Final Preparation
Questions?
Final Exam

• When: 8:00-10:00 am Wednesday, May 8th
• Location: here

• 1/3: midterm material
  – See lecture notes for midterm preparation

• 2/3: material since midterm

• 1 page of personal notes

• No electronic devices/books/other notes
Exam Parameters

• Most questions: multiple choice
  – Can grade your exam as you leave
• Some hand-written FSM questions
Sources of Material

- Zyante book and other assigned readings
- In-class and Zyante exercises
- Lecture notes
- Exams from prior years (both midterms and finals are available)
Pre-Midterm Material

- Number Representations (binary, hex, decimal)
- Arithmetic: adding, multiplying, incrementing, decrementing and shifting (<< and >>)
- Bit-wise operators: &, |, ~, ^
- Digital to analog conversion
- Digital I/O on the Teensy processors
- Basic circuits: LEDs, resistors, switches
- Motor control: H-bridges; PWM
- FSMs for control
- Serial communication: synchronous vs asynchronous serial
New Material

• Signed numbers
• Fixed point math
• Analog comparators
• Analog to digital conversion
• Proportional-derivative control
• Microprocessor components
• Performing multiple tasks
• Interrupts and interrupt service routines
• System safety & watchdog timers
• More FSMs
Representing Negative Integers

• Two’s complement representation
• Taking the negative of an integer
Fixed Point Math

• Converting between floating point and fixed point representations
• Addition, subtraction, multiplication and division of fixed point numbers

• Why do we do fixed point math?
Analog Circuits

- Analog comparator
- Analog-to-digital conversion
  - Flash ADC
  - Successive approximation
Proportional-Derivative Control

• Key PD control equation
• Meaning of the gains
• Phase plots
Key Microprocessor Components

- Data bus
- Data memory (RAM)
- Program memory (EEPROM in our case)
- General-purpose registers
- Special-purpose registers
  - Instruction register
  - Program counter
- Instruction decoder
- Arithmetic Logical Unit and Floating Point Unit
Performing Multiple Tasks

With PeriodicAction, we can define multiple, semi-independent code blocks (*tasks*)

- Naturally partition for the code
- Different tasks can be executed at different frequencies
- Some communication between tasks through global variables
Interrupts

• What are they?
• Interrupt service routines. Examples:
  – Pulse Width Modulation (PWM) generation (see slides)
  – Producing digital signals of various frequencies (e.g., can introduce software counters, too)
  – Using an ISR to ensure that a main-program task executed at a very regular period
  – Shared data problem
Safety and Watchdog Timers

Watchdog:

• Hardware counter that causes the processor to reset once it reaches a critical value
• The code’s job is to reset the counter fast enough to prevent this from happening ("feeding the dog")
• If the code does become stuck due to a bug or hardware problem, it is guaranteed that an ISR will be called (even resetting the processor)
Finite State Machines

• Definition
  – States
  – Inputs / Events
  – Transition function
  – Outputs / Actions
  – State transition diagrams

• FSMs for control

• At least one ‘draw the FSM’ question
Filtering

Low pass filter:

• Velocity smoothing
• Smoothing an error signal
• Numerical integration of LPF differential equation
C Code

• Be prepared to read (and possibly fix) simple C code
• Look to lecture discussions of code and your projects as you prepare