

Laboratorio 1: Modos de direccionamiento

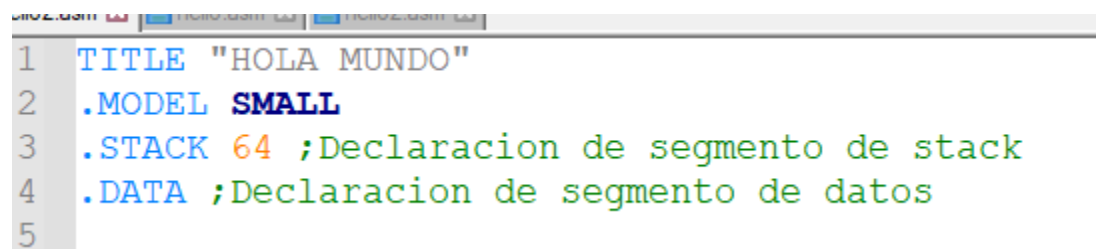
El siguiente laboratorio debe realizarse en parejas. Éste deberá subirse al portal para el día 4 de febrero a la media noche, solamente es necesario el archivo ASM. Su código deberá estar comentado y indentado para que sea fácil de entender.

Debe de agregar un encabezado donde se indiquen los nombres de los integrantes del grupo, y una descripción del funcionamiento del mismo escrito en sus palabras. En el resto del código con cada variable se deberá de comentar el uso de cada una de ella. También comentar la función general de bloques de código que realicen una función o tarea específica, por ejemplo una subfunción que muestre algo en pantalla deberá estar comentada indicando lo que hace, si espera parámetros y si los devuelve.

Teoría

Directivas

Nuestro código debe de ser capaz de comunicarse con el ensamblador para indicar cuanto espacio necesita o especificar que partes deben guardarse en el segmento de código o en el de datos. Para esto utilizamos **directivas**. Esto es código que no se traduce a binario, sin embargo, le dan contexto al ensamblador de como traducir el resto de nuestro código.



```
1 TITLE "HOLA MUNDO"
2 .MODEL SMALL
3 .STACK 64 ;Declaracion de segmento de stack
4 .DATA ;Declaracion de segmento de datos
5
```

En esta imagen vemos 4 directivas. Las cuales tienen la siguiente función:

Title: Indica el nombre del programa, en caso otra aplicación o el sistema operativo desee saberlo.

.MODEL: Esta indica el tamaño de nuestro programa, específicamente cuanto espacio apartar para los segmentos de código, datos. SMALL es un parámetro que dice que solamente vamos a necesitar un segmento de cada uno y que este puede tener un tamaño de 64K localidades de memoria. Existen otras posibilidades como MEDIUM que nos permite utilizar más segmentos de código pero solo uno de data o

LARGE que nos deja utilizar cualquier cantidad para ambos, pero para este curso SMALL será mas que suficiente.

STACK: Indica que en ese lugar se utilizara el segmento de datos del stack y el tamaño de 64 bytes.

DATA: Indica donde inicia el segmento de datos, no es necesario indicar donde termina ya que esto está implícito por utilizar el modelo SMALL.

```
0
1 .386
2 .CODE ;Declaracion de segmento de codigo
3 CSEG
4 MAIN     PROC FAR
```

.386: Es el procesador para el que se debe ensamblar, 386 indica que estamos programando para el 8086.

CODE: Funciona igual que data pero indica donde empieza el segmento de código.

PROC: Los programas van a estar segmentados por medio de procedimientos. FAR es un parámetro que indica el tamaño de este.

```
MAIN     INT 21H
         ENDP
         END MAIN
```

Esta última sección indica el fin del procedimiento y del programa.

Basándose en el ejemplo hello2.asm donde se realizarán 4 demostraciones de tipos de direccionamiento. Se le mostrará al usuario un título indicando que direccionamiento se esta ejemplificando seguido por el resultado del ejemplo. No hay entradas por el usuario y el programa de correr todos los ejemplos en la misma ejecución.

```
C:\>LAB1
Ejemplo de direccionamiento directo:
Laboratorio 1 de Arquitectura del computador1 ?

Ejemplo de direccionamiento indirecto:
Arquitectura del computador1 ?

Ejemplo 1 de direccionamiento con escalar:
VOCAL

Ejemplo 2 de direccionamiento con escalar:
BINGO
```

Dentro del programa usted debe de agregar variables en su segmento de datos que cumplan los siguientes requisitos:

- 3 variables estáticas para los títulos de cada uno de los ejemplos. Estas 3 variables serán los títulos.
Ejemplo:
 - 'Ejemplo de direccionamiento directo: '
 - 'Ejemplo de direccionamiento indirecto: '
 - 'Ejemplo 1 de direccionamiento con escalar: '
- 1 variable cuyo texto sea "Laboratorio 1 de Arquitectura del computador 1!", está se utilizará para los ejemplos de direccionamiento directo e indirecto.
- 4 variables que sean palabras del mismo número de caracteres. Estas deben de estar declaradas consecutivamente, osea las variables de los otros incisos no pueden ser declaradas en medio de estas. Éstas se usarán para los ejemplos de direccionamiento con escalar.

```
TEXT01 DB 'ARROZ'  
TEXT02 DB 'AUDIO'  
TEXT03 DB 'VOCAL'  
TEXT04 DB 'BINGO'
```

Declaración de variables

```
MESS    DB    'Hello',24H  
MESS1   DB    10 DUP('Hola'), 24H  
MESS2   DB    'Hello, $World!' ,0DH,0AH,24H
```

Debemos declarar nuestras variables en el segmento de datos. Podemos ver la declaración de variables como ponerles nombre a localidades de memoria. No indicamos donde termina una variable, solamente donde empieza. Por ejemplo podría parecer que estamos guardando "Hello, 24H" en MESS, pero en realidad solo le estamos indicando al ensamblador que a la localidad de memoria donde empieza ese mensaje, en la H, le ponga el nombre MESS. No hay limite para el espacio de una variable. DB nos indica el tamaño de cada uno de los siguientes elementos, en este caso es una double byte.

Función de escritura

```
MOV DX,OFFSET MESS ;Mover la dirección del mensaje a DX
MOV AH,9 ;Cargar la función queremos realizar en AH
INT 21H ;Ejecutar rutina 9h de la interrupción 21h
```

La microarquitectura cuenta con un set de interrupciones que ya realizan funciones complejas, como leer teclado, mouse o en este caso desplegar en pantalla. Aquí estamos utilizando la interrupción 21H que tiene varias funciones. Al momento de hacer la instrucción INT 21H esta va a buscar a AH el número de la función que queremos realizar, este caso es la 9 que es de desplegar texto en pantalla. Esta función asume que la dirección de la cadena que vamos a utilizar esta en DX por lo que la colocamos utilizando un MOV.

Es importante notar que MOV DX, OFFSET MESS no guarda el valor Hello en DX. Guarda la dirección de memoria donde empieza. A esto le llamamos direccionamiento directo.

Inciso 1: Direccionamiento directo

Muestre el título de este inciso al usuario seguido por el texto “Laboratorio 1 de Arquitectura del computador 1!” que debe estar almacenado en una variable. Para esto debe referenciar directamente el nombre de la variable.

Inciso 2: Direccionamiento indirecto

Muestre el título de este inciso al usuario seguido por el texto “Arquitectura del computador 1!” que debe estar almacenado en la misma variable que mostro en el inciso 1. Utilice el direccionamiento indirecto para desplegar solo la parte deseada de la cadena. No debe de cambiar el contenido de lo que esta almacenado.

Inciso 3: Direccionamiento con escalar

Muestre el título de este inciso al usuario seguido por el texto que este almacenado en la tercera variable, tal como se ve en el ejemplo de la salida. Debe de utilizar direccionamiento con escalar, utilizando la primera de las variables como base, el largo de lo que se almacena en cada una como escalar y el número de la palabra deseada como índice. No es necesario que use EAX y EBX para estos incisos.

Inciso 4: Direccionamiento con escalar 2

Utilizando direccionamiento indirecto cambie el “1” del título del inciso anterior por un “2”, utilice el valor del código ascii, y repita el inciso anterior pero mostrando la cuarta palabra en la lista.

Punteo

Comentarios y estilo: 20%

Funcionamiento de cada inciso: 20%

