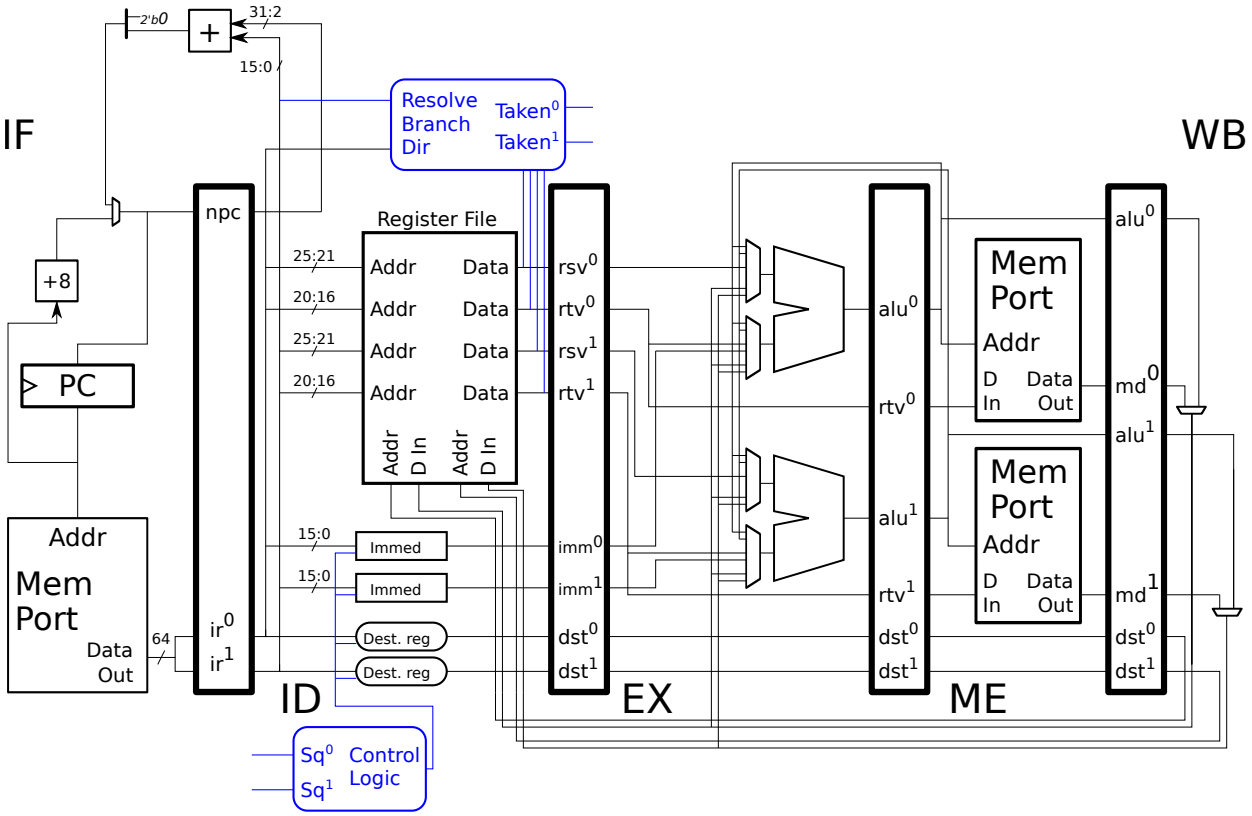
LSU EE 4720 **Homework 8** **Due: 11 April 2019**

Problem 1: The following problem is an enhanced version of 2018 Final Exam Problem 1 (c). Appearing below is our 2-way superscalar MIPS with ID-stage hardware to determine branch direction (near the top in blue) and ID-stage hardware to squash instructions (near the bottom in blue). The Inkscape SVG source for this image can be found at `https://www.ece.lsu.edu/ee4720/2019/hw08-ss.svg`.

There are two outputs of the branch direction hardware logic, indicating whether the respective ID-stage slot has a taken branch. For example, if `Taken0` is `1` then there is a branch in slot 0 and that branch is taken. Of course, assume that this logic is correct.

There is a squash logic with two inputs at the bottom. If input `Sq0` is 1 then the instruction in `ID`-stage slot 0 will be squashed, likewise for `Sq1`.

In this implementation fetch groups are not aligned.



(*a*) When a branch is taken we may need to squash one or two instructions (the number of instructions to squash depends on the whether the branch is in slot 0 or slot 1). Design logic to set the `Sq0` and `Sq1` inputs so that appropriate instructions are squashed. It will be very helpful to draw pipeline execution diagrams showing a taken branch in slot 0 and slot 1.

- ☐ Draw PEDs for the two cases.
- ☐ Add hardware to set `SQ` signals.

(*b*) Notice that the branch hardware shown can only provide the target for a branch in slot 1. Add hardware for providing the branch target of a branch in slot 0. Note that unlike the final exam, in this problem fetches are not aligned. That precludes the more efficient solution given in the final exam.

Do not add hardware for checking the branch condition. Show logic computing the select signals for any multiplexors you add, but do not show any other control logic. *Note: In the original assignment the direction to show logic computing select signals was omitted.*

- ☐ Add hardware for a slot-0 branch.
- ☐ **Pay attention to cost**.
- ☐ Be sure the hardware computes the correct target address. Think about the value of `NPC` (or related value) that's needed.