Slide 1

Administrivia

- Reminder: Homework 4 due Wednesday.
- Reminder: Abelson lecture 4pm today, Chapman Auditorium. Other talks:
  - "Amorphous computing" in Dr. Myers's AI class tomorrow at 11:20am in HAS 340.
  - Cryptography at 3:30pm in HAS 228.

Slide 2

Minute Essay From Last Lecture

- Question: Suppose you are given the address of a 32-bit word in the memory of a computer implementing the MIPS architecture. How can you tell whether the 32 bits there are an integer, a single-precision floating point number, or something else? (What are some of the other possibilities?)
- Answer?

Slide 3

Recap — What We've Done, What's Next

- We've talked about
  - Defining the smallest steps the processor can take — instructions.
  - What these smallest steps operate on — registers and memory elements.
  - How to do arithmetic in terms of Boolean algebra.
- Next is to talk about how to make something that executes a sequence of instructions — looking at a representative subset of the whole instruction set to keep things manageable.

  Sketch overall plan — figure 5.1.

Slide 4

Combinational Logic Blocks Versus State Elements

- We'll talk about two kinds of "functional units" (circuits with inputs, outputs), combinational logic blocks and state elements.
- Combinational logic blocks (e.g., our ALU) have no internal state — outputs depend only on inputs.
- State elements (e.g., registers, memory elements) *do* have internal/saved state — outputs depend on inputs and internal state, internal state can be changed.

Slide 5

Combinational Logic Blocks

- To review — CLB is a circuit that implements a boolean function or functions.
- In textbook's view, "circuit" is a combination of AND and OR gates and inverters. In a lower-level view, it's a combination of "switches".

  A "switch" here is something that allows current to flow / not flow between two points, depending on whether its control input is 1 or 0. By connecting switches in the right way (together with a "source of 1s" and a "source of 0s"), we can build gates that work as described.
- When inputs to a CL block change, these changes ripple through the circuit, changing which switches are open/closed. After a delay (how long depends on how many switches are connected "in sequence"), outputs to block change to reflect input changes.
- Examples include multiplexor (figure B4), ALU of chapter 4.

Slide 6

State Elements

- Combinational logic is good and useful, but — we also need some way to hold values in a stable way (for registers, memory, etc.).
- Simplest state element — unclocked set/reset latch (figure B12). When R input is 1, sets Q to 0; when S input is 1, sets Q to 1.

Slide 7

State Elements, Continued

- What we really want, though, is something that will hold a value and also only get updated when the intended new value is "stable".

  So, useful to have a "clock" (something that cycles back and forth between 0 and 1 at regular intervals) and only update "state elements" once per cycle.

- So, add a clock signal to set/reset latch to get "D latch" (figure B13). When clock signal (C) is 1, latch is "open" and output changes to match input D.

Slide 8

State Elements, Continued

- If we also want to be able to include state elements in a "feedback circuit" (outputs feed back into inputs), though, we'll need something more.
- So, add more logic to get "flip-flop" (figure B15):

  While clock is 1, input is being "sampled" and stored in first latch, but can't affect output.

  When clock goes to 0, stop sampling input and use sampled value to set output.

Slide 9

State Elements, Continued

- More generally, a state element is a block with inputs and outputs (like CL block), plus "internal state".
- Outputs depend on inputs and internal state.
- Inputs include "clock".
- Internal state is updated (at most) once per clock cycle – for convenience, we'll say "when clock goes from 1 to 0".

  ("At most" is because we might decide to update / not update based on other inputs.)

Slide 10

Combining CL Blocks and State Elements

- We'll typically build things from combinations of state elements and CL blocks, as in figure B10 and figure B11.
- Notice how this works:

  While clock is 1, values from state element #1 percolate through the CL block to produce outputs.

  When clock goes to 0, those outputs are used to set a new value for state element #2.

Slide 11

Minute Essay

- Sketch a combinational logic block with two inputs a and b and an output that's 1 when they're equal and 0 otherwise.