**LABORATORIO NO. 02**

“Ensamblador, DEBUG y Sistemas Numéricos”

**Ejercicio 1: Utilización del Ensamblador y el Enlazador**

Utilizando los archivos “Ejemplo1.asm” y “Ejemplo2.asm” genere el código objeto y el programa ejecutable utilizando el Ensamblador “TASM” y el Enlazador “TLINK”.

Conteste las siguientes preguntas:

1. Cuando se genera el ejecutable del archivo “Ejemplo2.asm”, ¿cuál es la advertencia que se muestra en pantalla? ¿Por qué muestra esa advertencia?

R// La advertencia que manda es “WARNING: NO STACK”, muestra esta advertencia debido a que en código no estaba definido desde el archivo asm la pila. STACK.

1. Modificado el programa para que no muestre la advertencia, ¿cuál es el resultado del programa, es decir, por qué se imprime ese carácter y no un 30?

R//El código imprimía de carácter a un asterisco debido a que lo esta tomando como hexadecimal, lo cual devuelve un 2A en hexadecimal que es el correspondiente en código ASCII a ese carácter.

1. Modifique el código del archivo “Ejemplo2.asm” y utilizando la tabla de códigos ASCII, imprima en pantalla una letra “Z”.

R// 5A en código ASCII.

**Ejercicio 2: Utilización del Modo “DEBUG”**

Comandos del Modo “DEBUG”:

* N Nombrar un programa.
* L Se encarga de cargar el programa.
* U        "Desensamblar" código máquina y pasarlo a código simbólico.
* A        Ensamblar instrucciones simbólicas y pasarlas a código máquina.
* D        Mostrar el contenido de un área de memoria.
* E        Introducir datos en memoria, iniciando en una localidad específica.
* G       Correr el programa ejecutable que se encuentra en memoria.
* P        Proceder o ejecutar un conjunto de instrucciones relacionadas.
* Q       Salir de la sesión con DEBUG.
* R       Mostrar el contenido de uno o más registros.
* T        Rastrear la ejecución de una instrucción.
* W       Escribir o grabar un programa en disco.

Utilizando el Modo “DEBUG” de DOS cargue el programa “Ejemplo2.exe” y responda las siguientes preguntas:

1. ¿En qué dirección de memoria inicia el código del programa?

R// 0B6C:0000 B86E0B

1. ¿En qué dirección de memoria termina el código del programa?
2. R// 0B67:001F 69
3. Aparecen los comentarios en pantalla ¿Sí? ¿No? ¿Por qué?

R// Las líneas en blanco y los comentarios son omitidos al momento de correr el programa debido a que no son sintaxis de assembler.

1. Para cada una de las instrucciones del programa, escriba la dirección de memoria que tiene asignada:

|  |  |
| --- | --- |
| Dirección de memoria | Instrucción |
| 0B6C:0000 B86E0B | Mov AX,@DATA |
| 0B6C:0003 8ED8 | Mov DS,AX |
| 0B6C:0005 B80000 | Mov AX,0000h |
| 0B6C:0008 BB0000 | Mov BX,0000h |
| 0B6C:000B B015 | Mov AL,15h |
| 0B6C:000D B315 | Mov BL,15h |
| 0B6C:000F 02C3 | Add AL,BL |
| 0B6C:0015 8AD0 | Mov DL,AL |
| 0B6C:0017 B402 | Mov AH,02 |
| 0B6C:0019 CD21 | Int 21h |
| 0B6C:001B B44C | Mov AH,4CH |
| 0B6C:001D CD21 | int 21h |

1. ¿Cuál es la dirección del segmento de código?

R// 0B6C:0000 B86E0B

1. Antes de iniciar la ejecución por pasos del programa, ¿cuáles son los valores de los registros de propósito general?

R// Los valores que tienen son los asignados previamente por otro programa que se está ejecutando o bien tienen los valores que vienen por defecto.

1. El valor del IP, ¿coincide con la dirección de inicio del programa?

R//El valor de IP es 0003 y si coincide con la segunda dirección.

1. Utilice el comando para el rastreo instrucción por instrucción y, por cada línea del código, escriba el contenido de los registros internos del CPU.



