**LABORATORIO NO. 01**

“Ensamblador, DEBUG y Sistemas Numéricos”

**Objetivo**

* Mostrar la estructura básica de un programa en Lenguaje ensamblador, así como la forma de ensamblarlo y generar el código objeto, además de su enlace como programa ejecutable.
* Demostrar el funcionamiento de la herramienta para realizar la depuración de un programa escrito en Lenguaje Ensamblador.
* Familiarizar los diversos sistemas numéricos, sus conversiones, operaciones y lógica apropiada para representar información en un programa en Lenguaje Ensamblador.

**Ejercicio 1: Declaración de variables e intercambio de valores**

Declarar dos variables, que almacenen dos valores,

* Variable 1: nombre del alumno
* Variable 2: carnet del alumno

Imprimir ambos valores, luego intercambiarlos, es decir, variable 1 = variable 2 y variable 2= variable 1, y de nuevo imprimir los valores.

**Ejercicio 2: 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?

* *No se puede ejecutar el programa Ejemplo2.asm*
* *Se muestra porque el nombre del archivo ASM es muy extenso.*

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 la consola imprime el valor ASCII equivalente al valor numérico.*
* *El resultado no es 30, es 2A, ya que la operación es hexadecimal.*

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

**Ejercicio 3: 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. **0CF4**
2. ¿En qué dirección de memoria termina el código del programa?
   1. **0CF4+001C = 0D10**
3. Aparecen los comentarios en pantalla ¿Sí? ¿No? ¿Por qué?
   1. *No, los comentarios son eliminados durante el ensamblaje*.
4. Para cada una de las instrucciones del programa, escriba la dirección de memoria que tiene asignada:

|  |  |
| --- | --- |
| **Dirección de memoria** | **Instrucción** |
| **B8F50C** | Mov AX,@DATA |
| **8ED8** | Mov DS,AX |
| **B80000** | Mov AX,0000h |
| **B80000** | Mov BX,0000h |
| **B015** | Mov AL,15h |
| **B315** | Mov BL,15h |
| **02C3** | Add AL,BL |
| **8AD0** | Mov DL,AL |
| **B402** | Mov AH,02 |
| **CD21** | Int 21h |
| **B44C** | Mov AH,4CH |
| **CD21** | int 21h |

1. ¿Cuál es la dirección del segmento de código?
   1. **CS = OCF4**
2. Antes de iniciar la ejecución por pasos del programa, ¿cuáles son los valores de los registros de propósito general?
   1. **AX = 0000**
   2. **BX = 0000**
   3. **CX = 01D6**
   4. **DX = 0000**
3. El valor del IP, ¿coincide con la dirección de inicio del programa?
   1. ***Sì.***
4. Utilice el comando para el rastreo instrucción por instrucción y, por cada línea del código, escriba el contenido de los registros internos del CPU.

|  |  |
| --- | --- |
| Instrucción | Datos |
| debug Ejemplo2.EXE | AX = 0000  BX = 0000  CX = 01D6  DX = 0000 |
| MOV AX, @DATA | AX = 0CF5  BX = 0000  CX = 01D6  DX = 0000 |
| MOV DS, AX | AX = 0CF5  BX = 0000  CX = 01D6  DX = 0000 |
| MOV AX, 0000 | AX = OCF5  BX = 0000  CX = 01D6  DX = 0000 |
| MOV BX, 0000 | AX = 0000  BX = 0000  CX = 01D6  DX = 0000 |
| MOV AL, 15H | AX = 0015  BX = 0000  CX = 01D6  DX = 0000 |
| MOV BL, 15H | AX = 0015  BX = 0015  CX = 01D6  DX = 0000 |
| ADD AL, BL | AX = 002A  BX = 0015  CX = 01D6  DX = 0000 |
| MOV DL, AL | AX = 002A  BX = 0015  CX = 01D6  DX = 002A |
| ADD DL, 30H | AX = 002A  BX = 0015  CX = 01D6  DX = 005A |
| MOV AH, 02 | AX = 022A  BX = 0015  CX = 01D6  DX = 005A |
| INT 21 | AX = 022A  BX = 0015  CX = 01D6  DX = 005A |
| MOV AH,4C | AX = 4C5A  BX = 0015  CX = 01D6  DX = 005A |
| FIN | |