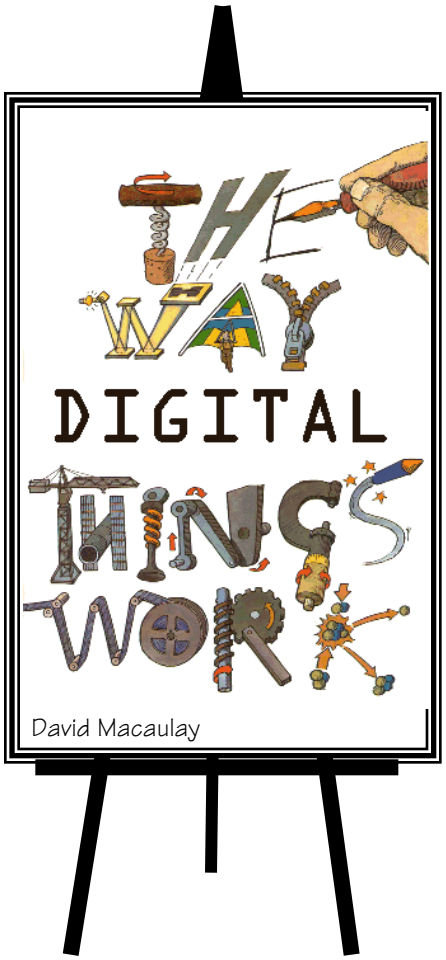
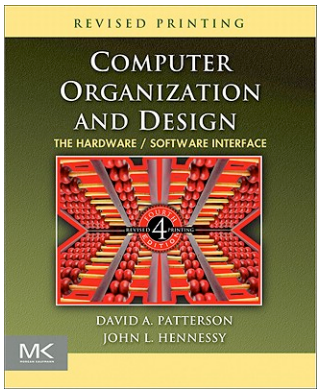
Welcome to Comp 411!



1) Course Mechanics
2) Course Objectives

Whos and Whats…



Lectures: Leonard McMillan (SN-311)

Office Hours Tu 2-3

TA: TBA

Book: Patterson & Hennessy

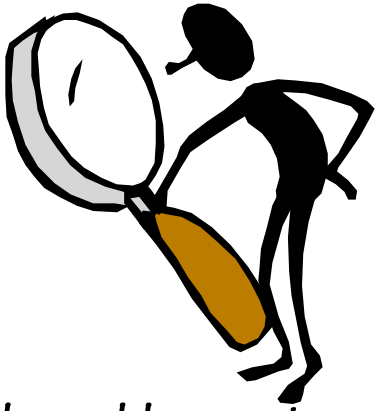
Computer Organization & Design

5th Edition, ISBN: 0-12374-750-3

(However, you won't need it until next week)

Course Mechanics

Grading:



Best 5 of 6 problem sets	25%
Best 9 of 10 Labs	25%
2 Quizzes	30%
Final Exam	20%

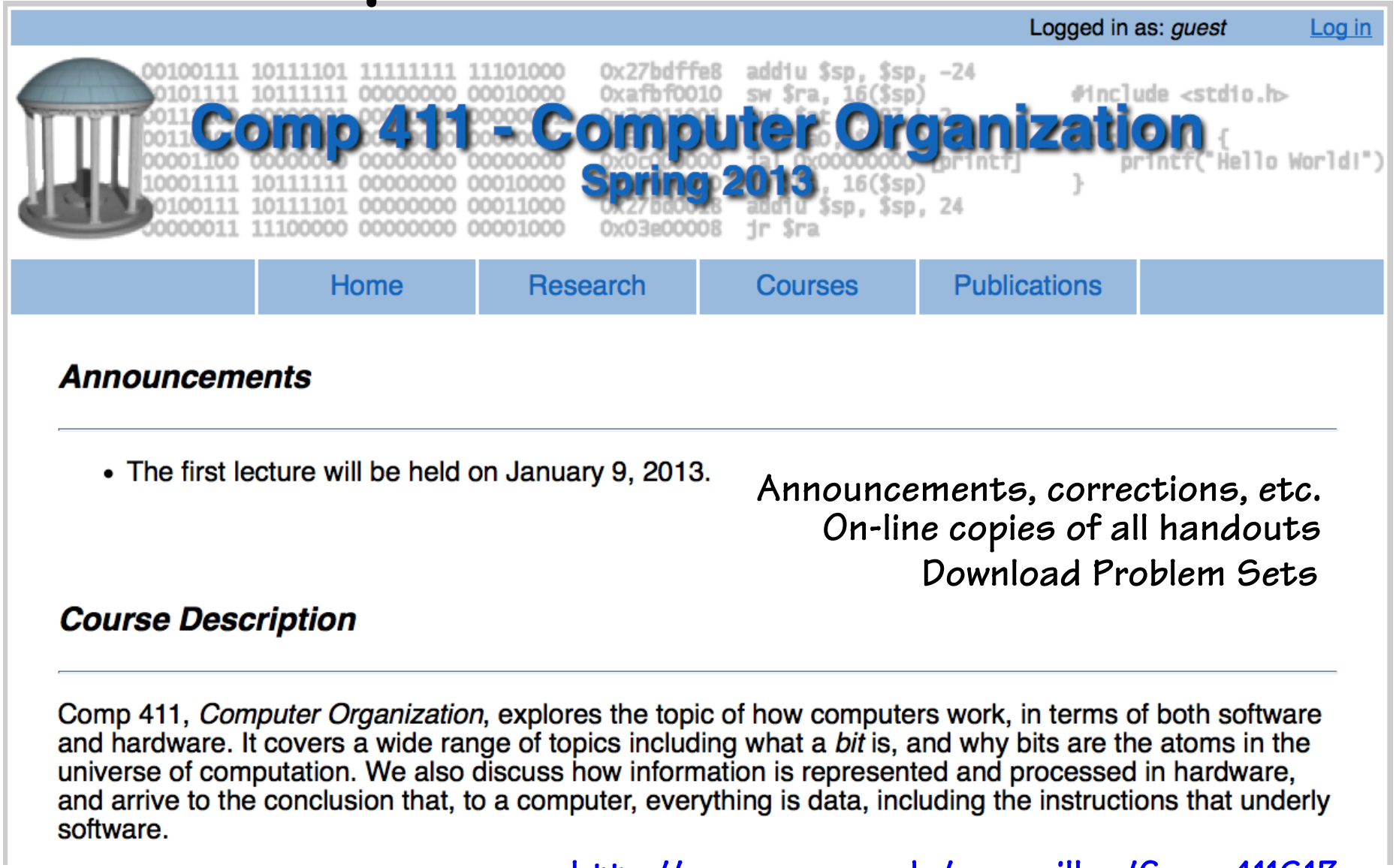
You will have at least two weeks to complete each problem set. Late problem sets will not be accepted, but the lowest problem-set score will be dropped.

Friday Labs are mandatory, and will meet on most Fridays, grade is based on completing a "lab check list."

Quizzes are multiple choice.

I will attempt to make Lecture Notes, Problem Sets, and other course materials available on the web before class on the day they are given.

Comp 411: Course Website



Announcements, corrections, etc.
On-line copies of all handouts
Download Problem Sets

http://www.cs.unc.edu/~mcmillan/Comp411S13

Goal 1: Demystify Computers

Strangely, most people (even some computer scientists I know) are afraid of computers.

We are only afraid of things we do not understand!

I do not fear computers. I fear the lack of them.

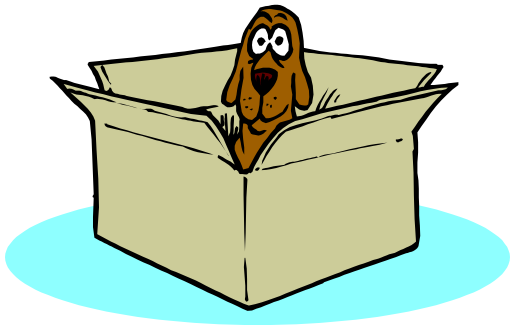
- Isaac Asimov (1920 - 1992)

Fear is the main source of superstition, and one of the main sources of cruelty. To conquer fear is the beginning of wisdom.

- Bertrand Russell (1872 – 1970)

Goal 2: Power of Abstraction

Define a function, develop a robust implementation, and then put a box around it.



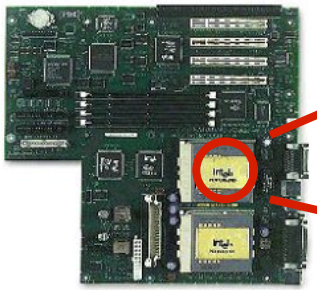
Abstraction enables us to create unfathomable systems (including computer hardware and software).

Why do we need ABSTRACTION…

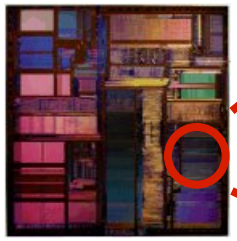
Imagine a billion --- 1,000,000,000

Understanding systems with >1G components

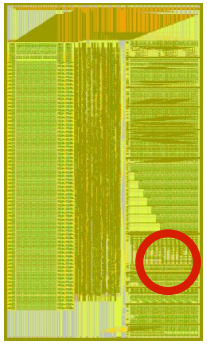
Personal Computer:
Hardware & Software



Circuit Board:
≈8 / system
1-2G devices

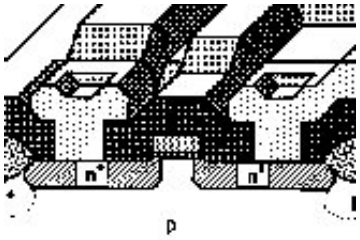


Integrated Circuit:
≈8-16 / PCB
.25M-16M devices

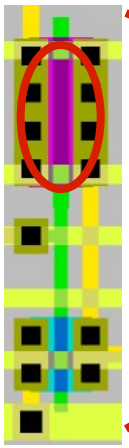


Module:
≈8-16 / IC
100K devices

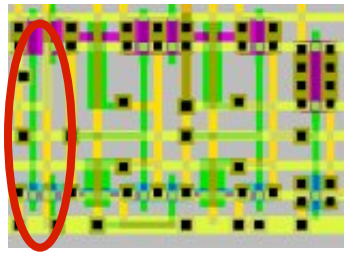
MOSFET



Scheme for
representing
information



Gate:
≈2-16 / Cell
8 devices



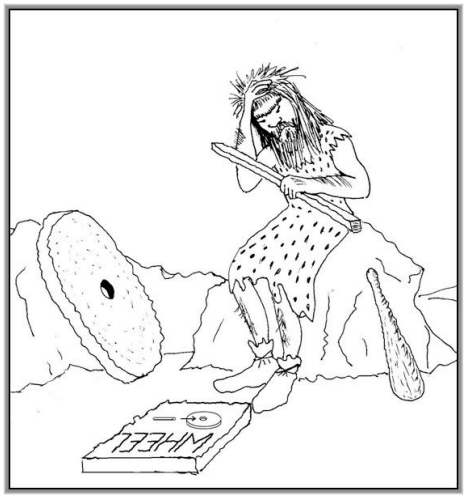
Cell:
≈1K-10K / Module
16-64 devices

By now, you all know how to program

```
int f(x) {
    int r;
    int odd = 1;
    for (r = 0; x >= odd; r++) {
        x -= odd;
        odd += 2;
    }
    return r;
}
```

What *does* this function *do*?

What does the computer do?



Problems with self-assembly are nothing new

```
int f(x) {
    int r;
    int odd = 1;
    for (r = 0; x >= odd; r++) {
        x -= odd;
        odd += 2;
    }
    return r;
}
```

What *does a computer do* with a program?

Where we are going…

How is data represented, stored, and manipulated in a computer?

What basic operations does a computer use?

What does mean to "compute"?

Are there limits to what can be computed?

Why are computers so fast?

What am I asking a computer to do when I give it a program to execute?

How are programs translated into computer instructions?

Why are some programs faster than others that perform the same function?

Summary

411 answers the following questions:

- How is information represented, stored, and manipulated by a computer?
- What does a computer really doing with my program?
- How to design, build, and manage large systems?

411 logistics:

- M-W: lectures and discussions
- F: ~2-hour Lab (starting next week 1/18)

Next time: What's a bit?