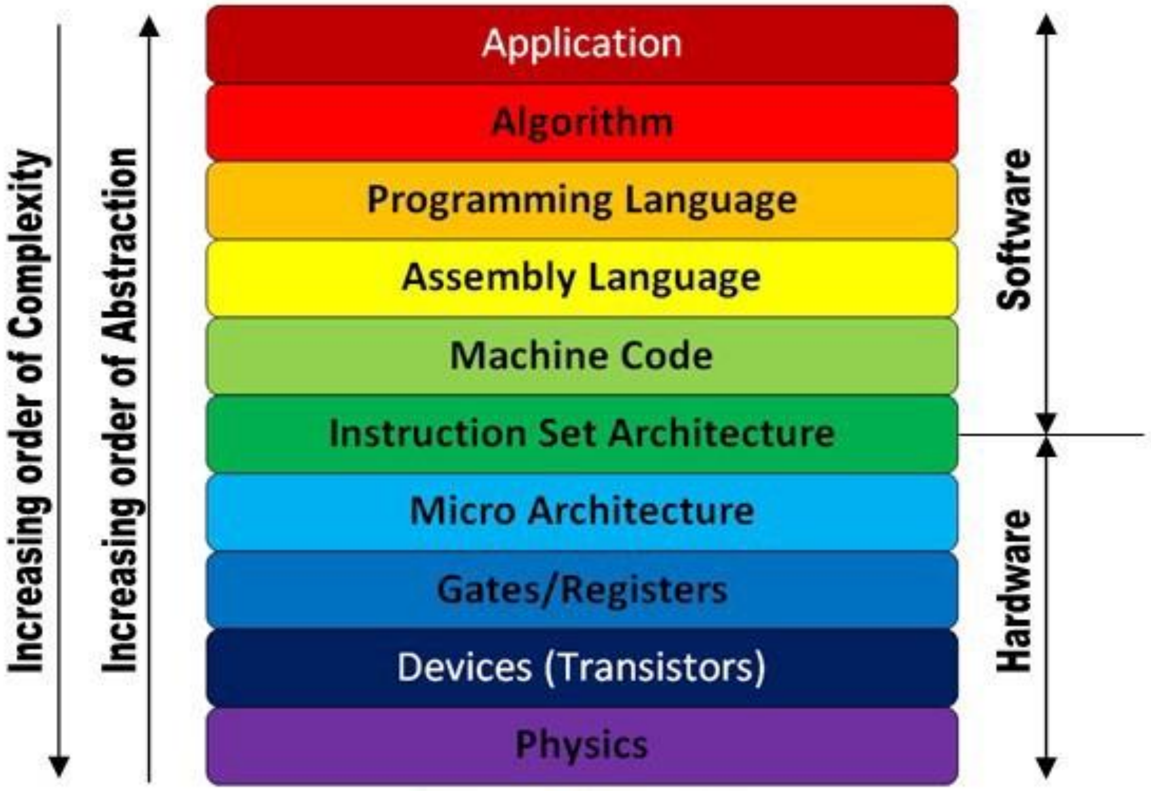
1. Introduction
    a. About me
        i. Second time teaching this specific course
        ii. TAed ECS 154B, next course in the sequence, for almost three years
    b. About this class (why computer architecture?)
        i. This is a required class for both CS and CSE
        ii. “Core material” for most CS graduate programs
        iii. My goal is to get you to remember some of this knowledge to use in industry or your next computer architecture class
        iv. My *hope* is that you understand why computer architecture is important after this class
    c. About this particular class
        i. Go over syllabus

2. Motivation
    a. Why do we care about any of this?
        i. Come back to this in a bit
    b. First, let’s talk about abstraction layers
        i. What are the different layers of computing, from the software you run to the hardware you run them on?
        ii. One interpretation below (unsure of original source for this image)



Layers of Abstraction

- c. Computing has become exponentially more complex since its inception
    - i. Few people in the world can say they know in detail every layer of the above
    - ii. Most people in academia/industry tend to focus on a few of these areas
    - iii. Very difficult to understand every single layer in detail
- d. Where does computer architecture lie?
    - i. Defined as the “hardware-software interface” by John Hennessey and Davis Patterson
        - 1. Two famous computer architects from Stanford and UCB, respectively
    - ii. Some debate, but usually consists of the middle three layers
        - 1. Machine code, instruction set architecture, and microarchitecture
    - iii. Back to our original question. Why do we care?
- e. Three reasons to care, taken from Hennessey and Patterson’s 2017 Turing Award Lecture
    - i. Can’t rely on hardware to pick up your slack and make your applications run faster anymore
        - 1. Death of Moore’s law and Dennard scaling
        - 2. Need to understand underlying hardware to keep making programs run faster
        - 3. Example: GPUs have become commonplace for running ML/AI workloads
    - ii. Security becoming a bigger and bigger concern
        - 1. Remember hearing the terms Spectre and Meltdown?
        - 2. Security vulnerability in some CPUs that let rogue processes read all memory
        - 3. Want secure applications as well as secure hardware that runs your applications
    - iii. “Golden Age” of computer architecture is right now according to them
        - 1. Many opportunities to innovate and contribute
        - 2. Great time for architects in academia and industry
- f. This is a very broad topic
    - i. You’re not going to know everything after this class, but will get a general overview
    - ii. ECS 154B covers more
- g. What we’re going to talk about
    - i. Digital design
        - 1. Basic building blocks
        - 2. Gates/registers layer in previous image
    - ii. How to build a computer from those building blocks
        - 1. Instruction set architecture and microarchitecture layers
    - iii. Other pieces of a computer
        - 1. Buses
        - 2. Memory, including caches and virtual memory

3. Basic definitions
   a. Computer architecture
      i. “Hardware-software interface”
      ii. Attributes of a system visible to the programmer
      iii. Those attributes which impact the logical execution of a program
   b. Computer organization/microarchitecture
      i. Units of a machine, and operations it performs
      ii. Interconnections between the two that realize the architectural specifications
   c. Tasks of a computer
      i. Transfer data between external devices
         1. Keyboard to monitor
         2. Microphone to speaker, so on
      ii. Storage device
         1. Network to memory
      iii. Data processing
         1. Internal or external source and destination
      iv. Control external devices
   d. Parts of a computer
      i. I/O
         1. Mouse, keyboard main examples we think of
         2. Other peripherals count too (like speakers)
      ii. Main memory: RAM, caches
      iii. System bus, move data around
      iv. CPU
         1. Registers, store values
         2. ALU, perform operations
         3. Internal bus, transfer data
         4. Control unit, the brain
            a. Sequencing logic, where to go next
            b. Control unit registers, decoders
            c. Control memory