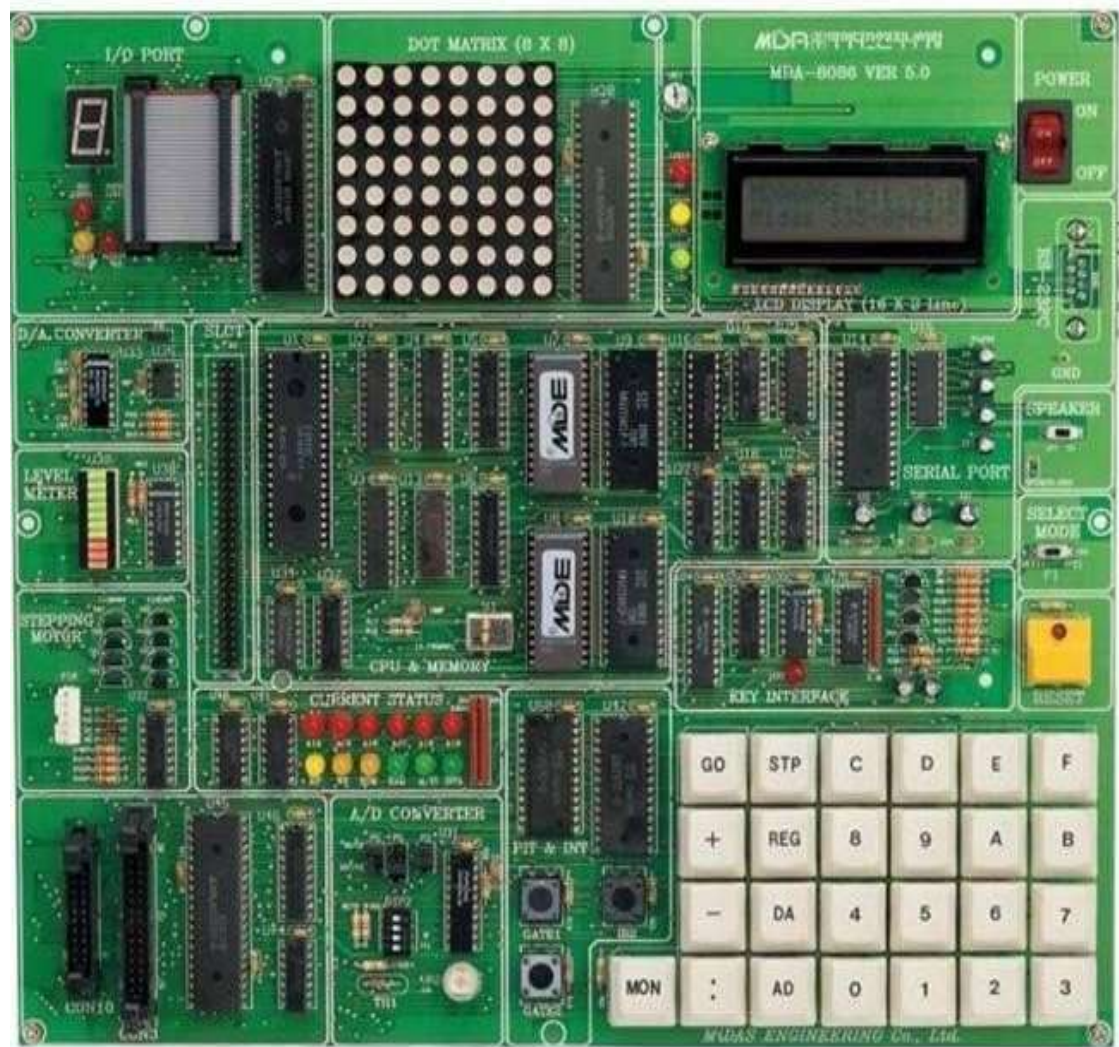


INDEX

Experiment No.	Experiment's Name	Page No.
01	Introductory Idea of 8086 Microprocessor and exploring MDA-Win8086 Trainer Kit.	
02	Introductory Idea of 8086 Microprocessor and exploring EMU8086 emulator.	
03	Write a and verify Assembly Language Program for Addition and Subtraction in emu8086.	
04	Write a and verify Assembly Language Program for Multiplication and Division in emu8086.	
05	Write a and verify Assembly Language Program for different logical operation in emu8086.	
06	Interfacing LED, 7 Segment display with 8086 using 8255 PPI in MDA- Win8086 Trainer Kit.	

Experiment No.: 01

MDA-Win8086 Overview:

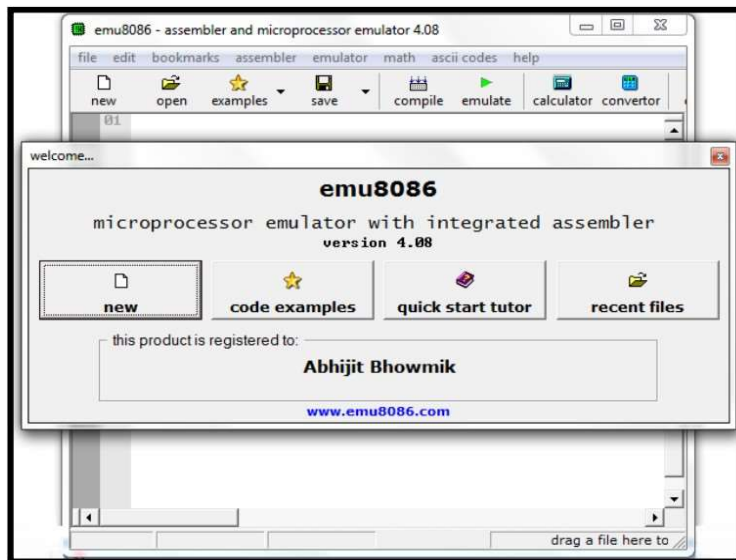


Experiment No.: 02

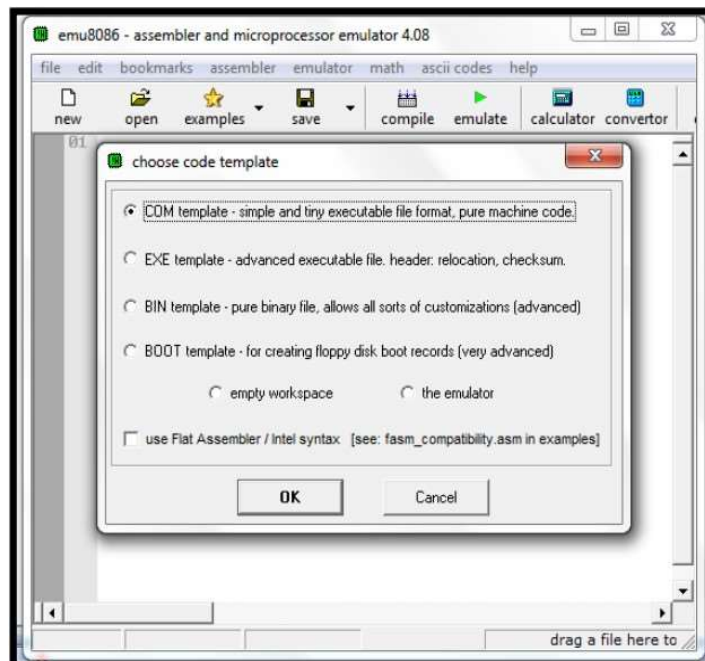
Emu8086:



Click “new”

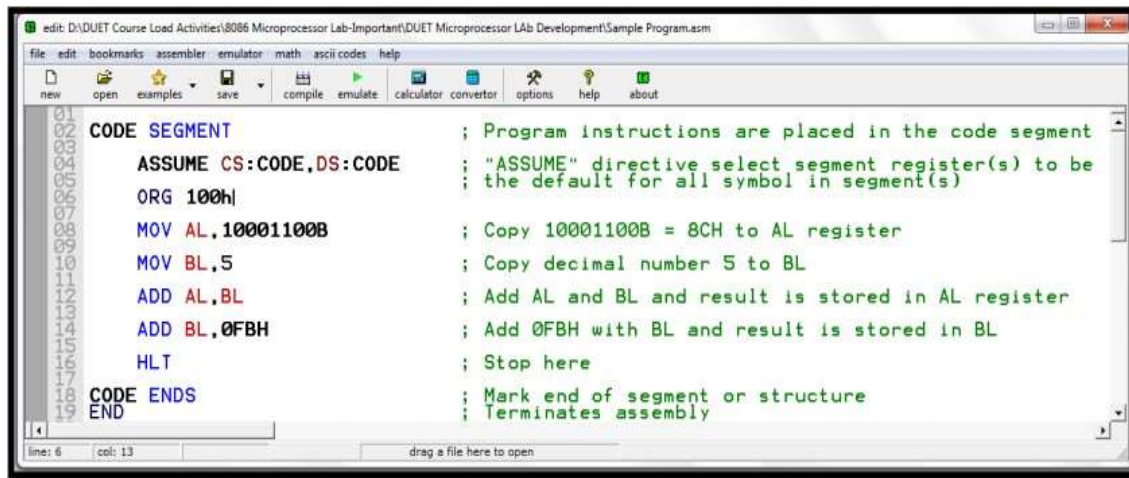


Choose COM template and then click Ok



Example: Create COM file and write the following 8086 assembly language

Example on the emulator to see how it works

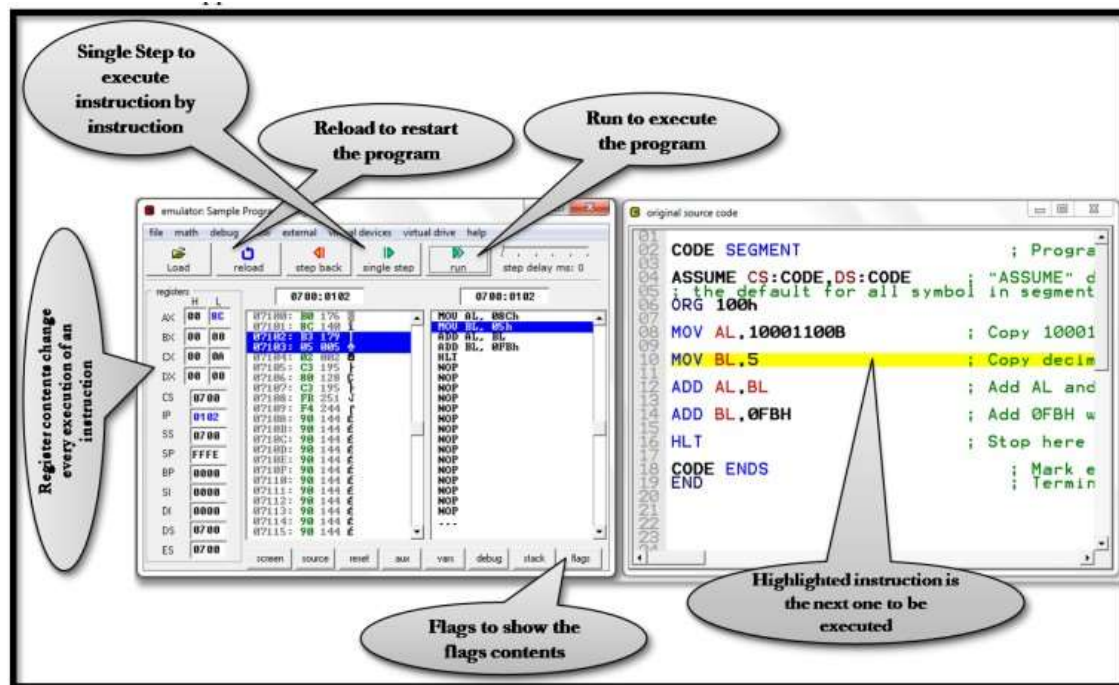


The screenshot shows a window titled "edit: D:\DUET Course Load Activities\8086 Microprocessor Lab-Important\DUET Microprocessor Lab Development\Sample Program.asm". The menu bar includes file, edit, bookmarks, assembler, emulator, math, ascii codes, and help. The toolbar contains icons for new, open, examples, save, compile, emulate, calculator, convertor, options, help, and about. The main text area contains the following assembly code:

```
CODE SEGMENT ; Program instructions are placed in the code segment
ASSUME CS:CODE,DS:CODE ; "ASSUME" directive select segment register(s) to be
                        ; the default for all symbol in segment(s)
ORG 100h
MOV AL,10001100B ; Copy 10001100B = 8CH to AL register
MOV BL,5 ; Copy decimal number 5 to BL
ADD AL,BL ; Add AL and BL and result is stored in AL register
ADD BL,0FBH ; Add 0FBH with BL and result is stored in BL
HLT ; Stop here
CODE ENDS ; Mark end of segment or structure
END ; Terminates assembly
```

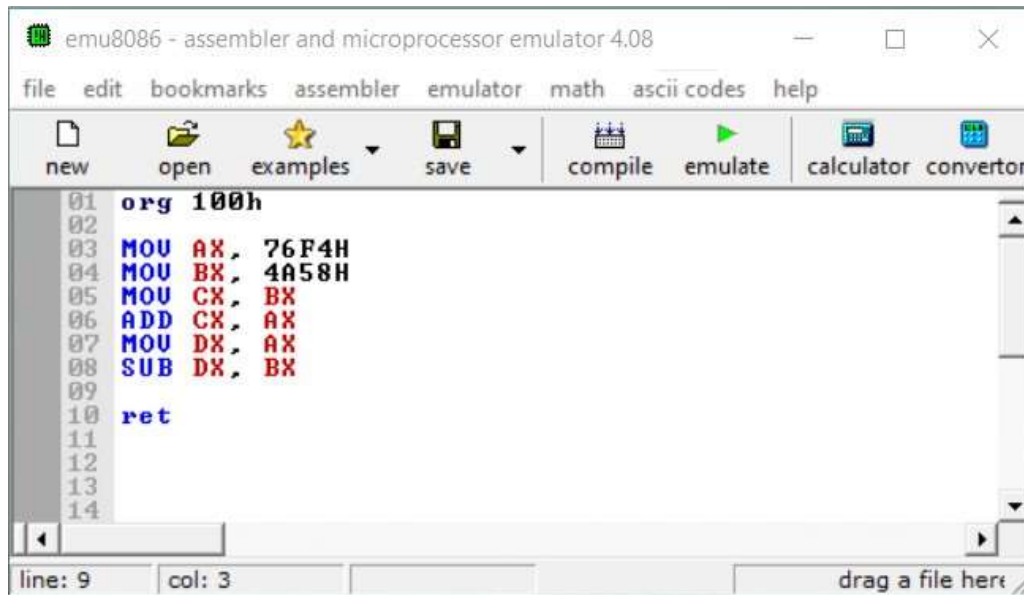
The status bar at the bottom shows "line: 6 col: 13" and "drag a file here to open".

Start emulation by clicking the “emulate” button on the toolbar. A new emulator window will appear



Experiment No.: 03

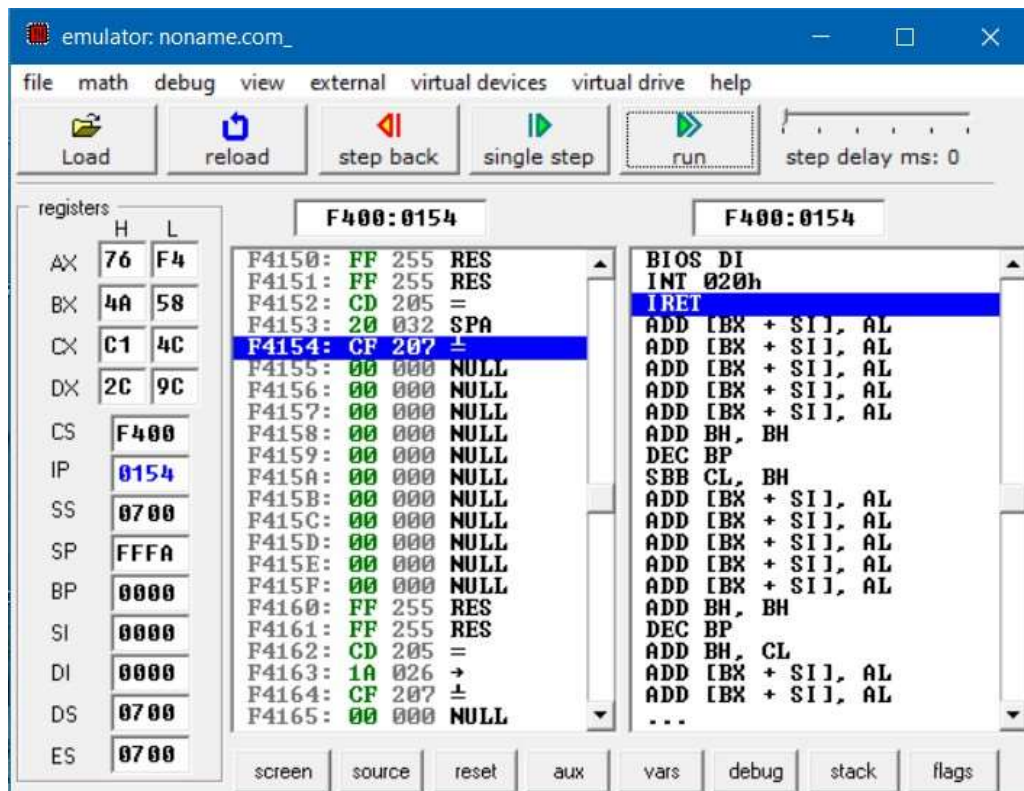
Assembly Program in emu8086:



```
01 org 100h
02
03 MOV AX, 76F4H
04 MOV BX, 4A58H
05 MOV CX, BX
06 ADD CX, AX
07 MOV DX, AX
08 SUB DX, BX
09
10 ret
11
12
13
14
```

line: 9 col: 3 drag a file here

Output:



emulator: noname.com_

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	76	F4
BX	4A	58
CX	C1	4C
DX	2C	9C
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154

F4150:	FF	255	RES
F4151:	FF	255	RES
F4152:	CD	205	=
F4153:	20	032	SPA
F4154:	CF	207	±
F4155:	00	000	NULL
F4156:	00	000	NULL
F4157:	00	000	NULL
F4158:	00	000	NULL
F4159:	00	000	NULL
F415A:	00	000	NULL
F415B:	00	000	NULL
F415C:	00	000	NULL
F415D:	00	000	NULL
F415E:	00	000	NULL
F415F:	00	000	NULL
F4160:	FF	255	RES
F4161:	FF	255	RES
F4162:	CD	205	=
F4163:	1A	026	→
F4164:	CF	207	±
F4165:	00	000	NULL


F400:0154

BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
...

screen source reset aux vars debug stack flags

Experiment No.: 04

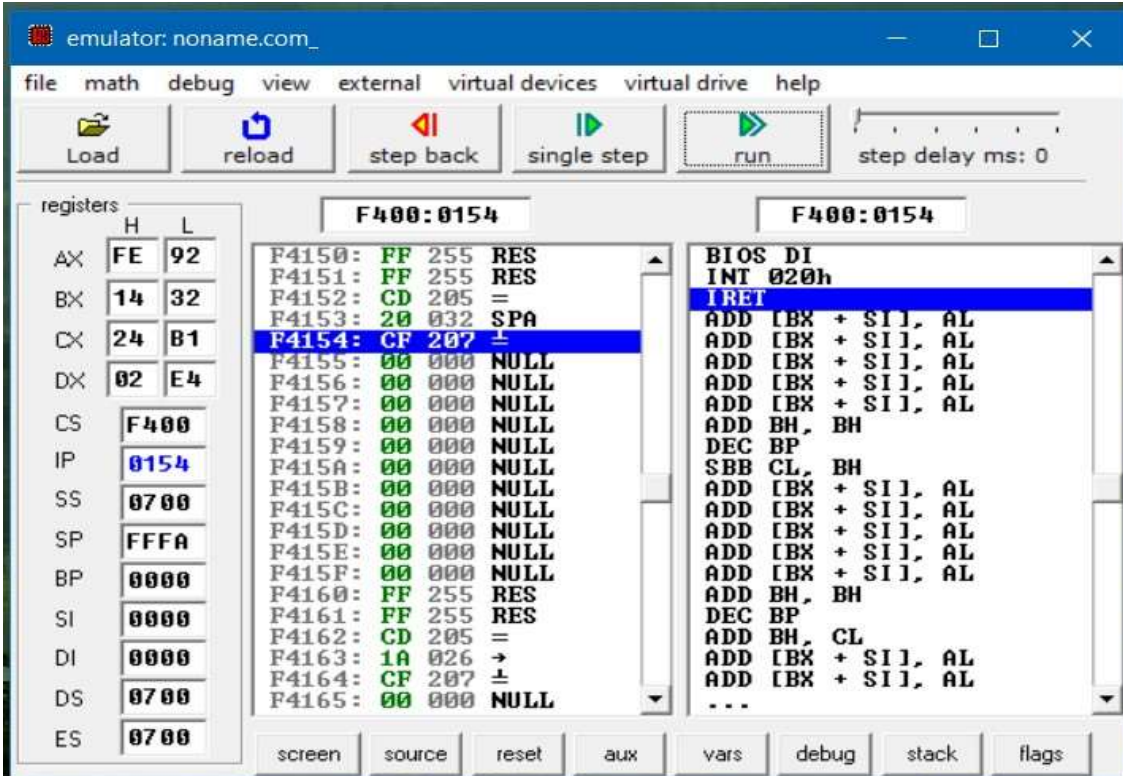
Assembly Program in emu8086 for Multiplication:



```
01 ; You may customize this and other start-up templates
02 ; The location of this template is c:\emu8086\inc
03
04
05 org 100h
06
07 MOV AX, 1432H
08 MOV BX, AX
09 MOV CX, 24B1H
10 MUL CX
11
12 ret
13
14
```

line: 8 col: 11 drag a file here

Output:



emulator: noname.com_

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	FE	92
BX	14	32
CX	24	B1
DX	02	E4
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154

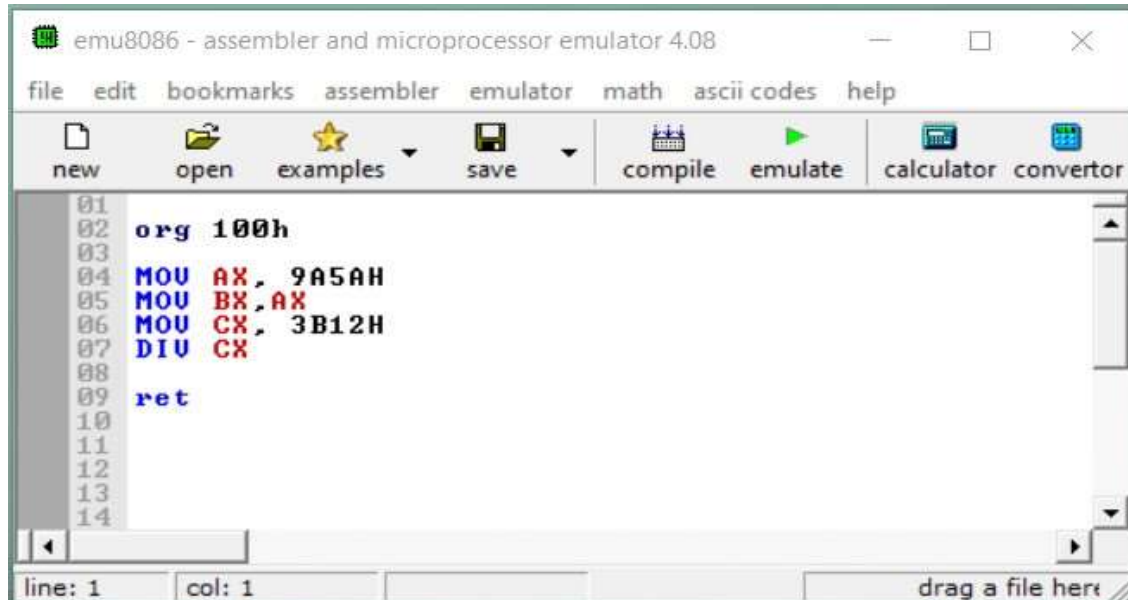
F4150:	FF	255	RES
F4151:	FF	255	RES
F4152:	CD	205	=
F4153:	20	032	SPA
F4154:	CF	207	±
F4155:	00	000	NULL
F4156:	00	000	NULL
F4157:	00	000	NULL
F4158:	00	000	NULL
F4159:	00	000	NULL
F415A:	00	000	NULL
F415B:	00	000	NULL
F415C:	00	000	NULL
F415D:	00	000	NULL
F415E:	00	000	NULL
F415F:	00	000	NULL
F4160:	FF	255	RES
F4161:	FF	255	RES
F4162:	CD	205	=
F4163:	1A	026	→
F4164:	CF	207	±
F4165:	00	000	NULL

F400:0154

BIOS DI
INT 020h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
...

screen source reset aux vars debug stack flags

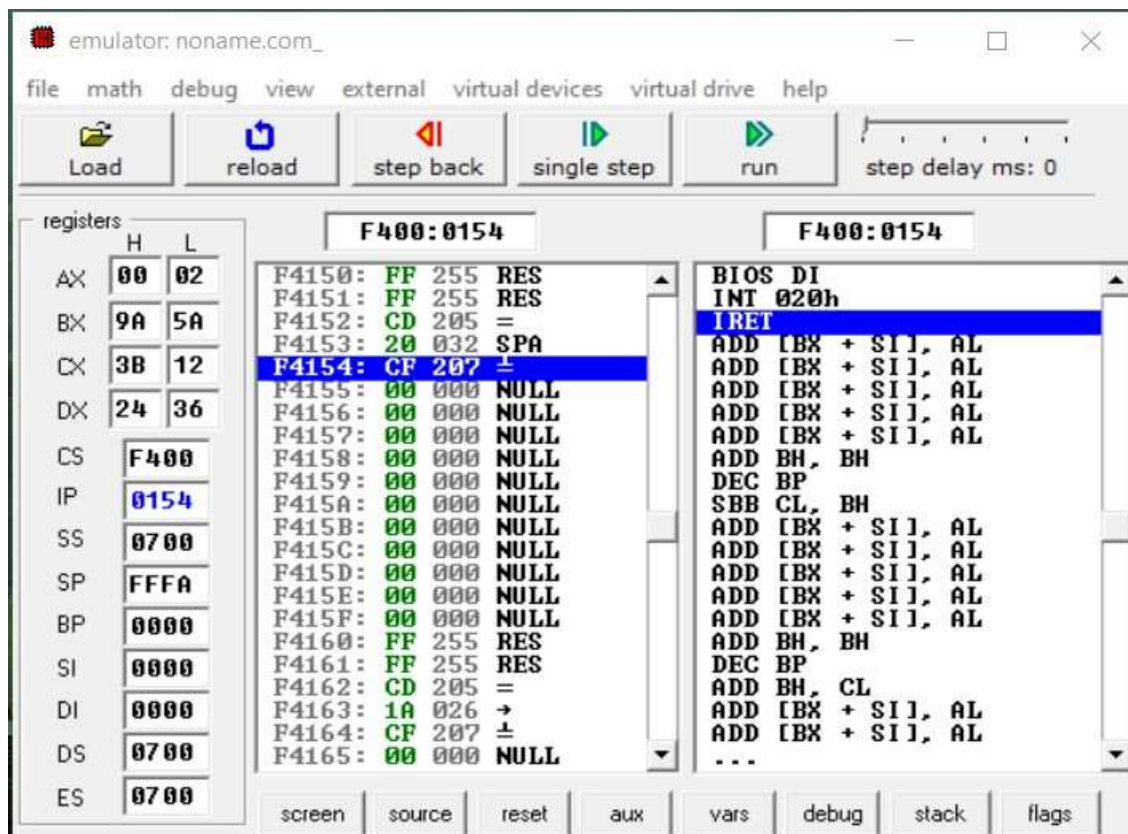
Assembly Program in emu8086 for Division:



```
01
02 org 100h
03
04 MOV AX, 9A5AH
05 MOV BX, AX
06 MOV CX, 3B12H
07 DIV CX
08
09 ret
10
11
12
13
14
```

line: 1 col: 1 drag a file here

Output:



emulator: noname.com_

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	02
BX	9A	5A
CX	3B	12
DX	24	36
CS	F400	
IP	0154	
SS	0700	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0154

F4150:	FF	255	RES
F4151:	FF	255	RES
F4152:	CD	205	=
F4153:	20	032	SPA
F4154:	CF	207	±
F4155:	00	000	NULL
F4156:	00	000	NULL
F4157:	00	000	NULL
F4158:	00	000	NULL
F4159:	00	000	NULL
F415A:	00	000	NULL
F415B:	00	000	NULL
F415C:	00	000	NULL
F415D:	00	000	NULL
F415E:	00	000	NULL
F415F:	00	000	NULL
F4160:	FF	255	RES
F4161:	FF	255	RES
F4162:	CD	205	=
F4163:	1A	026	→
F4164:	CF	207	±
F4165:	00	000	NULL

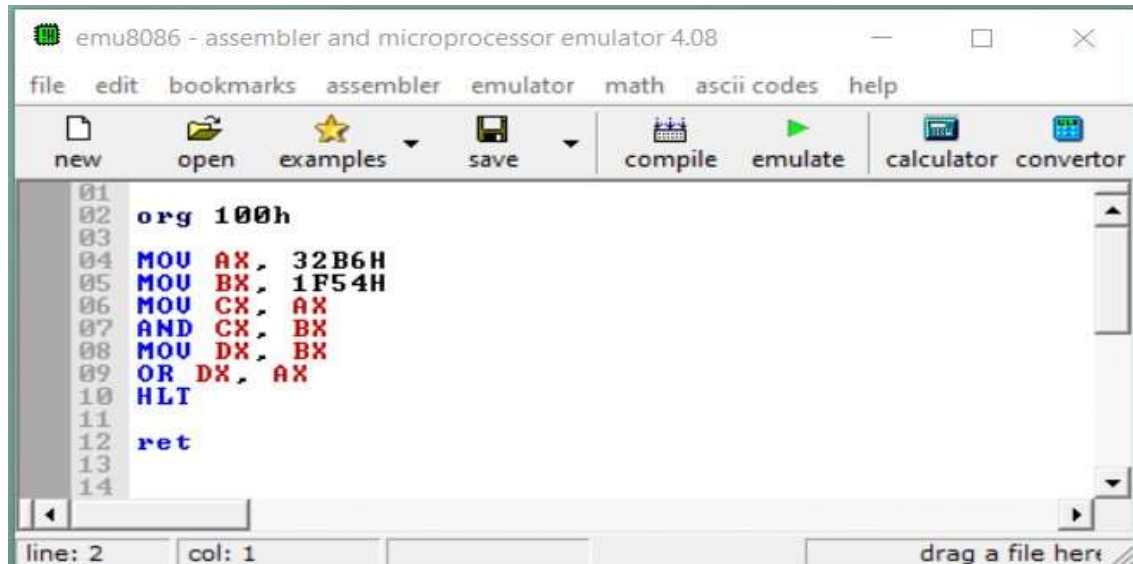
F400:0154

BIOS DI
INT 020h
I RET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
SBB CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADD BH, CL
ADD [BX + SI], AL
ADD [BX + SI], AL
...

screen source reset aux vars debug stack flags

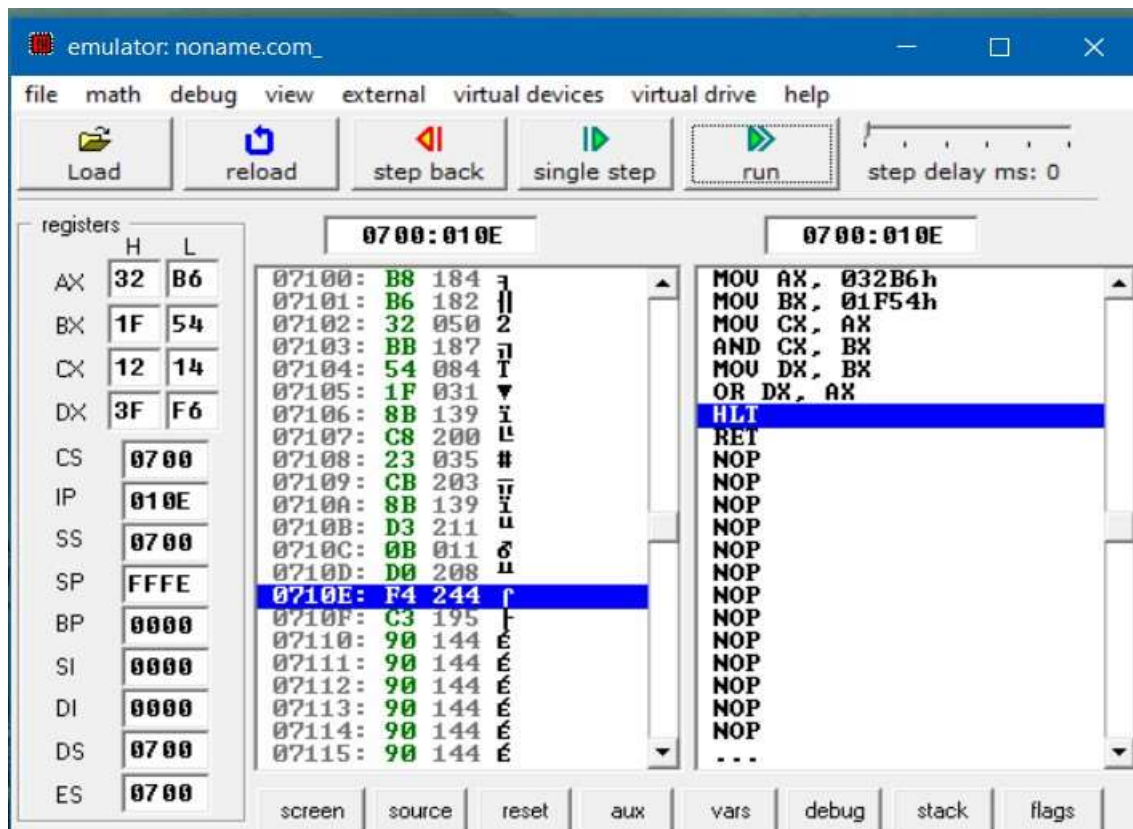
Experiment No.: 05

Assembly Program in emu8086 for Logical AND and OR operation:



```
01 org 100h
02
03
04 MOV AX, 32B6H
05 MOV BX, 1F54H
06 MOV CX, AX
07 AND CX, BX
08 MOV DX, BX
09 OR DX, AX
10 HLT
11
12 ret
13
14
```

Output:



registers	H	L
AX	32	B6
BX	1F	54
CX	12	14
DX	3F	F6
CS	0700	
IP	010E	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

memory	hex	dec	comment
07100:010E	B8	184	MOV AX, 032B6h
07101:010F	B6	182	MOV BX, 01F54h
07102:0110	32	050	MOV CX, AX
07103:0111	BB	187	AND CX, BX
07104:0112	54	084	MOV DX, BX
07105:0113	1F	031	OR DX, AX
07106:0114	8B	139	HLT
07107:0115	C8	200	RET
07108:0116	23	035	NOP
07109:0117	CB	203	NOP
0710A:0118	8B	139	NOP
0710B:0119	D3	211	NOP
0710C:011A	0B	011	NOP
0710D:011B	D0	208	NOP
0710E:011C	F4	244	NOP
0710F:011D	C3	195	NOP
07110:011E	90	144	NOP
07111:011F	90	144	NOP
07112:0120	90	144	NOP
07113:0121	90	144	NOP
07114:0122	90	144	NOP
07115:0123	90	144	NOP

Assembly Program in emu8086 for Logical XOR and NOT operation:

emu8086 - assembler and microprocessor emulator 4.08

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator converter

```

01
02  org 100h
03  MOV AX, 32B6H
04  MOV BX, 1F54H
05  MOV CX, AX
06  XOR CX, BX
07  MOV DX, CX
08  NOT DX
09  HLT
10  ret
11
12
13
14

```

line: 2 col: 9

drag a file here

Output:

The screenshot shows the x86-64 emulator interface. At the top, there's a title bar with the URL "emulator: noname.com_". Below it is a menu bar with options: file, math, debug, view, external, virtual devices, virtual drive, help. A toolbar contains buttons for Load, reload, step back, single step, run, and a step delay slider set to 0 ms.

The main area is divided into three sections:

- Registers:** A table on the left showing the state of various registers. The Address Register (IP) is highlighted with a value of 010E.
- Memory:** A central pane showing memory addresses from 07100 to 07115. Address 0710E is highlighted, containing the value F4.
- Instructions:** A right pane showing a list of instructions starting from 07100. The instruction at 0710E, "HLT", is highlighted.

At the bottom, there's a status bar with tabs for screen, source, reset, aux, vars, debug, stack, and flags.

Register	H	L
AX	32	B6
BX	1F	54
CX	2D	E2
DX	D2	1D
CS	0700	
IP	010E	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

Address	Hex	Dec	Symbol
07100	B8	184	
07101	B6	182	
07102	32	050	
07103	BB	187	
07104	54	084	
07105	1F	031	
07106	8B	139	
07107	C8	200	
07108	33	051	
07109	CB	203	
0710A	8B	139	
0710B	D1	209	
0710C	F7	247	
0710D	D2	210	
0710E	F4	244	
0710F	C3	195	
07110	90	144	
07111	90	144	
07112	90	144	
07113	90	144	
07114	90	144	
07115	90	144	

Address	Instruction
07100	MOV AX, 032B6h
07101	MOV BX, 01F54h
07102	MOV CX, AX
07103	XOR CX, BX
07104	MOV DX, CX
07105	NOT DX
07106	HLT
07107	RET
07108	NOP
07109	NOP
0710A	NOP
0710B	NOP
0710C	NOP
0710D	NOP
0710E	NOP
0710F	NOP
07110	NOP
07111	NOP
07112	NOP
07113	NOP
07114	NOP
07115	...