Today's topics

- Reason for late start
- Brief course overview
- Majors
- Administrative stuff
  - The fun stuff starts on Monday!

Reason for late start

- For the curious, student invented method of doubling usable memory on smartphones, PDAs, etc.,
- No hardware change
- No application change
- Little or no performance or power penalty
- Implemented as linux kernel module
- Plan to release for free personal use when patent clears
  - Need to protect industrial partner commercial use

What's your major?

Computer geek already?

- Good!
- You'll still probably see a lot of new things in this course
- Go ahead and ask questions that push beyond the basic material
- If you want to go beyond the normal labs, I'll be happy to make suggestions
- ECE 203 should lay the foundations for logic design and understanding the connections between electrons and software
Not a computer geek yet? Good!

• You’re going to be working with computers in almost any field
• Understanding how they work at the lowest levels and knowing how to build them will put you ahead your peers
• If you’re not a computer geek yet, sit in the front of the classroom and ask questions!
  – It’s the best way to keep the course’s pace sane

Backgrounds

• Different backgrounds
• ECE 203 can be a hard course
• However, if you work hard, I’m totally confident that you will learn how to build useful computers
  – TAs and I will help
• In the past, many Materials Science, BME, and IEMS did absolutely amazing work

Rules

• If something in lecture doesn’t make sense, please ask
  – If it doesn’t make sense to you, others have the same question!
• Do you feel like there is a gap in your background, e.g., forgot about resistance and capacitance?
  – It’s O.K. I have handouts and office hours to help but don’t fall behind!
• You’re paying a huge amount of money for this
• I expect a lot
• However, I’ll do whatever I can to make sure you get as much out of this course as you put in

Core course goal

By the end of this course, I want every one of you to be capable of designing and building simple but useful computer systems from integrated circuits, wires, and assembly language instructions
In fact, it’s a requirement

Administrative stuff

• How to get lab supplies
• How to subscribe to mailing list
• Some good references
• Decide grading policies
• Plan office hours
• Course overview (if time permits)

Mailing list

• Please subscribe to the ECE203 mailing list by sending an email to listserv@listserv.it.northwestern.edu with no subject and a body of “SUBSCRIBE ECE203 Firstname Lastname”.
• I encourage you to use the mailing list for discussion.
• Please don’t hesitate to use the list. If anybody thinks the traffic is too high, I’ll set up separate “Announce” and “Discuss” lists.
• In general, if I answer somebody’s question via email, I will also post the answer to the mailing list (without including the person’s name).
• Do this today! The mailing list is an important and required component of the course – used for announcements.

References

Grading

• I'm open to reasonable options
• Class needs to agree on scheme within the first week
• Once an option is chosen, it'll be my job to stick to it
• One possible option follows

Possible grading scheme

• 15% homeworks
• 35% labs
• 20% midterm exam
• 30% final exam

Late homework assignments

• After the class, on the due date: -5%
• After that, 10% per day penalty
• Three or more working days late: No credit
  – I'll hand out solutions

Late lab assignments

• Late lab verifications will be done at the discretion of the TAs
• In other words, although this will sometimes be possible, I'm not going to force the TA to skip their classes, research work, or meals to hold extra lab verification hours
• Late lab checks (without prior approval): -20%
• Three or more working days late: No credit

When to start labs

• The TAs spend a huge amount of time checking labs
• Having them do lab checks outside of the scheduled hours makes it difficult to keep up in their own classes and research
• Start labs early to see if you have questions
• The TAs and I will be happy to help
• Will need time to finish after pointed in right direction

Labs

• Open labs
• Tech EG27
• The TAs and I may leave a note and go from our offices to EG27 during office hours to answer lab questions
• You will need to sign up for a lab time slot

Lab check times

• Labs will be assigned on Fridays
• Can lab checks be held on Wednesday and Thursday throughout the day?
• Thursday and Friday would give you more time but might make it more difficult for those checked on Friday to attend the help sessions
• First lab much quicker than others
• Need to get go to get kit by Monday, though

Office hours options

1. Tuesday/Thursday throughout day
2. Monday/Thursday throughout day
3. Every day in morning
4. Every day in evening
Course overview

• Know what is computer engineering is
• Know some reasons to learn computer engineering
• Understand course goals
• Know which future courses ECE 203 can prepare you for
• Know course topics
• Start learning basic logic definitions

What is computer engineering?

• Design and implementation of computer systems
• Hardware and software design
• Related to electrical engineering and computer science with an emphasis on digital circuits
• The best computer engineers are also good at electrical engineers and computer science
• Knowing fundamentals helps in fields where computers are used

25

What is computer engineering?

• You need something solid to stand on
• Applications make more sense if you understand programming
• Programming makes more sense if you understand processors
• Processor make more sense if you understand logic design
• Logic design makes more sense if you understand circuits and discrete math
• Circuits make more sense if you understand transistors
• Every understanding rests on others
• Computer engineering requires understanding the many levels and the ways they fit together

26

Why computer engineering

• Why are you taking this class?
• What do you want to learn?
• What kind of background do you have?
  – When you see something cool do you reach for a screwdriver?
  – Who was electrocuted as a young child trying to figure out how something works?
  – Who has written code?
  – Who has designed something complicated for the fun of it?

27

Why computer engineering? Fun

• Computers are almost magical
  – You’ll learn how they work and how to build new ones
• You’ll learn (discrete) math, semiconductor physics, and the theory of algorithms
• You’ll be able to use your knowledge creatively

28

Why computer engineering? Flexible

Learn hardware and software design, can move in either direction
• Embedded system design
• Computer architecture
• VLSI design
• Digital circuit design
• Software engineering
• Algorithm design
• Information technology
If you finish a Ph.D., many other doors also open

29

Why computer engineering? Money

• According to a 2004 CNN survey, it’s the most lucrative degree, and it’s improving
• Average starting salary: $53,117 with a 0.7% increase from 2003–2004
• Compare with business administration ($37,368) and psychology ($25,032)
• Of course, this is not a good reason to pick a major
• Do what you love!
  – … but if you love computer engineering, the financial stuff might make it easier to justify to your relatives

30
Future courses

• Advanced digital logic design
• Computer architecture
• Design and analysis of algorithms
• Fundamentals of computer system software
• Introduction to computer networks

Future courses

• Introduction to VLSI CAD
• Introduction to mechatronics
• Microprocessor system design
• Programming for computer engineers
• VLSI systems design

Course topics in context

• Logic gates
  – Basic units of digital logic design
• Truth tables
  – Simple Boolean function representation
• Boolean algebra
  – Another way of representing and manipulating Boolean functions
• Two-level logic forms

Course topics

• Logic minimization: Boolean algebra, Karnaugh maps, and Quine-McCluskey’s method (if time permits)
  – Reduce area, power consumption, or improve performance
• Hazards
• Implementation in CMOS
• Number systems: decimal, binary, octal, hex, and Gray codes
• Signed and unsigned numbers

Course topics

• Arithmetic circuits, decoders, encoders, and multiplexers
• Sequential logic: Latches, flip-flops
• Finite state machines
• Assembly language programming

Course topics

• Overview of compilation of higher-level languages
• Computer organization
• Microcontrollers

Software

• Easy to change and design
• Usually has low performance compared to hardware implementation
• High power consumption
• General-purpose processor
• Digital signal processor (DSP)
• Field programmable gate array (FPGA)
• Application specific integrated circuit (ASIC)

Hardware

• Usually difficult to design and implement compared to software
  – Hardware description languages can make this easier
• Necessary (all software runs on hardware)
• High performance
• Low power
Hardware/software rules of thumb

- If you can do it in software, do it in software
  - However, some things can’t be done in a sane way with software
- If you can’t do it in software but you can do it with an HDL, do it with an HDL
  - Sometimes the results of automation aren’t good enough
- If you’re tired, don’t do hardware implementation
  - Software design errors usually mean wasted time
  - Hardware design errors often mean fried chips

Embedded systems

- Special-purpose computers, computers within devices which are generally not seen to be computers
- Larger market than general-purpose computers by volume and monetary value
- Microcontrollers rule
- Cool application-specific optimizations
  - Power
  - Size
  - Reliability
  - Hard deadlines

Market

- How large is the semiconductor market?

Digital and analog signals

- Analog: Continuously varying signal
- Digital: Discrete values, assumed instantaneous transition
- In reality, digital assumption is approximation
- Use thresholds

Digital voltage regeneration

Voltage regeneration hides input variation

Boolean algebra

- The only values are 0 (or false) and 1 (or true)
- One can define operations/functions/gates
  - Boolean values as input and output
- A truth table enumerates output values for all input value combinations

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>AND</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ a \land b = a \cdot b \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>OR</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ a \lor b = a + b \]
### NOT

<table>
<thead>
<tr>
<th>a</th>
<th>( \bar{a} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \text{NOT} \ a = \bar{a} = \bar{\bar{a}} \]

### Combinational vs. sequential logic

- **Combinational logic**
  - Outputs a function of the current inputs, only
  - No feedback between inputs and outputs
  - Plain old combinational logic
- **Sequential logic**
  - Outputs depend on current state and (maybe) current inputs
  - Next state depends on current state and input
  - For implementable machines, there are a finite number of states
  - Synchronous
    - State changes upon clock event (transition) occurs
  - Asynchronous
    - State changes upon inputs change, subject to circuit delays

### Summary

- Brief overview
- Administrative stuff
- Introduction and definitions

### Reading assignment (for next class)

  - Sections 1.1, 2.1, and 2.2
  - http://www.writphotec.com/mano/Supplements
- Read these by the next class

### Computer geek culture references

- \( Z_{i+1} = Z_i^2 + K \)

### Computers enabled many inventions

- Simulation, automation, knowledge discovery
- In astrophysics, chemistry, biology, medicine, etc.