

# Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías



División de Tecnologías para la Integración CiberHumana

**Ingeniería en Computación**

**Programación de Bajo Nivel**

D02 - IL358 - 209850

## 5. Manipulación de Archivos, Directorios y Fechas

**Profesor:** José Juan Meza Espinoza

**Alumno:** Alan Yahir Juárez Rubio

**Código:** 218517809

**Correo:** alan.juarez5178@alumnos.udg.mx

Este documento ha sido elaborado con fines estudiantiles.  
La información presentada puede contener errores.

21 de octubre de 2024



## Índice

<b>1. Introducción</b>	<b>4</b>
<b>2. Implementación</b>	<b>5</b>
2.1. Códigos . . . . .	5
2.2. Ejecución del Programa . . . . .	8
<b>3. Conclusión</b>	<b>11</b>



## Índice de figuras

1.	Ejecución del Programa: Muestreo de la ruta actual al entrar y salir de la carpeta creada .	8
2.	Valor almacenado en la variable <code>timestamp</code> . . . . .	9
3.	Impresión de la “fecha actual del sistema” ( <code>timestamp</code> ) almacenada en el archivo creado	10
4.	Valor obtenido al implementar el código 2, valor que debería ser almacenado en el archivo <code>txt</code> . . . . .	10



## Índice de códigos

1. Manipulación de Directorios: Crea un directorio, añade un archivo en él y se le guarda la fecha actual del sistema . . . . . 5
2. Conversión de `timestamp` a la fecha y hora actual del sistema . . . . . 7



## 5. Manipulación de Archivos, Directorios y Fechas

### 1. Introducción

La manipulación de directorios en ensamblador es una práctica fundamental ya que prácticamente todos los archivos se organizan en estos. La manipulación correcta de directorios y archivos te permite tener control acerca del cómo la información es almacenada, recuperada y manipulada. A continuación veremos un código con algunas de las operaciones que se pueden efectuar en archivos y directorios.

## 2. Implementación

### 2.1. Códigos

```
1 section .data
2     ; Constantes
3     BUFFER_SIZE equ 256
4
5     ; Variables
6     parent_dir DB "..", 0
7     dir DB "alan", 0
8     file DB "juarez.txt", 0
9
10    msg_1 DB "El directorio actual es: ", 0xA, 0
11    len_1 equ $-msg_1
12
13    new_line DB 0xA, 0xA, 0
14
15 section .bss
16     buffer RESB BUFFER_SIZE ; Buffer to hold the cwd
17     wd RESB BUFFER_SIZE ; Buffer to hold the cwd
18
19     timestamp RESD 1
20     file_descriptor RESD 1
21
22 section .text
23     global _start
24
25 _start:
26     ; Cleaning of registers (initializing them at 0)
27     XOR eax, eax
28     XOR ebx, ebx
29     XOR ecx, ecx
30     XOR edx, edx
31
32 get_cwd:
33     ; Get current working directory
34     MOV eax, 183 ; sys_getcwd() syscall number
35     MOV ebx, wd
36     MOV ecx, BUFFER_SIZE
37     INT 0x80
38
39 create_dir:
40     MOV eax, 39 ; sys_mkdir() syscall number
41     MOV ebx, dir
42     MOV cx, 0b110_100_000 ; RW permissions for Owner (6)
43                           ; R permissions for Group (4)
44                           ; No permissions for Others (0)
45     INT 0x80
46
47 change_dir:
48     MOV eax, 12 ; sys_chdir() syscall number
49     MOV ebx, dir ; Pointer to the directory name
50     INT 0x80
51
52 get_new_cwd:
53     ; Get current working directory
54     MOV eax, 183 ; sys_getcwd() syscall number
55     MOV ebx, buffer
56     MOV ecx, BUFFER_SIZE
57     INT 0x80
```



```
58
59 print_new_cwd:
60     ; Print msg_1
61     MOV eax, 4                ; sys_write() syscall number
62     MOV ebx, 1                ; File descriptor 1 (stdout)
63     MOV ecx, msg_1
64     MOV edx, len_1
65     INT 0x80
66
67     ; Get cwd
68     MOV eax, 4                ; sys_write() syscall number
69     MOV ebx, 1                ; File descriptor 1 (stdout)
70     MOV ecx, buffer
71     MOV edx, BUFFER_SIZE
72     INT 0x80
73
74     ; new_line
75     MOV eax, 4                ; sys_write() syscall number
76     MOV ebx, 1                ; File descriptor 1 (stdout)
77     MOV ecx, new_line
78     MOV edx, 2
79     INT 0x80
80
81 create_file:
82     MOV eax, 8                ; sys_creat() syscall number
83     MOV ebx, file
84     MOV ecx, 0b110_100_000    ; RW permissions for Owner (6)
85                                ; R permissions for Group (4)
86                                ; No permissions for Others (0)
87     INT 0x80                  ; Missing INT 0x80 added here
88
89 open_file:
90     MOV eax, 5                ; sys_open() syscall number
91     MOV ebx, file
92     MOV ecx, 1                ; Write access
93     MOV edx, 0b110_100_000    ; RW permissions for Owner (6)
94                                ; R permissions for Group (4)
95                                ; No permissions for Others (0)
96     INT 0x80
97
98     MOV [file_descriptor], eax
99
100 ; FIX: Convert the timestamp to a date and save it on file instead
101 ; NOTE: I couldn't convert it because it's too complex. I leave it as timestamp
102 ; HACK: I may use a C library. Instead of convert it manually
103
104 read_date:
105     MOV eax, 13                ; sys_time() syscall number
106     XOR ebx, ebx              ; Null pointer to store in eax
107     INT 0x80
108
109     MOV [timestamp], eax
110
111
112 write_file:
113     MOV eax, 4                ; sys_write() syscall number
114     MOV ebx, [file_descriptor]
115     MOV ecx, timestamp         ; Pass the address of the timestamp
116     MOV edx, 4                ; Writing 4 bytes for the timestamp
117     INT 0x80
118
```

```
119 close_file:
120     MOV eax, 6                ; sys_close() syscall number
121     MOV ebx, [file_descriptor]
122     INT 0x80
123
124 change_parent_dir:
125     MOV eax, 12               ; sys_chdir() syscall number
126     MOV ebx, parent_dir      ; Pointer to the directory name
127     INT 0x80
128
129 print_cwd:
130     ; Print msg_1
131     MOV eax, 4                ; sys_write() syscall number
132     MOV ebx, 1                ; File descriptor 1 (stdout)
133     MOV ecx, msg_1
134     MOV edx, len_1
135     INT 0x80
136
137     ; Get cwd
138     MOV eax, 4                ; sys_write() syscall number
139     MOV ebx, 1                ; File descriptor 1 (stdout)
140     MOV ecx, wd
141     MOV edx, BUFFER_SIZE
142     INT 0x80
143
144     ; new_line
145     MOV eax, 4                ; sys_write() syscall number
146     MOV ebx, 1                ; File descriptor 1 (stdout)
147     MOV ecx, new_line
148     MOV edx, 2
149     INT 0x80
150
151 exit:
152     MOV eax, 1                ; sys_exit() syscall number
153     XOR ebx, ebx              ; Exit code 0
154     INT 0x80
```

Código 1: Manipulación de Directorios: Crea un directorio, añade un archivo en él y se le guarda la fecha actual del sistema

```
1 import datetime
2
3 timestamp = 1729565904
4 print(datetime.datetime.utcfromtimestamp(timestamp))
```

Código 2: Conversión de timestamp a la fecha y hora actual del sistema





## 2.2. Ejecución del Programa

```
> ls
main main.asm
> ./main
El directorio actual es:
/home/donkey/Drives/github/escuela/6to/programacion-de-bajo-nivel/acts/5-directorios/src/alan

El directorio actual es:
/home/donkey/Drives/github/escuela/6to/programacion-de-bajo-nivel/acts/5-directorios/src

A> ~/Dr/gi/escuela/6to/p/a/5/src > on git main !9 ?8
```

Fig. 1: Ejecución del Programa: Muestreo de la ruta actual al entrar y salir de la carpeta creada

```

0x80490af <open_file+22>      mov     ds:0x804a238,eax
0x80490b4 <read_date>        mov     eax,0xd
0x80490b9 <read_date+5>      xor     ebx,ebx
0x80490bb <read_date+7>      int     0x80
>0x80490bd <read_date+9>    mov     ds:0x804a234,eax
0x80490c2 <write_file>      mov     eax,0x4
0x80490c7 <write_file+5>    mov     ebx,DWORD PTR ds:0x804a238
0x80490cd <write_file+11>   mov     ecx,0x804a234
0x80490d2 <write_file+16>   mov     edx,0x4
0x80490d7 <write_file+21>   int     0x80
0x80490d9 <close_file>      mov     eax,0x6
0x80490de <close_file+5>    mov     ebx,DWORD PTR ds:0x804a238
0x80490e4 <close_file+11>  int     0x80
0x80490e6 <change_parent_dir> mov     eax,0xc
0x80490eb <change_parent_dir+5> mov     ebx,0x804a000
0x80490f0 <change_parent_dir+10> int     0x80
0x80490f2 <print_cwd>      mov     eax,0x4
0x80490f7 <print_cwd+5>    mov     ebx,0x1
0x80490fc <print_cwd+10>   mov     ecx,0x804a013
0x8049101 <print_cwd+15>   mov     edx,0x1b
0x8049106 <print_cwd+20>   int     0x80
0x8049108 <print_cwd+22>   mov     eax,0x4
0x804910d <print_cwd+27>    mov     ebx,0x1
0x8049112 <print_cwd+32>   mov     ecx,0x804a134
0x8049117 <print_cwd+37>   mov     edx,0x100
0x804911c <print_cwd+42>   int     0x80
0x804911e <print_cwd+44>   mov     eax,0x4

native process 346906 (asm) In: read_date
Single stepping until exit from function print_new_cwd,
which has no line number information.
0x08049088 in create_file ()
Single stepping until exit from function create_file,
which has no line number information.
0x08049099 in open_file ()
Single stepping until exit from function open_file,
which has no line number information.
0x080490b4 in read_date ()
(gdb) stepi 2
0x080490bb in read_date ()
(gdb) info registers eax ebx
eax             0xd             13
ebx             0x0             0
(gdb) stepi
0x080490bd in read_date ()
(gdb) info registers eax
eax             0x671714d0      1729565904
(gdb)

```

Fig. 2: Valor almacenado en la variable timestamp

```
> ./main
El directorio actual es:
/home/donkey/Drives/github/escuela/6to/programacion-de-bajo-nivel/acts/5-directorios/src/alan

El directorio actual es:
/home/donkey/Drives/github/escuela/6to/programacion-de-bajo-nivel/acts/5-directorios/src

> cat alan/juarez.txt
Sg
```

Fig. 3: Impresión de la “fecha actual del sistema” (timestamp) almacenada en el archivo creado

```
> cat alan/juarez.txt
2024-10-22 02:58:24
```

Fig. 4: Valor obtenido al implementar el código 2, valor que debería ser almacenado en el archivo txt



### 3. Conclusión

Para finalizar, cabe destacar que el código 1 cuenta con un pequeño error, en vez de guardar la fecha actual guarda el la cantidad de segundos ocurridos a partir del 01 de enero de 1970, almacenada en `timestamp` y obtenida de la llamada a la función `sys_time` (13). Desafortunadamenete me fue imposible convertir correctamente este valor. Es importante mencionar que el código 1 es lenguaje ensamblador NASM x86 (linux), por lo cual, no existe ninguna llamada al sistema que te otorgue la fecha actual, tal como lo hace el lenguaje ensamblador MSDOS, utilizado en EMU8086.