Programming Project Part 1
University of Nevada, Reno
CS 219 - Computer Organization

1. Theory

In computer programming, instructions are broken down from higher-level languages into more machine interpret-able forms. An example of such would be designing a computer program in C, and once you compile the program, it will be compiled into forms the machine can understand.

In simple terms, this will be described as assembly. Assembly language is one step above machine code, or binary. Assembly language commands will follow the format:

operation operand1 operand2

This is the basic form of the operations that will be discussed. The result of the operation between the two operands will be returned to the machine.

HINT: The return of the resulting operation may not always happen behind the scenes.

2. Implementation Details

You are tasked with designing a machine language simulator. This project must be written in C or C++. If you elect to use C++ be careful with your objects. Your code should be versatile enough to be modified with more than one operation for future assignments. For part 1, you must only implement the ADD operation. You will be given a text file for input on this project, following the format discussed in the Theory section.

An example operation will be in the form **ADD 0x12345 0x678** When the operation to be performed is **addition,** it is adding the two numbers **0x12345** and **0x678**. These numbers will be in hexadecimal format. Recall that the hexadecimal prefix is always ”**0x**” This is not to be considered in any conversion process. **You might find it useful to use the uint32t in C/C++ to store the number as an unsigned 32-bit number.**

Using the provided text file as an example input. You will be tasked with creating a program that will **read the command (ADD, but future projects may have more), and it will perform the addition between the two hexadecimal numbers.** HINT: You may find it helpful to store this value in a variable, should you ever need to retrieve the information or store it in a simulated register. But this is not required for part 1.
The result of the addition operation must be displayed to the terminal or build environment. Your code should follow proper code commenting and indentation procedures.

3. Submission Requirements

For submission of this assignment, you are expected to submit your working code at a minimum. **If you elect to use a specific build environment please pick something simple to aid the compilation done by the TA**. **It must be detailed in your README file and include any specific build files required to compile**. All submissions are expected by the due date, and submission is done on canvas.
No extensions will be given except as permitted by the course Syllabus.

AFTER THE DUE TIME, YOU CAN SUBMIT YOUR PROJECT AS LATE SUBMISSION FOR ANOTHER WEEK WITH A 50% PENALTY. BUT AFTER THAT, NO SUBMISSION WILL BE ACCEPTED.
So, you need to submit a zipped file with the following items:

1. A running code (C/C++) with proper documentation on each part.
2. A README file with necessary information on how to run this code.
3. You must describe your result and working process in code as a commented-out form or in your README file.
4. Your code must be executable via the remote computers https://remote.engr.unr.edu/#/
5. Your file should be named with your name as ‘first-name_last-name proj1’.
6. If you have multiple files such as c/cpp and header, provide a make file so that the TAs can compile and execute your project easily. I have provided one on canvas which you can modify.
7. **If your zipped file’s naming doesn’t include your name and if your code is not executable on remote computers, there will be 10 points deduction from your project.**

4 Grading

30 points
30 points will be distributed based on the working code. As detailed in this document.
30 points
30 points will be distributed based on properly documented code and your description of your result and working process. As detailed in this document.
30 points
30 points will be distributed based on whether or not you solved the task you were given.
Did you properly read the commands from the text file?
Did you properly display the result?
Is the result correct?
Are you wrongly calculating values with the prefix?
Please Refer to implementation details.
10 points
10 points will be given for properly handling data types and hexadecimal numbers. Please refer to the Implementation details.