Homework 10: Simulated Control Circuit

Your assignment is to use Logisim make a rudimentary Control unit for a MIPS CPU that parses a 32-bit instruction in machine language. The file you turn in will be `control.circ`.

It will have a single 32-bit input on the west, which will be the MIPS instruction to be parsed. You need to implement the following instructions:

- `add` - `sub` - `addi` - `ori` - `lw` - `sw` - `beq` - `j`

Your control circuit will need the following outputs:

- **Three mutually-exclusive bits indicating if the instruction is *R*-, *I*-, or *J-type*.** These will be on the north side of the circuit, in this order.
- **The *opcode*, *rs*, *rt*, *rd*, *shamt*, *funct*, *immediate*, and *address* fields.** Output all of these for every instruction, regardless of its type. (So for example J-type instructions will have an output on *rs* using bits 21-25, even though it's meaningless.) These will be on the south side of the circuit (left to right), in the order indicated here.
- **The control outputs *RegWrite*, *RegDst*, *ALUSrc*, *Branch*, *Jump*, *MemRead*, *MemWrite*, *MemtoReg*, and *ALUInput*.** All of these are single bits except ALUInput, which is 4 bits long.
  - *RegWrite* is true when the instruction writes to a register.
  - *RegDst* is true when the register being written to is determined by bits 11-15 of the instruction, and false when it's determined by bits 16-20.
  - *ALUSrc* is true when the second ALU operand comes from an immediate value.
  - *Branch* is true when the instruction is a branch instruction (but not a jump).
  - *Jump* is true when the instruction is a jump instruction (not a branch).
  - *MemRead* is true when main memory must be read.
  - *MemWrite* is true when main memory must be written.
  - *MemtoReg* is true when a memory read needs to be directed back to a register.
  - *ALUInput* defines the operation to be given to the ALU's control as shown here:

ALU Con	Op	ALU Con	Op	ALU Con	Op
0000	and	0010	addition	0111	set-on-less-than
0001	or	0110	subtraction	1100	nor

  Note that the output in some cases won't matter (e.g. when *RegWrite* is false, *RegDst*'s value doesn't matter.) These outputs will be on the east side of the circuit, in the given order (top to bottom).

It may be easier to separate these three types of outputs into different subcircuits, to keep the design modular. As before, be sure to use the "dummy" template that is provided. (It uses a clock to cycle through outputs.)