Computer Science 104: Computer Organization, Design & Programming

Dr. Andrew (Drew) Hilton

General Information

Instructor: Andrew Hilton

Office: D213 LSRC

email: adhilton@cs.duke.edu

Office Hours: Monday 3pm-4pm, Wednesday 2pm-3pm

Teaching Assistants:

Razvan Dicu

Lindsay Kubasik

Geoffrey Lawler

Alex Sloan

More Information

- TA office hours: TBD
- Recitation Fridays

  Run by TA (Razvan)

  Ask questions

  Review material

  Etc…

Information

- **Questions encouraged**
- **Sakai**
  - **Turn in assignments**
  - **See announcments**
  - **Required reading**
- **Piazza**
  - **Discussions, questions, etc**
  - **Strongly recommended reading**
- **Course Web Page**

  http://www.cs.duke.edu/courses/spring12/cps104

Textbook, etc.

- **Text:** ***Computer Organization & Design.***
  (Patterson & Hennessy)
  - **You are expected to complete the assigned readings**
  - **Some material on the CD (e.g., Appendix)**
- **Read**
  - **Start reading Chapter 1 now**
  - **Optional: Brief History of Computers**
    - » **http://www.digitalcentury.com/encyclo/update/comp_hd.html**

- **Homework #1 Assigned due Feb 1.**

Grading

- Grade breakdown
  - Midterm Exam 1 23%
  - Midterm Exam 2 23%
  - Final Exam 30%
  - Homework 24%
- Late homework policy
  - 5 late days per person total for the semester
  - Days, not classes
  - After used up, no credit for late work.
- This course takes time, start assignments early.
  - Average 3-5 hrs/week from previous course evaluations.

Course Problems

- **Academic Conduct**
  - Duke Community Standard
  - Studying together in groups is encouraged
  - All written work must be your own, unless otherwise stated.
  - Common examples of cheating: running out of time on an assignment and then pick up someone else's output, person asks to borrow solution "just to take a look", copying an exam question, …
  - If you are not sure, please ask….
  - If you think I'd probably say no, its probably cheating
- **If I catch you cheating….**
  - You will receive a -100% grade (less than a 0)
  - You will be reported to the OSC

Course Problems

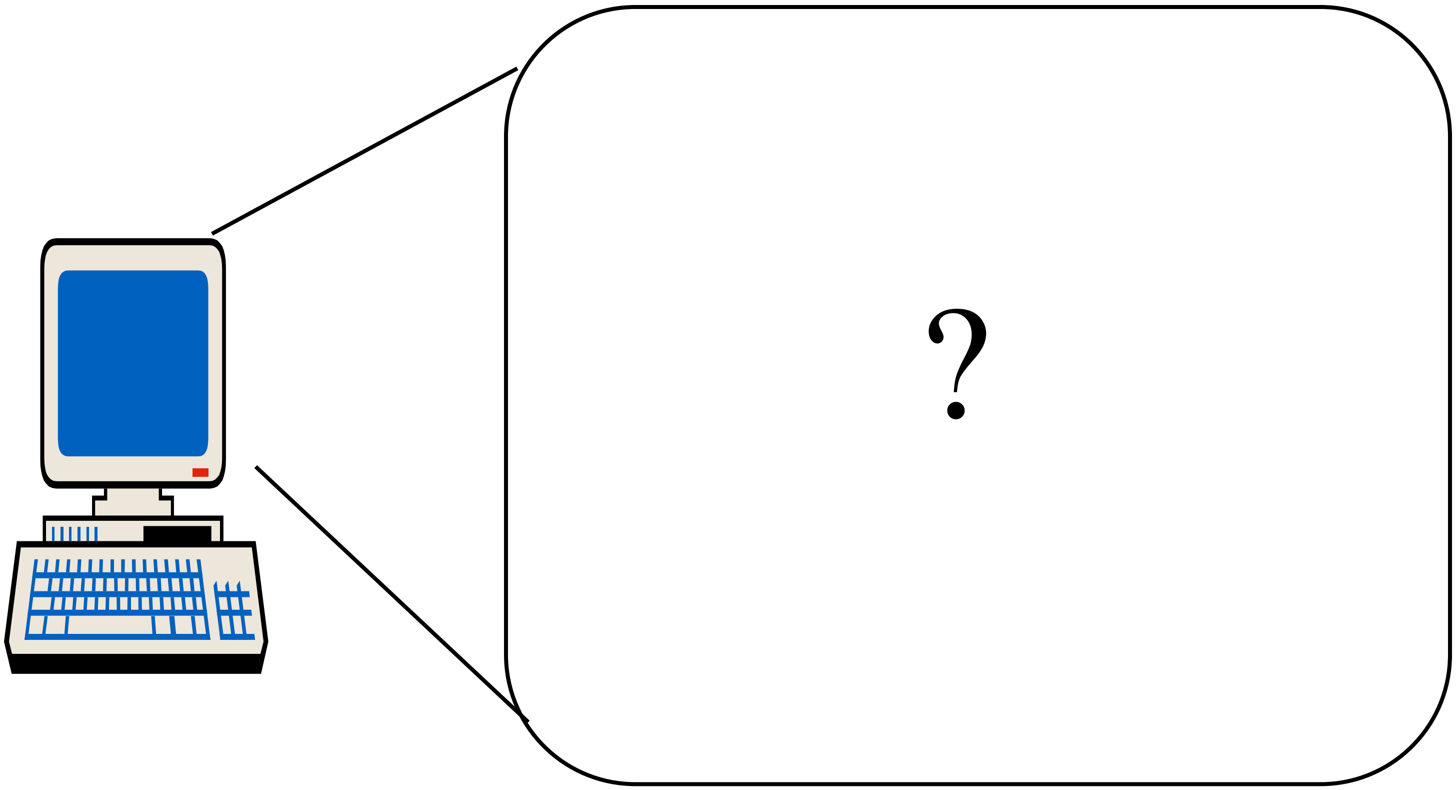
- Can't make midterms / final, other conflicts
  - Tell us early and we will schedule alternate time
- If you are having problems
  - See me
  - See DUS
  - See Academic Dean (very good resource)

Why Do You Have to Take This Course?

- You want to be a Computer Scientist
- You know how to program (CPS 6, 100)
- To be successful you don’t just program
- You have to understand the machine
  - Hardware: Processor, memory, disk, etc.
  - SW: Operating system, Programming Languages/Compilers
- What kind of computer scientist?
  - Databases, networks, facebook
  - Scientific computing (motion of planetary bodies, drug development, computational biology, economics, etc.)
  - Games, virtual reality
  - Embedded: Cell phones, mp3 player, cars
- Who’s code do you want controlling your brakes, airbag, financial transactions? Script kiddie or computer scientist.

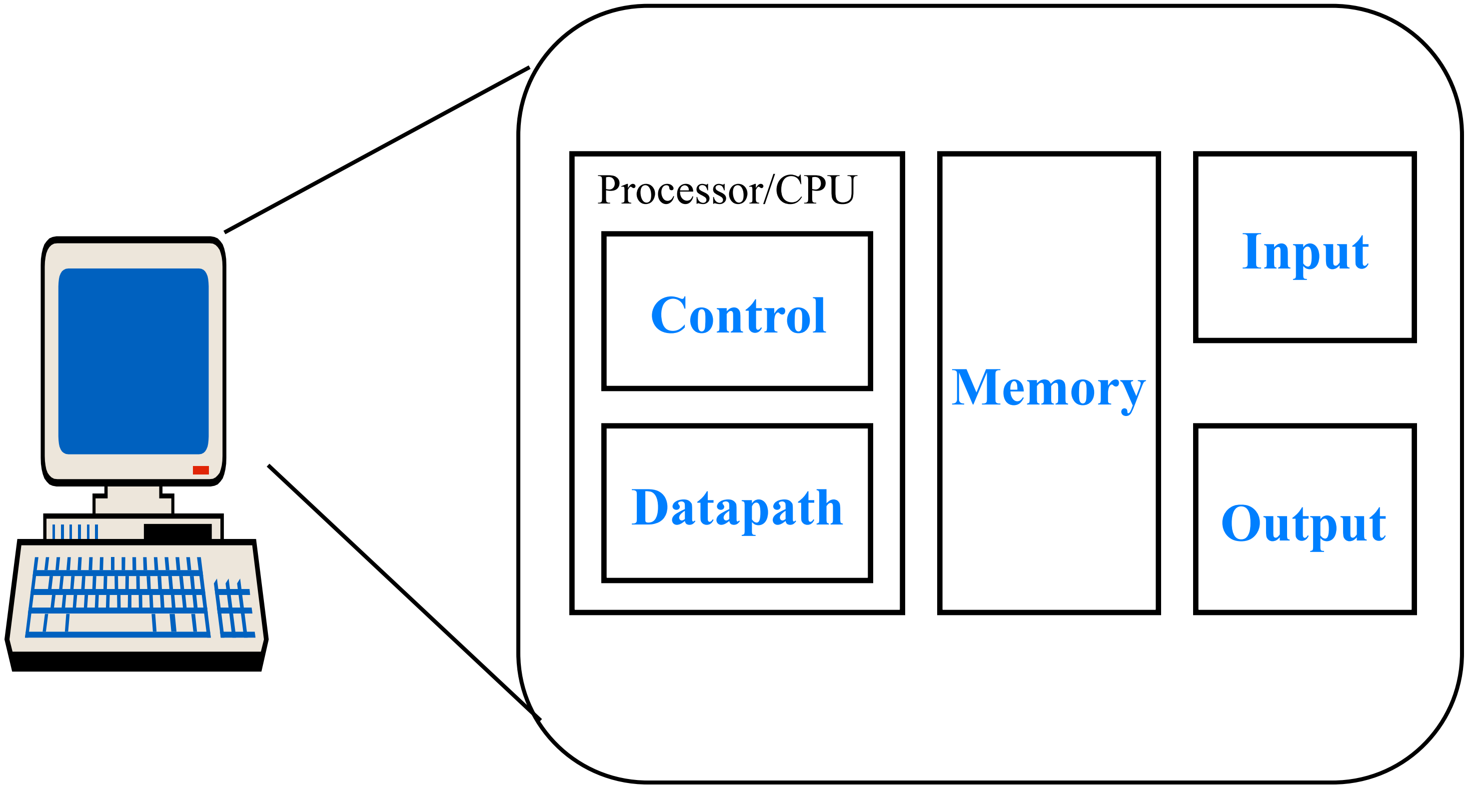
The Big Picture

- What is inside a computer?
- How does it execute a program?

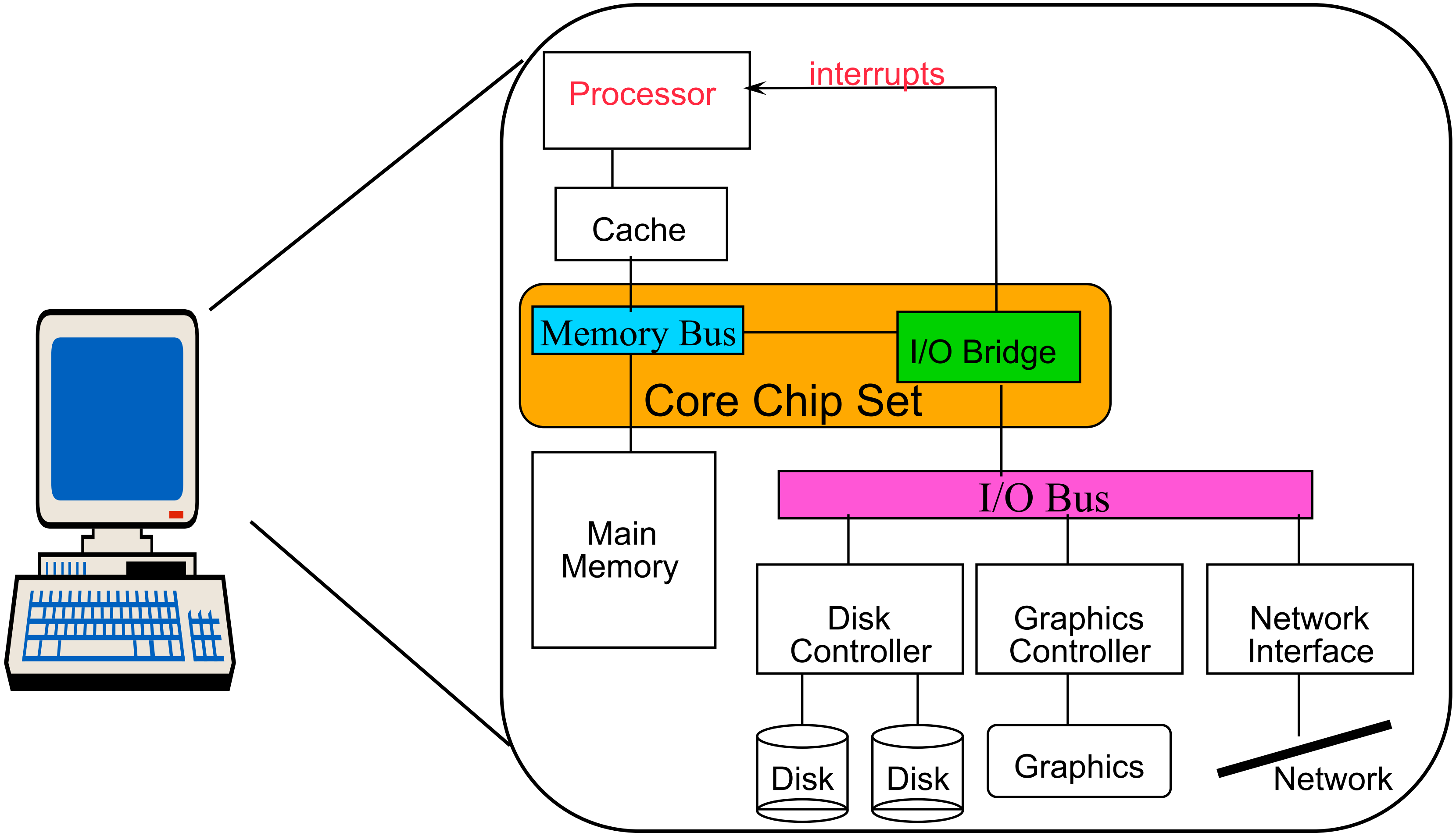


The Big Picture

- **The Five Classic Components of a Computer**

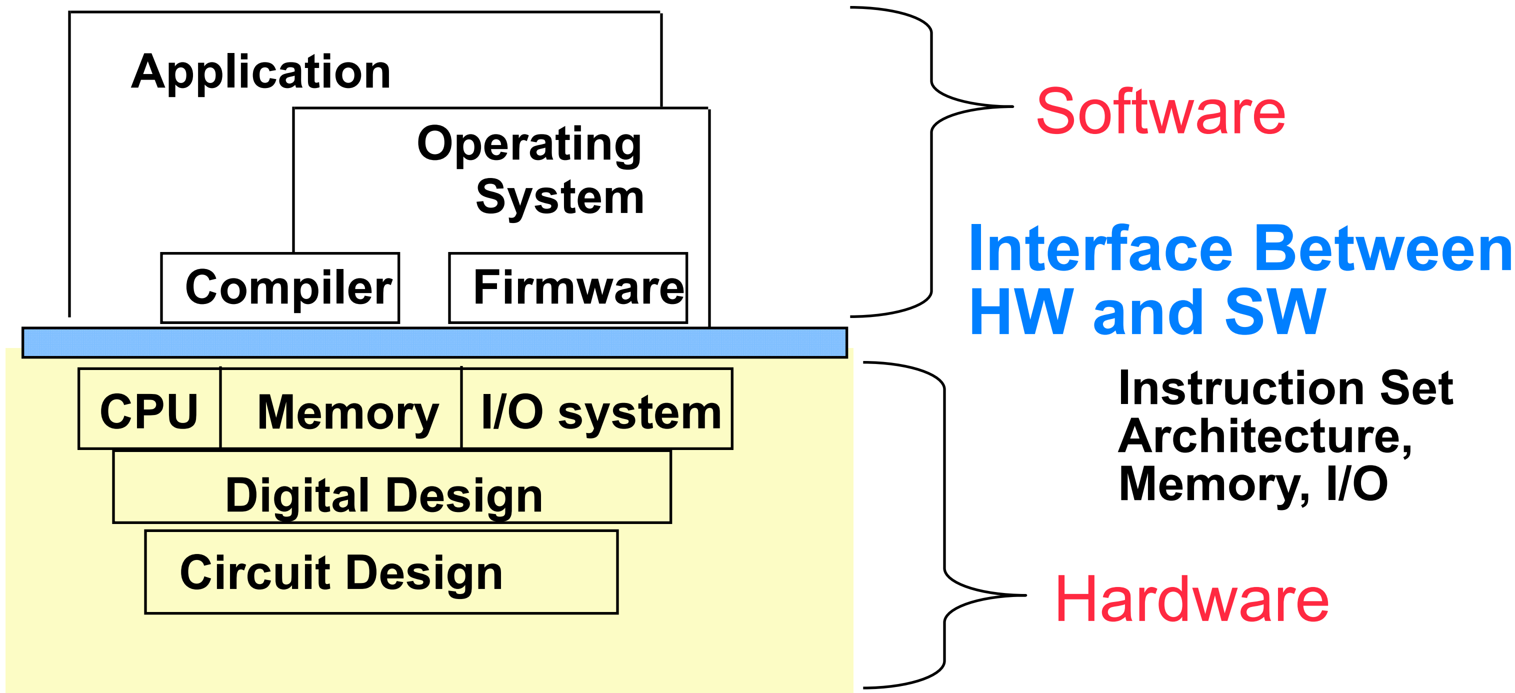


System Organization



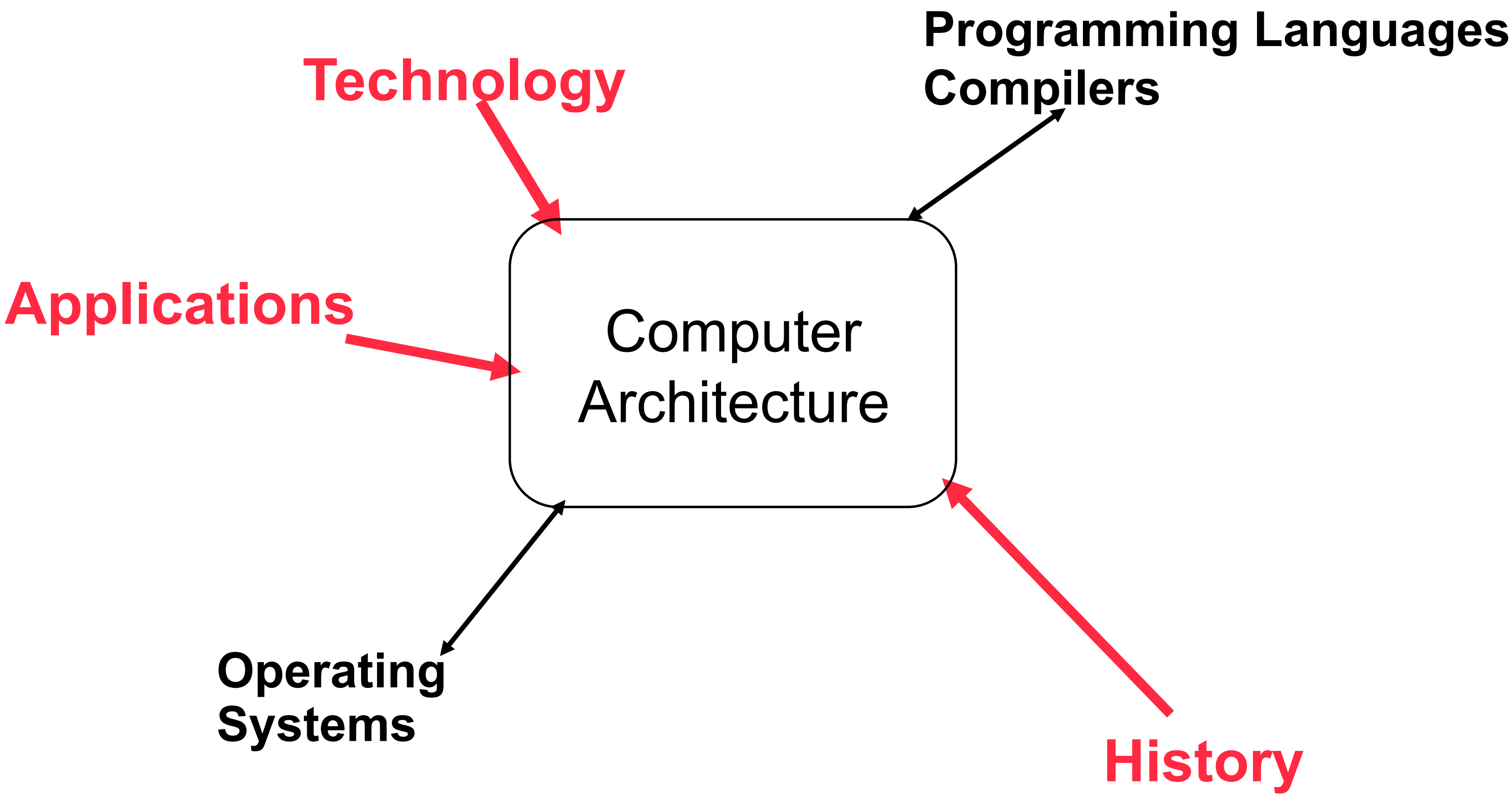
What is Computer Architecture?

- Coordination of levels of abstraction



- Under a set of rapidly changing *Forces*

Forces on Computer Architecture



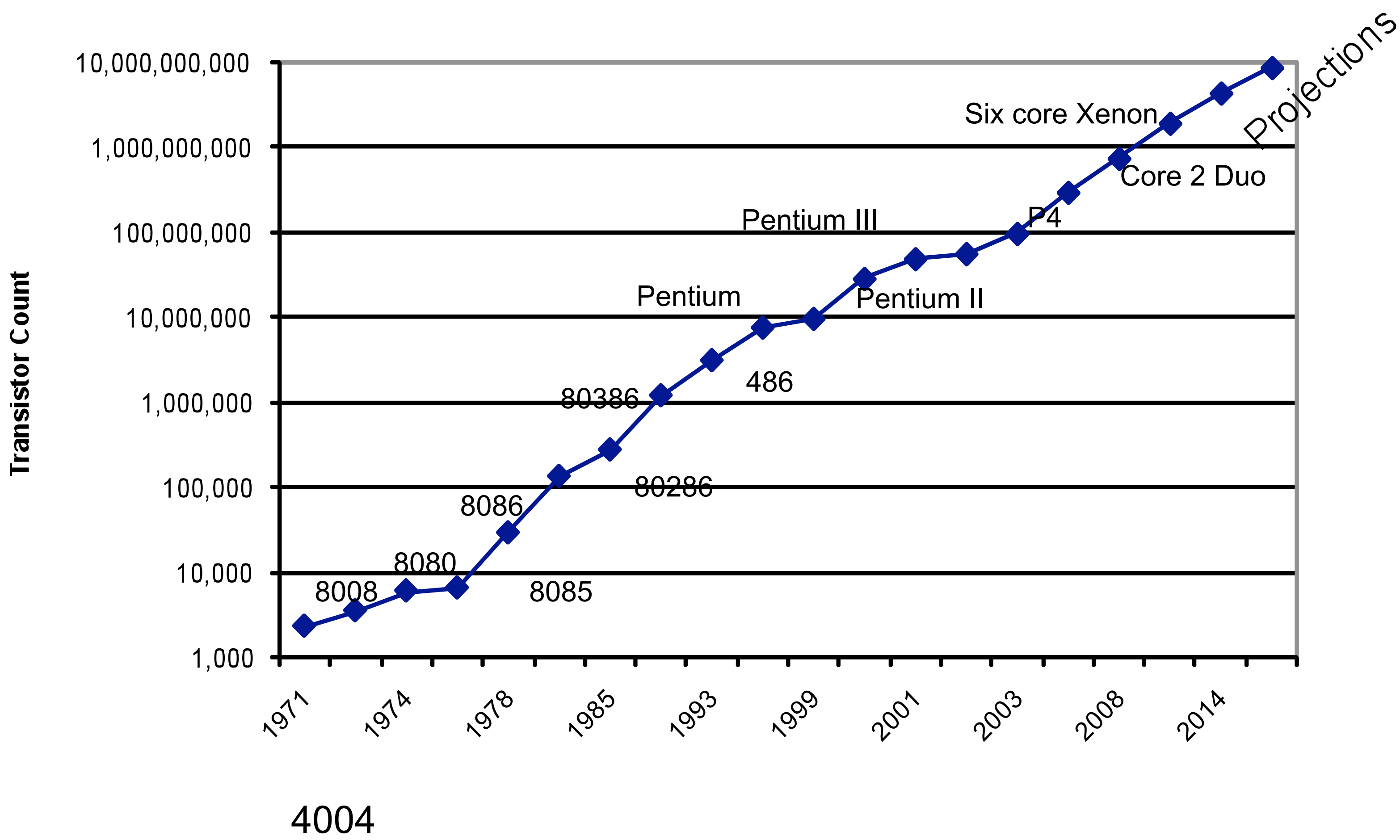
A Brief History of Computing

- 1645 Blaise Pascal Calculating Machine
- 1822 Charles Babbage
  - Difference Engine
  - Analytic Engine: Augusta Ada King first programmer (woman)
- < 1946 Eckert & Mauchly
  - ENIAC (Electronic Numerical Integrator and Calculator)
- 1947 John von Neumannn
  - Proposed Stored Program Computer
  - Properties of Today's computers
- 1949 Maurice Wilkes
  - EDSAC (Electronic Delay Storage Automatic Calculator)

Commercial Computers

Year	Name	Size (cu. ft.)	Adds/sec	Price
1951	UNIVAC I	1000	1,900	$1,000,000
1964	IBM S/360 Model 50	60	500,000	$1,000,000
1965	PDP-8	8	330,000	$16,000
1976	Cray-1	58	166,000,000	$4,000,000
1981	IBM PC	1	240,000	$3,000
1991	HP 9000 / model 750	2	50,000,000	$7,4000
1996	Intel Ppro PC	2	400,000,000	$4,400
2005	Intel Pentium4	0.25-2	4,000,000,000	< $1,000
2007	Intel Core2Duo	0.25-2	8,000,000,000	$300 - $1,000
2010	Quad Core	0.25-2	16,000,000,000	$100-$500

Microprocessor Trends



sense of scale

Other Technologies

- **Games**
  - **Console, handheld, PC**
  - **play each gameboy in the world for 60 seconds, finish in 190 years**
- **MP3 Players**
- **Cameras**
- **Cell Phones**

- **What is common among all these technologies?**

Levels of Representation

High Level Language Program

↓ **Compiler**

Assembly Language Program

↓ **Assembler**

Machine Language Program

↓ **Machine Interpretation**

Control Signal Specification

```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw $15,  0($2)
lw $16,  4($2)
sw       $16,     0($2)
sw       $15,     4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

Transistors turning on and off

What You Will Learn

- The basic operation of a computer
  - primitive operations (instructions)
  - arithmetic
  - Logic design (implement a simple processor)
  - instruction sequencing and processing
  - memory
  - input/output
  - etc.
- Understand the relationship between abstractions
  - interface design
  - high-level program to control signals (SW -> HW)
  - Astrachan "from the abstract to the ridiculous"
- Software performance depends on understanding underlying HW

Course Outline

- Introduction to Computer Organization
- Data Representations & Memory
- Instruction Set Architecture
- Assembly level programming
  - Instructions
  - Addressing, procedure calls and Exceptions
  - Linking & Loading
  - MIPS programming.
- Digital Logic
  - Digital Gates and Boolean Algebra
  - Arithmetic and Logic circuits, Finite State Machines (maybe)

Course Outline (continued):

- The Central Processing Unit (CPU)
  - The ALU
  - The data path
  - Finite State Control
- The Memory Hierarchy
  - Cache Memory
  - Virtual Memory and Paging
- Buses and Interrupts
- I/O Devices and Networks
- Advanced Computer Architecture (if there is time)
  - Pipelining
  - Multicore

Overview

- **First step in mapping high-level to machine**
  - **Data representations**

Outline

- **Representations**
- **Binary Numbers**
- **Integer numbers**
- **Floating-point numbers**
- **Characters**
- **Storage sizes: Bit, Byte, Word, Double-word**
- **Memory**
- **Arrays**
- **Pointers**

The Fundamental Rule of CS

- **There is one rule that governs all of CS….**
  - ➢**Anyone know what it is?**

The Fundamental Rule of CS

- There is one rule that governs all of CS….
  - Anyone know what it is?

Everything is a number

The Fundamental Rule of CS

- There is one rule that governs all of CS….
  - Anyone know what it is?

Everything is a number

- Computers can only work with numbers
  - If it's a number you can compute with it
  - If its not a number, a computer can't do anything with it.

What You Know Today

C++

```
...
int result;
double score;

double curve(double score) {
   return(score * 0.22124);
}
int main()
{
    int *x;
    ...
   result = x + result;
   cout << "Score is " <<
          curve(80) << endl;
   ...
}
```

JAVA

```
...
System.out.println("Please Enter
   In Your First Name: ");
String firstName =
   bufRead.readLine();
System.out.println("Please Enter
   In The Year You Were Born: ");
String bornYear =
   bufRead.readLine();
System.out.println("Please Enter
   In The Current Year: ");
String thisYear =
   bufRead.readLine();
int bYear =
   Integer.parseInt(bornYear);
int tYear =
   Integer.parseInt(thisYear);
int age = tYear - bYear ;
System.out.println("Hello " +
   firstName + ". You are " + age
   + " years old");
```

High Level to Assembly

High Level Language (C, C++, Fortran, Java, etc.)

- Statements
- Variables
- Operators
- Methods, functions, procedures

Assembly Language

- Instructions
- Registers
- Memory

Data Representation

- Compute two hundred twenty nine minus one hundred sixty seven divided by twelve
- Compute XIX - VII + IV
- We reason about numbers many different ways
- Computers store variables (data)
- Typically Numbers and Characters or composition of these
- The key is to use a representation that is “efficient”

Number Systems

- A number is a mathematical concept
  - 10
- Many ways to represent a number
  - 10, ten, 2x5, X, 100/10, ||||| |||||
- Symbols are used to create a representation

- Which representation is best for counting?
- Which representation is best for addition and subtraction?
- Which representation is best for multiplication and division?

More Number Systems

- **Humans use decimal (base 10)**
  - **digits 0-9 are composed to make larger numbers**

$$11 = 1*10^1 + 1*10^0$$

  - **weighted positional notation**
- **Addition and Subtraction are straightforward**
  - **carry and borrow (today called regrouping)**
- **Multiplication and Division less so**
  - **can use logarithms and then do adds and subtracts**

Changing Base (Radix)

- **Given 4 positions, what is the largest number you can represent?**

Number Systems for Computers

- Today's computers are built from transistors
- Transistor is either off or on
- Need to represent numbers using only off and on
  - two symbols
- off and on can represent the digits 0 and 1
  - BIT is Binary Digit
  - A bit can have a value of 0 or 1
- Binary representation
  - weighted positional notation using base 2

$$11_{10} = 1*2^3 + 1*2^1 + 1*2^0 = 1011_2$$

$$11_{10} = 8 + 2 + 1$$

What is largest number, given 4 bits?

Binary, Octal and Hexadecimal numbers

- Computers can input and output decimal numbers but they convert them to internal binary representation.
- Binary is good for computers, hard for us to read
  - Use numbers easily computed from binary
- Binary numbers use only two different digits: {0,1}
  - Example: 1200_{10} = 0000010010110000_{2}
- Octal numbers use 8 digits: {0 - 7}
  - Example: 1200_{10} = 04260_{8}
- Hexadecimal numbers use 16 digits: {0-9, A-F}
  - Example: 1200_{10} = $04B0_{16}$ = 0x04B0
  - does not distinguish between upper and lower case

Binary and Octal

- Easy to convert Binary numbers To/From Octal.
- Group the binary digits in groups of three bits and convert each group to an Octal digit.
- $2^3 = 8$

Bin.	Oct.
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Example:

$11\ 000\ 010\ 011\ 001\ 110\ 100\ 111\ 101\ 010\ 101_2$

$3\ \ 0\ \ 2\ \ 3\ \ 1\ \ 6\ \ 4\ \ 7\ \ 5\ \ 2\ \ 5_8$

Binary and Hex

- To convert to and from hex: group binary digits in groups of four and convert according to table
- $2^4 = 16$

Hex	Bin	Hex	Bin
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Example:

$1100\ 0010\ 0110\ 0111\ 0100\ 1111\ 1101\ 0101_2$

$C\quad 2\quad 6\quad 7\quad 4\quad F\quad D\quad 5_{16}$

Admin

- Read Ch. 1
- Optional: Brief History of Computers
  - http://www.digitalcentury.com/encyclo/update/comp_hd.html
- Homework #1 Assigned due Feb 1.

Next

- Start in on abstractions: data representation