548
Computer Architecture

M, W 2:30 - 3:50 EE 003

Mark Oskin, oskin@cs.washington.edu, CSE 564
Adrian Sampson, asampson@cs.washington.edu, CSE 352

Time schedule

- Class cancelled: 1/23. Holidays: 1/21, 2/18
- Week 1,2: Fundamental concepts
- Week 3,4: Microarchitecture Techniques
- Week 5: HW/SW Interface
- Week 6: Technology
- Week 7: Multiprocessors
- Week 8-End: Project reports

Overview

- No book, just topical literature
- Grading:
  - 20% warm-up project (become familiar with arch research thinking and tools)
  - 30% project (choose your own adventure)
  - 10% participation (SPEAK!)
  - 10% daily homework (slide prep on readings)
  - 30% Final exam (essay style)

Warm-up project

- Answer the following question:
  - Assume you can have up to 8MB of cache.
  - Assume that the delay to access (for read or write) a cache is given by the following formula: delay = ceil(sqrt(cache_size / 8KB) * Associativity)
  - Assume the delay to access DRAM is 200 processor clock cycles.
  - For some real benchmark or benchmark suite, what is the ideal **data** cache hierarchy design?
  - Other constraints:
    - No individual cache should be smaller than 8KB
- Due: Feb 4th: A short paper (2-3 pages) describing how you answered the question, what the answer is, references, etc. Likely this paper will have a few graphs, and will contain a thorough description of your technique, why the technique is sound, what the limitations of the technique are, and the results obtained.

Warm-up project

- Note 1: This is **not an easy project** and is **intentionally very open ended.**
    - I want you to fish around for tools, talk to Adrian and myself.
- Note 2: This page will help: http://pages.cs.wisc.edu/~arch/www/tools.html
    - Sub note 2: Personally, I would use pin for this and write my own cache simulator. But YMMV
- Note 3: Questions you must answer:
    - What is the design space?
        - How do I reasonably constrain it?
    - How do I actually answer this question?
- Note 4: **Start NOW** and set an aggressive schedule for yourself (this is not a group project):
    - This week: pick an infrastructure to work with, install it, and make sure it is going to work for you.
    - By next monday: understand the entire design space, and the (possibly sub-space) of designs you are going to explore.
    - By end of next week: Have something not quite working but that is producing numbers.
    - End of week 3: have something producing good data, graph it, and start writing.
- Note 5: Share your joys and sorrows in class with everyone. If you have something working, talk about it here in 548 with your classmates. Help each other, but please try and do most of the helping actually *in* class so that I know who's helping who (you *will be rewarded* for helping others).
- **Run for your life Charlie Brown! Yes, this is hard.**

Class project

- Class projects can be done solo or in pairs
- The goal of the class project is to answer some **architectural question** that **you (two) find interesting.**
- You should answer the question compellingly
- The answer should appear in the form of a research paper (~ 10 pages, normal form, etc) and a short talk (20 minutes).
- Many class projects turn into submitted and published research papers. Make that your goal.

Sample class projects

- Experimental work:
  - Trace a complete OS/app stack and compute the parallelization limit
  - Hack an OS/HV by exploiting a hardware bug
  - How much data redundancy is in the cache? How to exploit?
- Design work:
  - a < 200xtr microprocessor
  - a system capable of running "forever" from scavenged energy
  - A processor designed to run python
- Application work:
  - Parallelize something hard
  - Implement various algorithms on Grappa/XMT
- Literature work:
  - Survey of the last 20 years of commercial microprocessors
  - Survey of a particular topic in architecture, such as branch prediction, caching, pipelining, etc.

Daily Homework

- For each class 1-3 papers are assigned
- Read each, and produce 4 slides on each
    - 3 slides: summary
    - 1 slide: questions
- Email them to me ***before class***
- Depending on class size, 2-3 times a quarter you'll present your slides

Final exam

- Essay style
- 3-4 questions
- Not graded on much grammar or speling.

Office / Happy hour

- Every Monday after class Adrian and I will be at Big Time from 4 - 5:00 (later if people are around). Find us there if you have questions. Find us there if you just want to chat about architecture or grad school. Find us there if you want a (sometimes free) beer or other beverage (you don’t have to drink beer to attend office hours). Find us there if you want to buy Adrian a beer to thank him for his helpful and friendly service!
- Alternatively, I can also meet you most any time by appointment. Or you can reach me by cell 206-293-9456. Yes, I txt.

Other class stuff

- **SPEAK!** This class will bore you and bore me if you stay silent. Participation (speaking!) is a graded component of this class.
    - Contrary to what you've been told your whole life, there are stupid questions. But speak anyway! I don't hold it against you and I value you speaking far more than speaking correctly!
- Life happens. If you are going to be late by a day or two on the projects, that is fine. If you need to miss class for conferences, work, family, it's sunny outside finally, etc, that's fine too.
- You can miss up to two daily homeworks and none will be the wiser.
- Be nice to your TA. He has *chosen* to help you.
- As you are about to discover, this class requires your participation:
    - *You* take turns introducing the topic at hand.
    - *You* must speak for class to move forward.
    - Don't expect canned slides. What you see are canned topics of discussion. I serve as note taker (guide?), and *you* make the slides.
    - This will become clear shortly...

Who are you?

- 13 Grad students (2 5th years)
- 3 from WA, 1 MI, Switzerland, India, TX, *, NY, VA, FR/CAN, OR, IL,
- All in 20's, 30's
- Embedded, OS, Theory, PL, Graphics, Vision, Security, ML, Debugging, Architecture, ...
-

Who am I?

- Started hacking code pre-PC (1982) back when you bought books of source code in mall book stores.
- Burnt my first computer to the ground in 1986 (crossed wires). It would not be my last...
- I can't spelll: I don't know good english, I try and hide these facts by writing small and illegibly. Don't laugh, I have a tough time spelling parallell sometimes.
- Joined the faculty in 2001, worked on a variety of topics including: intelligent memory, quantum microarchitectures, dataflow, graph runtimes, and most recently, new memory interface abstractions for kilo-ILP processing.
- Hobbies: motorcycle riding, woodworking, house construction, good food, good beer, analog photography, outdoor*(fishing, hiking, sailing, snow shoeing, cross country skiing, ...), radio (KE7SCH)
- But these days Sky (my daughter) occupies most of my free time:

What do you want from this class?

- Free beer!
- better understanding of arch. choices for processors
- quals requirement
- how arch. changes require us to adapt SW stack
- what is the trade-offs in the alphabet soup of options (multicore, SMT, ILP, ...)
- Friends! (through shared pain)

What are some topics that you want to learn about in this class?

- Dataflow
- XMT/MTA, Graph stuff
- Lay of the land of modern processors
- Low power computing
- Dead branches: Crusoe, K2, CM2, CM5, ...
- Tension between generality and specificity
- Interconnects
- Who does what and why?
- Single threaded performance improvements
- What do people do when they design a chip?
- non-CMOS processors: DNA, memristors, carbon-nanotubes, ...
- quantum computers
-

What is a computer architecture?

- Micro-architecture:
  - Design of processors
  - Components and their interconnection
- Architecture:
  - The programming interface to users of a processor
- What about dynamic translation?

What do computer architects do?

- ISA design
- Leverage xtr's for performance
- Do more with less
- Matching at the interfaces
    - Manufacturing / Architecture
    - SW / Architecture
- Expand services to new markets

How do computer architects do what they do?

What challenges face computer architectures today?

What should HW do?
(And why?)