Slide 1

Administrivia

- Next homework coming soon. I will send mail.

Slide 2

Floating Point in MIPS Architecture

- Architecture defines 32 floating-point registers (`$f0` through `$f31`), used singly for single-precision, in pairs for double-precision.
- Instruction set includes:
  - Arithmetic instructions:
    `add.s`, `sub.s`, `mul.s`, `div.s`; `add.d`, `sub.d`, `mul.d`, `div.d`
  - Load/store instructions (single-precision):
    `lwc1`; `swc1`
  - Comparisons:
    `c.eq.s`, `c.lt.s`, etc.; `c.eq.d`, `c.lt.d`, etc.
    These set a bit true/false, which can be used by `bc1t`, `bc1f`.

Slide 3

A Little About Circuit Design

- Goal — sketch design of a (hardware) implementation of MIPS architecture in terms of some simple building blocks (AND and OR gates, inverters).
- Things we'll need:
  - Something to implement instructions: ALU (arithmetic/logic unit).
  - Something to implement registers: register file.
  - Something to implement fetch/decode/execute cycle: control logic.

Slide 4

Implementing Logic Gates — Executive-Level Summary

- The ones and zeros of low-level software become two distinct voltages in hardware, and the logic of Boolean algebra is implemented using "switches" (things that connect an input to an output, or not, depending on the state of a control input).
- Currently these switches are (usually?) transistors. In widely-used "CMOS technology", there are two types of switches, one that's good if the input is "one" and one that's good if the input is "zero". These can be combined to implement logic. Simple example: Inverter. (See link from "useful links" page.)

Slide 5

Minute Essay

- None — sign in.