ECE 252 / CPS 220 Homework #1

Due in class on Monday, Sept 7

(electronic submission of Question 5 due at 10:00am on Sept 7)

70 points

You must work with one other student on this assignment. Please turn in one assignment with both of your names on it. The idea is NOT for each of you to do half of the assignment—that only creates a situation in which each of you knows half of the material. The idea is for you to work together on all of the assignment. I can't enforce this, but I can strongly encourage you to do the right thing.

Note: Copying text from Wikipedia or other online sources will NOT be tolerated. I have discovered this form of plagiarism in the past, and the consequences are severe.

Performance Analysis (Chapter 1)

1) (10 points) H&P 1.14

2) (10 points) You have been named the chief architect of a next-generation gaming system. Which processor do you choose? Your choices are: the triple-core IBM Xenon processor (used in the Xbox 360), the Cell Broadband Engine (used in the PlayStation3), or the not-yet-released Intel Larrabee chip (assume it will have 40 cores). You must explain your answer, and you must provide citations for any data (or other material) you use in your argument.

3) (10 points) Little's Law: Assume (with no basis in reality) that YouTube can process requests to view videos at a steady state rate of 2000 requests per second. Each request takes, on average, 0.1 seconds to process. On average, how many requests are being processed at any given instant? Given your result, if you were sizing the system, how many requests would you size it to be processing at once (be careful here - simple algebra will not solve this question)? Explain your answers.

4) (10 points) You have two different computers, Computer A and Computer B. They are identical (same ISA, caches, memory, I/O) except for their CPU microarchitectures. They run the exact same binary program. Computer A consumes more power but less energy than Computer B while running this program. Explain how this is possible.

Learning how to use the SimpleScalar simulator

5) (30 points) SimpleScalar is a set of simulator tools that we will use throughout the semester to study computer architectures. The SimpleScalar toolset (see `www.simplescalar.com` for more information), which is written in C, is used widely in research for evaluating microarchitectural ideas, and it includes several types of simulators. These

simulators trade off speed of simulation versus modeling detail. They can all simulate several ISAs, but in this class we will only use them to simulate the Alpha ISA (used by DEC and then Compaq).

On an x86/Linux machine supported by either Duke OIT[1], Computer Science[2], or ECE[3], create a working directory. Copy the files `instruct-progs.tar.gz` and `simplesim-3v0d.tgz` from `~sorin/ece252-public/` (this directory exists on OIT, CS, and ECE machines) to your working directory. For both of these files, gunzip (`gunzip file.tar.gz`) and then untar (`tar -xvf file.tar`) them. Move to the newly created `simplesim-3.0` directory and then build the purely functional simulator, `sim-safe`, by typing `make config-alpha; make sim-safe`. You may or may not see many instances of the following warning (or similar): "warning: passing argument 2 of '_panic' discards qualifiers from pointer target type". Don't worry about these warnings—they're normal.

Now you are ready to run the benchmarks that are in the newly created `benchmarks` directory. Follow the instructions for running 3 out of the 4 benchmarks (all but `compress`) that are in `benchmarks/README`. The target is "alpha", since we are simulating the Alpha ISA. To make sure everything is running correctly, here are the instruction counts for `go` (545811989), `gcc` (337360339), and `anagram` (25595137). It is possible your instruction counts will be slightly different (less than 1% different). This is OK. Also, do not worry if the output is not the same as the reference outputs—this is also OK.

Experiment: For each of these three benchmarks, evaluate the following design idea. Assume that you have a 3GHz processor whose clock rate is bottlenecked by the time to access the L1 data cache for loads and stores), and that all instructions currently take 1 cycle (there is no pipelining and no parallelism of any kind). We could increase the clock rate to 3.5 GHz if we let loads and stores take 2 cycles each (but all other instructions are still 1 cycle). Is this a good idea? Show your math!

To evaluate this idea, you must modify `sim-safe` to count loads and stores of all types. (You do NOT need to modify the simulator to change latencies.) When you're ready to submit your modified `sim-safe.c`, rename it to `homework1.c` and submit it by directing your web browser to `https://www.ee.duke.edu/sorin-upload/`. Upload your file called `homework1.c`.

Note: sim-safe reports "sim_elapsed_time" when it is done. This is a measure of how long the simulation took to run, **NOT** how long it would take the simulated machine to run the benchmark. This is a *very* important distinction.

---

1. For OIT, these machines include teer{1-45}.oit.duke.edu and hudson{1-21}.oit.duke.edu. Please refer to the following website for more info: http://www.oit.duke.edu/comp-print/labs/locations/teer.html

2. For CS, these machines are all aliased to linux.cs.duke.edu. If you are a CS graduate student with an x86/Linux desktop, please use that machine. If you are a CS graduate student with a SPARC/Solaris desktop, you may try to use that machine, although the portability of SimpleScalar on certain versions of Solaris is sketchy at best. The TA and I will *not* support SimpleScalar on Solaris.

3. For ECE, these machines include dsil{1-14}.