VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Microprocessors and Interfacing

(CSE – 3002)

LAB EXPERIMENT– 3

Name: **Vibhu Kumar Singh**

Reg. No: **19BCE0215**

Teacher: **Mr. Konguvel E.**

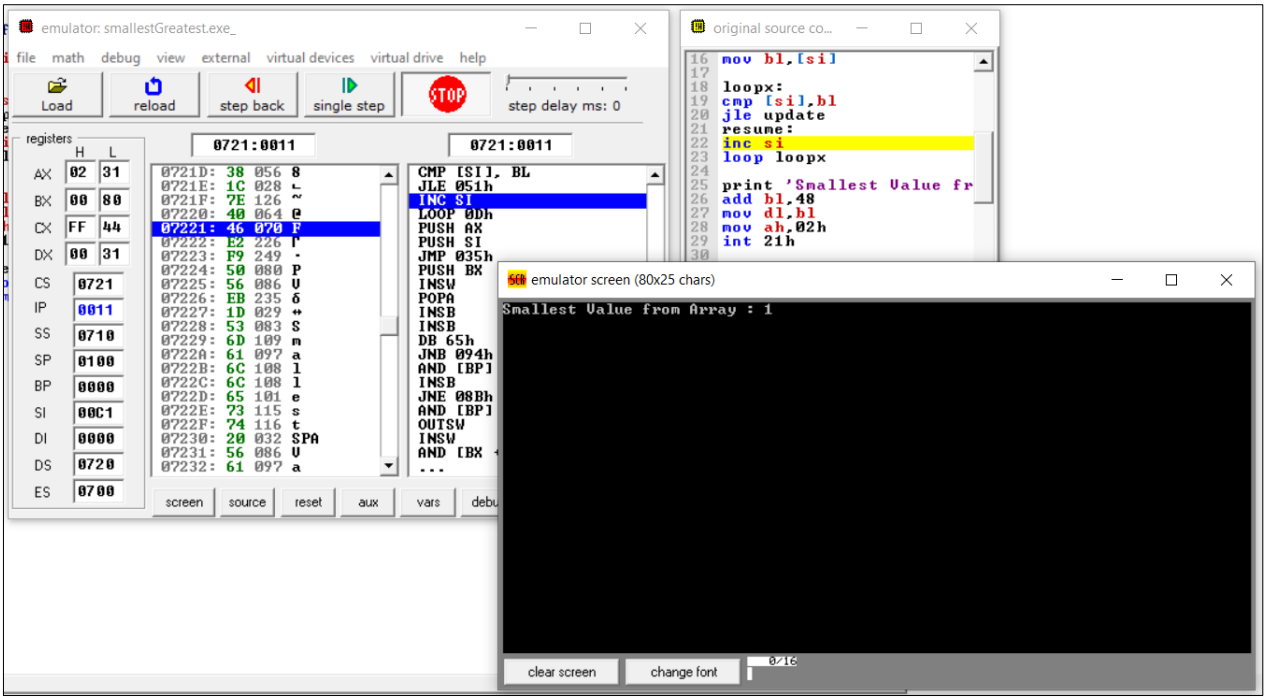
Q1) Write and execute 8086 ALP to find out largest and smallest number in 16-bit Array.

Ans 1)

Smallest Number in Array:

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

.data

    array db 7,3,4,1,5

.code
    main 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
    main endp
end main
```

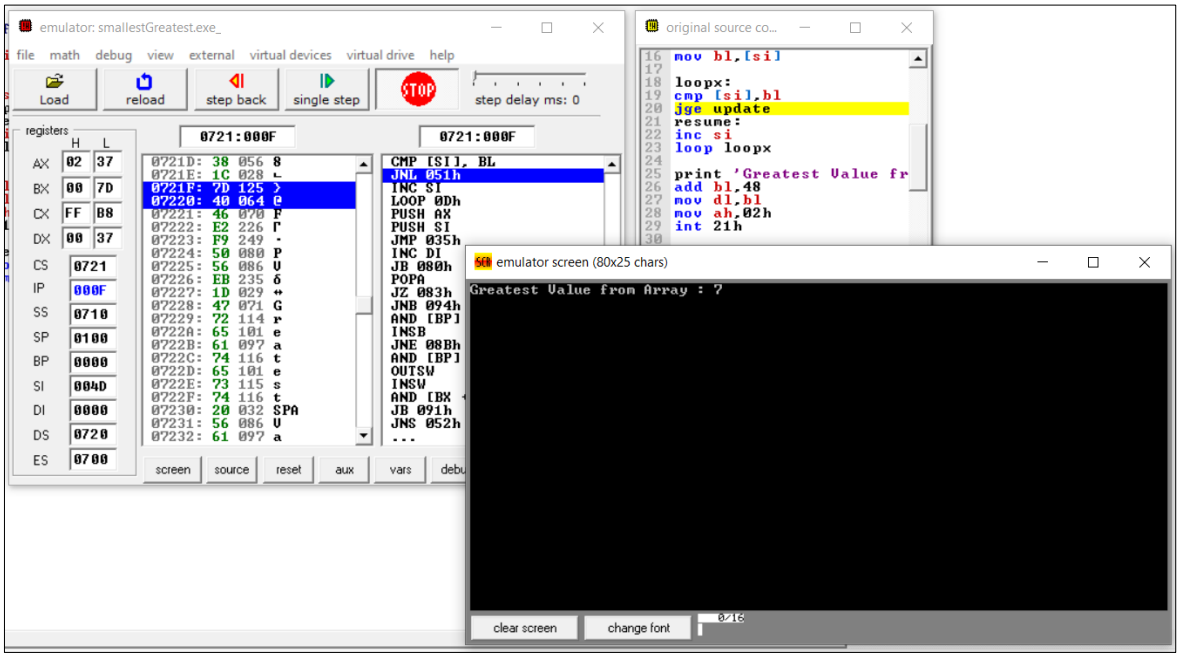
Output:



Greatest Number in Array:

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

.data

    array db 7,3,4,1,5

.code
    main proc
        mov ax,@data
        mov ds,ax

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

        loopx:
            cmp [si],bl
            jge update
            resume:
            inc si
            loop loopx

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

            update:
                mov bl,[si]
                jmp resume
    main endp
end main
```

Output:



Q2) Write and execute 8086 ALP to sort the given 16-bit numbers in Ascending and Descending order in an Array.

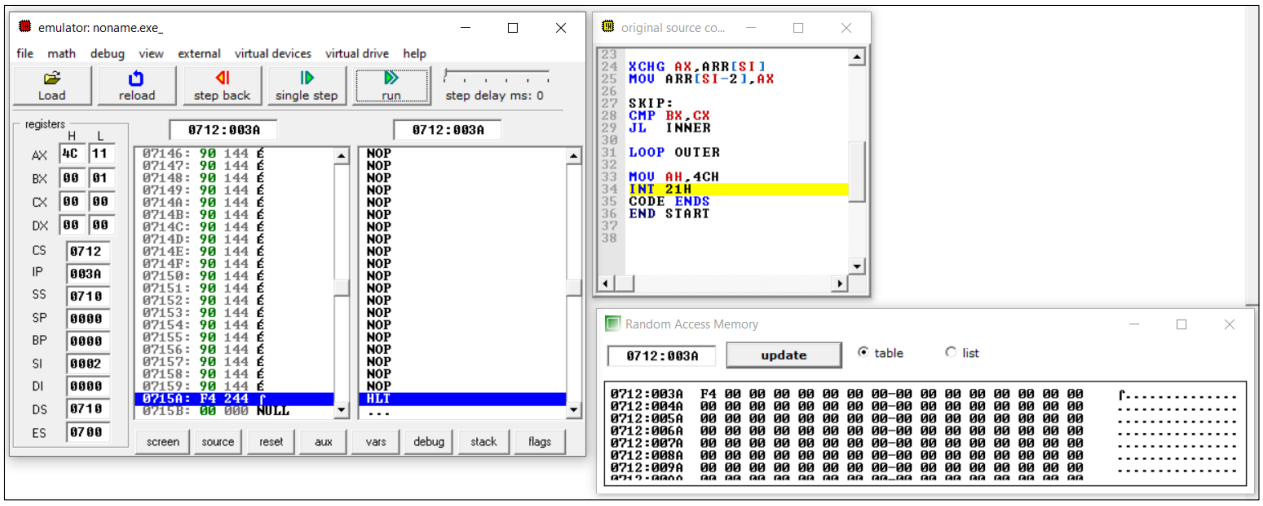
Ans 2)

Ascending Order:

```
DATA SEGMENT
     ARR DW 3333h,4444h,1111h,9999h,5555h,2222h,7777h,8888h,6666h
     LEN DW $-ARR

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

      MOV CX,(LEN/2)-1
OUTER:
      LEA SI,ARR
      MOV BX,0
INNER:
      INC BX
      MOV AX,ARR[SI]
      INC SI
      INC SI
      CMP AX,ARR[SI]
      JB SKIP

      XCHG AX,ARR[SI]
      MOV ARR[SI-2],AX
```

```
SKIP:
      CMP BX,CX
      JL  INNER

      LOOP OUTER

      MOV AH,4CH
      INT 21H
CODE ENDS
END START
```

Output:



Descending Order:

```
DATA SEGMENT
     ARR DW 30h,40h,10h,90h,50h,20h,70h,80h,60h
     LEN DW $-ARR

DATA ENDS
CODE SEGMENT
ASSUME DS:DATA CS:CODE
START:
       MOV AX,DATA
       MOV DS,AX
       MOV CX,(LEN/2)-1
       OUTER:
              LEA SI,ARR
              MOV BX,0
       INNER:
              INC BX
              MOV AX,ARR[SI]
              INC SI
              INC SI
              CMP ARR[SI],AX
              JB SKIP
              XCHG AX,ARR[SI]
              MOV ARR[SI-2],AX
       SKIP:
              CMP BX,CX
              JL  INNER
              LOOP OUTER
              MOV AH,4CH
```

```
            INT 21H
        CODE ENDS
END START
```

Output: