CIS501– Computer Architecture — Final Exam

Prof. Martin — **Wednesday, Dec. 10, 2008**

This exam is an individual-work exam. Write your answers on these pages. Additional pages may be attached (with staple) if necessary. Please ensure that your answers are concise and legible. Read and follow the directions of each question carefully. Please attempt to answer all the questions (don't allow yourself to get stuck on a single question). You have 120 minutes to complete the exam (approximately one point per minute).

If you are taking this exam as a WPE-I:

Your WPE-I Number: _______

If you are *NOT* taking this exam as a WPE-I:

Name: ____

Problem	Page	Possible	Score
1	2	10	
2	3	5	
3	4	6	
4	5	9	
5	6	6	
6	7	8	
7	8	8	
8	9	11	
9	10	8	
10	12	8	
11	13	11	
Total		90	

1. **[10 Points] True or False**. If a statement is "false," briefly explain how so by describing how the statement may most simply be made true. Unjustified (or poorly justified) "false" answers will be marked wrong. Simply stating the negation of the false statement is *not* sufficient justification. *Please be specific!*

   (a) Power-related issues are relevant only when designing battery-powered devices.

   (b) Smaller transistors generally use less energy when switching.

   (c) Reliability is an increasing concern as transistors continue to shrink in size.

   (d) For a cache of a fixed capacity, doubling the associativity of the cache will halve the size of the tag array.

   (e) A victim buffer improves the average miss penalty of a cache.

   (f) A write buffer (also called a store buffer) is primarily intended to hide store latency.

   (g) Assuming similar pipeline depths, improving branch prediction accuracy is generally more important in a superscalar (multiple-issue) processor than in a scalar (single-issue) processor.

   (h) Because branch prediction is so critical for performance, a processor today typically uses a branch predictor that is 1 MB or larger.

   (i) A branch target buffer (BTB) is tagged because it must be correct; in contrast, a branch direction predictor is untagged because it is only a predictor and thus does not always need to be correct.

   (j) When a cache block in the "E" cache coherence state is evicted from the cache, it must be written back to the memory.

2. **[5 Points]**

   (a) **Processor Performance**. Assume a typical program has the following instruction type breakdown:

   - 35% loads
   - 10% stores
   - 50% adds
   - 3% multiplies
   - 2% divides

   Assume the current-generation processor has the following instruction latencies:

   - loads: 4 cycles
   - stores: 4 cycles
   - adds: 2 cycles
   - multiplies: 16 cycles
   - divides: 50 cycles

   If for the next-generation design you could pick one type of instruction to make twice as fast (half the latency), which instruction type would you pick? Why?

   (b) **Caches**. You've been asked to determine the cache parameters that minimize average memory access latency for a next-generation flagship processor. The processor has a simple non-pipelined datapath in which all non-memory instructions take one cycle and all memory operations that hit in the cache take either two or three cycles (depending on the below cache configuration). After several experiments, you have narrowed down the caches to two choices:

   i. A 64 KB cache which has a 2-cycle access time and a 90% hit rate

   ii. A 128 KB cache which has a 3-cycle access time and a 95% hit rate

   Under what condition would you select cache (*i*) over cache (*ii*)? Use a calculation to justify your answer.

3. **[6 Points] Miscellaneous Short Answer**. Answer the following questions in just a few sentences.

   (a) What is the primary trade-off between coarse-grained locking and fine-grained locking? That is, what is the primary advantage/disadvantage of one over the other?

   (b) What are micro-operations (uops)? What problem do they attack?

   (c) If a single-cycle datapath with delay of n nanoseconds is converted into a pipelined datapath with p stages, the clock period will be longer (slower) than n/p nanoseconds. Give two reasons for this:

   1.

   2.

   (d) In a multiprocessor, increasing the cache block size can have an additional effect on performance. What causes this effect? Is it a positive or negative impact?

4. **[9 Points] Branch Prediction and Pipeline Depth**.

Branch prediction accuracy affects the choice of pipeline depth (i.e., number of pipeline stages). Consider the two simple single-issue pipelines below:

Pipeline A: A traditional five-stage pipeline with a single-cycle load-use penalty a two-cycle branch misprediction penalty at a clock frequency of 250Mhz (4ns clock cycle).

Pipeline B: A twelve-stage pipeline with a two-cycle load-use penalty and a six-cycle branch misprediction penalty at a clock frequency of 333Mhz (3ns clock cycle).

Consider a program that is 30% loads, 20% branches, and 50% other instructions. One third of loads have no dependent instructions within two instructions after it, one third of the loads have a dependent instruction immediately following the load, and the remaining one third of loads are followed by an independent instruction and then a dependent instruction.

(a) Give the cycles per instruction (CPI) and nanoseconds per instruction (NPI) for each of the two pipelines assuming a simplistic "predict not taken" that gives a prediction accuracy of 50% on this program.

(b) At what branch prediction accuracy does the twelve-stage pipeline outperform the five-stage pipeline?

(c) Based on typical branch prediction accuracies, which pipeline would you choose for best performance?

5. **[6 Points] Storage I**.

Consider a disk with rotational speed of 6000 RPM (which is 100 rotations per second or one full rotation every 10ms), an average seek time of 10ms (with zero seek time to an adjacent track), and negligible controller overhead. Each track on the disk has 1024 sectors and each sector is 1024 bytes.

(a) Calculate the maximum read throughput of the disk for sequential reads.

(b) Now consider random reads for the same disk. How large must the chunks of data read from the disk be to ensure a non-sequential read throughput of 14.25 MB/second?

6. **[8 Points] Storage II**.

You've been asked to design a storage subsystem that has a total capacity of 5 TB (1 TB is 1024 GBs) and that can support 30,000 random reads per second.

(a) If each disk costs $100, can support 100 non-sequential reads per second, and has a capacity of 1TB. How much would such a storage subsystem cost?

(b) Flash memory is non-volatile storage without any moving parts, thus it avoids all seek and rotation latencies to provide a dramatic increase in throughput for non-sequential reads. On October 15th, Intel released the X25-E flash-based solid-state storage device that uses a standard disk interface, has a capacity of 64GBs, can perform 35,000 non-sequential reads per second, and costs around $1000. Given the same storage system requirements as above, how much would such a storage system cost if it was comprised entirely of these X25-E storage devices?

(c) How might you build a storage subsystem using a combination of the disk drives and solid-state drives described above?

(d) Would your implementation of this hybrid storage system use extra hardware? Change the software? Both? Neither? Justify your answer.

(e) Assuming the best case, how much might such a hybrid storage subsystem cost?

(f) What property would need to be present in the workload for such a system to work?

7. **[8 Points] Static and Dynamic Scheduling**.

   (a) What are the three primary impediments that limit the compiler's ability to perform effective static scheduling?

   1.

   2.

   3.

   (b) What techniques do dynamically scheduled processors use to overcome each of these three limitations?

   1.

   2.

   3.

   (c) Although today's mainstream laptop and desktop processors employ dynamic scheduling and wide-issue superscalar (four or more instructions per cycle) execution. In contrast, some newer chips in other domains do not. Give two reasons we're seeing a resurgence in-order processor designs and more modest superscalar execution (for example, only two instructions per cycle)?

   1.

   2.

8. **[11 Points]** **Superscalar versus Vectors**. Superscalar execution and vector instructions are two ways of increasing the amount of computation a processor can perform each cycle. Consider adding these features to a 64-bit pipelined processor.

   (a) What impact–*if any*–would adding **superscalar execution** (for example, the ability to perform four instructions per cycle) have on the following six aspects of the design:
   Instruction fetch:

   Stall logic:

   Bypassing:

   Register file:

   Execution units:

   Data cache:

   (b) What impact–*if any*–would adding **vectors** (for example, using four-element vectors to perform four 64-bit operations in parallel for each vector instruction) have on the following six aspects of the design:
   Instruction fetch:

   Stall logic:

   Bypassing:

   Register file:

   Execution units:

   Data cache:

   (c) Apart from the above hardware implementation differences, what are the two most important advantages of superscalar execution over vector operations:
   1.

   2.

9. **[8 Points]** **Multithreading**. Consider a video encoding computation running on an in-order dual-issue superscalar processor. You have two video files to encode, each taking 10 minutes to execute. Thus, encoding both files back-to-back on this simple processor will take 20 minutes.

   This computation spends 30% of its cycles executing two instructions, 50% executing one instruction, and 20% of its cycles executing zero instructions because of a cache miss. The cache miss latency is 20 cycles.

   Consider the change to the runtime when running these two computations at the same time on various multithreaded version of the above two-way superscalar processor.

   (a) Calculate a reasonable best-case estimate of the combined runtime on a coarse-grained multi-threaded processor (which switches threads on a cache miss) with a thread-switch time of 10 cycles.

   (b) Calculate a reasonable best-case estimate of the combined runtime on a fine-grained multi-threaded processor (which can switch threads every cycle without penalty).

   (c) Calculate a reasonable best-case estimate of the combined runtime on a simultaneously multi-threaded processor.

Continued on next page

Now consider a different implementation of the computation. This computation spends 10% of its cycles executing two instructions, 30% executing one instruction, and 60% of its cycles executing zero instructions because of a cache miss.

(a) Calculate a reasonable best-case estimate of the combined runtime on a coarse-grained multi-threaded processor (which switches threads on a cache miss) with a thread-switch time of 10 cycles.

(b) Calculate a reasonable best-case estimate of the combined runtime on a fine-grained multi-threaded processor (which can switch threads every cycle without penalty).

(c) Calculate a reasonable best-case estimate of the combined runtime on a simultaneously multi-threaded processor.

(d) Give two general reasons these simple calculations are optimistic estimates of performance? That is, what two assumptions did you make in the above calculations?

1.

2.

10. **[8 Points] Cross-Cutting ISA Issues**.

(a) Most ISAs have distinct "integer" and "floating point" register files. Give an advantage and a disadvantage of having separate register files versus a single register file for both types of data.

Advantage:

Disadvantage:

(b) Throughout the semester, we discussed several ISA extensions. What new instruction(s) have been added for each of the following:

- To reduce branch mispredictions?
- To reduce cache misses?
- To express data-level parallelism?
- To facilitate shared-memory multiprocessing?

(c) What aspect of an ISA has the most impact when implementing multithreading? Why?

11. **[11 Points] Parallelism At All Levels**.

Throughout the semester we've explored parallelism at many different levels: beginning with parallelism at the level of instructions (or phases of instructions) to multiple processors cores on a chip.

Based on what we've discussed this semester, give an example use of parallelism at at least five different levels of granularity. For each of them also describe both: (1) the specific reasons for (or benefits of) employing parallelism, and (2) any disadvantages or challenges of exploiting parallelism at that level of granularity.

1.

2.

3.

4.

5.

Give an example of a system that exploits all of these forms of parallelism: