Homework 7

Building the MARIE Architecture: Part 3

Due Date: Before 11:59 p.m. on Thursday, April 14th.

Step 0: Setup

Start with your Logisim circuit from the previous assignment (or you can use my solution). Make a copy and name it `part_3.circ`. Make sure that you understand a few things:

- How to transfer values between registers by changing the bus signals, which I have renamed `Write To ID` and `Read From ID` to remove any ambiguity.
- How to read a value from any location in memory by first writing an address to the MAR, then reading the output from memory.
- How to write a value to memory by first writing an address to the MAR, then writing to memory.
- How to load values in the AC and MBR, then add or substract them using the ALU.

If you're confused about any of these, talk to me ASAP.

Step 1: PC

The program counter is a normal 12-bit register with one slight difference: its output can either be its contents (like a normal register), or it can be the value of the contents + 1. Consider, during the fetch phase, we need to implement the following RTL instruction:

```
PC ← PC + 1
```

This means we need to be able to get (PC + 1) without using the ALU. However, for the `JnS` instruction, we need this RTL instruction:

```
MBR ← PC
```

So, we need to be able to PC *or* (PC + 1) from the PC. Create a subcircuit called `PC`. It should have the following inputs and outputs:

- A 12-bit `D` input.
- A 12-bit `Q` output.
- A 1-bit `en` input. When 0, the PC ignores the clock; otherwise, it updates the stored value.
- A 1-bit `clock` input.
- A 1-bit `inc` input. When 0, the `Q` output is the last stored value; when 1, the `Q` output is the stored value increased by 1.

You can see that it looks very similar to a normal 12-bit register. To build it, you don't need anything more than a 12-bit register, an adder, and a mux.

Once you've built the PC, add it to the main circuit and connect it to the bus. Add a "PC+1" input to your main circuit and connect it to the PC `inc` input. The "PC+1", "Write to ID", "Read from ID", and "ALU Func" inputs are the control signals you will use to control exeuction of the MARIE CPU.

Step 2: Instruction, Input, and Output Registers

Simply connect normal 16-bit registers to the bus for these three components.

Step 3: Connecting the ALU to the AC

Now you can delete the extra "ALU Output" pin from your main circuit, and connect the output of the ALU to the AC.

The trick here is that the AC can read from either the bus, *or* the ALU. Whenever the ALU is producing an output, the AC should store it. Put another way, the AC should update its value if:

- The ALU function is anything other than "do nothing", or 00. If this is the case, the AC should store the ALU output.
- If the ALU function control is 00, then the AC should obey the `busif` signals as usual.

Add logic to your main circuit to effect this behavior.

Step 4: Testing

At this point, the datapath for MARIE is complete. You should be able to manually perform any of the instructions in Table 4.7 of the MARIE pdf (p. 38 of the pdf, labeled p. 232).

Turning it in

Make a folder labeled "Homework 7" in your turn-in folder and copy in your `part3_circ`.