Lab 10 Report

Kiran Nadkarni

Elizabeth

11/8/19

Shift Registers

Objectives:

The goal of this lab is to utilize the flip-flops (specifically d-flip flops) that we have learned about from the modules to create a register circuit using these flip-flops, as well as a debouncer circuit to make sure that the shift register will only register one every time we flip the switch. Like all of our previous hardware labs, this one will test our ability to use LogicWorks to design the circuit and construct it on our protoboards.

Introduction:

Shift registers are, as mentioned before, created using D-flip flops. These devices have the D input, which can directly control the output of the flip flop, the PRE or set which sets the flip flop to 1, the CLR or reset which resets the flip flop to 0, and the CLK or clock which works with the D input to set the output Q to whichever setting D is on for every clock cycle. The outputs are Q and Q' as most flip flops have. We are going to chain these flip flops by connecting their Q's to the next flip flop's D input, and we will do this for 4 flip flops to create our register. We are going to make this circuit synchronous, as in each flip flop will share the same clock switch, but they will each have their own PRE switch and we will design them to share a CLR switch, even though we could do it either way with this.

As mentioned before, we are creating a debouncer circuit to precede our shift register to make sure that we don't have multiple actuations of our shift register for only one switch. We will be making this with NAND gates and two resistors and a switch, which we will represent in our protoboard with a wire attached to a ground that we will move between the resistors to represent the switch itself. The way the debouncer works is that the NAND gates will touch both the set and reset switches to Vcc to give us 1 and 0 respectively. For this lab we are going to be using the SN7474 D-Flip flop chip, which will contain 2 of the 4 flip flops we need, as well as a NAND gate chip that we can put our two NAND gates on.

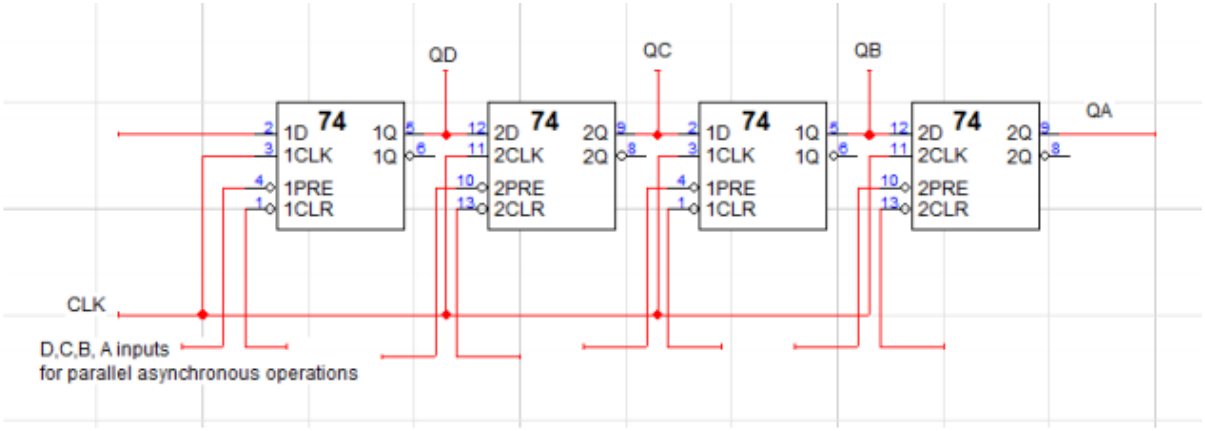

Figure 1: This is a diagram of the general connections of the flip flops to make the shift register. Notice all of the appropriate I/O for each D flip flop, and the QD-QA outputs that will go to our LEDs.

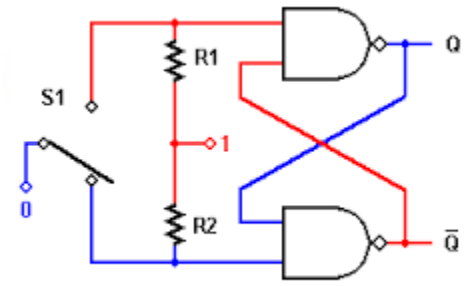

Figure 2: This is the diagram of the debouncer circuit that we will be using. Notice the NAND gates and the resistors, as well as the connection to the 1, which is essentially Vcc. This diagram is color coded as well to represent the affect of the switch touching ground for the Q output, which is the output we attach to the Flip-Flops.

Procedure:

1- The procedure to create this circuit is fairly straightforward. We can follow the diagram set before us in the introduction to create a LogicWorks template that we will then implement on our protoboard. The first step would be to place the flip flops and then connect them by their Q output to D inputs. We are also attaching these connections to binary probes, or our LEDs on the protoboard to see the outputs of the flip flops, with the rightmost bit being the output of the rightmost flip-flop.

2- We then want to attach the CLKs of all of the flip flops together, because we are making a synchronous circuit with a common clock. There will be wires attached to switches labeled D, C, B, and A for the leftmost to rightmost flip-flops as well, which will be attached to the PRE input of the flip flops. The last input is attached to the CLR input, and we are going to use only one switch and chain these inputs together, although it is not necessary to only use one switch for our design, it does simplify it on the physical hardware.

3- Now we are finally creating the debouncer circuit. This is fairly easy to create by following the diagram and explanation in the introduction, but we will start with our NAND gates and connecting one input of each to the output of the other for both of them. The other input of each will be connected to two resistors, with the other side of both resistors both being connected to Vcc. It is important to note that the resistance of the resistors doesn't matter too much, but it is important to have two of the same magnitude. This same, non-Vcc side of the wire will be used as the ends of our switch, with the other end of it being a wire with one end in GND. Using this one end in GND, we will pivot the wire between the two resistors to represent a switch flipping between two resistors.

Of course, in LogicWorks we can use an actual switch, which will make our results more accurate as a result. As a last part, remember that the Q output of the debouncer is what goes to the Flip-Flop portion of the circuit.

Results:

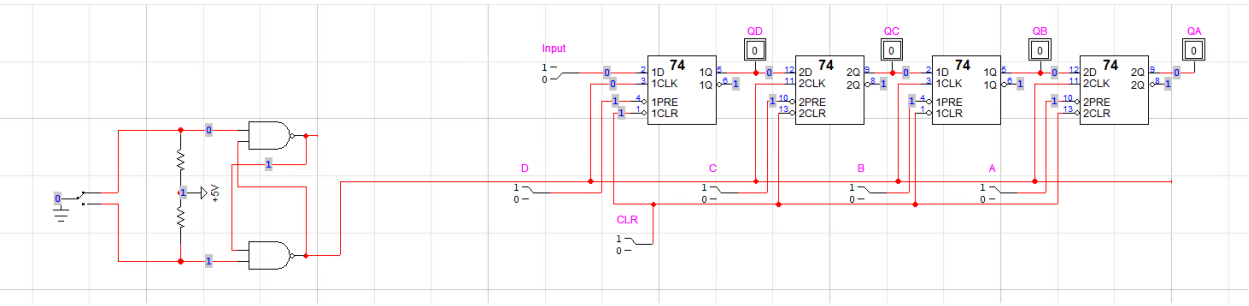

Figure 3: This is my full design of the circuit in LogicWorks. Notice the debouncer circuit, as well as the 4 output bits. There are also the individual set switches for each bit, the synchronous clock and the clear switch, as well as the set switch as D flip flops have.

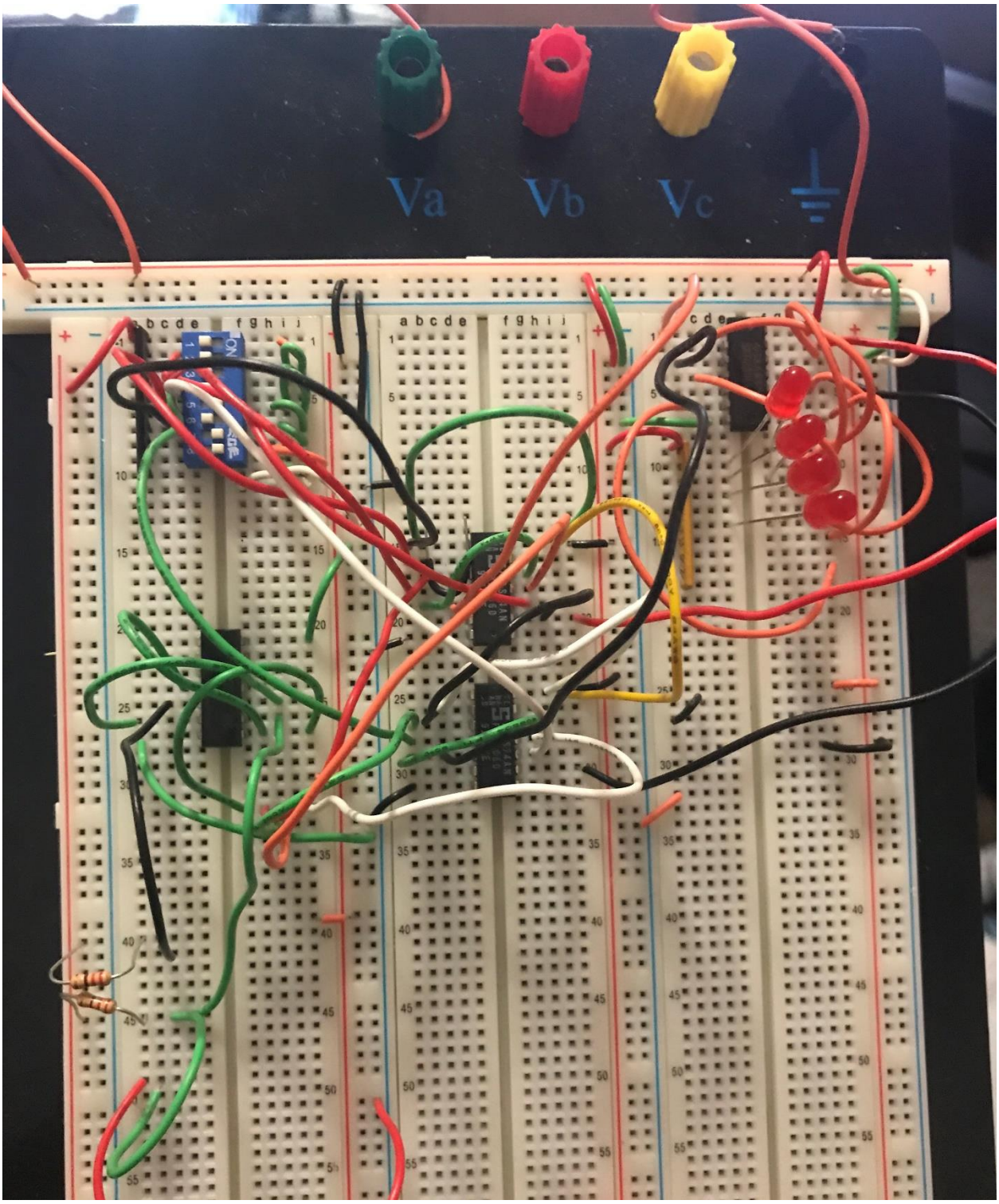

Figure 4: This is the finished circuit on my protoboard. Notice the 4 LEDs that will show our bits, from MSB at top to LSB at bottom. There is also the leftmost chip which is the NAND chip and the two resistors that make the debouncer. The right two chips are the two sets of two d-flip flops, the SN7474 chip.

For a register that is twice the length of mine (8-bits), last two numbers are 88 in decimal, so 1000-1000 in BCD. As stated before, this shows up as 1000-1000 in the register, and if I shift the bits to the right with my input set to 0, the numbers become 0100-0100 in BCD or 44 in binary. I could set the input to 1 as well, which would shift to 1100-0100 which technically would be 0 and 4 in decimal because BCD would not recognize 1100 as a digit from 0-9.

Discussion:

I believe that my results in this lab were accurate, since the circuit only had to perform operations that I would know the answer to. My circuit was able to perform all of the operations with the proper results. Something of note was that the circuit did not work too perfectly from the beginning of my creation of it. I think the hardware for this circuit is fairly delicate so even a slight bump can change the display of the circuitry. I experienced this when I realized all of my wires were attached in the correct spots and were all adjusted, but when I moved the switches too violently, the display of the LEDs changed. Another problem that made this even more tricky was the fact that my switch set did not fit properly in the protoboard. The switch would not sit still when flipping them, and required extreme care to not change the LED output while both serially and parallel inputting data.

Conclusion:

I believe that the hardest part of this lab, as mentioned before, was conducting the tests to make sure my register worked. Any additional movement of the breadboard could change or delete the LED output, and this would constantly make me second-guess the integrity of every part of my circuit. This did however, help me improve it by using the binary probe to figure out where some of the potential errors could be stemming from, and I think that this definitely helped the stability of my circuit in the long run. Overall however, because my circuit did function correctly, and later on I did get the circuit to do this without any hiccups, I would say that this lab was a success.

Question:

A popular use for shift registers is the storage of data, as well as the ability to both input and output data into that storage. The shift register can input data as well, which we have done in two ways, serially and parallel. This controlled data flow in multiple ways is what makes shift registers so useful and popular.

An SN5474 is different from a SN7474 because it is a military-grade chip. This means that it has a wider voltage tolerance range as well as can operate under higher and lower maximum temperatures of -55 degrees C to 125 degrees C.