David J Tinley

11/19/2023

Computer Organization

Lab 5

For the lab 5 assignment we were provided with a starter file that had a good portion of the assembly code written for us. Our main task was to implement a function that reverse the case of a string of characters input by the user and another function to find the minimum ascii value after reversing the case. For the data section pretty much everything was provided for us already.

A screenshot of a computer program

Description automatically generated

The main section of the code was provided for us for the most part. All that I needed to add were the jumps to the functions and moving some of the values to be used in the findMin function. I also added a message to print before exiting to make sure we reached the end of the program.

A screenshot of a computer program

Description automatically generated

For the reverse case function, I first loaded 2 integers into separate registers and then loaded the addresses of the string and the reversed string. Next for the loop, it first loads a character from the user string and checks if it needs to be converted to lower case and jumps to the toLower label where it adds 32 to the value to make it lower case. If the character was already lower case, then it is converted to upper case by subtracting 32 from it’s ascii value. After all that I print the reversed string back to the user. Finally it jumps back to where it was previously in the main function.

A screen shot of a computer

Description automatically generated

For the findMin function, I loaded 2 integers to be used in the find\_min\_loop and then load the addresses of reversed string and the minimum character. The loop for finding the minimum character works by loading a character from the reversed string and comparing it to the previously loaded character. If the newly character has a lower ascii value than the previous than it is set as the new minimum character. After the loop is complete the minimum character is printed back to the user and then jumps back to the previous part where it was in the main function.

A screen shot of a computer

Description automatically generated

Overall, I found this lab to be the most difficult so far. It had the most complex implementation so far by using multiple jumps and loops to get the results that were required. Memory alignment seemed like a very important concept for this lab assignment as well. Finally, here is a screenshot of my compiled and executed program in MARS.

A screenshot of a computer

Description automatically generated