Lecture 2

Machine language

Let my machine talk to me...

CS 241: Foundations of Sequential Programs
Winter 2018

Troy Vasiga et al
University of Waterloo

In Linux (and all operating systems)

Bit sequences matter!
Lots of ways of Linux-ifing your Windows machine:

- cygwin (and use
  `ssh -Y yourname@linux.student.cs.uwaterloo.ca`)
- Linux Live CD
- install your own linux (Ubuntu is well supported and easy to use)
- putty

You may also use the lab machines running Linux in MC 3xxx or MC 2xxx.

Grouping of Bits

- most common grouping is a byte: 8 bits
- Integers:
- Characters:

Larger Groupings of Bits

- "standard" ASCII
- "extended" ASCII
- Unicode (e.g., UTF-8)

A word about words

A word on a machine is:

- 8
- 16
- 32
- 64

Files

- a sequence of bytes
- the interpretation is in the eye of the beholder
  - `cat` interprets the file as a sequence of ASCII characters
  - `eog` interprets the file as a description of an image
  - `xxd` indicates exactly what bits are in the file

Demo

Files:

- `beep`
- `hasX1`
- `babe.jpg`

Programs (that interpret the files):

- `cat`
- `eog`
- `xxd`

Our Machine: A Stored Program Computer

CPU

Memory (RAM)

Machine Language

- Instructions to the machine
- MIPS: 18 different 32-bit instructions encoded in two basic instruction formats
- See the MIPS Machine Language Reference sheet (from the course webpage)

MIPS as a Programming Language

- the language that the CPU speaks
- example: `add $1, $2, $3`
- human meaning
- computer meaning

Communication between CPU and RAM

- load

- store

Machine Cycle

PC =

IR =

Fetch-Decode-Execute Loop

- PC $\leftarrow$ 0
- loop
  - fetch word from RAM whose address is in the PC
  - place that word in IR
  - PC $\leftarrow$ PC + 4
  - decode and execute the instruction that is in IR

Examples

See website. In particular:

- `add` example
- `add` and `lis` example
- `slt` example
- `beq` example

Note: you are writing *subprograms* (for the most part) in this course, and thus you should always *return*.