Name: Jigar Siddhpura **SAPID:** 60004200155

DIV: C/C2 **Branch:** Computer Engineering

POA EXPERIMENT 6

60004210155
Jigar Siddhpura

Experiment 6 : Sorting Numbers in 8086

Aim : Implement program to sort numbers in ascending / descending order.

Theory :

Bubble sort is implement in 8086 microprocessor to arrange an array of values. Here, element are compared to next adjacent element and swaped if they are in wrong order; until the array is sorted. The assembly code shows both ascending & descending sorting.

Ascending :

Here, array (segment) is defined in DATA segment & iterated through using loops. Outer loop 'L1' iterates through array & inner loop 'L2' is used for comparison. If current element is greater than next element, swapping is performed using XCHG instruction. This continues until entire array is sorted.

Descending : Here

Here, the code is almost same as above, however the primary difference is on Conditional Jump instructions. Here, swap is performed if current element is less

than the next.

Here instructions used are – MOV, LEA (Load effective address), DEC, CMP (compare), JNZ, JC, XCHG (swap operation)

Here, code segment contains instructions whereas data segment is used to define array. In code, DATA refers to 0710h address, CH refers to counter, LEA SI, N1 stores starting address of N1. Under the hood, CMP performs subtraction to detect the greater element. JC means Jump if Carry i.e. if carry flag is set, jump to particular procedure. JNZ means Jump if not zero i.e. jump if flag is non-zero.

Conclusion: Here, we perform sorting in both possible orders on 8086 microprocessor using bubble sort with the help of instruction set of 8086.

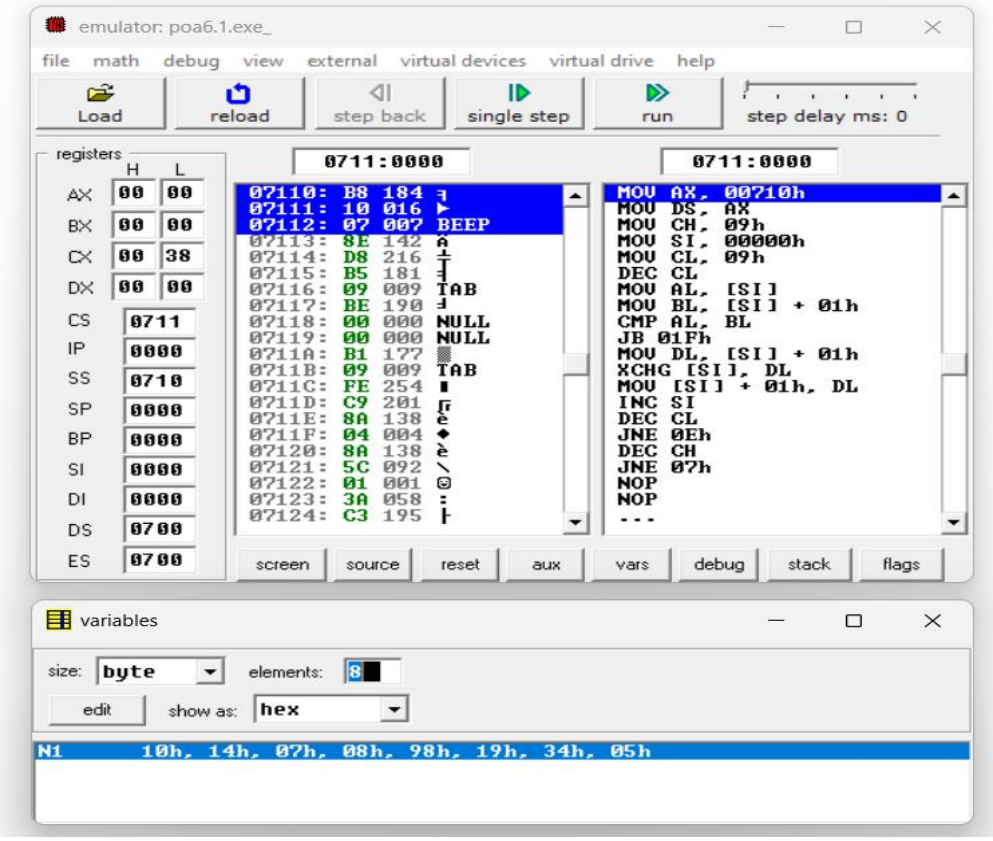
Code :

1. **Sorting in Ascending Order** :

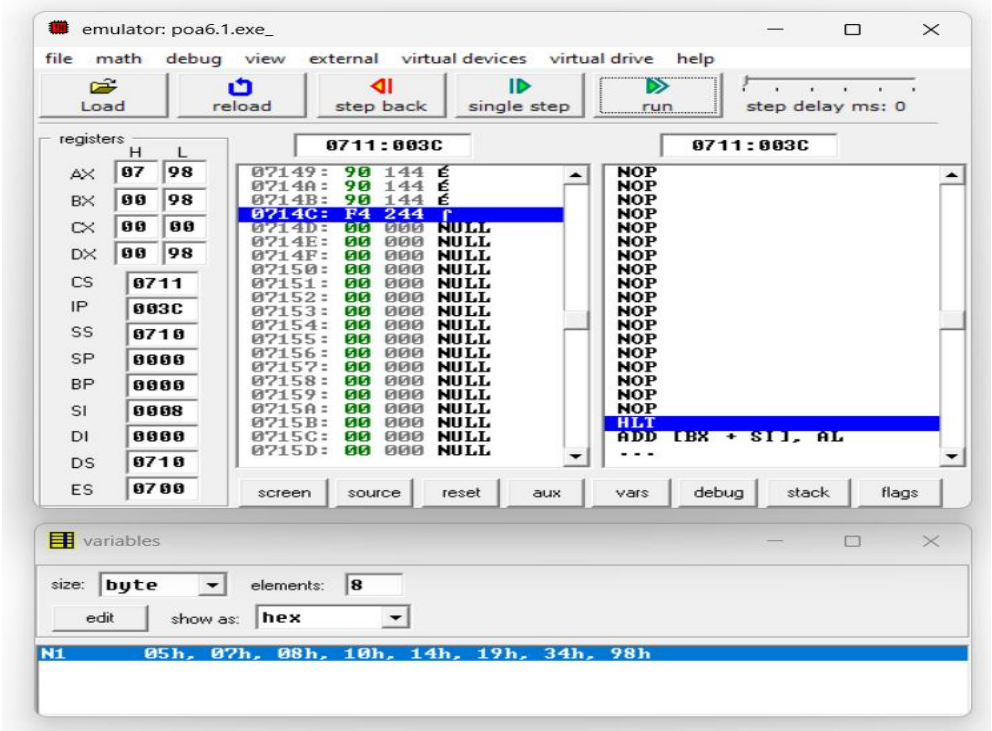
```
DATA SEGMENT
   N1 DB 10h,14h,7h,8h,98h,19h,34h,5h, 98h
   SIZE equ $ - N1
ENDS


CODE SEGMENT
   START:
   MOV AX,DATA
   MOV DS,AX

   MOV CH,SIZE

   L1:
   LEA SI,N1
   MOV CL,
   SIZE DEC CL

   L2:
   MOV AL, [SI]
   MOV BL, [SI+1]
   CMP AL,BL
   JC DOWN
   MOV DL,[SI+1]
   XCHG [SI],DL
   MOV [SI+1],DL

   DOWN:
   INC SI
   DEC CL
   JNZ L2

   DEC CH
   JNZ L1

CODE ENDS
END START
```

- Here the elements are sorted in ascending order in N1 data segment

Before Sorting -



After sorting -



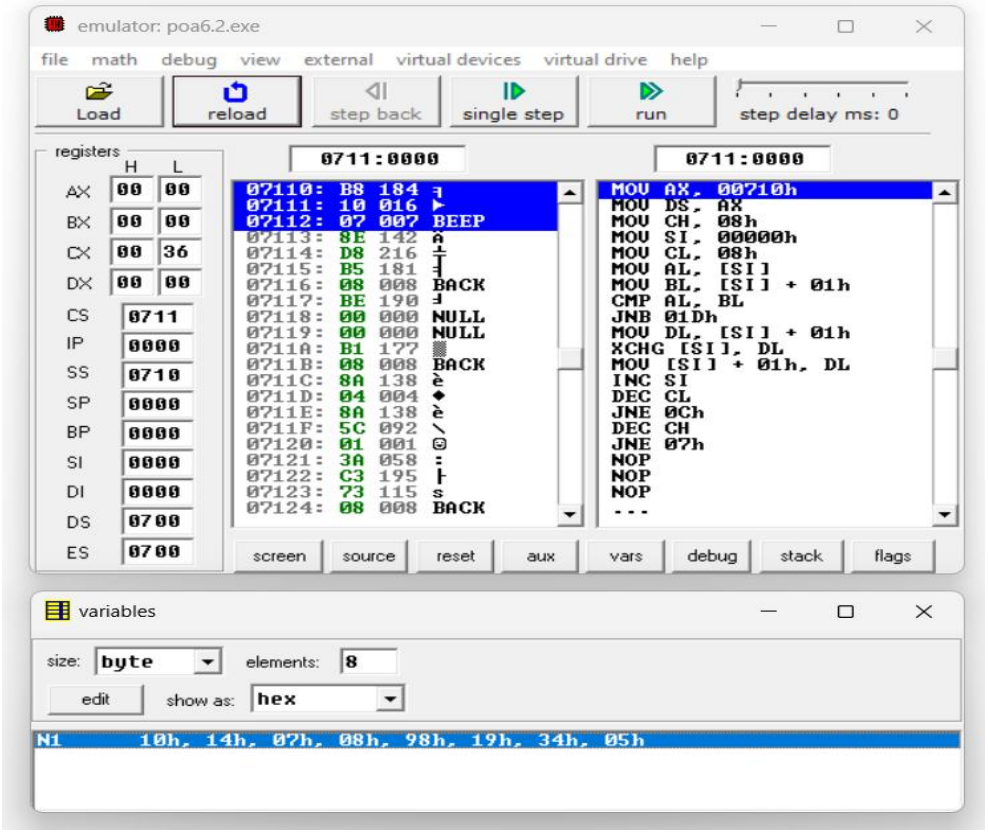
Code :

2. **Sorting in Descending Order** :

```
DATA SEGMENT
   N1 DB 10h,14h,7h,8h,98h,19h,34h,5h
   SIZE equ $ - N1
ENDS
CODE SEGMENT
   START:
   MOV AX,DATA
   MOV DS,AX

   MOV CH,SIZE

   L1:
   LEA SI,N1
   MOV CL,SIZE

   L2:
   MOV AL, [SI]
   MOV BL, [SI+1]
   CMP AL,BL
   JNC DOWN
   MOV DL,[SI+1]
   XCHG [SI],DL
   MOV [SI+1],DL

   DOWN:
   INC SI
   DEC CL
   JNZ L2
   DEC CH
   JNZ L1

CODE ENDS
END START
ret
```

- Here the elements are sorted in descending order in N1 data segment

Before Sorting -



After sorting -