CS 3410: Computer System Organization and Programming

Hakim Weatherspoon

Spring 2012

Computer Science

Cornell University

Computer System Organization

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

Computer Organization

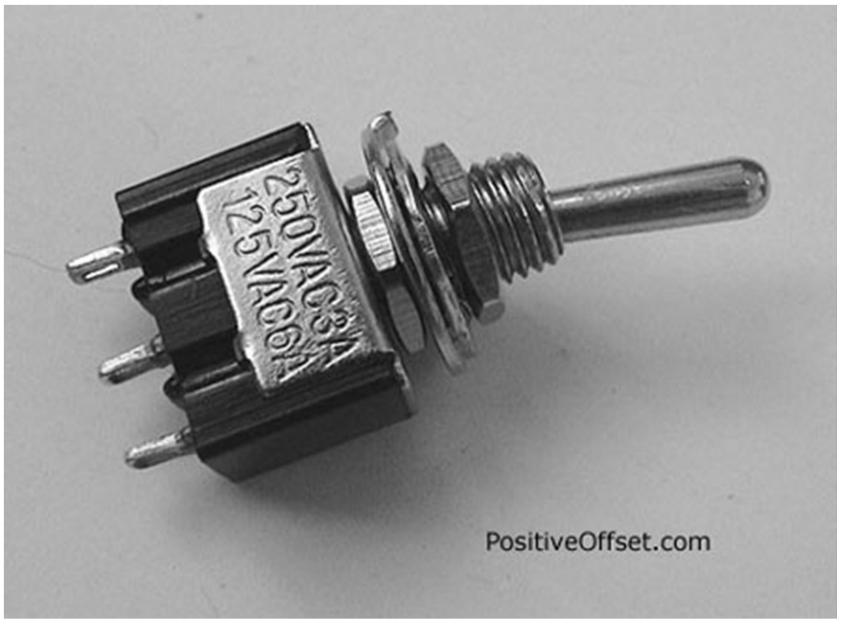
- The most amazing and likely to be most long-lived invention of the 1800's was…
  - (a) The steam engine?
  - (b) The lightning rod?
  - (c) The carbonated beverage?
  - (d) All of the above
  - (e) None

Computer Organization

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

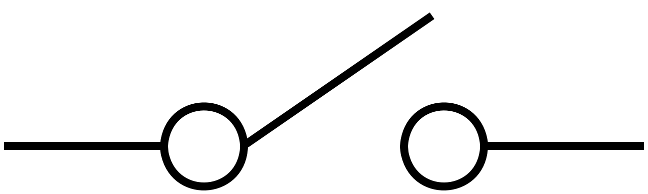
THE ELECTRIC SWITCH

Basic Building Blocks: A switch



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

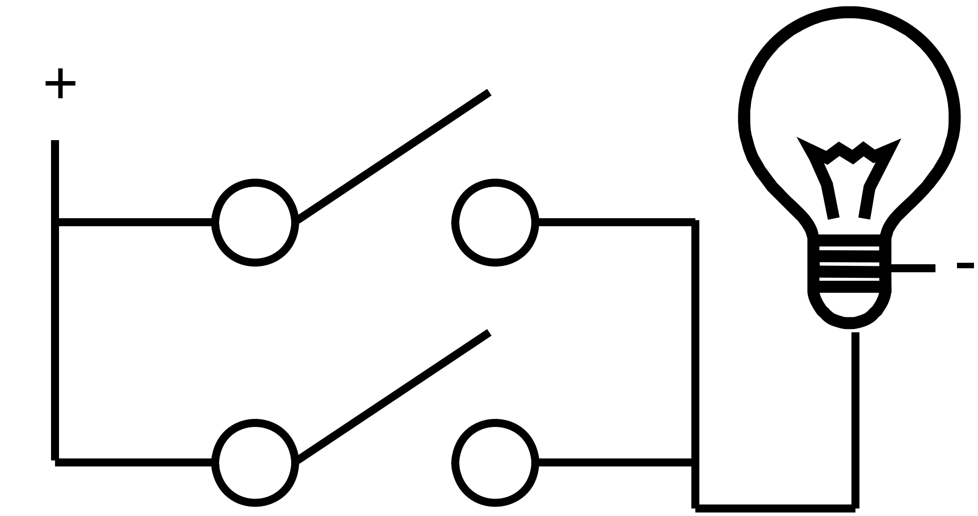
- Can be used for amazing things…



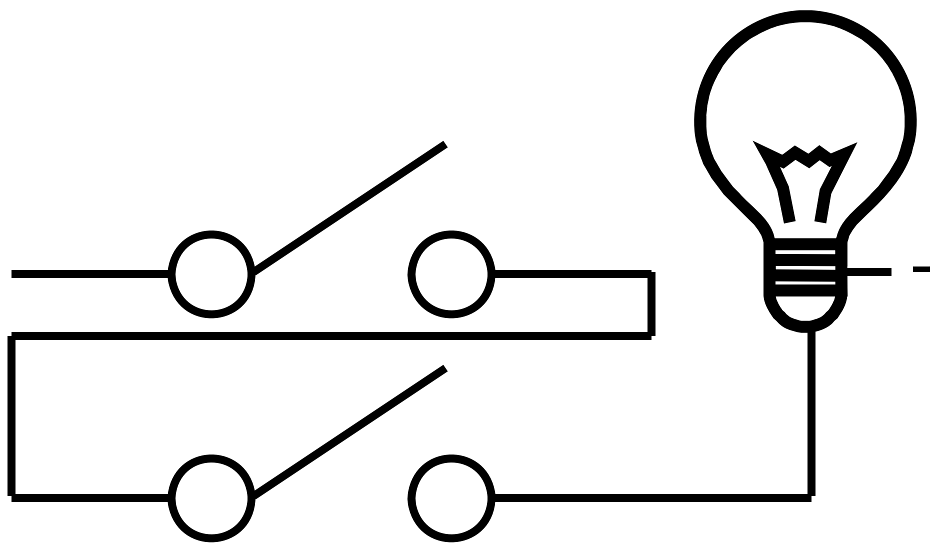
In what language do computers think?

- (a) Java
- (b) C/C++
- (c) Matlab
- (c) Python
- (d) Binary Digits

Basic Building Blocks: Switches

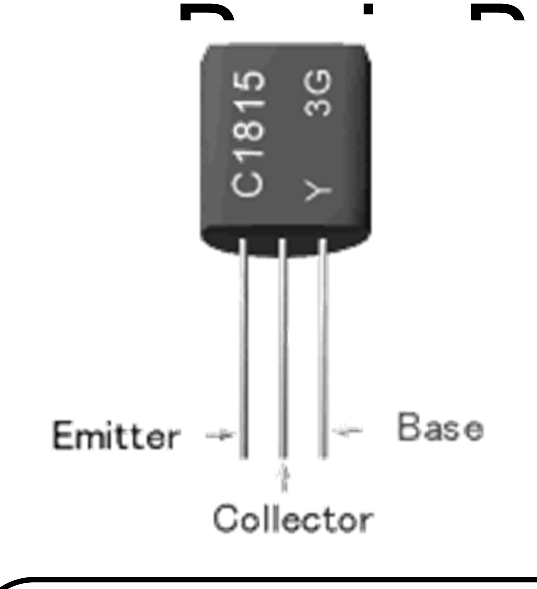


- Either (OR)

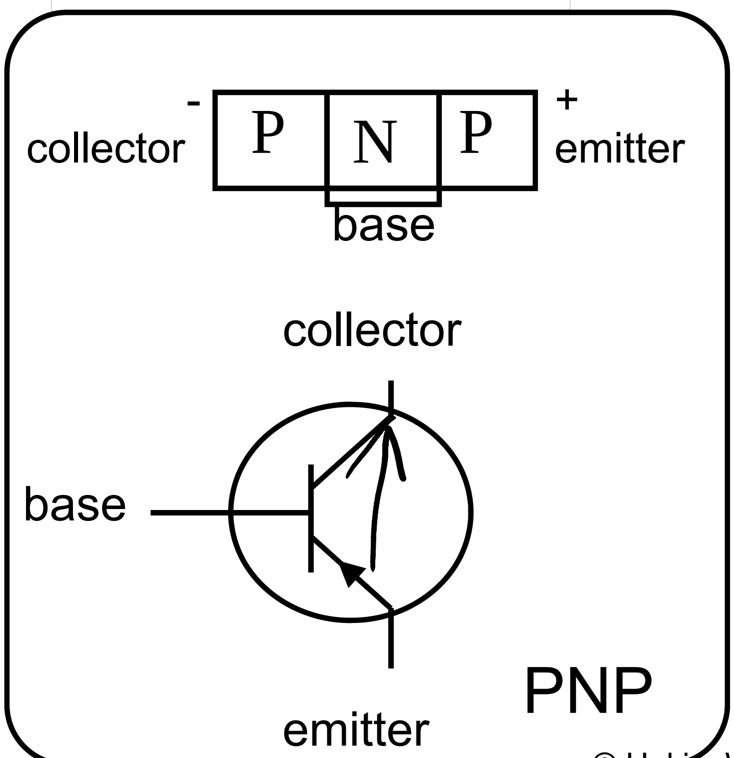


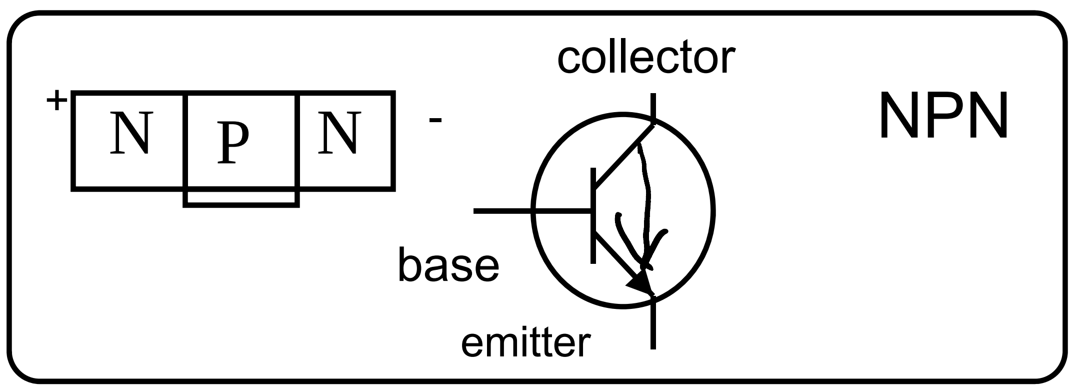
- Both (AND)
- But requires mechanical force

Basic Building Blocks: Transistors



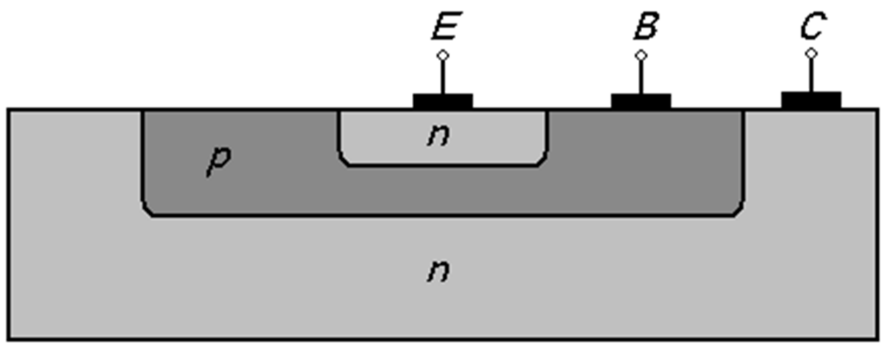
- Solid-state switch
  - The most amazing invention of the 1900s
- PNP and NPN

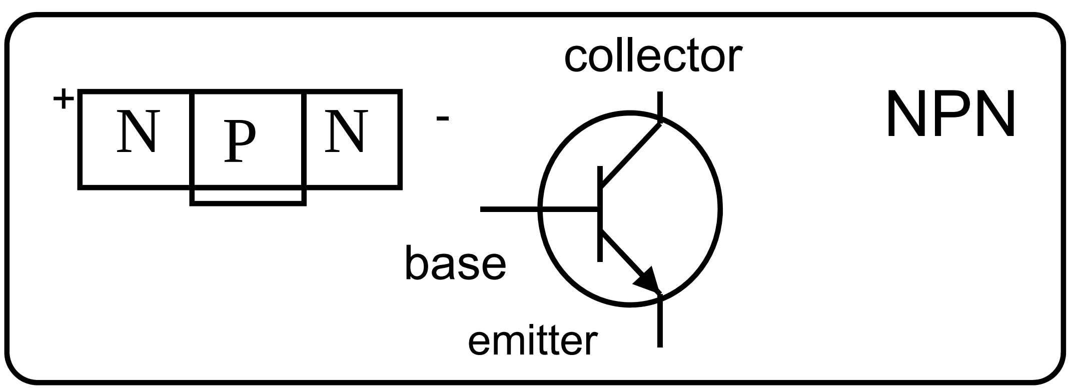


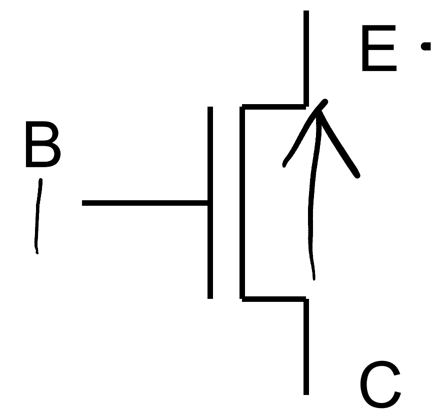


Basic Building Blocks: NPN Transistors

- Semi-conductor



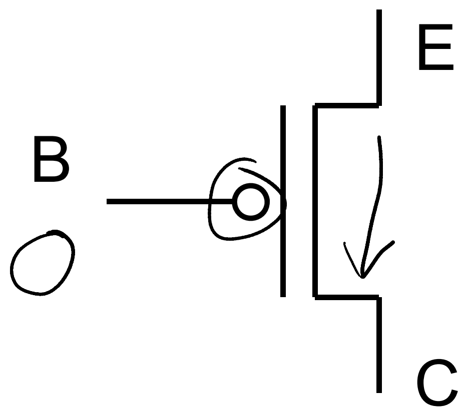




- Connect E to C when base = 1

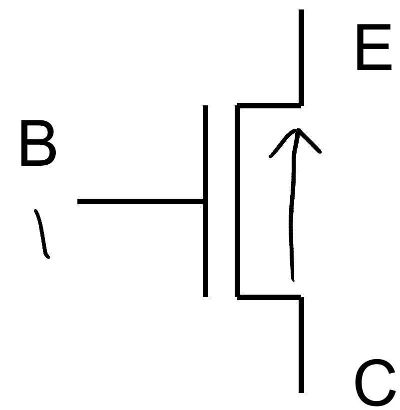
P and N Transistors

- PNP Transistor



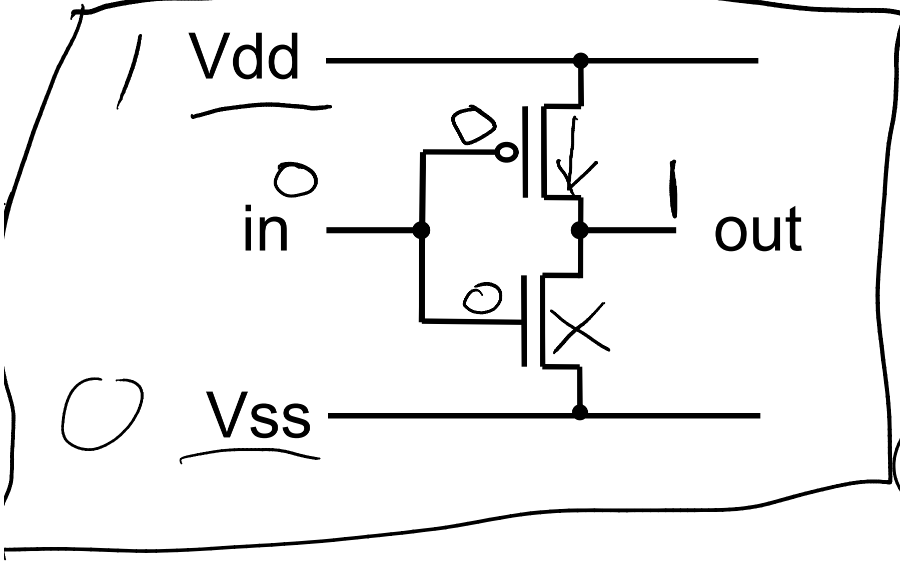
- Connect E to C when base = 0

- NPN Transistor

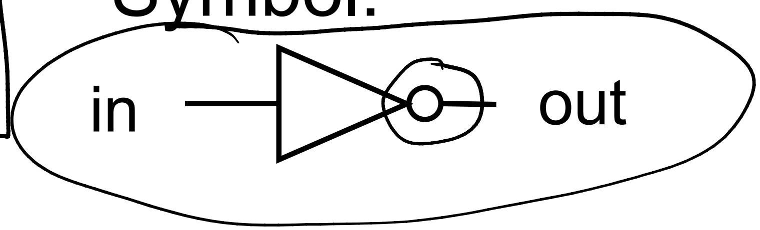


- Connect E to C when base = 1

Inverter



- Function: NOT
- Called an inverter
- Symbol:



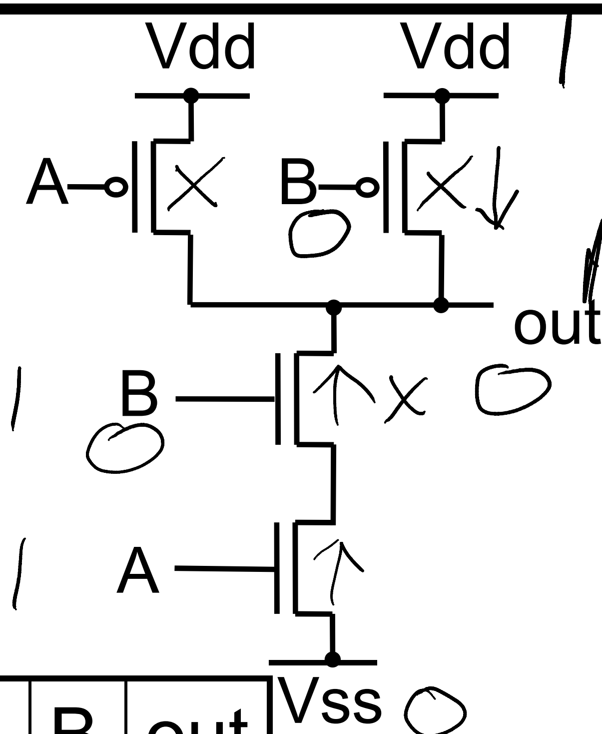
In	Out
0	1
1	0

Truth table

- Useful for taking the inverse of an input

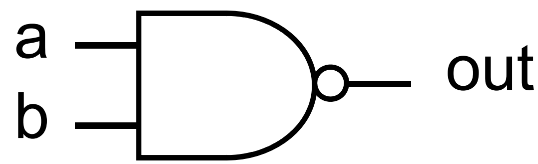
- CMOS: complementary-symmetry metal**–oxide–semiconductor**

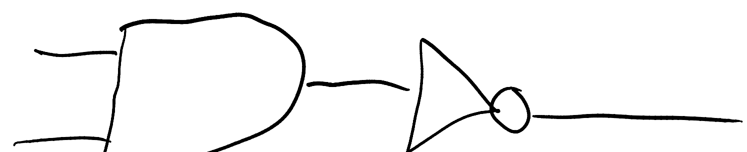
NAND Gate



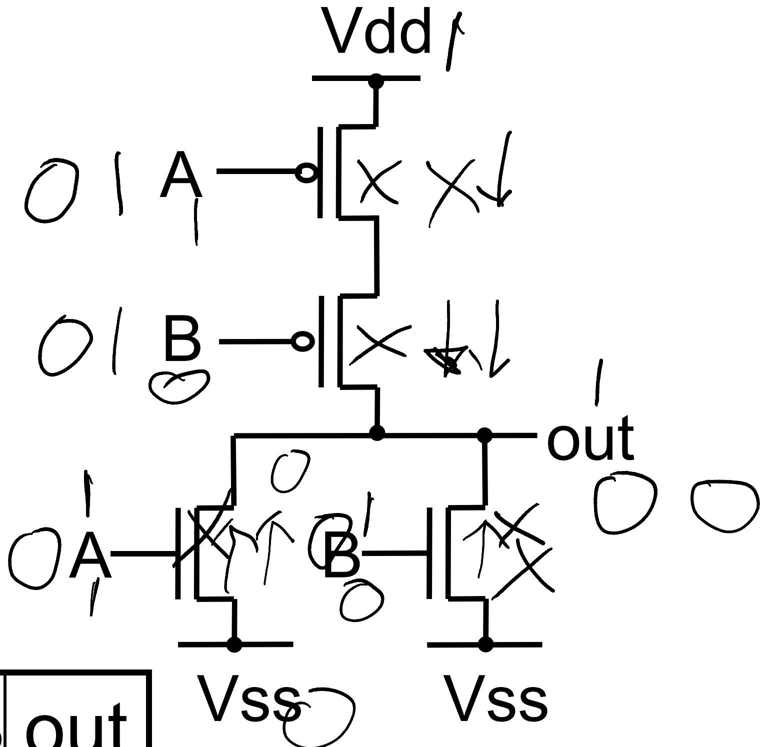
A	B	out
0	0	1
1	0	1
0	1	1
1	1	0

- Function: NAND
- Symbol:

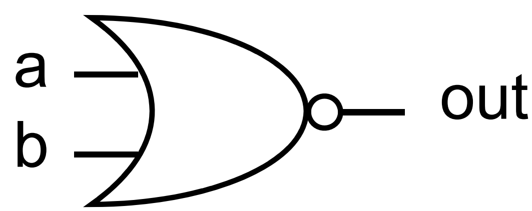


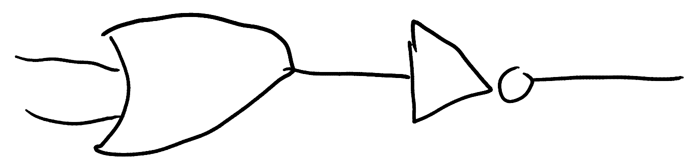


NOR Gate



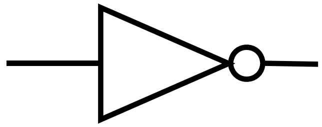
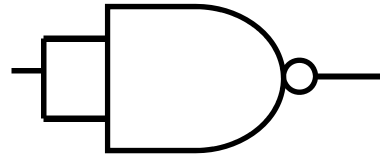
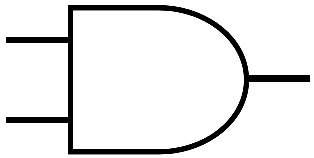
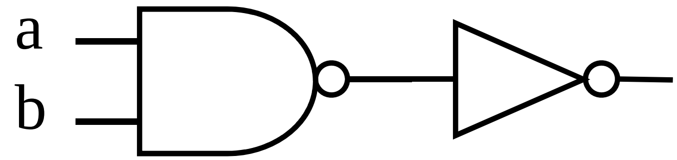
- Function: NOR
- Symbol:

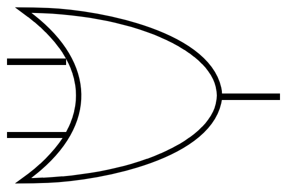
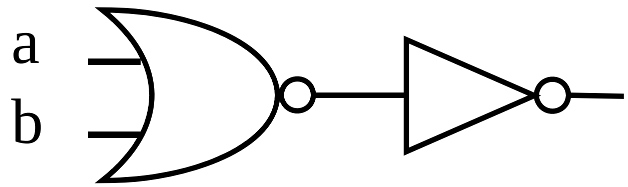




A	B	out
0	0	1
1	0	0
0	1	0
1	1	0

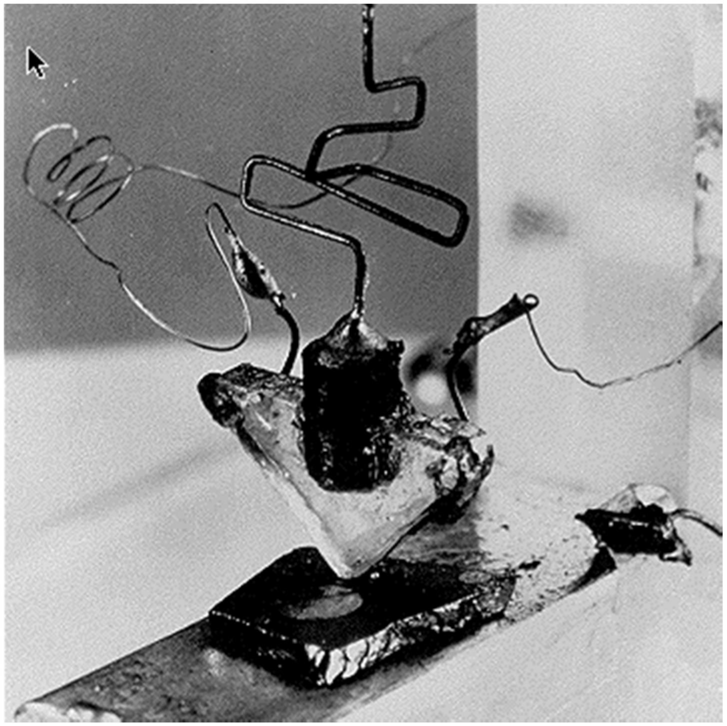
Building Functions

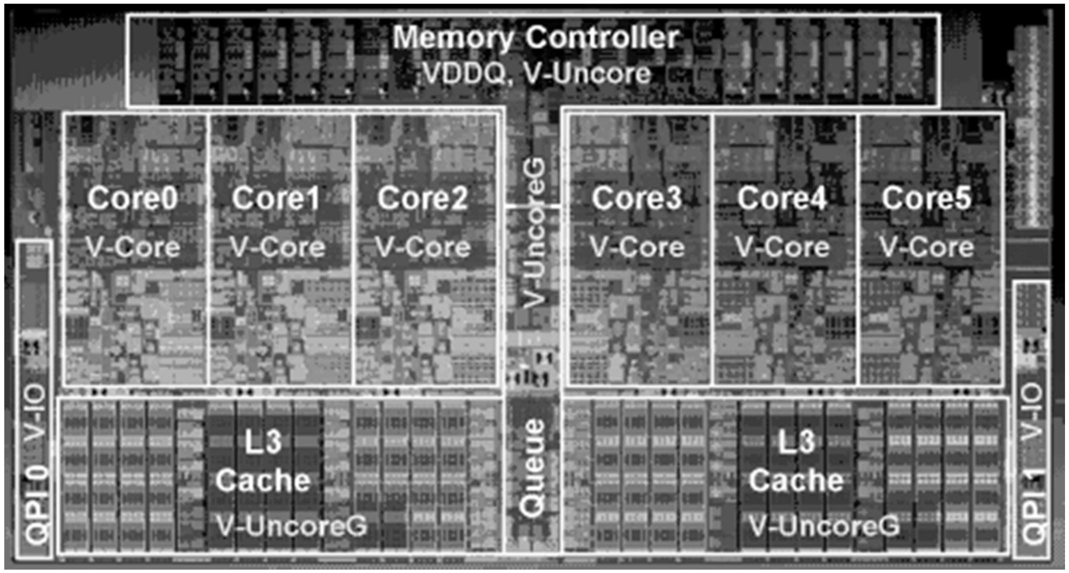
- NOT:  
- AND:  

- 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
  - Six processing cores

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

Course Objective

- Bridge the gap between hardware and software
  - How a processor works
  - How a computer is organized

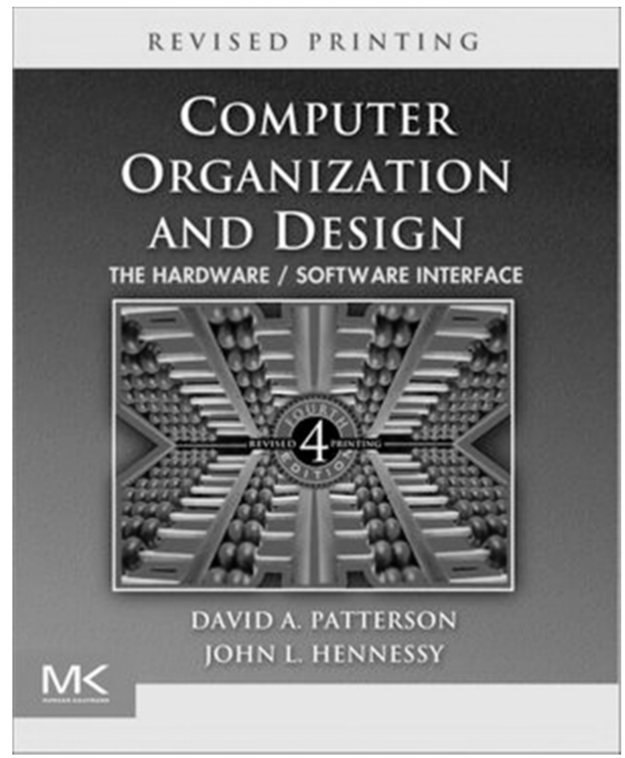
- Establish a foundation for building higher-level applications
  - How to understand program performance
  - How to understand where the world is going

Announcements: How class organized

- Instructor: Hakim Weatherspoon (hweather@cs.cornell.edu)

- Lecture:
  - Tu/Th 1:25-2:40
  - Hollister B14

- Lab Sections:
  - Carpenter 235 (Red Room)

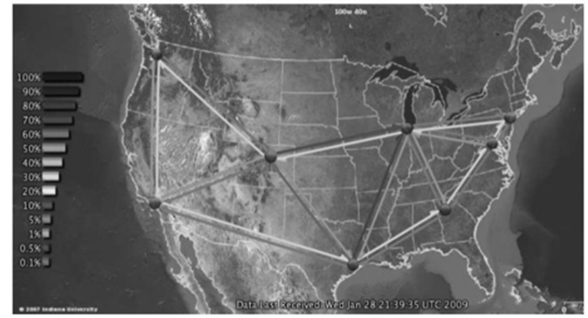


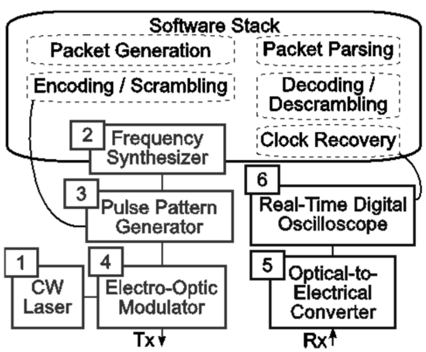
Who am I?

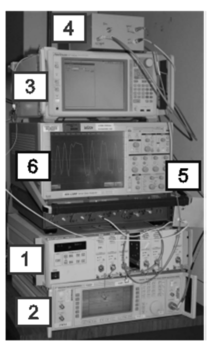
- Prof. Hakim Weatherspoon
  - (Hakim means Doctor, wise, or prof. in Arabic)
  - Background in Education
    - Undergraduate University of Washington
      - Played Varsity Football
        - Some teammates collectively make $100's of millions
        - I teach!!!
    - Graduate University of California, 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

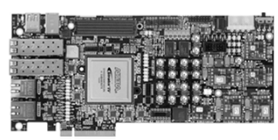
Who am I?

- 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
  - ***Colin Ponce*** ***(cponce@cs.cornell.edu)*** ***(lead)***
  - Anish Ghulati (ag795@cornell.edu)
  - Ming Pan (mp492@cornell.edu)

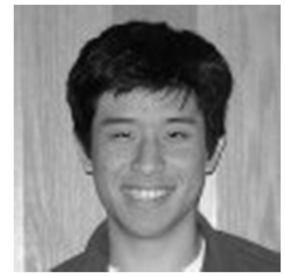
- Lab TAs
  - ***Han Wang*** ***(hwang@cs.cornell.edu)*** ***(lead)***
  - Zhefu Jiang (zj46@cs.cornell.edu)

- Lab Undergraduate consultants
  - Doo San Baik (db478@cornell.edu)
  - Erluo Li (el378@cornell.edu)
  - Jason Zhao (jlz27@cornell.edu)
  - ***Peter Tseng*** ***(pht24@cornell.edu)*** ***(lead)***
  - Roman Averbukh (raa89@cornell.edu)
  - Scott Franklin (sdf47@cornell.edu)

Administrative Assistant:

- Randy Hess (rbhess@cs.cornell.edu)

Course Staff

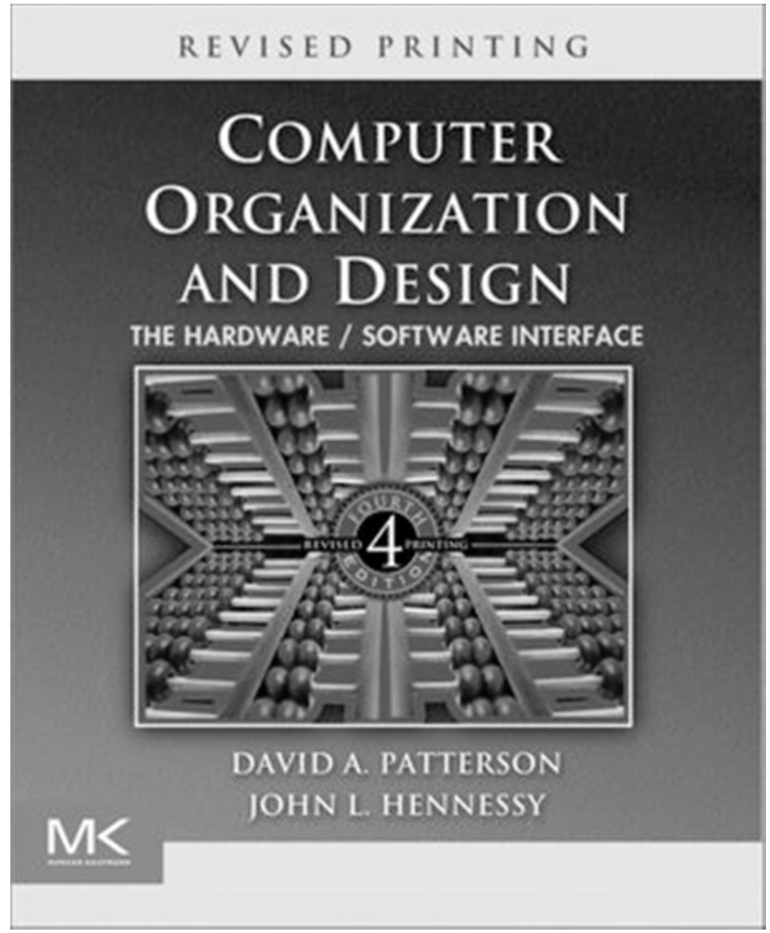


Doo San Baik Roman Averbukh Peter Tseng

Book

- Computer Organization and Design
  – The Hardware/Software Interface

- David Patterson, John Hennessy
  – Get the 4th Edition Revised



Pre-requisites and scheduling

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

- CS 3420 (ECE 3140)
  - Take either CS 3410 ***or*** CS 3420
    - both satisfy CS and ECE requirements
  - *However, Need ENGRD 2300 to take CS 3420*

- CS 3110
  - Not advised to take CS 3110 and 3410 together

Grading

- Lab (45-50%)
  - 4-5 Individual Labs (15-20%)
  - 4 Group Projects (30-35%)
- Lecture (45-50%)
  - 3 Prelims (35-40%)
  - Homework (10%)
- Participation/Discretionary (5%)

Prelims
Tu Feb 28
Th Mar 29
Th Apr 26

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
  - 20% deducted per day late after slip days are exhausted

Administrivia

- http://www.cs.cornell.edu/courses/cs3410/2012sp
  - 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/cs3410/2012sp
  - 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 235 (Red Room)
W	3:35 – 4:50pm	Carpenter Hall 235 (Red Room)
W	7:30—8:45pm	Carpenter Hall 235 (Red Room)
R	11:40 – 12:55pm	Carpenter Hall 235 (Red Room)
R	2:55 – 4:10pm	Carpenter Hall 235 (Red Room)
F	2:55 – 4:10pm	Carpenter Hall 235 (Red 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

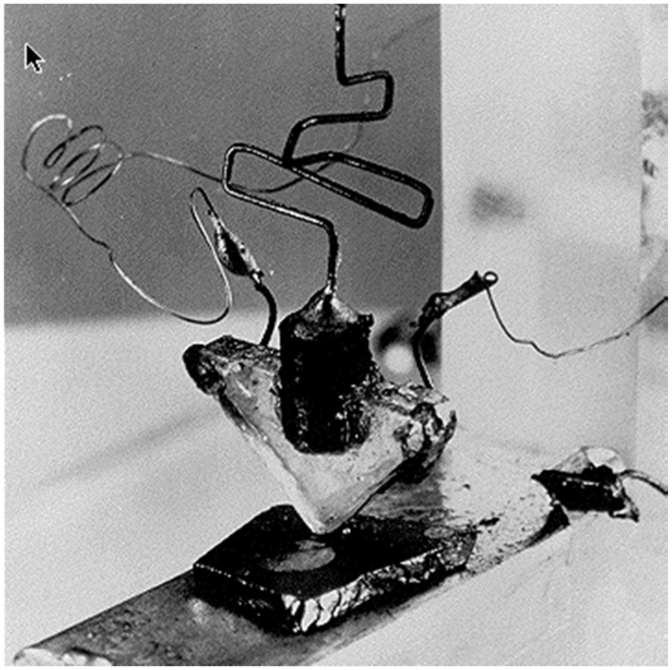
Academic Integrity

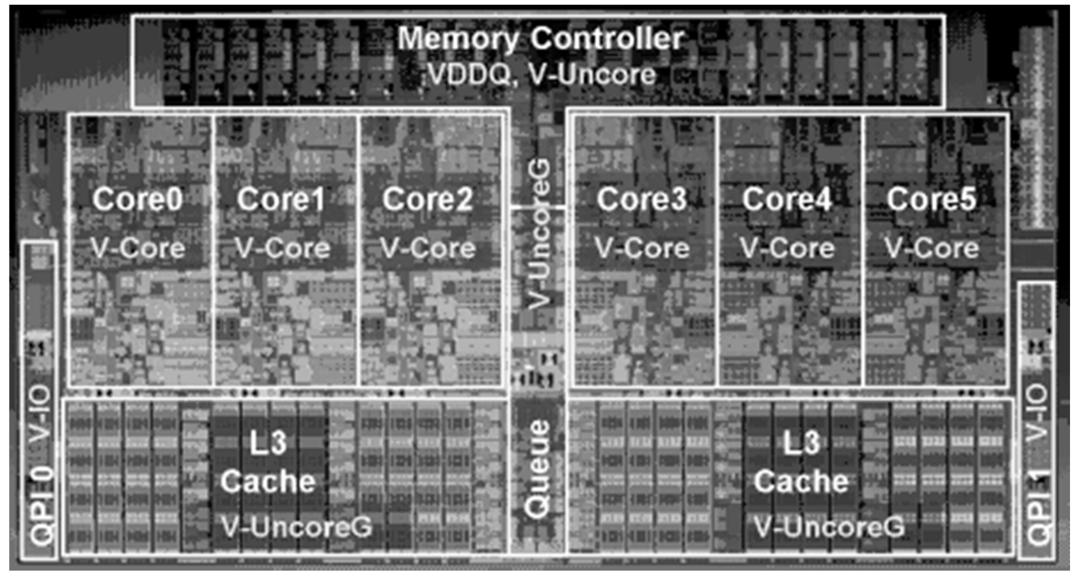
- All submitted work must be your own
  - OK to study together, but do not share soln’s
  - Cite your sources
- Project groups submit joint work
  - Same rules apply to projects at the group level
  - Cannot use of someone else’s soln
- Closed-book exams, no calculators

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

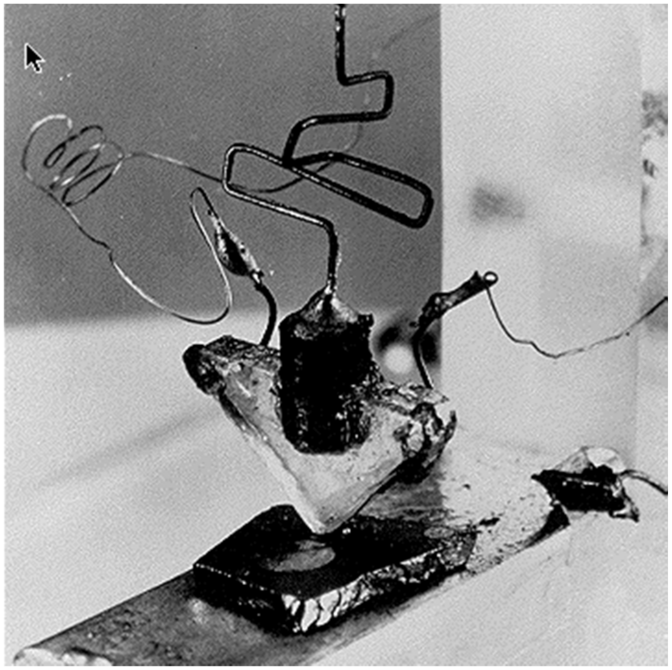
Plagiarism in any form will not be tolerated

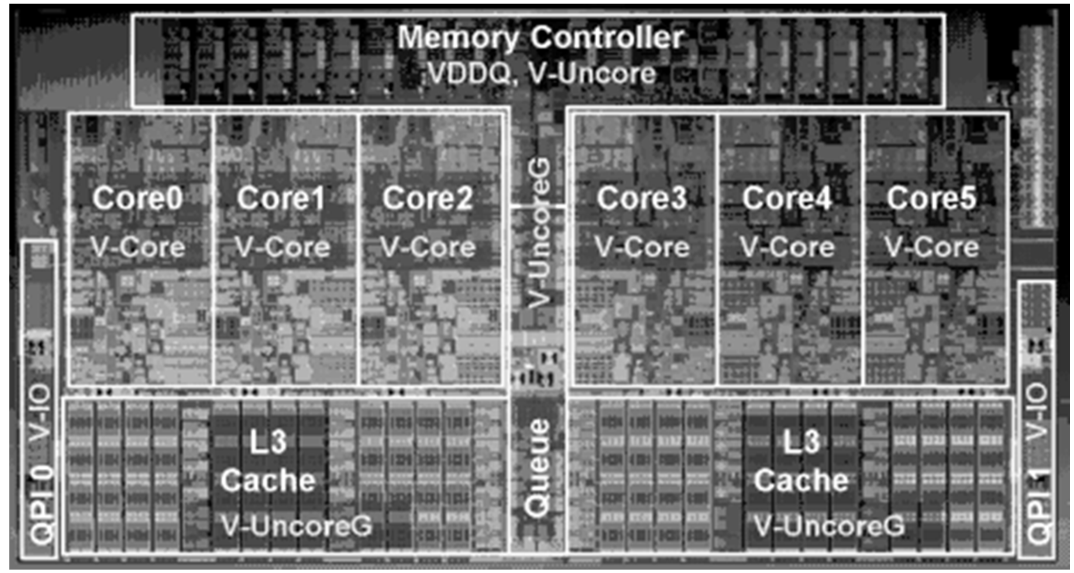
Why do CS Students Need Transistors?





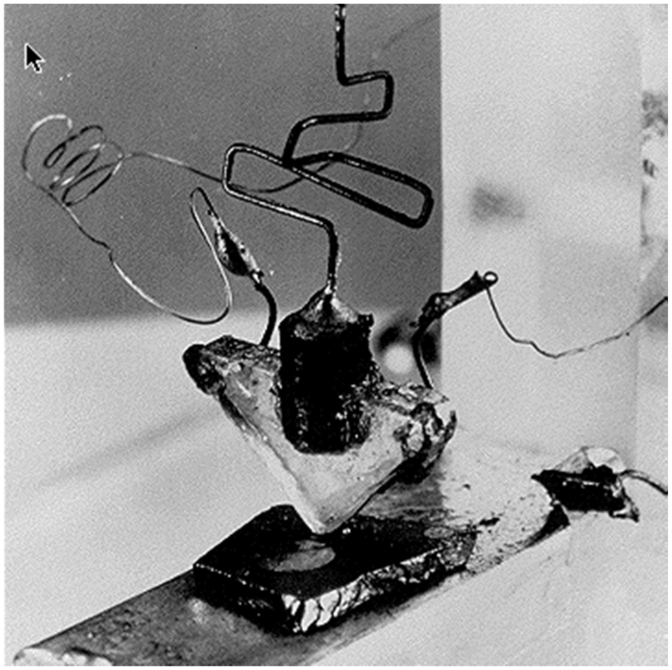
Why do CS Students Need Transistors?

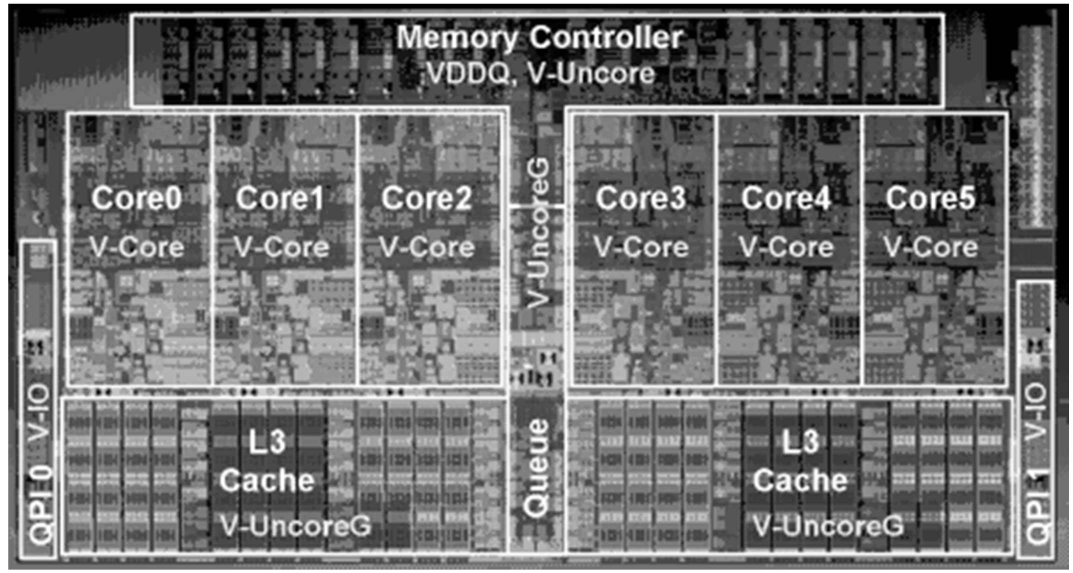




- ***Functionality and Performance***

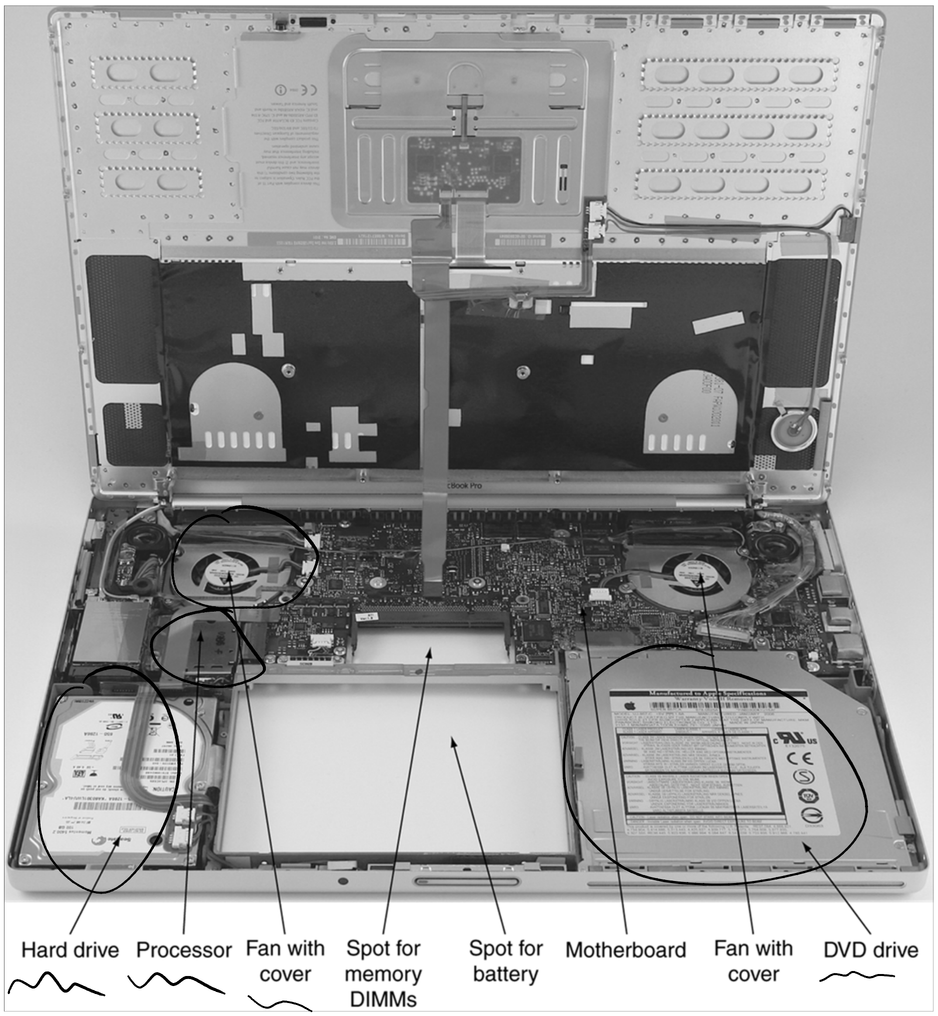
Why do CS Students Need Transistors?

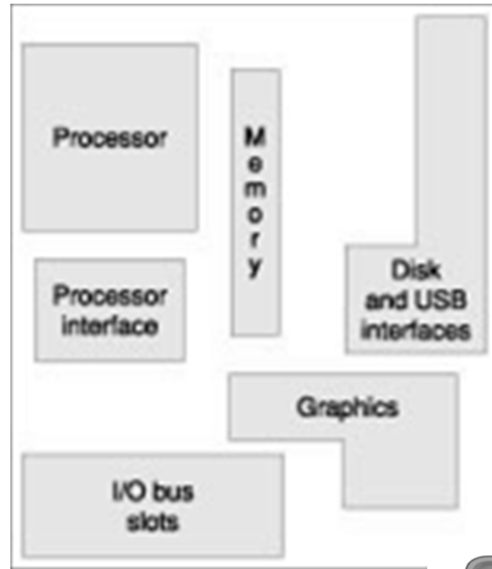


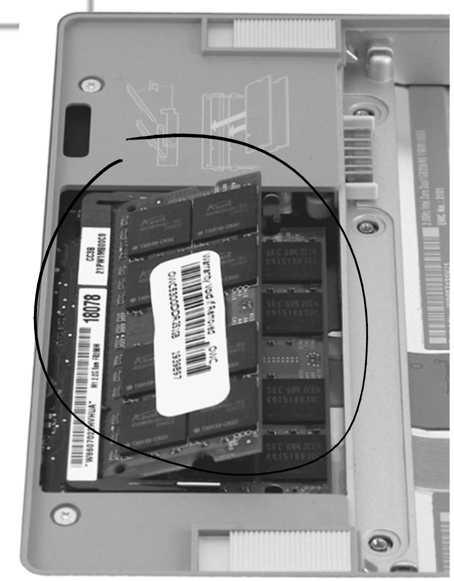


- To be better Computer Scientists and Engineers
  - Abstraction: simplifying complexity
  - How is a computer system organized? How do I build it?
  - How do I program it? How do I change it?
  - How does its design/organization effect performance?

Computer System Organization



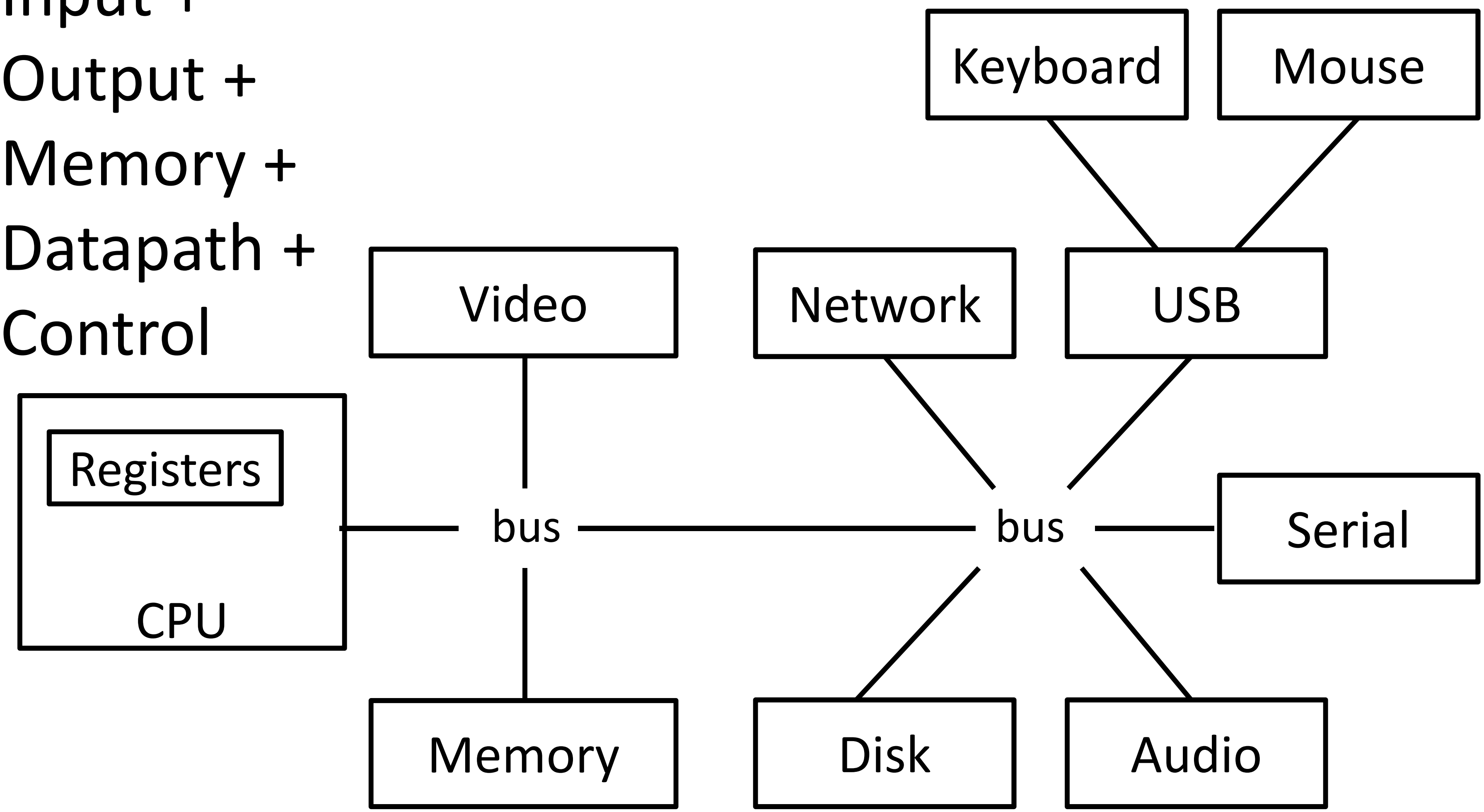




Computer System Organization

Computer System = ?

Input +
Output +
Memory +
Datapath +
Control



Compilers & Assemblers

C

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

r0 = 0

compiler

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

op = addi

r0 r5 10

MIPS machine language

```
00100000000001010000000000001010
00000000000001010010100001000000
00100000101001010000000000001111
```

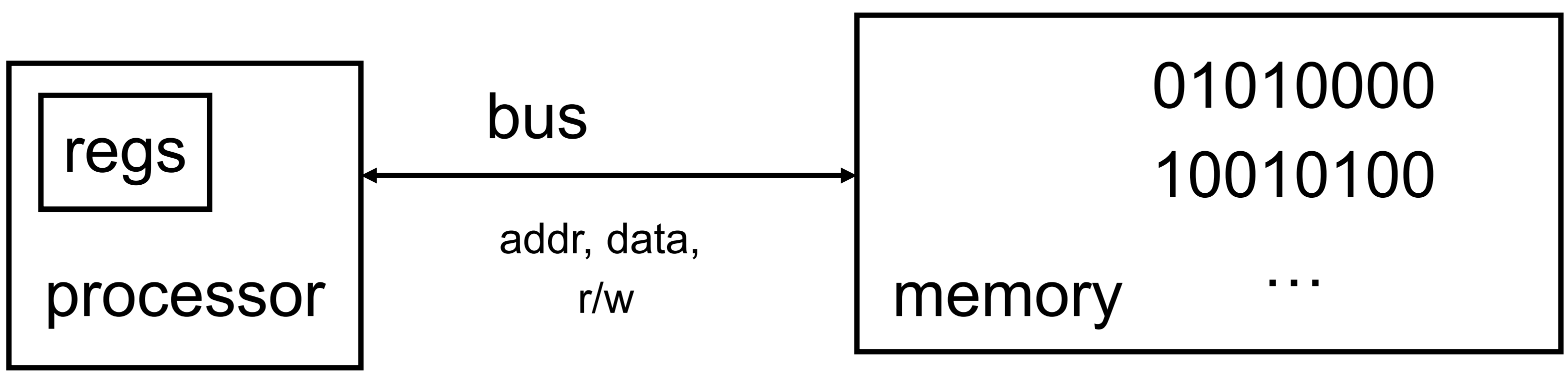
addi r5 r5 15

Instruction Set Architecture

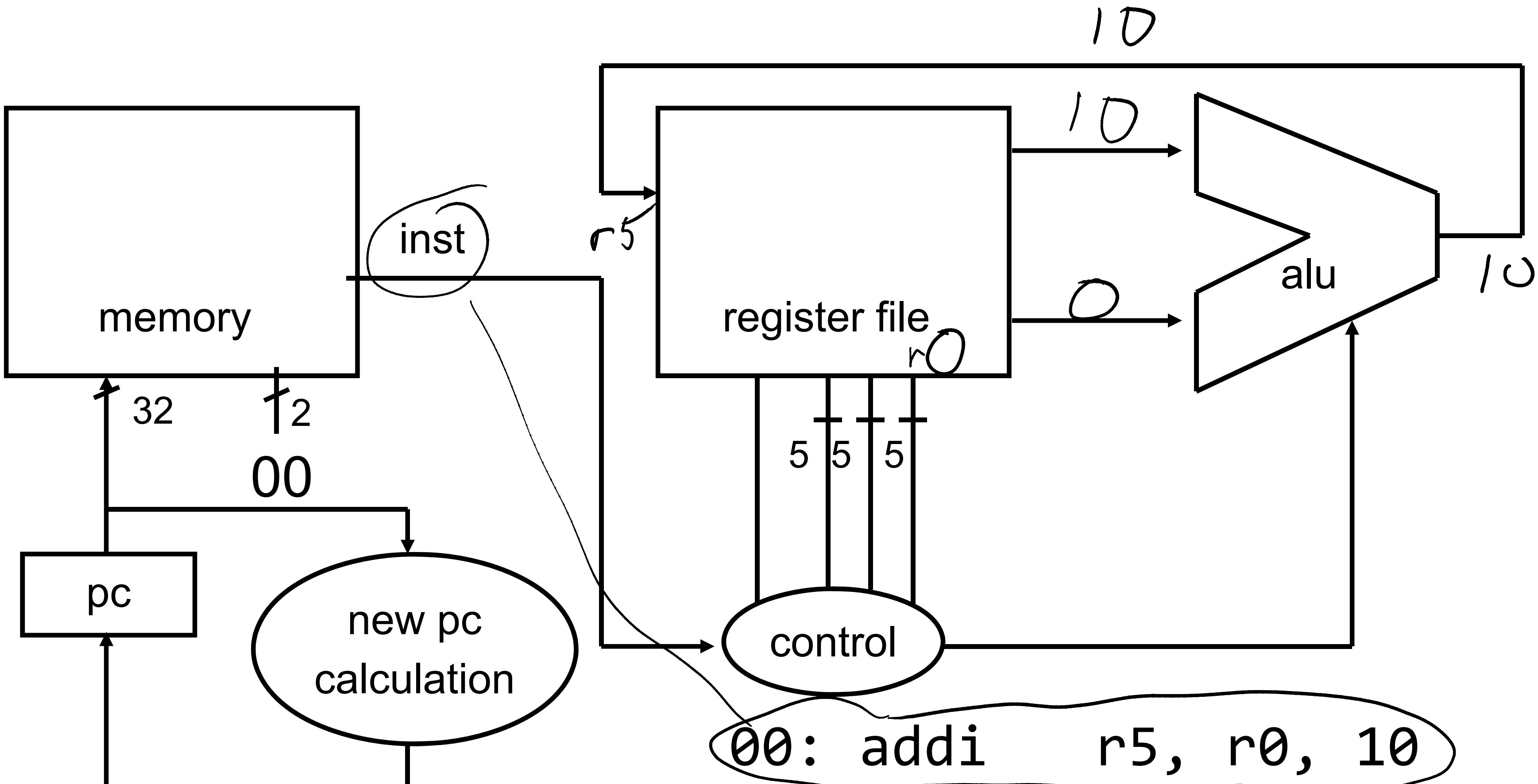
- ISA
  - abstract interface between hardware and the lowest level software
  - user portion of the instruction set plus the operating system interfaces used by application programmers

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

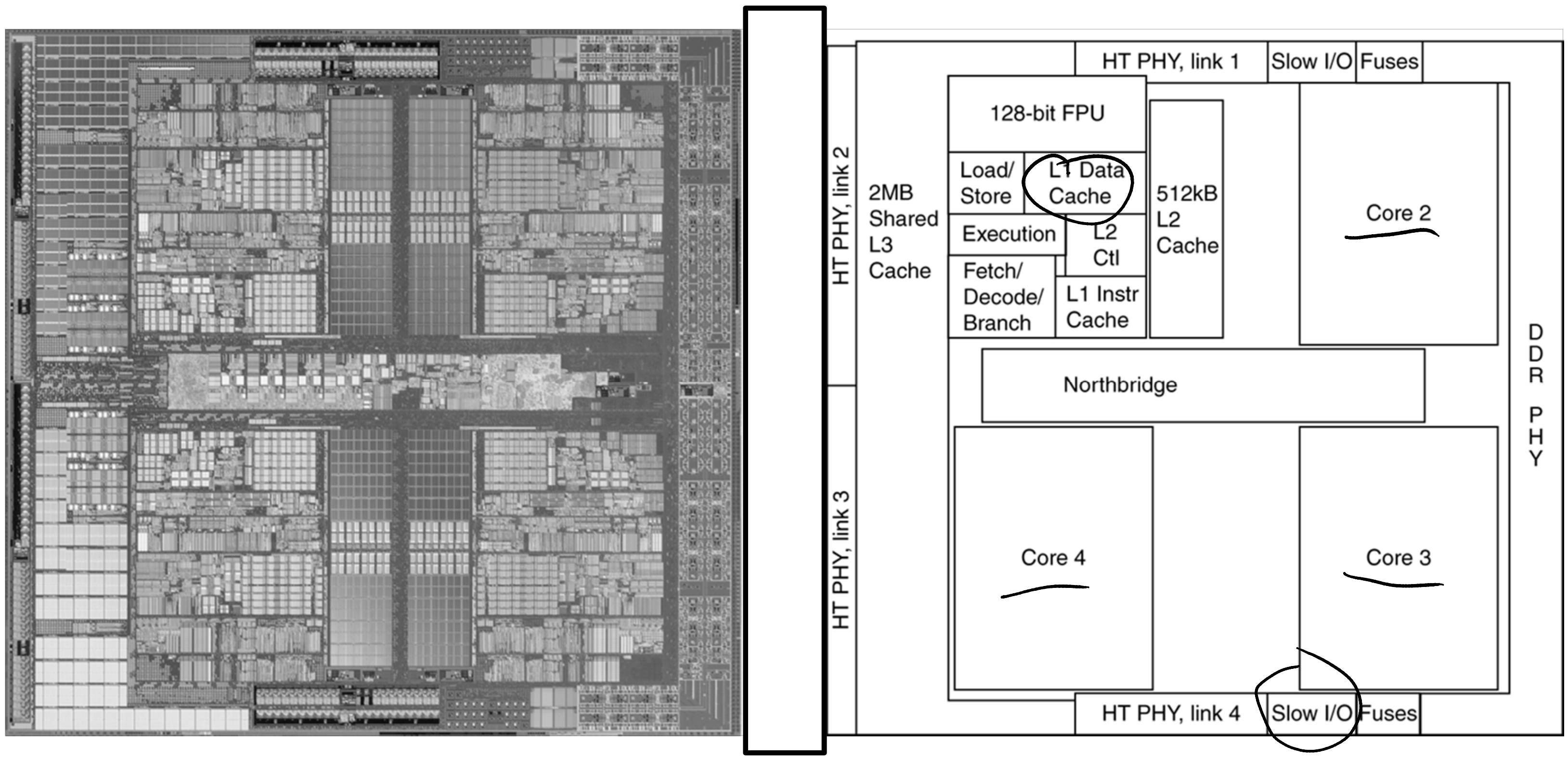


Figure from Patterson & Hennesssy, Computer Organization and Design, 4th Edition

How to Program the Processor: MIPS R3000 ISA

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

Registers

R0 - R31

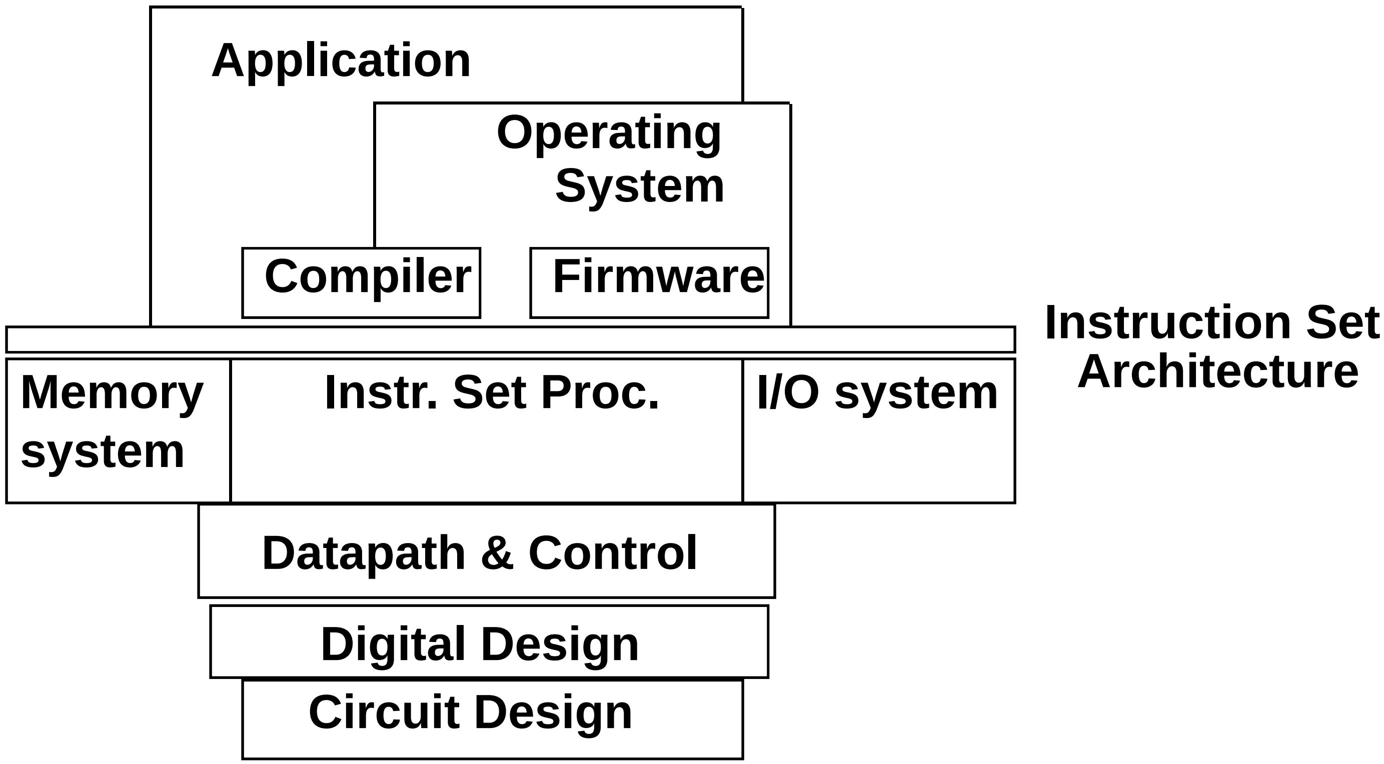
PC

HI

LO

R	**OP**	**rs**	**rt**	**rd**	**sa**	**funct**
I	**OP**	**rs**	**rt**	**immediate**		
J	**OP**	**jump target**				

Overview

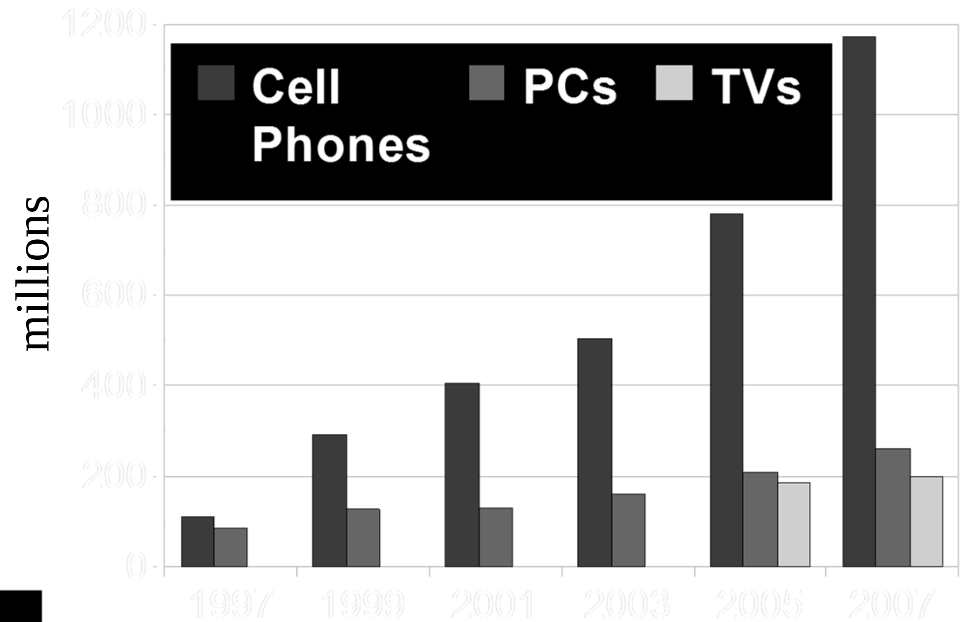

Application
Operating System
Compiler
Firmware
Instruction Set Architecture
Memory system
Instr. Set Proc.
I/O system
Datapath & Control
Digital Design
Circuit Design

Applications

- Everything these days!
  - Phones, cars, televisions, games, computers,…

Example 3: New Devices

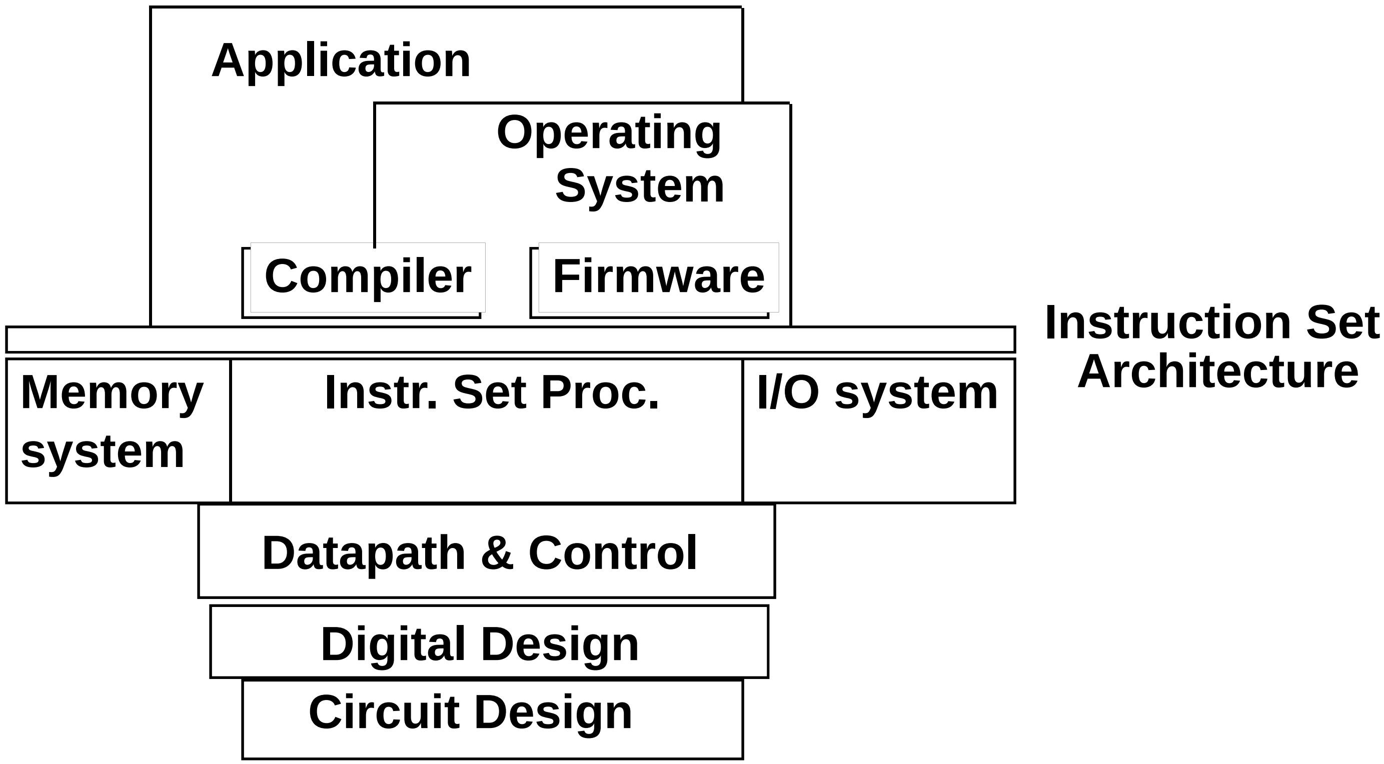
Xilinx FPGA



NVidia GPU

Berkeley mote

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