Microprocessor Lab Programs

Software – 1 :

Search a key element in a list of n-16 bit numbers using the binary search algorithm

;Binary search

```
data segment
        a dw 1111h,2222h,3333h,4444h,5555h
        len dw ($-a)/2          ;length of data
        key dw 2222h
        res dw ?
data ends

code segment
        assume cs:code,ds:data
        start:  mov ax,data
                mov ds,ax

                mov bx,01               ;low
                mov cx,key
                mov dx,len              ;high

        assign: cmp bx,dx               ;low>high
        ja fail

        mov ax,dx           ;mid=ax
        add ax,bx
        shr ax,01           ;mid=(low+high)/2

        mov si,ax
        cmp cx,a[si]        ;check the mid element
        je yes              ;key==mid
        ja big              ;key>mid

        dec ax
        mov dx,ax           ;high=mid-1
        jmp assign

big:    inc ax
        mov bx,ax           ;low=mid+1
        jmp assign

fail:   mov res,0000h
        jmp stop

yes:    mov res,1111h

stop:   int 3h
```

```
code ends

end start
```

Software – 2:

Write an ALP to implement

- **To read a character from the keyboard in the module(1) in a different file**
- **To display a character in the module(2) from different file**
- **Use the above two modules to read a string of character from keyboard terminated by the carriage return and print the string in the display in the next line**

;Main :

```
data segment
             msg dw 20 dup(?)
data ends


code segment
          assume cs:code, ds:data
          extrn read:far, display:far       ;inform about the far functions


          start:   mov ax,data
                   mov ds,ax


                   mov si, offset msg
                   mov cl,00
```

```
again:  call read
        cmp al,0dh              ;check for enter
        jz next
        mov [si],al             ;store char to memory
        inc si
        inc cl                  ;count
        jmp again

next:   mov dl, 0ah
        mov ah,02h
        int 21h
        mov dl, 0dh             ;new line
        mov ah,02h
        int 21h

        mov si, offset msg
disp:   mov dl,[si]
        call display
        inc si
        loop disp
        mov ah,4ch              ;terminate
        int 21h
code ends
end start
```

;Read

```
accept macro                    ;macro to read a character
      mov ah,01h
      int 21h
endm
code segment
      assume cs:code
      start:  public read
              read proc
              accept            ;call macro
              ret
              read endp
code ends
end start
```

;Write

```
write macro                     ;macro for writing a char
      mov ah,02h
      int 21h
endm
code segment
      assume cs:code
      start:  public display
              display proc
              write             ;call macro
```

```
                ret
                display endp
code ends
end start
```

Software – 3:

Write an ALP to check whether the given number is prime or not. Display appropriate message

```
;Prime Number

data segment
        a db 0ch
        msg1 db "Prime$"
        msg2 db "Not Prime$"
data ends

code segment
        assume cs:code, ds:data
        start:  mov ax,data
                mov ds,ax
                mov al,a                ; number is in ax
                mov ah,00
                mov cx,ax
                shr ax,1                ;last value ofi=n/2
```

```
            mov bl,al
            mov bh,02
     check: cmp bh,bl              ;initialize looping index
            jg prime               ; if(i>n/2)
            mov ax,cx
            div bh
            cmp ah,00              ;n%i==0
            je not_prime
            inc bh                 ; increment looping index
            jmp check
     prime: mov dx, offset msg1    ; display prime
            jmp last
 not_prime: mov dx, offset msg2    ;display not prime
     last:  mov ah, 09h
            int 21h
            mov ah,4ch
            int 21h
code ends
end start
```

<u>Software – 4:</u>

Read your name from the keyboard and display it at a specified location on a screen in front of the message “What is your name?”. You must clear the entire screen before display.

;Read and display at specified position

```
strread macro loc
        mov ah,01h                  ;read char
        int 21h
        mov loc,al
endm

data segment
        m1 db "What is your name ?$"
        arr db 20 dup(?)
data ends

code segment
        assume cs:code, ds:data
        start:  mov ax,data
                mov ds,ax

                mov si,0h

        loop1:  strread arr[si]
                inc si
                cmp al,13           ;compare with enter
                jnz loop1

                mov arr[si],"$"
                mov ah,00h      ;
                mov al,03h      ;
                int 10h         ;
                mov ah,02h      ;
                mov bh,00h      ;clearscreen

                mov dh,07h      ;x-axis
                mov dl,15h      ;y-axis
                int 10h

                lea dx,m1
                mov ah,09h
                int 21h

                mov si,0h
                lea dx,arr[si]
                mov ah,09h      ;display
                int 21h

                mov ah,4ch
                int 21h
code ends
end start
```

Software – 5:

To read a string and find the frequency of occurrence of a given character in that string

```
;Occurance of a given character

data segment
        msg1 db 10,13,"Enter any string $"
        msg2 db 10,13,"Enter any character $"
        msg3 db 10,13,"$"
        msg4 db 10,13,"No, Character not found$"
        msg5 db "character(s) found in the given string$"
        char db ?
        count db 0
        p1 label byte
        m1 db 0ffh
        l1 db ?
        p11 db 0ffh dup('$')
data ends

display macro msg                       ;display macro
        mov ah,09h
        lea dx,msg
        int 21h
endm

code segment
```

```
assume cs:code,ds:data


start:  mov ax,data

        mov ds,ax


        display msg1


        lea dx,p1

        mov ah,0ah              ;input string

        int 21h


        display msg2


        mov ah,01h              ;input character

        int 21h

        mov char,al


        display msg3


        lea si,p11

        mov cl,l1

        mov ch,00


check:  mov al,[si]

        cmp char,al             ;check key and p11[si]

        jne skip
```

```
            inc count           ;count

    skip:   inc si
            loop check

            cmp count,00
            je not_found

            display msg3

            mov dl,count        ;print count
            add dl,30h          ;converting to ascii
            mov ah,02h
            int 21h

            display msg5

            jmp exit

not_found:  display msg4

    exit:   mov ah,4ch
            int 21h
code ends
end start
```

Software – 6:

Read two strings , store them at locations str1 of data segment and str2 in extra segment check whether they are equal or not .Display appropriate message

```
;String comparison

disp macro msg                          ;display macro
            lea dx,msg
            mov ah,09h
            int 21h
endm

data segment
            m1 db 10,13,"Enter string 1:$"
            m2 db 10,13,"Enter string 2:$"
            m3 db 10,13,"Length 1:$"
            m4 db 10,13,"Length 2:$"
            m5 db 10,13,"STR1=STR2$"
            m6 db 10,13,"STR2!=STR2$"
            str1 db 80 dup('$')
            l1 db ?
data ends

extra segment
```

```
                    str2 db 80 dup('$')

                    l2 db ?

extra ends


code segment

          assume cs:code, ds:data, es:extra

          start:    mov ax,data

                    mov ds,ax

                    mov es,ax


                    disp m1

                    lea dx,str1              ;read string1

                    call read


                    disp m2


                    lea dx,str2              ;read string2

                    call read

                    mov bl,[str1+1]          ;length of string1

                    mov l1,bl

                    mov al,[str2+1]          ;length of string2

                    mov l2,al

                    cmp bl,al

                    jne strnode

                    mov ch,00

                    mov cl,l1
                    cld
                    lea si,str1+2           ;address of string1
                    lea di,str2+2           ;address of string 2
                    repe cmpsb
                    jnz strnode             ;if comparison fails
                    disp m5
                    jmp next
          strnode:disp m6
          next:     disp m3
                    mov al,l1
                    call displ
                    disp m4
                    mov al,l2
                    call displ
                    mov ah,4ch
                    int 21h

                    read proc               ;read a string
                    mov ah,0ah
                    int 21h
                    ret
                    read endp

                    displ proc              ;print length
                    aam
                    mov bx,ax
```

```
                    add bx,3030h
                    mov ah,2
                    mov dl,bh
                    int 21h
                    mov dl,bl
                    int 21h
                    ret
                    displ endp
code ends
end start
```

Software – 7:

Multiply two double precision number

;Double precision multiplication

```
data segment
          m1 dw 0003h,0000h
          m2 dw 0003h,0000h
          ans dw 4 dup(00h)
data ends
```

```
code segment
          assume cs:code, ds:data
          start:    mov ax, data
                    mov ds,ax

                    mov ax, m1                ;m1(l)*m2(l)
                    mul m2
                    mov ans,ax
                    mov ans+2,dx

                    mov ax, m1+2              ;m1(h)*m2(l)
                    mul m2
                    add ans+2,ax
                    adc ans+4,dx
                    adc ans+6,00

                    mov ax,m1                 ;m1(l)*m2(h)
                    mul m2+2
                    add ans+2,ax
                    adc ans+4,dx
                    adc ans+6,00

                    mov ax,m1+2               ;m1(h)*m2(h)
                    mul m2+2
                    add ans+4,ax
                    adc ans+6,dx
          int 3h
code ends
end start
```

Software – 8:

;Bubble sort

```
data segment
     a db 20h,70h,40h,10h,50h
     cnt equu ($-a)
data ends

code segment
     assume cs:code, ds:data
     start:   mov ax,data
              mov ds,ax

              mov bl,cnt          ;i=cnt
              dec bl              ;i--

     l1:      lea si,a
              mov cl,bl           ;j=i

     top:     mov al,[si]
```

```
                inc si
                cmp al,[si]
                jle skip                ;a[j-1]>a[j]
                xchg al,[si]
                xchg al,[si-1]

        skip:   dec cl                  ;j--
                jnz top
                dec bl
                jnz l1

                int 3h
code ends
end start
```

Software – 9:

Compute nCr using recursive procedure. Assume that n and r are non-negative integers

;nCr computation

```
data segment
         n db 6
        r db 4
        res db 0
data ends

code segment
        assume cs:code, ds:data
        start:  mov ax,data
                mov ds,ax

                mov al,n                ;al=n
                mov bl,r                ;bl=r
                call nCr
                int 3h

                nCr proc near
                cmp bl,00h              ;if r=0
                jne l1
                add res,01h
                ret

        l1:     cmp al,bl               ;if n=r
                jne l2
                add res,01
                ret

        l2:     cmp bl,01               ;if r=1
                jne l3
                add res,al
                ret

        l3:     dec al
                cmp al,bl           ;if n=r+1
                jne l4
                inc al
                add res,al
                ret

        l4:     push ax
                push bx
                call nCr            ;call nCr
                pop bx
                pop ax
                dec bx
                push ax
                push bx
                call nCr            ;call nCr-1
                pop bx
                pop ax
                ret
                nCr endp
code ends
end start
```

Software – 10:

Store n packed BCD numbers at memory at memory location BCD and find the sum. Display the result on the monitor in packed BCD form

```
;BCD addition

data segment
        bcd db 10h,10h,10h,10h,10h
        n   db 5
data ends

code segment
        assume cs:code, ds:data
        start:  mov ax,data
                mov ds,ax
                mov si, offset bcd
                mov cl,n
                mov ax,0000h
        again:  mov bl,[si]
                add al,bl
                daa                     ;convert to decimal
                jnc t1
                inc ah                  ;add carry to ah
        t1:     inc si
                loop again
                call disp
                mov ah,4ch
                int 21h

                disp proc near
                mov cl,04
                mov bh,al               ;al has the packed bcd
                shr bh,cl               ;mov the higher nibble to lower nibble
                mov bl,al
                and bl,0fh              ;mask the higher nibble
                add bx,3030h            ;convert to ASCII
                mov dl,bh               ;display higher nibble
                mov ah,02h
                int 21h
                mov dl,bl               ;display lower nibble
                int 21h
                ret
                disp endp
code ends
end start
```

Software – 11:

Generate the first n Fibonacci numbers and store all the Fibonacci numbers starting at even address

;Fibonacci numbers

```
data segment
        n db 9
        even
        fib db 20h dup(?)
data ends

code segment
        assume cs:code, ds:data
        start:  mov ax,data
                mov ds,ax
                lea si,fib
                mov cl,n                ;count=n
                mov al,0
                mov bl,1
                mov [si],al             ;fib[0]=0
                inc si
                dec cl
                mov [si],bl             ;fib[1]=1
                inc si
                dec cl
        top:    mov al,[si-1]
                add al,[si-2]
                mov [si],al             ;fib[si]=fib[si-1]+fib[si-2]
                inc si
                dec cl
                jnz top
                int 3h
code ends
end start
```

Software – 12:

To multiply two 2 digit unpacked BCD number

;Multiply unpacked BCD

```
data segment
        a db 02h,00h
        b db 02h,00h
        c db 4 dup (00h)
        c0 db 4 dup (00h)
        c1 db 4 dup (00h)
data ends

code segment
        assume cs:code,ds:data
        start:  mov ax,data
                mov ds,ax
                mov ah,00h
                mov al,a
```

```
mul b
aam
mov word ptr c0,ax

mov al,a
mul b+1
aam
add al,c0+1
aam
mov word ptr c0+1,ax

mov al,a+1
mul b
aam
mov word ptr c1,ax

mov al,a+1
mul b+1
aam
add al,c+1
aaa
mov word ptr c1+1,ax
mov al,c0
mov c,al
mov al,c0+1
mov al,c1
```

```
          aaa
          mov c+1,al
          mov al,c0+2
          adc al,c1+1
          aaa
          mov c+2,al
          mov al,c1+2
          adc al,00
          aaa
          mov c+3,al
          int 3h
code ends
end start
```

<u>Hardware – 1:</u>

; **Ring Counter**

```
data segment
    pa equ 0c800h
    pctrl equ 0c803h
data ends

code segment
        assume cs:code , ds:data
        start:  mov ax,data
                mov ds,ax

                mov al,80h
                mov dx,pctrl
                out dx,al

                mov al,01
        top:    mov dx,pa
                out dx,al

                mov bl,al
                mov ah,01h
                int 21h
                cmp al,'Q'
                je stop

                call delay

                mov al,bl
                rol al,01
                jmp top

                delay proc near
                mov si,1234h
        t2:     mov di,0ffffh
        t1:     dec di
                jnz t1
                dec si
                jnz t2
                ret
                delay endp

        stop:   mov ah,4ch
                int 21h
code ends
end start
```

Hardware - 2:

; **Program to count the number of 1's in a given number**

```
data segment
        pa equ 0c800h
        pb equ 0c801h
        pctrl equ 0c803h
data ends

code segment
        assume cs:code, ds:data
        start:  mov ax, data
                mov ds, ax

                mov al,82h
                mov dx,pctrl
                out dx,al

                mov dx,pb
                in al,dx

                mov cl,08h
                mov bl,00

        top:    shr al,01
                jnc next_rot
                inc bl
```

```
          next_rot:dec cl
                   jnz top

                   mov al,bl
                   mov dx,pa
                   out dx,al

                   mov ah,4ch
                   int 21h
code ends
end start
```

Hardware – 3:

; program BCD Counter

```
data segment
           pa equ 0c800h
           pctrl equu 0c803h
data ends

code segment
          assume cs:code,ds:data
          start:   mov ax,data
                   mov ds,ax

                   mov al,80h
                   mov dx,pctrl
                   out dx,al
```

```
        mov al,00h
top:    mov dx,pa
        out dx,al
        call delay
        add al,01h
        daa
        cmp al,20h
        jle top

bottom: sub al,1
        das
        mov dx,pa
        out dx,al
        call delay
        cmp al,00
        jz exit
        jmp bottom

exit:   mov ah,4ch
        int 21h

        delay proc
        mov bx,1234h
t:      mov cx,0ffffh
loop1:  loop loop1
        dec bx
        jnz t
        ret
        delay endp
```

```
code ends
end start
```

Hardware - 4:

; left to right rolling fassion

```
data segment
        pctrl equ 0c803h
        pc equ 0c802h
        pb  equ 0c801h
        code1 db 0ffh,0ffh,0ffh,0ffh,99h,0b0h,0a4h,0f9h,80h,0f8h,82h,92h
data ends

code segment
        assume cs:code,ds:data
        start:  mov ax,data
                mov ds,ax
                mov dx,pctrl
                mov al,80h
                out dx,al

                mov cl,12
                mov si,offset code1
        again:  call display
                call delay
                inc si
                dec cl
                jnz again
```

```
mov ah,4ch
int 21h

display proc near
mov bl,08h
mov al,[si]
top:   rol al,01

mov ch,al
mov dx,pb
out dx,al

mov al,00h
mov dx,pc
out dx,al

mov al,0ffh
mov dx,pc
out dx,al

mov al,ch
dec bl
jnz top
ret
display endp

delay proc near
push bx
mov di,0ffffh
```

```
        t:      mov bx,0ffffh
        t1:     dec bx
                jnz t1
                dec di
                jnz t
                pop bx
                ret
                delay endp
code ends
end start
```

Hardware – 5:

; Seven segment

; Flickering effect (1234 5678)

```
data segment
        portc equ 0c802h
        portb equ 0c801h
        cw equ 0c803h
        cod1 db 99h,0b0h,0a4h,0f9h
        cod2 db 80h,0f8h,82h,92h
        count db 5
data ends

code segment
        assume cs:code,ds:data
        start:  mov ax,data
                mov ds,ax
        mov al,80h
        mov dx,cw
        out dx,al

again:  lea si,cod1
        call display
        call delay
        lea si,cod2
        call display
        call delay
        dec count
        jnz again
        mov ah,4ch
        int 21h

        display proc
        mov di,0004
nextchar: mov al,[si]
        mov bh,08
nextbit: rol al,01

        mov cl,al
        mov dx,portb
        out dx,al

        mov al,00
        mov dx,portc
        out dx,al
```

```
            mov al,0ffh
            out dx,al

            mov al,cl
            dec bh
            jnz nextbit
            inc si
            dec di
            jnz nextchar
            ret
            display endp

            delay proc
            mov si,0ffffh
        t2: mov di,0bbbbh
        t1: dec di
            jnz t1
            dec si
            jnz t2
            ret
            delay endp
code ends
end start
```

Hardware - 6:

;Display even numbers in a table using Seven segment display

```
data segment
```

```
        pb equ 0cd01h
        pc equ 0cd02h
        pctrl equ 0cd03h
        tab db 1,0ah,0bh,0ch,0dh,0eh,0fh,2
        count db 8
        seg_tab db 0c0h,0ffh,0a4h,0ffh,99h,0ffh,82h,0ffh,80h,0ffh,88h,0ffh,0c6h,0ffh,86h,0ffh
data ends

code segment
        assume cs:code,ds:data
        start:  mov ax,data
                mov ds,ax

                mov al,80h
                mov dx,pctrl
                out dx,aL

                lea si,tab
                mov cl,count

        top:    mov al,[si]
                shr al,1
                jc neven

                mov bx,offset seg_tab
                mov al,[si]
                xlat

                call disp
```

```
          neven: inc si
                 dec cl
                 jnz top

                 mov ah,4ch
                 int 21h

                 disp proc near
                 mov bl,8h
          again: rol al,01

                 mov bh,al
                 mov dx,pb
                 out dx,al

                 mov al,00h
                 mov dx,pc
                 out dx,al

                 mov al,0ffh
                 out dx,al

                 mov al,bh
                 dec bl
                 jnz again
                 ret
                 disp endp
code ends
end start
```

Hardware - 7:

;stepper motor

```
data segment
          n dw 0005
          pc equ 0c802h
          pctrl equ 0c803h
data ends

code segment
          assume cs:code,ds:data
          start:   mov ax,data
                   mov ds,ax
                   mov al,80h    ; Move control word
                                 ; to al
                   mov dx,pctrl
                   out dx,al     ;Contents of al is
                                 ;moved to o/p port C
                   mov cx,n
                   mov al,0eeh

                   mov dx,pc
           t1:     out dx,al
                   call delay

                   rol al,1

                   dec cx
```

```
            jnz t1
            mov cx,n
            mov al,77h
             mov dx,pc
     t2:    out dx,al
            call delay
            ror al,1
            dec cx
            jnz t2

            mov ah,4ch
            int 21h

            delay proc near
            mov si, 0ffffh
     t4:    mov di,0ffffh
     t5:    dec di
            jnz t5
            dec si
            jnz t4
            ret
            delay endp
code ends
end start
```

Hardware - 8:

; key scan

;Display key,column,row values

```
data segment
        col db 00h
        row db 00h
        pa equ 0c800h
        pc equ 0c802h
        pctr1 equ 0c803h
        key db 00h
        newline db 0ah,0dh,'$'
data ends

code segment
        assume cs:code,ds:data
        start :  mov ax,data
                 mov ds,ax

                 mov dx,pctr1
                 mov al,90h
                 out dx,al

                 call keyscan
                 mov row,bh
                 call display

                 mov ch,row
```

```
            inc ch
            call display

            mov ch,col
            inc ch
            call display

            mov ah,4ch
            int 21h

            keyscan proc near
    repeat:mov bh,02h
            mov ch,10h
            mov bl,04h
            mov ah,00h

    nextrow: mov al,bl
            mov dx,pc
            out dx,al

            mov dx,pa
            in al,dx

            cmp al,00h
            jnz findkey

            sub ch,08h
            ror bl,01
            dec bh
```

```
                cmp bh,0ffh
                jnz nextrow
                jmp repeat

        findkey: rcr al,01h
                jc keyfound
                inc col
                inc ch
                jmp findkey
        keyfound: ret
                keyscan endp

                display proc near
                mov dl,ch
                add dl,30h
                mov ah,02h
                int 21h

                mov dx,offset newline
                mov ah,09h
                int 21h
                display endp
code ends
end start
```

\

Hardware - 9:

;Elevator

```
data segment
          pctrl equ 0c803h
          pa equ 0c800h
          pb equ 0c801h
          flor db 00,03,06,09,0e0h,0d3h,0b6h,79h
data ends

code segment
          assume cs:code,ds:data
          start:    mov ax,data
                    mov ds,ax

                    mov dx,pctrl
                    mov al,82h
                    out dx,al

                    mov bl,00
                    mov dx,pa
                    mov al,bl
                    or al,0f0h

                    out dx,al              ;elevator in the ground floor
          top:      mov dx,pb
                    in al,dx               ;check for the request
```

```
        or al,0f0h
        cmp al,0ffh
        jz top
decide:ror al,01                  ;check from ehich floor the request has come
        jnc up
        inc si
        jmp decide

up:     cmp bl,[si]               ;keep moving the ele until it reaches
        jz reset
        inc bl                    ;the requested floor
        mov al,bl
        or al,0f0h
        mov dx,pa
        out dx,al

        call delay
        jmp up

reset:  add si,04                 ;service the request
        mov al,[si]
        mov dx,pa
        out dx,al

        call delay
down:   dec bl                    ;move ele down until it reaches ground floor
        cmp bl,0ffh
        jz stop
        mov al,bl
        or al,0f0h
        mov dx,pa
        out dx,al

        call delay
        jmp down

stop:   mov ah,4ch
        int 21h

        delay proc near
        mov cx,0ffffh
t1:     mov di,0ffffh
t:      dec di
        jnz t
        loop t1
        ret
        delay endp
code ends
end start
```

Hardware - 10:

; key scan

;Input twon nos from keypad and divide these nos n

;Print quotient and remainder

```
data segment
        pa equ 0cd00h
        pc equ 0cd02h
        pctr1 equ 0cd03h
        op1 db ?
        op2 db ?
        newline db 0ah,0dh,'$'
data ends

code segment
        assume cs:code,ds:data
        start :  mov ax,data
                 mov ds,ax

                 mov dx,pctr1
                 mov al,90h
                 out dx,al

                 call keyscan
                 mov op1,ch
                 call display

                 mov ah,01h              ; DOS interrupt to wait for the next
                 int 21h                 ; character from the keyboard

                 call keyscan
                 mov op2,ch
                 call display
```

```
          mov al,op1
          mov ah,00
          div ch
          mov ch,al
          mov cl,ah             ; a copy of ah to cl
          call display

          mov ch,cl
          call display

          mov ah,4ch
          int 21h

          keyscan proc near
repeat:mov bh,02h
          mov ch,10h
          mov bl,04h

nextrow: mov al,bl
          mov dx,pc
          out dx,al

          ror bl,01
          mov dx,pa
          in al,dx
          cmp al,00h

          jnz findkey
          sub ch,08h
```

```
            dec bh
            cmp bh,0ffh
            jnz nextrow
            jmp repeat

        findkey:rcr al,01h
            jc keyfound
            inc ch
            jmp findkey
        keyfound:ret
            keyscan endp

            display proc near
            mov dl,ch
            add dl,30h
            mov ah,02h
            int 21h

            mov dx,offset newline
            mov ah,09h
            int 21h
            ret
            display endp
code ends
end start
```

<u>**Hardware - 11:**</u>

;Elevator

```
data segment
          val1 db 03
          val2 db 02
          pctrl equ 0cd03h
          pa equ 0cd00h
          pb equ 0cd01h
          flor db 00,03,06,09,0e0h,0d3h,0b6h,79h
data ends

code segment
          assume cs:code,ds:data
          start:    mov ax,data
                    mov ds,ax
                    mov dx,pctrl
                    mov al,82h

                    out dx,al
                    lea si,flor

                    mov bl,00
                    mov dx,pa
                    mov al,bl
                    or al,0f0h
                    out dx,al                    ;elevator in the ground floor

                    mov al,val1
                    mul val2
```

```
        cmp al,00
        jz move

        cmp al,03
        jz move

        cmp al,06
        jz move

        cmp al,09
        jz move
        jmp stop
move:   inc bl                  ;the requested floor
        mov bh,al
        mov al,bl
        or al,0f0h
        mov dx,pa
        out dx,al

        call delay
        cmp bl,bh
        jnz move

down:   dec bl                  ;move ele down until it reaches ground floor
        cmp bl,0ffh
        jz stop
        mov al,bl
        or al,0f0h
        mov dx,pa
```

```
                out dx,al

                call delay
                jmp down

        stop:   mov ah,4ch
                int 21h

                delay proc near
                mov cx,0ffffh
        t1:     mov di,0ffffh
        t:      dec di
                jnz t
                loop t1
                ret
                delay endp
code ends
end start
```

Hardware - 12:

; key scan

;Input a number n and display the given msg n times

```
data segment
        pa equ 0cd00h
        pc equ 0cd02h
        pctr1 equ 0cd03h
        msg db "PandBabu$"
        newline db 0ah,0dh,'$'
data ends

code segment
        assume cs:code,ds:data
        start :  mov ax,data
                 mov ds,ax

                 mov dx,pctr1
                 mov al,90h
                 out dx,al

                 call keyscan
                 mov ah,02h
                 int 21h
                 lea si,msg
        again:   call display
                 mov dx,offset newline
                 mov ah,09h
                 int 21h
                 dec ch
                 jnz again
                 mov ah,4ch
                 int 21h

                 keyscan proc near
        repeat:mov bh,02h
                 mov ch,10h
                 mov bl,04h
```

```
nextrow:mov al,bl
        mov dx,pc
        out dx,al

        ror bl,01
        mov dx,pa
        in al,dx
        cmp al,00h

        jnz findkey
        sub ch,08h

        dec bh
        cmp bh,0ffh
        jnz nextrow
        jmp repeat

findkey:rcr al,01h
        jc keyfound
        inc ch
        jmp findkey
keyfound:ret
        keyscan endp

        display proc near
        mov dx,offset msg
        mov ah,09h
        int 21h
```

```
                    ret
                    display endp

code ends
end start
```