

# **UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA**

## **Facultad de Ingeniería, Arquitectura y Diseño**

**Ingeniero en Software y Tecnologías Emergentes**



### **Manual de Talleres de la materia Organización de Computadoras**

**AUTOR(ES)**  
Jonatan Crespo Ragland

## **Taller No. 10 Desplazamiento de bits**

**Objetivo: Desarrollar el uso de desplazamiento de bits**

## **Fundamentos teóricos del taller**

- Lenguaje Ensamblador x86

## **Instrucciones para el desarrollo del taller**

1. Desarrolla los siguientes puntos

### **Recursos**

1. Apuntes de clase.
2. Fuentes bibliográficas.
3. Lápiz.
4. Equipo de cómputo.

## **Tiempo e instrucciones de entrega**

Duración: 2 horas.

**Desarrollar lo siguiente en su cuaderno o computadora: 1. Modifica el código de prueba para incluir instrucciones de SHL, SHR, ROR y ROL (las 4 para cada inciso). Tu debes de realizar los desplazamientos y rotación a tu consideración, para imprimir cada uno de los caracteres que se piden (cada uno en un programa por separado). Para cada caso documenta tus resultados en el taller. Para tu repositorio solo es necesario que subas el código del**

**inciso g. Adelante puedes encontrar el código de prueba y el enlace correspondiente al mismo código.**

**a. A**

**b.0**

**c. g**

**d.=**

shl:

section .data

char db 0

newline db 10

section .text

global \_start

\_start:

mov al, 112 ; AL = 112

shl al, 1 ; Desplaza a la izquierda 1 vez → AL = 224

mov [char], al

; Escribir carácter

mov eax, 4

mov ebx, 1

mov ecx, char

mov edx, 1

int 0x80

; Nueva línea

mov eax, 4

mov ebx, 1

mov ecx, newline

mov edx, 1

int 0x80

; Salir

mov eax, 1

xor ebx, ebx

int 0x80

shr:

section .data

char db 0

newline db 10

section .text

global \_start

\_start:

mov al, 112 ; AL = 112

shr al, 1 ; Desplaza a la derecha 1 vez → AL = 56  
mov [char], al

```
; Escribir carácter
mov eax, 4
mov ebx, 1
mov ecx, char
mov edx, 1
int 0x80
```

```
; Nueva línea
mov eax, 4
mov ebx, 1
mov ecx, newline
mov edx, 1
int 0x80
```

```
; Salir
mov eax, 1
xor ebx, ebx
int 0x80
```

```
ro:
section .data
char db 0
newline db 10
```

```
section .text
global _start
```

```
_start:
mov al, 112 ; AL = 112
rol al, 1 ; Rota a la izquierda 1 vez → AL = 224
mov [char], al
```

```
; Escribir carácter
mov eax, 4
mov ebx, 1
mov ecx, char
mov edx, 1
int 0x80
```

```
; Nueva línea
mov eax, 4
mov ebx, 1
mov ecx, newline
mov edx, 1
int 0x80
```

```

; Salir
mov eax, 1
xor ebx, ebx
int 0x80

ror:
section .data
char db 0
newline db 10

section .text
global _start

_start:
mov al, 112 ; AL = 112
ror al, 1 ; Rota a la derecha 1 vez → AL = 56
mov [char], al

; Escribir carácter
mov eax, 4
mov ebx, 1
mov ecx, char
mov edx, 1
int 0x80
; Nueva línea
mov eax, 4
mov ebx, 1
mov ecx, newline
mov edx, 1
int 0x80

; Salir
mov eax, 1
xor ebx, ebx
int 0x80

```

## Código prueba

<https://onecompiler.com/assembly/43g8tnypc>

```
section .data  
char db 0  
newline db 10
```

```
section .text  
global _start
```

```
_start:  
; Aqui van las instrucciones del taller 10
```

```
; Guardar en char  
mov [char], al
```

```
; Escribir carácter en consola  
mov eax, 4 ; syscall write  
mov ebx, 1 ; stdout  
mov ecx, char  
mov edx, 1 ; 1 byte  
int 0x80
```

```
; Salto de línea  
mov eax, 4  
mov ebx, 1  
mov ecx, newline  
mov edx, 1  
int 0x80
```

```
; Salir  
mov eax, 1 ; syscall exit  
xor ebx, ebx ; código de salida 0
```

# int 0x80

## Referencias

1. Ornare quam viverra orci sagittis eu volutpat. Aenean et tortor at risus. Feugiat in ante metus dictum at tempor commodo.
2. Senectus et netus et malesuada fames ac. Dictum sit amet justo donec enim diam vulputate ut pharetra. Tristique senectus et netus et malesuada fames

## Anexos

Incluir un anexo con modelos de rúbricas, formatos de evaluación y otros recursos que faciliten la implementación de los talleres