Introduction to Computing: Task 3
Fall 2018
Dr. Ramesh Yerraballi
Designed by Matthew Barondeau
The University of Texas at Austin
Due: Friday 11/30, 11:59pm

Purpose

The datapath is the part of computer architecture that implements a program. In this task, you will use Logisim to create a register file, ALU, and shifter. You will then use a bus and tri-state drivers to perform simple operations on registers.

Task Requirements

1. Open the starter project and label the four 8 bit registers R0-R3
2. Implement two read ports and one write port on the register file
3. Test the register file operation
4. Create a logical shifter
5. Test the shifter
6. Create a ripple carry adder
7. Create an 8-bit XOR
8. Create an 8-bit AND
9. Combine the XOR, AND, and ADD into the ALU as described below
10. Test the ALU
11. Attach tri-state drivers to the outputs of the ALU and shifter
12. Create a bus that is connected to the write port of the register file, ALU output, and the shifter output
13. Test operations on registers

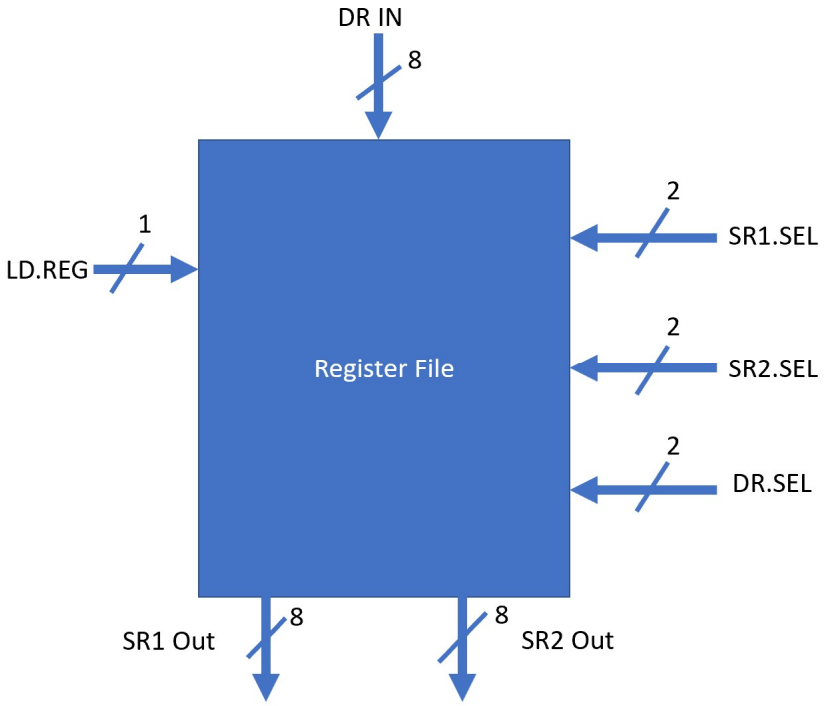
Software Installation

For this lab you will use Logisim, which should be installed on your computer.

If you feel stuck on how to use the program, the tutorial is located here:
http://www.cburch.com/logisim/docs/2.7/en/html/guide/tutorial/index.html

Register File

You will design a register file with 4 registers labeled R0, R1, R2, and R3. These registers will already be created in the starter file for this project. Your job is to implement the interface shown below. Whenever SR1.SEL is a 00, SR1 Out should get whatever the value of R0 is. Whenever SR2.SEL is a 10, SR2 Out should get whatever is in R2. These are two of the possible cases. Ensure that all the output cases work. Whenever DR.SEL is 00 **AND** LD.REG is a 1, R0 will get whatever value is in DR.IN.



Your register file will be tested on the following:

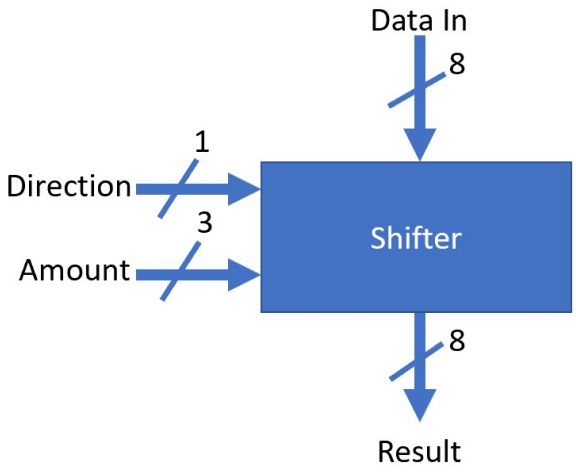
1. The Output of Source 1 and Source 2 are correct based on select signals
2. The Destination Register is modified on positive clock edge
3. The Destination Register is only modified when LD.REG is a 1

Shifter

The shifter you will build has 3 inputs and 1 output. The three inputs are:

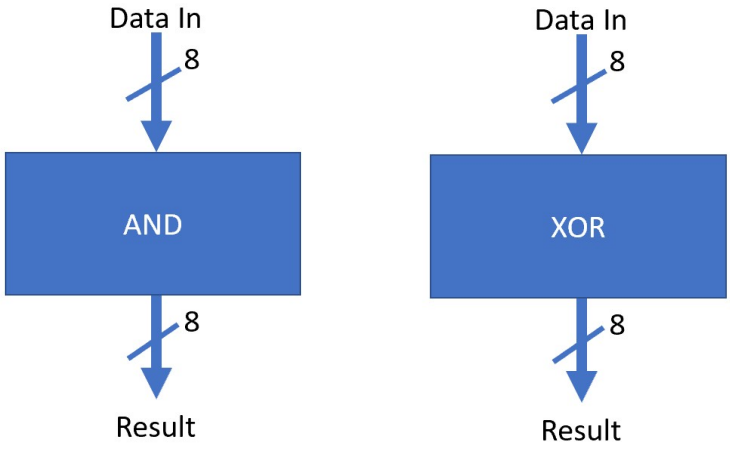
1. Data In – 8-bit value of register prior to shifting
2. Direction – If direction is a 1, shift right. Otherwise shift left.
3. Amount – 3-bit value corresponding to how many positions to shift

The output of the shifter is the 8-bit result. You can use the built-in shift register but ensure that the functionality is as specified above. A block diagram of the shifter is shown below:



XOR/AND

You will create an 8-bit XOR and an 8 bit AND logic block. The logic blocks should perform bitwise AND and a bitwise XOR. Their block diagrams are shown below:



Adder

Your adder will perform addition between two 8-bit numbers. The final carry should remain disconnected. The initial carry in for bit 0 should also be disconnected. Be prepared to know how a ripple carry adder works and any disadvantages.

ALU

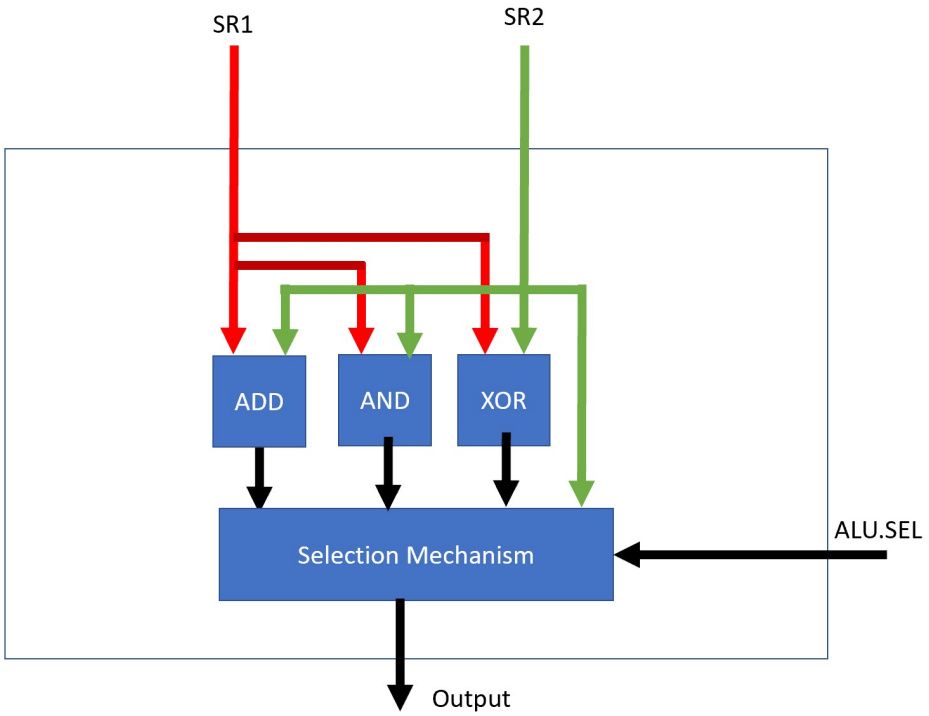
Finally, you will tie together the pieces you made in the past three parts into one ALU. The ALU will be able to perform 4 functions:

1. ADD – Add two 8 bit inputs
2. AND – And two 8-bit inputs
3. XOR – Exclusive or two 8-bit inputs
4. PASS – Pass Source 2 through the ALU unaltered

The operation will be determined by a signal called "ALU.SEL". Only one operation will be performed each cycle. The following encoding for ALU.SEL will be used:

ALU.SEL Value	Operation
00	ADD
01	AND
10	XOR
11	PASS

The block diagram is as follows:



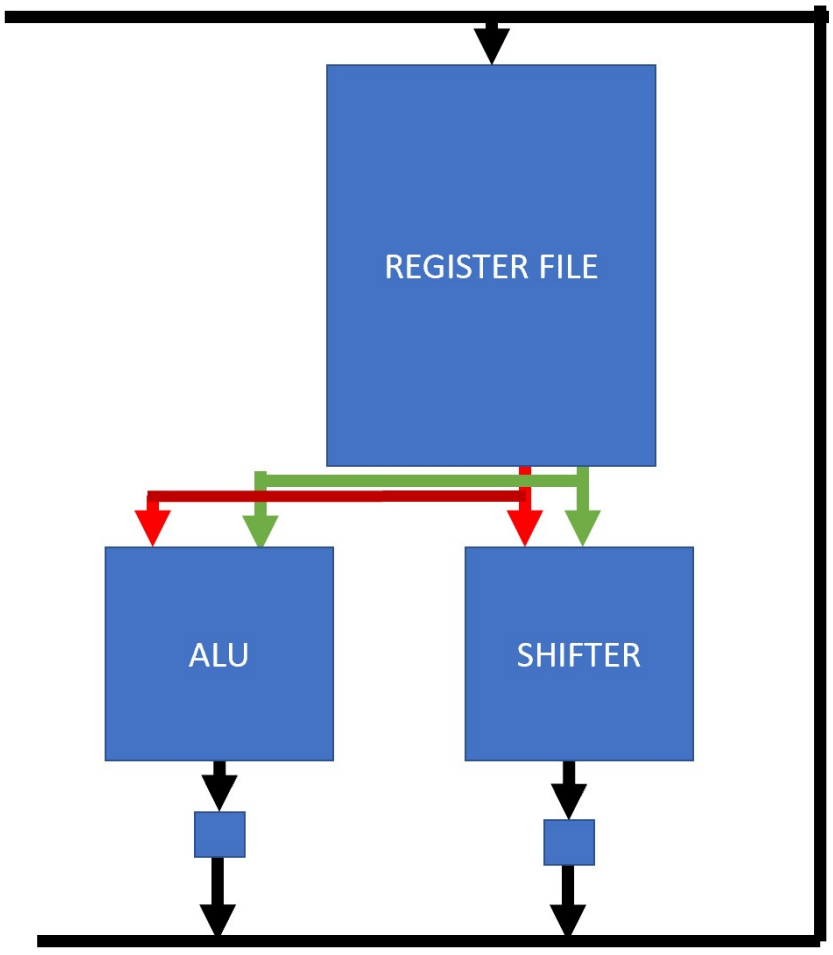
Bus

You will create a bus to feedback the result to the register. This bus will have two inputs to it:

1. ALU Output
2. Shifter Output

Tie one control signal to the tri-state buffers. When BUS.SEL is a 1, allow the ALU on the bus. When BUS.SEL is a 0, allow the shifter output onto the bus.

These inputs will need a tri-state buffer connected between the output and the bus. A tri-state buffer only allows the output to propagate if the enable is a 1. Otherwise the output is not connected, and the other input can write to the bus. Your final system layout should look as follows:



Submission
You will submit 3 snapshots and the final bus system. Name your files “Register_EID”, “Shifter_EID”, “ALU_EID”, and “BUS_EID”. No late submissions will be accepted.

Grading

Item	Points
Working Register File	15
Working Shifter File	15
Working Add	10
Working And	10
Working XOR	10
Working Pass	10
Working Tri-State Buffer	10
Working Full-System	20