**Lab 11: Datapath Construction**

**Primary Objectives**

1. Design and implement an 8 x 4 register file

2. Design and implement a function unit

3. Implement a datapath using previous components

*Objective 1 Register File*

The purpose of this lab is to create a datapath for a microprocessor. The first portion of the system to design is an 8 x 4 register file. This portion of the lab was fairly simple, given that all register files follow a template. This meant that I just had to adjust this template to allow for the specifications of this system. Given that it is an 8 x 4 register file, it had to include eight total registers, each of which could hold a four-bit value. After implementing these registers and the simple combinational logic necessary to allow for data writing functionality, I had to allow the register file to output A and B values dependent on given values from the control word. This was done with the use of two 16-to-1 4-bit value multiplexers. Below is what my register file looks like in Logisim.

A diagram of a circuit

Description automatically generated

Register File Implementation

*Objective 2 Function Unit*

For the next portion of the datapath, I had to construct a function unit capable of doing one of 16 designated operations to the supplied data values either from the register file or the outside world. For the design I decided to use a 16-to-1 multiplexer that used an input labeled FS to select which operation to supply as Data Out. The individual operations were extremely simple to implement, since most of them either took data directly from A or B or used a single gate/arithmetic operator supplied by Logisim. In addition to its primary function-select capabilities, the function unit also had to have an output PZN that showed whether the output of the function unit was positive, zero, or negative with corresponding values 100, 010, and 001, respectively. To do this, I used another 16-to-1 multiplexer. Because the microprocessor is going to use all values in two’s complement notation, I supplied the corresponding value (100, 010, 001) based on whether the data value’s MSB was 0 (positive), 1 (negative), or the value was all zeros (zero). Below is the Logisim implementation of my function unit.

A diagram of a computer

Description automatically generated

Function Unit Implementation

*Objective 3 Datapath*

For the final portion of the lab, I had to put my register file and function unit together in a datapath together, with a 16-bit control word controlling the full operation of the datapath based on given specifications. Given that this microprocessor’s datapath functionality was nearly identical to a standard microprocessor’s, the connecting logic could be done with two multiplexers and a few additional outputs and inputs to communicate with the control unit/outside world. After making the connecting logic, I used a 16-bit input as the control word for the datapath, using several splitters to separate the input values into its necessary components to allow the datapath to function. Below is my datapath.

A computer diagram of a computer

Description automatically generated with medium confidence

Datapath Implementation

**Conclusion**

The lab went smoothly. I was able to use the templates provided in the lecture slides to make both the register file and overall datapath design much simpler. The control word makes the datapath function as intended.