ESE-2025 Embedded Operating Systems

Computer Studies

Course Number:	Co-Requisites:	Pre-Requisites:
ESE-2025	N/A	N/A
Prepared by:	Takis Zourntos, Outline Creator	
Approved by:	Chris Slade, Dean School of Business and International Ed.	
Approval Date:	Thursday, August 22, 2019	
Approved for Academic Year:	2019-2020	
Normative Hours:	75.00	

Course Description

This course provides a comprehensive study of the Linux operating system, describing its multi-tasking operation and structure (kernel, kernel space and user space, filesystem), boot process, commands and filesystem operations. Both host and embedded target Linux distributions will be compared/contrasted. Students will learn advanced shell programming using BASH, developing scripts for searching, organizing and manipulating data. Students will also learn advanced C++ techniques, developing multi-threaded applications using POSIX pthreads, queues, mutexes and semaphores, under Linux, and using make to build applications. Students will learn about developing and deploying custom embedded Linux distributions using the Yocto Project. Students will use emulation tools and burn U-boot and Linux images to be deployed on a real ARM Cortex-A platform.

Course Learning Outcomes/Course Objectives

1. **Apply various operating system basics and features in system development.**
   - 1.1 Explain the concept of a multi-tasking operating system, cooperative versus preemptive modes of kernel/scheduler operation;
   - 1.2 Explain the origins of Linux: the open source movement, GNU
   - 1.3 Describe Linux/Unix architecture: kernel and user space, filesystems, user privileges and file permissions;
   - 1.4 Perform basic Linux commands and file system operations using the Command Line Interpreter (CLI);
   - 1.5 Set up an Internet connection and install programs (Ethernet, Ethernet over USB, etc.) on an embedded Linux platform;
   - 1.6 Explain the Linux boot process, start-up services (systemd), and the Linux device tree, comparing/contrasting Linux on the host machine and Linux on the embedded system;
   - 1.7 Download, compile and configure a Linux kernel;

1.8 Use GNU editors such as nano, vim and gedit.

2. Compose shell scripts and device-tree source code for managing data, applications andhardware

2.1 Compose simple shell scripts based on straightforward command sequences from memory;

2.2 Compose more advanced shell scripts using command-line operands, error handling, flow control, variables/parameters (integers and strings), I/O redirection (>>), arithmetic expressions, regular expressions and globbing;

2.3 Explain the purpose of environment variables and the customizing of user Linux environments;

2.4 Explain variables/parameters, special characters, quotes and test constructs in shell scripts;

2.5 Apply branching and looping structures for shell programming;

2.6 Explain how to access devices and memory-mapped I/O through shell scripts using sysfs;

2.7 Create basic Makefile scripts and the use of make to build applications;

2.8 Pass data into applications written in C/C++ via the filesystem;

2.9 Explain the purpose of the Linux device tree and how the device tree is used by U-Boot and Linux;

2.10 Describe the structure of a node in a device tree source (.dts) file;

2.11 Introduce a custom device node to a dts file;

2.12 Deploy a compile a new device node from a dts file using the device tree compiler (dtc) to produce a device tree blob (.dtb).

3. Compose embedded applications in C++ under Linux:

3.1 Write applications that use generic programming, exception handling, and multiple threads;

3.2 Write programs that use templated functions;

3.3 Design and implement template classes;

3.4 Design classes that extend the STL exception class and provides a constructor that accepts an exception message of type string;

3.5 Explain the meaning of stack unwinding;

3.6 Implement programs that use try, catch, finally blocks and use functions that re-throw exceptions;

3.7 Describe when to use exceptions;

3.8 Write applications that create and manage STL thread objects;

3.9 Write embedded applications that correctly use condition variables, shared memory, inter-thread communication, semaphores, critical sections and mutexes;

3.10 Write applications that use condition variables to signal multiple threads;

3.11 Write applications that use std::stringstream to read from and write to a file;

3.12 List the different std::ios flags and describe their purpose;

3.13 Describe the difference between random access files and sequential access files;

3.14 Design a stream management class that automatically opens and closes streams;

3.15 Use the stream manipulators to set the width, precision and base of output data;

3.16 Create programs that handle input/output routines, streams, serial communication and socket communication.

4. Develop Custom Embedded Linux Distributions using the Yocto Project:

4.1 Setup and install the Yocto Project, Poky, BitBake and OpenEmbedded;

4.2 Describe the purpose of the Yocto Project, Poky, BitBake and OpenEmbedded;

4.3 Setup and install the Poky build environment;

4.4 Create a simple BitBake recipe;

4.5 Generate a root filesystem using BitBake;

4.6 Describe how to provide additional features to Poky using BitBake recipes in layers;

4.7 Describe the purpose of QEMU;

4.8 Test a custom built Linux distribution using QEMU.

Learning Resources

a. Required

Mastering Embedded Linux Programming by Chris Simmonds; Packt Publishing, (Dec.2015).

b. Supplemental

Embedded Linux Primer: A Practical Real-World Approach by Christopher Hallinan; Prentice Hall, 2nd ed. (Oct. 2010).

Student Evaluation

Tests 25%

Tests (1@10%, 1@15%)

Validates Outcomes: CLO 1, CLO 2, CLO 3, CLO 4, VLO 1, VLO 2, VLO 3

Assignments 15%

Assignments (3 equally weighted @5% each)

Validates Outcomes: CLO 1, CLO 2, CLO 3, CLO 4, VLO 1, VLO 2, VLO 3

Laboratory Sessions 60%

Laboratory Sessions (10 equally weighted @ 6% each)

Validates Outcomes: CLO 1, CLO 2, CLO 3, CLO 4, VLO 1, VLO 2, VLO 3

Grade Scheme

The round off mathematical principle will be used. Percentages are converted to letter grades and grade points as follows:

Mark (%)	Grade	Grade Point	Mark (%)	Grade	Grade Point
94-100	A+	4.0	67-69	C+	2.3
87-93	A	3.7	63-66	C	2.0
80-86	A-	3.5	60-62	C-	1.7
77-79	B+	3.2	50-59	D	1.0
73-76	B	3.0	0-49	F	0.0
70-72	B-	2.7			

Prior Learning Assessment and Recognition

Students who wish to apply for prior learning assessment and recognition (PLAR) need to demonstrate competency at a post-secondary level in all of the course learning requirements outlined above. Evidence of learning achievement for PLAR candidates includes:

- Not Applicable: Post graduate course and not eligible for PLAR.

Course Related Information

This course is designed to provide an emphasis on hands on experience through the use of laboratory sessions and assignments.

College Related Information

Academic Integrity

Lambton College is committed to high ethical standards in all academic activities within the College, including research, reporting and learning assessment (e.g. tests, lab reports, essays).
The cornerstone of academic integrity and professional reputation is principled conduct. All scholastic and academic activity must be free of all forms of academic dishonesty, including copying, plagiarism and cheating.

Lambton College will not tolerate any academic dishonesty, a position reflected in Lambton College policies. Students should be familiar with the Students Rights and Responsibilities Policy, located at lambtoncollege.ca. The policy states details concerning academic dishonesty and the penalties for dishonesty and unethical conduct.

Questions regarding this policy, or requests for additional clarification, should be directed to the Lambton College Centre for Academic Integrity.

Students with Disabilities

If you are a student with a disability please identify your needs to the professor and/or the Accessibility Centre so that support services can be arranged for you. You can do this by making an appointment at the Accessibility Centre or by arranging a personal interview with the professor to discuss your needs.

Student Rights and Responsibility Policy

Acceptable behaviour in class is established by the instructor and is expected of all students. Any form of misbehaviour, harassment or violence will not be tolerated. Action will be taken as outlined in Lambton College policy.

Date of Withdrawal without Academic Penalty

Please consult the Academic Regulations and Registrar's published dates.

Waiver of Responsibility

Every attempt has been made to ensure the accuracy of this information as of the date of publication. The content may be modified, without notice, as deemed appropriate by the College.

Students should note policies may differ depending on the location of course offering. Please refer to campus location specific policies:

- **Student Rights & Responsibilities & Discipline policy** (2000-5-1) - https://www.lambtoncollege.ca/custom/Pages/Policies/Policy.aspx?id=2147491640 Mississauga Campus Policies - https://www.lambtoncollege.ca/Programs/International/Lambton_in_Mississauga/Student_Policies/ Toronto Campus Policies
- https://www.lambtoncollege.ca/Programs/International/Lambton_in_Toronto/Student_Policies/

Note: It is the student’s responsibility to retain course outlines for possible future use to support applications for transfer of credit to other educational institutions.