Final Exam

• When: 8:00-10:00 am Thursday, May 10th
• Location: here (Carson 439)

• 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
Pre-Midterm Material

- Basic gates
- Boolean algebra
- Digital circuits and circuit reduction
- Number representations (binary, hex)
- Bit-wise operators
- Sequential logic
- Components of microprocessors
- Memory behavior (input/output signals, buses, addressing)
Key Microprocessor Components

• General- versus special-purpose registers
• Instruction decoder
• Data memory (RAM)
• Program memory (EEPROM in our case)
• I/O modules
  – Digital input/output
  – Serial UART
Special-Purpose Registers

What does each do?

- Program counter
- Instruction register
- Status register
Timer/Counters

- Prescalars
- Counters (hardware)
  - Timer0, timer2: 8-bit
  - Timer1: 16-bit
- Interrupts on timerX overflow

- Computing timerX count frequencies/periods
- Computing timerX interrupt frequencies/periods
Interrupts

• What are they?
• Types of interrupts
• Interrupt service routines. Examples:
  – Pulse Width Modulation (PWM)
  – Producing digital signals of various frequencies (e.g., can introduce software counters, too)
  – Sensor control (sonar)
  – Serial buffering
Interrupt Woes

The Shared Data Problem

• One segment of code can be interrupted at any time to execute another segment of code
• Both access the same data structures
• Solution: ensure that a critical section of code cannot be interrupted
  – In our case: use disable/enable interrupts
Input/Output Systems

- Polling vs interrupt-driven input/output
- Buffers
  - Why do we need buffers?
  - Circular buffer implementation
- Modes of communication:
  - Parallel, serial, analog, PWM
Serial Communication

• What is it?
• How does it work?
  – Start/stop/parity bits
• Hardware implementation
  – A byte is shifted out one bit at a time
• Software implementation
  – getchar(), putchar()
Finite State Machines

• Definition
  – States
  – Events
  – Transition function
  – Outputs and output function
• State transition diagrams
• Relationship to sequential logic
• FSMs for control
Basics of Digital Port I/O

- Input/output selection
- Output value
- Input
C Code

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

• Digital to analog:
  – Pulse-width modulation
  – Resistive network

(last two lectures)

Note: we did not cover analog to digital conversion (don’t worry about this material)