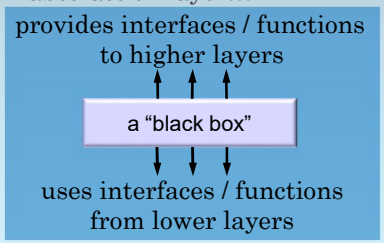
University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

ECE 120: Introduction to Computing

Abstraction Layers in Digital Systems

Abstraction Separates Function from Implementation

An abstraction layer…



Many implementations are possible!

Humans Learn to Use Many Abstractions

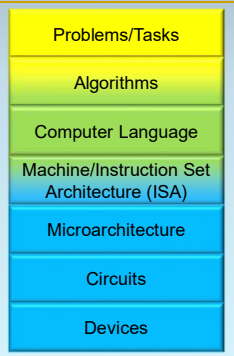
Example: taxi
- function: take customer to a human-specified location
- lower layers: car / van / truck / limousine / motorcycle, driver / autonomous control!

Example: water faucet
- function: get water at a specific (fuzzy) rate
- lower layers: plumbing, water tanks / cisterns / wells / aquaducts, valves, knobs

Digital Systems are Comprised of Seven Layers

The colors indicate the typical basis for each layer
- **human language / theory**
- **software**
- **digital hardware**

(figure based on Patt & Patel Ch. 1)

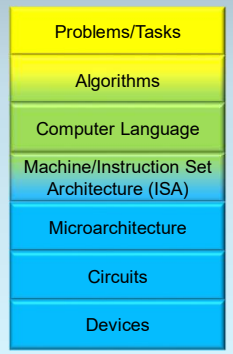


Don't Talk to Electrons. Please.

Below the device layer are the electrons.

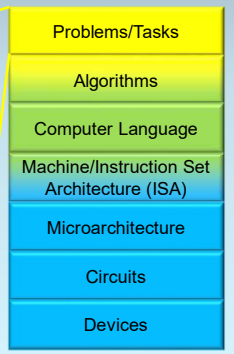
We'd like to just tell them what we want done.

But they don't seem to listen.



Human Problem Descriptions are the Top Layer

Problems/Tasks
- stated in natural (human) language
- For example: What's the sum of numbers between 1 and 3?



Sorry, But Your Answer is Wrong

Question:
What's the sum of numbers between 1 and 3?

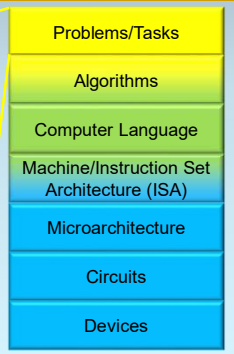
Did you answer 6? (Did you include 1 and 3?)
- What's between the bread in a peanut butter sandwich? **Is the bread between the bread?**

Did you answer 2? (Did you exclude 1 and 3?)
- What about **2.5**? What about **Π/2**? **e**?

Did you answer infinity?
- You're still wrong! (Too geeky! You'll probably end up as a professor one day.)

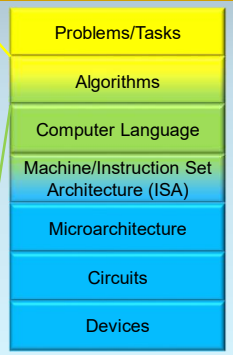
Human Languages Suffer from Ambiguity

Problems/Tasks
- stated in natural (human) language
- For example: What's the sum of numbers between 1 and 3?
- **Problem inherent to natural language: ambiguity.**
- Another example: Time flies like an arrow.



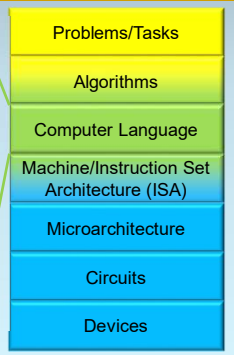
A Task Can be Solved by Many Algorithms

Algorithms
- a step-by-step process to solve a problem
- requires three things:
  - **definiteness** (no ambiguity)
  - **effective computability** (each step simple enough for a computer)
  - **finiteness** (finishes)



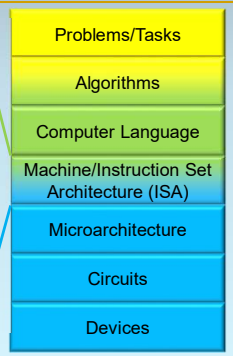
An Algorithm Can be Implemented in Many Languages

Computer Language
- 1000s of choices
- examples: C, C++, Java, Python
- we use C in 120 & 220
  - easy mapping to lower levels
  - a subset of other languages



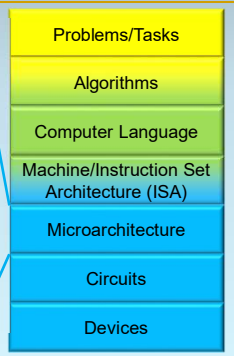
A Language Can be Implemented with Many ISAs

Machine/Instruction Set Architecture (ISA)
- interface between software and hardware
- examples: x86, ARM, PowerPC



An ISA Can be Executed by Many Microarchitectures

Microarchitecture
- digital hardware
- executes instructions from an ISA
- examples
  - X86 ISA: i5, i7, Opteron, Phenom
  - ARM: Cortex A15, Cortex A9, Kynetis K



Our Class Builds from the Ground Up

Your future work!

Future classes (CS374)

Week #3 (plus ECE220)

ECE120 covers these layers from the ground up (more on circuits in ECE110/210; more on devices in later years).