CS1101: Lecture 9

Contemporary Multilevel Machines

Dr. Barry O'Sullivan
b.osullivan@cs.ucc.ie

Course Homepage
http://www.cs.ucc.ie/~osullb/cs1101

Lecture Outline

- A Multi-Level Machine
- A Six-Level Computer
  - Digital Logic Level
  - Microarchitecture Level
  - Instruction Set Architecture Level
  - Operating System Machine Level
  - Between Levels 3 and 4
  - Assembly Language Level
  - Problem-oriented Language Level
- Computer Architecture
- **Reading**: Tanenbaum, Chapter 1.

A Multi-Level Machine

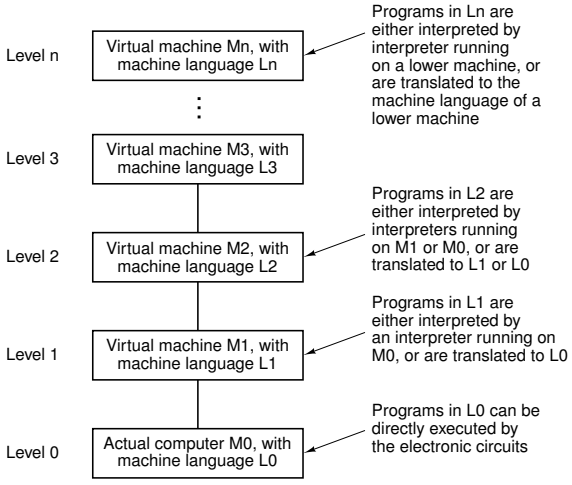

Figure 1.1 A multi-level machine

A Six-Level Computer

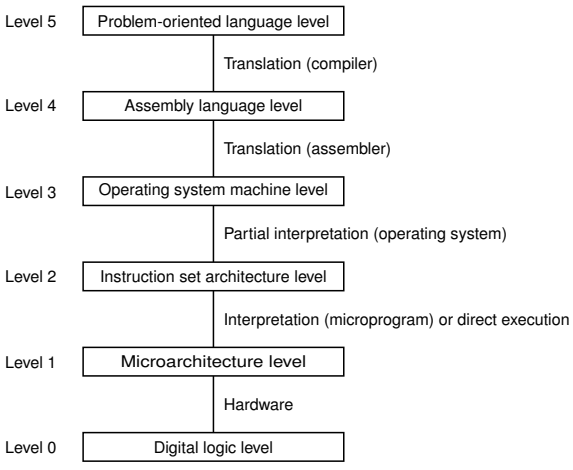

Figure 1.1 A six-level computer

Digital Logic Level

- The interesting objects at this level are **gates**;
- Each gate has one or more digital inputs (signals representing a 0 or 1) and computes as output some simple function of these inputs, such as AND or OR;
- Each gate is built of at most a handful of transistors
- A small number of gates can be combined to form a 1-bit memory, whcih can store a 0 or 1;
- The 1-bit memories can be combined in groups of, for example, 16, 32 or 64 to form **registers**
- Each register can hold a single binary number up to some maximum;
- Gates can also be combined to form the main computing engine itself.

Microarchitecture Level

- At this level we see a collection of (typically) 8-32 registers that form a local memory and a circuit called an **ALU (Arithmetic Logic Unit)** that is capable of performing simple arithmetic operations;
- The registers are connected to the ALU to form a **data path** over which the data flow;
- The basic operation of the data path consists of selecting one or two registers having the ALU operate on them;
- For example, adding them together, with the result being stored back in some register;
- On some machines the operation of the data path is controlled by a program called a **microprogram**, on other machine it is controlled by hardware.

Instruction Set Architecture Level

- The ISA level is defined by the machine's instruction set
- This is a set of instructions carried out interpretively by the microprogram or hardware execution sets;

Operating System Machine Level

- At this level there is a different memory organisation, a new set of instructions, the ability to run one or more programs concurrently, and various other features;
- The new facilities added at level 3 are carried out by an interpreter running at level 2, which, historically, has been called an operating system (OS);
- Those level 3 instructions identical to level 2's are carried out directly by the microprogram (or hardwired control), not by the OS;
- In other words, some of the level 3 instructions are interpreted by the OS and some of the level 3 instructions are interpreted directly by the microprogram;
- This the OS level is a hybrid.

Between Levels 3 and 4

- The lower 3 levels are not for the average programmer – Instead they are primarily for running the interpreters and translators needed to support the higher levels;
- These are written by **system programmers** who specialise in developing new virtual machines;
- Levels 4 and above are intended for the **applications programmer**
- Levels 2 and 3 are always interpreted, Levels 4 and above are usually, but not always, supported by translation;
- Languages at Levels 1, 2 and 3 are generaly numeric – long series of numbers – from Level 4 and higher, the languages contain words and abbreviations meaningful to humans.

Assembly Language Level

- This level is really a symbolic form for the one of the underlying languages;
- This level provides a method for people to write programs for levels 1, 2 and 3 in a form that is not as unpleasant as the virtual machine languages themselves;
- Programs in assembly language are first translated to level 1, 2 or 3 language and then interpreted by the appropriate virtual or actual machine;
- The program that performs the translation is called an **assembler**.

Problem-oriented Language Level

- This level usually consists of languages designed to be used by applications programmers;
- These languages are generally called **higher level languages**
- Some examples: Java, C, BASIC, LISP, Prolog;
- Programs written in these languages are generally translated to Level 3 or 4 by translators known as **compilers**, although occasionally they are interpreted instead;
- For example, in some cases Level 5 consists of an interpreted for a specific application domain, such as symbolic mathematics.

Computer Architecture

- Thus, computers are designed as a series of levels, each one built on its predecessors;
- Each level represents a distinct abstraction, with different operations and objects present;
- This view allows us to handle the complexity of computer design;
- The set of data types, operations and features of each level is called its **architecture**;
- Implementation aspects, such as what kind of chip technology is used to implement the memory, are not part of the architecture;
- The study of how to design those parts of a computer system that are visible to the programmers are called **computer architecture**;