COMPUTER SYSTEMS I — LAB 3B
Flip-flops and 4-bit counters

1 Building a flip-flop

Congratulations! If you're here, you know what the problem is with your 1-bit counter and your *D latch*. So, build a *D flip-flop* to fix your 1-bit counter. Make it work. Then move onto the next section.

2 The 4-bit counter

Build a *sequential logic circuit* that counts from 0 to 15, and then wraps back around to 0 again. 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 counter forward by one value. Thus, after 16 clock cycles, the counter should be back to the value at which it started, having progressed through every value once.

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—that is, this chip behaves as an *8-bit register*. 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*—that is, they adopt a new value as the CLK pin transitions from 0 to 1.

The other new and noteworthy 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 CLK 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.

2.1 How to do it (roughly)

To make a 4-bit counter, you need a 4-bit register—a role that can be performed by a single 74LS273 chip. The value stored by this register, and thus its output value, is, at any moment, the *current state* of the counter—that is, it will emit the counter's current value. You then need a combinational circuit that can use the register's **output** value as an **input**, and then generate the counter's **next** value.

Show your work before moving onto the final part of the assignment.

3 Changing the sequence

Once you have created your basic counter, you need to create a slightly different counter. Specifically, you need 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.

Once you have this final counter sequence working, show your work.

This assignment is due on Friday, October 15, at the beginning of your lab section