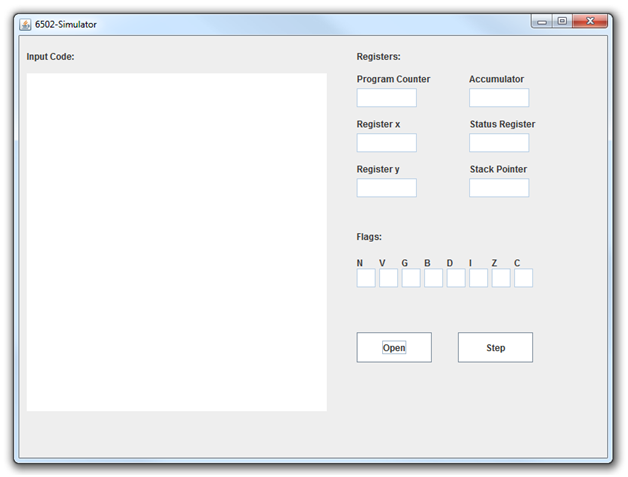
User Guide

Introduction

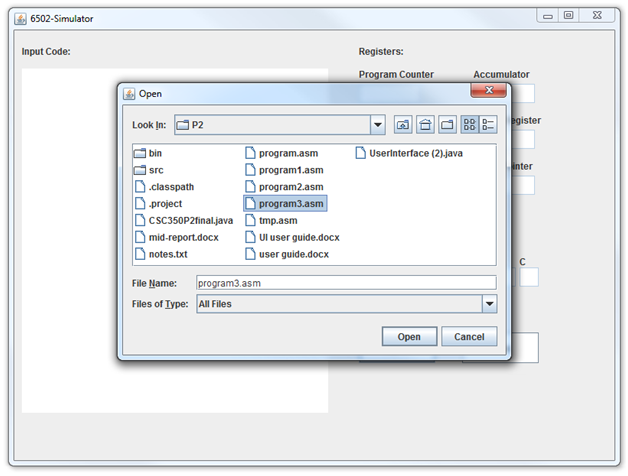
This simulator has a simple user interface that can run all of the 6502 instructions. We have provided 3 sample programs with instructions on how to run them below. Also, there are instructions on how to create your own programs at the end of the guide.

Start

To run the program ensure that Java 1.6 (or newer) is installed on your machine. Double-click the CSC350P2.jar file and the program will automatically start and show up in a small frame as shown below.

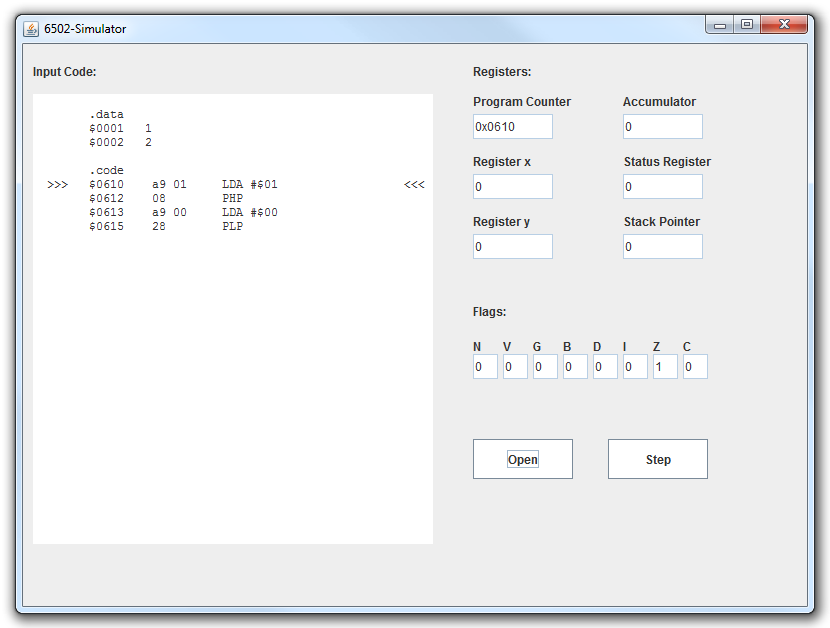
Choosing a file

Click the “Open” button and a file choosing frame will pop up. This is where you will choose the program you would like to run. Choose a 6502 machine language file saved as a .txt or .asm. The example ones provided are called program1.asm, program2.asm, etc. Once you found a program click “open”.



Explanation of the UI

The program should appear on the input text area on the left hand side of the UI. The flags will all be set to 0 (false) and the program counter will display the address for the next instruction. The arrows indicate which line you’re currently on.



Once the file is chosen, you can run one instruction of the program by clicking on the “Step” button. The UI will step through the program line by line, updating the values of the registers and flags as they change for each line. Acronyms for flag names are used on the UI for simplicity. These represent:

* N - Negative
* V - Overflow
* G - Ignored
* B - Break
* D - Decimal
* I - Interrupt
* Z - Zero
* C - Carry

The values of the flags will output to the text boxes as a 1 or 0, corresponding to true or false. Registers are labeled and all will output to the text boxes in base 10, with the exception of the program counter. The program counter will output as a hexadecimal value and it will increment as each line of the program is processed. You can use this value to determine where in the program you currently are. When the program is finished, it will halt and the window will need to be closed.

Example Programs

program1.asm

* This program demonstrates how to put data into the specific memory locations and to use the arithmetic instructions.
* The values stored in memory at 0001, 0042, 0503 are all added together and stored in the accumulator
* Then the sum is stored back in memory at 0503

program2.asm

* This program shows how to use a branch.
* First the number 8 is loaded into the X register.
* Then it is decremented by 1 and compared with the number 3
* If the number is the not equal to 3 then it will branch back to the instruction on line 0602

program3.asm

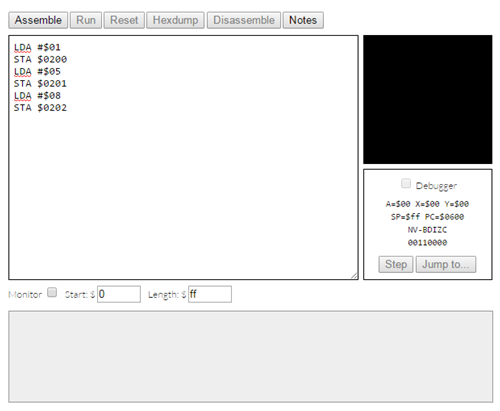
* This program demonstrates how to use the jump and system functions
* The BRK instruction forces an interrupt and gets the address stored FFFE, and pushes the PC and the flags to the stack.
* The address stored points to RTI (return from interrupt) which pulls the PC and flags back off the stack and returns back to where the program was running.
* Next the program will add 1 to the accumulator and then jump back to the beginning.

program4.asm

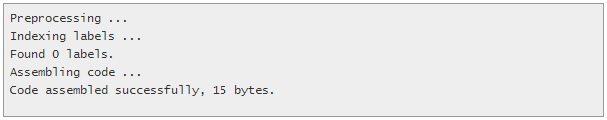
* This program uses jump to subroutine and return to subroutine by simulating a for loop
* The first JSR jumps to the LDX which loads 0 into the X register then returns
* The second JSR jumps to INX which increments X then compares it to 5
* This will repeat until X equals 5 and then it will reach the final JSR which jumps to the end

Since the simulator can only read machine language a separate tool is needed to assemble the assembly language. There are many free 6502 assembler’s on the internet but I recommend using http://skilldrick.github.io/easy6502/ because it’s simple and easy to use. This assembler lets you write assembly code and converts it into machine language. The full instruction set for the 6502 can be found at http://www.obelisk.demon.co.uk/6502/instructions.html. Together these tools can be used to create any 6502 program that can run in our simulator as shown on the next page.

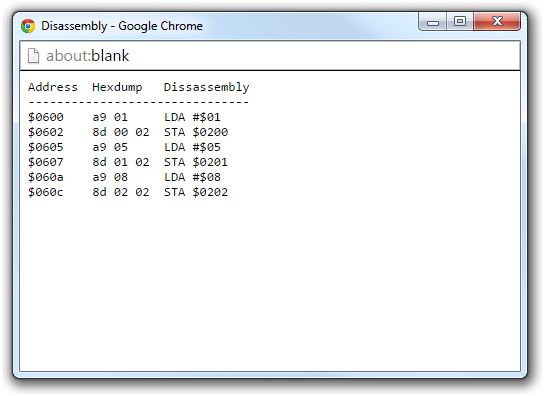
Creating a file

1) Go to the 6502 assembler and write a program using any instructions for the 650

2) Click the “Assemble” button and it will say “Code assembled successfully” unless there is an error with your code.



3) Click the “Disassemble” button and a popup will appear with the machine language.



4) Copy the machine language into a file, removing the first 2 lines, and saving as a .txt or .asm.

5) If you want to add your own data use the format shown in program3.asm. Start your program with “.data” and specify the address and value you want to put in memory. After you’ve added all of your data, write “.code” and then copy the rest of your program like in step 4 above.