASCII adjust after addition)

Program:

```
MOV AL,36H
ADD AL,37H
AAA
HLT
```

Output:

AX : 0103H



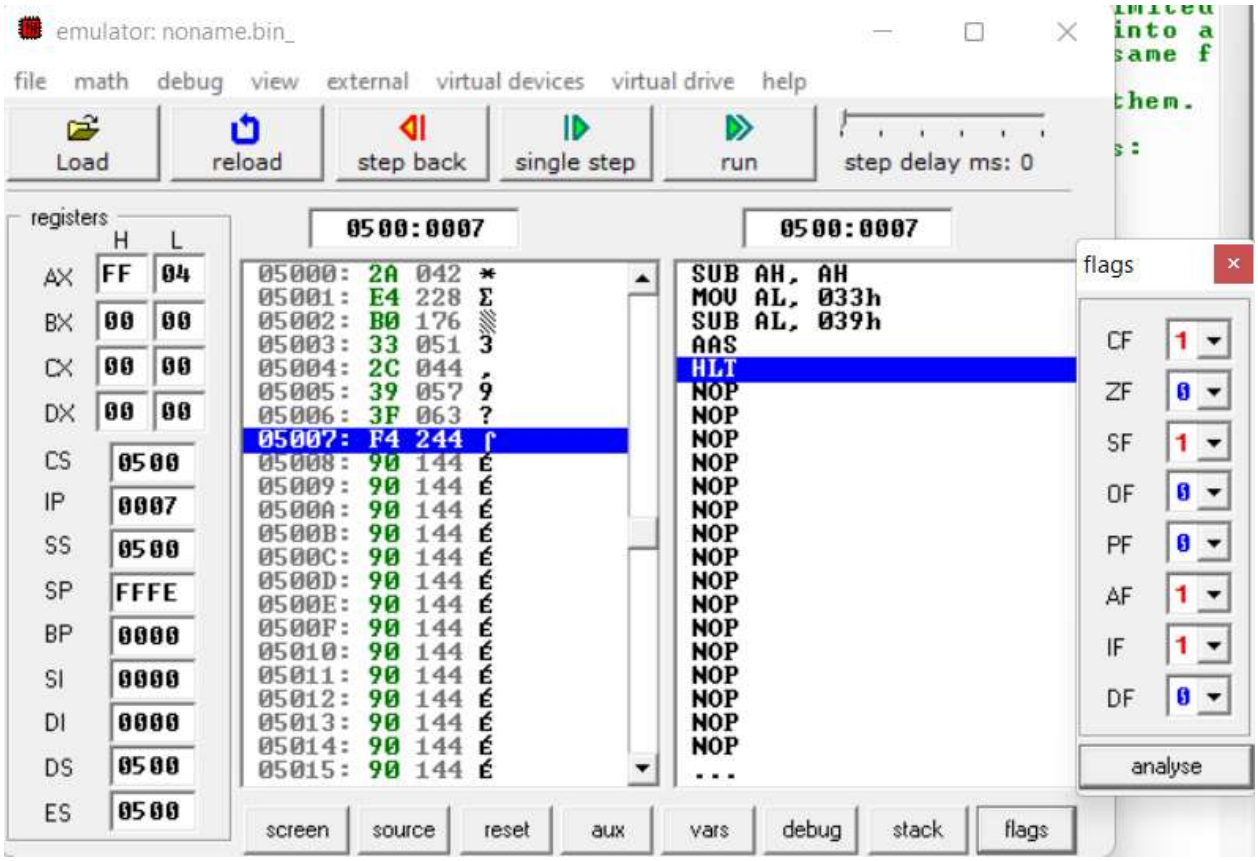
AAS (ASCII Adjust after Subtraction)

Program:

```
SUB AH,AH
MOV AL,33H
SUB AL,39H
AAS
HLT
```

Output:

AX : FF04H



AAM (ASCII ADJUST AFTER MULTIPLICATION)

Program:

```
MOV AL,[1234H]
MOV BL,[1235H]
MUL BL
AAM
MOV [1236H],AX
HLT
```

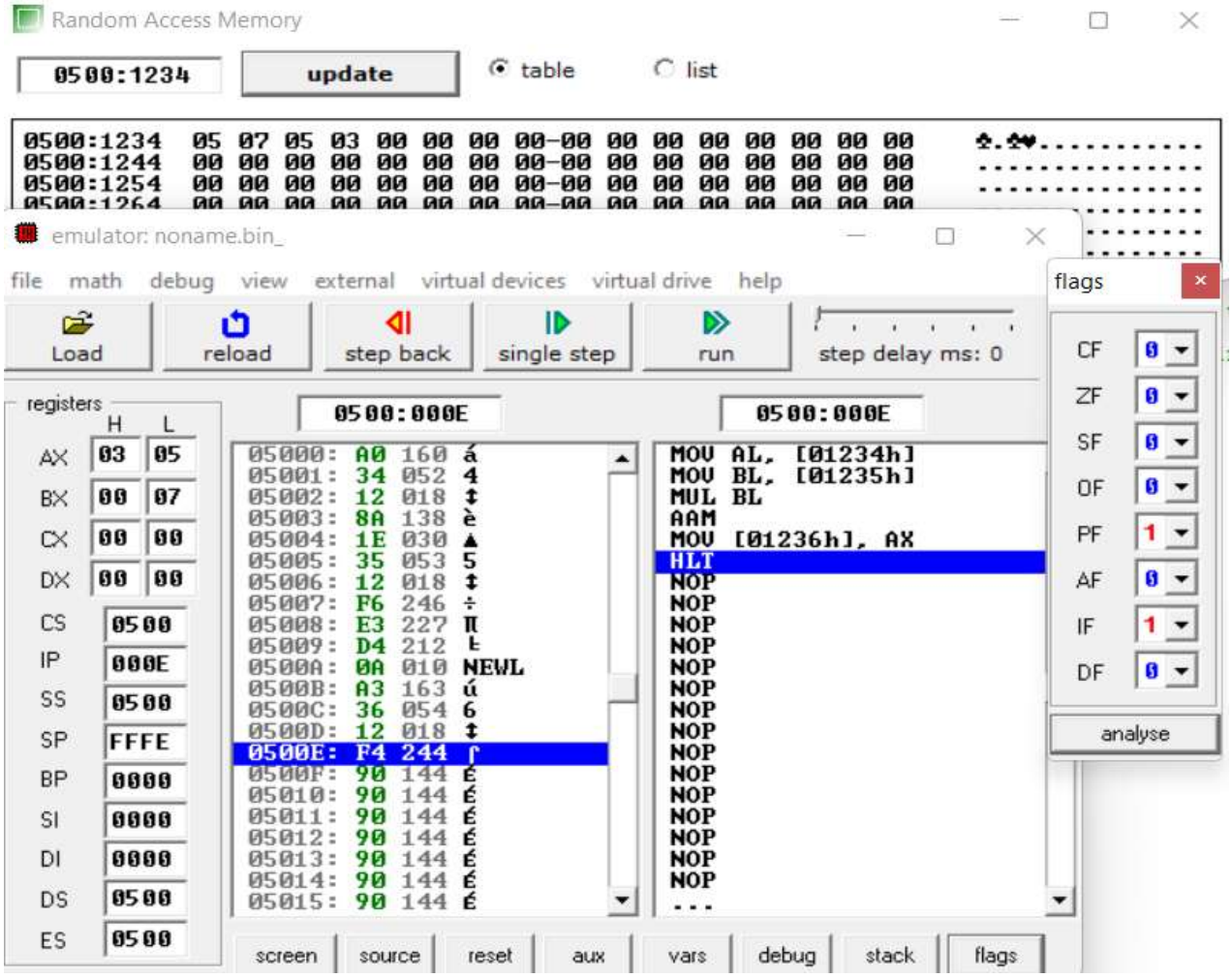
Output:

[1234H] : 05H [1235H] : 07H

Input:

AX : 0305H

[1236H]: 05H [1237H]: 03H



AAD (ASCII ADJUST BEFORE DIVISION)

Program:

```
MOV AX,[1234H]
MOV BL,[1236H]
AAD
DIV BL
MOV [1237H],AX
HLT
```

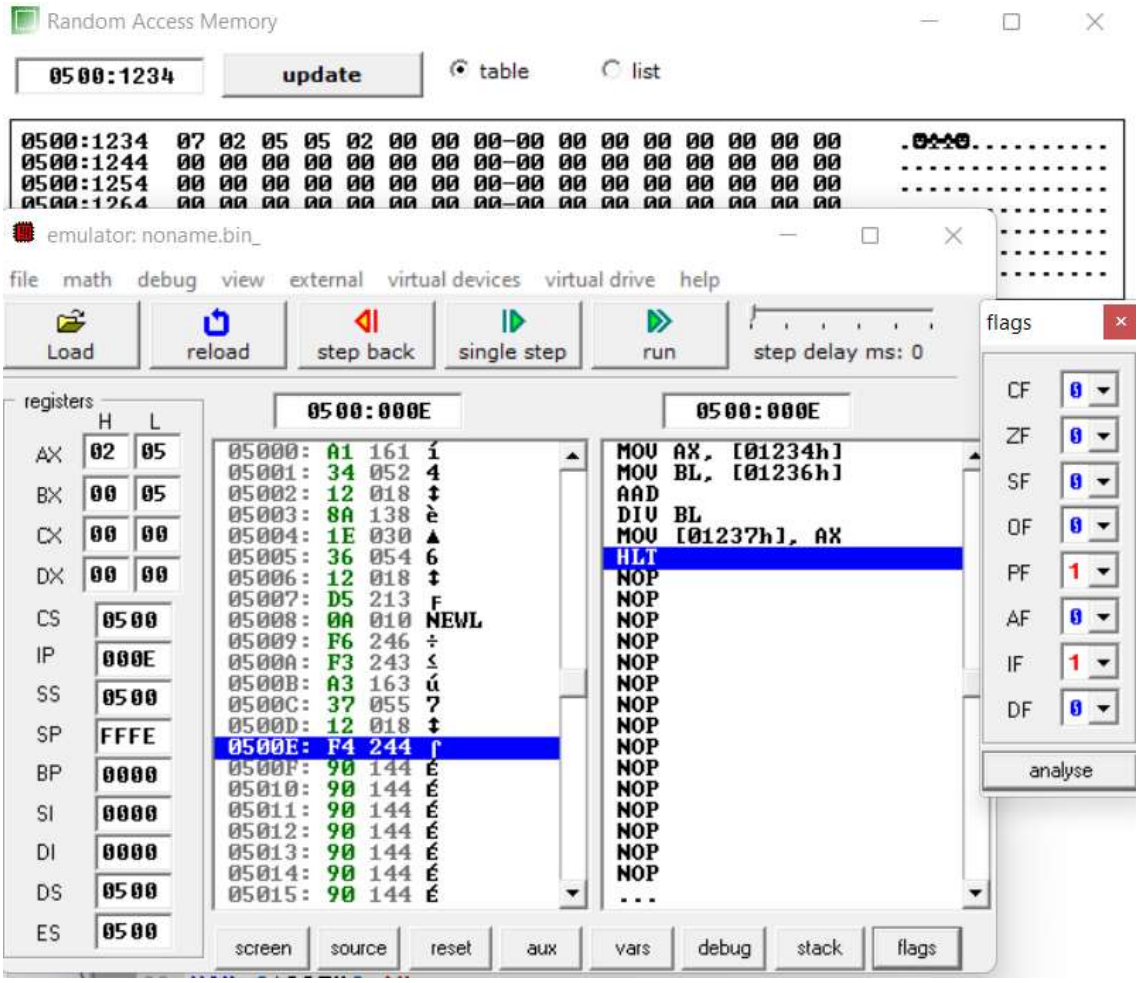
Output:

[1234H] : 07H [1235H] : 02H [1236H] : 05H

Input :

AX : 0205H

[1237H] : 05H [1238H] : 02H



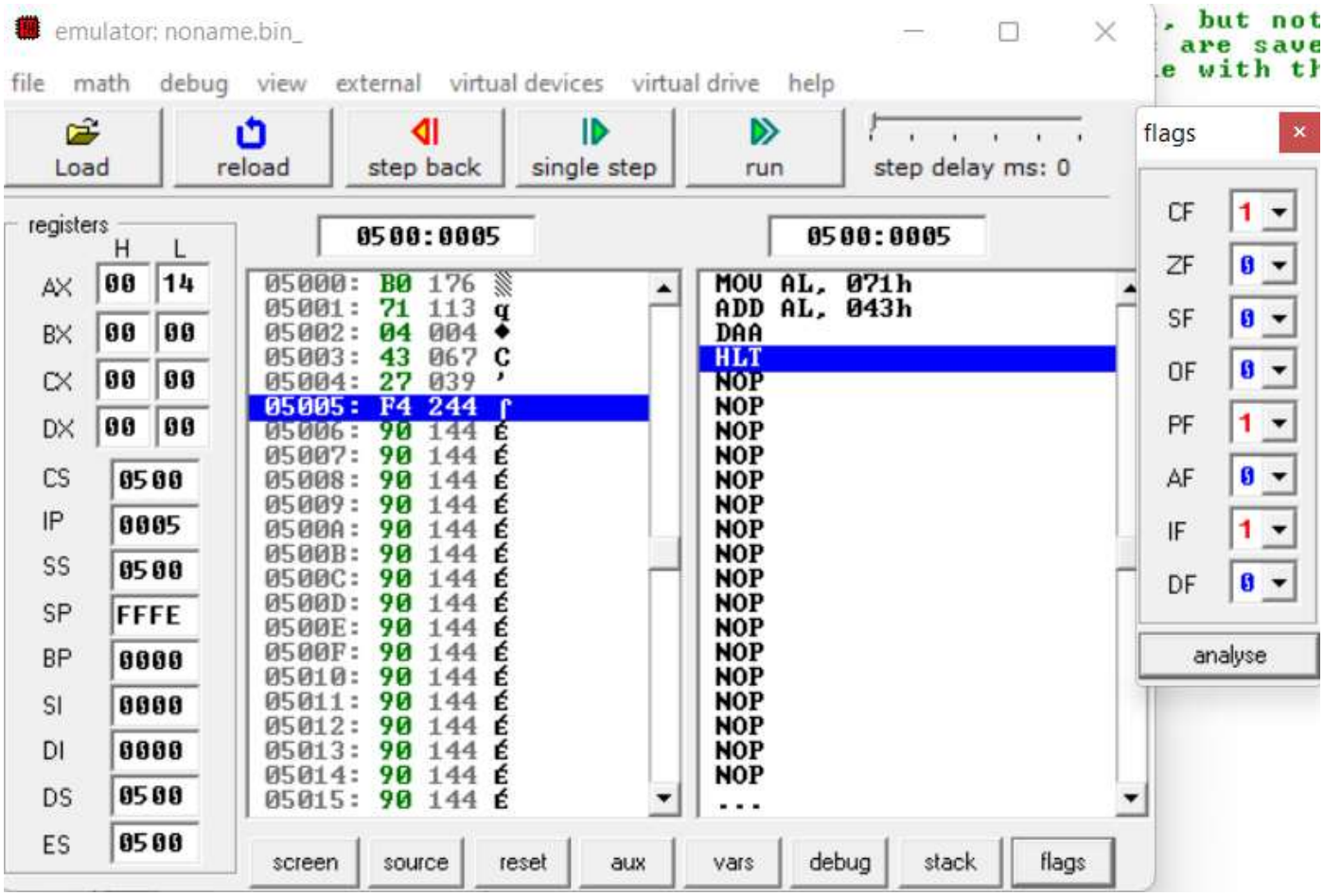
DAA (DECIMAL ADJUST FOR ADDITION)

Program:

```
MOV AL,71H
ADD AL,43H
DAA
HLT
```

Output:

AX: 0014H



DAS (DECIMAL ADJUST FOR SUBTRACTION)

Program:

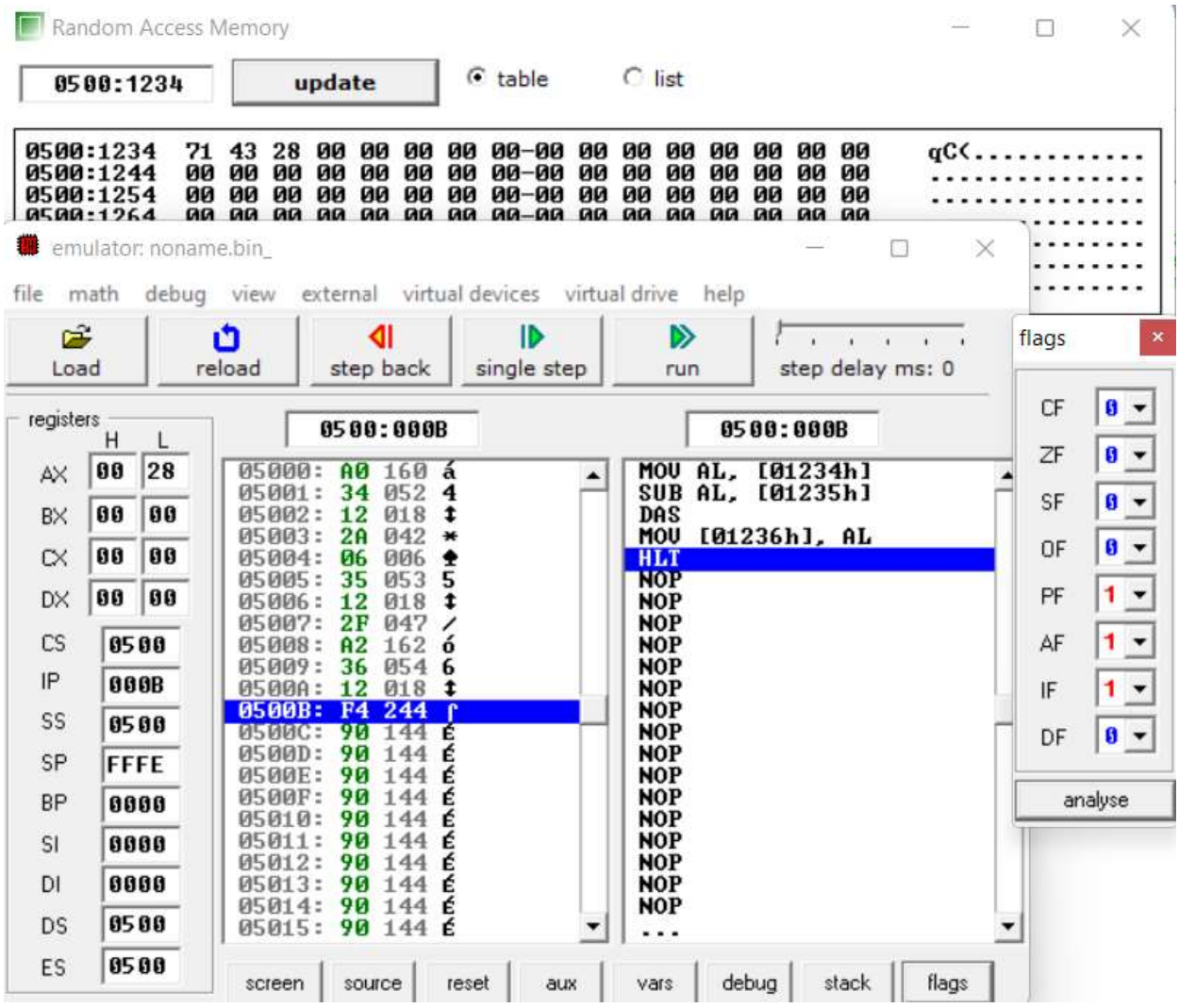
```
MOV AL,[1234H]
SUB AL,[1235H]
DAS
MOV [1236H],AL
HLT
```

Input:

[1234H] : 71H [1235H] : 43H

Output:

AL : 28H [1236H] : 28H



Assignment 21

Q. Write an assembly language program to find out the count of positive numbers and negative numbers from a series of signed numbers in 8086.

Ans.

Program:

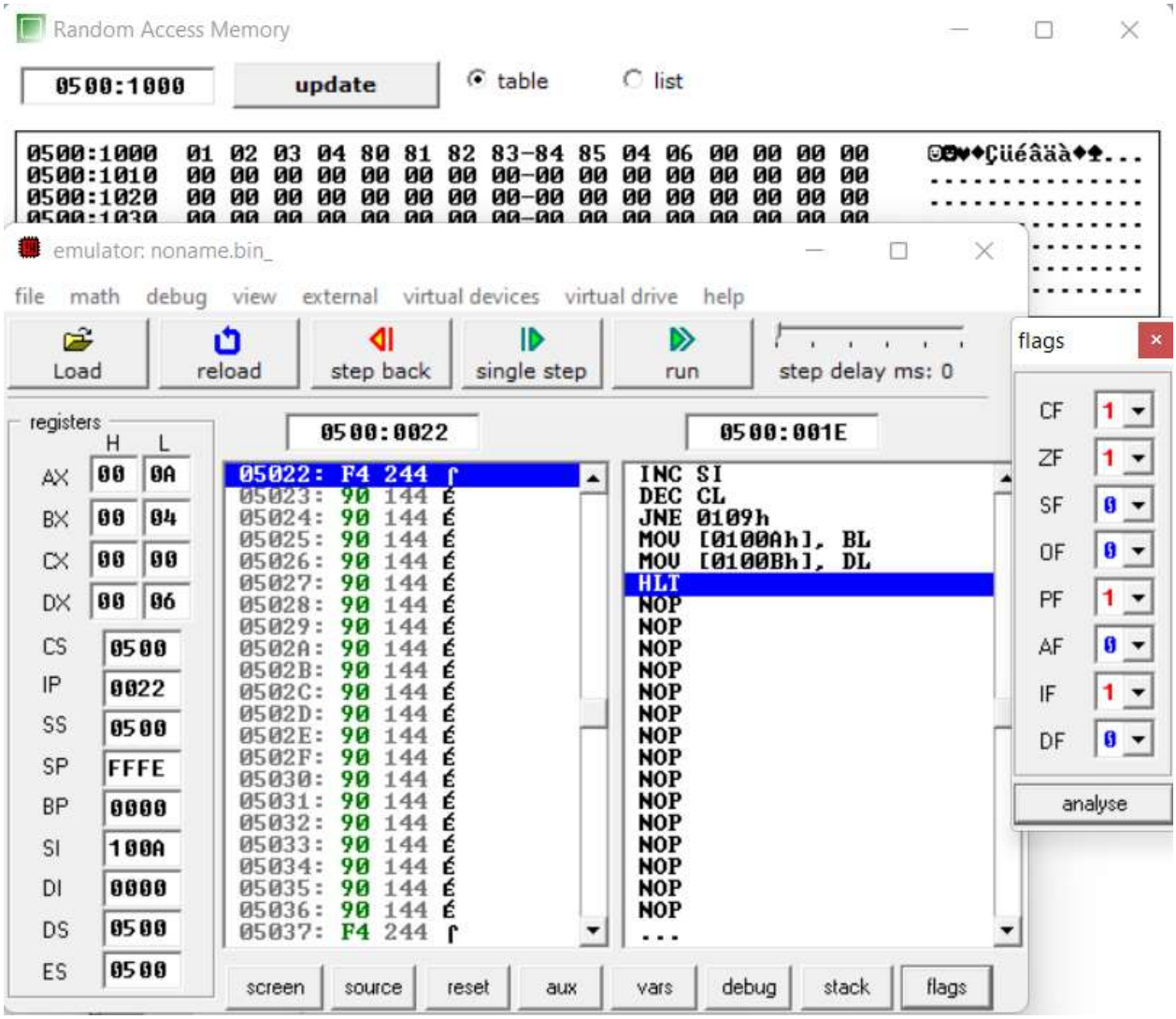
```
MOV CL, 0AH
MOV BL, 00H
MOV DL,00H
LEA SI,[1000H]
L1:MOV AL,[SI]
SHL AL,01
JNC L2
INC DL
JMP L3
L2: INC BL
L3: INC SI
DEC CL
JNZ L1
MOV [100AH],BL
MOV [100BH],DL
HLT
```

Input:

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

Output:

[100AH] : 04	[100BH] : 06


Random Access Memory
0500:1000
update
table
list
0500:1000 01 02 03 04 80 81 82 83-84 85 04 06 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
emulator: noname.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
flags
registers
H L
AX 00 0A
BX 00 04
CX 00 00
DX 00 06
CS 0500
IP 0022
SS 0500
SP FFFE
BP 0000
SI 100A
DI 0000
DS 0500
ES 0500
0500:0022
0500:001E
05022: F4 244 ⌠
05023: 90 144 É
05024: 90 144 É
05025: 90 144 É
05026: 90 144 É
05027: 90 144 É
05028: 90 144 É
05029: 90 144 É
0502A: 90 144 É
0502B: 90 144 É
0502C: 90 144 É
0502D: 90 144 É
0502E: 90 144 É
0502F: 90 144 É
05030: 90 144 É
05031: 90 144 É
05032: 90 144 É
05033: 90 144 É
05034: 90 144 É
05035: 90 144 É
05036: 90 144 É
05037: F4 244 ⌠
INC SI
DEC CL
JNE 0109h
MOV [0100Ah], BL
MOV [0100Bh], DL
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
CF 1
ZF 1
SF 0
OF 0
PF 1
AF 0
IF 1
DF 0
analyse
screen
source
reset
aux
vars
debug
stack
flags

Assignment 22

Q. 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.

Ans.

Program:

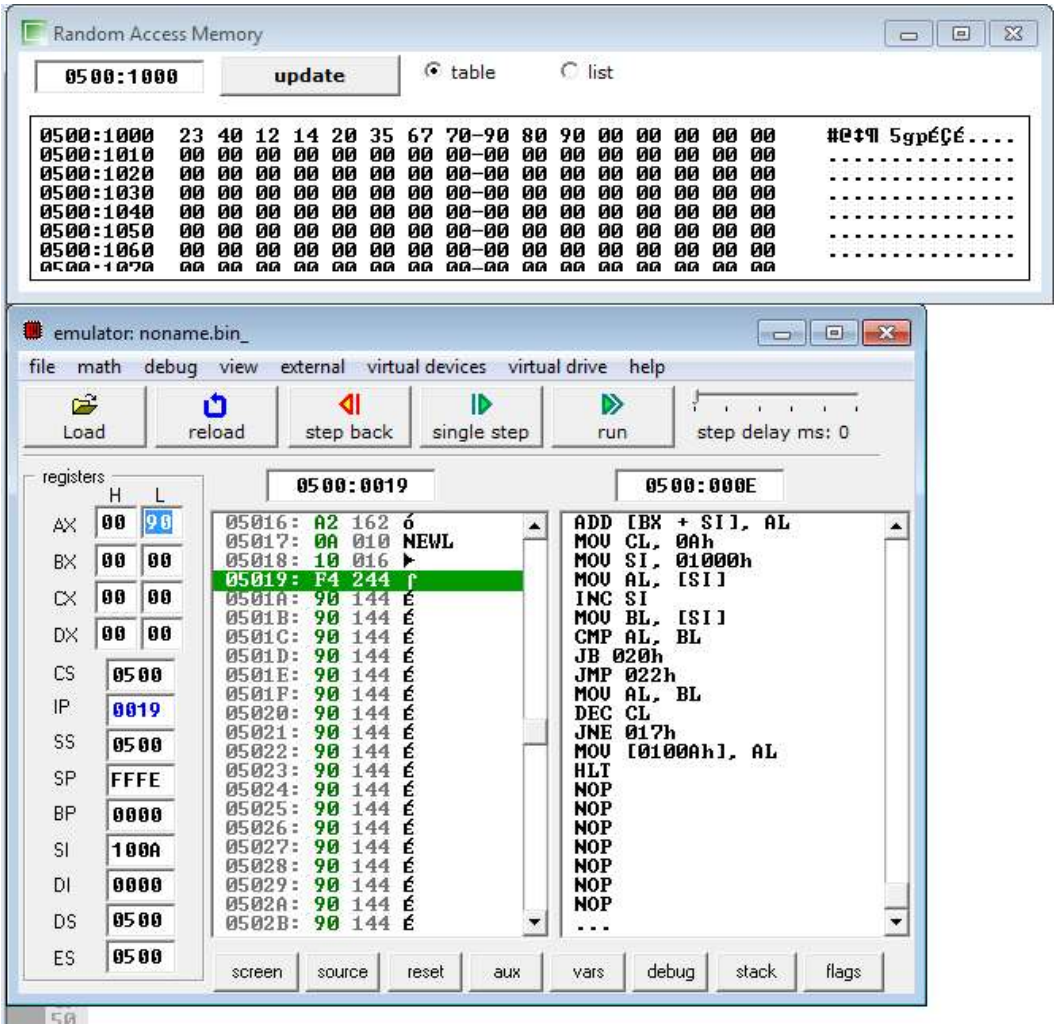
```
MOV CL, 0AH
LEA SI,[1000H]
MOV AL,[SI]
L1: INC SI
MOV BL,[SI]
CMP AL,BL
JC L2
JMP L3
L2: MOV AL,BL
L3: DEC CL
JNZ L1
MOV [100AH],AL
HLT
```

INPUT::

[1000]- 23, [1001]- 40, [1002]- 12, [1003]- 14, [1004]- 20, [1005]- 35, [1006]- 67, [1007]- 70, [1008]- 90, [1009]- 80

OUTPUT:

[100A]- 90


Random Access Memory
0500:1000
update
table
list
0500:1000 23 40 12 14 20 35 67 70-90 80 90 00 00 00 00 00 #@↕¶ 5gpÉÇÉ....
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 ....
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 90
BX 00 00
CX 00 00
DX 00 00
CS 0500
IP 0019
SS 0500
SP FFFE
BP 0000
SI 100A
DI 0000
DS 0500
ES 0500
0500:0019
0500:000E
05016: A2 162 ó
05017: 0A 010 NEWL
05018: 10 016 ►
05019: F4 244 ⌠
0501A: 90 144 É
0501B: 90 144 É
0501C: 90 144 É
0501D: 90 144 É
0501E: 90 144 É
0501F: 90 144 É
05020: 90 144 É
05021: 90 144 É
05022: 90 144 É
05023: 90 144 É
05024: 90 144 É
05025: 90 144 É
05026: 90 144 É
05027: 90 144 É
05028: 90 144 É
05029: 90 144 É
0502A: 90 144 É
0502B: 90 144 É
ADD [BX + SI], AL
MOV CL, 0Ah
MOV SI, 01000h
MOV AL, [SI]
INC SI
MOV BL, [SI]
CMP AL, BL
JB 020h
JMP 022h
MOV AL, BL
DEC CL
JNE 017h
MOV [0100Ah], AL
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags

Assignment 23

Q. 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.

Ans.

Program:

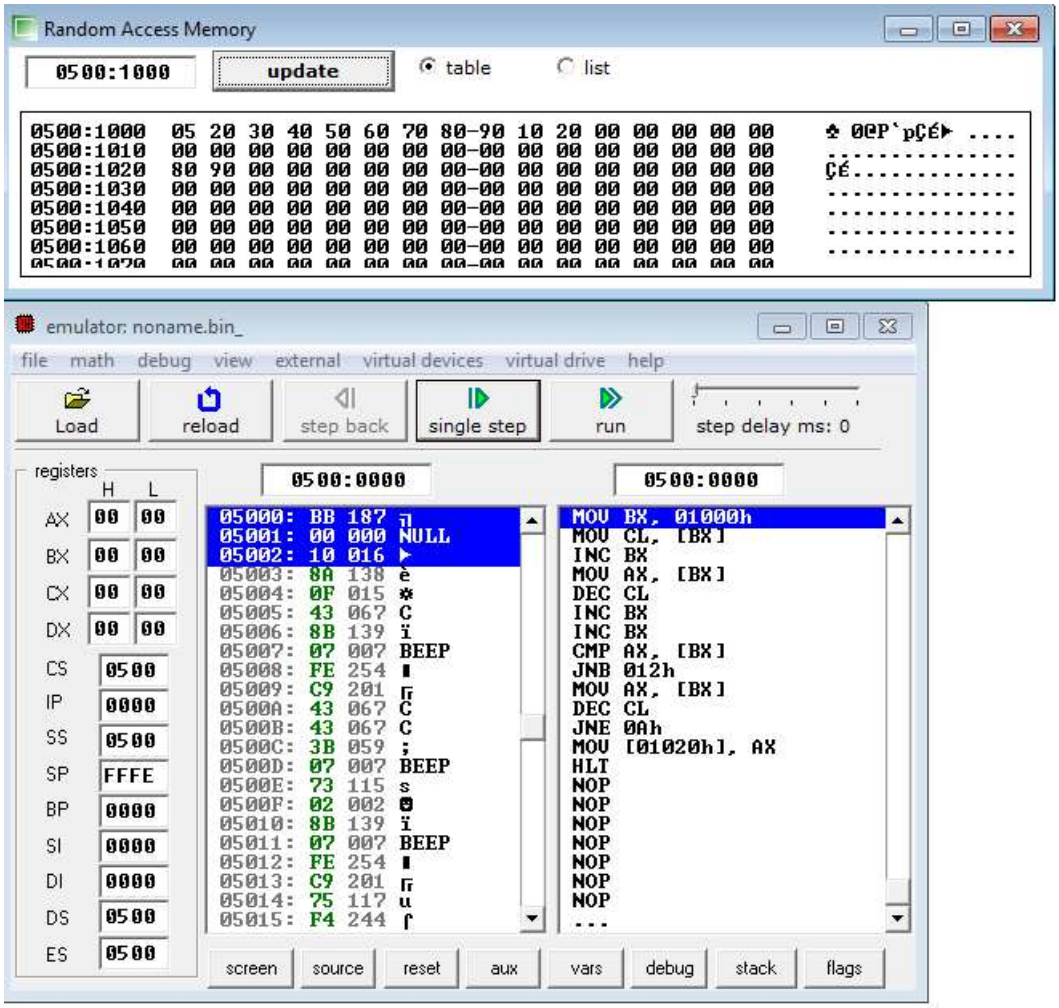
```
MOV BX, 1000H
MOV CL, [BX]
INC BX
MOV AX, [BX]
DEC CL
Back: INC BX
INC BX
CMP AX,[BX]
JNC Next
MOV AX, [BX]
Next: DEC CL
JNZ Back
MOV [1020H],AX
HLT
```

INPUT::

[1000]- 05, [1001]- 20, [1002]- 30, [1003]- 40, [1004]- 50, [1005]- 60, [1006]- 70, [1007]- 80, [1008]- 90, [1009]- 10, [100A]: 20

OUTPUT:

[1020]- 80, [1021]- 90


Random Access Memory
0500:1000
update
table
list
0500:1000 05 20 30 40 50 60 70 80-90 10 20 00 00 00 00 00
0500:1010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1020 80 90 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
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 00
CX 00 00
DX 00 00
CS 0500
IP 0000
SS 0500
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0500
ES 0500
0500:0000
05000: BB 187
05001: 00 000 NULL
05002: 10 016
05003: 8A 138 è
05004: 0F 015
05005: 43 067 C
05006: 8B 139 ï
05007: 07 007 BEEP
05008: FE 254
05009: C9 201
0500A: 43 067 C
0500B: 43 067 C
0500C: 3B 059 ;
0500D: 07 007 BEEP
0500E: 73 115 s
0500F: 02 002
05010: 8B 139 ï
05011: 07 007 BEEP
05012: FE 254
05013: C9 201
05014: 75 117 u
05015: F4 244
0500:0000
MOV BX, 01000h
MOV CL, [BX]
INC BX
MOV AX, [BX]
DEC CL
INC BX
INC BX
CMP AX, [BX]
JNB 012h
MOV AX, [BX]
DEC CL
JNE 0Ah
MOV [01020h], AX
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags

Assignment 24

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

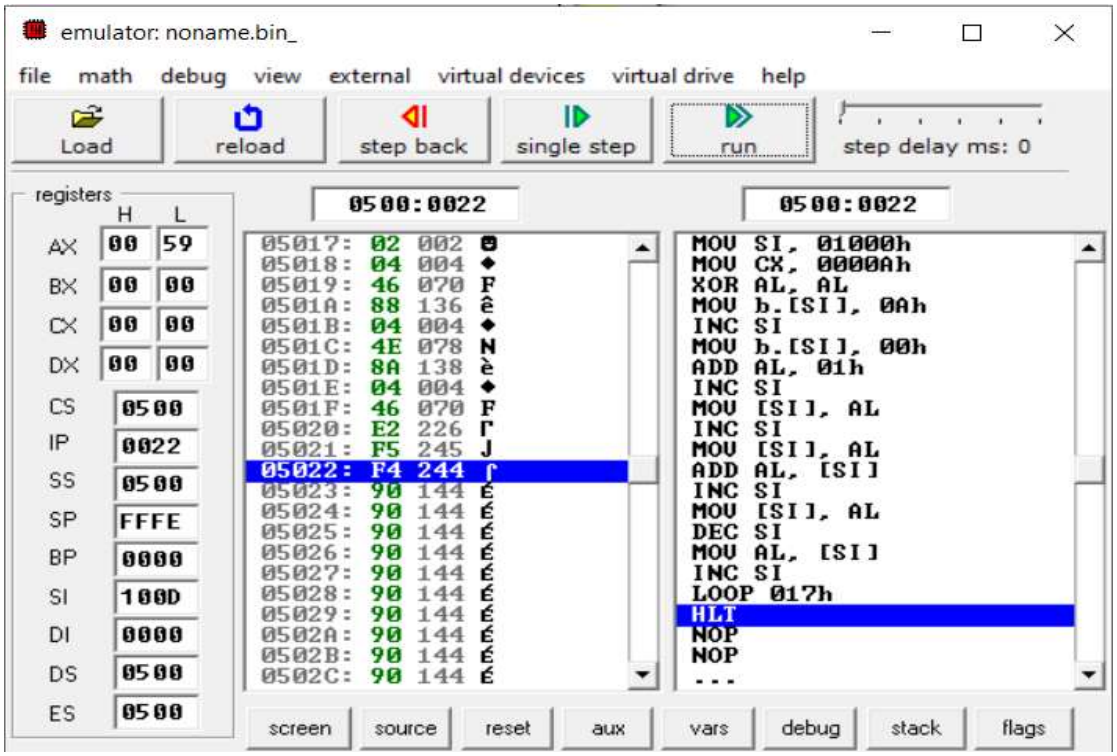
Ans.

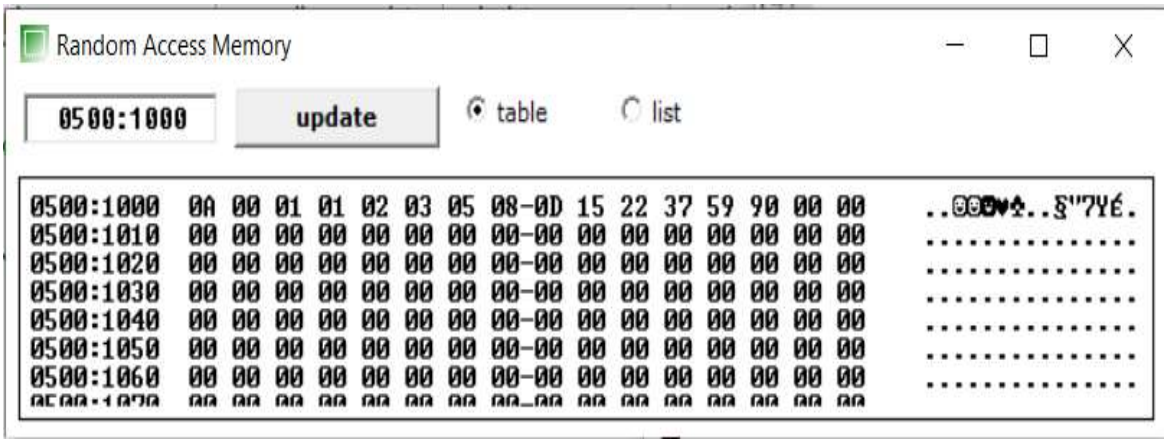
Program:

```
MOV SI,1000H
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:

[1001]- 00, [1002]- 01, [1003]- 01, [1004]- 02, [1005]- 03, [1006]- 05, [1007]- 08, [1008]- 0D, [1009]- 15, [100A]- 22, [100B]- 37, [100C]- 59, [100D]- 90


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 59
BX 00 00
CX 00 00
DX 00 00
CS 0500
IP 0022
SS 0500
SP FFFE
BP 0000
SI 100D
DI 0000
DS 0500
ES 0500
0500:0022
0500:0022
05017: 02 002
05018: 04 004
05019: 46 070 F
0501A: 88 136 ê
0501B: 04 004
0501C: 4E 078 N
0501D: 8A 138 è
0501E: 04 004
0501F: 46 070 F
05020: E2 226 Γ
05021: F5 245 J
05022: F4 244 ⌠
05023: 90 144 É
05024: 90 144 É
05025: 90 144 É
05026: 90 144 É
05027: 90 144 É
05028: 90 144 É
05029: 90 144 É
0502A: 90 144 É
0502B: 90 144 É
0502C: 90 144 É
MOV SI, 01000h
MOV CX, 0000Ah
XOR AL, AL
MOV b.[SI], 0Ah
INC SI
MOV b.[SI], 00h
ADD AL, 01h
INC SI
MOV [SI], AL
INC SI
MOV [SI], AL
ADD AL, [SI]
INC SI
MOV [SI], AL
DEC SI
MOV AL, [SI]
INC SI
LOOP 017h
HLT
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags


Random Access Memory
0500:1000
update
table
list
0500:1000 0A 00 01 01 02 03 05 08-0D 15 22 37 59 90 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

Assignment 25

Q. Write an assembly language program to perform the division 15/6 using the ASCII codes. Store the ASCII codes of the result in register DX.

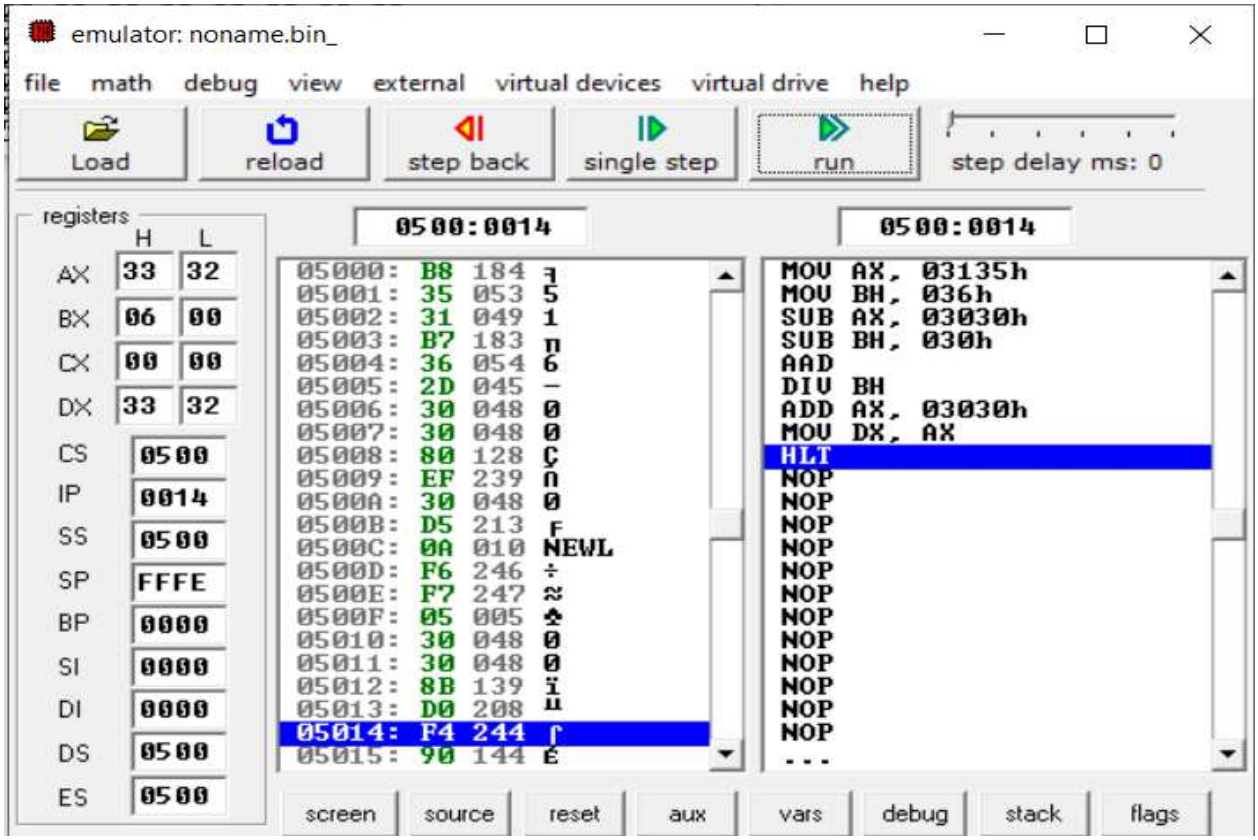
Ans.

Program:

```
MOV AX, '15'
MOV BH, '6'
SUB AX,3030h
SUB BH,30H
AAD
DIV BH
ADD AX,3030H
MOV DX,AX
HLT
```

OUTPUT:

DX- 33 32, AX- 33 32



Steps of Execution on Kit

1. Press Reset
2. Press E from the keyboard
3. Press Enter
4. Write A1000:1000
5. Press Enter
6. Now write your code line by line
7. In the last line of code write INT A5.
8. Press Enter
9. Press Reset
10. Press G
11. Press Enter
12. It will show Burst mode; it means that compile runs the code in one turn.
13. Press Enter
14. Now it will asks the segment address i.e. 1000
15. Press Enter
16. Now it will asks the offset address i.e. 1000
17. Press Enter
18. Now it has executed the program
19. It will show Cmd_word
20. Press S
21. Press Enter
22. Now you can check the contents from any place either from memory/register/IO
23. From whichever place you want to see the contents Press Enter at there.
24. For example I want to check the contents from register then I press Enter at Register
25. Now it will asks the name of the register i.e. AX (It will show the contents at AX).
26. Finally you have successfully write the program in assembly language and execute it.

Addition on 8086 kit using Dynamic Input

Program:

```
MOV AX, 1122
MOV [ 0301H ], AX
MOV AX, 1122
MOV [ 0303H ], AX
MOV AX, [ 0301H ]
MOV BX, [ 0303H ]
ADD AX, BX
INTA5
```

Subtraction on 8086 kit using Static Input

Program:

```
MOV AX, 2244
MOV BX, 1122
SUB AX, BX
INTA5
```