SYSTEMS I — LAB 3
A 4-bit counter

A counter

Build a circuit that counts from 0 to 15, and then wraps back around to 0 again. (Sound a bit like lab-2?) The circuit should be driven by a *clock* input, where each *cycle* of the clock (from 0 to 1 and back to 0 again) moves the clock forward by one value. Thus, after 16 clock cycles, the counter should be back where it started, having progressed through every value once.

Useful components

There are three new parts worth describing here:

- **Flip-flop chips:** The 74LS273 chip contains 8 1-bit D flip-flops. Examining the chip diagram, we see that this is a 20-pin chip (unlike the 14-pin chips that we've used so far). It still requires power (V_{cc}) and ground (GND) connections. Each of the 8 flip-flops has a data input (D) and a data output (Q).

  There are two new pins that have not appeared on previous chips for us. The first is the **clock** (CLK) pin. This pin is connected to the clock input of all of the 8 flip-flops in the chip. Thus, when you cycle the input on this pin, all of its flip-flops will adopt new values. Note that these flip-flops are *rising edge-triggered.*

  The other new and noteworth pin on this chip in the **negated clear** ($\overline{CLR}$). When this input to this pin is 1, the flip-flops will behave normally. However, if the input of this pin is set to 0, all of the flip-flops in the chip will have their value immediately reset to 0, irrespective of the value on the clock input. I suggest connecting this pin to its own pulse switch (see below), allowing you to reset your counter to zero with the press of one button.

- **Pulse switches:** In the lower-right corner of your ETS-7000, there are two red buttons, P_A and P_B. There are also four outputs from these switches, A, $\bar{A}$, B, and $\bar{B}$. When P_A is not being depressed, $A = 0$ and $\bar{A} = 1$; when it is depressed, $A = 1$ and $\bar{A} = 0$.

  Thus, these buttons can be used in two ways for this project. First, you could connect $\bar{A}$ to the $\overline{CLR}$ input of your flip-flop chip(s). Thus, the input to $\overline{CLR}$ is normally 1, leaving the chip to operate as normal. To clear the contents of your memory element, just depress P_A for a moment. Similarly, B can be connected to the CLK input. Thus, each time you depress and release P_B, you cycle the clock, advancing your counter by one step.

- **Digital probes:** There are, floating around the lab, a number of yellow devices with a pair of wires coming out of the bottom, a pointy metal thing coming out of the top, and a few LED's and switches on its face. This device allows you to test easily the values flowing through your circuit.

  To "plug in" this device, you need two (short) wires, one plugged into power and the other into ground—you can do so by plugging the wires directly into a connected portion of your power channels. Then, you can connect the red *alligator clip* of the digital probe to the power wire, and the black clip to the ground wire. Now you can stick the pointy end of the probe into any hole on your breadboard that is also connected to a wire and/or pin. It will emit a lower sound and light the "low" LED if that row carries a 0, and it will emit a higher sound and light the "high" LED if it carries a 1. You can now use this device to test intermediate parts of your circuit to find errors.

How to do it (roughly)

To make a 4-bit counter, you need 4 flip-flops. Each one of those will store the *current state* of the counter—that is, it will emit the counter's current value. Your task is to use the counter's current value as an **input** to a combinational circuit that computes the **next** value that the counter should take. Thus, the **output** of that combinational circuit should the **input** to your memory elements. Thus, when the clock cycles, the counter's next value is loaded into the flip-flops, thus making it the **current** counter value.

An extra challenge

Have you figured out how to make this counter work? Does it nicely count forward, from 0 to 15 and then wrap around again, as the clock "ticks"? If so, here's a slightly more challenging version of this problem:

Make a **3-bit counter** that counts through the following arbitrary sequence of values:

000
011
110
100
010
101

Note that the counter should start at 000, progress through the above values to 101, and then wrap around to 000 again. Also note that the period of this counter is 6—that is, not all 8 possible 3-bit values are used. Every 6 clock cycles, the counter should repeat itself.

Finishing up

The usual rules for finishing your work apply:

- When done working in the lab, **clean up**. Put away wires, probes, excess chips, wire trimmers, chip pullers, and any other devices that you've used. If you have created garbage by cutting and stripping wire, throw it out.
- To "submit" your work, show it to me. You'll need to demonstrate that your adder handles all 16 cases correctly.

This assignment is due on Friday, September 26, at 11:00 am