A Laboratory Manual
for

Microprocessor

(22415)

Semester – IV

(CO, CM, CW)

Maharashtra State
Board of Technical Education, Mumbai
(Autonomous) (ISO 9001:2015) (ISO/IEC 27001:2013)

Maharashtra State
Board of Technical Education

Certificate

This is to certify that Mr. / Ms. ……. Roll No……of Fourth Semester of Diploma in …… of Institute ……..…….. (Code……..) has attained predefined practical outcomes (PROs) satisfactorily in course **Microprocessor (22415)** for the academic year 20…….to 20…..... as prescribed in the curriculum.

Place ……. Enrollment No……

Date: ..... Exam Seat No. ……..

Course Teacher **Head of the Department** **Principal**

Seal of the Institute

Sr.no	Practical Outcome	Date of performance	Date of submission	Marks	Remark
1	Identify various pins of the given microprocessor.	6/1/24	13/1/24	25/25	
2	Use Assembly Language Programming Tools and functions	13/1/24	20/1/24	24/25	
3	Use different addressing mode instruction in program Write an Assembly Language Program (ALP) to add two given 8 and 16 bit numbers. (b) Write an Assembly Language Program (ALP) to subtract two given 8 and 16 bit numbers.	20/1/24	27/1/24	23/25	
4	(a) Write an ALP to multiply two given 8 and 16 bit unsigned numbers. (b) Write an ALP to multiply two given 8 and 16 bit signed numbers.	27/1/24	3/2/24	23/25	
5	(a) Write an ALP to divide two unsigned numbers 02 (b) Write an ALP to divide two signed numbers	3/2/24	10/2/24	23/25	
6	Write an ALP to add, subtract, multiply, divide two BCD numbers.	10/2/24	17/2/24	23/25	
7	(a) Write an ALP to perform block transfer data using string instructions 02 (b) Write an ALP to perform block transfer data without using string instructions.	17/2/24	24/2/24	23/25	
8	Implement loop in assembly language program (a) Write an ALP to find sum of series of Hexadecimal Numbers. (b) Write an ALP to find sum of series of BCD numbers.	24/2/24	2/3/24	23/25	

9	(a) Write an ALP to find smallest number from array of n numbers. (a) Write an ALP to find largest number from array of n numbers.	2|3|24	9|3|24	23/25	
10	(a) Write an ALP to arrange numbers in array in ascending order. (b) Write an ALP to arrange numbers in array in descending order.	9|3|24	16|3|24	22/25	
11	Write an ALP for Z = (A + B) * (C + D) using Procedure	16|3|24	16|3|24	22/25	
12	Write an ALP for Z = (A + B) * (C + D) using MACRO.	16|3|24	16|3|24	22/25	

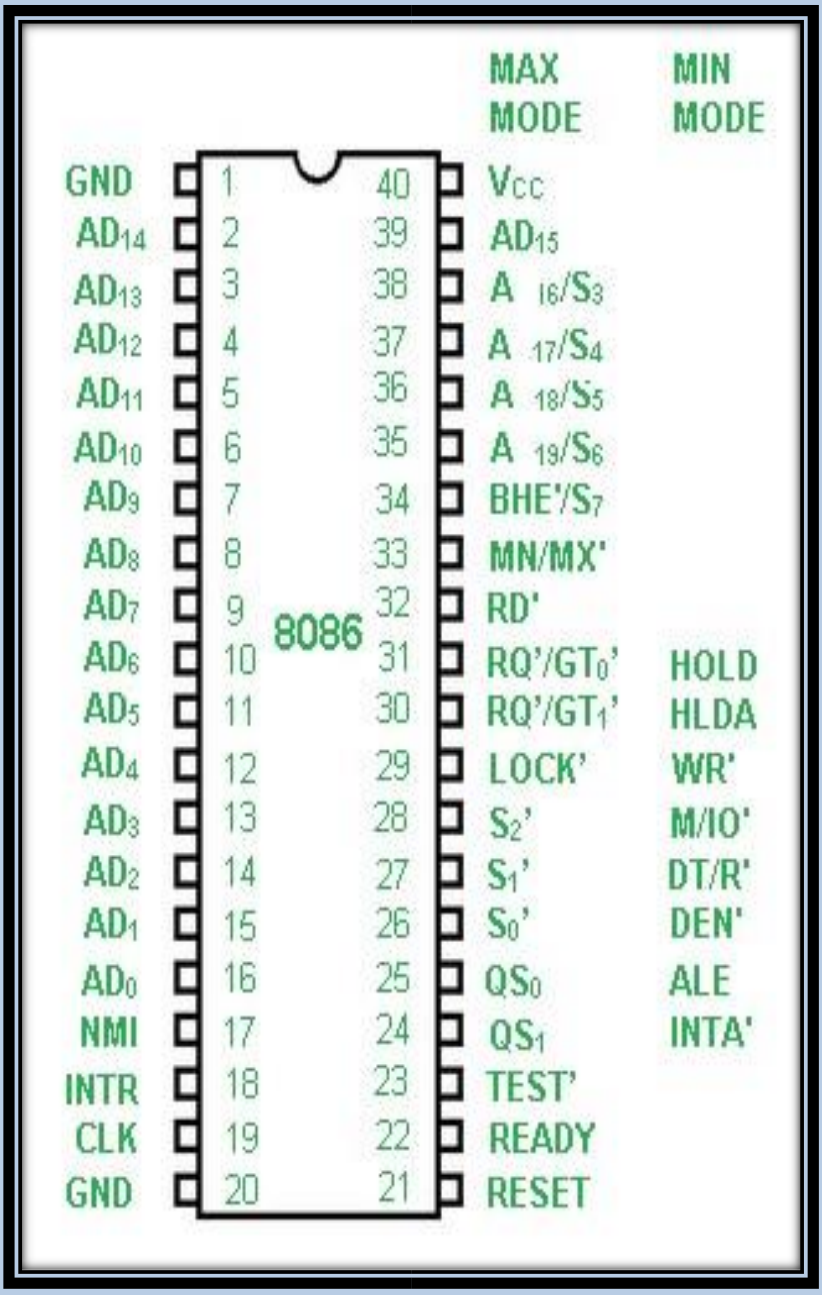
Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: Microprocessor	Subject Code:22415
Semester:4th Semester	Course: Computer Engineering
Laboratory No:L004B	Name of Subject Teacher: Pragati Mali
Name of Student: Siddharth Shah	Roll Id: 22203A0041

Experiment No:	1
Title of Experiment	Identify the various pins of the 8086 Microprocessor

AIM: Identify the various pins of the 8086 Microprocessor.

DIAGRAM:


MAX MODE
MIN MODE
8086
GND 1
AD14 2
AD13 3
AD12 4
AD11 5
AD10 6
AD9 7
AD8 8
AD7 9
AD6 10
AD5 11
AD4 12
AD3 13
AD2 14
AD1 15
AD0 16
NMI 17
INTR 18
CLK 19
GND 20
40 VCC
39 AD15
38 A16/S3
37 A17/S4
36 A18/S5
35 A19/S6
34 BHE'/S7
33 MN/MX'
32 RD'
31 RQ'/GT0' HOLD
30 RQ'/GT1' HLDA
29 LOCK' WR'
28 S2' M/IO'
27 S1' DT/R'
26 S0' DEN'
25 QS0 ALE
24 QS1 INTA'
23 TEST'
22 READY
21 RESET

PIN DESCRIPTION:

Pin number	Pin name	Function
16-2, 39	AD0 – AD15	Address/Data bus. These are low order address bus. They are multiplexed with data.
38-35	A16-A19	High order address bus. These are multiplexed with status signals.
28	M/IO’	M/IO’ signal is used to distinguish between memory and I/O operations. When it is high, it indicates I/O operation and when it is low indicates the memory operation
27	DT/R’	DT/R’ stands for Data Transmit/Receive signal. It decides the direction of data flow through the transreceiver. When it is high, data is transmitted out and vice-versa.
26	DEN’	DEN’ stands for Data Enable and is available at pin 26. It is used to enable Transreceiver 8286. The transreceiver is a device used to separate data from the address/data bus.

25	ALE	ALE stands for address enable latch and is available at pin 25. A positive pulse is generated each time the processor begins any operation. This signal indicates the availability of a valid address on the address/data lines.
29	WR' OR LOCK	WR' stands for write signal and is available at pin 29. It is used to write the data into the memory or the output device depending on the status of M/IO signal.
18	INTR	It is an interrupt request signal, which is sampled during the last clock cycle of each instruction to determine if the processor considered this as an interrupt or not
21	RESET	It causes the processor to immediately terminate its present activity. This signal is active high for the first 4 clock cycles to RESET the microprocessor.
22	READY	It is an acknowledgement signal from I/O devices that data is transferred. It is an active high signal. When it is high, it indicates that the device is ready to transfer data. When it is low, it indicates wait state.
23	TEST'	When this signal is high, then the processor has to wait for IDLE state, else the execution continues

24	INTA’	When the microprocessor receives this signal, it acknowledges the interrupt.
30	HLDA	This signal acknowledges the HOLD signal.
31	HOLD	HOLD signal indicates to the processor that external devices are requesting to access the address/data buses.
32	RD’	RD’ is available at pin 32 and is used to read signal for Read operation
33	MN/MX’	MN/MX’ stands for Minimum/Maximum and is available at pin 33. It indicates what mode the processor is to operate in; when it is high, it works in the minimum mode and vice-versa.
34	BHE’	BHE stands for Bus High Enable. It is available at pin 34 and used to indicate the transfer of data using data bus AD8-AD15. This signal is low during the first clock cycle, thereafter it is active.
17	NMI	NMI stands for non-maskable interrupt and is available at pin 17. It is an edge triggered input, which causes an interrupt request to the microprocessor.
1,20	GND	It provides ground for the microprocessor
40	VCC	It has 5V DC supply

CONCLUSION:

8086 was the first 16-bit microprocessor available in 40-pin DIP chip .By this practical we can understand the pin Number with their respective functions.

Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: Microprocessor	Subject Code: 22415
Semester: 4th Semester	Course: Computer Engineering
Laboratory No: L004B	Name of Subject Teacher: Pragati Mali
Name of Student: Siddharth.P.Shah	Roll Id: 22203A0041

Experiment No:	**2**
Title of Experiment	**Use Assembly Language Programming Tools and Functions**

Aim: To Use Assembly Language Programming Tools and Functions

Table:

Tools	Function	Software Used
Assembler	An assembler is a program that converts source code program written in assembly language into object files in machine language.	TASM
Linker	A linker is a program that combines object file created by the assembler with other object files and link libraries and produces a single executable program.	TLINK for TASM and LINK.EXE

Debugger	A debugger is a program that allows you to trace the execution of a program and examine the content of registers and memory.	Turbo Debugger for TASM
Editor	An editor is used to create assembly language source files.	Notepad

Conclusion:

Learned about essential programming tools, including text editors for code creation assemblers for converting code to machine code , linker for combining multiple code modules, and debuggers for identifying and resolving programming errors

Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject:- Microprocessors	Subject Code: 22415
Semester: 4th Semester	Course: Computer Engineering
Laboratory No: L004B	Name of Subject Teacher: Pragati Mali
Name of Student: Aditya Makwana	Roll Id: 22203A0042

Experiment No:	3
Title of Experiment	Write an Assembly Language program to perform Addition and subtraction of two 8-bit and 16-bit numbers

- Software Used:-

TASM 1.4
Text Editor (Notepad) Addition of 8-bit numbers:-

- Program Code

```
DATA SEGMENT

    NUM1 DB 55H

        NUM2 DB 11H

    DATA ENDS

CODE SEGMENT
```

ASSUME CS: CODE, DS: DATA

START:

MOV AX, DATA

MOV DS, AX

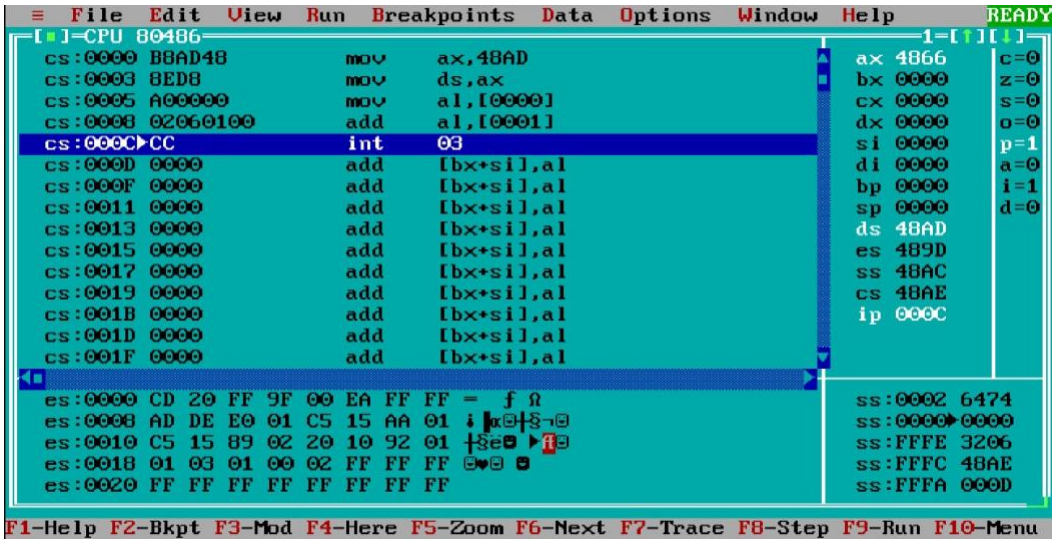
MOV AL, NUM1 ADD AL, NUM2

INT 3

CODE ENDS

END START

- Output

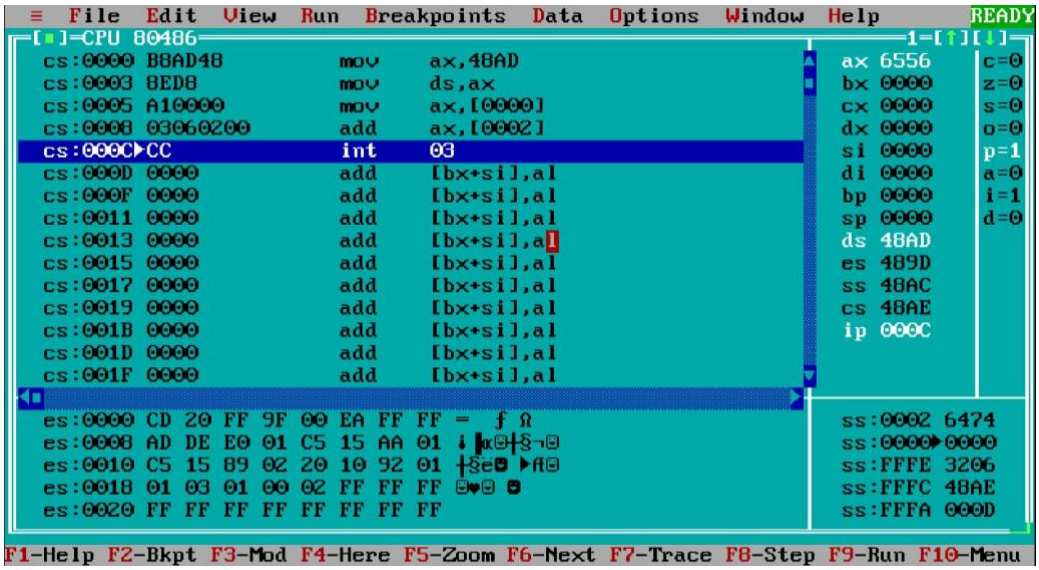


Addition of 16-bit numbers:-

- Program Code

```
DATA SEGMENT

        NUM1 DB 5555H
      NUM2 DB 1001H

DATA ENDS

      CODE SEGMENT
        ASSUME CS: CODE, DS: DATA

      START:

        MOV AX, DATA

          MOV DS, AX
        MOV AX, NUM1

  ADD AX, NUM2

          INT 3

CODE ENDS

      END START
```

- Output

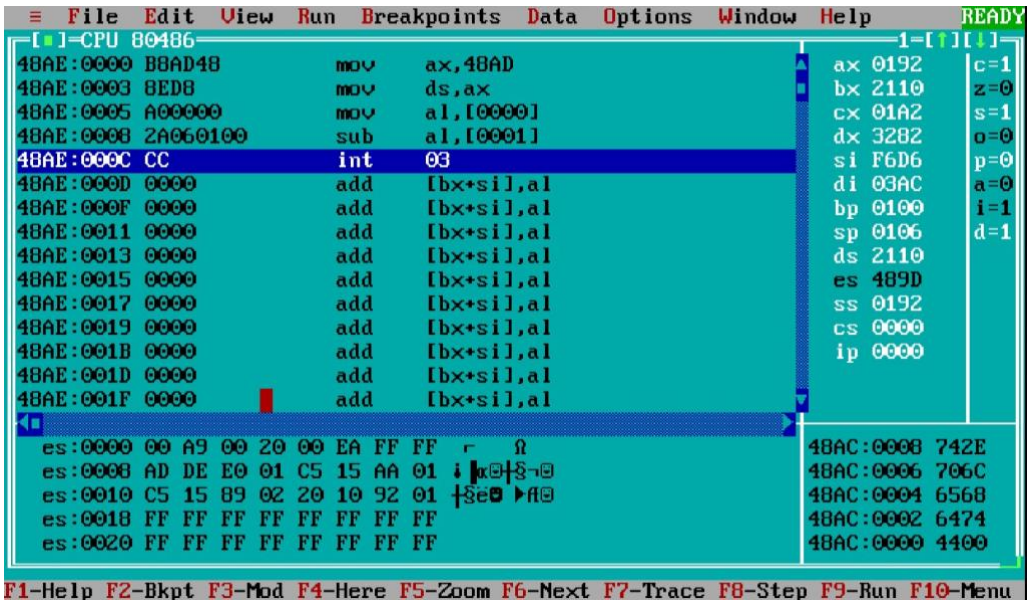

File Edit View Run Breakpoints Data Options Window Help READY
CPU 80486
cs:0000 B8AD48 mov ax,48AD
cs:0003 8ED8 mov ds,ax
cs:0005 A10000 mov ax,[0000]
cs:0008 03060200 add ax,[0002]
cs:000C CC int 03
cs:000D 0000 add [bx+si],al
cs:000F 0000 add [bx+si],al
cs:0011 0000 add [bx+si],al
cs:0013 0000 add [bx+si],al
cs:0015 0000 add [bx+si],al
cs:0017 0000 add [bx+si],al
cs:0019 0000 add [bx+si],al
cs:001B 0000 add [bx+si],al
cs:001D 0000 add [bx+si],al
cs:001F 0000 add [bx+si],al
ax 6556
bx 0000
cx 0000
dx 0000
si 0000
di 0000
bp 0000
sp 0000
ds 48AD
es 489D
ss 48AC
cs 48AE
ip 000C
c=0
z=0
s=0
o=0
p=1
a=0
i=1
d=0
es:0000 CD 20 FF 9F 00 EA FF FF
es:0008 AD DE E0 01 C5 15 AA 01
es:0010 C5 15 89 02 20 10 92 01
es:0018 01 03 01 00 02 FF FF FF
es:0020 FF FF FF FF FF FF FF FF
ss:0002 6474
ss:0000 0000
ss:FFFE 3206
ss:FFFC 48AE
ss:FFFA 000D
F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

Subtraction of 8-bit numbers:-

- Program Code

```
DATA SEGMENT

      NUM1 DB 55H

      NUM2 DB 11H

DATA ENDS

CODE SEGMENT

      ASSUME CS: CODE, DS: DATA

START:

      MOV AX, DATA

      MOV DS, AX

      MOV AL, NUM1

      SUB AL, NUM2

      INT 3
CODE ENDS  END
START
```

- Output

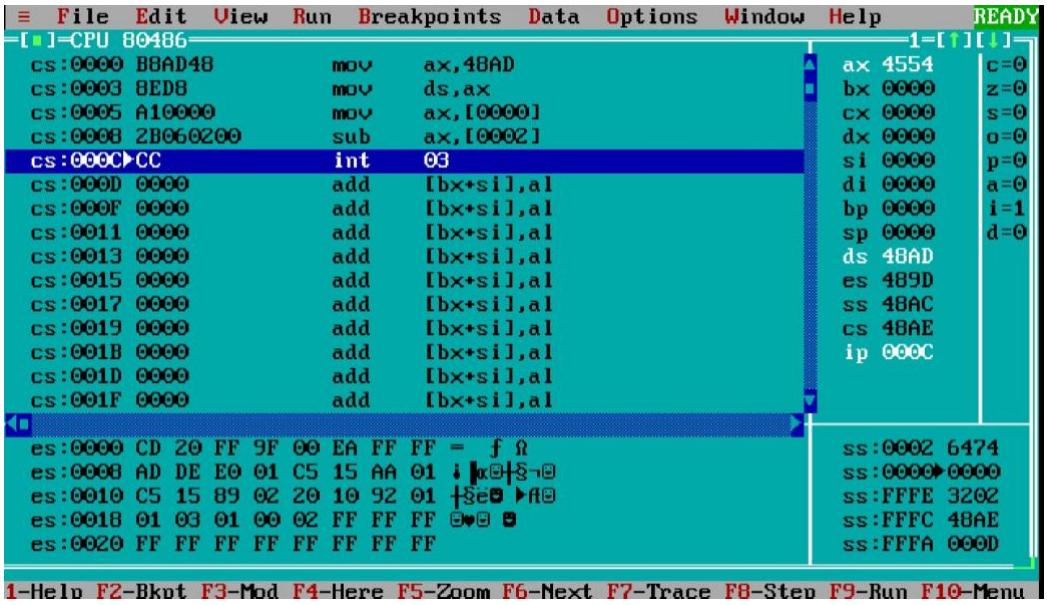


Subtraction of 16-bit numbers:-

- Program Code

```
DATA SEGMENT

        NUM1 DW 5555H
        NUM2 DW 1001H

DATA ENDS       CODE SEGMENT
        ASSUME CS: CODE, DS: DATA

    START:
        MOV AX, DATA

MOV DS, AX

        MOV AX, NUM1

SUB AX, NUM2

        INT 3

CODE ENDS       END START
```

- Output


File Edit View Run Breakpoints Data Options Window Help READY
CPU 80486
cs:0000 B8AD48 mov ax,48AD
cs:0003 8ED8 mov ds,ax
cs:0005 A10000 mov ax,[0000]
cs:0008 2B060200 sub ax,[0002]
cs:000C CC int 03
cs:000D 0000 add [bx+si],al
cs:000F 0000 add [bx+si],al
cs:0011 0000 add [bx+si],al
cs:0013 0000 add [bx+si],al
cs:0015 0000 add [bx+si],al
cs:0017 0000 add [bx+si],al
cs:0019 0000 add [bx+si],al
cs:001B 0000 add [bx+si],al
cs:001D 0000 add [bx+si],al
cs:001F 0000 add [bx+si],al
es:0000 CD 20 FF 9F 00 EA FF FF
es:0008 AD DE E0 01 C5 15 AA 01
es:0010 C5 15 89 02 20 10 92 01
es:0018 01 03 01 00 02 FF FF FF
es:0020 FF FF FF FF FF FF FF FF
ax 4554
bx 0000
cx 0000
dx 0000
si 0000
di 0000
bp 0000
sp 0000
ds 48AD
es 489D
ss 48AC
cs 48AE
ip 000C
c=0
z=0
s=0
o=0
p=0
a=0
i=1
d=0
ss:0002 6474
ss:0000 0000
ss:FFFE 3202
ss:FFFC 48AE
ss:FFFA 000D
1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

- CONCLUSION :-

In this practical we learned how to perform addition and subtraction operations on 8-bit and 16-bit numbers in assembly language programming.

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: MIC	Subject Code:22415
Semester:4th Semester	Course: Computer Engineering
Laboratory No: L004B	Name of Subject Teacher: Pragati mali
Name of Student: Siddharth.P.Shah	Roll Id: 22203A0041

Experiment No:	**4**
Title of Experiment	**Write an assembly language program to perform Multiplication and signed Multiplication of two 8- and 16-bit numbers**

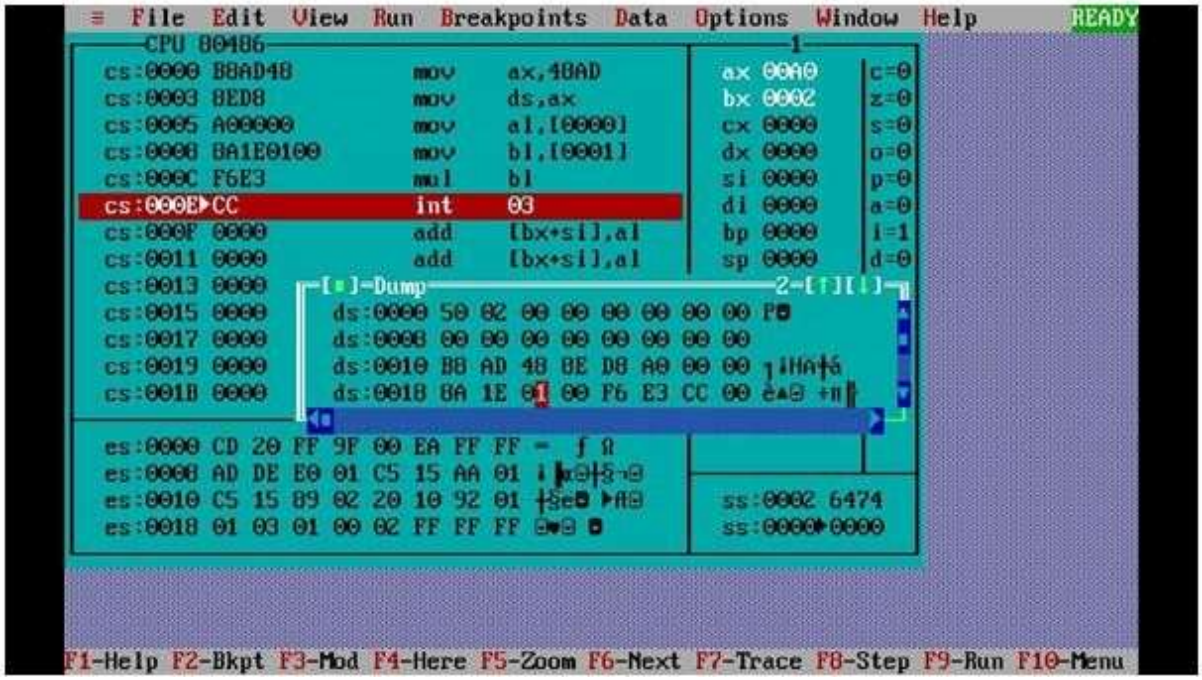
Aim: Write an assembly language program to perform Multiplication and signed Multiplication of two 8- and 16-bit numbers

Software Used : TASM & EMU8086

Write an assembly language program to calculate 8-bit multiplication.

```
DATA SEGMENT
   NUM1 DB 10H
  NUM2 DB 05H
DATA ENDS

CODE SEGMENT ASSUME CS: CODE  , DS:DATA
START:
   MOV AX,DATA
   MOV DS,AX
   MOV AL,NUM1
   MUL AL,NUM2
   INT 3
  CODE ENDS
END START
```

Output:

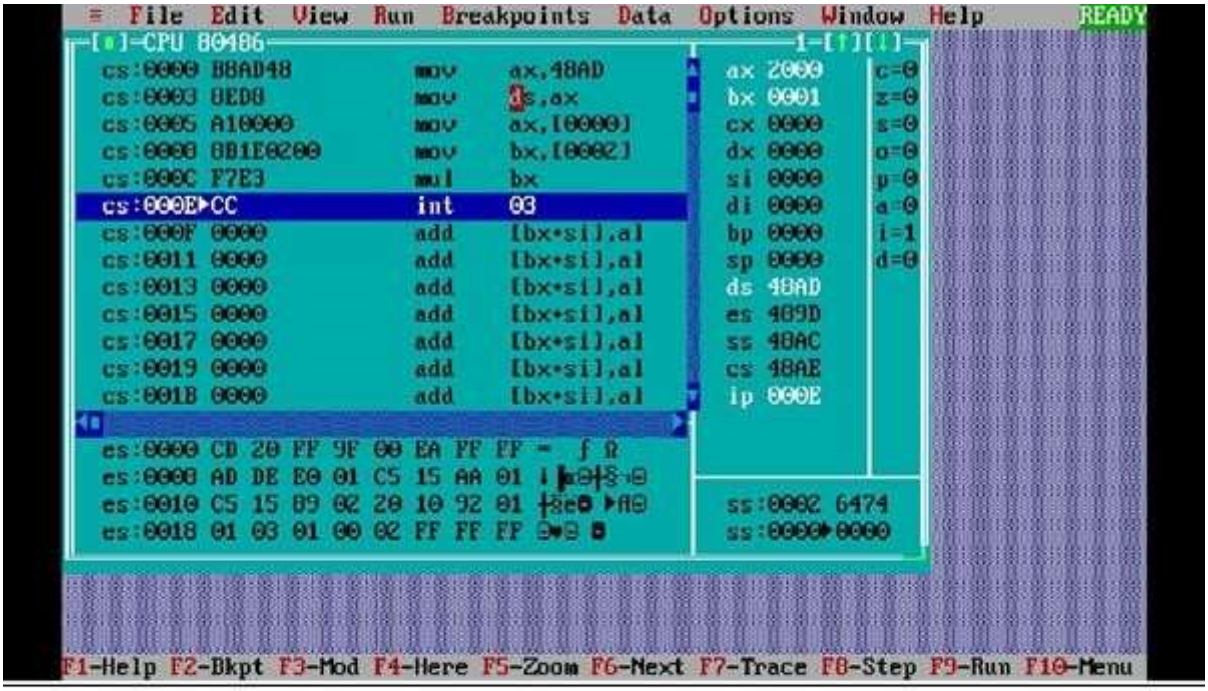


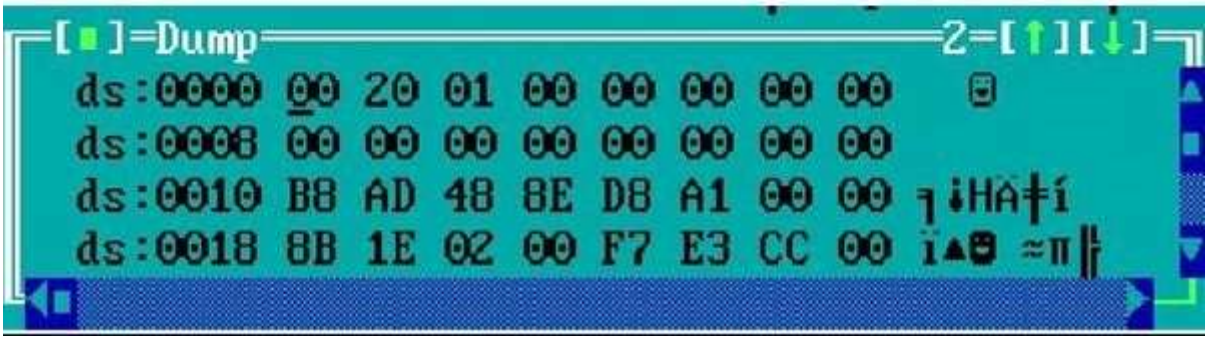
Write an assembly language program to calculate 16-bit multiplication.

Code :

```
DATA SEGMENT

    NUM1 DW 2000H

   NUM2 DW 0001H

DATA ENDS

CODE SEGMENT ASSUME CS:CODE,DS:DATA START:

MOV AX,DATA

MOV DS,AX

MOV AX,NUM1

MOV BX,NUM2

MUL BX

INT 3

CODE ENDS

END START
```

Output:



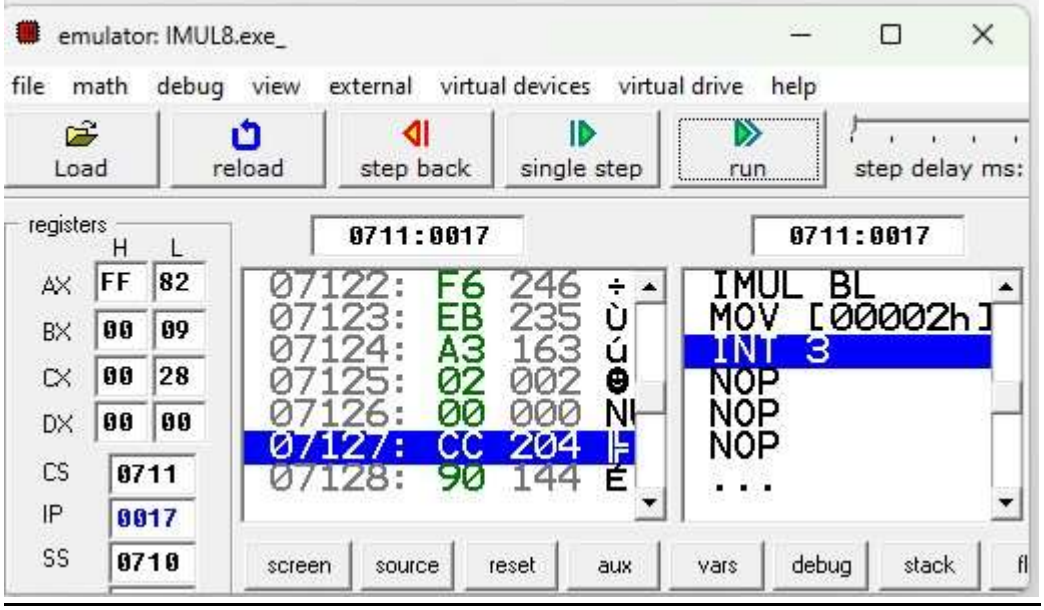


Write an Assembly Language Program (ALP) to perform Signed Multiplication for two given 8-bit numbers.

```
DATA SEGMENT
 A DB 0F2H
B DB 09H  C
DW ?
DATA ENDS
CODE SEGMENT
   ASSUME CS:CODE,DS:DATA
   START:
   MOV AX,DATA
   MOV DS,AX
   MOV AX,0000H
   MOV BX,0000H
   MOV AL,A
   MOV BL,B
   IMUL BL
   MOV C,AX
   INT 03H
   CODE ENDS
END START
```

Output:

```
DATA SEGMENT
  A DB 0F2H
  B DB 09H
  C DW ?
DATA ENDS
CODE SEGMENT
      ASSUME CS:CODE,DS:DATA
      START:
      MOV AX,DATA
      MOV DS,AX
      MOV AX,0000H
      MOV BX,0000H
      MOV AL,A
      MOV BL,B
      IMUL BL
      MOV C,AX
      INT 03H
      CODE ENDS
END START
```



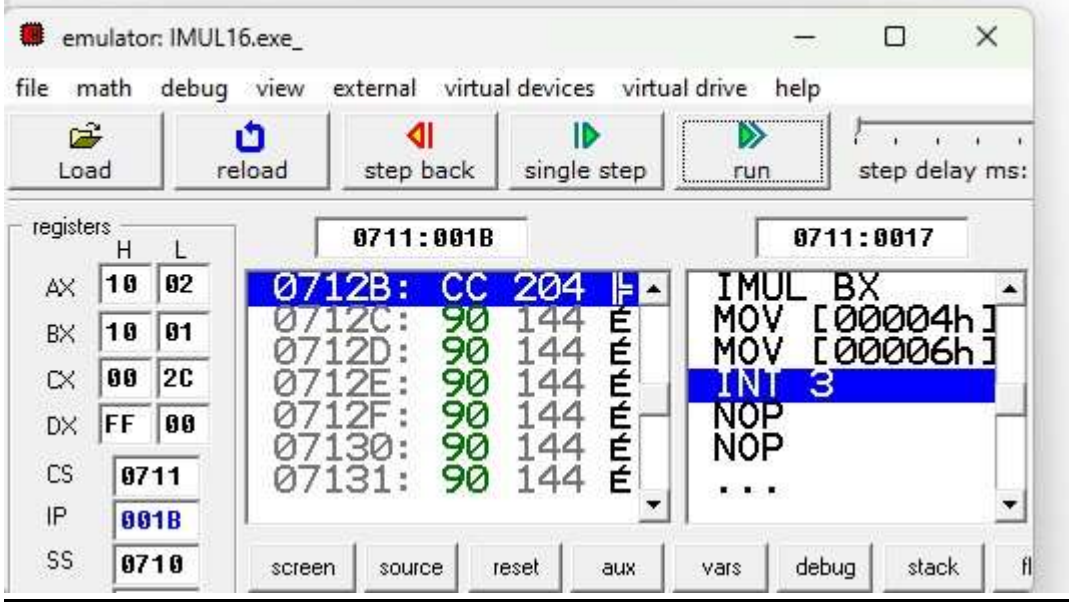
Write an Assembly Language Program (ALP) to perform Signed Multiplication for two given 16-bit numbers.

DATA SEGMENT

```
  A DW 0F002H
B DW 1001H   C
DW ?
DATA ENDS
CODE SEGMENT
   ASSUME CS:CODE,DS:DATA
   START:
   MOV AX,DATA
   MOV DS,AX
   MOV AX,0000H
   MOV BX,0000H
   MOV AX,A
   MOV BX,B
   IMUL BX
   MOV WORD PTR C,AX
   MOV WORD PTR C+2,DX
   INT 03H
   CODE ENDS
END START
```

Output:

```
DATA SEGMENT
A  DW  0F002H
B  DW  1001H
C  DW  ?
DATA ENDS
CODE  SEGMENT
      ASSUME CS:CODE,DS:DATA
      START:
      MOV  AX,DATA
      MOV  DS,AX
      MOV  AX,0000H
      MOV  BX,0000H
      MOV  AX,A
      MOV  BX,B
      IMUL  BX
      MOV  WORD PTR C,AX
      MOV  WORD PTR C+2,DX
      INT  03H
      CODE  ENDS
END  START
```



Conclusion:

From This pratical we get to know how to perform 8-16 bit Multiplication and Signed Multiplication.

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: Microprocessor	Subject Code:22415
Semester:4th Semester	Course: Computer Engineering
Laboratory No: L004B	Name of Subject Teacher: Pragati Mali
Name of Student: Siddharth Shah	Roll Id: 22203A0041

Experiment No:	**5**
Title of Experiment	**Write an assembly language program to perform Divison and signed Division of two 8- and 16-bit numbers**

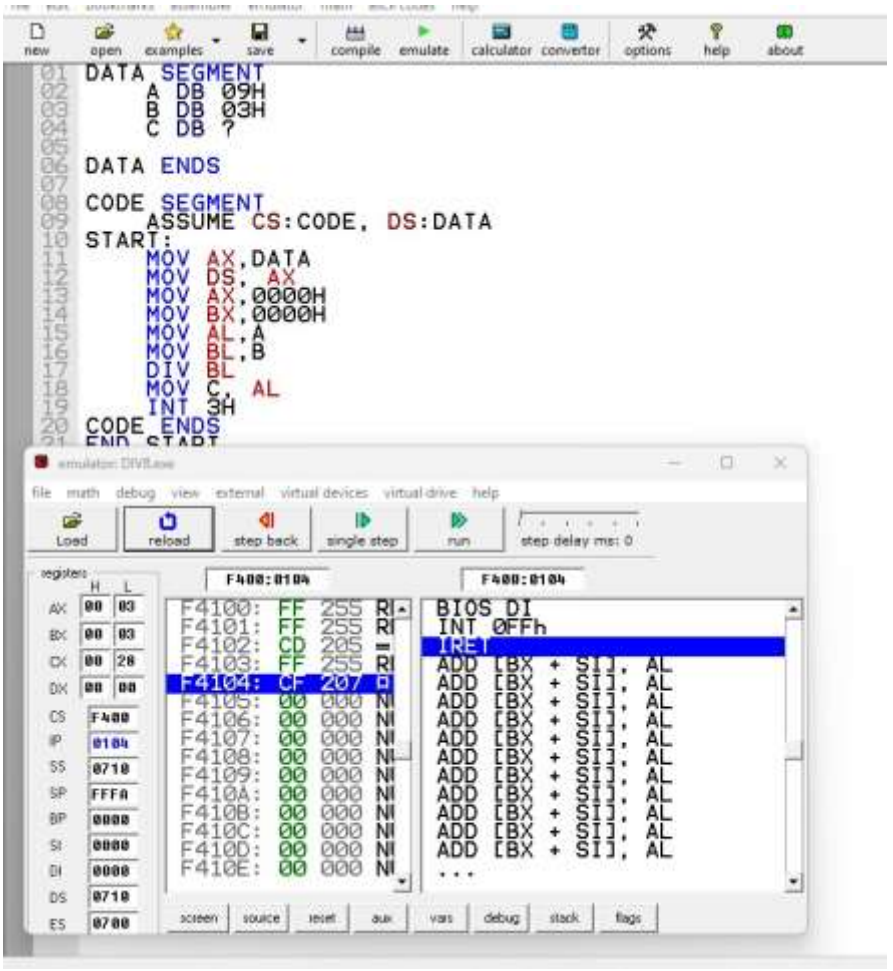
AIM: -Write an assembly language program to perform Divison and signed Division of two 8- and 16-bit numbers

Used Software:

Emu8086

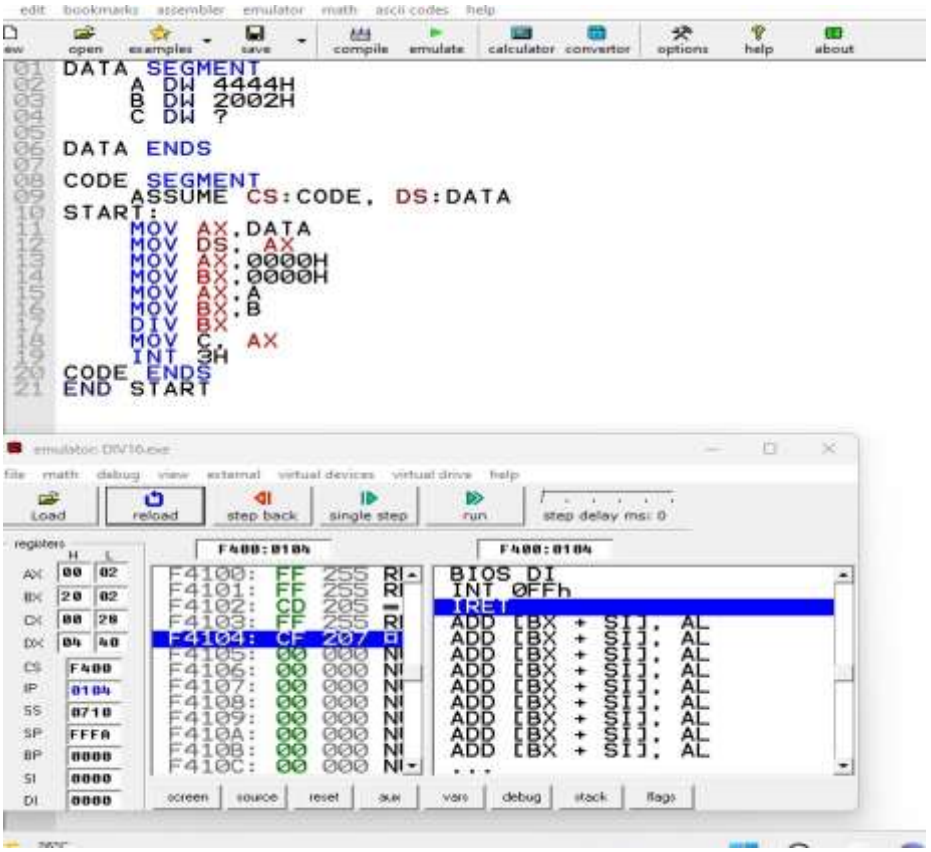
Write an Assembly Language Program (ALP) to Division two given 8-bit numbers.

```
DATA SEGMENT
   A DB 09H B
   DB 03H C
   DB ?

DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE, DS:DATA START:
   MOV AX,DATA
   MOV DS, AX
   MOV AX,0000H
   MOV BX,0000H
   MOV AL,A
   MOV BL,B
   DIV BL
   MOV C, AL
   INT 3H
   CODE
   ENDS END
   START
```



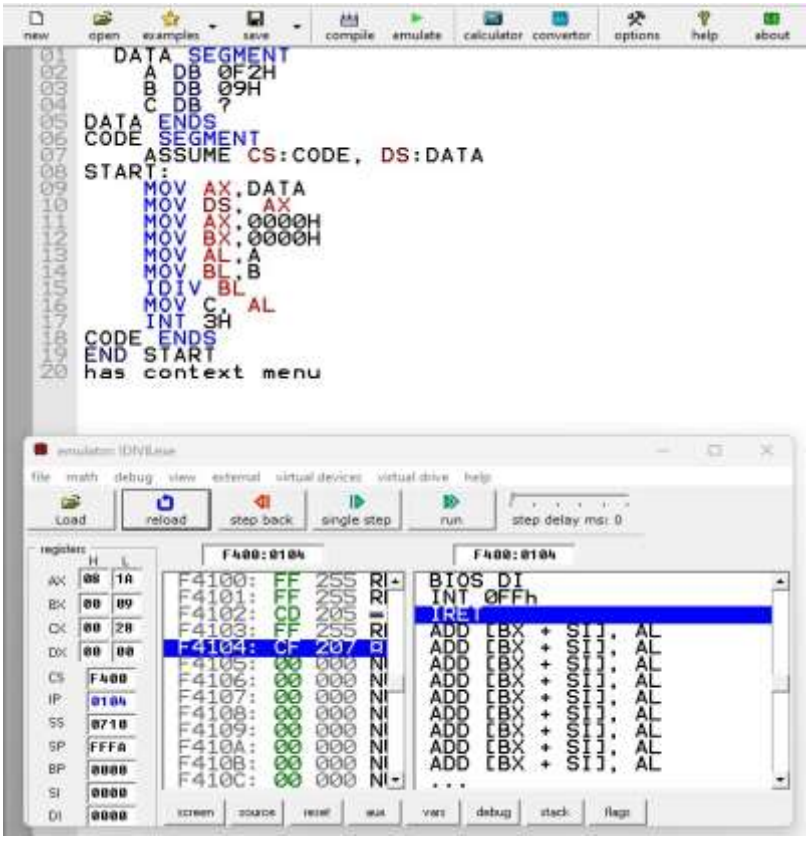
Write an Assembly Language Program (ALP) to Division two given16-bit numbers.

```
DATA SEGMENT
   A DW 4444H
B DW 2002H
C DW ?

DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE, DS:DATA START:
   MOV AX,DATA
   MOV DS, AX
   MOV AX,0000H
   MOV BX,0000H
   MOV AX,A
   MOV BX,B
   DIV BX
   MOV C, AX
   INT 3H
CODE ENDS
END START
```



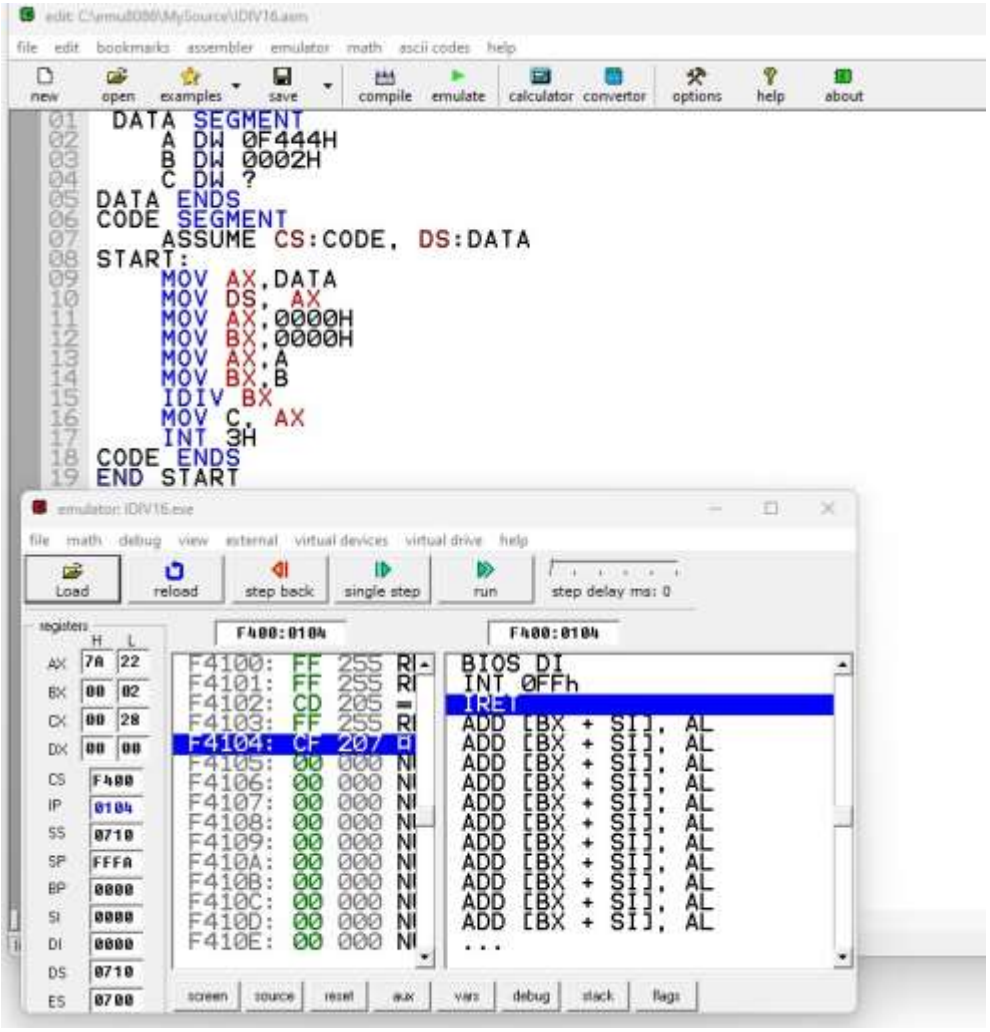
Write an Assembly Language Program (ALP) to Signed Division two given 8-bit numbers.

```
DATA SEGMENT
   A DB 0F2H B
   DB 09H C DB ?
DATA ENDS
CODE SEGMENT
   ASSUME CS:CODE, DS:DATA
START:
   MOV AX,DATA MOV
   DS, AX
   MOV AX,0000H
   MOV BX,0000H
   MOV AL,A
   MOV BL,B
   IDIV BL
   MOV C, AL
   INT 3H
CODE ENDS
END START
```



Write an Assembly Language Program (ALP) to Signed Divison two given 16-bit numbers.

```
DATA SEGMENT
   A DW 0F444H
   B DW 0002H C
   DW ?
DATA ENDS
CODE SEGMENT
   ASSUME CS:CODE, DS:DATA
START:
   MOV AX,DATA MOV
   DS, AX
   MOV AX,0000H
   MOV BX,0000H
   MOV AX,A
   MOV BX,B
   IDIV BX
   MOV C, AX
   INT 3H
CODE ENDS
END START
```



Conclusion: From This pratical we get to know how to perform 816 bit Division and Signed Divison.

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: MIC	Subject Code:22415
Semester:4th Semester	Course: Computer Engineering
Laboratory No: L004B	Name of Subject Teacher: Pragati Mali
Name of Student: Siddharth Shah	Roll Id: 22203A0041

Experiment No:	**6**
Title of Experiment	**Write an ALP to Add, Sub, Mul, Div two BCD numbers**

Aim: Write an assembly language program to perform BCD Addition, Subtraction, Multiplication & Division of 8-bit numbers.

Software Used:

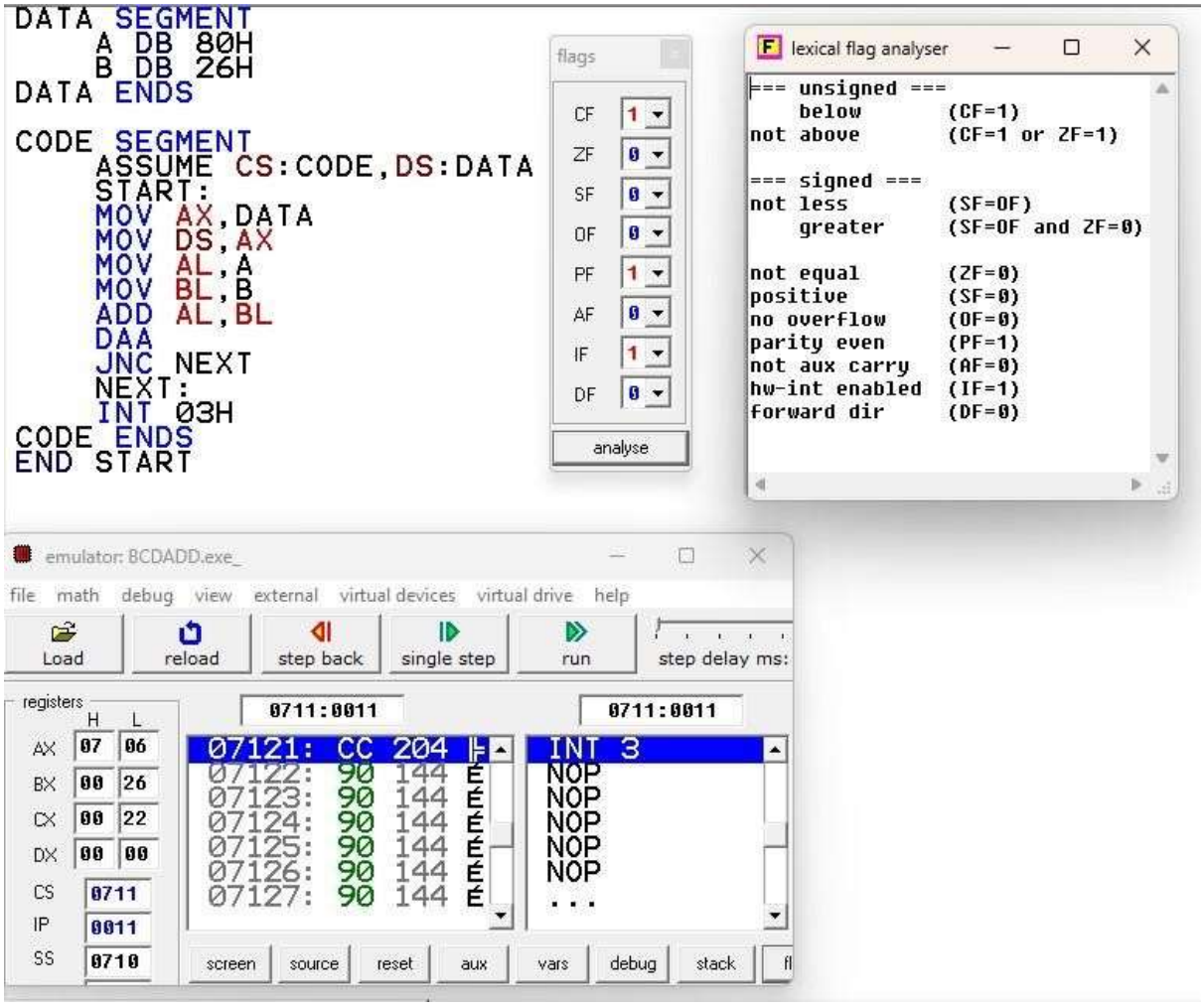
EMU8086

1. Write an assembly language program to Perform 8-bit BCD Addition.

CODE:

```
DATA SEGMENT
A DB 80H
B DB 26H
   RES_LSB DB 0H
   RES_MSB DB 0H
DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE, DS:DATA
   START:
   MOV AX, DATA
   MOV DS, AX
   MOV AL, A
   MOV BL, B
   ADD AL, BL
   DAA
   JNC NEXT
INC RES_MSB
NEXT:
   MOV RES_LSB , AL
   INT 03H CODE
ENDS
END START
```

Output:

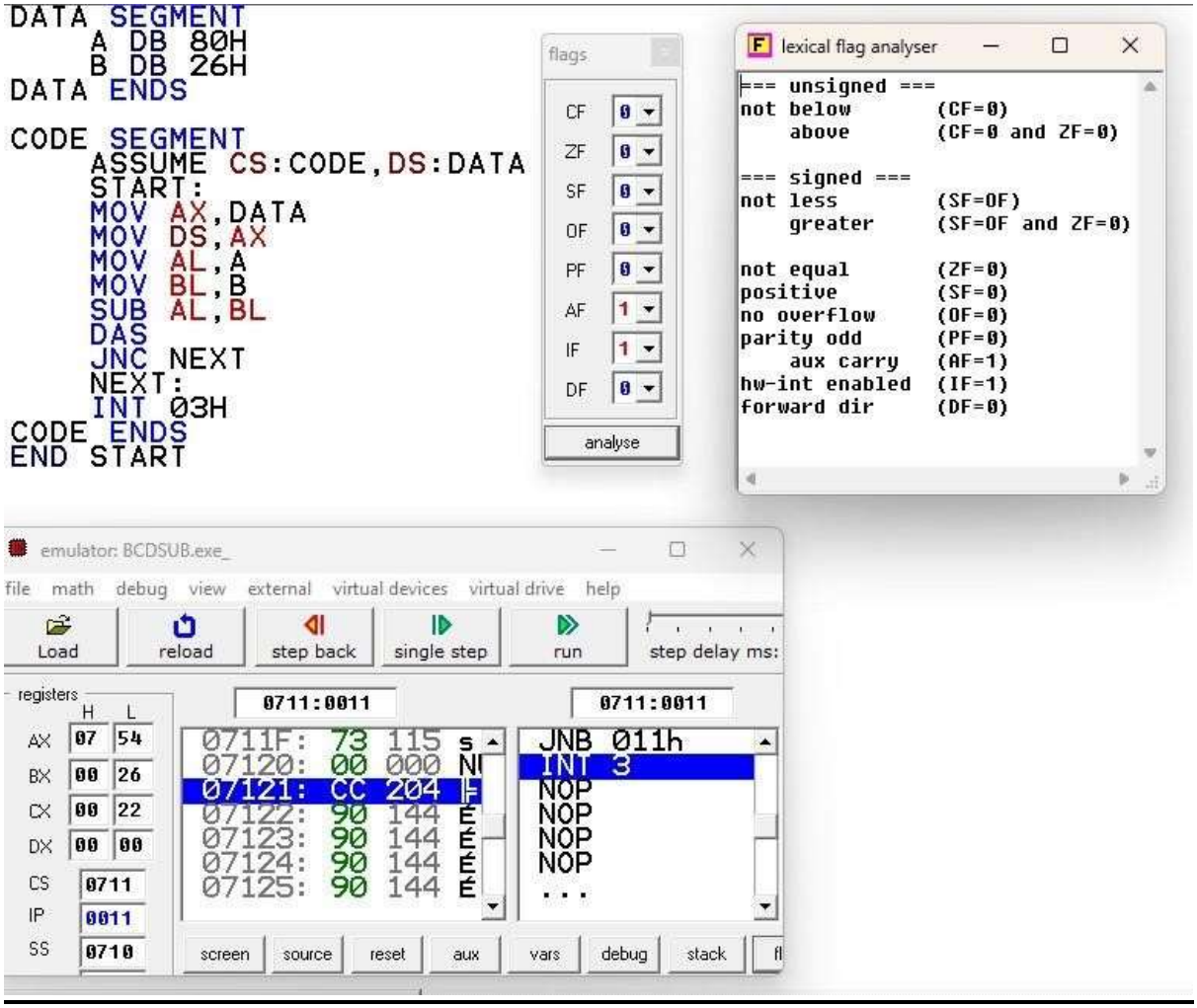

DATA SEGMENT
A DB 80H
B DB 26H
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV AL,A
MOV BL,B
ADD AL,BL
DAA
JNC NEXT
NEXT:
INT 03H
CODE ENDS
END START
flags
CF 1
ZF 0
SF 0
OF 0
PF 1
AF 0
IF 1
DF 0
analyse
lexical flag analyser
=== unsigned ===
below (CF=1)
not above (CF=1 or ZF=1)
=== signed ===
not less (SF=OF)
greater (SF=OF and ZF=0)
not equal (ZF=0)
positive (SF=0)
no overflow (OF=0)
parity even (PF=1)
not aux carry (AF=0)
hw-int enabled (IF=1)
forward dir (DF=0)
emulator: BCDADD.exe_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms:
registers
H L
AX 07 06
BX 00 26
CX 00 22
DX 00 00
CS 0711
IP 0011
SS 0710
0711:0011
07121: CC 204 INT 3
07122: 90 144 NOP
07123: 90 144 NOP
07124: 90 144 NOP
07125: 90 144 NOP
07126: 90 144 NOP
07127: 90 144 ...
0711:0011
screen source reset aux vars debug stack

2. Write an assembly language program to Perfrom 8-bit BCD Subtraction.

CODE:

```
DATA SEGMENT
A DB 80H
   B DB 26H
   RES_LSB DB 0H
DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE, DS:DATA
   START:
   MOV AX, DATA
   MOV DS, AX
   MOV AL, A
   MOV BL, B
   SUB AL, BL
   DAS
   MOV RES_LSB , AL
   INT 03H CODE
ENDS
END START
```

Output:

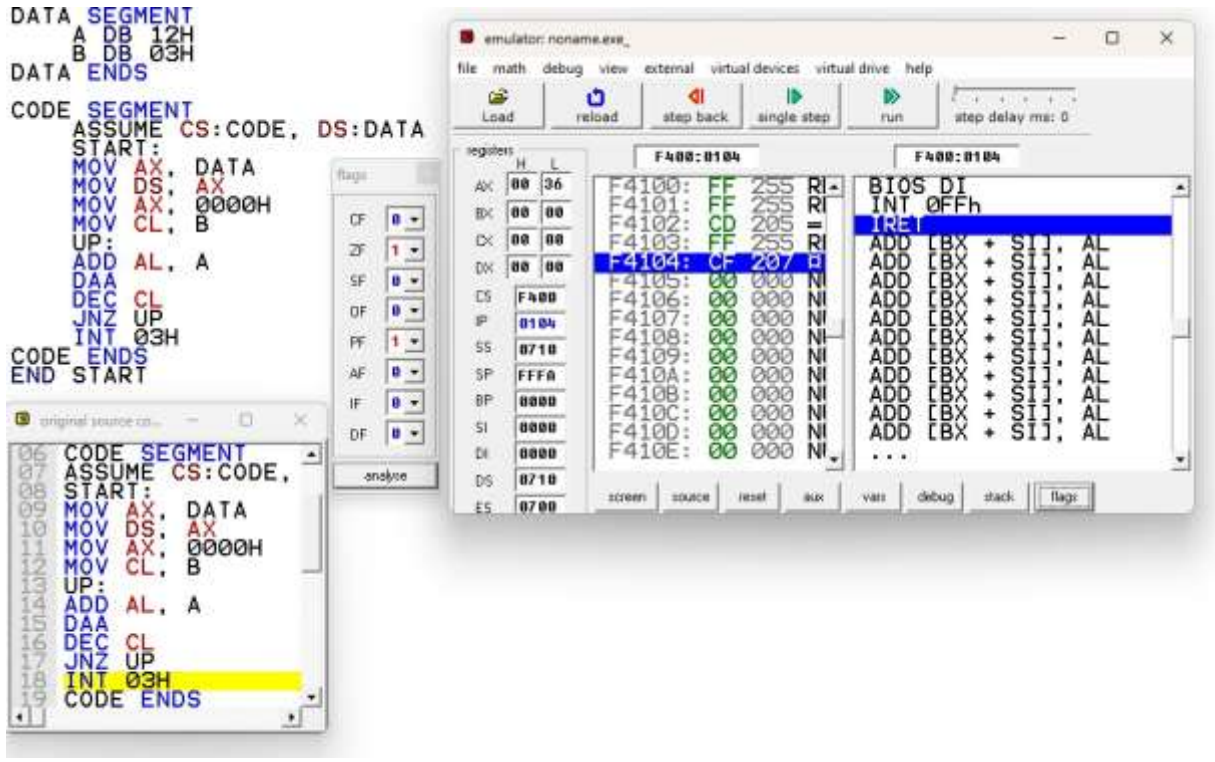

DATA SEGMENT
A DB 80H
B DB 26H
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV AL,A
MOV BL,B
SUB AL,BL
DAS
JNC NEXT
NEXT:
INT 03H
CODE ENDS
END START
flags
CF 0
ZF 0
SF 0
OF 0
PF 0
AF 1
IF 1
DF 0
analyse
lexical flag analyser
=== unsigned ===
not below (CF=0)
above (CF=0 and ZF=0)
=== signed ===
not less (SF=OF)
greater (SF=OF and ZF=0)
not equal (ZF=0)
positive (SF=0)
no overflow (OF=0)
parity odd (PF=0)
aux carry (AF=1)
hw-int enabled (IF=1)
forward dir (DF=0)
emulator: BCDSUB.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms:
registers
H L
AX 07 54
BX 00 26
CX 00 22
DX 00 00
CS 0711
IP 0011
SS 0710
0711:0011
0711F: 73 115 s
07120: 00 000 NU
07121: CC 204 F
07122: 90 144 E
07123: 90 144 E
07124: 90 144 E
07125: 90 144 E
0711:0011
JNB 011h
INT 3
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack

3. Write an assembly language program to Perform 8-bit BCD Multiplication.

CODE:

```
DATA SEGMENT
A DB 12H
B DB 03H
DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE, DS:DATA
   START:
   MOV AX, DATA
   MOV DS, AX
   MOV AX, 0000H
   MOV CL, B
   UP:
   ADD AL, A
   DAA
   DEC CL
   JNZ UP
   INT 03H
CODE ENDS
END START
```

Output:


DATA SEGMENT
A DB 12H
B DB 03H
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
MOV AX, 0000H
MOV CL, B
UP:
ADD AL, A
DAA
DEC CL
JNZ UP
INT 03H
CODE ENDS
END START
emulator: noname.exe
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers
F400:0104
BIOS DI
INT 0FFh
IRET
screen source reset aux vars debug stack flags

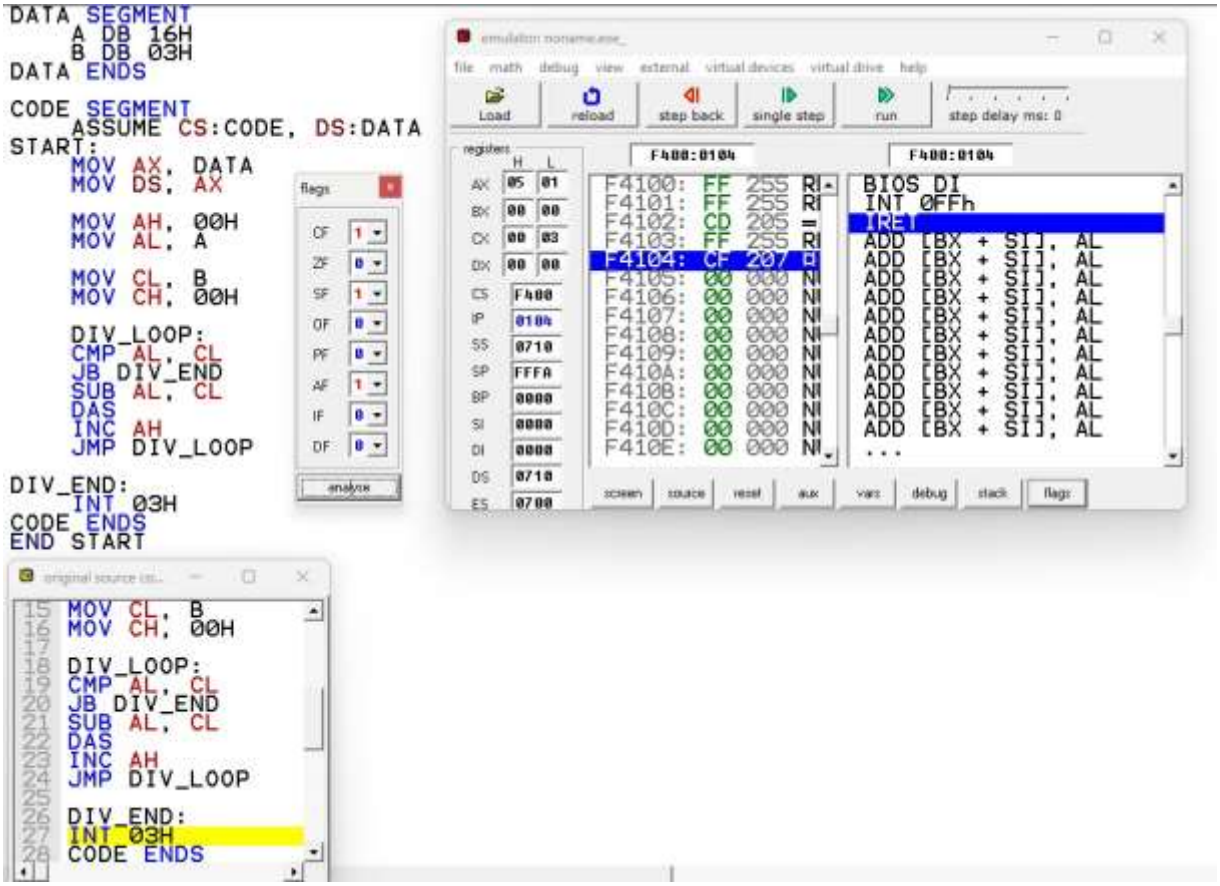
4. Write an assembly language program to Perform 8-bit BCD Division.

CODE:

```
DATA SEGMENT
A DB 16H
B DB 03H     DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE, DS:DATA START:
   MOV AX, DATA
   MOV DS, AX

   MOV AH, 00H
   MOV AL, A

   MOV CL, B
   MOV CH, 00H

   DIV_LOOP:
   CMP AL, CL
   JB DIV_END
   SUB AL, CL
   DAS
   INC AH
   JMP DIV_LOOP

DIV_END:
```

INT 03H
CODE ENDS
END START

Output:



<u>Conclusion</u>: From this practical we get to know how to perform 8 bit BCD Addition, Subtraction ,Multiplication & Division.

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: Microprocessor	**Subject Code**:22415
Semester: 04	**Course:** CO4I-A
Laboratory No: L004B	**Name of Subject Teacher:** Pragati Mali
Name of Student: Siddharth Shah	**Roll ID: 22203A0041**

Experiment No:	**7**
Aim	a. Write an ALP to perform block transfer data using string instruction. b. Write an ALP to perform block transfer data without using string instruction.

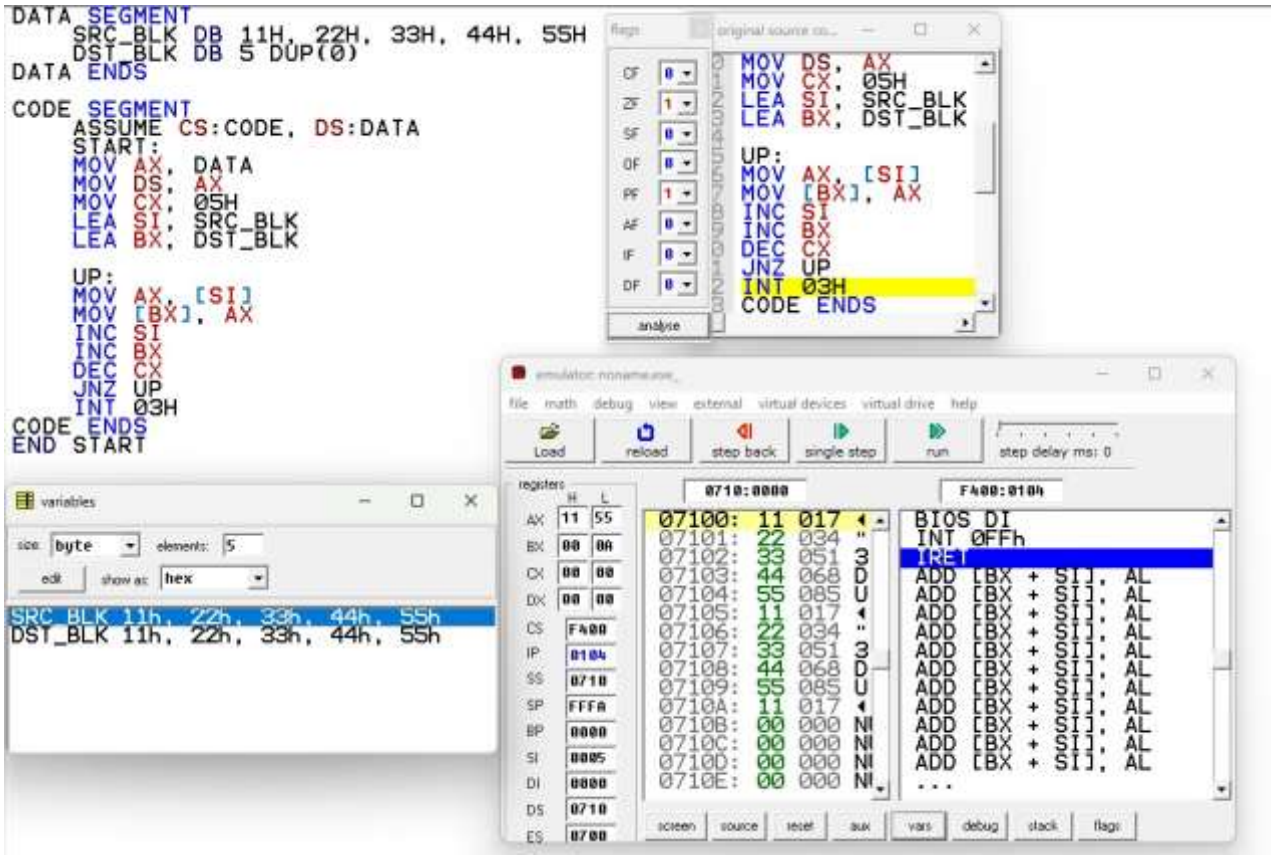
1. Block Transfer data without using string instruction.

CODE:

```
DATA SEGMENT
SRC_BLK DB 11H, 22H, 33H, 44H, 55H
DST_BLK DB 5 DUP(0)
DATA ENDS

CODE SEGMENT ASSUME
CS:CODE, DS:DATA START:
MOV AX, DATA
```

```
MOV DS, AX
MOV CX, 05H
LEA SI, SRC_BLK
LEA BX, DST_BLK

UP:
MOV AX, [SI]
MOV [BX], AX
INC SI
INC BX
DEC CX
JNZ UP
INT 03H
CODE ENDS
END START
```

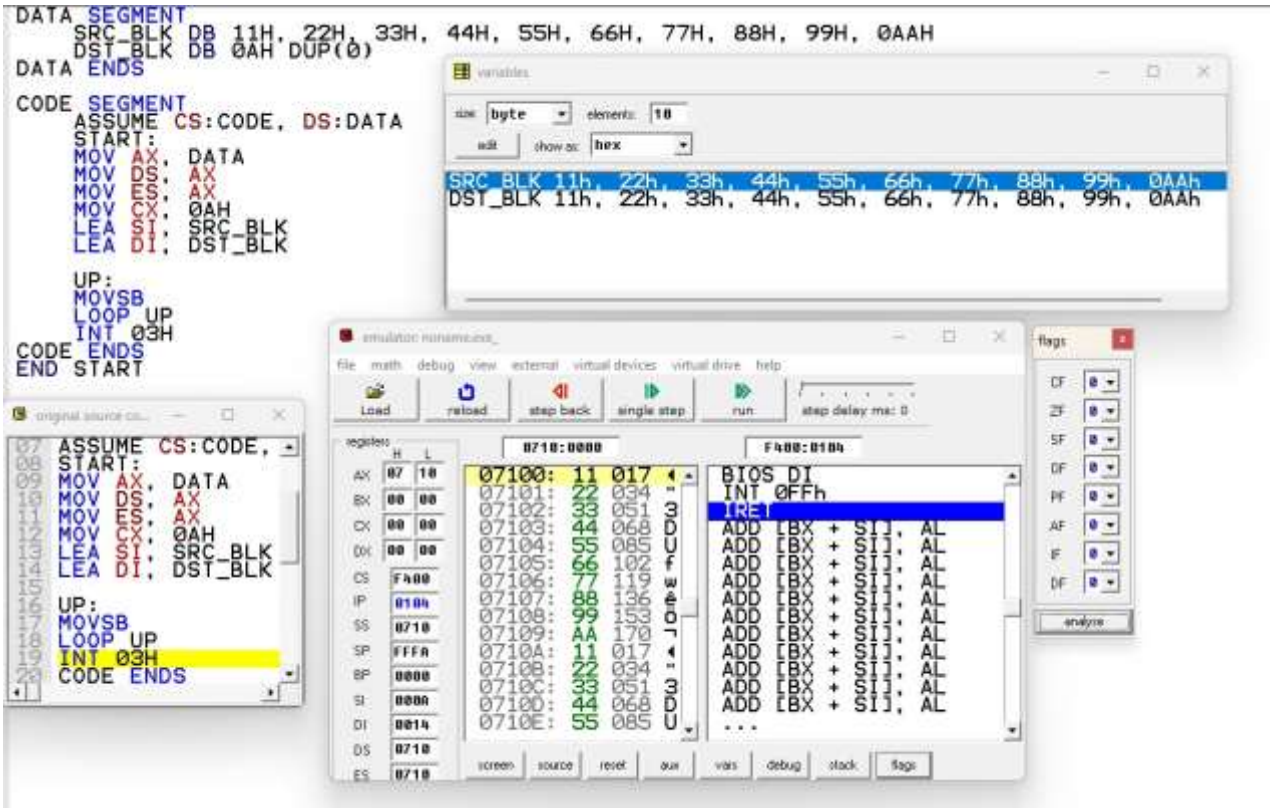
Output:


DATA SEGMENT
SRC_BLK DB 11H, 22H, 33H, 44H, 55H
DST_BLK DB 5 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
MOV CX, 05H
LEA SI, SRC_BLK
LEA BX, DST_BLK
UP:
MOV AX, [SI]
MOV [BX], AX
INC SI
INC BX
DEC CX
JNZ UP
INT 03H
CODE ENDS
END START
variables
SRC_BLK 11h, 22h, 33h, 44h, 55h
DST_BLK 11h, 22h, 33h, 44h, 55h
MOV DS, AX
MOV CX, 05H
LEA SI, SRC_BLK
LEA BX, DST_BLK
UP:
MOV AX, [SI]
MOV [BX], AX
INC SI
INC BX
DEC CX
JNZ UP
INT 03H
CODE ENDS
Load
reload
step back
single step
run
BIOS_DI
INT 0FFh
IRET

2. Block Transfer data with string instruction.

```
 SRC_BLK DB 11H, 22H, 33H, 44H, 55H, 66H, 77H, 88H,
99H, 0AAH
  DST_BLK DB 0AH DUP(0)
DATA ENDS

CODE SEGMENT
  ASSUME CS:CODE, DS:DATA    START:
  MOV AX, DATA
  MOV DS, AX
  MOV ES, AX
  MOV CX, 0AH
  LEA SI, SRC_BLK
  LEA DI, DST_BLK

  UP:
  MOVSB
  LOOP UP
  INT 03H
CODE ENDS END START
```

Output:



Conclusion:

With this practical we learned to perform block data transfer with and without using String Instruction

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: Microprocessor	**Subject Code**:22415
Semester: 04	**Course:** CO4IA
Laboratory No: L004A	**Name of Subject Teacher:** Prof. Pragati Mali
Name of Student: Siddharth Shah	**Roll ID:** 22203A0041

Experiment No:	8
Aim	Implement Loop in Assembly language Program. (A) Write an ALP to find sum of series of Hexadecimal Numbers. (B) Write an ALP to find sum of series of BCD Numbers.

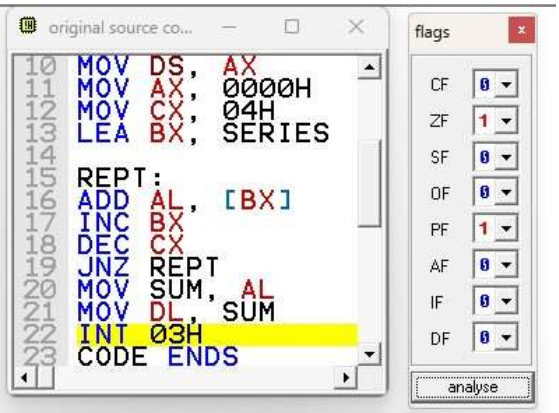
1. Write an ALP to find sum of series of Hexadecimal Numbers.

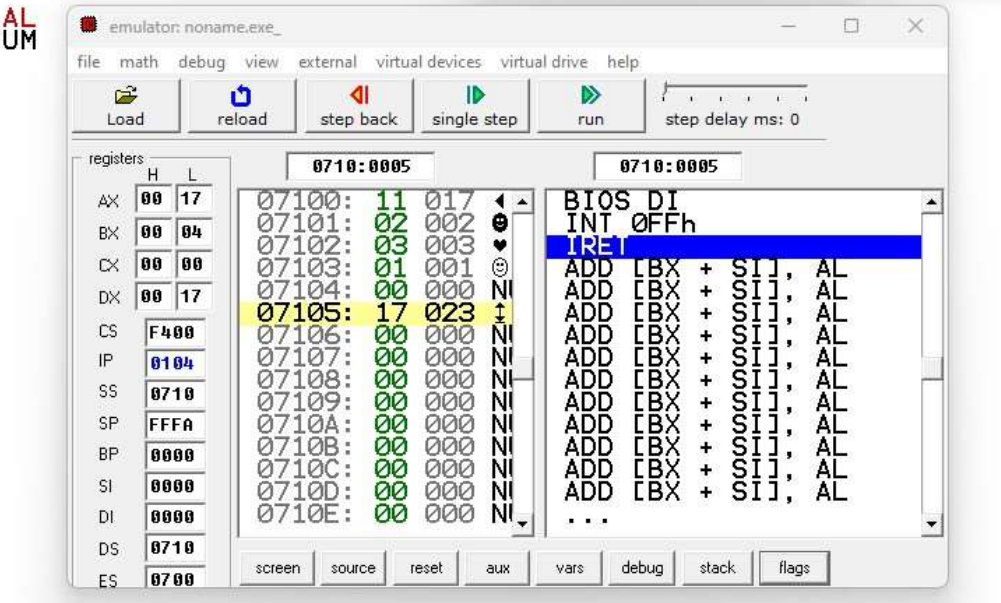
```
DATA SEGMENT
  SERIES DB 11H, 02H, 03H, 01H,
00H
  SUM DB 00H
DATA ENDS

CODE SEGMENT
  ASSUME CS:CODE, DS:DATA
START:
  MOV AX, DATA
  MOV DS, AX
```

```
    MOV AX, 0000H
    MOV CX, 04H
    LEA BX, SERIES

    REPT:
    ADD AL, [BX]
    INC BX
    DEC CX
    JNZ REPT
    MOV SUM, AL
    MOV DL, SUM
    INT 03H
CODE ENDS  END
START
```

Output:

```
DATA SEGMENT
    SERIES DB 11H, 02H, 03H, 01H, 00H
    SUM DB 00H
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
    START:
    MOV AX, DATA
    MOV DS, AX
    MOV AX, 0000H
    MOV CX, 04H
    LEA BX, SERIES

    REPT:
    ADD AL, [BX]
    INC BX
    DEC CX
    JNZ REPT
    MOV SUM, AL
    MOV DL, SUM
    INT 03H
CODE ENDS
END START
```



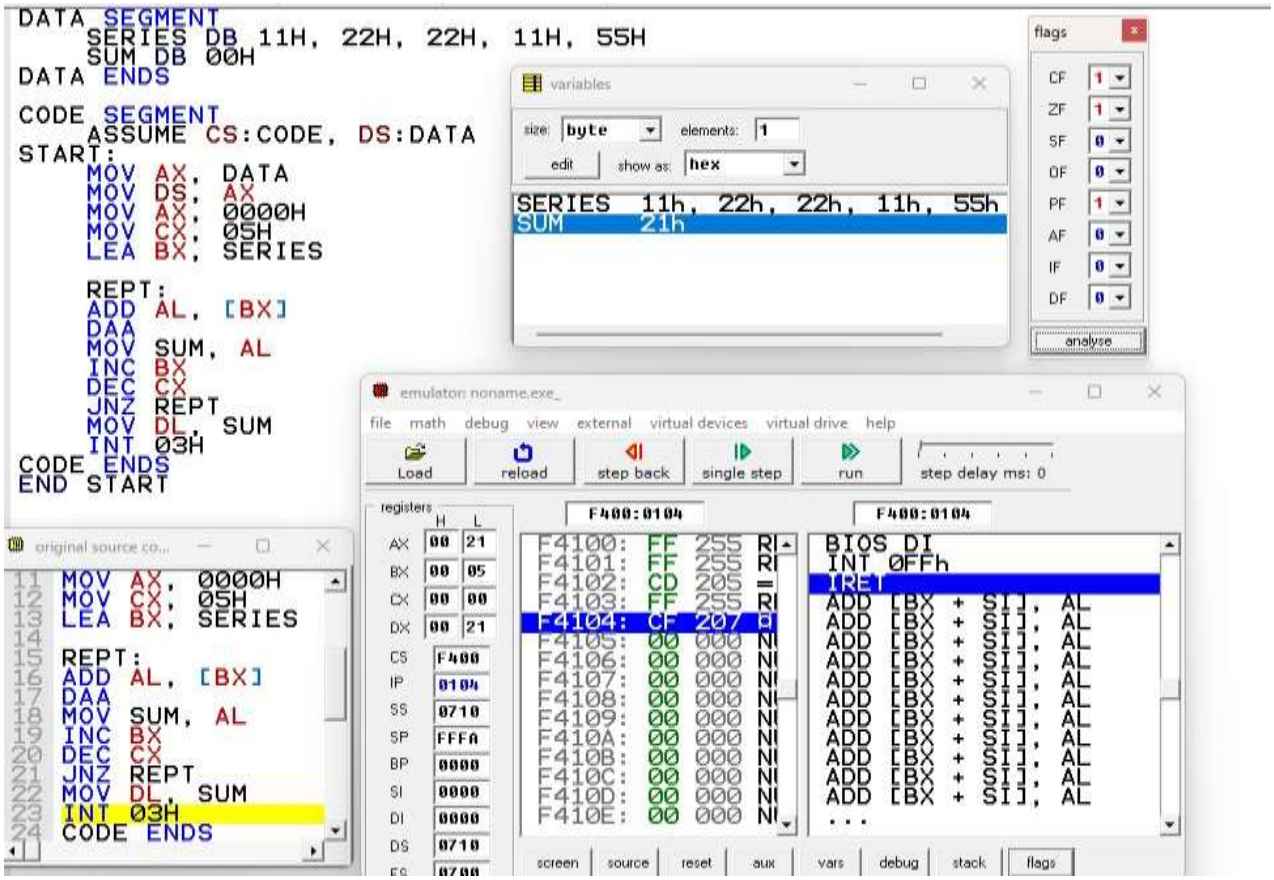


2. Write an ALP to find sum of series of BCD Numbers

```
DATA SEGMENT
   SERIES DB 11H, 22H, 22H, 11H, 55H
   SUM DB 00H
DATA ENDS


CODE SEGMENT
ASSUME CS:CODE,DS:DATA
     START:
     MOV AX, DATA
     MOV DS, AX
    MOV AX, 0000H
    MOV CX, 05H      LEA BX, SERIES      REPT:
    ADD AL, [BX]
    DAA
    MOV SUM, AL
    INC BX
    DEC CX
   JNZ REPT
   MOV DL, SUM
   INT 03H
CODE ENDS
```

END START

Output:



Conclusion:

With this practical we learned to perform sum of series of hexadecimal and BCD numbers.

VP Vidyalankar Polytechnic	**DEPARTMENT OF COMPUTER ENGINEERING**

Subject: Microprocessor	**Subject Code:**22415
Semester:4th Semester	**Course:** Computer Engineering
Laboratory No: L004B	**Name of Subject Teacher:** Prof. Pragati Mali
Name of Student: Siddharth Shah	**Roll Id**: 22203A0041

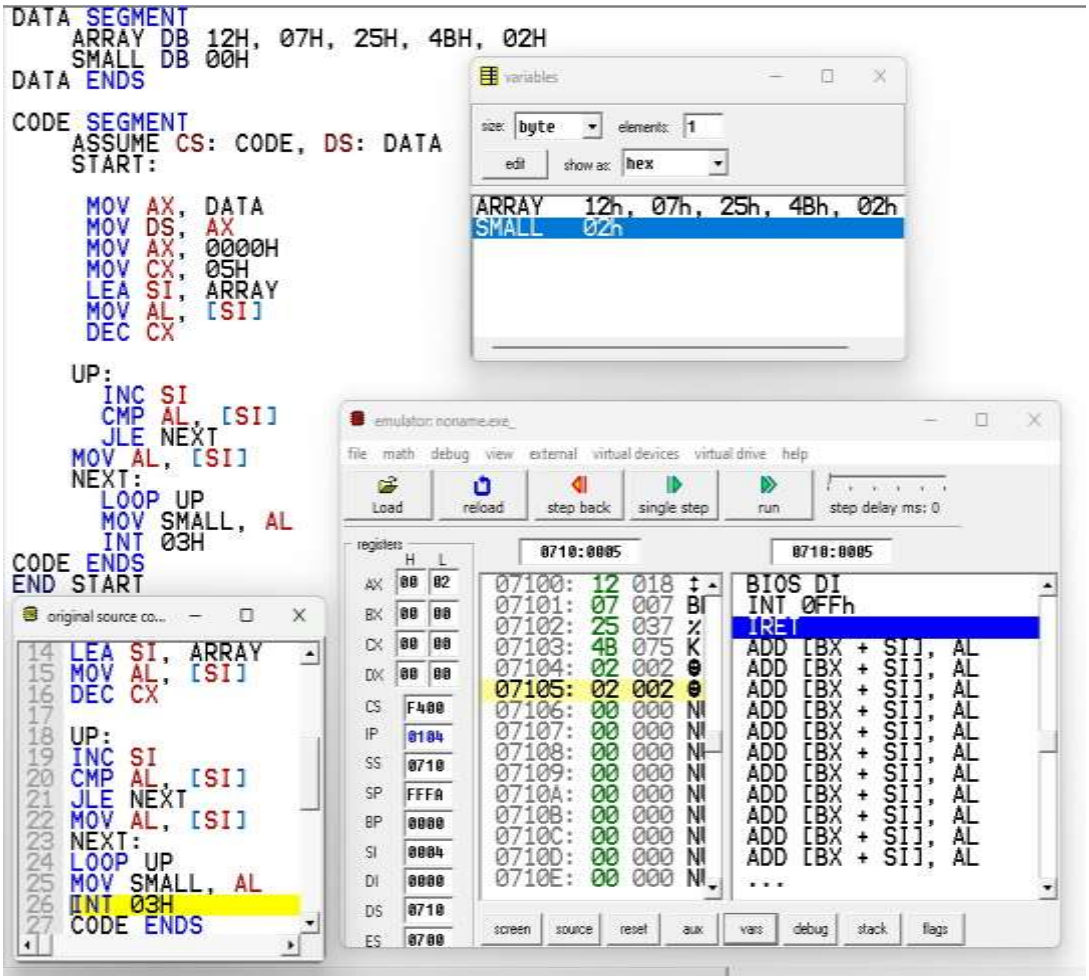
Experiment No:	9
Title of Experiment:	(A) Write an ALP to find the smallest number from Array of n numbers. (B)Write an ALP to find largest number from Array of n numbers

1.Write an ALP to find the smallest number from Array of n numbers.

```
DATA SEGMENT
   ARRAY DB 12H, 07H, 25H, 4BH, 02H
   SMALL DB 00H
DATA ENDS

CODE SEGMENT
   ASSUME CS: CODE, DS: DATA
START:

    MOV AX, DATA
    MOV DS, AX
    MOV AX, 0000H
    MOV CX, 05H
```

```
    LEA SI, ARRAY
    MOV AL, [SI]
    DEC CX

   UP:
     INC SI
     CMP AL, [SI]
     JLE NEXT
MOV AL, [SI]
NEXT:
     LOOP UP
     MOV SMALL, AL
     INT 03H
CODE ENDS
END START
```

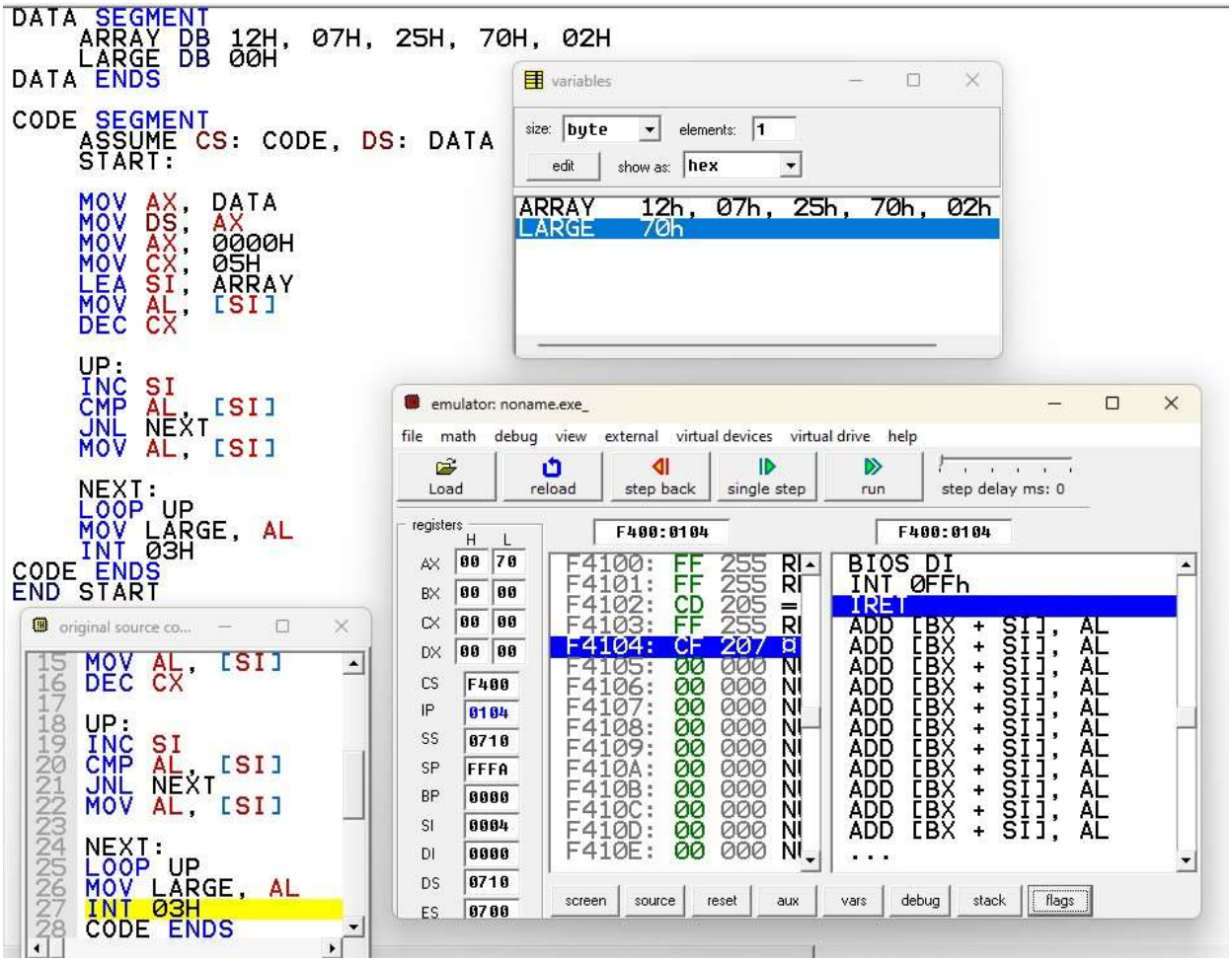
OUTPUT:


DATA SEGMENT
ARRAY DB 12H, 07H, 25H, 4BH, 02H
SMALL DB 00H
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS: DATA
START:
MOV AX, DATA
MOV DS, AX
MOV AX, 0000H
MOV CX, 05H
LEA SI, ARRAY
MOV AL, [SI]
DEC CX
UP:
INC SI
CMP AL, [SI]
JLE NEXT
MOV AL, [SI]
NEXT:
LOOP UP
MOV SMALL, AL
INT 03H
CODE ENDS
END START
variables
size: byte
elements: 1
edit
show as: hex
ARRAY 12h, 07h, 25h, 4Bh, 02h
SMALL 02h
emulator: noname.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
0710:0005
0710:0005
BIOS DI
INT 0FFh
IRET
ADD [BX + SI], AL
screen
source
reset
aux
vars
debug
stack
flags
original source co...
LEA SI, ARRAY
MOV AL, [SI]
DEC CX
UP:
INC SI
CMP AL, [SI]
JLE NEXT
MOV AL, [SI]
NEXT:
LOOP UP
MOV SMALL, AL
INT 03H
CODE ENDS

2.Write an ALP to find the largest number from Array of n numbers.

```
DATA SEGMENT
   ARRAY DB 12H, 07H, 25H, 70H, 02H
   LARGE DB 00H
DATA ENDS

CODE SEGMENT
   ASSUME CS: CODE, DS: DATA
START:

   MOV AX, DATA
   MOV DS, AX
   MOV AX, 0000H
   MOV CX, 05H
   LEA SI, ARRAY
   MOV AL, [SI]
   DEC CX

   UP:
   INC SI
   CMP AL, [SI]
   JNL NEXT
   MOV AL, [SI]

   NEXT:
   LOOP UP
   MOV LARGE, AL
   INT 03H
CODE ENDS
END START
```

OUTPUT



CONCLUSION: -

Different addressing mode of instructions are implemented in above program:

➢ To find the smallest number from Array of n numbers.

➢ To find the largest number from Array of n numbers.

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: Microprocessor	**Subject Code**:22415
Semester: 04	**Course:** CO4I-A
Laboratory No: L004B	**Name of Subject Teacher:** Prof. Pragati Mali
Name of Student: Siddharth Shah	**Roll ID:** 22203A0041

Experiment No:	10
Aim	(A) Write an ALP to arrange numbers in array in ascending order (B) Write an ALP to arrange numbers in array in descending order

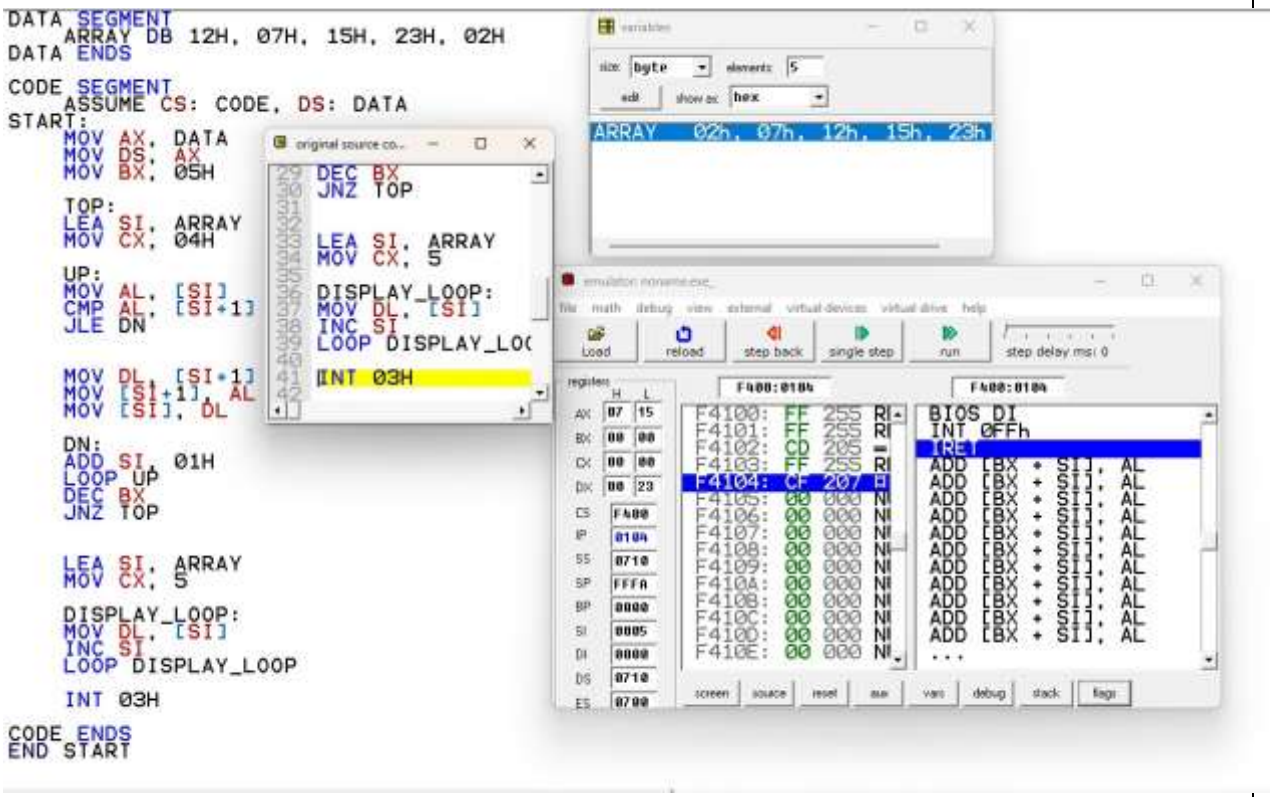
1. Write an ALP to arrange numbers in array in ascending order.

Code:

```
DATA SEGMENT
   ARRAY DB 12H, 07H, 15H, 23H, 02H
DATA ENDS

CODE SEGMENT
   ASSUME CS: CODE, DS: DATA START:
   MOV AX, DATA
   MOV DS, AX
   MOV BX, 05H



   TOP:
   LEA SI, ARRAY
   MOV CX, 04H

   UP:
   MOV AL, [SI]
   CMP AL, [SI+1]
   JLE DN


   MOV DL, [SI+1]
   MOV [SI+1], AL
   MOV [SI], DL

   DN:
   ADD SI, 01H
```

```
    LOOP UP
    DEC BX
    JNZ TOP

    LEA SI, ARRAY
    MOV CX, 5

    DISPLAY_LOOP:
    MOV DL, [SI]
    INC SI
    LOOP DISPLAY_LOOP

    INT 03H

CODE ENDS
END START
```

Output:

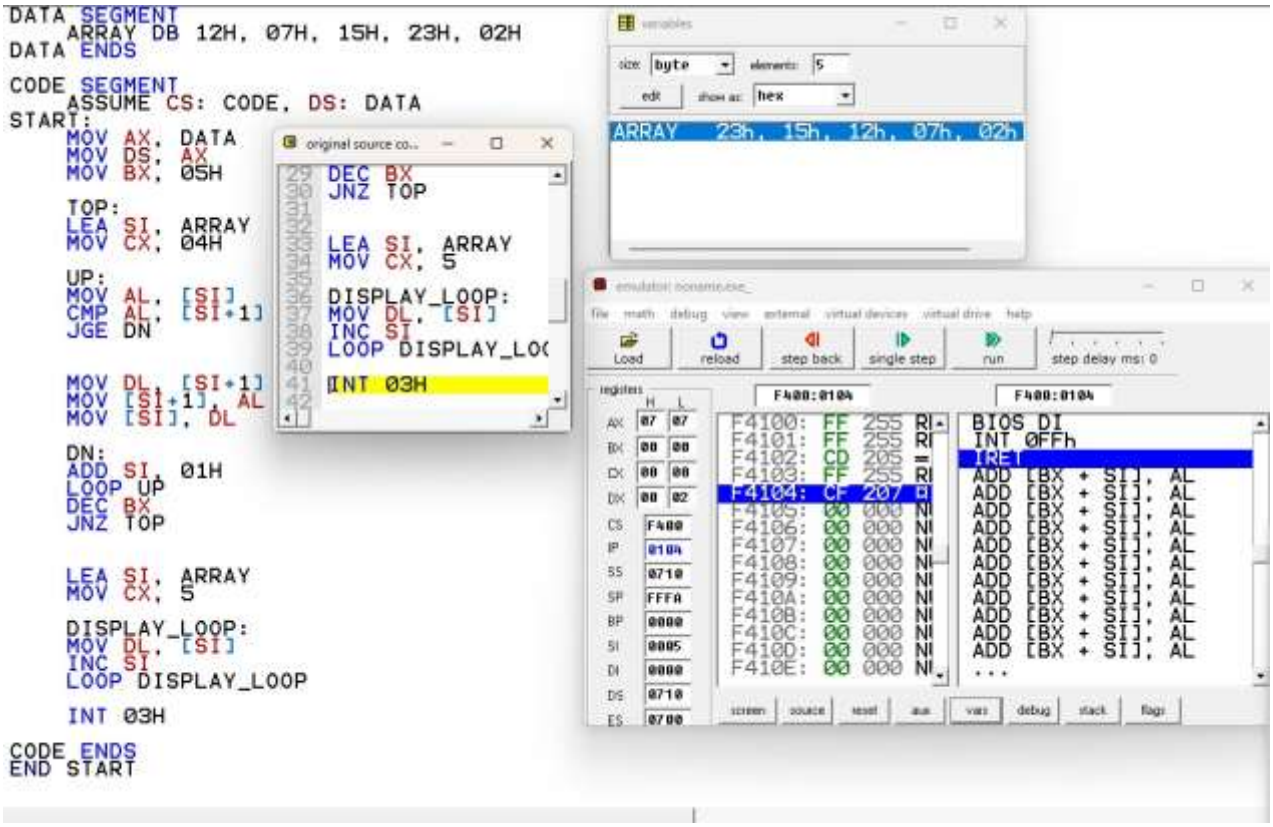

DATA SEGMENT
ARRAY DB 12H, 07H, 15H, 23H, 02H
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS: DATA
START:
MOV AX, DATA
MOV DS, AX
MOV BX, 05H
TOP:
LEA SI, ARRAY
MOV CX, 04H
UP:
MOV AL, [SI]
CMP AL, [SI+1]
JLE DN
MOV DL, [SI+1]
MOV [SI+1], AL
MOV [SI], DL
DN:
ADD SI, 01H
LOOP UP
DEC BX
JNZ TOP
LEA SI, ARRAY
MOV CX, 5
DISPLAY_LOOP:
MOV DL, [SI]
INC SI
LOOP DISPLAY_LOOP
INT 03H
CODE ENDS
END START
ARRAY 02h, 07h, 12h, 15h, 23h
Load
reload
step back
single step
run
BIOS DI
INT 0FFh
IRET

2. Write an ALP arrange numbers in array in descending order.

```
DATA SEGMENT
   ARRAY DB 12H, 07H, 15H, 23H, 02H
DATA ENDS

CODE SEGMENT
   ASSUME CS: CODE, DS: DATA START:
   MOV AX, DATA
   MOV DS, AX
   MOV BX, 05H

   TOP:
   LEA SI, ARRAY
   MOV CX, 04H

   UP:
   MOV AL, [SI]
   CMP AL, [SI+1]
   JGE DN


   MOV DL, [SI+1]
   MOV [SI+1], AL
   MOV [SI], DL

   DN:
   ADD SI, 01H
   LOOP UP
   DEC BX
   JNZ TOP


   LEA SI, ARRAY
   MOV CX, 5

   DISPLAY_LOOP:
   MOV DL, [SI]
   INC SI
   LOOP DISPLAY_LOOP
   INT 03H
```

CODE ENDS
END START

Output:



Conclusion:

With this practical we learned how to arrange arrays in Ascending and Descending Order.

VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: MIC	Subject Code:224145
Semester:4th Semester	Course: Computer Engineering
Laboratory No: L004B	Name of Subject Teacher: Pragati Mali
Name of Student: Siddharth Shah	Roll Id: 22203A0041

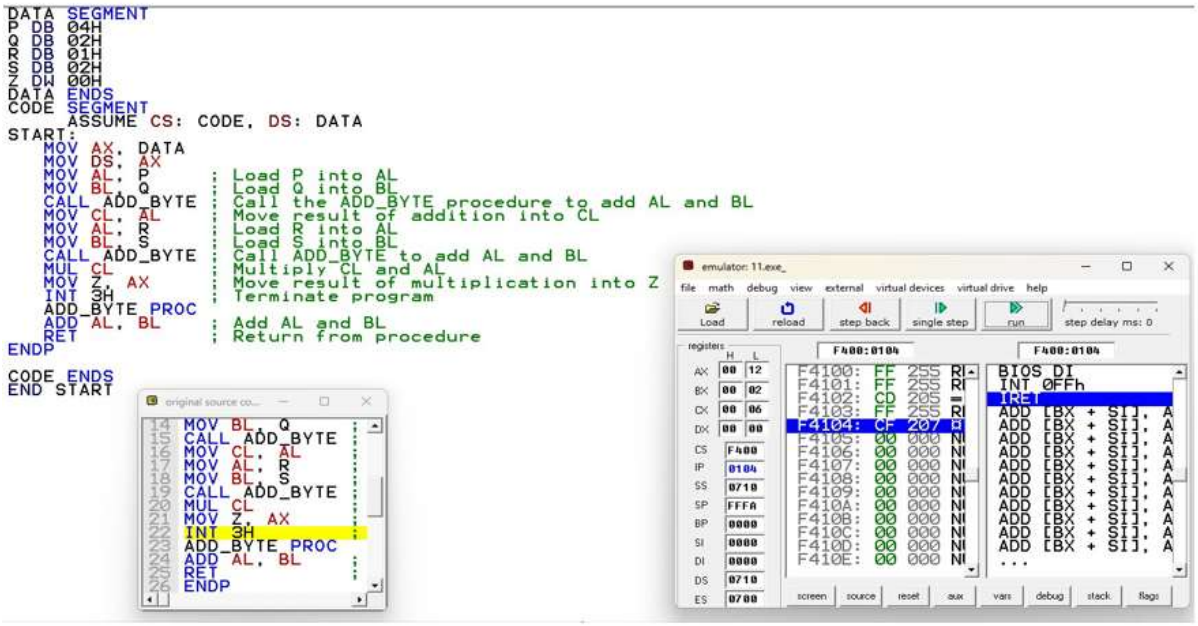
Experiment No:	11
Title of Experiment	Write an ALP using Procedure to solve equation such as Z=(A+B)*(C+D).

Q. Write an ALP using Procedure to solve equation such as Z=(A+B)*(C+D).

```
DATA SEGMENT
P DB 04H
Q DB 02H
R DB 01H
S DB 02H
Z DW 00H
DATA ENDS
CODE SEGMENT
    ASSUME CS: CODE, DS: DATA
START:
  MOV AX, DATA
  MOV DS, AX
  MOV AL, P
  MOV BL, Q
  CALL ADD_BYTE
  MOV CL, AL
  MOV AL, R
  MOV BL, S
  CALL ADD_BYTE
  MUL CL
  MOV Z, AX
```

```
 INT 3H
 ADD_BYTE PROC
ADD AL,BL
 RET
ENDP

CODE ENDS
END START
```

OUTPUT:



VP Vidyalankar Polytechnic	DEPARTMENT OF COMPUTER ENGINEERING

Subject: MICROPROCESSOR	Subject Code:22415
Semester:4th Semester	Course: Computer Engineering
Laboratory No:L004	Name of Subject Teacher: Pragati Mali
Name of Student: Siddharth Shah	Roll Id: 22203A0041

Experiment No:	12
Title of Experiment	Write an ALP for Z=(A+B)*(C+D) using macro.

- **PROGRAM:-**

```
DATA SEGMENT
  A DB 04H
B DB 04H
C DB 01H
D DB 02H
  R1 DB 00H
  R2 DB 00H
  Z DW 00H
DATA ENDS
SUM_BYTE MACRO A1, A2, RES
  MOV AL, A1    ; Move A1 into AL
  ADD AL, A2    ; Add A2 to AL
  MOV RES, AL   ; Move the result in AL to RES
ENDM
```

```
CODE SEGMENT
  ASSUME CS: CODE, DS: DATA
START:
  MOV AX, DATA
  MOV DS, AX
SUM_BYTE A, B, R1   ; Call SUM_BYTE macro to add A and B, store result in R1
SUM_BYTE C, D, R2   ; Call SUM_BYTE macro to add C and D, store result in R2
  MOV AL, R1     ; Move R1 into AL
  MUL R2         ; Multiply R2 with AL, result in AX
  MOV Z, AX      ; Move result of multiplication into Z
  INT 3H
CODE ENDS
END START
```

- **OUTPUT: -**