Instructor
Tashfeen Karamat
Office ME4230
Email: TashfeenKaramat@cunet.carleton.ca

TAs
1. Feng Su
2. Ghazal Rouhafzay
3. Esther Akinwumi

Office Hours
Thursdays from 1130 hrs to 1255 hrs

Course Objectives
Engineers working with computers/microcontrollers (including mobile devices) must understand computer systems at hardware/software interface level. The concepts at this level form the foundation on which more powerful computer systems are based (such as desktop systems, servers, multi-processors, and supercomputers). To realize this, the course objectives include the following:

- Understand the basic functional units in a typical computer system.
- Understand binary number representation including two's-complement signed values, hexadecimal number representation, and character representation using the ASCII encoding.
- Understand the common basic features of a processor, including the program counter register, instruction register, general-purpose registers, and memory interface.
- Describe the essential steps for fetching, decoding, and executing instructions with respect to the common basic features of a processor.
- Understand memory organization concepts, including endian-ness and byte addressability.
- Understand assembly-language instruction syntax including mnemonics, operand specification, and addressing modes.
- Understand subroutine linkage, the use of the processor stack, and subroutine nesting.
- Understand and implement basic assembly-language programs and how it manipulates
information at the hardware/software interface and can support a high-level language

- Understand basic input/output interfaces and assembly-language implementation of program-controlled input/output operations.
- Understand the partition of instruction processing into basic steps that correspond to stages of execution in hardware.
- Understand the design of the control portion of a processor, and its datapath consisting of the general-purpose register file, the arithmetic/logic unit, the memory interface, internal registers between stages, and multiplexers for selecting inputs to components.
- Understand the common organizational and operational characteristics of semiconductor-based memory components.

**Learning Outcomes**

- Describe the overall architecture of a basic computer system, including:
  - How a processor, memory and I/O devices are connected using a bus
  - How a processor, memory and I/O devices communicate via a bus protocol
  - Explain the concept related to computer memory
- Describe how a processor is organized
- Describe how a processor executes instructions; in particular:
  - Identify the actions taken at each stage of a processor
  - Able to explain the detailed actions taken to execute an instruction
- Understand and write simple assembly language programs that perform computations and drive input/output devices
- Explain how a higher-level language (such as C) uses the features of assembly language to implement language features
- Understand basic input/output interfaces and assembly-language implementation of program-controlled input/output operations

**Graduate Attributes**

- Computer Systems, Digital systems and computers, Electronics and Circuits
- 2.3 Validity of results (reasonableness, effects of assumptions, analytical approach
- 4.4 Design solution(s)
- 5.3 Tools for design, experimentation, simulation, visualization and analysis (work on wording)

**Course Web Site**

The course will use the SYSC3006A Fall 2018 website on cuLearn.

Course materials will be placed on the website for student personal use, and students are responsible for checking the website frequently. Student, TA or professor materials created for this course (including presentations and posted notes, labs, case studies, assignments and exams) remain the intellectual property of the author(s). They are intended for personal use and may not be reproduced or redistributed without prior written consent of the author(s).

**Textbook and References**
A specific textbook is not recommended in the course. Course notes and supporting material will be supplied via the course webpage for student personal use.

**Links to Software, libraries, additional resources**

1. Link to software tool used for labs: http://www.cburch.com/logisim/

**Evaluation and Marking Scheme**

1. Labs: 25%
2. Mid-term test: 25%
3. Final exam: 50%

**Labs**

There will be 10 graded laboratories that will be posted on the course website. Lab attendance is mandatory, with work to be demonstrated by the end of the lab period. Marks will only be awarded for labs demonstrated during the student’s schedule lab time (according to section registration). During the scheduled lab times, TA assistance will be available. The computer lab is open seven days a week, whenever the building is open. You may use the lab at any time other than those timeslots when the lab is reserved for other courses or for other sections of this course.

**Exams**

**Midterm Exam:** There will be one closed book, no-calculator, midterm test. Date and time TBD.

**Final Exam:** A closed book, no-calculator, final exam will be held during the University’s formal examination period.

Students who miss the final exam may be granted permission to write a deferred examination (see the Undergraduate Calendar for regulations on deferred exams).

**General Regulations**

- **Copyright on Course Materials:** The materials created for this course (including course outline, slides, posted notes, labs, project, assignments, quizzes, exams and solutions) are intended for personal use and may not be reproduced or redistributed or posted on any web site without prior written permission from the author(s).
- **Attendance:** Students are expected to attend all lectures and lab periods. The University requires students to have a conflict-free timetable. For more information, see the current Undergraduate Calendar, Academic Regulations of the University, Section 1.2, Course Selection and Registration and Section 1.5, Deregistration. Requests to accommodate a missed midterm exam, lab periods, etc., because of conflicts with jobs or vacation plans will not be considered.
- **Health and Safety:** Every student should have a copy of our Health and Safety Manual. A PDF copy of this manual is available online: http://sce.carleton.ca/courses/health-and-safety.pdf.
• **Deferred Term Work**: Students who claim illness, injury or other extraordinary circumstances beyond their control as a reason for missed term work are held responsible for immediately informing the instructor concerned and for making alternate arrangements with the instructor and in all cases this must occur no later than three (3.0) working days after the term work was due. The alternate arrangement must be made before the last day of classes in the term as published in the academic schedule. For more information, see the Academic Regulations of the University, Section 2.6, Deferred Term Work.

• **Appeal of Grades**: The processes for dealing with questions or concerns regarding grades assigned during the term and final grades is described in the Academic Regulations of the University, Section 2.7, Informal Appeal of Grade and Section 2.8, Formal Appeal of Grade.

• **Academic Integrity**: Students should be aware of their obligations with regards to academic integrity. Please review the information about academic integrity at: https://carleton.ca/registrar/academic-integrity/ This site also contains a link to the complete Academic Integrity Policy that was approved by the University's Senate.

• **Academic Accommodations**: Requests for Academic Accommodation You may need special arrangements to meet your academic obligations during the term. For an accommodation request, the processes are as follows:
  - Pregnancy obligation
    Please contact your instructor with any requests for academic accommodation during the first two weeks of class, or as soon as possible after the need for accommodation is known to exist. For more details, visit the Equity Services website: carleton.ca/equity/wp-content/uploads/Student-Guide-to-Academic-Accommodation.pdf
  - Religious obligation
    Please contact your instructor with any requests for academic accommodation during the first two weeks of class, or as soon as possible after the need for accommodation is known to exist. For more details, visit the Equity Services website: carleton.ca/equity/wp-content/uploads/Student-Guide-to-Academic-Accommodation.pdf
  - Academic Accommodations for Students with Disabilities
    If you have a documented disability requiring academic accommodations in this course, please contact the Paul Menton Centre for Students with Disabilities (PMC) at 613-520-6608 or pmc@carleton.ca for a formal evaluation or contact your PMC coordinator to send your instructor your Letter of Accommodation at the beginning of the term. You must also contact the PMC no later than two weeks before the first in-class scheduled test or exam requiring accommodation (if applicable). After requesting accommodation from PMC, meet with your instructor as soon as possible to ensure accommodation arrangements are made. carleton.ca PMC
  - Survivors of Sexual Violence
    As a community, Carleton University is committed to maintaining a positive learning, working and living environment where sexual violence will not be tolerated, and is survivors are supported through academic accommodations as per Carleton's Sexual Violence Policy. For more information about the services available at the university and to obtain information about sexual violence and/or support, visit: carleton.ca/sexual-violence-support
  - Accommodation for Student Activities
    Carleton University recognizes the substantial benefits, both to the individual student and for the university, that result from a student participating in activities beyond the classroom experience. Reasonable accommodation must be provided to students who compete or perform at the national or international level. Please contact your instructor with any requests for academic accommodation during the first two weeks of class, or as soon as possible after the need for accommodation is known to exist. https://carleton.ca/senate/wp-content/uploads/Accommodation-for-Student-Activities-1.pdf
Additional Information

Course Description

This course introduces basic computer structure, instruction set architecture, assembly-language programming, input/output considerations, processor design based on digital logic, and memory technology and memory system design principles. The increasing use of high level languages such as C is reducing the need for assembly language programming proficiency. As a result, the course will focus more on understanding how an assembly language manipulates information at the hardware/software interface and can support a high-level language and will not emphasize developing complete applications in assembly language. The primary intent is to provide a foundation for subsequent courses on hardware/software interfacing for microprocessor-based systems, computer system architecture, and digital systems engineering. A secondary intent is to provide an appreciation of the low-level representation of software compiled from high-level languages into machine instructions. The practical aspects of the course are illustrated with Logisim software tool which is an educational tool for designing and simulating digital logic circuits. This course builds on and supplements knowledge from other courses on digital logic, circuits and electronics, and software/programming.

Prerequisites

(SYSC-2002 or SYSC-2006) and ELEC-2607. This course precludes additional credit for SYSC-2001 and SYSC-2003. This course may not be taken for credit by students in Computer Systems Engineering, Communications Engineering, or Software Engineering.

Students who have not satisfied the prerequisites for this course must either a) withdraw from the course, or b) obtain a prerequisite waiver online at http://www.carleton.ca/engineering-design/current-students/undergrad-academic-support/, or c) will be deregistered from the course after the last day to register for courses.

Study Questions

Study questions will be posted on the course website but will not be collected and graded.

Students are encouraged to discuss issues when working on labs; however, they are expected to do their own lab work individually. Suspected plagiarism will be investigated and may result in a mark of zero for the lab. As well, alleged instructional offences will be reported to the Associate Dean of Engineering. (Please see the current undergraduate calendar, “Instructional Offences”, in the Undergraduate Calendar Supplement).

Students are warned that the labs and study questions form a very important part of this course doing these (by yourself) is an excellent way for you to learn the material. In this context, it should be noted that copying labs is, even if you are not caught, a self-defeating exercise. Historically, most of the students who resorted to copying did not do particularly well on the mid-term or final exam.

Lecture Contents

The following themes are interwoven in the lectures throughout the course.

1. Computer system architecture (basics)
• Computer system components: processor, memory, I/O, interconnection bus
• Information encoding, data representation in binary, hexadecimal
• Number systems, unsigned integers, signed integers, 2’s complement
• Computer hardware organization: datapath and control
• Registers, instruction cycle

2. Hardware/software interface (Instruction Set Architecture)
• Instructions: data manipulation, data transfer, control flow, instruction encoding
• Computer arithmetic, flags

3. Microcontroller example
• Microcontroller concept
• System on Chip, memory model, ISA

4. Assembly language programming
• Code snippets, examples
• Assembly process, linker, loader

5. High-level Language Support
• Variables, arrays, structures, assignment, looping, conditional statements
• Procedures and functions, parameter passing

6. Peripheral I/O and Interrupts
• Register model of peripheral devices: parallel I/O, serial I/O, timers
• Polling
• Hardware interrupts: vectored and prioritized, Nested Vector Interrupt Controller(NVIC)
• Examples: timer, serial
• Software interrupts, o/s calls