Roll No. - 2
Aaron Philip
1032210163

MAIOT ASSIGNMENT 11

PROBLEM STATEMENT :

Write an ALP to sort 8-bit number in ascending & descending order

OBJECTIVES :

1. To learn the instruction set of Pentium Processors.
2. To learn displaying 2 digit hex numbers stored in an array.

THEORY :

Explain new instructions used

- XCHG

The XCHG instruction is a processor instruction used in x86 architecture, including the 8086 & pentium processors. It stands for "exchange" and is used to swap the contents of two operands.

The XCHG instruction can be used with various operands, including registers & memory locations. The syntax of the instruction -

XCHG dest, source

eg) XCHG AX, BX

After this instruction, the contents of AX & BX are swapped

ALGORITHM :

① SORTING

```
up: mov  al, byte [rsi]
cmp  al, byte [rsi+1]
jbe  only_inc
xchg  al, byte [rsi+1]
mov  byte [rsi], al

only_inc : inc rsi
dec  cl
jnz  up
dec  bl
jnz  loop_outer
operate  1,1 , msg, msglen
```

② DISPLAYING SORTED NUMBERS

```
mov  rdi,arr ; unpacking
mov  rsi, result
mov  dl, 10
disp_loop1 :
mov  cl, 2 ; displaying  2 digit  number
mov  al, [rdi]
 againx:
rol  al, 4
mov  bl, al
and  al, 0Fh
cmp  al, 09h
jbe  downx
add  al, 07H
```

```
downx:
add al,30h
mov byte [rsi],al
mov al,bl
inc rsi
dec cl
jnz againx
```

PLATFORM

Assembler - NASM (Netwide Assembler)

Linker - LD, (a GNV Linker)

System calls used:

sys_write , sys_exit

INPUT :

8 bit Numbers stored in an array.

OUTPUT :

Sorted numbers in ascending & descending order

CONCLUSION :

Thus the program is implemented in an assembly language to sort 8 bit numbers in ascending & descending order

FAQs

Q1) Write down the algorithm if we need to accept numbers from the users.

A1) STEPS -

① Initialise the segment & offset values of the data segment register.

② Set up a loop to iterate through the array & read each element into a register or memory location.

③ Use the INT 21h or 16h interrupt to read input from the keyboard.

④ Store each value in a consecutive memory location in the array.

Q2) Explain the following instructions.

A2) ① CMPXCHG

CMPXCHG stands for compare & exchange, it is an instruction that provides a way to automaticaly compare a value/register & then exchange it.

② BSWAP

BSWAP stands for byte-swap, it is an instruction that provides a way to byte-swap (i.e reverse order of bytes)

③ PUSHA

Push A is an instruction that provides a way to push all the general purpose registers onto a stack.

④ POPA

POPA is an instruction that provides a way to pop all general purpose registers from the stack.

CODE :

```
%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 ; 3x5 = 15

section .text
global _start
_start:
mov bl,5 ; 5 iterations for outer loop
loop_outer:mov cl,4 ; 4 iterations for inner loop
mov rsi,arr

up:mov al,byte[rsi]
cmp al,byte[rsi+1]
jbe only_inc
xchg al,byte[rsi+1]
mov byte[rsi],al

only_inc:inc rsi
dec cl
jnz up
dec bl
jnz loop_outer
operate 1,1,msg,msglen

mov rdi,arr ; unpacking
mov rsi,result
mov dl,10 ; for one number there are 2 digits

disp_loop1:
mov cl,2 ; displaying 2 digit number
mov al,[rdi]
```

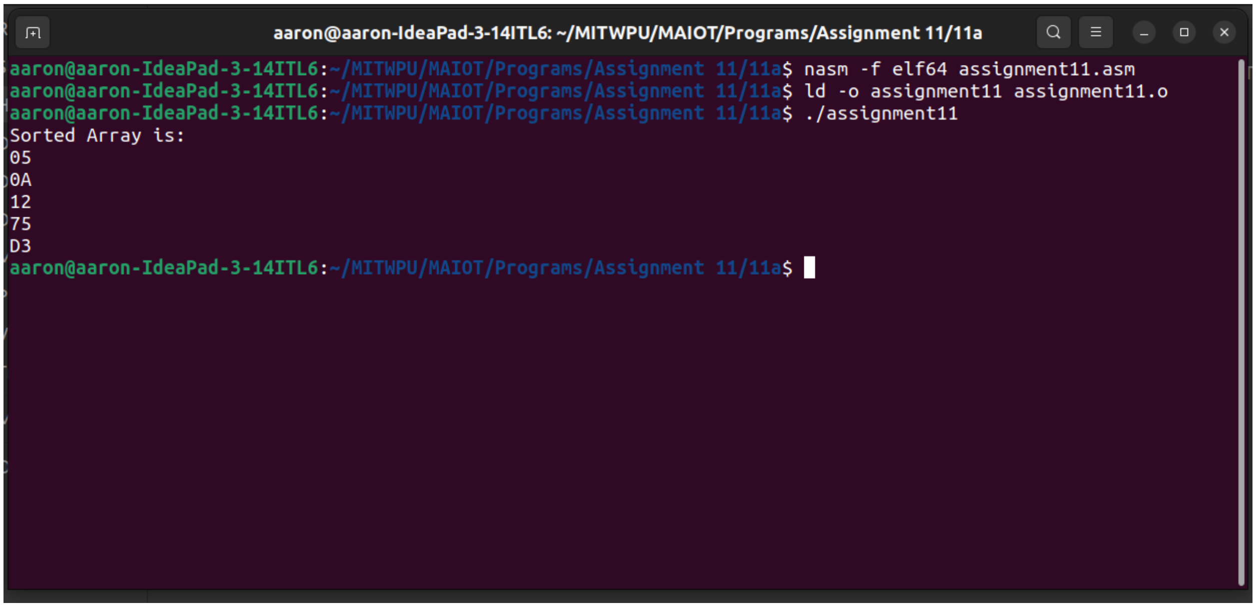
```
againx:
rol al,4
mov bl,al
and al,0Fh
cmp al,09h
jbe downx
add al,07h

downx:
add al,30h
mov byte[rsi],al
mov al,bl
inc rsi
dec cl
jnz againx


mov byte[rsi],0Ah
inc rsi
inc rdi
dec dl
jnz disp_loop1
operate 1,1,result,15

operate 60,0,0,0 ; exit syscall
```

OUTPUT :



aaron@aaron-IdeaPad-3-14ITL6: ~/MITWPU/MAIOT/Programs/Assignment 11/11a

```
aaron@aaron-IdeaPad-3-14ITL6:~/MITWPU/MAIOT/Programs/Assignment 11/11a$ nasm -f elf64 assignment11.asm
aaron@aaron-IdeaPad-3-14ITL6:~/MITWPU/MAIOT/Programs/Assignment 11/11a$ ld -o assignment11 assignment11.o
aaron@aaron-IdeaPad-3-14ITL6:~/MITWPU/MAIOT/Programs/Assignment 11/11a$ ./assignment11
Sorted Array is:
05
0A
12
75
D3
aaron@aaron-IdeaPad-3-14ITL6:~/MITWPU/MAIOT/Programs/Assignment 11/11a$
```