16.682 - Prototyping Avionics
Spring 2006

Instructor
Alvar Saenz-Otero

DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
Outline

• Class organization
  – Motivation, objectives

• (Re-) Introduction to Design
  – Requirements, Design Processes, Why “Prototype” now?

• Subject overview
  – Overview of EE concepts
  – Show & tell of software we will use

“The scientist seeks to understand what is; the engineer seeks to create what never was”

– Von Karman
Motivation

• Before

• Now

PCB ~$2-5K

~$50

~$500
Objectives

• Bottom line:
  – Learn to make printed circuit boards

• More in depth:
  – Identify the main components of an avionics prototype
  – Use schematic capture software to create detailed schematics
  – Use PCB layout programs to create avionics boards
    • Assembly avionics PCBs
  – Understand the test and debug processes for avionics
Class Organization

- “Prototype” class
  - Builds upon the “rapid prototyping” of mechanical hardware (by Oliver deWeck) given during IAP
  - First class to introduce prototyping of PCB’s at MIT Aero/Astro
- 6 unit class
  - 2-2-2
    - 2 hour of lecture practically every week
    - you are expected to spend two hours in lab during the PCB design phase
    - about 2 hours of homework during the first month, then documentation of project
Class Overview

- Two main sections of the class
  - 1) Electronics review
    - 4 weeks to review analog and digital electronics
    - Will reverse-engineer circuits and create some of our own
  - 2) PCB design
    - Schematic capture - how your circuit works, captured electronically
    - PCB layout - actually “drawing” how the printed circuit board looks like

<table>
<thead>
<tr>
<th>Week</th>
<th>Week 2</th>
<th>Week 3</th>
<th>Week 4</th>
<th>Week 5</th>
<th>Week 6</th>
<th>Week 7</th>
<th>Week 8</th>
<th>Week 9</th>
<th>Week 10</th>
<th>Week 11</th>
<th>Week 12</th>
<th>Week 13</th>
<th>Week 14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electronics Review</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schematics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCB's</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assembly and debug</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Grading & Deliverables

• Based on the two parts of the class
  – 4 problem sets, total 30%, of electronics review
    • Will directly cover the topics of that week
  – 50% will be the schematics and layout projects
    • Will incrementally build on the
  – 20% on documenting the projects
    • Creating a “data sheet” of the project and a final report

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homework</td>
<td>30%</td>
</tr>
<tr>
<td>Documentation: functional</td>
<td>10%</td>
</tr>
<tr>
<td>Schematic</td>
<td>25%</td>
</tr>
<tr>
<td>Layout</td>
<td>25%</td>
</tr>
<tr>
<td>Documentation: test, debug, results</td>
<td>10%</td>
</tr>
</tbody>
</table>
Resources

• Textbook
  – None required - we’ll learn to find and read data-sheets

• Laboratory
  – Main area: (through April)
    • All computers have schematic and layout software
      – Altium Designer
  – Assembly area: Gelb Lab (once our boards are made)
    • Assembly (soldering) and debugging of boards
  – Demo area: Space Systems Lab
    • Microscope use
    • Complex embedded systems demo
(Re)Introduction to Design

reproduced with permission, Oliver deWeck
Design: an iterative process

1. Define Requirements
2. Create/Choose Concept
3. Perform Design
4. Analyze System
5. Build Prototype
6. Test Prototype
7. Accept Final Design

easy to modify, unreliable, stand-alone

can modify, reliable, integrated

hard to modify, very reliable, final
Typical Design Phases

Requirements Definition
- General arrangement and performance
- Representative configurations
- General internal layout

Conceptual Design
- Conceptual baselines

Preliminary Design
- Selected baseline

Detailed Design
- Systems specifications
- Detailed subsystems
- Internal arrangements
- Process design

Production baseline

Production and support

Sophisticated Analysis
- Problem Decomposition
- Multidisciplinary optimization
Typical Design Phases

- Sophisticated Analysis
- Problem Decomposition
- Multidisciplinary optimization

Selected baseline

- Systems specifications
- Detailed subsystems
- Internal arrangements
- Process design

Detailed Design

Production baseline
Phased vs. Spiral PD Processes

Phased, Staged, or Waterfall PD Process
(dominant for over 30 years)

Spiral PD Process
(primarily used in software development)

- Process Design Questions:
  - How many spirals should be planned?
  - Which phases should be in each spiral?
  - When to conduct gate reviews?
Spiral PD Process

- Rapid Prototyping is typically associated with this process.
Basic Trade-offs in Product Development

- **Performance**
  - ability to do primary mission
- **Cost**
  - development, operation life cycle cost
- **Schedule**
  - time to first unit, production rate
- **Risk**
  - of technical and or financial failure

Where do the different phases of avionics design fit in these trade-offs?

Ref: Maier, Rechtin, “The Art of Systems Architecting”
Class Content
Remember 8.02?

- Analog electronics ultimately go back to the basic concepts of E&M:
  - Voltage
  - Current

- Power system design and analysis depends on E&M concepts:
  - Inductance
  - Capacitance
  - Watts

- We’ll review:
  - $V = IR$
  - $P = IV = I^2R$
  - $I = C\frac{dV}{dt}$
  - $V = L\frac{dl}{dt}$
Discrete Components

- Discrete components are some of the most basic parts of a circuit
  - Usually passive
  - Normally one component per package
  - Implement basic E&M

- Include:
  - Resistors, capacitors, inductors, and diodes
  - First and second order systems created with passive components
  - Emphasis on voltage dividers and low-pass filters

- Also some active elements: amplifiers
  - Assume an ideal amplifier
  - Review general equations to implement different types of common circuits
Transistors

• These days each microprocessor contains millions of transistors...

• But what does a single transistor do?
  – It's essential to understand what a single transistor does to understand better how many analog and digital circuit operate

• There are many types of transistors, most importantly
  – NPN/PNP
  – MOSFETs

• Transistors have two behaviors: transient and saturated
  – Analog circuits operate in the transient behavior
  – Digital circuits saturate transistors
  – We’ll emphasize the use of transistors in digital electronics
Power Components

• You have a single AAA 1.5V battery... how do you power a 5V microcontroller?
  – Remember that E&M has a lot to do with power: RLC circuits can create a "voltage pump"

• You have 120Vac and need to power a 5V micro-controller, how?
  – Remember discrete components: diodes and capacitors help you create a DC signal out of an AC signal
  – Remember E&M? Voltage transformers use E&M to swap voltage for current

• I have 5V and I need 2.5V, how do I do that?
  – Could use a resistor-based voltage divider, a linear regulator, or a switching regulator... which one do I choose?
The most basic block of a digital system is called a “gate”
- and, or, nor, xor
- Normally specified using “truth tables”

Gates are put together to create more complex parts
- multiplexers
- registers
- latches, and flip-flops

Tens, hundreds, even thousands of these make microprocessors and other parts

BUT individual blocks are still needed from time to time independently, so we need to know what their basic functions are
- One bit at a time
Digital Thinking

• Digital systems work with many bits at a time, so it’s important to understand how to work with many bits at a time:
  – Bit groupings: bit vs. nibble vs. byte vs. word
  – Notations: octal vs. decimal vs. hexadecimal
  – Operations: bitwise vs. byte/word operations
  – Data types: how do computers represent data

• It’s also important to understand how computers maintain the data:
  – Stacks and queues
  – Pointers
  – Memory operations
Processors

- Processors can be very simple (e.g. an 8-bit PIC) or extremely complex (e.g. a dual-core 65-bit Intel)
  - Embedded processors are usually considered for low-power consumption (which means low heat), so we’ll talk about simple processors in general

- General purpose processor architecture
  - Processing unit
  - Program memory and stack
  - Data memory
  - Peripherals
  - Interfaces

- To really learn processors you’ll have to take 6.115 for now
• You have a 32-bit processor and a temperature sensor that connects to a resistor, what do you do?
  – ADC: analog to digital conversion
  – What do you need to learn:
    • Sample rates (aliasing), discretization, bit-depth
• The processor calculates the motor needs to run at 40% of its maximum speed, how do you tell it to do it?
  – Multiple options:
    • Pulse-width modulation + amplifiers
    • DAC: digital to analog conversion + amplifiers
Schematics

• Start basic:
  – Single-page schematic with pre-existing components

• Add complexity:
  – Define your own parts that are not found in the included libraries

• More complexity:
  – Larger schematic that requires multiple pages

• Re-simplify?
  – Group repeated circuits into “multiple channels” schematics

• Finalize
  – Number all the components (automatically, of course) and check for errors
What is a printed circuit board?
  - All of you have seen one by now, but what are its parts?

You designed your circuit, but exactly which parts do you use?
  - Surface mount vs. through-hole components
    - Package definitions
  - Annular rings, clearance, and thermal relieves
  - Hole sizes, drills, and mounting holes

Routing a board
  - Manual routing: take care of noise, ground lines, power lines
  - Automatic routing: when to use it

Finishing the board
  - Error checking, plotting/printing, file output, manufacturing details
• By now you will have seen many datasheets…
  – Time to create your own
• Describe the functionality of the avionics board
  – Timing diagrams
  – Truth tables
  – Ratings
  – Plots

• Have all the parts?
  – Time to solder!
  – Soldering techniques
    (microscopes are not only for bio)
  – Part management
    • What happens when the part does not fit?
• The first time you power up your PCB everything will work perfectly, right?
  – power check
  – critical net check
  – assembly in stages

• How do you debug?
  – Oscilloscope operation
  – Signal generators
  – Logic analyzer introduction
What’s next?

• Fill out the questionnaire at the end of the syllabus handout
  – Think if you have any specific avionics in mind for the class

• Tell others about the class

• Problem Set 1 will be out until Monday Feb 13th, will be due **Tue** Feb 21st (Monday schedule)

• See you Monday!