|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FACULTAD: | **Tecnología Informática** | | | | |
| CARRERA: | **Analista Programador** | | | | |
| ALUMNO/A: | **Cynthia A. Della Maggiora** | | | | |
| SEDE: |  | | LOCALIZACIÓN: | **Buenos Aires** | |
| ASIGNATURA: | **Sistemas de Computación II** | | | | |
| COMISIÓN: |  | | TURNO: |  | |
| PROFESOR: | **Douce** | | FECHA: | **16/07/21** | |
| TIEMPO DE RESOLUCIÓN: | | **120 min** | EXAMEN PARCIAL N°: | | **2 (dos)** |
| MODALIDAD DE RESOLUCIÓN: | | | A Distancia Sincrónico - Escrito | | |
| CALIFICACIÓN: | | |  | | |
| RESULTADOS DE APRENDIZAJE ESPERADO: **Asegurar la comprensión de Interrupciones, manejo de la pila y programación en Assembler.** | | | | | |

Dado el siguiente estado de la UCP (los registros que no aparecen no son usados por el programa que se está ejecutando):

AX, IP, RE, SP, CS, DS, SS (tomar los valores de la planilla provista por el profesor)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alumno | **AX** | **IP** | **RE** | **SP** | **CS** | **DS** | **SS** | **CALL** | **INT** | **Vec: IP** | **Vec: CS** |
|  | F012 | 2300 | 7574 | 6890 | 5000 | 5000 | B100 | 4399 | 64 | 4713 | 8000 |

**PUNTO 1:**

Si la siguiente instrucción a ejecutar es CALL **4399** que ocupa 3 bytes, **indicar:**

**1.a** Mediante las componentes XXXX:YYYY de la memoria, en qué dirección de memoria está la instrucción CALL 4399 y cuál es la dirección que aparecerá en el bus de direcciones al buscar esa instrucción.

**1.b** Qué movimientos de información ocurren durante la ejecución de CALL **4399** y con qué valores quedan los registros involucrados.

**1.c** Cómo queda la pila luego de la ejecución de CALL **4399**.

**1.d** Represente usando DEBUG esta ejecución mostrando con el comando R y T como se produce el salto y como cambia SP, y con E como ha cambiado la pila.

**PUNTO 2:**

Si la subrutina empieza con PUSH AX, **indicar:**

**2.a** Mediante las componentes XXXX:YYYY, donde se encuentra la instrucción PUSH AX y cual es la dirección que aparecerá en el bus de direcciones al buscarse la instrucción PUSH AX.

**2.b** Qué movimientos de información ocurren durante la ejecución de PUSH AX y con qué valores quedan los registros involucrados.

**2.c** Cómo queda la pila luego de la ejecución de PUSH AX

**PUNTO 3:**

Si a PUSH AX (de un byte de longitud) sigue la instrucción INT 64 (de dos bytes de longitud), cuyo vector de interrupciones contiene las componentes CS=8000 e IP=4713, **indicar:**

**3.a** Mediante las componentes XXXX:YYYY, en qué posición de memoria se encuentra INT nn y cómo aparecerá la misma en el bus de direcciones.

**3.b** Qué movimientos de información se realizan durante su ejecución para poder resguardar información en la Pila.

**3.c** Cómo queda la pila luego de los movimientos de información que dispara la ejecución de INT 64.

**3.d** Cuáles son las acciones que faltan realizar para concluir con la ejecución de INT nn en relación con el flag I y la localización de la subrutina del SO que atiende INT nn. Describir la ubicación del vector de interrupción dentro de la Tabla de vectores.

**3.e** Con qué instrucción termina esta rutina que atiende la Interrupción por software, y qué movimientos de información genera su ejecución.

**3.f** Cómo queda la pila cuando finaliza la ejecución de esta subrutina del SO.

**3.g** Se puede enmascarar este tipo de interrupciones?

**3.h** Represente usando DEBUG esta ejecución mostrando con el comando R y T como se produce el salto y como cambia SP, y con E como ha cambiado la pila y el contenido del vector de interrupción

**PUNTO 4:**

Puesto que la subrutina empezó con PUSH AX e INT 64, **indicar:**

**4.a** Con qué instrucciones debe terminar.

**4.b** Qué movimientos ocurren cuando se ejecuta cada una, y cómo queda la pila luego de cada ejecución.

**Respuestas:**

**PUNTO 1:**

**a** La instrucción CALL 4399 está en la dirección de memoria 5000:2300 y la dirección que aparecerá en el bus de direcciones al buscar esa instrucción será 52300 en hexadecimal (0101 0010 0011 0000 0000 binario).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 5 | 0 | 0 | 0 | 0 |
| + |  |  |  |  |  |
|  |  | 2 | 3 | 0 | 0 |
|  | 5 | 2 | 3 | 0 | 0 |

CS:IP

5000:2300

**1.b** Se detallan a continuación los movimientos de información y el valor con el que quedan los registros involucrados:

CALL 4399

SP 🡨 SP-2 SP=6890-2=688E

[SP] 🡨 IP [SP]=[688E]=2303 (2300+3)

IP 🡨 DA+IP (2303+2096)=IP=4399

**1.c** La pila queda de la siguiente forma:

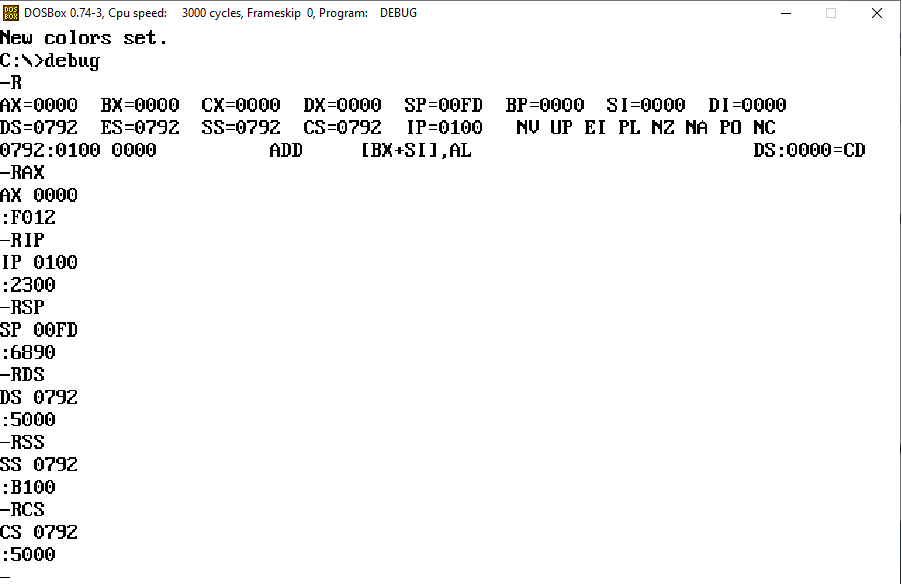
SS:SP

B100:688E **03 (IP)** 🡨**cima de la pila**

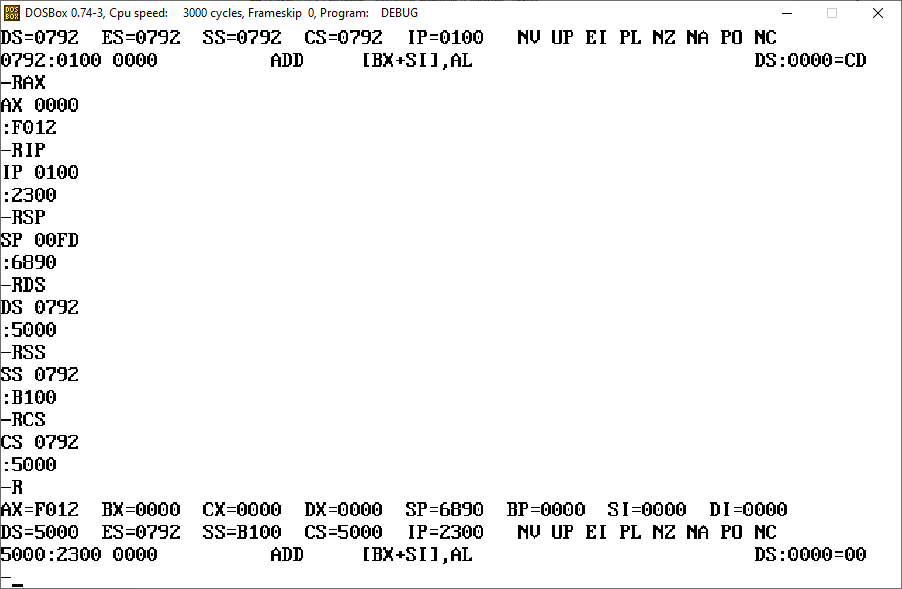
B100:688F **23 (IP)**

B100:6890 **xx**

**1.d** Se representa pasa a paso lo solicitado:

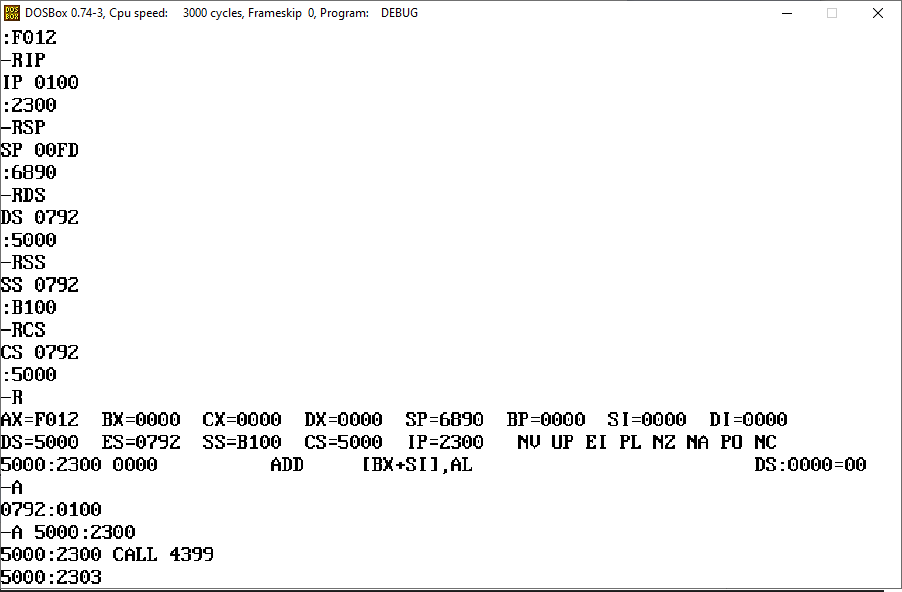
****

*Imagen 1: En esta pantalla del Debug, se inicializa todos los datos proporcionados por el profesor para realizar el parcial (se cargan todos menos el Registro de Estado (RE) ya que no es posible manipularlo).*

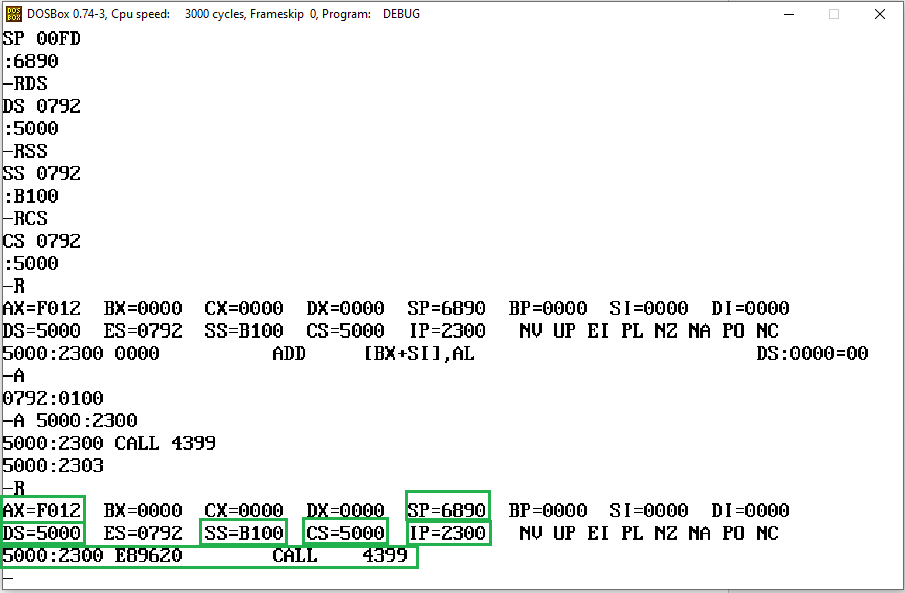
****

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alumno | **AX** | **IP** | **RE** | **SP** | **CS** | **DS** | **SS** | **CALL** | **INT** | **Vec: IP** | **Vec: CS** |
|  | F012 | 2300 | 7574 | 6890 | 5000 | 5000 | B100 | 4399 | 64 | 4713 | 8000 |

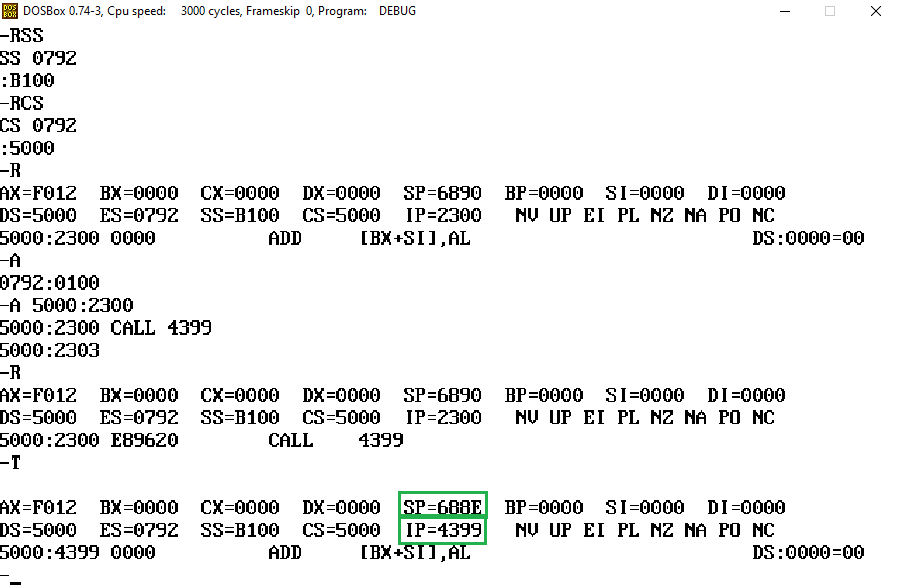
*Imagen 2: Con el comando R, se chequea que todo se encuentre cargado ok según los requerimientos.*

****

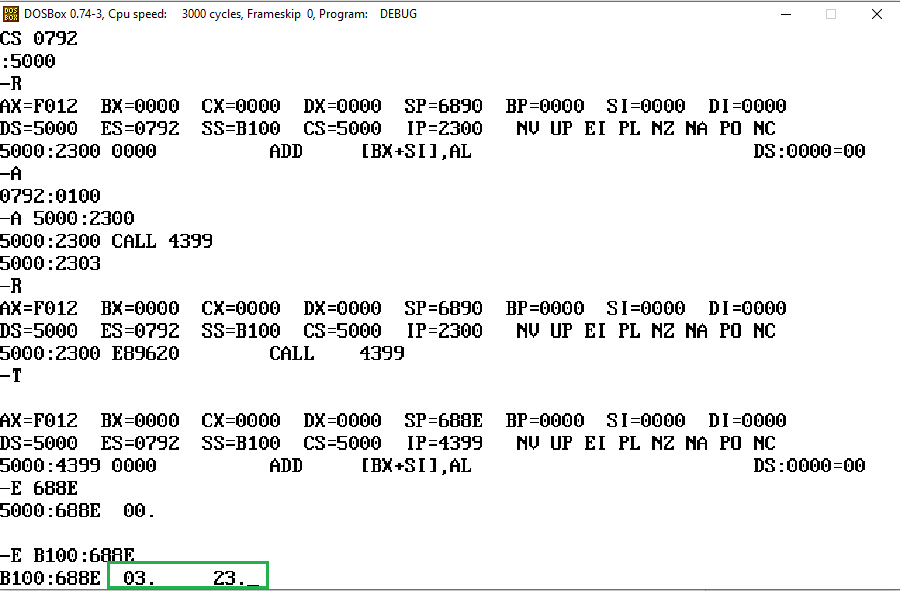
*Imagen 3: Con el comando A, compruebo que no estoy en 5000:2300 como me pide el ejercicio y modifico con el mismo comando para encontrarme en el segmento correcto y cargo CALL 4399 en la dirección especificada en la consigna; seguido de eso, doy enter y compruebo que la instrucción cargada ocupa 3 posiciones.*

****

*Imagen 4: Con el comando R, controlo los valores para luego ejecutar.*

****

*Imagen 5: Utilizo al comando T, y ejecuto. Controlo entonces el punto 1.b se encuentra ok, ya que SP=688E e IP=4399.*

****

*Imagen 6: Paso a utilizar el comando E para controlar los valores de la pila expresados en el ejercicio 1.c. Como en una primer instancia el debug me muestra otra dirección de memoria y no la correspondiente a la pila, vuelvo a utilizar el comando E para revisar la dirección correcta (B100:688E) y compruebo con barra espaciadora que los valores se encuentran ok, dando por finalizada la comprobación.*

**PUNTO 2:**

Si la subrutina empieza con PUSH AX se indica**:**

**2.a** La instrucción PUSH AX está en la dirección 5000:4399 y la dirección que aparecerá en el bus de direcciones es 54399 hexadecimal (0101 0100 0011 1001 1001 binario).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 5 | 0 | 0 | 0 | 0 |
| + |  |  |  |  |  |
|  |  | 4 | 3 | 9 | 9 |
|  | 5 | 4 | 3 | 9 | 9 |

CS:IP

5000:4399

**2.b** Se detallan a continuación los movimientos de información que ocurren durante la ejecución de PUSH AX y con qué valores quedan los registros involucrados:

PUSH AX

SP 🡨 SP-2 SP=688E-2=688C

[SP] 🡨 AX AX= F012(AX no cambia, se mantiene)

[SP]=[688C]=F012

**2.c** Cómo queda la pila luego de la ejecución de PUSH AX

SS:SP

B100:688C **12 (AX)** 🡨**cima de la pila**

B100:688D **F0 (AX)**

B100:688E **03 (IP)**

B100:688F **23 (IP)**

B100:6890 **XX**

**PUNTO 3:**

Si a PUSH AX (de un byte de longitud) sigue la instrucción INT 64 (de dos bytes de longitud), cuyo vector de interrupciones contiene las componentes CS=8000 e IP=4713, **se indica:**

**3.a** INT 64 se encuentra en la dirección 5000:439A porque PUSH AX se encontraba en 5000:4399 (ocupando solo una posición) y aparecerá la misma en el bus de direcciones 5439A hexadecimal (0101 0100 0011 1001 1010 binario).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 5 | 0 | 0 | 0 | 0 |
| + |  |  |  |  |  |
|  |  | 4 | 3 | 9 | A |
|  | 5 | 4 | 3 | 9 | A |

CS:IP

5000:439A (IP=4399+1)

**3.b** Qué movimientos de información se realizan durante su ejecución para poder resguardar información en la Pila.

INT 64

• SP 🡨 SP-2 SP=688C-2 SP=688A

• [SP] 🡨 RE RE=7574 se guarda en SP

• SP 🡨 SP-2 688A-2=6888

• [SP] 🡨 CS CS=5000 se guarda en SP

• SP 🡨 SP-2 6888-2=6886

• [SP] 🡨 IP 439C se guarda en SP

**SP= 6886** 🡨**cima de la pila**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alumno | **AX** | **IP** | **RE** | **SP** | **CS** | **DS** | **SS** | **CALL** | **INT** | **Vec: IP** | **Vec: CS** |
|  | F012 | 2300 | 7574 | 6890 | 5000 | 5000 | B100 | 4399 | 64 | 4713 | 8000 |

**3.c** Cómo queda la pila luego de los movimientos de información que dispara la ejecución de INT 64.

SS:SP

B100:6886  **9C(IP)** 🡨**cima de la pila**

B100:6887  **43(IP)**

B100:6888  **00(CS)**

B100:6889  **50(CS)**

B100:688A  **74(RE)**

B100:688B **75(RE)**

B100:688C **12 (AX)**

B100:688D **F0 (AX)**

B100:688E **03 (IP)**

B100:688F **23 (IP)**

B100:6890 **XX**

**3.d** Los movimientos que faltan realizar son los siguientes:

• IP 🡨 [XX\*4] 64\*4=190

• CS 🡨 [XX\*4+2] 64\*4+2=192

• DI 🡨 Flag I=0 (estaba en 1, ejecuto int y pasa a 0)

*IP= 4713 (queda con el valor del vector)*

*CS= 8000 (queda con el valor del vector)*

**El vector de interrupción estará en:**

0000:0190 13

0000:0191 47

0000:0192 00

0000:0193 80

**3.e** La instrucción con la que termina esta rutina que atiende la Interrupción por software es IRET. Dicha instrucción realiza los siguientes movimientos:

IRET (“interrupt return”)

* IP  [SP] SP=6886
* SP  SP+2 SP=6888 (6886+2)
* CS  [SP] SP=CS
* SP  SP+2 SP=688A (6888+2)
* RE  [SP] RE=SP=688A
* SP  SP+2 SP=688C (688A+2)
* EI  Flag I=1

**3.f** La pila queda con la misma situación que antes de ejecutar la operación INT 64. Se detalla:

B100:688C **12 (AX)** 🡨**cima de la pila**

B100:688D **F0 (AX)**

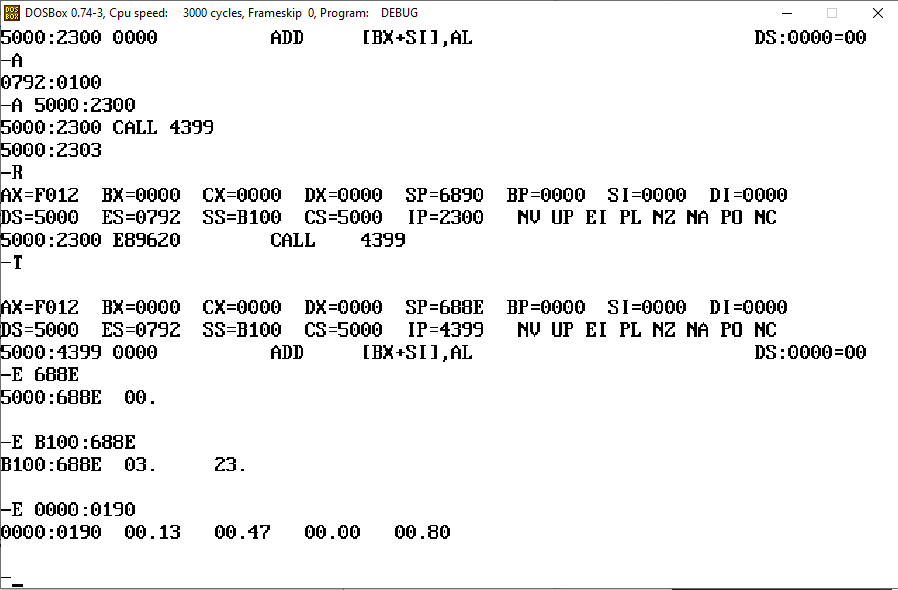
B100:688E **03 (IP)**

B100:688F **23 (IP)**

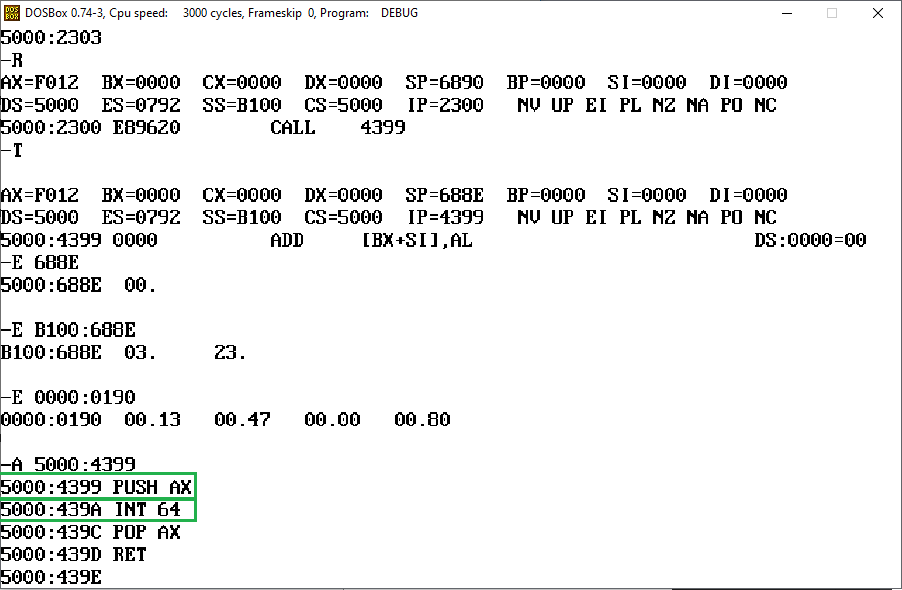
B100:6890 **XX**

**3.g** Si, se puede enmascarar este tipo de interrupciones por medio del registro RE (que contiene al flag I y cambia su estado con la ejecución de la instrucción INT NN). Antes de ejecutar la interrupción INT NN, el flag I se encuentra en 1, y luego, pasa a 0 deshabilitando (enmascarando) cualquier interrupción por hardware que ocurra mientras se ejecuta la subrutina que atiende a INT NN. Esto ocurre porque se está ejecutando una subrutina del sistema operativo, por lo que, al ponerse el flag I=0, la CPU se pone en modo “kernell” (y cuando el flag I=1, está en modo usuario).

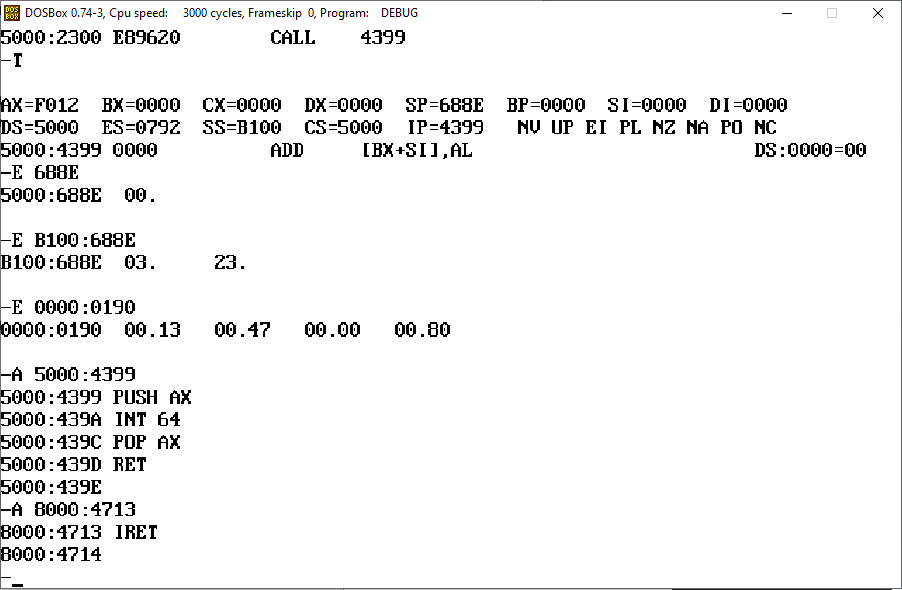
**3.h** Se representa usando DEBUG esta ejecución mostrando con el comando R y T como se produce el salto y como cambia SP, y con E como ha cambiado la pila y el contenido del vector de interrupción

****

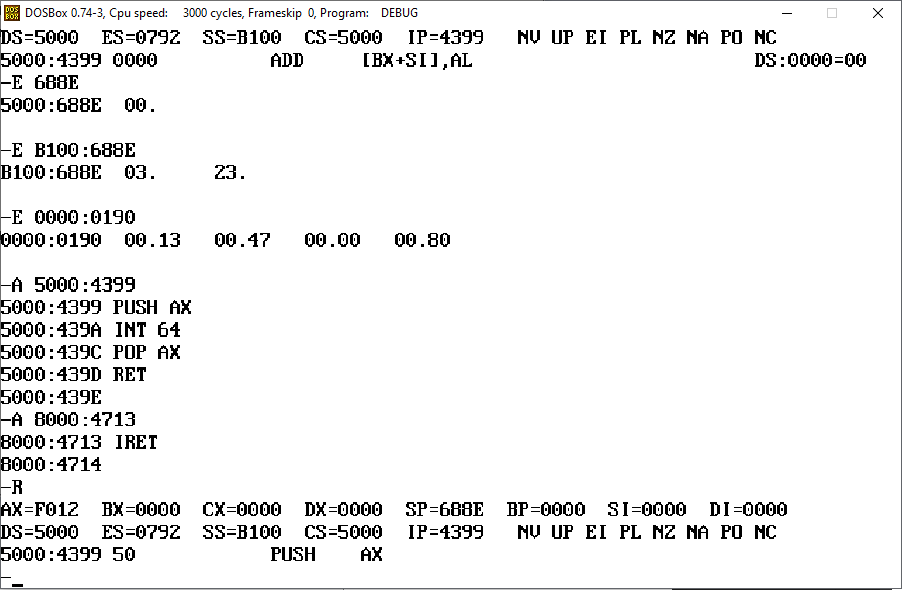
*Imagen 7: Utilizo el comando E para cargar el vector de interrupción.*

****

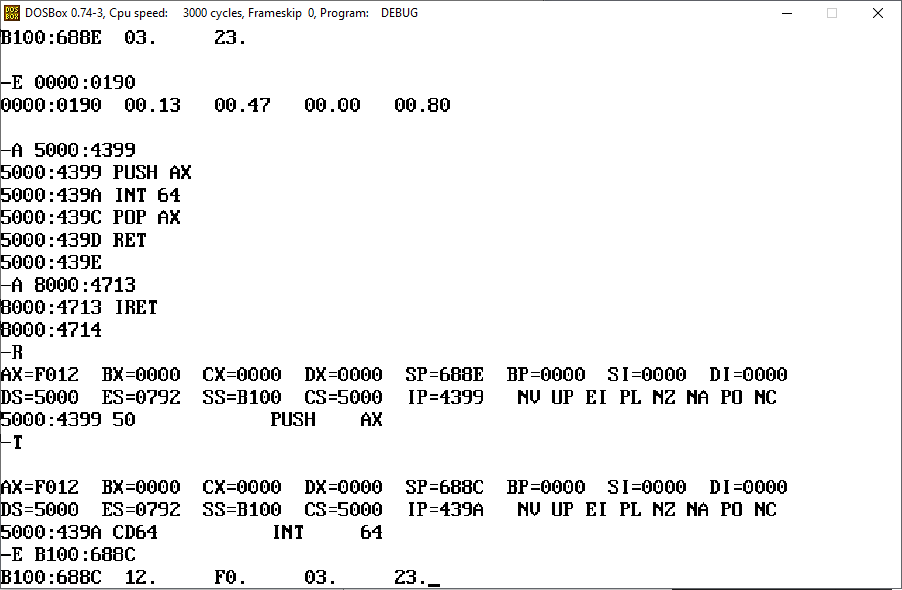
*Imagen 8: Utilizo el comando A para cargar las instrucciones de la subrutina en SS:IP (teniendo en cuenta que el IP de subrutina es 4399). Reviso y confirmo que coinciden con los valores de los puntos realizados en el parcial.*

****

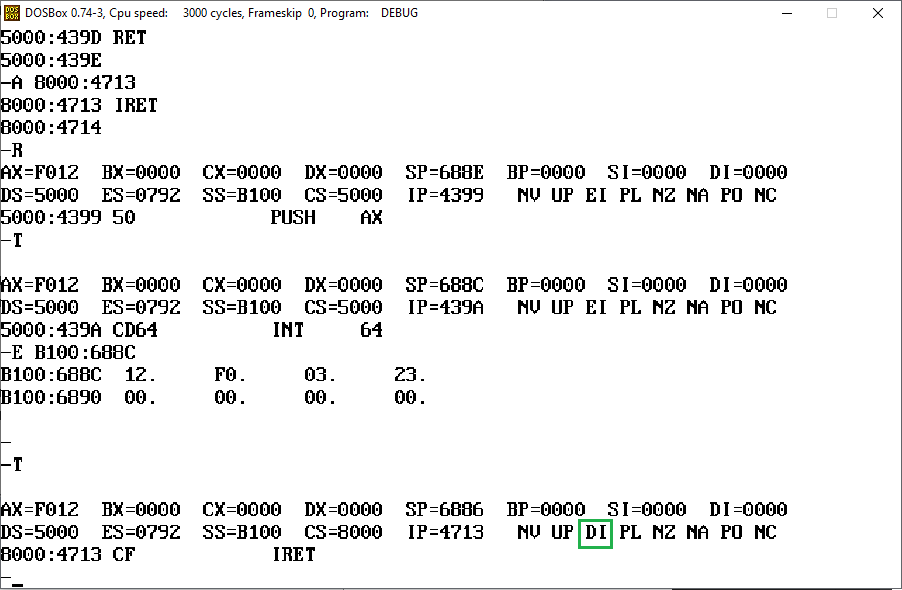
*Imagen 9: Con el comando A, cargo IRET en CS:IP (utilizando los datos proporcionados por la cátedra para el vector).*

****

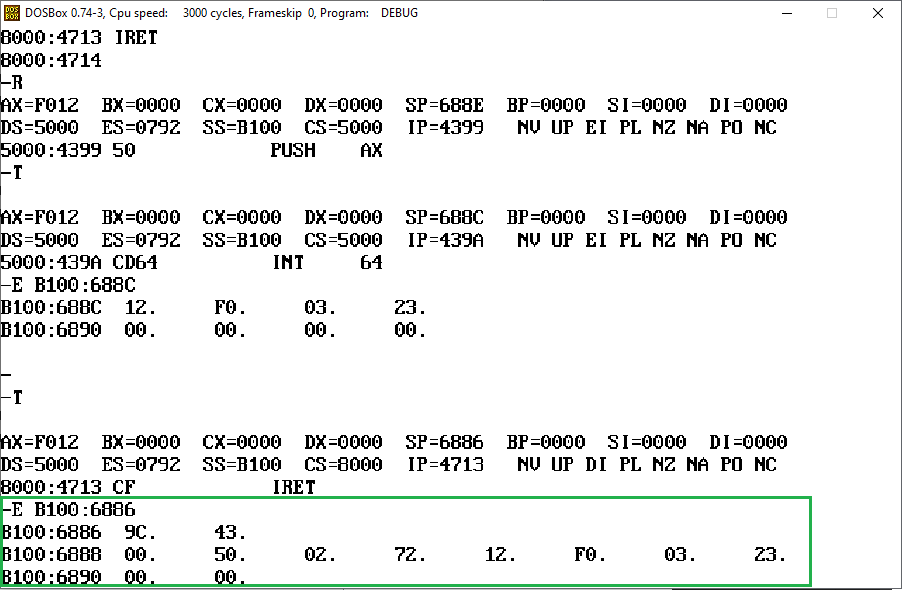
*Imagen 10: Con el comando R, reviso que la siguiente instrucción se encuentre ok (y sea PUSH AX y el siguiente IP sea 4399, además de que los registros estén bien cargados).*

****

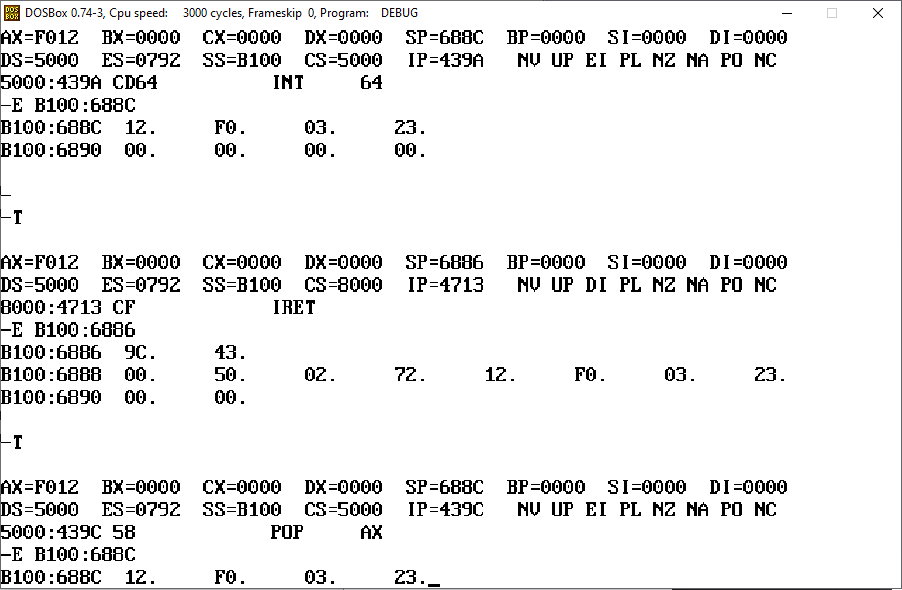
*Imagen 11: Con el comando T, ejecuto PUSH AX y con el E, controlo la pila (coincide con lo desarrollado durante el parcial)*

****

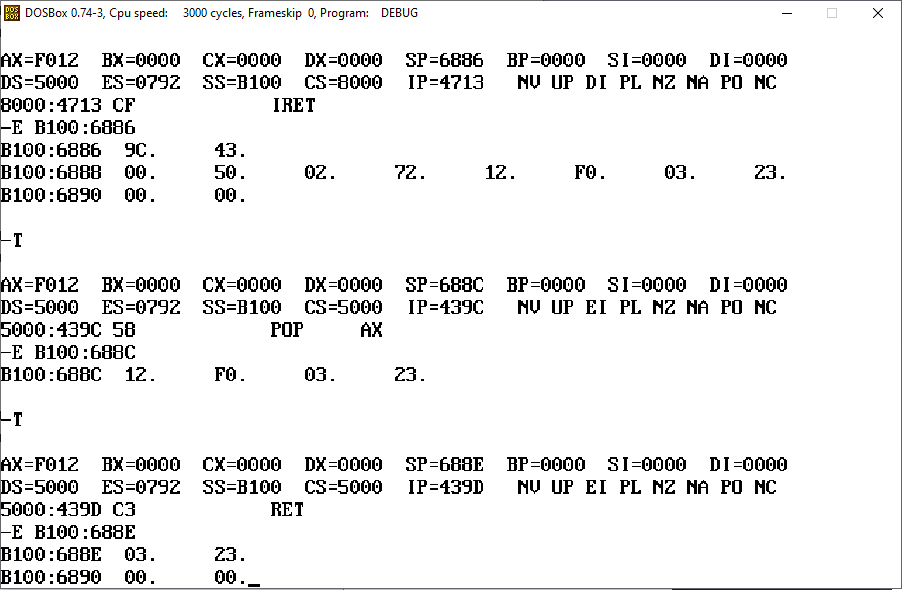
*Imagen 12: Nuevamente con el comando T, ejecuto la siguiente instrucción. En esta ejecución controlo valores; registro el cambio del flag I, ya que aparece DI entre los flags por la instrucción INT 64.*

****

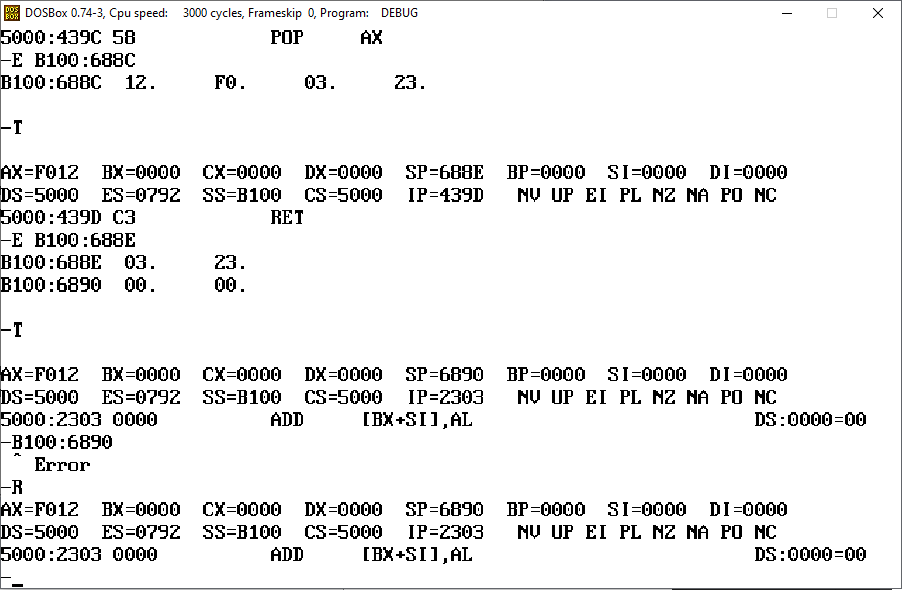
*Imagen 13: Con el comando E, compruebo como queda la pila luego de dicha instrucción y confirmo los resultados obtenidos en el análisis.*

****

*Imagen 14: vuelvo a ejecutar la siguiente instrucción con el comando T y compruebo que la pila vuelve, tal cual se había detallado, al paso anterior a ejecutar la instrucción INT. Compruebo que el flag I vuelve a su anterior estado.*

****

*Imagen 15: Ejecuto la siguiente instrucción y compruebo como se redujo la pila y confirmo valores obtenidos.*

****

**Imagen 16: Ejecuto la última instrucción y doy por finalizado el ejercicio.**

**PUNTO 4:**

Puesto que la subrutina empezó con PUSH AX e **INT 64**, **indicar:**

**4.a** La subrutina para funcionar correctamente debe terminar con las siguientes instrucciones:

POP AX

RET

**4.b** Qué movimientos ocurren cuando se ejecuta cada una, y cómo queda la pila luego de cada ejecución.

POP Ax

• Ax 🡨 [SP] SP=688C AX=F012

• SP 🡨 SP+2 SP=688E (688C+2)

RET

• IP 🡨 [SP] IP=688E

• SP 🡨 SP+2 SP=6890 🡨 **Cima de la pila**