ECE 252 / CPS 220 Homework #2
Due in class on Monday, September 22nd
Total Points 100

Explain all of your answers to get full credit! You may NOT work with other people on this assignment. I will not tolerate academic misconduct.

Pipelining (Appendix A)

1) (10 points) H&P A.5
2) (10 points) H&P A.7
3) (10 points) To avoid a hypothetical structural hazard at the register file, we can add another port to it. Adding a port, though, slows it down. Assume the added port leads to a 5% cycle time increase. Assume the pipeline can complete one instruction per cycle unless there is a structural hazard, hazards occur for 1 out of every 20 instructions, and hazards incur a 1-cycle stall penalty. Is adding a register port a good idea or not?

Superscalar and Dynamic ILP (Chapter 3)

4) (10 points) H&P 3.1
5) (10 points) H&P 3.3

Pipelining and Dynamic Scheduling with SimpleScalar

6) (50 points total) Start with the sim-DLX simulator. sim-DLX extends sim-func by adding simple timing along with branch prediction, caches, and some other fun stuff. The main loop of the simulator (the sim_sample function in sim-DLX.c) executes each instruction and increments the cycle counter by one or more (if there was a cache miss or branch misprediction). Note 1: sim-DLX does NOT model any specific pipeline depth. Note 2: sim-DLX actually executes the instruction at the beginning of the loop and then figures out its impact on the timing. This is counter-intuitive, but it makes SimpleScalar go faster. Assume we’ll be using sim-DLX to model (the timing of) a 10-stage pipeline. Now run the following two experiments (which will involve modifying the code) for each of the three working benchmarks, and do not change the default configuration of sim-DLX (i.e., just run it without adding flags to change the cache configurations, etc.). In addition to the results you’ll turn in, you must email me sim-DLX.c before class on Sept 22.

(a) Evaluate the importance of early resolution of branch mispredictions (and prediction accuracy) in deep pipelines: compare the performance of a pipeline with a 2-cycle misprediction latency vs. a 5-cycle misprediction latency.

(b) Evaluate the importance of memory disambiguation (i.e., figuring out that no older stores will write the address of a younger load): compare the performance of a pipeline with perfect disambiguation (you don’t have to change sim-DLX to model this) to a system where a load must stall in stage 5 until there are no older stores in the pipe (you must change sim-DLX to model this). Hint: you will need to create a data structure to keep track of what kinds of instructions are still in the pipeline. Note that this experiment is slightly simplified for this homework, since a load really only cares if there are older stores that could write to the same address as the load.