LSU EE 4720 Homework 4 Due: 4 December 2006

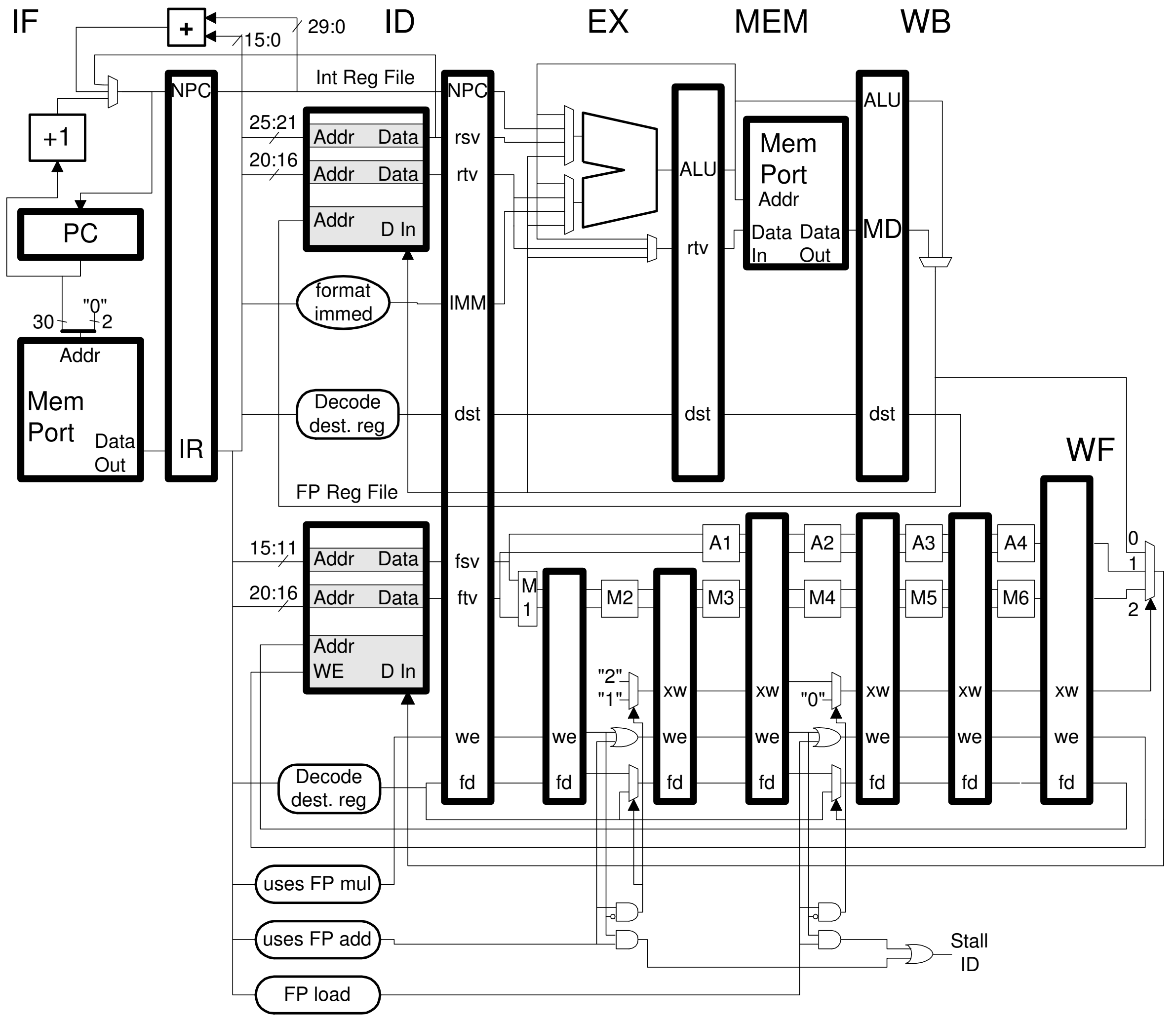
Problem 1: The floating point pipeline in the MIPS implementation illustrated below must sometimes stall instructions to avoid the WF structural hazard. The WF structural hazard could be avoided by requiring all instructions that use WF to go through the same number of stages. Note that instructions that use WB all pass through five stages, even though some instructions, such as `xor`, could write back earlier.

Redesign the illustrated implementation so that the WF structural hazard is eliminated by having WF instructions (consider `add.d`, `sub.d`, `mul.d`, and `lwc1`) all pass through the same number of stages. The functional units themselves shouldn't change (still six multiply steps and four add steps) but their positions might change.

(*a*) Show the possibly relocated functional units and their connections. Don't forget connections for the `lwc1` instruction.

(*b*) Show any changes to the logic generating the `fd`, `we`, and `xw` signals. *Note: The original assignment did not ask for* `xw` *changes.*

(*c*) Show bypass paths needed to avoid stalls between any pair of floating point instructions mentioned above.



Problem 2: Consider the changes to avoid structural hazard stalls from the previous problem. Provide an argument, either for making the changes and or against making the changes. For your argument use whatever cost and performance estimates can be made from the previous problem. Add to that the results of fictitious code analysis experiments and alternative ways of using silicon area to improve performance.

The code analysis experiments might look at the dynamic instruction stream of selected programs. For these experiments explain what programs were used and what you looked for in the instruction stream. Make up results to bolster your argument.

For the alternative ways of using silicon area, consider other ways of avoiding the structural hazard stalls, or other ways of improving performance. This does not have to be very detailed, but it must be specific. (For example, "use the silicon area for pipeline improvement" is too vague.)

The argument should be about a page and built on a few specific elements, rather than meandering long-winded generalities.

Problem 3: In the previous problem structural hazards were avoided by having all WF instructions pass through the same number of stages. If both WB and WF instructions passed through the same number of stages then, were it not for stores, it would *easily* be possible for floating-point instructions to raise precise exceptions without added stalls (even if exceptions could not be detected until `M6`).

(*a*) For this part, ignore store instructions. Explain why having all instructions pass through the same number of stages makes it easier to implement precise exceptions (without added stalls, etc.) for floating point instructions.

(*b*) For this part, include store instructions. Explain how store instructions preclude precise exceptions for the implementation outlined above, or at least for a simple one.

(*c*) For this part, include store instructions. Do something about stores so that the all-instructions-use-the-same-number-of-stages implementation can provide precise exceptions to floating point instructions. It is okay if the modified implementation adds stalls around loads and stores. A good solution balances cost with performance.

If your solution is costly say so and justify it. If your solution is low cost but lowers performance say so and show the execution of code samples that encounter stalls.