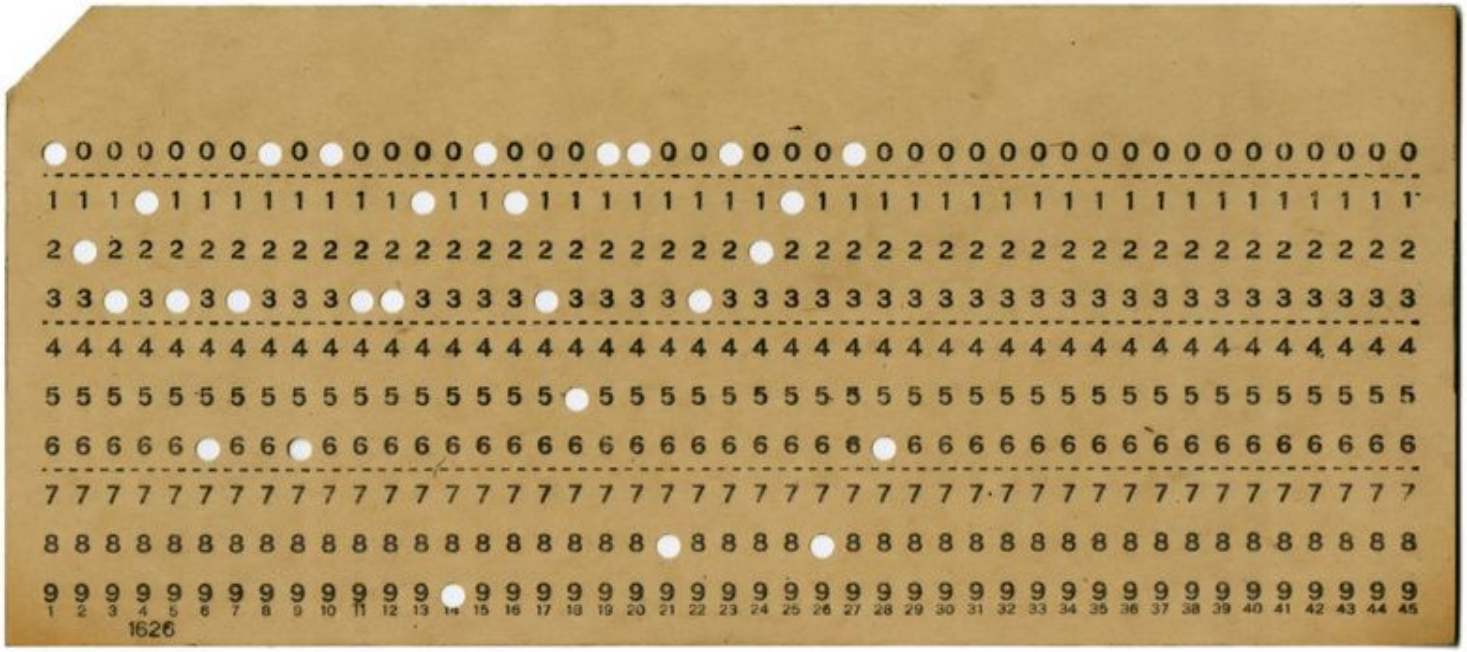
Great Ideas in Computer Architecture

Course Intro, Number Representation

Instructors: Steven Ho, Nick Riasanovsky



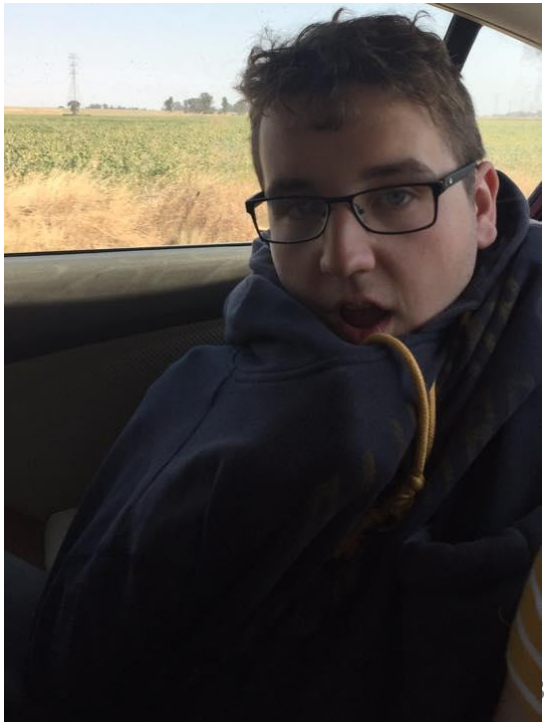
Introducing Steven

- Instructor
- **Upbringing:** Born and raised in LA
- **Education:** Newly-graduated CS & Ling double major

- **Teaching:** 61C course staff for 6 semesters, 188 TA last semester
- **Interests:**

Introducing Nick

- **Upbringing:** Born and raised in Orange County
- **Education:** Rising fourth year majoring in CS
- **Teaching:**
  - TA for 61C for 3 semesters
- **Interests:** Eating, Watching Sports, Video games



Introducing Your TAs

Sean

Damon: Head TA

Jonathan

Suvansh

Sruthi

Emaan

Sukrit

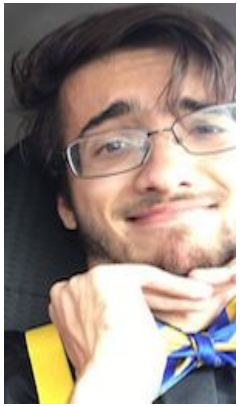
Introducing Your Tutors

Hersh

Alex

Anusha

Spencer



Stephan

Agenda

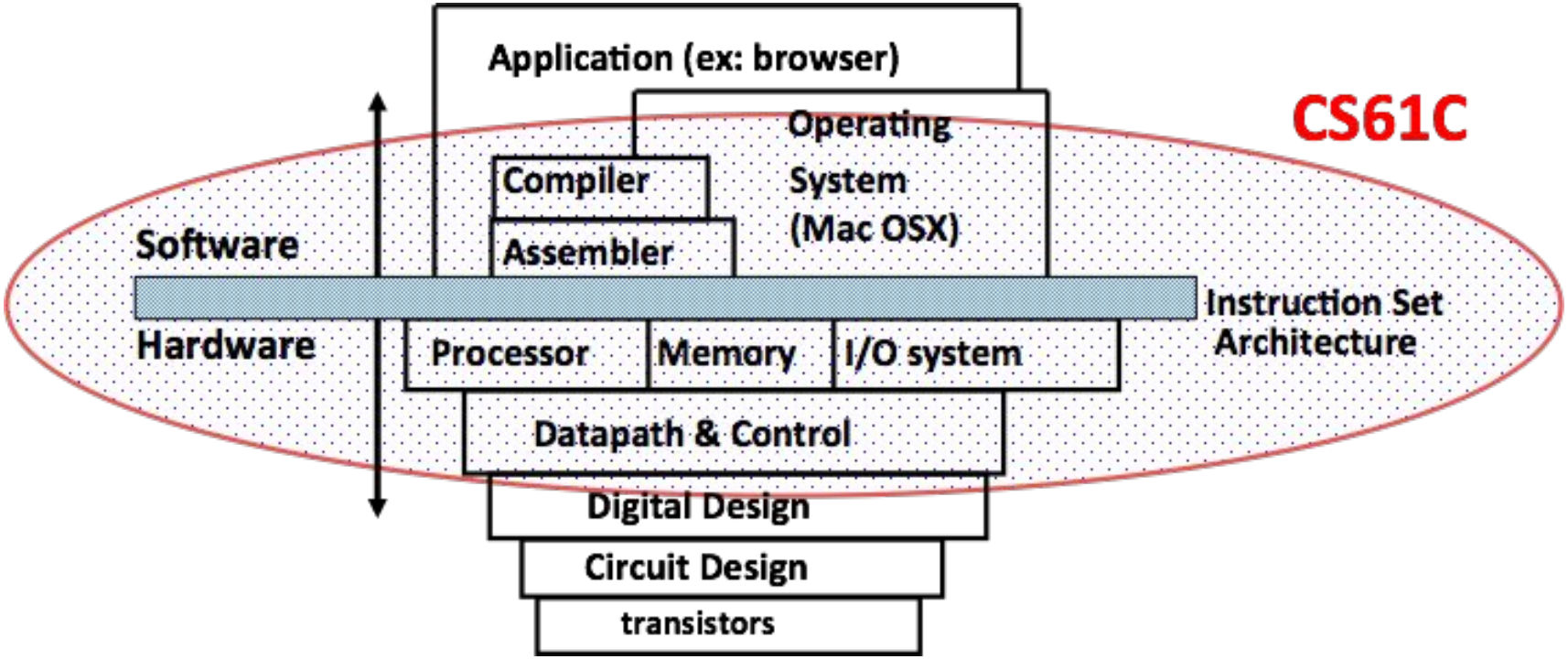
- Course Overview
- Course Policies
- Administrivia
- Number Representation
  - Number Bases
  - Signed Representations
  - Overflow
  - Sign Extension

61C Mission Statement

"CS61C introduces students to **foundational concepts** that form the backbone of different branches of computer science such as **computer architecture**, **low-level software**, **computer security**, and **large-scale computing**. This course demystifies some of the vulnerabilities inherent in **abstraction barriers** that programmers of all levels confide in when the **performance** and **correctness** of high-level code is under fire. After taking this course, students will be able to see computing through a **bit-level perspective** and understand the central **trade-offs** and choices involved in making fast hardware and software of all levels that work together efficiently and safely."

— Steven and Nick

Hardware-Software Interface



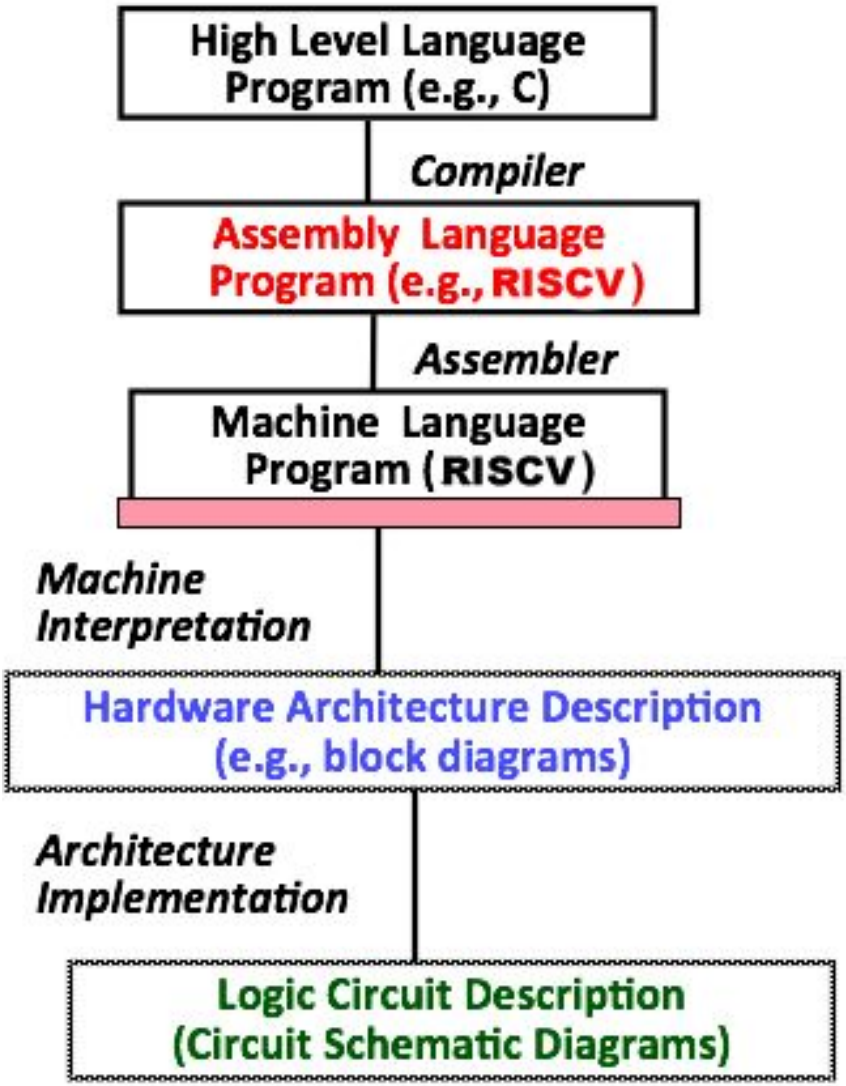
CS61C For Software Development

- Knowing the tools of the trade – computers!
  - "Computers" come in all shapes and sizes
  - Computing achieved in many different ways nowadays
- When performance matters
  - Ex: taking advantage of parallelism
- Differences between programming languages under the hood
- Design large systems – abstraction in hardware
- Security
- Design methodology – limitations and tradeoffs

Course Learning Objectives

- After taking this class students should be able to:
  - ✓ Identify and explain the various layers of abstraction that allow computer users to perform complex software tasks without understanding what the computer hardware is actually doing
  - ✓ Judge the effect of changing computer components (e.g. processor, RAM, HDD, cache) on the performance of a computer program
  - ✓ Explain how the memory hierarchy creates the illusion of being almost as fast as the fastest type of memory and almost as large as the biggest memory
  - ✓ Construct a working CPU from logic gates for a specified instruction set architecture
  - ✓ Identify the different types of parallelism and predict their effects on different types of applications

Great Idea #1: Abstraction (Levels of Representation/Interpretation)

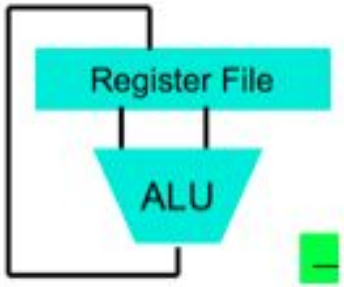



```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw   $t0, 0($2)
lw   $t1, 4($2)
sw   $t1, 0($2)
sw   $t0, 4($2)
```

Anything can be represented as a *number*, i.e., data or instructions

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101                1
0101 1000 0000 1001 1100                1
```



Course Learning Objectives

- In addition, this class will requires students to work on the following skills:
  - Creating and modifying designs to meet a given set of *specifications*
  - Identifying unexpected or problematic situations and create test cases to ensure proper behavior
  - Defending design choices based on tradeoffs and limitations

Six Great Ideas in Computer Architecture

1) Abstraction
2) Technology Trends
3) Principle of Locality/Memory Hierarchy
4) Parallelism
5) Performance Measurement & Improvement
6) Dependability via Redundancy

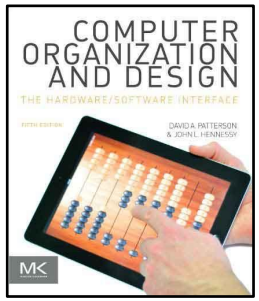
Agenda

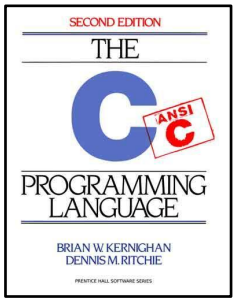
- Course Overview
- Course Policies
- Administrivia
- Number Representation
  - Number Bases
  - Overflow
  - Signed Representations
  - Sign Extension

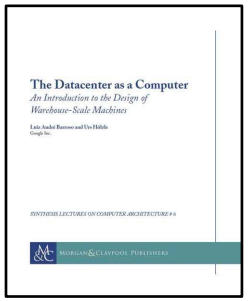
Course Information

More info found on the course policies page

- **Website:** http://inst.eecs.berkeley.edu/~cs61c/su18
  - Check for assignments, weekly schedule, contact info
- **Textbooks:**







- **Piazza:** http://piazza.com/class#summer2018/cs61c
  - Will have every announcement, discussion, clarification
- **Inst account:** http://inst.eecs.berkeley.edu/webacct
  - Submit assignments, check grades, access lab files

Course Assignments and Grading

- **Homework** (10%) – 8 total on edX
- **Labs** (10%) – 13 total
- **Projects** (30%) – 5 total, weighted by difficulty
  - Proj1 (7%) – Proj2-1 (3%) – Proj2-2 (4%)
  - Proj3 (9%) – Proj4 (7%)
- **Exams**
  - **Midterm 1** (13%): Monday, July 3 during class time
  - **Midterm 2** (13%): Monday, July 25 during class time
  - **Final** (23%): Thursday, August 9 @ 7pm
- **EPA** (5%) – Effort, Participation, and Altruism
  - Extra credit awarded to encourage learning environment

Lab Grading Policy

- Out of 2 points. A new lab is assigned every Tue/Thu
  - In order to receive credit for a lab, you must get it checked off by a staff member
  - You will receive full points if you get it checked off the day the lab was assigned, or one lab session late
  - You will receive half credit if you get it checked off two lab sessions late.
  - You will receive no points on a lab that is more than 2 labs late.
  - Note: You will only have one attempt to check off any lab that is late (aka not assigned that day)

Project Late Policy – Slip Days

- Project submissions due at 11:59:59pm
  - Count lateness in *days* (even if just by a second)
- Late penalty is 33% deduction of score per day
- You are given 3 slip day tokens
  - Will be distributed amongst your late submissions at the end of the semester to maximize your grade
  - No benefit to having leftover tokens
- Use at own risk – don't want to fall too far behind

EPA

- Encourage class-wide learning!
- Effort
  - Attendance and active engagement in additional course events (eg: office hours, guerilla sessions, project parties, etc)
- Participation
  - Attendance and active engagement in required course events (eg: discussion, lab…)
  - ...and lecture: iClickers!
- Altruism
  - Helping others anywhere (eg: lab, discussion, OH and Piazza)

EECS Grading Policy

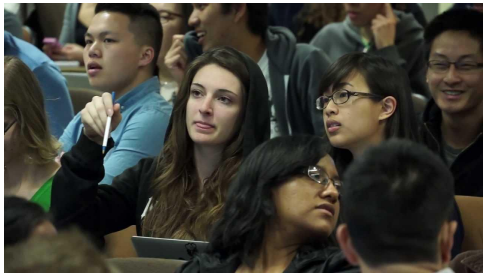
- http://www.eecs.berkeley.edu/Policies/ugrad.grading.shtml

  "A typical GPA for a lower division course will fall in the range 2.8 - 3.3, depending on the course and the students who enroll. For example, a GPA of 3.0 would result from 35% A's, 45% B's, 13% C's, and 7% D's and F's."
- Spring 2018: GPA 3.08
- 36% A's, 43% B's, 17% C's, 3% D's, 1% F's
- Graded using fixed bins

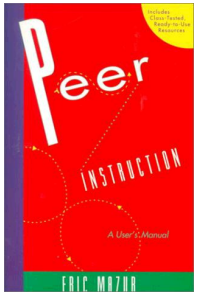
Raw Score	290+	[270,290)	[260,270)	[250,260)	[230,250)	[220,230)	[210,220)	[190,210)	[180,190)	[140,180)	[0,140)
Grade	A+	A	A-	B+	B	B-	C+	C	C-	D	F

- If the grade bins result in an average GPA that is too low, the course will be curved to match department guidelines BEFORE adding EPA.

Peer Instruction



- Increase real-time learning in lecture, test your understanding, increase student interactions
  - Lots of research supports its effectiveness
- Multiple choice questions sprinkled throughout lecture
  - 1 minute to decide on your own
  - 2 minutes in pairs to reach consensus
  - Learn through discussion
- Vote using i>clicker remotes
  - Can borrow for summer with $40 check as collateral



Question: Which statement is FALSE about the CS61C policies this summer?

(A) Both homework and labs are being graded on completion, not correctness

(B) Activities in person (lab, disc) and online (Piazza) can only help your class grade

(C) There are no dropped homeworks or labs

(D) Labs checked off two sections after they were assigned will be given half credit

Working with Others

- We are encouraging partner/group work
- Beneficial for the class
  - Someone to bounce ideas off of and work with and keep you on task
  - Don’t have to go it alone – meet new people!
- Valuable life skill: working well with others
  - Distribution of work, communication skills, time management, teaching skills

Policy on Assignments and Independent Work

- All submissions are expected to be yours and yours alone (or group's when applicable)
- You are encouraged to discuss your assignments with other students (*ideas*), but we expect that what you turn in is yours
- It is NOT acceptable to copy solutions from other students
- It is NOT acceptable to copy (or start your) solutions from the Web (including Github)

Policy on Assignments and Independent Work

- We have tools and methods, developed over many years, for detecting this. You WILL be caught, and the penalties WILL be severe.
- Both the *cheater* and the *enabler* receive –100% for the assignment. Letter to your university record documenting the incidence of cheating.
  - Possible automatic F in the course
- People are caught every semester of 61C

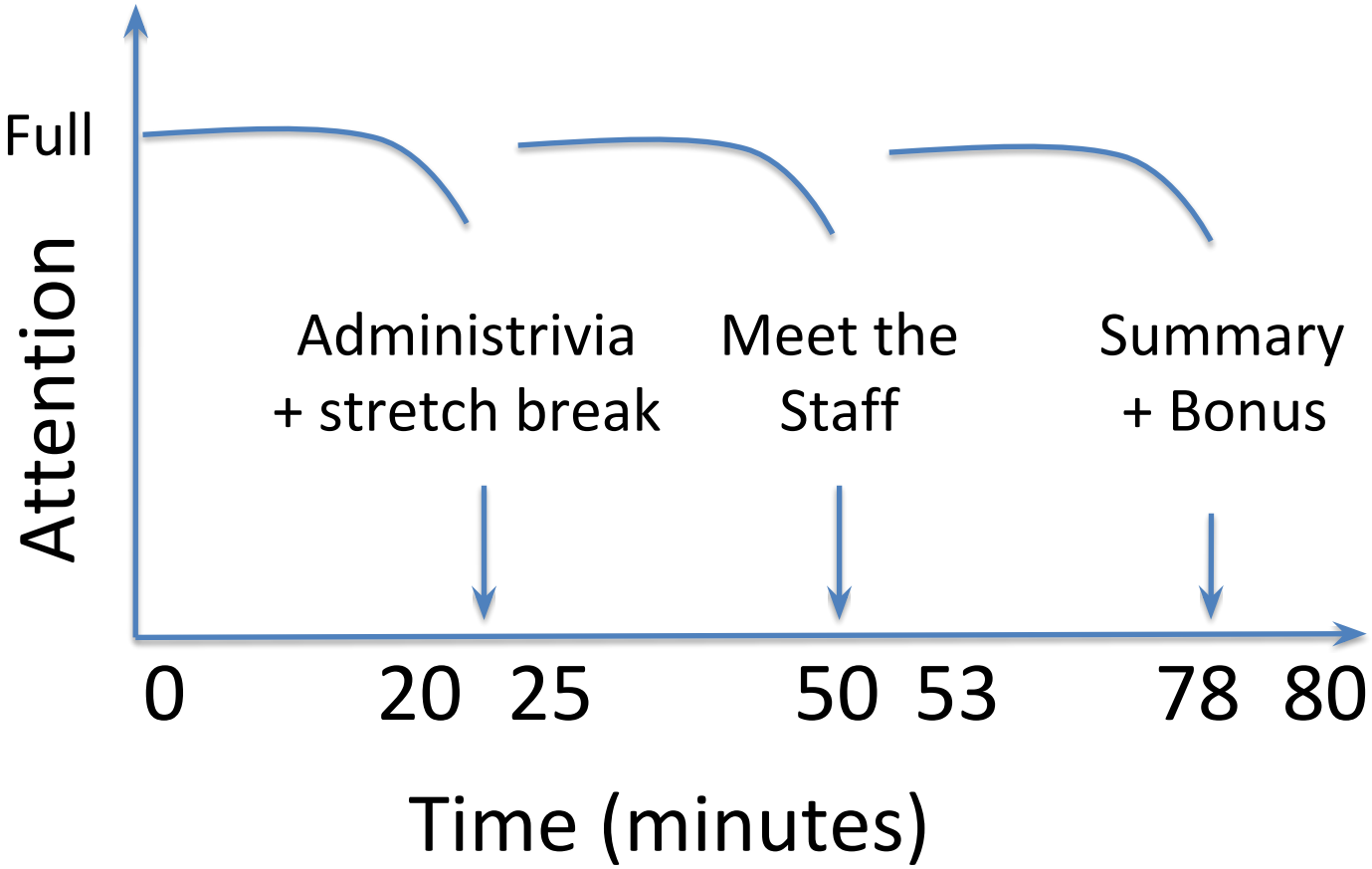
Course Resources

- Weekly Guerrilla Sessions
  - Led by tutors
  - Review sessions coming exam problems on specific topics
- Small-group tutoring
  - Sign up for a weekly tutoring session w/ a tutor
  - Groups are as small as 3-6 people; personalized help
- Project Parties
  - (Near) weekly events staffed by TAs, tutors, and possibly lab assistants to help you with your projects
- Office hours
  - Please come chat with us about any questions or concerns!

Successful Behaviors

- Practice, practice, practice
  - Learn by doing: deep learning doesn't happen in lec
  - Growth mindset: success through effort
- Find a learning community
  - Learning is much more fun with friends
  - Learn via teaching or different perspectives
- Avoid electronic screens during lecture
- You learn best from your mistakes
  - Don't be afraid to be wrong; you lose if you remain silent

Architecture of a Lecture



Administrivia

- Discussions, labs, and OHs start immediately
  - Yes, that means today!
  - Switching labs: if you find another 61C student willing to swap lab, talk to your TAs
  - Attend whichever discussion you want, as long as there is enough physical room
- HW0 and mini-bio due next Monday, June 25th
  - Find a digital image of yourself
  - Both worth points, neither can be dropped

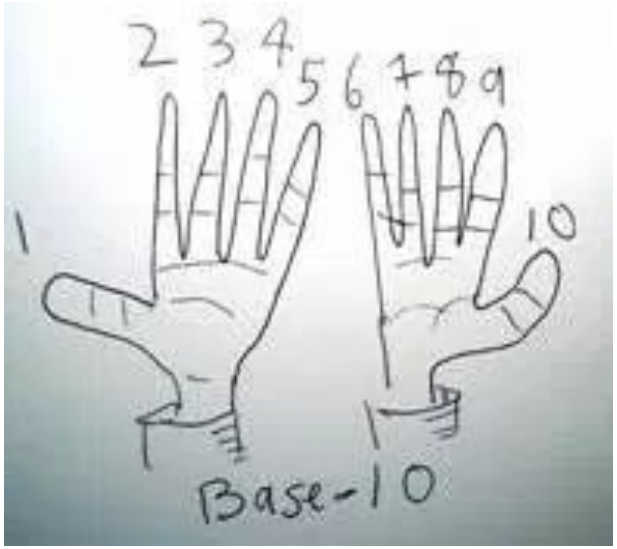
Meet the Staff

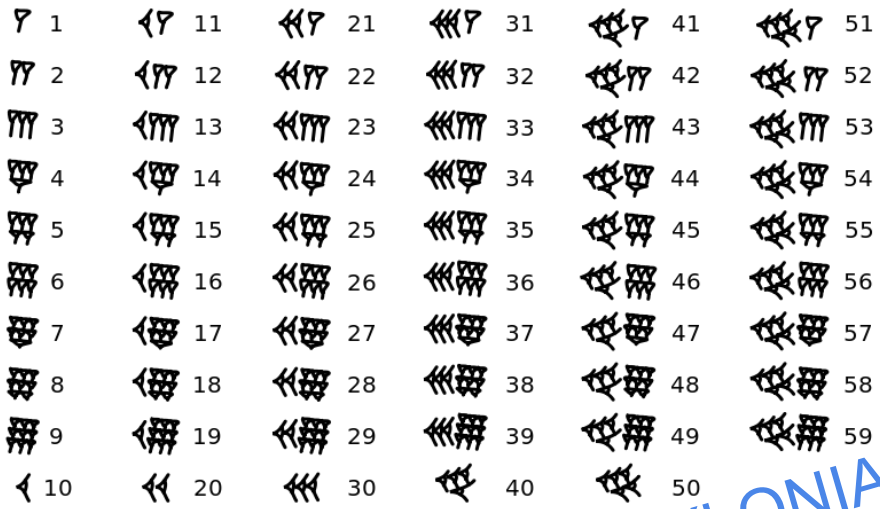
	 Steven	Nick 
Bribe Food	Pomegranates	Candy
Biggest Fear	Ramses' Curse	Heights
Unpopular Opinion	Pizza is terrible	Tabs > Spaces
Café Getaway	Asha Tea House	Nefali's

Agenda

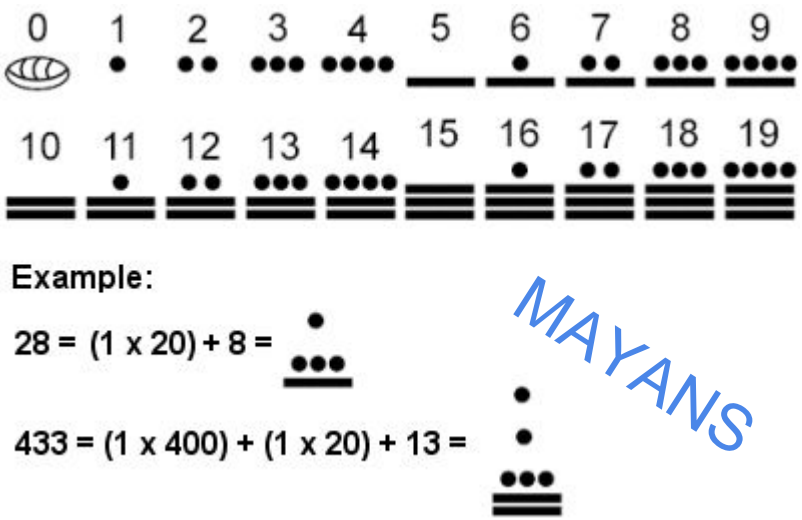
- Course Overview
- Course Policies
- Administrivia
- Number Representation
  - Number Bases
  - Signed Representations
  - Overflow
  - Sign Extension

Number Representation



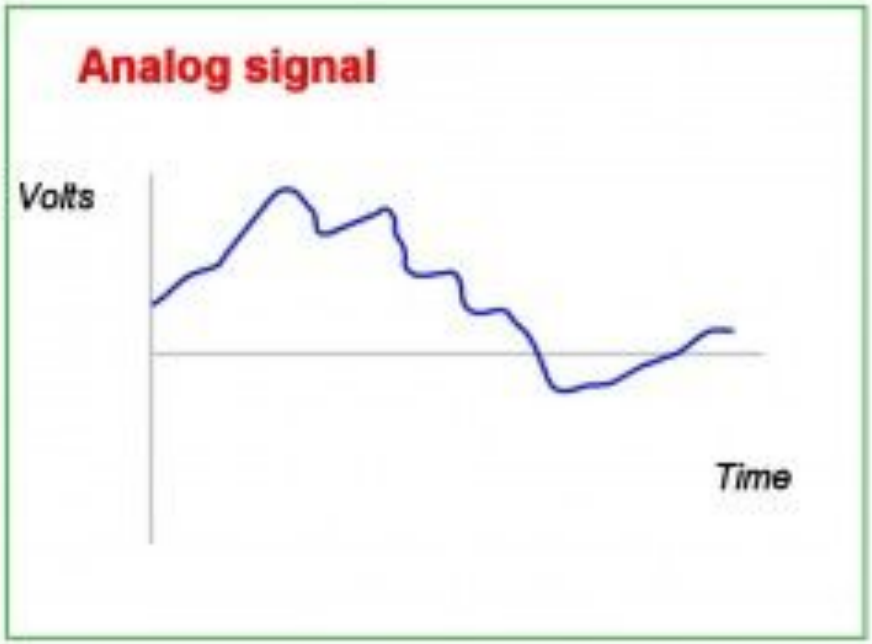


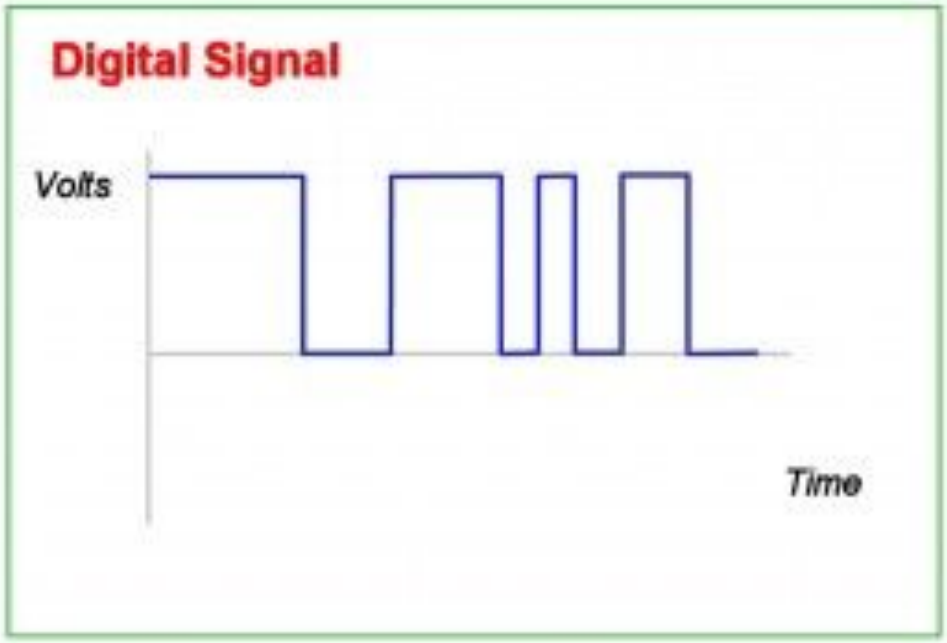
BABYLONIANS



MAYANS

Computers: Base 2





Number Base Examples

thousand's place → **9472**$_{ten}$ ← one's place

hundred's place → **9472**$_{ten}$ ← ten's place

$$\mathbf{9472}_{ten} = 9000 + 400 + 70 + 2$$
$$= 9 \times 1000 + 4 \times 100 + 7 \times 10 + 2 \times 1$$
$$= 9 \times 10^3 + 4 \times 10^2 + 7 \times 10^1 + 2 \times 10^0$$

General Formula:

$d_{n-1} d_{n-2} \ldots d_1 d_0$ (n-digit number in base B)

$$= d_{n-1} \times B^{n-1} + d_{n-2} \times B^{n-2} + \ldots + d_1 \times B^1 + d_0 \times B^0$$

a digit d can take on values from 0 to $B - 1$

Commonly Used Number Bases

- **Decimal** (base 10)
  - Symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - Notation: 9472_{ten} = 9472
- **Binary** (base 2)
  - Symbols: 0, 1
  - Notation: 101011_{two} = 0b101011
- **Hexadecimal** (base 16)
  - Symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
  - Notation: $2A5D_{hex}$ = 0x2A5D

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

i-not-Clicker Question

Convert **0b11110** to **decimal.**

(A) 11

(B) 30

1*2^4 + 1*2^3 + 1*2^2 + 1*2^1 + 0*2^0

(C) -2

(D) I'm so lost

(E) Who even cares?

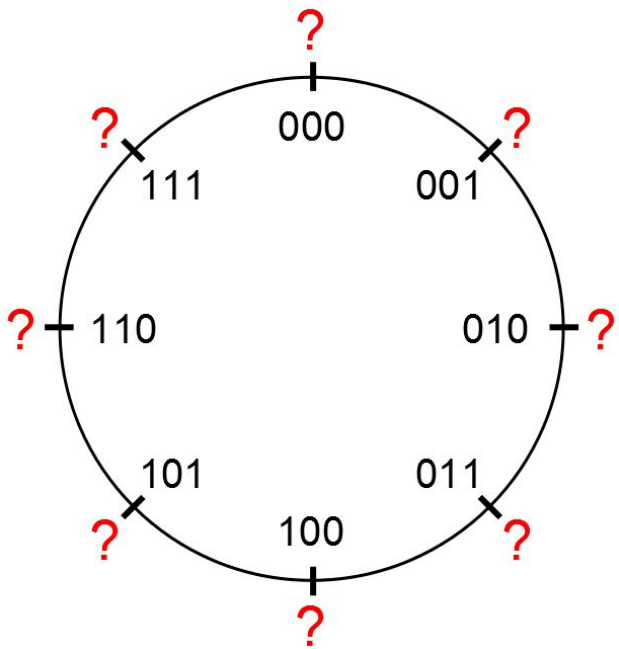

CAUTION:
BIG IDEA AHEAD
WHAT DOES IT ALL MEAN?!?!
quickmeme.com

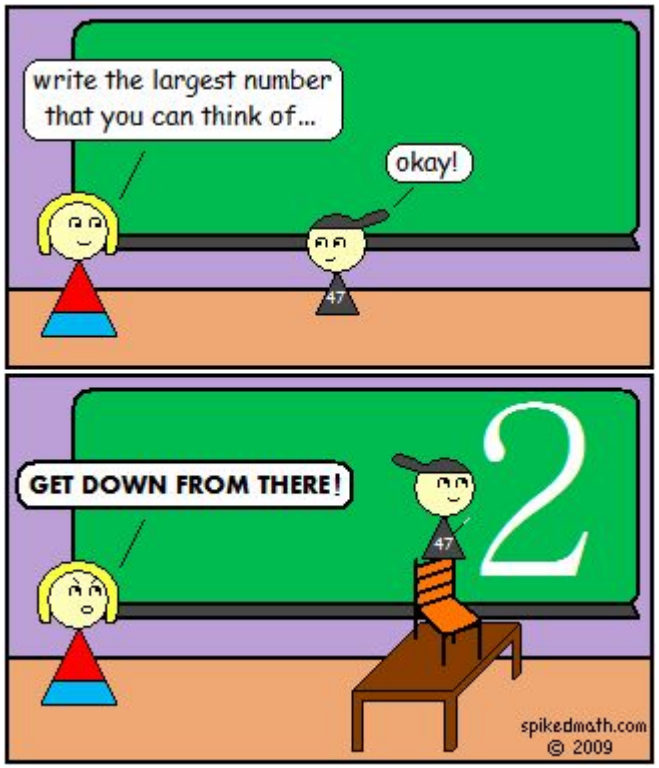
BIG Idea:
Bits Can Represent ANYTHING

- ...entirely depending on our *interpretation*
- **Characters?!**
  - 26 letters require 5 bits ($2^5 = 32 > 26$)
  - upper/lower case + punctuation → 7 bits ("ASCII")
- **Logical values?!** 0 → False, 1 → True
- **Students in this class?!** 2^8 = 256 > 220 → 8 bits
- **Colors?!** Red (00) Green (01) Blue (11)
- ONLY ONE constraint:

  n digits (base B) ⇒ <= B^n things
  - Each of the n digits is one of B possible symbols
  - Have more things? Add more digits!

How do we assign values to the Binary numerals available to us?





Unsigned Integers

Represent only non-negative (unsigned) integers:

$000_{two} = 0_{ten}$
$001_{two} = 1_{ten}$
$010_{two} = 2_{ten}$
$011_{two} = 3_{ten}$
$100_{two} = 4_{ten}$
$101_{two} = 5_{ten}$
$110_{two} = 6_{ten}$
$111_{two} = 7_{ten}$

Zero?

$0\ldots0_{two} = 0_{ten}$

Most neg number?

$0\ldots0_{two} = 0_{ten}$

Most pos number?

$1\ldots1_{two} = (2^n-1)_{ten}$

Increment?

Signed Integers

- n bits ⇒ 2^n things
  - half positive, half negative?
- First logical step:

  +534 -256
- Let the first bit be interpreted as a sign!

Sign and Magnitude

"first" bit gives sign, rest treated as unsigned (magnitude):

$000_{two} = +0_{ten}$
$001_{two} = +1_{ten}$
$010_{two} = +2_{ten}$
$011_{two} = +3_{ten}$

$100_{two} = -0_{ten}$
$101_{two} = -1_{ten}$
$110_{two} = -2_{ten}$
$111_{two} = -3_{ten}$

Zero?

$0\ldots0_{two}$ and $10\ldots0_{two} = \pm0_{ten}$

TWO ZEROS?!

Most pos number?

$01\ldots1_{two} = (2^{(n-1)}-1)_{ten}$

Most neg number?

$1\ldots1_{two} = -(2^{(n-1)}-1)_{ten}$

Increment?

Biased Notation

Like unsigned, but "shifted" so zero is (roughly) in the middle: (value = "unsigned value" - bias)

PRO

$000_{two} = -3_{ten}$

$001_{two} = -2_{ten}$

$010_{two} = -1_{ten}$

CON $011_{two} = 0_{ten}$ ← Conventional Bias: $(2^{n-1}-1)$

$100_{two} = +1_{ten}$

$101_{two} = +2_{ten}$

$110_{two} = +3_{ten}$

$111_{two} = +4_{ten}$ PRO

Zero?

$01\ldots1_{two} = 0_{ten}$

Most neg number?

$0\ldots0_{two} = -(2^{(n-1)}-1)_{ten}$

Most pos number?

$1\ldots1_{two} = 2^{(n-1)}{}_{ten}$

Increment?

PRO just like unsigned!

One's Complement

New negation procedure – complement the bits:

$000_{two} = +0_{ten}$
$001_{two} = +1_{ten}$
$010_{two} = +2_{ten}$
$011_{two} = +3_{ten}$

$100_{two} = -3_{ten}$
$101_{two} = -2_{ten}$
$110_{two} = -1_{ten}$
$111_{two} = -0_{ten}$

Zero?

$0\ldots0_{two}$ and $1\ldots1_{two}$ = $\pm0_{ten}$

TWO ZEROS AGAIN……

Most neg number?

$10\ldots0_{two} = -(2^{(n-1)}-1)_{ten}$

Most pos number?

$01\ldots1_{two} = (2^{(n-1)}-1)_{ten}$

Increment?

Two's Complement

Like One's Complement, but "shift" negative #s by 1:

$000_{two} = +0_{ten}$
$001_{two} = +1_{ten}$
$010_{two} = +2_{ten}$
$011_{two} = +3_{ten}$

$100_{two} = -4_{ten}$
$101_{two} = -3_{ten}$
$110_{two} = -2_{ten}$
$111_{two} = -1_{ten}$

Zero?

$0\ldots0_{two} = 0_{ten}$

Most neg number?

$10\ldots0_{two} = -2^{(n-1)}{}_{ten}$

Most pos number?

$01\ldots1_{two} = (2^{(n-1)}-1)_{ten}$

Increment?

just like unsigned!

Two's Complement Summary

- Used by all modern hardware
- Roughly evenly split between positive and negative
  - One more negative # because positive side has 0
- Can still use MSB as sign bit
- To negate: Flip the bits and add one
  - Example: +7 = 0b 0000 0111, -7 = 0b 1111 1001

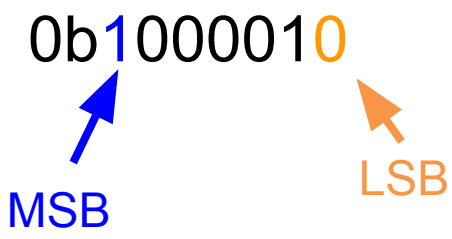
Two's Complement Review

- Suppose we had 5 bits. What range of integers can be represented in two's complement?

(A)	-31 to +31	← need 6 bits
(B)	-15 to +15	← one's complement
(C)	0 to +31	← unsigned
(D)	-16 to +15	← two's complement
(E)	-32 to +31	← need 6 bits

Numbers in a Computer

- Numbers really have ∞ digits, but hardware can only store a finite number of them (fixed)
  - Usually ignore *leading zeros*
  - Leftmost is *most significant bit* (MSB)
  - Rightmost is *least significant bit* (LSB)



Overflow

- Overflow is when the result of an arithmetic operation can't be represented by the (FINITE) hardware bits
  - i.e. the result is mathematically incorrect
- Examples:
  - Unsigned: $0b1\ldots1 + 1_{10} = 0b0\ldots0 = 0?$
  - Two's: $0b01\ldots1 + 1_{10} = 0b10\ldots0 = -2^{(n-1)}{}_{ten}?$

- Solution: add more bits

Sign Extension

- Want to represent the same number using more bits than before
  - Easy for positive #s (add leading 0's), more complicated for negative #s
  - Sign and magnitude: add 0's *after* the sign bit
  - One's complement: copy MSB
  - Two's complement: copy MSB
- Example:
  - Sign and magnitude: 0b 11 = 0b 1001
  - One's/Two's complement: 0b 11 = 0b 1111

Summary

- Number Representation: How to represent positive and negative integers using binary
  - Unsigned: Interpret numeral in base 2
  - Signed: Two's Complement
  - Biased: Subtract bias
  - Sign extension must preserve signed number