Práctica: "Hola Mundo" en Ensamblador

Objetivo de la Práctica

Esta práctica tiene como objetivo introducirte a la programación en ensamblador utilizando NASM dentro de un contenedor Docker. Crearemos un programa básico que imprimirá "Hola Mundo" en la terminal. Aprenderás a escribir el código en ensamblador, ensamblarlo, vincularlo y ejecutarlo en un sistema Linux.

Requisitos Previos

1. **Docker**: Debes tener Docker instalado en tu sistema.
2. **Editor de texto**: Puedes usar Visual Studio Code, Vim, Nano, o cualquier editor de texto de tu preferencia.

---

1. Preparación del Entorno

1. Abre una terminal en tu sistema.
2. Crea una carpeta para la práctica y navega a ella:
   `bash mkdir practica_hola_mundo cd practica_hola_mundo`
3. Crea un archivo llamado HolaMundo.asm:

   `touch HolaMundo.asm`

2. Escribir el Código en Ensamblador

Abre el archivo HolaMundo.asm en tu editor de texto y escribe el siguiente código:

```
msg db "Hola Mundo", 0xA, 0xD   ; Mensaje a imprimir con salto de línea
len equ $ - msg                 ; Calcula la longitud del mensaje

section .text
global _start                   ; Punto de entrada

_start:
    ; Syscall para escribir en pantalla
    mov eax, 4                  ; Syscall número 4 (sys_write)
    mov ebx, 1                  ; File descriptor 1 (stdout)
    mov ecx, msg                ; Dirección del mensaje
    mov edx, len                ; Longitud del mensaje
    int 0x80                    ; Llama a la interrupción del sistema

    ; Syscall para terminar el programa
    mov eax, 1                  ; Syscall número 1 (sys_exit)
    xor ebx, ebx                ; Código de salida 0
    int 0x80                    ; Llama a la interrupción del sistema
```

Explicación del Código

- Segmento de datos (section .data): Aquí se define el mensaje "Hola Mundo" que queremos imprimir en pantalla.
- msg: etiqueta que almacena el texto y los caracteres de nueva línea (0xA, 0xD).
- len: calcula la longitud del mensaje msg (número de bytes) utilizando $ - msg.
- Segmento de texto (section .text): Este es el segmento de código, donde está la lógica del programa.
- global _start: define el punto de entrada del programa.
- _start: etiqueta de inicio del programa.
- mov eax, 4: prepara la llamada al sistema (sys_write) para imprimir en pantalla.
- mov ebx, 1: especifica la salida estándar (1) como destino.
- mov ecx, msg: establece la dirección del mensaje en msg.
- mov edx, len: longitud del mensaje a escribir.
- int 0x80: llama a la interrupción de Linux para ejecutar la syscall.
- mov eax, 1: prepara la llamada al sistema (sys_exit) para salir del programa.
- xor ebx, ebx: establece el código de salida como 0 (éxito).
- int 0x80: llama a la interrupción de Linux para finalizar el programa.

3. Iniciar el Contenedor Docker

Inicia un contenedor Docker con NASM y monta el directorio donde está el código para ensamblar y ejecutar el programa:

```
docker run -it --name ensamblador -v $(pwd):/code codeneomatrix/nasm sh
```

- -it: inicia el contenedor en modo interactivo.
- --name ensamblador: le asigna el nombre ensamblador al contenedor.
- -v $(pwd):/code: monta el directorio actual ($(pwd)) en la ruta /code del contenedor.
- codeneomatrix/nasm: imagen Docker con NASM preinstalado.
- sh: inicia una shell dentro del contenedor.

4. Ensamblar el Código

Una vez en la terminal del contenedor, navega al directorio montado y ensambla el código para generar un archivo objeto:

`nasm -f elf32 HolaMundo.asm -o HolaMundo.o`

- -f elf32: especifica el formato de salida como elf32, compatible con sistemas de 32 bits.
- HolaMundo.asm: archivo fuente en ensamblador.
- -o HolaMundo.o: archivo objeto de salida.

5. Vincular el Archivo Objeto

Vincula el archivo objeto para generar un binario ejecutable:

`ld -m elf_i386 HolaMundo.o -o HolaMundo`

- -m elf_i386: especifica el formato de 32 bits para compatibilidad.
- HolaMundo.o: archivo objeto ensamblado.
- -o HolaMundo: nombre del archivo ejecutable de salida.

6. Ejecutar el Programa

Para ejecutar el programa, usa el siguiente comando:

`./HolaMundo`

Si todo está correcto, deberías ver la salida **Hola Mundo** en la terminal.

Ejemplo de Ejecución Completo:

1. Escribir el código en HolaMundo.asm.
2. Iniciar el contenedor Docker:
   `docker run –it ––name ensamblador –v $(pwd):/code codeneomatrix/nasm sh`
3. Ensamblar:

   ```
   nasm –f elf32 HolaMundo.asm –o HolaMundo.o```
   ```
4. Vincular:
   `ld –m elf_i386 HolaMundo.o –o HolaMundo`
5. Ejecutar:
   `./HolaMundo`

Preguntas de Reflexión

```
1.  ¿Qué hace cada sección (section .data y section .text) en el código ensamblador?
2.  ¿Por qué es necesario el uso de int 0x80 en Linux para realizar llamadas al sistema?
3.  ¿Qué diferencia hay entre un archivo objeto (.o) y un ejecutable?
```

Conclusión

Esta práctica te ha enseñado cómo crear y ejecutar un programa básico en ensamblador NASM en un entorno Docker en Linux. Has aprendido a definir datos y segmentos de código, y cómo realizar llamadas al sistema para interactuar con el sistema operativo.