05-combo-notes

Combinatorial Logic

Agenda

0. Re-Orienting

1. Combinatorial Logic

2. Adders

3. Reality affects logic (again)

---

0. Re-Orienting

Where are we, and how did we get here?

From the top, going down:

- mental model of programming
- layers of software
- OS
- ...
- functions on integers
- integers

From the bottom, going up, sort of...

- atomic physics
- voltage levels
- 1, 0
- logic gates
- binary numbers
  - (also hex, octal, BCD representations)
- combinatorial logic...
- ...

---

1. Combinatorial Logic

In the previous class, we showed how to build logic circuits that

- test if two binary numbers are equal
- recognize if a binary number is a specific value
- calculate an arbitrary 1-bit function on k-bit binary numbers

(And... if you scroll to the bottom of the notes for the previous class, you'll find more complete, annotated versions of the skeletons we played with in class.)

Since we also now know how to use transistors to build electrical machines that carry out these gate operations.... we also know how to build electrical machines that calculate these arbitrary 1-bit functions!

This is the general idea of **combinatorial** logic: circuits whose outputs are determined exclusively by the combination of values sent in on the inputs.

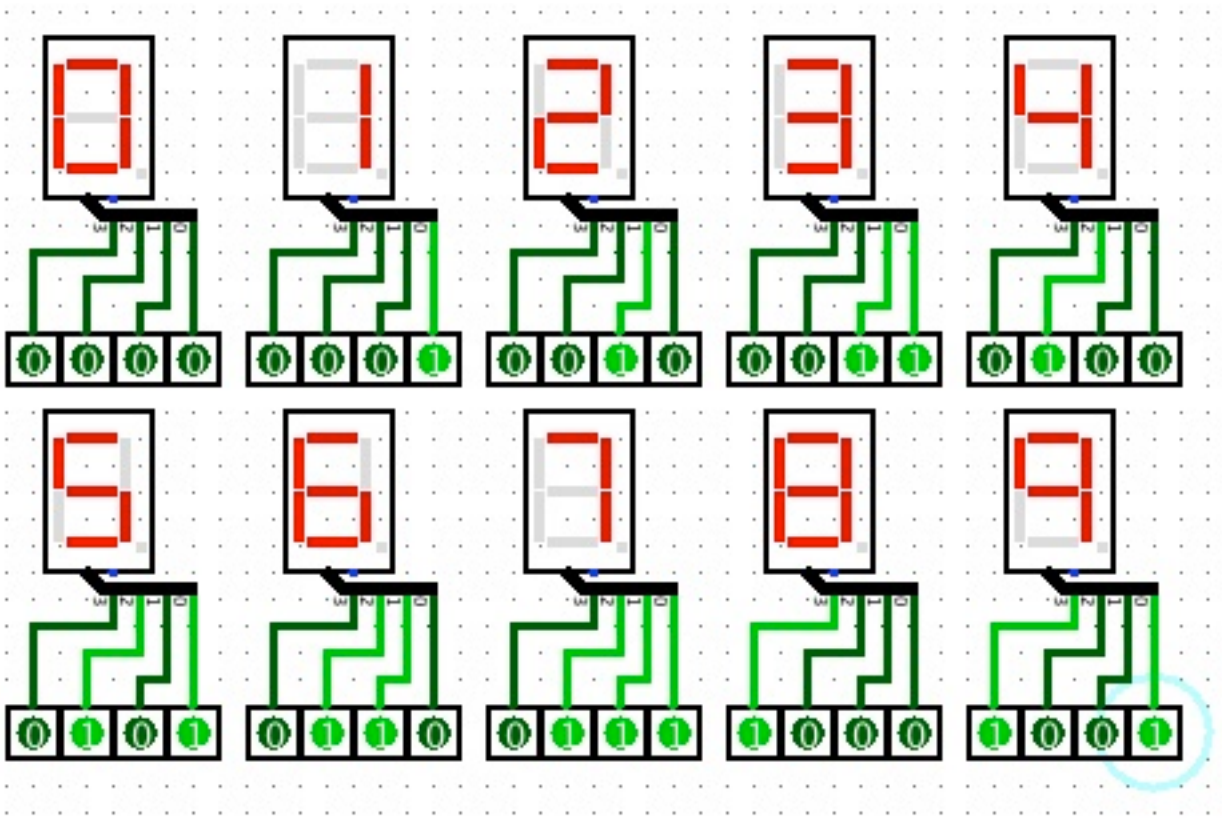
We know how to solve the general case. However, the decoder/OR approach isn't always the most expressive or efficient way to approach a problem... so we'll cover some other approaches (and useful examples).

---

review: seven-segment LED display

---

seven-segment LED display



[digits.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/digits.zip)](https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/digits.zip)

---

a seven-segment LED driver!

input: BCD

output: turning on the right segments

7seg-1.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/7seg-1.zip)

7seg-matrix.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/7seg-matrix.zip)

- note the use of a "decoder" module
- the idea of physically embodying the "matrix"
- but also the trouble we get into if we just fill in the "1"s with connections

multiplexor

mux.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/mux.zip)

demultiplexor

demux.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/demux.zip)

2. Adders

adding two bits?

need carry

half-add.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/half-add.zip)

what about the *next* two bits?

need to add in the PREVIOUS carry

draw out a truth table...

full-add.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/full-add.zip)

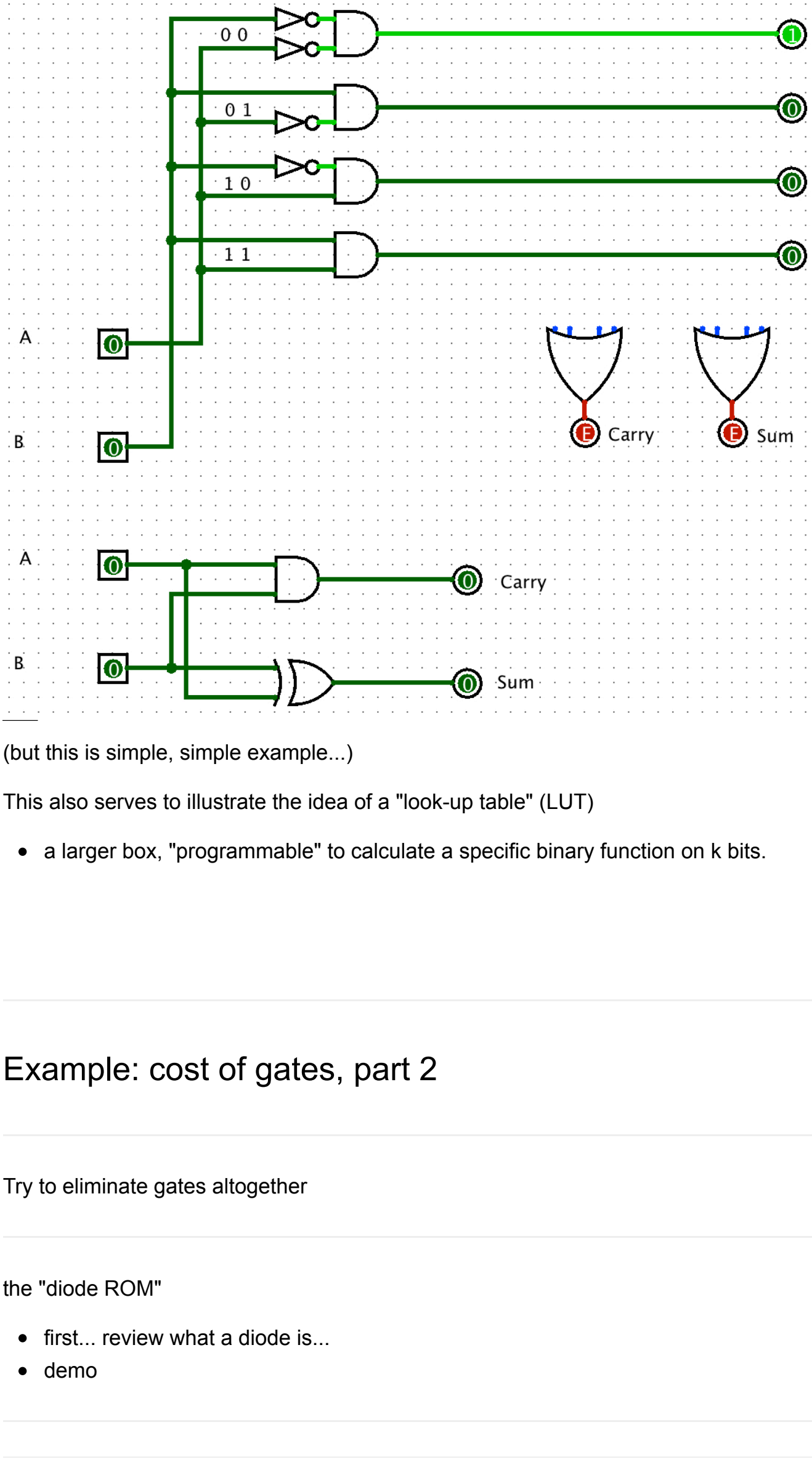
generalizing

4adder-blank.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/4adder-blank.zip)

3. Reality affects logic, revisited

Example: cost of gates

half-adders.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/05-combo/demo/half-adders.zip)



(but this is simple, simple example...)

This also serves to illustrate the idea of a "look-up table" (LUT)

- a larger box, "programmable" to calculate a specific binary function on k bits.

Example: cost of gates, part 2

Try to eliminate gates altogether

---

the "diode ROM"

- first... review what a diode is...
- demo

Now, return to our seven-segment LED driver. We really just wanted to encode a 23 by 7 matrix of zeros and ones.... but ended needing big honking gates on each output line. Can you think of how to use "electrical" properties of discrete components to make it simpler?

drom.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/03-reality/demo/drom.zip)

Example: time delay through gates

Remember my simulation of a delays through gates?

What happens if we try it with a 4-bit adder? See "4-bit delayed" in **delayed.circ (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/03-reality/demo/delayed.zip)**

Suppose it takes 1 second to go through a gate.

- How long is it before S0 is ready?
- How long before S3 is ready?

If you've read this far

Here (https://ssl.cs.dartmouth.edu/~sws/cs51-s14/04-combo/demo/completed.zip) are the completed circuits for today's demos.