1. Download the Verilog code for our example simple CPU from http://www.swarthmore.edu/NatSci/mzucker1/e15_f2014/SimpleCPU.zip and look for the five new CPU instructions added since class.

   a. What five instructions were added? Describe how each one works in terms of the source register (or immediate data), destination register, and what operation is being performed.

   b. How did the ALU need to get modified to implement these instructions? What is the purpose of the giant nested `:` statement inside the updated ALU module?

2. What binary operation is being performed by `mystery.v` ROM locations 2-7? How, in terms of Boolean Algebra and register assignments, is this binary operation being performed? Note: You can run this program by modifying the `include` directive inside of `E15Process.v`. If you want to view the contents of the registers inside your waveform viewer to determine what’s going on, you will likely want to display the results in binary, not decimal.

3. Write a program called `triangle.v`, similar to `fibonacci.v`, which instead generates the sequence of triangular numbers\(^1\), stored in register 3. Include a printout of your `triangle.v` program along with a printout from your waveform viewer showing a successful run of the program. Please adjust the horizontal scale on your waveform plot to make sure it’s clear what’s going on inside your program.

4. Difficult: Write a program called `rightshift.v`, starting from ROM location 1, to right-shift the value stored in register 0, and which stores the result in register 3 before terminating (via a relative jump of zero). Test your program by using ROM location 0 to store a particular binary value in register zero. For instance, if you store the value `4'b1101` in R0 at the start of your program, when it terminates, the value `4'b0110` should be stored in R3.

   Please hand in a printout of your `rightshift.v` program, along with two printouts of waveform traces demonstrating successful runs of your program on both the input `4'b1101` and the input `4'b1010`.

---

\(^1\)http://en.wikipedia.org/wiki/Triangular_number