Assignment no.6

Name:- Gaurav Patil

Roll No. - 21259

Batch – G2

Title :- Write X86/64 ALP to convert 4-digit Hex number into its equivalent BCD number and 5-digit BCD number into its equivalent HEX number. Make your program user friendly to accept the choice from user for: (a) HEX to BCD b) BCD to HEX (c) EXIT. Display properstrings to prompt the user while accepting the input and displaying the result. (Wherever necessary, use 64-bit registers).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* C O D E \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

section .data

nl db 10,10

ln: equ $-nl

menu db 10,"Menu"

db 10,"1.Hex to BCD "

db 10,"2.BCD to Hex"

db 10,"3.Exit "

db 10

db 10,"Enter your choice: "

len: equ $-menu

m1 db 10,"Hex to BCD "

db 10,"Enter 4-digit Hex number: "

ln1: equ $-m1

m2 db 10,"BCD to Hex "

db 10,"enter 5-digit BCD number: "

ln2: equ $-m2

m3 db 10,13,"Equivalent Hex number is: "

ln3: equ $-m3

m4 db 10,13,"Equivalent BCD number is: "

ln4: equ $-m4

m5 db 10,"You entered Invalid Data!!!",10

ln5: equ $-m5

section .bss

buf resb 6

buf\_len: equ $-buf

digitcount resb 1

ans resw 1

char\_ans resb 4

%macro print 2

mov rax,1

mov rdi,1

mov rsi,%1

mov rdx,%2

syscall

%endmacro

%macro read 2

mov rax,0

mov rdi,0

mov rsi,%1

mov rdx,%2

syscall

%endmacro

%macro exit 0

print nl, ln

mov rax, 60

xor rdi, rdi

syscall

%endmacro

section .text

global \_start

\_start:

print menu,len

read buf,2

mov al,[buf]

c1:

cmp al,'1'

jne c2

call hex\_bcd

jmp \_start

c2:

cmp al,'2'

jne c3

call bcd\_hex

jmp \_start

c3:

cmp al,'3'

jne err

exit

err:

print m5,ln5

jmp \_start

hex\_bcd:

print m1,ln1

call accept\_16

mov ax,bx

mov rbx,10

back:

xor rdx,rdx

div rbx

push dx

inc byte[digitcount]

cmp rax,0h

jne back

print m4, ln4

print\_bcd:

pop dx

add dl,30h

mov [char\_ans],dl

print char\_ans,1

dec byte[digitcount]

jnz print\_bcd

ret

bcd\_hex:

print m2,ln2

read buf,buf\_len

mov rsi,buf

xor rax,rax

mov rbx,10

mov rcx,05

back1:

xor rdx,rdx

mul ebx

xor rdx,rdx

mov dl,[rsi]

sub dl,30h

add rax,rdx

inc rsi

dec rcx

jnz back1

mov [ans],ax

print m3,ln3

mov ax,[ans]

call display\_16

ret

accept\_16:

read buf,5

xor bx,bx

mov rcx,4

mov rsi,buf

next\_digit:

shl bx,04

mov al,[rsi]

cmp al,"0"

jb error

cmp al,"9"

jbe sub30

cmp al,"A"

jb error

cmp al,"F"

jbe sub37

cmp al,"a"

jb error

cmp al,"f"

jbe sub57

error:

print m5,ln5

exit

sub57:

sub al,20h

sub37:

sub al,07h

sub30:

sub al,30h

add bx,ax

inc rsi

loop next\_digit

ret

display\_16:

mov rsi,char\_ans+3

mov rcx,4

cnt:

mov rdx,0

mov rbx,16

div rbx

cmp dl,09h

jbe add30

add dl, 07h

add30:

add dl,30h

mov [rsi],dl

dec rsi

dec rcx

jnz cnt

print char\_ans,4

ret

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* O U T P U T \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*