# LAB ACTIVITY REPORT FILE

## UCS617 : MICROPROCESSOR BASED SYSTEM DESIGN

8086 MICROPROCESSOR


**SUBMITTED BY:**

JATIN BAGGA - 102203713

KUSHAGR SHARMA - 102203714

SHAURYA DAMATHIA - 102203798

JIYA - 102203801

**GROUP : 3CO18 (3C43)**

**SUBMITTED TO:**

DR. ROHAN SHARMA




**COMPUTER SCIENCE AND ENGINEERING DEPARTMENT**

**THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY**


**JAN-MAY 2025**

# TABLE OF CONTENTS

# EXPERIMENT NO. 1

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

**CODE:**

```
MOV AX,1234H
MOV BX,1236H
ADD AX, BX
HLT
```

**IMAGES:**

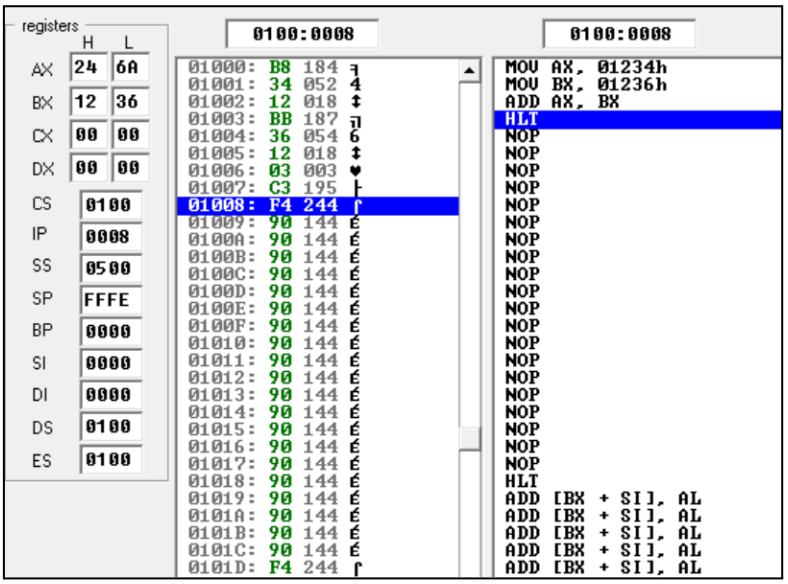


Fig 1.1

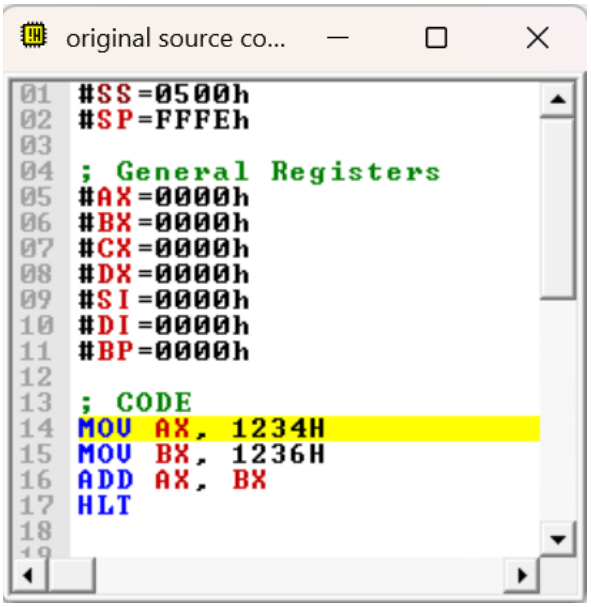


Fig 1.2

# EXPERIMENT NO. 2

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

**CODE:**

```
MOV AX,1234H
MOV BX,1236H
SUB AX, BX
HLT
```

**IMAGES:**

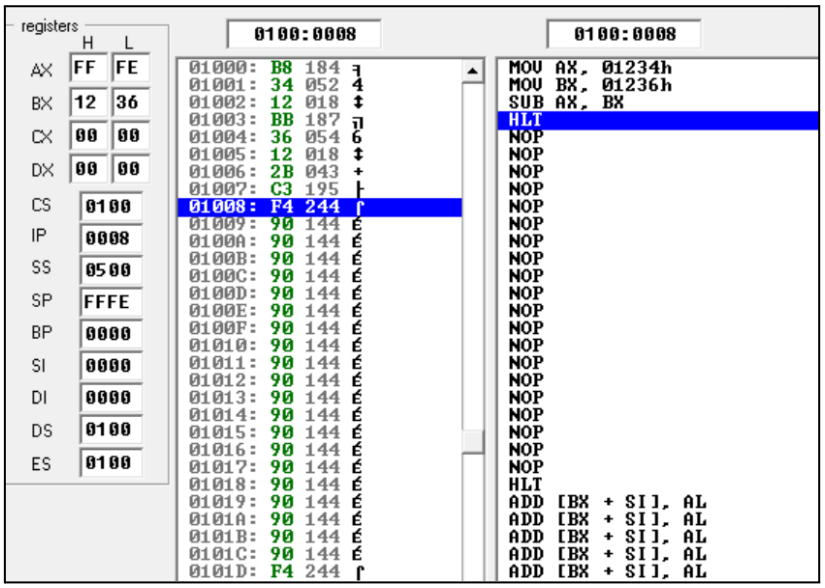


Fig 2.1

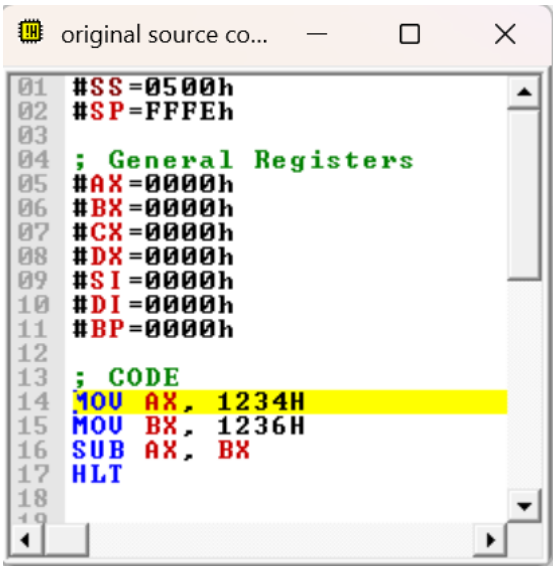


Fig 2.2

# EXPERIMENT NO. 3

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

**CODE:**

```
MOV AX, [0301H]
MOV BX, [0303H]
MUL BX
HLT
```

**IMAGES:**

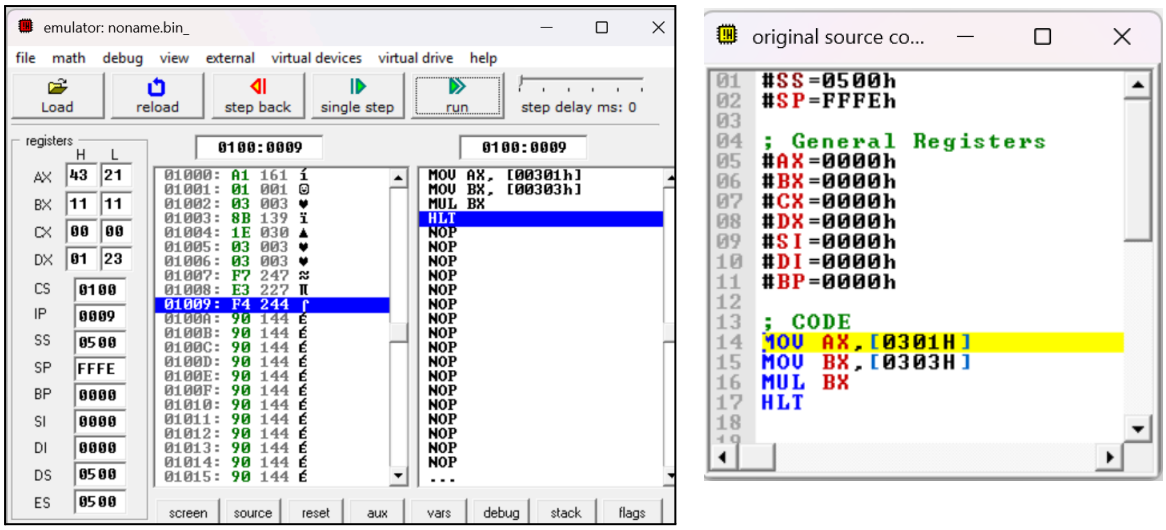


Fig 3.1

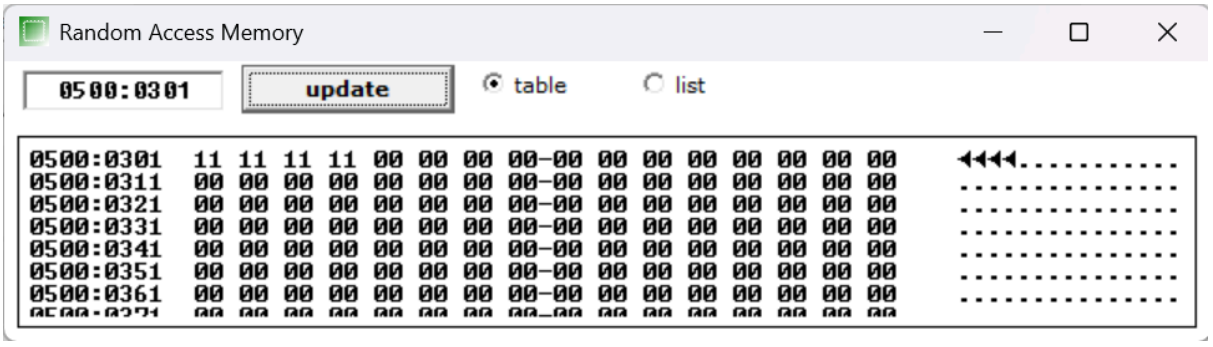


Fig 3.2

# EXPERIMENT NO. 4

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

**CODE:**

```
MOV AX, [0301H]
MOV BX, [0303H]
MUL BX
HLT
```

**IMAGES:**

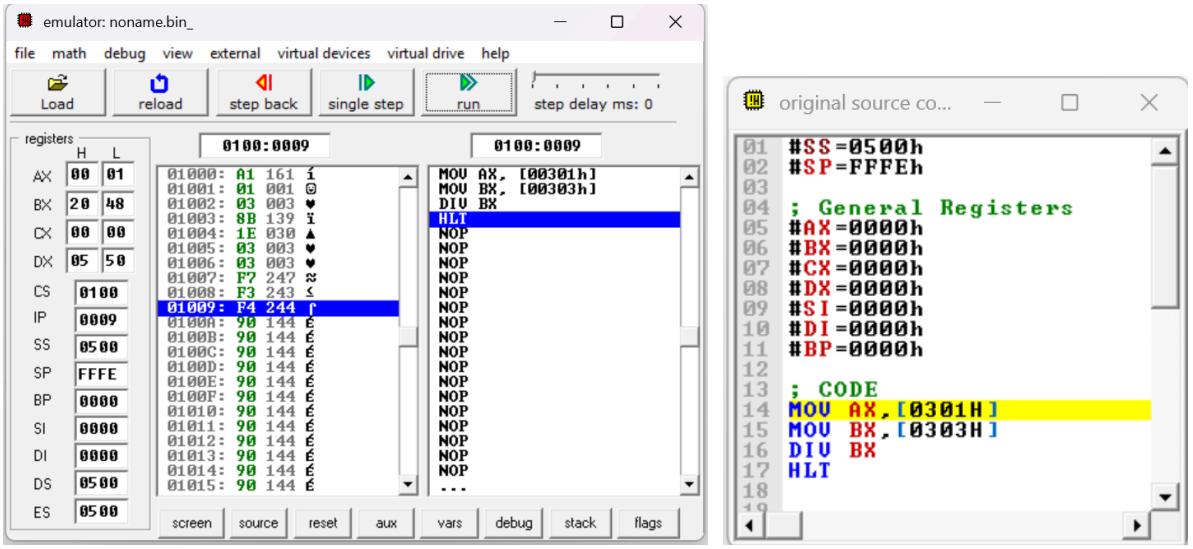


Fig 4.1

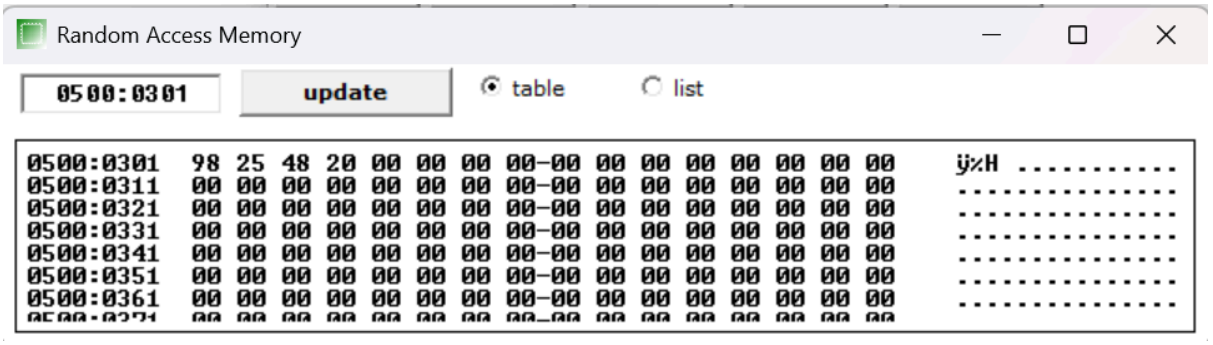


Fig 4.2

# EXPERIMENT NO. 5

**Q)** Write an assembly language program to demonstrate AAA, AAS, AAM, AAD, DAA and DAS in 8086. MOV AX,[0301H]

**CODES:**

1. **AAA**

```
MOV AX,0032H
MOV BX,0033H
ADD AX,BX
AAA
HLT
```

2. **AAS**

```
MOV AL,0033H
SUB AX,0039H
AAS
HLT
```

3. **AAM**

```
MOV AL,03H
MOV BL,09H
MUL BL
AAM
HLT
```

4. **AAD**

```
MOV AX,0033H
MOV BX,0032H
AAD
DIV BX
HLT
```

5. **DAA**

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

6. **DAS**

```
MOV AL,71H
SUB AL,43H'
DAS
HLT
```

**IMAGES:**

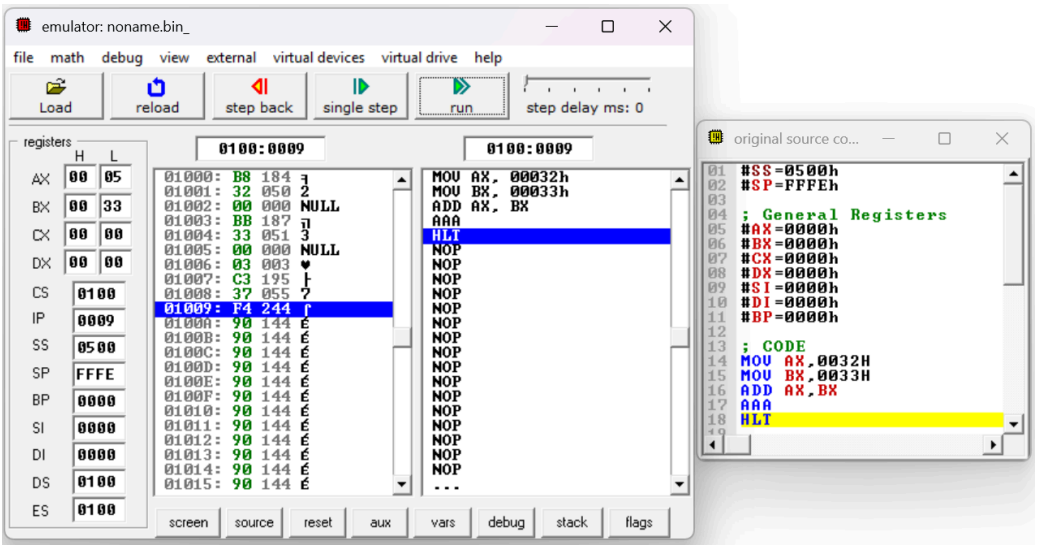


Fig 5.1: AAA

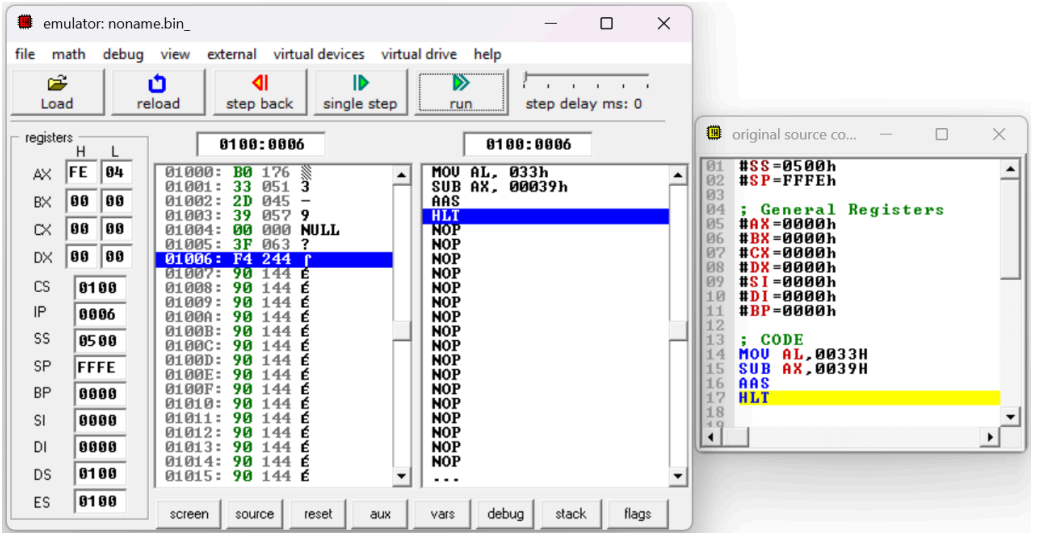


Fig 5.2: AAS

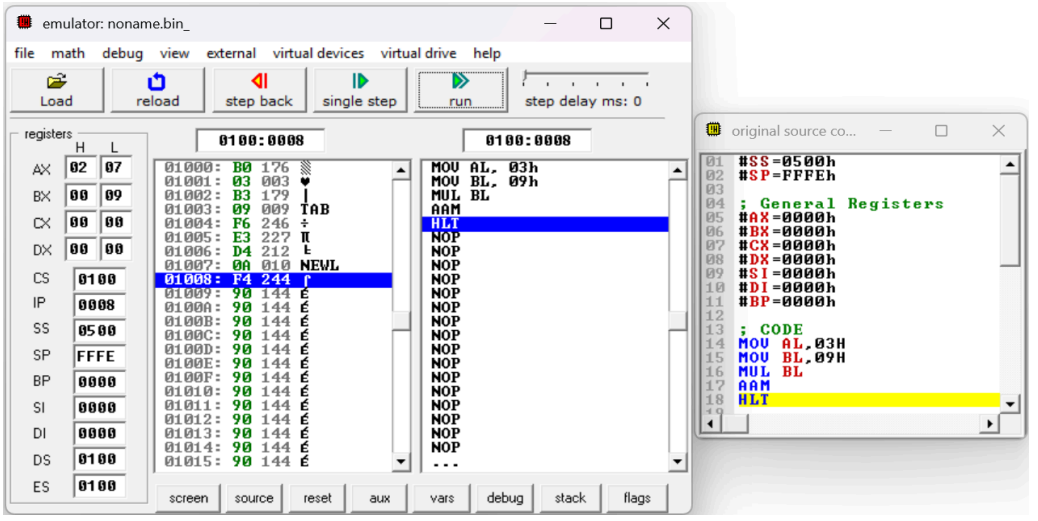


Fig 5.3: AAM

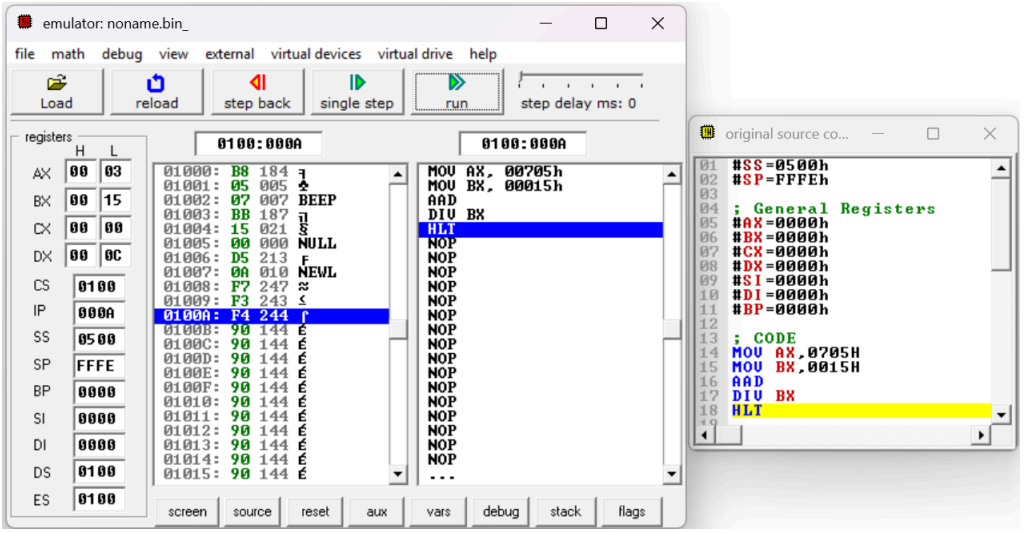


Fig 5.4: AAD

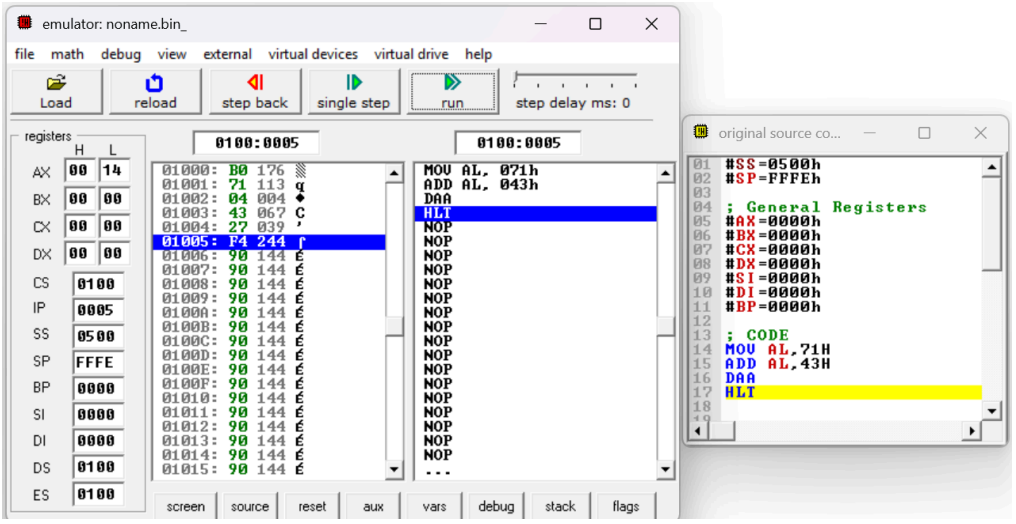


Fig 5.5: DAA

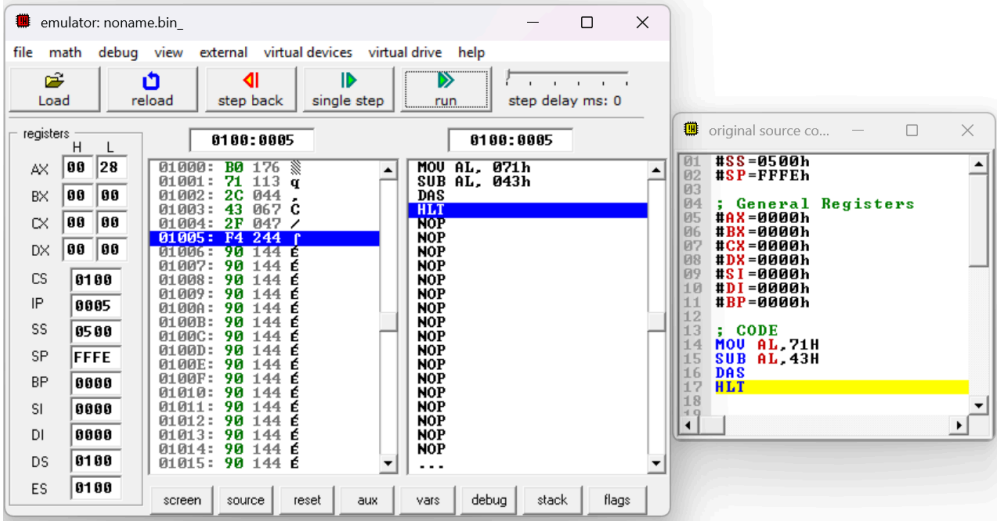


Fig 5.6: DAS

# EXPERIMENT NO. 6

**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.

**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
```

**IMAGES:**

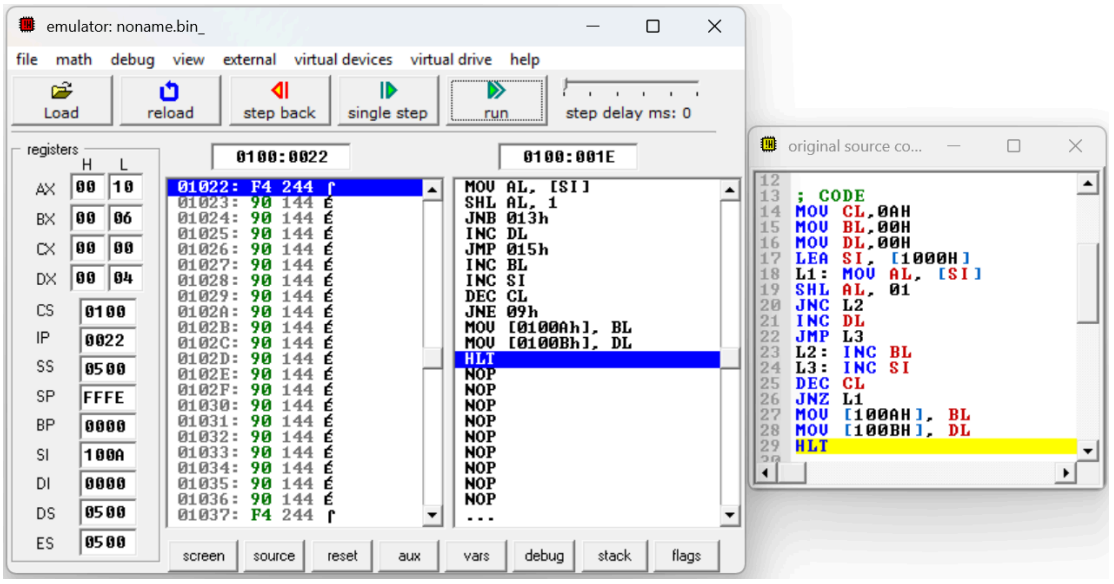


Fig 6.1

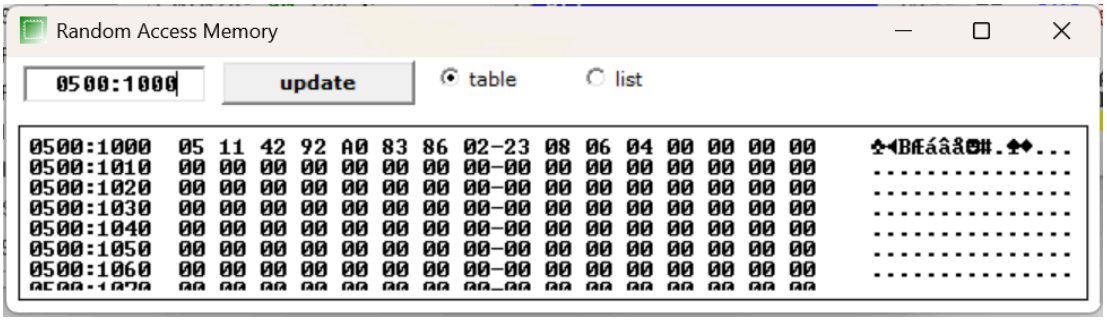
Random Access Memory
0500:1000
update
table
list
0500:1000 05 11 42 92 A0 83 86 02-23 08 06 04 00 00 00 00
0500:1010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1040 00 00 00 00 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

Fig 6.2

# EXPERIMENT NO. 7

**Q)** Write an assembly language program 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
```

**IMAGES:**

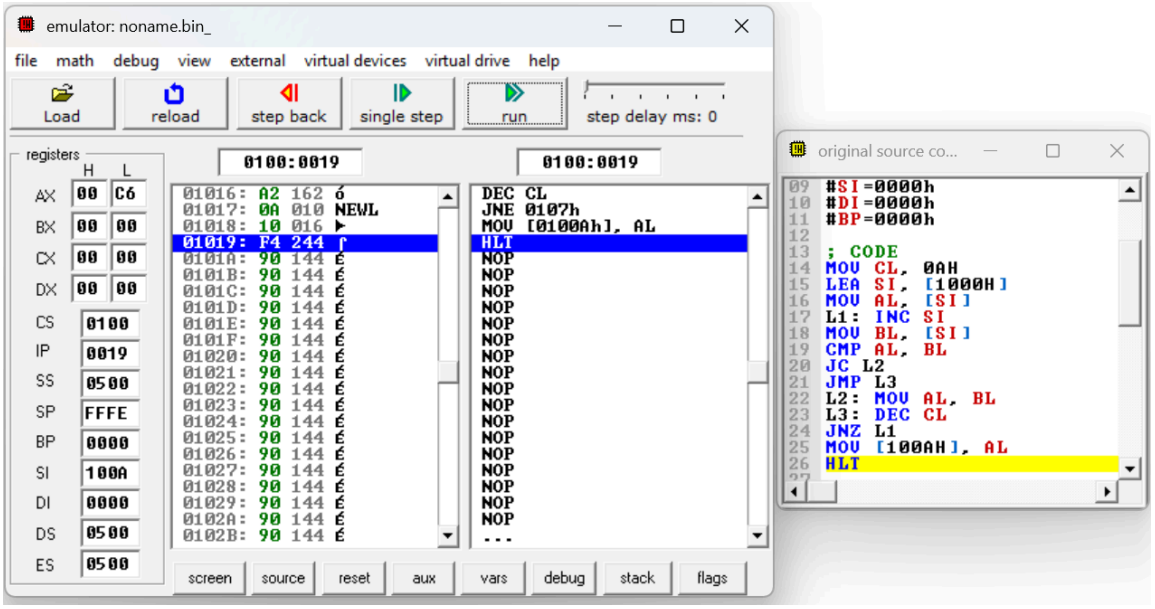


Fig 7.1

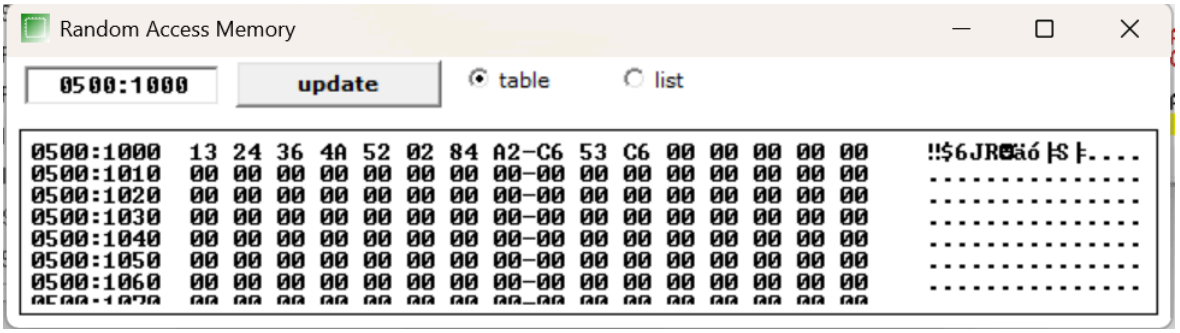
Random Access Memory
0500:1000
update
table
list
0500:1000 13 24 36 4A 52 02 84 A2-C6 53 C6 00 00 00 00 00
0500:1010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500:1040 00 00 00 00 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

Fig 7.2

# EXPERIMENT NO. 8

**Q)** Write an assembly language program 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 CL, 0AH
LEA SI, [1000H]
MOV AX, [SI]
L1: INC SI
INC SI
MOV BX, [SI]
CMP AX, BX
JC L2
JMP L3
L2: MOV AX, BX
L3: DEC CL
JNZ L1
MOV [100AH], AX
HLT
```

**IMAGES:**

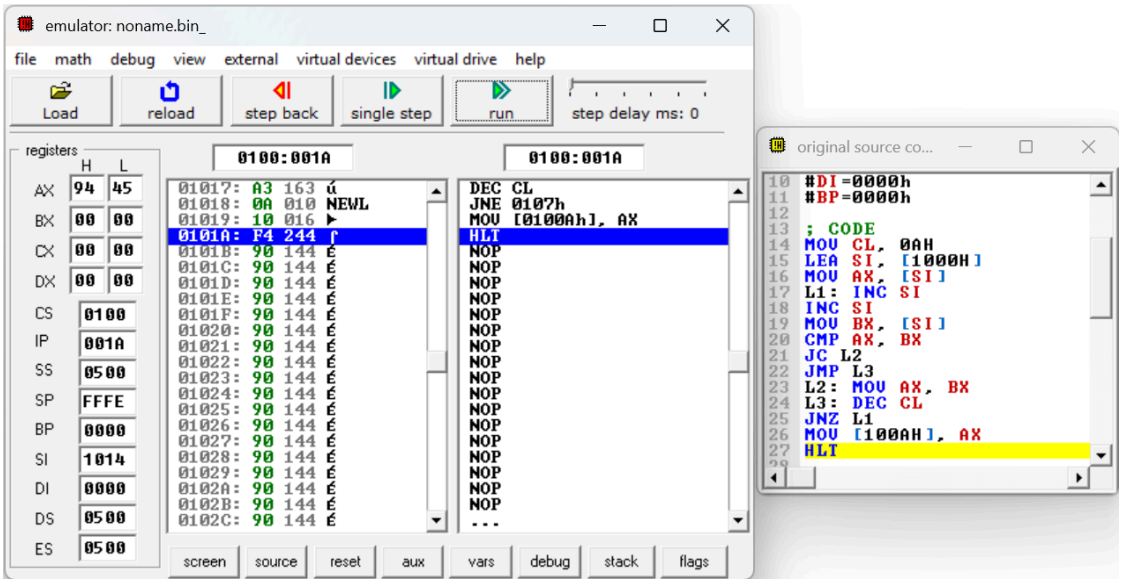


Fig 8.1

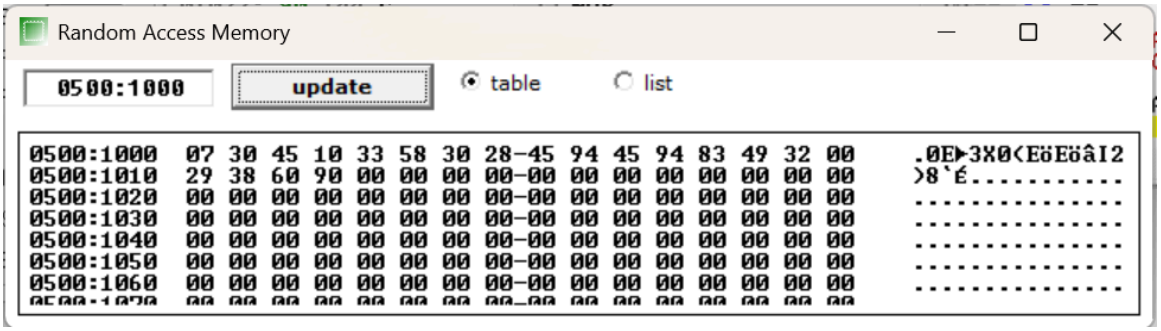
Random Access Memory
0500:1000
update
table
list
0500:1000 07 30 45 10 33 58 30 28-45 94 45 94 83 49 32 00 .0E►3X0(E÷E÷âI2
0500:1010 29 38 60 90 00 00 00 00-00 00 00 00 00 00 00 00 )8`É............
0500:1020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0500:1030 00 00 00 00 00 00 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 ................

Fig 8.2