Goals

Practice creating circuits using Logisim, and designing those circuits using truth tables.

General Instructions

Paste a screenshot of your circuits from Logisim into a document (PDF or open office or plain text). Upload this document to Sakai and hand in a hard copy by the due date. BE SURE YOU PUT YOUR NAME AND THE LAB NUMBER at the top of the document! Also upload the circuits themselves as a separate file.

Converting a Boolean Expression to a Truth Table [5 points]

Construct a truth table for the expression $Z = \sim ([A + B] \cdot (\sim A + C)] + (B \cdot \sim C)$

Converting a Truth Table to a Karnaugh Map [5 points]

Take the output of the truth table from the previous question, and simplify the expression using a Karnaugh Map.

Converting a Simplified Boolean Expression to a Circuit [10 points]

Take the simplified Boolean expression that resulted from the previous question, and draw a circuit in Logisim that produces the correct input/output relationship.
Grandma Ann’s Cookie Watcher [20 points]

Grandma Ann, owner and founder of Grandma Ann’s Crumbling Cookies, recognizes that to stay competitive in today’s marketplace, she must integrate technology into her business in order to keep quality up and prices down.

To assist her business, she enrolled in CMPSC 210 last year. Using her acquired knowledge, she designed a cookie watcher. The cookie watcher “watches” multiple batches of cookies baking simultaneously. It is precisely timed to an individual recipe’s cooking time. The cookie watcher alerts an employee to when a batch is ready to be taken out of the oven, as well as when a batch is ruined (overcooked). It is easily scalable to watch multiple cookie batches at once. Thanks to the cookie watcher, fewer cookies are being burnt, and the cookies are consistently baked the proper amount of time. As a result, Grandma Ann’s profits soared. She has since retired to a private island in the Antilles.

In this section of the lab, you will recreate her cookie watcher circuit in Logisim. Naturally, Grandma Ann first tested her design in Logisim and then eventually soldered her own physical version. First, look at the final design on the next page.

Now start the following steps:

1. Start a new Logisim circuit up.

2. Click “Project” then ”Add Circuit” to add a subcircuit. We can design a subcircuit once, then instantiate multiple copies of it. Name this subcircuit “CookieWatcher.” We will first design this circuit. It will be capable of watching exactly 1 batch of cookies.

3. Add 2 input pins and give them the proper labels.

4. Add a counter (available from the “Memory” section on the side). Change its “Action on Overflow” property to be “Stay on Value.” When the counter reaches its maximum, it will not wrap around back to 0. Change its “Data Bits” to 10.

5. Connect the input pins to the counter as shown in the CookieWatcher design.

6. Draw a short wire coming out of the “Q” output of the counter. This is the counter’s value. The output is, by default, a bundle. A bundle is several (in this case 10) wires all connected together. Using a bundle can save us space when working with multiple bits at once. Bundles are always black in color, and we are unable to see the values on the individual wires within the bundle.

7. Connect a splitter (from the “Base” category) to the end of the bundle. Change the splitter’s BitWidthIn to 10 (because the counter’s output is 10 bits). Change the splitter’s BitWidthOut to be 10. This will split the 10-bit input into 10 pieces. A 10-bit input split 10 ways will result in 10 1-bit outputs.

8. Connect the 10 outputs from the splitter to AND gates as shown. Connect the AND gates as shown.
9. Add 2 output pins and give them the names as shown. The CookieWatcher’s output is whether the batch of cookies it is watching is done (1 of the output pins states this) as well as whether that batch has been burnt (the other output pin states this).

10. Before moving on, think about the circuit you’ve just built. Why are the outputs from the counter connected up in the way that they are? What do you think will happen as the counter changes? When will the outputs of the cookie watcher become true? False?

Figure 1: The CookieWatcher subcircuit. One input is whether that batch of cookies was handled. The other input is the block signal. The inputs are fed into a counter. The counter’s value is used to determine whether the cookies are done and whether they are ruined (overbaked).

You have now completed the CookieWatcher circuit. Now, we can connect multiple CookieWatchers to create an automated cookie baking notification system capable of watching any number of baking cookies. Switch to the “main” circuit, which should currently be blank. To do this, double-click on the word “main” (which should be above the words “CookieWatcher” on the left hand pane).

1. Add a block component (available in the “Wiring” category).

2. Add 6 buttons (available from the “Input/Output” category).

3. Add 6 CookieWatcher subcircuits. To insert one subcircuit, single-click on the “CookieWatcher” entry in the side pane. Then, click in the main circuit to place it somewhere. It will appear as a small box. Do this 6 times.

4. With the arrow tool active, however over the four dots on a CookieWatcher circuit. Notice that the left-hand ones are blue, meaning that they are unconnected, expecting an input. The right-hand ones are dull green right now, meaning that they are outputting a 0. We call these dots pins. Hovering over a pin tells you the purpose of the pin, assuming you have the subcircuit’s pin a label. For example, hovering over the top pin on the right-hand side of a CookieWatcher should show that the pin is the CookiesRuined pin.

5. Add 8 LEDs (under “Input/Output”). Connect them as shown in the figure.
Figure 2: Six cookie watchers. The second and fifth batches are ready to be taken out of the oven. The first and fourth batches are burnt. The third and sixth batches are not ready to be taken from the oven.

Your circuit should now be done. To test it, we will force the clock to tick, thus updating the circuit state. We can poke the clock to have it switch from low (0) to high (1) and vice versa. This lets us step through time, one cycle at a time.

To really test this circuit though, let’s force the clock to update automatically at some fixed frequency. Under the “Simulate” menu, go to “Ticks Frequency” and select “128 Hz” (128 Hertz, 128 times per second).

Now, under “Simulate” choose “Ticks Enabled” (keyboard shortcut Ctrl-K). The clock will automatically tick away, advancing the circuit state.

The clock is connected to each CookieWatcher, and the CookieWatcher circuit will, eventually, turn on the lights. At 128Hz, the bottom light should come on in about 4 seconds. The bottom light turning on informs the employee that a batch of cookies are done. The employee would then take out the cookies, put in a new batch, and press (i.e. poke) the batch’s button to reset the timer.

If the employee waits too long, the top light will come on. This lets the employee know that the cookies have been burnt. The employee should then put in a new batch, throw out the old batch, and press the reset button. At a blockrate of 128Hz, the top light will come on after 8 seconds.

Make sure that your circuit is working as designed. You should understand what each of the components are doing and why they are doing what they do.
Ask yourself the following (you do not have to submit these answers):

- When exactly does the CookiesDone light turn on?
- When exactly does the CookiesRuined light turn on?
- When the user pokes a button, what happens? How does the counter respond?