**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?

Turbo Link Version 3.01 Copyrigth (c) 1987, 1990 Borland International.

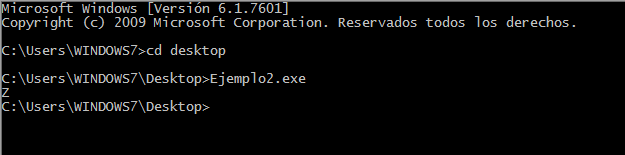
Warning: No Stack

Cuando se genera el ejecutable del archivo “Ejemplo2.asm”, La advertencia que muestra en pantalla dice “No Stack” y se muestra porque en el código original no tiene el “.stack” definido, osea que el segmento Stack no está definido.

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?

Porque se suma el contenido de cada uno de los registros, tanto el AL que contiene un 30h más los nuevos 30h que tiene la nueva. Daría un 60h, lo cual escrito en hexadecimal es un 5A, el cual tiene como símbolo una Z

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



**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?



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



1. Aparecen los comentarios en pantalla ¿Sí? ¿No? ¿Por qué?

No, ya que al momento de debugear solo le importa mostrar direcciones de memoria e instrucciones realizadas dentro del programa (movimientos entre registros y direcciones de memoria para poder realizar el proceso completo)

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

|  |  |
| --- | --- |
| Dirección de memoria | Instrucción |
|  | Mov AX,@DATA |
|  | Mov DS,AX |
|  | Mov AX,0000h |
|  | Mov BX,0000h |
|  | Mov AL,15h |
|  | Mov BL,15h |
|  | Add AL,BL |
|  | Mov DL,AL |
|  | Mov AH,02 |
|  | Int 21h |
|  | Mov AH,4CH |
|  | int 21h |

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

OB6E

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

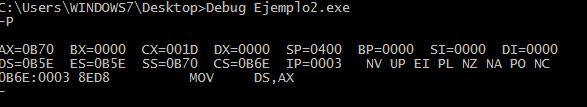
AX = 0B70

BX = 0000

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

IP = 0003

Si, si coiincide.



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.

