

# Actividad 7

**Christian Geovany Muñoz Rodríguez**

**Ingeniería en computación**

**Código: 221350605**

**Seminario de Solución de Problemas de  
Traductores de lenguajes I – D09 (Lunes y  
Miércoles de 1 a 3)**

**Maestro: José Juan Meza Espinosa**

**Universidad de Guadalajara**

**Centro Universitario de Ciencias Exactas e  
Ingenierías**

**11 de marzo del 2023**



## Código:

```
org 100h
```

```
jmp inicio
```

```
direccion db "C:\EMU8086\MyBuild\christian", 0 ;direccion del  
directorio
```

```
direccion2 db "C:\EMU8086\MyBuild", 0 ;direccion del  
directorio padre
```

```
dirAux db "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$", 0
```

```
dirAux2 db "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$", 0
```

```
archivo db "C:\EMU8086\MyBuild\ christian \munoz.txt", 0  
;direccion del archivo
```

```
mensaje db "holacomoestasesperoestesbien " ;cadena de  
caracteres que se escribira en el archivo
```

```
handler dw ? ;manejador del archivo
```

```
inicio:
```

```
mov ah, 39h ;Funcion para establecer el directorio actual
```

```
mov dx, offset direccion
```

```
int 21h
```

```
    mov ah, 3Bh      ;Funcion para crear un directorio (si no  
existe) o cambiar al directorio especificado
```

```
    mov dx, offset direccion
```

```
    int 21h
```

```
    mov ah, 47h      ;Funcion para obtener el nombre del directorio  
actual
```

```
    mov dl, 0
```

```
    mov si, offset dirAux
```

```
    int 21h
```

```
    mov ah, 3ch      ;Funcion para crear un archivo
```

```
    mov cx, 0
```

```
    mov dx, offset archivo
```

```
    mov ah, 3ch
```

```
    int 21h
```

```
    mov handler, ax ;Guarda el manejador del archivo en la  
variable "handler"
```

```
    mov bx, handler ;Coloca el manejador del archivo en BX para  
usarlo en la funcion de escritura
```

```
    mov ah, 40h      ; Funcion para escribir en un archivo
```

```
    mov cx, 26       ; Longitud del mensaje que se escribir en el  
archivo
```

```
    mov dx, offset mensaje
```

```
    int 21h
```

```
mov ax, 0          ;Funcion para cerrar el archivo
```

```
mov ah, 3eh
```

```
int 21h
```

```
; Funcion para borrar el archivo
```

```
mov ah, 41h
```

```
mov dx, offset archivo
```

```
int 21h
```

```
; Funcion para salir del directorio actual
```

```
mov ah, 3bh
```

```
mov dx, offset direccion2
```

```
int 21h
```

```
; Funcion para borrar el directorio
```

```
mov ah, 3ah
```

```
mov dx, offset direccion
```

```
int 21h
```

```
; Obtener el directorio actual
```

```
mov ah, 47h
```

```
mov dl, 0
```

```
mov si, offset dirAux2
```

```
int 21h
```

```
ret
```

## Desarrollo:

Este programa es un ejemplo básico de cómo abrir, escribir y borrar un archivo de texto utilizando interrupciones del sistema DOS en el ensamblador 8086.

Primero se define la etiqueta "inicio" para comenzar la ejecución del programa. Luego se define la cadena de caracteres "direccion" que contiene la ruta del archivo de texto que se desea abrir y escribir.

```
org 100h
jmp inicio
direccion db "C:\EMU8086\MyBuild\christian", 0 ;direccion del directorio
direccion2 db "C:\EMU8086\MyBuild", 0 ;direccion del directorio padre
dirAux db "$$$$$$$$$$$$$$$$$$$$$$$$$$$$", 0
dirAux2 db "$$$$$$$$$$$$$$$$$$$$$$$$$$$$", 0
archivo db "C:\EMU8086\MyBuild\christian\munoz.txt", 0 ;direccion del archivo
mensaje db "holacomoestasesperooestesbien" ;cadena de caracteres que se escribira en el archivo
handler dw ? ;manejador del archivo

inicio:
mov ah, 39h ;Funcion para establecer el directorio actual
mov dx, offset direccion
int 21h

mov ah, 3Bh ;Funcion para crear un directorio (si no existe) o cambiar al directorio especificado
mov dx, offset direccion
int 21h
```

La siguiente instrucción es un salto incondicional a la etiqueta "inicio". Luego se definen las cadenas de caracteres "direccion2", "dirAux" y "dirAux2" que serán utilizadas más adelante en el programa.

A continuación se utiliza la interrupción 21h con la función 39h para establecer el directorio actual. Luego se utiliza la interrupción 21h con la función 3Bh para crear un directorio (si no existe) o cambiar al directorio especificado.

```
inicio:
mov ah, 39h ;Funcion para establecer el directorio actual
mov dx, offset direccion
int 21h

mov ah, 3Bh ;Funcion para crear un directorio (si no existe) o cambiar al directorio especificado
mov dx, offset direccion
int 21h
```

Nombre	Fecha de modificación	Tipo	Tamaño
christian	11/03/2023 09:31 p. m.	Carpeta de archivos	
Act4.com_	08/03/2023 11:03 p. m.	assembly source c...	1 KB
Act4.com_~asm	08/03/2023 11:03 p. m.	Archivo ~ASM	2 KB
Act4.com_debug	08/03/2023 11:03 p. m.	Archivo DEBUG	2 KB
Act4.com_list	08/03/2023 11:03 p. m.	Archivo LIST	7 KB
Act4.com_symbol	08/03/2023 11:03 p. m.	Archivo SYMBOL	2 KB
Act7.com_	11/03/2023 09:31 p. m.	assembly source c...	1 KB
Act7.com_~asm	11/03/2023 09:31 p. m.	Archivo ~ASM	2 KB
Act7.com_debug	11/03/2023 09:31 p. m.	Archivo DEBUG	3 KB
Act7.com_list	11/03/2023 09:31 p. m.	Archivo LIST	7 KB
Act7.com_symbol	11/03/2023 09:31 p. m.	Archivo SYMBOL	2 KB
file.txt	11/03/2023 08:30 p. m.	Documento de te...	1 KB
logicos.com_	10/02/2023 12:11 p. m.	assembly source c...	1 KB
logicos.com_~asm	10/02/2023 12:11 p. m.	Archivo ~ASM	2 KB
logicos.com_debug	10/02/2023 12:11 p. m.	Archivo DEBUG	2 KB
logicos.com_list	10/02/2023 12:11 p. m.	Archivo LIST	6 KB
logicos.com_symbol	10/02/2023 12:11 p. m.	Archivo SYMBOL	1 KB
noname.com_~asm	11/03/2023 09:02 p. m.	Archivo ~ASM	2 KB
noname.com_list	11/03/2023 09:02 p. m.	Archivo LIST	5 KB

Después se utiliza la interrupción 21h con la función 47h para obtener el nombre del directorio actual.

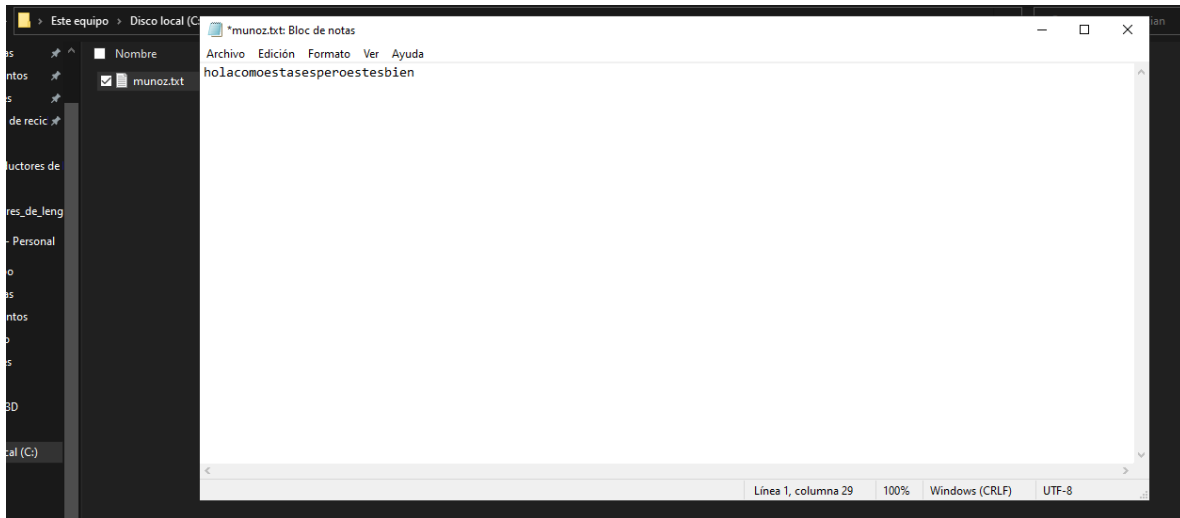
```
mov ah, 47h      ;Funcion para obtener el nombre del directorio actual
mov dl, 0
mov si, offset dirAux
int 21h
```

Se utiliza la interrupción 21h con la función 3ch para crear el archivo de texto en caso de que no exista. Se define la cadena de caracteres "archivo" que contiene la ruta y el nombre del archivo de texto que se va a escribir.

Nombre	Fecha de modificación	Tipo	Tamaño
munoz.txt	08/03/2023 11:03 p. m.	Documento de te...	0 KB

Luego se utiliza la instrucción "mov bx, handler" para mover el handle del archivo a la variable "bx". Después se utiliza la interrupción 21h con la función 40h para escribir en el archivo. La cadena de caracteres "mensaje" es escrita en el archivo.

```
mov handler, ax ;Guarda el manejador del archivo en la variable "handler"
mov bx, handler ;Coloca el manejador del archivo en BX para usarlo en la funcion de escritura
mov ah, 40h     ; Funcion para escribir en un archivo
mov cx, 26      ; Longitud del mensaje que se escribir en el archivo
mov dx, offset mensaje
int 21h
```

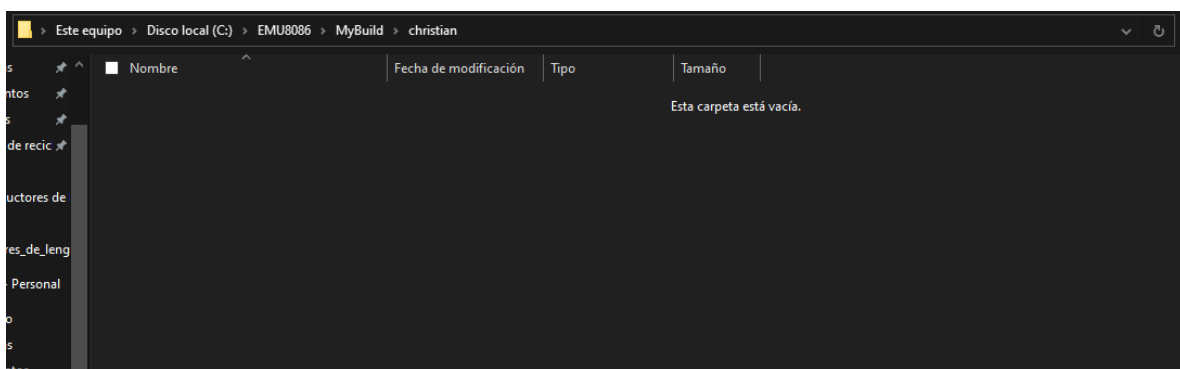


Finalmente, se utiliza la interrupción 21h con la función 3eh para cerrar el archivo.

```
mov ax, 0           ;Funcion para cerrar el archivo
mov ah, 3eh
int 21h
```

La función "borrar el archivo" se encarga de eliminar el archivo "munoz.txt" del directorio actual.

```
; Funcion para borrar el archivo
mov ah, 41h
mov dx, offset archivo
int 21h
```



La función "salir del directorio actual" se encarga de salir del directorio actual y volver al directorio anterior.

```

; Funcion para salir del directorio actual
mov ah, 3bh
mov dx, offset direccion2
int 21h

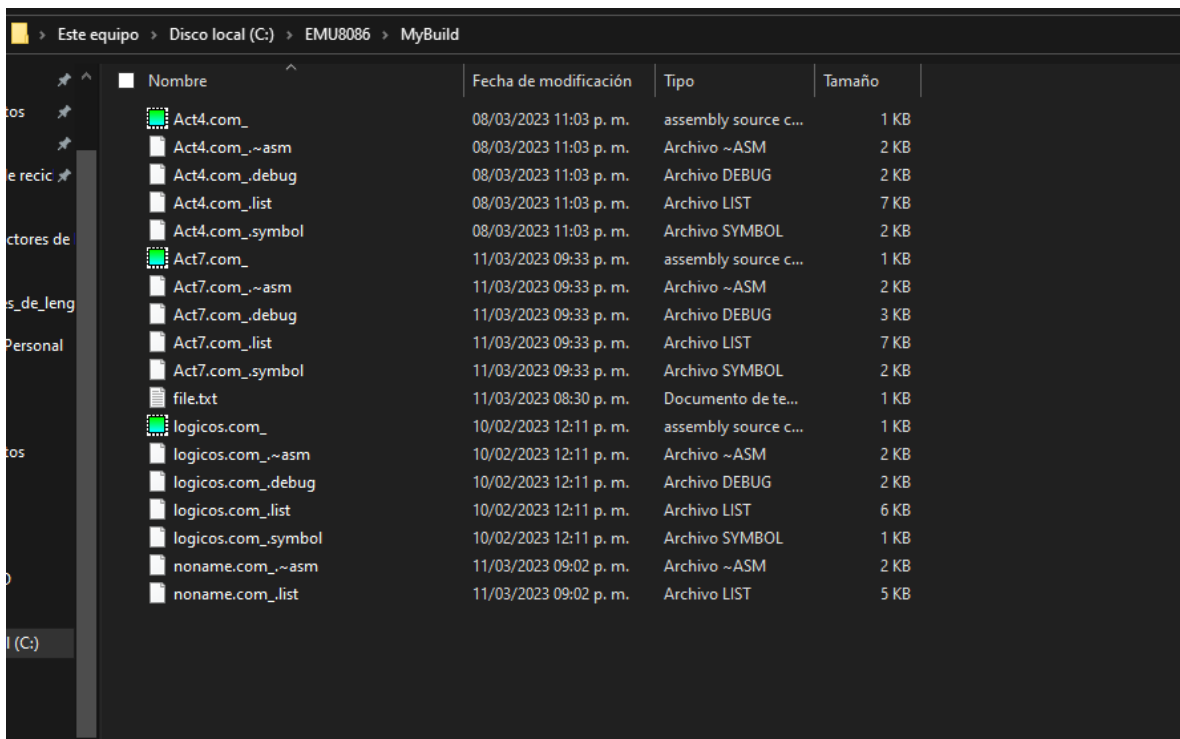
```

La función "borrar el directorio" se encarga de eliminar el directorio actual y todo su contenido.

```

; Funcion para borrar el directorio
mov ah, 3ah
mov dx, offset direccion
int 21h

```



Nombre	Fecha de modificación	Tipo	Tamaño
Act4.com_	08/03/2023 11:03 p. m.	assembly source c...	1 KB
Act4.com_~asm	08/03/2023 11:03 p. m.	Archivo ~ASM	2 KB
Act4.com_debug	08/03/2023 11:03 p. m.	Archivo DEBUG	2 KB
Act4.com_list	08/03/2023 11:03 p. m.	Archivo LIST	7 KB
Act4.com_symbol	08/03/2023 11:03 p. m.	Archivo SYMBOL	2 KB
Act7.com_	11/03/2023 09:33 p. m.	assembly source c...	1 KB
Act7.com_~asm	11/03/2023 09:33 p. m.	Archivo ~ASM	2 KB
Act7.com_debug	11/03/2023 09:33 p. m.	Archivo DEBUG	3 KB
Act7.com_list	11/03/2023 09:33 p. m.	Archivo LIST	7 KB
Act7.com_symbol	11/03/2023 09:33 p. m.	Archivo SYMBOL	2 KB
file.txt	11/03/2023 08:30 p. m.	Documento de te...	1 KB
logicos.com_	10/02/2023 12:11 p. m.	assembly source c...	1 KB
logicos.com_~asm	10/02/2023 12:11 p. m.	Archivo ~ASM	2 KB
logicos.com_debug	10/02/2023 12:11 p. m.	Archivo DEBUG	2 KB
logicos.com_list	10/02/2023 12:11 p. m.	Archivo LIST	6 KB
logicos.com_symbol	10/02/2023 12:11 p. m.	Archivo SYMBOL	1 KB
noname.com_~asm	11/03/2023 09:02 p. m.	Archivo ~ASM	2 KB
noname.com_list	11/03/2023 09:02 p. m.	Archivo LIST	5 KB

Finalmente, el programa obtiene el directorio actual y termina su ejecución.

## Conclusiones:

Este programa demuestra cómo se pueden utilizar las interrupciones de DOS para interactuar con el sistema de archivos en un entorno de programación de nivel de sistema. Aunque el uso de estas interrupciones está limitado a sistemas operativos basados en DOS o en Windows 9x, todavía se pueden encontrar aplicaciones prácticas para estas técnicas en entornos de programación de nivel de sistema, como el desarrollo de controladores de dispositivos o aplicaciones de bajo nivel que requieren acceso directo al hardware.



En general, este programa es un buen ejemplo de cómo utilizar las interrupciones de DOS para realizar operaciones de archivos y directorios. Sin embargo, es importante tener en cuenta que las operaciones de archivos y directorios son específicas del sistema operativo y pueden variar en otros sistemas operativos. Por lo tanto, es importante asegurarse de que cualquier programa que realice operaciones de archivos y directorios sea compatible con el sistema operativo en el que se va a ejecutar.

## **Bibliografía:**

Brey, B. B. (2006). *Microprocesadores Intel : 8086/8088, 80186/80188, 80286, 80386 y 80486, Pentium, procesador Pentium Pro, Pentium II, Pentium III y Pentium 4: arquitectura, programación e interfaces*. Pearson Educación.