Course Syllabus

Professor

Dr. Jonathan Geisler

Contact: (765) 998-5269 and jgeisler@cse.taylor.edu, but best found on the CSE Slack.

Hours

Lecture: MWF 9:00 - 9:50 in Euler 200

Office: M 13:00 - 15:00, T 14:00 - 16:00, WF 8:00 - 9:00, R 15:00 - 17:00 in Euler Room 212

I'm in my office often at times other than indicated above. Feel free to stop by or make an appointment. I'll be happy to work with you on a time that fits your schedule, too.

Course Description

A study of the hardware structure of computer systems, including arithmetic/logic units, memory organization, control unit design, pipelining, and instruction set design. A brief introduction to advanced topics, such as out-of-order execution, branch prediction, multi-core systems, and parallel processing cache coherency will prepare the student

for graduate level courses in architecture. Prerequisites: COS 284.

Course Outcomes

By the end of the course, the student should ...

1. be comfortable with assembly programming
2. understand the internal structure of a modern CPU
3. be able to justify performance optimizations based on a specific architecture
4. have a basic understanding of what future CPU architectures may be like
5. understand various addressing modes used by assembly instructions
6. explain the fetch-decode-execute cycle of a von Neumann architecture
7. understand and explain internal binary representations of MIPS instructions
8. explain the basic operation of an assembler
9. thoroughly understand instruction-level pipelining
10. explain design choices between various instruction set architectures
11. thoroughly explain how a cache works
12. describe the memory hierarchy and explain why it is necessary in modern processors
13. describe the underlying philosophies of RISC architectures

14. be familiar with common fallacies and pitfalls that computer architects encounter
15. describe the importance of the instruction set architecture to both the hardware architect and software application developer
16. have detailed knowledge of the arithmetic and logic unit of a modern processor
17. identify the driving design principles (simplicity favors regularity, smaller is faster, make the common case fast, and good design demands good compromises) and when they apply to real-life designs
18. explain what a pseudoinstruction is and why they exist
19. build a single-cycle MIPS-lite simulator
20. build a pipelined MIPS-lite simulator
21. describe how a machine performs integer multiplication
22. describe how a machine performs integer division
23. describe how a machine performs floating-point arithmetic
24. explain how performance is best measured on computer applications. This includes how the number of instructions, CPI, and clock speed are involved
25. explain how the PC drives the flow of instructions through the CPU
26. explain how a register file is organized and used
27. explain the role of control logic in a CPU
28. explain the difference between sequential and combinational logic

29. have a basic understanding of how the hardware handles interrupts and exceptions
30. describe how to handle data hazards
31. describe how to handle control hazards
32. describe how to handle structural hazards
33. motivate and describe delay slots for load and branch instructions
34. describe when a pipeline stage must be stalled or killed
35. describe how branch prediction works
36. describe how out-of-order execution works
37. describe how speculative execution works
38. describe how superscalar processors work
39. describe how superpipelined processors work
40. describe how register renaming works
41. state the "3 C's" of cache misses
42. explain how computer architects deal with writes to caches

Texts

Computer Organization and Design RISC-V Edition: The Hardware/Software Interface, 2nd edition, by David Patterson and John Hennessy. ISBN 978-0128203316.

You will be expected to keep up with the reading by completing a set of questions for each course period. In general, the reading will be due prior to our discussion of the material so that we can cover anything that is confusing from

the reading or so that we can go into greater depth than the book covers.

Content

This course covers the foundations of how a computer is designed. We will look at modern CPU designs to drive our understanding of important issues. Additionally, we will construct a model of a simple CPU using C++ to understand how modern hardware is designed.

Assessment

Lab assignments	50 %
Final lab project	10 %
Reading assignments	10 %
Exams (2 total)	20 %
Final	10 %
Total	100 %

The final grade will be assigned on the following scale:

		≥ 93	A	≥ 90	A-
≥ 87	B+	≥ 83	B	≥ 80	B-
≥ 77	C+	≥ 73	C	≥ 70	C-
≥ 67	D+	≥ 63	D	≥ 60	D-
< 60	F				

Attendance

Attendance is expected and can alter the course grade. Taylor policy regarding excused absences is followed. It is the student's responsibility to verify that the professor has been notified of excused absences for any reason, including official university functions. The due date of an assignment will NOT be automatically extended as a result of an excused absence.

Three unexcused absences will be permitted without penalty. A penalty of 3% will be deducted from the final grade for each additional unexcused absence. Excessive tardiness may be counted as absence.

Late Work

All assignments are due before class starts. Assignments will be submitted via Canvas or Gitlab, which leaves a timestamp that you cannot modify. No late work will be accepted. If you know you will miss a test due to an excused absence, you must contact me ahead of time to schedule a make-up session.

Labs

There are no structured laboratory sections for this class. It is your responsibility to ensure you are able to complete the

assignments before they are due. Use the CSE labs or any other resource that is convenient for you. The labs will consist of two parts. The first part is a set of near-weekly assignments to develop portions of a CPU simulator. Before the end of the course, the assignments can be combined to create a fully functional simulator. The second part is an extended project the expand the CPU simulator in some new, interesting way.

Examples of the final project could include developing:

- A branch predictor
- An out-of-order execution unit
- A multi-level cache system
- A port to a different Instruction Set Archecture (e.g., x86, ARM, MIPS, etc.)
- Exceptional Condition handling (e.g., system calls, traps, etc.) with user/kernel mode

Exams

Tests

The tests will be a mixture of multiple choice, essay, design, or other appropriate question format. You should be able to demonstrate your knowledge of the subject using terminology and techniques covered in class and/or the textbook.

There is not enough time in class to cover everything in the book. The tests will cover more material than we go over in class; all the assigned readings will be fodder for questions on the test. It is your responsibility to work with the instructor to ensure you understand any material that you need more help with.

Final

The final exam will be similar to the other tests, but will be longer and comprehensive; however, the final exam will emphasize the material covered since the last test. The final exam will be 1:00 PM, Wednesday, December 15 in class. Students must take their final examinations at the assigned time. Exceptions will be made only because of serious illness or the death of an immediate member of the family. Reasons such as plane schedules, availability of flights, and rides leaving early are not acceptable exceptions. Students having three or more examinations in one day should report this to the Registrar's Office ten days prior to the beginning of finals week. Reasonable alternatives in alleviating this dilemma will be pursued by the registrar and the student in consultation with the appropriate faculty.

Cheating

The standard departmental cheating policy is expected to be

followed.

Taylor University Plagiarism Statement

Definition: In an instructional setting, plagiarism occurs when a person presents or turns in work that includes someone else's ideas, language, or other (not common-knowledge[1]) material without giving appropriate credit to the source. Plagiarism will not be tolerated and may result in failing this course, and may also result in further consequences as stipulated in the Taylor catalog: https://public.taylor.edu/academics/files/undergrad-catalog/current/policies.pdf.

Academic dishonesty constitutes a serious violation of academic integrity and scholarship standards at Taylor that can result in substantial penalties, at the sole discretion of the University, including but not limited to denial of credit in a course as well as dismissal from the University. ... In short, a student violates academic integrity when he or she claims credit for any work not his or her own (*words, ideas, answers, data, program codes, music,* etc.) or when a student misrepresents any academic performance.

[1] **Common knowledge** means any knowledge or facts that could be found in multiple places or as defined by a discipline, department, or faculty member.

Laptops in the Classroom

Laptops will be permitted in the classroom provided that they are not a distraction to you or other students. The purpose of the laptop should be to enhance your learning. This means you should be taking notes, or searching for information related to class. It also means that you shouldn't be on Facebook, sending email to friends, or working on homework in other classes.

Academic Resources

- Academic Enrichment Center: aecenter@taylor.edu
- Peer tutoring services: drnurkkala@taylor.edu
- Writing center: julie_moore2@taylor.edu or http://public.taylor.edu/academics/academic-support/writing-center.shtml
- Zondervan Library: zonlib@taylor.edu
- Disability support services: lswallace@taylor.edu