EXPERIMENT NO- 10

AIM: WAP for device driver (Accept the input from keyboard and display it)

Resource Required: P-IV and above RAM 128MB, Dot Matrix Printer, Emu 8086, MASM 611/ TASM, Turbo C/C++, Printer, Printout Stationary.

THEORY:

DOS Interrupts: MS-DOS provides many common services through INT 21h. Entire books have been written about the variety of functions available.

Most basic ones for console input and output here.

1) Input a character.

```
MOV AH, 01h

INT 21h
```

After the interrupt, AL contains the ASCII code of the input character. The character is echoed (displayed on the screen). Use function code 8 instead of 1 for no echo.

2) Input a string.

```
SECTION .data

Buffer DB BUFSIZE ;BUFSIZE is max number of chars to read, <= 255

RESB BUFSIZE + 1

SECTION .text

MOV DX, Buffer

MOV AH, 0Ah

INT 21h
```

After the interrupt, BYTE [Buffer + 1] will contain the number of characters read, and the characters themselves will start at Buffer + 2. The characters will be terminated by a carriage return (ASCII code 13), although this will not be included in the count

3) Output a character.

```
MOV DL, ...

MOV AH, 02h
```

INT 21h

Load the desired character into DL, then call the interrupt with function code 2 in AH.

4) Output a string.

MOV DX, ...

MOV AH, 09h

INT 21h

Load the address of a '$'-terminated string into DX, then call the interrupt with function code 9

in AH.

5) Exit.

MOV AL, ...

MOV AH, 4Ch

INT 21h

Load the return code (0 for normal exit, non-zero for error) into AL, then call the interrupt with function code 4Ch in AH.

ALGORITHM:

Step I : Accept input from keyboard

Step II : Check if input=’0’.if yes goto Step IV otherwise continue next line

Step III : Jump to step 1

Step IV : Stop

CONCLUSION: We have successfully:

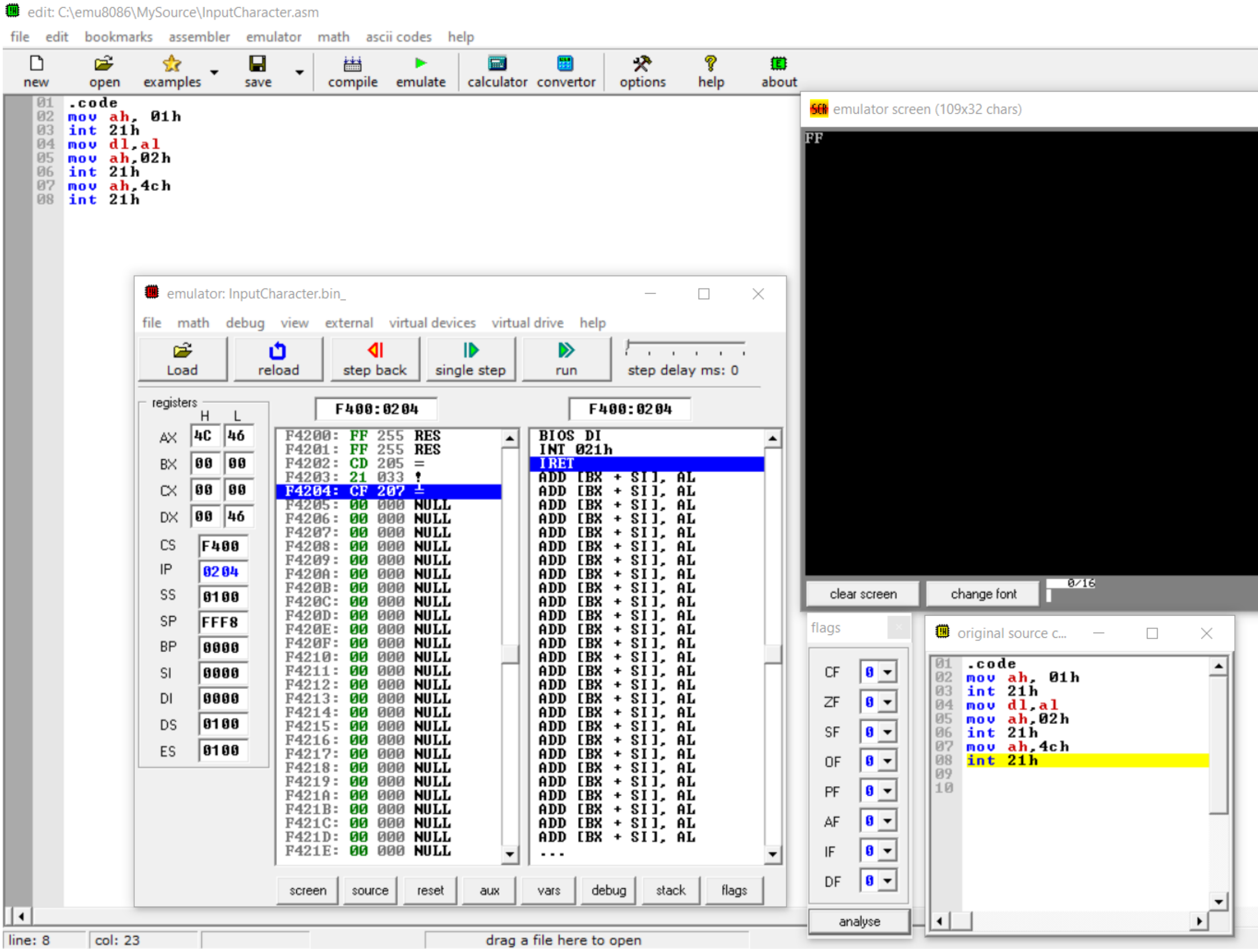
I) Accepted input of character from the keyboard and display it on the terminal using INT 21h Instruction in Assembly Language using EMU8086.

II) Accepted input of string from the keyboard using INT 21h Instruction in Assembly Language using EMU8086.

III) Display string on the terminal using INT 21h Instruction in Assembly Language using EMU8086.

Code and Output:

I) Accepting character and displaying it:



edit: C:\emu8086\MySource\InputCharacter.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator: InputCharacter.bin_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
screen source reset aux vars debug stack flags
emulator screen (109x32 chars)
clear screen change font 0/16
flags
analyse
original source c...
line: 8 col: 23 drag a file here to open

```
.code
mov ah, 01h
int 21h
mov dl,al
mov ah,02h
int 21h
mov ah,4ch
int 21h
```

```
FF
```

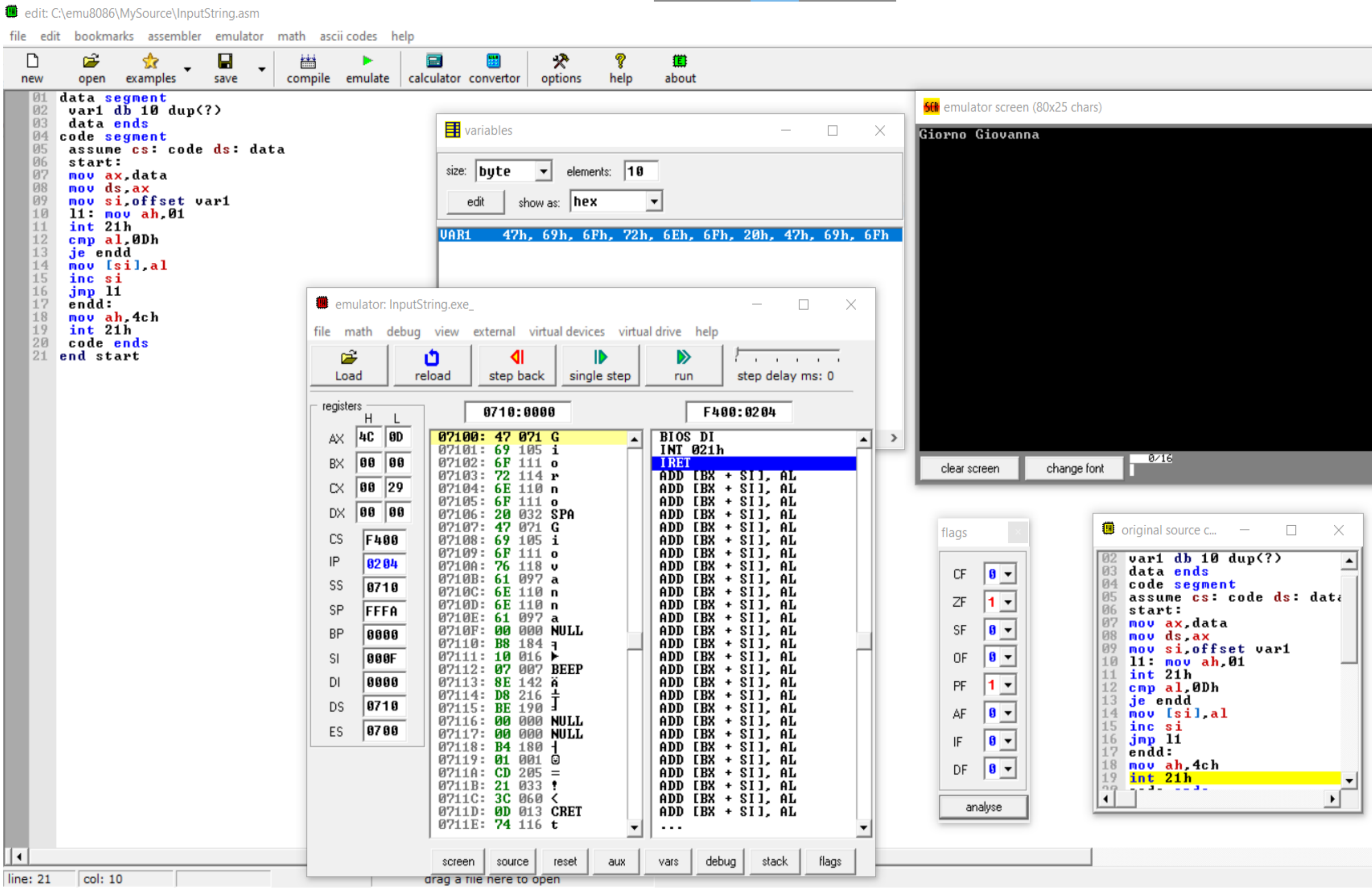
```
registers
     H   L
AX  4C  46
BX  00  00
CX  00  00
DX  00  46
CS  F400
IP  0204
SS  0100
SP  FFF8
BP  0000
SI  0000
DI  0000
DS  0100
ES  0100
```

```
F400:0204                   F400:0204

F4200:  FF 255 RES          BIOS DI
F4201:  FF 255 RES          INT 021h
F4202:  CD 205 =            IRET
F4203:  21 033 !            ADD [BX + SI], AL
F4204:  CF 207 ╧            ADD [BX + SI], AL
F4205:  00 000 NULL         ADD [BX + SI], AL
F4206:  00 000 NULL         ADD [BX + SI], AL
F4207:  00 000 NULL         ADD [BX + SI], AL
F4208:  00 000 NULL         ADD [BX + SI], AL
F4209:  00 000 NULL         ADD [BX + SI], AL
F420A:  00 000 NULL         ADD [BX + SI], AL
F420B:  00 000 NULL         ADD [BX + SI], AL
F420C:  00 000 NULL         ADD [BX + SI], AL
F420D:  00 000 NULL         ADD [BX + SI], AL
F420E:  00 000 NULL         ADD [BX + SI], AL
F420F:  00 000 NULL         ADD [BX + SI], AL
F4210:  00 000 NULL         ADD [BX + SI], AL
F4211:  00 000 NULL         ADD [BX + SI], AL
F4212:  00 000 NULL         ADD [BX + SI], AL
F4213:  00 000 NULL         ADD [BX + SI], AL
F4214:  00 000 NULL         ADD [BX + SI], AL
F4215:  00 000 NULL         ADD [BX + SI], AL
F4216:  00 000 NULL         ADD [BX + SI], AL
F4217:  00 000 NULL         ADD [BX + SI], AL
F4218:  00 000 NULL         ADD [BX + SI], AL
F4219:  00 000 NULL         ADD [BX + SI], AL
F421A:  00 000 NULL         ADD [BX + SI], AL
F421B:  00 000 NULL         ADD [BX + SI], AL
F421C:  00 000 NULL         ADD [BX + SI], AL
F421D:  00 000 NULL         ADD [BX + SI], AL
F421E:  00 000 NULL         ...
```

```
CF 0
ZF 0
SF 0
OF 0
PF 0
AF 0
IF 0
DF 0
```

II) Accepting String:



edit: C:\emu8086\MySource\InputString.asm

file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about

```
data segment
 var1 db 10 dup(?)
 data ends
code segment
 assume cs: code ds: data
 start:
 mov ax,data
 mov ds,ax
 mov si,offset var1
 l1: mov ah,01
 int 21h
 cmp al,0Dh
 je endd
 mov [si],al
 inc si
 jmp l1
 endd:
 mov ah,4ch
 int 21h
 code ends
end start
```

line: 21 col: 10

variables

size: byte elements: 10 show as: hex

```
VAR1   47h, 69h, 6Fh, 72h, 6Eh, 6Fh, 20h, 47h, 69h, 6Fh
```

emulator screen (80x25 chars)

```
Giorno Giovanna
```

emulator: InputString.exe_

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	4C	0D
BX	00	00
CX	00	29
DX	00	00
CS	F400	
IP	0204	
SS	0710	
SP	FFFA	
BP	0000	
SI	000F	
DI	0000	
DS	0710	
ES	0700	

```
0710:0000

07100: 47 071 G
07101: 69 105 i
07102: 6F 111 o
07103: 72 114 r
07104: 6E 110 n
07105: 6F 111 o
07106: 20 032 SPA
07107: 47 071 G
07108: 69 105 i
07109: 6F 111 o
0710A: 76 118 v
0710B: 61 097 a
0710C: 6E 110 n
0710D: 6E 110 n
0710E: 61 097 a
0710F: 00 000 NULL
07110: B8 184 ╕
07111: 10 016 ►
07112: 07 007 BEEP
07113: 8E 142 Ä
07114: D8 216 ╪
07115: BE 190 ╛
07116: 00 000 NULL
07117: 00 000 NULL
07118: B4 180 ┤
07119: 01 001 ☺
0711A: CD 205 ═
0711B: 21 033 !
0711C: 3C 060 <
0711D: 0D 013 CRET
0711E: 74 116 t
```

```
F400:0204

BIOS DI
INT 021h
IRET
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 [BX + SI], AL
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 [BX + SI], AL
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 [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

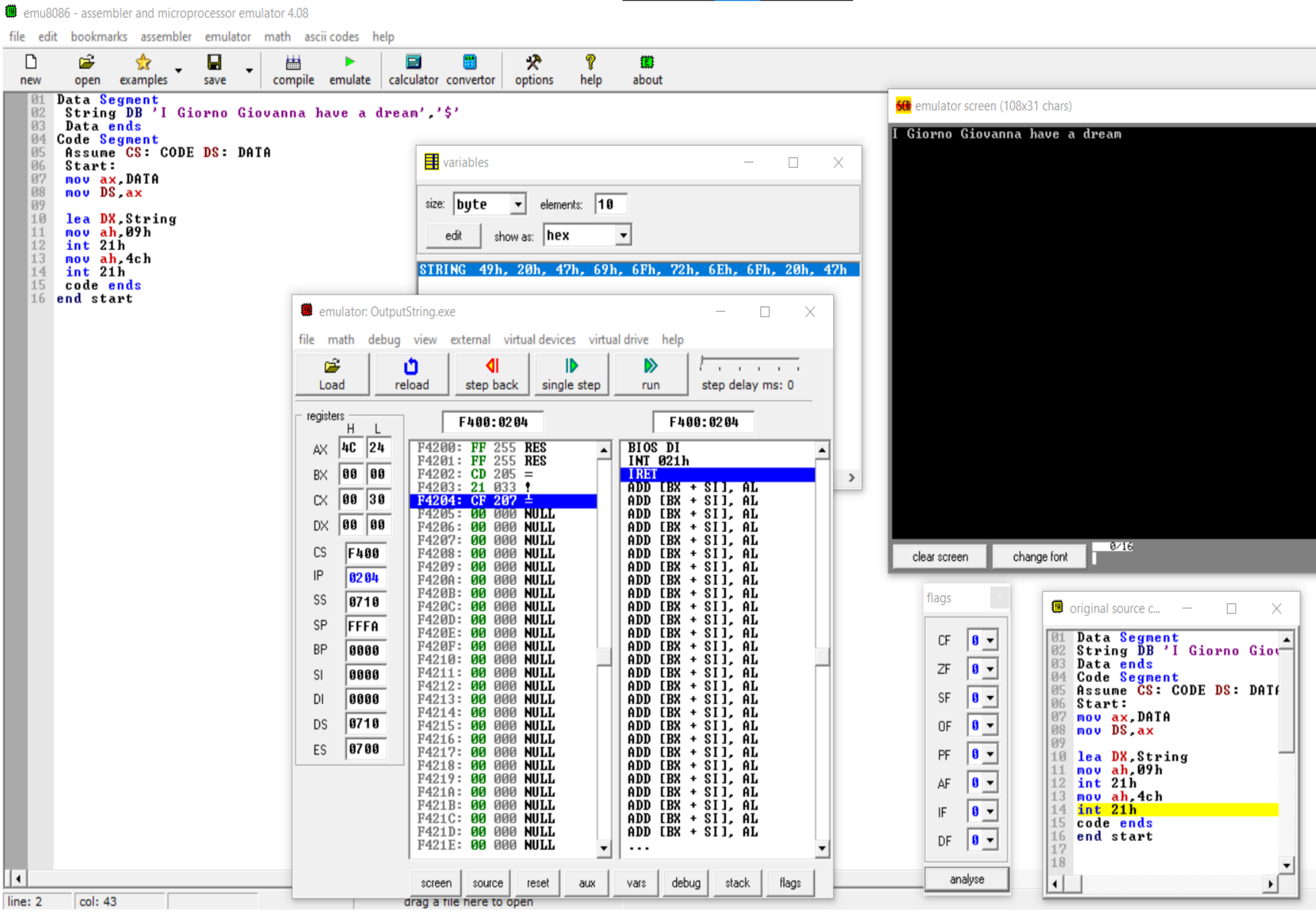
flags

CF	0
ZF	1
SF	0
OF	0
PF	1
AF	0
IF	0
DF	0

original source c...

```
var1 db 10 dup(?)
data ends
code segment
assume cs: code ds: data
start:
mov ax,data
mov ds,ax
mov si,offset var1
l1: mov ah,01
int 21h
cmp al,0Dh
je endd
mov [si],al
inc si
jmp l1
endd:
mov ah,4ch
int 21h
```

III) Displaying String:


emu8086 - assembler and microprocessor emulator 4.08
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator: OutputString.exe
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
screen source reset aux vars debug stack flags
emulator screen (108x31 chars)
clear screen change font
original source c...
analyse
line: 2 col: 43
drag a file here to open

```
Data Segment
 String DB 'I Giorno Giovanna have a dream','$'
 Data ends
Code Segment
 Assume CS: CODE DS: DATA
 Start:
 mov ax,DATA
 mov DS,ax

 lea DX,String
 mov ah,09h
 int 21h
 mov ah,4ch
 int 21h
 code ends
end start
```

variables: size: byte elements: 10 show as: hex

```
STRING  49h, 20h, 47h, 69h, 6Fh, 72h, 6Eh, 6Fh, 20h, 47h
```

Emulator screen:

```
I Giorno Giovanna have a dream
```

registers	H	L
AX	4C	24
BX	00	00
CX	00	30
DX	00	00
CS	F400	
IP	0204	
SS	0710	
SP	FFFA	
BP	0000	
SI	0000	
DI	0000	
DS	0710	
ES	0700	

```
F400:0204
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ╧
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
F4216: 00 000 NULL
F4217: 00 000 NULL
F4218: 00 000 NULL
F4219: 00 000 NULL
F421A: 00 000 NULL
F421B: 00 000 NULL
F421C: 00 000 NULL
F421D: 00 000 NULL
F421E: 00 000 NULL
```

```
F400:0204
BIOS DI
INT 021h
IRET
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 [BX + SI], AL
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 [BX + SI], AL
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 [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

flags	
CF	0
ZF	0
SF	0
OF	0
PF	0
AF	0
IF	0
DF	0