Manual Técnico

Jose Eduardo Moran Reyes

201807455

Proyecto

DOSBox

DOSBox es un emulador que recrea un entorno similar al sistema DOS con el objetivo de poder ejecutar programas y videojuegos originalmente escritos para el sistema operativo MS-DOS de Microsoft en computadoras más modernas o en diferentes arquitecturas.

DOSBox es un emulador de CPU completo, no solo una capa de compatibilidad como DOSEmu o las máquinas con DOS virtual de Windows y OS/2, que aprovechan las posibilidades de virtualización de la familia de procesadores Intel 80386. No requiere un procesador x86 ni una copia de MS-DOS o cualquier otro DOS para ejecutarse, y puede ejecutar juegos que requieran que la CPU esté en modo real o modo protegido.

```
printPixel macro x,y, color
    pushear
    mov ax,y;y*320 +
    mov bx,320
    mul bx
    xor bx, bx
    mov bx, x
    add ax,bx
    mov di, ax
    mov dl, color
    mov es:[di],dl


    popear
endm
```

```
moverCursor macro x,y
pushear
    xor ax,ax
    xor dx,dx
    xor bx, bx
    mov ah,02h
    mov dh,x
    mov dl,y
    int 10h
popear
```

```
printVideo macro x, y, string       ;x0,y0,cadena
    pushear
    xor ax,ax
    xor dx,dx
    xor bx, bx
    xor cx, cx
    mov ah,02h
    mov dh,x
    mov dl,y
    int 10h


    mov dx,offset string
    mov ah,9h
    int 21h
```

```
    popear
endm
```

```
Sound macro hz
    push ax
    push bx
    push cx
    push dx
    xor cx,cx
    xor dx,dx

    mov al, 86h
    out 43h, al
    mov ax, 1193180 ;numero de hz
    mov bx,hz
    div bx
    out 42h, al
    mov al, ah
    out 42h, al
    in al, 61h
    or al, 00000011b
    out 61h, al
    delay 0ffffh
    in al, 61h
    and al, 11111100b
    out 61h, al

    pop dx
    pop cx
    pop bx
    pop ax

endm
```

```
verificarNumeros macro buffer, length
    LOCAL salir, omitir, ciclo
    mov siEsNumero[0],'0'
    mov cx, length
    mov si, 0
```

```
    ciclo:
        cmp buffer[si],48
        jl salir
        cmp buffer[si],57
        jg salir

        omitir:
            inc si

        loop ciclo
    mov siEsNumero[0],'1'
    salir:

endm
```

```
escribirUser:
        mov cx,7
        mov si,0
        inc di
            limpiarCadena userAux, 9
        mientras:
            mov al,bufferUsuarios[di]
            mov userAux[si],al
            inc si
            inc di
            loop mientras

        ;print userAux
        mov cx,4
        mov si,0
        limpiarCadena contraAux, 9
        mientras2:
            mov al,bufferUsuarios[di]
            mov contraAux[si],al
            inc si
            inc di
            loop mientras2
        compararCadenas userAux, usuario, igual1
        cmp igual1[0],'1'
        jne siguiente

        compararCadenas contraAux, contra, igual2
        cmp igual2[0],'1'
```

```
        jne siguiente
        mov retornoExiste[0],'1'

        siguiente:
            sub bx, 12
            cmp bx, 0
        jg escribirUser

        fin:
endm
```

```
printPelota macro pos, color

    pushear
    mov di, pos
    mov dl, color
    mov es:[di],dl
    mov es:[di+1],dl
    mov es:[di+2],dl
    mov es:[di+320],dl
    mov es:[di+321],dl
    mov es:[di+322],dl
    mov es:[di+640],dl
    mov es:[di+641],dl
    mov es:[di+642],dl

  popear
endm
```