CpSc 418/538E **Midterm** February 28, 2007

45 minutes. Open papers, notes, homework, books. Calculators allowed. No laptop computers, PDAs, cell phones or other communication devices. 5 questions, 100 points.

0. Your name: _____. Your student number: _____.

1. (**24 points**) Here's a list of papers (in the order that they were assigned) of papers that have been assigned so far this term:

   A: *The Landscape of Parallel Computing Research: A View from Berkeley*

   B: *The Case for the Reduced Instruction Set Computer*

   C: *Instructions Sets and Beyond: Computers, Complexity, and Controversy*

   D: *A 32-Bit VLSI CPU with 15 MIPS Peak Performance*

   E: *The Pentium Chronicals: Introduction*

   F: *Computer Architecture: A Quantitative Approach (2nd edition, pp. 372–411)*

   G: *The MIPS R1000 Superscalar Microprocessor*

   H: *A 0.18μm CMOS IA-32 Processor with a 4 GHz Integer Execution Unit*

   I: *The Limits of Instruction Level Parallelism*

   J: *The Cosmic Cube*

   K: *A Message Passing Standard for MPP and Workstations*

   L: *Using Cache Memory to Reduce Processor Memory Traffic*

   M: *The Sun Fireplane SMP Interconnect in the Sun Fire 3800-6800*

   N: *An Industry-Standard API for Shared-Memory Programming*

   O: *Architectural and Organizational Trade-Offs in the Design of the MultiTitan CPU*

   To which paper would you refer to find information about each topic listed below. You can answer each just by giving the letter for the paper (i.e., "**A**" for the *Landscape* paper, "**B**" for *The Case for RISC*, etc.).

   (a) __J__ Hypercube interconnect networks.

   (b) __F__ The three C's of cache misses.

   (c) __C__ Six distinguishing characteristics of a RISC processor.

   (d) __H__ An integer ALU that can perform two operations during each CPU clock cycle.

   (e) __L__ A cache coherence protocol for a multiprocessor with a single, shared, memory bus.

   (f) __A__ Autotuners.

   (g) __K__ Collective communication actions for message passing computers.

   (h) __G__ A description of register mapping including free lists, active lists and busy-bit tables.

2. (**24 points**) The following questions ask you to explain several points raised in the *MultiTitan* paper.

   (a) Does the paper consider high clock frequency or low instruction latency to be more important for achieving performance? What justification does the author give for his choice?

   Solution:

   - The paper claims that low latency is more important.
   - He claims that while high-latency pipelines can have higher bandwidths (i.e. clock rates) data dependencies and limited instruction level parallelism make deeply pipelined processors lose most of their performance due to stalling.

   (b) Give three example of where the answer to part (a) guided design decisions for the MultiTitan.

   Solution:

   - The pipeline is short, four stages.
   - The use of direct mapped caches allows data to be used during tag comparison.
   - Simple instruction formats and control structures reduce latency.
   - A 64-bit off-chip data bus reduces load and cache-refill latencies.

   (c) Is the I-cache direct mapped, set-associative, or fully associative? What was the main reason for the choice? What is the main drawback of the choice?

   Solution:

   - The cache is direct mapped.
   - This allows instruction decode and register fetch to be performed in parallel with the tag comparison.
   - The miss rate is higher than that for a set-associative cache with the same capacity.

3. (**16 points**) The following questions ask you to explain a few points raised in the *0.18μm CMOS IA-32 Processor* paper. Give a 1-3 sentence answer to each question.

   (a) Does the paper consider high clock frequency or low instruction latency to be more important for achieving performance? What justification does the author give for his choice?

   Solution:

   - The paper claims that high clock frequency is more important.
   - The paper notes that performance does not scale directly with clock frequency, but asserts

     Deeper pipelines make many operations take more clock cycles, such as mispredicted branches and cache misses, but usually more than make up for the lower per-clock-efficiency by allowing the design to run at a much higher clock rate.

     There are other assertions of the value of high clock frequency throughout the paper, but little hard data to back it up. The most compelling evidence is probably the comparison with the Pentium-III in Figure 11.

   (b) Decoding multiple instructions in parallel is challenging for the IA-32 architecture because instructions don't have a fixed size. How does the Pentium-4 address this problem?

   Solution:

   The Pentium-4 uses a trace-cache. Instructions are decoded at a rate of one instruction per cycle when loading the L1 I-cache after a cache miss. The L1 cache holds traces of decoded instructions for use on subsequent re-executions of those instructions.

4. (**12 points**) Goodman's coherence protocol has four possible states for each cache line: invalid, valid, reserved and dirty.

   (a) Describe two situations in which a line can transition from reserved to invalid. Does the cache need to write the line to memory in either of these situations?

   Solution:

   - If the line is evicted due to a cache miss to an address with the same cache index.
   - If another CPU writes to an address included in this cache line.
   - Neither needs a write to memory. When a line is in the reserved state, the CPU has performed one write to the location that was sent on to memory so the caches of the other CPUs could see it and invalidate any copies they had of that line.

   (b) Describe a situation in which a line can transition from dirty to valid. Does the cache need to write the line to memory for this transition?

   Solution:

   - If another processor reads a location that is part of this line, the cache provides the data and transitions to valid.
   - Yes. The cache writes its data for this line to memory as when providing it to the other cache. This means that a cache is free to drop a valid line without writing it to memory.

5. (**24 points**) As we described superscalar processors in class, a physical register can be in one of four possible states: free, busy, ready or graduated.

   (a) When does a register move from free to busy?

   Solution:
   When the logical destination register of an instruction is mapped to this physical register.

   (b) When does a register move from busy to ready?

   Solution:
   When the instruction for which this is the destination has computed its result.

   (c) When does a register move from ready to graduated?

   Solution:
   When the all previous instructions, by program order, have graduated.

   (d) When does a register move from graduated to free?

   Solution:
   When another instruction that writes to the same logical register graduates.

   (e) When can an instruction use the value of one of its register operands?

   Solution:
   When the physical register for that operand is ready or graduated.

   (f) There may be several physical registers that correspond to the same logical register. How does the superscalar CPU ensure that an instruction reads the right physical registers when it fetches its operands?

   Solution:
   The register operand is mapped to the physical register when the instruction was decoded. This physical register will have its value set by the most recent instruction (in program order) that wrote the corresponding logical register.