Computer Science 50
Fall 2009
Scribe Notes

Week 10 Monday: November 9, 2009
Andrew Sellergren

Contents

Computer Science 50
Fall 2009
Scribe Notes

Week 10 Monday: November 9, 2009
Andrew Sellergren

1 Announcements (0:00–17:00)

- This is CS 50.
- 1 floppy disk (pick any color).
- 1 new handout.
- Check out Apple's commercial from 1984 which aired during the Super Bowl. Back then, Apple's introduction of a computer to compete with IBM's Big Blue was a huge deal.
- Read About Quiz 1. For those who received the print copy, forgive the typo in the first sentence which refers to Quiz 0. Quiz 1 will take place on Wednesday, November 18.
- Monday's lecture will be split amongst several guest lecturers who will introduce you to the computer science courses they teach at Harvard. Taking computer science at Harvard has evolved from a single, set path to many, varied, more interesting paths. After CS 50, you are prepared to take CS 51, which deals with functional programming and abstraction; CS 61, which covers systems programming (in which you'll "defuse" a binary bomb); CS 121, which discusses what computers theoretically can and can't do; CS 124, which examines algorithms; CS 171, which explorers visualization; CS 105, which delves into privacy and security; and CS 179, which looks at user interfaces, in particular the iPhone. About half of you will never take a CS course again (which is fine), but all of you are encouraged to dabble in, to minor in, or to major in CS.
- Apply to be a TF or CA for next year! Being a CS 50 TF is an intense but rewarding experience. TFs are responsible for leading sections, holding office hours, grading problem sets. CAs are generally CS 50 alumni who volunteer for two hours a week to help with answer students' questions via e-mail, bulletin board, and office hours.
- Like CS? (Or cider or pie?) Attend HCS and SEAS's Computer Science Concentration Cider & Pie on Fri 11/13, 3pm, Maxwell Dworkin lobby. (Win prizes.)
- Shuttleboy Cards are still available at shuttleboy.cs50.net.
- Congrats to Drew Robb for earning 113 quintillion dollars and trouncing everyone on Problem Set 7's Big Board.
- Check out The CS 50 Store.
- Last year, Alex Bick, Joy Ding, Drew Robb, Cameron Spickert and Winston Yan won the AT&T Big Mobile on Campus Challenge for their Rover iPhone app, which grew out of a CS 50 final project. Next year it could

be you! Don't worry, though, the finished product took several years of development. The bar isn't quite *that* high for final projects.[1]

2 Underneath the Hood (17:00–50:00)

2.1 The Compiler

- So far we've taken for granted what happens when we compile a program in C. We know that our source code gets translated into binary, but more specifically, it gets translated into instructions which the CPU can understand. These instructions are actually CPU-specific to a certain extent. There's a set of instructions, for example, that only Intel CPUs can understand—and perhaps even more astounding, these instructions are no more complicated than simple arithmetic operations like addition and subtraction.

- `hello.c` will be familiar to you from Week 1 of the course:

```
/**************************************************************************
 * hello.c
 *
 * Computer Science 50
 * David J. Malan
 *
 * Says hello to the world.
 **************************************************************************/

#include <stdio.h>

int
main(int argc, char *argv[])
{
    printf("hello, world!\n");
}
```

If we compile this program using the `-S` flag for GCC, we'll get output in a file called `hello.s`, a snippet of which is to follow:

```
    .file    "hello.c"
    .section    .rodata
.LC0:
    .string "hello, world\n"
    .text
.globl main
```

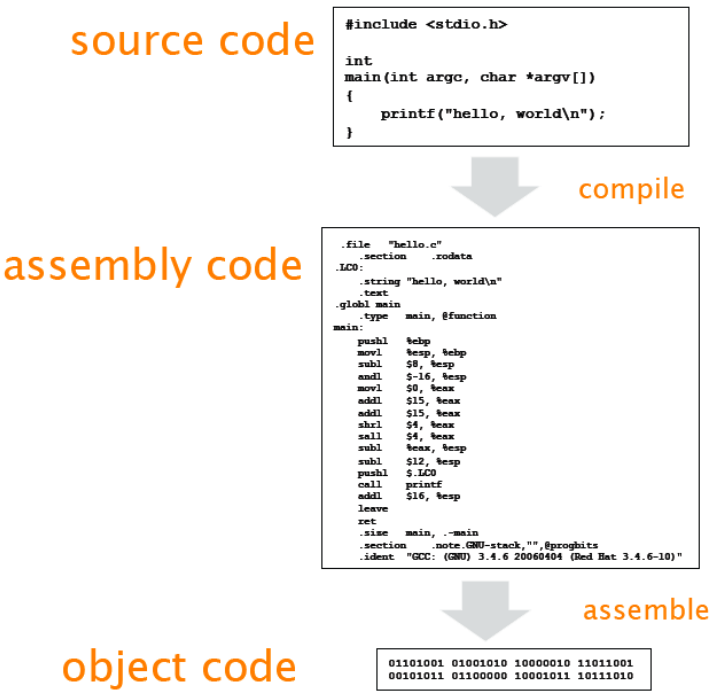
[1] Yes it is. My final project proved the Riemann Hypothesis and solved global warming. In LOLCODE.

```
    .type    main, @function
main:
    pushl    %ebp
    movl     %esp, %ebp
    subl     $8, %esp
    andl     $-16, %esp
    movl     $0, %eax
    addl    $15, %eax
    addl    $15, %eax
    ...
```

`.rodata` signifies read-only data and is follow shortly thereafter by the string `"hello, world\n"`. It makes sense that this string is in a read-only section since it is hard-coded into our program. The `.text` section points to our actual source code. As you can see, the instructions boil down to addition and subtraction. `addl` and `subl` actually stand for "add long" and "subtract long," in reference to the data type `long`.

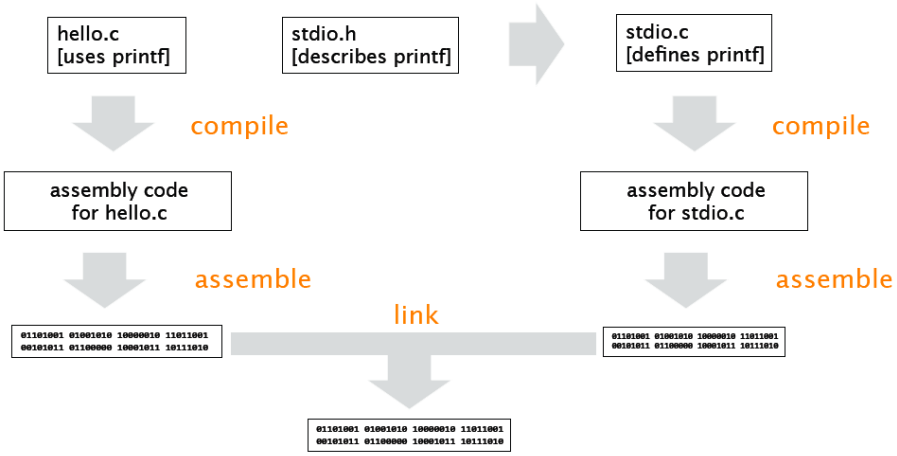
- In general, there are five steps to the creation of a working program (after writing the source code):
  - Pre-Processing
    - pre-processor directives, including `#define` and `#include` are interpreted—constants are replaced with their actual values and other source code files are fetched
  - Compiling
    - the compiler converts source code into assembly code
  - Assembling
    - assembly code is converted into object code (`.o` file), which consists of 0's and 1's
  - Linking
    - the `-l` compiler flag ensures that library code is added to the program
    - earlier, during the pre-processing step, when helper files were fetched, they were themselves compiled and assembled; now all the resulting `.o` files are merged in memory
  - Executing

  This figure below shows the first three steps of the software creation process:



The actual object code that is created during this process is much lengthier than this figure suggests. As we've already learned, most CPUs have a 32-bit architecture. What this literally means is that when a program is executed, its 0's and 1's are fed into the CPU 32 at a time. The first few bits generally denote what CPU instruction should be used. The last 8 or 16 usually refer to registers, the smallest useful chunks of memory (of which there are 32 to 64 on a system) which store the actual operands on which instructions are performed. If two numbers are being added together, for example, each is stored in a separate register, as is the result of operation.

- Whenever our code makes use of library functions like `printf`, the source code for those functions needs to be compiled and assembled in parallel to our own program's code. Then, during the linking step, the `.o` files are combined, as this figure implies:



If this figure is accurate, then why have we never had to explicitly link to `stdio.h` at compile-time? Turns out this library is common enough that GCC automatically links to it.

- This type of linking is called *static linking*. In contrast, files with the `.dll` extension, which are common as Windows drivers, are *dynamic link libraries*. One major difference between these two types of linking is that statically linked programs are generally portable. You can move them between systems which have the same CPUs because everything that is needed to run them is packaged within the executable. On the other hand, dynamically linked programs require additional files to be able to execute properly. This is why you can't simply copy and paste Microsoft Word's `.exe` file from one computer to another and expect the program to work. A program which is statically linked with a library contains that library's actual bits. A program which is dynamically linked with a library contains only a path to the library's actual bits. One advantage of dynamically linking is space optimization since you don't have to copy over and over again the bits of a library whose code will be reused often.

2.2 Permanent Data Storage

- Hard drives consist of circular platters which spin as data is being written to and read from them. When data is to be stored on the hard drive, it passes from RAM along with software signals that designate how the data is to be stored. Certain signals control how the platters spin and others control the read-write heads. The distance between the heads and platters is less than the width of a human hair, yet the platters spin 5400 RPM or faster. Multiple platters make for greater efficiency than a single platter.
- A read-write head in a hard drive contains a tiny electromagnet which conducts the software signals it is passed and, during writing, flips single

bits on the platter either on or off by polarizing it in one direction or the opposite. During reading, the process is reversed and the electromagnet conducts signals from the bits on the platter. A single file may be stored in locations scattered widely across a platter, so a separate file keeps track of the locations of all the bits of files.

- One compelling aspect of computer science is that unlike biology, in which the objects of study (e.g. the human body) are still largely a mystery, the object of study in computer science is quite well understood and is relatively easily understood.

- CDs can store much more data in a more compact space because they are read with an extremely narrow beam of light. The actual surface of a CD consists of pits and lands. When the laser strikes a pit, the light is scattered. When the laser strikes a pit, the light is reflected directly back. The pattern of blanks (where no light is reflected back) and pulses (where light **is** reflected back) is translated into electric current and ultimately 0's and 1's.

- CD-Rs which you read and write from yourselves generally consist of a piece of plastic with one side covered in a label and the other side covered in a dye. When writing to the CD, this dye is distorted in a specific places so that it will scatter the light from a laser when it's being read.

- Floppy disks, which store only 1.44 MB, are cheaper and much slower than CDs. They consist of a hard plastic coating and an inner metal-coated "cookie" which is protected by a sliding shield. When the disk is inserted in the drive, levers hold back the shield and read-write heads enclose themselves around the cookie, which spins just like a hard drive platter or a CD. In the bottom corner there is a write-protect tab which, if open, signals to the computer that data should not be written to the disk. As with a hard drive platter, data is written to the surface of a floppy disk cookie via a tiny electromagnet that polarizes the bits one direction or the opposite.

- Feel free to rip off the sliding shield and break open the plastic coating so you can play with the cookie they protect!