LAB D

Gavin Sit

215 043 870

2021E

**Part 1:**

Code:

addi x2, x0, 1600 //initialize the stack to 1600, x2 = stack pointer

ecall x6, x0, 5 //read the input to x5

add x10, x6, x0 //put the parameter in x10

addi x2, x2, -8 //make room to store x5

sd x6, 0(x2)

jal x1, myswap

ld x6, 0(x2) //restore x5 from the stack

addi x2, x2, 8 //pop the stack

ecall x0, x10, 2 //print the original value returned from the functions

ecall x0, x6, 2 //print the original value

ORG 96

myswap:

addi x2, x2, -8

sd x8, 0(x2)

addi x5, x0, 255 // x5 is a 0000 ..011111111

and x6, x10, x5 // x6 is m=i&a, m has lest significant byte

srai x7, x10, 8 //x7 is I shifted to left by 8

and x7, x7, x5 //x7 is the second least sig. Byte of I

srai x8, x10, 16 // shift I by 16 put it in x8

slli x8, x8, 8

or x8, x8, x6 //append the second least significant byte

slli x8, x8, 8

or x8, x8, x7

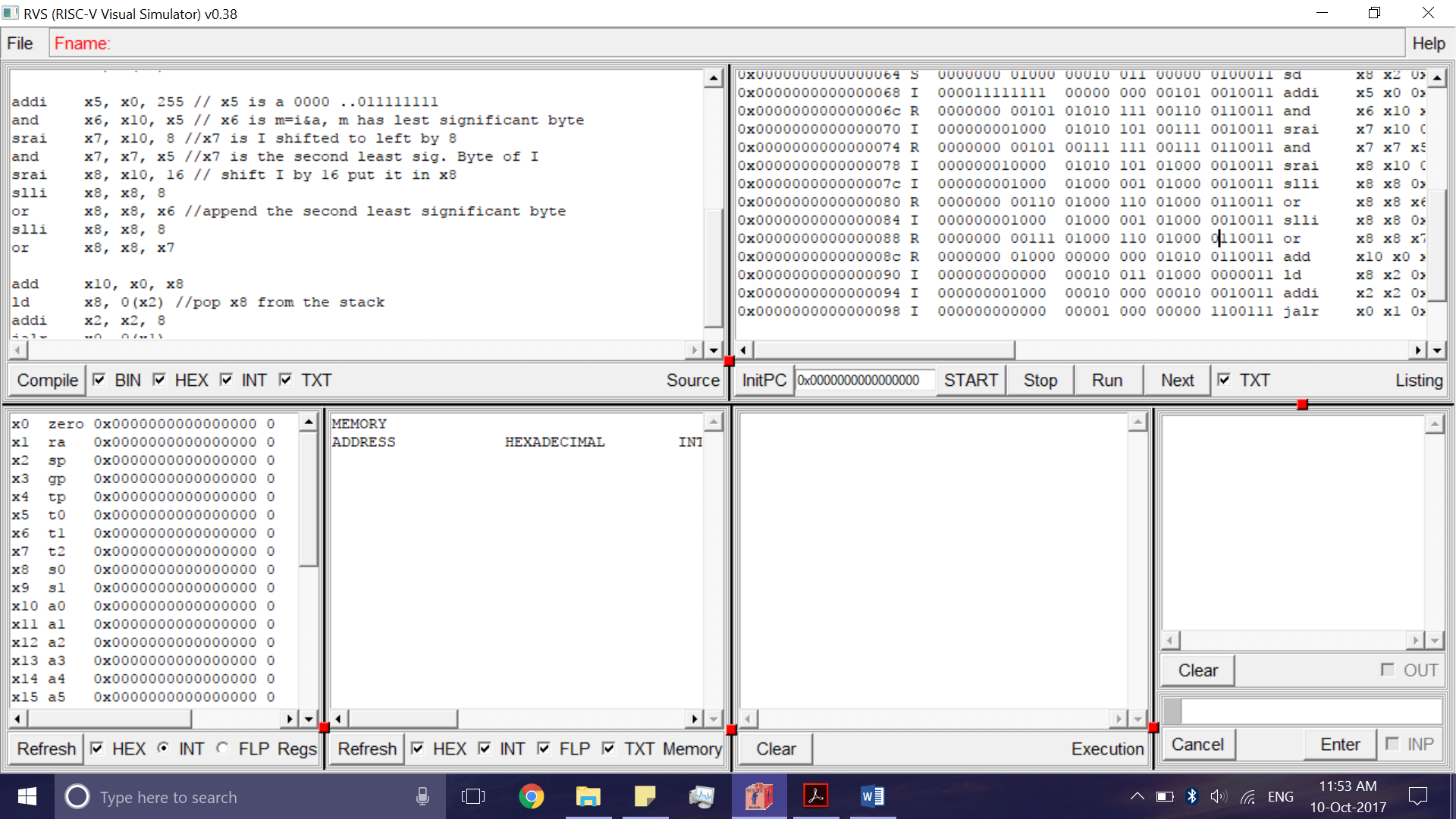
add x10, x0, x8

ld x8, 0(x2) //pop x8 from the stack

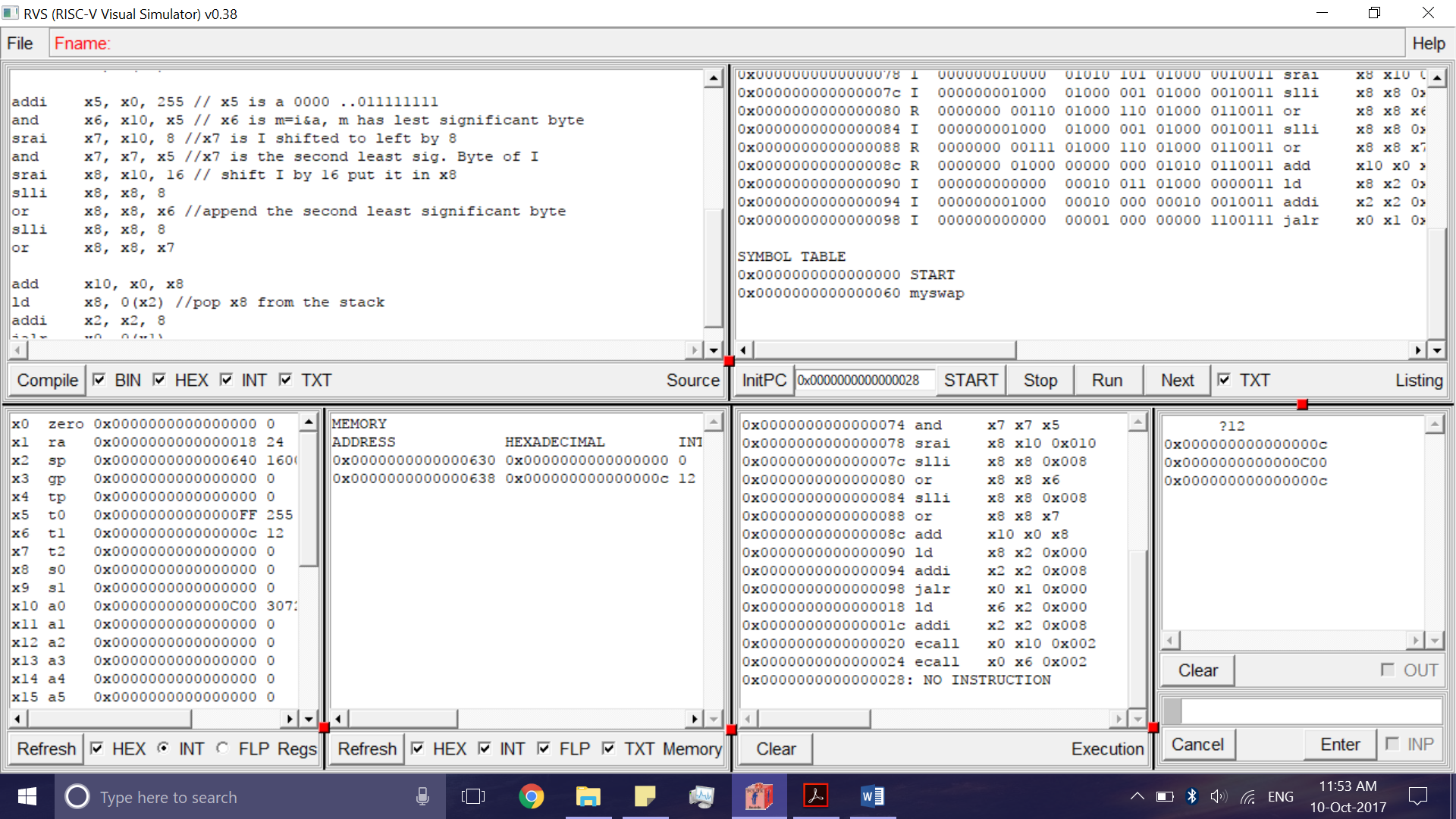
addi x2, x2, 8

jalr x0, 0(x1)

After Compiling:



After Running:



**Part 2**

Code:

ecall x10, x0, 5 //get a number

fact: addi sp, sp, -16

sd x1, 8(sp)

//sd x10, 0(sp)

addi x5, x10, -2 //x5 = n - 2

bge x11, x5, L1 //it wont equal 0, but if it's greater

addi x10, x0, 1 //return one

addi x11, x0, 3 //returns three

addi sp, sp, 16 //pop 2 items off stack

jalr x0, 0(x1) //return to caller

addi x6, x10, 0 //return from jal: move result of fact (n-2) to x6

ld x10, 0(sp) //restore argument n

ld x1, 8(sp) //restore the return address

addi sp, sp, 16 //adjust stack pointer to pop 2 items

mul x10, x10, x6 //return n \* fact(n-2)

addi x10, x10, 1 //adds 1 to x10

jalr x0, 0(x1)

L1: addi x10, x10, -2

jal x1, fact

After Compiling:

After Running

fact: addi sp, sp, -16

sd x1, 8(sp)

sd x10, 0(sp)

addi x5, x10, -1 //x5 = n - 1

bge x5, x0, L1 // if (n - 1) >= 0, go to L1

addi x10, x0, 1 //return one

addi sp, sp, 16 //pop 2 items off stack

jalr x0, 0(x1) //return to caller

addi x6, x10, 0 //return from jal: move result of fact (n-1) to x6

ld x10, 0(sp) //restore argument n

ld x1, 8(sp) //restore the return address

addi sp, sp, 16 //adjust stack pointer to pop 2 items

mul x10, x10, x6 //return n \* fact(n-1)

jalr x0, 0(x1)

L1: addi x10, x10, -1

jal x1, fact