CSE 466 – Software for Embedded Systems

- **Instructor:**
  - Gaetano Borriello
  - CSE 572, Hours: by app’t
  - gaetano@cs.washington.edu

- **Teaching Assistants:**
  - Brain French
    - CSE 003, Hours TTh 2:30-5:30
    - bmf@cs.washington.edu
  - Chris Grand
    - bitabur@cs.washington.edu
    - CSE 003, Hours TTh 2:30-5:30

---

CSE 466 – Software for Embedded Systems

- **Class Meeting Times and Location:**
  - Lectures: EE1 037, MWF 12:30-1:20
  - Lab: CSE 003, T – Section A, 2:30-5:20
    - Section B, 2:30-5:20

- **Exams**
  - Exam-I: Friday, 10 February, EE1 037, 12:30-1:20
  - Final demo: Friday, 10 March, CSE Atrium, 12:30-1:20
  - Exam-II: Thursday, 16 March, EE1 037, 8:30-10:20 (but only 1 hour)

- **Evaluations**
  - Wednesday, 8 March, EE1 037, 12:30-1:20
Embedded system – from the web

- **Definitions**
  - A device not independently programmable by the user.
  - Specialized computing devices that are not deployed as general purpose computers.
  - A specialized computer system which is dedicated to a specific task.
  - An embedded system is preprogrammed to perform a narrow range of functions with minimal end user or operator intervention.

- **What it is made of**
  - Embedded systems range in size from a single processing board to systems with operating systems.
  - A combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a dedicated function.
  - In some cases, embedded systems are part of a larger system or product, as is the case of an anti-lock braking system in a car.
  - A specialized computer system that is part of a larger system or machine.
  - Typically, an embedded system is housed on a single microprocessor board with the programs stored in ROM.
  - Some embedded systems include an operating system, but many are so small and specialized that the entire logic can be implemented as a single program.

- **Examples**
  - Virtually all appliances that have a digital interface – watches, microwaves, VCRs, cars – utilize embedded systems.
  - A computer system dedicated to controlling some non-computing hardware, like a washing machine, a car engine or a missile.
  - Examples of embedded systems are medical equipment and manufacturing equipment.
  - While most consumers aren’t aware that they exist, they are extremely common, ranging from industrial systems to VCRs and many net devices.

---

What is an embedded system?

- **Different than a desktop system**
  - Fixed or semi-fixed functionality (not user programmable)
  - Different human interfaces than screen, keyboard, mouse, audio
  - Usually has sensors and actuators for interface to physical world
  - May have stringent real-time requirements

- **It may:**
  - Replace discrete logic circuits
  - Replace analog circuits
  - Provide feature implementation path
  - Make maintenance easier
  - Protect intellectual property
  - Improve mechanical performance
What do these differences imply?

- **Less emphasis on**
  - Graphical user interface
  - Dynamic linking and loading
  - Virtual memory, protection modes
  - Disks and file systems
  - Processes

- **More emphasis on**
  - Real-time support, interrupts (very small OS, if we’re lucky)
  - Tasks (threads)
  - Task communication primitives
  - General-purpose input/output
  - Analog-digital/digital-analog converters
  - Timers
  - Event capture
  - Pulse-width modulation
  - Built-in communication protocols

Examples of embedded systems
What is an embedded system? (cont’d)

- Figures of merit for embedded systems
  - Reliability – it should never crash
  - Safety – controls things that move and can harm/kill a person
  - Power consumption – may run on limited power supply
  - Cost – engineering cost, manufacturing cost, schedule tradeoffs
  - Product life cycle – maintainability, upgradeability, serviceability
  - Performance – real-time requirements, power budget

Example: a temperature controller

- **Task: Tachometer (external interrupt)**
  
  ```
  now = getTime();
  period = then - now; //overflow?
  then = now;
  return;
  ```

- **Task: FanPWM (periodic, hard constraint)**
  
  ```
  count++;
  if (count == 0) PD6 = 1;
  if (count > Thi) PD6 = 0;
  return;
  ```

- **Task: TempControl (periodic, soft constraint)**
  
  ```
  if (Temp > setpoint) Thi++;
  if (Temp < setpoint) Thi--;
  if (period<min || period>max) PD0 = 1;
  ```

- **Task: Main**
  
  ```
  Thi = 0;
  setup timer for 1ms interrupt;
  setup timer for 100ms interrupt;
  while (1):
  ```
Capacity

- Assume:
  - 8 MHz processor @ one instruction/cycle
  - Assume fan runs between 30Hz and 60Hz
  - Assume 256ms period on speed control PWM, with 1ms resolution.

- What percent of the available cycles are used for the temperature controller?
  - \[
  \frac{\text{total instructions in one second}}{8 \text{MInstr/sec}}
  \]

- How much RAM do you need?

- How much ROM?

Resource analysis of temp controller

<table>
<thead>
<tr>
<th>Task: Tachometer (external interrupt)</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>now = getTime();</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>period = then - now; //overflow?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>then = now;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>return;</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Task: FanPWM (periodic, hard constraint) |   |   |   |
| count++;                               |   |   |   |
| if (count == 0) GP0 = 1;               |   |   |   |
| if (count > Thi) GP0 = 0;              |   |   |   |
| return;                                |   |   |   |

| Task: TempControl (periodic, soft constraint) |   |   |   |
| if (Temp > setpoint) Thi++;                  |   |   |   |
| if (Temp < setpoint) Thi--;                  |   |   |   |
| if (period<min || period>max) GP4 = 1;       |   |   |   |

| Task: Main |   |   |   |
| Thi = 0;   |   |   |   |
| setup timer for 1ms interrupt;              |   |   |   |
| setup timer for 100ms interrupt;            |   |   |   |
| while (1) ;                                  |   |   |   |

<table>
<thead>
<tr>
<th>Task</th>
<th>ROM</th>
<th>RAM</th>
<th>Instructions/Sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tach</td>
<td>~4</td>
<td>2 (period, then)</td>
<td>(4 \times 60 = 240)</td>
</tr>
<tr>
<td>FanPWM</td>
<td>~8</td>
<td>1 (count)</td>
<td>(8 \times 1000 = 8000)</td>
</tr>
<tr>
<td>TempControl</td>
<td>~10</td>
<td>1 (Thi)</td>
<td>(10 \times 2 = 20)</td>
</tr>
</tbody>
</table>

Total Instructions/Sec = 8260, at 8MIPS, that’s only 0.1% utilization!
Other resources? local variables, stack
What Are You Going to Learn?

- **Hardware**
  - I/O, memory, busses, devices, control logic, interfacing hw to sw

- **Software**
  - Lots of C and assembly, device drivers, low level OS issues
  - Concurrency

- **Software/Hardware interactions**
  - Where is the best place to put functionality hardware or software?
  - What are the costs:
    - performance
    - memory requirements (RAM and/or ROM)
    - integration of hardware and software courses
  - Programming, logic design, architecture
  - Algorithms, mathematics and common sense

What are you going to learn? (cont’d)

- Understanding of basic microcontroller architecture
- Understanding of interfacing techniques
- Appreciation of power management methods
- Understanding of basic communication protocols
- Facility with a complete set of tools for design/debug
- Experience implementing some real systems
Class logistics – see course web

- [http://www.cs.washington.edu/466](http://www.cs.washington.edu/466)
  - expands to [http://www.cs.washington.edu/education/courses/cse466/06wi/](http://www.cs.washington.edu/education/courses/cse466/06wi/)
- Class structure
- Business matters
- Grading
- Syllabus
- What we'll be doing

Class structure

- Lecture (25)
  - Closely linked to laboratory assignments
  - Cover main concepts, introduced laboratory problems
- Lab (8)
  - Implementation of two projects
  - Lab reports due prior with 30 minutes of start of next lab section
- Exams (2)
  - Based on lectures, labs, and reading assignments
- Final demo (1)
  - During scheduled final time – participation required
- Reading and source material (lots)
  - Some assigned, most you'll find on your own
Business Matters

- Lecture slides will be on line after class (links in several places)
- Get the CoursePak for CSE466 ($33.50, Communications B-042)
- Bring a $200 personal check to the first lab to check out a kit
- Random lab partner assignments, changed mid-quarter
- Sign up for CSE466 mailing list

Grading

- Lab reports:
  - Demonstration(s) required
  - Brief answers to questions embedded in assignment
  - Sometimes hand-in code
  - Do with your partner

- Distribution:
  - Labs: 40%
  - Exams: 30% (10 Feb and 10 Mar – both Fridays)
  - Demo: 15%
  - Class Participation: 15%
CSE466 Lab Projects

- Two multi-week projects
  - Four lab assignments each
  - Different lab partners
- First project
  - Familiarize with microcontroller
  - Learn how to interface devices to it
  - Test and debug
  - Basic communication between chips and between chip and PC
- Second project
  - Wireless communication
  - Embedded operating system
  - Real-time issues
  - Test and debug
  - Coordinated behavior among many devices

CSE466 Lab Projects (cont’d)

- Project 1 – USB device
  - Platform: ATmega16 AVR microcontroller
  - Accelerometer and multi-color LED used to select a color
  - Connects sensor and actuator to PC through USB port
  - Color Selector
    - Tilt accelerometer to move cursor on a color map
    - Send selected color back to device
    - Pulse-width modulation of multi-color LED to reproduce color
    - Eventually would be wireless USB device to home PC to control room lighting
CSE466 Lab Projects (cont’d)

- Project 2 – Ad hoc wireless network (“flock”)
  - Platform: UC Berkeley wireless sensor nodes (UCB “motes”)
  - Sound generation coordinated with neighbors and time of day
  - Coordinated behavior between different nodes
- **Flock-IV – the deployment**
  - Install in Allen Center atrium for pleasing auditory display
  - Focus on ease of installation and command/control
  - Switch between bird songs, crickets, water sounds