CS 3410: Computer System Organization and Programming

Hakim Weatherspoon

CS 3410, Spring 2013

Computer Science

Cornell University

Computer System Organization

The most amazing and likely to be most long-lived invention of the 1800’s was…

Computer System Organization

The most amazing and likely to be most long-lived invention of the 1800's was…

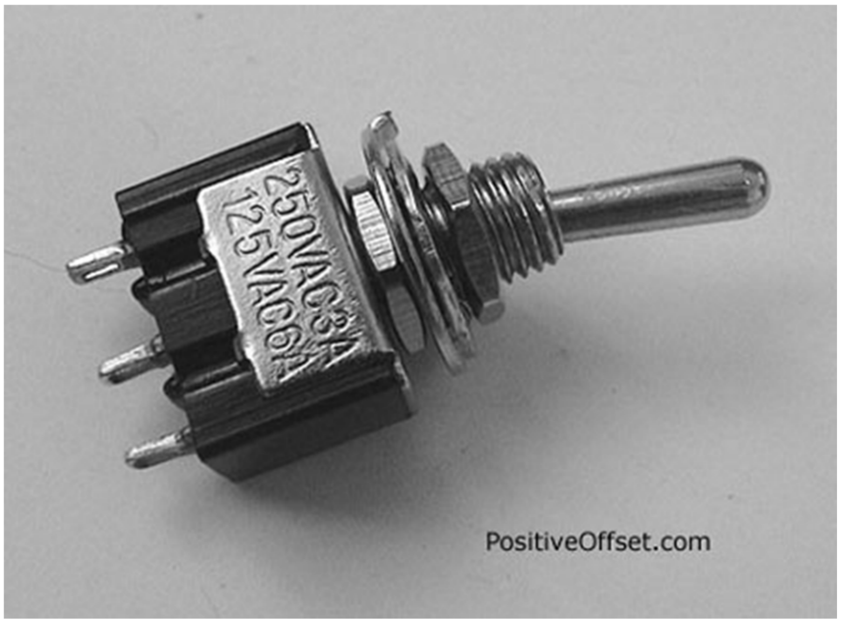
- (a) The steam engine?
- (b) The lightning rod?
- (c) The carbonated beverage?
- (d) All of the above
- (e) None

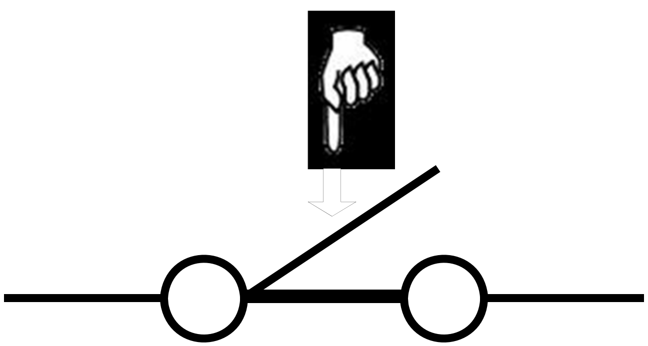
Computer System Organization

The most amazing and likely to be most long-lived invention of the 1800's was…

THE ELECTRIC SWITCH

Basic Building Blocks: A switch



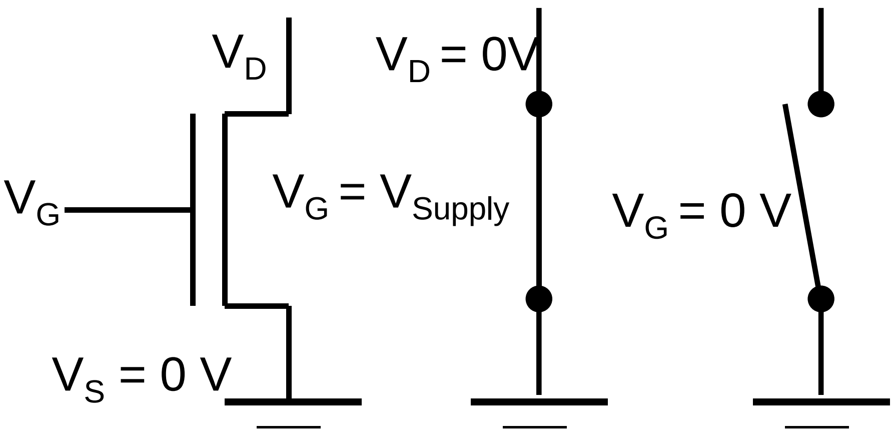


A switch is a simple device that can act as a conductor or isolator

Can be used for amazing things…

NMOS and PMOS Transistors

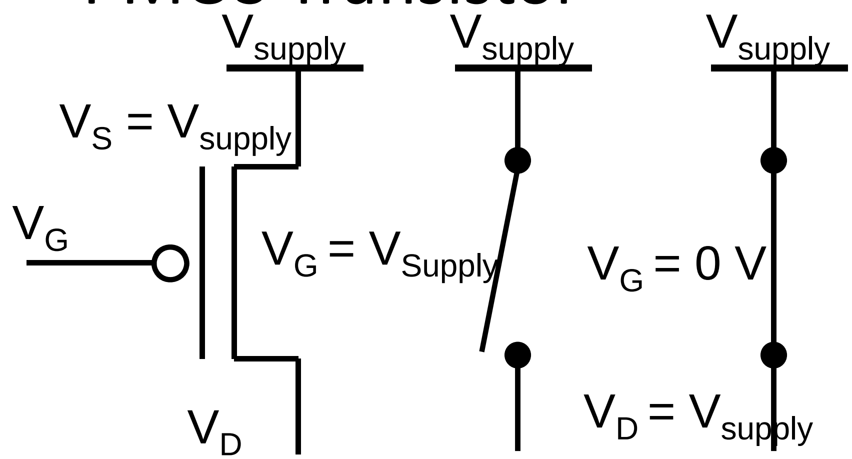
- NMOS Transistor



Closed switch
When $V_G = V_{supply}$

- Connect source to drain when $V_G = V_{supply}$
- N-channel transistor

PMOS Transistor



Closed switch
When $V_G = 0$ V

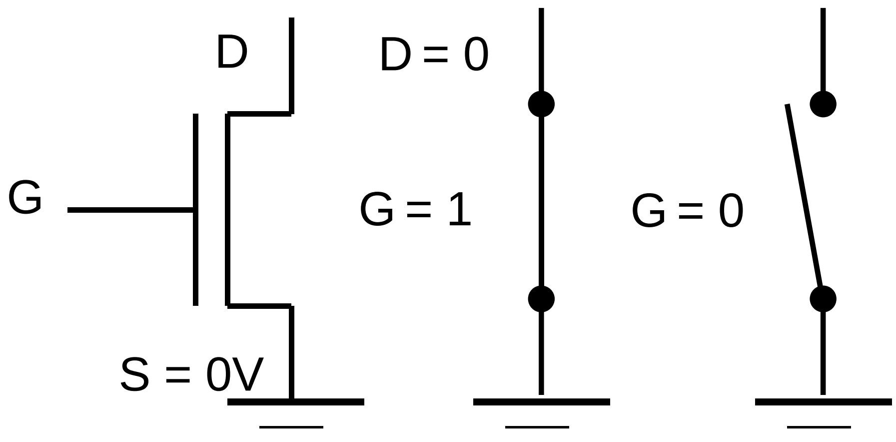
Connect source to drain when $V_G = 0$ V

P-channel transistor

V_S: voltage at the source
V_D: voltage at the drain
V_{supply}: max voltage (aka a logical 1)
(ground): min voltage (aka a logical 0)

NMOS and PMOS Transistors

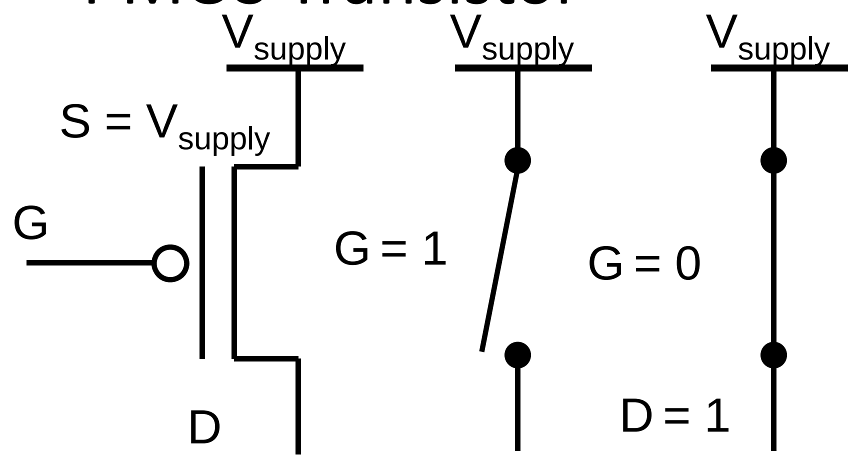
- NMOS Transistor



Closed switch
When $V_G = V_{supply}$

- Connect source to drain when gate = 1
- N-channel transistor

PMOS Transistor



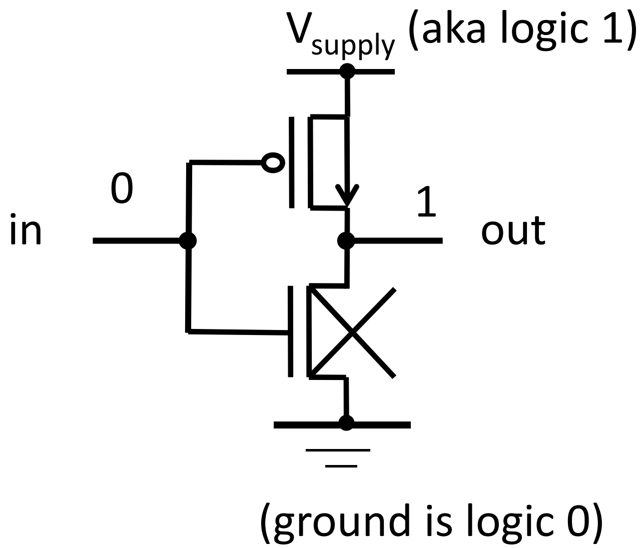
Closed switch
When V_G = 0 V

Connect source to drain when gate = 0

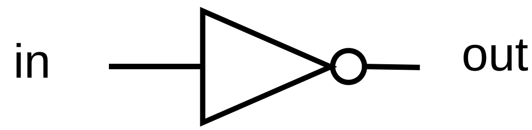
P-channel transistor

V_S: voltage at the source
V_D: voltage at the drain
V_{supply}: max voltage (aka a logical 1)
(ground): min voltage (aka a logical 0)

Inverter



- Function: NOT
- Called an inverter
- Symbol:

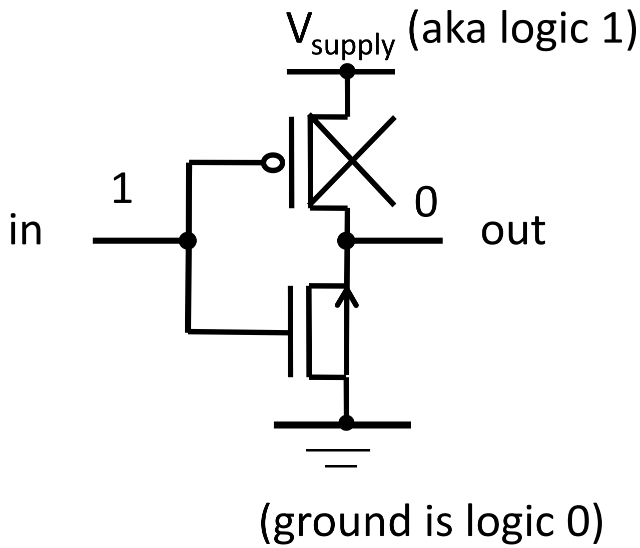


In	Out
0	1
1	0

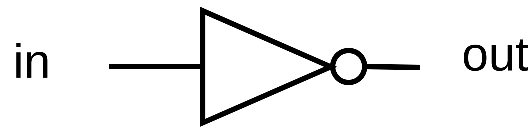
Truth table

- Useful for taking the inverse of an input
  - CMOS: complementary-symmetry metal**–oxide–semiconductor**

Inverter



- Function: NOT
- Called an inverter
- Symbol:



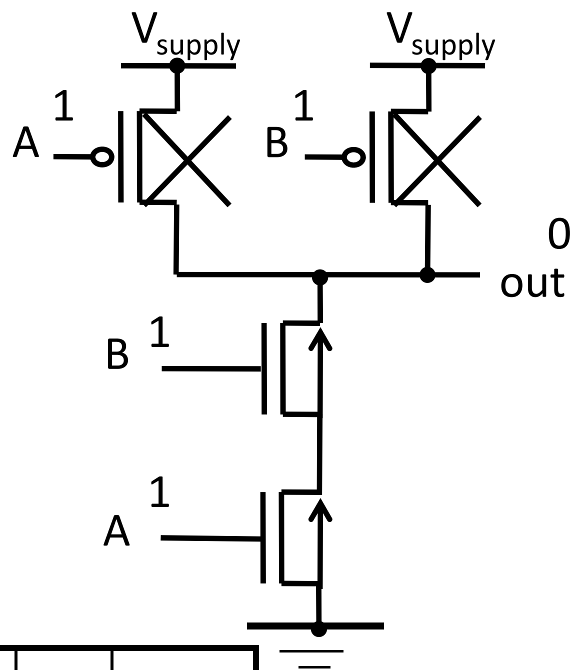
In	Out
0	1
1	0

Truth table

- Useful for taking the inverse of an input

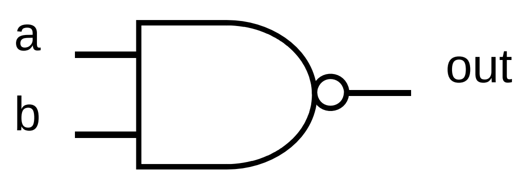
- CMOS: complementary-symmetry metal**–oxide–semiconductor**

NAND Gate

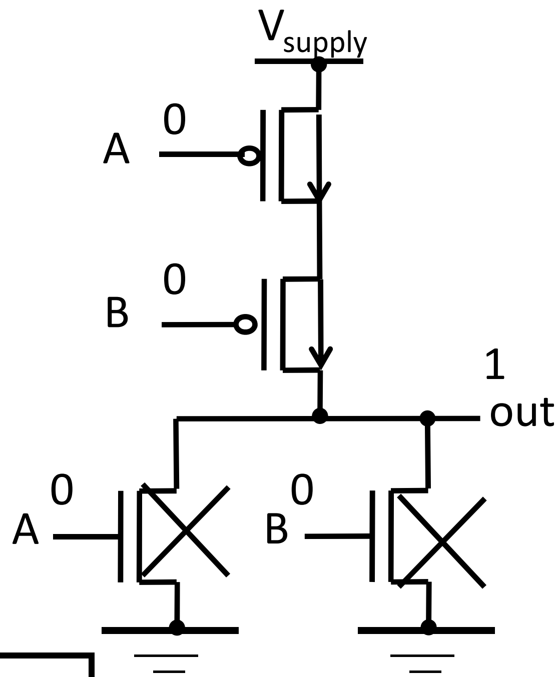


A	B	out
0	0	1
1	0	1
0	1	1
1	1	0

- Function: NAND
- Symbol:

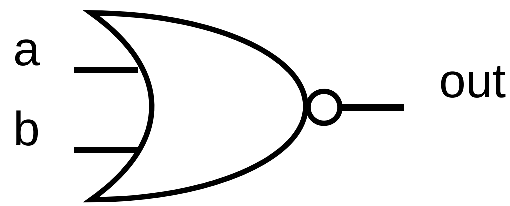


NOR Gate



A	B	out
0	0	1
1	0	0
0	1	0
1	1	0

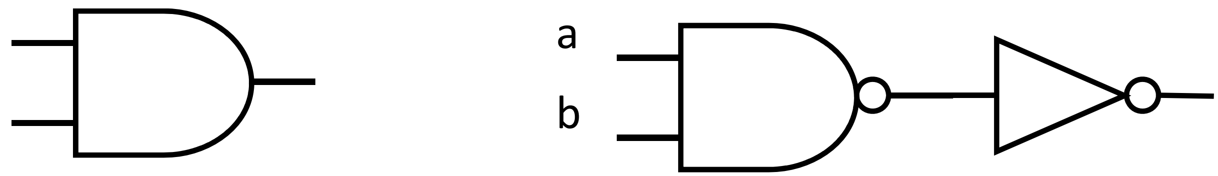
- Function: NOR
- Symbol:



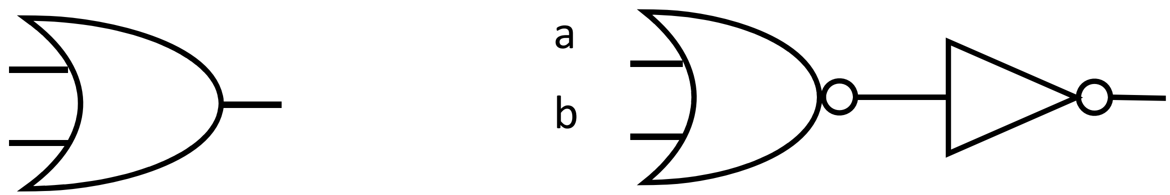
Building Functions

NOT:

AND:



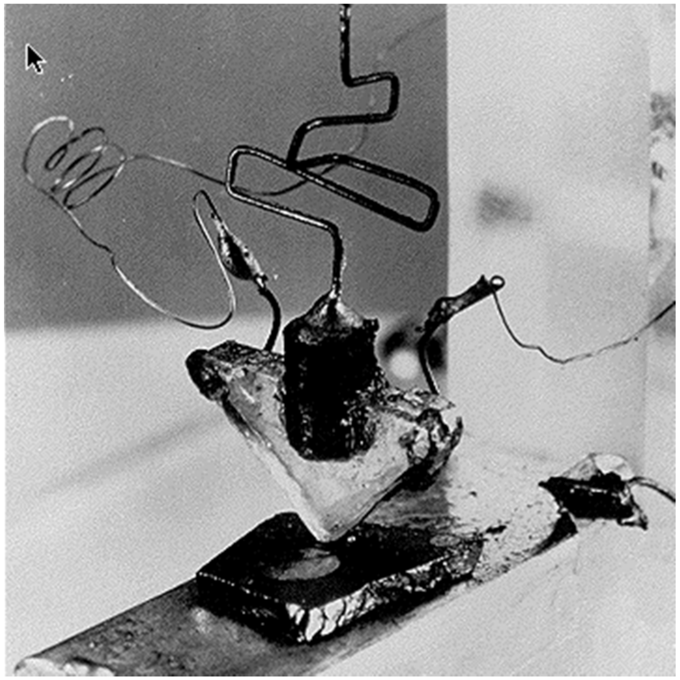
OR:

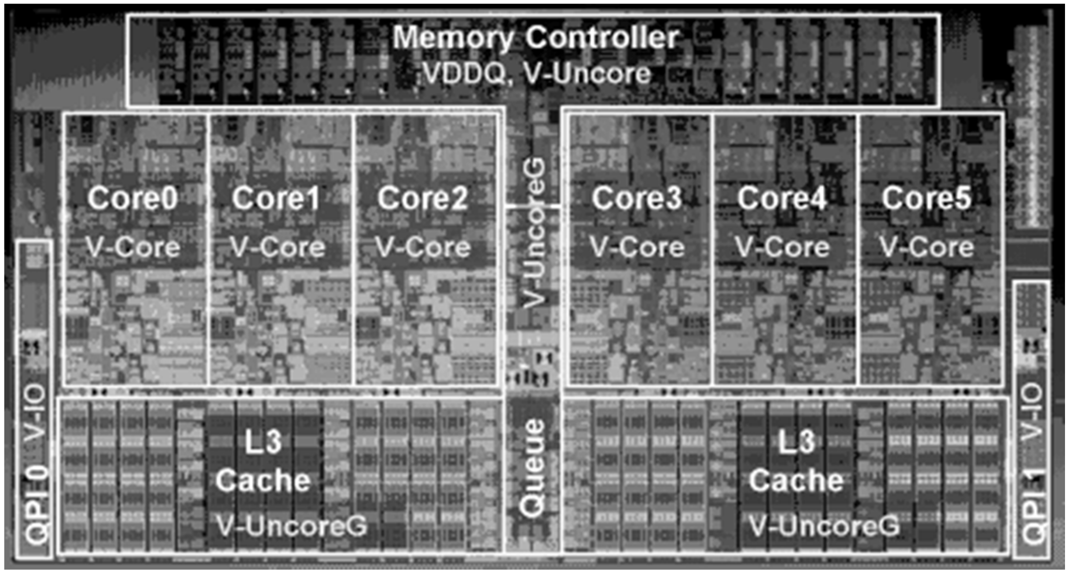


NAND and NOR are universal

- Can implement any function with NAND or just NOR gates
- useful for manufacturing

Then and Now





http://www.theregister.co.uk/2010/02/03/intel_westmere_ep_preview/

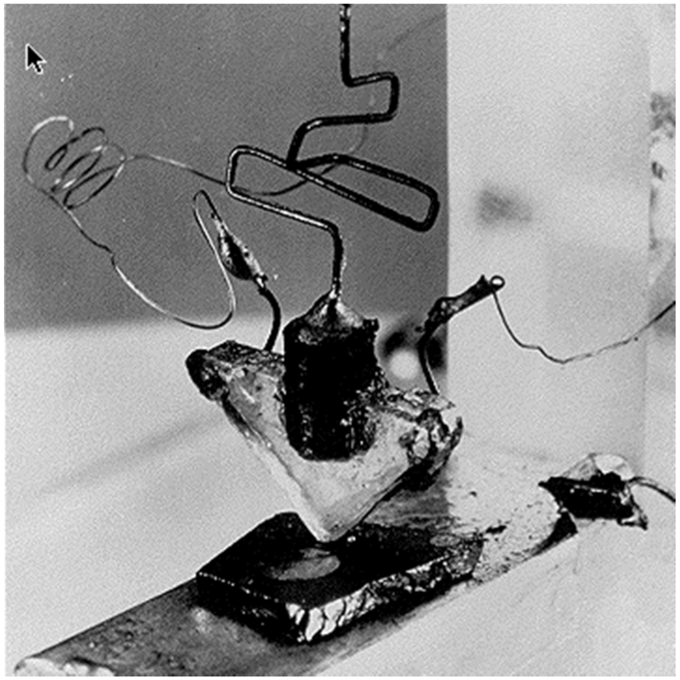
The first transistor

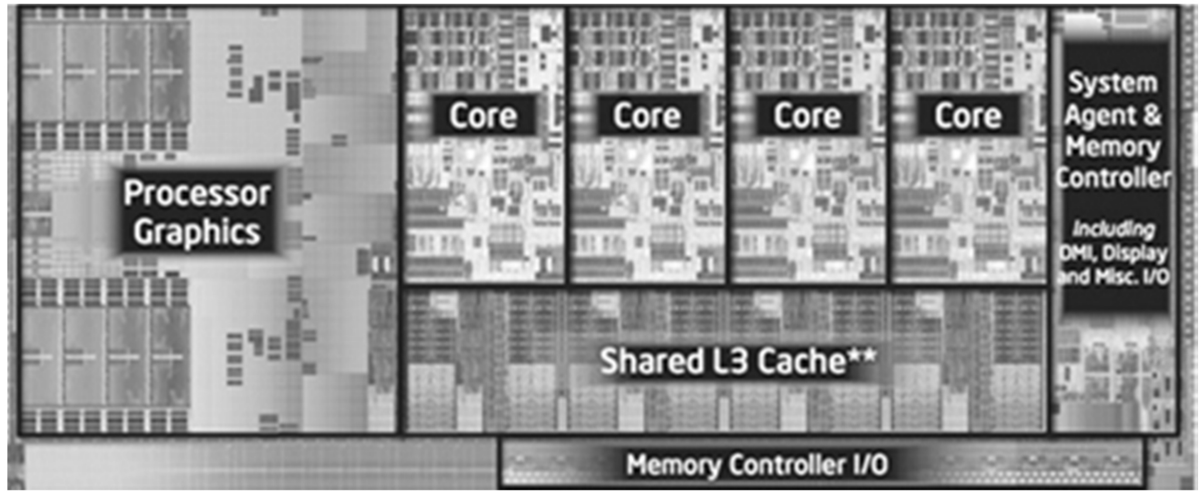
- on a workbench at

  AT&T Bell Labs in 1947
- Bardeen, Brattain, and Shockley

- An Intel Westmere
  - 1.17 billion transistors
  - 240 square millimeters
  - 32 nanometer: transistor gate width
  - Six processing cores
  - Release date: January 2010

Then and Now





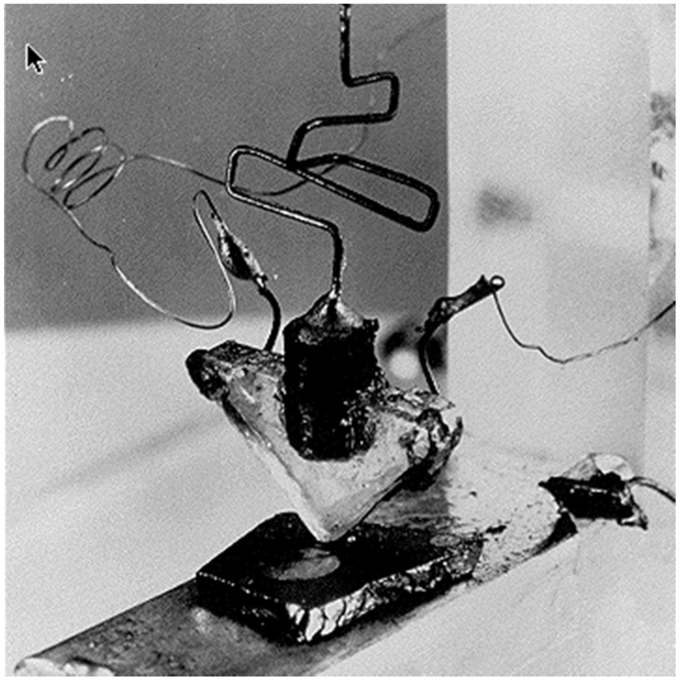
http://forwardthinking.pcmag.com/none/296972-intel-releases-ivy-bridge-first-processor-with-tri-gate-transistor

The first transistor

- on a workbench at
  AT&T Bell Labs in 1947
- Bardeen, Brattain, and Shockley

- An Intel Ivy Bridge
  - 1.4 billion transistors
  - 160 square millimeters
  - 22 nanometer: transistor gate width
  - Up to eight processing cores
  - Release date: April 2012

Then and Now



http://www.anandtech.com/show/6386/samsung-galaxy-note-2-review-t-mobile-/3

The first transistor

- on a workbench at
  AT&T Bell Labs in 1947
- Bardeen, Brattain, and Shockley

- Samsung Galaxy Note II
  - Eynos 4412 System on a Chip (SoC)
  - ARM Cortex-A9 processing core
  - 32 nanometer: transistor gate width
  - Four processing cores
  - Release date: November 2012

Moore's Law

The number of transistors integrated on a single die will double every 24 months...

– Gordon Moore, Intel co-founder, 1965

Amazingly Visionary

1971 – 2300 transistors – 1MHz – 4004

1990 – 1M transistors – 50MHz – i486

2001 – 42M transistors – 2GHz – Xeon

2004 – 55M transistors – 3GHz – P4

2007 – 290M transistors – 3GHz – Core 2 Duo

2009 – 731M transistors – 2GHz – Nehalem

2012 – 1400M transistors – 2-3GHz – Ivy Bridge

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

Announcements: How class organized

Instructor: Hakim Weatherspoon

(hweather@cs.cornell.edu)

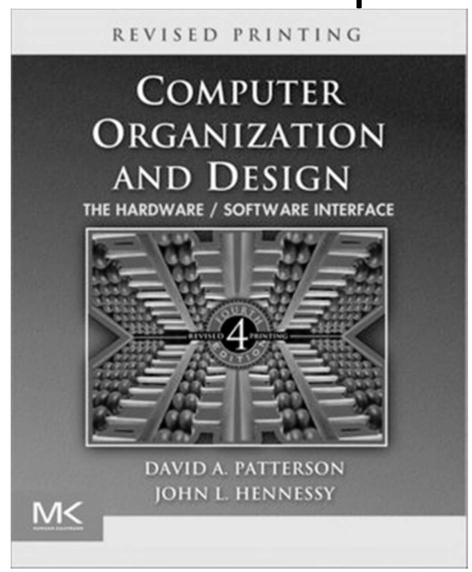
Lecture:

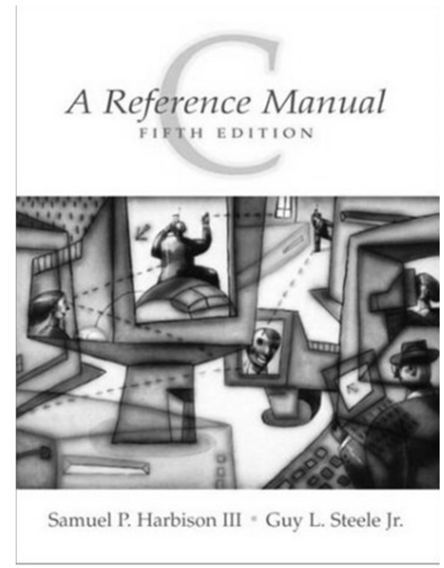
- Tu/Th 1:25-2:40
- Olin 155

Lab Sections:

- Carpenter 104 (Blue Room)
- Carpenter 235 (Red Room)

Required Textbooks





Suggested Textbook

Who am I?

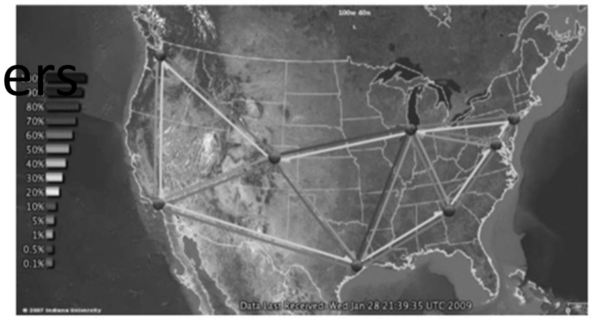
Prof. Hakim Weatherspoon

- (Hakim means Doctor, wise, or prof. in Arabic)
- Background in Education
  - Undergraduate University of Washington
    - Played Varsity Football
      - » Some teammates collectively make $100’s of millions
      - » I teach!!!
  - Graduate University of California, Berkeley
    - Some class mates collectively make $100’s of millions
    - I teach!!!
- Background in Operating Systems
  - Peer-to-Peer Storage
    - Antiquity project - Secure wide-area distributed system
    - OceanStore project – Store your data for 1000 years
  - Network overlays
    - Bamboo and Tapestry – Find your data around globe
  - Tiny OS
    - Early adopter in 1999, but ultimately chose P2P direction

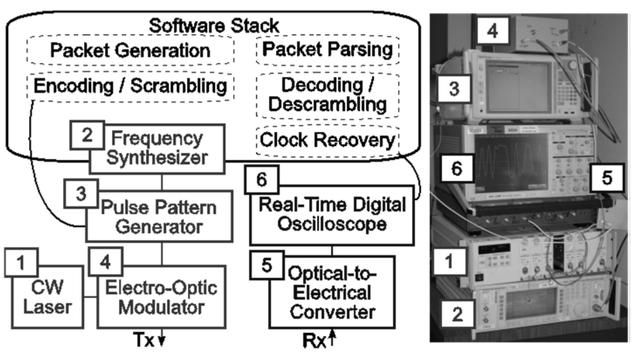
Who am I?

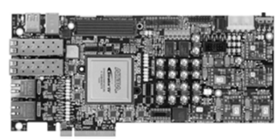
Cloud computing/storage

- Optimizing a global network of data centers
- Cornell Ntional λ-Rail Rings testbed
- Software Defined Network Adapter
- Energy: KyotoFS/SMFS



Antiquity: built a global-scale storage system





Course Staff

cs3410-staff-l@cs.cornell.edu

Lecture/Homwork TA’s

- Detian Shi (ds629@cornell.edu)
- ***Paul Upchurch (paulu@cs.cornell.edu) (lead)***
- Paul Heran Yang (hy279@cornell.edu)

Lab TAs

- Efe Gencer (gencer@cs.cornell.edu)
- ***Erluo Li (el378@cornell.edu***)
- ***Han Wang (hwang@cs.cornell.edu) (lead)***

Lab Undergraduate consultants

- ***Roman Averbukh (raa89@cornell.edu)***
- Favian Contreras (fnc4@cornell.edu)
- Jisun Jung (jj329@cornell.edu)
- Emma Kilfoyle (efk23@cornell.edu)
- Joseph Mongeluzzi (jam634@cornell.edu)
- Sweet Song (ss2249@cornell.edu)
- ***Peter Tseng (pht24@cornell.edu)***
- Victoria Wu (vw52@cornell.edu)
- ***Jason Zhao (jlz27@cornell.edu)***

Administrative Assistant:

- Molly Trufant (mjt264@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 Phillips (PHL) 203
- Class started yesterday and ends March 1st

CS 2022 (Introduction to C)

- 1-credit course will greatly help with CS 3410
- *Unfortunately, offered in the fall, not spring*
- Instead, we will offer a primer to C next Monday, January 28th, 6-8pm. Location TBD.

Schedule (subject to change)

Week	Date (Tue)	Lecture#	Lecture Topic	HW	Prelim	Lab Topic	Lab/Proj
1	22-Jan	1	Intro			Logisim	Lab 0: Adder/Logisim intro Handout
		2	Logic & Gates				
2	29-Jan	3	Numbers & Arithmetic	HW1: Logic, Gates, Numbers, & Arithmetic		ALU	lab 1: ALU Handout (design doc due one-week, lab1 due two-weeks)
		4	State & FSMs				
3	5-Feb	5	Memory			FSM	Lab 2: (IN-CLASS) FSM Handout
		6	Simple CPU				
4	12-Feb	7	CPU Performance & Pipelines	HW2: FSMs, Memory, CPU, Performance, and pipelined MIPS		MIPS	Proj 1: MIPS 1 Handout
		8	Pipelined MIPS				
5	19-Feb	9	Pipeline Hazards			Fast Adder?	Proj 1: Design Doc Due
		10	Control Hazards & ISA Variations				
6	26-Feb	11	RISC & CISC		Prelim 1	MIPS Help Lab?	
		12	Calling Conventions				
7	5-Mar	13	Calling Conventions	HW3: Calling Conventions, RISC, CISC, Linkers		MIPS 2	Proj 2: MIPS 2 Handout
		14	Calling Conventions				
8	12-Mar	15	Linkers	-		C for Java Programmers	Proj 2: Design Doc Due
		16	Linkers & Caches 1			MIPS 2 Help	
	19-Mar		Spring Break				
			Spring Break				
9	26-Mar	17	Caches 1			Intro to UNIX/Linux	
		18	Caches 2		Prelim 2	ssh, gcc, How to tunnel	
10	2-Apr	19	Virtual Memory 1			Stack Smashing	Lab 3: Buffer Overflows handout
		20	Virtual Memory 2				
11	9-Apr	21	Virtual Memory 3 & Traps	HW4: Virtual memory, Caches,		Caches	Proj 3: Caches Handout
		22	Multicore Architectures	Traps, Multicore,			Exceptions???
12	16-Apr	23	Synchronization			Caches Help?	
		24	Synchronization 2				
13	23-Apr	25	Prelim 3 Review			Virtual Memory	Lab 4: (IN-CLASS) Virtual Memory
		26	Synchronization 2		Prelim 3		
14	30-Apr	27	I/O			Synchronization	Proj 4: Multicore/NW Handout
		28	Future Directions			Proj 4 Help Lab?	
	7-May						Proj 4: Design Doc Due
	15-May						
	5/15/2012 4:30pm						Proj 4 Due

Grading

Lab (45-50%)

- 5-6 Individual Labs (15-17.5%)
  - 2 out-of-class labs (10%)
  - 3-4 in-class labs (5-7.5%)
- 4 Group Projects (30%)
- Quizzes in lab (2.5%)

Lecture (45-50%)

- 3 Prelims (32.5 - 37.5%)
  - Tue Feb 26th, Thur Mar 28th, and Thur Apr 25th
- Homework (10%)
- Quizzes in lecture (2.5%)

Participation/Discretionary (5%)

Grading

Regrade policy

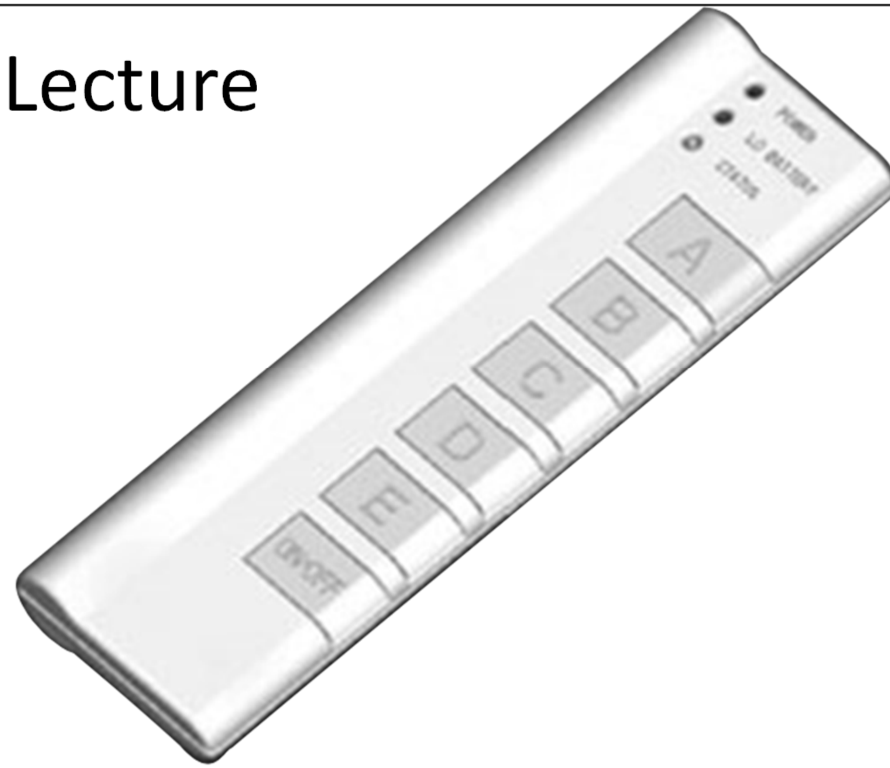
- Submit written request to lead TA,
  and lead TA will pick a different grader
- Submit another written request,
  lead TA will regrade directly
- Submit *yet* another written request for professor to regrade.

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

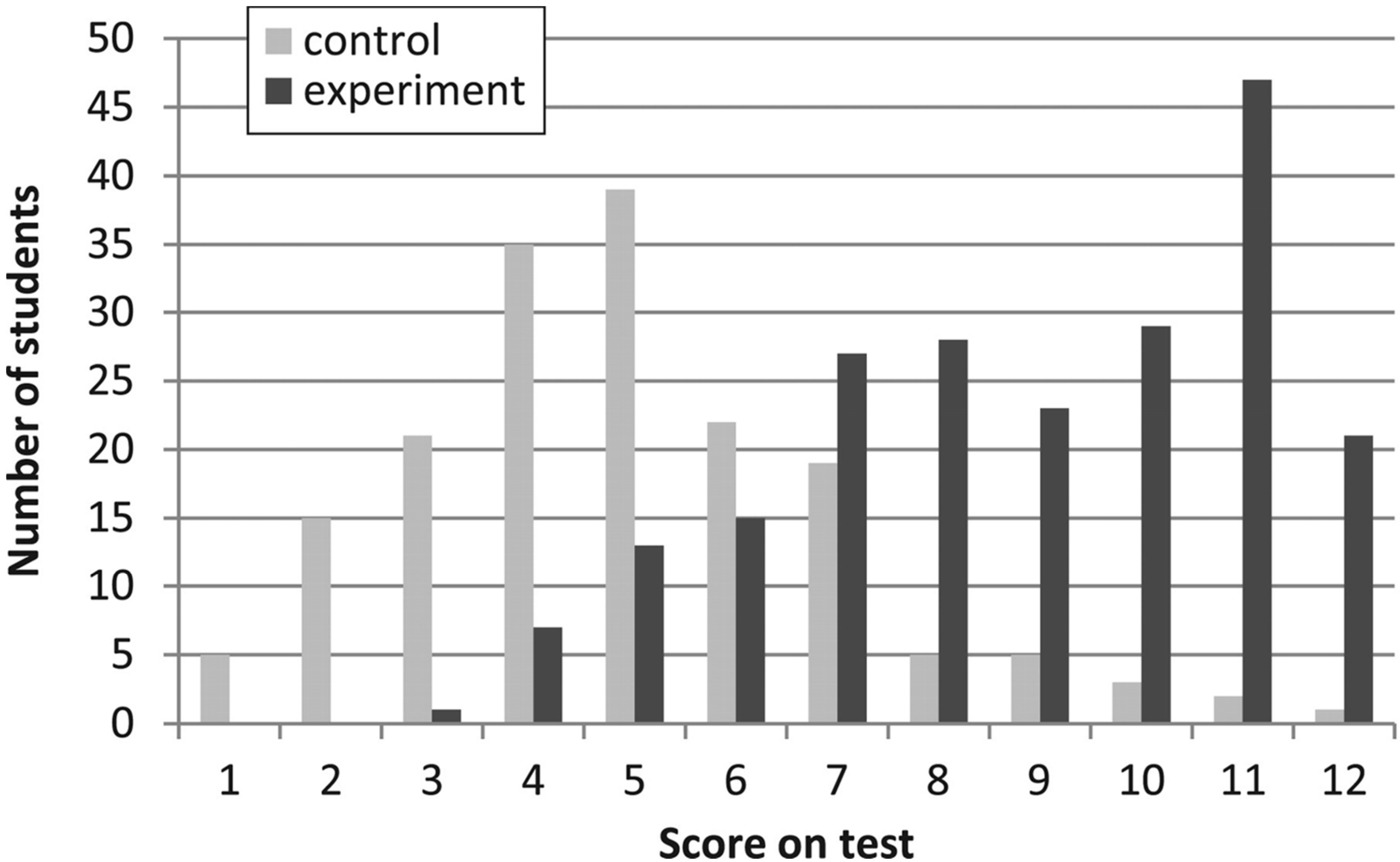
Active Learning

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.

Published by AAAS

Administrivia

http://www.cs.cornell.edu/courses/cs3410/2013sp

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

Lab Sections (start ***today***)

- Labs are separate than lecture and homework

- Bring laptop to Labs (optional)

Administrivia

http://www.cs.cornell.edu/courses/cs3410/2013sp

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

Lab Sections (start ***today***)

T	2:55 – 4:10pm	Carpenter Hall 104 (Blue Room)
W	3:35 – 4:50pm	Carpenter Hall 104 (Blue Room)
W	7:30—8:45pm	Carpenter Hall 235 (Red 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	2:55 – 4:10pm	Carpenter Hall 104 (Blue Room)

- Labs are separate than lecture and homework
- Bring laptop to Labs
- ***This*** week: intro to logisim and building an adder

Communication

Email

- cs3410-staff-l@cs.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/spring2012/cs3410
- For students

iClicker

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

Lab Sections & Projects

Lab Sections start ***this*** week

- Intro to logisim and building an adder

Labs Assignments

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

Projects

- two-person teams
- Find partner in same section

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

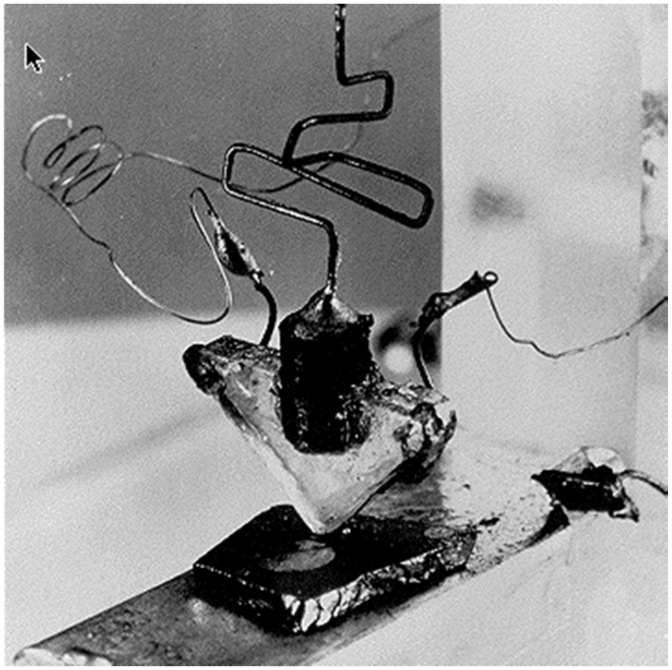
- Same rules apply to projects at the group level
- Cannot use of someone else's soln

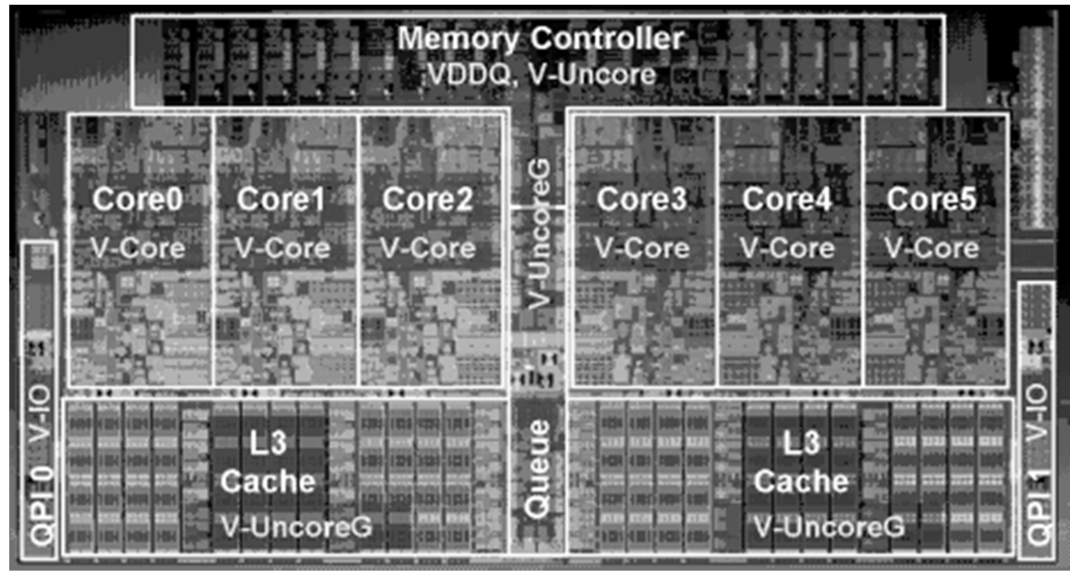
Closed-book exams, no calculators

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

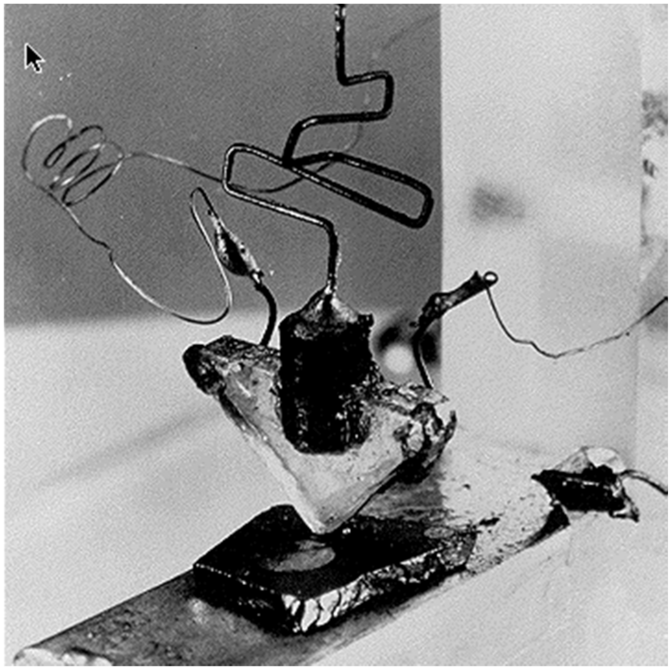
Plagiarism in any form will not be tolerated

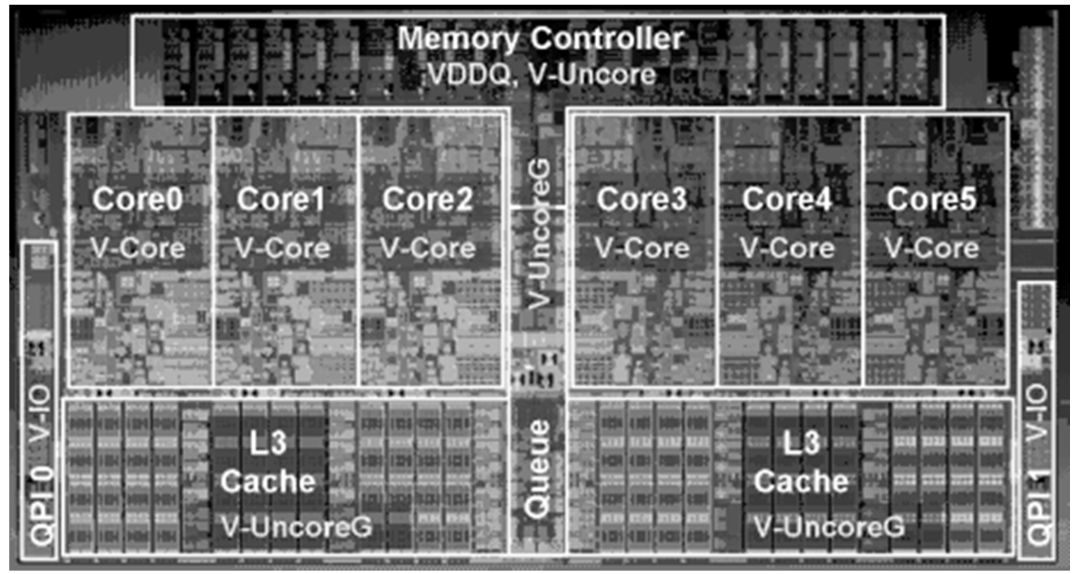
Why do CS Students Need Transistors?





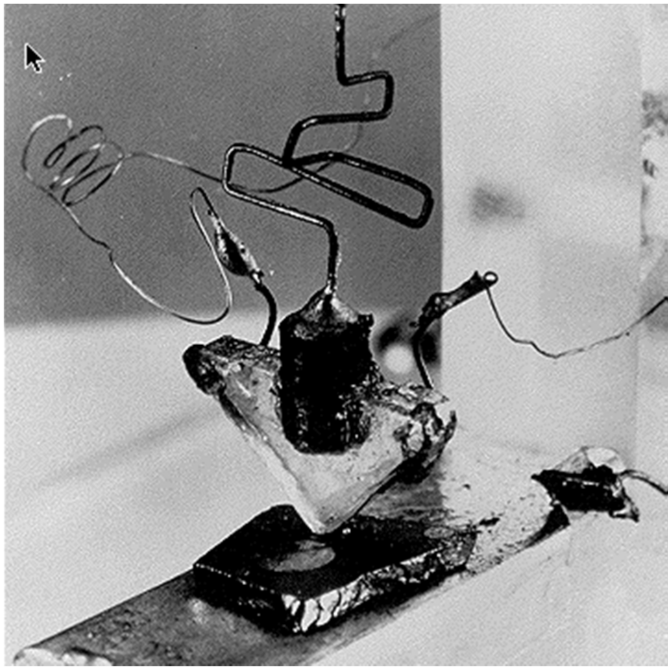
Why do CS Students Need Transistors?

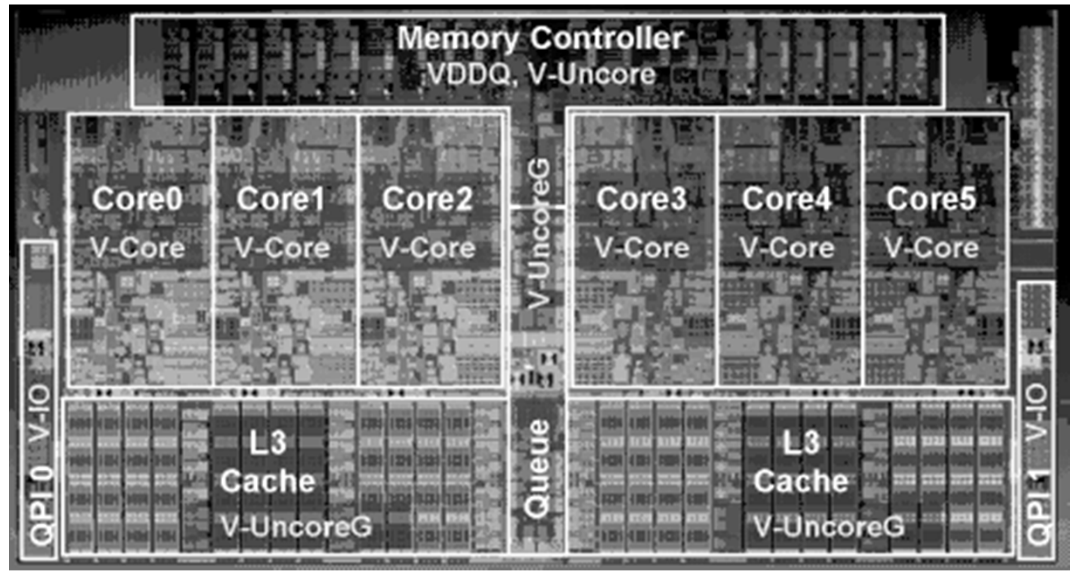




Functionality and Performance

Why do CS Students Need Transistors?

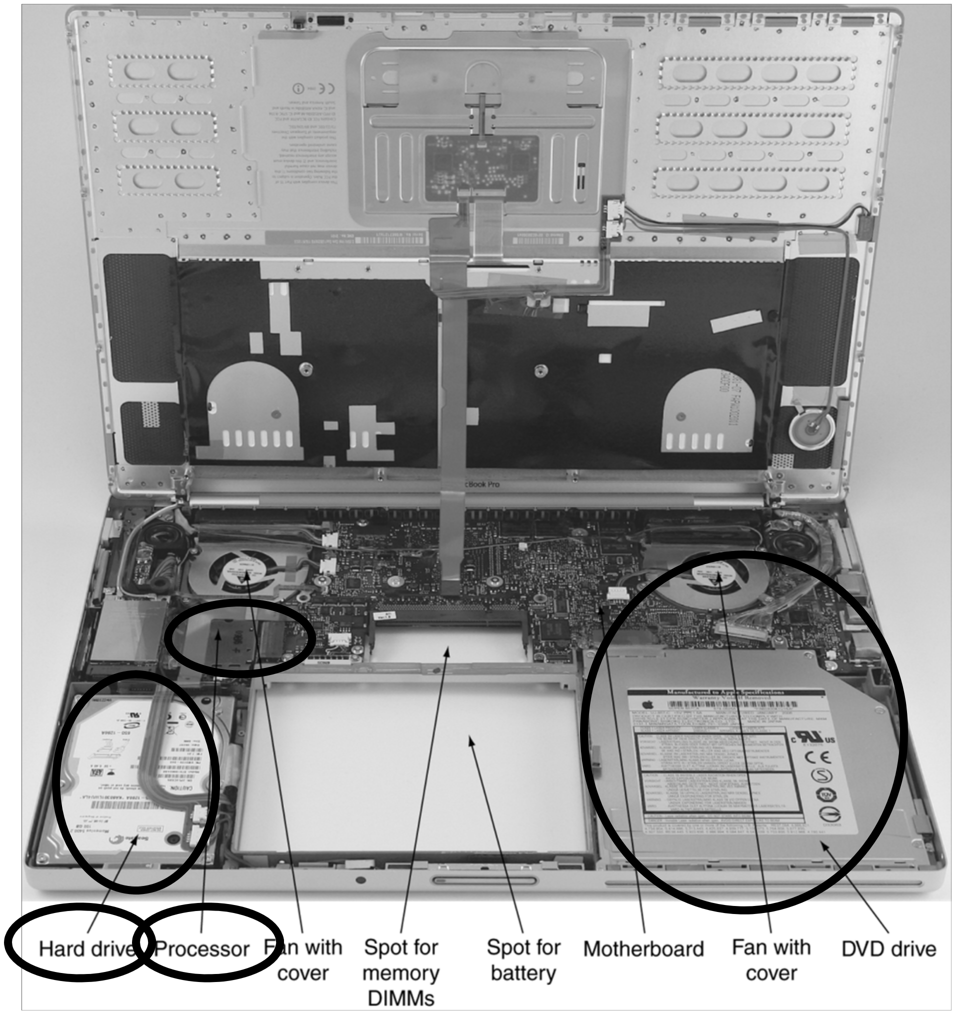


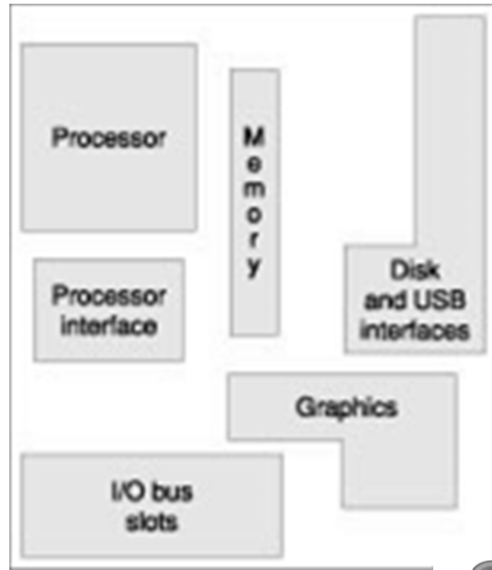


To be better Computer Scientists and Engineers

- 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

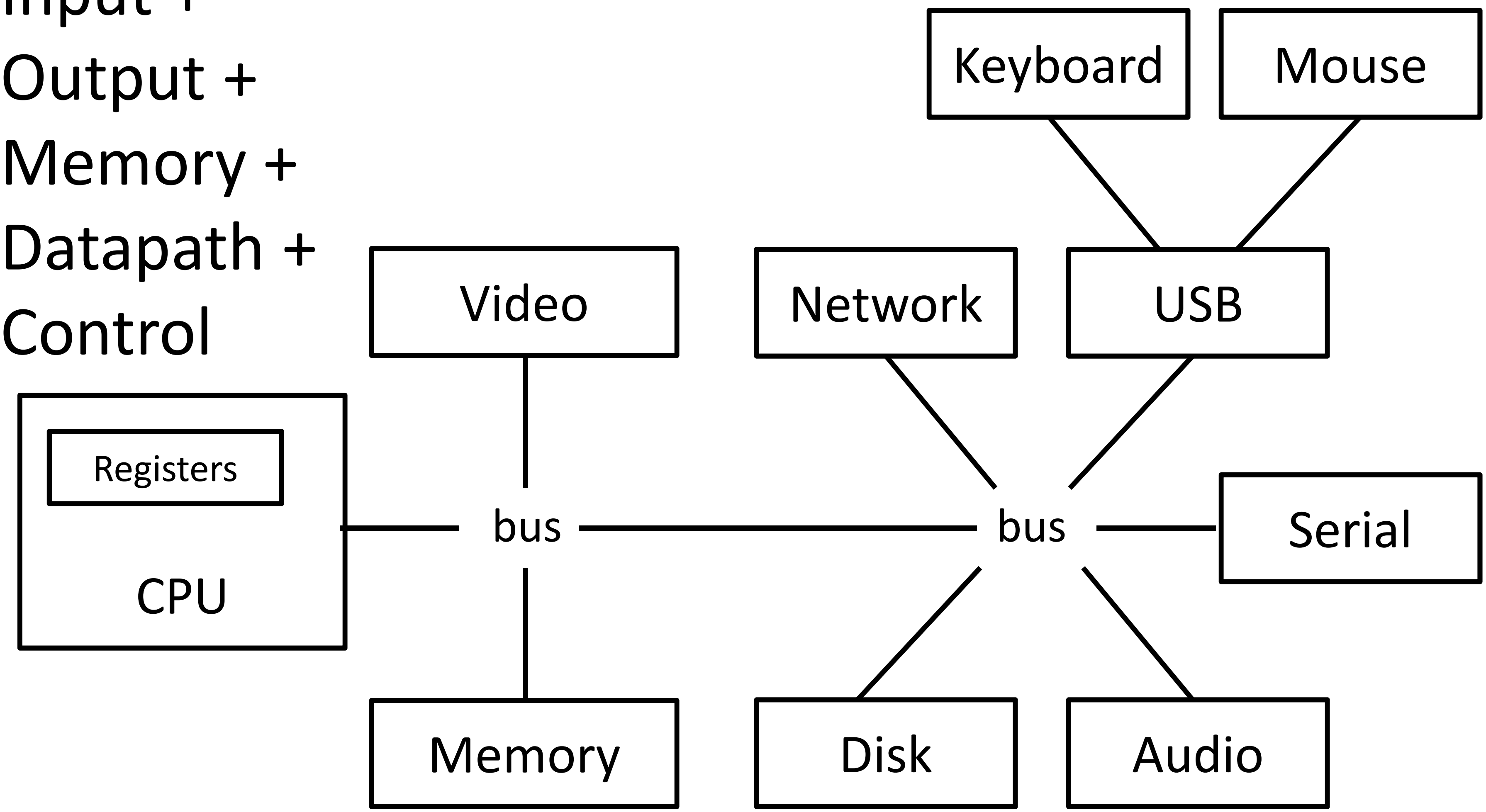




Computer System Organization

Computer System = ?

Input +
Output +
Memory +
Datapath +
Control



Compilers & Assemblers

C

```
int x = 10;
x = 2 * x + 15;
```

compiler

r0 = 0

MIPS assembly language

```
addi r5, r0, 10
muli r5, r5, 2
addi r5, r5, 15
```

r5 = r0 + 10

r5 = r5 * 2

20

r5 = r15 + 15

35

assembler

MIPS machine language

op = addi	r0	r5	10
001000	00000	00101	0000000000001010

00000000000001010010100001000000

001000	00101	00101	0000000000001111
op = addi	r5	r5	15

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

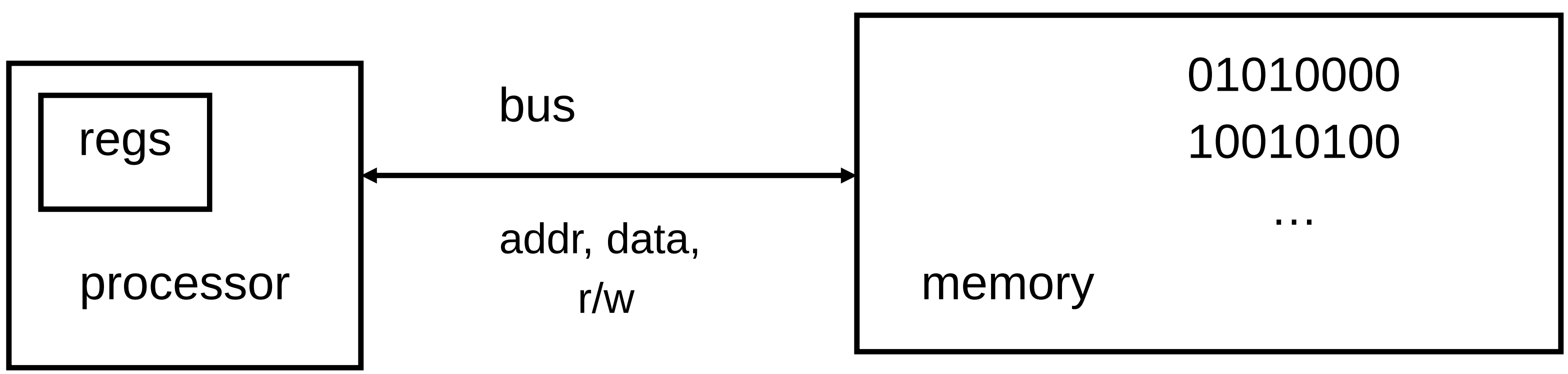
A processor executes instructions

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

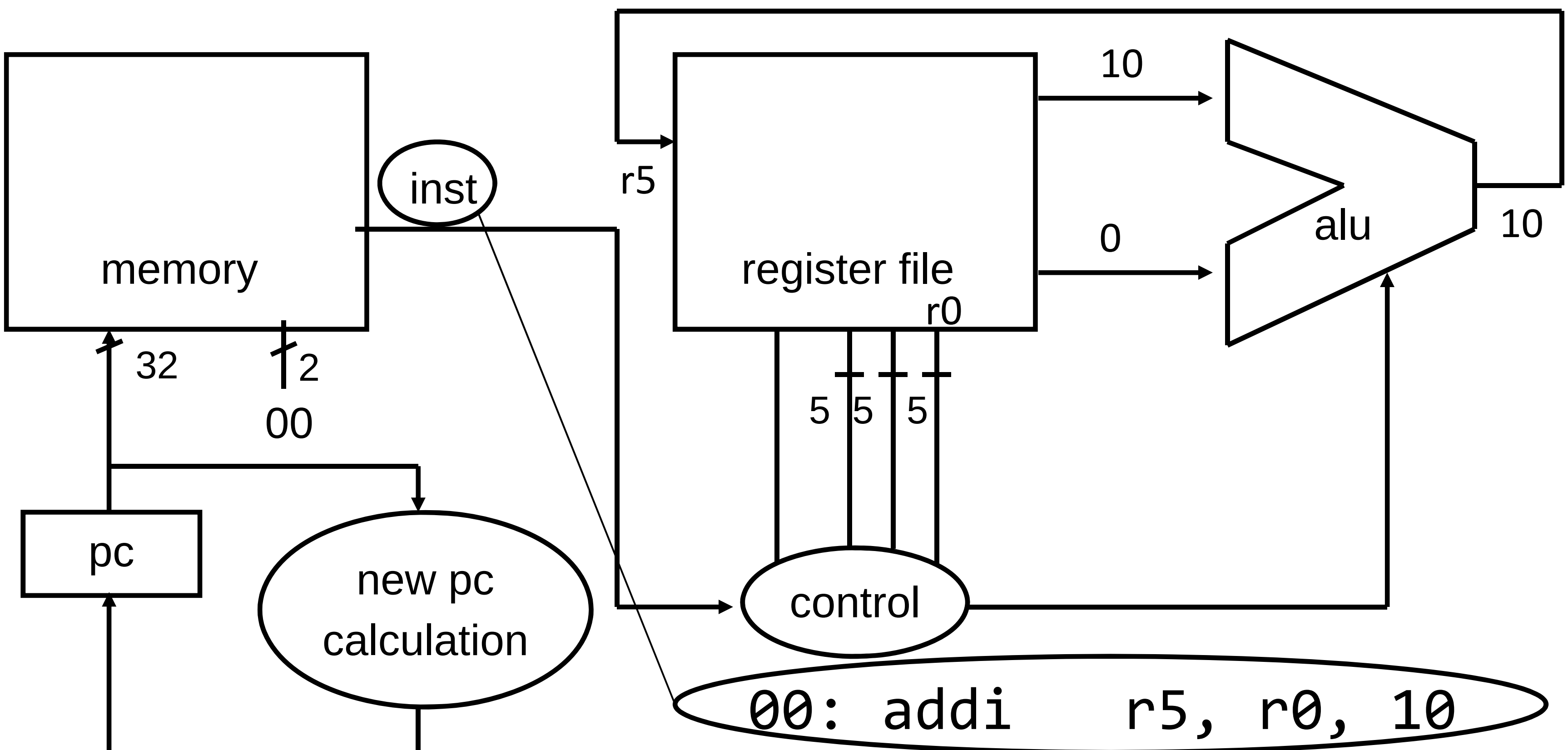
A memory holds instructions and data

- von Neumann architecture: combined inst and data

A bus connects the two



How to Design a Simple Processor




```
00: addi   r5, r0, 10
04: muli   r5, r5, 2
08: addi   r5, r5, 15
```

Inside the Processor

AMD Barcelona: 4 processor cores

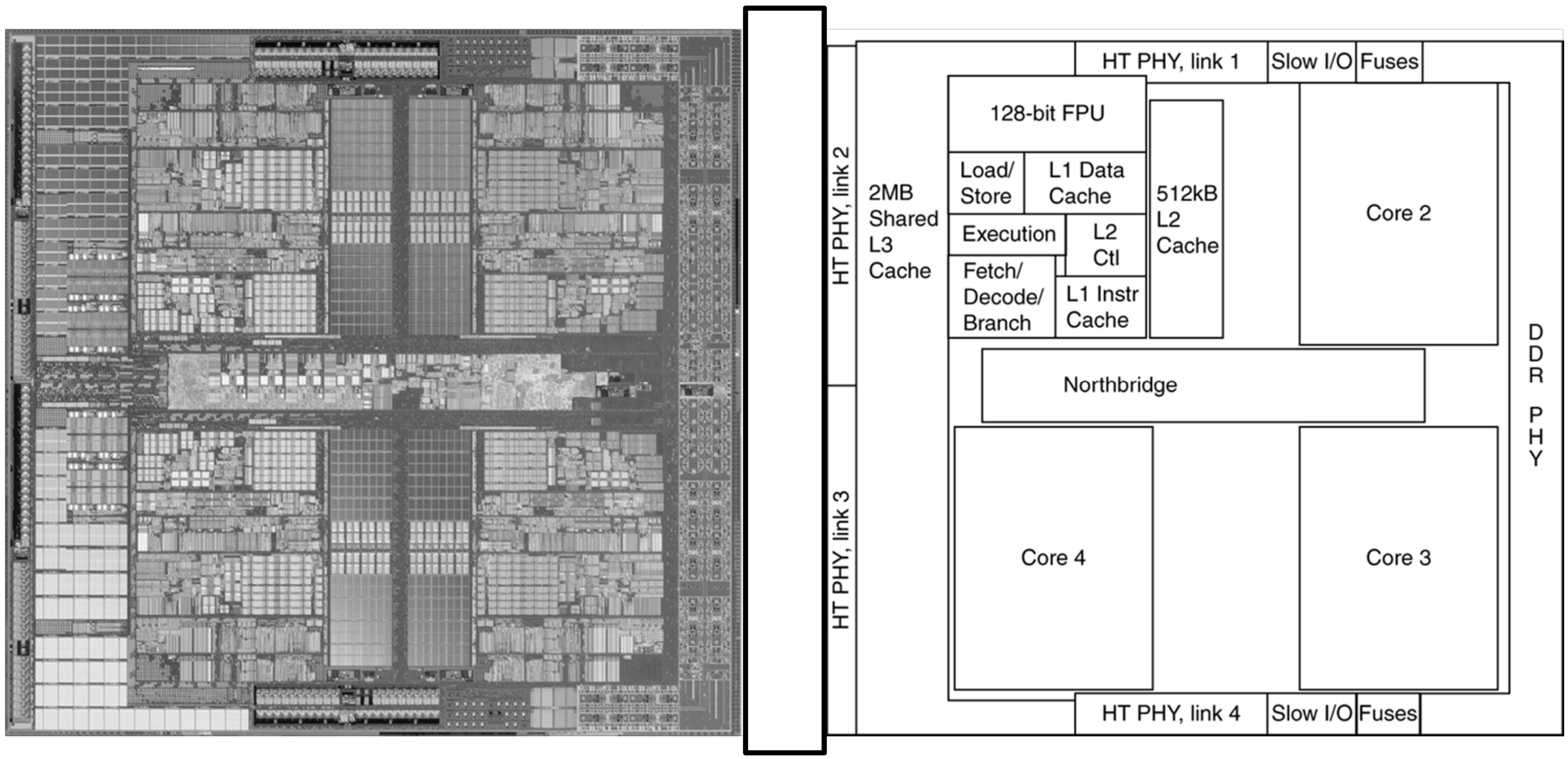


Figure from Patterson & Hennesssy, Computer Organization and Design, 4th Edition

How to Program the Processor: MIPS R3000 ISA

Instruction Categories

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

Registers

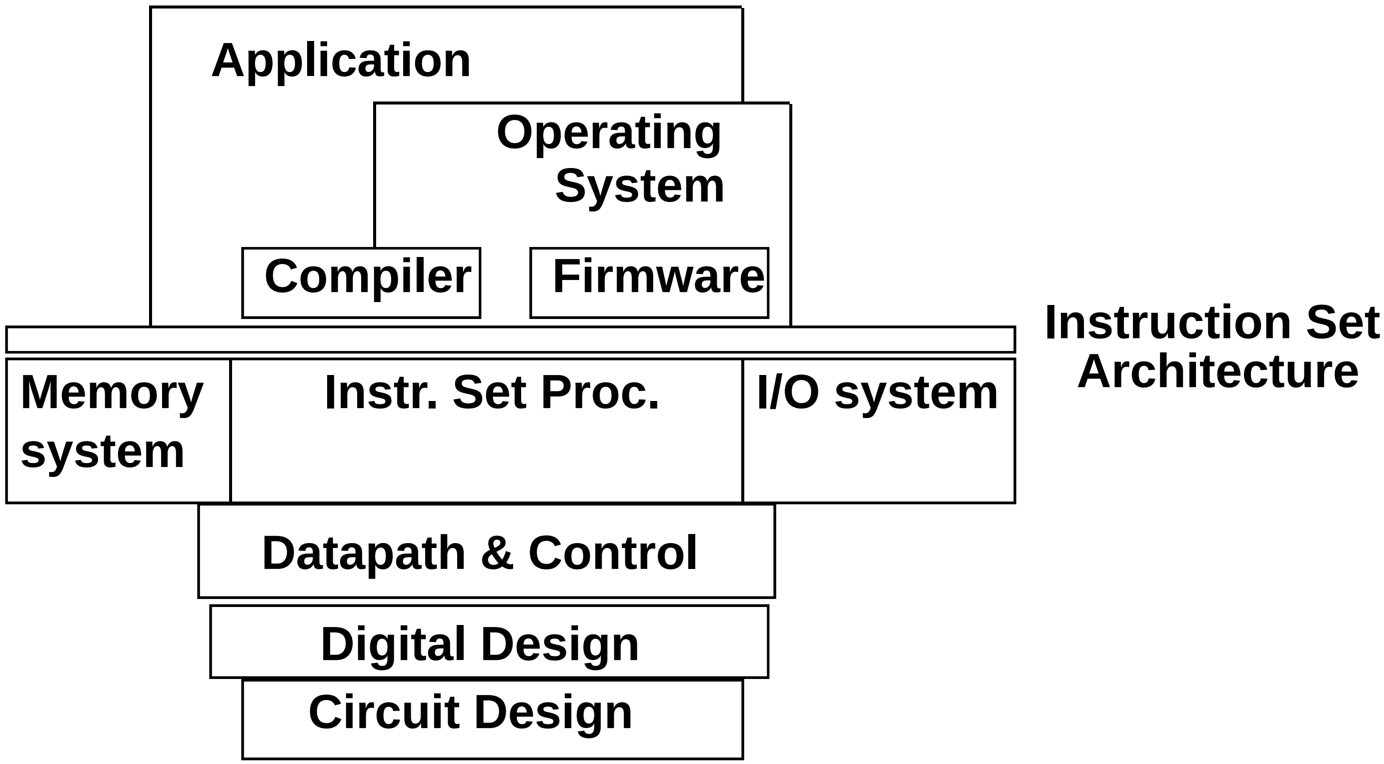
R0 - R31
PC
HI
LO

OP	**rs**	**rt**	**rd**	**sa**	**funct**

OP	**rs**	**rt**	**immediate**

OP	**jump target**

Overview



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

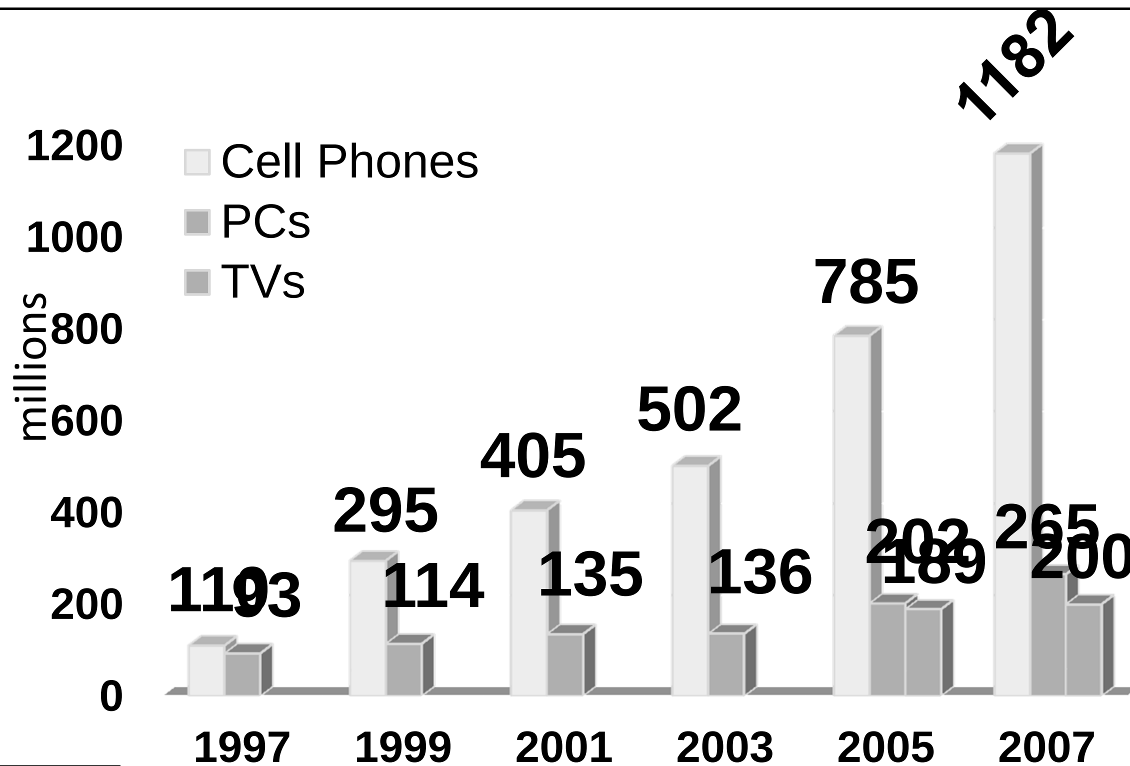
Applications

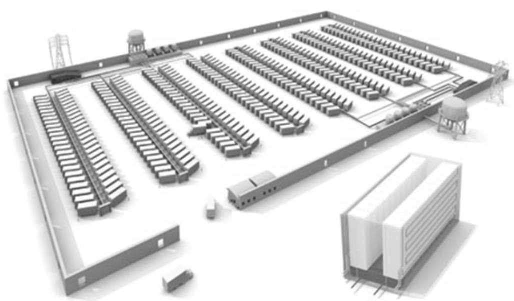
Everything these days!

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

Applications

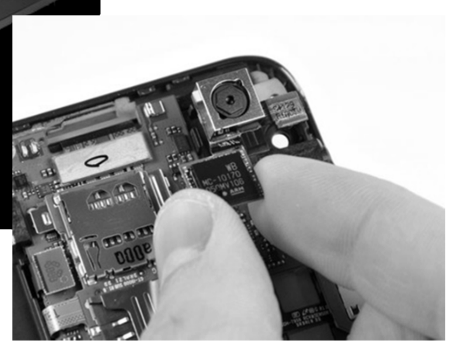
Xilinx FPGA





Cloud Computing

NVidia GPU

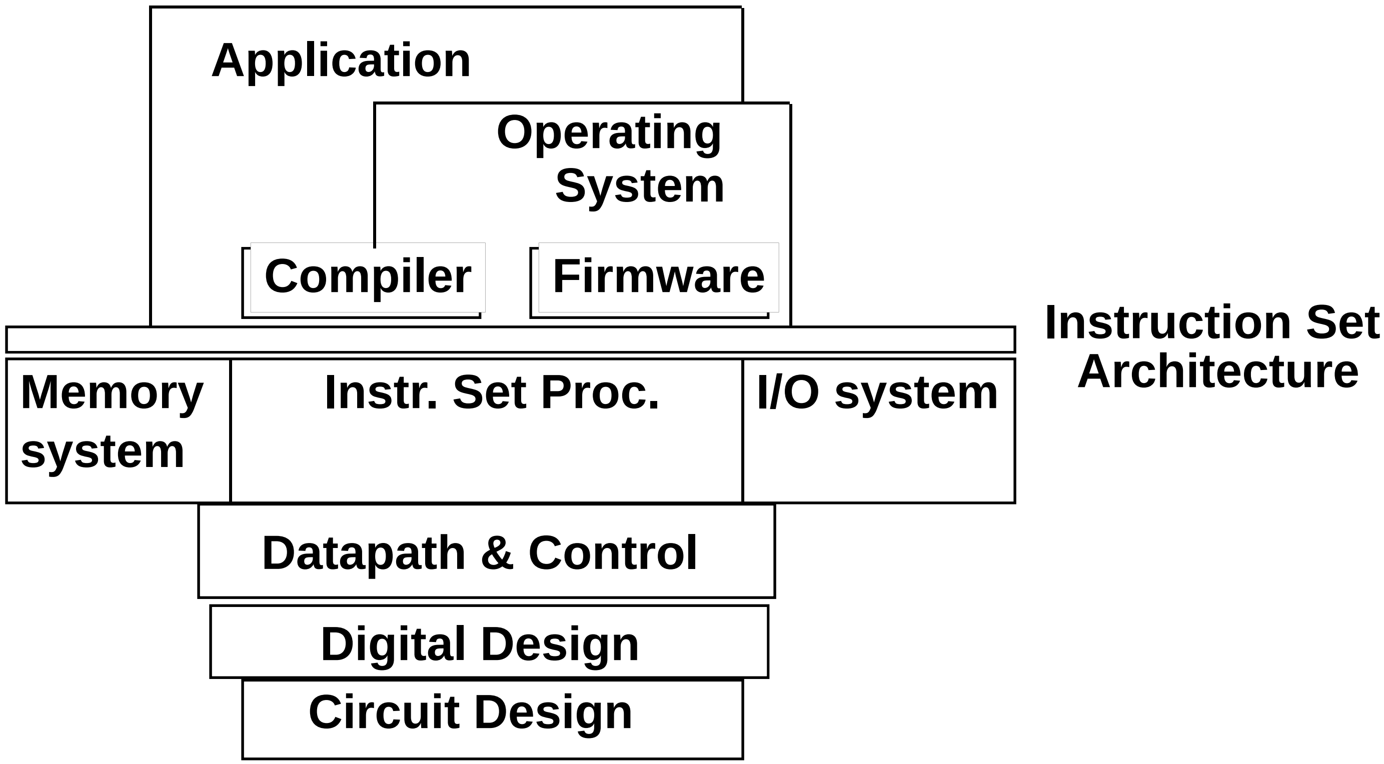


Cell Phone

Berkeley mote

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