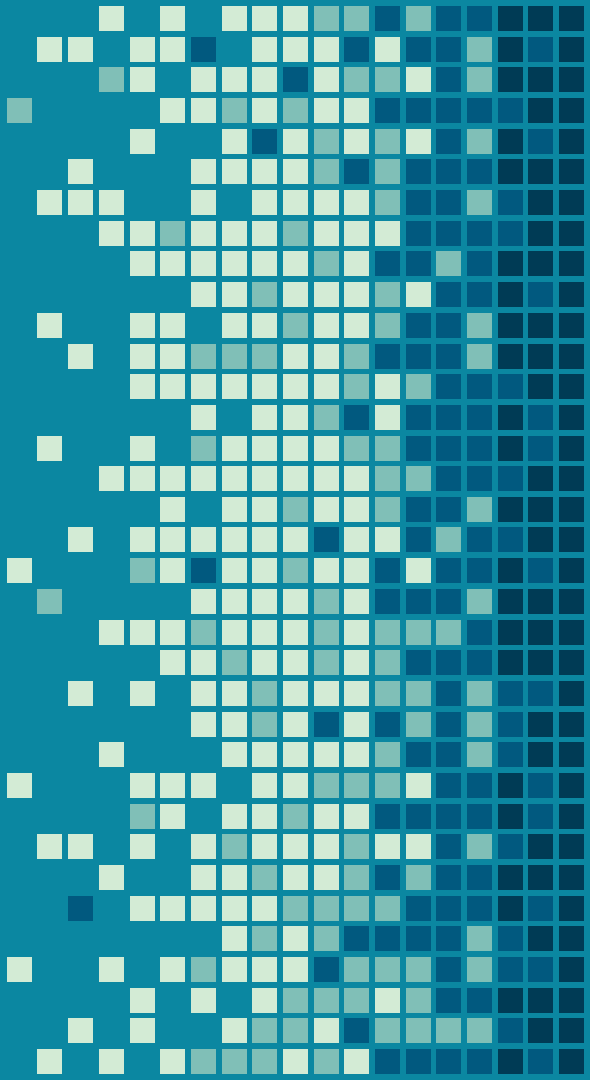
INTRODUCTION TO COMPUTING

Michael D'Argenio – mjdargen@ncsu.edu
Electrical Engineering – SS 2019 – Duke TIP



WHAT IS A COMPUTER?

- “A computer is a machine that can be instructed to carry out sequences of arithmetic or logical operations automatically.”

A computer is a machine that can:

1. Accept input
2. Execute a procedure
3. Produce output

Computer must have:

- Input/Output or I/O
- Processor
- Memory (requirement for modern computers)

Programming

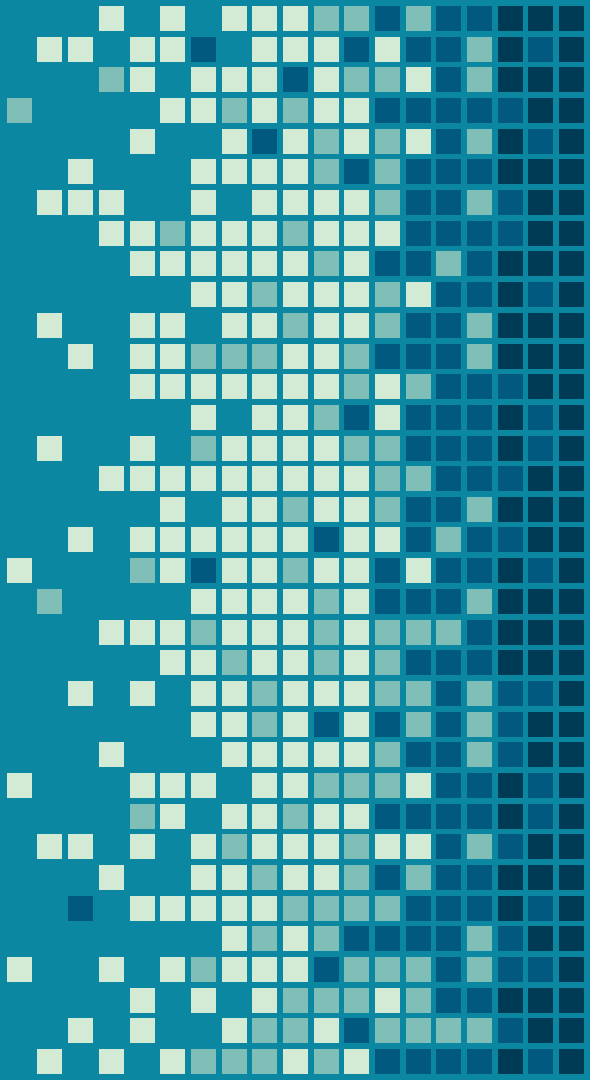
- How we tell processors what to do.
- Programming can either be
  - Mechanical
  - Hardware
  - Software

How computers differ from machines:

1. Machines amplify or extend our physical abilities while computers amplify and extend our mental abilities.
2. Machines are designed to perform a few specific tasks while computers can be programmed to perform many tasks.

There exists very simple computer models that provide the framework to perform any possible computation.

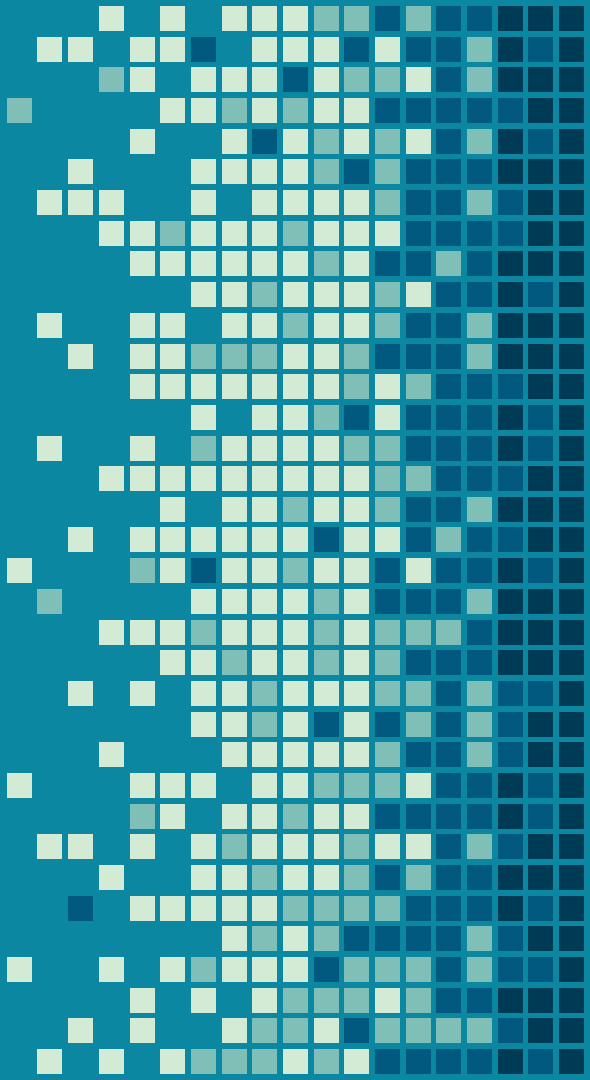
HISTORY OF COMPUTING



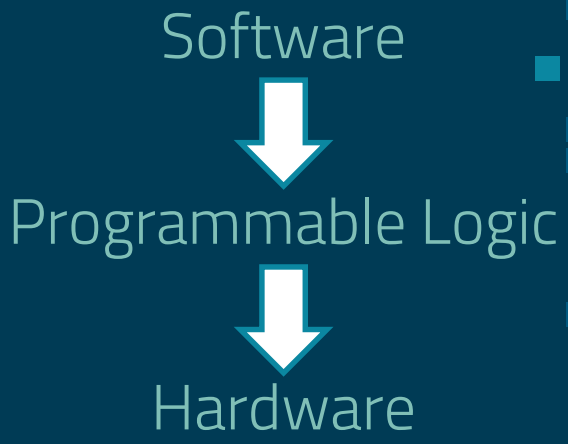
Crash Course History of Computing

- https://youtu.be/O5nskjZ_GoI
- https://youtu.be/LN0ucKNX0hc

HOW IS IT ELECTRICAL ENGINEERING?



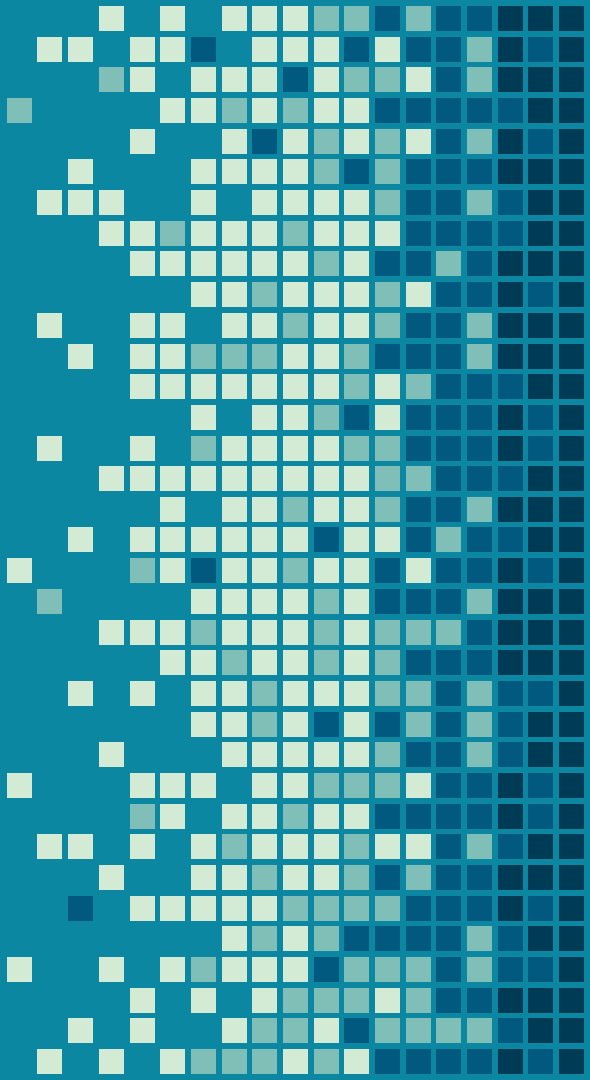
LEVELS OF ABSTRACTION



Fundamental Theorem of Software Engineering:

"We can solve any problem by introducing an extra level of indirection.“ – Andrew Koenig

INTRO TO THE WORLD OF COMPUTING



The World of Computing

- Computers are dumb.
  - They do exactly what we tell them to do, nothing more.
  - If something is wrong, 99.99999999% of the time:
  - It is because we told it to do the wrong thing OR
  - We told it to do the right thing in the wrong way.

- Approach:
  - Build understanding from the bottom up.

Bits → Gates → Processor → Instructions → C Programming

Two Recurring Themes

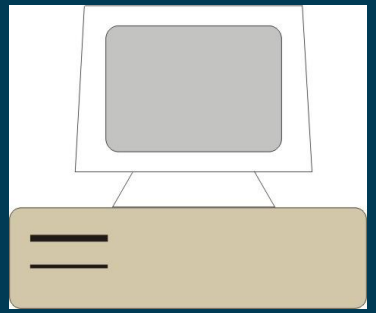
- Abstraction
  - Productivity enhancer – don't need to worry about details.
  - Can drive a car without knowing how the internal combustion engine works… until something goes wrong!
  - Where is the spark plugs? What is a spark plug?
  - Important to understand the components and how they work together.
- Hardware vs. Software
  - It's not either/or – both are important components of a computer system.
  - Even if you specialize in one, you should understand capabilities and limitations of both.

Big Idea #1: Universal Computing Device

- All computers, given enough time and memory, are capable of computing exactly the same things.

Smart phone

=



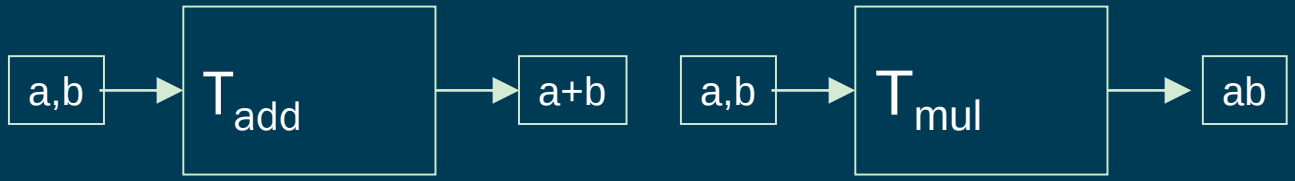
Workstation

=

Supercomputer

Turing Machine

- Mathematical model of a device that can perform any computation – Alan Turing (1937)
  - ability to read/write symbols on an infinite "tape"
  - state transitions, based on current state & input
- Every computation can be performed by some Turing machine. *(Turing's thesis)*



Turing machine that adds

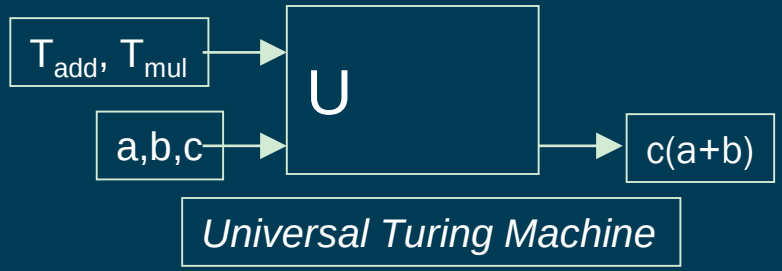
Turing machine that multiplies

For more info about Turing machines, see http://www.wikipedia.org/wiki/Turing_machine/

For more about Alan Turing, see http://www.turing.org.uk/turing/

Universal Turing Machine

- A machine that can implement all Turing machines -- this is also a Turing machine!
  - inputs: data, plus a description of computation

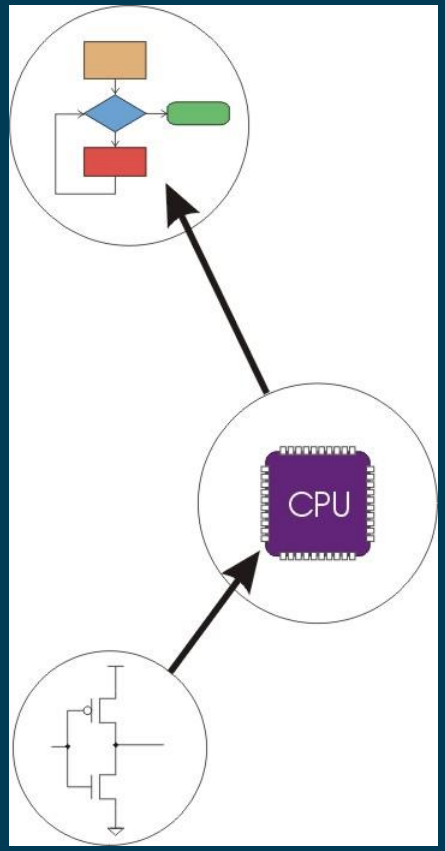


- U is programmable – so is a computer!
  - instructions are part of the input data
  - a computer can emulate a Universal Turing Machine
- *A computer is a universal computing device.*

From Theory to Practice

- In theory, a computer can *compute* anything that's possible to compute given enough *memory* and *time.*
- In practice, *solving problems* involves computing under constraints.
  - time
    - weather forecast, next frame of animation, ...
  - cost
    - cell phone, automotive engine controller, ...
  - power
    - personal electronics, remote sensors, …

Big Idea #2: Transformation betw. Layers



Problems

Algorithms

Language

Instruction Set Architecture

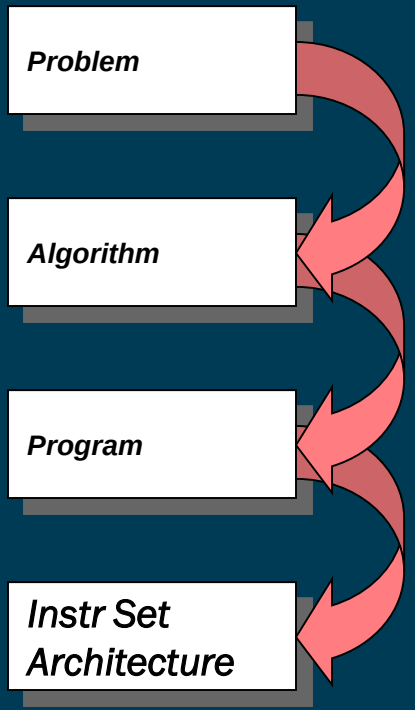
Microarchitecture

Circuits

Devices

How to solve problems using a computer?

A systematic sequence of transformations between levels of abstraction.

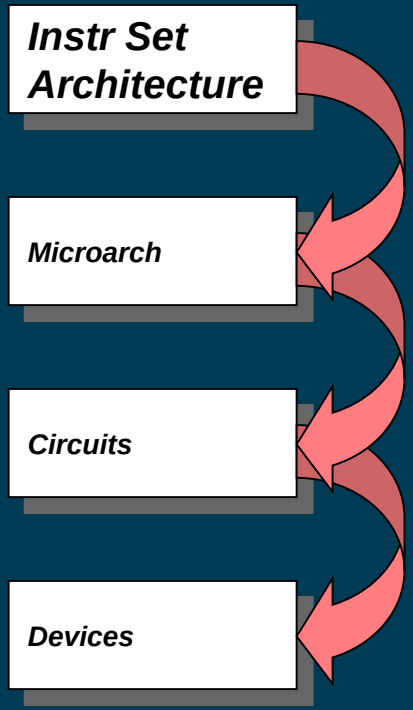


Software Design:
choose algorithms and data structures

Programming:
use language to express design

Compiling/Interpreting:
convert language to machine instructions

Deeper and Deeper…



Processor Design:
choose structures to implement ISA

Logic/Circuit Design:
gates and low-level circuits to implement components

Process Engineering & Fabrication:
develop and manufacture lowest-level components

Descriptions of Each Level

- **Problem Statement**
  - stated using "natural language"
  - may be ambiguous, imprecise
- **Algorithm**
  - step-by-step procedure, guaranteed to finish
  - definiteness, effective computability, finiteness
- **Program**
  - express the algorithm using a computer language
  - high-level language, low-level language
- **Instruction Set Architecture (ISA)**
  - specifies the set of instructions the computer can perform
  - data types, addressing mode

Descriptions of Each Level (cont.)

- **Microarchitecture**
  - detailed organization of a processor implementation
  - different implementations of a single ISA
- **Logic Circuits**
  - combine basic operations to realize microarchitecture
  - many different ways to implement a single function (e.g., addition)
- **Devices**
  - properties of materials, manufacturability

Many Choices at Each Level