MICROPROCESSOR AND INTERFACING

Lab Manual # 08

Implementation of Assembly Language programs using Procedures and CALL using EMU8086

Procedure:

Procedure is a part of code that can be called from your program in order to make some specific task (remember the functions in C). Procedures make program more structural and easier to understand.

Procedure Syntax:

name PROC

; here goes the code

; of the procedure ...

RET

name ENDP

Example:

```
ORG 100h

MOV AL, 1

MOV BL, 2

CALL m2

RET; return to operating system.

m2 PROC
```

```
MUL BL          ; AX = AL * BL.
RET             ; return to caller
m2 ENDP
ret
```

Generally, procedure returns to the same point from where it was called

CALL a Procedure: -

Transfer control to a procedure. RET is used to return control to the instruction after the call.

TASK: 1

Write a program to find the factorial of given number by using procedure of FACT?

SOURCE CODE:

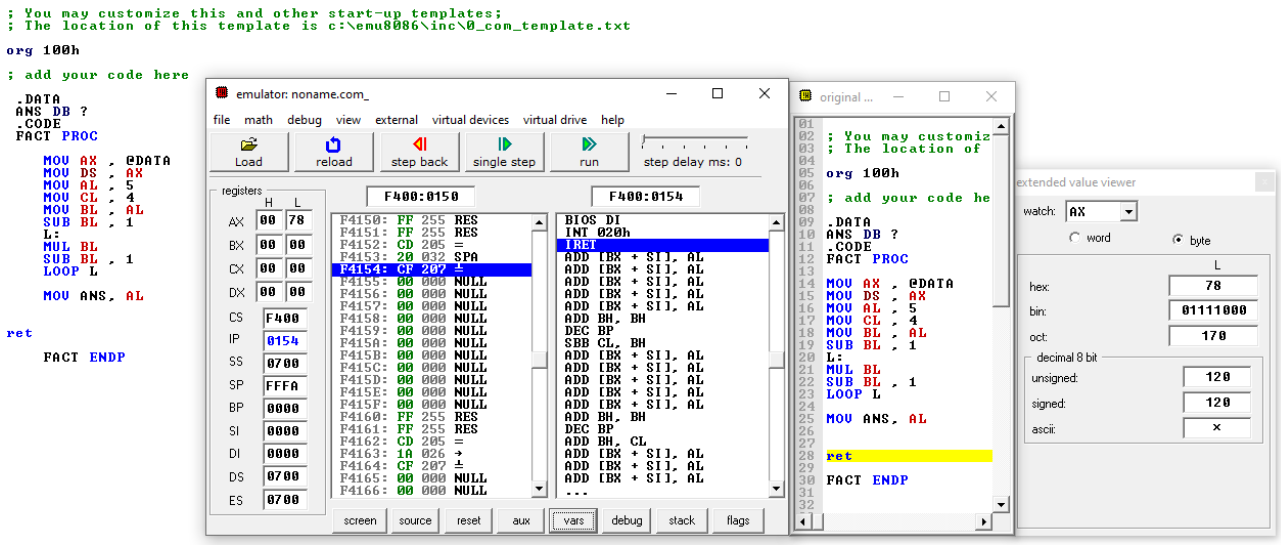
```
org 100h
; add your code here
.DATA
 ANS DB ?
 .CODE
 FACT PROC

 MOV AX , @DATA
```

```
MOV DS , AX
MOV AL , 5
MOV CL , 4
MOV BL , AL
SUB BL , 1
L:
MUL BL
SUB BL , 1
LOOP L

MOV ANS, AL
ret
FACT ENDP
```

OUTPUT:



TASK: 2

Write a program to swap any two numbers by using procedure of SWAP?

SOURCE CODE:

```
org 100h
; add your code here

 .model  small
 .stack  100h
 .data

 n1 db 11h
 n2 db 88h
 .code

 SWAP PROC

 MOV AX, @data
 MOV DS, AX

 MOV AL, n1
 MOV BL, n2
```

MOV n1, BL

MOV n2, AL

MOV AH, 4ch

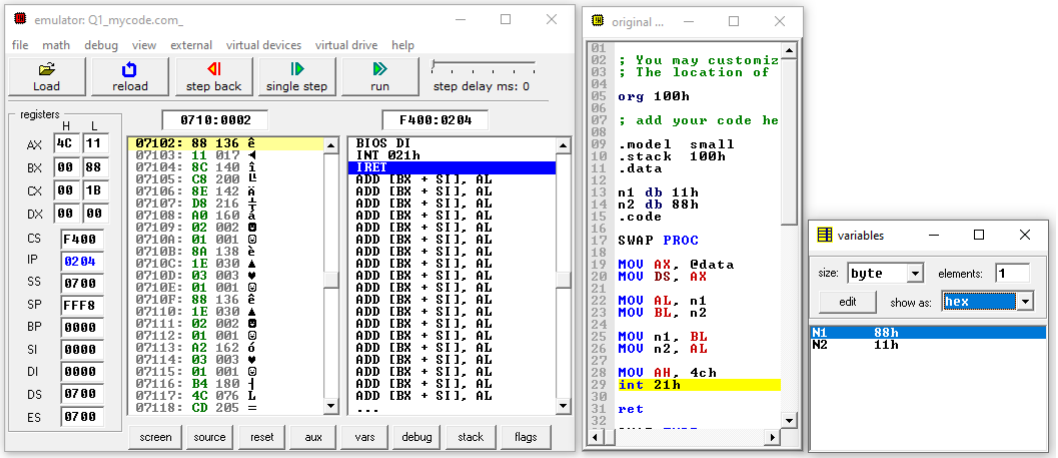
int 21h

ret

SWAP ENDP

OUTPUT:

```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt
org 100h
; add your code here
    .model  small
    .stack  100h
    .data
      n1 db 11h
      n2 db 88h
      .code
      SWAP PROC
      MOV AX, @data
      MOV DS, AX
      MOV AL, n1
      MOV BL, n2
      MOV n1, BL
      MOV n2, AL
      MOV AH, 4ch
      int 21h
ret
      SWAP ENDP
```



TASK: 3

Write a program to find the minimum number of a byte sized array and store it in a variable min, by using procedure of MIN?

SOURCE CODE:

org 100h

```
; add your code here

 include 'emu8086.inc'
.model small
.stack 100h
.data

array  db 7,3,4,6,5

.code

MIN  PROC
mov ax, @data
mov ds, ax

mov si, offset array
mov cx, 5
mov bl, [si]

loopx:
cmp [si], bl
jle update
resume:
inc si
```

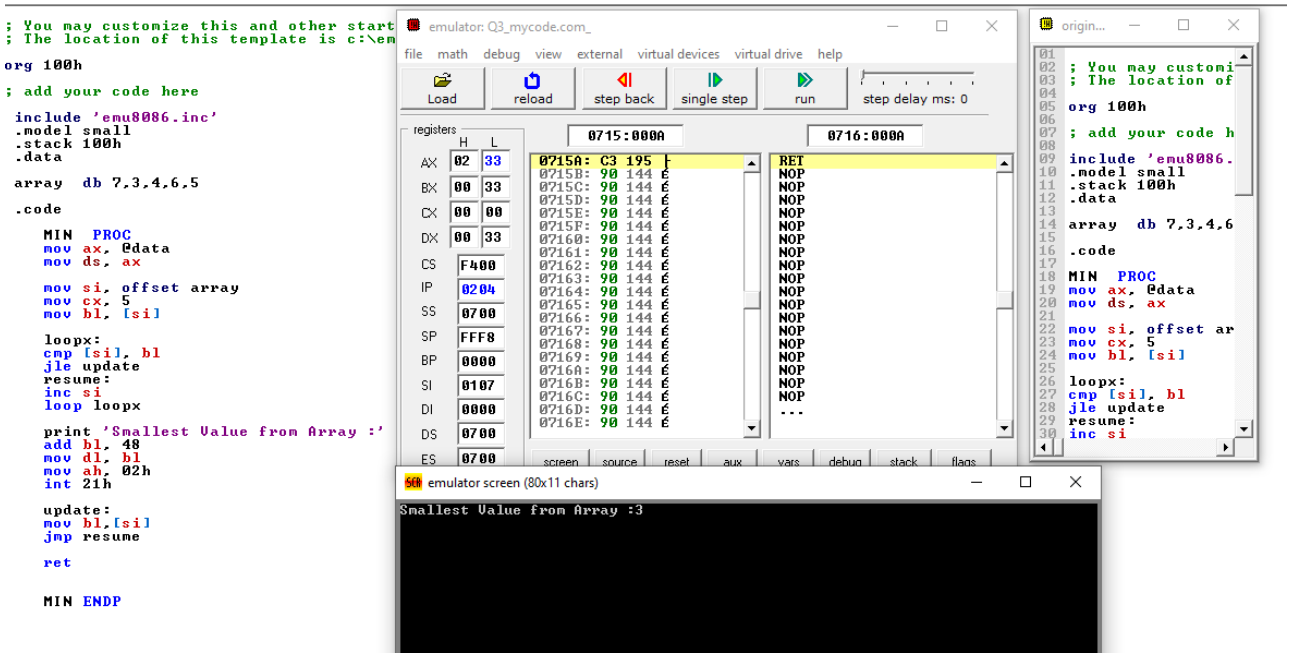
```
loop loopx


print 'Smallest Value from Array :'
 add bl, 48
 mov dl, bl
 mov ah, 02h
 int 21h

 update:
 mov bl,[si]
 jmp resume

 ret
 MIN ENDP
```

OUTPUT:



TASK: 4

Write a procedure which searches a number in an array of words. The procedure should receive the start address of the array from the SI register, the number to be searched from the BX register, and the number of elements in the array from the CX register. The procedure should return the offset of the number in DI register. If the number is not found, DI should be zero

SOURCE CODE:

```
; add your code here

include emu8086.inc

ORG 100H

; Put some data into 2500H..

MOV SI, 2500H

MOV AX, 10

MOV CX, 10


LOOP1:

MOV [SI], AX

INC SI

INC SI

DEC AX

LOOP LOOP1
```

```
; Set the parameters:
MOV SI, 2500H ; start address
MOV CX, 10    ; number of elements
MOV BX, 15    ; number to be searched
; Call the procedure:
CALL FIND
; Check whether the number is found or not found:
CMP DI, 0
JNE NUMFOUND
PRINT 'The number is not found'
RET
NUMFOUND:
PRINT 'The number is found.'
FIND PROC
L1:
MOV AX, [SI] ; Read the next element from the array
CMP AX, BX ; Compare it to the value on AX
JE FOUND ; If found, jump to FOUND
INC SI ; If not found, check the next element in the array
INC SI ; Since the array is a word-array, SI must be increased by 2
LOOP L1
; If this line is achieved, the number is not found in the array:
JMP NOTFOUND
FOUND:
```

```
MOV DI, SI ; Copy SI into DI and return
RET
NOTFOUND:
MOV DI, 0 ; Copy zero to DI and return
RET
FIND ENDP
Ret
```

OUTPUT: