HW#2: Pipelining and Data Dependencies

Start Assignment

Due Friday by 11:59pm **Points** 100 **Submitting** a text entry box or a file upload **File Types** doc, pdf, and txt

Assignment 2

Note, read the following carefully:

All homeworks will have to be turned in electronically on Canvas.

Plan accordingly and don't leave your submission until the last minute.

Make reasonable assumptions where necessary and clearly state them.

Feel free to discuss problems with classmates, but the only written material that you may consult while writing your solutions are the textbook, lecture notes, and lecture slides.

I am allowing you to submit HWs in groups of two at the most. You can self-assemble in groups of two through Canvas.

You will be able to review other people's solutions.

Grading Policy: I am not grading these for correctness to the decimal point but I am more interested in seeing that you understand the concepts and can apply what you have learned. Therefore, grading here will be more lenient than in Exams. But the problems are exam quality problems. I do not provide solutions to any homework but I am more than happy to solve whichever ones you want in class if asked to. The discussion and peer-review of your fellow students' solutions is part of the requirements and part of what will get graded. You actually learn from your peers sometimes more than from your instructors. Check the problems you are not sure about. Most probably, one of your peers has done it right. Or maybe you can point out that someone did something wrong. Point it out, politely.

1. **Pipelining (40 points)**

   An unpipelined processor takes 10 ns to work on one instruction. It then takes 0.2 ns to latch its results into latches. I was able to convert the circuits into 10 sequential pipeline stages. The stages have the following lengths: 1.3ns; 0.7ns; 0.8ns; 1.2ns; 0.7ns; 1.4ns; 0.7ns; 0.4ns; 1.5ns; 1.3ns. Answer the following, assuming that there are no stalls in the pipeline.

   1. What are the cycle times in both processors?
   2. What are the clock speeds in both processors?
   3. What are the IPCs in both processors?
   4. How long does it take to finish one instruction in both processors (in nano-seconds and cycles)?
   5. What is the speedup provided by the 10-stage pipeline?
   6. If I was able to build a magical 1000-stage pipeline, where each stage took an equal amount of time, what speedup would I get?

2. **Data Dependences (60 points)**

   Consider a 32-bit in-order pipeline that has the following stages. Note the many differences from the examples in class: a stage that converts CISC instructions to micro-ops, one stage to do register reads, one stage to do register writes, three stages to access the data memory, and 4 stages for the FP-ALU. For the questions below, assume that each CISC instruction is simple and is converted to a single micro-op.

Fetch	Convert to micro-ops	Decode	Regread	IntALU	Regwrite			
				IntALU	Datamem1	Datamem2	Datamem3	Regwrite
				FPALU1	FPALU2	FPALU3	FPALU4	Regwrite

   After instruction fetch, the instruction goes through the micro-op conversion stage, a Decode stage where dependences are analyzed, and a Regread stage where input operands are read from the register file. After this, an instruction takes one of three possible paths. Int-adds go through the stages labeled "IntALU" and "Regwrite". Loads/stores go through the stages labeled "IntALU", "Datamem1", "Datamem2", "Datamem3", and "Regwrite". FP-adds go through the stages labeled "FPALU1", "FPALU2", "FPALU3", "FPALU4", and "Regwrite". Assume that the register file has an infinite number of write ports so stalls are never introduced because of structural hazards. How many stall cycles are introduced between the following pairs of successive instructions (i) for a processor with no register bypassing and (ii) for a processor with full bypassing?

   1. Int-add, followed by a dependent Int-add
   2. FP-add, followed by a dependent FP-add
   3. Load, providing the address for a store
   4. Load, providing the data for a store
   5. FP-add, providing the data for a store