CS 3410: Computer System Organization and Programming

Prof. Hakim Weatherspoon

CS 3410, Spring 2015

Computer Science

Cornell University

“Sometimes it is the people that no one imagines anything of

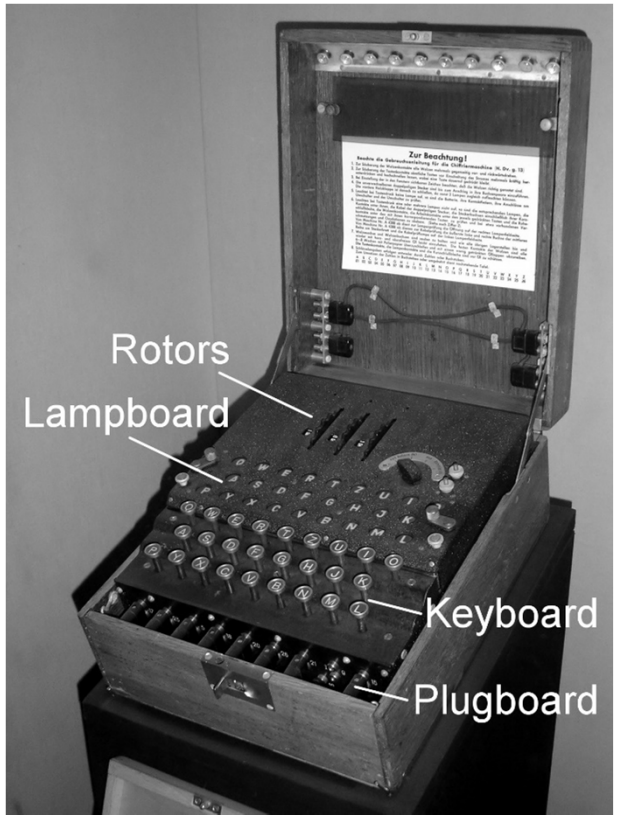
who do the things that no one can imagine”

--quote from the movie The Imitation Game

“Can machines think?”

-- Alan Turing, 1950

Computing Machinery and Intelligence

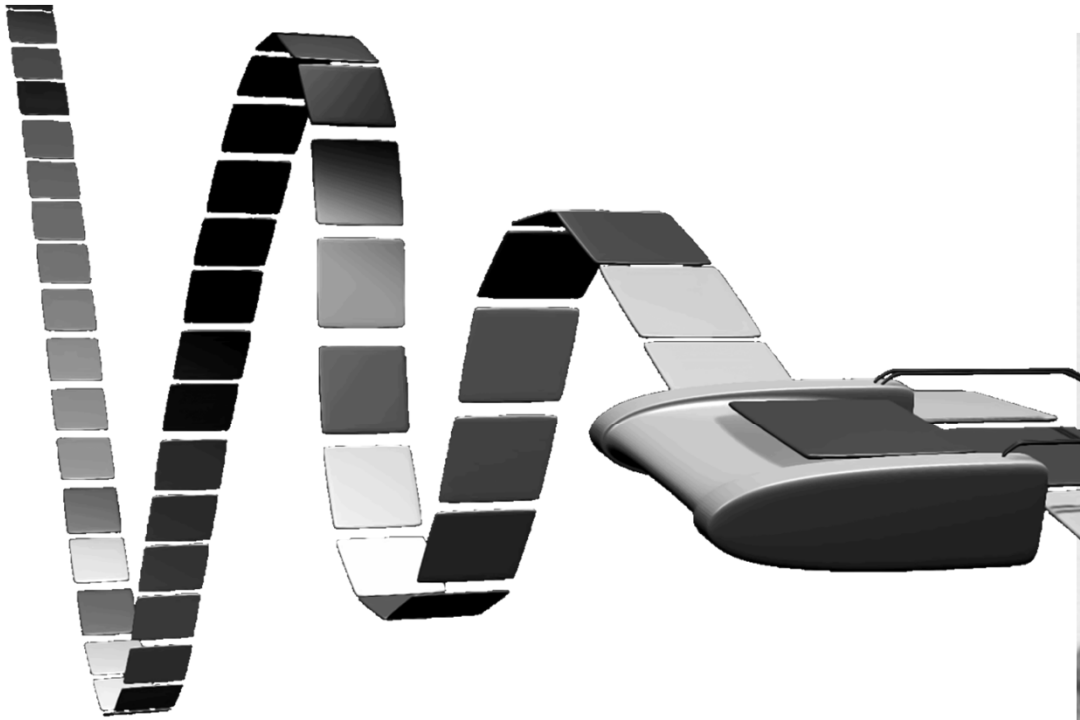


Enigma machine

Used by the Germans during World War II to encrypt and exchange secret messages

The Bombe

used by the Allies to break the German Enigma machine during World War II



Turing Machine
1936

Alan Turing

Course Objective

Bridge the gap between hardware and software

- How a processor works
- How a computer is organized

Establish a foundation for building higher-level applications

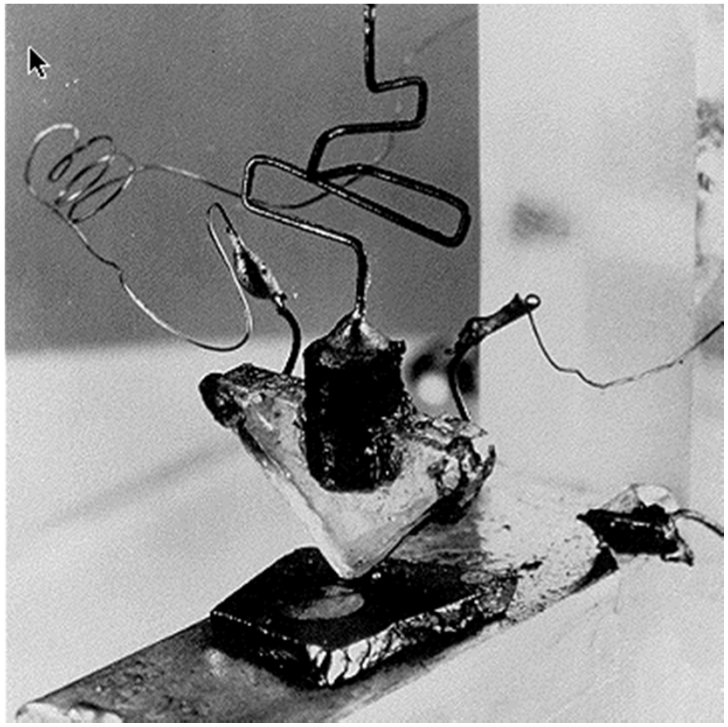
- How to understand program performance
- How to understand where the world is going

Where did it begin?

Electrical Switch

- On/Off
- Binary

Transistor



The first transistor on a workbench at AT&T Bell Labs in 1947

Moore's Law

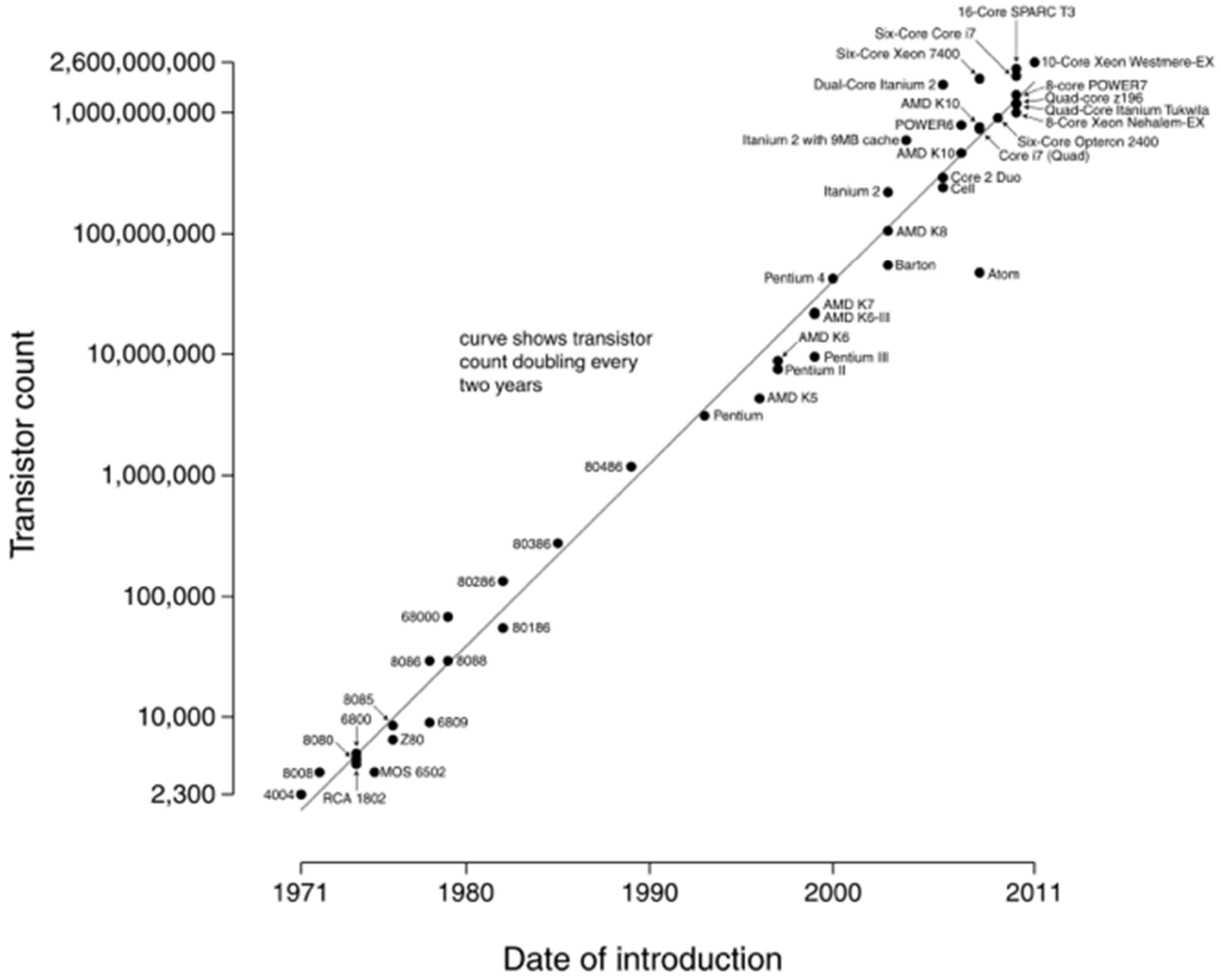
1965

- number of transistors that can be integrated on a die would double every 18 to 24 months (i.e., grow exponentially with time)

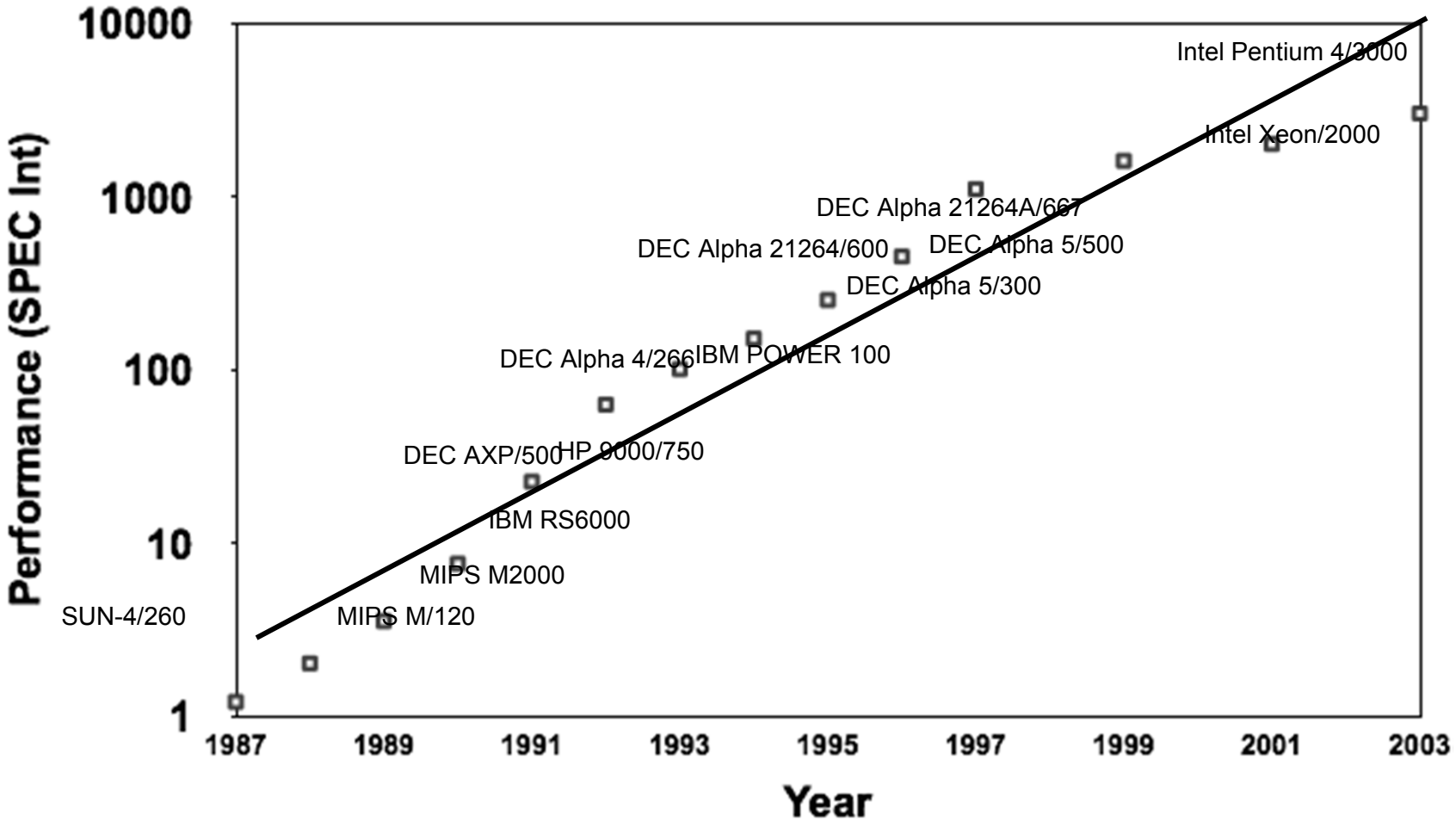
Amazingly visionary

- 2300 transistors, 1 MHz clock (Intel 4004) - 1971
- 16 Million transistors (Ultra Sparc III)
- 42 Million transistors, 2 GHz clock (Intel Xeon) – 2001
- 55 Million transistors, 3 GHz, 130nm technology, 250mm^2 die (Intel Pentium 4) – 2004
- 290+ Million transistors, 3 GHz (Intel Core 2 Duo) – 2007
- 721 Million transistors, 2 GHz (Nehalem) - 2009
- 1.4 Billion transistors, 3.4 GHz Intel Haswell (Quad core) – 2013

Microprocessor Transistor Counts 1971-2011 & Moore's Law



Processor Performance Increase



Moore's Law

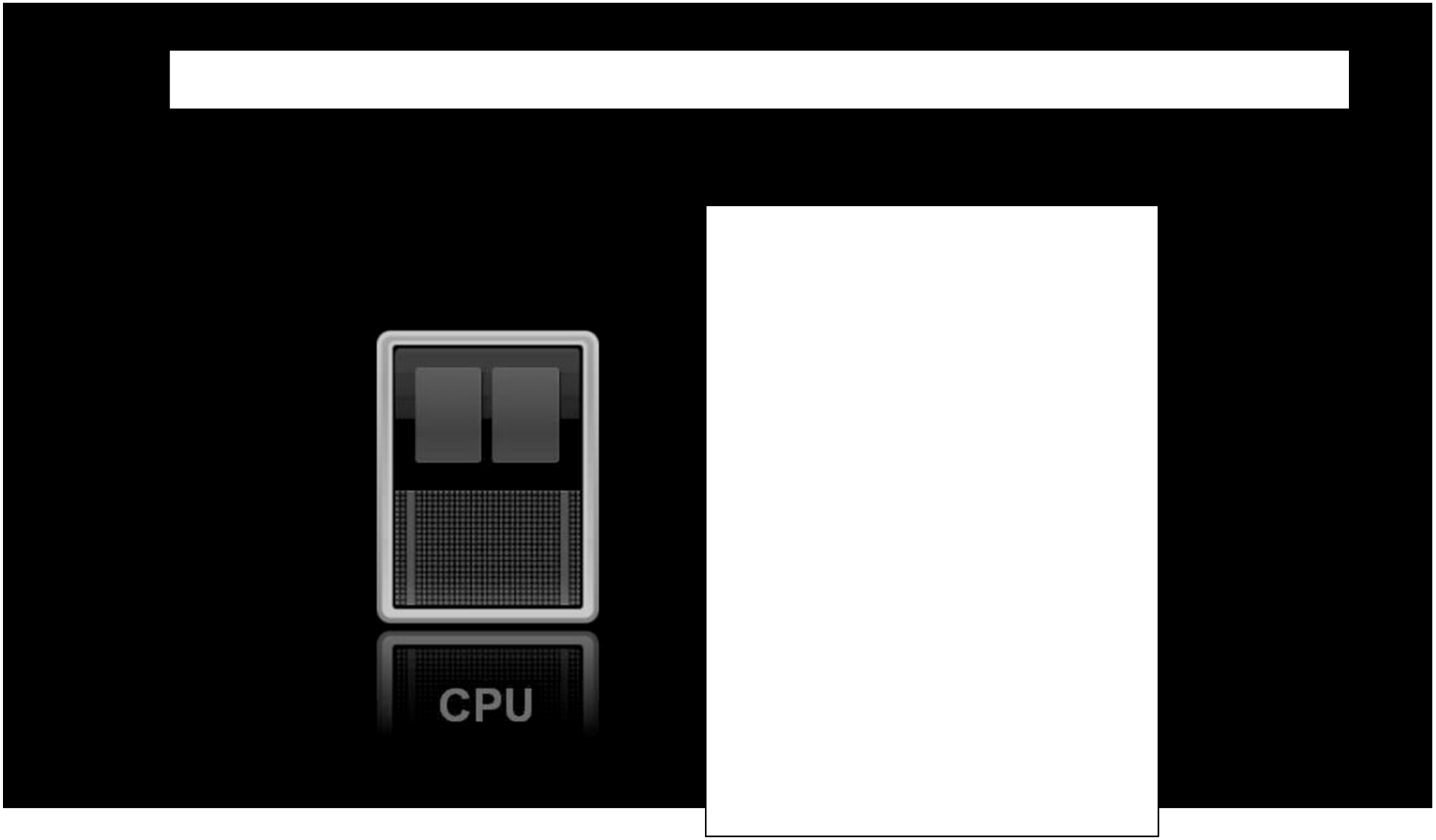
1965

- number of transistors that can be integrated on a die would double every 18 to 24 months (i.e., grow exponentially with time)

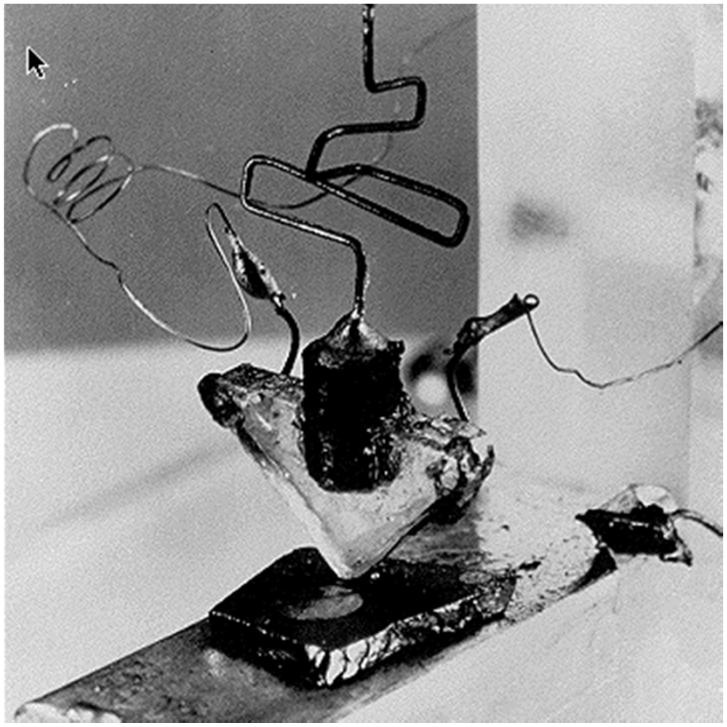
Amazingly visionary

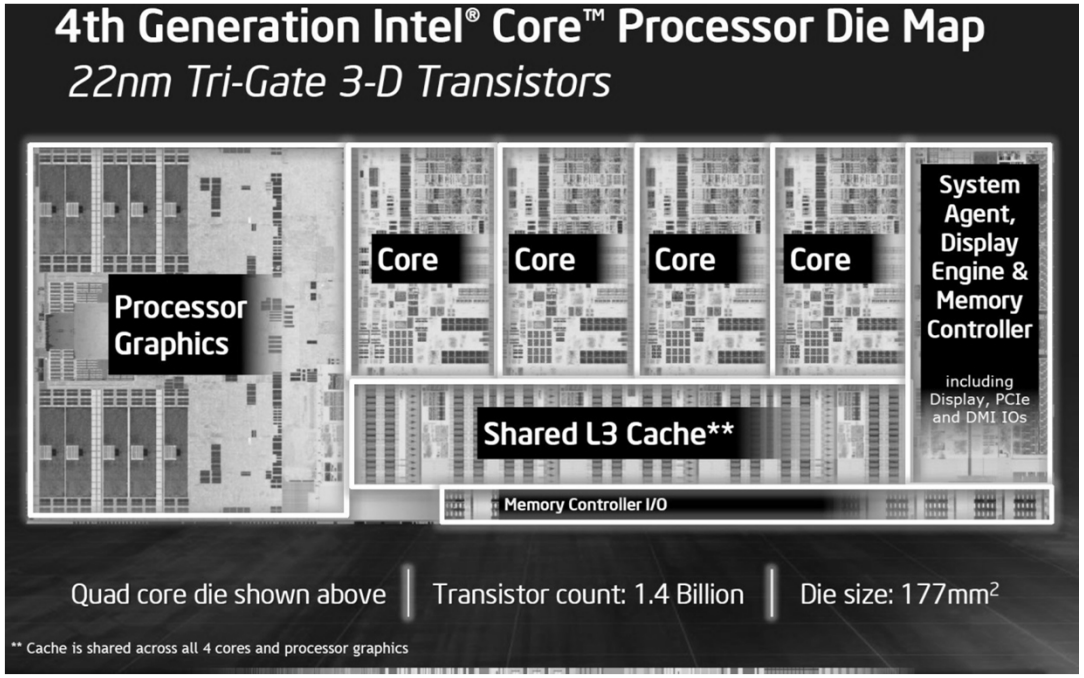
- 2300 transistors, 1 MHz clock (Intel 4004) - 1971
- 16 Million transistors (Ultra Sparc III)
- 42 Million transistors, 2 GHz clock (Intel Xeon) – 2001
- 55 Million transistors, 3 GHz, 130nm technology, 250mm^2 die (Intel Pentium 4) – 2004
- 290+ Million transistors, 3 GHz (Intel Core 2 Duo) – 2007
- 721 Million transistors, 2 GHz (Nehalem) - 2009
- 1.4 Billion transistors, 3.4 GHz Intel Haswell (Quad core) – 2013

Parallelism



Then and Now

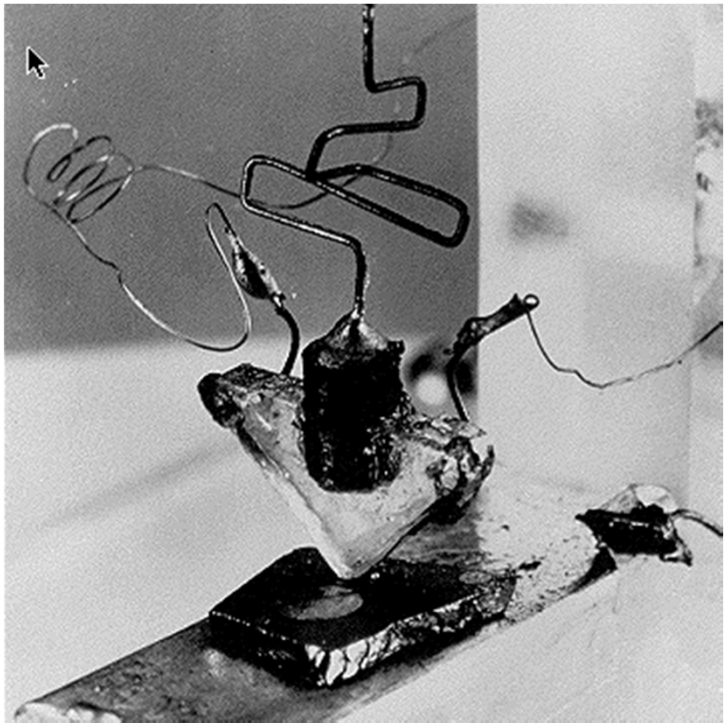


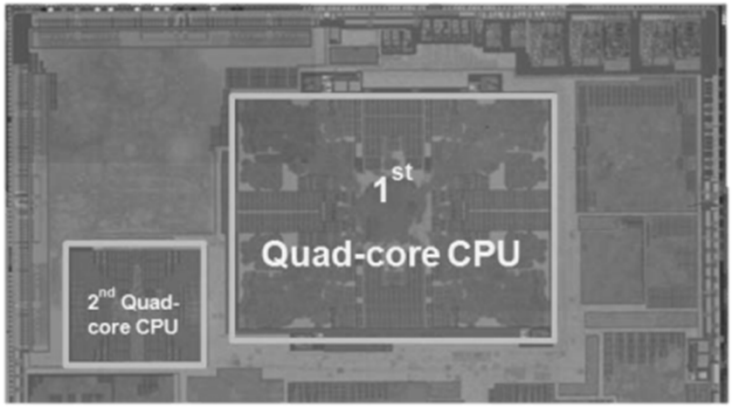


http://techguru3d.com/4th-gen-intel-haswell-processors-architecture-and-lineup/

- The first transistor
  - One workbench at AT&T Bell Labs
  - 1947
  - Bardeen, Brattain, and Shockley
- An Intel Haswell
  - 1.4 billion transistors
  - 177 square millimeters
  - Four processing cores

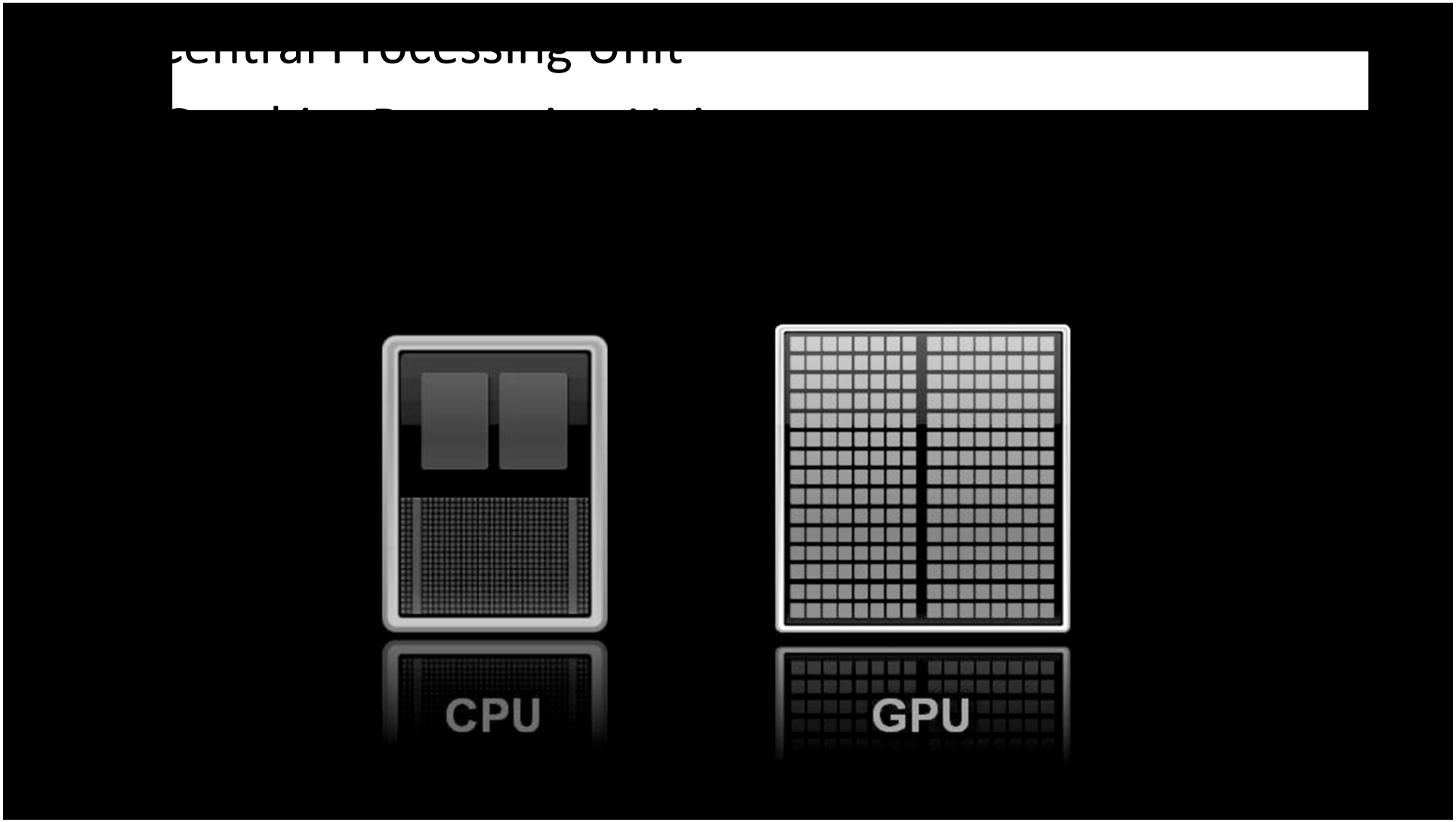
Then and Now



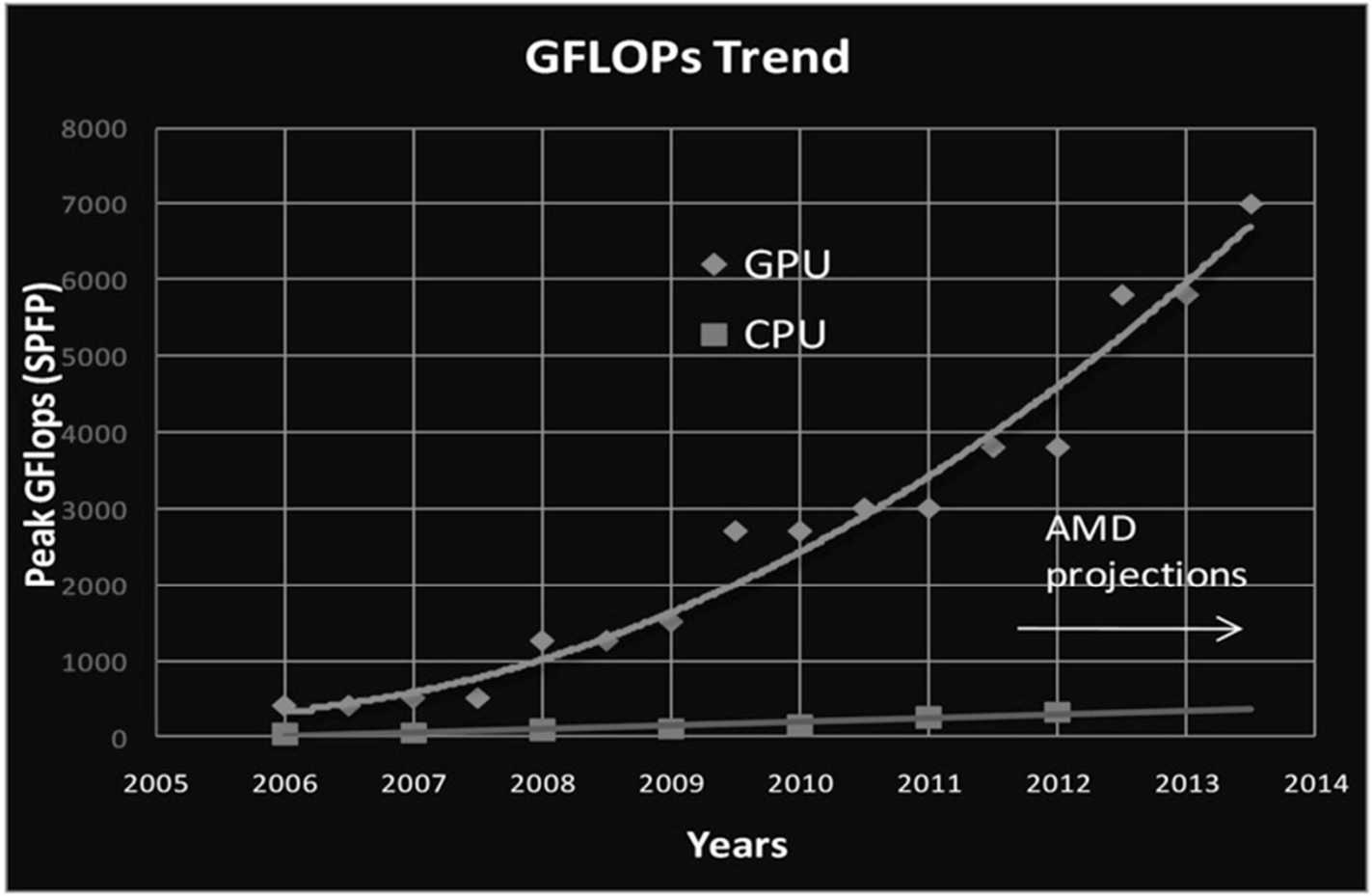


- The first transistor
  - One workbench at AT&T Bell Labs
  - 1947
  - Bardeen, Brattain, and Shockley
- Galaxy Note 3
  - 8 processing cores

Parallelism

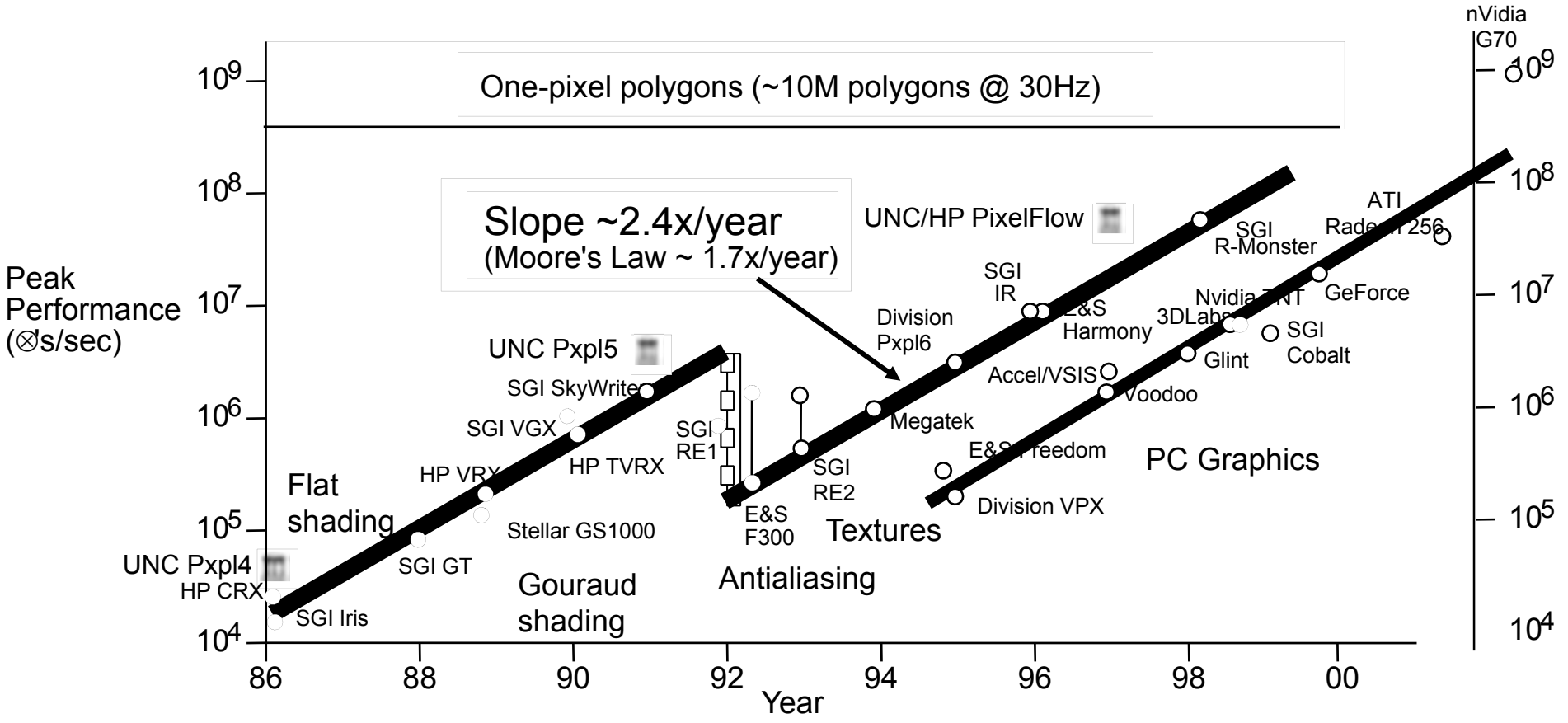


GPU-type computation offers higher GFlops



(Source: Sam Naffziger, AMD)

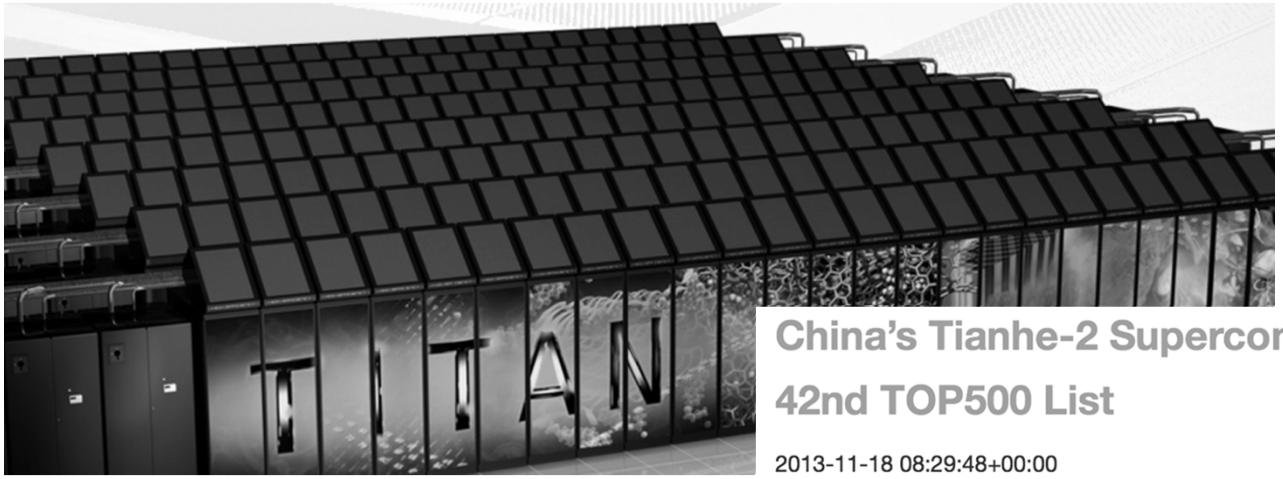
GPUs: Faster than Moore's Law



Graph courtesy of Professor John Poulton (from Eric Haines)

Supercomputers

- Petaflops (10^{15})
  - GPUs/multicore/100s-1000s cores



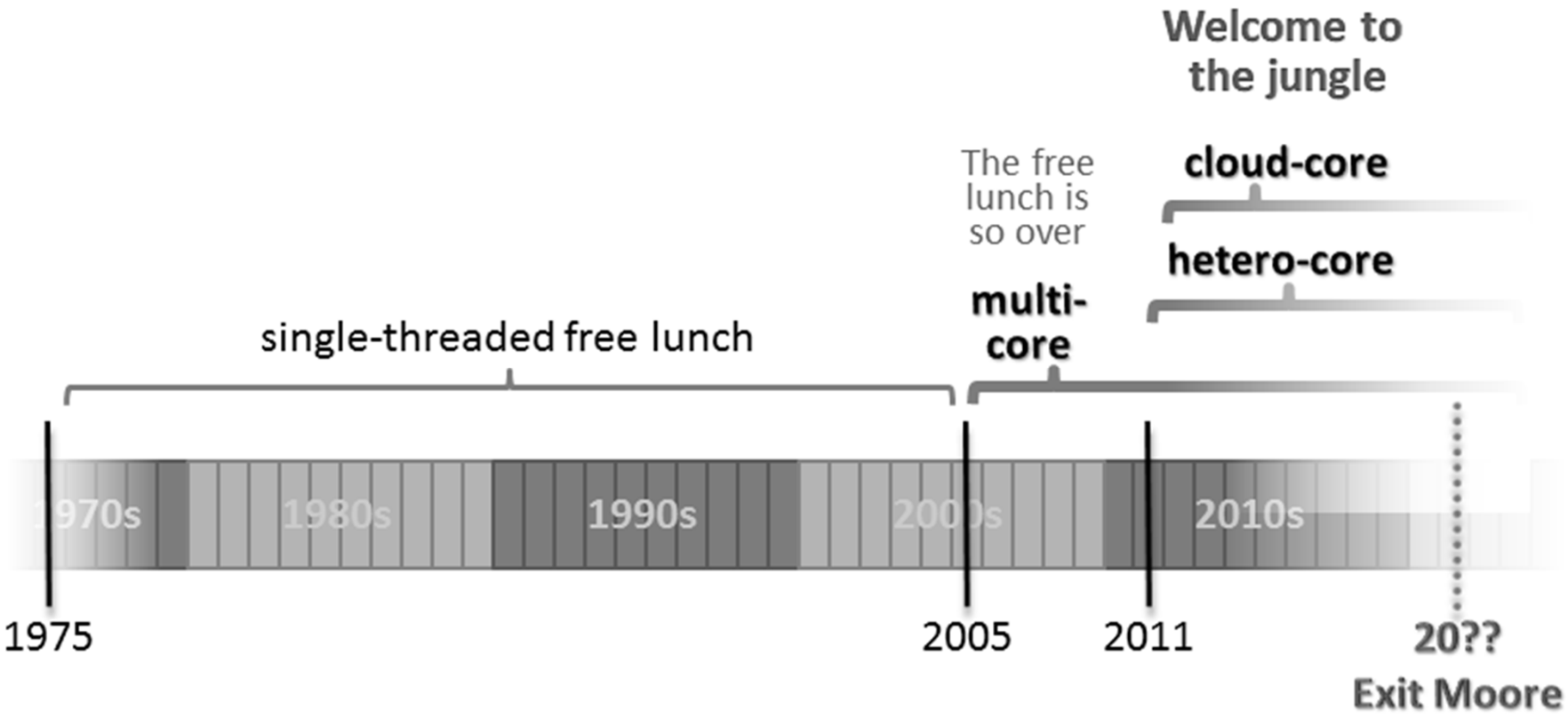
China's Tianhe-2 Supercomputer Maintains Top Spot on 42nd TOP500 List

2013-11-18 08:29:48+00:00

MANNHEIM, Germany; BERKELEY, Calif.; and KNOXVILLE, Tenn.—Tianhe-2, a supercomputer developed by China's National University of Defense Technology, retained its position as the world's No. 1 system with a performance of 33.86 petaflop/s (quadrillions of calculations per second) on the Linpack benchmark, according to the 42nd edition of the twice-yearly TOP500 list of the world's most powerful supercomputers. The list was announced Nov. 18 at the SC13 conference in Denver, Colo.

Titan, a Cray XK7 system installed at the Department of Energy's (DOE) Oak Ridge National Laboratory, remains the No. 2 system. It achieved 17.59 Pflop/s on the Linpack benchmark. Titan is one of the most energy efficient systems on the list consuming a total of 8.21 MW and delivering 2.143 gigaflops/W.

Sequoia, an IBM BlueGene/Q system installed at DOE's Lawrence Livermore National Laboratory, is again the No. 3 system. It was first delivered in 2011 and achieved 17.17 Plop/s on the Linpack benchmark.


Welcome to the jungle
The free lunch is so over
cloud-core
hetero-core
multi-core
single-threaded free lunch
1970s
1980s
1990s
2000s
2010s
1975
2005
2011
20??
Exit Moore

Why?

- Parallelism

- Pipelining

Programmable Hardware

- Started in 1999
- Flexible, programmable
  - Vertex, Geometry, Fragment Shaders
- And much faster, of course
  - 1999 GeForce256: 0.35 Gigapixel peak fill rate
  - 2001 GeForce3: 0.8 Gigapixel peak fill rate
  - 2003 GeForceFX Ultra: 2.0 Gigapixel peak fill rate
  - ATI Radeon 9800 Pro : 3.0 Gigapixel peak fill rate
  - 2006 NV60: ... Gigapixel peak fill rate
  - 2009 GeForce GTX 285: 10 Gigapixel peak fill rate
  - 2011
    - GeForce GTC 590: 56 Gigapixel peak fill rate
    - Radeon HD 6990: 2x26.5
  - 2012
    - GeForce GTC 690: 62 Gigapixel/s peak fill rate

Course Objective

Bridge the gap between hardware and software

- How a processor works
- How a computer is organized

Establish a foundation for building higher-level applications

- How to understand program performance
- How to understand where the world is going

How class is organized

Instructor: Hakim Weatherspoon

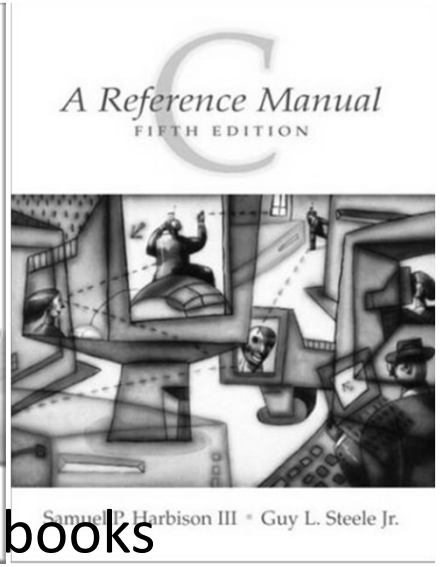
(hweather@cs.cornell.edu)

Lecture:

- Tu/Th 1:25-2:40
- Statler Auditorium

Lab sections:

- Start next week
- Carpenter 104 (Blue room)
- Carpenter 235 (Red room)
- Upson B7



books

Suggested Textbook

Who am I?

Prof. Hakim Weatherspoon

- (Hakim means Doctor, wise, or prof. in Arabic)

Career Path

- Undergrad→grad→post-doc→professor
- Washington → Berkeley → Cornell → Cornell

Who am I?

The promise of the Cloud

- *ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.* NIST Cloud Definition

Who am I?

The promise of the Cloud

- *ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.* NIST Cloud Definition

Who am I?

The promise of the Cloud

- *ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.* NIST Cloud Definition

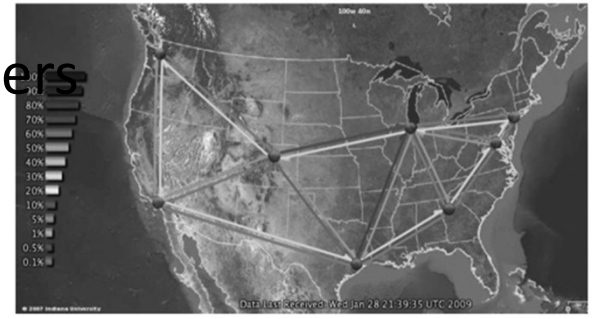
Requires fundamentals in distributed systems

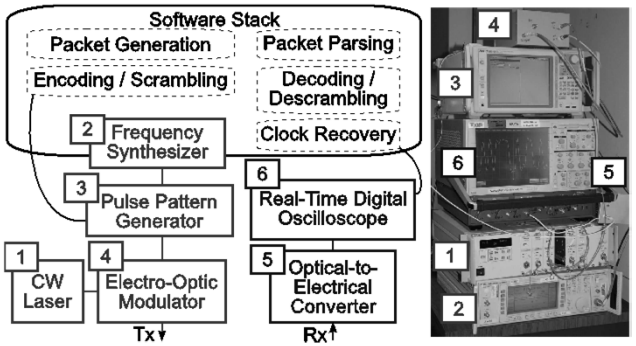
- Networking
- Computation
- Storage

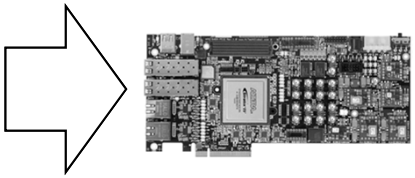
Who am I?

Cloud computing/storage

- Optimizing a global network of data centers







Course Staff

cs-3410-staff-l@cornell.edu

Lab/Homework TA's

- Deniz Altinbuken <deniz@cs.cornell.edu> (PhD)
- Adam Campbell <atc89@cornell.edu> (PhD)
- Praveen Kumar <praveenk@cs.cornell.edu> (PhD)
- Vishal Shrivastav <vishal@cs.cornell.edu> (PhD)
- Rob Mcguinness <jrm369@cornell.edu> (MEng)
- Akshay Navalakha <adn47@cornell.edu> (MEng)
- Andrew Weymouth <asw75@cornell.edu> (MEng)
- Naman Agarwal <na298@cornell.edu>
- Maxwell Dergosits <mad293@cornell.edu>
- Antoine Pourchet <app63@cornell.edu>
- Gary Zibrat <gdz4@cornell.edu>
- Ari Karo <aak82@cornell.edu>
- Amy Chen <yc624@cornell.edu>
- Stephanie Guo <lg399@cornell.edu>
- Kylar Henderson <kdh59@cornell.edu>
- Megan Carpenter <mnc29@cornell.edu>
- Rebecca Stambler <rls462@cornell.edu>
- Yogisha Dixit <yad4@cornell.edu>
- Mahak Goel <mg785@cornell.edu>
- Spandan Agrawal <sga35@cornell.edu>
- Adwit Tumuluri <at627@cornell.edu>
- Daniel Liu <dl596@cornell.edu>
- Rishab Gupta <rsg246@cornell.edu>
- Oscar Pacheco <ofp3@cornell.edu>
- Lucas Derraugh <ldd49@cornell.edu>
- Brian Wang <bhw45@cornell.edu>
- Anthony Lin <al744@cornell.edu>
- Charles Lai <cjl223@cornell.edu>
- Jonathan Behrens <jkb229@cornell.edu>

Administrative Assistant:

- Jessica Depew <jd648@cs.cornell.edu>

Pre-requisites and scheduling

CS 2110 is required (Object-Oriented Programming and Data Structures)

- Must have satisfactorily completed CS 2110
- *Cannot take CS 2110 concurrently with CS 3410*

CS 3420 (ECE 3140) (Embedded Systems)

- Take either CS 3410 ***or*** CS 3420
  - both satisfy CS and ECE requirements
- *However, Need ENGRD 2300 to take CS 3420*

CS 3110 (Data Structures and Functional Programming)

- Not advised to take CS 3110 and 3410 together

Pre-requisites and scheduling

CS 2043 (UNIX Tools and Scripting)

- 2-credit course will greatly help with CS 3410.
- Meets Mon, Wed, Fri at 11:15am-12:05pm in Hollister (HLS) B14
- Class started yesterday and ends March 5th

CS 2022 (Introduction to C) and CS 2024 (C++)

- 1 to 2-credit course will greatly help with CS 3410
- *Unfortunately, offered in the fall, not spring*
- Instead, we will offer a primer to C during lab sections
  and include some C questions in homeworks

Schedule (subject to change)

Week	Date (Tue)	Lecture#	Lecture Topic	HW	Prelim Evening	Lab Topic	Lab/Proj
1	22-Jan	1	Intro				
	27-Jan	2	Logic & Gates			Logisim	Lab 0: (IN-CLASS) Adder/Logisim intro
2		3	Numbers & Arithmetic				
	3-Feb	4	State & FSMs			ALU/Design Docs	Lab 1: ALU Handout
3		5	Memory				
	10-Feb	6	Simple CPU	HW1: Logic, Gates, Numbers, & Arithmetic,		FSM	Lab 2: (IN-CLASS) FSM Handout
4		7	CPU Performance & Pipelines	FSMs, Memory, CPU, Performance,			
	17-Feb		Winter Break	and pipelined MIPS		MIPS	Proj 1: MIPS 1 Handout
		8	Pipelined MIPS				
5	24-Feb	9	Pipeline Hazards			C for Java Programmers	C Lab 1 (IN-CLASS)
		10	Control Hazards & ISA Variations				
6	3-Mar	11	RISC & CISC & Prelim 1 Review		Prelim1	C lecture 2	C Lab 2 (IN-CLASS)
		12	Calling Conventions				
7	10-Mar	13	Calling Conventions			MIPS 2	Proj 2: MIPS 2 Handout
		14	Calling Conventions				
8	17-Mar	15	Linkers	-		Intro to UNIX/Linux	C Lab 2 (cont. IN-CLASS)
		16	Caches 1			ssh, gcc, How to tunnel	
9	24-Mar	17	Caches 2			VM & Unix	VM & Unix
		18	Caches 3				
	31-Mar		Spring Break				
			Spring Break				
10	7-Apr	19	Virtual Memory 1			Stack Smashing	Lab 3: Buffer Overflows handout
		20	Virtual Memory 2				
11	14-Apr	21	Traps	HW2: Calling Conventions, RISC, CISC		Caches	Proj 3: Caches Handout
		22	Multicore Architectures & GPUs	Linkers & and more calling conventions			
12	21-Apr	23	Synchronization	HW2: Virtual memory, Caches,		Virtual Memory	Lab 4: (IN-CLASS) Virtual Memory
		24	Synchronization 2	Traps, Multicore, Synchronization			
13	28-Apr	25	GPUs & Prelim 2 Review			Synchronization	Proj 4: Multicore/NW Handout
		26	I/O		Prelim 2		
14	5-May	27	Future Directions				Proj 3 Tournament: Monday, May 4th
	12-May						
	19-May						

Grading

Lab (50% approx.)

- 5-6 Individual Labs
  - 2 out-of-class labs (5-10%)
  - 3-4 in-class labs (5-7.5%)
- 4 Group Projects (30-35%)
- Participation/Quizzes in lab (2.5%)

Lecture (50% approx.)

- 2 Prelims (35%)
  - Dates: March 3, April 30
- Homework (10%)
- Participation/Quizzes in lecture (5%)

Grading

Regrade policy

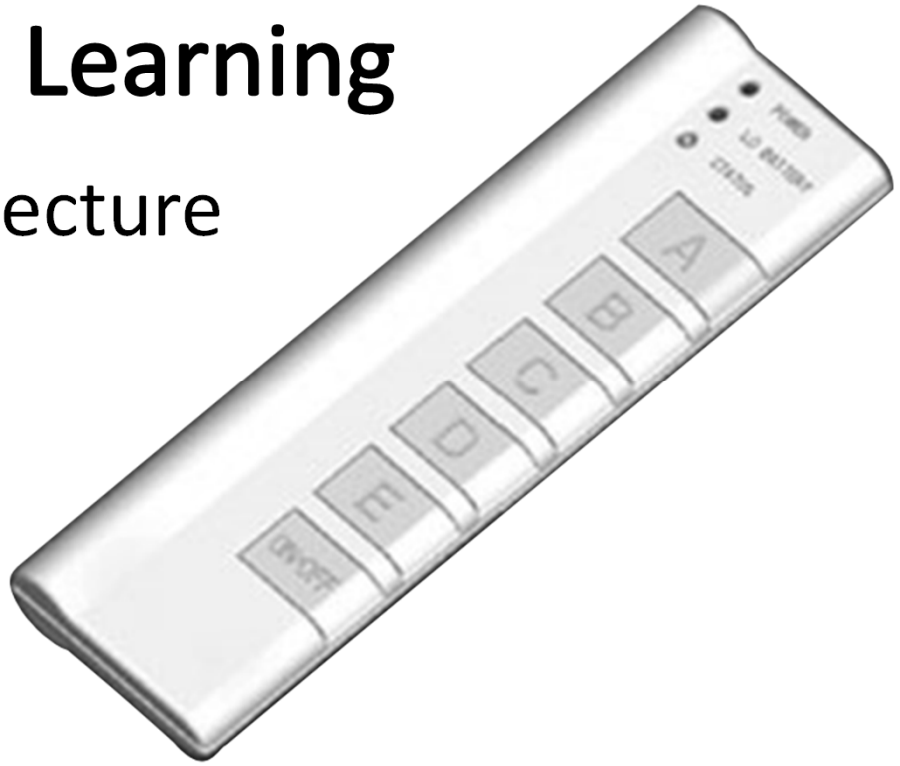
- Submit written request to lead TA,
  and lead TA will pick a different grader
- Submit another written request,
  lead TA will regrade directly
- Submit *yet* another written request for professor to regrade

Late Policy

- Each person has a total of ***four*** "slip days"
- Max of ***two*** slip days for any individual assignment
- For projects, slip days are deducted from all partners
- 25% deducted per day late after slip days are exhausted

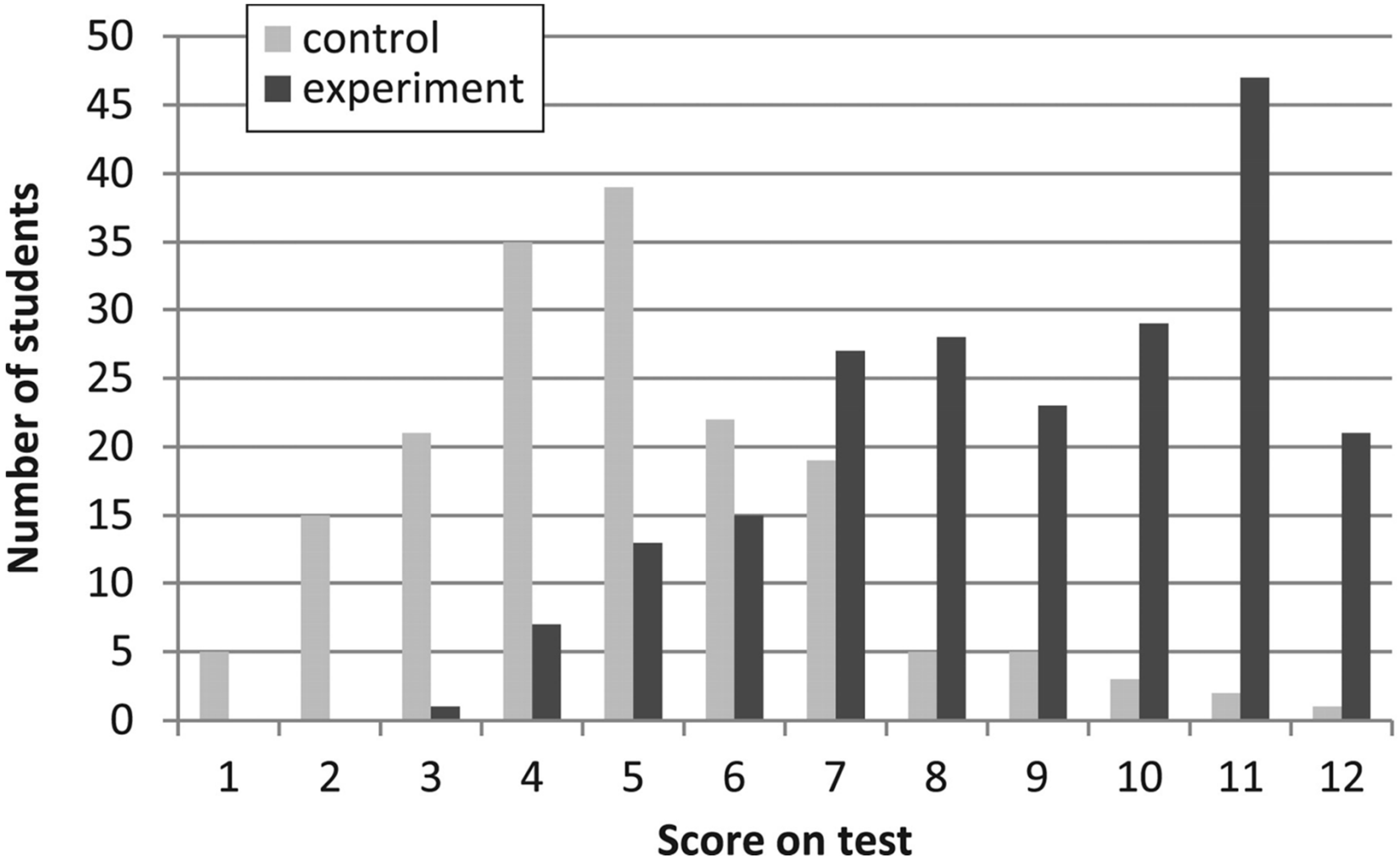
Active Learning

iClicker: Bring to every Lecture



Put all devices into ***Airplane Mode***

Active Learning



L Deslauriers et al. Science 2011;332:862-864

Fig. 1 Histogram of 270 physic student scores for the two sections: Experiment w/ quizzes and active learning. Control without.

Active Learning

Demo: What year are you in school?

a) Freshman
b) Sophomore
c) Junior
d) Senior
e) Other

Active Learning

Also, activity handouts will be available before class

In front of doors before you walk in

Administrivia

http://www.cs.cornell.edu/courses/cs3410/2015sp

- Office Hours / Consulting Hours
- Lecture slides, schedule, and Logisim
- CSUG lab access (esp. second half of course)

Lab Sections (start ***next week***)

T	2:55 – 4:10pm	Carpenter Hall 104 (Blue Room)
W	8:40—9:55am	Carpenter Hall 104 (Blue Room)
W	11:40am – 12:55pm	Carpenter Hall 104 (BlueRoom)
W	1:25—2:40pm	Carpenter Hall 104 (Blue Room)
W	3:35 – 4:50pm	Carpenter Hall 104 (Blue Room)
W	7:30—8:45pm	Carpenter Hall 235 (Blue Room)
R	8:40 – 9:55pm	Carpenter Hall 104 (Blue Room)
R	11:40 – 12:55pm	Carpenter Hall 104 (Blue Room)
R	2:55 – 4:10pm	Carpenter Hall 104 (Blue Room)
F	8:40 – 9:55am	Carpenter Hall 104 (Blue Room)
F	11:40am – 12:55pm	Carpenter Hall 104 (Blue Room)
F	1:25 – 2:40pm	Carpenter Hall 104 (Blue Room)
F	2:55 – 4:10pm	Carpenter Hall 104 (Blue Room)

- Labs are separate than lecture and homework
- Bring laptop to Labs
- ***This week:*** "hello world" lab: Intro to C and virtual machines
- ***Next*** week: Intro to logisim, logic circuits, and building an adder

Administrivia

http://www.cs.cornell.edu/courses/cs3410/2015sp

- Office Hours / Consulting Hours
- Lecture slides, schedule, and Logisim
- CSUG lab access (esp. second half of course)

Course Virtual Machine (VM)

- Identical to CSUG Linux machines
- Download and use for labs and projects
- **https://confluence.cornell.edu/display/coecis/CSUG+Lab+VM+Informatior**

Communication

Email

- cs-3410-staff-l@cornell.edu
- The email alias goes to me and the TAs, not to whole class

Assignments

- CMS: http://cms.csuglab.cornell.edu

Newsgroup

- http://www.piazza.com/cornell/spring2015/cs3410
- For students

iClicker

- http://atcsupport.cit.cornell.edu/pollsrvc/

Lab Sections, Projects, and Homeworks

Lab Sections start ***this*** week

- This week: “hello world” lab: Intro to C and virtual machines
- Next week: Intro to logisim, logic circuits and building an adder

Labs Assignments

- Individual
- One week to finish (usually Monday to Monday)

Projects

- two-person teams
- Find partner in same section

Homeworks

- One before each prelim
- Will be released a few weeks ahead of time
- Finish question after covered in lecture

Academic Integrity

All submitted work must be your own

- OK to study together, but do not share soln's
- Cite your sources

Project groups submit joint work

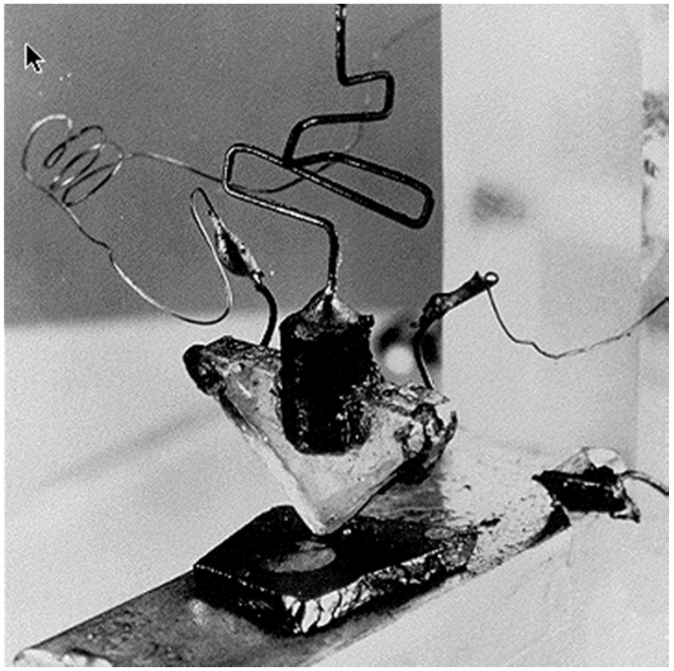
- Same rules apply to projects at the group level
- Cannot use of someone else's soln

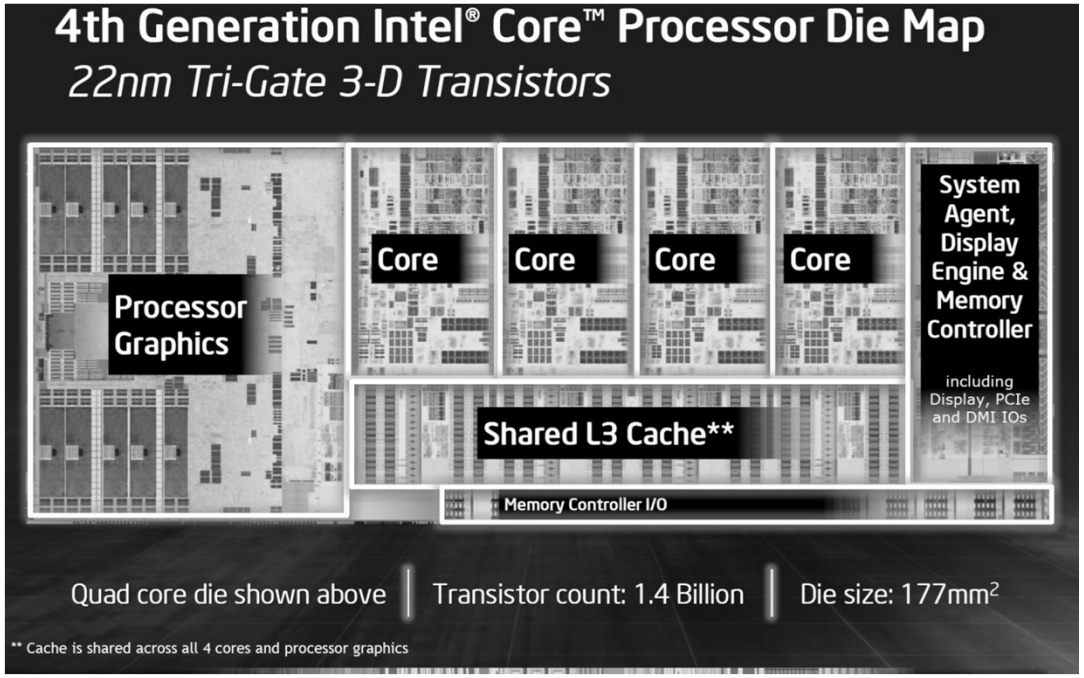
Closed-book exams, no calculators

- Stressed? Tempted? Lost?
  - Come see us before due date!

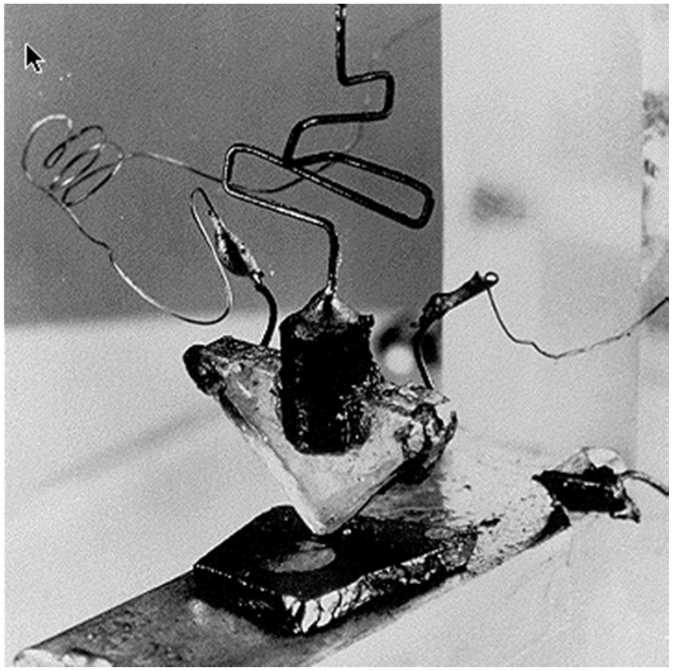
Plagiarism in any form will not be tolerated

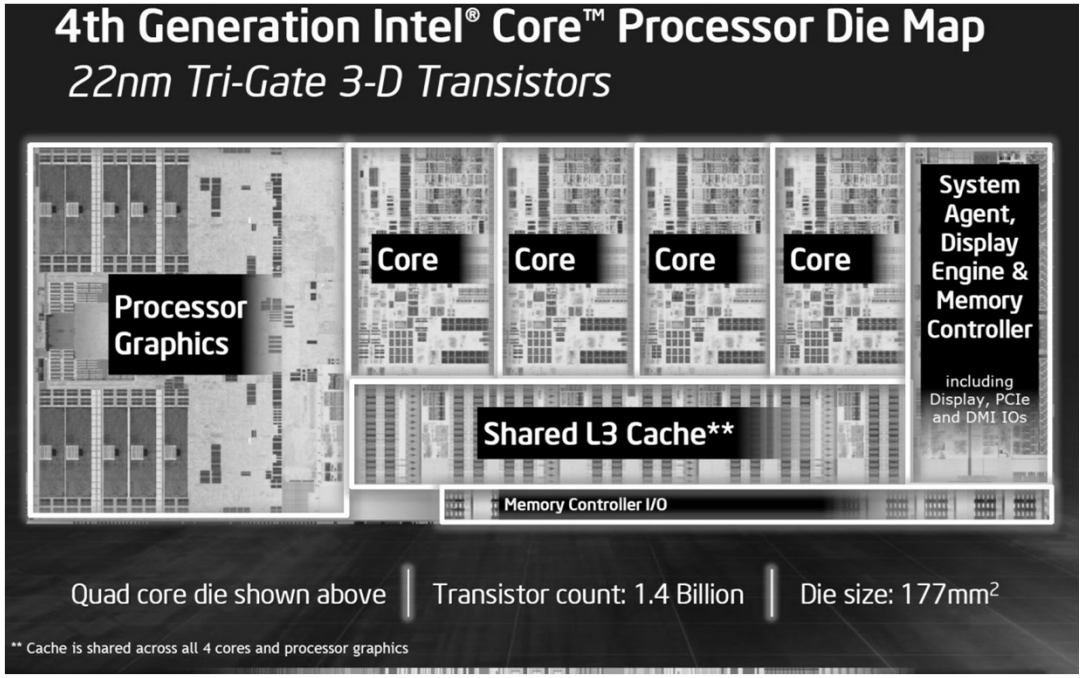
Why do CS Students Need Transistors?





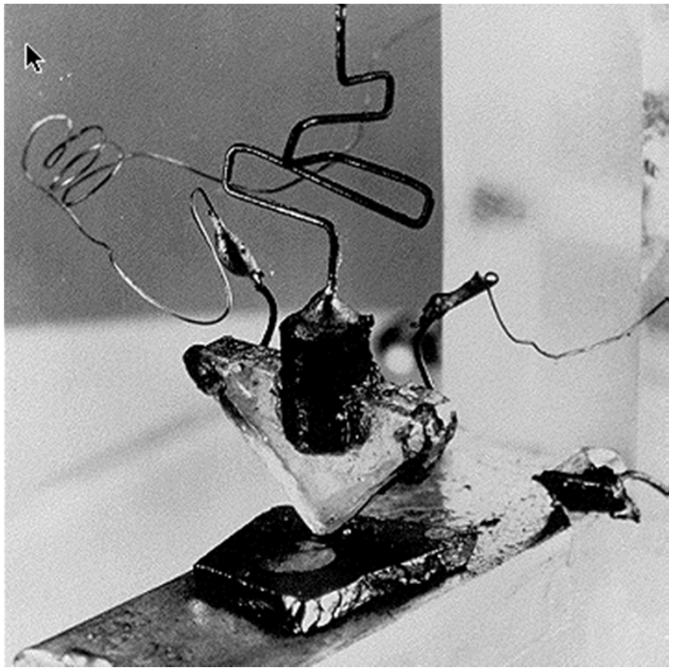
Why do CS Students Need Transistors?

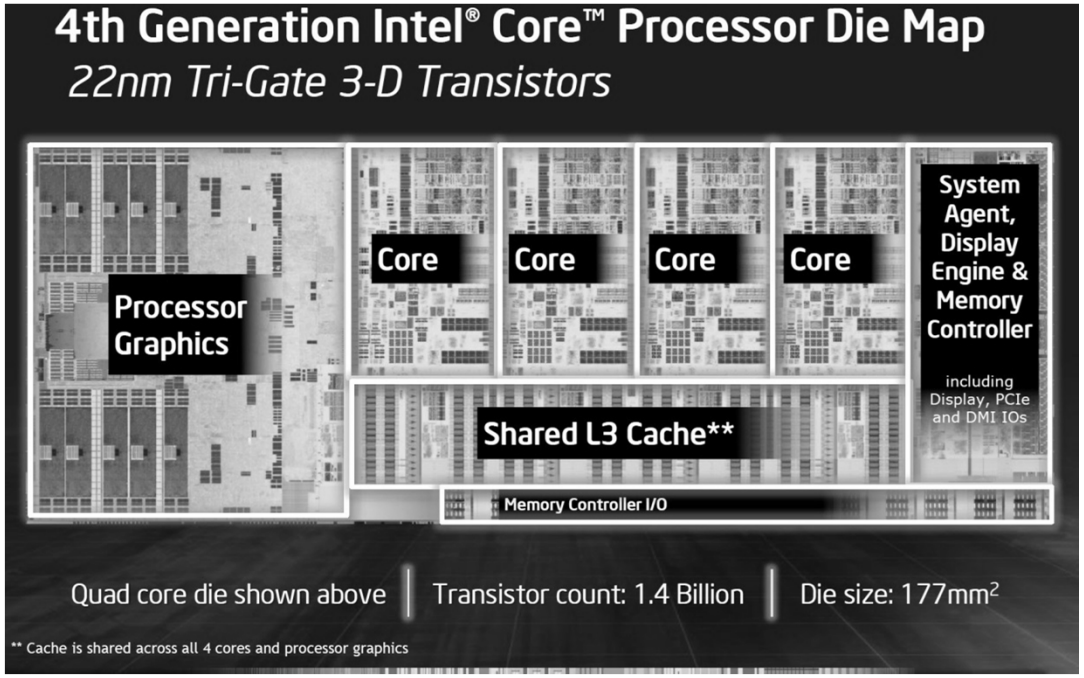




Functionality and Performance

Why do CS Students Need Transistors?

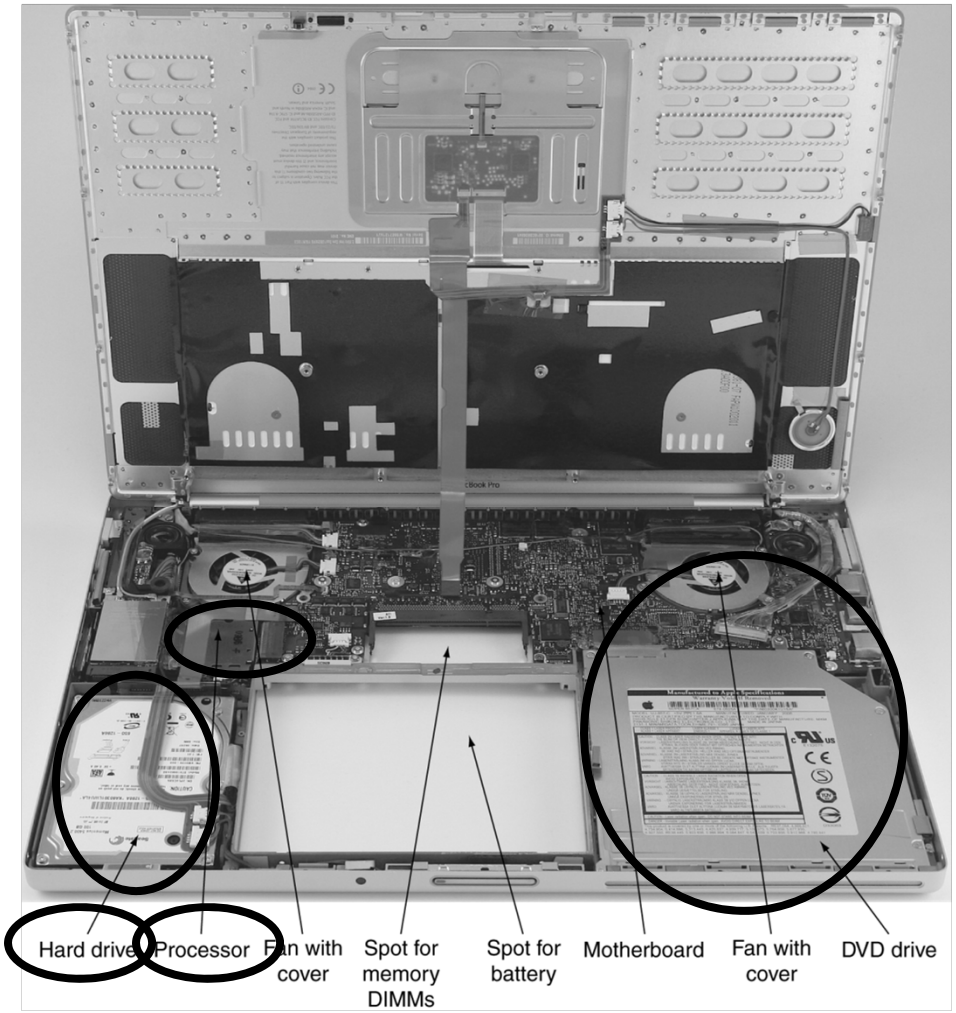


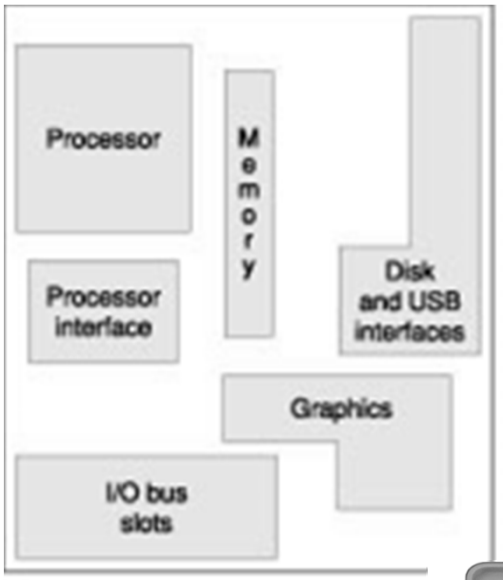


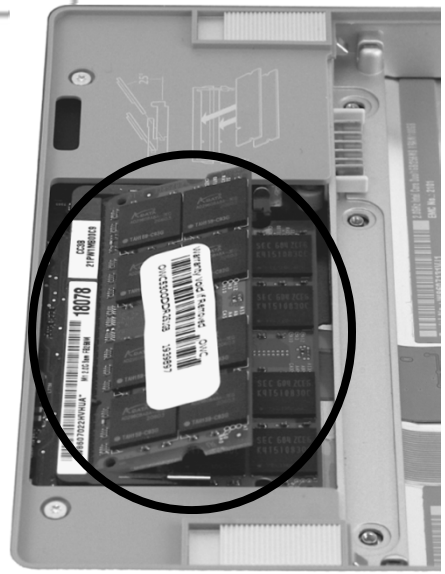
To be better Computer Scientists and Engineers

- Abstraction: simplifying complexity
- How is a computer system organized? How do I build it?
- How do I program it? How do I change it?
- How does its design/organization effect performance?

Computer System Organization







Computer System Organization

Computer System = ?

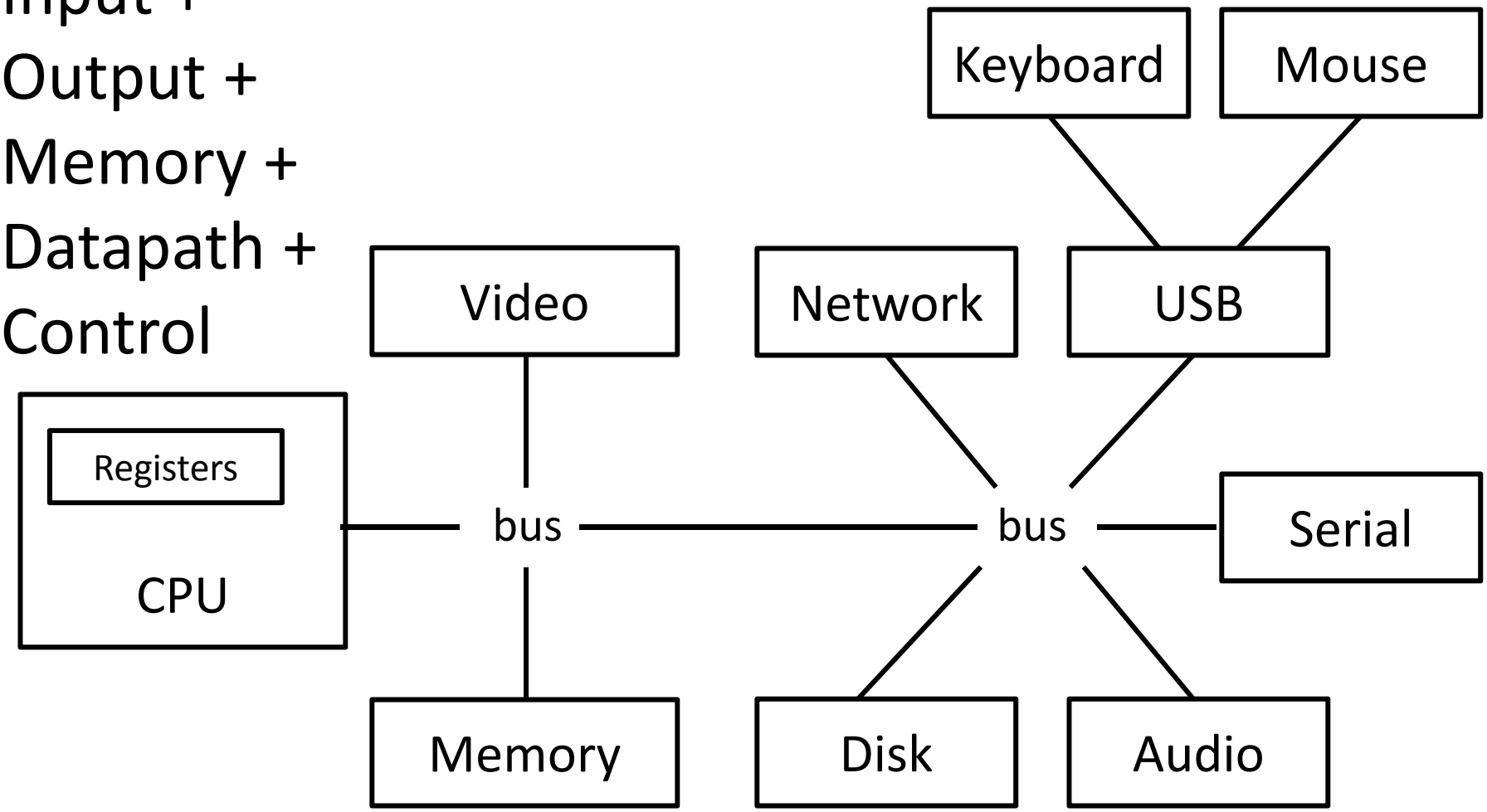
Input +

Output +

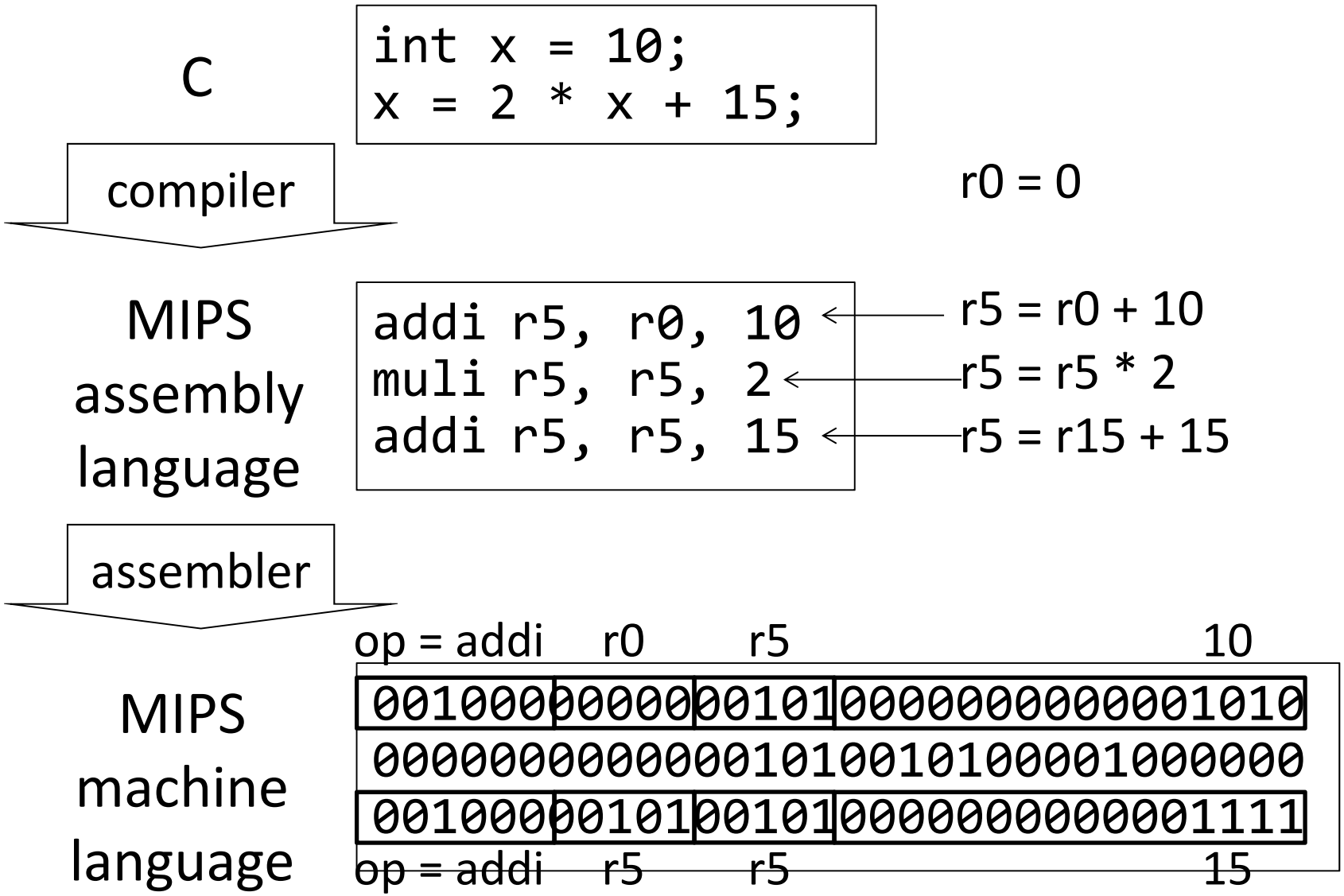
Memory +

Datapath +

Control



Compilers & Assemblers



Instruction Set Architecture

ISA

- abstract interface between hardware and the lowest level software

- user portion of the instruction set plus the operating system interfaces used by application programmers

Basic Computer System

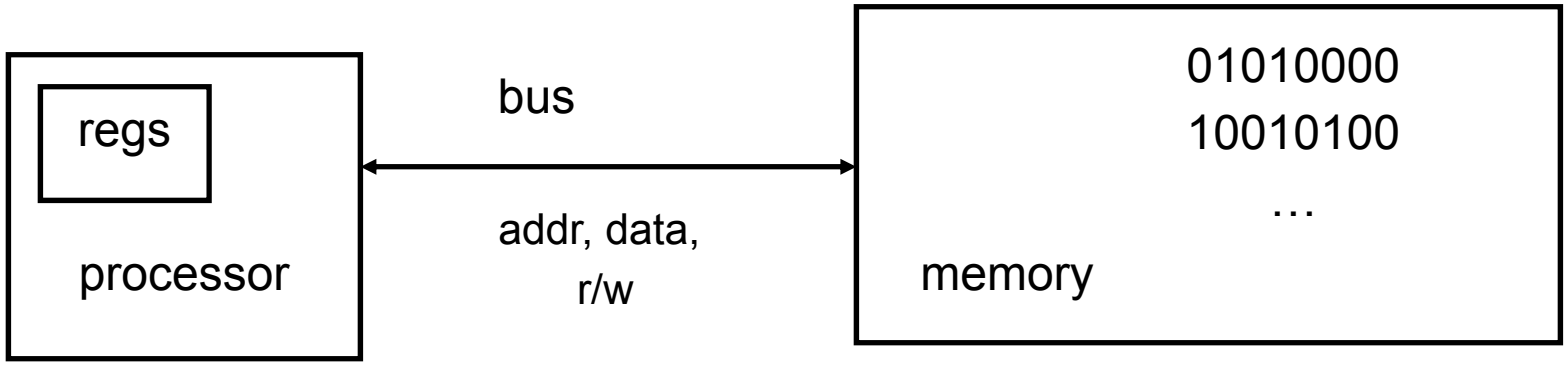
A processor executes instructions

- Processor has some internal state in storage elements (registers)

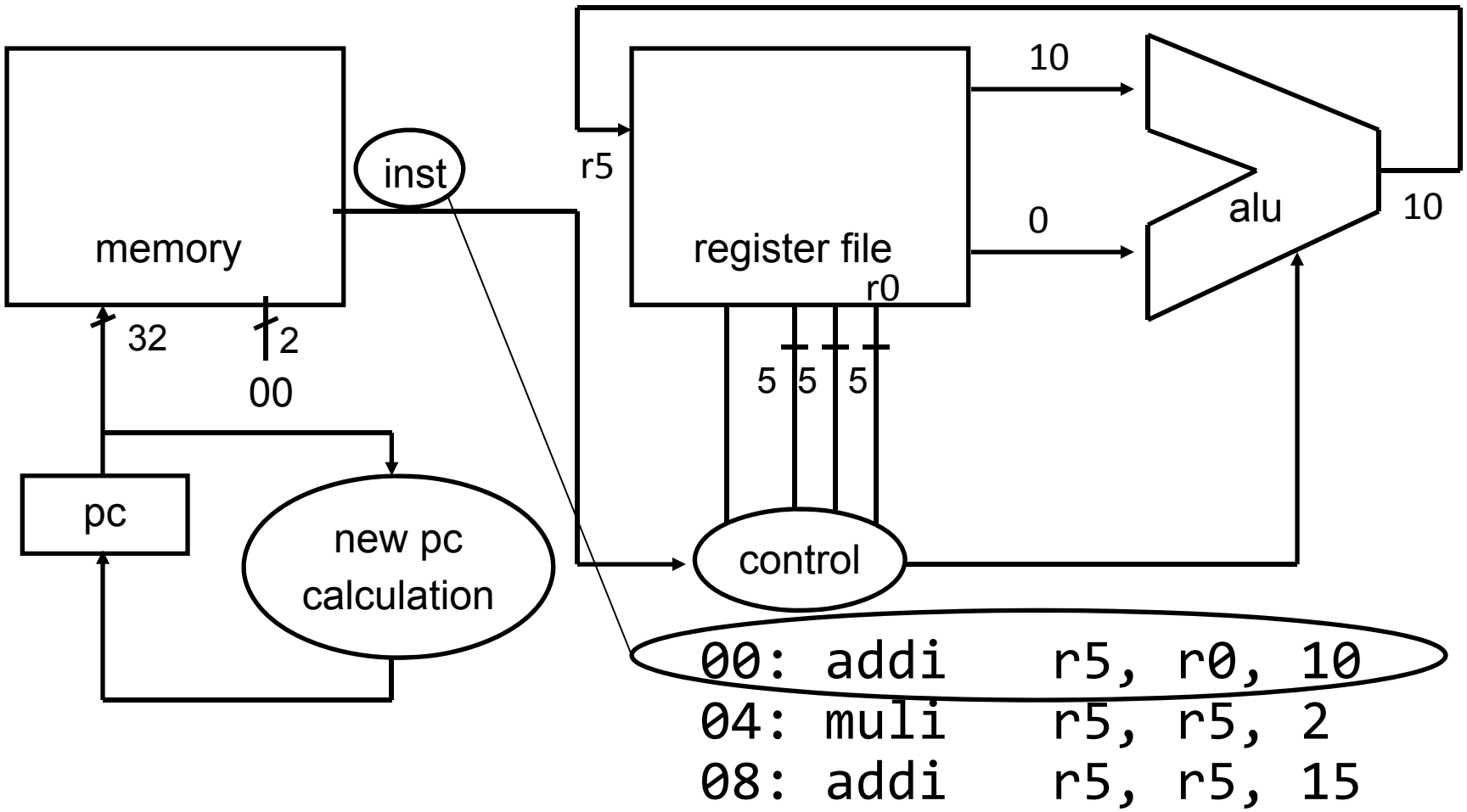
A memory holds instructions and data

- von Neumann architecture: combined inst and data

A bus connects the two



How to Design a Simple Processor



Inside the Processor

AMD Barcelona: 4 processor cores

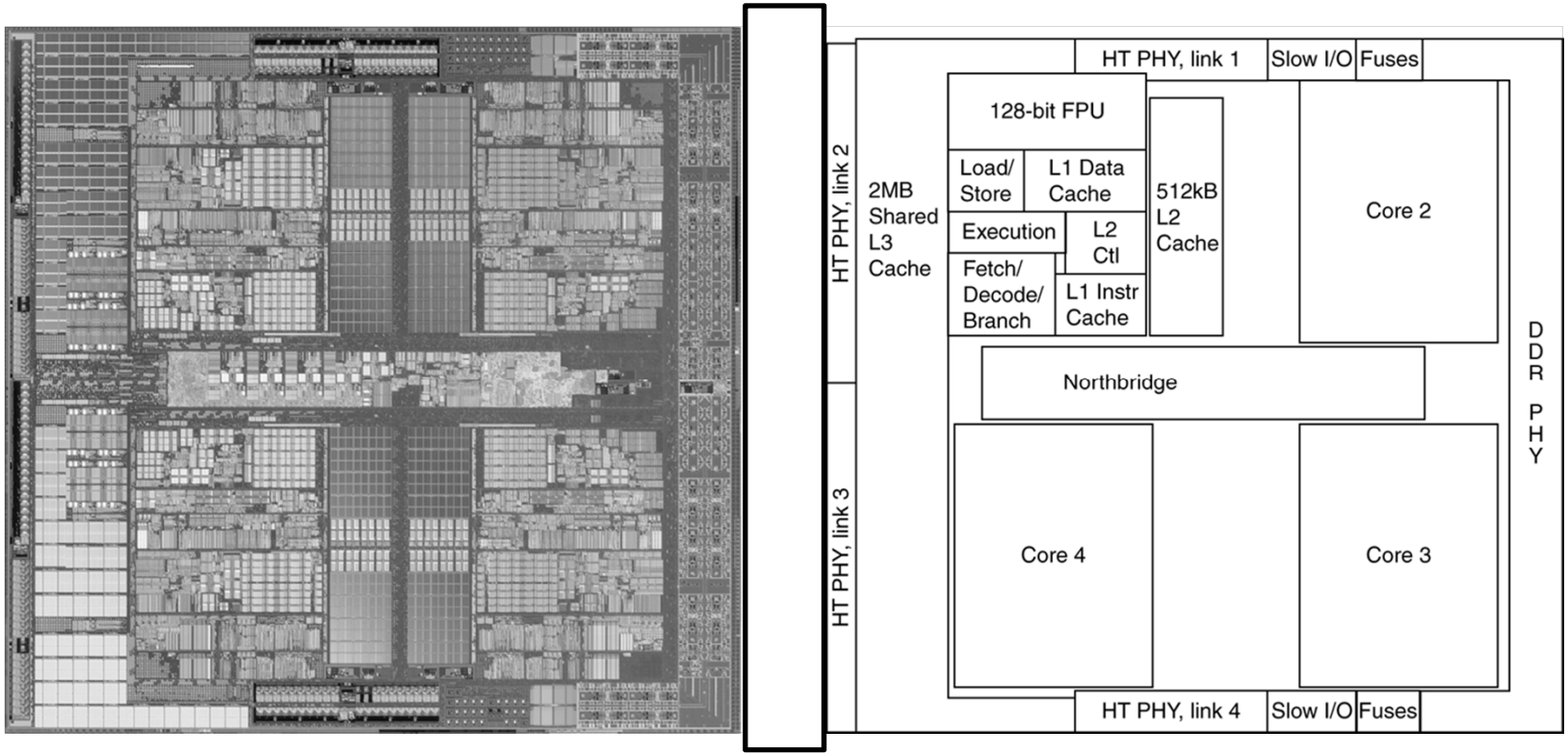
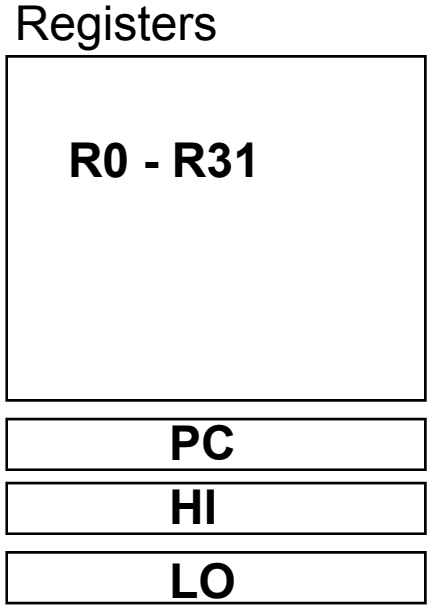


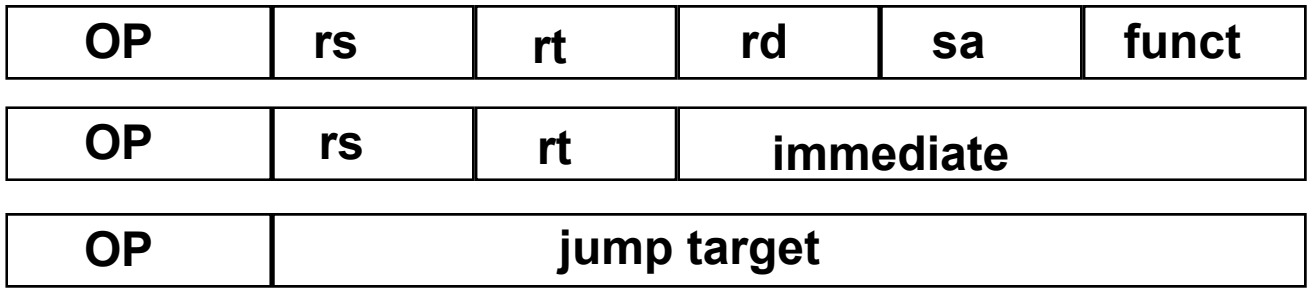
Figure from Patterson & Hennesssy, Computer Organization and Design, 4th Edition

How to Program the Processor: MIPS R3000 ISA

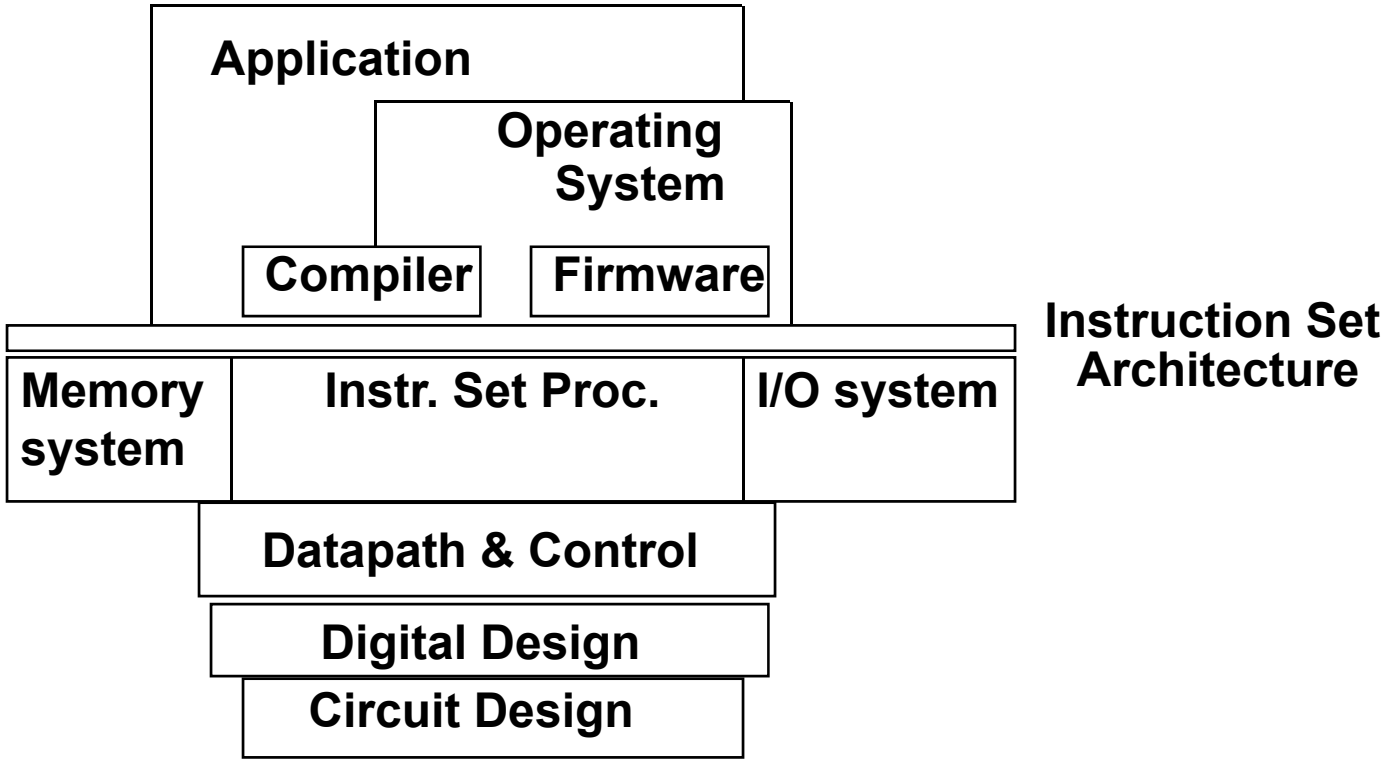
Instruction Categories

- Load/Store
- Computational
- Jump and Branch
- Floating Point
  - coprocessor
- Memory Management





Overview


Application
Operating System
Compiler
Firmware
Instruction Set Architecture
Memory system
Instr. Set Proc.
I/O system
Datapath & Control
Digital Design
Circuit Design

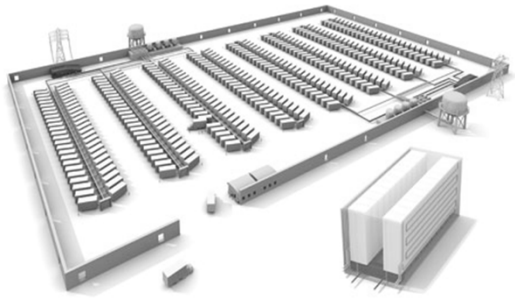
Applications

Everything these days!

- Phones, cars, televisions, games, computers,…

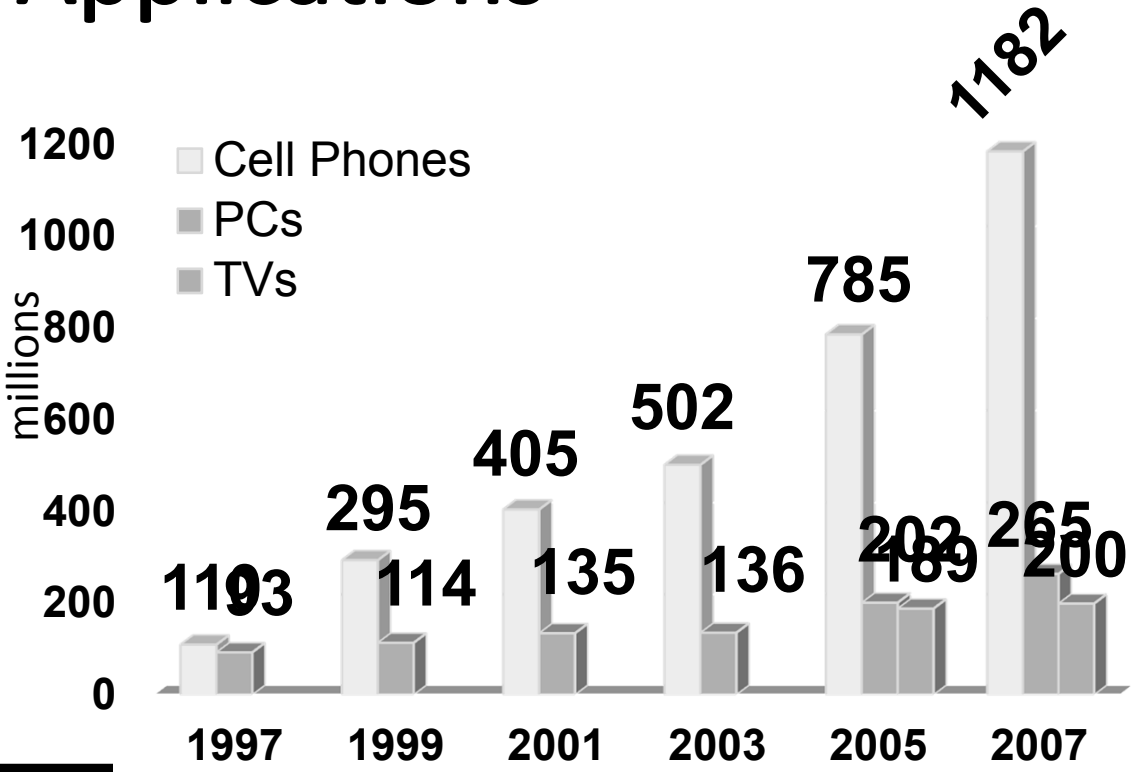
Applications

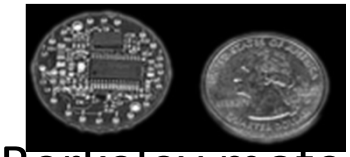
Xilinx FPGA



Cloud Computing

NVidia GPU



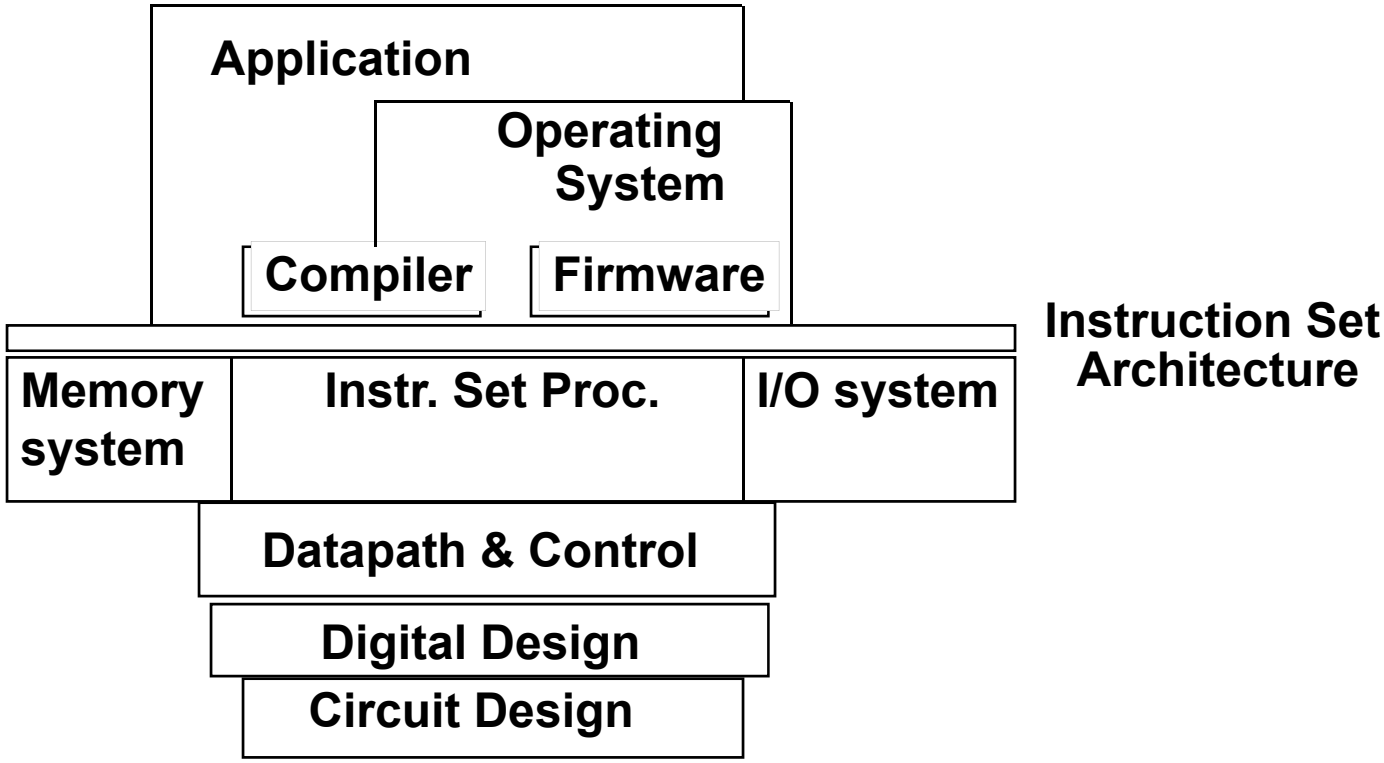


Berkeley mote

Cell Phone

Cars

Covered in this course



Reflect

Why take this course?

- Basic knowledge needed for *all* other areas of CS:
  operating systems, compilers, ...
- Levels are not independent
  hardware design ↔ software design ↔ performance
- Crossing boundaries is hard but important
  device drivers
- Good design techniques
  abstraction, layering, pipelining, parallel vs. serial, ...
- Understand where the world is going