Computer Science 324
Computer Architecture
Mount Holyoke College
Fall 2009

Topic Notes: Introduction and Overview

Welcome to Computer Architecture!

Why take Computer Architecture?

To build upon your knowledge of how computers are designed and built, right from the low-level details of the electrical circuits up to the assembly language code generated by compilers.

You have seen the idea of an abstract data type.

This idea of *abstraction* is key in this course as well. Think of the levels of abstraction of a computer:

Level 0: Physics and Digital Logic

- properties of atoms, electrons, and quantum dynamics
- semiconductors/silicon used to build transistors
- transistors used to build logic gates (CMOS)
- Boolean logic
- combinational logic, arithmetic circuits
- sequential logic, finite state machines
- architectural issues: *i.e.*, caches, virtual memory, pipelining

Level 1: Conventional Machine Language

- CPUs, vonNeumann architecture
- most of the processors we think about operate at this level
- we will consider MIPS, other modern examples include IA-32, PowerPC, Sparc, ARM
- programmed in “machine code” – just a bunch of 0’s and 1’s
- the *instruction set architecture*: capabilities of the machine

Level 2: Assembly Language

- human readable/writable description, closely related to machine language
- translated to machine language by an *assembler*

Level 3: Operating System

- runs programs on the hardware
- manages sharing of resources
- *e.g.*, Mac OS X, Linux, FreeBSD, Windows, etc.

Level 4: Conventional Programming Langauages

- applications software – the stuff people actually use
- *compilers* translate this code into assembly, which the assembler in turn translates to machine code to run on the computer
- make use of services provided by the operating system
- *e.g.*, C, C++, Fortran

Not everything fits in directly here.

Level 3.5: Virtual Machines

- an abstract "machine" that runs programs in a higher-level "architecture"
- VM is in turn implemented to execute in the traditional environment
- *e.g.*, JVM, .NET

Level 4.5: Interpreted Languages

Java, BASIC, Python, Perl

To understand what's going on inside the computer, you need to understand all of these levels.

Our focus is on the lowest levels.

- Digital logic
- Microarchitecture and Instruction set architecture
- Assembly language programming
- Higher-level computer architecture topics (parallelism)