; MIL Assignment No. : 3

; Assignment Name : Write 64 bit 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.

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

section .data

menumsg db 10,10,'###### Menu for Code Conversion ######'

db 10,'1: Hex to BCD'

db 10,'2: BCD to Hex'

db 10,'3: Exit'

db 10,10,'Please Enter Choice::'

menumsg\_len equ $-menumsg

wrchmsg db 10,10,'Wrong Choice Entered....Please try again!!!',10,10

wrchmsg\_len equ $-wrchmsg

hexinmsg db 10,10,'Please enter 4 digit hex number::'

hexinmsg\_len equ $-hexinmsg

bcdopmsg db 10,10,'BCD Equivalent::'

bcdopmsg\_len equ $-bcdopmsg

bcdinmsg db 10,10,'Please enter 5 digit BCD number::'

bcdinmsg\_len equ $-bcdinmsg

hexopmsg db 10,10,'Hex Equivalent::'

hexopmsg\_len equ $-hexopmsg

section .bss

numascii resb 06 ;common buffer for choice, hex and bcd input

ansbuff resb 02

dnumbuff resb 08

%macro disp 2

mov eax,04

mov ebx,01

mov ecx,%1

mov edx,%2

int 80h

%endmacro

%macro accept 2

mov eax,3

mov ebx,0

mov ecx,%1

mov edx,%2

int 80h

%endmacro

section .text

global \_start

\_start:

disp menumsg,menumsg\_len

accept numascii,2

cmp byte [numascii],'1'

jne case2

call hex2bcd

jmp \_start

case2: cmp byte [numascii],'2'

jne case3

call bcd2hex

jmp \_start

case3: cmp byte [numascii],'3'

je exit

disp wrchmsg,wrchmsg\_len

jmp \_start

exit:

mov eax,1

mov ebx,0

int 80h

hex2bcd:

disp hexinmsg,hexinmsg\_len

accept numascii,5

call atoh

mov ax,bx

mov bx,10 ;Base of Decimal No. system

mov ecx,0

h2b1: mov dx,0

div bx

push edx

inc ecx

cmp ax,0

jne h2b1

mov edi,ansbuff

h2b2: pop edx

add dl,30h

mov [edi],dl

inc edi

loop h2b2

disp bcdopmsg,bcdopmsg\_len

disp ansbuff,5

ret

;---------------------------------Logic End-------------------------

bcd2hex:

disp bcdinmsg,bcdinmsg\_len

accept numascii,6

mov esi,numascii

mov eax,0

mov ebx,0AH

mov ecx,5

b2h1: mov dl,0

mul ebx

mov dl,[esi]

sub dl,30h

add eax,edx

inc esi

DEC ECX

JNZ b2h1

mov ebx,eax

call disp32

ret

atoh:

mov bx,0

mov ecx,04

mov esi,numascii

up1:

rol bx,04

mov al,[esi]

sub al,30h

cmp al,09h

jbe skip1

sub al,07h

skip1:add bl,al

inc esi

loop up1

ret

disp32:

mov edi,dnumbuff ;point esi to buffer

mov ecx,04 ;load number of digits to display

disp:

rol bx,4;as only bx contains result so rotate number left by 4 bits

mov dl,bl ;move lower byte in dl

and dl,0fh ;mask upper digit of byte in dl

cmp dl,09h ;compare with 09h

jbe next ;if less than 39h akip adding 07 more

add dl,07h ;else add 07

next: add dl,30h ;add 30h to calculate ASCII code

mov [edi],dl ;store ASCII code in buffer

inc edi ;point to next byte

loop disp ;decrement the count of digits to display

;if not zero jump to repeat

disp hexopmsg,hexopmsg\_len

disp dnumbuff,4 ;Dispays only lower 4 digits as upper four are '0'

ret