CS359 Computer Architecture

Yanyan Shen

Department of Computer Science and Engineering

Agenda

- What is computer architecture?
  - Definition and trends
- How to design computer architecture?
- Why study computer architecture?
  - Contribute to the future
- How to study CS359?

Computer Architecture is…

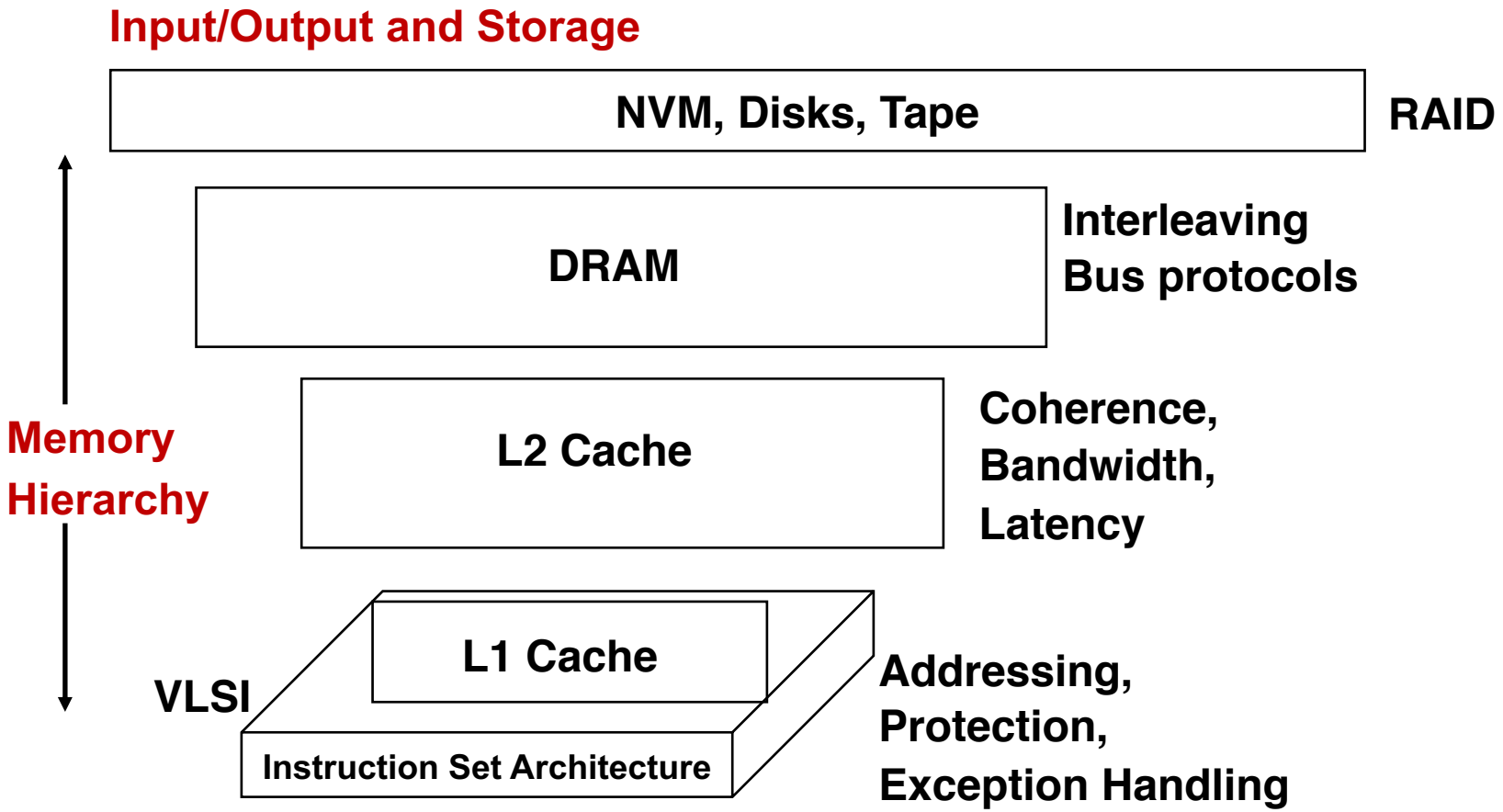
Term coined by Fred Brooks and colleagues at IBM:

"…the **structure of a computer** that a machine language programmer must understand to write a correct (timing independent) program for that machine."

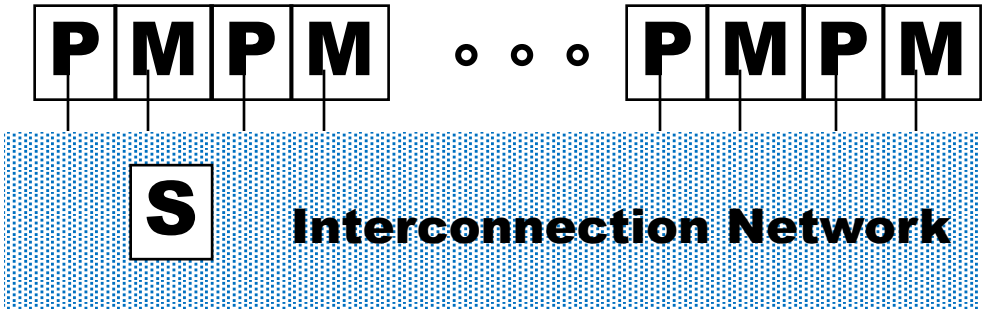
Amdahl, Blaauw, and Brooks, 1964

"Architecture of the IBM System 360",
IBM Journal of Research and Development

Computer Architecture Topics



Computer Architecture Topics



Processor-Memory-Switch

Shared Memory,
Message Passing,
Data Parallelism

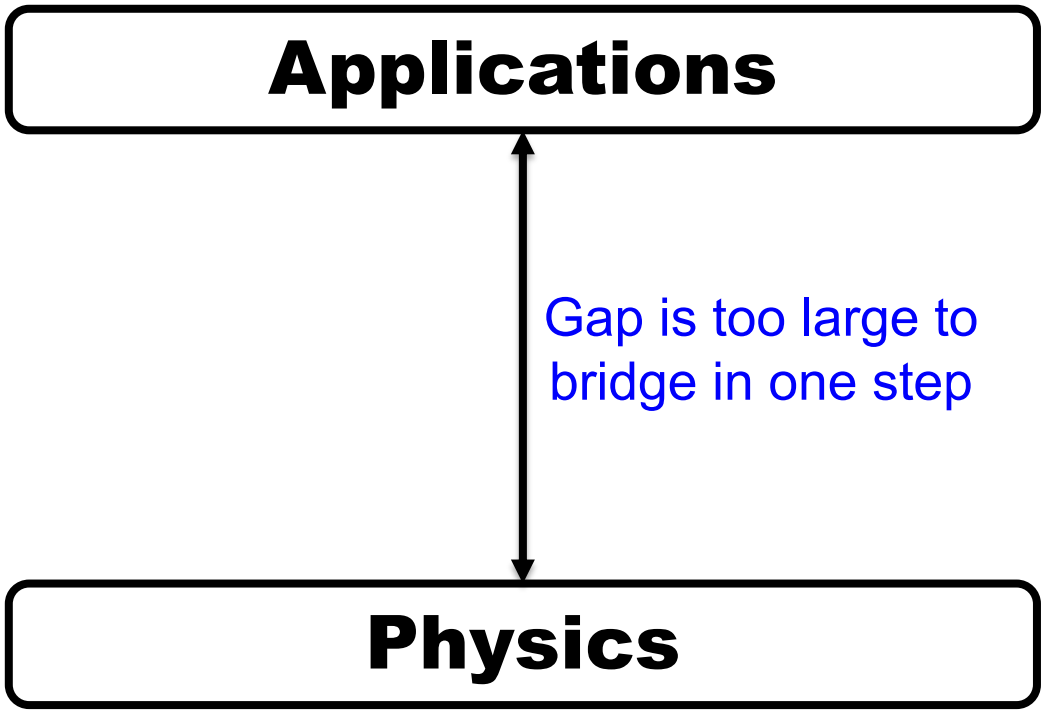
Network Interfaces

Topologies,
Routing,
Bandwidth,
Latency,
Reliability

Multiprocessors

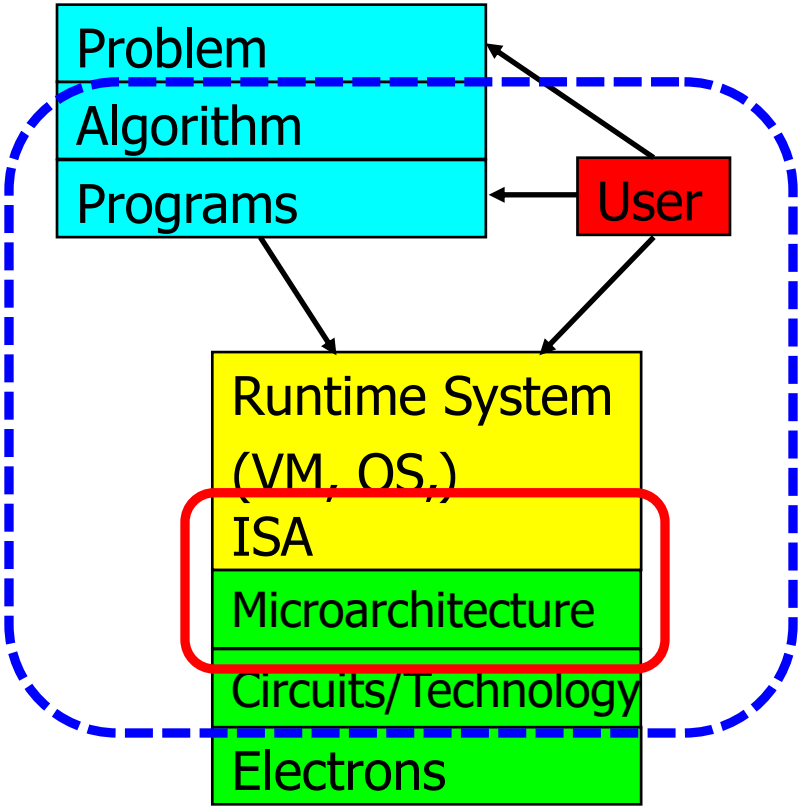
Networks and Interconnections

The Big Picture



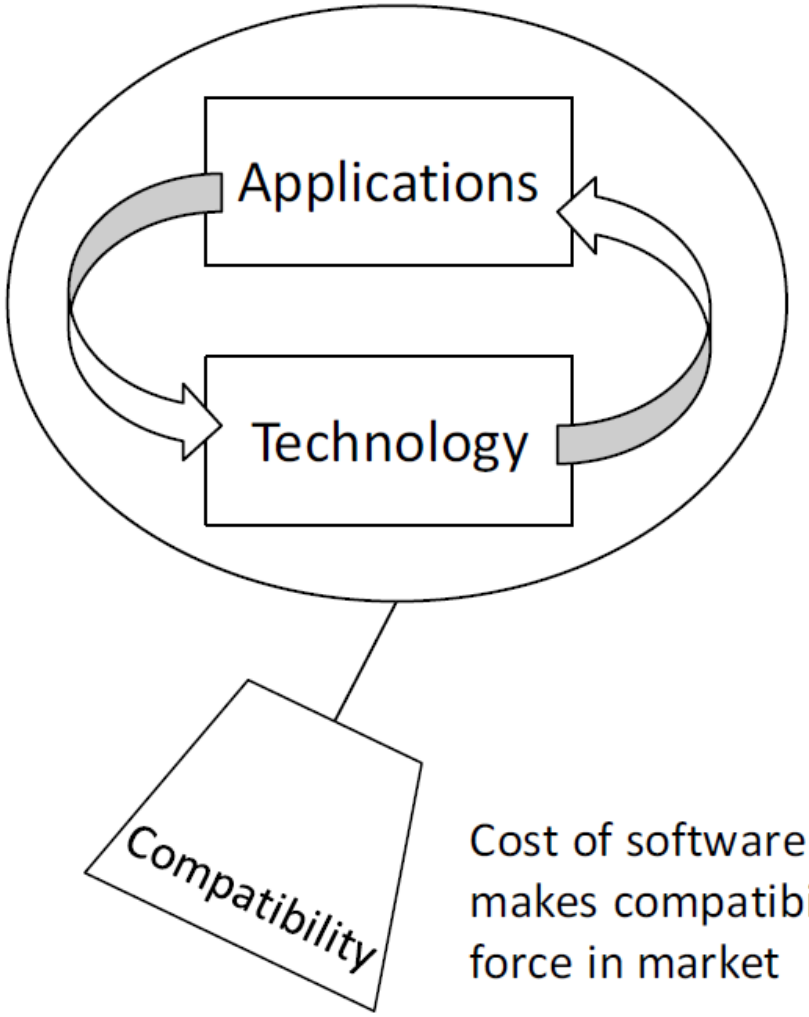
In its broadest definition, computer architecture is the *design of the abstraction layers* that allow us to implement information processing applications efficiently using available manufacturing technologies.

Levels of Transformation



Architecture continually changing

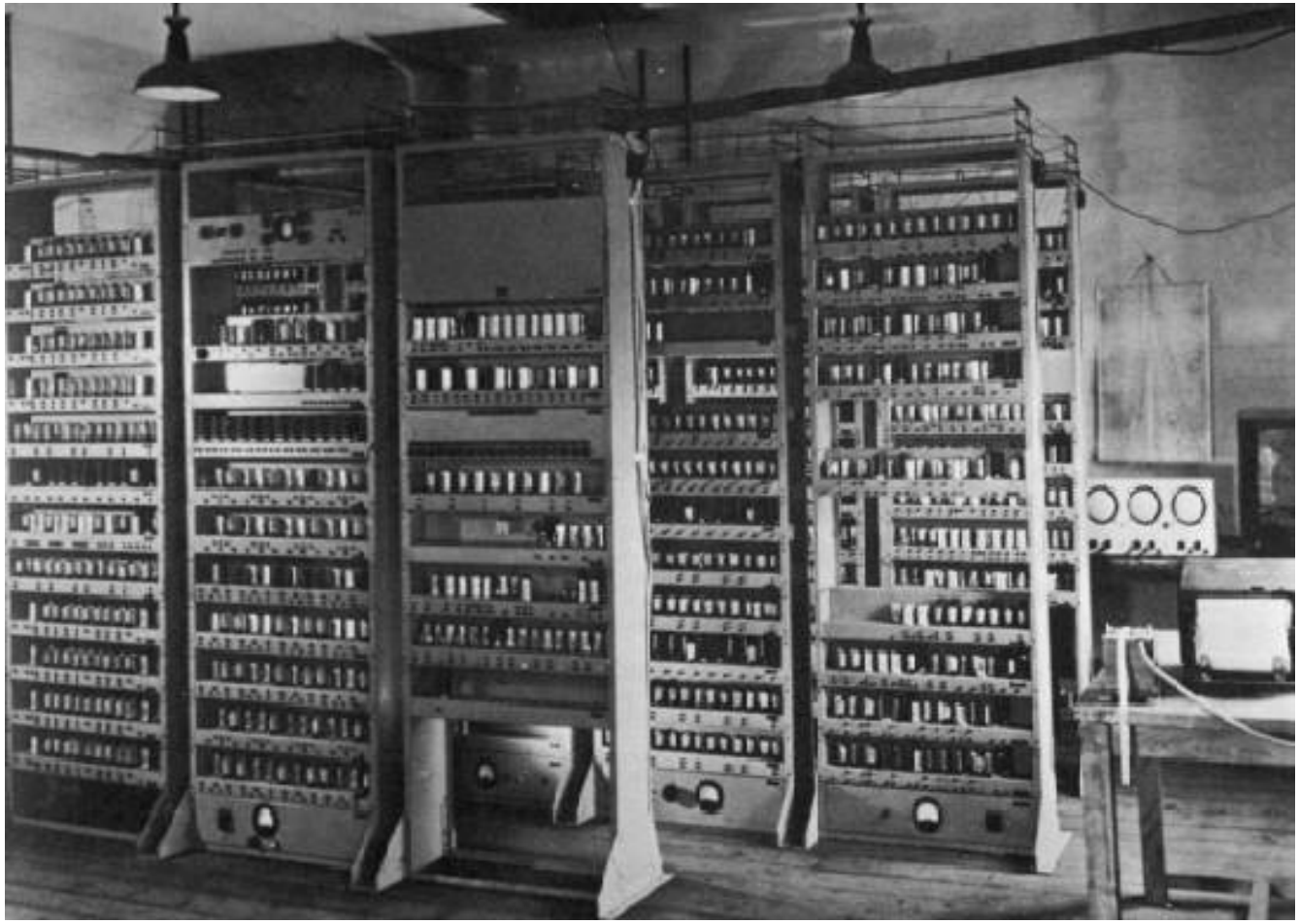
Applications provide need to improve technology, provide revenue to fund development



Improved technologies make new applications possible

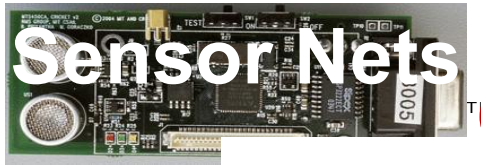
Cost of software development makes compatibility a major force in market

Computing Devices Then…



EDSAC, University of Cambridge, UK, 1949

Computing Devices Now



Cameras

Games

Set-top boxes

Media Players

Laptops

Servers

Routers

Robots

Smart phones

Automobiles

Supercomputers

Hardware Trends – at a glance

- Integrated circuit technology
  - Transistor density – 35% per year
  - Die size – 10-20% per year
- Semiconductor DRAM
  - Density – 40-60% per year (4x in 3-4 years)
  - Cycle time – 33% in 10 years
  - Bandwidth – 66% in 10 years
- Magnetic disk technology
  - Density – 100% per year
  - Access time – 33% in 10 years
- Network technology (depends on switches and transmission technology)
  - 10Mb-100Mb (10years), 100Mb-1Gb (5 years)
  - Bandwidth – doubles every year (for USA)

Software Trends – at a glance

- No longer just executing C/FORTRAN code
- Object Oriented Programming
- Middleware
  - Layer(s) between client and server applications
  - Hides complexity of client/server communications
- Complex applications
  - Computation-intensive
  - Communication-intensive

An Integrated Approach

- Computer architecture is not just about transistors, individual instructions, or particular implementations

- Functioning of the complete system
  - Hardware, runtime system, compiler, operating system, and application
  - New technology, new capability, new tradeoffs

Agenda

- What is computer architecture?
  - Definition and trends
- How to design computer architecture?
- Why study computer architecture?
  - Contribute to the future
- How to study CS359?

Design Goals & Constraints

- **Functional**
  - Needs to be correct
    - unlike software, difficult to update once deployed
  - What functions should it support
- **Reliable**
  - Does it ***continue*** to perform correctly?
  - Hard fault vs transient fault
  - Space satellites vs desktop vs server
- **High performance**
  - "Fast" is only meaningful in the context of a set of important tasks
  - Impossible goal: fastest possible design for all programs

Design Goals & Constraints

- **Low cost**
  - Per unit manufacturing cost (wafer cost)
  - Cost of making first chip after design (mask cost)
  - Design cost (huge design teams)
- **Low power/energy**
  - Energy in (battery life, cost of electricity)
  - Energy out (cooling and related costs)

Application Areas

- **Scientific**: weather prediction
  - Need: large memory, heavy-duty floating point
  - Examples: CRAY T3E, IBM BlueGene, Intel Xeon Phi, GPUs
- **Commercial**: database/web serving, e-commerce
  - Need: data movement, high memory + I/O bandwidth
  - Examples: IBM POWER, AMD Opteron, Intel Xeon
- **Desktop**: home office, multimedia, games
  - Need: integer, memory bandwidth, integrated graphics/network?
  - Examples: Intel Core i*, AMD Athlon

Application Areas

- **Mobile**: laptops, tablets, phones
  - Need: **low power**, integer performance, integrated wireless
  - Examples: Intel Core i*, Atom, AMD APUs, ARM
- **Embedded**: microcontrollers in cars, …
  - Need: low power, **low cost**
  - Examples: ARM chips, dedicated digital signal processors (DSPs)
  - 10 billion ARM chips sold in 2013

Challenges

- Balancing the relative importance of these goals
  - E.g., general-purpose vs application-specific
  - And the balance is constantly changing
    - No goal is absolutely important at expense of all others
  - Our focus: *performance*, only touch on cost, power, reliability

Challenges - tradeoff

Agenda

- What is computer architecture?
  - Definition and trends
- How to design computer architecture?
- Why study computer architecture?
  - Contribute to the future
- How to study CS359?

Why Study Computer Architecture?

- **Understand where computers are going**
  - Future capabilities drive the (computing) world
  - Real world-impact: no computer architecture → no computers!
- **Understand high-level design concepts**
  - The best architects understand all the levels
    - Devices, circuits, architecture, compiler, applications
- **Understand computer performance**
  - learn valid experimental methodologies

Why Study Computer Architecture?

- **Write better software**
  - The best software designers also understand hardware
  - Need to understand hardware to write fast software
- **Design hardware**
  - At Intel, AMD, IBM, ARM, Qualcomm, Oracle, NVIDIA, Samsung

Why Study Computer Architecture?

- CHANGE
- It's exciting!
- It has never been more exciting!
- It impacts every other aspect of electrical engineering and computer science

Agenda

- What is computer architecture?
  - Definition and trends
- How to design computer architecture?
- Why study computer architecture?
  - Contribute to the future
- How to study CS359?

Prerequisites

- Basic computer organization
  - Basic digital logic: gates, boolean functions, latches
  - Binary arithmetic: adders, hardware mul/div, floating-point
  - Basic datapath: ALU, register file, memory interface
  - Basic control: single-cycle control, microcode
  - Assembly language: e.g., MIPS
  - Pipelining
- Programming experience
  - No specific language required

Estimated Schedule

- Introduction (1 lecture, 90min)
- Fundamentals (3)
- Instructions – MIPS (2)
- Processor – single-cycle (2)
- Processor – multi-cycle (2)
- Pipelining and ILP theory (6)
- Memory hierarchy design (4)
- Data level parallelism(4)

Expectations

- See the "big ideas" in computer architecture
  - Pipelining, parallelism, caching, locality, abstraction, etc
- Exposure to examples of good (and some bad) engineering
- Understanding computer performance and metrics
- Design and implementation
  - some techniques on ISA, cache, architecture

- *Raise questions…*

Coursework & Grading

(Rough, not the final decision…)

- Quizzes and participants: 5%
- English notes: 10%
  - 2 people per group, 10min quick review at the beginning of the classes
- Assignments: 15%
- Projects: 20%
- Final exam: 50%

All homework assignments and project files should be submitted before the deadline.

**** Late submissions will NOT be accepted.*

Plagiarism

- Discussions about homework in small groups are encouraged.
- However, homework must be written up individually and independently. Projects are to be completed individually unless otherwise specified.
- NO cheating is allowed!