ECE 391 - Computer Systems Engineering

Instructors:

This class is taught by many professors. Prof. Matt Frank is known for being down to earth, in touch with students' experience in the lab, and his willingness to meet with students to discuss anything. Prof. Steve Lumetta is known for his good presentation of material, and his high expectations. Lately, Profs. Kalbarczyk and Borisov have been teaching the course. Prof. Kalbarczyk generally teaches during the fall semester while Prof. Borisov teaches during the spring semester.

Prerequisites:

You really do need ECE 220 and ECE 120 before you take ECE 391. Your understanding of LC3 assembly and processor architecture will be crucial in transitioning to x86 assembly. Additionally, knowledge of pointers and basic data structures (linked lists) will be crucial to success in this class. CS 225 is not a listed prerequisite for this course, but it is recommended for students to take it first in order to develop the programming experience and maturity to handle ECE 391.

When to Take It:

Many students take ECE 391 during their junior year or during their first semester senior year, just in time to serve as a pre-req for ECE 411, which they take their last semester. A good handful of students also take this class during the second semester of their sophomore year. If you are a consistently good student and/or have a very capable group for the final project ready, you will benefit from taking this class earlier. This is an outstanding class to discuss in interviews if you want an internship or career in low-level programming. It is a requirement for CompEs, and a good class for any EE interested in a software career.

Class Content:

This course is all about bridging the gap between hardware and software. The course covers I/O semantics, synchronization, interrupts, multitasking, virtualization of resources, protection, and resource management concepts. Students begin by learning x86 assembly. x86 is similar to the LC3 in many ways, but is more complex; while the LC3 is just a textbook example, x86 is an industry standard. Students learn how processors organize and access RAM, and how processors keep track of and switch between multiple programs running at once. Students learn about how computers handle inputs like key-strokes and moving the mouse, and how programs use an operating system's system calls to execute commands in the processor. The final goal is a thorough understanding of how operating systems work, using Linux as an example.

Work:

This course is lab based (open lab), and has a substantial work load. Try to avoid taking other heavy labs at the same time. There are three MPs (four including MP0). MP1 and MP2 are different between fall and spring semesters, but the concepts and implementations are very similar. MP handins consist of live demonstrations to TAs in the lab, generally during Monday evenings. Each MP has a pre-lab/problem set. Pre-labs are a small time commitment, and can be completed in groups. Do pre-labs diligently, since they cover the materials you need to understand for the MPs and the exams. The MPs themselves are the bulk of the work, but rest assured there are no post-lab reports. The first two (three) MPs are individual; the third MP is completed in a group of four. In MP0, students prepare and familiarize themselves with the work environment used in all future MPs by setting up a virtual machine within Windows. In the first MP, students write several routines in x86 assembly and are provided with an introduction to how drivers accomplish tasks in the Linux kernel. The second MP consists of two checkpoints. In the first checkpoint, students modify given code for an already written video game to expand the VGA graphics capabilities and add additional graphics features. In the second checkpoint, students write hardware device drivers for a hand-held controller and enable multi-threads. MP3, the longest out of all the MPs, consists of five checkpoints. In the third MP, students write their very own Linux-based operating systems in groups of four, starting from almost nothing. Those who may wish to enter the design competition for the course may also add extra functionalities to their kernels that are not listed inside the documentations. All MPs are done in an emulator, but for your own satisfaction, it is possible to load your own operating system onto your actual computer. The last MP is extremely challenging, but even more rewarding.

The class has two evening midterms and a final exam. Exams cover material learned from both the lectures and the programming assignments.

Life After:

If you liked this class more than ECE 120 and ECE 385, you are probably on track for a software career. If you liked ECE 385 better, you are probably on track for a hardware career, and should hurry up to take ECE 411, so you can talk about it in your interviews. The combination of ECE 391 and ECE 411 leads to the deep understanding of hardware, software, and their interactions, which defines a quality computer engineer. ECE 391 trains students to be good coders. You can use this course to help you into any software job. Specifically, it is good preparation for low level programming jobs, such as operating systems and device drivers for PCs, smart phones, or any handheld electronic gadget. Any company that makes processors, ASICs, or FPGAs, needs software written for their specific hardware. Other low level programming classes you would likely enjoy are ECE 428 (Distributed Systems), ECE 435 and 438 (Communication Networks and Lab), and almost any 400 level CS class. Students may also consider taking CS 461 (ECE 422) and CS 463 (ECE 424) (Computer Security).