California Institute of Technology
Department of Computer Science
Computer Architecture

CS184b, Winter 2001   **Final**   Sunday, March 11

Due: Friday, March 16, 5:00PM

This does have a somewhat different flavor than last quarter's final, but should invoke the same kind of thinking.

My goal here is not to trick you up but to have you think about how to put together and exercise the concepts we have discussed.

No time limit. Open book and notes. Work alone.

Points are given in parentheses at the beginning of each (sub)problem.

1. (5) Increasingly we see a rising semantic gap between the architecture model presented to the software or user and the hardware implementation (*e.g.* sequential ISA versus parallel, out-of-order execution). Outside of computing, detail another example of this phenomena.

2. (20) Consider adding native loop support to a new ISA. Sketch the implementation (instructions, datapath to support, etc.). Outline the potential benefits of these instructions and the support facilities versus the conventional case where we just have branches and comparisons to implement looping constructs. Identify any issues which may hinder scaling and usability. How can compilers exploit this addition? What could prevent the compiler from being able to use these instructions?

3. (20) Detail the correct behavior of a write buffer system when the main memory is banked (I want to see a full description of how the write buffer needs to behave, including a datapath diagram). What issues do you have to worry about for correctness? Sketch what implementation parameters should impact the proper depth of the write buffer and how to minimize implementation costs (area, delay) while maximizing performance. Remember, the goal of the write buffer is to allow computation to continue without waiting for the latency of the write to complete to main memory.

4. Standard techniques for increasing processor performance ignore energy. In what ways do typical optimizations tend to help reduce energy? What implementation details would go differently if your primary goal was energy reduction rather than top performance? Give a set of first order equations that might serve as a starting point to quantify the energy/performance tradeoffs (these will be like the CPU performance and cache equations we saw in class; some parameters in these equations might be something you would need to measure from code).

   - (13) memory system (caching)
   - (12) control flow (branch acceleration and predicated instructions)

5. The number of registers in the register file is one of the few places where a fixed constant shows up in the architecture. Despite our appreciation of the pitfalls of exposing fixed constants, even modern architectures like EPIC retain a visible, finite register set size.

   - (5) How does a fixed number of registers limit parallelism and hence performance scaling?
   - (5) Identify the negative effects of having a large number of registers.
   - (20) Propose a scheme to eliminate registers from the ISA model.
     1. Describe the new model.
     2. Sketch the hardware mechanisms/support necessary to do this and achieve comparable or better performance (compared to the traditional alternative) from your design.
     3. Critique your design — what are the potential drawbacks of the design? what are the critical things it would be necessary to pull of (in the compiler, microarchitecture, implementation) in order for this to be beneficial.