E85: Digital Design and Computer Engineering

Lab 11: Multicycle Processor

Objective

In this lab, you will complete a multicycle ARM processor. You must get your processor working to pass the course.

1. Multicycle ARM Processor

Figure 7.30 of the textbook shows the complete multicycle processor. Write a hierarchical Verilog description of the processor. The processor should have the following module declaration. The memory signals are tapped out for testing purposes. Use your controller from Lab 10 and any Verilog building blocks you need from Section 7.6.2.

```
module top(input  logic        clk, reset,
           output logic [31:0] WriteData, Adr,
           output logic        MemWrite);
```

2. Test Bench

The arm_testbench.sv and test code (in assembly and machine language) are on the class web page. Note that the test code has been enhanced somewhat beyond what is in Figure 7.60. Your memory should read the test code from the memory file at startup with the line:

```
initial $readmemh(“memfile.dat”, RAM);
```

Generate simulation waveforms at least for clk, reset, PC, Instr, state, SrcA, SrcB, ALUResult, Adr, WriteData, and MemWrite. Display the 32-bit signals in hexadecimal for ease of reading. You may wish to add other signals to help debug. Fix any problems you may find until your code executes the program as expected and the testbench reports success.

Refer to the previous lab for debugging hints. In particular, it will save you much time to carefully predict what each of the signals in your waveforms should be doing on each cycle, and to systematically debug beginning with the first known discrepancy and working your way backward until you have good inputs and bad outputs and have isolated the bug.

What to Turn In

1. Please indicate how many hours you spent on this lab. This will be helpful for calibrating the workload for next time the course is taught.
2. Hierarchical SystemVerilog for your top-level processor module (and submodules) matching the declaration given above.

3. Simulation waveforms (in this order) at least for the specified signals. Does your system pass your testbench?

If you have suggestions for further improvements of this lab, you're welcome to include them at the end of your lab.