Computer Simulation and Modeling
CSCI 361
Spring 2017 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:	Jesse Johnson
Office:	406A Interdisciplinary Science Building
Telephone:	(406) 243-2356
Email:	jesse.johnson@umontana.edu
Web:	http://hs.umt.edu/hs/faculty-list/faculty-details.php?id=540
Office Hours:	T 15:00–16:00 and Th 10:00-12:00 , Interdisciplinary Science Building 406A *Or, by appointment.*

Prerequisites

Students taking this course are expected to have:

- Programming experience demonstrated by passing CSCI136 or a similar course.
- Organizational skills and familiarity with computers sufficient to install new software and create a filesystem for the course.
- 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.

Textbook

This semester I'll be using the following text book. You'll need to purchase a copy at the UM Bookstore, or online.
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.

Software

This course uses simulators to test the design of your hardware. They are written in Java, so run on Windows, OSX, and Linux. The software should be downloaded and configured according to the instructions here.

Methodology

This is mostly a hands-on course, evolving around building a series of hardware and software modules. Each module development task 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. Each lecture will start by reviewing the work that was done thus far, and giving guidelines on what to do next. The projects can be done in pairs.

Meeting Times/Place

Times: Monday, Wednesday 15:00–16:20
Place: Social Science 362

Final Exam Time and Place

15:20–17:20 Tuesday, May 9, 2017
Social Science 362

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-	70-82
D+	67-69
D	63-76
D-	60-62
F	0-59

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**	**Number**	**Weight**
In-class problems	Problems worked on in the classroom, by the instructor and groups of students. $\frac{1}{3}$ of the grade will be attendance, $\frac{1}{3}$ assessment from classmates, and $\frac{1}{3}$ correctness of solution. All group members may submit the same work for these assignments.	12	30%
Homework	Assessment of individual student performance on the assigned problems. These are to be worked by students outside of the classroom. Students are encouraged to discuss solutions with their groups, but each submitted assignment must represent the student's own work and be unique.	6	40 %
Midterm Exam	Test of your knowledge of material presented in class and done in homework.	1	10%
Final Exam	Test of your knowledge of all material presented in class and done in homework.	1	20%

Tentative schedule:

MONDAY	WEDNESDAY
Jan 23rd **1** Course introduction and demonstration of tools	25th **2** Introduction to Hardware Description Language (HDL), logic gates
30th **3** Combinational logic and the ALU (Arithmetic-Logic Unit)	Feb 1st **4** Combinational logic and the ALU (Arithmetic-Logic Unit)
6th **5** Sequential logic: memory hierarchy	8th **6** Sequential logic: flip-flop gates, registers, and RAM
13th **7** Machine language: instruction set, assembly and binary versions	15th **8** Machine language: assembly language programs
20th *Presidents Day*	22nd **9** Computer architecture: integrations of chipsets built thus far in class
27th **10** Computer architecture: integrations of chipsets built thus far in class	Mar 1st **11** Assembler: language translation - parsing and symbol table

MONDAY	WEDNESDAY
6th **12** Assembler: language translation - macro-assembly and construction of assembler	8th **13** **Midterm Exam**
13th **14** Virtual machine I: modern virtual machines, stack based arithmetic, logical and memory access operations	15th **15** Virtual machine I: implementation of a VM from assembler language previously developed
20th *Spring Break*	22nd *Spring Break*
27th **16** Virtual machine II: stack-based flow-of-control and subroutine call-and-return techniques, complete VM implementation	29th **17** High level language: introduce *Jack*, a simple high level language with Java like syntax
Apr 3rd **18** High level language: trade-offs in language design and a simple, interactive game in *Jack*	5th **19** Compiler I: context-free grammars and recursive parsing algorithms, building a tokenizer and parser for *Jack*.
10th **20** Compiler I: syntax analyzer and XML output	12th **21** Compiler II: code generations, low-level handling of arrays and objects
17th **22** Compiler II: a full-scale compiler, generating VM code from XML produced previous week	19th **23** Operating system: design of OS/hardware and OS/software with regard to time/space efficiency of design
24th **24** Operating system: classic algorithms in OS design	26th **25** More fun to go: improvement of our computer in terms of optimization and functional extensions
May 1st **26** More fun to go: HTTP servers, FPGA implementations	3rd **27** Wrap up/Course evaluation

Attendance Policy

Attendance is required and enters your grade as part of the in class assessment (10% of total grade). The policy for excusing absences is identical to that of late assignments.

Late Assignments

Other than in in exceptional circumstances, such as family or medical emergencies *late homework will not be accepted* unless an extension was agreed upon *well in advance* of the due date. All excpetional circumstances must be documented in writing.

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.