P0B

David Teofilo Garitagoitia Romero

SISTEMAS BASADOS EN MICROPROCESADORES P0B

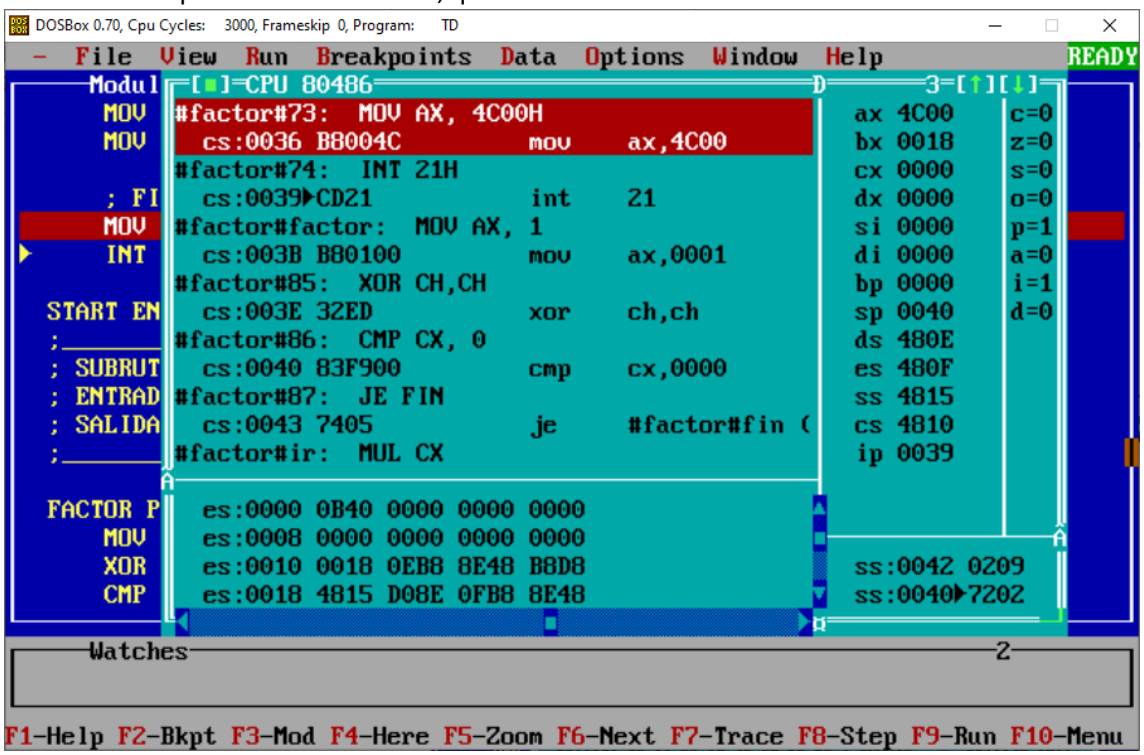
Contenido

EJERCICIO 1

1. 4! x 5! =

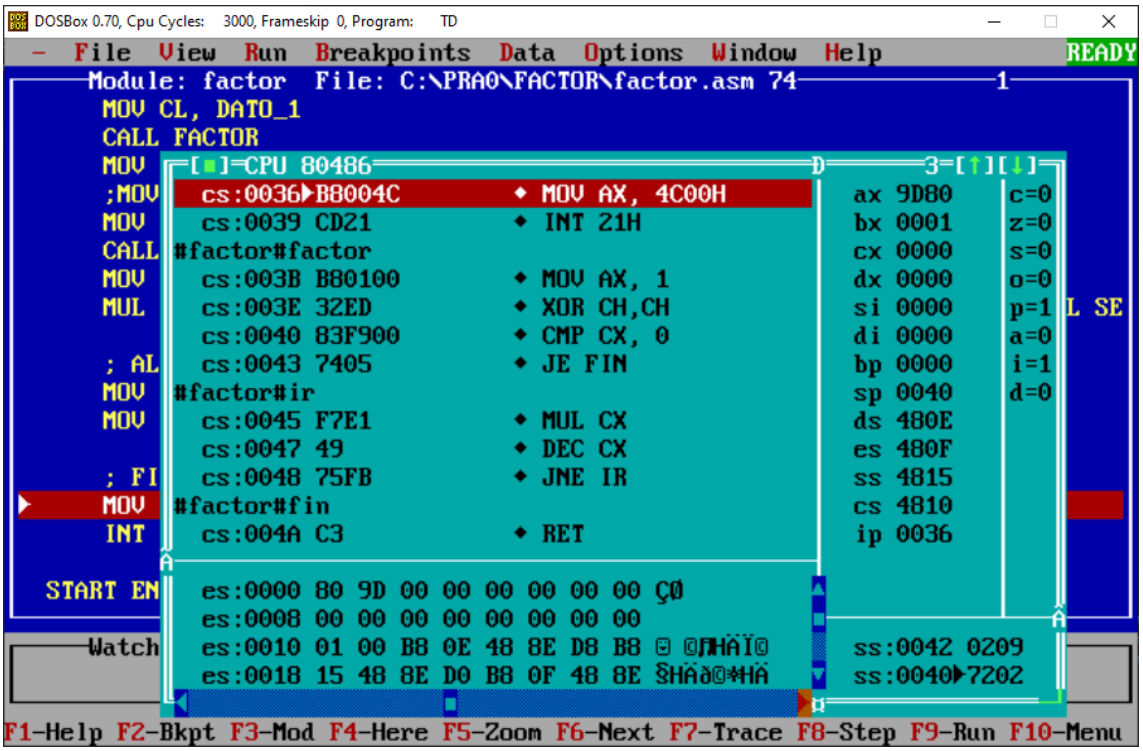
Para el ejercicio 1 basta con modificar los valores de DATO_1 y DATO_2 en data segment,

Para el primero muestra como resultado 0B40, convirtiendo ese valor de hexadecimal a decimal corresponde con 2.88*10^3, que es el valor de 4!*5!



2. 8! =

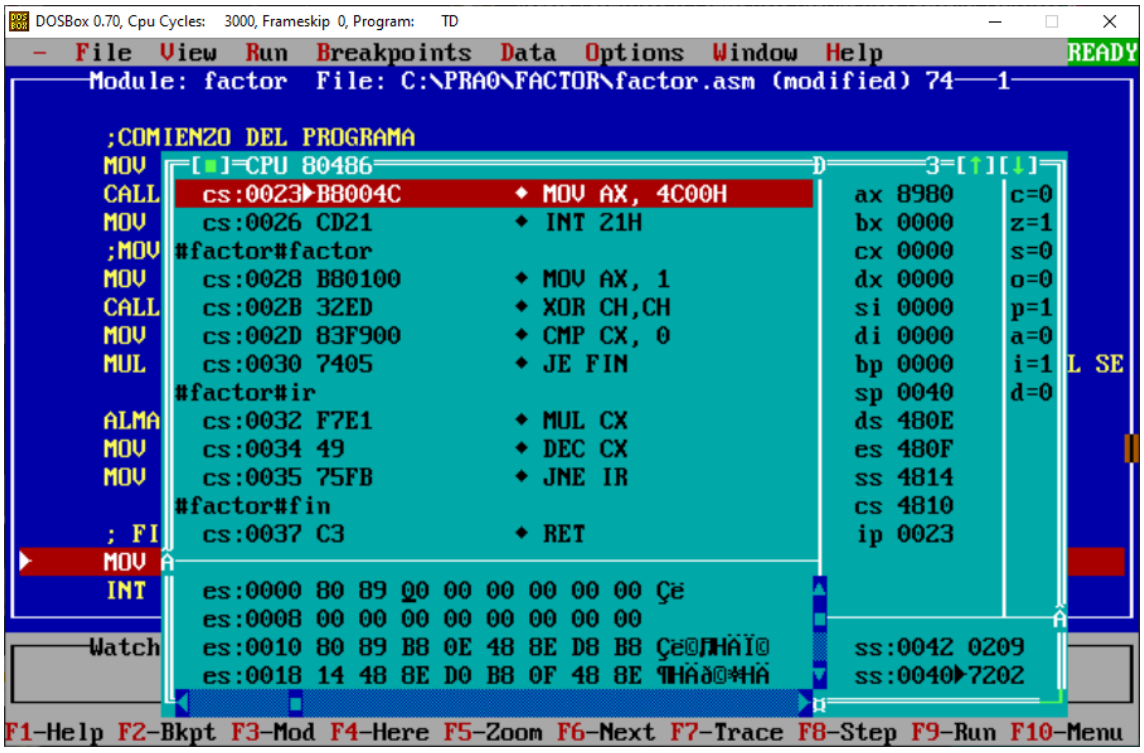
Para este se puede, tanto cambiar el DATO_2 por un 1, como modificar el programa para que únicamente haga el factorial de 1 dato, como podemos ver el resultado es 9D80 por lo que es correcto



factor.asm

```
49    MOV SS, AX
50
51    MOV AX, EXTRA
52    MOV ES, AX
53
54    ; CARGA EL PUNTERO DE PILA CON EL VALOR MAS ALTO
55    MOV SP, 64
56
57    ; FIN DE LAS INICIALIZACIONES
58
59    ;COMIENZO DEL PROGRAMA
60    MOV CL, DATO_1
61    CALL FACTOR
62    MOV FACT_DATO_1, AX       ; ALMACENA EL RESULTADO
63    MOV RESULT, AX
64    ;MOV CL, DATO_2
65    ;CALL FACTOR
66    ;MOV BX, FACT_DATO_1
67    ;MUL BX                   ; EN AX ESTA EL RESULTADO DEL FACTORIAL DEL SEGUNDO
68
69    ; ALMACENA EL RESULTADO
70    ;MOV RESULT, AX
71    ;MOV RESULT+2, DX
72
73    ; FIN DEL PROGRAMA
74    MOV AX, 4C00H
75    INT 21H
```

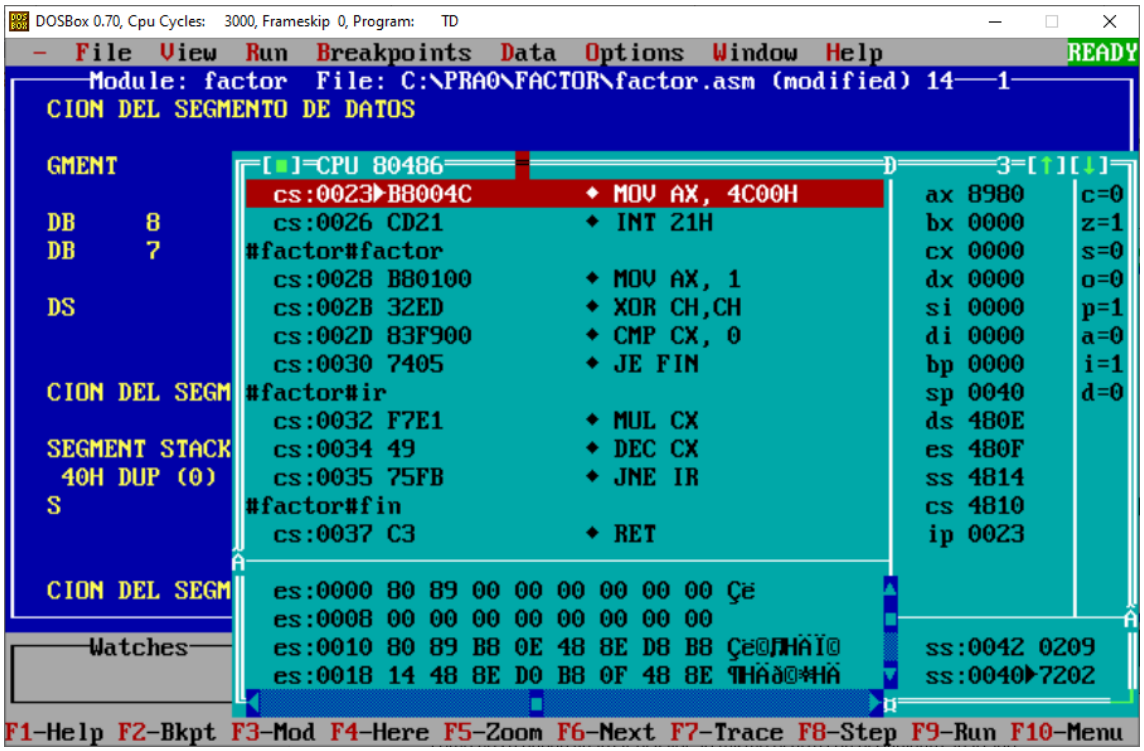
3. 9!=



El resultado no coincide porque no entra el valor en la variable ya que como máximo llega a FFFF, por lo que ocurre overflow como se puede ver en la bandera O que se activa dentro de la función factor, lo mismo en el apartado 4 de este mismo ejercicio

4. 8! x 7! =

Ocurre lo mismo que en el caso anterior



En cada uno de los casos el valor de FACT_DATO_1 es el valor del factorial del dato 1

```
;COMIENZO DEL PROGRAMA
MOV CL, DATO_1
CALL FACTOR
MOV FACT_DATO_1, AX      ; ALMACENA EL RESULTADO
```

Ejercicio 2

Para que calcule la multiplicación y después haga el factorial, basta con eliminar una llamada a factor y hacer la multiplicación antes.

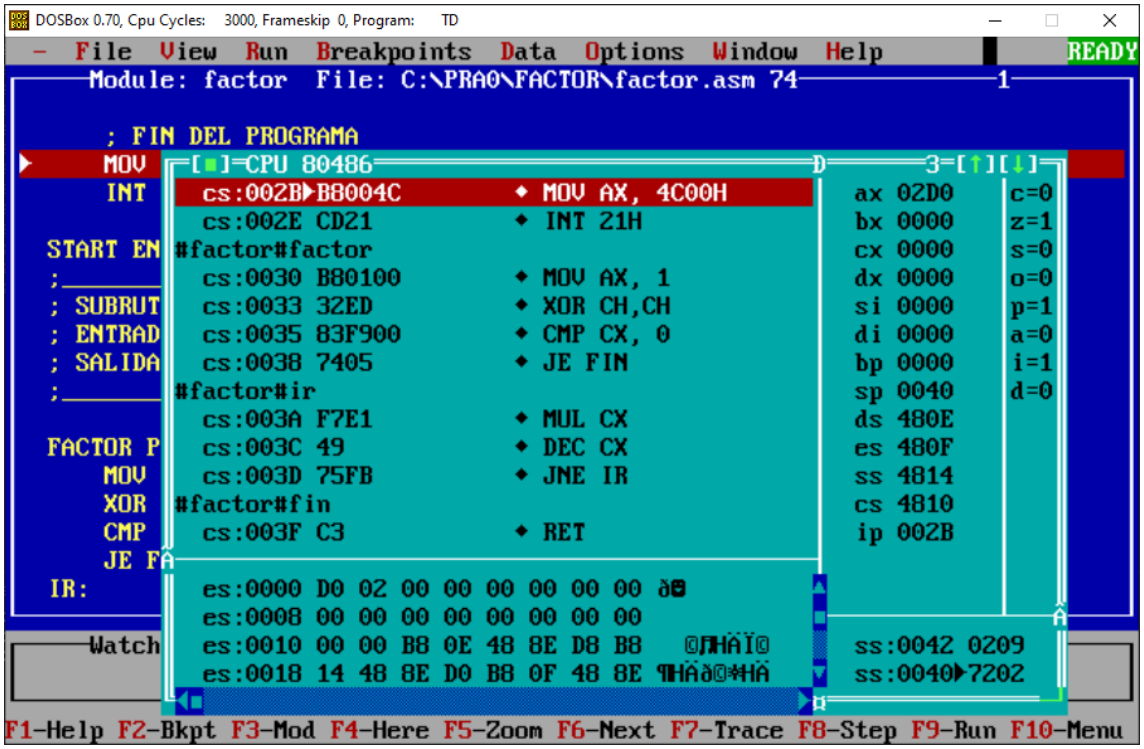
```
    ;COMIENZO DEL PROGRAMA
    MOV AL, DATO_1
    ;CALL FACTOR
    ;MOV FACT_DATO_1, AX       ; ALMACENA EL RESULTADO
    MOV CL, DATO_2 ;Tiene que ser en L porque es little endian
    ;CALL FACTOR
    ;MOV BX, FACT_DATO_1
    MUL CX                    ; EN AX ESTA EL RESULTADO DEL FACTORIAL DEL SEGUNDO
    MOV CX, AX ;Se pasa el valor de la multiplicación al registro C que es el que se usa en factor
    ; ALMACENA EL RESULTADO
    CALL FACTOR
    MOV RESULT, AX  ;Se pasa AX que contiene el valor del resultado a RESULT
    MOV RESULT+2, DX

    ; FIN DEL PROGRAMA
    MOV AX, 4C00H
    INT 21H

START ENDP
```

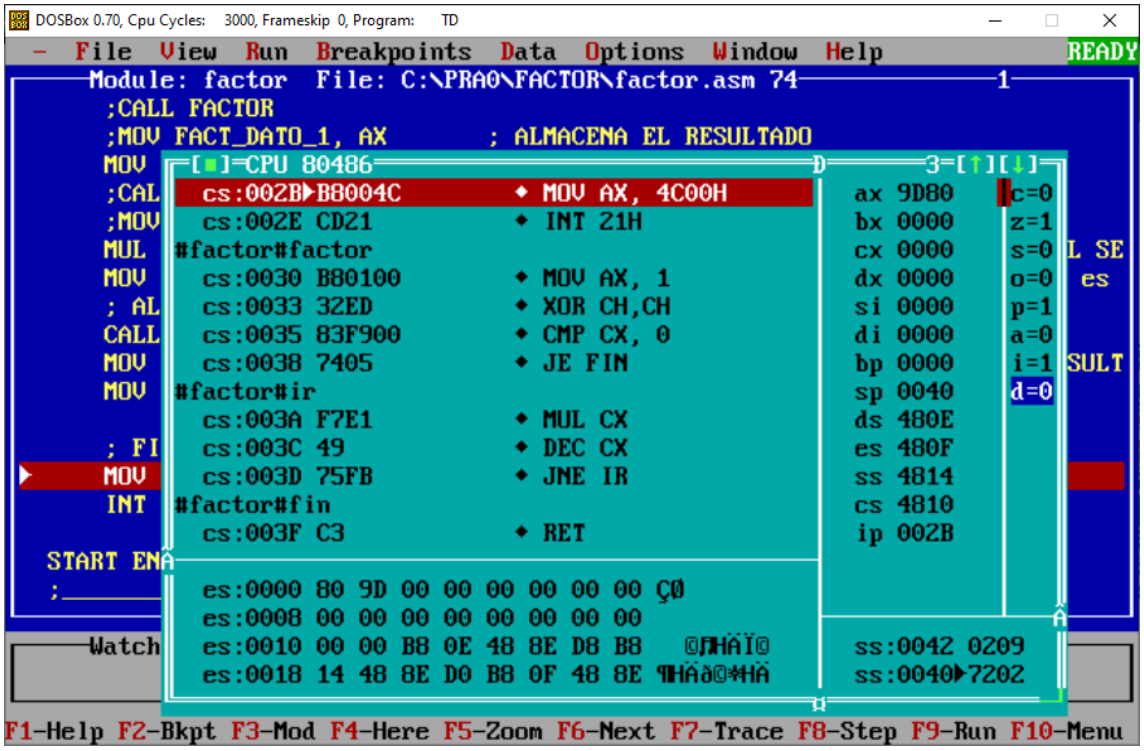
1. (2 x 3) ! =

Se muestra como resultado 02D0 que traducido de hexadecimal en decimal corresponde con el valor 720=5!=(2x3)!



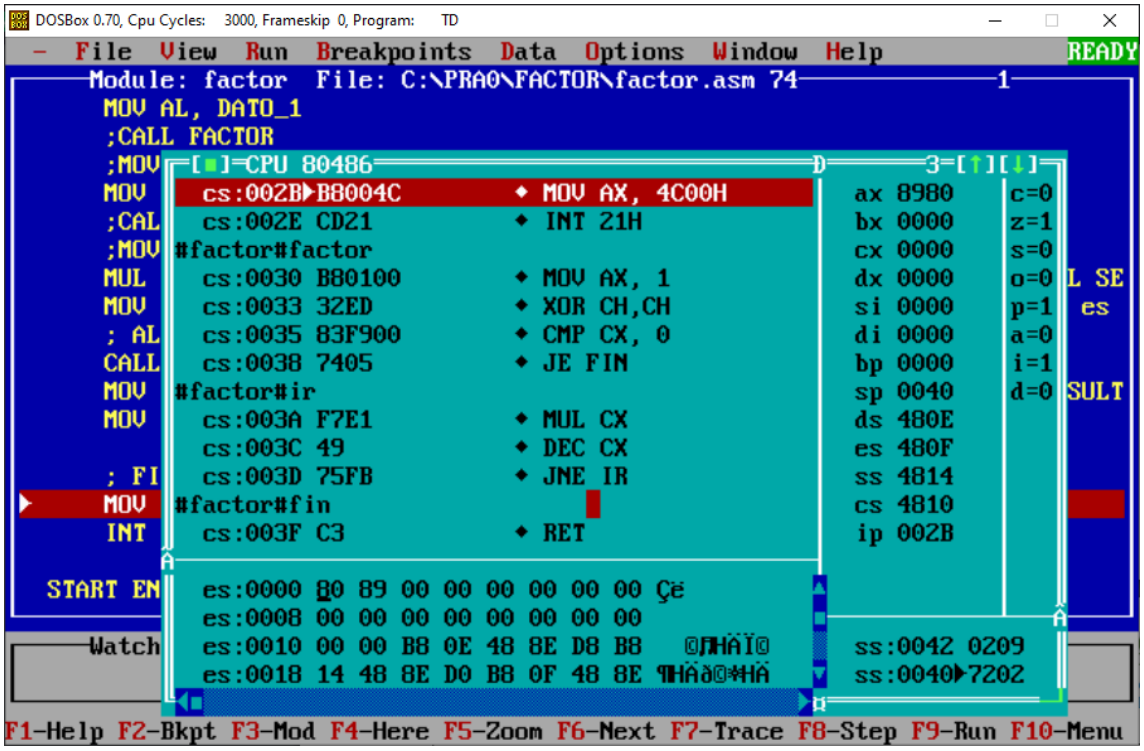
2. (2 x 4) ! =

El resultado de la operación es 4.032*10^4, que en hexadecimal corresponde con 9D80



3. (3 x 3) !=

Overflow, nuevamente volvemos al caso de 9! cuyo resultado no entra en FFFF



4. (5 x 2) ! =

Nuevamente el resultado es incorrecto, ya que se produce overflow