Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías

DIVISIÓN DE TECNOLOGÍAS PARA LA INTEGRACIÓN CIBER-HUMANA

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES

Actividad 18(practica 10)

TEMA: “DISEÑO DE UN RETARDO EN ENSAMBLADOR”

NOMBRE DEL ESTUDIANTE: Padilla Perez Jorge Daray

NOMBRE DE LA MATERIA: Seminario de solución de problemas de traductores de lenguaje

NOMBRE DEL PROFESOR: Roberto Patiño Ruiz

17/03/2023

Titulo

“DISEÑO DE UN RETARDO EN ENSAMBLADOR”

BREVE RESUMEN (1/2 cuartilla)

El programa PROG10.EXE muestra en pantalla una cadena la cual va cambiando de color junto con una pantalla (rectángulo) que también van cambiando de colores.

Define una cadena de caracteres " " en el segmento de datos, la cual es la que se imprimirá en una posición centrada de la pantalla la cual cambiara de color hasta que se imprimen los 256 colores.

En resumen, el programa muestra en pantalla 1 cadena la cual cambia de color junto con una pantalla, con la utilización de retardos se pueden ver mas despacio o mas rápido, el programa se adaptó para ver los cambios cada 2 segundos en promedio.

PLANTEAMIENTO DEL PROBLEMA A RESOLVER:

Codificar el programa de la práctica 10 a fin de poder hacer que podamos mostrar la cadena puesta que cambie de colores y entender como funcionan los retardos en el emu8086.

RESULTADOS LOGRADOS (redacta un objetivo específico y los resultados alcanzados):

Se logro cumplir el objetivo principal ya que se pudo ejecutar el programa correctamente y entender cómo funciona la manera de hacer retardos(duración, y llamadas a estos).

FUNDAMENTO TEÓRICO (Hipótesis)

El programa es una aplicación de consola que inicializa y solicita al usuario que escriba un nombre. A continuación, llama repetidamente a varias funciones para mostrar el mensaje, leer la entrada del usuario, validar la entrada, cambiar de colores la cadena y la pantalla

OBJETIVO DEL PROGRAMA

Poder entender el funcionamiento del programa el cual imprime en pantalla una cadena que cambia de color junto con su pantalla, y gracias a los retardos implementados se pueda controlar el tiempo en que tarda en cambiar este color.

DESARROLLO (Incluir Código comentado, imágenes y/o diagramas)

```
PAGE 60,132          ;Numero de paginas programa
TITLE PRACT10.EXE    ;Nombre del programa
.MODEL SMALL         ;Habilita el segmento 64 bits
;_____________________________________
.STACK 64            ;Habilita la pila
.DATA                ;Habilita el segmento de datos
REG0 DW 00001H  ;0FFFFH     ;Habilitamos el registro REG0
REG1 DW 00001H       ;Habilitamos el registro REG1
REG2 DW 00001H       ;Habilitamos el registro REG2
COLOR DB 00H         ;Registro para los colores
MSJE DB "RETARDO","$" ;Cadena MSJE
;_____________________________________
.CODE                ;Inicio del codigo
INICIO PROC FAR      ;Funcion principal INICIO
MOV AX, @DATA        ;Mover el segmento de datos a AX
MOV DS, AX           ;Movemos el valor de AX a DS
CALL PANT1           ;LLamada a funcion PANT1
CALL CAMBIO          ;LLamada a funcion CAMBIO
CALL CURSOR1         ;LLamada a funcion CURSOR1
MOV AX,4C00H         ;Movemos el valor 4C00h a AX
INT 21H              ;Interrupcion 21H
INICIO ENDP          ;Terminamos la funcion inicio
;_____________________________________
PANT0 PROC NEAR      ;Inicio de la funcion PANT0
MOV AX,0600H         ;Movemos el valor 0600H a AX
MOV BH, COLOR        ;Movemos el valor de COLOR a BH
MOV CX,0205H         ;Movemos el valor 0205H a CX
MOV DX,1545H         ;Movemos el valor 1545H a DX
INT 10H              ;Interrupcion 10H
```

```
RET             ;Retornamos a la funcion principal
PANT0 ENDP      ;Fin de PANT0
;_____________________________________
PANT1 PROC NEAR     ;Inicio de la funcion PANT1
MOV AX,0600H        ;Movemos el valor 0600H a AX
MOV BH,10H          ;Movemos el valor 10H a BH
MOV CX,0000H        ;Movemos el valor 0000H a CX
MOV DX,1950H        ;Movemos el valor 1950H a DX
INT 10H             ;Interrupcion 10H
RET             ;Retornar a la funcion principal
PANT1 ENDP          ;Fin de PANT1
;_____________________________________
CURSOR0 PROC NEAR    ;Iniciamos la funcion CURSOR0
MOV AH,02H          ;Movemos el valor 02H a AH
MOV BH,00H          ;Movemos el valor 00H a BH
MOV DX,0C1AH        ;Movemos el valor 0C1AH a DX
INT 10H             ;Interrupcion 10H
RET             ;Retornamos a la funcion principal
CURSOR0 ENDP        ;Fin de CURSOR0
;_____________________________________
CURSOR1 PROC NEAR    ;Inicio de funcion CURSOR1
MOV AH,02H          ;Movemos el valor 02H a AH
MOV BH,00H          ;Movemos el valor 00H a BH
MOV DX,1400H        ;Movemos el valor 1400H a DX
INT 10H             ;Interrupcion 10H
RET             ;Retornamos a la funcion principal
CURSOR1 ENDP        ;Fin de CURSOR1
;_____________________________________
CAMBIO PROC NEAR     ;Inicio de funcion CAMBIO
CALL PANT0          ;Llamada a funcion CURSOR0
```

```
CALL CURSOR0        ;Llamada a funcion PANT0
CALL DESP           ;Llamada a funcion DESP
CALL RETARd         ;Llamada a funcion RETARd
CALL RETARi         ;Llamada a funcion RETARi
CALL RETARd         ;Llamada a funcion RETARd
CALL RETARi         ;Llamada a funcion RETARi
INC COLOR           ;Incrementador de COLOR
CMP COLOR,0FFH      ;Comparamos el valor 0FFH con COLOR
JE SALE             ;Saltamos a la funcion SALE si ZF es 1
JMP CAMBIO          ;Salto a la funcion CAMBIO
SALE: RET           ;Inicia la funcion SALE y retorna
CAMBIO ENDP         ;Terminamos la funcion CAMBIO
;_____________________________________
DESP PROC NEAR      ;Inicia la funcion DESP
MOV AH,09H          ;Movemos el valor 09H a AH
LEA DX, MSJE        ;Cargamos el valor de MSJE a DX
INT 21H             ;Interrupcion 21H
RET                 ;Retornamos a la funcion principal
DESP ENDP           ;Fin de DESP
;_____________________________________
RETARd PROC NEAR    ;Inicio de la funcion RETARd
DECR0: DEC REG0     ;Iniciamos la funcion DECR0;decrementa REG0
CMP REG0,00000H     ;Comparamos REG0 con 00000H
JE DECR1            ;Saltamos a la funcion DECR1 si ZF es 1
JMP DECR0           ;Salto a DECR0
DECR1: DEC REG1     ;Inicio de DECR1;Decrementamos REG1
CMP REG1,00000H     ;Comparamos REG1 con 00000H
JE DECR2            ;Saltamos a la funcion DECR2 si ZF es 1
JMP DECR1           ;Salto a DECR1
DECR2: DEC REG2     ;Inicio de DECR2;Decrementamos REG2
```

```
CMP REG2,00000H      ;Comparamos REG2 con 00000H
JE SALIR0            ;Saltamos a la funcion SALIR0 si ZF es 1
JMP DECR2            ;Salto a DECR2
SALIR0: RET          ;Inicia la funcion SALIR0 y retornamos
RETARd ENDP          ;Fin de RETARd
;_____________________________________
RETARi PROC NEAR     ;Inicio de la funcion RETARi
INCR0: INC REG0      ;Iniciamos la funcion INCR0;Incrementamos REG0
CMP REG0,00005H   ;0FFFFH   ;Comparamos REG0 con 0FFFFH
JE INCR1             ;Saltamos a la funcion INCR1 si ZF es 1
JMP INCR0            ;Salto a INCR0
INCR1: INC REG1      ;Inicio de INCR1;Incrementamos REG1
CMP REG1,00005H      ;Comparamos REG1 con 0FFFFH
JE INCR2             ;Saltamos a la funcion INCR2 si ZF es 1
JMP INCR1            ;Salto a INCR1
INCR2: INC REG2      ;Inicio de INCR2;Incrementamos REG2
CMP REG2,00005H      ;Comparamos REG2 con 0FFFFH
JE SALIR1            ;Saltamos a la funcion SALIR1 si ZF es 1
JMP INCR2            ;Salto a INCR2
SALIR1: RET          ;Inicia la funcion SALIR1 y retornamos
RETARi ENDP          ;Fin de RETARi
;_____________________________________
END INICIO           ;Terminamos la funcion INICIO
```

Documenta cada una de las líneas de código del resto del programa y explica en está sección del reporte, cada uno de los Procedimientos que a continuación se enlistan:

```
 - PANT0 PROC NEAR                  ;Establece el color de fondo de pantalla.
 - CAMBIO PROC NEAR                 ; es el procedimiento principal que llama a los otros
;procedimientos para mostrar un mensaje repetidamente con un retraso intermedio. Utiliza
;un bucle para disminuir un registro (REG0) hasta que llegue a cero, y luego salta a otro
;bucle que incrementa el color de fondo y continúa el proceso. El retraso es introducido por
;el bucle que disminuye el registro.
 - RETARd PROC NEAR                 ; disminuye los valores de REG0, REG1 y REG2 hasta
```

;que cada uno alcanza un valor de 00000H. Utiliza las subrutinas DECR0, DECR1 y DECR2
;para disminuir los valores de los registros, y la subrutina SALIR0 para salir del bucle
;cuando todos los registros se han reducido a 00000H.
  - RETARi PROC NEAR ; incrementa los valores de REG0, REG1 y REG2 hasta
;que cada uno alcanza un valor de 0FFFFH. Utiliza las subrutinas INCR0, INCR1 e INCR2
;para incrementar los valores de los registros, y la subrutina SALIR1 para salir del bucle
;cuando todos los registros se han incrementado a 0FFFFH.

CONCLUSIONES (Breve descripción de los resultados obtenidos)

Para concluir los resultados obtenidos fueron satisfactorios ya que se elaboró el programa y funciono de la manera correcta, al momento de su realización en clase no se pudo concluir por falta de tiempo.

Además, se logró entender de manera correcta el programa y el cómo funcionan los retardos que es lo importante en esta practica.

REFERENCIAS BIBLIOGRÁFICAS (Formato APA, en caso de consultar otras fuentes)

Barlau, S. (2022, 3 abril). *¿Qué es la instrucción RET en microprocesador?* – – –

Veintipico. https://veintipico.com/que-es-la-instruccion-ret-en-microprocesador/