Brooklyn College
Department of Computer and Information Science

CISC 3310 [27.1] Principles of Computer Architecture
4 hours; 4 credits

Introduction to digital logic. Basic digital circuits. Boolean algebra and combinational logic, data representation and transfer, digital arithmetic. Instruction sets. Introduction to assembly languages: ALU and memory reference instructions, flow control, subroutine linkage, arrays and structures. Memory. I/O systems. Performance. Relationship between software and architecture. (Not open to students who are enrolled in or have completed Computer and Information Science 3305 [27] or 3315 [28].)

Objectives

This course will be a required course for majors and is the only required course in which students study computer hardware: approximately half the course deals with how computer hardware is designed, and the other half deals with implications of hardware design for programmers and users.

By the end of this course students should be able to:

1. Understand computer data representations and arithmetic operations.
2. Understand computer instruction sets and assembly language.
3. Understand central processing unit organization and operation.
4. Understand computer memory systems.
5. Understand computer input/output operations and interfacing.

Syllabus

Week 1 Introduction. Boolean algebra. Truth tables. Logic gates. Combinatorial circuits (basic gates, multiplexors, decoders, adders)

Week 2 Combinatorial circuits. Sequential circuits (flip-flops, registers). Clocks and the basic clock cycle.

Week 3 Number representations (binary, decimal, octal, and hexadecimal). Integers (ones complement, twos complement, sign and magnitude, biased). Character representations (ASCII, Unicode).

Week 4 Floating point numbers. Error detection and correction. CPU basics (fetch/execute cycle, data path, control unit)

Week 5 Instructions and formats: arithmetic operations, logical operations, shift operations, instructions, conditional instructions, call and return.

Week 6 Addressing modes (PC relative, indirect, indexed). Alternative architectures. Pipelining (conflicts, branch predictions, superscalar processors, out-of-order execution).

Week 7 Review and midterm.

Week 8 Introduction to assembly language. High- and low-level languages. Assemblers. Pseudo commands. Basic instructions (ALU instructions, memory reference instructions). Variables, registers and memory.

Week 9 Flow control (jumps, comparison, and conditional branches). Subroutine linkage (call, return, argument passing, local variables).

Week 10 Arrays and structures. Memory hierarchy. Locality. Cache.

Week 11 Virtual memory. Secondary storage (disks). I/O systems: buses; device control.

Week 12 I/O systems: Interrupts; DMA; disks (RAID); data compression.

Week 13 Performance measures. Alternative architectures. RISC. Flynn's taxonomy. Parallel architectures.

Week 14 Operating systems. Compilation. Interpreters and virtual machines. Review for final.

Bibliography:

Anthony Dos Reis: *Assembly Language and Computer Architecture Using C++ and Java*, Thomson Course Technology, Boston, MA., 2004.

Karen Miller: An Assembly Language Introduction to Computer Architecture, *Oxford UniversityPress, 1999.*

John L. Hennessy, David A. Patterson: Computer Architecture, Fourth Edition: A Quantitative Approach, Morgan Kaufman, San Francisco, CA., 2006.

William B. Jones: Assembly Programming: Programming for the IBM PC Family, 3d ed., Scott/Jones Inc., El Segundo, CA, 2001.

Kip Irvine: Assembly Language for Intel-Based Computers, 5th ed., Prentice-Hall, Saddle River, NJ, 2006.