UNIVERSITY OF OSLO

Summary of INF2270 - Computer Architecture

Ivar Stangeby

January 7, 2015

Contents

Chapter 1

A Tour of Computer Systems

1.1 Information is Bits + Context

A *source program* is a sequence of *bits*, each with a value of 0 or 1, organized in 8-bit chunks called *bytes*. Files that consist exclusively of ASCII characters are known as *text files*. All other files are known as *binary files*.
All information in a system is represented as a bunch of bits.

1.2 Programs Are Translated by Other Programs into Different Forms

In order for a machine to understand the instructions written in a high-level C program, the program must first be compiled into a sequence of *machine-language* instructions. The result is an *executable object program*. This process is performed by a *compiler driver*, and is done in four steps:

1. *Preprocessing Phase*. The preprocessor (cpp) modifies the original C program according to directives (starting with # in the source code). The result is is another C program (usually with the .i extension).
2. *Compilation phase*. The compiler (cc1) translates the .i file into an *assembly-language program* that contains machine-language instructions (usually with the .s extension).
3. *Assembly phase*. The assembler (as) packages the machine-language instructions into a *relocatable object program*. This is a binary file (usually with the .o extension).
4. *Linking phase*. The linker (ld) merges separate precompiled files (.o files) that the C program uses. This results in an *executable object file* that can be executed by the system.

1.3 It Pays to Understand How Compilation Systems Work

There are some important reasons why programmers need to understand how compilation systems work:

1. *Optimizing program performance.* Is a switch-statement always better than a sequence of if-else statements? Is a while-loop more efficient than a for-loop? What is the difference in accessing local or global variables in terms of performance?

2. *Understanding link-time errors.* What does it mean when the linker reports that it cannot resolve a reference? Difference between a static and a global variable? What happens if you define two global variables in different files with the same name?

3. *Avoiding security holes.* Learning to understand the consequences of the way data and control information are stored on the stack.