Actividad 10

CARRERA: Ingeniería en Computación

NOMBRE: *Efrain Robles Pulido*

CÓDIGO: 221350095

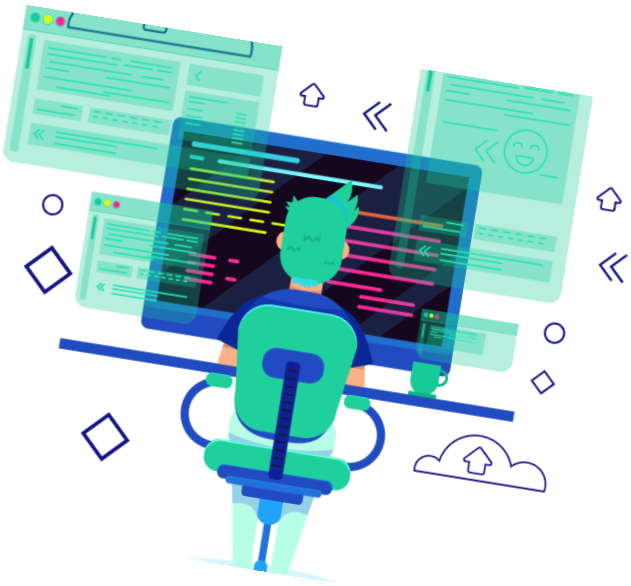
MATERIA: Seminario de Solución de Problemas de Traductores de Lenguajes I

MAESTRA: José Juan Meza Espinoza

SECCIÓN: D09 **CALENDARIO:** 2023A

UNIVERSIDAD DE GUADALAJARA

CUCEI



Desarrollo

- Dibujar un rectángulo.
- Posicionar con el mouse la coordenada 1 (x0, y0) y la segunda coordenada (x1, y1).
- Dibujarlo.

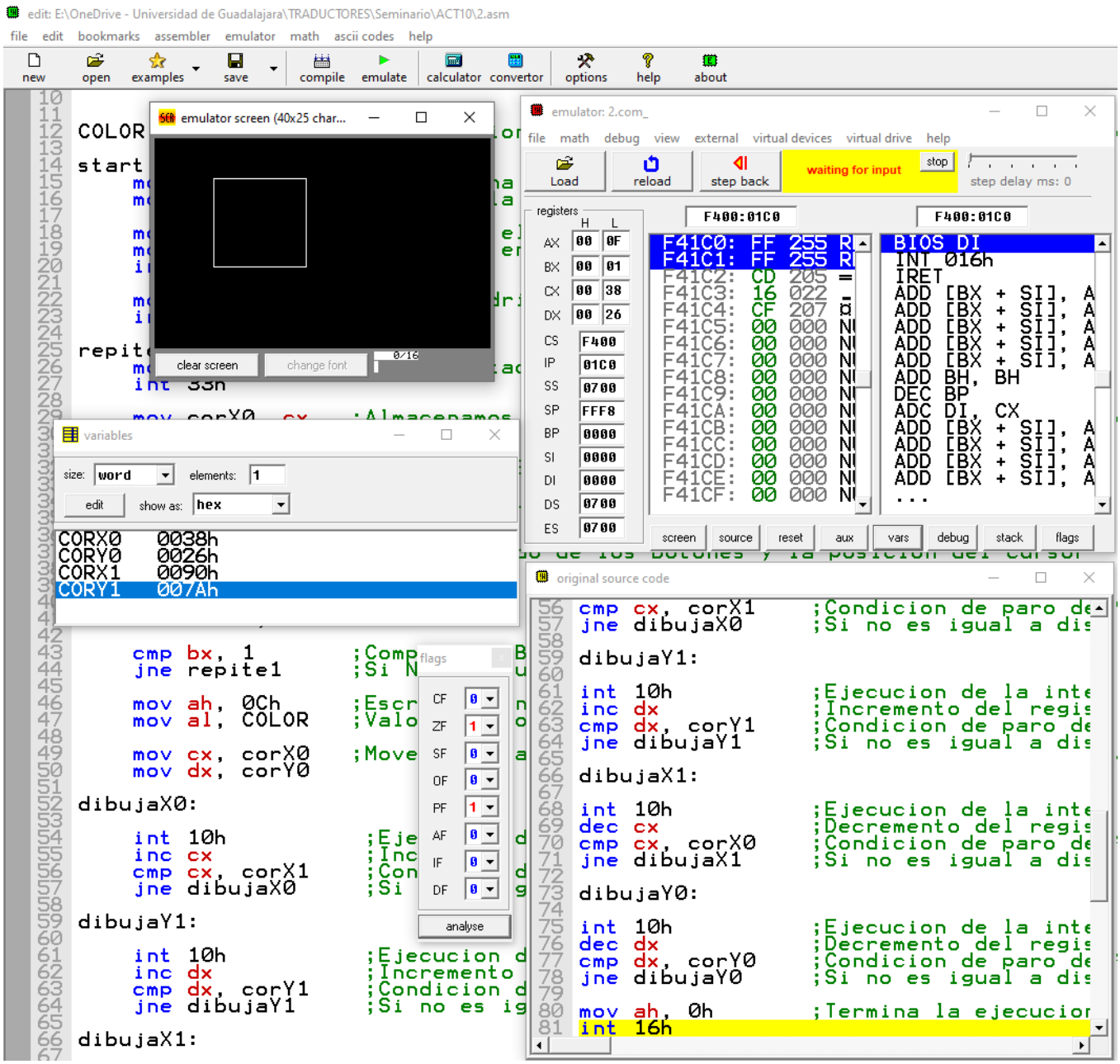
El código comenzará con el almacenamiento de la dirección de memoria de nuestras variables a utilizar, como las 2 coordenadas del mouse a utilizar y definiendo una constante numérica para determinar el color del píxel a pintar, después se establecerá el modo de video en Ah en 0 y su dimensión de la pantalla en Al en 13h mediante la interrupción 10h, después moveremos en Ah el valor de 0 para ejecutar la interrupción 33h para reiniciar y verificar el driver del mouse.
A continuación, se iniciará un ciclo en el que se estará moviendo el valor de 3h en Ax con interrupción 33h para leer el estado de los botones del mouse y su respectiva posición en la pantalla de video, en el que se estará almacenando el estado de los botones en el registro Bx, y la coordenada 0´s, X0 en Cx y Y0 en Dx, y a su vez se estará almacenando en memoria esas mismas coordenadas. También estaremos comparando si en Bx tenemos el valor 1 (botón izquierdo) para realizar el respectivo salto en “repite”, si no es igual a 1 se estará ciclando hasta apretar el botón del mouse.
Una vez presionado el botón, pasaremos a otro ciclo en donde repetiremos el mismo procedimiento, pero ahora lo estaremos almacenando la coordenada 1´s, es sus respectivos ejes, X1 en Cx y Y1 en Dx, estará saltando en “repite1” para ciclar esta parte. A continuación, movemos 0Ch a Ah para configurar para escribir un punto en pantalla y moveremos el valor COLOR en Al para configurar el color de aquel punto a pintar, después moveremos las coordenadas X0, Y0 a Cx y Dx respectivamente para empezar a pintar este punto. En el que estaremos ciclando y incrementando en 1, el valor de la coordenada X0 (Cx) para irlo pintando continuamente con la condición de que no sea igual a la coordenada X1. Pasando a otro muy parecido en el que ahora es incrementara en 1 el valor de Dx (en el eje de las Ys) y se cambiara el valor de la condición con la coordenada Y1.
El siguiente ciclo es parecido a los anteriores, pero ahora se decrementará en 1 valor de Cx (eje de la Xs) hasta llegar a la coordenada de X0, y el siguiente ciclo con decrementara en 1 el valor en Dx(eje de la Ys) hasta llegar a la coordenada Y0.
Por lo que cada ciclo tendrá su respectiva etiqueta para realizar sus ciclos individuales para formar el rectángulo. Finalmente terminara el código con la instrucción 0h de la interrupción 16h.

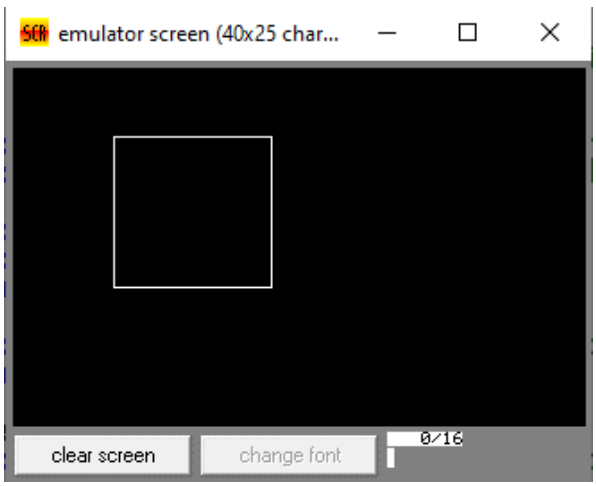
edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm

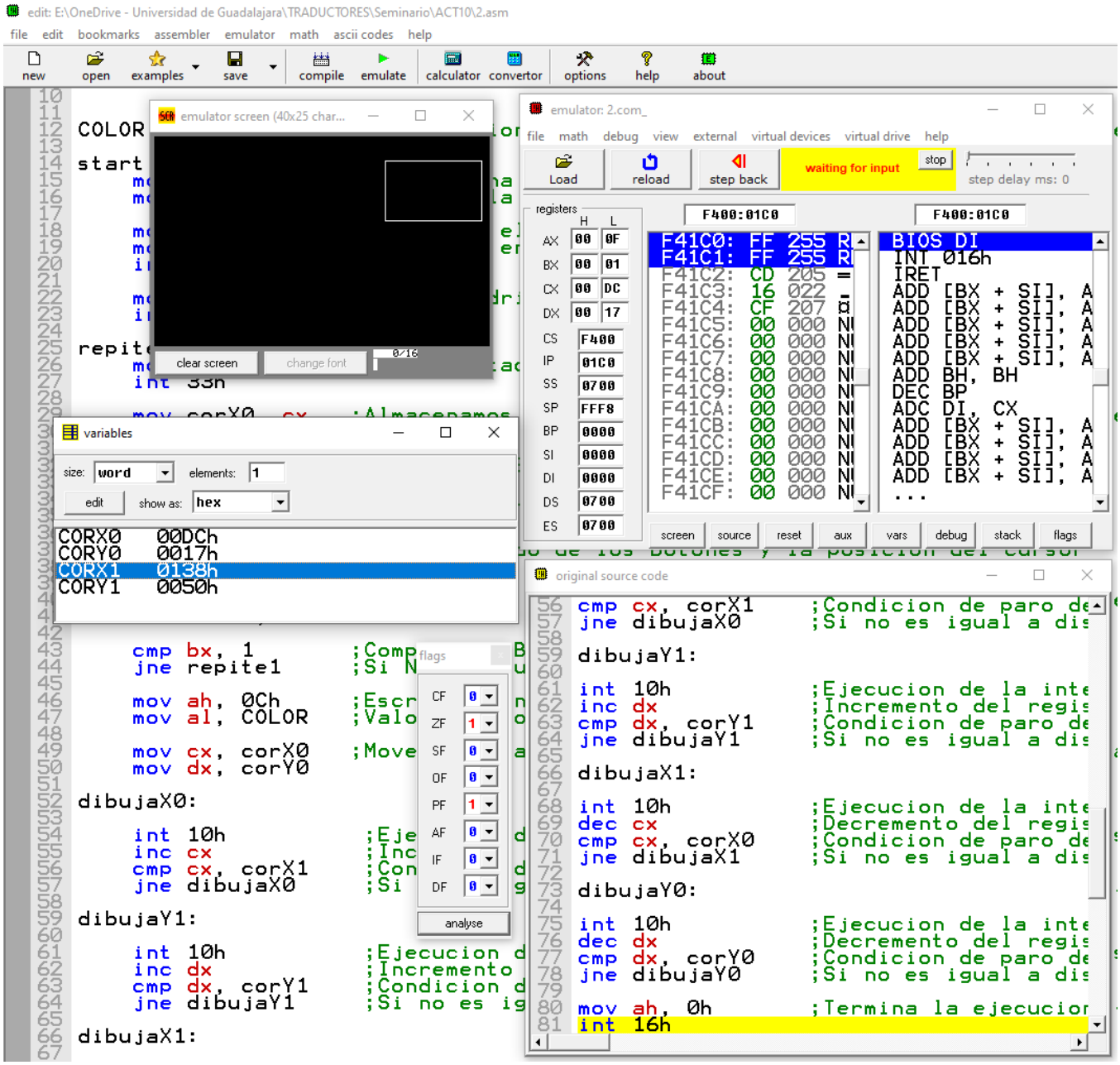
file edit bookmarks assembler emulator math ascii codes help

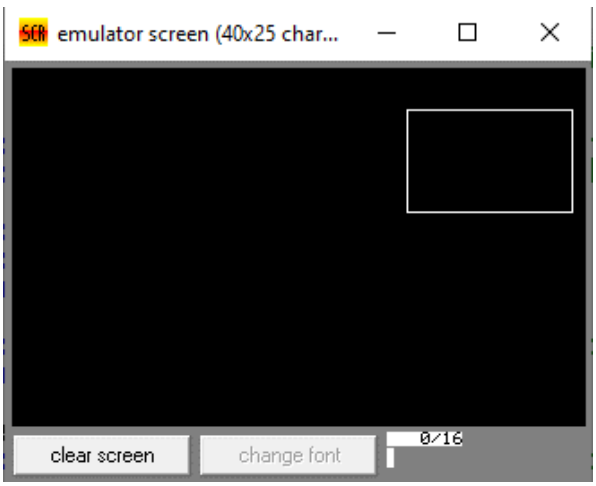
new open examples save compile emulate calculator convertor options help about

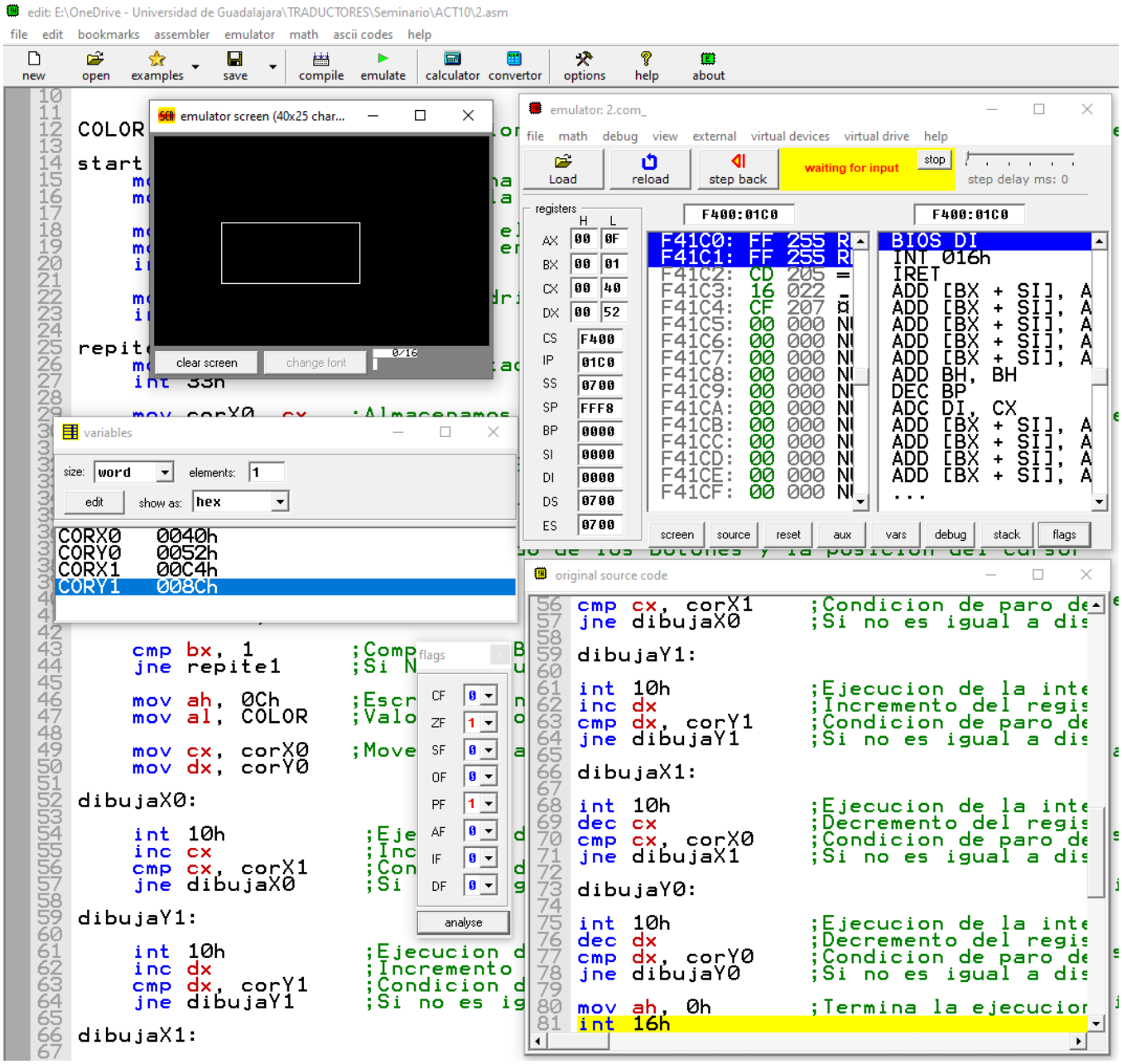
```
org 100h

jmp start

corX0 dw ?              ;Valor en memoria para la coordena X0 del mouse
corY0 dw ?              ;Valor en memoria para la coordena Y0 del mouse

corX1 dw ?              ;Valor en memoria para la coordena X1 del mouse
corY1 dw ?              ;Valor en memoria para la coordena Y1 del mouse


COLOR EQU 0Fh          ;La expresion COLOR es igual a 0Fh (255d), como constante numerica

start:
    mov ax, @data      ;Se almacena la direccion de memoria de nuestro arreglos
    mov ds, ax         ;Guardandola en el registro para las direcciones

    mov ah, 0          ;Establece el modo video
    mov al, 13h        ;320 x 200 en grafico
    int 10h

    mov ah,0h          ;Verifica driver del raton
    int 33h

repite:
    mov ax, 03h        ;Lee el estado de los botones y la posicion del cursor
    int 33h

    mov corX0, cx      ;Almacenamos en memoria las coordenadas X0 y Y0 del mouse
    mov corY0, dx

    cmp bx, 1          ;Compara si Bx es 1 ya que significa el boton Izquierdo
    mov bx, 0
    jne repite         ;Si NO es igual a 1, saltara a la etiqueta repite

repite1:
    mov ax,03h         ;Lee el estado de los botones y la posicion del cursor
    int 33h

    mov corX1, cx      ;Almacenamos en memoria las coordenadas X1 y Y1 del mouse
    mov corY1, dx

    cmp bx, 1          ;Compara si Bx es 1 ya que significa el boton Izquierdo
    jne repite1        ;Si NO es igual a 1, saltara a la etiqueta dibuja

    mov ah, 0Ch        ;Escribira un punto en la pantalla en modo video, con lo que tenga en Cx y Dx, con la int 10h
    mov al, COLOR      ;Valor del color a pintar

    mov cx, corX0      ;Moveremos las coordenas de X0, Y0, para empezar a dibujar la figura
    mov dx, corY0

dibujaX0:

    int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesarios
    inc cx             ;Incremento del registro Cx para recorrer en el eje X
    cmp cx, corX1      ;Condicion de paro del ciclo
    jne dibujaX0       ;Si no es igual a distancia X1, saltara a la etiqueta dibujaX0

dibujaY1:

    int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesario
    inc dx             ;Incremento del registro Dx para recorrer en el eje Y
    cmp dx, corY1      ;Condicion de paro del ciclo
    jne dibujaY1       ;Si no es igual a distancia Y1, saltara a la etiqueta dibujaY1

dibujaX1:

    int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesario
    dec cx             ;Decremento del registro Cx para recorrer en el eje X
    cmp cx, corX0      ;Condicion de paro del ciclo
    jne dibujaX1       ;Si no es igual a distancia X0, saltara a la etiqueta dibujaX1

dibujaY0:

    int 10h            ;Ejecucion de la interrupcion 10h, ya con sus parametros necesario
    dec dx             ;Decremento del registro Cx para recorrer en el eje Y
    cmp dx, corY0      ;Condicion de paro del ciclo
    jne dibujaY0       ;Si no es igual a distancia Y0, saltara a la etiqueta dibujaY0

    mov ah, 0h         ;Termina la ejecucion del programa
    int 16h

ret
```

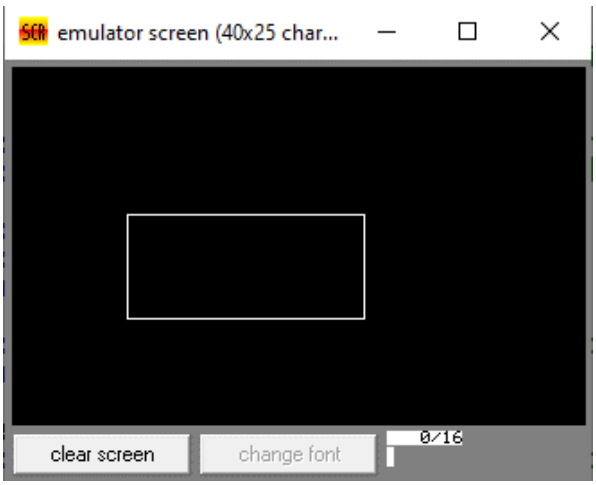

edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator screen (40x25 char...
clear screen
change font
0/16
emulator: 2.com_
file math debug view external virtual devices virtual drive help
Load reload step back waiting for input stop step delay ms: 0
registers
AX BX CX DX CS IP SS SP BP SI DI DS ES
F400:01C0
F41C0: FF 255 R
F41C1: FF 255 R
BIOS DI
INT 016h
IRET
ADD [BX + SI], A
ADD BH, BH
DEC BP
ADC DI, CX
screen source reset aux vars debug stack flags
variables
size: word elements: 1
edit show as: hex
CORX0 0038h
CORY0 0026h
CORX1 0090h
CORY1 007Ah
flags
CF ZF SF OF PF AF IF DF
analyse
original source code
cmp cx, corX1 ;Condicion de paro de
jne dibujaX0 ;Si no es igual a dis
dibujaY1:
int 10h ;Ejecucion de la inte
inc dx ;Incremento del regis
cmp dx, corY1 ;Condicion de paro de
jne dibujaY1 ;Si no es igual a dis
dibujaX1:
int 10h ;Ejecucion de la inte
dec cx ;Decremento del regis
cmp cx, corX0 ;Condicion de paro de
jne dibujaX1 ;Si no es igual a dis
dibujaY0:
int 10h ;Ejecucion de la inte
dec dx ;Decremento del regis
cmp dx, corY0 ;Condicion de paro de
jne dibujaY0 ;Si no es igual a dis
mov ah, 0h ;Termina la ejecucion
int 16h

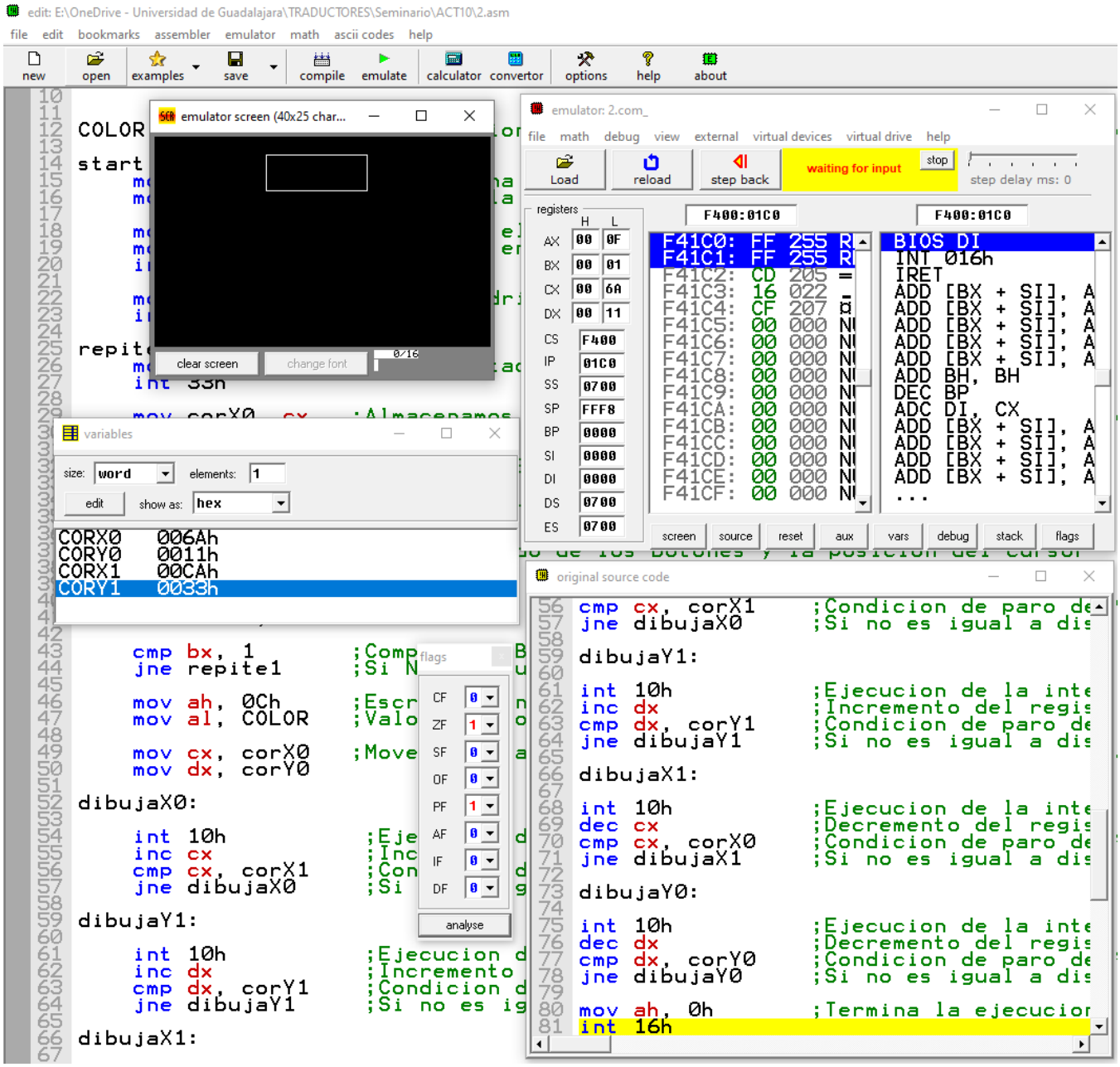

emulator screen (40x25 char...
clear screen
change font
0/16

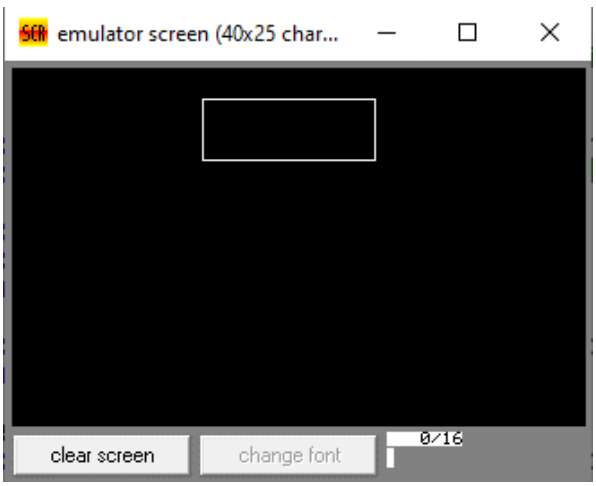

edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator screen (40x25 char...
clear screen
change font
0/16
emulator: 2.com_
file math debug view external virtual devices virtual drive help
Load reload step back waiting for input stop step delay ms: 0
registers
H L
AX 00 0F
BX 00 01
CX 00 DC
DX 00 17
CS F400
IP 01C0
SS 0700
SP FFF8
BP 0000
SI 0000
DI 0000
DS 0700
ES 0700
F400:01C0
F400:01C0
F41C0: FF 255 R
F41C1: FF 255 R
F41C2: CD 205 =
F41C3: 16 022 -
F41C4: CF 207 ¤
F41C5: 00 000 NU
F41C6: 00 000 NU
F41C7: 00 000 NU
F41C8: 00 000 NU
F41C9: 00 000 NU
F41CA: 00 000 NU
F41CB: 00 000 NU
F41CC: 00 000 NU
F41CD: 00 000 NU
F41CE: 00 000 NU
F41CF: 00 000 NU
BIOS DI
INT 016h
IRET
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD BH, BH
DEC BP
ADC DI, CX
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
...
screen source reset aux vars debug stack flags
variables
size: word elements: 1
edit show as: hex
CORX0 00DCh
CORY0 0017h
CORX1 0138h
CORY1 0050h
flags
CF 0
ZF 1
SF 0
OF 0
PF 1
AF 0
IF 0
DF 0
analyse
original source code
cmp cx, corX1 ;Condicion de paro de
jne dibujaX0 ;Si no es igual a dis
dibujaY1:
int 10h ;Ejecucion de la inte
inc dx ;Incremento del regis
cmp dx, corY1 ;Condicion de paro de
jne dibujaY1 ;Si no es igual a dis
dibujaX1:
int 10h ;Ejecucion de la inte
dec cx ;Decremento del regis
cmp cx, corX0 ;Condicion de paro de
jne dibujaX1 ;Si no es igual a dis
dibujaY0:
int 10h ;Ejecucion de la inte
dec dx ;Decremento del regis
cmp dx, corY0 ;Condicion de paro de
jne dibujaY0 ;Si no es igual a dis
mov ah, 0h ;Termina la ejecucion
int 16h
COLOR
start
repite
int 33h
cmp bx, 1 ;Comp
jne repite1 ;Si N
mov ah, 0Ch ;Escr
mov al, COLOR ;Valo
mov cx, corX0 ;Move
mov dx, corY0
dibujaX0:
int 10h ;Eje
inc cx ;Inc
cmp cx, corX1 ;Con
jne dibujaX0 ;Si
dibujaY1:
int 10h ;Ejecucion d
inc dx ;Incremento
cmp dx, corY1 ;Condicion d
jne dibujaY1 ;Si no es ig
dibujaX1:


emulator screen (40x25 char...
clear screen
change font
0/16


edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator screen (40x25 char...
clear screen
change font
0/16
emulator: 2.com_
file math debug view external virtual devices virtual drive help
Load reload step back waiting for input stop step delay ms: 0
registers
AX BX CX DX CS IP SS SP BP SI DI DS ES
F400:01C0
BIOS DI
INT 016h
IRET
variables
size: word elements: 1
edit show as: hex
CORX0 0040h
CORY0 0052h
CORX1 00C4h
CORY1 008Ch
flags
CF ZF SF OF PF AF IF DF
analyse
original source code
cmp cx, corX1 ;Condicion de paro de
jne dibujaX0 ;Si no es igual a dis
dibujaY1:
int 10h ;Ejecucion de la inte
inc dx ;Incremento del regis
cmp dx, corY1 ;Condicion de paro de
jne dibujaY1 ;Si no es igual a dis
dibujaX1:
int 10h ;Ejecucion de la inte
dec cx ;Decremento del regis
cmp cx, corX0 ;Condicion de paro de
jne dibujaX1 ;Si no es igual a dis
dibujaY0:
int 10h ;Ejecucion de la inte
dec dx ;Decremento del regis
cmp dx, corY0 ;Condicion de paro de
jne dibujaY0 ;Si no es igual a dis
mov ah, 0h ;Termina la ejecucion
int 16h


emulator screen (40x25 char...
clear screen
change font
0/16


edit: E:\OneDrive - Universidad de Guadalajara\TRADUCTORES\Seminario\ACT10\2.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator screen (40x25 char...
clear screen
change font
0/16
emulator: 2.com_
file math debug view external virtual devices virtual drive help
Load reload step back waiting for input stop step delay ms: 0
registers
AX BX CX DX CS IP SS SP BP SI DI DS ES
F400:01C0
BIOS DI
INT 016h
IRET
variables
size: word elements: 1
edit show as: hex
CORX0 006Ah
CORY0 0011h
CORX1 00CAh
CORY1 0033h
flags
CF ZF SF OF PF AF IF DF
analyse
original source code
screen source reset aux vars debug stack flags
COLOR
start
repite
dibujaX0:
dibujaY1:
dibujaX1:
dibujaY0:
int 33h
cmp bx, 1 ;Comp
jne repite1 ;Si N
mov ah, 0Ch ;Escr
mov al, COLOR ;Valo
mov cx, corX0 ;Move
mov dx, corY0
int 10h ;Eje
inc cx ;Inc
cmp cx, corX1 ;Con
jne dibujaX0 ;Si
int 10h ;Ejecucion d
inc dx ;Incremento
cmp dx, corY1 ;Condicion d
jne dibujaY1 ;Si no es ig
cmp cx, corX1 ;Condicion de paro de
jne dibujaX0 ;Si no es igual a dis
dibujaY1:
int 10h ;Ejecucion de la inte
inc dx ;Incremento del regis
cmp dx, corY1 ;Condicion de paro de
jne dibujaY1 ;Si no es igual a dis
dibujaX1:
int 10h ;Ejecucion de la inte
dec cx ;Decremento del regis
cmp cx, corX0 ;Condicion de paro de
jne dibujaX1 ;Si no es igual a dis
dibujaY0:
int 10h ;Ejecucion de la inte
dec dx ;Decremento del regis
cmp dx, corY0 ;Condicion de paro de
jne dibujaY0 ;Si no es igual a dis
mov ah, 0h ;Termina la ejecucion
int 16h


emulator screen (40x25 char...
clear screen
change font
0/16

Conclusión

Al principio me costo trabajo como empezar a dibujar el rectángulo. Así que primero se empezó por realizar el código para almacenar las coordenadas de ambos puntos en la memoria en el que reutilice parte del código de la práctica anterior. Para después ir aumentando en uno en uno los puntos de cada eje respectivamente. Por lo que se iba recorriendo los ejes respectivamente para dibujar el píxel correspondiente, teniendo las coordenadas almacenadas como condiciones para parar o continuar dibujando. Entonces, una vez entendido la lógica se puedo realizar correctamente la práctica, así que considero que se podría hacer aún más eficiente el código, pero con este cumple su propósito.

Bibliografía:

Brey, B. B. (2006). Microprocesadores Intel - 7 Edición (7a). Pearson Publications Company.