Actividad 10

Christian Geovany Muñoz Rodríguez

Ingeniería en computación

Código: 221350605

Seminario de Solución de Problemas de Traductores de Lenguajes I – D09 (Lunes y Miércoles de 1 a 3)

Maestro: José Juan Meza Espinoza

Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías

9 de mayo del 2023

Código:

```
ORG 100h ; Direccion de inicio del programa

JMP START ; Salto incondicional a la etiqueta START

CORX0 DW ? ; Variable para almacenar la coordenada X inicial
CORY0 DW ? ; Variable para almacenar la coordenada Y inicial

CORX1 DW ? ; Variable para almacenar la coordenada X final
CORY1 DW ? ; Variable para almacenar la coordenada Y final

COLOR EQU 0Fh ; Definicion de la constante COLOR con el valor 0Fh

MACRO DRAW ; Macro para dibujar con el color definido
    MOV AH, 0Ch ; Carga el número de la funcion en AH
    MOV AL, COLOR ; Carga el color en AL
    INT 10h ; Llama a la interrupcion del BIOS para dibujar
ENDM ; Fin de la macro DRAW

START: ; Etiqueta START: punto de inicio del programa

    MOV AX, @DATA ; Carga el segmento de datos en AX
```

```
    MOV DS, AX ; Asigna el valor del segmento de datos al registro DS


    MOV AH, 0 ; Limpia el registro AH

    MOV AL, 13h ; Carga el modo de video 13h en AL

    INT 10h ; Llama a la interrupcion del BIOS para establecer el modo de video


    MOV AH, 0h ; Limpia el registro AH

    INT 33h ; Llama a la interrupcion del BIOS para obtener las coordenadas del
mouse


REPITE: ; Etiqueta REPITE: inicio del bucle


    MOV AX, 03h ; Carga el número de la funcion en AX

    INT 33h ; Llama a la interrupcion del BIOS para actualizar las coordenadas
del mouse


    MOV CORX0, CX ; Almacena la coordenada X inicial en la variable

    MOV CORY0, DX ; Almacena la coordenada Y inicial en la variable


    CMP BX, 1 ; Compara el valor de BX con 1

    MOV BX, 0 ; Si no son iguales, se establece BX en 0

    JNE REPITE ; Salta a la etiqueta REPITE si no son iguales


REPITE1: ; Etiqueta REPITE1: inicio del bucle


    MOV AX, 03h ; Carga el número de la funcion en AX
```

```
    INT 33h ; Llama a la interrupcion del BIOS para actualizar las coordenadas
del mouse


    MOV CORX1, CX ; Almacena la coordenada X final en la variable

    MOV CORY1, DX ; Almacena la coordenada Y final en la variable


    CMP BX, 1 ; Compara el valor de BX con 1

    JNE REPITE1 ; Salta a la etiqueta REPITE1 si no son iguales


    MOV AH, 0Ch ; Carga el número de la funcion en AH

    MOV AL, COLOR ; Carga el color en AL


    MOV CX, CORX0 ; Carga la coordenada X inicial en CX

    MOV DX, CORY0 ; Carga la coordenada Y inicial en DX


DIBUJAX0: ; Etiqueta DIBUJAX0: inicio del bucle


    INT 10h ; Llama a la interrupcion del BIOS para dibujar

    INC CX ; Incrementa la coordenada X

    CMP CX,CORX1 ; Compara la coordenada X con CORX1

    JNE DIBUJAX0 ; Salta a la etiqueta DIBUJAX0 si no son iguales


DIBUJAY1: ; Etiqueta DIBUJAY1: inicio del bucle


    INT 10h ; Llama a la interrupcion del BIOS para dibujar

    INC DX ; Incrementa la coordenada Y
```

```
    CMP DX, CORY1 ; Compara la coordenada Y con CORY1
    JNE DIBUJAY1 ; Salta a la etiqueta DIBUJAY1 si no son iguales

DIBUJAX1: ; Etiqueta DIBUJAX1: inicio del bucle

    INT 10h ; Llama a la interrupcion del BIOS para dibujar
    DEC CX ; Decrementa la coordenada X
    CMP CX, CORX0 ; Compara la coordenada X con CORX0
    JNE DIBUJAX1 ; Salta a la etiqueta DIBUJAX1 si no son iguales

DIBUJAY0: ; Etiqueta DIBUJAY0: inicio del bucle

    INT 10h ; Llama a la interrupcion del BIOS para dibujar
    DEC DX ; Decrementa la coordenada Y
    CMP DX, CORY0 ; Compara la coordenada Y con CORY0
    JNE DIBUJAY0 ; Salta a la etiqueta DIBUJAY0 si no son iguales

    MOV AH, 0h ; Limpia el registro AH
    INT 16h ; Llama a la interrupcion del BIOS para esperar una tecla

RET ; Retorno del programa principal
```

Desarrollo:

El programa es un código en lenguaje ensamblador que utiliza interrupciones del BIOS para dibujar una figura en modo de video 13h. El objetivo del programa es obtener las coordenadas del mouse y utilizarlas para dibujar un rectángulo en la pantalla.

El programa comienza estableciendo el modo de video 13h, que es un modo gráfico que permite mostrar 320x200 píxeles con 256 colores. Luego, se utiliza la interrupción del BIOS (INT 33h) para obtener las coordenadas del mouse. Las coordenadas se almacenan en las variables CORX0 y CORY0 para representar la coordenada inicial del rectángulo.

```
START: ; Etiqueta START: punto de inicio del programa
    MOV AX, @DATA ; Carga el segmento de datos en AX
    MOV DS, AX ; Asigna el valor del segmento de datos al registro DS

    MOV AH, 0 ; Limpia el registro AH
    MOV AL, 13h ; Carga el modo de video 13h en AL
    INT 10h ; Llama a la interrupcion del BIOS para establecer el modo de video

    MOV AH, 0h ; Limpia el registro AH
    INT 33h ; Llama a la interrupcion del BIOS para obtener las coordenadas del mouse
```

```
CORX0 DW ? ; Variable para almacenar la coordenada X inicial
CORY0 DW ? ; Variable para almacenar la coordenada Y inicial
```

A continuación, se entra en un bucle (etiqueta REPITE) donde se actualizan las coordenadas del mouse y se almacenan en las variables CORX1 y CORY1 para representar la coordenada final del rectángulo. Este bucle se repite hasta que se detecte un clic del mouse, momento en el cual se sale del bucle.

```
REPITE: ; Etiqueta REPITE: inicio del bucle
    MOV AX, 03h ; Carga el n·mero de la funcion en AX
    INT 33h ; Llama a la interrupcion del BIOS para actualizar las coordenadas del mouse

    MOV CORX0, CX ; Almacena la coordenada X inicial en la variable
    MOV CORY0, DX ; Almacena la coordenada Y inicial en la variable

    CMP BX, 1 ; Compara el valor de BX con 1
    MOV BX, 0 ; Si no son iguales, se establece BX en 0
    JNE REPITE ; Salta a la etiqueta REPITE si no son iguales
```

Después de obtener las coordenadas iniciales y finales del rectángulo, se utiliza la macro DRAW para establecer el color de dibujo. La macro utiliza la interrupción del BIOS (INT 10h) con la función 0Ch para dibujar con el color especificado en la constante COLOR.

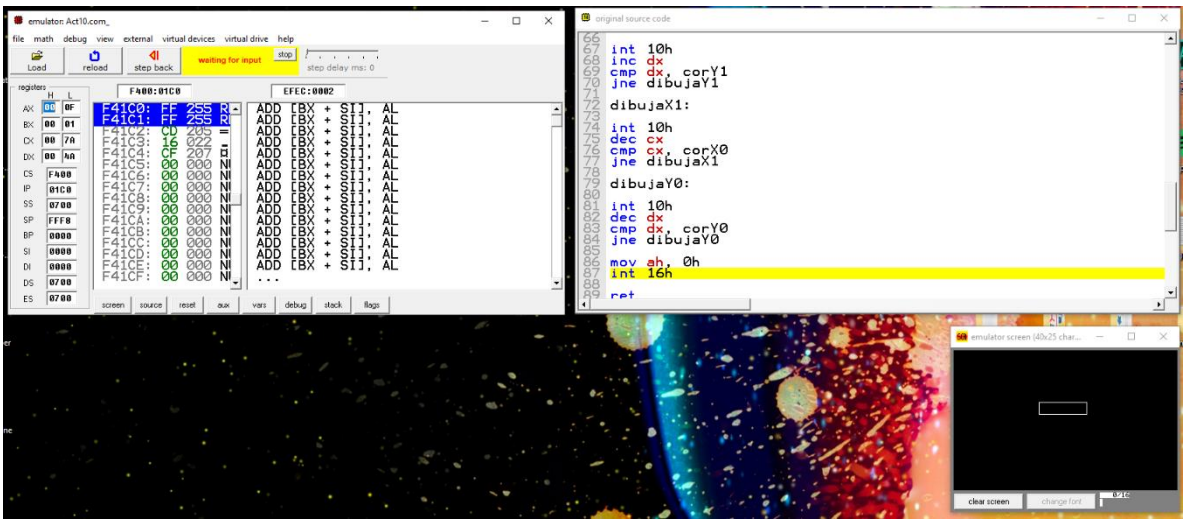
```
MACRO DRAW ; Macro para dibujar con el color definido
    MOV AH, 0Ch ; Carga el n·mero de la funcion en AH
    MOV AL, COLOR ; Carga el color en AL
    INT 10h ; Llama a la interrupcion del BIOS para dibujar
ENDM ; Fin de la macro DRAW
```

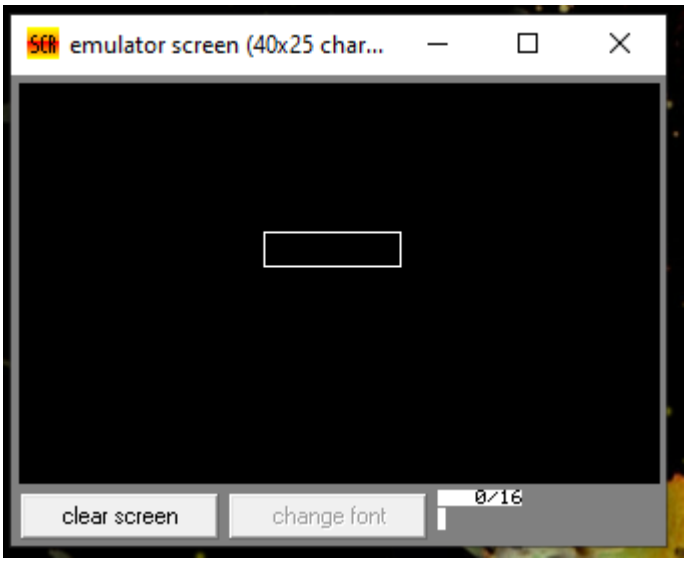
A continuación, se inicia un proceso de dibujo del rectángulo utilizando las coordenadas almacenadas. Se utilizan bucles y las instrucciones de la interrupción del BIOS (INT 10h) para dibujar líneas horizontales y verticales del rectángulo. Se incrementan o decrementan las coordenadas X e Y según corresponda, y se compara con las coordenadas finales para determinar si se ha completado el dibujo del rectángulo.

```
DIBUJAX0: ; Etiqueta DIBUJAX0: inicio del bucle

    INT 10h ; Llama a la interrupcion del BIOS para dibujar
    INC CX ; Incrementa la coordenada X
    CMP CX,CORX1 ; Compara la coordenada X con CORX1
    JNE DIBUJAX0 ; Salta a la etiqueta DIBUJAX0 si no son iguales

DIBUJAY1: ; Etiqueta DIBUJAY1: inicio del bucle

    INT 10h ; Llama a la interrupcion del BIOS para dibujar
    INC DX ; Incrementa la coordenada Y
    CMP DX, CORY1 ; Compara la coordenada Y con CORY1
    JNE DIBUJAY1 ; Salta a la etiqueta DIBUJAY1 si no son iguales

DIBUJAX1: ; Etiqueta DIBUJAX1: inicio del bucle

    INT 10h ; Llama a la interrupcion del BIOS para dibujar
    DEC CX ; Decrementa la coordenada X
    CMP CX, CORX0 ; Compara la coordenada X con CORX0
    JNE DIBUJAX1 ; Salta a la etiqueta DIBUJAX1 si no son iguales

DIBUJAY0: ; Etiqueta DIBUJAY0: inicio del bucle

    INT 10h ; Llama a la interrupcion del BIOS para dibujar
    DEC DX ; Decrementa la coordenada Y
    CMP DX, CORY0 ; Compara la coordenada Y con CORY0
    JNE DIBUJAY0 ; Salta a la etiqueta DIBUJAY0 si no son iguales

    MOV AH, 0h ; Limpia el registro AH
    INT 16h ; Llama a la interrupcion del BIOS para esperar una tecla
```

Finalmente, el programa espera la pulsación de una tecla antes de finalizar.

Ejecución:





Conclusión:

Este programa demuestra el nivel de control y precisión que se puede lograr al programar en lenguaje ensamblador, ya que se accede directamente a las funciones y características del hardware. Aunque puede resultar más complejo de entender y escribir en comparación con lenguajes de programación de alto nivel, el lenguaje ensamblador sigue siendo utilizado en situaciones donde se requiere un control más fino del hardware o se necesita optimizar el rendimiento.

Bibliografía:

Brey, B. B. (2006). Microprocesadores Intel : 8086/8088, 80186/80188, 80286, 80386 y 80486, Pentium, procesador Pentium Pro, Pentium II, Pentium III y Pentium 4: arquitectura, programación e interfaces. Pearson Educación.