

Universidad Autónoma de Baja California

Ingeniería en Computación



Facultad de Ciencias Químicas e Ingeniería

Organización y Arquitectura de Computadoras

Práctica 6

Zavala Román Irvin Eduardo

Grupo: 551

29/09/2021

Periodo 2021-2

Objetivo

Identificar los modos de direccionamiento adecuados para manejo de memoria en aplicaciones de sistemas basados en microprocesador mediante la distinción de su funcionamiento, de forma lógica y responsable.

Desarrollo

Copie el código del Listado 1 en un archivo llamado entrada.asm. Abra una terminal en Linux y ensamble el código con NASM por medio del comando: `nasm -f elf entrada.asm`

El cual generará el archivo objeto entrada.o. Encadene el archivo por medio de uno de los siguientes comandos:

- a) En un sistema operativo de 32 bits: `ld -s -o entrada entrada.o`
 - b) En un sistema operativo de 64 bits: `ld -m elf_i386 -s -o entrada entrada.o`
- El cual generará el archivo ejecutable entrada.

Ejecute el archivo por medio del comando: `./entrada` El programa solicita al usuario el ingreso de su nombre y despliega un mensaje de saludo.

2. Cree un programa llamado P6.asm que contenga las instrucciones necesarias para hacer lo que se indica a continuación:

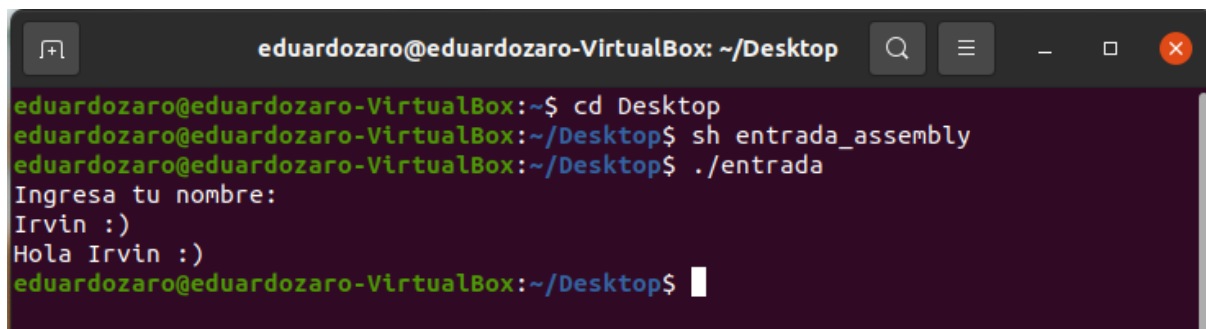
- a) Reservar dos espacios en memoria no inicializados, uno de 64 bytes etiquetado como A y el otro de 1 byte etiquetado como N.
- b) Solicitar una cadena que se almacene en A.
- c) Copiar el primer carácter de A en la variable N. Use un modo de direccionamiento base.
- d) Reemplazar el tercer carácter A por un guion -, usando un modo de direccionamiento base con desplazamiento.
- e) Reemplazar el octavo carácter de A por una X usando un direccionamiento base con índice escalado.

f) Copiar el segundo carácter de A y almacenarlo en los bits 15-8 del acumulador.

g) Reemplazar el noveno carácter de A por el carácter en los bits 15-8 del acumulador, usando un direccionamiento base con índice escalado y desplazamiento.

Resultados

1. La ejecución del programa entrada queda de la siguiente manera:



```
eduardozaro@eduardozaro-VirtualBox: ~/Desktop
eduardozaro@eduardozaro-VirtualBox:~/Desktop$ cd Desktop
eduardozaro@eduardozaro-VirtualBox:~/Desktop$ sh entrada_assembly
eduardozaro@eduardozaro-VirtualBox:~/Desktop$ ./entrada
Ingresa tu nombre:
Irvin :)
Hola Irvin :)
eduardozaro@eduardozaro-VirtualBox:~/Desktop$
```

Figura 1. Ejecución de entrada

2. El código resultante de los ejercicios del inciso 2 es lo siguiente:

```
section .data ; Datos inicializados
    msg1:    db "Ingresa A: ",10
    msg1_L:  equ $-msg1

    msg2:    db "N = "
    msg2_L:  equ $-msg2

    NL:      db 13, 10
    NL_L:    equ $-NL

section .bss ; Datos no inicializados
    A resb 64
    N resb 1

section .text
    global _start

_start:
    mov eax, 4 ;Escribe
    mov ebx, 1
    mov ecx, msg1
    mov edx, msg1_L
    int 80h

    mov eax, 3 ;Lee
    mov ebx, 0
    mov ecx, A
```

```
mov edx, 64  
int 80h
```

```
mov al,[A] ;INCISO C) GUARDAR PRIMER DIGITO DE A EN N DIR BASE  
mov [N], al  
mov eax, 4 ;Escribe  
mov ebx, 1  
mov ecx, msg2  
mov edx, msg2_L
```

```
int 80h  
mov eax, 4 ;Escribe  
mov ebx, 1  
mov ecx, N  
mov edx, 1  
int 80h
```

```
mov eax, 4 ;Salto  
mov ebx, 1  
mov ecx, NL  
mov edx, NL_L  
int 80h
```

```
mov al, "-" ;INCISO D) CAMBIAR TERCER CARACTER DE A POR GUION, DIR B + I  
mov [A+ebx+1], al
```

```
mov eax, 4 ;Escribe  
mov ebx, 1  
mov ecx, A  
mov edx, 64  
int 80h
```

```
mov eax, 4 ;Salto  
mov ebx, 1  
mov ecx, NL  
mov edx, NL_L  
int 80h
```

```
mov ebx, 7;INCISO E) CAMBIAR OCTAVO CARACTER DE A POR X, DIR Bx  
mov byte [A+ebx], "X"
```

```
mov eax, 4 ;Escribe  
mov ebx, 1  
mov ecx, A  
mov edx, 64  
int 80h
```

```
mov eax, 4 ;Salto  
mov ebx, 1
```

```
mov ecx, NL
mov edx, NL_L
int 80h
```

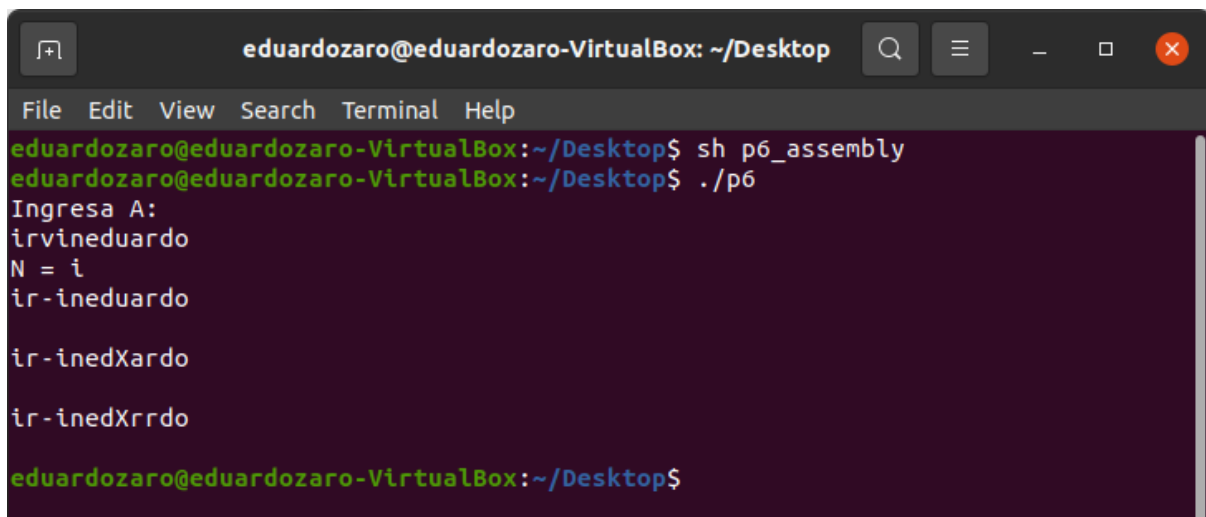
```
mov ah, [A+1] ;INCISO F) COPIAR 2DO CARACTER DE A Y ALMACENARLO EN
BITS 15-8 DE ACUMULADOR
mov [A+ebx*4+4], ah ;INCISO G) COPIAR EN EL 9NO CARACTER DE A LO DE AH
```

```
mov eax, 4 ;Escribe
mov ebx, 1
mov ecx, A
mov edx, 64
int 80h
```

```
mov eax, 4 ;Salto
mov ebx, 1
mov ecx, NL
mov edx, NL_L
int 80h
```

```
mov eax, 1
mov ebx, 0
int 80h
```

Dando el siguiente resultado:



```
eduardozaro@eduardozaro-VirtualBox: ~/Desktop
File Edit View Search Terminal Help
eduardozaro@eduardozaro-VirtualBox:~/Desktop$ sh p6_assembly
eduardozaro@eduardozaro-VirtualBox:~/Desktop$ ./p6
Ingresa A:
irvineduardo
N = i
ir-ineduardo

ir-inedXardo

ir-inedXrrdo

eduardozaro@eduardozaro-VirtualBox:~/Desktop$
```

Figura 2. Ejecución de P6

Conclusión

Los modos de direccionamiento nos permiten acceder a direcciones de memoria de distintas maneras, permitiendo realizar acciones como guardar o pasar datos entre registros y variables, esto es algo interesante aunque el funcionamiento me sigue pareciendo un poco confuso.