Page No.
Date

Devanshu Surana
Panel C, Batch C1
PC-23, 1032210755

MAIoT Lab Assignment - 11

Problem statement:
Write an ALP to sort 8 bit numbers in ascending and descending order.

Objectives:
1. To learn an Instruction set of Pentium processors.
2. To learn displaying 2 digit hex numbers stored in an array.

Theory: Explain new instructions used eg. XCHG

The xchg instruction exchanges the content of a register with the content of another register or with the content of memory location(s). It cannot directly exchange the content of two memory locations. The source and destination must both be of the same type. (bytes or words).
The xchg (exchange data) instruction exchanges the contents of two operands.
There are 3 variants:

```
XCHG reg, reg
XCHG reg, mem
XCHG mem, reg
```

We can exchange data between registers or between registers and memory, but not from memory to memory.

```
xchg ax, bx    ; Put AX in BX and BX in AX.
xchg memory, ax     ; Put 'memory' in AX and AX
                      in 'memory'.
mem1, mem2          ; can't exchange memory locations.
```

the rules for operands in the XCHG instruction are the same as those for the MOV instruction.

Algorithm:

Write down the algorithms:

1. To sort the integers number in ascending/descending order and display.

   1. Declare numbers to be sorted
   2. Specify counter to perform sorting
   3. Select pointer
   4. Use 'CMP' instruction to compare elements and 'XCHG' if required
   5. Arrange sorted array in ascending/descending order.
   6. Use two digit display for unpacking of number
   7. Print all numbers and terminate the code.

Platform.

OS - Ubuntu 16, 64-bit

System calls used:

sys_write, sys_exit

Conclusion:

Thus the program is implemented in ALP to sort 8 bit numbers in ascending and descending order.

FAQ's

Ans 1) 1. Declare an array of required size
2. set the counter variable to a
3. set a pointer to the starting index of array.
4. Accept 2-digit hex numbers from user
5. Pack each number before storing it in array
6. Increment the counter variable and pointer to array.
7. Repeat steps 4-6 until counter variable has the value of the required no of array elements.

Ans 2) i) CMPXCHG : Used to compare the contents of a register or memory locations with the content of accumulator and swap the two values if they are not equal.

Syntax → CMPXCHG destination ,source

Ex → CMPXCHG [10011], AX

ii) BSWAP : Used to reverse the byte order of a 32-bit register or memory location.

Syntax → BSWAP register /memory location

Ex → BSWAP EAX.

iii) PUSH A : Used to pull all general-purpose registers onto the stack in the order (AX, BX, CX, DX, SI, DI, BP, SP)

Syntax → PUSHA

iv) POPA : Used to pop all general-purpose registers from the stack in order (SP, BP, DI, SI, DX, CX, BX, AX)

syntax → POPA.

Name: Devanshu Surana **Roll No.:** 23
Panel: C
Batch: C1

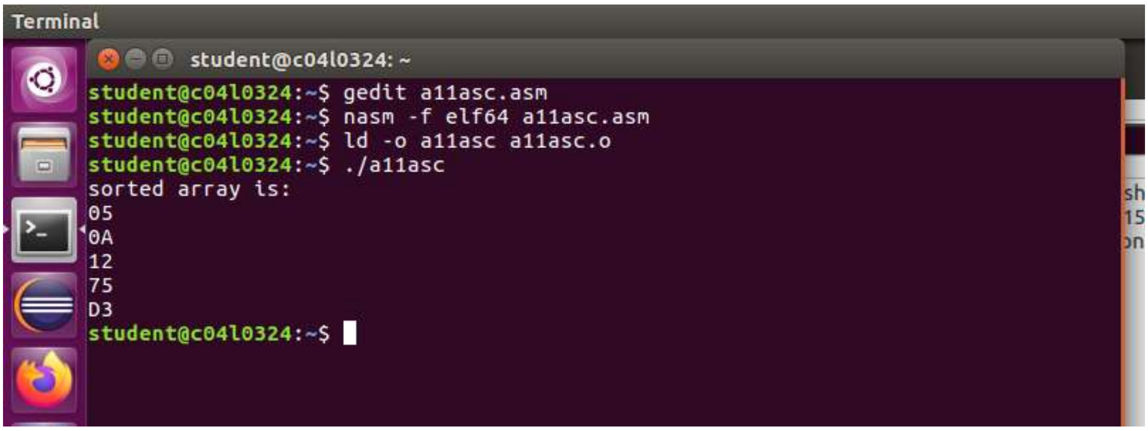
MAIoT Assignment 11

CODE:
(Ascending)

```
section .data
msg db"sorted array is: ", 10 msglen equ $-msg

arr db 05h,0Ah,75h,0D3h,12h

%macro operate 4 mov rax,%1
mov rdi,%2
mov rsi,%3

mov rdx,%4 syscall %endmacro section .bss result resb 15 section .text
global _start _start:

mov bl,5 ;;outer loop runs for n times

loop_outer:mov cl,4 ;inner loop runs n-1 times mov rsi,arr

up: mov al,byte[rsi]
cmp al,byte[rsi+1]
jbe only_inc ;no swapping
xchg al,byte[rsi+1] ;swap
mov byte[rsi],al
only_inc:inc rsi
dec cl ;decrementing inner loop
jnz up
dec bl ;decrementing outer loop
jnz loop_outer
operate 1,1,msg,msglen
mov rdi,arr ;unpacking
mov rsi,result
mov dl,10 ; for one number there are two digits disp_loop1:
mov cl,2
mov al,[rdi]
againx:
```

```
rol al,4 ;rotate by 34
mov bl,al
and al,0FH
cmp al,09H
jbe downx ;for ascending order

add al,07H downx:
add al,30H mov byte[rsi],al mov al,bl

inc rsi
dec cl
jnz againx

mov byte[rsi],0AH ;inserting enter inc rsi ;result
inc rdi
dec dl

jnz disp_loop1 operate 1,1,result,15

operate 60,0,0,0
```

OUTPUT:



```
student@c04l0324:~$ gedit a11asc.asm
student@c04l0324:~$ nasm -f elf64 a11asc.asm
student@c04l0324:~$ ld -o a11asc a11asc.o
student@c04l0324:~$ ./a11asc
sorted array is:
05
0A
12
75
D3
student@c04l0324:~$
```

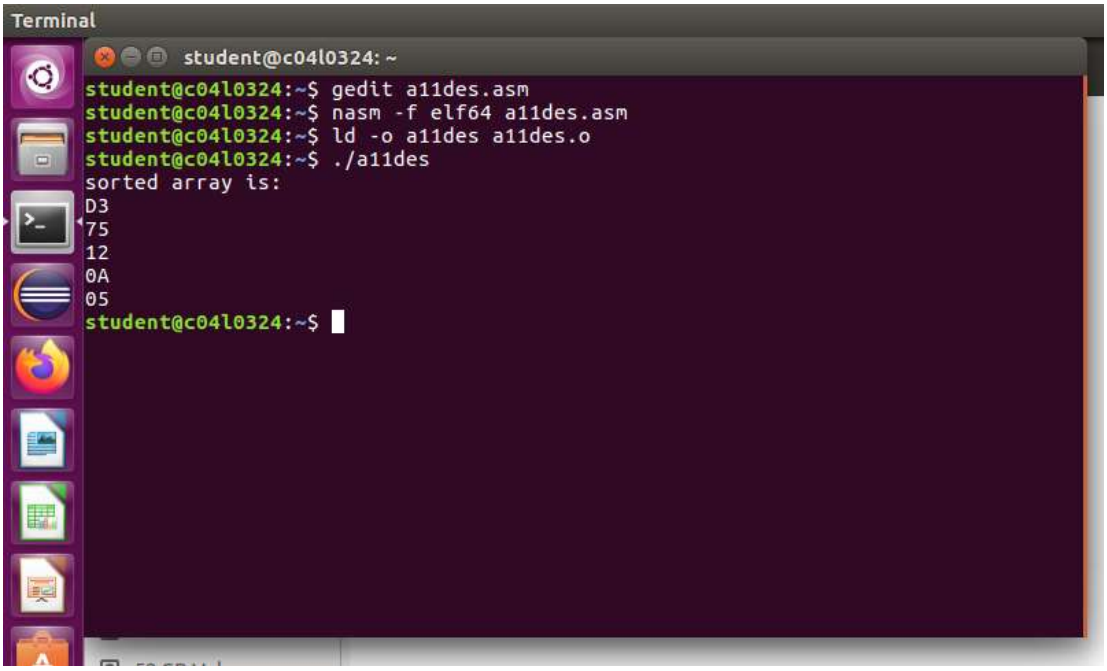
(Descending)

```
%macro operate 4 mov rax,%1
mov rdi,%2
mov rsi,%3

mov rdx,%4 syscall %endmacro
```

```
section .data
msg db"sorted array is: ", 10 msglen equ $-msg
arr db 05h,0Ah,75h,0D3h,12h

section .bss result resb 15 section .text global _start _start:

mov bl,5 ;outer loop runs for n times loop_outer:mov cl,4 ;inner loop runs
n-1 times mov rsi,arr

up: mov al,byte[rsi]
cmp al,byte[rsi+1]
jae only_inc ;no swapping
xchg al,byte[rsi+1] ;swap
mov byte[rsi],al
only_inc:inc rsi
dec cl ;decrementing inner loop
jnz up
dec bl ;decrementing outer loop
jnz loop_outer
operate 1,1,msg,msglen
mov rdi,arr ;unpacking
mov rsi,result
mov dl,10 ; for one number there are two digits disp_loop1:
mov cl,2
mov al,[rdi]
againx:
rol al,4 ;rotate by 34
mov bl,al
and al,0FH
cmp al,09H
jbe downx
add al,07H
downx:
add al,30H

;for ascending order

mov byte[rsi],al mov al,bl
inc rsi
dec cl

jnz againx
```

```
mov byte[rsi],0AH ;inserting enter inc rsi ;result
inc rdi
dec dl

jnz disp_loop1 operate 1,1,result,15

operate 60,0,0,0
```

OUTPUT:



```
student@c04l0324:~$ gedit a11des.asm
student@c04l0324:~$ nasm -f elf64 a11des.asm
student@c04l0324:~$ ld -o a11des a11des.o
student@c04l0324:~$ ./a11des
sorted array is:
D3
75
12
0A
05
student@c04l0324:~$ 
```