CIS 216: Computer Organization and Design

Scott D. Anderson

Spring 1999

Instructor

My office hours are Monday 1:00–3:30, Wednesday 3:00–5:00, and Thursday 2:30–4:00. You can also meet me by appointment. Contact me by email at `anderson@spelman.edu` or by phone at 223-7617. My office is room 201B in the Academic Computer Center. My web page is `http://www.spelman.edu/~anderson/`.

Policies

My policies regarding academic dishonesty, attendance, late assignments, grading standards, and so forth are listed on my web page. You are expected to read them and understand them.

Description and Prerequisites

An introduction to the theoretical foundations of computer science with emphasis on topics such as finite state machines, formal languages, boolean logic and combinatorics. Also a study of the vision of the field of computer science with exposure to the breadth of the basic areas of study. Prerequisite: CIS 122 or consent of the department.

Purpose

This course explains the basic workings of a computer, from the stored-program concept and the fetch-execute cycle, to the representation of code and data. By discussing the Instruction Set Architecture, it bridges the gap between hardware and software, particularly the operating system.

This course will also provide the students with an understanding of combinational and sequential logic gates and the internal operation of modern computer hardware. The emphasis is on the functional logic design level, and the mechanics of data transfer and control, rather than circuit details.

Objectives

At a very primitive level, computers work by executing machine language: programs represented in binary. In this course, we will study the more readable counterpart of machine language called assembly language. We will also look at the hardware implementation of assembly language, building up from logic gates to flip-flops, registers, and, ultimately, to complete computer architectures. We will also see how the architecture is designed for usability, efficiency, and speed.

Specifically, by the end of the course, you will be able to

- describe how data structures are represented in binary—objects such as numbers, character strings, and records,
- give a high-level view of computer architecture (how the hardware works),
- state the basic principles of digital logic (combinational and sequential) and computer design.
- describe the design of the ALU and the processor, datapath and control
- explain register transfer and micro-operations
- sketch the organization of the central processing unit, input/output devices and memory of modern computers
- describe the general architecture of MIPS architecture, with emphasis on pipelining
- explain cache and virtual memory techniques
- define the premise of advanced architectures, such as SIMD and MIMD.

This course prepares students for more advanced courses such as compiler design, parallel processing and computer architecture.

Instruction

Instruction will consist mostly of lectures based on the text. We will use SPIM, a simulator of the MIPS architecture, for executing and debugging programs in MIPS assembly language. There will required laboratory sessions, meeting in the ACC Sun lab. These lab sessions will cover either assembly language programming in SPIM or digital logic using pencilboxes or a software simulator.

There will be at least five assembly language programming assignments during the semester. Assembly language programming will appear on the tests.

The material to be covered in each lecture is given below. Students are expected to have read the assigned material prior to class.

Textbook

Computer Organization and Design: The Hardware/Software Interface by David A. Patterson and John L. Hennessy, published by Morgan Kaufmann, 1994, ISBN 0-55860-281-X.

Other references that you may find helpful:

- *Computer Systems Design and Architecture*, by Vincent P. Heuring and Harry F. Jordan, pub. by Addison-Wesley Longman, 1997, ISBN 0-8053-4330-X.

- *Computer Organization and Architecture: Designing for Performance*, by William Stallings. Fourth edition, pub. by Simon and Schuster, 1996, ISBN 0-13-359985-X.

- *Computer Architecture*, by Robert J. Baron and Lee Higbie, pub. by Addison-Wesley Longman, 1992, ISBN 0-201-50923-7.

Requirements

There will be three in-class tests (about one a month) and a final exam. The dates of the in-class exams are tentatively scheduled below. The tests are cumulative; they will cover all material given in the lectures up to the test date. The final exam will cover all course material. There will be no make-up exams except for excused absences where I have been notified in advance.

The preceding components of your grade are weighted as follows:

Component	percent
in-class tests	30
final exam	20
programming assignments	30
homework	20

Course Outline

Students are expected to come to class having read the material for the week. The following reading endpoints are *inclusive*; in other words, if it says read 2.1–2.4, that means you must read all of section 2.4.

Week	Reading	Topics
1	Chapter 1	Computer Abstractions and Technology
2	2.1 – 2.4	Performance Metrics
	B.1 – B.3	lab reading
3	2.5 – end	Performance Metrics
	B.4 – B.5	lab reading
4	3.1 – 3.5	Simple Arithmetic Instructions
	A.1 – A.4	lab reading
5	3.6 – end	**test 1** Branch and Jump Instructions
	A.5 – A.6	lab reading
6	4.1 – 4.4	Computer Arithmetic
7	4.5 – 4.7	Multiplication and Division
8	5.1 – 5.3	Single Cycle Datapaths
9	5.4 – 5.5	Multi-cycle Datapaths **test 2**
		Spring Break
10	5.6 – end	Exceptions
11	6.1 – 6.6	Pipelining **test3**
12	7.1 – 7.4	Memory Hierarchy
13	7.4 – end	Memory Hierarchy
14	8.1 – end	Input and Output devices
15	9.1 – 9.3	SIMD and MIMD computers
16		Catchup and Review