CS252 Graduate Computer Architecture
Spring 2014
Lecture 1: Introduction

Krste Asanovic
krste@eecs.berkeley.edu
http://inst.eecs.berkeley.edu/~cs252/sp14

Welcome to CS252!

- An exciting time for computer architecture
- Which means a terrifying time for computer users

Upheaval in Computer Design

- Most of last 50 years, Moore's Law ruled
  - Technology scaling allowed continual performance/energy improvements without changing software model
- Last decade, technology scaling slowed/stopped
  - Dennard scaling over (supply voltage ~fixed)
  - Moore's Law (cost/transistor) over?
  - No competitive replacement for CMOS anytime soon
  - Energy efficiency constrains everything
- No "free lunch" for software developers, must consider:
  - Parallel systems
  - Heterogeneous systems

Today's Dominant Target Systems

- Mobile (smartphone/tablet)
  - >1 billion sold/year
  - Market dominated by ARM-ISA-compatible general-purpose processor in system-on-a-chip (SoC)
  - Plus sea of custom accelerators (radio, image, video, graphics, audio, motion, location, security, etc.)
- Warehouse-Scale Computers (WSCs)
  - 100,000's cores per warehouse
  - Market dominated by x86-compatible server chips
  - Dedicated apps, plus cloud hosting of virtual machines
  - Starting to see some GPU usage, but mostly general-purpose CPU code
- Embedded computing
  - Wired/wireless network infrastructure, printers
  - Consumer TV/Music/Games/Automotive/Camera/MP3

CS252 Goals

- Provide background for Berkeley architecture oral prelim exam
- Prepare graduate students for research in computer architecture
- Provide advanced architecture material for graduate students in related areas (operating systems, compilers, networking, high-performance programming)

CS252 Prerequisites

- Upper division graduate architecture class (CS152 or equivalent)
- Thoroughly familiar with ISAs, assembly programming, in-order pipelining and caches.
- Should have seen and understood most of following:
  - Out-of-order superscalar
  - Vectors
  - VLIW
  - Multithreading
  - Virtual memory
  - Cache coherency
- Will be quickly reviewing this material, but no time to catch up if you have not seen material before

Prereq Pop Quiz

- 10 minute multiple choice quiz
- Write name, ungrad or grad, and year, email
- No books/computers/phones

CS252 Approach

Understand computer architecture through:

- History
- Applications
- Technology Trends
- Architectural Design Patterns
- Programming models
- Business models

Why Study Architecture History (1)?

- Appreciate the rich architecture lore
- Understand how the current architecture landscape was shaped by design decisions driven by earlier application, technology, or business concerns
- Help write better related work sections in your research papers

Why Study Architecture History (2)?

- *"Those who don't know history are doomed to repeat it."* Edmund Burke
- Many mistakes made in mainframe design, were repeated in minicomputers, then again in microprocessors, …
- Many proposed *"revolutionary"* computer architecture ideas repeat earlier proposals that were investigated and later abandoned for good reason
  - Negative results not well recorded in literature, as advocates only occasionally reflect on mistakes
  - (*Of course, applications and technology might change to make an old bad idea a new good idea*)

Applications

- Computers exist to run applications
- Need to understand demands of current and future applications to guide architecture design decisions
  - Also, historical applications that guided current designs
- Real applications are complex, and include much legacy code (if only in OS and libraries)
- Important to understand how applications are written, tuned, and maintained by developers
  - Architects often overoptimistic about effort developers will expend on exploiting hardware features
- Benchmarks and kernels are often substituted for applications in architecture studies, but often with poor correlation to real application behavior
  - Understand limitations of workloads used in evaluation

Technology Trends

- Computing technology is a very fast-moving field, so must constantly track changing technology abilities
- New emerging technologies in 2014:
  - Non-volatile memory, possible NAND flash replacements
  - Integrated silicon photonics
  - Extensive 3D stacking and new packaging technologies

Applications and Technology Trends

A virtuous circle between applications and technology trends:

- New technologies make new applications possible
  - E.g., the microprocessor enabled personal computing
- Revenues from popular applications fund and guide technology development
  - E.g., flash memory for digital cameras and MP3 players

Architectural Design Patterns

- Understand architecture space through long-lived, recurring standard architectural design patterns, for processors, memory systems, and interconnect
- Almost any "new" architecture can be understood as composition of standard architecture design patterns
  - Including custom accelerators
- We will be looking at case studies of real machines and breaking down into standard patterns

Programming Models

- Major architectural design patterns are usually associated with an expected programming model
  - Serial code for uniprocessors (C)
  - Loop nests for vector machines (FORTRAN)
  - Element function code for GPUs (CUDA/OpenCL)
  - Annotated loops for shared memory multiprocessors (OpenMP)
  - Explicit message passing for clusters (MPI)
  - CSP or Kahn process networks for distributed embedded systems (Occam)

Business Models

- Viability of different designs depends on expected business model
- Some factors to consider:
  - Volume of design: billions of units/year (smartphone) or 100s of units/year (supercomputer)
  - Non-recurring engineering costs: new complex custom chip requires $50M, new FPGA board $100K
  - Horizontal (Wintel) versus Vertical models (Embedded)

CS252 Architectural Design Patterns

- Microcoding/Pipelining/Decoupling
- In-order/Out-of-order superscalars
- SIMD (Vectors, Packed SIMD, GPUs)
- VLIW
- Multithreading
- Memory system (Regfiles, Caches, DRAM)
- Message-Passing systems (MPPs, WSCs)
- Shared-Memory systems (coherence, synchronization)
- Protection,Security,Virtual Memory&Virtual Machines
- Other parallel (Associative, Systolic, Dataflow)
- Networking and NICs
- Storage and device interfaces

CS252 Architectural Design Issues

- Applications, workloads, and benchmarks
- Technology trends
- Business Models
- Power/Energy/Thermal issues
- Resiliency, coping with faults and soft errors
- Simulation methodologies

CS252 Course Grade Allocation

- Reading assignments and summaries (25%)
- Midterm Exam (25%)
- Course project (50%)

Reading assignments and summaries (25%)

- You'll be reading many papers this semester (~2 per class) – mostly "must read" papers for architects
- Require 200-300 word review (NOT summary) per paper (review as if on program committee, describing strengths and weaknesses)
- Review must be ASCII plain text in email to krste@eecs ***before*** class (zero credit after 10:30AM)
- Lowest 3 days' scores ignored (so can skip up to 3)
- Each class starts with discussion of papers (~20 minutes) before lecture (~60 minutes)
- Want good discussion, will call upon all students
- 15% grade for text, 10% for discussion participation

Midterm (25%)

- In-class midterm (80 minutes)
- April 2 (Wednesday, week after spring break)
- Covers lecture material up till spring break

- Closed book, no notes, no computer, no phone, …

- Test will emphasize understanding not memorization

Course Project (50%)

- Students work in groups of 2 to complete a project
- Topic that could be paper at top architecture conference (ISCA, ASPLOS, MICRO, HPCA)
- Two-page proposal due Monday March 17
- Two weeks of 1-1 project advising during class time
  - No TA for CS252 this semester, so must find/support own infrastructure for project
- Final presentation (15%) in RRR week, date TBD
- Final paper (35%) due Friday May 9, 11:59PM Pacific Time, mail to **`krste@eecs`**
  - Must be in PDF, conference format (10-page, 2-column)
  - No extensions

A Good Project Proposal

Must have title and two authors' names.

Should answer:

- What are you are trying to improve
- How is it currently done, or what has been tried before
- What is your potential upside if successful
- How will you evaluate your idea
- What are intermediary milestones to measure progress

Krste's Office Hours

- Mondays 5-6pm in ASPIRE Lab
- Email ahead of time to confirm

- Can arrange other times to meet (M-W only) if this doesn't work

Class Website

`http://inst.eecs.berkeley.edu/~cs252`

- Class schedule
- Course info
- Lecture slides, posted morning before lecture
- Reading assignments
- Supplementary material (additional reading on each topic)

Video Taping

- Will experiment with video taping of class for external use.
- Will not video discussion or use student faces/voices, but will repeat/use questions asked.

Next Time Reading Assignments

- "Architecture of the IBM System/360", Amdahl, Blaauw, Brooks, 1964
- "Design of the B5000 System", Lonergan, King, 1961

Acknowledgements

- This course is partly inspired by previous MIT 6.823 and Berkeley CS252 computer architecture courses created by my collaborators and colleagues:
  - Arvind (MIT)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
  - David Patterson (UCB)