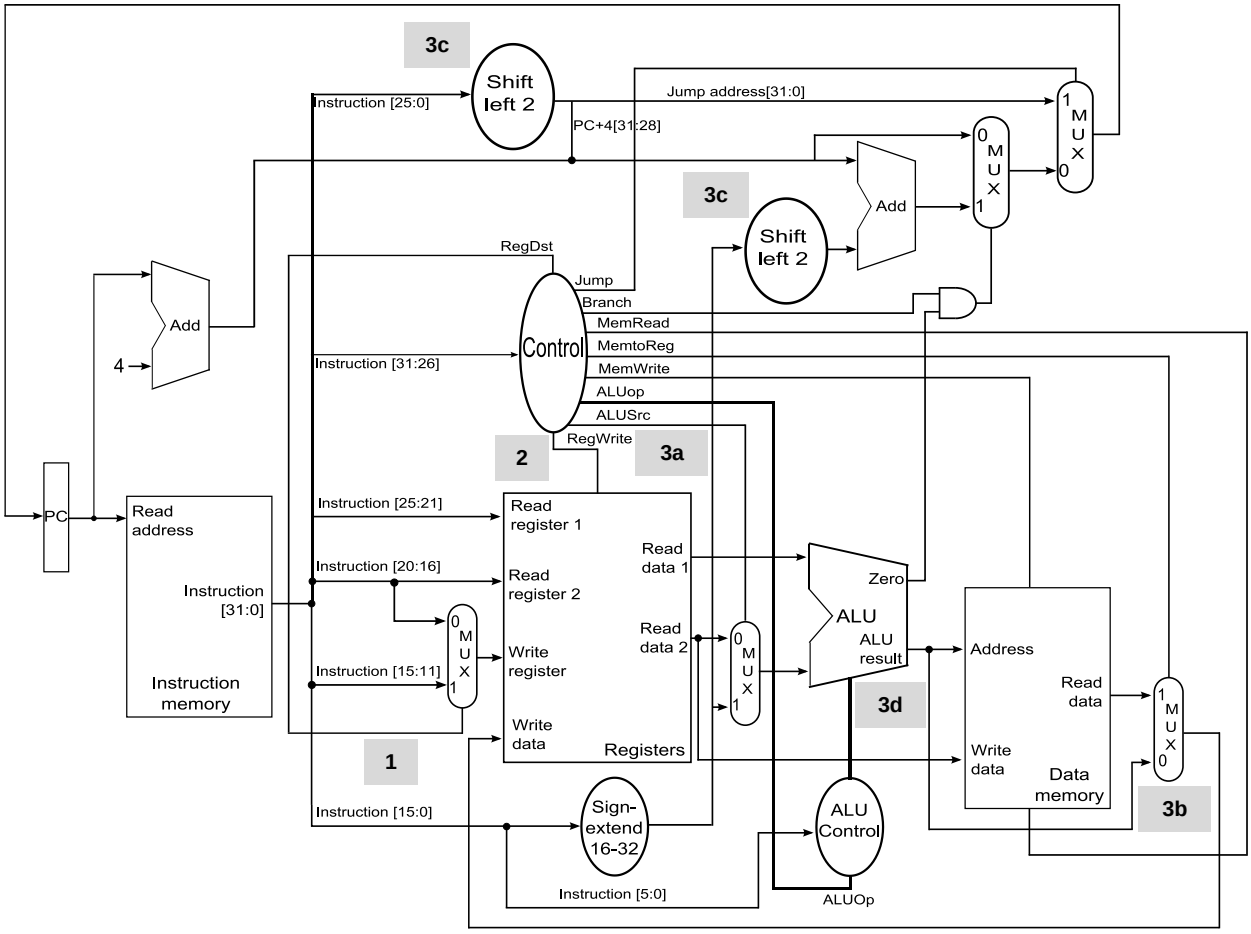
You may work in pairs for this assignment. If you choose to work with a partner, make sure only one of you submits a solution, and you paste a copy of the Partners Template that contains the names and PIDs of both students at the beginning of the file.

Prepare your answers to the following questions in a plain text file. Submit your file to the Curator system by the posted deadline for this assignment. No late submissions will be accepted.

You will submit your answers to the Curator System (www.cs.vt.edu/curator) under the heading `MIPS01`.

Questions 1 through 5 refer to the completed single-cycle datapath design, reproduced below, which supports execution of the following MIPS instructions: `add`, `sub`, `and`, `or`, `slt`, `lw`, `sw`, `beq` and `j`.



1. Suppose that, due to a manufacturing defect, the RegDst control signal suffers a *stuck-at-0 error*. That is, the RegDst signal is always set to 0 regardless of circumstances. Assume that the rest of the hardware operates as designed.

a) [12 points] Would any of the supported instructions still always execute correctly? If so, which one(s) and why?

b) [12 points] Would any of the other supported instructions possibly execute correctly (i.e., produce all the logically correct results and not produce any unintended side effects)? If so, which one(s) and under what circumstances?

2. Suppose that, due to a manufacturing defect, the RegWrite control signal suffers a *stuck-at-1 error*. That is, the RegWrite line is always set to 1 regardless of circumstances. Assume that the rest of the hardware operates as designed.

a) [12 points] Would any of the supported instructions still always execute correctly? If so, which one(s) and why?

b) [12 points] Would any of the other supported instructions possibly execute correctly (i.e., produce all the logically correct results and not produce any unintended side effects)? If so, which one(s) and under what circumstances?

3. [10 points] The above single-cycle datapath (SCD) includes two Shift left 2 units: one for `beq` instruction, another for `j` instruction. An alternative design could have a single Shift left 2 unit and use a MUX to choose an input to the shifter. Discuss why this alternative design is less efficient than the original design with two Shift left 2 units

4. A few easy ones about the SCD. Just state your conclusion; explanations will not be considered.

a) [6 points] Among the supported I-type instructions, which instruction(s) requires the ALUSrc control signal to be set to 0?

b) [6 points] The MUX to the right of the Data memory unit is necessary because of a difference between the R-format machine instructions and what other (single) supported machine instruction?

c) [6 points] For `slt $t0, $t1, $t2` instruction in an assembly format, what should be the value of ALU result (an output of the ALU unit) if `$t1` is 1 and `$t2` is 2.

5. Suppose we want to support two additional branch instructions: `bne` (Branch on Not Equal) and `blt` (Branch on Less Than). The `bne` instruction compares 2 registers and takes a branch if two registers are not equal; and the `blt` instruction compares 2 registers and takes a branch if one register is less than another.

a) [12 points] Describe one specific way to extend the above SCD to support `bne` instruction. You may propose to extend hardware and have additional control signals and/or logic gates (if needed).

b) [12 points] Suppose we have the SCD design that supports `bne` instruction (as proposed in 4a)). Describe how to support `blt` instruction without additional hardware modification? (Hint: think about how pseudo-instructions work)