Course Overview
Policies, Grading & Outline

CIT 593
Fall 2008

Staff

- **Instructor: Diana Palsetia**
  - Email: palsetia@cis.upenn.edu
  - Location: Moore 174
  - Office hours: TBA
  - course web: [http://www.seas.upenn.edu/~cit593](http://www.seas.upenn.edu/~cit593)

- **TA: Ganggang HU**
  - Email: ganggang@seas.upenn.edu
  - Office location and hours: TBA

About CIT 593-95 in General

- How computers work
- How they are organized internally
- What are the design tradeoffs
- How design affects programming and applications
  - How to fix computers
  - How to build myself one real cheap

Computer Organization

- Software
  - Application Program
  - Algorithms
  - Language

- Hardware
  - Instruction Set Architecture (and I/O Interfaces)
  - Microarchitecture
  - Circuits
  - Devices
Why take CIT 593-95 sequence?

- Automobile – Combustion Engine Analogy
  - One does not have to understand the internal combustion engine to drive an automobile
  - Knowing mechanics can enhance the life of the engine and to improve its performance.

- The same is equally true of computer architecture & organization
  - Knowledge of the computer systems can improve the performance of software if the software is written to exploit the underlying hardware.

Why take CIT 593-95 sequence?

- Understand how computer represents and processes information
  - Understand some of high-level ideologies such as data type, subroutine linkage, memory allocation

- Understand what it takes to write machine dependent software
  - E.g. compilers, operating systems, and device drivers

- Interested in high performance studies
  - E.g. how to arrange/design programs to gain maximum performance from a given machine architecture & organization

- Benchmarking
  - E.g. compare computer architectures for a purchase decision as a member of the IT computer purchasing team.

Overview of Topics CIT 593-95

- Hardware
  - Data Representation, von Neumann machine model, transistors, gates, digital logic

- Instruction Set Architecture (ISA)
  - Basic Operations that machine can perform

- Assembly language
  - Structured programming which is specific to a machine

- C programming
  - Language features + relationship to assembly language

- Operating System
  - A management program that allows application programs to interface the machine

- Input/Output
  - How does the computer communicate with the outside word?
  - How does simultaneous interaction with the computer work?

- Performance
  - Caches, Virtual Memory, Pipelining and Alternate Architectures

Why learn Hardware?

- To learn the foundations
  - How does a problem stated in programming language is actually solved by electrons moving around inside the electronics of the computer

- To know its limitations
  - What makes a computer system tick before you can attempt to optimize programs?
Why learn Assembly?
- Computers can only understand voltages and abstractly we represent them in binary notation (0 or 1)
  - What computer understand is called as machine language
- Assembly is the lowest form of human-readable notation for the machine language
  - Better able to understand & appreciate a high-level language

Why Learn C Language?
- What is C?
  - High-level language
  - In between assembly and Java/VB/C#
- Very common
  - Operating systems and even general applications
  - Foundation for C++/Java/C#
  - Still widely used in conjunction with C++ by many companies
  - Assembly-to-C migration for embedded applications

Why Learn about OS & I/O
- You use these in day to day when you interact with the computer
- Get perspective on what is happening
  - E.g.1 When interacting with multiple programs
  - E.g.2 Blue Screen of Death

Course Topics for CIT 593
- Bits and Bytes
  - Data Representation
- C Programming in Unix/Linux Environment
  - Language Basics to write application programs
- Instruction Set Architecture
  - Formal specification of all the functions a particular machine can carry out
- Assembly Language Programming
  - How do we use processor instructions to implement algorithms?
  - How do we implement high-level programming constructs?
  - How do we write modular, reusable code (subroutines), OOP
- I/O, Traps, and Interrupts
  - How does processor communicate with outside world?
Transition to CIT 595

- Implementation
  - How are instructions actually carried out using transistors (abstraction logic gates) ?
  - Are there more than one way of building a computer?
    - Micro-architecture
- Memory Hierarchy and Performance Impact
  - Different kinds of computer memory
- Overview on the role of the Operating System
- Some Advanced Topics in Computer Architecture

Textbooks

- Required
  - Introduction to Computing Systems from bits & gates to C & beyond
    - Author: Yale N. Patt and Sanjay J. Patel
    - ISBN: 0072467509 (2nd Ed)
- Recommended
  - Practical C Programming - by Steve Oualline
  - Online text thru Safari Online via Penn Library

Online Resources

- Plenty of online resources
- See course website
  - http://www.cis.upenn.edu/~palsetia/technicalFAQ.html

Grading

- Course grades will be on class performance
  - 90% or above is not necessarily an A
- Assignments (45% of grade)
  - Written and Programming Assignments
- Two exams (45% of grade)
  - Midterm (20%) & Final (25%)
  - Exams will be closed book
- Participation (10 % of grade)
  - Participation means coming to lab (most Thursday lectures will be hands on activities) on time and working on course matter
  - You may be asked to submit some of them for homework.

Dates

- Midterm exam: Tentatively 21st Oct (during class time)
- Final exam: According to final week schedule
- Assignments (regularly check website)
  - Programming/Homework: As they are posted
### Assignment Grading

- If submitted later than due date
  - 10% per day for five days and then no credit
  - Weekend days count as late days

- Program grading is generally
  - Good documentation 20%
  - Sound structure 20%
  - Correct output or results 60%

- Turning assignments
  - Submitted via Digital Dropbox via Blackboard
  - Instructions on how to submit are provided in the syllabus (see course website)

### Academic Honesty

- **Appropriate**
  - Discuss the assignments with one another to clear doubts
  - Help others debug their work

- **Inappropriate**
  - Work together unless stated otherwise
  - Copy another's code, or allow your code to be copied
  - Lend your code to someone else, or allow them to copy it
  - Use any code from a book or the web without my permission

- **Penalty for first offence:**
  - You will be reported to the Office of Student Conduct
  - If you may have accidentally broken a rule, talk to Instructor or TA immediately

### How much time to put into the course?

- On average, at least **10 hours** per week, assuming:
  - You take 3-4 courses
  - You are a full-time student
  - Your studies occupy a 40-hour work week

- 3 hours in class + office hours
- 4 hours on projects and practice
- 3 hours on readings

### Computing Facilities at Penn

- **Penn Eng.**:
  - [http://www.seas.upenn.edu/cets/answers/](http://www.seas.upenn.edu/cets/answers/)
  - Information on accounts, labs, software

- **Transferring Files your PC to Penn acc**
  - Mac Software: Fetch, DataComet-Secure X
  - Windows: Filezella, SecureCRT
  - [http://www.upenn.edu/computing/product/](http://www.upenn.edu/computing/product/)
Questions?