CS 3432: Computer Organization
Spring 2025
Syllabus

Title:	CS 3432: Computer Organization, CRN 26954 (CRN 26962 for the LAB)
Term:	Spring 2025
Duration:	Tue 01/21/2025 through Thu 05/08/2025
Lecture Time:	Tue, Thu 4:30PM to 5:50PM
Lecture Location:	EDUC 112
Lab Time:	Tue, Thu 6:00PM to 7:20PM
Lab Location:	CCSB 1.0704
Instructor:	Dr. Christoph Lauter `cqlauter@utep.edu` +19157475939 (office) +19152963175 (urgent cases, text message only)
Instructors Office Hours and Location:	Email anytime, Phone/Teams during business hours, Office (CCSB 3.0610) Mon-Thu 10:00AM-11:50AM
Class Web page:	`https://www.christoph-lauter.org/utep-arc/`

Prerequisites:

- To succeed in this course, students need familiarity and maturity with concepts and techniques taught in digital design (EE 2369/2169), elementary data structures and algorithms (CS 2401), and discrete math (Math 2300).
- As stipulated in the course catalog, the usual way to demonstrate this familiarity is by earning a C or above in (1) all of these courses and (2) Data Structures (CS 2302).
- Students who earn Bs or above in EE 2369/2169, CS 2401, and Math 2300 and are taking CS 2302 concurrently, are also considered ready.

Lab:

- Students must enroll in the associated lab section.
- Lab sessions and assignments will be managed by the TA.
- Participation is mandatory. It is extremely easy to fall behind and imperative that you make arrangements with the instructor or TA to make-up missed lessons and work. Students at risk of failing due to not engaging with the lab section may be dropped.
- There will be scheduled and unscheduled quizzes in the lab section.

Description:

CS 3432 is a first systems course about how computer systems work. Whereas computer architecture describes the functional behavior of a computer, computer organization describes the structural relationships. CS 3432 is a course in computer organization that answers questions such as the following:

1. How are programs written in a high-level language, such as C or Java, translated into the language of the hardware, and how does the hardware execute the resulting program?
2. What is the interface between the software and the hardware, and how does software instruct the hardware to perform the necessary functions?
3. What determines the performance of a program, and how can hardware designers or programmers improve the performance?
4. What are the reasons for and the consequences of the recent switch from sequential to parallel processing?

Student Learning Outcomes:

See appendix below.

Textbooks:

Required:

- David A. Patterson and John L. Hennessy. Computer Organization and Design RISC-V Edition: The Hardware Software Interface, Second edition, Morgan Kaufmann, ISBN: 978-0128203316.
- Kernighan, Brian, Ritchie, Dennis M. The C Programming Language, Second edition, Prentice Hall, ISBN: 0-13-115817-1.

Optional:

- Other readings and resources will be posted on the course webpage.

Homework Assignments:

- Students are expected to review topics taught in class, work on solutions to assigned problems, and be able to demonstrate skills and solutions during class. Homework assignments will be posted on the course website. Answer will be posted for selected problems.
- To obtain full credit for the homework grade, you will need to present solutions to all 3 homework problems.

Workload & Attendance: This course requires participation during the in-class lectures, in addition to the daily readings and the homeworks. Students must read the textbook as well as the posted lecture slides before the corresponding class. Attendance in the lecture will be checked. This course requires student committment to understanding low-level concepts and complex data-structures. Students are encouraged to favor simple and working solutions over complex and incomplete approaches.
Please form study groups. You are encouraged to discuss the approach and understand the problem. However, the write up, programming, and actual solutions to the homeworks are individual work. If you use someone's work for your own, you are committing plagiarism.

Readings: Students are expected to read and understand the textbook chapters before these chapters are discussed in the lecture.

Computers: Students need access to a computer that allows them using a native Linux environment or that allows them `ssh` access to a Linux server, provided by the instructor. For homework assignments that compile in one environment and not the other, the result obtained on the system provided by the instructor will be deemed the canonical result. Risc-V development boards will be distributed to students; these kits need to be returned at the end of the semester.

Slides, Script: No scholae sed vitae discimus. This class is praxis-oriented; we will code in class and discuss all pieces of hardware needed for a computer in class. Slides are hindering our comprehension, as they only give us the impression that we understood things. The instructor will hence refrain from presenting slides. There is no official script. **Make your own script by taking extensive notes.** The instructor's handwriting is not faster than yours, so you can essentially copy everything from the blackboard.

Tests and Testing Policy: Three tests will be given: two mid-term tests and a final. All tests are cumulative, with an emphasis on recent material. No make up tests are given but for documented medical emergencies.

Grading: Letter grade:

Points range	Grade
$[90; 100]$	A
$[80; 90)$	B
$[70; 80)$	C
$[60; 70)$	D
$[0; 60)$	F

Grading breakdown:

Lab	50%
Homework	5%
Midterm I	10%
Midterm II	10%
Final exam	20%
Attendance	5%

Make-Ups: Make-ups are not allowed. Make-up work will be given only in the case of a documented emergency. Note that make-up work may be in a different format than the original work, may require more intensive preparation, and may be graded with penalty points. If you miss an assignment and the reason is not considered excusable, you will receive a zero. It is therefore important to reach out to the instructor **before** the respective exam starts, resp. **before** the homework assignment deadline is up, and explain with proper official documentation why you missed a given course requirement. Once a deadline has been established for make-up work, no further extensions or exceptions will be granted.

Approximate Course Schedule: The course schedule of topics and assignments will be posted on the course website.

Accomodations: The University is committed to providing reasonable accommodations to students with documented disabilities. Students who become pregnant may also request reasonable accommodations, in accordance with state and federal laws and regulations and University policy. Accommodations that constitute undue hardship are not reasonable. To make a request, please register with the UTEP Center for Accommodations and Support Services (CASS). Contact CASS at 915-747-5148, email them at `cass@utep.edu`, or apply for accommodations online via the CASS portal.

Cell phones: Please silence your mobile devices or put them into a vibrate mode for the duration of class - they are disruptive for your fellow students. No mobile devices (cell phones, PDAs, laptops etc.) are allowed during the exams and will result in your expulsion from the test.

Class recordings: The instructor will not take the burden of recording the classes. If the group of students wants to self-organize and record the classes, the instructor is all in favor of this solution. However, in this case, students must abide by federal and state law governing in-class recordings, in particular FERPA, as well as with UTEP policies. When in doubt of the legality of making a recording available to other students in this class, students should first consult with the instructor. **FERPA prevents in-class recordings to be made publicly available.** Students may not share recordings outside of this course. Doing so may result in disciplinary action and legal consequences. The instructor will of course abide by the same laws and will respect your privacy. This means he will not and cannot share class recordings with people not enrolled in the class or TAs/IAs.

Scholastic Integrity:

- Academic dishonesty is prohibited and is considered a violation of the UTEP Handbook of Operating Procedures. It includes, but is not limited to, cheating, plagiarism, and collusion. Cheating may involve copying from or providing information to another student, possessing unauthorized materials during a test, or falsifying research data on laboratory reports. Plagiarism occurs when someone intentionally or knowingly represents the words or ideas of another as ones' own. Collusion involves collaborating with another person to commit any academically dishonest act. Any act of academic dishonesty attempted by a UTEP student is unacceptable and will not be tolerated. All suspected violations of academic integrity at The University of Texas at El Paso must be reported to the Office of Community Standards (`https://www.utep.edu/student-affairs/standards/`) for possible disciplinary action. To learn more, please visit HOOP: Student Conduct and Discipline (`https://www.utep.edu/hoop/section-2/student-conduct-and-discipline.html`).

- Submitted work should be unmistakably your own. You may not transcribe or copy a solution taken from another person, book, or other source (e.g., a web page). Professors are required to report academic dishonesty and any other violation of the Standards of Conduct to the Dean of Students.

- Some AI technologies or automated tools, particularly generative AI such as ChatGPT or DALL-E, can be beneficial during the early brainstorming stages of an activity, and you are welcome to explore them for that purpose. However, keep in mind that AI-generated ideas are not your own and may hinder your ability to think critically and creatively about a problem. It is also important to remember that these technologies often "hallucinate" or produce materials and information that are inaccurate or incomplete—even providing false citations for use. That said, you are not allowed to submit any AI-generated work in this course as your own. If you use any information or materials created by AI technology, you are required to cite it like you would any other source. Consider how this will affect your credibility as a writer and scholar before doing so. Any direct use of AI-generated materials submitted as your own work will be treated as plagiarism and reported to the Office of Community Standards.

- Permitted collaboration: Students may discuss requirements, background information, test sets, solution strategies, and the output of their programs. However, implementations and documentation must be their own creative work. Students are required to document advice received from others and all resources utilized in the preparation of their assignments.
- If academic dishonesty is suspected: The Dean of Students office will be contacted for adjudication. A temporary "incomplete" grade will be issued if their investigation extends beyond the grading period.

Appendix: CS 3432 Learning Outcomes

Level 1:

- A1. Define and explain the purpose of an instruction set architecture (ISA).
- HSI2. Explain the relationship and differences between a high-level programming language, assembly language, and machine language.
- HSI3. Describe the fetch-execute cycle in terms of the hardware-software interface between machine instructions and processor components.
- NR1. Explain the relationship between a high-level language basic data type (e.g., signed or unsigned integer, floating point number) and its representation as a bit pattern inside the computer.
- A2. Describe the basic components of a processor (e.g., register file, special-purpose registers, control unit, memory) and how they interact with one another.
- HSI4. Explain how procedures are supported by processor hardware.
- HSI5. Explain exception/interrupt handling in terms of the hardware-software interface.
- HSI6. Explain various ways an operand can be addressed in an assembly language instruction.
- HSI7. Describe the process of compiling/assembling, linking, loading, and executing a program.

Level 2:

- NR2. Convert between different integer data representations (e.g., decimal, binary, hexadecimal, octal).
- NR3. Interpret the bit representation of a floating-point number.
- NR4. Perform addition and subtraction on two's complement representation of integers.
- NR5. Use bitwise operators to access and manipulate values stored in a subset of bits within a byte or word.
- NR6. Determine range and precision (if applicable) of numbers that can be stored for a given data type and determine whether an integer operation will result in overflow.
- HSI8. Convert between machine and assembly language representations of instructions – i.e., encode and decode instructions.
- HSI9. Trace the datapath through the processor for a given class of instructions (e.g., arithmetic-logical, memory access, conditional branch)
- HSI10. Trace the execution of an assembly language program with procedure calls in terms of allocation and deallocation of stack frames.

- L1. Translate expressions and assignment statements from C to assembly language.
- L2. Translate Boolean logic and control flow constructs (decisions, loops) from C to assembly language.
- L3. Translate operations on arrays, structs, and pointers from C to assembly language.

Level 3:

- HSI11. Implement/debug simple imperative programs in assembly/machine language.
- HSI12. Write or call a procedure with local variables, parameters, and return value in assembly language.
- HSI13. Implement a simple interrupt handler.
- T1. Compose, compile/assemble, execute, and debug simple programs in a command-line environment, using appropriate modularization and multiple files.