

---

# **CS430 - Computer Architecture**

**William J. Taffe**

**Fall 2002**

**using slides from**

**CS61C - Machine Structures**

**Dave Patterson**

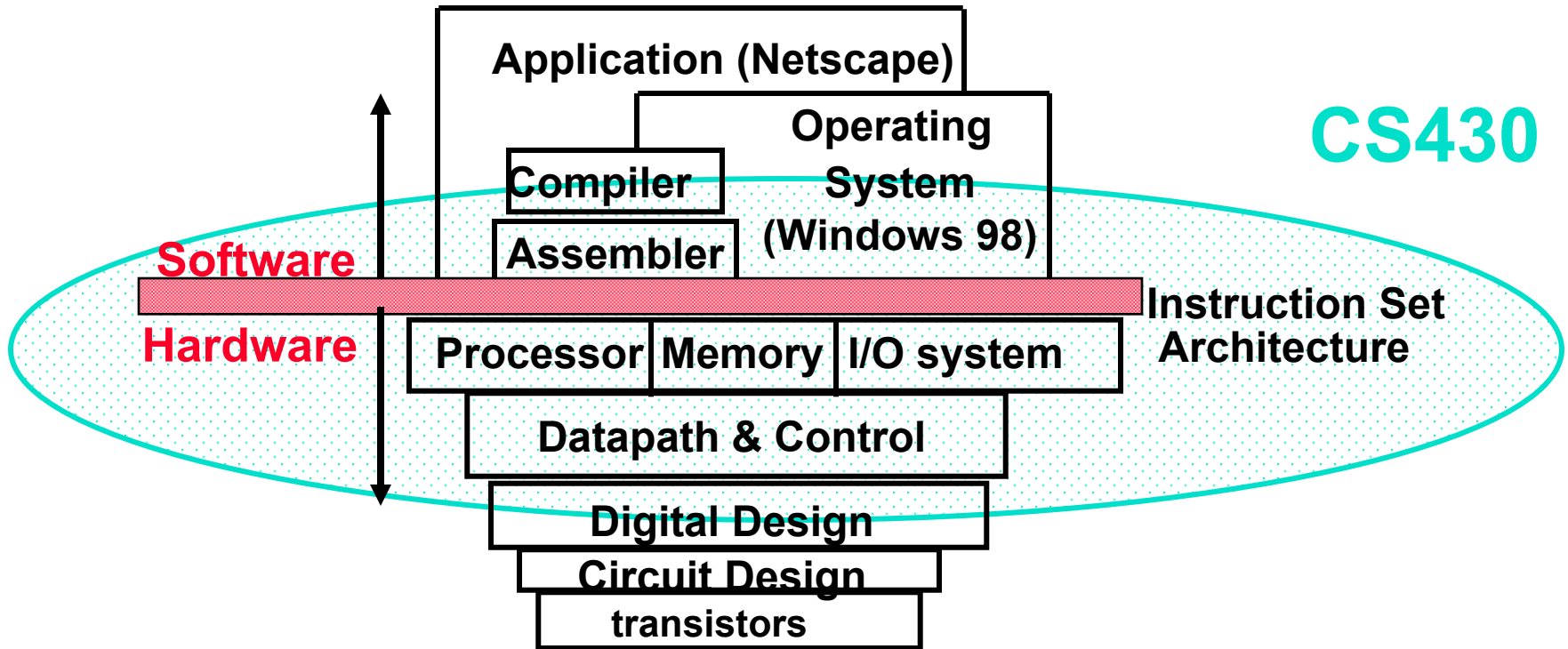
**Fall 2000**

# Overview

---

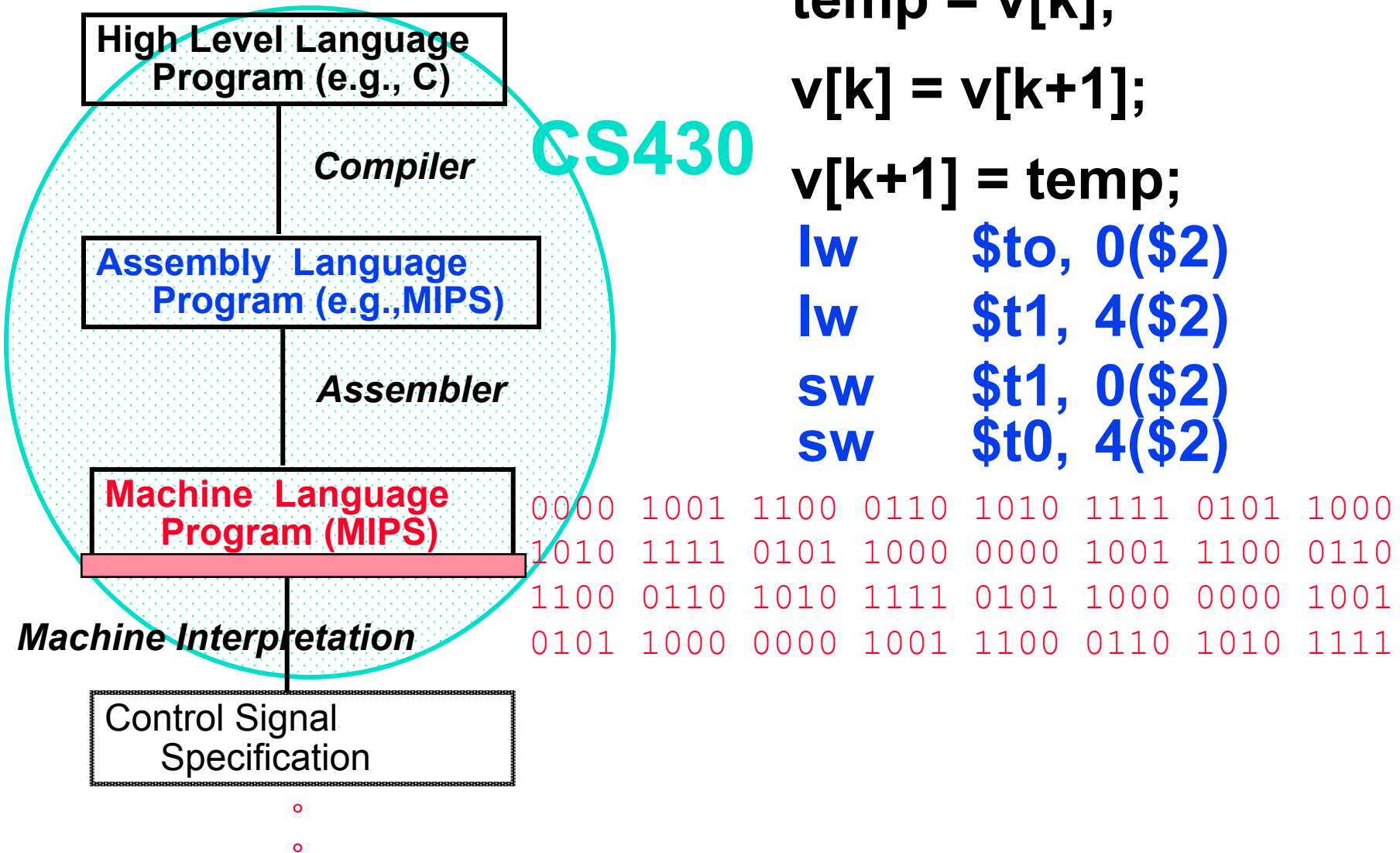
- **Intro to Machine Structures**
- **Organization and Anatomy of a Computer**
- **Rapid Technological Change**
- **Course Style, Philosophy and Structure**
- **Conclusion**

# What are “Machine Structures”?

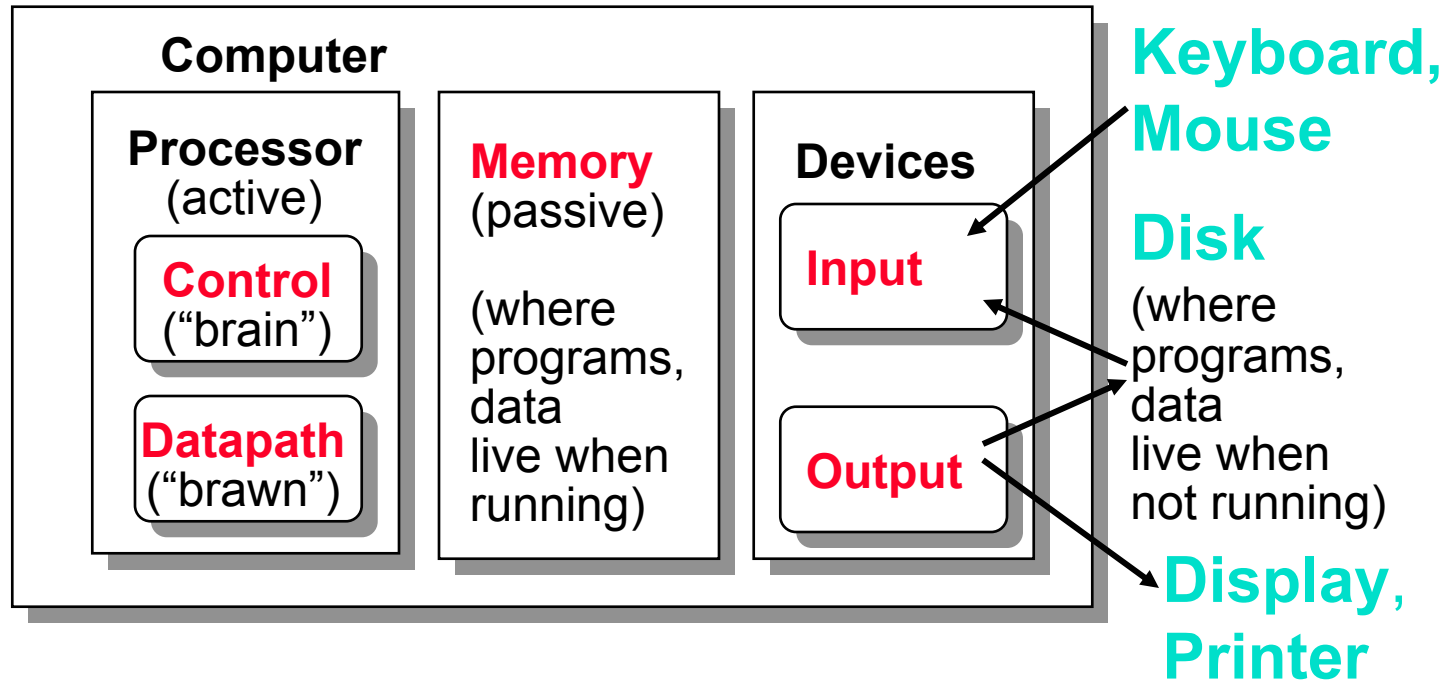
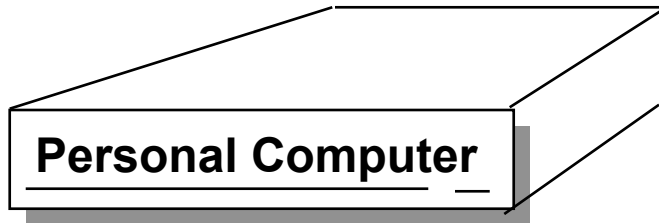


- Coordination of many *levels of abstraction*

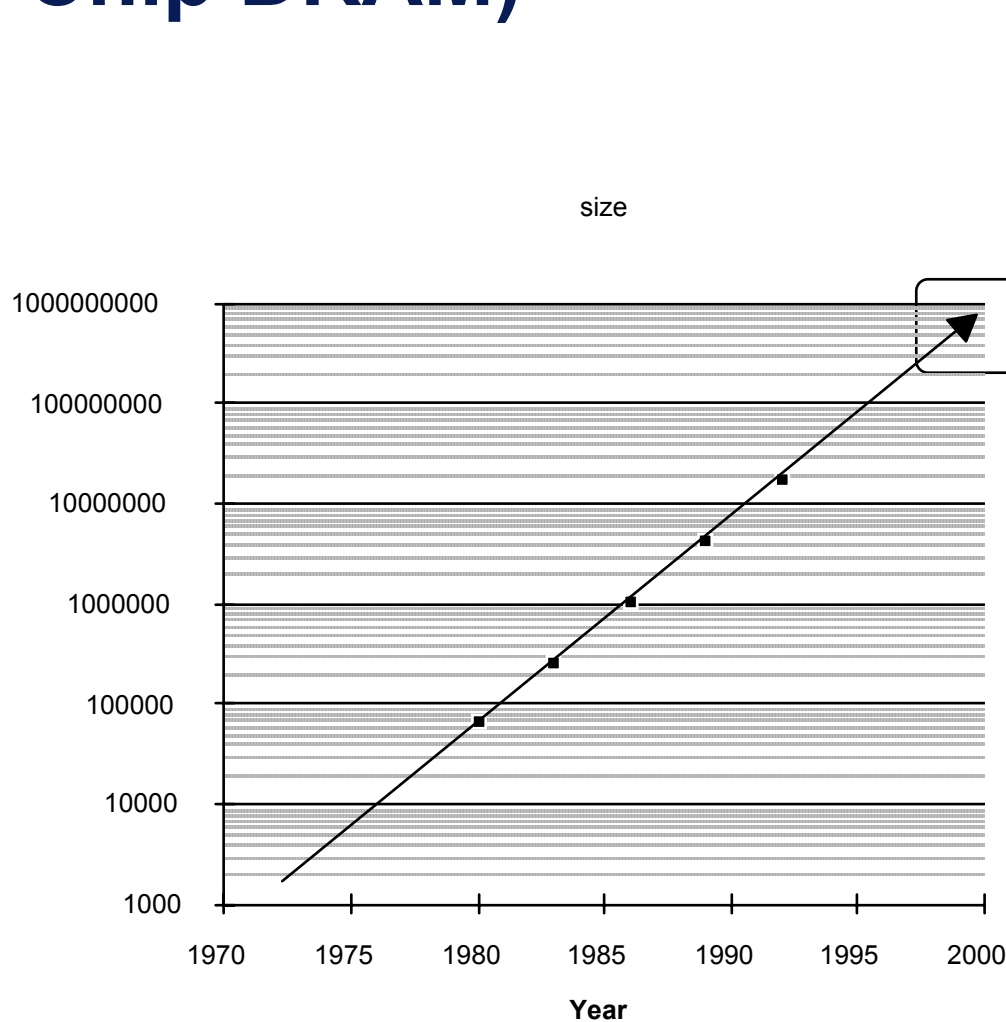
# Levels of Representation



# Anatomy: 5 components of any Computer



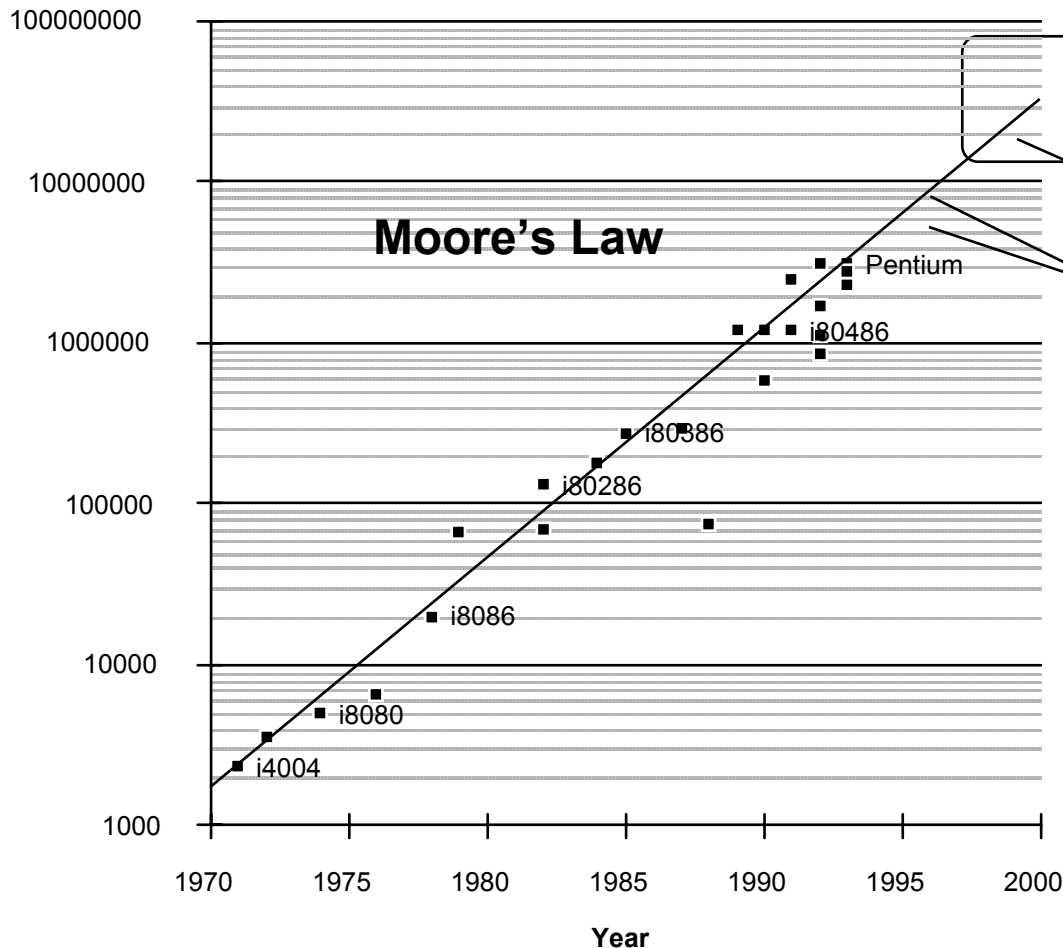
# Technology Trends: Memory Capacity (1 Chip DRAM)



year	size(Megabit)
1980	0.0625
1983	0.25
1986	1
1989	4
1992	16
1996	64
2000	256

**Now 1.4X/yr, or  
doubling every 2 years  
4000X since 1980**

# Technology Trends: Microprocessor Capacity

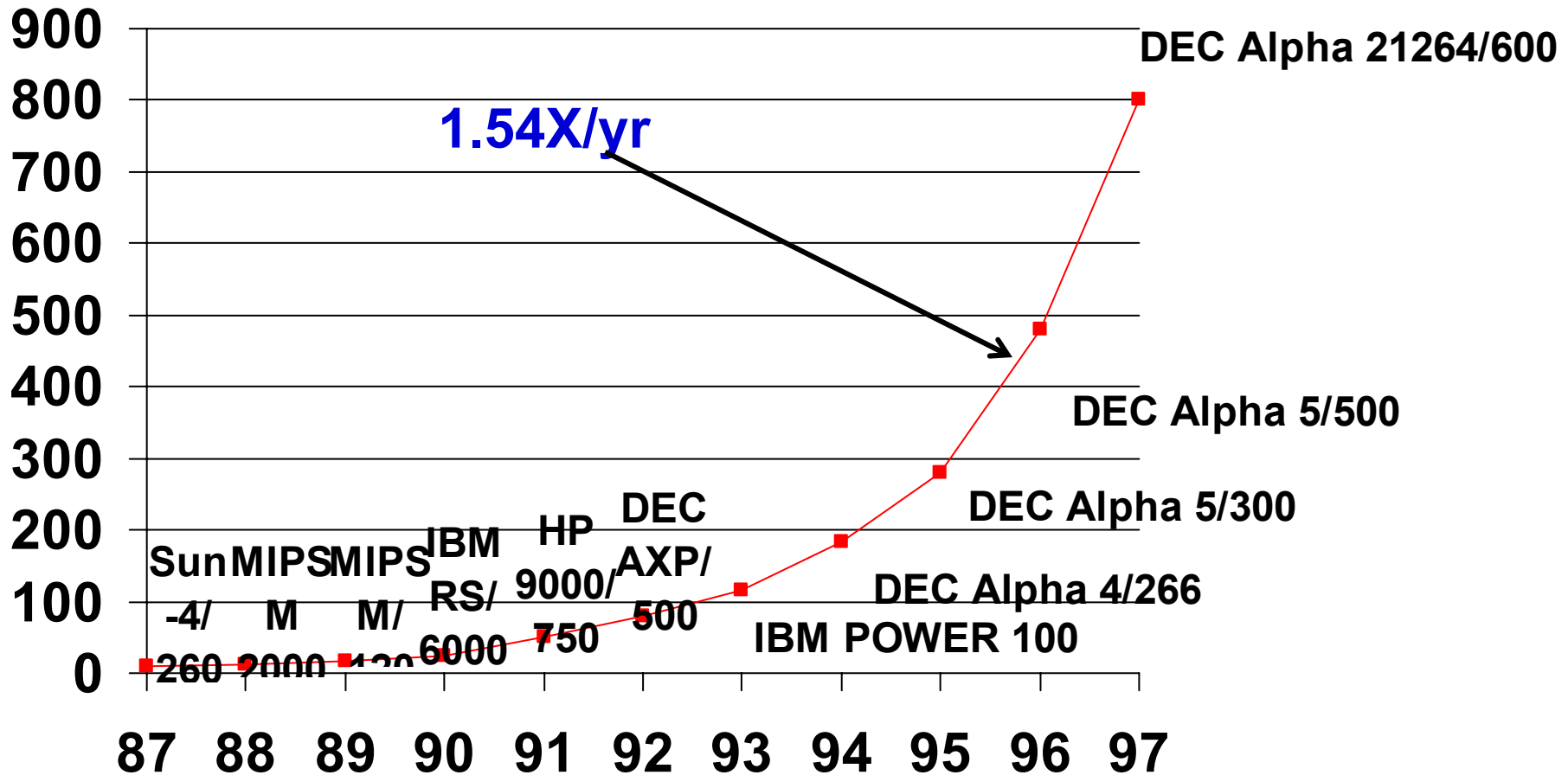


**Alpha 21264: 15 million**  
**Pentium Pro: 5.5 million**  
**PowerPC 620: 6.9 million**  
**Alpha 21164: 9.3 million**  
**Sparc Ultra: 5.2 million**

**2X transistors/Chip  
Every 1.5 years**

**Called**  
**“Moore’s Law”**

# Technology Trends: Processor Performance



**Processor performance increase/year, mistakenly referred to as Moore's Law (transistors/chip)**

# Computer Technology => Dramatic Change

## ◦ Processor

- 2X in speed every 1.5 years;  
100X performance in last decade

## ◦ Memory

- DRAM capacity: 2x / 2 years; 64X size in last decade
- Cost per bit: improves about 25% per year

## ◦ Disk

- capacity: > 2X in size every 1.0 years
- Cost per bit: improves about 100% per year
- 120X size in last decade

# Computer Technology => Dramatic Change

## ◦ State-of-the-art PC when you graduate:

- Processor clock speed: 4000 MegaHertz  
(4.0 GigaHertz)
- Memory capacity: 1000 MegaByte  
(1.0 GigaBytes)
- Disk capacity: 1000 GigaBytes  
(1.0 TeraBytes)
- New units! Mega => Giga, Giga => Tera

# Why Study Machine Structures?

◦ **CHANGE; It's exciting!; It has never been more exciting!**

◦ **It impacts every other aspect of computer science**



Bionics:  
Sensors in latex fingers instantly register hot and cold, and an electronic interface in his artificial limb stimulates the nerve endings in his upper arm, which then pass the information to his brain. The \$3,000 system allows his hand to feel pressure and weight, so for the first time since losing his arms in a 1986 accident, he can pick up a can of soda without crushing it or having it slip through his fingers. *One Digital Day*

# CS430: So what's in it for me?

---

- **Machine structures from a programmer's view**
  - **What the programmer writes**
  - **How it is converted to something the computer understands**
  - **How the computer interprets the program**
  - **What makes programs go slow**

# **CS430: So what's in it for me?**

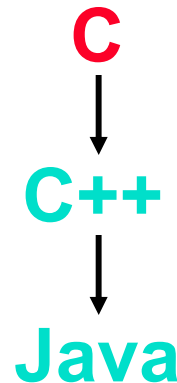
- **Learn big ideas in CS and engineering**
  - **5 Classic components of a Computer**
  - **Data can be anything (integers, floating point, characters): a program determines what it is**
  - **Stored program concept: instructions just data**
  - **Principle of Locality, exploited via a memory hierarchy (cache)**
  - **Greater performance by exploiting parallelism**
  - **Principle of abstraction, used to build systems as layers**
  - **Compilation v. interpretation thru system layers**
  - **Principles/Pitfalls of Performance Measurement**

# What 61C is not

---

## ◦ Learning C

- If you know one, you should be able to learn another programming language on your own
- Given that you know Java, should be easy to pick up its ancestor, C



## ◦ Assembly Language Programming

- This is a skill you will pick up, as a side effect of understanding the Big Ideas

## ◦ Hardware design

- Hardware at abstract level, with only a little bit of physical logic to give things perspective
- CS 215 teaches this

# CS430 Prerequisite

---

- **Students who have not taken CS215:**
  - **Will be dropped from class**

# CS 430 exams

---

° **Reduce the pressure of taking exams**

- **Three hour exams**
- **Our goal: test knowledge vs. speed writing**
- **Review meetings: in class/lab on Monday**
- **Open Notes**

# Course Problems

---

## ◦ What is cheating?

- Studying together in groups is encouraged
- Work must be your own
- Common examples of cheating: running out of time on a assignment and copy, person asks to borrow solution “just to take a look”, copying an exam question, ...
- Better off to skip assignment (3 exams, many homeworks, several labs, lots of writing, etc. How much can one assignment mean?)

## **And in Conclusion...**

- **14 weeks to learn big ideas in CS&E**
  - **Principle of abstraction, used to build systems as layers**
  - **Pliable Data: a program determines what it is**
  - **Stored program concept: instructions are just data**
  - **Principle of Locality, exploited via a memory hierarchy (cache)**
  - **Greater performance by exploiting parallelism (pipeline)**
  - **Compilation v. interpretation to move down layers of system**
  - **Principles/Pitfalls of Performance Measurement**

# And in Conclusion...

---

## ◦ Continued rapid improvement in Computing

- 2X every 1.5 years in processor speed; every 2.0 years in memory size; every 1.0 year in disk capacity; Moore's Law enables processor, memory (2X transistors/chip/ ~1.5 yrs)

## ◦ 5 classic components of all computers

Control   Datapath   Memory   Input   Output



Processor