Slide 1

Administrivia

- (Via e-mail?)

Slide 2

Designing a Processor — Recap/Review

- We sketched out a design for something that implements the designated subset of MIPS instructions.
- What we have is the “datapath” part . . .

Slide 3

The "Datapath" — What's Missing

- Inputs to some blocks (e.g. PC) can come from more than one source. *That* can't work. So we need multiplexors to control which is used.
- Inputs to ALU / adder are 32 bits, but for some instructions we want to get one of them from 16 bits in instruction. So, need something to extend that to 32 bits by extending sign.
- Both control-flow instructions include something that must be shifted two bits before being used to compute target address, so need to support that.
- Add these to "datapath" part of Figure 4.1 to get Figure 4.15. Leaves out "control" part, substituting not-connected-yet control inputs (blue in figures.)
- Right now we're showing the whole instruction as input to all elements that need part of it; we'll refine this in the next step.

Slide 4

Control Logic

- So we have a "datapath" that can do things, but there are still some inputs that aren't connected to anything. An analogy — the datapath is a puppet, and these inputs are its strings.
- Who/what pulls the strings? the "control logic" — combinational logic whose input is the current instruction plus any other needed information and whose output is those disconnected inputs to datapath.

Slide 5

Control Logic

- Figure 4.16 shows names of "control signals" and what they mean. (Note: Textbook uses "asserted" and "deasserted"; I'll just use 1 and 0 — textbook indicates that this is a bit sloppy but I think okay for our purposes.)

  (Why `MemRead`? textbook says/implies that data memory is constructed such that attempts to read from invalid address could cause problems, and sometimes address *won't* be valid.)
- How to generate them? As mentioned in Appendix B, tools exist to transform truth tables into combinational logic, so it will be enough to come up with a truth table that maps inputs to the needed signals.

Slide 6

Control Logic, Continued

- Figure 4.17 adds needed combinational logic blocks to Figure 4.15.
- Section 4.4 works through details. A lot of it should seem like common sense (viewed from the right angle?). Only potentially tricky part is input to ALU "which operation?" . . .

Slide 7

ALU Control Input

- ALU as designed in Appendix B uses 4 bits to represent which operation is to be done — 2-bit input to multiplexor plus (see Figure B.5.10 and table at start of 4.4 — e.g., 0010 to add, 0110 to subtract). Seems like it would be simple enough for the main control unit to generate these directly?
- However, turns out to be even simpler to split functionality into two parts: Generate a 2-bit “ALU operation” from just the opcode field, and then use that plus (for arithmetic instructions) function field to tell the ALU what to do.

  (How is this simpler? Everything but the ALU operation depends only on opcode, and the ALU operation can be generated from 2 bits based on the opcode and the function field. Textbook seems to say two smaller combinational-logic blocks are preferable to one big one.)

Slide 8

Additions for Jump

- Figure 4.24 shows additions for `j`. Again, sort of common sense (from the right angle?)

Slide 9

Instruction Execution Details — Tracing What Happens

- Tracing through what happens as various instructions are executed is tedious but (I think!) instructive:
- Work from Figure 4.17 (revised/improved version of 4.15) and the tables in Figure 4.18 and Figure 4.13.
- Start out by writing down what you know: Output of PC (its current value), fields of instruction.
- Use figure and tables to fill in other things, tracing through how bits flow.
- What you come up should be consistent with what the instruction is supposed to do.

Slide 10

Instruction Execution Details — Examples

- Example `add`. (Solution online as part of Homework 6 assignment.)
- Example `lw`. (Solution online as part of Homework 6 assignment.)
- Example `beq`. (Solution online as part of Homework 6 assignment.)

Slide 11

Minute Essay

- Questions?