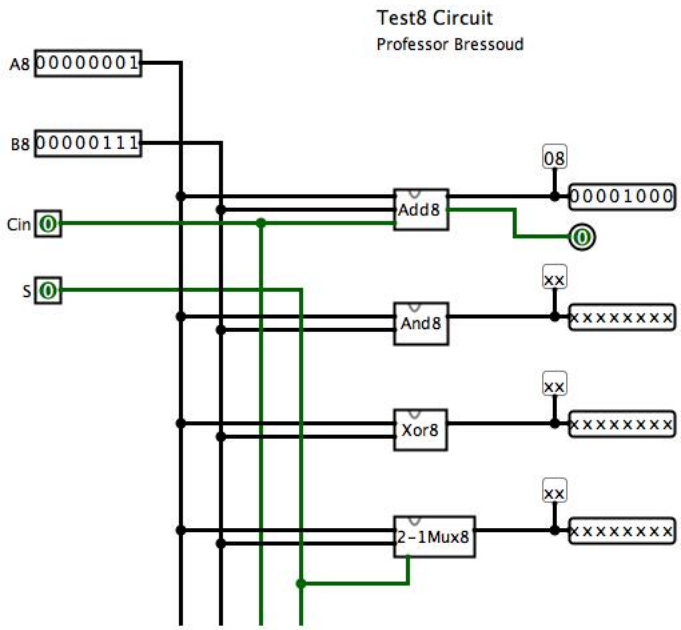
cs281: Introduction to Computer Systems

Project Lab 2, Part I: Logisim and an Arithmetic Logic Unit (ALU) Library

Assigned: Friday, Sept. 25, Due: Wednesday Sept. 30 by midnight

1. The purpose of this project laboratory is to:
   - introduce you to Logisim, a software package for designing and implementing digital logic,
   - reinforce our experience in designing and implementing combinational logic circuits,
   - gain experience in abstraction at the hardware level to prepare us to build a larger whole from smaller parts.
2. This lab is to be completed individually. You will only submit a copy of your Logisim circuit file; there is no accompanying report. You will be given a template `ALUlibrary.circ` file so that you have a framework from which to build your ALU circuit elements, and it will help ensure that the external interface (pin definitions and placement) will be consistent with that expected for our grading of your work.
3. Your goal is to design and build the following circuits in Logisim in the file `ALUlibrary.circ`:
   - **FullAdd**: the full adder circuit, as designed in the Tuesday hardware lab. This circuit has inputs A, B, Cin and has outputs S, Cout. The circuit performs the binary sum of a column of a binary addition, adding the inputs and resulting in a sum and a carry-out.
   - **2-1Mux**: the two-to-one mux circuit, as designed and implemented in the Tuesday hardware lab. This circuit has inputs `A, B, S` and output `Y`. When $S = 0$, the output Y is given by the current value of A, and when $S = 1$, the output is given by the current value of B.
   - **Add8**: an 8-bit wide adder circuit. Utilizing the FullAdd circuit, build a circuit that performs an 8-bit wide addition, adding inputs $A8$, $B8$ and Cin, and resulting in $F8$ and $Cout$, where $A8$, $B8$ and $F8$ are 8-bit wide inputs/outputs and Cin and Cout are each 1 bit wide.
   - **2-1Mux8**: an 8-bit wide two-to-one mux. For this circuit, the inputs are $A8$, $B8$ and S, where $A8$ and $B8$ are each 8 bits wide, while S is 1 bit wide. The output is $Y8$. When $S = 0$, the 8 bits of $Y8$ are determined by the 8 corresponding bits of $A8$ and when $S = 1$, the 8 bits of $Y8$ are determined by the corresponding bits of $B8$.
   - **And8**: an 8-bit wide bit-vector AND. The inputs are $A8$ and $B8$, and the output $F8$ is given by the 8 independent AND operations of A_i with B_i for $i \in 0..7$.
   - **Xor8**: an 8-bit wide bit-vector XOR. The inputs are $A8$ and $B8$, and the output $F8$ is given by the 8 independent XOR operations of A_i with B_i for $i \in 0..7$.
4. In your realization of the above circuits, you are restricted to 2-input AND and OR gates and 1-input NOT gates. You may use splitters, probes, and constant wiring elements as well. This means that the XOR gate is not legal, nor are any of the existing arithmetic and multiplexers already built-in to Logisim.
5. In addition to templates for each of the above individual hardware circuits, I have also provided you with a circuit, Test8, that will allow you to try different inputs to each of your 8-bit-wide circuits and see the results.



The inputs are shown on the left and the outputs are on the right of the circuit. As a matter of convention, we use pins on the left of a chip for general inputs, the bottom of the chip for *control/select* inputs, the right for outputs. Note that $A8$ and $B8$ are 8-bit-wide inputs, 8 and Cin and S are 1-bit-wide input2. Each tested circuit has its own 8 bit wide output.

6. This lab part will be graded based on a total of 35 points. Each circuit is worth 5 points and there are 5 points for the quality, clarity, and neatness of your design. Some of the grading criteria for the 5 points on these design elements:

   - Following conventions of inputs on the left, outputs on the right, and being able to read the "flow" of the circuit from left to right.
   - Neatness – appropriate "clean" routing of wires and placement of sub circuits to convey an organized and orderly design.

Guidance

The guidance given here should help you to order and prioritize your development of the ALU8 circuit. It would be helpful for you to open the `ALU.circ` file and look at the template structure provided as you read this guidance.

1. When you open the template structure, the *current* circuit (the one with the magnifying glass icon) should be `Test8` and the design pane should correspond to the top-level picture above. **You should make NO changes to this circuit.** By extension, this means that you should make no changes to the interface for any of the 8 bit circuits that are used by Test8, because this would then result in needing to change Test8.

2. Notice that, in addition to `Test8`, all of your needed (sub-)circuits have been created. If you double-click any of these circuits to see their contents, you will observe that the only elements in each of these are the input pins and the output pins that define the external interface of the circuit. In Logisim, the placement of the pin devices on the design page determine their order and placement when the device is used. Pins on the top (and facing down) of the design panel are on the top of the device when it is used, and the order of the pins from left to right corresponds to the order on the design page. Likewise for pins on the left, right, and bottom of the design page. So for each of these already-created circuits, you must keep the top/bottom/left/right positioning of the pins as given to you,

but as long as you keep them on the same "edge" and relative position, you can move them to accommodate your circuit design.

3. Before you can build an 8-bit wide adder to fill in the body of the Add8 circuit, you will need to design and build the 1 bit FullAdd circuit. You can use the design steps from the hardware lab to help this process. This is its own circuit and you should test it well before proceeding.

4. Similarly, you will want to implement the 1-bit 2-1 multiplexer and then use it in the 8-bit wide 2-1 multiplexer.