Brian Patino
CSE 2304 - Computer Architecture
2016-02-15
Lab 1 Report

Assembly code is very interesting and time consuming programming language. What can be done in a matter of seconds in Java, takes me several dozen minutes in MIPS. On the other hand, every new programming language that I try takes about the same length of time to learn the syntax. This is especially true for a closer to machine level language. So far in my short experience in MIPS, a lot of the lines of code has to do with arithmetic operations even working with just strings. This is something new to me since in java, python, javascript and any other high level language I've used, I don't directly work with arithmetic operations (unless I'm getting substrings etc).

In this lab, I got to work with several basic algebraic operations, input/output prompts, and string displays. Right from the start I began thinking of how to display the requested information to the user. I thought, "since I need some input values X, Y, Z, I should let the user know to please input values X, Y, Z". Thus, the prompts were labeled "Enter value X:", "Enter value Y:", and "Enter value Z:". These input string prompts are written after the *.data* but before *.text*. This is nice and reminiscent of placing all new variables at the top of a method/function. I then placed the body of the code between the *main:* and *exit:* labels. The *exit:* label contains the two lines needed to end the MIPS application which is convenient since I can jump to this at any time.

The only major problems I faced when I first started writing this application was syntax and generally working with bits and bytes. For almost every line of code I had to insert a comment to explain or define what exactly is going on. I usually do this for every new language that I work with since it helps me remember the syntax and how things work. Another problem I faced was how MIPS hands multiplying and dividing integers. At first, I didn't quite understand that MIPS stores numbers less than 32 bits in Lo and higher numbers in Hi, or that when dividing integers, MIPS stores the result in Lo and the remainder in Hi. Now I do.