CMPT 295 - D200 Intro. to Computer Systems

Instructor(s): Arrvindh Shriraman

Calendar Objective/Description:
Intro. to Computer Systems

Instructor's Objectives:

Prerequisites:

Basic programming skills (C++ or C). Basic background at Unix/Linux.
https://www.cs.sfu.ca/~ashriram/Courses/CS431/assets/notebooks/essentialC.pdf (We assume familiarity with the Essential programming constructs listed in the pdf)

https://www.cs.sfu.ca/~ashriram/Courses/CS431/assets/notebooks/unixProgrammingTools.pdf (We assume familiarity with basic unix shell programming tools)

We assume linux environmental experience; typically obtained through CMPT 127, 135 or related course at SFU. All assignments and lab in this course are administered through the linux environment and shell. If you are not comfortable working with linux shell, talk to an advisor.

As of May 26, 2021, the university has announced its plan for approximately 70-80% of teaching in person in Fall 2021. It has also stated that: "not all courses will be delivered in person. The fall will be a transitional term. Deans, supported by the work of chairs and directors, will make final decisions about whether courses will be taught remotely or in person."

Please continue to check our course outline for further information.
Should this course be taught remotely, students must have access to a computer with internet access, allowing the use of a conferencing system such as Zoom or BB Collaborate Ultra. Some components of the course will require synchronous (real-time) participation during the scheduled lecture and/or exam times. Visual proctoring will most likely be required for midterms and exams.

Overview

This course should develop students' sense of what really happens when software runs — and that this question can be answered at several levels of abstraction, including the hardware architecture level, the assembly level and the C programming level. The core around which the course is built is C, assembly, and low-level data representation, but this is connected to higher levels (roughly how basic other programming languages could could be implemented), lower levels (the general structure of a processor), and the role of the processor and its how it is implemented.

This course provides an introduction to computer organization, systems programming and the hardware/software interface. Topics include instruction sets, computer arithmetic, datapath design, data formats, addressing modes, memory hierarchies including caches and virtual memory, and multicore architectures. Students learn assembly language programming and design a pipelined RISC V processor. The course is open to any undergraduate who has mastered the 120s material.
RISC V
This version of the course will study the core components in computer architecture through the lens of the RISC-V (V: Five) ISA. What is RISC V? It is a modern open source instruction set that enables students to learn both assembly-level programming and the digital design of a processor.

**This course is a hands-on, programming-heavy course**. Expect to get down and get your hands dirty with C programming and hardware design. We will discuss the fundamental design and engineering trade-offs in computer architecture at every level.

Course Offer Plan:

**Lab:** There will be dedicated lab days. (required lab). Attendance is required and counts towards 2.5% of your overall grade. zoom/in-person (To be decided)

**Instructor Hours (50 minute session) will be instructor hours. zoom/in-person (To be decided).**

**Lecture (1hr 50minute)** will be live lecture day. I will be reviewing the quiz from the previous week. I will be on presenting some of the material from the week's module. Further time will be spent on answering class-wide questions. zoom/in-person (To be decided)

**Videos** - All lecture content will be made available as videos 1 week prior. The expectation is that these videos are watched prior to the class. https://en.wikipedia.org/wiki/Flipped_classroom

Text books. Please try and obtain the eversions of these text books.

Prerequisites:

see go.sfu.ca

Topics:

- RISC-V
- Bits, Arrays, Pointers
- Number formats
- Caches
- Memory allocator
- Processor Pipeline and Digital logic
- Parallelism

Grading:

Tentative: 6 programming-heavy assignments (45-50%) 1 Midterm (15%) 1 Final (30%) 9-10 Labs (2.5%) 9-10 Weekly Quizes (2.5%) Students must attain an overall passing grade on the weighted average of exams in the course in order to obtain a clear pass (C- or better).

Required Books:


All of Programming, Andrew Hilton and Anne Bracy, 0000000000000, https://play.google.com/store/books/details?id=-zViCgAAQBAJ&pcampaignid=books_web_aboutlink

Academic Honesty Statement:

Academic honesty plays a key role in our efforts to maintain a high standard of academic excellence and integrity. Students are advised that ALL acts of intellectual dishonesty will be handled in accordance with the SFU Academic Honesty and Student
Conduct Policies (http://www.sfu.ca/policies/gazette/student.html).