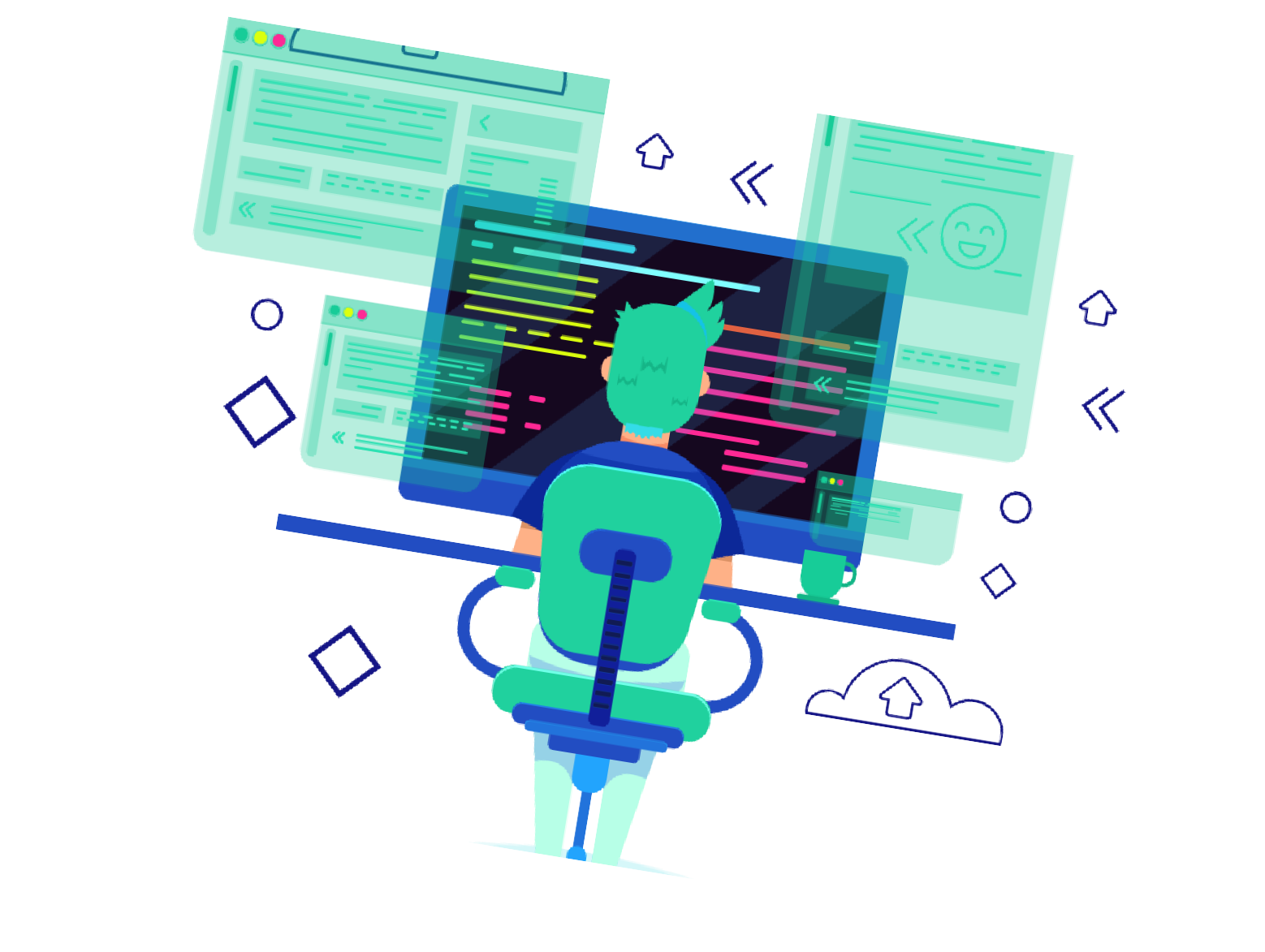
Actividad 3



**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**

**e**

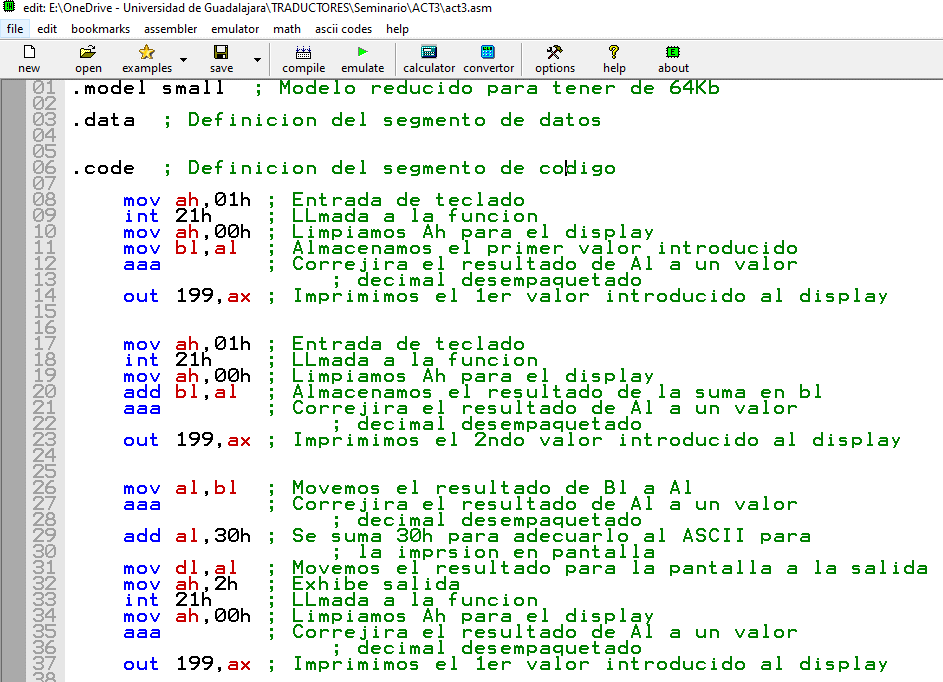
# Desarrollo

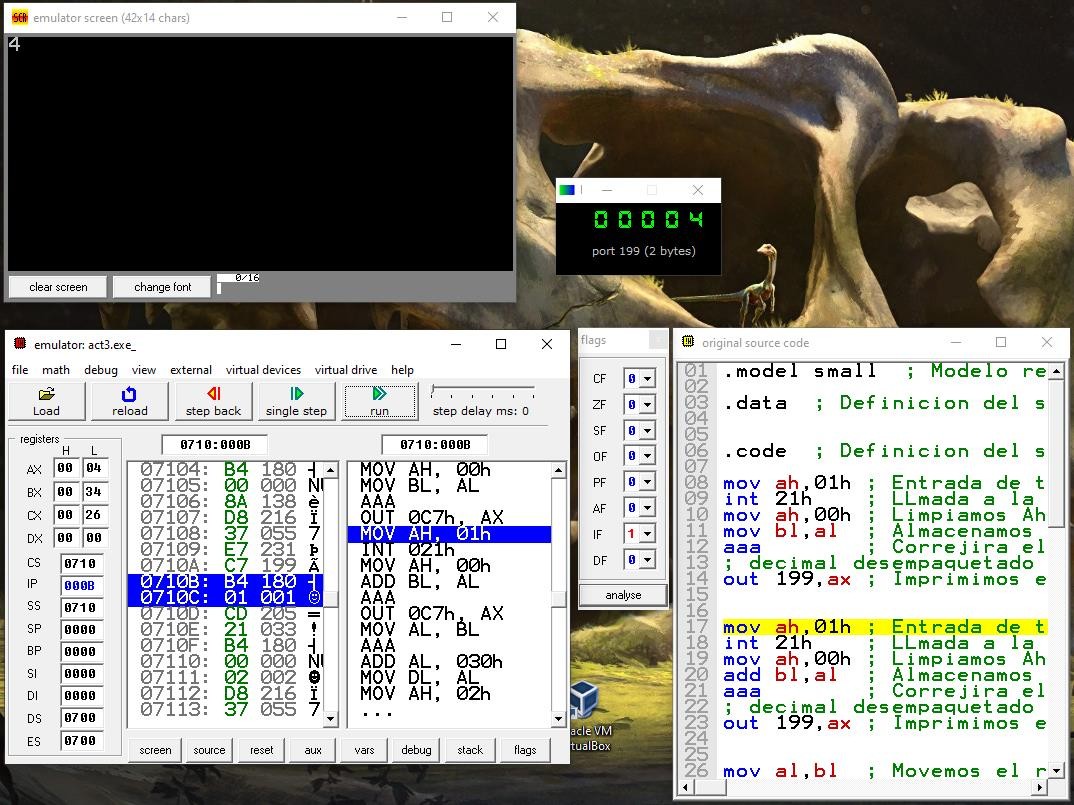
Implementar las instrucciones ADD, ADC, SUB, SBB, MUL y DIV.

Imprimir el resultado en pantalla (monitor) de la suma y además imprimirlo con el display de emu8086.

## ADD

Se realizo la operación ADD, en el que el usuario puede introducir sus propios valores para su respectivo almacenamiento, en el que se usó algunas estrategias para administrar los datos para obtener la suma final deseado, así como de la función de AAA para convertirlo en un número fijo y se le sumara 30 para adecuarlo al símbolo de números de la tabla ASCII. También se utilizó las llamadas de función para introducir valores del teclado y las respectivas impresiones en el display como en la consola.

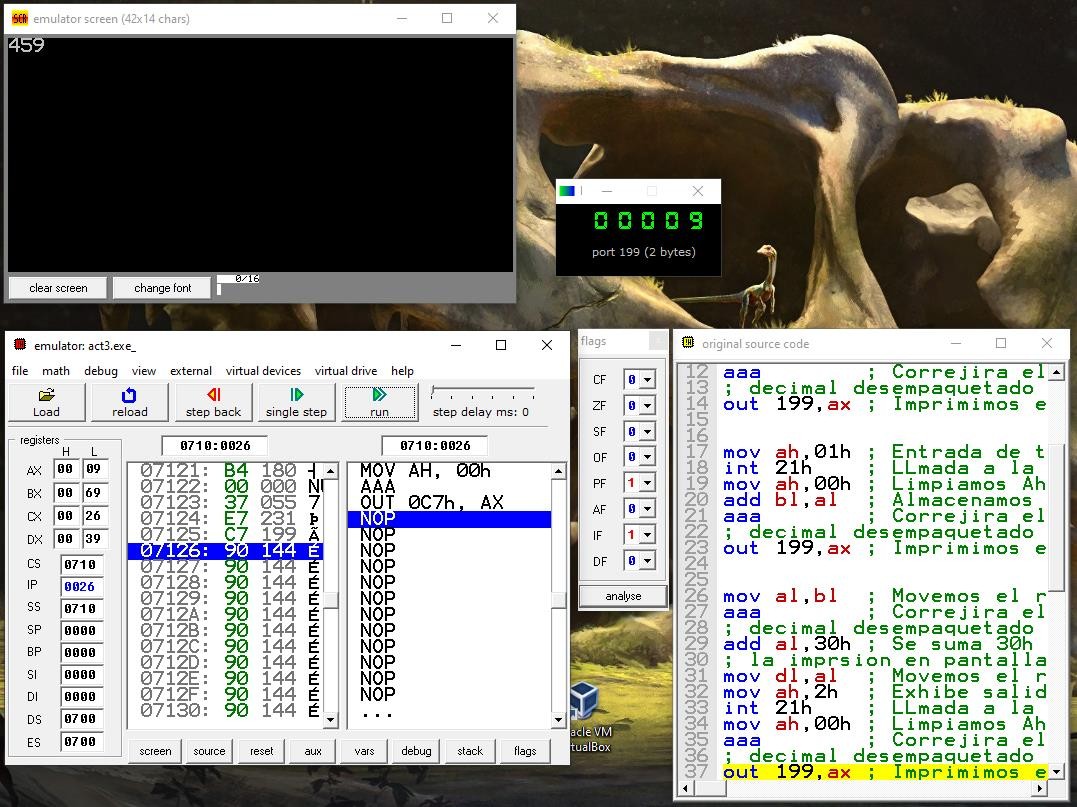




Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (BL).



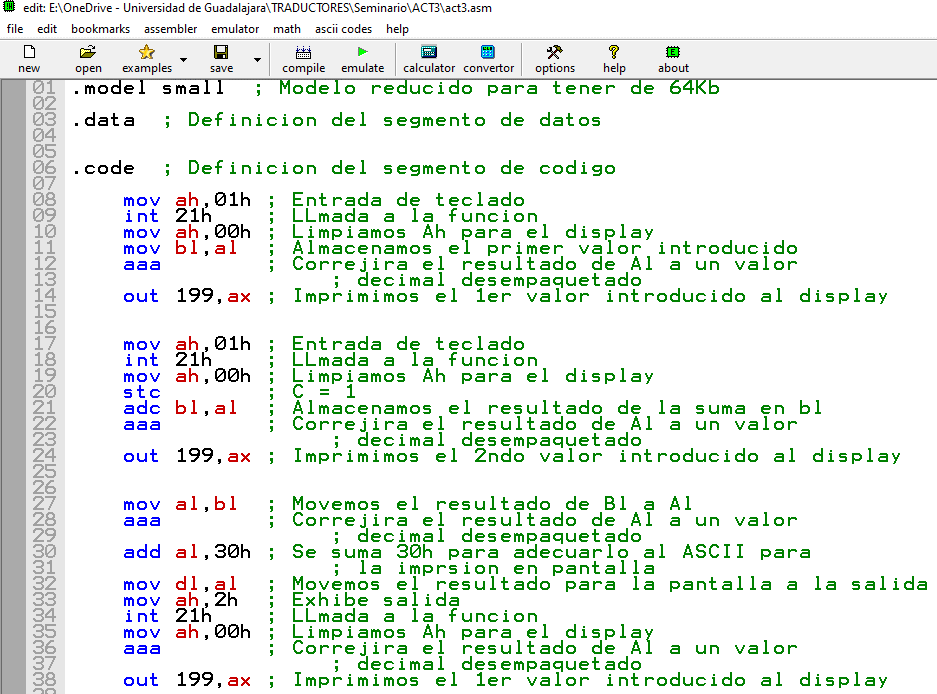
Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior.

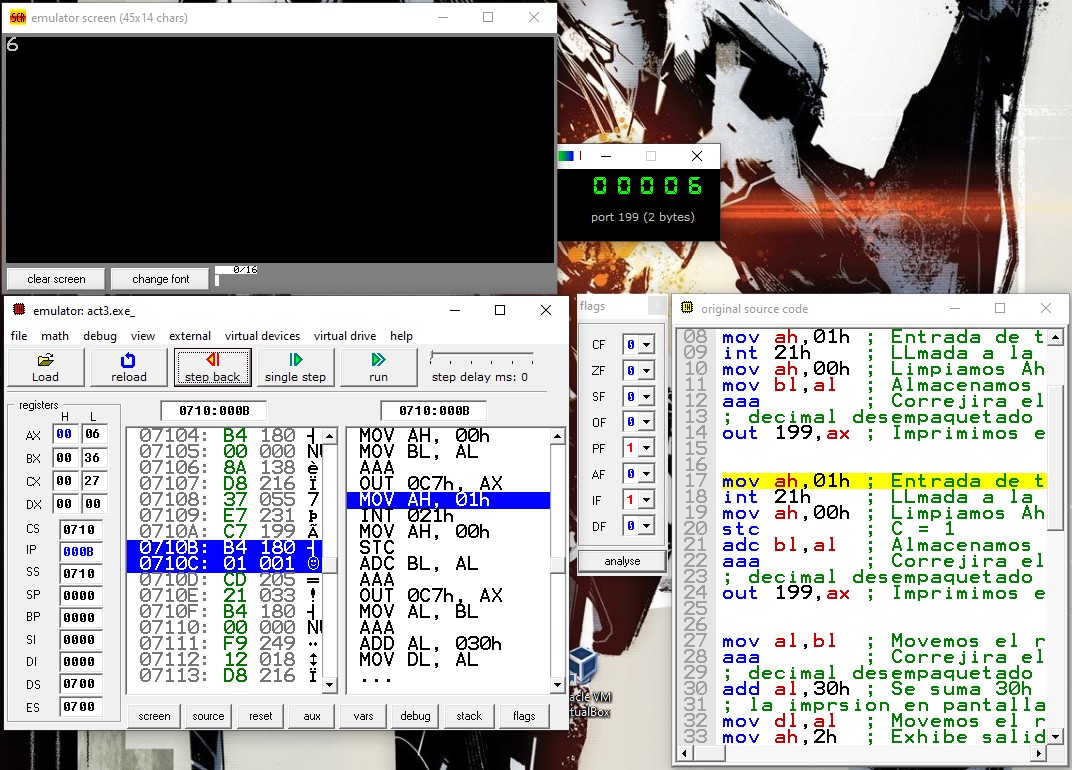


Para el resultado final de la operación se tuvo que hacer movimientos hacia (AL) para su respectiva conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAA para la display y para la consola se usó la función anterior y se le sumo 30.

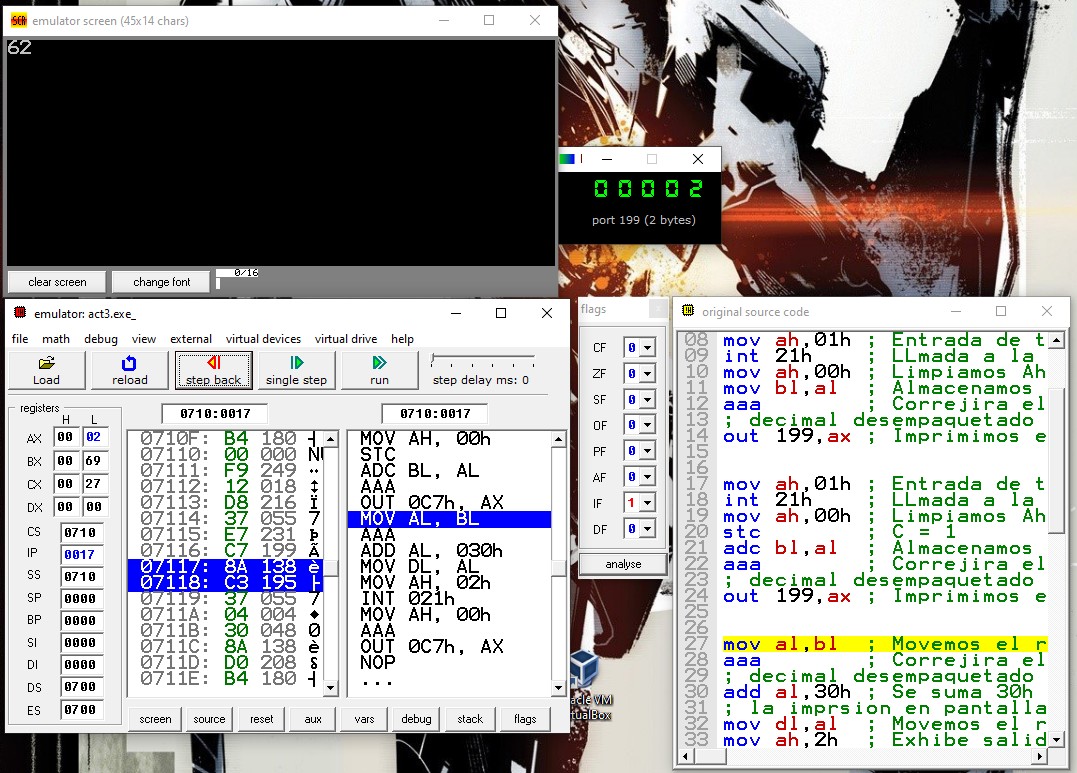
## ADC

Se realizo la operación ADC que es muy parecido al ADD, pero con un arrastre que le agregara a nuestra suma final según el estado de C, en el que el usuario puede introducir sus propios valores para su respectivo almacenamiento, en este caso se puso C en 1 para demostrar su arrastre, en el que se usó algunas estrategias para administrar los datos para obtener la suma con arrastre final deseado, así como de la función de AAA para convertirlo en un número fijo y se le sumara 30 para adecuarlo al símbolo de números de la tabla ASCII. También se utilizó las llamadas de función para introducir valores del teclado y las respectivas impresiones en el display como en la consola.

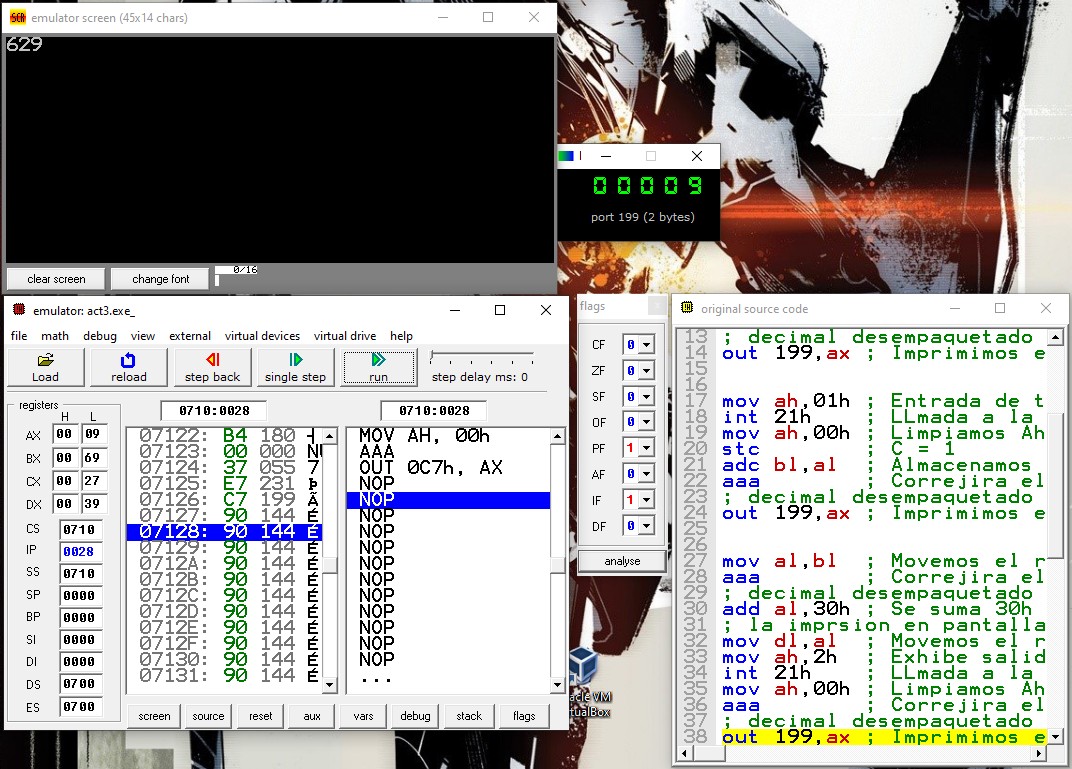




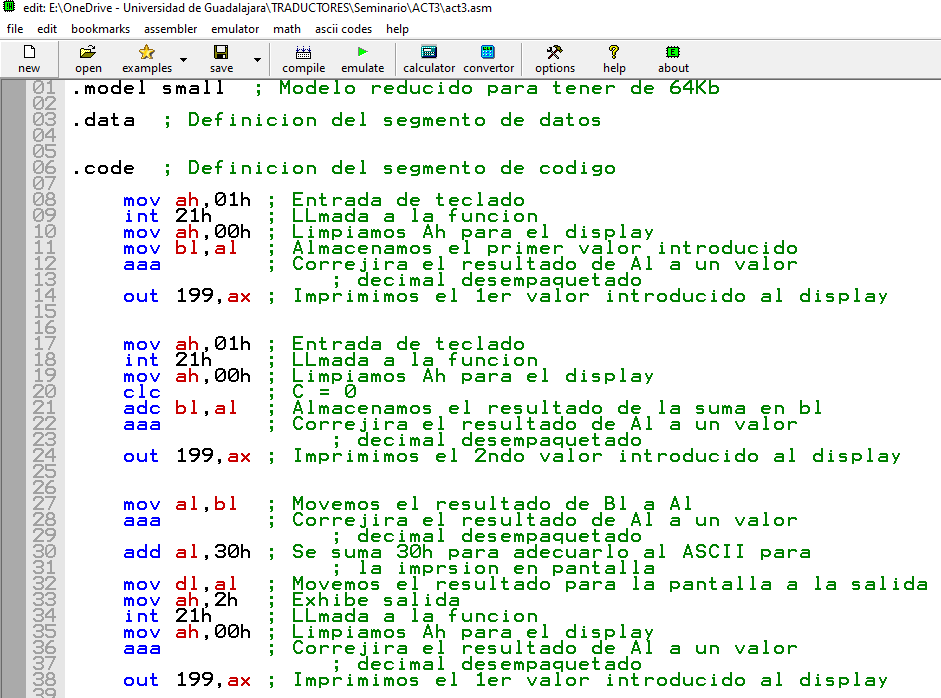
Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (BL).

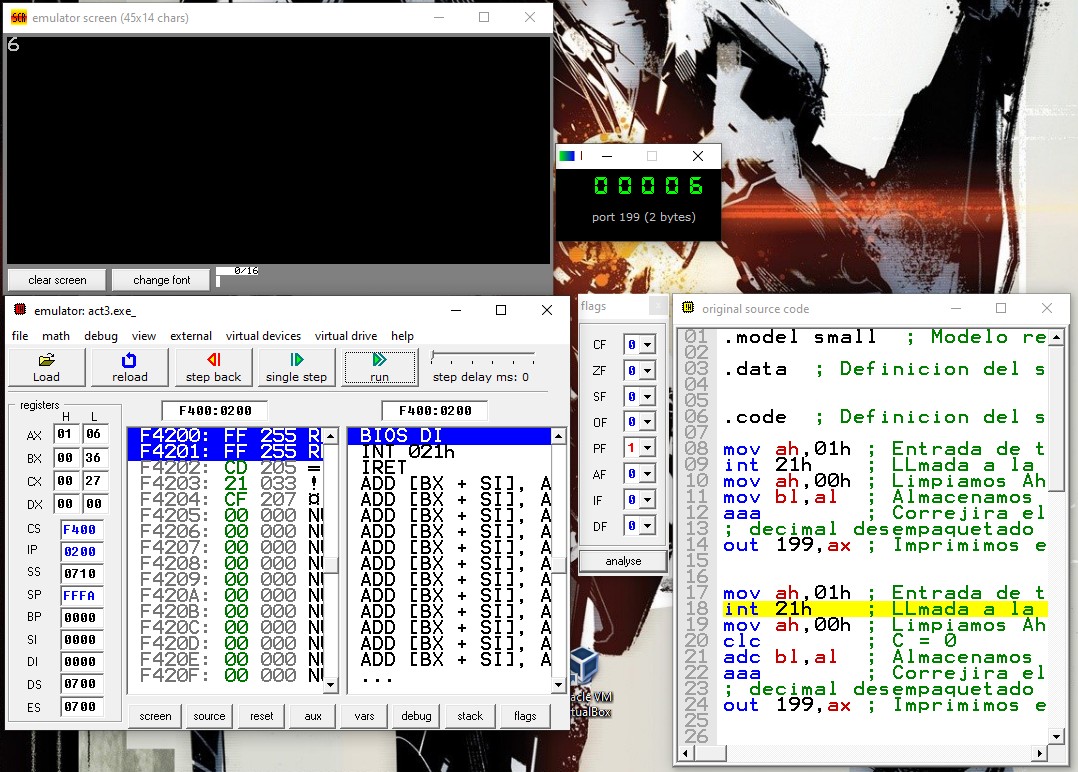


Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior.

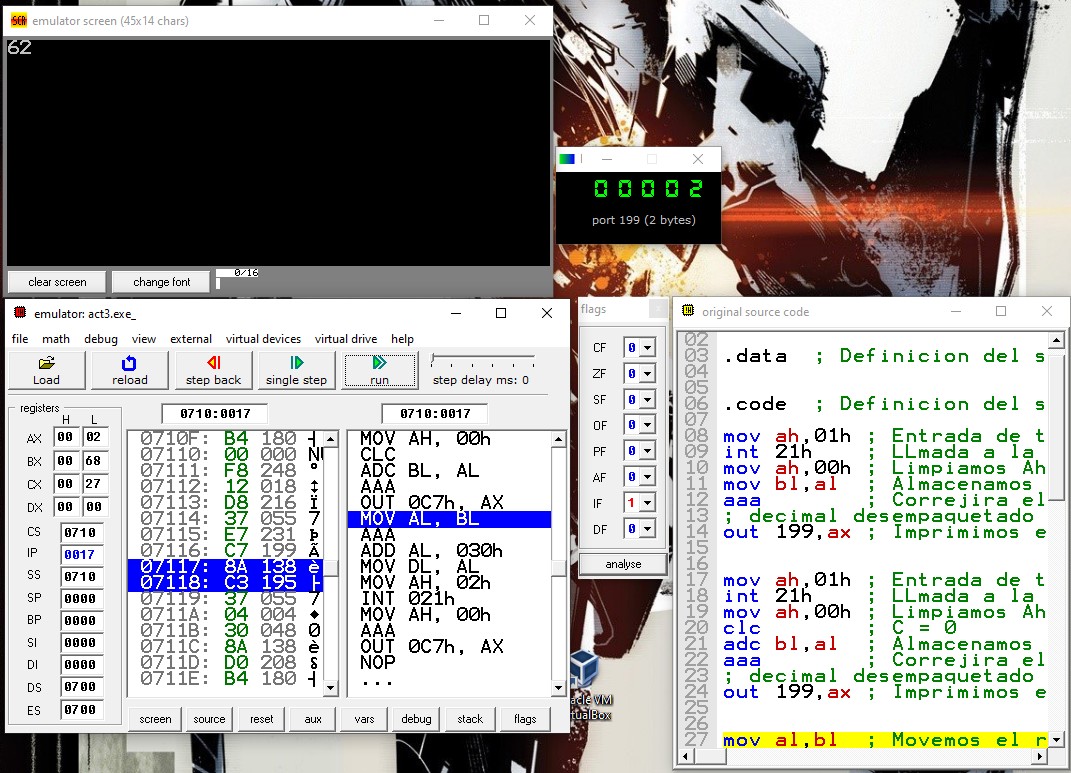


Para el resultado final de la operación se tuvo que hacer movimientos hacia (AL) para su respectiva conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAA para la display y para la consola se usó la función anterior y se le sumo 30.

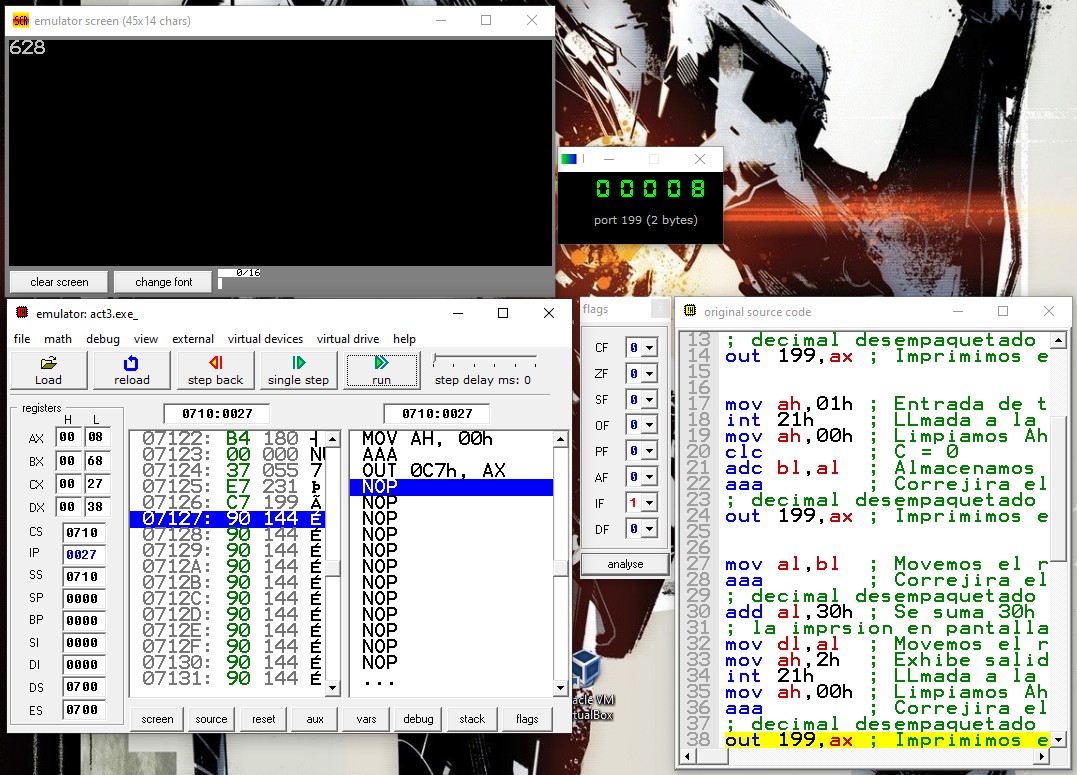
También se ejecutará la operación ADC que no tenga arrastre poniendo C en 0.



Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (BL).



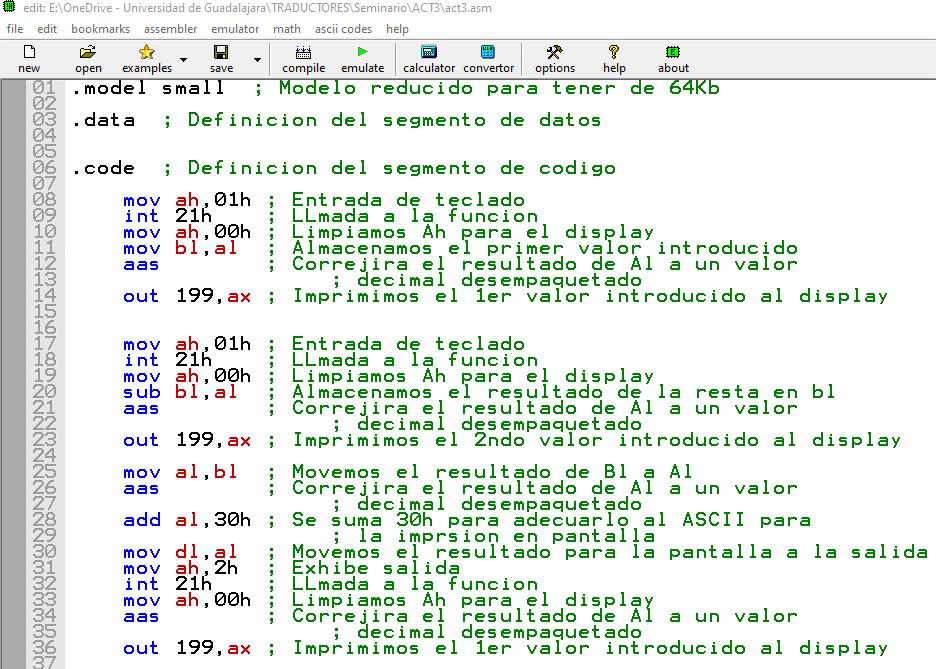
Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior.

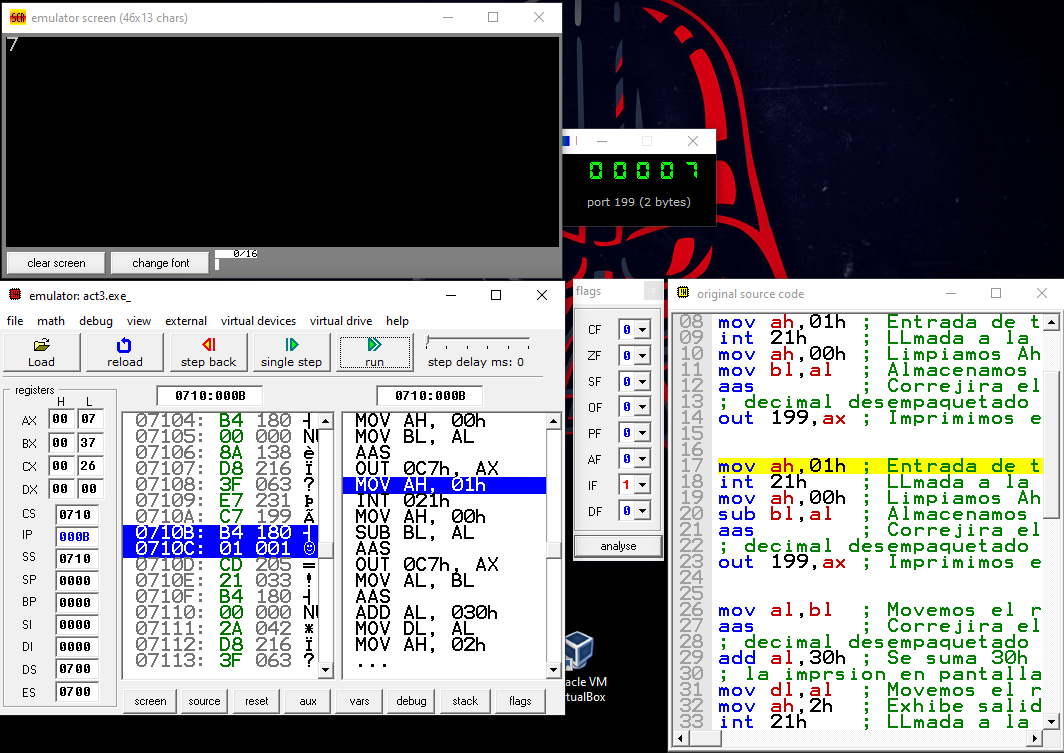


Para el resultado final de la operación se tuvo que hacer movimientos hacia (AL) para su respectiva conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAA para la display y para la consola se usó la función anterior y se le sumo 30.

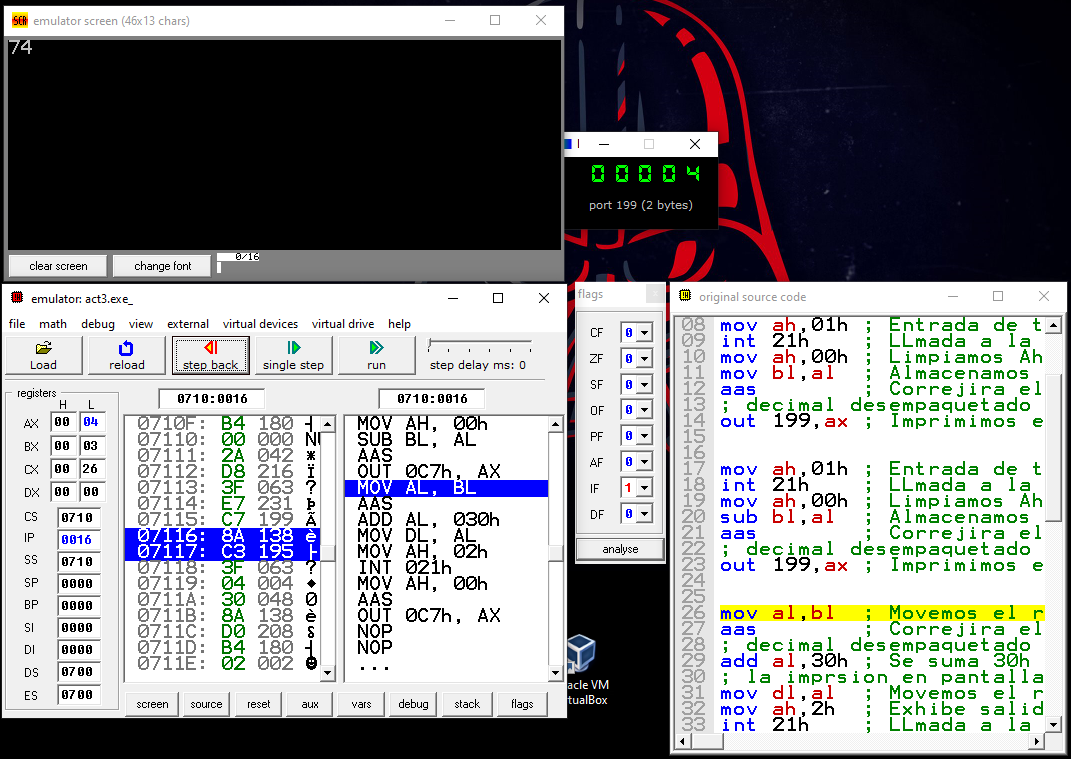
## SUB

Se realizo la operación SUB, en el que el usuario puede introducir sus propios valores para su respectivo almacenamiento, en el que se usó algunas estrategias para administrar los datos para obtener la resta final deseado, así como de la función de AAS para convertirlo en un número fijo y se le sumara 30 para adecuarlo al símbolo de números de la tabla ASCII. También se utilizó las llamadas de función para introducir valores del teclado y las respectivas impresiones en el display como en la consola.

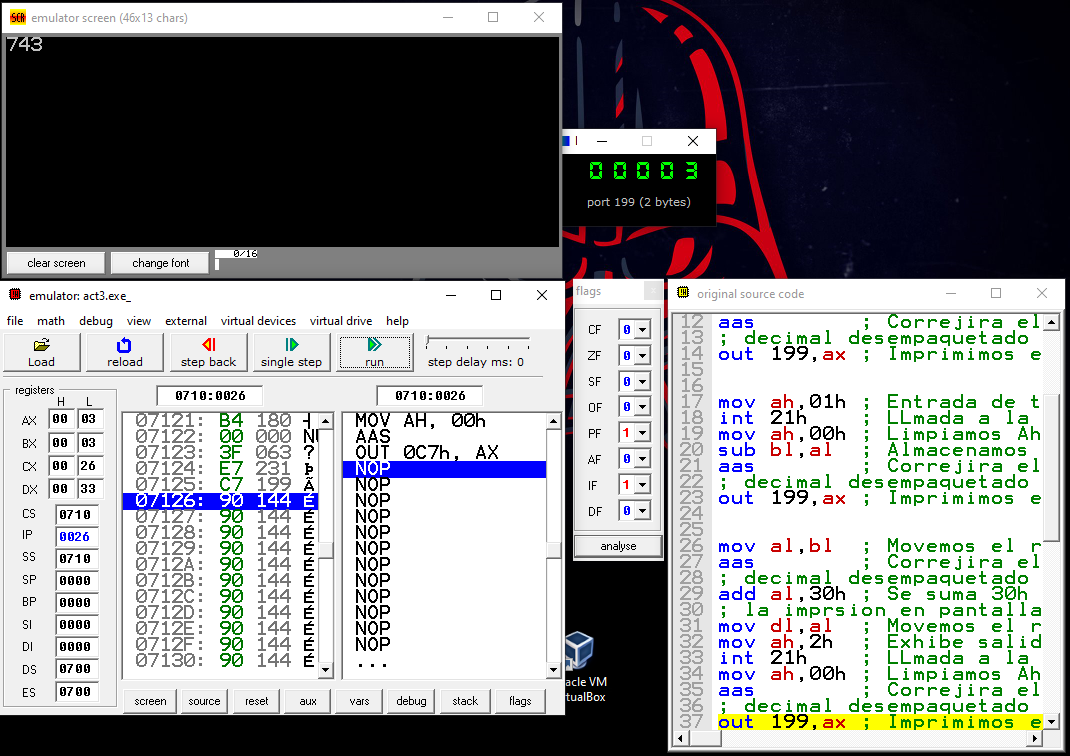




Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAS para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (BL).



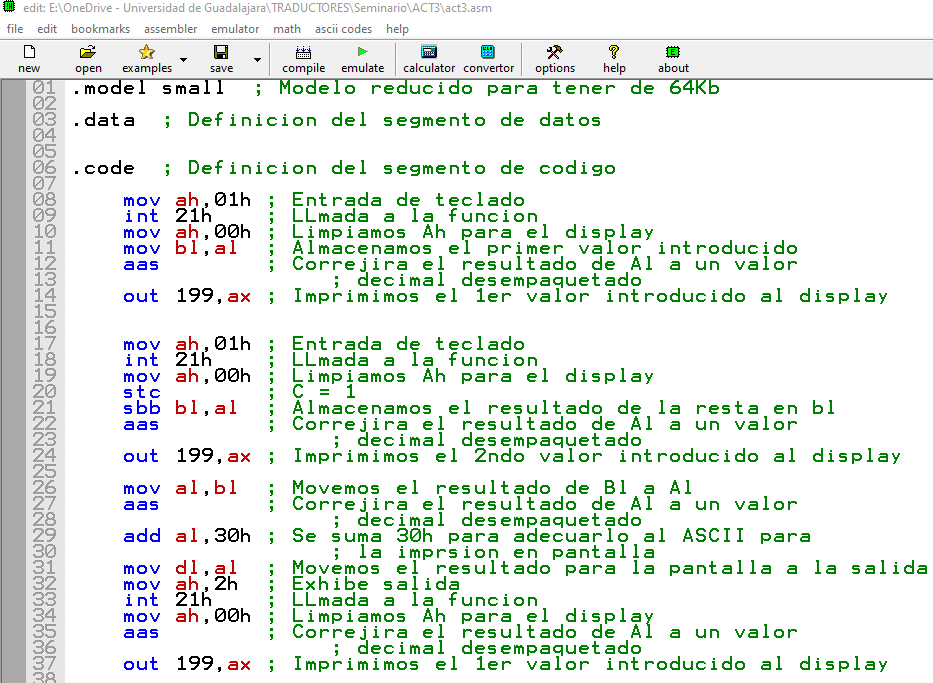
Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAS para la display y para la consola se usó la función anterior.

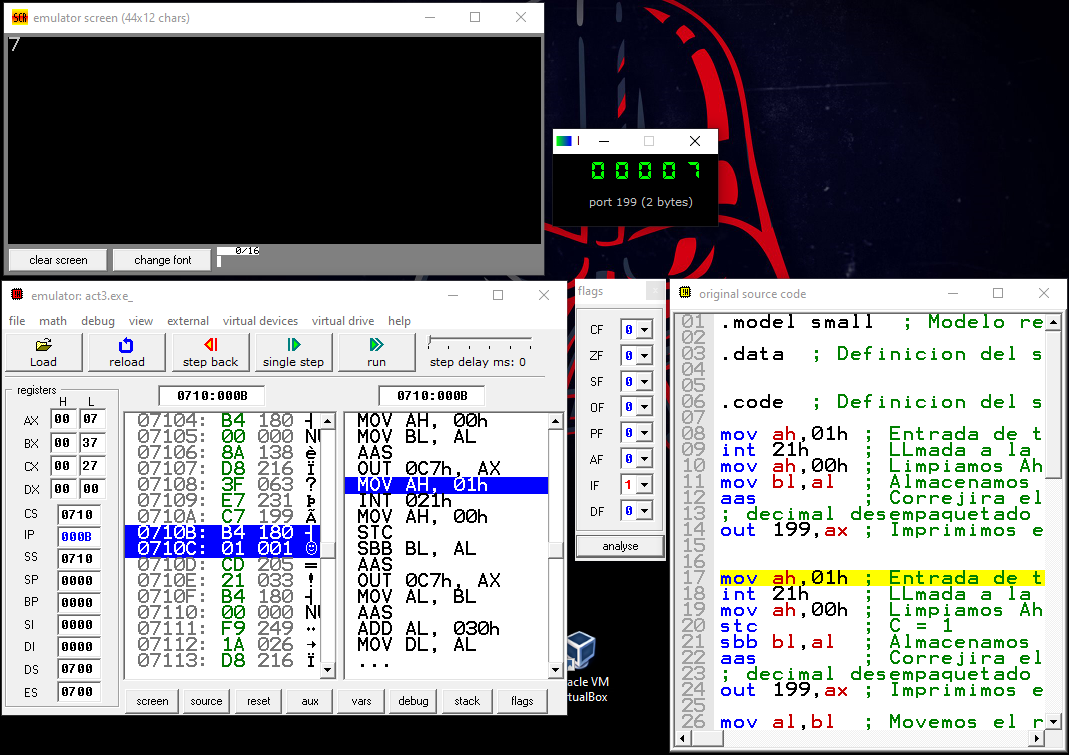


Para el resultado final de la operación se tuvo que hacer movimientos hacia (AL) para su respectiva conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAS para la display y para la consola se usó la función anterior y se le suma 30.

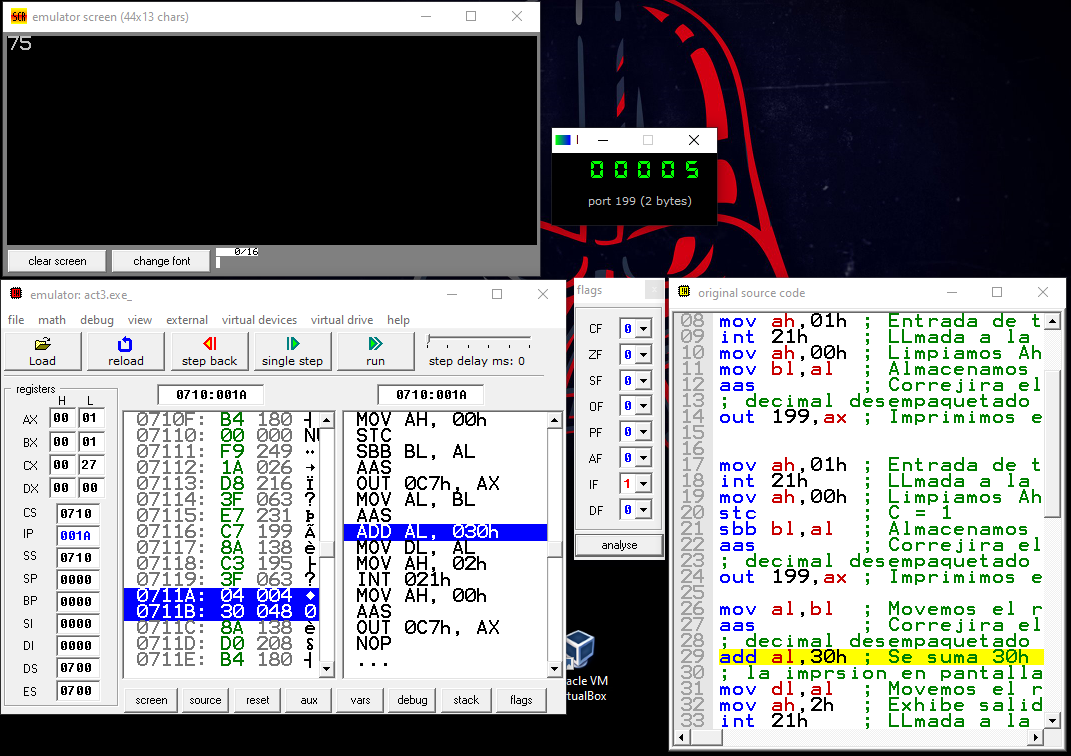
## SBB

Se realizo la operación SBB que es muy parecido al SUB, pero con un préstamo que le sumara a nuestro operando fuente a nuestra resta final según el estado de C, en el que el usuario puede introducir sus propios valores para su respectivo almacenamiento, en este caso se puso C en 1 para demostrar su préstamo, en el que se usó algunas estrategias para administrar los datos para obtener la resta con préstamo final deseado, así como de la función de AAS para convertirlo en un número fijo y se le sumara 30 para adecuarlo al símbolo de números de la tabla ASCII. También se utilizó las llamadas de función para introducir valores del teclado y las respectivas impresiones en el display como en la consola.

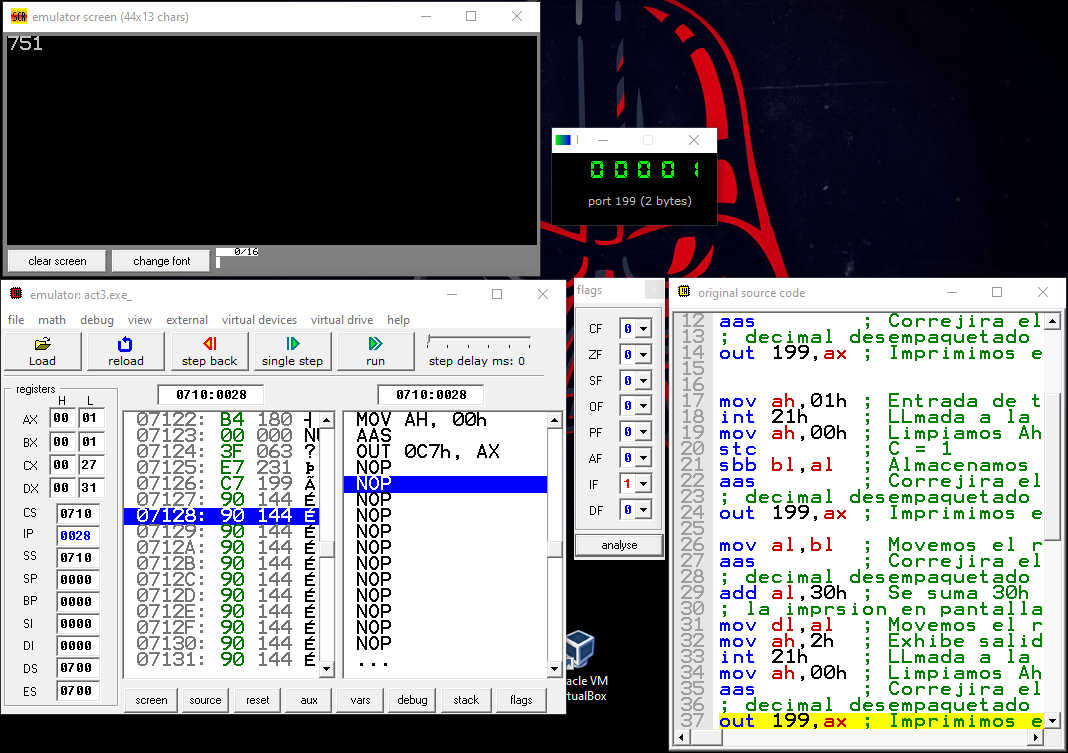




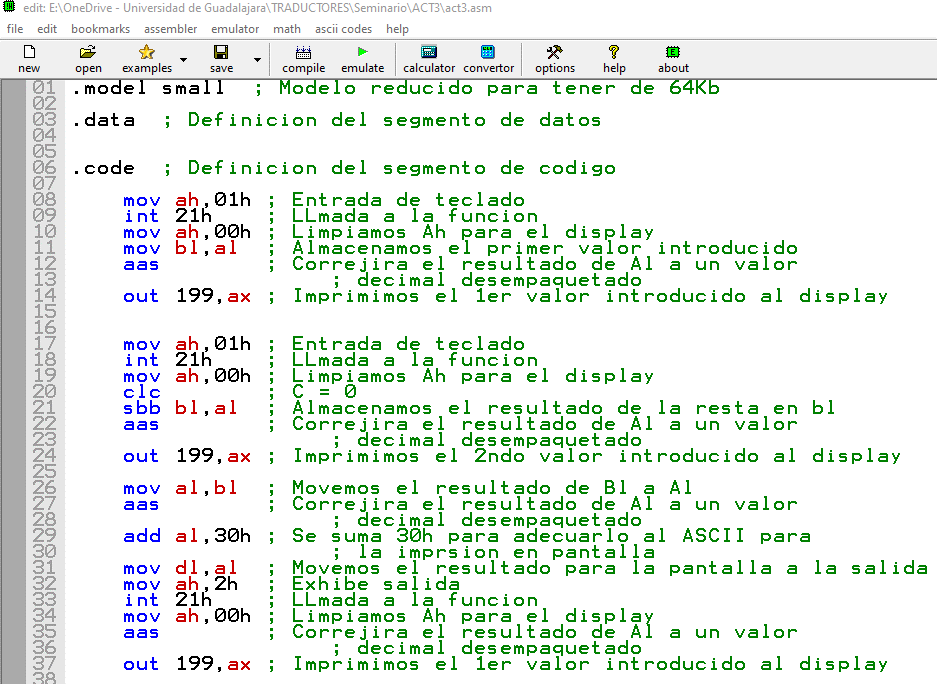
Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAS para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (BL).

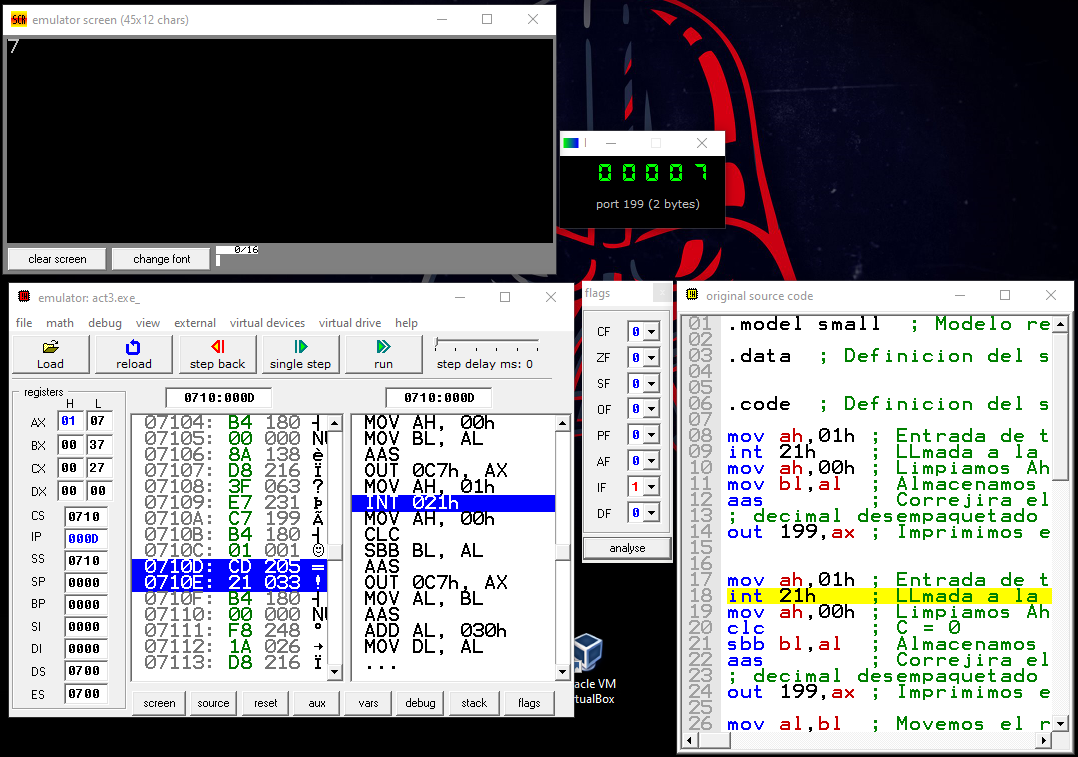


Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAS para la display y para la consola se usó la función anterior.

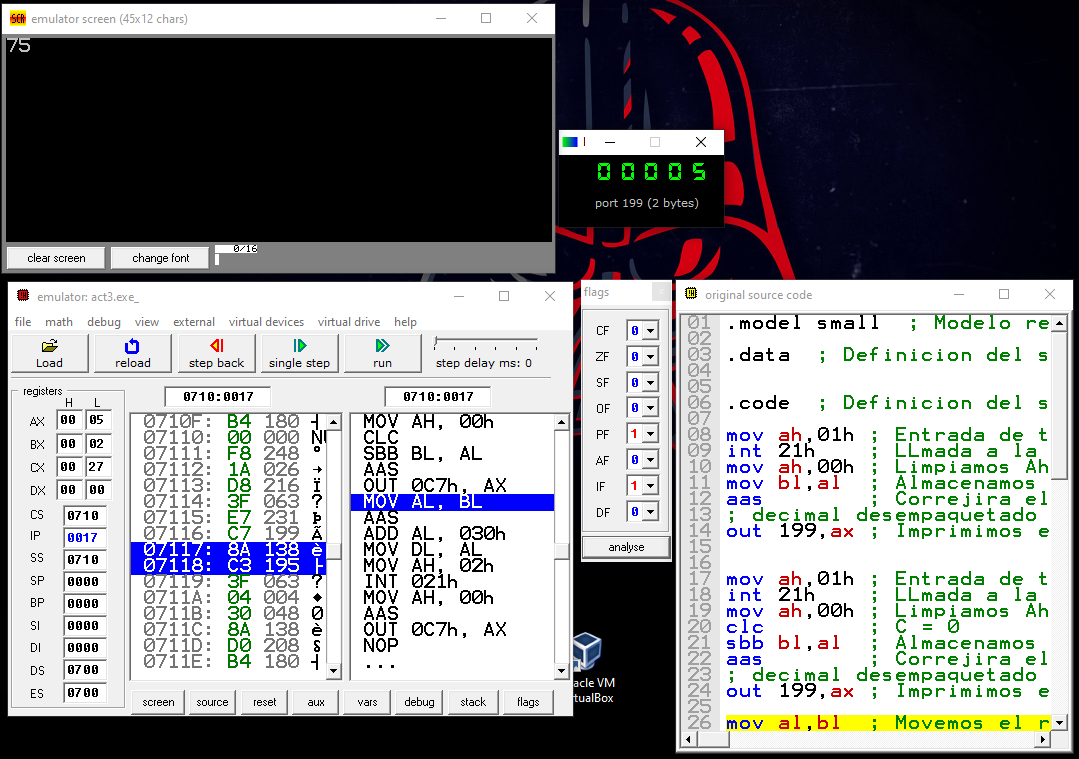


Para el resultado final de la operación se tuvo que hacer movimientos hacia (AL) para su respectiva conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAS para la display y para la consola se usó la función anterior y se le sumo 30.

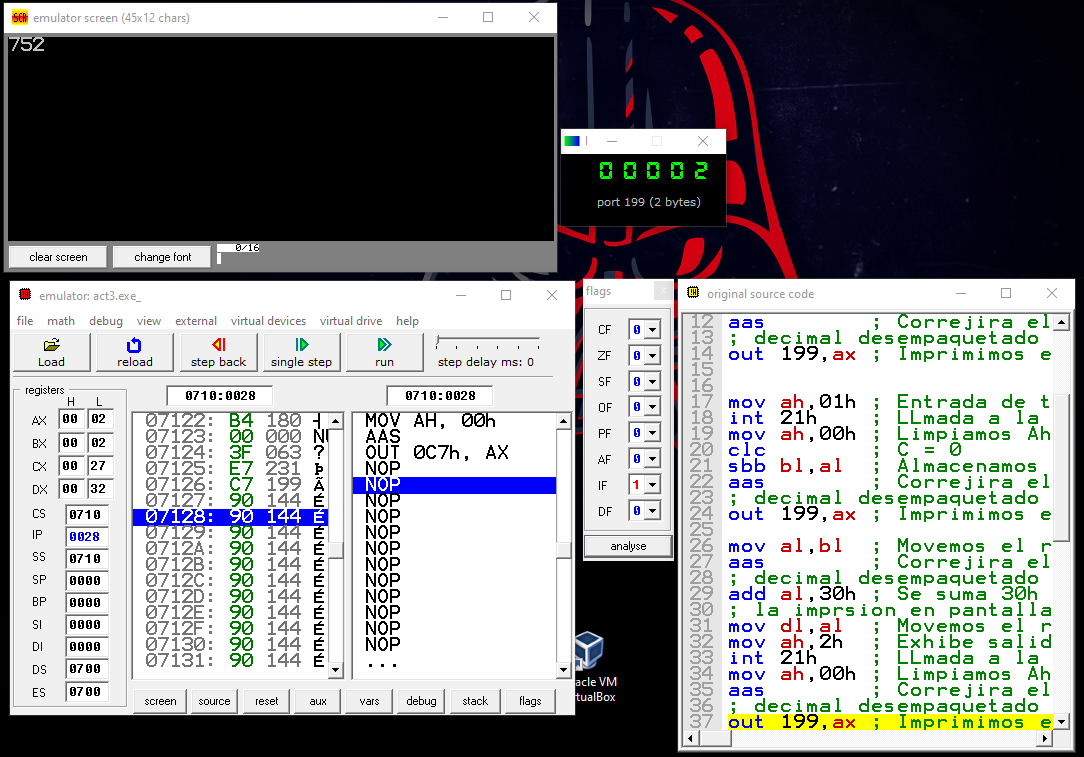
También se ejecutará la operación SBB que no tenga arrastre poniendo C en 0.



Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAS para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (BL).



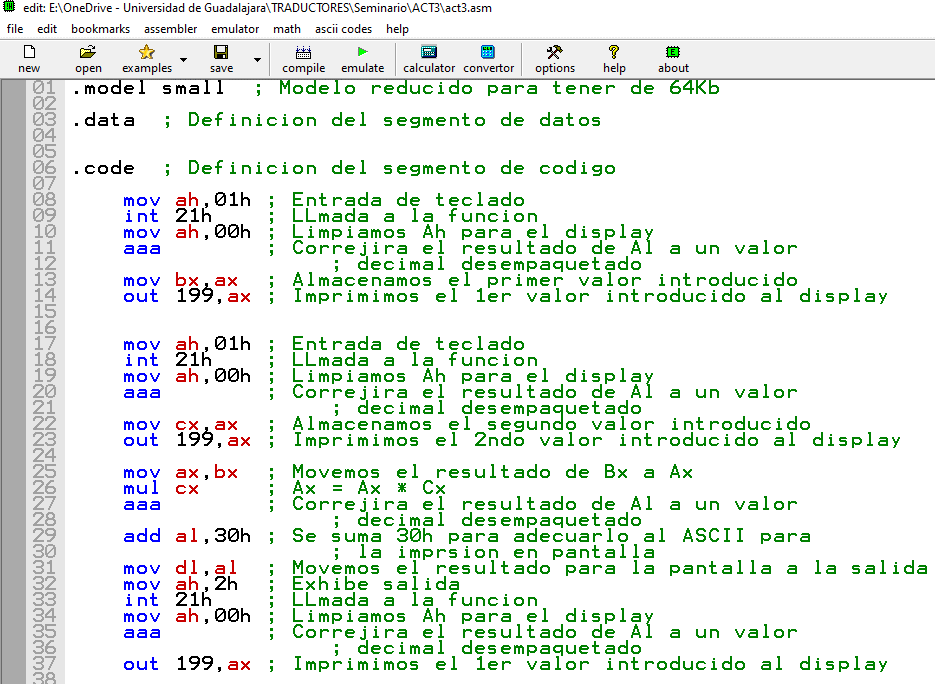
Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAS para la display y para la consola se usó la función anterior.

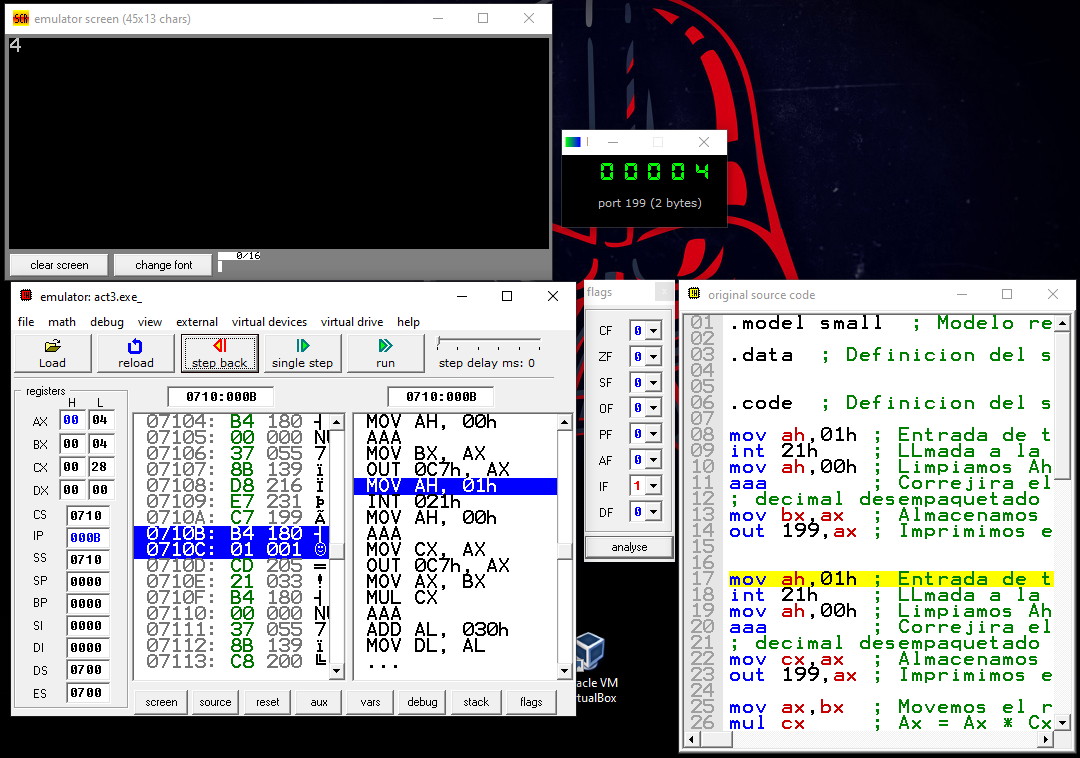


Para el resultado final de la operación se tuvo que hacer movimientos hacia (AL) para su respectiva conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAS para la display y para la consola se usó la función anterior y se le sumo 30.

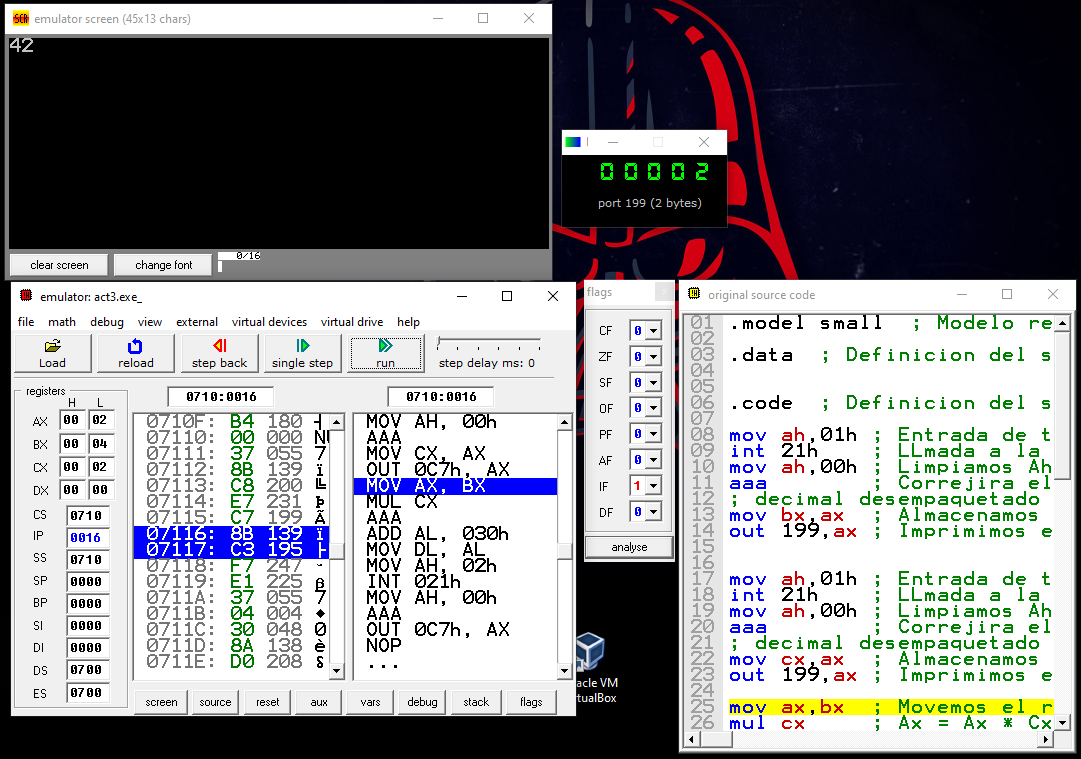
## MUL

Se realizo la operación MUL, en el que el usuario puede introducir sus propios valores para su respectivo almacenamiento, en el que se usó algunas estrategias para administrar los datos para obtener la multiplicación final deseado, así como de la función de AAA para convertirlo en un número fijo y se le sumara 30 para adecuarlo al símbolo de números de la tabla ASCII. También se utilizó las llamadas de función para introducir valores del teclado y las respectivas impresiones en el display como en la consola.

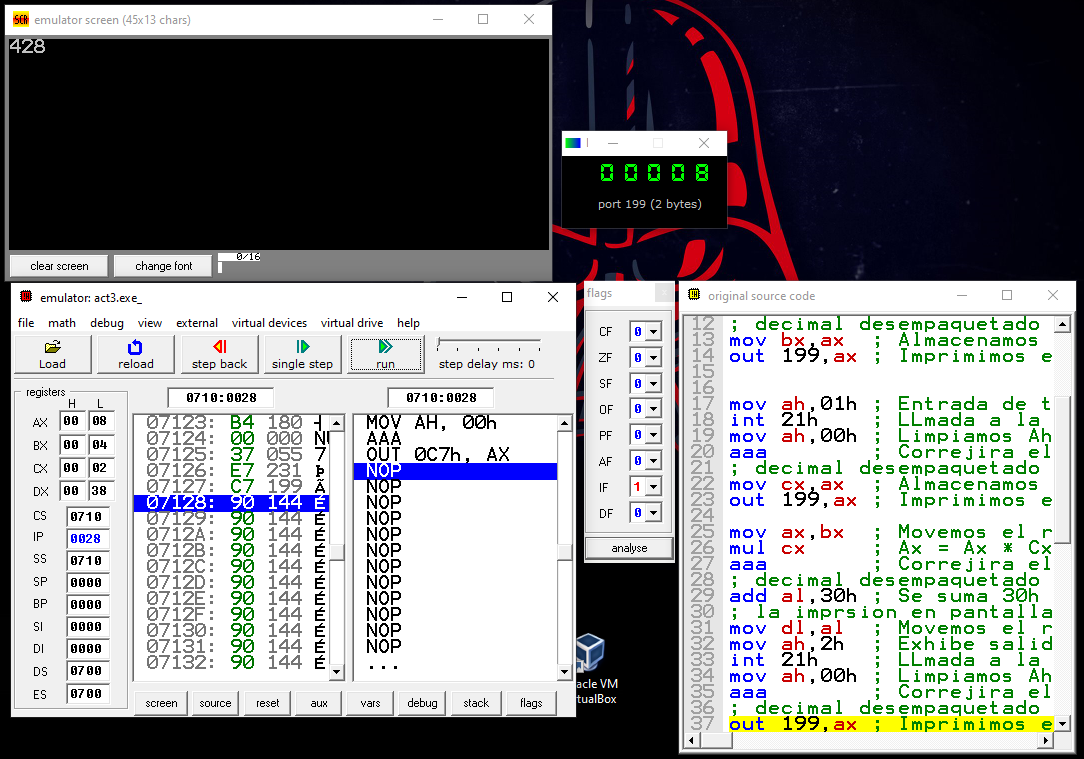




Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (Bx).



Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. También se tuvo que almacenar en otro registro para utilizarlo para la operación final (Cx).

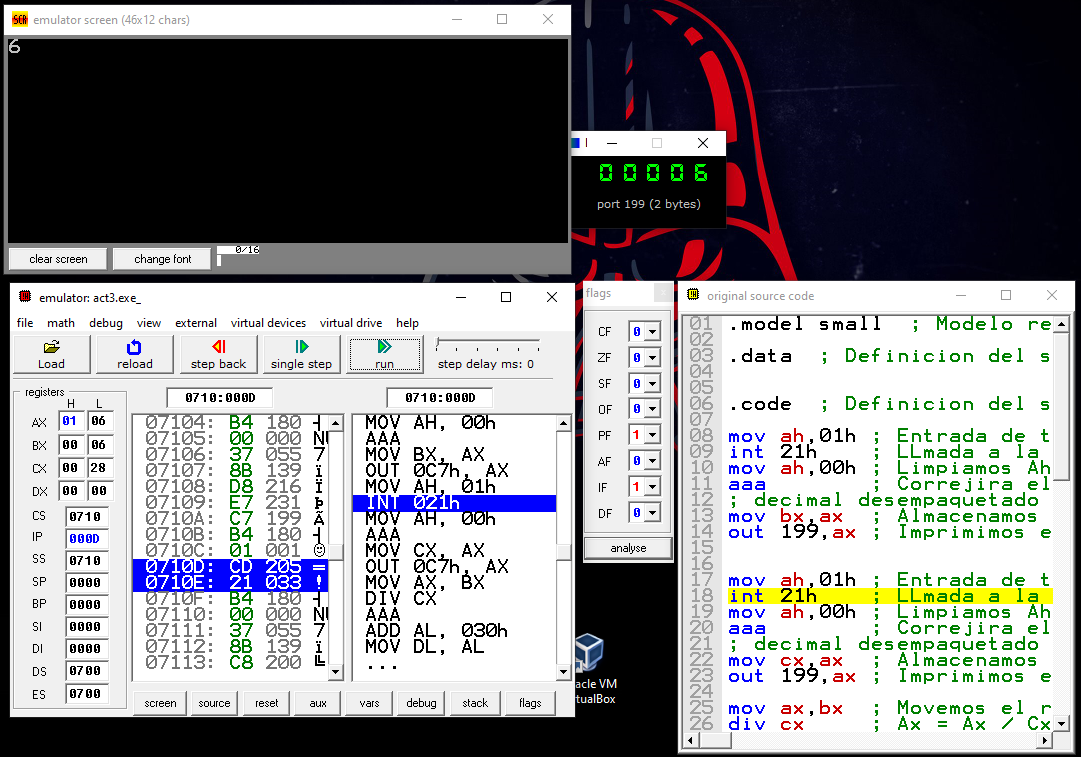


Para el resultado final de la operación se tuvo que hacer movimientos hacia (Ax) para su respectiva operación y conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAA para la display y para la consola se usó la función anterior y se le sumo 30.

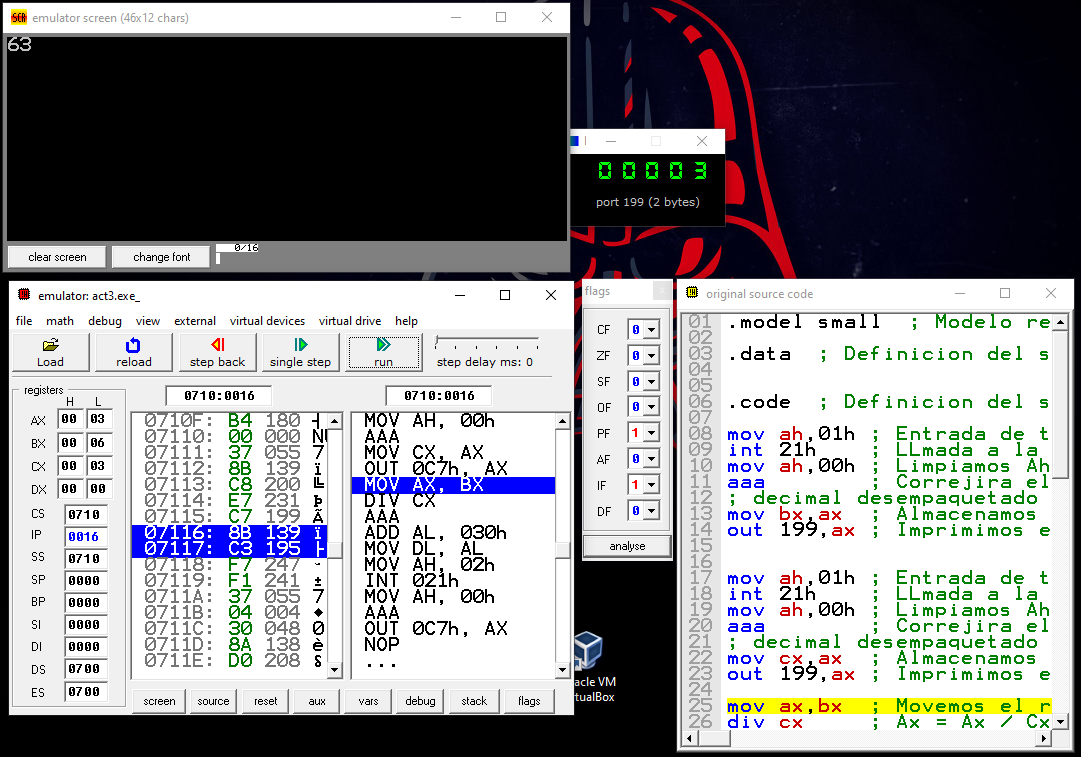
## DIV

Se realizo la operación DIV, en el que el usuario puede introducir sus propios valores para su respectivo almacenamiento, en el que se usó algunas estrategias para administrar los datos para obtener la división final deseado, así como de la función AAA para convertirlo en un número fijo y se le sumara 30 para adecuarlo al símbolo de números de la tabla ASCII. También se utilizó las llamadas de función para introducir valores del teclado y las respectivas impresiones en el display como en la consola.





Se introdujo el primer valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. Además de que se tuvo que almacenar en otro registro para utilizarlo para la operación final (Bx).



Se introdujo el segundo valor para su respectivo almacenamiento y correspondiente conversión para la consola y el display a utilizar. Utilizando solo la función AAA para la display y para la consola se usó la función anterior. También se tuvo que almacenar en otro registro para utilizarlo para la operación final (Cx).



Para el resultado final de la operación se tuvo que hacer movimientos hacia (Ax) para su respectiva operación y conversión hacia el símbolo de la tabla ASCII para su impresión en el display y en la consola. Utilizando solo la función AAA para la display y para la consola se usó la función anterior y se le sumo 30.

# Conclusión

En esta actividad aprendí mucho sobre el uso de los operadores aritméticos, las funciones para la visualización en la consola y en el display, y de las funciones para la conversión de los números de ASCII a decimal. Tuve algunos problemas al usar los operadores MUL y DIV ya que estos usan mas bits que los otros operadores por lo que tuve que adecuar sus valores en el código. Por otro lado, aprendí las diferentes funciones que hay en el lenguaje ensamblador para hacer las conversiones de ASCII según sea el operador a utilizar ya que como en el MUL y DIV son especificas en funcionamiento ya que usan diferentes estrategias para tener el valor correcto según el código ASCII. Finalmente, tuve algunos problemas con el display para mostrar mis valores deseados, además considero que los códigos realizados aun no son los mas eficientes por lo que aun necesito más práctica. Pero dentro de lo que cabe se lograron los objetivos deseados de practicar estos nuevos recursos del EMU8086.

# Bibliografía:

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