# Bits, Bytes, and Integers – Part 1

15-213/18-213/15-513: Introduction to Computer Systems
2nd Lecture, May 22, 2019

**Instructors:**

Sol Boucher

# *But first…* an overview of course topic progression and labs

# Programs and Data

## Topics

- Bit operations, arithmetic, assembly language programs
- Representation of C control and data structures
- Includes aspects of architecture and compilers

## Assignments

- L1 (devicelab): Manipulating bits, characters, and strings
- L2 (bomblab): Defusing a binary bomb
- L3 (attacklab): The basics of code injection attacks

## Note: new first lab vs. past terms, current 513!

# The Memory Hierarchy

## Topics

- Memory technology, memory hierarchy, caches, locality
- Includes aspects of architecture and OS

## Assignments

- L4 (cachelab): Building a cache simulator and optimizing for locality.
  - Learn how to exploit locality in your programs.

# Memory Allocation

## Topics

- Dynamic storage allocation, virtual memory, address translation
- Includes aspects of architecture and OS

## Assignments

- L5 (malloclab): Writing your own malloc package
  - Get a real feel for systems-level programming

## Note: different topic/lab order vs. past terms!

# Exceptional Control Flow

## Topics

- Hardware exceptions, processes, process control, Unix signals, nonlocal jumps
- Includes aspects of compilers, OS, and architecture

## Assignments

- L6 (tshlab): Writing your own Unix shell.
  - A first introduction to concurrency

# Networking, and Concurrency

## Topics

- High level and low-level I/O, network programming
- Internet services, Web servers
- concurrency, concurrent server design, threads
- I/O multiplexing with select
- Includes aspects of networking, OS, and architecture

## Assignments

- L7 (proxylab): Writing your own Web proxy
  - Learn network programming and more about concurrency and synchronization.

# Lab Rationale

- **Each lab has a well-defined goal such as solving a puzzle or winning a contest**
- **Doing the lab should result in new skills and concepts**
- **We try to use competition in a fun and healthy way**
  - Set a reasonable threshold for full credit
  - Post intermediate results (anonymized) on Autolab scoreboard for glory!

# Policies: Grading

- **Exams (50%): midterm (20%), final (30%)**
- **Labs (50%): weighted according to effort**
- **Final grades based on a straight scale (90/80/70/60) with a small amount of curving**
  - Only upward

# Doing the Lab

- **https://autolab.andrew.cmu.edu/courses/15213-m18**
  - Hosts a writeup with instructions for downloading and completing the lab
  - Access will be granted on Friday when the first lab is released
- **If you have questions**
  - Piazza
  - Office hours…

# Office Hours

- **Prof. Railing**
  - Immediately after lectures, i.e. class meetings *except*:
    - Bootcamps
    - Recitations
    - Exam reviews
- **Prof. Boucher**
  - Wednesdays at 3 PM in GHC–***8***115
  - Fridays at 3 PM in GHC–***9***115
- **TAs**
  - Primary time TBA
- **Availability may vary week to week**
  - Check Office Hours page on website

# Lecture Schedule Preview

- This week: *Everything* you need for devicelab!
  - Yesterday: Course Overview (Prof. Railing)
  - **Today: Bits, Bytes, Integers (Prof. Boucher)**
  - Tomorrow: *More* Bits, Bytes, Integers (Prof. Boucher)
  - Friday: Floating Point (Prof. Railing)
- Next week
  - Tuesday: Linux/Git bootcamp (Tas)
  - …

# Waitlist questions

- 15-213: Amy Weis alweis@andrew.cmu.edu
- 18-213: Zara Collier zcollier@andrew.cmu.edu
- 15-513: Amy Weis alweis@andrew.cmu.edu

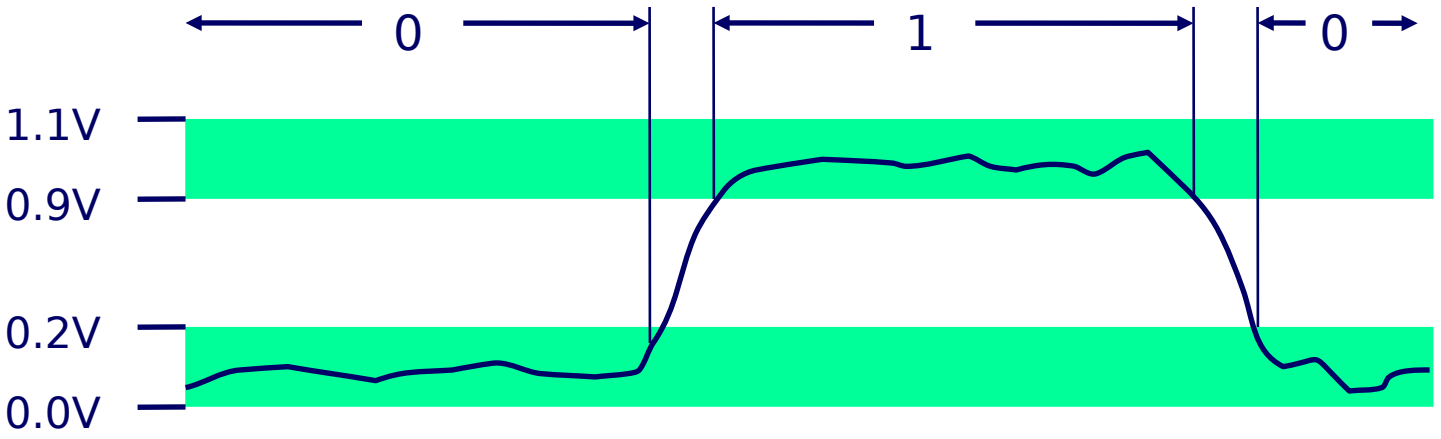
- Please don't contact the instructors with waitlist questions.

# Today/tomorrow: Bits, Bytes, Integers

- **Representing information as bits**
- **Integers**
  - Representation: unsigned and signed
  - Bit-level manipulations
  - Conversion, casting
  - Expanding, truncating
  - Addition, negation, multiplication, shifting
  - Summary
- **Representations in memory, pointers, strings**

# Everything is bits

- Each bit is 0 or 1
- By encoding/interpreting sets of bits in various ways
  - Computers determine what to do (instructions)
  - … and represent and manipulate numbers, sets, strings, etc…
- Why bits? Electronic Implementation
  - Easy to store with bistable elements
  - Reliably transmitted on noisy and inaccurate wires



# For example, can count in binary

- **Base 2 Number Representation**
  - Represent $15213_{10}$ as $11101101101101_2$
  - Represent $1.20_{10}$ as $1.0011001100110011[0011]\ldots_2$
  - Represent $1.5213 \times 10^4$ as $1.1101101101101_2 \times 2^{13}$

# Encoding Byte Values

- **Byte = 8 bits**
  - Binary $00000000_2$ to $11111111_2$
  - Decimal: $0_{10}$ to $255_{10}$
  - Hexadecimal $00_{16}$ to $FF_{16}$
    - Base 16 number representation
    - Use characters '0' to '9' and 'A' to 'F'
    - Write $FA1D37B_{16}$ in C as
      - 0xFA1D37B
      - 0xfa1d37b

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

**Activity: binary, hexadecimal, two's complement**

```
15213: 0011 1011 0110 1101
         3    B    6    D
```

# Example Data Representations

| C Data Type | Typical 32-bit | Typical 64-bit | x86-64 |
|---|---|---|---|
| `char` | 1 | 1 | 1 |
| `short` | 2 | 2 | 2 |
| `int` | 4 | 4 | 4 |
| `long` | 4 | 8 | 8 |
| `float` | 4 | 4 | 4 |
| `double` | 8 | 8 | 8 |
| pointer | 4 | 8 | 8 |

**Preview: negative numbers and fixed widths**