UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
CS 433: Computer System Organization
http://courses.engr.illinois.edu/cs433/
Prof. Christopher Fletcher
Spring 2019

Syllabus

Instructor: Prof. Christopher Fletcher, email: cwfletch@illinois.edu, office: 4106 SC, office hours: Wed 3-4pm or by appointment.

Software Support: We will use the Piazza newsgroup (see link on course home page). We will use the website `http://courses.engr.illinois.edu/cs433/` to access the course slides, handouts, homeworks, etc.

Class: Tu and Th, 2:00PM - 3:15PM in 1109 Siebel Center.

Prerequisites: CS 233 (Computer Architecture) or ECE 390.

Credit: Graduate students: 1 unit; undergraduate students: 0.75 units. Graduate students are asked to do extra work in each assignment. Graduate and undergraduate students are graded using two different curves.

Text: "Computer Architecture: A Quantitative Approach," **sixth** edition, by Hennessy and Patterson. We will follow this book in class. To get another perspective on the issues, you can read "Parallel Computer Organization and Design" by Dubois, Annavaram, and Stenstrom, or "Modern Processor Design: Fundamentals of Superscalar Processors" by Shen and Lipasti.

Format: I plan to use slides in class. You can access the slides from the course web site.

Assignments: There will be six (6) written homeworks. See the schedule for assignment turn-in dates. All assignments are due at the beginning of class the day they are due and will be turned in on paper. Assignments will include extra questions for graduate students only (although undergraduates should check the solutions). Note that the assignments have only a modest weight toward the final grade. However, doing the assignments is a good way to prepare for the exams.

Exams: There will be a midterm and a final exam. The final is comprehensive. Both exams are closed book and closed notes. Bring a calculator. The midterm will be in class (see the schedule).

Grading: Final grades will be computed based on the following: midterm 35%, final 40%, homeworks 25%. Requests for regrading will be accepted up to one week after the return of graded assignments. Such requests must be accompanied with a written justification for why the score should change. No late assignments will be accepted without prior arrangement. Typical grade distributions in the past have been: Graduate students (A 45%, B 50%), Undergraduate students (A 35%, B 35%). I give letter grades with + or − as well. Note that these distributions may change.

Collaboration: For assignments, it is legitimate for students to discuss possible interpretations of the assignment. However, once you understand the assignment and have begun formulating a solution, collaboration must cease. If students are found to have collaborated excessively, all involved will receive a grade of 0 for that assignment (this counts as cheating, see below). There is no collaboration on exams, and the collaboration policy on assignments is meant to help prepare you for the exams.

Cheating: Cheating will be dealt with harshly. We will follow the policy on cheating given at `http://www-bsac.eecs.berkeley.edu/~pister/etc/Cheating.htm`.

Goal of the Course: This is an advanced course in computer architecture that moves fast. You should not fall behind in your Hennessy and Patterson readings because you will find it difficult to catch up. It is assumed that you already understand the basics of computer architecture, including:

1. Computer elements (logic, FSM, memory)
2. Assembly language programming (addressing, control transfers, procedure linkage)
3. Pipelines and pipelined instruction execution (instruction fetch, decode, execute, store, etc.)
4. Basic machine organization (datapath, control, buses, memory interface, interrupts)

If you feel that your background is lacking in any of these areas, please consult an introductory computer architecture textbook such as "Computer Organization and Design: The Hardware / Software Interface," by Patterson and Hennessy.

Topics in the Course:

1. Introduction: review of fundamental performance issues, power and reliability, cost vs. price, basic pipeline structure.
2. Instruction level parallelism: hardware and software techniques (e.g., dynamic scheduling, superscalar, static and dynamic branch prediction, VLIW, loop unrolling).
3. Memory hierarchy: advanced concepts in caches, main memory, and virtual memory.
4. Multiprocessors/multicore: overview of different models, cache coherence with shared-memory systems/multicore (snoopy and directory solutions), synchronization, memory consistency models.
5. Data parallel architectures: vectors, SIMD, GPUs.
6. Special topics (time permitting): domain-specific accelerators (DSAs) for deep neural networks, warehouse-scale computing, hardware security, storage systems/IO.