UCSD Extended Studies
Business, Science and Technology Department

Embedded Linux Online
ECE-40105

Course Syllabus

- Instructor: Doug Abbott. Email: doug@intellimetrix.us, phone: (575) 590-2788
- Tentative virtual office hours: Tuesdays 12 – 2 pm PST, Thursdays 6 – 8 pm PST
- Section 122807, Spring 2017
- 3 units
- Start date: 4/13, End date: 6/8
- All classes online

Course Description:

Welcome to the exciting world of building computing devices with Linux, the free, open source operating system. Linux is rapidly emerging as the leading platform for embedded devices using high-performance, 32-bit and 64-bit processors. This course is designed to give you practical, hands-on experience in writing and debugging application programs on real target hardware.

The course is intended for engineers and programmers involved in developing embedded computing systems and devices using the Linux operating system. Prerequisites include:

- Fluency in C programming
- Some familiarity with basic digital hardware components such as network and serial ports
- Some experience using Linux is helpful
- Some experience debugging application code is helpful

Goals and Objectives:

The goal of this course is to teach you how to use Linux to build high performance embedded computing systems and devices.

By the end of this course, you will be able to:

- Describe the nature of open source software and why that's important
- Describe the structure and overall contents of the Linux file system
- Use shell commands and write simple scripts
- Use the Eclipse IDE to write, build, and test application software both on the host workstation and an embedded target board
- Configure and build the Linux kernel for a specific target environment
- Configure a target system to boot directly into the application
- Configure the boot and initialization environments

- Write and test simple device drivers

Student/Course Requirements:

In order to satisfy course requirements, class participants must participate in discussions, complete all assignments on time (on or before the due date), and of course, complete all quizzes.

Late assignments (anything posted or sent after the due date) may be graded -1/2 point for each day late unless due to a verifiable medical or family emergency. Assignments sent with the wrong naming convention or in the wrong format will be considered late until they are sent correctly. Late assignments will be accepted at the discretion of the instructor who may also waive the point penalty depending on circumstances.

Expect and plan for contingencies and technical problems (they WILL happen!).

Assignments

There is an assignment for every week of class. These are in two forms: programming assignments where you are expected to modify existing source code to implement new functionality, and procedural assignments where you follow a specific procedure to accomplish something.

Programming assignments are submitted in the form of C source code files. Generally, each programming assignment results in a single ".c" file. Name it week<n>-<last_name>.c and use the Assignment tool within Blackboard to submit. Also please put your name in the header comment at the top of every file you submit.

Points will be deducted for code that is not adequately commented or is otherwise difficult to understand.

There may be times when your solution to a programming problem involves modifying files other than the one the assignment specifically calls for. That's OK. I encourage creativity, but be sure to submit all files that you modify so I can rebuild your solution.

I will not tolerate programs that don't build! Programming assignments that do not build without error will be graded as zero and you will have exactly one chance to plead with me over why you should be allowed to submit a second time.

Procedural assignments will require that you submit some evidence that you have successfully carried out the procedure. The exact submission requirements are listed with each assignment. In general this will involve uploading a file using the same procedure described above for programming assignments.

Quizzes

There are two interim quizzes worth 15 points each and a final worth 40 points. Of necessity, these quizzes are "open book" because there's no way to enforce a closed book policy in a remote environment. Nevertheless, answers are expected to be original. Obvious copy-and-paste from the lesson scripts or PowerPoint files will be marked down.

Quizzes must be completed by 12 noon, Pacific Daylight Time, on the indicated day. They will no longer be available after that time. Plan ahead and give yourself plenty of time to complete them. The quizzes are based upon the lessons and assignments so do both before completing a quiz. You may only attempt each quiz once.

Discussion Board

A regular presence is expected in Blackboard discussions including contributions about class topics and discussion questions, and helping other students having problems. Think of the discussion board as a mail list or a newsgroup. You should drop in on the discussion board regularly (certainly more than once a week) to see what's happening.

So what constitutes contribution/participation? Here are some ideas:

- start a discussion on Blackboard (add a thread)
- respond thoughtfully to a topic posed by the instructor or another student
- provide links and resources related to the topic
- pose a thought-provoking question related to the topic
- help another student who's having difficulties

In grading discussion board participation, both quantity and quality are considered. Regular contributions that add to the knowledge base of other students, links to additional resources, and providing substantive food for thought get points. If you don't know a lot (yet!) about the topics, feel free to pose some questions to others and/or search the Internet and share what you find with the class.

You can sometimes learn just as much from the person sitting next to you (even in the virtual classroom!) as you can from the instructor. We're all here to learn.

Course Materials:

Since the class is primarily hands-on, practical programming, there is no required textbook. On the other hand, you are required to complete programming assignments on a target computer, specifically the Embedded Linux Learning Kit from Intellimetrix, available to UCSD students at a reduced price (www.intellimetrix.us/ucsdkit.htm)

Software and supplementary notes are available in the Resources section. There are also a number of books recommended for additional study.

Grading System:

Grades are based on points and letter grades are given as follows:

A+	191-200
A	180-190
A-	170-179
B+	160-169
B	149-159
B-	139-148
C+	129-138

C	118-128
C-	108-117
D+	98-107
D	87-97
D-	77-86
F	0-76

Weighted Grades

Discussion Board Participation:	15%
Assignments:	45%
Interim Quizzes:	20%
Final	20%
TOTAL	100%

Course Structure

Session 1: Introducing Linux and open source

- Introducing open source software and Linux.
- Explore the Linux file system, command shell, and scripting
- Assignment/Activities
  - Introduce yourself on the Discussion Board
  - Install Linux on your workstation

Session 2: Getting familiar with the target board

- Connecting and configuring the target board and its environment
- Our first program
- Assignment/Activities
  - Set up the target board
  - Build and run the LED program

Session 3: Application development with Eclipse

- The Eclipse development environment
- Assignment/Activities
  - Install and configure Eclipse
  - Build and debug a simple application
  - Interim quiz

Session 4: Accessing hardware, high-level simulation, multi-threaded programming

- Accessing peripheral devices from User Space
- Workstation as a debug environment – simulation

- Multi-thread programming – Posix threads
    - Build and run measure example
    - Convert to thermostat. Test in both simulation and target environments
    - Build and run Posix example

Session 5: Network programming

- The socket programming model
- Web servers
- Multiple clients
- Assignment/Activities
    - Build and test examples
        - Sockets
        - Networked thermostat
        - Multi-client server

Session 6: Working with the Linux kernel. Busybox

- Patching, configuring and building the kernel
- Busybox
- Assignment/Activities
    - Configure and build the kernel. Test the new kernel on the target
    - Configure and build Busybox
    - Interim quiz

Session 7: Graphics device driver. Booting directly into the application

- User view of device drivers
- Console and framebuffer devices
- Linux initialization
- Putting it all in flash
- Assignment/Activities
    - Build and test display example
    - Modify initialization script
    - Load kernel and root file system to flash

Session 8: Kernel programming – modules and device drivers

- Kernel modules
- Debugging kernel code
- Character device drivers
- Blocking I/O
- Assignment/Activities
    - Build and test module and character driver examples

Session 9: Device drivers continued

- Accessing hardware
- Course review
- Assignment/Activities
  - Build and test hardware driver
  - Modify thermostat application to use driver

Session 10: Final exam

- Good luck!
- Fill out course evaluation

In Case of Emergency

Emergencies happen! Know where to find emergency information online.

> *In the event of an emergency, information for UC San Diego Extension will be posted on the website at* ***extension.ucsd.edu.*** *Extension students must access the website to find out the status of the emergency situation. Email and or phone lines may not be accessible. Information will be updated online as the situation progresses and an ALL CLEAR will be posted once the situation is resolved.*