INTRODUCTION TO COMPUTER ARCHITECTURE

LAB 3

A counter

1 Building a 4-bit counter

We want to make a circuit that is controlled by a *clock* input: a button that we press to change a signal from 0 to 1 and then, when we let go of the button, back to 0 again. This circuit should count through a sequence of values, from 0 to 15, and then wrap back around to 0 again. Each time you press the *clock* button, the counter moves forward by one value in the sequence.

Your incrementor from Lab-2 will be quite handy here, although you shouldn't use the whole thing at once. Use the 1's bit of your incrementor to build a 1-bit counter; then add the 2's bit, then the 4's, and finally the 8's. Specifically, we will use a memory chip (described below) to hold the value of the counter. That chip's outputs should be wired to your incrementor's inputs (which should no longer be connected to switches); the incrementor's outputs then become the memory chip's inputs. The clock controls when the memory chip adopts new values from the inputs.

Flip-flop chips: The 74LS273 chip contains eight 1-bit *D flip-flops*. That is, it has 8 D inputs paired with 8 Q outputs. 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 ($D0$ to $D7$) and a data output ($Q0$ to $Q7$).

There are two new pins that have not appeared on previous chips for us. The first is the *clock pulse* (CP) 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 a collection of eight 1-bit memories whose values are adopted (a.k.a., *clocked in*) at the same time. We call such a collection of 1-bit memories a *register*, making this chip an *8-bit register*. Thus, when you cycle the input on the CP pin, all of the D inputs are passed through to the Q outputs.

The other new and noteworthy pin on this chip in the *master reset* ($\overline{MR}$) input. When this pin is *asserted* (its value set to 1), the chip's flip-flops will behave normally. However, if this pin is deasserted (set to 0), **all of the flip-flops in the chip will have their value immediately reset to** 0, irrespective of the CP input. I suggest connecting this pin to its own button (see below), allowing you to reset your counter to zero with the press of one button.

1.1 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. Notice that you have already created this combinational circuit in the form of the *incrementor* from Lab 2.

2 Adding one more capability

As an optional, extra challenge: Add one more input E, from a switch. This switch should control whether the counter counts *forward*, incrementing at each step when $E = 0$, or it counts *backward*, decrementing at each step when $E = 1$. Figure out the change in the incrementor logic that would allow this switch to control the direction of the counting.

3 Submitting your work

Once you have a counter working, demonstrate the working circuit:

1. **Take a video:** Capture a video f the circuit by resetting the counter to 0, and then pressing the clock button to count through the full sequence. If you have added the direction-controlling input E, flip that switch and then show it counting backwards.

2. **Submit the video:** Go to the CS submission site at:
   `<https://www.cs.amherst.edu/submit>`
   Login and navigate your way to COSC-163, Lab-3, and submit your video file.

This assignment is due Thursday, Sep-26, at 11:59 pm.