;-------------------------------------------------------------------------------

;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 proper strings to prompt the user while accepting the input and displaying ;the result. (wherever necessary, use 64-bit registers)

;-------------------------------------------------------------------------------

%macro display 2

mov rax,1

mov rdi,1

mov rsi,%1

mov rdx,%2

syscall

%endm

%macro accept 2

mov rax,0

mov rdi,1

mov rsi,%1

mov rdx,%2

syscall

%endm

global \_start

section .data

msg2: db 10,'Equivalent BCD number is: '

len2: equ $-msg2

msg3: db 10,'Equivalent HEX number is: '

len3: equ $-msg3

menu: db 10,'\*\*\*\*\*\*Menu\*\*\*\*\*\*\*'

db 10,'1.HEX-->BCD'

db 10,'2.BCD-->HEX'

db 10,'3.Exit'

db 10,' Enter choice: '

Lmenu: equ $-menu

msgbcd: db 10,'Enter 5 digit BCD number: '

Lmsgbcd: equ $-msgbcd

msghex: db 10,'Enter 4 digit Hex number: '

Lmsghex: equ $-msghex

counter db 04h

cnt db 00

arr dw 2710h,03E8h,0064h,000Ah,0001h

bcount db 05h

count db 10h

result dw 00h

section .bss

var resb 5

choice resb 02

bcdascii resb 5

bcd resq 1

hexascii resb 4

hex resq 1

temp resb 01h

section .text

\_start:

dispmenu:

display menu,Lmenu

accept choice,02

cmp byte[choice],31h

je HEXBCD

cmp byte[choice],32h

je BCDHEX

cmp byte[choice],33h

je ENDP

HEXBCD:

call hextobcd

jmp dispmenu

BCDHEX:

call bcdtohex

jmp dispmenu

ENDP:

mov rax,60

mov rbx,00

syscall

bcdtohex:

display msgbcd,Lmsgbcd

accept var,6

mov rsi,var

mov rdi,arr

mov word[result],0

call convrsi

display msg3,len3

call dissum

ret

hextobcd:

display msghex,Lmsghex

accept hexascii,05

display msg2,len2

xor rbx,rbx

call convhex

xor rax,rax

mov rax,qword[hex]

mov rcx,0Ah

mov rbx,0

L0: xor rdx,rdx

mov rcx,0Ah

div rcx

ror rbx,04

add bl,dl

cmp rax,00h

jnz L0

ror rbx,04

mov qword[bcd],0

mov qword[bcd],rbx

call disbcd

ret

convrsi:

mov byte[bcount],05

L1:xor rcx,rcx

xor rax,rax

xor rbx,rbx

mov bl,byte[rsi]

cmp bl,39h

jbe nxt

sub bl,07h

nxt:sub bl,30h

mov ax,word[rdi]

mul bx

mov cx,word[result]

add cx,ax

mov word[result],cx

add rdi,2

inc rsi

dec byte[bcount]

jnz L1

ret

dissum:

xor rax,rax

mov byte[bcount],04

mov rsi,result

mov ax,word[result]

L2:rol ax,04

mov bx,ax

and al,0Fh

cmp al,09h

jbe nxt1

add al,07h

nxt1:add al,30h

mov byte[temp],al

display temp,01

mov ax,bx

dec byte[bcount]

jnz L2

ret

disbcd: mov byte[counter],05

mov rax,0

mov rax,qword[bcd]

L3: rol rax,04h

mov rbx,rax

and al,0Fh

cmp al,09h

jbe next

add al,07h

next:add al,30h

mov byte[temp],al

display temp,1

mov rax,rbx

dec byte[counter]

jnz L3

ret

convhex:

mov rsi,hexascii

xor rbx,rbx

mov byte[counter],04h

L4: rol rbx,4

xor rax,rax

mov al,byte[rsi]

cmp al,39h

jbe nxtt

sub al,07h

nxtt: sub al,30h

add bl,al

inc rsi

dec byte[counter]

jnz L4

mov qword[hex],rbx

ret

**OUTPUT :**

manu@ubuntu:~/MP$ nasm -felf64 A3.nasm

manu@ubuntu:~/MP$ ld -o A3 A3.o

manu@ubuntu:~/MP$ ./A3

\*\*\*\*\*\*Menu\*\*\*\*\*\*\*

1.HEX-->BCD

2.BCD-->HEX

3.Exit

Enter choice: 1

Enter 4 digit Hex number: AAAA

Equivalent BCD number is: 43690

\*\*\*\*\*\*Menu\*\*\*\*\*\*\*

1.HEX-->BCD

2.BCD-->HEX

3.Exit

Enter choice: 2

Enter 5 digit BCD number: 12345

Equivalent HEX number is: 3039

\*\*\*\*\*\*Menu\*\*\*\*\*\*\*

1.HEX-->BCD

2.BCD-->HEX

3.Exit

Enter choice: 3