Outline

- Enrollment & Attendance
- Course Materials & Content
- Course Structure & Grading
- A Few Basic Principles of Digital Design

Enrollment

- If you are enrolled and plan to take the course you must attend your lab section next week, if not you will be dropped from the class roster. No exceptions!

- If you are on the waitlist - we have room for you, however, you must:
  1. Have taken the prerequisites - CS61C & EECS40.
  2. Attend lectures and do the homework, the first two weeks.
  3. In the second week of classes, go to the lab section in which you wish to enroll. Give the TA your name and student ID. If the lab section is full (the TA will tell you if so), you must find a different section.
  4. Later, we will process the waitlist based on these requests, and lab section openings.
  5. Note: If you are not on the waitlist, you will not be considered for enrollment.

- No lab (or discussion) sections this week. Yes, lab lecture Friday
Attendance

- Attend regular lectures and ask questions.
- Attend weekly lab lecture (Friday). Lab lecture will be webcast and recorded on the website.
- Attend your lab section. You must stick with the same lab section all semester.
  - Lab exercises will be done individually; project with a partner.
  - We will put together a lab section exchange in a few weeks to help you move to a different section.
- Attend any discussion section. You may attend any discussion section that you want regardless of which one you are enrolled in.
- The instructor and TAs hold regular office hours (see class webpage). Please take advantage of this opportunity!

Lab Lecture

- Held in the lab - 125 Cory, so we can do demonstrations.
- This is an important part of the course. You get background and practical information regarding the lab exercises and project checkpoints.
- Also, we will have a mandatory short quiz at the beginning of each lab lecture (we might move some of these to Thursday class).
  - Quiz will be based on one of the weekly homework problems.
  - Two quiz scores for each student will be dropped at the end of the semester, so you can miss two quizzes (save this option for important dates – like job interview trips, etc.)
- Telebears say Friday 1-2, Website says Friday 2-3. Which one is it?

Course Materials

- Textbook: Katz & Borriello, Publisher: Prentice Hall
  Other useful books:
  (on reserve in Eng Library)

- Class notes, homework & lab assignments, solutions, and other documentation will be available on the class webpage:
  http://www-inst.eecs.berkeley.edu/~cs150/index.html
  - Check the class webpage and newsgroup often!
  - You are responsible for checking the class webpage at least once every 24 hours.

Course Content

Components and Design Techniques for Digital Systems
more specifically
Synchronous Digital Hardware Systems

- Synchronous: “Clocked” – all changes in the system are controlled by a global clock and happen at the same time (not asynchronous).
- Digital: All inputs/outputs and internal values (signals) take on discrete values (not analog).
  - Example digital representation: music waveform
  - A series of numbers is used to represent the waveform, rather than a voltage or current, as in analog systems.
Course Content

- Hardware Architectures
- Arithmetic units, controllers
- Memory elements, logic gates, busses
- Transistor-level circuits
- Transistors, wires
  - Not a course on transistor physics and transistor circuits. Although, we will look at these to better understand the primitive elements for digital circuits.
  - Not a course on computer architecture or the architecture of other systems. Although we will look at these as examples.

Course Content

- IC processing
- Transistor Physics
- Devices
- Circuits
- EE 40
- Gates
- FlipFlops
- HDL
- Machine Organization
- Instruction Set Arch

Deep Digital Design Experience

- Fundamentals of Boolean Logic
- Synchronous Circuits
- Finite State Machines
- Timing & Clocking
- Device Technology & Implications
- Controller Design
- Arithmetic Units
- Encoding, Framing
- Testing, Debugging
- Hardware Architecture
- Hardware Design Language (HDL)
- Design Flow (CAD)

Course Evolution

- Final project circa 1980:
  - Example project: pong game with buttons for paddle and LEDs for output.
  - Few 10’s of logic gates
  - Gates hand-wired together on “bread-board” (protoboard).
  - No computer-aided design tools
  - Debugged with oscilloscope and logic analyzer

Course Evolution

- Final project circa 1995:
  - Example project: MIDI music synthesizer
  - Few 1000’s of logic gates
  - Gates wired together internally on field programmable gate array (FPGA) development board with some external components.
  - Circuit designed “by-hand”, computer-aided design tools to help map the design to the hardware.
  - Debugged with circuit simulation, oscilloscope and logic analyzer
Moore’s Law – 2x stuff per 1-2 yr

Course Evolution

• Final project circa 2000:
  – Example project: eTV - streaming video broadcast over Ethernet, student project decodes and displays video
  – Few 10,000’s of logic gates
  – Gates wired together internally on FPGA development board and communicate with standard external components.
  – Circuit designed with logic-synthesis tools, computer-aided design tools to help map the design to the hardware.
  – Debugged with circuit simulation, logic analyzer, and in-system debugging tools.

• Final project circa 2005:
  – Example project: Multi-user networked Quake in hardware
  – 10Million logic gates
  – Circuit designed with logic-synthesis tools, and library of high-level blocks.
  – Debugged with in-system debugging tools.

But wait!

Course Evolution

• Final project it getting too complex.
• Design mythology getting too “high-level”. Losing touch with fundamentals of logic-design.
• Our Plan:
  – Emphasize fundamental design concepts, de-emphasize advanced techniques.
  – Stay with current development platform (Calinx board).
  – Somewhat scale-down scope of project.
• New course to cover advanced design techniques - CS151:
  – First offering will be Fall 2005.
  – Covers advanced digital system design
  – Projects will combine processor cores with dedicated hardware circuits.
Final Project: Streaming-Audio Receiver

- Audio streams (music) will be broadcast over the local area network (LAN) in the lab.
- Networked attached devices, pick off one audio stream and play it to headphones or speakers.
- Optional display of "out of band content", song titles, etc., on mini-LCD Display.
- Everyone (working in groups of 2) will design, implement, debug, and demo a networked audio receiver.

Course Grading

- Weekly homework based on reading and lectures.
  - graded on effort only,
  - out at the end of each week, due before next week lab lecture.
- Weekly quiz closely related to one of the homework problems. Given at the beginning of the lab lecture or during Thursday lecture (will announce which one).
  - Most of "HW/quiz" grade points based on quiz grades.
  - Lab exercises for weeks 2-6, followed by project checkpoints and final checkoff.
  - Labs and checkpoints due before your next lab session.
- Three exams of approximately equal weight - held in class.
  - No Final.

Tips on How to Get a Good Grade

The lecture material is not the most challenging part of the course.

- You should be able to understand everything as we go along.
- Do not fall behind in lecture and tell yourself you “will figure it out later from the notes or book”.
- Notes will be online before the lecture (usually the night before). Look at them before class. Do assigned reading (only the required sections).
- Ask questions in class if you don’t understand. If you are not getting it then probably nobody is. Ask questions.
- The exams will test your depth of knowledge. You need to understand the material well enough to apply it in new situations (beyond the homework). The homework is a starting point, not the ending point.
You need to do well on the project to get a good course grade.

- Take the labs very seriously. They are an integral part of the course.
- Choose your partner carefully. Your best friend may not be the best choice.
- Most important (this comes from 30 years of hardware design experience):
  - Be well organized and neat.
  - Add complexity a little bit at a time - always have a working design.
  - Don’t be afraid to throw away your design and start fresh.
Course Structure

*A week in the life of an EECS150 student*

Monday (for example):
- Discussion section 1 hours
Tuesday: Lecture 12-1 1.5
Wednesday (for example):
- Lab section 3
Thursday: Lecture 12-1 1.5
Friday: Lab Lecture 1
Reading book, reviewing notes 3
Homework 4
**TOTAL** 15 hours/week

Extra time in lab to finish weekly assignments.
Significantly more time once project starts.

Cheating

- Any act that gives you unfair advantage at the expense of another classmate.
- Examples:
  - copying on exams, homework,
  - copying design data,
  - modifying class CAD software,
  - modifying or intentionally damaging lab equipment.
- If you ever have a question about what will be considered cheating, please ask me.
- We have software that automatically compares your submitted work to others.
- If we detect you are involved in cheating you will be:
  - Get a 0 for that portion of the course, and a letter of reprimand to your student file.
  - Egregious and multiple instances of cheating will be turned over to the University Student Conduct Office for further action.

Example Digital Systems

- **Digital Computer**
  - Usually design to maximize performance. "Optimized for speed"
- **Handheld Calculator**
  - Usually designed to minimize cost. "Optimized for low cost"
  - Of course, low cost comes at the expense of speed.

Example Digital Systems

- **Digital Watch**
  - Designed to minimize power. Single battery must last for years.
  - Low power operation comes at the expense of:
    - lower speed
    - higher cost
Basic Design Tradeoffs

- You can improve one at the expense of worsening one or both of the others.
- These tradeoffs exist at every level in the system design - every sub-piece and component.
- Design Specification -
  - Functional Description,
  - Performance, cost, power constraints.
- As a designer you must make the tradeoffs necessary to achieve the function within the constraints.

Design Representation

- Helps control complexity -
  - by hiding details and reducing the total number of things to handle at any time.
- Modularizes the design -
  - divide and conquer
  - simplifies implementation and debugging
- Top-Down Design
  - Starts at the top (root) and works down by successive refinement.
- Bottom-up Design
  - Starts at the leaves & puts pieces together to build up the design.
- Which is better?
  - In practice both are needed & used.
    - Need top-down divide and conquer to handle the complexity.
    - Need bottom-up because in a well designed system, the structure is influence by what primitives are available.

Hierarchy in Designs

Digital Design

- Given a functional description and performance, cost, & power constraints, come up with an implementation using a set of primitives.
- How do we learn how to do this?
  1. Learn about the primitives and how to generate them.
  2. Learn about design representation.
  3. Learn formal methods to optimally manipulate the representations.
  4. Look at design examples.
  5. Use trial and error - CAD tools and prototyping.
- Digital design is in some ways more an art than a science. The creative spirit is critical in combining primitive elements & other components in new ways to achieve a desired function.
- However, unlike art, we have objective measures of a design: