CS1101: Lecture 8

Introduction to Computer Organisation

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

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

Lecture Outline

- Recommended Text
- Structured Computer Organisation
- Machine Language
- Structured Computer Organisation
- Languages, Levels & Virtual Machines
  - Translation
  - Interpretation
  - Virtual Machine
- Multiple Levels
- A Multi-Level Machine
- **Reading**: Tanenbaum, Chapter 1.

Recommended Text

- "*Structured Computer Organization*" (Fourth Edition) by Andrew S. Tanenbaum;
- Published by Prentice Hall International, 1999;
- ISBN: 0-13-020435-8.

Structured Computer Organisation

- A digital computer is a machine that can solve problems by carrying our instructions given to it;
- Sequence of instructions = **program**;
- The electronic circuits of each computer can recognise and directly execute a limited set of simple instructions into which all its programs must be converted before they can be executed;
- These instructions are rarely more complicated than:
  - Add 2 numbers;
  - Check a number to see if it's zero;
  - Copy a piece of data from one part of the computer's memory to another.

Machine Language

- A computer's primitive instructions form a language in which it is possible to communicate with the computer;
- This language is called a **machine language**;
- Each computer has a machine language;
- Usually the set of instructions is consistent with the intended use and the expected performance of the computer
- Aim is to reduce the cost and complexity of the electronics needed in a computer
- Because most machine languages are simple, it is difficult and tedious for people to use them.

Structured Computer Organisation

- It is convenient to structure computers as a series of abstractions, each building on the one beneath;
- Complexity can be mastered;
- Promotes systematic design of computer systems;
- This is **structured computer organization**.
- How do we reconcile what is convenient for people with what is convenient for computers?

Languages, Levels & Virtual Machines

- The problem can be addressed in two ways;
- Both involve designing a new set of instructions that is more convenient for people to use rather than the set of built-in machine instructions;
- These new instructions also form a language, L1;
- The built-in machine instructions form a language, L0;
- The two approaches we will discuss differ in the way programs written in L1 are executed by the computer, which, after all, can only execute programs written in its machine language, L0.
- The two approaches are:
  - Translation
  - Interpretation

Translation

- One method of executing a program written in L1 is first to replace each instruction in it by an equivalent sequence of instructions in L0;
- The resulting program consists entirely of L0 instructions;
- The computer then executes the new L0 program instead of the old L1 program;
- This technique is called **translation**
- In other words, the entire L1 program is first converted into an L0 program, then the L0 program is executed.

Interpretation

- The other method is to write a program in L0 that take a program written in L1 as input data and carries them out by examining each instruction in turn and executing the equivalent set of L0 instructions directly;
- In other words, after each L1 instruction is examined and decoded, it is carried out immediately;
- This technique does not require first generating a new program in L0;
- The technique is called **interpretation**
- The program used is called an **interpreter**.

Virtual Machine

- Rather than thinking in terms of translation or interpretation, it is often simpler to imagine the existence of a hypothetical computer or **virtual machine** whose machine language is L1 – lets call this machine M1 (and call the virtual machine for L0, M0)
- If such machines could be built cheaply enough we would not need L0 or a machine that executed programs in L0 at all;
- To make translation or interpretation practical, the languages L0 and L1 must not be "too" different – this means that L1, although better than L0, will still be far from ideal for most applications;
- Our aim is to relieve the programmer of the burden of having to express a language more suited to machines than people;

- So what do we do?

Multiple Levels

- Solution: invent another set of instruction that is more people-oriented and less machine-oriented than L1 – we will call this L2, with a virtual machine M2.
- The invention of a whole series of languages, each one more convenient than its predecessors, can go on until a suitable one is found;
- Each language uses its predecessor as a basis, so we may view a computer using this technique as a series of **layers** or **levels**, one on top of the other;
- The bottommost language or level is the simplest and the highest language or level is the most sophisticated;
- An *n* level machine comprises *n* different virtual machines and *n* different languages.

A Multi-Level Machine

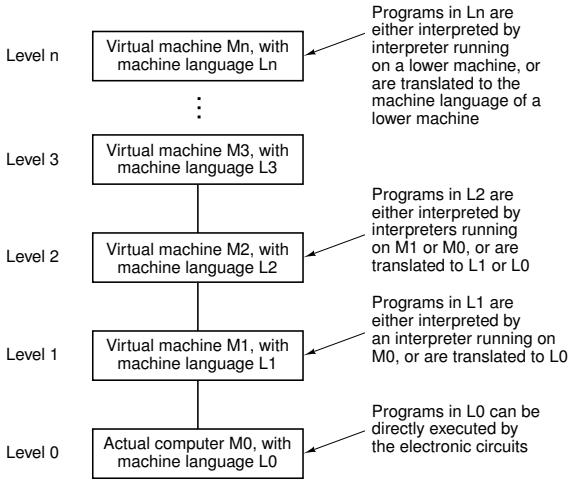

Figure 1.1 A multi-level machine