

15-740/18-740 Computer Architecture

Homework 1

Due Friday, September 16, at 12:00 PM

1. Coming challenges in microarchitecture and architecture

▷ Read Ronen, R.; Mendelson, A.; Lai, K.; Shih-Lien Lu; Pollack, F.; Shen, J.P.; , “Coming challenges in microarchitecture and architecture,” *Proceedings of the IEEE* , vol.89, no.3, pp.325-340, Mar 2001.

Consider the following set of instructions provided by the programmer:

```
ADD r3 ← r2, r1
MOV r0 ← r9
MOV r1 ← r10
MUL r4 ← r0, r1
ADD r5 ← r3, r2
SUB r8 ← r6, r7
```

You are the instruction scheduler! Determine the order of execution and number of cycles to execute these instructions (while minimizing number of cycles) on a system with a:

- (a) Scalar, in-order core.
- (b) Super-scalar, in-order core.
- (c) Super-scalar, out-of-order core.

Assume you can execute up to two instructions in the same cycle, each instruction takes one cycle to execute, and that the number of functional units (adder and multiplier) and ports on the register file (for the move instruction) are infinite.

Next, recall the concept of “eager execution” presented in the paper. Though this mechanism could be used as a complete alternative to branch prediction, it may not make sense to treat all branches the same.

- (d) Qualitatively describe the characteristics of branches where using eager execution would be more beneficial for performance than using branch prediction.
- (e) Now, consider the alternative: Qualitatively describe the characteristics of branches where using *branch prediction* would be more beneficial for performance than using *eager execution*.

2. Requirements, bottlenecks, and good fortune

▷ Read Patt, Y.; , “Requirements, bottlenecks, and good fortune: agents for microprocessor evolution,” *Proceedings of the IEEE* , vol.89, no.11, pp.1553-1559, Nov 2001.

The paper notes that “at each step in the hierarchy, from choice of algorithm, to language, to ISA to microarchitecture, to circuits, there are choices and therefore tradeoffs.”

- (a) Based on your knowledge from the readings, what is an example of a trade-off present at a step in the hierarchy?

Though computer architects typically operate in the domain of the ISA/Microarchitecture/Circuits, the decisions they make can affect higher levels of the hierarchy.

- (b) What are some of the trade-offs present in making architectural decisions which may require changes to the higher levels of the hierarchy, such as the compiler, operating system, and programming languages?

Refer to the list of applications in Section I.D., "Application Space."

- (c) Pick two of these applications. Imagine you have been asked by a leading processor manufacturer to design two specialized cores, tailored to the unique needs of each of the two applications you picked. How might the constraints of each application affect your designs? What trade-offs would you consider in each of your designs?

3. Concepts in caching

You should be familiar with caching concepts from 18-447 or a similar computer architecture course. These questions are designed to bring that knowledge back into your "cache."

- (a) What program characteristics do caches exploit? (In less than 20 words.)
- (b) Is it possible to have a program that achieves a 100% hit rate in a processor cache (i.e., all load/store accesses hit in the cache)? If so, write a sample program (pseudo-code) that results in 100% hit rate in a processor cache. State your assumptions and necessary conditions.
- (c) Is it possible for a program to have 0% hit rate in a processor cache (i.e., no load/store access is a hit in the cache)? If so, write a sample program (pseudo-code) that results in a 0% hit rate in a processor cache. State your assumptions and necessary conditions.
- (d) What are the advantages of (i.e., reasons for) including multiple levels of caches in a processor?
- (e) What are the disadvantages of including multiple levels of caches in a processor?
- (f) What is the difference between a cache implemented in hardware and a cache implemented in software? List some constraints and considerations that make these two different.

4. Instruction scheduling and execution

As we discussed in lecture, modern processors give the illusion of in-order execution of instructions, even though in the architecture instructions may be executed out-of-order.

- (a) Why can dynamic instruction scheduling provide higher performance than static instruction scheduling?
- (b) Why is the abstraction of in-order execution useful to the programmer?
- (c) What would happen if the abstraction of in-order execution were not present (as in Tomasulo's original algorithm)?