EXPERIMENT NO- 3

AIM: WAP to find factorial of number using procedure.

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

THEORY:

Instructions used in this program are:

1) JNZ (Jump if not Zero): This is conditional Jump. This instruction will jump to specified label when zero flag is not set.

2) DEC: DEC decrements the source by one

Syntax: DEC source

3) CMP: Compare the numerical value of the destination with the source and set flags appropriately. This comparison is carried out in the form of a subtraction to determine which of the operands has a greater value. After a CMP instruction, OF, SF, ZF and CF are set appropriately. For example, if the operands have equal values, then ZF if set.

Syntax:

CMP destination, source

4) CALL AND RET : These instructions interrupts the flow of a program by passing control to an internal or external subroutine .The return instruction returns the control from a subroutine back to a calling program .CALL passes the control to a label specified after the call keyword. When the subroutine ends with return instruction, the instruction following CALL are processed.

ALGORITHM:

Step I : Initialize the data segment

Step II : Initialize the variable A to number

Step III : Move the contents of variable A to AX

Step IV : Call procedure factorial

Step V : Decrement the value of variable A

Step VI : Multiply A with AX

Step VII : Copy the value of A to CX

Step VIII : Compare contents of CX with 01

Step IX : If equal go to step X else step V

Step X : Return to calling program

Step XI : Display the value in Fact variable

Step XII : Stop

CONCLUSION: We have successfully calculated Factorial of desired number using Procedural as well as normal programming approach in Assembly language using EMU 8086.

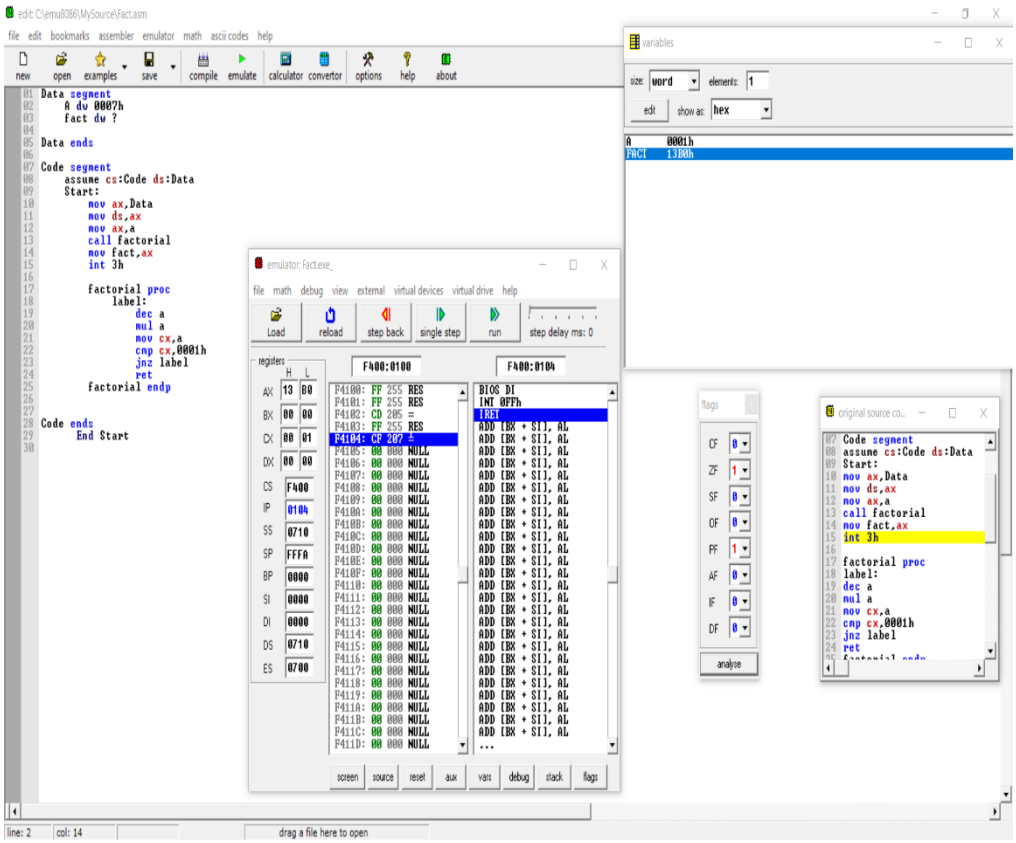
Procedural Code:

```
Data segment
    A dw 0007h
    fact dw ?

Data ends

Code segment
    assume cs:Code ds:Data
    Start:
        mov ax,Data
        mov ds,ax
        mov ax,a
        call factorial
        mov fact,ax
        int 3h

        factorial proc
```

```
label:
    dec a
    mul a
    mov cx,a
    cmp cx,0001h
    jnz label
    ret
factorial endp
```

```
Code ends
    End Start
```

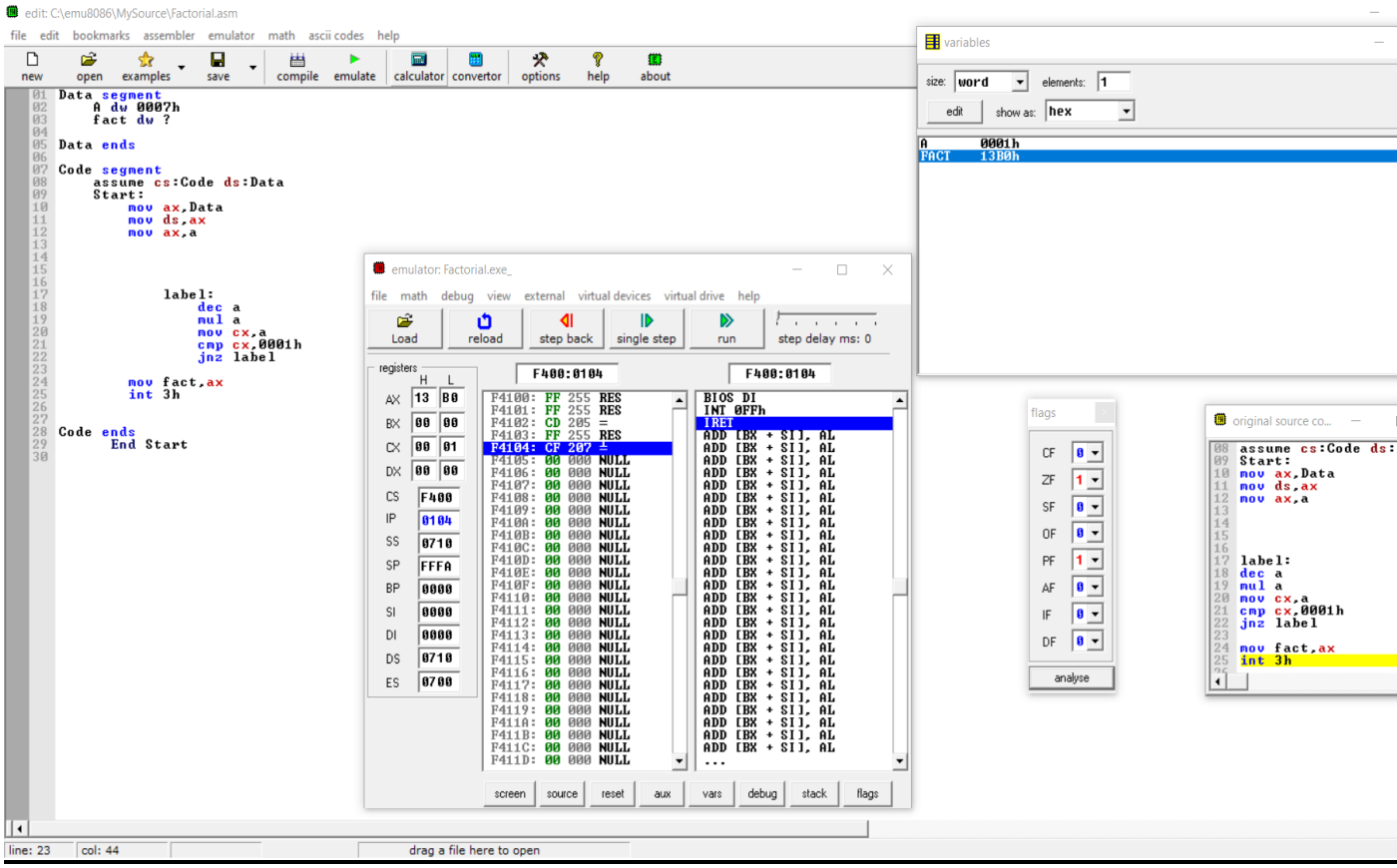
Output:



Normal code:

```
Data segment
   A dw 0007h
   fact dw ?

Data ends

Code segment
   assume cs:Code ds:Data
   Start:
      mov ax,Data
      mov ds,ax
      mov ax,a
         label:
            dec a
            mul a
            mov cx,a
            cmp cx,0001h
            jnz label

      mov fact,ax
      int 3h

Code ends
     End Start
```

Output:


edit: C:\emu8086\MySource\Factorial.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
Data segment
A dw 0007h
fact dw ?
Data ends
Code segment
assume cs:Code ds:Data
Start:
mov ax,Data
mov ds,ax
mov ax,a
label:
dec a
mul a
mov cx,a
cmp cx,0001h
jnz label
mov fact,ax
int 3h
Code ends
End Start
line: 23
col: 44
drag a file here to open
emulator: Factorial.exe_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers
AX 13 B0
BX 00 00
CX 00 01
DX 00 00
CS F400
IP 0104
SS 0710
SP FFFA
BP 0000
SI 0000
DI 0000
DS 0710
ES 0700
F400:0104
F4100: FF 255 RES
F4101: FF 255 RES
F4102: CD 205 =
F4103: FF 255 RES
F4104: CF 207 ±
F4105: 00 000 NULL
F4106: 00 000 NULL
F4107: 00 000 NULL
F4108: 00 000 NULL
F4109: 00 000 NULL
F410A: 00 000 NULL
F410B: 00 000 NULL
F410C: 00 000 NULL
F410D: 00 000 NULL
F410E: 00 000 NULL
F410F: 00 000 NULL
F4110: 00 000 NULL
F4111: 00 000 NULL
F4112: 00 000 NULL
F4113: 00 000 NULL
F4114: 00 000 NULL
F4115: 00 000 NULL
F4116: 00 000 NULL
F4117: 00 000 NULL
F4118: 00 000 NULL
F4119: 00 000 NULL
F411A: 00 000 NULL
F411B: 00 000 NULL
F411C: 00 000 NULL
F411D: 00 000 NULL
F400:0104
BIOS DI
INT 0FFh
IRET
ADD [BX + SI], AL
screen source reset aux vars debug stack flags
variables
size: word elements: 1
edit show as: hex
A 0001h
FACT 13B0h
flags
CF 0
ZF 1
SF 0
OF 0
PF 1
AF 0
IF 0
DF 0
analyse
original source co...
assume cs:Code ds:
Start:
mov ax,Data
mov ds,ax
mov ax,a
label:
dec a
mul a
mov cx,a
cmp cx,0001h
jnz label
mov fact,ax
int 3h