

## MP Practical- 5

Name: Siddhesh Dilip Khairnar  
Division: B Roll No: 272028  
PRN No: 22110398

PAGE NO.:
DATE: / /

### Experiment no 5

Aim: Write X86 ALP to convert 4 digit Hex number into its equivalent BCD number HEX to BCD. Display appropriate message to prompt the user while accepting the input & Displaying the result.

Theory: Memory model dissection: The directives instruct the assembler as to how large various segment & what sort of segmentation register will be required.

Segment Directives: The directive indicate to assembler the order in which to load segment when it encounter one of these directive, it interpret all subsequent instruction as belonging to the indicated segment.

New directive used: →

- |           |   |
|-----------|---|
| 1) .MODEL | 5. .OFFSET - It inform the assembler to determine the offset / displacement of a named data item. |
| 2) .STACK |   |
| 3) .DATA  | 6. PTR - assign a specific type of variable / label.  |
| 4) .CODE  |   |

### Algorithm:

- 1) Define variable on data segment.
- 2) Display message on screen ENTER 4 DIGIT HEX NO.
- 3) Accept BCD No. from user.
- 4) Transfer 0AH as a divisor in one of the register.
- 5) Divide the no. by 0AH.
- 6) PUSH remainder in one of the register.
- 7) Increment count 1
- 8) Repeat till BCD NO. is not zero goto steps.
- 9) POP the content of remainder
- 10) Display result by calling display procedure.
- 11) Decrement count -1, till count is not zero repeat step 9 else goto step 12.
- 12) Stop.

Conclusion: Thus we successfully converted HEX number to BCD using Assembly code.

## Code:

```
global _start
```

```
_start:
```

```
section .text
```

```
; macro for system call for write
```

```
%macro disp 2
```

```
    mov rax,1
```

```
    mov rdi,1
```

```
    mov rsi,%1
```

```
    mov rdx,%2
```

```
    syscall
```

```
%endmacro
```

```
; macro for system call for read
```

```
%macro accept 2
```

```
    mov rax,0
```

```
    mov rdi,0
```

```
    mov rsi,%1
```

```
    mov rdx,%2
```

```
    syscall
```

```
%endmacro
```

```
;-----First Choice Hex to BCD-----
```

ch1:

; accept numbers

disp msg1,len1

accept num,02

call convert

mov [no.1],al

accept num,03

call convert

mov [no.2],al

disp msg2,len2

; Form ax as input

mov ah,[no.1]

mov al,[no.2]

;Point esi to predefined array in .data

mov esi,array1

; Hex to BCD conversion

I5:

mov dx,0000h

mov bx,[esi]

div bx

mov [rem],dx

mov [t1],al

push rsi

call disp\_proc

pop rsi

inc esi

```
    inc esi
    mov ax,[rem]
    dec byte[cnt]
jnz l5
```

```
    disp msg,len
```

```
;To exit program.
```

```
ch3:
```

```
    mov rax,60
    mov rdi,0
    syscall
```

```
;CONVERT procedure
```

```
convert:
```

```
    mov esi,num
    mov al,[esi]
    cmp al,39h
    jle l1
        sub al,07h
l1: sub al,30h
    rol al,04h    ;to swap number

    mov bl,al

    inc esi
```

```

        mov al,[esi]
        cmp al,39h
        jle l2
            sub al,07h
l2: sub al,30h

        add al,bl
        mov [t1],al
ret

```

```

;CONVERT2 procedure
convert2:
        mov al,[num]
        cmp al,39h
        jle l8
            sub al,07h
l8:sub al,30h
ret

```

```

;DISPLAY
procedure
disp_proc:
;for unt's place

        mov al,[t1]
        cmp al,09h
        jle l4
        add al,07h
l4:add al,30h
        mov [t2],al

```

```
        disp t2,1
ret
```

```
;DISPLAY@ procedure
```

```
display2:
```

```
        mov rsi,charans+3
        mov rcx,04h

l12: mov rdx,0
        mov rbx,10h
        div rbx
        cmp dl,09h
        jle l3
        add dl,07h
            l3:add dl,30h
            mov [rsi],dl
            dec rsi
        dec rcx
jnz l12
```

```
        mov rax,1
        mov rdi,1
        mov rsi,charans
        mov rdx,4
        syscall

ret
```

## section .data

```
msg: db "",10
len: equ $-msg
msg1: db "Enter Hex number : ",10
len1: equ $-msg1
msg2: db "BCD equivalent is : ",10
len2: equ $-msg2
msg3: db "#####MENU#####",10
      db "1.Hex to BCD.",10
      db "2.BCD to Hex.",10
      db "3.Exit.",10
len3: equ $-msg3
msg4: db "Enter your choice : ",10
len4: equ $-msg4
msg5: db "Enter BCD number : ",10
len5: equ $-msg5
msg6: db "Hex equivalent is : ",10
len6: equ $-msg6

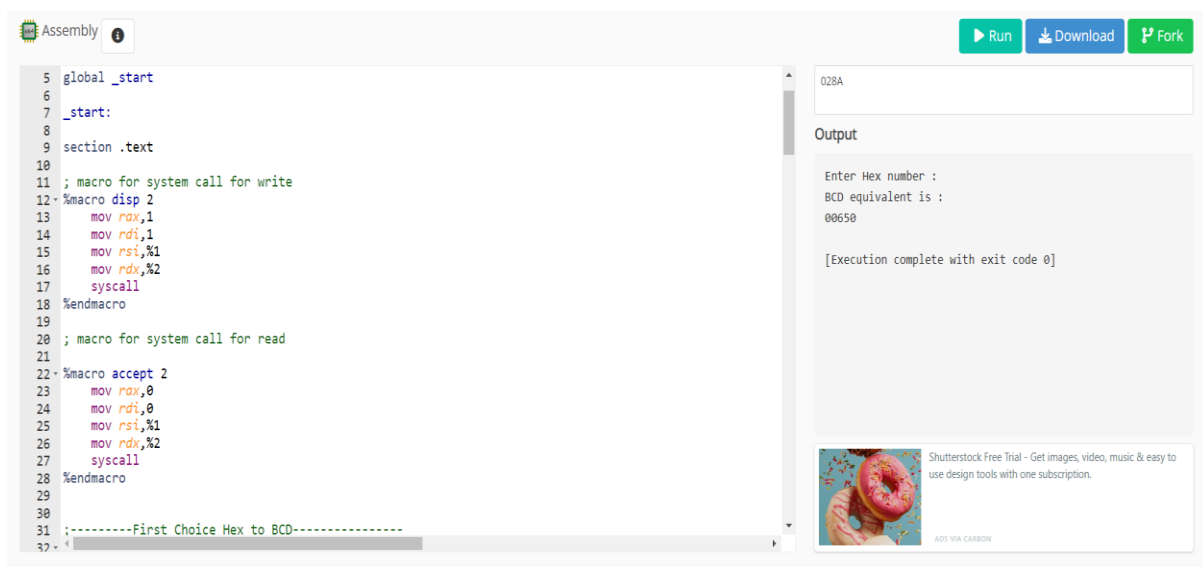
array1 dw 2710h,03E8h,0064h,000Ah,0001h
cnt db 5
cnt2 db 5
```

## section .bss

```
num resb 03
no.1 resb 02
no.2 resb 02
t1 resb 03
t2 resb 03
```

t3 resb 03  
rem resw 02  
result resw 03  
choice resb 03  
charans resb 08

## Output:



The screenshot shows an online assembly compiler interface. On the left, the assembly code is displayed with line numbers 5 through 32. The code includes a global \_start, a section .text, and two macros: %macro disp 2 and %macro accept 2. The code uses x86-64 instructions like mov, rdi, rsi, rdx, and syscall. On the right, the output section shows the execution result. The input was 028A, and the output was 00650. The execution completed with exit code 0. There is also a small advertisement for Shutterstock at the bottom right.

```
5 global _start
6
7 _start:
8
9 section .text
10
11 ; macro for system call for write
12 %macro disp 2
13     mov rax,1
14     mov rdi,1
15     mov rsi,%1
16     mov rdx,%2
17     syscall
18 %endmacro
19
20 ; macro for system call for read
21 %macro accept 2
22     mov rax,0
23     mov rdi,0
24     mov rsi,%1
25     mov rdx,%2
26     syscall
27 %endmacro
28
29
30
31 ;-----First Choice Hex to BCD-----
32
```

028A

Output

Enter Hex number :  
BCD equivalent is :  
00650

[Execution complete with exit code 0]

Shutterstock Free Trial - Get images, video, music & easy to use design tools with one subscription.

Name- Siddhesh Dilip Khairnar

Roll No.- 272028

Batch- B2