ASSIGNMENT NO 7

PROBLEM STATEMENT:

Write X86/64 ALP to detect protected mode and display the values of GDTR, LDTR, IDTR, TR and MSW Registers also identify CPU type using CPUID instruction.

SOURCE CODE:

```
section .data
        rmodemsg db 10,'processor is in real mode'
        rmsg_len:equ $-rmodemsg

        pmodemsg db 10,'processor is in protected mode'
        pmsg_len:equ $-pmodemsg

        gdtmsg db 10,'GDT Contents are::'
        gmsg_len:equ $-gdtmsg

        ldtmsg db 10,'LDT Contents are::'
        lmsg_len:equ $-ldtmsg

        idtmsg db 10,'IDT Contents are::'
        imsg_len:equ $-idtmsg

        trmsg db 10,'Task Register Contents are::'
        tmsg_len:equ $-trmsg
```

```
        mswmsg db 10,'Machine Status Word::'
        mmsg_len:equ $-mswmsg

        colmsg db ':'

        nwline db 10

section .bss
        gdt resd 1    ;base register (upper part)
           resw 1   ;limit (lower part)
        ldt resw 1
        idt resd 1     ;base register (upper part)
           resw 1    ;limit  (lower part)

        tr  resw 1           ;16 bit

        cr0_data resd 1 ;32 bit

        dnum_buff resb 04  ;lowest TR,LDTR,Limit (2 times call)

%macro disp 2
        mov eax,04
        mov ebx,01
        mov ecx,%1
        mov edx,%2
        int 80h
```

```
%endmacro

section .text
        global _start
_start:
        smsw eax   ;reading CR0

        mov [cr0_data],eax

        bt eax,1      ;checking PE bit,if 1=protectrd mode else realmode
        jc prmode
        disp rmodemsg,rmsg_len
        jmp nxt1

prmode:     disp pmodemsg,pmsg_len

nxt1:   sgdt [gdt]
          sldt [ldt]
          sidt [idt]
          str [tr]
        disp gdtmsg,gmsg_len

        mov bx,[gdt+4]            ;higher part base address
        call disp_num

        mov bx,[gdt+2]            ;lower part limit
        call disp_num
```

```
disp colmsg,1

mov bx,[gdt]                ;lower part
call disp_num

disp ldtmsg,lmsg_len
mov bx,[ldt]
call disp_num

disp idtmsg,imsg_len

mov bx,[idt+4]
call disp_num

mov bx,[idt+2]
call disp_num

disp colmsg,1

mov bx,[idt]
call disp_num

disp trmsg,tmsg_len

mov bx,[tr]
call disp_num
```

```
        disp mswmsg,mmsg_len

        mov bx,[cr0_data+2]     ;32 bit higher part
        call disp_num

        mov bx,[cr0_data]
        call disp_num

        disp nwline,1
exit: mov eax,01
         mov ebx,00
         int 80h

disp_num:
        mov esi,dnum_buff               ;point esi to buffer

        mov ecx,04          ;load no. of digits to display
up1:
        rol bx,4                        ;rotate no. left by four bits
        mov dl,bl           ;mov lower byte in dl
        add dl,0fh
        add dl,30h          ;add 30 h to calculate ASCII code
        cmp dl,39h ;compare with 39h
        jbe skip1
        add dl,07h          ;else add 07
skip1:
```

```
mov [esi],dl
inc esi
loop up1

disp dnum_buff, 4  ;display the no.from buffer'
ret
```

OUTPUT: