Syllabus

Instructor:	(Jonathan) Sumner Evans (jonathanevans@mines.edu)
TA:	Adam Sandstedt (asandstedt@mymail.mines.edu)
Course Advisor:	Dr. Bo Wu (bwu@mines.edu)[1]
Lecture:	Mondays and Wednesdays, 16:30 to 17:45 in GC MET
Zoom:	https://mines.zoom.us/j/91249307693?pwd=VTFqYkw5SEoyeE9TTEJqeFZjcFY5QT09 (Password: 856310)
Office Hours:	See the Office Hours page on the course website for a listing of the current schedule or email the instructor to schedule an appointment.
Course Website:	https://sumnerevans.com/teaching/csci564-s21
Matrix Chat:	https://matrix.to/#aca-s21:sumnerevans.com
Piazza:	https://piazza.com/mines/spring2021/csci564
Prerequisites:	CSCI 341 Computer Organization (or similar). Additionally, you should be able to program in C or C++.

Course Description & Learning Objectives

CSCI 564 is a course covering **advanced topics in computer architecture, including pipelining, caching, virtual memory, storage, multi-core processing and power**. We will focus on understanding the various options available to computer architects when designing computer systems, with an emphasis on developing quantitative justifications for those options.

In addition to being invaluable for computer architects, this topic is highly relevant for both future software engineers and computer scientists. Software engineers rely on performance gains made in the field of computer architecture, and understanding the fundamentals of concepts such as the memory hierarchy helps guide software engineers' optimization strategies. Additionally, concepts such as caching are widespread in industry (albeit mainly at a higher level than the caches we will discuss). Future computer scientists will benefit from this class because it quantifies many core concepts of the underlying hardware of all modern computing systems.

Expectations

It is **highly recommended** that you use Linux for all of the projects in this course. If you choose to use a different type of system (such as macOS or Windows), the instructor nor the TAs will not be able to assist you with any platform-specific issues.

Students are also expected to have be able to write either C or C++ before entering this course. *Neither the instructor nor the TA will help debug your code for you.*

Textbooks and Other Resources

- **Required:** Computer Architecture: A Quantitative Approach, Fifth Edition, John L. Hennessy and David A. Patterson.
- *Recommended:* Synthesis Lectures on Computer Architecture. Link: http://www.morganclaypool.com/toc/cac/1/1
- The instructor may also require and/or recommend readings from other online sources

[1] You probably won't ever need to contact the course advisor, but if you have concerns you wish not to bring to the attention of the instructor (e.g., anonymous concerns), you are welcome to contact the advisor.

Topics

This is a tentative list of topics that we will discuss in class.

Performance Metrics	Out-of-Order Execution
Amdahl's Law	Main Memory
Cache	SIMD
Cache Optimizations	Multiprocessors
Virtual Memory	Consistency and Coherency
Pipelining	Interconnection
Handling Branches	

Grading Policy

This course is worth 1000 points. The points are allocated as follows:

1. **Midterm.** 100 points.
2. **Final.** 200 points.
3. **Three Projects.** Total 400 points.
    - Project 1: Cache Replacement Policies Simulator. 150 points.
    - Project 2: Cache Prefetcher Simulator. 100 points.
    - Project 3: Branch Prediction Simulator. 150 points.
4. **Four Homework Assignments.** Total 200 points. 50 points each.
5. **Class Participation (Mainly Worksheets).** 100 points.

Your grade letter will be derived using the **standard plus-minus grading scale**. The instructor reserves the right to move the scale down, but it will never move up.

All projects will be due at 23:59 of the day listed on the schedule. The penalty for late submissions is 20% per day. If you need an extension for any reason, email the instructor to work something out.

Autograded Projects

Most programming assignments have an auto-graded component where a portion of your grade is calculated immediately upon submission. The grading scripts are designed to mark all fully correct programs as correct, and do the best job possible assigning partial credit where applicable. However, under some cases, you may not receive as many points as you deserve. If you believe this to be the case, contact the course instructor so they can either fix the autograder to handle your case better, or manually grade your assignment.

You are only allowed 3 submissions on autograded homework. This is because your grade is received quickly after submitting, and it may be tempting to use the autograder as the only means of testing. Students are expected to test their own code before submitting. If you run out of submissions and need more, then contact the instructor.

Warning

Autograded assigments are reviewed for plagarism a few weeks after the deadline, not when graded. While copying an assigment from another student may give you a good grade in the short term, the long term consequences could be severe. Plagarism is taken very seriously on these assignments.

Collaboration Policy for Programming Projects in CS Courses

The following policy exists for all courses in the CS department. This policy is a minimum standard; your instructor may decide to augment this policy.

1. If the project is an individual effort project, you are not allowed to give code you have developed to another student or use code provided by another student. If the project is a group project, you are only allowed to share code with your group members.

2. You are encouraged to discuss programming projects with other students in the class, as long as the following rules are followed:

   a) You view another student's code only for the purpose of offering or receiving debugging assistance. Students can only give advice on what problems to look for; they cannot debug your code for you. **All changes to your code must be made by you.**

   b) Your discussion is subject to the **empty hands policy**, which means you leave the discussion without any record (electronic, mechanical, or otherwise) of the discussion.

3. Any material from any outside source such as books, projects, and in particular, from the Web, should be properly referenced and should only be used if specifically allowed for the assignment.

4. To prevent unintended sharing, any code stored in a hosted repository (e.g., on GitHub) must be private. For group projects, your team members may, of course, be collaborators.

5. If you are aware of students violating this policy, you are encouraged to inform the professor of the course. Violating this policy will be treated as an academic misconduct for all students involved. See the Student Handbook for details on academic dishonesty.

Disability Support

The Colorado School of Mines is committed to ensuring the full participation of all students in its programs, including students with disabilities. If you are registered with Disability Support Services (DSS) and your instructor has received your letter of accommodations, please contact your instructor at your earliest convenience so you can discuss your needs in this course. For questions or other inquiries regarding disabilities, we encourage you to visit Disability Support Services (DSS) for more information.