Embedded Systems Programming

Introduction
(Module 1)

Yann-Hang Lee
Arizona State University
yhlee@asu.edu
(480) 727-7507

Summer 2014
Course Syllabus

Course Goals:

- fundamental issues as well as practical development skill in the area of embedded systems programming
  - Design issues of embedded software and the knowledge of development and execution environment on target processors.
  - The functions and the internal structure of device interfaces, drivers, and real-time operating systems.
  - Multi-threaded embedded software in target environment.
  - Task scheduling and schedulability analyses.

Pre-requisites:

- Computer organization, data structures, and C/C++ programming.
- Helpful if you have some knowledge of operating systems and computer architecture.
## Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Classes (4 20-minutes lectures/day)</th>
<th>Lab and self-study (3 hours/day)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday, July 7</td>
<td>Introduction, Linux loadable modules</td>
<td>Exercise: data structures for Linux device drivers</td>
</tr>
<tr>
<td>Tuesday, July 8</td>
<td>Linux device driver</td>
<td>Lab: Linux loadable module</td>
</tr>
<tr>
<td>Wednesday, July 9</td>
<td>Quark SOC and Galileo architecture</td>
<td>Exercise: Galileo board design and GPIO programming</td>
</tr>
<tr>
<td>Thursday, July 10</td>
<td>Linux ISR and device driver</td>
<td>Exercise: user-level I2C programming</td>
</tr>
<tr>
<td>Friday, July 11</td>
<td>Thread and kernel synchronization</td>
<td>Lab: I2C-based EEPROM driver</td>
</tr>
<tr>
<td>Monday, July 14</td>
<td>Embedded programming</td>
<td>Exercise: set jmp and long jmp</td>
</tr>
<tr>
<td>Tuesday, July 15</td>
<td>Embedded programming</td>
<td>Lab: signal and asynchronous control</td>
</tr>
<tr>
<td>Wednesday, July 16</td>
<td>Real-time scheduling and analysis</td>
<td>Self-study: course review</td>
</tr>
<tr>
<td>Thursday, July 17</td>
<td>Real-time scheduling and analysis</td>
<td>Lab: real-time task management</td>
</tr>
<tr>
<td>Friday, July 18</td>
<td>Real-time scheduling and analysis</td>
<td>Final exam</td>
</tr>
</tbody>
</table>
Lab Setup

- OMAP beagleboard XM and trainer board
- Prepare your own micro SD card (and adapter) for booting
- Preferred development software
  - Linux, Eclipse, command lines, GUN tools
Real-time Embedded Systems

- **Embedded system**
  - the software and hardware component that is an essential part of another system

- **Real-time system**
  - provide well-timed computation
  - deadlines, jitters, periodicity
  - temporal dependency

```
Plant
  sensor
    A/D
    A/D
Control-raw computation
    D/A
  actuator
    Controller
```

Reference input
Embedded Systems -- Examples
Emerging Embedded Systems
Embedded Systems

- They are everywhere
- What are they?

**Hardware (chips) + Software (programs) for specific applications**

- **CPU (micro-processor)**
- **memory**
- **Timer**
- **I/O**
SW Development for RT ES

- To write the control software for a smart washer
  - initialize
  - read keypad or control knob
  - read sensors
  - take an action

- System current state
  - state transition diagram
  - external triggers via polling or ISR

- If there are multiple triggers and external conditions – single or multiple control loops
RT ES vs. General Software

- Multi-tasking for concurrent events
- Machine and device interface dependence and portability
- Control timing and scheduling
  - predictable actions in response to external stimuli
  - deadline (absolute or relative), and jitter
- Resource constraints and sharing
  - CPU time, stack, memory, and bandwidth
- Software abstraction, modular design
  - information hiding, OO, separate compilation, reusable
  - a sorting procedure -- function, input, output specification