CSE 466 – Software for Embedded Systems

- Instructor:
  - Bruce Hemingway
  - CSE 464, Office Hours: 1:30-2:30 p.m., Monday, Wednesday,
  - or whenever the door is open
  - bruceh@cs.washington.edu

- Teaching Assistants:
  - Phil Roan
  - and Brian Mayton, IRS

Class Meeting Times and Location:

- Lectures: EE 003, MWF 12:30-1:20
- Lab: CSE 003E, Tuesday, 2:30-5:20

Exams

- Midterm: Wednesday, 11 February, EE 003, 12:30-1:20
- Final: Thursday, 19 March, EE 003, 8:30-10:20
CSE 466 – Software for Embedded Systems

- Grading Policy
  - There will be two exams, as shown on the class schedule.
  - Lab reports: Demo usually required, sometimes questions

- Ratios:
  - Lab: 50%
  - Exams total: 20%
  - Homework: 10% (reports)
  - Class Participation: 20%

Embedded systems
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.
Embedded system – from the web

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

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

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

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 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

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

### 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 * 60 = 240</td>
</tr>
<tr>
<td>FanPWM</td>
<td>~8</td>
<td>1 (count)</td>
<td>8 * 1000 = 8000</td>
</tr>
<tr>
<td>TempControl</td>
<td>~10</td>
<td>1 (THI)</td>
<td>10 * 2 = 20</td>
</tr>
</tbody>
</table>

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

### Class logistics – see course web
- Class structure
- Business matters
- Grading
- Syllabus
- What we’ll be doing
Class structure

- Lecture
  - Closely linked to laboratory assignments
  - Cover main concepts, introduce laboratory problems
- Lab
  - Implementation of two projects
  - Lab reports due prior with 30 minutes of start of next lab section
- Exams
  - Two, based on lecture, lab, and datasheet reading
- Final demo
  - During last class time – participation required

Business Matters

- Lecture slides will be on line after class (links in several places)
- 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: 50%
  - Exams: 20%
  - Homework: 10%
  - Class Participation: 20%

CSE466 Lab Projects

- Two multi-week projects
  - Four lab assignments each
  - Different lab partners

- First project
  - Familiarize with microcontroller
  - Learn how to interface various devices
  - Testing and debugging
  - Basic communication between chips and between chip and PC

- Second project
  - Wireless communication
  - Embedded operating system
  - Real-time issues
  - Testing and debugging
  - Emergent behavior of a collection of devices
CSE466 Lab Projects (cont’d)

- Project 1 – USB device
  - Platform: ATmega16 AVR microcontroller
  - Accelerometer and push-button used to control a tri-color LED
  - Connects sensor and actuator to PC through USB port
  - **Color Controller**
    - Tilt the controller to change color mix
    - Accelerometer senses movement – tilting
    - Mapping problem- two axes of motion to three-color space
  - Make a one-button virtual mouse
    - Tilt the accelerometer to move the PC cursor

CSE466 Lab Projects (cont’d)

- Project 2 – Ad hoc wireless network (“Mr. Poe”)
  - Platform: Intel iMote2
  - Sound generation coordinated with neighbors and time of day
  - Emergent behavior between different nodes
  - **Mr. Poe-- magnet poetry**
    - Install in Allen Center atrium for pleasing auditory display
    - On-the-fly poetry
    - Wireless nodes say words
    - And react to neighbors words