15-346: Seminar in Perspectives in Computer Architecture

Syllabus

School of Computer Science
Carnegie Mellon University, Qatar
Spring 2013

1 Overview

Title: Seminar in Perspectives in Computer Architecture
Units: 6 units
Pre-requisites: A grade of "C" or better in 15-213, Introduction to Computer Systems
Lectures: Monday and Wednesday, 8:30 – 9:50 AM, Room 1190
Webpage: http://www.qatar.cmu.edu/~msakr/15346-s13/

Description:

This course will provide various perspectives in the field of computer architecture by world renowned scientists. The course will bring together basic architecture principles and designs of uniprocessor and multicore computers. First, we will introduce concepts, notations, attributes and computational elements emerged over several thousands of years. Computational elements pre and post 1900 (e.g., memory, processing and I/O elements of Babbage's Analytic Engine, Turing and Von Neumann machines) will be compared and contrasted. Second, we will identify factors that contributed to the many-fold improvements in computer performance, size, and power consumption over the last 60 years. Students will further recognize the architectural challenges imposed by the 21's century emerging applications and unfold the reasons behind the recent slowdown in performance, size and power improvements. Third, we will present few basic architectural techniques including instruction level parallelism, pipelining and memory hierarchy. An overview of multicore architectures, specifically on how they differ from uniprocessor ones, the promises they offer, and the serious challenges they pose, will also be provided. Lastly, we will survey a range of architecture types from digestible (e.g., sensors) to the largest and fastest (e.g., Clouds and supercomputers), how they came across and which to choose when. As a case use, we will focus on mobile systems, their different deployments, usage models, and challenges. The concepts delivered in the lectures will be reinforced and extended through student presentations on multiple directions in computer architecture.

Instructors:

Gordon Bell
gbell@microsoft.com Room TBD, Phone TBD, Office Hours: Wednesday 11AM-12PM

Mohammad Hammoud
mhhammou@qatar.cmu.edu, 1013, 4454-8506, Office Hours: Thursday, 11AM – 12PM

Raj Reddy
rr@cmu.edu Room TBD, Phone TBD, Office Hours: Wednesday 11AM-12PM

Majd F. Sakr
msakr@qatar.cmu.edu, 1016, 4454-8625, Office hours: Tuesday, 3:00 – 4:00 PM

Daniel P. Siewiorek
dps@cs.cmu.edu By Skype from CMU Pittsburgh Thursday 4PM-5PM

Chuck Thacker
cthacker@microsoft.com Room TBD, Phone TBD, Office Hours: Wednesday 11AM-12PM

2 Course Objectives

Computer architecture is the science and art of selecting and interconnecting hardware components to produce a system that meets functional, performance and cost goals. Similar to the architecture of a building that describes only its functionality and appearance, the architecture of a computer system reflects only the functionality and appearance of the system, with no exposure to implementation details (i.e., the actual embodiment of the architecture).

Our aim in this course is six-fold:

1. Introduce you to the area of computer architecture.
2. Provide various perspectives on the origins of computers, pre 1900 and post 1900, as well as on the computing trajectory over two centuries, the 20th and the 21st, with a special focus on the computer architecture angle.
3. Learn some conventional and crucial uniprocessor architectural techniques and principles that define computer architecture.
4. Examine new architectural paradigms such as multicore architectures, how they differ from uniprocessor ones, and what challenges they pose on computer architects.
5. Survey various computer architecture designs for specific domains, ranging from embedded systems, through a single HPC computer, to Cloud and supercomputers.
6. Introduce mobile hardware technologies, review development and deployment strategies for mobile systems, and highlight some research challenges and lessons learned over years.

Through these objectives the course will give you a good understanding of the computer architecture domain as well as various perspectives on its past, present and future designs, usages and directions. This will enable you at the end of the course to write a technical paper-like survey about one of the presented directions that interested you.

3 Learning Outcomes

The high-level learning outcomes of the course are as follows:

1. Students will review the emergence of computational aids, tools, mechanical calculators, electro-mechanical calculators over the millennia and relatively recent arrival stored program electronic computer and identify which device contributed to what features currently now in use in modern systems.

2. Students will evaluate the factors that led to exponential improvement in speed, size and power consumption over the last 60 years and explore avenues for continued improvements in the 21st century.

3. Students will discuss Instruction Level Parallelism (ILP), apply pipelining as a mechanism

to exploit ILP, identify structural, data and control hazards in pipelined systems, and identify the difference between super-scalar and VLIW systems.

4. Students will illustrate memory hierarchy as a strategy to bridge the CPU-Memory speed gap, recognize the cost-performance trade-offs of using multiple levels of caches, describe cache misses, placement and replacement policies, and identify the difference between virtual and physical systems.

5. Students will outline the difference between uniprocessor and multicore architectures, and identify major multicore challenges such as distributed cache organization, network on-chip and synchronization.

6. Students will examine the structures and functions of a wide range of computer architectures varying from digestible and wearable computers to the largest and fastest peta-scale computers, whereby the application seems to define the final form and function of the architecture.

7. Students will study mobile computer architectures. Mobile hardware and software tend to be application specific requiring careful reengineering where aesthetics and ease of use dominate the design considerations.

3.1 Origins of Computers – Pre 1900

Concepts, notations, attributes and elements of computation emerged over several thousand years. The emergence of representation of numbers and arithmetic operations on numbers was followed by attempts to build mechanical tools to perform these operations. Concepts for input and output of data, logical expressions for precise definition of operations had to be invented. This unit will discuss the contributions to advancement of computing concepts by Pascal, Liebniz, Boole, Babbage and Lovelace over a 300 year period prior to 1900CE.

In this unit the student will:

- Discuss the mechanical computational elements of designs of Pascal and Liebniz and their strengths and limitations compared modern calculators.
- Understand the structure of memory, processing and I/O elements of Babbage's Analytic Engine and compare with corresponding elements of modern computers.
- Analyze Lovelace's program for Babbage's Analytic Engine and compare it with current programming conventions.

3.2 Origins of Computers – Post 1900

The foundations of modern stored program computers emerged out of the work of Turing and von Neumann. They in turn were influenced by the formalisms and results of Hilbert and Gödel. During the World War II, several electro-mechanical (relay) and electronic (vacuum tube) computers were independently built by George Stibitz, Konrad Zuse, John Atanasoff and Howard Aiken which become the fore-runners of modern computing revolution. John von Neumann's EDVAC report of 1945 synthesized disparate concepts from Turing, Aiken, Eckert and Mauchly, and many others to create the standard computer architecture which is universally used and called von Neumann architecture. In this unit we will review these efforts and identify elements of their architecture that became standard elements of future systems.

In particular, in this unit the student will:

- Discuss the computational elements of the Turing Machine and why it is called Universal, in that any computation done by any other computer can also be done on the Turing Machine.
- Create a comparative Chart showing presence or absence of various computing elements that are deemed to be essential in modern computers.
- Present the factors that led to the demise of punch cards and paper tape in modern computers.

3.3 Computer Architecture in the 20th Century:

During the last century, computers emerged from a few post-war laboratories to become an indispensable part of all our lives. Driven by advances in the underlying technologies used in their design, the performance and storage capacity of computers grew by many orders of magnitude, while their size shrank from machines that occupied an entire building to devices that we carry in our pockets. In this unit, we will examine some of the underlying technologies that have been employed in computing, and describe some of the architectural ideas they made possible. We will discuss the relevance of some of these ideas to today's architectures, especially with the continual rapid progress in technology.

In this unit the student will:

- Identify factors that contribute to a billion fold improvement in computer performance over the past 60 years.
- Identify factors that contribute to a 10 million fold improvement in size reduction over the past 60 years.
- Identify factors that contribute to a billion fold reduction in power consumption of computing elements over the past 60 years.

3.4 Computer Architecture in the 21st Century:

During the last century, computers became pervasive as their performance and usability rapidly improved. Today, much of the improvement that we saw during the first sixty years of computing has slowed or stopped, as we reach limits imposed by physical law. In this unit, we will examine the reasons for this slowdown, and present some of the challenges facing the computer architects of the 21st century. We no longer have the luxury of a constantly-improving technology base, and must employ new ways to continue to improve the effectiveness and usefulness of our computers. We will discuss some of the challenges we face as architects, and discuss some potential solutions.

In this unit the student will:

- Discuss the factors leading to slowdown in exponential improvement in computing devices.
- Identify new avenues for improving price/performance of future computing devices.
- Propose research projects that would preserve exponential improvement in computer performance for the next 60 years.

3.5 Instruction Level Parallelism and Pipelining:

Parallelism can be applied at the instruction level in advanced microprocessors whereby a CPU can begin executing a second instruction before the first one is completed. All instructions can be run in parallel on a unique pipeline, with each being at a different processing stage. Such a mechanism is usually referred to as Instruction Level Parallelism (ILP) and serves in improving CPU performance. Students will learn how performance can be improved using ILP. They will examine instruction pipelines, such as the classic RISC pipeline. Specifically, after finishing this unit students will be able to:

- Identify the different steps of instruction execution such as fetch, decode, execute, memory and writeback.
- Discuss how parallelism can be exploited at the instruction level with the same circuitry for improved performance.
- Apply pipelining as a mechanism to exploit ILP.
- Identify structural, data and control hazards in pipelined systems.
- Identify super-scalar and VLIW systems and recognize the primary difference between them.

3.6 Memory Hierarchy:

In addition to ILP and pipelining, performance of uniprocessor machines can be improved using many other architectural techniques such as memory hierarchy. A memory system is a hierarchy of storage devices with different capacities, costs, and access times. Memory hierarchies can improve performance by bridging the CPU-Memory speed gap. In this unit, students will:

- Define the CPU-Memory speed gap and discuss how memory hierarchy can serve in bridging this gap.
- Illustrate the memory hierarchy and recognize the cost-performance trade-offs of using multiple levels of caches and memory.
- Describe different cache organizations, identify cache miss types and examine placement, replacement, write and inclusion policies.
- Identify virtual memory systems and recognize the difference between virtual and physical memories.

3.7 Multicore Chips and Challenges:

In the past, architects could rely on speeding up a uniprocessor machine in order to enable increasingly complex software to be built and run. Those days are gone. Nowadays, multicore machines are ubiquitous, present in all mainstream architectures for servers, desktops, and embedded systems. In this unit students will describe multicore architectures, recognize how they differ from uniprocessor ones, identify the promises they offer, and discuss the serious challenges they introduce. Specifically, after finishing this unit, students will be able to:

- Outline the difference between uniprocessor and multicore architectures.
- Identify main challenges in designing multicore systems such as cache organization, network on-chip and synchronization.

3.8 Computer Architectures from Digestible to the Largest and Fastest:

This unit will look at various kinds of computers, how they got that way, and attempt to answer the question—which to choose or use?

Computers have evolved from functional bifurcation of calculation and record keeping to controlling other system interfaces, communications and switching. Every information carrier, from greeting cards to phones, is an embedded computer. Similarly, physical carriers from networks to autos can be viewed as computers with wheels, UIs, memory, etc. In each area the choice may be cost, power, reliability, specialization for the task, and the like. Computer structures will include the following:

- Cloud and supercomputers for high performance computing are almost indistinguishable. They are both scalable and constructed from many low cost computing nodes interconnected to each other and to storage (disks). A proposed triple crown measures their Top500, Graph500, and Green500 rankings. Cray Research, Fujitsu, IBM, as well as China all compete for these titles. In 2012 a single HPC computer operates at almost 20 peta-flops. The goal of exa-flops has been recently moved to after 2020.
- Communications and network computers are another special breed. Within this category, companies compete for the ability to execute stock trades electronic with the lowest latency. One company Zeptronics claims a delay of less than 10 ns.
- Game computers have even better performance/price than supercomputers. Graphics Processing Units were introduced in games and these have become a component of HPC. What makes each special?
- Wireless Sensor Networks Industrial control, building automation, and scientific and engineering data acquisition usually boils down to interconnecting a variety of highly distributed sensors and effectors to some central or distributed computers for control or record keeping. Wireless Sensor Networks provide reliable communication in the face of electrical noise, sharing communication channels with Bluetooth, 802.11x, wireless phones, etc.
- Ingestible and in-body computers from pill-cams to pacemakers. For instance, one of the smallest computers, which is designed to remain in the eye for measuring pressure, is a 1 mm cube that includes sensing, battery, and telemetry.

In this unit the student will:

- Identify factors in different applications that dictate architecture decisions that contribute to a billion fold improvement in computer performance over the past 60 years.
- Describe the structure and function and features of 1mm cube computer with sensing and battery and telemetry.
- Discuss the factors that lead to reliable communications in Wireless Sensor Networks in the face of electrical noise, sharing communication channels with Bluetooth, 802.11x, wireless phones, etc.

3.9 Futuristic Designs: Mobile Hardware Technologies:

A confluence of technologies including miniature electronics, multi-modal interaction such as speech and gesture recognition, digital communications, and machine learning makes possible the creation of mobile, highly interactive intelligent assistants that monitor and communicate with users, understand their needs and goals, and augment their capabilities. This unit will highlight the research challenges and lessons learned from deploying mobile systems. Several example mobile systems will be presented.

In this unit the student will:

- Categorize system goals.
- Identify mobile system usage models.
- Recognize various combinations of sensors.
- Understand the profile of energy usage in mobile systems.

- Appreciate the central role of the end user.

3.10 End-to-End Mobile Application Considerations:

We will review over 15 years of experience in developing and deploying mobile systems in domains as diverse as manufacturing, maintenance, operations, museums, forest fires, volcanoes, and medical hospitals leading to lessons that are not taught in the classroom. These experiences are summarized in categories for easier recall: User, Corporal, Attention, Manipulation, and Power (UCAMP).

In this unit the student will:

- Identify hardware and software issues to be addressed in creating and deploying complete systems that service a public need.
- Utilize several check lists that can be used to anticipate deployment challenges.

4 Textbooks

The primary textbooks for this course are:

- David A. Patterson and John L. Hennessy, "**Computer Organization and Design: The Hardware/Software Interface**", Fourth Edition, Morgan Kaufmann, 2011.
- John L. Hennessy and David A. Patterson, "**Computer Architecture: A Quantitative Approach**", Fifth Edition, Morgan Kaufmann, 2011.
- Siewiorek, Bell and Newell, "**Computer Structures**", McGraw-Hill Inc.,US Jan. 1, 1971.

In addition, we recommend the following textbooks:

- Randal E. Bryant and David R. O'Hallaron, "**Computer Systems: A Programmer's Perspective**"*,* Prentice Hall, 2003.
- Jack Copeland, "**The Essential Turing**", Oxford University Press, USA, Nov. 18, 2004.
- Martin Davis, "**The Universal Computer**", W. W. Norton & Company, October 2000.
- George Dyson, "**Darwin among the Machines**", Basic Books; First Trade Paper Edition, October 8, 1998.
- Freeman Dyson, "**Origins of Life**", Cambridge University Press; 2nd, Sep. 28, 1999.

We have several reference books in the library covering most of the topics of the course. We will also be reading tutorials, journals and conference publications on the subject.

5 Course Organization

Students' participation in the course will involve three forms of activities:

- Attending lectures and participating in class discussions
- A topic-specific survey paper
- Final presentations at the end of the semester

6 Getting Help

For urgent communication with the teaching staff, it is best to send an email (preferred) or phone. If you want to talk to an instructor in person, please remember that our posted office hours are merely nominal times when we guarantee that we will be in our offices. You are always welcome to visit us outside of our office hours if you need help or want to talk about the course. For visiting faculty or faculty teaching remotely, it is best to send email.

We will use the course web-page as the central repository for all information about the class. Using the web-page, you can:

1. Obtain copies of any handouts. This is especially useful if you miss a class.
2. Find links to any electronic data you need for studying material covered in the lectures.
3. Read clarifications and changes made to any schedules and policies.
4. Provide healthy feedback about the course.

You can use a mailing list specific for the class (instructions on this mailing list will be provided in class and on the course webpage) to post messages and make queries about the course. The messages on this mailing list will be distributed to all the students and the instructors of the course.

7 Appealing Grades Policy

After you are assigned a grade on your paper and final presentation, you have seven calendar days to appeal your grade. All your appeals should be provided in writing. If you are still not satisfied, please come and visit Prof. Sakr.

8 Assessment

Each student will receive a numeric score for the course, based on a weighted average of the following:

1. **Final Presentation:** Each student will be delivering a presentation at the end of the semester on a specific computer architecture topic that she/he chooses in coordination with one of the teaching instructors. The presentation will count for a total of 30% of your score.

2. **A Survey Paper:** Each student will write a technical paper-like document (not more than 12 pages) about the topic that she/he will be presenting at the end of the semester. The document has to *survey* the current and past work on the selected and presented topic. In addition, the document will include the expected future trend(s) concerning the topic. The document will count for 60% of the grade.

3. **Class Participation and Attendance:** Your attendance and participation in the different discussions held in the class will account towards 10% of your final grade.

Type	#	Weight
Final Presentation	1	30%
Survey Paper	1	60%
Class Participation and Attendance	14	10%

Grades for the course will be determined by absolute standards. The total score will be plotted as a histogram. Cutoff points are determined by examining the quality of work by students on the borderlines. Individual cases, especially those near the cutoff points may be adjusted upward or downward based on factors such as attendance, class participation, improvement observed throughout the course, and special circumstances.

9 Cheating and Plagiarism

Each presentation and survey paper must be the sole work of the student turning it in. The following are guidelines on what is authorized and what is not:

What is cheating?

1. Sharing electronic files: either by copying, retyping, looking at, or supplying a copy of a file.
2. Submission of work that is not the student's own.
3. Sharing written surveys: Looking at, copying, or supplying a survey.

What is Plagiarism?

Plagiarism includes, but is not limited to, failure to indicate the source with quotation marks or footnotes where appropriate if any of the following are reproduced in the work submitted by a student:

1. A written phrase.
2. A graphic element.
3. Specific language.
4. An idea derived from the work, published or unpublished, of another person.

What is NOT cheating?

4. Clarifying ambiguities or vague points in class handouts.
5. Helping others use the computer systems, networks, compilers, debuggers, profilers, or other system facilities.
6. Helping others with high-level conceptual issues.

Cheating and plagiarism in survey papers will be strictly monitored and penalized. Be aware of what constitutes cheating (and what does not) while interacting with your colleague students. You cannot share or use written documents and other electronic material. If you are unsure, ask the teaching staff. Be sure to store your work in protected directories. The penalty for cheating is severe, and might jeopardize your career – cheating is not worth the trouble. By cheating in the course, you are cheating yourself; the worst outcome of cheating is missing an opportunity to learn. In addition, you will be removed from the course with a failing grade. We also place a record of the incident in the student’s permanent record.

10 Class Schedule

Table 1 shows the tentative schedule for the class. An updated schedule will be maintained on the class webpage.

Week	Session	Date	Topic	Teaching Method	Instructor
1	1	14 Jan	Origins of Computers- Part I	Lecture	Raj Reddy
	2	16 Jan	Origins of Computers- Part II	Lecture	Raj Reddy
2	3	21 Jan	Computer Architecture in the 20th Century	Lecture	Chuck Thacker
	4	23 Jan	Computer Architecture in the 21st Century	Lecture	Chuck Thacker
3	5	28 Jan	Instruction Level Parallelism and Pipelining- part I	Lecture	Majd Sakr
	6	30 Jan	Instruction Level Parallelism and Pipelining- Past II	Lecture	Majd Sakr
4	7	4 Feb	The Memory Hierarchy	Lecture	Mohammad Hammoud
	8	6 Feb	Multicore Chips: Introduction and Challenges	Lecture	Mohammad Hammoud
5	9	11 Feb	A Walk-Through Computer Architectures from Digestible to the Largest and Fastest- Part I	Lecture	Gordon Bell
	10	13 Feb	A Walk-Through Computer Architectures from Digestible to the Largest and Fastest- Part I	Lecture	Gordon Bell
6	11	18 Feb	Mobile Hardware Technologies	Lecture	Daniel P Siewiorek
	12	20 Feb	End-to-End Application Considerations	Lecture	Daniel P. Siewiorek
7	13	25 Feb	Student Presentations	Presentations	All
	14	27 Feb	Student Presentations	Presentations	All

Table 1: Tentative Time-Line of the 15-346 Course.