Spring 2012 CSE/EEE 420

Computer Architecture I

Class Meetings
TT, 1:30 pm - 2:45 pm, BYAX 190
Instructor:
Aviral Shrivastava
Office- BY 408
Website
Email
Office Hours – TT 12:30pm – 1:30pm, BY 408

Teaching Assistant:
Jing Lu
Website
Email
Office Hours: MW 1:00pm - 2:00pm, BY 407CA

Textbooks:

There are 2 textbooks for this course:

   4th Edition revised printing
   Authors - David A. Patterson and John L. Hennessy
   Publisher - Morgan Kaufman, now Elsevier
   ISBN – 978-0-12-374750-1

   4th Edition
   Authors - Hennessey and Patterson
   Publisher - Elsevier
   ISBN – 978-0-12-383872-8

Course Abstract

"It (Computer Architecture) is not a deary science of paper machines that will never work. No! It’s a discipline of keen intellectual interest, requiring balance of marketplace forces to cost-performance-power, leading to glorious failures and some notable successes." - John Hennessy and David Patterson.
This course will start with bridging the gap between the high-level programming languages (e.g. C/C++, Java, perl, python) that we so conveniently use, and the low-level electronic components (e.g. transistors, AND gate, OR gate, multiplexors).

This Computer Architecture course then delves into details of designing a processor. The class is divided into 4 modules, i) A simple processor design, ii) Pipelining and ILP, iii) Memory Organization iv) Multi-core processors. Starting from a simple design, we will strive to make it better. We will learn several fundamental techniques, like pipelining, caching, and parallel execution, which enable the modern computing-based world. Although this course takes the MIPS architecture as a vehicle to explain the complexities and trade-offs in computer architecture, the concepts are applicable in a much broader scope. We will take a hands-on approach to understanding computer architecture. We'll develop simulators to model and thereby understand computer architecture.

Course Learning Outcomes

Gain an understanding of the fundamentals of computer architecture. After doing this course, students will be able to:

- Understand the role and importance of assembly language.
- Understand implementation of a simple pipelined processor.
- Understand the promise, and effects of pipelining. Students will be able to know the basic challenges of pipelining, and will have fundamental knowledge on how to resolve those challenges and achieve high performance with pipelining.
- Understand branch prediction, and bypassing.
- Understand that there are limits to parallelism, and just increasing the number of ALUs will not increase processor performance.
- Understand the concept of caches, and know the fundamental mechanism of how they work, and why they are so effective.
- Understand the need, importance and basic implementation of virtual memory.
- Appreciate the need for multi-core processors.
- Have a general idea of the landscape of multi-core processors.
- Realize the programming challenges brought forth by multi-core processors.

Course Timeline

- ISA and simple Implementation -- 3 weeks
- Pipelining and ILP -- 4 weeks
- Memory Management -- 3 weeks
- Multi-core processors -- 4 weeks

Course Structure

Classroom: The lectures will be fast paced, and may cover stuff not in the text book. Thus I strongly recommend to attend all the lectures and that you scan the sections before coming to the class, and follow up after the class.

Quarter-terms: There will be 4 scheduled and announced quarter-terms in this course. The quarter-terms will be on Thursdays near the end of each module in the class. Unless otherwise specified, all exams in this class are open notes, books, calculators, computers and internet. Just do not talk to any other human during the exam.

Homework Projects: 4 programming projects will be given in the course. A lot of programming will be involved in the course. The first project requires programming in MIPS assembly, and the rest can be done in C. Homework projects will be done in groups of 2. Under exception, you can make a group of 3.

Final: There will be one Final, in the same format as the quarter-terms.

Extra Project: If you are on track to a D grade or less after module 3, then you can contact me regarding extra projects that can demonstrate your learning towards this course. I will be happy to work with you to define such projects for extra grades.

Scoring Policy

- 4 Quarter terms: Each 10% Total 40%
- 4 Homework Projects: Each 10% Total 40%
• 1 Final: Total 20%

Rescore Requests
Rescore requests can be made within a week of receiving your graded exams. After that it is too late.

Grading Policy

> 90% -- A  
> 80% -- B  
> 70% -- C  
< 60% -- E

Course Announcements
All announcements will be made at the beginning of the class, and/or on the course blackboard. Any handouts will be distributed at the beginning of the class, and will be available on the course web site. It is your responsibility not to miss any announcements made in the class and/or on the CSE 420 blackboard.

Course Withdrawal Policy
University policies will be adhered to for course withdrawal. There will be no exceptions to this course withdrawal policy.

Cheating Policy
If you are not already familiar with the school’s cheating and plagiarism policies and procedures, read the document on Academic Integrity here.
Ignorance is not an excuse!

Formula for Success in CSE 420
• Scan lecture material before class.
• Annotate textbook with notes during discussions.
• Review lecture material after the class.
• Meet with TA to resolve any discrepancies promptly.

Who to see for help
Questions pertaining to the lectures: Instructor.  
Questions regarding suggested homework problems: TA.  
Rescoring requests: TA.  
Errors or corrections on posted scores: TA
Motivation (1 week)
1. Welcome
2. Field of computer architecture
3. Multicore introduction

Module 1: ISA to Implementation (2 weeks)
1. MIPS ISA
2. Function Calls
3. Assembly
4. Single Cycle Implementation

Module 2: Pipelining
1. Pipelining and Reservation Tables
2. Structural and Data Hazards
3. Control Hazards
4. Branch Prediction

Module 3: Memory Hierarchy
1. Memory
2. Direct Mapped Cache
3. Set Associative Caches
4. Virtual Memories
5. Inclusive Caches

Module 4: Superscalar Processors
1. In-order Scheduling
2. Out-of-order Scheduling
3. Speculative Execution

Module 5: Multi-threading and Multi-cores
1. Multi-threading
2. IBM Cell processor architecture and programming
3. Cache Consistency and Coherency
4. How GPUs work