# Computer Science 104:
## Computer Organization, Design & Programming

Alvin R. Lebeck

---

**General Information**

**Instructor:** Alvin R. Lebeck  
*Office:* D308 LSRC  
*Email:* alvy@cs.duke.edu  
*Office Hours:* Thurs 10:30-11:30, Fri 1:00 to 2:00, or by appointment

**Teaching Assistant:** Yang Liu  
*Office:* 005 North  
*Email:* yliu@cs.duke.edu  
*Office Hours:* Wed 3:00 to 4:00 (one more TBD)

---

**More Information**

- **Undergraduate TAs**  
  - To be decided  
  - Office Hours  
  - Read Forums  
  - Grade Homework

- **Altera DE2 Development Board**  
  - Fridays ("recitation") we will be hands on in class  
  - Yes, we really are meeting 3 times a week for 75min!

---

**Information**

- **I AM NOT PERFECT**  
  - Ask Questions!!

- **Course Web Page**  
  - [http://www.cs.duke.edu/courses/spring07/cps104](http://www.cs.duke.edu/courses/spring07/cps104)  
  - Lecture slides available on web before or shortly after class  
  - See lectures link for readings also

- **Blackboard ([http://courses.duke.edu](http://courses.duke.edu))**  
  - Grades  
  - Questions, discussions (communication->forums)

- **You are required to monitor both web sites**  
  - Homework will appear on web page  
  - If necessary, additional information about homework on blackboard  
  - You can post questions about homework on blackboard

---

**Textbook, etc.**

- **Text:** *Computer Organization & Design 3rd Ed.* (Patterson & Hennessy)  
  - You are expected to complete the assigned readings  
  - Some material on the CD (e.g., Appendix)

- **Read**  
  - Chapter 1, Ch 3.1-3.2 pgs 160-168, 3.6 pgs 189-197  
  - Optional: Brief History of Computers  

- **C tutorial** [http://computer.howstuffworks.com/c.htm](http://computer.howstuffworks.com/c.htm)

- **Homework #1 Assigned due Jan 24.**

---

**Grading**

- **Grade breakdown**  
  - Midterm Exams 30%  
  - Final Exam 20%  
  - Homework 25%  
  - Project 25%

- **Late homework policy**  
  - 10 point reduction for each day late  
  - No credit after the homework is graded and handed back.  
  - Feedback >> return results quickly >> grade almost immediately >> late homework is a hassle

- **This course takes time, start assignments early.**  
  - Average 3-5 hrs/week from previous course evaluations.  
  - May be different this semester with new hardware
Course Problems

- Academic Conduct
  - Duke Community Standard
  - Studying together in groups is encouraged
  - All written work must be your own, unless otherwise stated. Programs that are substantially the same as others will receive a grade of 0
  - 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, ...
- 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 plastic surgeon
  - You all know how to use knife, needle, thread
- To be successful you don’t just "cut & paste"
- You have to learn about the body
  - Skin
  - Bones
  - Muscles
- Different parts of the body require different skills
  - Nose
  - Hands
  - Legs
- Who do you want performing your surgery?

Why Do You Have to Take This Course?

- You want to be a race car driver
  - You all know how to drive
- To be successful you don’t just drive
  - You must “be in touch with your vehicle”
- You have to learn about the vehicle
  - Engine
  - Suspension
  - Tires
- Is it drag racing, monster trucks, NASCAR, endurance
  - Different cars
  - Different style of driving
- Who is going to win the Indy 500, 16 year old or Jeff Gordon?

Why Do You Have to Take This Course?

- You want to be a Computer Scientist
  - You know how to program (CPS 6, 100, 108)
- 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
  - 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 Five Classic Components of a Computer

- Control
- Memory
- Input
- Output
What is Computer Architecture?

- Coordination of levels of abstraction

Software

Interface Between HW and SW

Instruction Set Architecture, Memory, I/O

Hardware

- Under a set of rapidly changing Forces

Forces on Computer Architecture

Technology

Programming Languages
Compilers

Applications

Computer Architecture

Operating Systems

History

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 Neumann
  - Proposed Stored Program Computer
  - Properties of Today's computers
- 1949 Maurice Wilkes
  - EDSAC (Electronic Delay Storage Automatic Calculator)

Commercial Computers

<table>
<thead>
<tr>
<th>Year</th>
<th>Name</th>
<th>Size (cu. ft.)</th>
<th>Adds/sec</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1951</td>
<td>UNIVAC I</td>
<td>1000</td>
<td>1,000</td>
<td>$1,000,000</td>
</tr>
<tr>
<td>1964</td>
<td>IBM S/360 Model 50</td>
<td>60</td>
<td>500,000</td>
<td>$1,000,000</td>
</tr>
<tr>
<td>1965</td>
<td>PDP-8</td>
<td>8</td>
<td>330,000</td>
<td>$16,000</td>
</tr>
<tr>
<td>1976</td>
<td>Cray-1</td>
<td>58</td>
<td>166,000,000</td>
<td>$4,000,000</td>
</tr>
<tr>
<td>1981</td>
<td>IBM PC</td>
<td>1</td>
<td>240,000</td>
<td>$3,000</td>
</tr>
<tr>
<td>1991</td>
<td>HP 9000 / model 750</td>
<td>2</td>
<td>50,000,000</td>
<td>$7,4000</td>
</tr>
<tr>
<td>1996</td>
<td>Intel Pyro PC</td>
<td>2</td>
<td>400,000,000</td>
<td>$4,4000</td>
</tr>
<tr>
<td>2005</td>
<td>Intel Pentium4</td>
<td>0.25-2</td>
<td>4,000,000,000</td>
<td>&lt; $1,000</td>
</tr>
<tr>
<td>2007</td>
<td>Intel Core2Duo</td>
<td>0.25-2</td>
<td>8,000,000,000</td>
<td>$3.00 - $1,000</td>
</tr>
</tbody>
</table>

Microprocessor Trends

<table>
<thead>
<tr>
<th>Year</th>
<th>Transistor Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1971</td>
<td>1,000</td>
</tr>
<tr>
<td>1974</td>
<td>10,000</td>
</tr>
<tr>
<td>1978</td>
<td>100,000</td>
</tr>
<tr>
<td>1980</td>
<td>1,000,000</td>
</tr>
<tr>
<td>1985</td>
<td>10,000,000</td>
</tr>
<tr>
<td>1993</td>
<td>100,000,000</td>
</tr>
<tr>
<td>1999</td>
<td>1,000,000,000</td>
</tr>
<tr>
<td>2003</td>
<td>10,000,000,000</td>
</tr>
<tr>
<td>2008</td>
<td>100,000,000,000</td>
</tr>
</tbody>
</table>
Other Technologies

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

What is common among all these technologies?

Levels of Representation

<table>
<thead>
<tr>
<th>High Level Language Program</th>
<th>Assembly Language Program</th>
<th>Machine Language Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>temp = v[k];</td>
<td>hw $15, 0($2)</td>
<td>0100 0101 0110 0111 0110 0100 0101 0100</td>
</tr>
<tr>
<td>v[k] = v[k+1];</td>
<td>hw $16, 4($2)</td>
<td>0110 0111 0110 0111 0110 0100 0100 0101</td>
</tr>
<tr>
<td>v[k+1] = temp;</td>
<td>sw $16, 0($2)</td>
<td>0110 0111 0110 0111 0110 0100 0100 0101</td>
</tr>
</tbody>
</table>

Machine Interpretation

Control Signal Specification

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
  - NiosII programming.
- Digital Logic
  - Digital Gates and Boolean Algebra
  - Arithmetic and Logic circuits, Finite State Machines
  - You will design and program a simple processor in this class

Course Outline (continue):

- 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
  - Multiprocessors

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
Representations (Group task)

• Form partners
• Using only the three symbols @ # $ specify a representation for the following:
  - All integers from 0 to 10
  - Commands to 1) walk, 2) turn, 3) sit, 4) raise right arm, 5) raise left arm
• Using only your representation write down series of commands & integers (if appropriate, e.g., raise left arm-3, turn-2)
  - Must have at least 5 commands

What You Know Today

C++

```cpp
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

```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();
...
```

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\times10^1 + 1\times10^0
  - 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
    \[ 1110_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 1011_2 \]
    \[ 1110 = 8 + 2 + 1 \]
- 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: \(1200_2 = 0000010010110000\)
    - Octal numbers use 8 digits: \{0 - 7\}
      - Example: \(1200_8 = 04230\)
    - Hexadecimal numbers use 16 digits: \{0-9, A-F\}
      - Example: \(1200_8 = 04B0_8 = 0x04B0\)
      - 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.
- \(2^3 = 8\)

### Example:

<table>
<thead>
<tr>
<th>Bin</th>
<th>Oct</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>1</td>
</tr>
<tr>
<td>001</td>
<td>2</td>
</tr>
<tr>
<td>010</td>
<td>3</td>
</tr>
<tr>
<td>011</td>
<td>4</td>
</tr>
<tr>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>101</td>
<td>6</td>
</tr>
<tr>
<td>110</td>
<td>7</td>
</tr>
</tbody>
</table>

Binary and Hex

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

<table>
<thead>
<tr>
<th>Hex</th>
<th>Bin</th>
<th>Hex</th>
<th>Bin</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>8</td>
<td>1000</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>9</td>
<td>1001</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>A</td>
<td>1010</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>B</td>
<td>1011</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>C</td>
<td>1100</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>D</td>
<td>1101</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>E</td>
<td>1110</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>F</td>
<td>1111</td>
</tr>
</tbody>
</table>

### Example:

\[
\begin{align*}
1100 & \quad 0010 \\
0110 & \quad 0111 \\
0100 & \quad 1101 \\
& \quad 0101
\end{align*}
\]

\[
\text{C} \quad 2 \quad 6 \quad 7 \quad 4 \quad F \quad D \quad 5_{16}
\]

Admin

- Read Chapter 1, Ch 3.1-3.2 pgs 160-168, 3.6 pgs 189-197
- Optional: Brief History of Computers
- C tutorial [http://computer.howstuffworks.com/c.htm](http://computer.howstuffworks.com/c.htm)
- Homework #1 Assigned due Jan 24.

Next

- Start in on abstractions: data representation