Assignment Solution Report

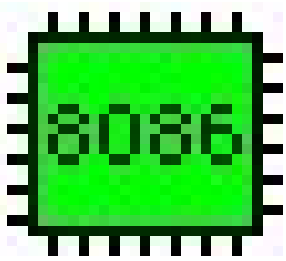
Contents

Summary

In this assignment, the x86 emulator software is used to develop an 8086 assembly program that performs different functionalities including the following:

- Declaring an array of integers
- Sum of integers
- Sum of Positive integers
- Zero-Crossing Count (ZCC)

About x86 emulator



In this assignment the 8086 Emulator tool was used to run the assembly code. EMU8086 - MICROPROCESSOR EMULATOR is a free emulator for multiple platforms. It provides its user with the ability to emulate old 8086 processors, which were used in Macintosh and Windows computers from the 1980s and early 1990s.

https://emu8086-microprocessor-emulator.en.softonic.com/

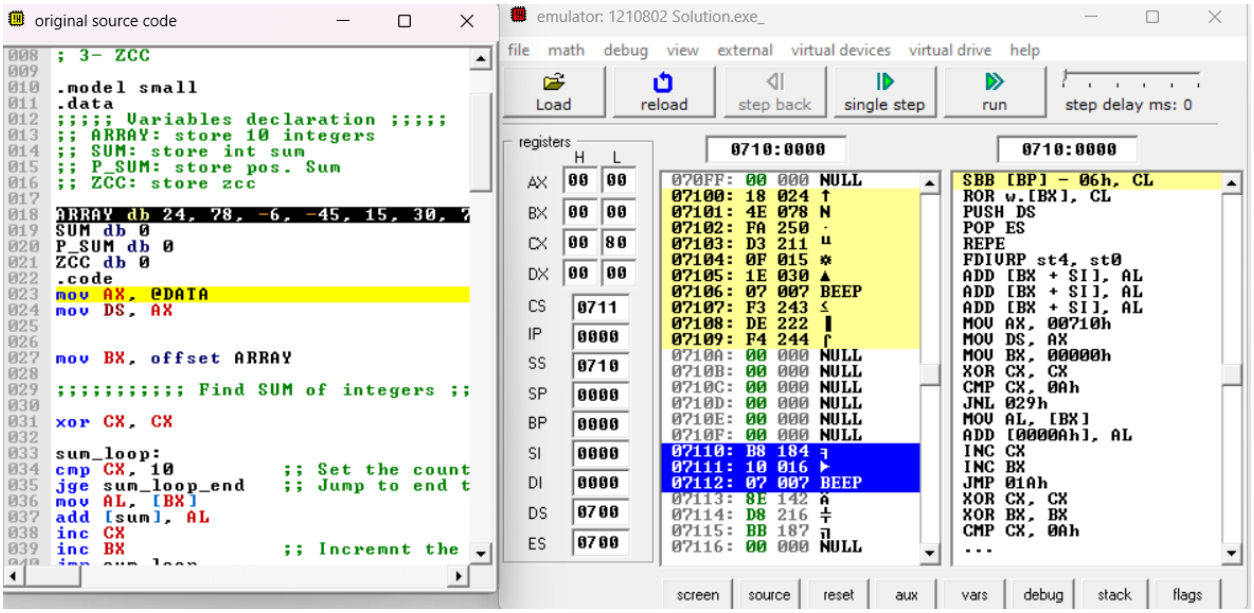
Variables

These are the variables used in the program:

- ARRAY: used to store the array of integers.
- SUM: used to store the sum of integers of ARRAY.
- P_SUM: used to store the sum of the ARRAY positive integers.
- ZCC: Used to store the Zero-Crossing Count of the ARRAY integers.

Declaring the array

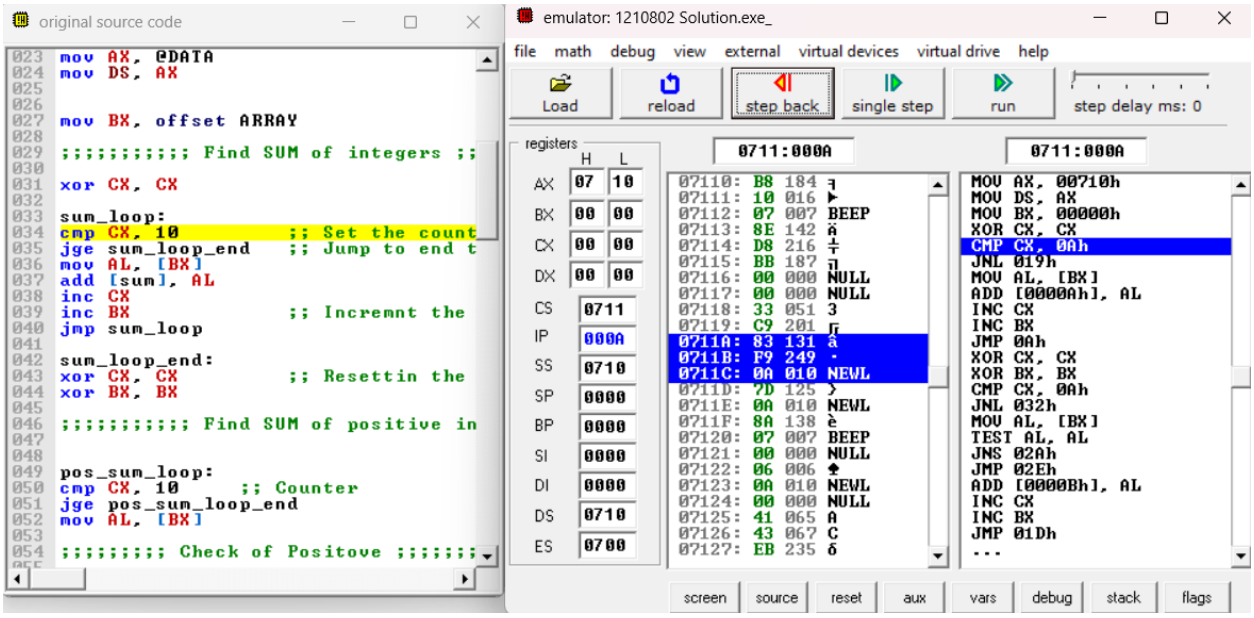
First thing that this program does is declare an array of 10 signed integer numbers (8-bit) in the memory with their initial values. It was chosen to be half positive and half negative integers.

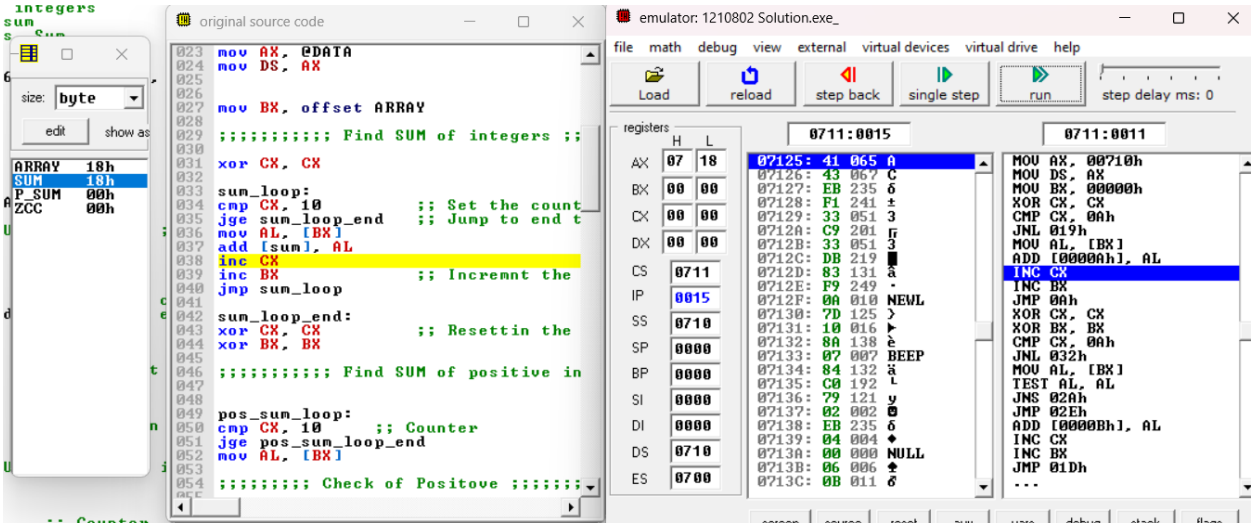


It is seen on the labeled lines using the yellow color, that the array is declared for 10 elements.

Finding the Sum

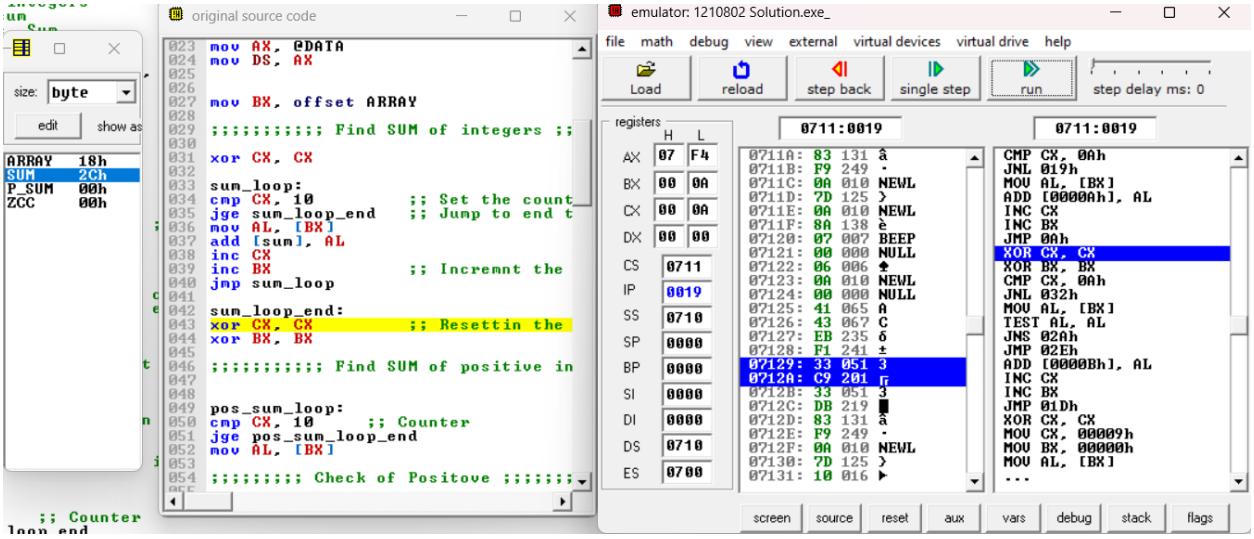
The second function that this program does is finding the arithmetic sum of all elements of the array and store it in the memory in a variable called SUM. This was done using the CX register and looping until it reaches 10.





It is noticed that after starting a single step of the sum loop that the SUM variable value has changed from 00h to 18h.

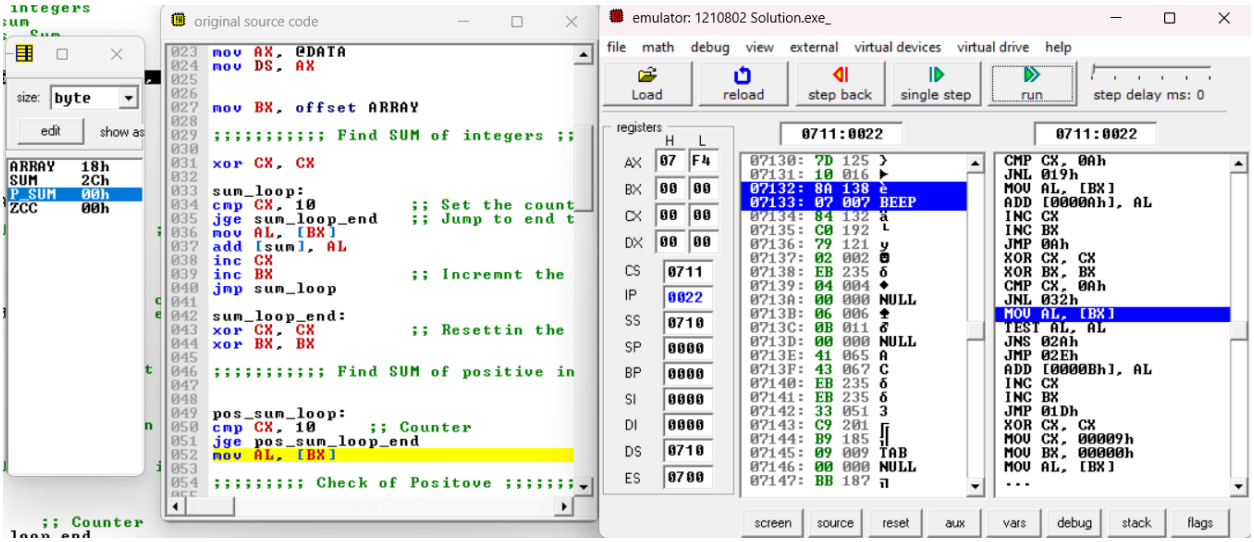
When the final element is added then we noticed the SUM variable value is 2Ch



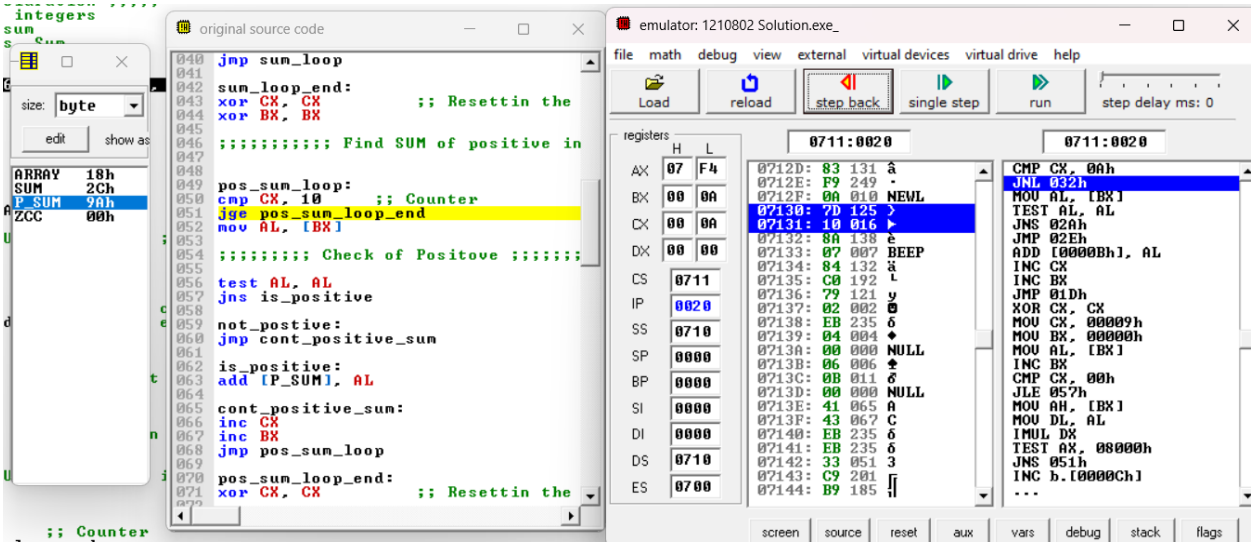
Which is a valid output, since the arithmetic sum of the elements equal 44 decimal equivalent to 2C in hexadecimal.

Finding the Positive Sum

Then the program finds the sum of the elements that are greater than **zero** in the array. And store it in a memory variable called **P_SUM**. This is done using the (test) mnemonic or instruction which is a bitwise AND.



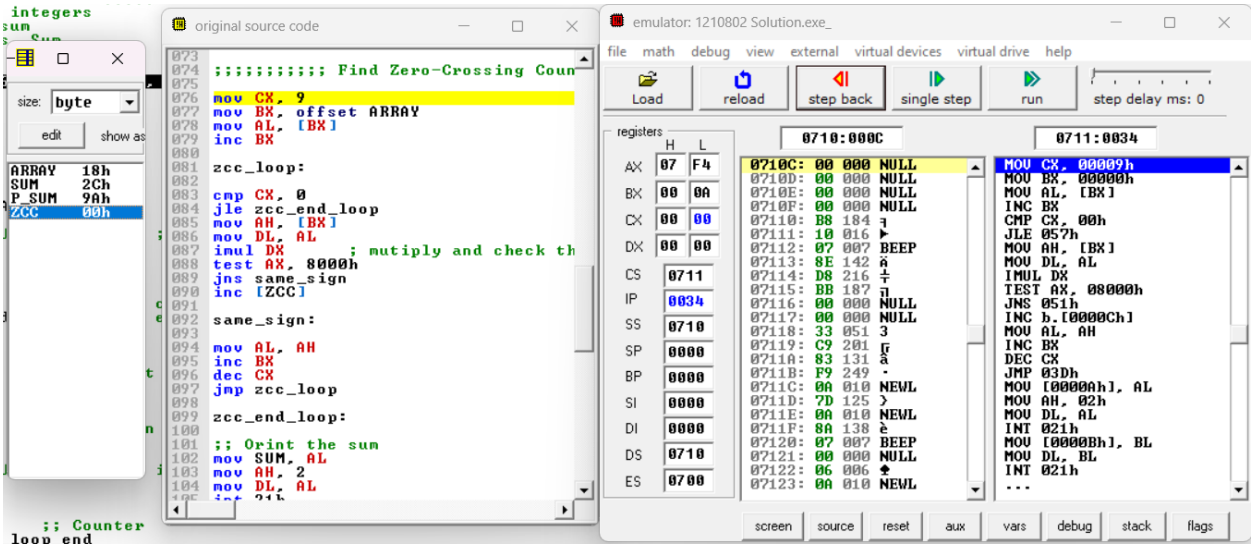
It is noticed that the P_SUM initial value is 00h. And it starts to change once the loop changes.



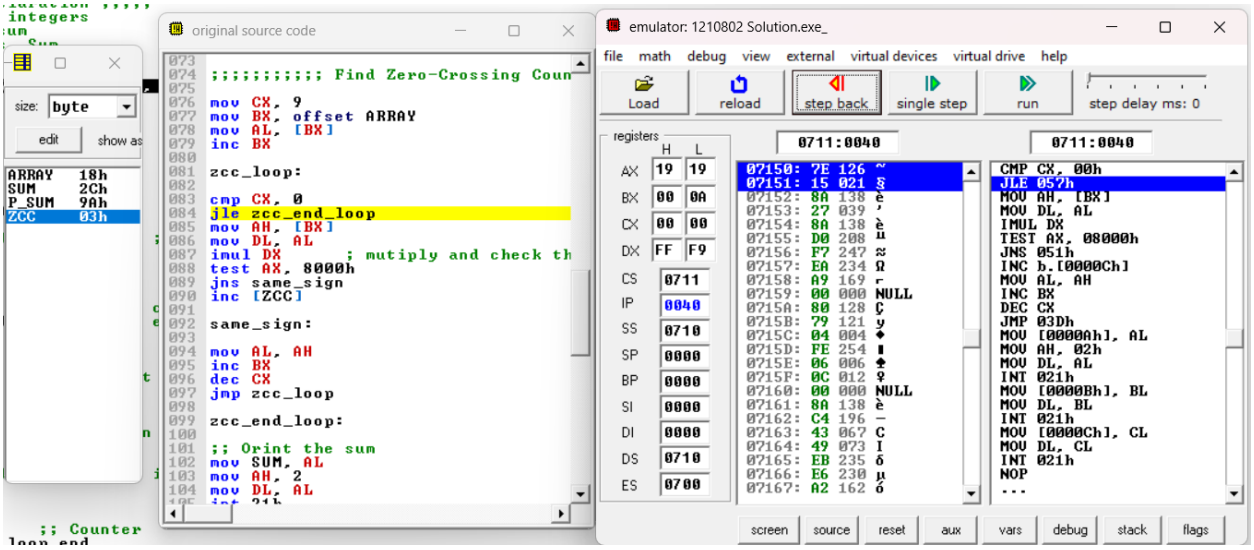
It is noticed that the final output is valid for the positive SUM which is supposed to be for the running ARRAY 154 in decimal that is 9A in hexadecimal.

Finding the Zero-Crossing Count (ZCC)

Finally the program finds the Zero-Crossing Count (ZCC) of the array elements. Which is computed by counting the number of times the successive samples of the array change their signs. For example, for an array A = {+1,-2, -8, +3, -4, +5, +6}, the ZCC is 4, i.e. the samples cross zero-axes 4 times. And the value is stored in the variable called ZCC.



After running the loop:



In the running example array there exists 4 Zero-Crossing values. So the final output is valid that is stored in ZCC variable 03 in Hexadecimal.