# Computer Architectures
# CSCI 361
# Spring 2019 Syllabus

> We can only see a short distance ahead, but we can see plenty there that needs to be done.
>
> –Alan Turning

## Instructor Details


| | |
|---|---|
| **Name:** | Professor Jesse Johnson |
| **Office:** | 417 Social Science Building |
| **Telephone:** | (406) 243-2356 |
| **Email:** | jesse.johnson@umontana.edu |
| **Web:** | Faculty Home Page |
| **Office Hours:** | TWTh* 14:00–15:00 , Social Science Building 417 |
| | *Thursday office hours impacted by chair's meetings* |
| | *Or, by appointment.* |


## Prerequisites

Students taking this course are expected to have:

- Programming experience demonstrated by passing CSCI 136 (Introduction to CS II) or a similar course.
- Organizational skills and familiarity with computers sufficient to install new software, create a file system for the course, and execute Java programs from the command line.
- Knowledge of the Python language adequate to make alterations to an existing Python program.
- The ability to attend class.

## Course Objectives

The course objective is to integrate key notions from algorithms, computer architecture, operating systems, compilers, and software engineering in one unified framework. This will be done constructively, by building a general-purpose computer system from the ground up. In the process, we will explore many ideas and techniques used in the design of modern hardware and software systems, and discuss major trade-offs and future trends. Throughout this journey, you will gain many cross-section views of the computing field, from the bare bone details of switching circuits to the high level abstraction of object-based software design.

## Student Outcomes

Upon successful completion of this course, student will be better able to:

- apply knowledge of computing and mathematics appropriate to the programs student outcomes and to the discipline.
- analyze a problem, and identify and define the computing requirements appropriate to its solution.
- design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs.
- function effectively on teams to accomplish a common goal.
- communicate effectively with a range of audiences.

## Textbook

This semester I'll be using the following text. You need to purchase a copy.

**The Elements of Computing Systems**
*Nisan and Schocken*
MIT Press
2005

## Online Resources

Please bookmark the following online resources immediately:

- with the exception of the textbook, all course material will be made available online, through the University of Montana's Moodle system,
- the textbook has a web site, and
- there is a Coursera Course for this text, and videos of lectures are available if you register (helpful if you have to miss a class).

### Software

This course uses simulators to test the design of your hardware. They are written in Java and run on Windows, OSX, or Linux. Of course, you'll need the Java runtime environment installed on your computer. The software should be downloaded and configured according to the instructions here.

## Course Format

This is a hands-on course, structured around building a series of twelve hardware and software projects. Each project is accompanied by a design document, an API, an executable solution, a test script (illustrating what the module is supposed to do), and a detailed implementation plan (proposing how to build it). The projects are spread out evenly, so there will be no special pressure towards the semesters end. That said, they are also quite 'vertical' in that many depend on successful completion of the previous project.

Our time together in the classroom will be spent as follows.

**Board work** Each contact period begins by randomly selecting students to go to the board and demonstrate their work. I will randomly sample the calls *with replacement*, meaning that the probility of going to the board is the same for each student, every time. This will take about 20 minutes and will involve 4–5 students. Each student will be graded based on a rubric appearing on the course Moodle. The same grade will be awarded to the student's group, forcing groups to take responsibility for all members. While presenting, students may ask their group two questions that have one sentence answers.

**Bug bounty** While students are presenting, other students can call out mistakes to claim a "bug bounty". A student identifying a significant error or demonstrating a correct way of doing a problem to a student that is "stuck" will receive 5% of extra-credit on any in-class or group work assignment they scored less than 100% on. Student's must be given at least 30 seconds before "bug bounty" is called out. Despite extra-credit opportunities, no more that 100% will be awarded as a final grade for in-class and group work problems.

**Lecture** After board work, I will lecture for about 20-30 minutes. Beyond the background theory, I will focus on working examples that are similar to the assigned work.

**Group problem solving** Either mixed through the lecture, or at the end of lecture, groups of students will have time to work on in-class quizes based on old test questions. Answers to these in-class quizes will be submitted at and graded the group level. There will be about 20 minutes of time in each class dedicated to this.

## Meeting Times/Place

**Times:** Tuesday, Thursday 12:30–13:50
**Place:** Liberal Arts Building 201

## Final Exam Time and Place

**Time:** 8:00-10:00, Friday, May 3
**Place:** Liberal Arts Building 201

## Grading Policy

### Grading scale

| | |
|---|---|
| A | 94-100 |
| A- | 90-93 |
| B+ | 87-89 |
| B | 83-86 |
| B- | 80-82 |
| C+ | 77-79 |
| C | 73-76 |
| C- | 72-70 |
| D+ | 67-69 |
| D | 63-76 |
| D- | 60-62 |
| F | 0-59 |

Students achieving the numerical scores above are guaranteed the associated letter grade. However, if average performance is low, I may decide to assign a higher letter grade for a lower score; e.g. a B+ for a numerical score of 84.

Students taking the course pass/no pass are required to earn a grade of D or better in order to pass.

### Assessments and weights

The following assessments will be used and weighted according to the values in the table to determine final grades.

| Component | Description | Weight |
|---|---|---|
| In-class problems | Problems worked on the board, by individual students. A rubric of assessment appears on the course web site. | 20% |
| Group work | Assessment of individual student performance at the board will be given to each member of the group the student is in. This will be given equal wieght in an average with the in-class quizes to give the final group work grade. | 20% |
| Midterm I | Test of your knowledge of material presented in class and projects. Inclusive of material presented since first day of class. | 15% |
| Midterm II | Test of your knowledge of material presented in class and projects. Inclusive of material presented after midterm I. | 15% |
| Final Exam | Test of your knowledge of all material presented in class and projects. Inclusive of all material. | 30% |

## Tentative schedule:

| TUESDAY | THURSDAY |
|---|---|
| Jan 8th **1** | 10th **2**<br>Course introduction and demonstration of tools, Introduction to Hardware Description Language (HDL), logic gates |
| 15th **3**<br>Combinational logic and the ALU (Arithmetic-Logic Unit) | 17th **4**<br>Combinational logic and the ALU (Arithmetic-Logic Unit) |
| 22nd **5**<br>Sequential logic: memory hierarchy | 24th **6**<br>Sequential logic: flip-flop gates, registers, and RAM |
| 29th **7**<br>Machine language: instruction set, assembly and binary versions | 31st **8**<br>Machine language: assembly language programs |
| Feb 5th **9**<br>Computer architecture I | 7th **10**<br>Computer architecture II |
| 12th **11**<br>Assembler: language translation - parsing and symbol table | 14th **12**<br>Assembler: language translation - macro-assembly and construction of assembler |
| 19th **13**<br>Virtual machine I: modern virtual machines, stack based arithmetic, logical and memory access operations | 21st **14**<br>Virtual machine I: implementation of a VM from assembler language previously developed |
| 26th **15**<br>Virtual machine II: stack-based flow-of-control and subroutine call-and-return techniques, complete VM implementation | 28th **16**<br>High level language: introduce *Jack*, a simple high level language with Java like syntax |
| Mar 5th **17**<br>High level language: trade-offs in language design and a simple, interactive game in *Jack* | 7th **18**<br>**Midterm Exam I** |
| 12th **19**<br>Compiler I: context-free grammars and recursive parsing algorithms, building a tokenizer and parser for *Jack*. | 14th **20**<br>Compiler I: syntax analyzer and XML output |

| TUESDAY | THURSDAY |
|---|---|
| 19th **21**<br>Compiler II: code generations, low-level handling of arrays and objects | 21st **22**<br>Compiler II: a full-scale compiler, generating VM code from XML produced previous week |
| 26th<br>*Spring Break* | 28th<br>*Spring Break* |
| Apr 2nd **23**<br>Operating system: design of OS/hardware and OS/software with regard to time/space efficiency of design | 4th **24**<br>Operating system: classic algorithms in OS design |
| 9th **25**<br>ARM Architecture I | 11th **26**<br>ARM Architecture II |
| 16th **27**<br>ARM Architecture III | 18th **28**<br>ARM Architecture IV |
| 23rd **29**<br>Wrap up/Course evaluation | 25th **30**<br>**Midterm Exam II** |
| 30th **31**<br>Finals Week | May 2nd **32**<br>Finals Week |

## Attendance Policy

Attendance will not be taken. Students absent when called up to work problems on the board will be given a grade of 0%. Another team member will be selected to go to the board at random. *The team will not be punished for the unexcused absense of a member.* Students informing the instructor of a valid reason for missing class *in advance*, via email, will not be called to the board. Valid reasons include family emergencies and illness. I may ask for documentation of absence (doctors note, death certificate, etc.). Cell phone photos are very useful for this - a selfie in the doctors office, or next to a car that won't start...

## Academic Integrity

All students must practice academic honesty. Academic misconduct is subject to an academic penalty by the course instructor and/or a disciplinary sanction by the University. All students need to be familiar with the Student Conduct Code. I will follow the guidelines given there. In cases of academic dishonesy, I will seek out the maximum allowable penalty. If you have questions about which behaviors are acceptable, especially regarding use of code found on the internet or shared by your peers, please ask me.

## Disabilities

Students with disabilities may request reasonable modifications by contacting me. The University of Montana assures equal access to instruction through collaboration between students with disabilities, instructors, and Disability Services for Students. Reasonable means the University permits no fundamental alterations of academic standards or retroactive modifications.