CS252 Spring 2017
Graduate Computer Architecture

Lecture 1: Introduction

Lisa Wu
lisakwu@eecs.berkeley.edu
http://inst.eecs.berkeley.edu/~cs252/sp17
Welcome!

- What is computer architecture?
  - Myopic view from years ago: Instruction Set Architecture (ISA)
  - Updated view: Instruction Set Architecture, Microarchitecture, and Hardware

Computer architects must design a computer to meet functional requirements as well as price, power, performance, and availability goals.

~H&P 5th Edition
What we are going to cover today...

• Computer design then, computer design now
• The CS252 approach to understanding computer architecture
• Course structure
Computer Design Then, Computer Design Now
Moore made the observation in 1965; 50 years later…

B. Sutherland, “No Moore? A golden rule of microchips appears to be coming to an end”, The Economist, Nov 2013.

No Moore?
**Dennard Scaling**

“Design of ion-implanted MOSFET's with very small physical dimensions”
Robert H. Dennard, Fritz H. Gaensslen, Hwa-Nien Yu, V. Leo Rideout, Ernest Bassous, and Andre R. LeBlanc
IEEE Journal of Solid-State Circuits, October 1974

**Table 1**
Scaling Results for Circuit Performance

<table>
<thead>
<tr>
<th>Device or Circuit Parameter</th>
<th>Scaling Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device dimension $t_{ox}, L, W$</td>
<td>$1/\kappa$</td>
</tr>
<tr>
<td>Doping concentration $N_a$</td>
<td>$\kappa$</td>
</tr>
<tr>
<td>Voltage $V$</td>
<td>$1/\kappa$</td>
</tr>
<tr>
<td>Current $I$</td>
<td>$1/\kappa$</td>
</tr>
<tr>
<td>Capacitance $\epsilon A/t$</td>
<td>$1/\kappa$</td>
</tr>
<tr>
<td>Delay time/circuit $VC/I$</td>
<td>$1/\kappa$</td>
</tr>
<tr>
<td>Power dissipation/circuit $VI$</td>
<td>$1/\kappa^2$</td>
</tr>
<tr>
<td>Power density $VI/A$</td>
<td>1</td>
</tr>
</tbody>
</table>

The failure of Dennard Scaling

CMOS technology scaling

• This classical technology scaling enabled Moore’s law but is coming to an end in 6-9 years (3 generations or so)

• Dennard scaling that enabled constant power per chip despite increasing CMOS integration density ended in the mid-2000’s

• Intel has abandoned it’s tick-tock model and moving to 3 designs per technology generation

• ITRS (International Technology Roadmap for Semiconductors) has stopped publishing roadmaps because there is no clear path to continued scaling.
Moore made the observation in 1965; 50 years later…

Target systems today: PMD

• PMD (Personal Mobile Device): a collection of wireless devices with multimedia user interfaces such as cell phones and tablet computers.

• More than 1 billion units sold per year

• Market dominated by ARM-ISA-compatible general purpose processor(s) in a SoC (system-on-a-chip) and a sea of custom accelerators (radio, video, audio, image, graphics, motion, location, security, etc.)
Target systems today: Desktop

- Desktop Computing: spans from low-end netbooks to high-end workstations.
- Half of the desktop computers made each year have been battery operated laptop computers.
- The newest and highest performance microprocessor and cost-reduced microprocessors often appear first in desktop systems.
Target systems today: Servers

- Availability and scalability requirements set them apart from desktop systems.
- In most cases, throughput trumps single-thread performance.

Cost of System Downtime per Hour:

- Brokerage Operations: $1,000,000
- Credit Card Authorization: $1,000,000
- Package Shipping Services: $1,000,000
- Home Shopping Channel: $1,000,000
- Airline Reservation: $1,000,000
- ATM Service Fees: $1,000,000

Source: H&P Fifth Edition
Target systems today: WSCs

- WSCs (Warehouse-Scale Computers)/Clusters
  
  - 100,000’s cores per warehouse
  
  - Market dominated by x86-compatible server chips
  
  - Dedicated apps plus cloud hosting of virtual machines
  
  - Starting to see more GPU usage, even some FPGAs, but mostly general purpose CPU code
Target systems today: Embedded

- Embedded Computers
- IoTs (Internet of Things)
- Wired/wireless network infrastructure
- Consumer TV/Automotive/Camera/MP3
- Able to run third-party software compared to PMDs
The CS252 Approach to Understanding Computer Architecture
Understanding Computer Architecture

Applications

Architectural Design Patterns

Business Models

History

Technology Trends

Programming Models

Berkeley Architecture Research
History

• Understand how the current architecture landscape was shaped by design decisions driven by earlier application, technology, or business concerns

• Don’t make the same mistakes

• Application and technology maturing can turn an old did-not-work/old bad idea into a new good idea
Applications

• Need to understand demands of current and future applications to guide architecture design decisions

• Real applications are complex and include much legacy code (if only in OS and libraries)

• Benchmarks and kernels are often used instead of real applications in architectural studies; need to understand workload modeling
Technology Trends

• Computing technology is a very fast-moving field, so one must constantly track changing technology abilities to make future-looking design decisions.

• e.g. New memory devices: NVM technology such as Intel/Micron 3D XPoint memory that offers fundamentally different cost, density, latency, throughput, reliability, and endurance tradeoffs than traditional DRAM, flash, or magnetic disks.

• e.g. Deep 3D integration: global wires too expensive relative to compute, shortening interconnects by routing in three dimensions. This enables greater energy efficiency, higher bandwidth, lower latency between system components inside the 3D structure such as 3D stacking.
A virtuous circle

• New technologies make new applications possible
e.g. the microprocessor enabled personal computing
e.g. exponential growth in compute performance
enabled interpreted languages and scripting languages
to deliver acceptable application performance
e.g. 3D stacking enabled near-memory compute

• Revenues from popular applications fund and guide
technology development
e.g. flash memory for digital cameras and mp3 players
Architectural Design Patterns

• We will understand computer architecture 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 architectural design patterns

• We will look at case studies of real machines and break the design down into standard architectural design patterns
Architectural Design Patterns

- Example patterns we will see in CS252:
  
  Microcode, Pipeline, Decouple, In-Order/Out-of-Order
  Superscalar, SIMD, VLIW, Multithreading, Memory
  subsystem, Shared-memory system, Protection,
  Security, Virtual Memory, Networking, Storage and
  device interfaces, etc.
Programming Models

- Major architectural design patterns are usually associated with an expected programming model

Serial code :: Uniprocessors (C)
Loop nests :: Vector machines (FORTRAN)
Annotated loops :: Shared memory multiprocessors (OpenMP)
Element function code :: GPUs (CUDA/OpenCL)
Explicit message passing :: Clusters (MPI)
Business Models

• Viability of different computer designs depends on the expected business model

• Some factors to consider:
  Volume: billions of units/year for smartphones or hundreds of units/year for supercomputers
  Non-recurring engineering costs: new complex custom chip requires $10-50M or new FPGA board requires $10-100K
CS252 Course Structure
• Berkeley Architectural Oral Prelim Exam
• Research in Computer Architecture and Related Areas
• Possible Publication Submission
Prerequisites

• Upper division architecture class (CS152 or equivalent)
• Very familiar with ISAs, assembly programming, in-order pipelining, and caches.
• Somewhat familiar with out-of-order superscalar, vectors, VLIW, multithreading, virtual memory, and cache coherency.
• We will be reviewing this material, however, if you have not seen them before, there won’t be time to catch up.
POP QUIZ!
# Course Grade Allocation

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reading Assignments and “Reviews”</td>
<td>15%</td>
</tr>
<tr>
<td>Problem Sets</td>
<td>15%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>30%</td>
</tr>
<tr>
<td>Course Project</td>
<td>40%</td>
</tr>
</tbody>
</table>
Reading Assignments and Reviews (15%)

• Read 2 or 3 classical papers for architects per class

• Submit a review as if you are on a PC (Program Committee) for an architecture conference

• Review must be entered on Google Forms the night before class (zero credit after 11:59pm)

• Lowest two scores are dropped

• We will discuss the papers for half an hour each class (will employ something similar to the socratic method)

• 5% grade for reviews, 10% grade for discussion participation
Problem Sets (15%)

• Weekly problem sets: assigned on Mondays, due the following Sunday at 11:59pm

• Must be done individually, no collaboration

• Questions often don’t have simple correct answers; justify your thinking

• First problem set will be out next Monday, due the following Sunday at 11:59pm
Midterm Exam (30%)

• In-class midterm exam (80 minutes)

• Covers lecture material, reading assignments, and problem sets

• Closed book, no notes, no computer, no phone; test will emphasize understanding not memorization

• Date either before or after Spring Break
Course Project (40%)

• Work in groups of 2 to complete a project

• Pick a topic that could be a paper at a top-tier architecture conferences (ISCA, MICRO, ASPLOS, HPCA)

• Two-page proposal due mid-March

• 1-1 project advising during class time in the second half of the semester

• Final presentation (15%) on Thursday May 4 from 10am-12:30pm at Woz Lounge

• Final project paper due by Friday May 5 at 11:59pm — must be in PDF, conference format, Latex template will be provided, no extensions
The 1-2-3 of a Good Project Proposal

• Will have a list of example projects for consideration
• Can come up with your own projects
• Should be able to answer the following questions:
  1. What problem are you trying to solve?
  2. What idea/solution are you proposing?
  3. Are there other solutions that you know of? If so, compare and contrast.
  4. What is the potential upside using your idea/solution?
  5. How will you evaluate your idea/solution?
  6. What are the intermediary milestones to measure your progress?
Course Info

• Class website
  http://inst.eecs.berkeley.edu/~cs252/sp17

• My Office Hour: Tuesdays 3-4pm

• TA: Eric Love
  TA Office Hour: Thursdays 3-5pm

• Sign up for course Piazza to communicate with staff and students
Acknowledgements

• The content of this course is largely provided by Professor Krste Asanovic.

• In addition, the original course material was inspired and created by previous MIT 6.823 and Berkeley CS252 courses: Arvind (MIT), Joel Emer (NVIDIA/MIT), James Hoe (CMU), John Kubiatowicz (UCB), and David Patterson (UCB).
FINI