

Student Name:

CMSC 311 Computer Organization

LAB 6: Assembly Language programming Using LC-3 Simulator

Instructor: Dr. Cang Ye

Email: cye@vcu.edu

Office & Hours: E2240, TR 12-1 PM

TA: Lingqiu Jin

Email: jinl@mymail.vcu.edu

Office & Hours: E2264, MW 2-3 PM

Assignment:

- 1) Write an LC-3 assembly language program that divides a positive number X by another positive number Y and stores the integer part of the result in R3, i.e., $R3 = \text{int}(X/Y)$. Your code should use an iterative construct.
- 2) Write an assembly program that produce the modulo of 2 positive numbers (i.e., the remainder of the division) and store the result in R4, i.e., $R4 = X \% Y$.

The value of X should be stored in 0x3100 and the value of Y should be stored in 0x3101. Also, let $X = 0x005A$ and $Y = 0x0007$. Your program should start at 0x3000.

Instructions:

1. Your lab report is due no later than the date of the next class. Your lab report (in pdf format) should include your assembly code and screenshot(s). Submit both your lab report and source codes in separate files via blackboard. Please comment your code to make it easy to understand.
2. The screenshot(s) should show both your assembly code and simulation result. The code should start at address x3000. All register information, the assembly code, and the input data should be visible on the screenshot(s).
3. Grading policy: lab report 20%, source code 80%.

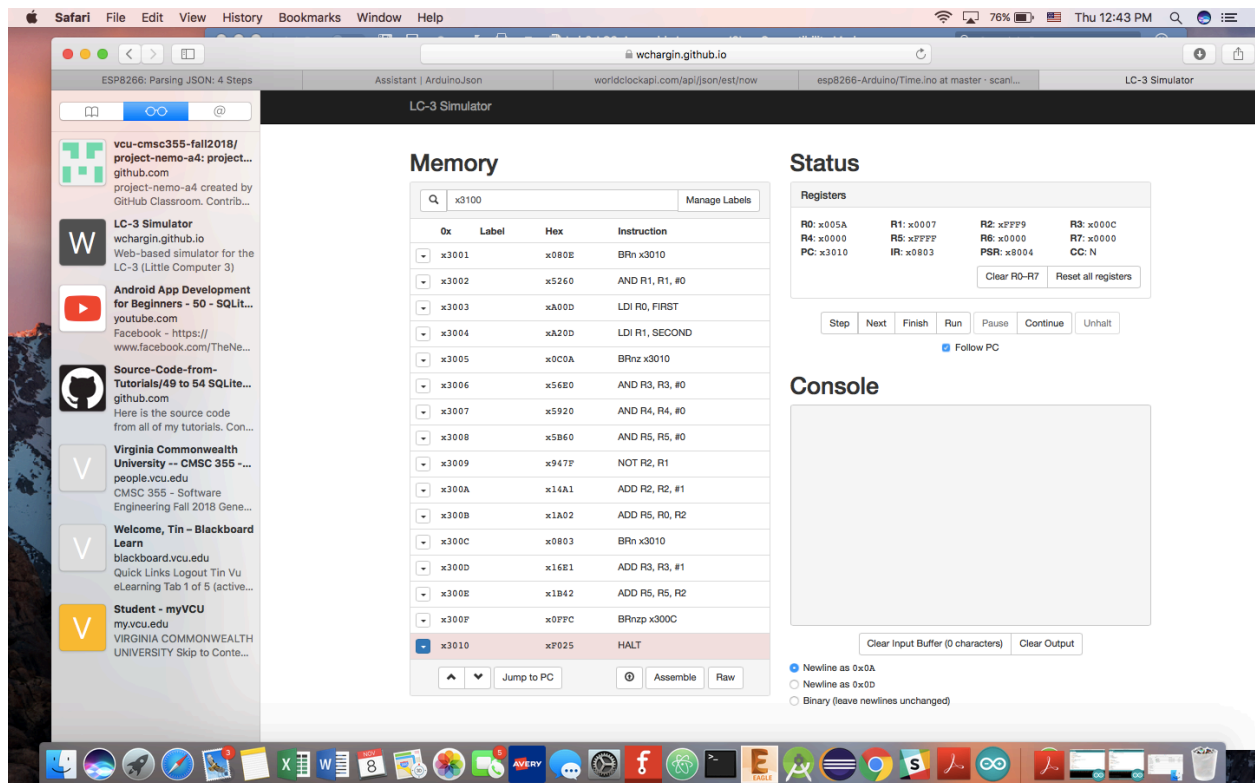
- 1) Write an LC-3 assembly language program that divides a positive number X by another positive number Y and stores the integer part of the result in R3, i.e., $R3 = \text{int}(X/Y)$. Your code should use an iterative construct.
 - a. In this program I used the subtraction method to find the division result of X and Y. First the program check if the bottom number is zero. Conditions are set to have no negative number as input in this program. The result will be stored in R3. We should see the top number and bottom number at R0 and R1 respectively at the end of each run also. Before running, we have to input the top and. Due to the available space of the screen! The first ADD instruction isn't shown.

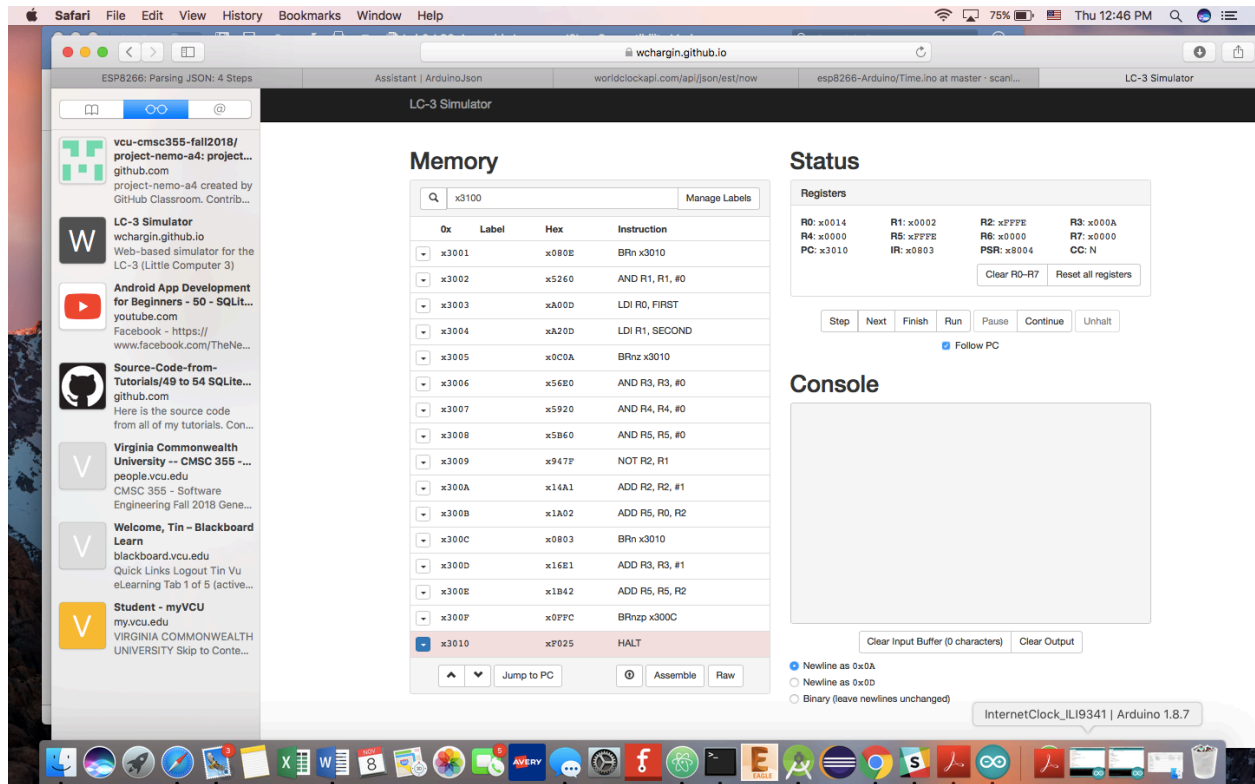
NOTE: THE .ASM FILES FOR BOTH 1 AND 2 WERE COMPILED FROM WEBSITE

<http://spacetechn.github.io/LC3Simulator/>

CODE:

```
.ORIG x3000
AND R0,R0, #0 ;; R0, WILL HOLD THE FIRST NUMBER
BRn #14    ;; NO NEGATIVE INTEGER IN THIS PROGRAM
AND R1,R1, #0 ;; R1, WILL HOLD THE SECOND NUMBER
LDI R0, FIRST
LDI R1, SECOND
BRnz #10    ;; CAN'T DIVIDE BY ZERO AND ALSO NO NEGATIVE INTEGER IN THIS PROGRAM
AND R3,R3, #0 ;; R3 WILL HOLD THE RESULT
AND R4,R4, #0 ;; R4, WILL HOLD THE REMAINING
AND R5, R5, #0 ;; HOLD THE TEMP RESULT
NOT R2, R1  ;; CHANGE SIGN OF THE SECOND NUMBER
ADD R2, R2, #1 ;;
ADD R5, R0, R2 ;; FIRST NUMBER - SECOND NUMBER
BRn #3      ;;IF THE RESULT IS NEGATIVE--> HALT
;;ADD R0,R5,#0 ;; ELSE PUT THE RESULT OF R3 BACK INTO R0
ADD R3, R3, #1 ;; INCREASE THE COUNTER
ADD R5, R5,R2 ;; FIRST - SECOND AGAIN
BRnzp #-4   ;; UNCONDITIONAL BRANCH
HALT
FIRST .FILL x3100
SECOND .FILL x3101
.END
```





- 1) Write an assembly program that produce the modulo of 2 positive numbers (i.e., the remainder of the division) and store the result in R4, i.e., $R4 = X \% Y$.
 - a. This program is just a modified version of the first program in this lab. I added 2 lines after the loop to calculate the remainder of X and Y. all the preconditions and postconditions stay the same. Input at the same addresses. The remainder is stored at register R4.

CODE:

```
.ORIG x3000
AND R0,R0, #0 ;; R0, WILL HOLD THE FIRST NUMBER
BRn #14      ;; NO NEGATIVE INTEGER IN THIS PROGRAM
AND R1,R1, #0 ;; R1, WILL HOLD THE SECOND NUMBER
LDI R0, FIRST
LDI R1, SECOND
BRnz #10     ;; CAN'T DIVISE BY ZERO AND ALSO NO NEGATIVE INTEGER IN THIS PROGRAM
AND R3,R3, #0 ;; R3 WILL HOLD THE RESULT
AND R4,R4, #0 ;; R4, WILL HOLD THE REMAINING
AND R5, R5, #0 ;; HOLD THE TEMP RESULT
NOT R2, R1   ;; CHANGE SIGN OF THE SECOND NUMBER
ADD R2, R2, #1 ;;
```

```

ADD R5, R0, R2 ;; FIRST NUMBER - SECOND NUMBER
BRn #3      ;;IF THE RESULT IS NEGATIVE--> HALT
;;ADD R0,R5,#0 ;; ELSE PUT THE RESULT OF R3 BACK INTO R0
ADD R3, R3, #1 ;; INCREASE THE COUNTER
ADD R5, R5,R2 ;; FIRST - SECOND AGAIN
BRnzp #-4   ;; UNCONDITIONAL BRANCH
ADD R4, R5, R1 ;; STORE THE REMAINDER
HALT
FIRST .FILL x3100
SECOND .FILL x3101
.END

```

The screenshot shows the LC-3 Simulator web interface. The left sidebar contains navigation links for various resources. The main area is divided into three panels: Memory, Status, and Console.

Memory Panel: Displays a table of memory locations. The search bar shows 'x3100'. The table has columns for Address (0x), Label, Hex, and Instruction.

0x	Label	Hex	Instruction
x300D		x16E1	ADD R3, R3, #1
x300E		x1B42	ADD R5, R5, R2
x300F		x0F9C	BRnzp x300C
x3010		x1941	ADD R4, R5, R1
x3011		x7025	HALT
x3012	FIRST	x3100	ST R0, x2F13
x3013	SECOND	x3101	ST R0, x2F15
x3014		x3101	ST R0, x2F16
x3015		x0000	NOP
x3016		x0000	NOP
x3017		x0000	NOP
x3018		x0000	NOP
x3019		x0000	NOP
x301A		x0000	NOP
x301B		x0000	NOP
x301C		x0000	NOP

Status Panel: Shows the current state of registers and program counter.

Registers			
R0: x005A R4: x0006 PC: x3011	R1: x0007 R5: xFFFF IR: x1941	R2: xFFFF R6: x0000 PSR: x8001	R3: x000C R7: x0000 CC: P

Buttons: Stop, Next, Finish, Run, Pause, Continue, Unhalt. Follow PC button.

Console Panel: Contains a text area for output and buttons for 'Clear Input Buffer (0 characters)' and 'Clear Output'.

SAMPLE RUN WITH X = 90 AND Y = 7

RESULT IS IN R3: 12 (x00C)

REMAINDER IS IN R4: 6 (x006)