

## Introduction

---

- **Rapidly changing field:**
  - vacuum tube -> transistor -> IC -> VLSI (see section 1.4)
  - doubling every 1.5 years:
    - memory capacity*
    - processor speed* (Due to advances in technology and organization)
- **Things you' ll be learning:**
  - how computers work, a basic foundation
  - how to analyze their performance (or how not to!)
  - issues affecting modern processors (caches, pipelines)
- **Why learn this stuff?**
  - you want to call yourself a “computer scientist”
  - you want to build software people use (need performance)
  - you need to make a purchasing decision or offer “expert” advice

## What is a computer?

---

- **Components:**
  - input (mouse, keyboard)
  - output (display, printer)
  - memory (disk drives, DRAM, SRAM, CD)
  - network
- **Our primary focus: the processor (datapath and control)**
  - implemented using millions of transistors
  - Impossible to understand by looking at each transistor
  - We need...

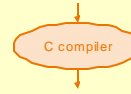
## Abstraction

- Delving into the depths reveals more information
- An abstraction omits unneeded detail, helps us cope with complexity

*What are some of the details that appear in these familiar abstractions?*

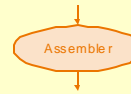
High-level  
language  
program  
(in C)

```
swap(int v[], int k)
{int temp;
 temp = v[k];
 v[k] = v[k+1];
 v[k+1] = temp;
}
```



Assembly  
language  
program  
(for MIPS)

```
swap:
 muli $2, $5, 4
 add $2, $4, $2
 lw $15, 0($2)
 lw $16, 4($2)
 sw $16, 0($2)
 sw $15, 4($2)
 jr $31
```



Binary machine  
language  
program  
(for MIPS)

```
00000000101000010000000000 00 11000
0000000010001110000110000 01 00001
10001100011000100000000000 00 00000
10001100111100100000000000 00 00100
10101100111100100000000000 00 00000
10101100011000100000000000 00 00100
00000011111000000000000000 00 01000
```

## Instruction Set Architecture

- A very important abstraction
  - interface between hardware and low-level software
  - standardizes instructions, machine language bit patterns, etc.
  - advantage: *different implementations of the same architecture*
  - disadvantage: *sometimes prevents using new innovations*

*True or False: Binary compatibility is extraordinarily important?*

- Modern instruction set architectures:
  - 80x86/Pentium/K6, PowerPC, DEC Alpha, MIPS, SPARC, HP

## Where we are headed

---

- Performance issues (Chapter 2) *vocabulary and motivation*
- A specific instruction set architecture (Chapter 3)
- Arithmetic and how to build an ALU (Chapter 4)
- Constructing a processor to execute our instructions (Chapter 5)
- Pipelining to improve performance (Chapter 6)
- Memory: caches and virtual memory (Chapter 7)
- I/O (Chapter 8)

**Key to a good grade: reading the book!**