Department of Electrical and Computer Engineering
University of Massachusetts/Amherst

ECE322: Systems Software & Networking I, Fall 2019
(Required Course)

Catalog Data
This course provides the theoretical and practical foundations for engineering the production of contemporary and future software intensive systems, and provides the basis for the analysis and co-design of complex hardware and software systems. The course enables advanced engineering problem solving concepts and skills by means of state of the art tools. The primary objectives of the course are to provide a deep introduction to both i) "systems" software programming in a Unix environment and ii) the basic suite of tools for engineering software. Systems programming topics include process control, static and dynamic linking, exceptional control flow, system-level I/O, network programming, inter-process communication, and concurrent programming (4 credits).

Prerequisites
E&C-ENG 242 (Data Structures and Algorithms) with a grade of 'C' or better.

Instructors
Lectures: David E. Irwin, 211D Knowles Engineering Building, irwin@ecs.umass.edu
Teaching Assistant: Pradeep Ambati, 8C Marcus, lambati@umass.edu

Course Meeting Times
Lectures: M W F, 12:20 PM – 1:10 PM, 304 Engineering Building

Textbook
Computer Systems: A Programmer's Perspective, Third Edition, R. Bryant and D. O'Hallaron

Course Goals
Students completing this course will be able to do the following. Each goal includes an associated assignment that requires "…apply[ing] engineering, science, and math to identify, formulate, and solve complex problems." These assignments also require "…apply[ing] engineering design to produce solutions that meet specified needs."

1. Use Unix OS, Unix program control, and C programming language to OS software, e.g., a shell
2. Use and understand exception flow control in Unix (i.e., signals)
3. Use and understand the Unix compilation and linking process
4. Use and understand user-level memory management in C, i.e., malloc, free, pointers, etc..
5. Use and understand the sockets interface to implement networked applications
6. Use and understand basics of concurrent programming using locks/mutexes, semaphores, etc.

Relation of Course Goals to Student Outcomes:

PROGRAM OUTCOMES	COURSE OBJECTIVES					
	1	2	3	4	5	6

1. Able to apply engineering, science, and math to identify, formulate, and solve complex problems	Y	Y	Y	Y	Y	Y
2. Able to apply engineering design to produce solutions that meet specified needs	Y	Y	Y	Y	Y	Y
3. Able to communicate effectively with a range of audiences	Y	Y	Y	Y	Y	Y
4. Able to recognize ethical and professional responsibilities, and make informed judgments	N	N	N	N	N	N
5. Able to function effectively on a team	Y	Y	Y	Y	Y	Y
6. Able to develop and conduct experiments, analyze and interpret data, and draw conclusions	Y	Y	Y	Y	Y	Y
7. Able to acquire and apply new knowledge as needed, using appropriate learning strategies	Y	Y	Y	Y	Y	Y

Course Topics

1. **Introduction –** Unix history, operating environment (e.g., file layout and file permissions), and basic tools, such as shell interface, make, gdb, version control (git/subversion), scripting, etc.
2. **Unix Program Control –** System call interface, inter-process communication introduction, pipes, redirection operators
3. **C Programming –** C control and data structures, bit-level operations, arrays, pointers, structures, header files, macros, typdefs, libraries, etc.
4. **Exception Control Flow –** Hardware exceptions, processes and process control (fork/exec/join), Unix signals, non-local jumps
5. **Compilation and Linking –** Multi-stage compilation process, object files, executables, static and dynamic linking, symbols, symbol tables, symbol resolution, relocation, library interpositioning
6. **Memory Management –** Management of the heap and stack; implementations of malloc() and free(); focus on user-level memory management (rather than system-level virtual memory
7. **Network Programming** – High- and low-level I/O, network programming with sockets, web servers
8. **Concurrent Programming –** concurrent server designs, threads, I/O multiplexing with select

Prepared by: David E. Irwin April 20, 2019