ECE/CS 250 – Recitation #8 – Prof. Sorin
Working Towards a Processor Core

Objective: In this recitation, you will start building components that can be used (with modification) in homework #4 (processor design).

Complete as much of this as you can during recitation. If you run out of time, please complete the rest at home.

1. Task 1: Building a Register File

1) Build a single 16-bit register with a Reset input that sets all 16 bits to zero. You'll use 16 D flip-flops. Make this into a module for reuse.
2) Start building a register file by putting together 16 of the 16-bit register modules you just made.
3) Add a global Reset input to the register file that resets all of the registers to all-zero.
4) Add two read ports to the register file. Each read port consists of a 4-bit input to specify which register you want to read and a 16-bit output (for the 16 bits of the register you're reading).
5) Add one write port to the register file. The write port consists of a 4-bit input to specify which register you want to write and a 16-bit input (for the 16 bits of the register you're writing).
6) Add a global RegWriteEnable input for the register file.
7) Modify register 0 such that it always equals zero (i.e., can't be written).
8) Make this register file into a module for reuse. This register file has the following inputs: Reset (1-bit), RegWriteEnable (1-bit), ReadReg1 (4-bit), ReadReg2 (4-bit), WriteReg (4-bit), WriteData (16-bit). The register file has the following outputs: ReadData1 (16-bit), ReadData2 (16-bit).

2. Task 2: Building an ALU

Create an ALU module. The module has three inputs: OpType (2-bit), InputData1 (16-bit), and InputData2 (16-bit). The module has one output: OutputData (16-bit). Based on the OpType, the ALU computes the desired operation on the inputs and produces the output. The OpType mapping is as follows: 00=Add, 01=Subtract, 10=XOR, 11=NAND. For this recitation (and not on the homework!!), you may use functional units straight from the Logisim library.

3. Task 3: A Simplistic Core-Like Substance

Connect the RegFile and ALU. The two read ports of the RegFile connect to the two inputs of the ALU. The output of the ALU connects to the write port of the RegFile.

Add a clock. On every rising edge of the clock, change the values (in any way you want) of RegRead1, RegRead2, RegWrite, RegWriteEnable, and OpType. Observe what happens.