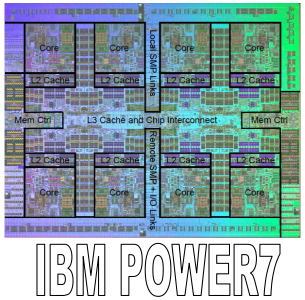
Lecture 1

Advanced Computer Architecture



Fall 2016

Pejman Lotfi-Kamran

Adapted from slides originally developed by Profs. Falsafi, Hill, Hoe, Lipasti, Shen, Smith, Sohi, and Vijaykumar of Carnegie Mellon University, EPFL, Purdue University, and University of Wisconsin.

Where Are We?

Fr	Sa	Su	Mo	Tu
	27-Shahrivar		29-Shahrivar	
	3-Mehr		5-Mehr	
	10-Mehr		12-Mehr	
	17-Mehr		19-Mehr	
	24-Mehr		26-Mehr	
	1-Aban		3-Aban	
	8-Aban		10-Aban	
	15-Aban		17-Aban	
	22-Aban		24-Aban	
	29-Aban		1-Azar	
	6-Azar		8-Azar	
	13-Azar		15-Azar	
	20-Azar		22-Azar	
	27-Azar		29-Azar	
	4-Dey		6-Dey	

Class intro
- Logistics
- Grades
- Topical intro

Monday:
- How to measure computer performance

Homework 0
- Due Monday Shahrivar 29th

Who Should Take This Course?

Graduate students (MS/PhD)
1. Computer architects to be
2. Computer system designers
3. Those interested in computer systems

Required Background
- Introduction to Computer Architecture

About the Course
- Heavily discussion oriented
- With emphasis on cutting-edge issues/research

Feedback
- Individual feedback upon request

Where do I find info about this course?

Anything you ever wanted to know:

CW

E.g.,

Where to go and when

Syllabus: grading, what the course assumes, etc.

Class notes, homework, project description etc.

Logistics for the Course

Class times

- Lectures: Sa 9:00-10:30am, 006
  Mo 9:00-10:30am, 006

Lecturer

- Pejman Lotfi-Kamran
- Research Interests
  - Memory systems
  - Interconnection networks
  - Approximate computing

TA

- Mohammad Bakhshalipour

Components

Text

- *Computer Architecture: A Quantitative Approach, 5th ed.*
- recommended: *Readings in Computer Architecture*

Homework

- list of papers: classic + state of the art
- short written review per paper

Programming assignments (individual write-up, group discussion okay)

Quiz

Project (Optional)

- mostly original research
- groups of two

Homework

Weekly paper reading

To answer questions related to a paper

The questions will be posted on the web

Due at the beginning of the class

I do not accept late homework

Homework 0 due Monday Shahrivar 29th

- My way to learn about you
- You will not receive grades for any subsequent homework unless you complete homework 0

Programming Assignments

Simple cache simulator

Starts in one week

Assignment description will be posted

Deliver a short write-up

Project (Optional)

- Find a partner
- Project
  - I will hand out a proposed list of projects
  - You can propose a project
  - You can reproduce results from recent publications
    → Top original results will be submitted to a conference
- Starts in two weeks
- Will have milestones for the project
- Final report (and presentation/poster)

Announcements

All announcements appear on the Class mailing list

Give us your mailing address (in Homework 0)

All graded homework, projects, exams

Grading

Grade breakdown

Homework:	20%
Programming assignments:	25%
Quiz	10%
Midterm:	20%
Final:	25%
Project:	25% (Extra points)

Participation + Discussion count

Academic Dishonesty

Group studies ok

Homework solution/code must be individual effort

What is not ok (not kosher):

Group discussion of homework solution

Copying homework solution/code from each other or from prior semesters

Class Meeting Time

Notice!

- each lecture is 90 minutes long
- class meets between 9:00am – 10:30am on Sa-Mo
- office hours: By Appointment (usually after the lectures)
-

Required Background

- Computer Architecture
- Basic OS
- C/C++ programming

What Is Computer Architecture?

"The term *architecture* is used here to describe the attributes of a system as seen by the programmer, i.e., the conceptual structure and functional behavior as distinct from the organization of the dataflow and controls, the logic design, and the physical implementation."

Gene Amdahl, IBM Journal of R&D, April 1964

Architecture, Organization, Implementation

Computer architecture: SW/HW interface

- instruction set
- memory management and protection
- interrupts and traps
- floating-point standard (IEEE)

Organization: also called microarchitecture

- number/location of functional units
- pipeline/cache configuration
- datapath connections

Implementation:

- low-level circuits

What Is This Course All About?

State-of-the-art computer hardware design

Microprocessor architecture

Memory architecture

Multiprocessors

System-level interconnect architecture

CMOS issues (wires, power, bit error, yield, etc.)

Blue Sky

Roadmap for the Course

Performance Measurement	Evaluation
Basic Caches	Parallel programming
Low-Miss-Ratio Caches	Cache Coherence
High-B/W Caches	Memory Consistency
Prefetching	Synchronization
Virtual Memory	Interconnect
DRAM	Storage
Pipelining	Scaling
Exploiting ILP Dynamically	Servers
Frontend	Data centers/Supercomputers

Take Advanced Computer Architecture!

Computer Architecture Curriculum

Introduction to computer architecture
Advanced computer architecture
Advanced Microarchitecture
Advanced multiprocessor architecture
Advanced topics in memory systems
Topics in datacenter design
Proposals for future architectures

Related areas:

- circuits: VLSI, digital circuit design, CAD
- systems: compilers, OS, database systems, networks, embedded computing, fault-tolerant computing
- evaluation: queuing theory, analysis of variance, confidence intervals, etc.

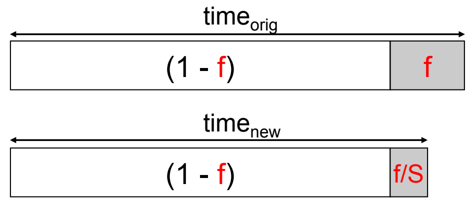
Amdahl's Law

Speedup= $time_{without\ enhancement} / time_{with\ enhancement}$

Suppose an enhancement speeds up a fraction f of a task by a factor of S

$time_{new} = time_{orig} \cdot ((1-f) + f/S)$

$S_{overall} = 1 / ((1-f) + f/S)$

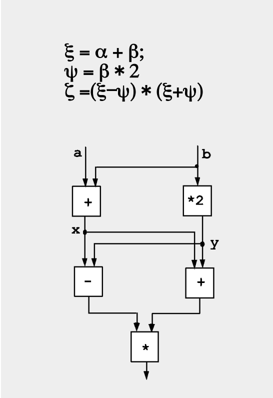


Parallelism: Work and Critical Path

Parallelism - the amount of independent sub-tasks available

Work=T_1 - time to complete a computation on a sequential system

Critical Path=T_∞ - time to complete the same computation on an infinitely-parallel system



Average Parallelism

$P_{avg} = T_1 / T_\infty$

For a p wide system

$T_p \geq \max\{ T_1/p, T_\infty \}$

$P_{avg} >> p \Rightarrow T_p \approx T_1/p$

Locality Principle

One's recent past is a good indication of her/his near future.

- Temporal Locality: If you looked something up, it is very likely that you will look it up again soon
- Spatial Locality: If you looked something up, it is very likely you will look up something nearby next

Locality == Patterns == Predictability

Converse:

Anti-locality : If you haven't done something for a very long time, it is very likely you won't do it in the near future either

Memoization

Dual of temporal locality but for computation

If something is expensive to compute, you might want to remember the answer for a while, just in case you will need the same answer again

Why does memoization work??

Real life examples:

- whatever results of work you will soon reuse

Examples

- Trace caches

Amortization

overhead cost : one-time cost to set something up

per-unit cost : cost for per unit of operation

total cost = overhead + per-unit cost x N

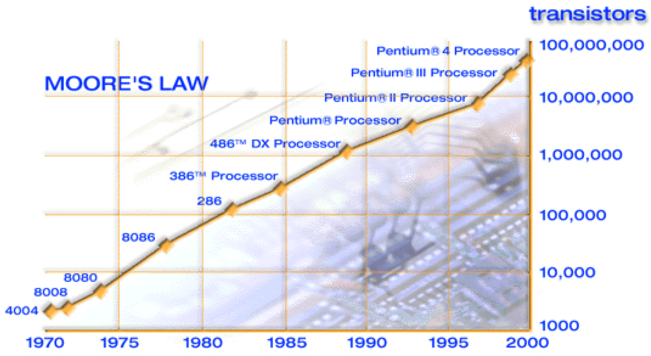
It is often okay to have a high overhead cost if the cost can be distributed over a large number of units

$\Rightarrow$ low the *average cost*

average cost = total cost / N

= (overhead / N) + per-unit cost

Why Study Computer Architecture?



http://www.intel.com/research/silicon/mooreslaw.htm

Why Study Computer Architecture?

Answer #1: Optimize cost/performance as technology changes

What do these intervals have in common?
- 1776—1997 (222 years)
- 1998—1999 (2 years)

Absolute speed improvement of computers comparable!
- If performance improves by 50%, $1.5^2 = 2.25$

Technology	Annual Improvement
Transistor count	25%
Transistor speed	20%-25%
DRAM density	60%
DRAM speed	4%
Disk density	25%
Disk speed	4%

Why Study Computer Architecture?

Answer #2: Innovation built into performance trends

Initially, transistor counts limited performance
- ~35% performance improvement per year

Later, larger transistor counts ➟ advanced microarchitecture
- > 50% performance improvement per year
- the added growth due to implementation/organization

1996 performance	Clock (MHz)	Performance (SPECInt)
1989 projections	150	2.5
Actual	200	10

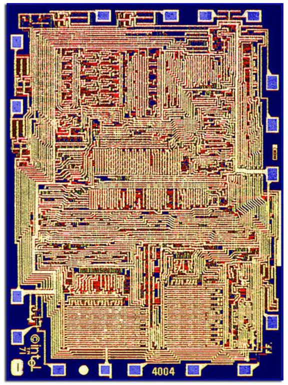
This course will cover:
- technologies enabling this performance growth
- technologies sustaining future growth

Why Study Computer Architecture?

Answer #3: User requirements change rapidly

Previously infeasible solutions become ubiquitous products!
- multimedia
- entertainment
- portable computing
- virtual reality/wearable computing
- web/network computing
- whatever you can think of….

Intel 4004, circa 1971



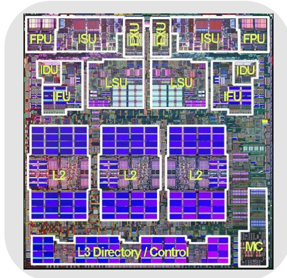
The first single chip CPU

- 4-bit processor for a calculator.
- 1K data memory
- 4K program memory
- 2,300 transistors
- 16-pin DIP package
- 740kHz (eight clock cycles per CPU cycle of 10.8 microseconds)
- ~100K OPs per second

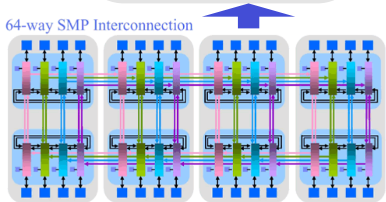
Molecular Expressions: Chipshots

IBM Power 5, circa 2006

Powe5 Chip



Power5 System



Performance leader

- Two 64-bit processors
  - Four threads
- 2 MByte in cache!!
- 276 million transistor
- 2 GHz, issue up to 10 instructions per cycle

In ~30 years, about 100,000 fold growth in transistor count and chip performance!

Any info missing? Ask now…

Next Lecture

How to measure and report computer performance and cost?