Lecture 1: Course overview

David Hovemeyer

August 29, 2022

601.229 Computer Systems Fundamentals

Welcome!

- Welcome to CSF!
- Today:
  - Administrative stuff
  - Course overview
  - Binary data representation

Administrative stuff

About the course

- Instructor
  - David Hovemeyer, daveho@cs.jhu.edu, Malone 240A
- CAs
  - Coming soon, see course web page for details

Where to find stuff

- Course website: `https://jhucsf.github.io/fall2022`
  - Syllabus, schedule, lecture notes, assignments, etc.
  - All public course information will be here
- Piazza `https://piazza.com/jhu/fall2022/601229`
  - Non-public course information such as homework/exam solutions
  - Discussion forum, Q/A: please post questions here!

Syllabus highlights

- Please read the syllabus carefully:
  `https://jhucsf.github.io/fall2022/syllabus.html`
- Highlights:
  - Grades: 55% homework, 40% exams, 5% participation
  - *Probably* 6 assignments, mostly programming based, expect them to be challenging!
  - Late policy: you have 120 late hours to use as needed (assignment submissions which exceed the late hour limit receive no credit)
  - Three exams (two during semester, one during final exam period)
    - Exams will be in-class
    - Will focus on recently-covered material

Participation

- What counts as participation?
  - What officially counts:
    - Participation in clicker quizzes in class
  - Also valuable, but won't officially count:
    - Activity on Piazza (asking questions, answering questions)
    - Attending office hours (but we won't track this closely)
    - Reviewing lecture recordings
- I would like to see *reasonably consistent* participation

Academic integrity

- Please read the academic integrity policy in the syllabus carefully
- Highlights:
  - Follow the CS Academic Integrity Code: `https://www.cs.jhu.edu/academic-integrity-code/`
  - Homework assignments
    - Individual: code sharing is not allowed
    - Pair: you can work with one partner
  - Exams are (obviously) individual effort
  - Violations of academic integrity will be reported to the Student Conduct office
- Be careful about using web as a resource
  - Do *not* copy code
  - *Always* cite sources used

Class meetings

- Typical class meeting: lecture/discussion, peer instruction questions, occasional group activities, discussion of current assignment, time for free-form Q&A
- *Do the reading in advance!*
- Come prepared to actively engage with the material!
  - Learning is not passive
  - More productive class time $\rightarrow$ better outcomes
  - Ask questions!

Peer instruction

- How peer instruction works:
  - Slide with a multiple choice question
  - Answer individually, discuss with peers, then answer again
  - Shown to improve outcomes!
  - Questions may be challenging
  - Graded for participation only
- You may have done this in other courses

Getting an iClicker remote

- You will need an iClicker remote
  - iClicker 2, iClicker+, and the original iClicker all should work
  - Could potentially get an iClicker 2 at Barnes and Noble or the JHU Technology Store
  - You could get a used one
    - from another student who no longer needs it
    - on EBay (they should be $10 to $20)
  - Use the google form linked from the Piazza resources page to register your iClicker remote ID
- Using the iClicker phone app will *not* be an option

Peer instruction etiquette

- Be respectful:
  - Let everyone participate
  - Don't put down anyone else's ideas
- Work together and think carefully about the question!

First clicker quiz!

Clicker quiz omitted from public slides

Computing requirements

- All assignments will be done using x86-64 Linux
- Autograders will use Ubuntu 18.04
- **You will need an x86-64 Linux development environment!**
- Recommendations:
  - Ugrad machines (different version of Linux, but should work fine)
  - Run Linux on your laptop or PC
  - Run Ubuntu 18.04 using WSL2 under Windows (great option!)
  - Run an Ubuntu virtual machine image using VirtualBox
- I'm not aware of any way to set up a usable development environment on an M1 Mac

Course overview

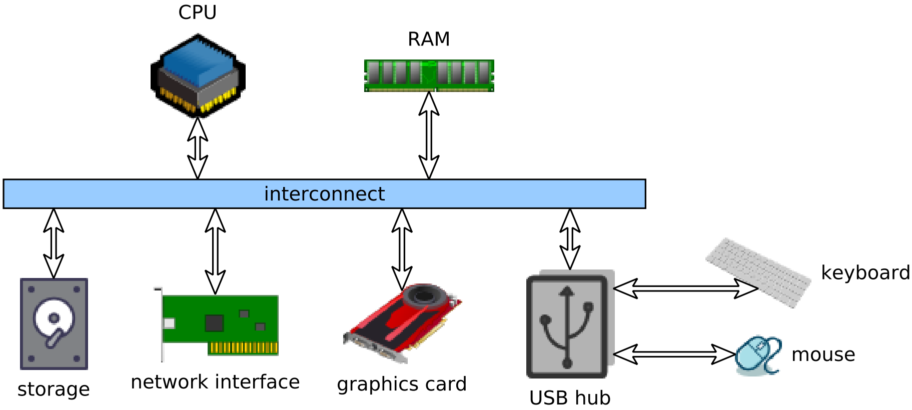
What the course is about

- Course is about *computer systems* from the *programmer's perspective*
- Computer system = hardware + software
  - Much of our concern is the interaction between hardware and software — how they work together

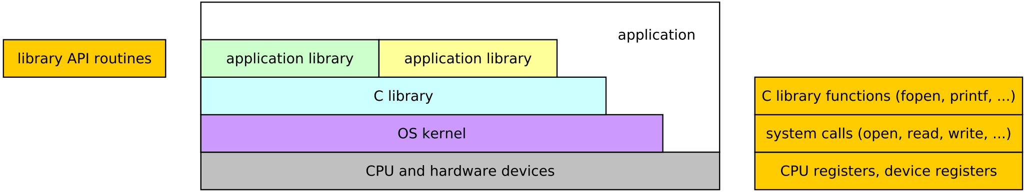
Goals of course

- "Deep" understanding of how computers work (down to hardware)
  - OS and runtime library interfaces
  - Machine-level ISA / assembly language
  - Processor features
  - Operating system features
- Apply this understanding to...
  - Optimize application performance
  - Avoid pitfalls such as security vulnerabilities
  - Take full advantage of the computer's and operating system's capabilities

A computer system (hardware)

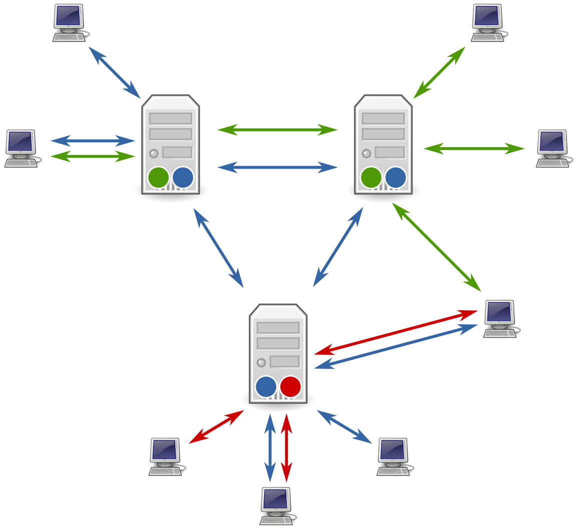


A computer system (software)



- Your application program is supported by lower layers of software and hardware
- Each layer provides an interface to the layer above

A computer network



Computer networks allow your program to communicate with peer systems.

Thanks to the global Internet, the peer systems could be anywhere on earth!

Binary data representation

Discrete data representation

- Digital computers use a *discrete* representation for all data
- Consider a representation of a number:

Discrete data representation

- Digital computers use a *discrete* representation for all data
- Consider a representation of a number:
  - A *continuous* representation would allow the number to have *any* value
    - We think of physical phenomena (mass, velocity, etc.) as being continuous

Discrete data representation

- Digital computers use a *discrete* representation for all data
- Consider a representation of a number:
  - A *continuous* representation would allow the number to have *any* value
    - We think of physical phenomena (mass, velocity, etc.) as being continuous
  - A *discrete* representation would allow the number to have one of a set of possible values, where the set of possible values is *enumerable*
    - Often we think of discrete values as corresponding to a range of integers

Why discrete representation?

- Why do digital computers use a discrete representation for all data?

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*
  - High voltage (1) vs. low voltage (0)
  - Digital circuits (with discrete high vs. low voltages) have many advantages over *analog* circuits, where voltages can vary continuously

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*
  - High voltage (1) vs. low voltage (0)
  - Digital circuits (with discrete high vs. low voltages) have many advantages over *analog* circuits, where voltages can vary continuously
- OK, let's think about what discrete data representations will look like...

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*
  - High voltage (1) vs. low voltage (0)
  - Digital circuits (with discrete high vs. low voltages) have many advantages over *analog* circuits, where voltages can vary continuously
- OK, let's think about what discrete data representations will look like...
  - Starting with *integers* (if you can represent integers, you can represent anything)

Decimal numbers

- We're all familiar with decimal (base 10) numbers
- E.g.,

$$42 = 4 \cdot 10^1 + 2 \cdot 10^0$$

  - Digits are 0–9
  - Places are powers of 10

Other bases

- Base 10 is arbitrary!
- Representing decimal 42 using base 5:
$$42_{10} = 132_5 = 1 \cdot 5^2 + 3 \cdot 5^1 + 2 \cdot 5^0$$
  - "Digits" are 0–4
  - Places are powers of 5

Try it!

How to express decimal 42 using base 6?

$$\underline{\hspace{2em}} \cdot 6^2 + \underline{\hspace{2em}} \cdot 6^1 + \underline{\hspace{2em}} \cdot 6^0$$

How to express decimal 79 using base 6?

$$\underline{\hspace{2em}} \cdot 6^2 + \underline{\hspace{2em}} \cdot 6^1 + \underline{\hspace{2em}} \cdot 6^0$$

Reference:

$$6^2 = 36$$
$$6^1 = 6$$
$$6^0 = 1$$

Binary

- Binary = base 2
- Representing decimal 42 using base 5:

$$\begin{aligned} 42_{10} &= 101010_2 \\ &= 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \end{aligned}$$

  - "Digits" are 0 and 1
  - Places are powers of 2
- Computers use binary representations for all data, because
  - *Digital circuits* use two voltage levels, high and low
  - By convention, 1=high voltage, 0=low voltage
  - So, computer hardware fundamentally operates on binary data

Try it!

How to express decimal 29 using base 2?

$$\_\_\_\_ \cdot 2^5 + \_\_\_\_ \cdot 2^4 + \_\_\_\_ \cdot 2^3 \; \_\_\_\_ \cdot 2^2 + \_\_\_\_ \cdot 2^1 + \_\_\_\_ \cdot 2^0$$

Reference:

$$2^5 = 32$$
$$2^4 = 16$$
$$2^3 = 8$$
$$2^2 = 4$$
$$2^1 = 2$$
$$2^0 = 1$$