

Lab Report

ECPE 170 – Computer Systems and Networks – Spring 2018

Name: Melissa Chow

Lab Topic: MIPS Assembly Programming (Basic) (Lab #: 10)

PART 1:

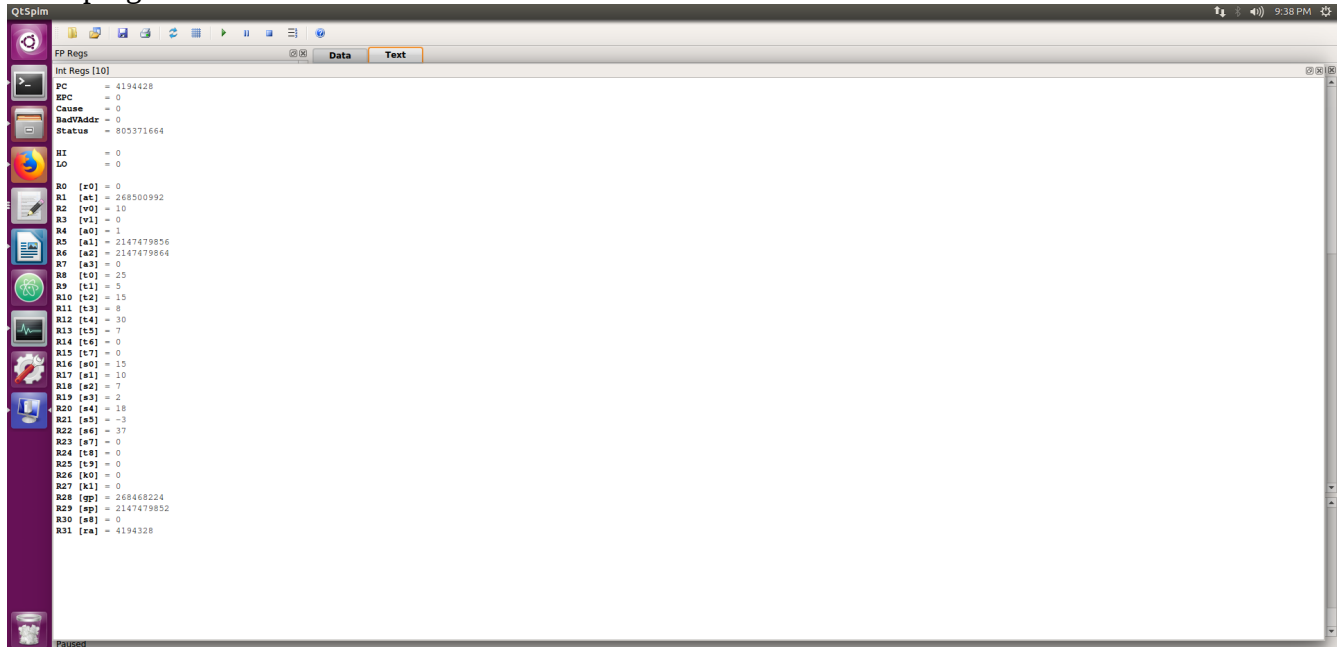
Question #1:

Take two screenshots of the MIPS register panel: one before your program runs, and one after your program finishes. Put the register panel in Decimal mode (right-click) so it is easy to see register values.

Before program runs:



After program runs:

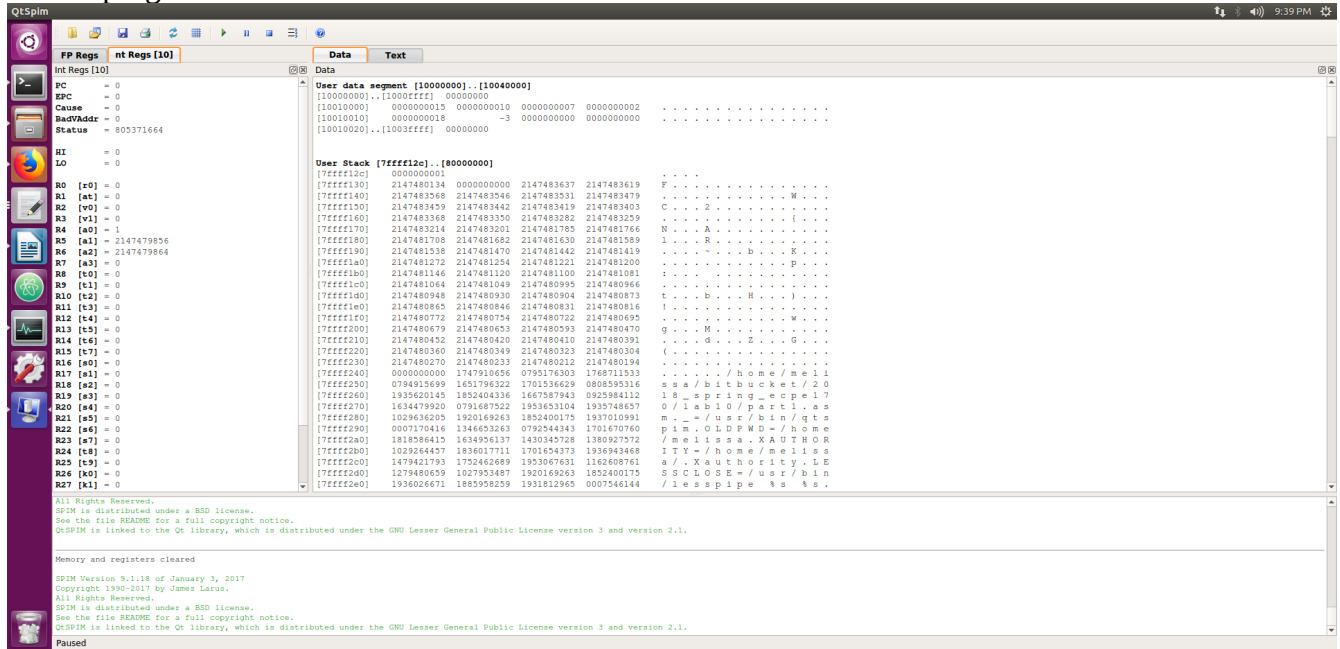


Question #2:

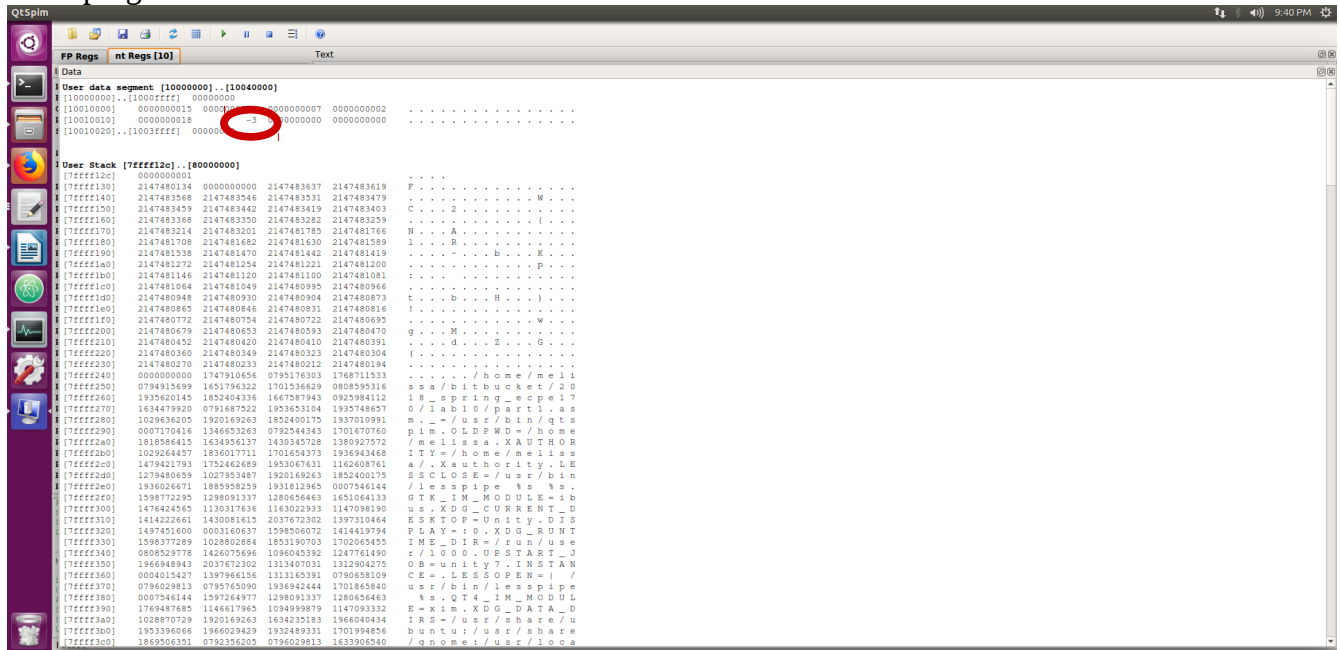
Take two screenshots of the MIPS memory panel (data tab): one before your program runs, and one after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. In the after-execution capture, **circle the memory location (not register) that**

contains the final calculated value of Z.

Before program runs:



After program runs:



PART 2:

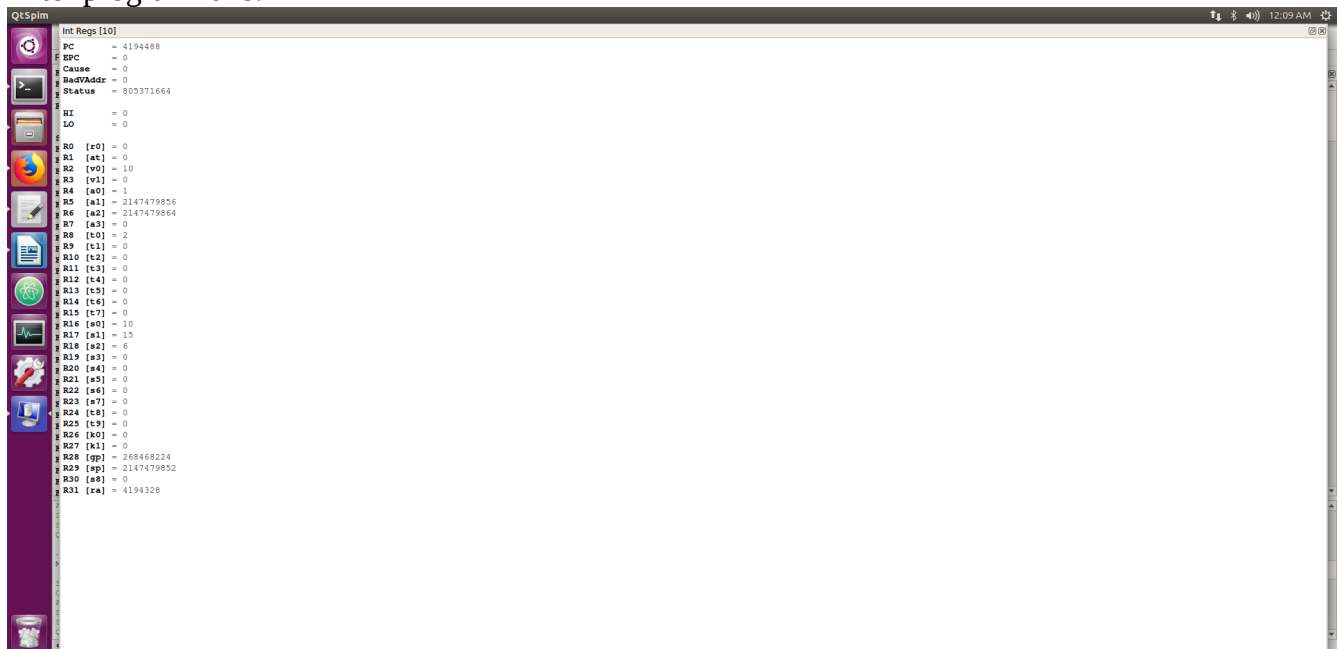
Question #3:

Take two screenshots of the MIPS register panel: one before your program runs, and one after your program finishes. Put the register panel in Decimal mode (right-click) so it is easy to see register values.

Before program runs:



After program runs:



Question #4:

Take two screenshots of the MIPS memory panel (data tab): one before your program runs, and one after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. In the after-execution capture, **circle the memory location (not register) that contains the final calculated value of Z.**

Before program runs:

The screenshot shows the QtSpim memory dump window. The 'Data' tab is selected, displaying memory addresses from 10000000 to 175237924. The 'User data segment' is visible, showing a series of zeros. The 'User Stack' is also visible, showing a series of zeros. The 'FP Regs' and 'nt Regs' tabs are visible at the top.

After program runs:

The screenshot shows the QtSpim memory dump window after the program has run. The 'Data' tab is selected, displaying memory addresses from 10000000 to 175237924. The 'User data segment' is visible, showing a series of zeros. The 'User Stack' is also visible, showing a series of zeros. The 'FP Regs' and 'nt Regs' tabs are visible at the top. A red circle highlights the value 00000000 in the 'User data segment' at address 10000000.

PART 3:

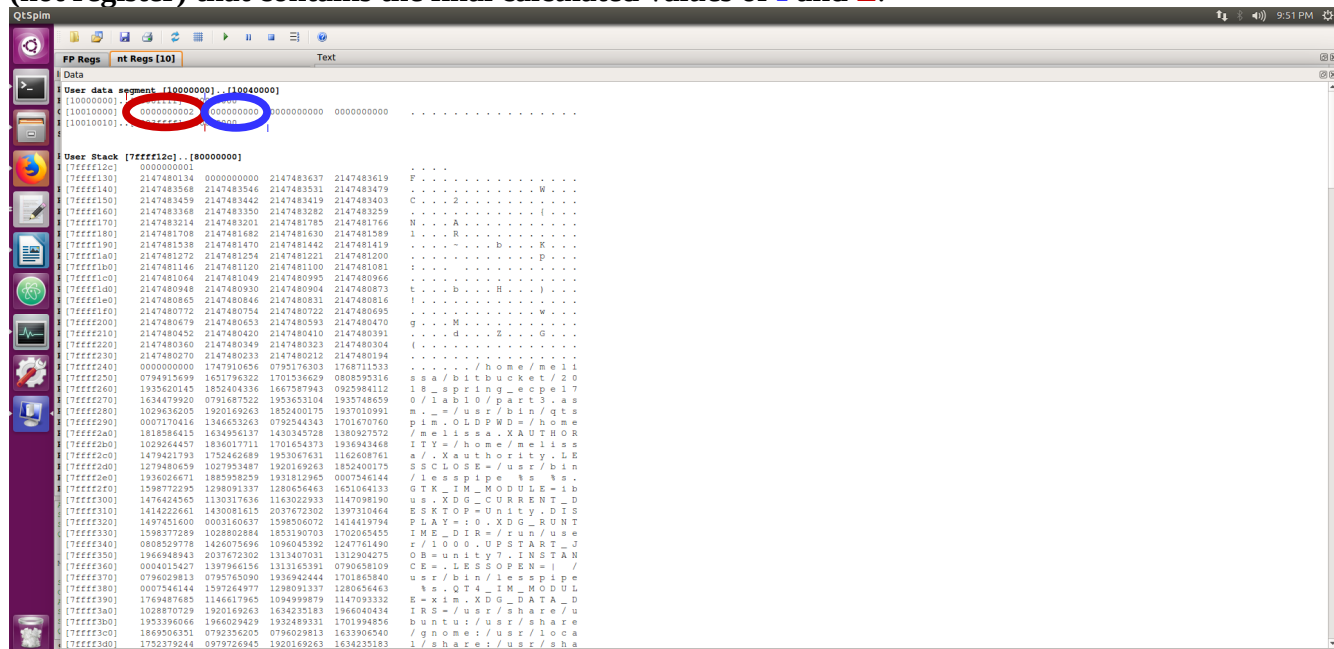
Question #5:

Take a screenshot of the MIPS register panel after your program finishes. Put the register panel in Decimal mode (right-click) so it is easy to see register values.



Question #6:

Take a screenshot of the MIPS memory panel (data tab) after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. **Circle the memory location (not register) that contains the final calculated values of I and Z.**

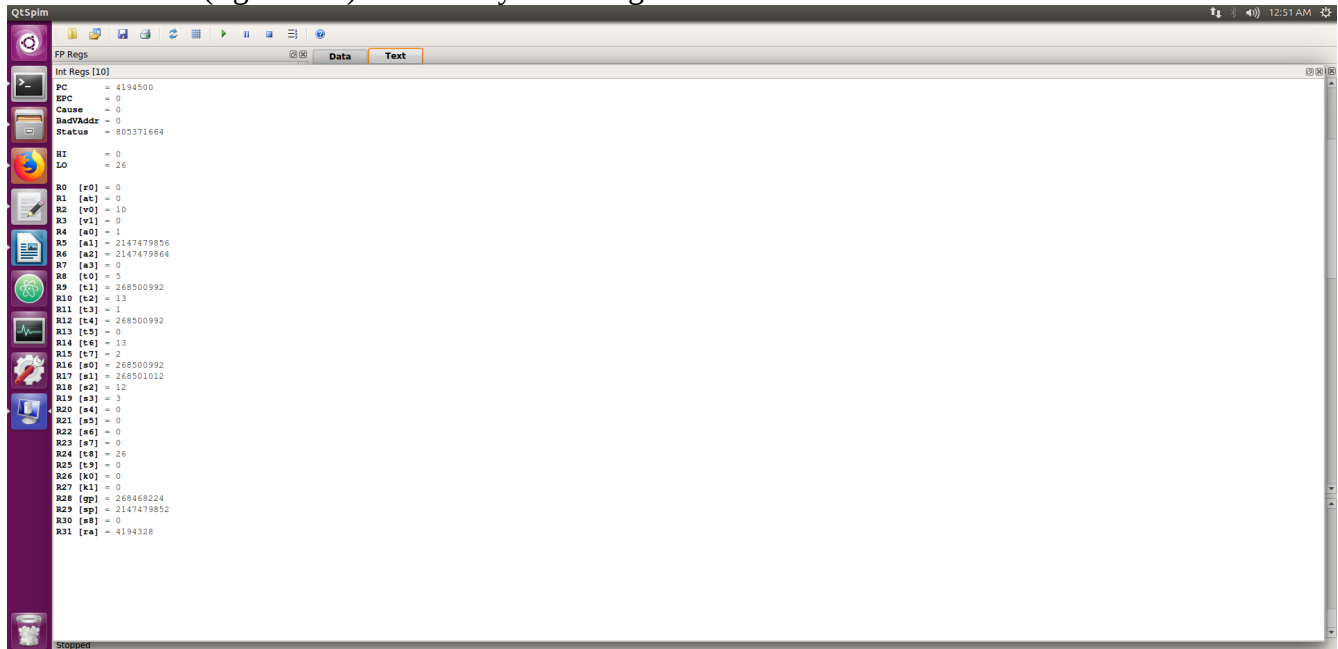


PART 4:

Question #7:

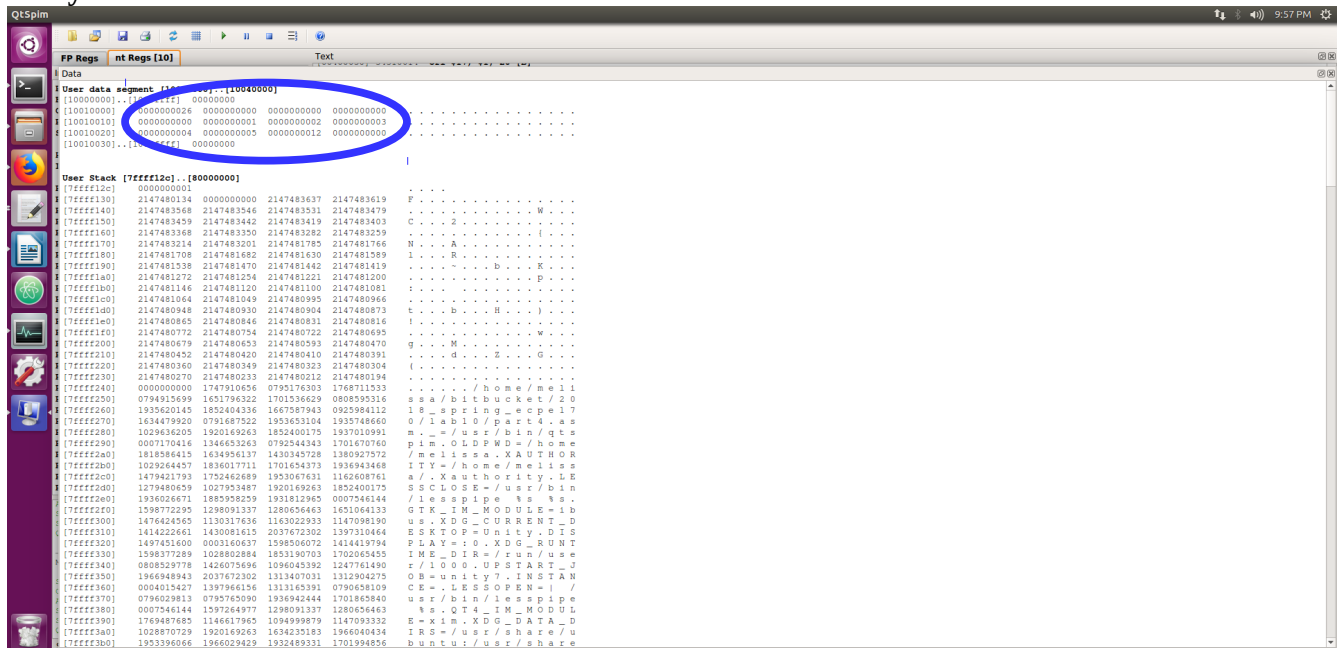
Take a screenshot of the MIPS register panel after your program finishes. Put the register panel in

Decimal mode (right-click) so it is easy to see register values.



Question #8:

Take a screenshot of the MIPS memory panel (data tab) after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. **Circle the final values of array A.**

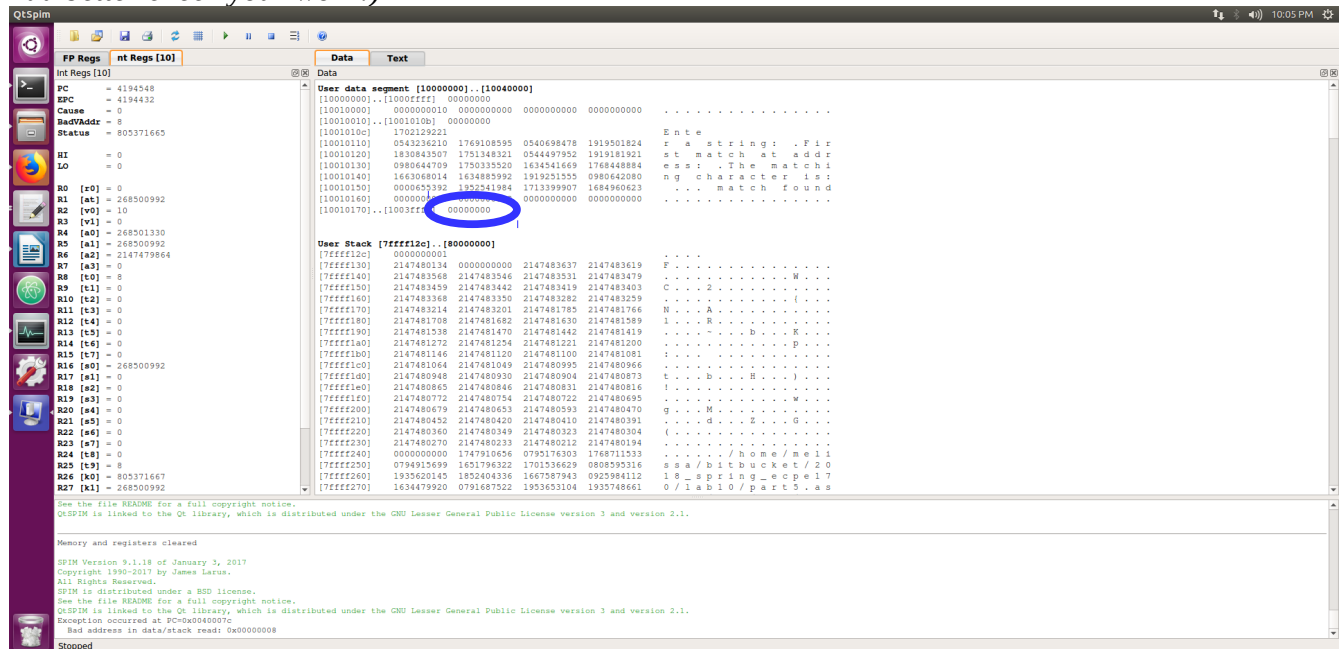


PART 5:

Question #9:

Take a screenshot of the MIPS memory panel (data tab) after your program finishes. Put the memory panel in Hex mode (right-click), since Decimal mode will not allow us to distinguish between bytes.

Circle two things: the final value of the pointer 'result' in memory, and the corresponding location that result points to. Does that location in memory contain the ASCII code for the character 'e'? (*If not, you had better check your work!*)



Yes.

Extra Credit (10 points):

Extend your assembly program to also print out the address of the matching character (i.e. the result variable) as a hexadecimal number in addition to a decimal number. This should be the equivalent of `printf("%X", result);`