Bits of Architecture

Introduction to Instructions

Computers Understand Instructions

Back to Basics

- **Instructions** are the language of processors
  - **Instruction Sets** are the vocabulary
- There are many instruction sets out there
  - **RISC-V**, x86, MIPS, Power
- **Stored-Program Concept**
  - Instructions and data can be stored in memory and are easy to change

High-Level Languages vs Instructions

How Assembly and High-Level Code Differs

- High-level languages often allow us to be more expressive
  - std::sort
- Instructions are (generally) primitive operations
  - add, multiply, divide, etc.
- Instructions use **registers**

C++

```
a = b + c + d + e;
```

Assembly

```
add a, b, c
add a, a, d
add a, a, e
```

Classes of Instructions

Classes of Instructions

- **Arithmetic**
  - E.g., Add/Subtract
- **Data Transfer**
  - E.g., Load/Store
- **Logical**
  - E.g., And/Xor
- **Shift**
  - E.g., Shift Logical Left/Right
- **Control Flow**
  - E.g., Conditional/Unconditional Branches

Design Principles

Design Principles

1. Simplicity favors regularity
2. Smaller is Faster
3. Good design demands good compromises