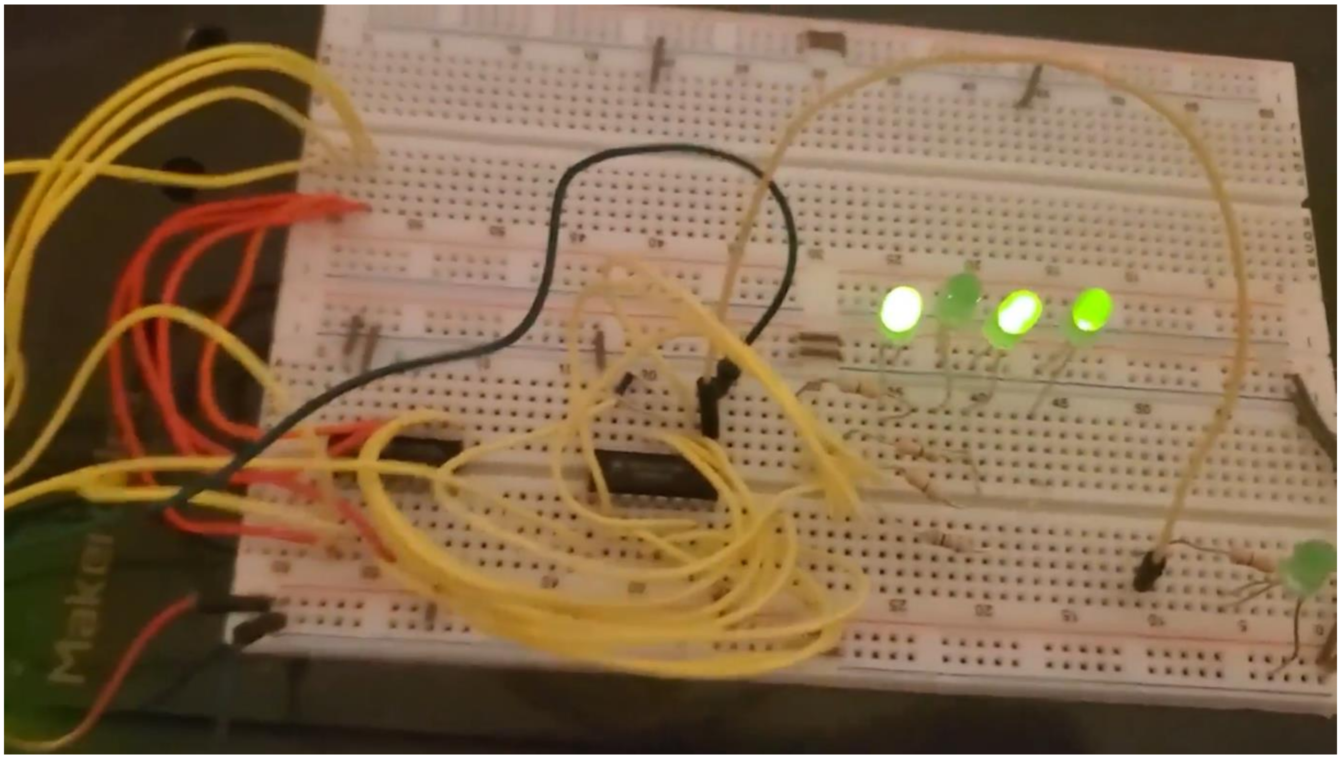
Sequential Logic Applications

ECE2300L Module 5 Report



Kyler Martinez

November 18th, 2020

Introduction

The learning outcomes of the module is for students to acquire the skills necessary to verify the function of a load and shift register and be able to combine their functionality into one circuit. Students would also be able to verify the function of a binary up/down counter and be able to implement these circuits in simulation and using physical devices. Finally, these circuits would need to be reused and altered to be applied to create modulo-n counters and linear feedback shift registers.

Activity 5.1 — Registers

a.

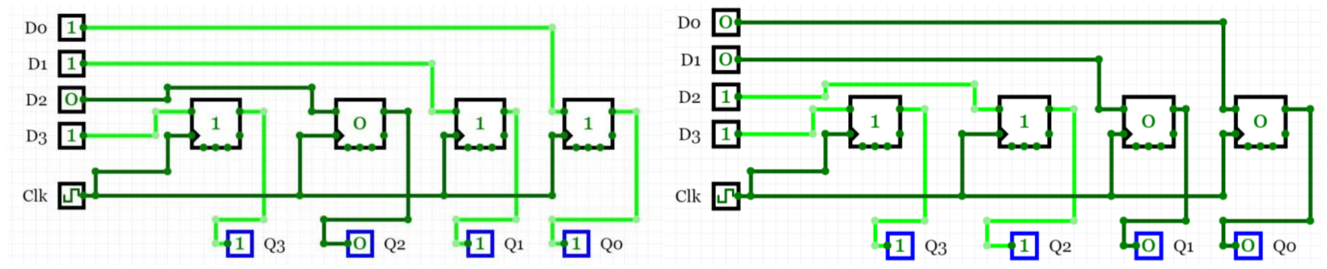


Figure 1: 4-Bit Load Register

To keep $Q_3Q_2Q_1Q_0$ at the same value, $(1011)_2$, come the next clock cycle we would need to keep the D inputs at $(1011)_2$, since if they are changed, then the Q outputs will reflect the inputs. If we wish to increase the output by 1, Q must be $(1100)_2$. This is achievable with this circuit is to configure each input to $(1100)_2$.

b.

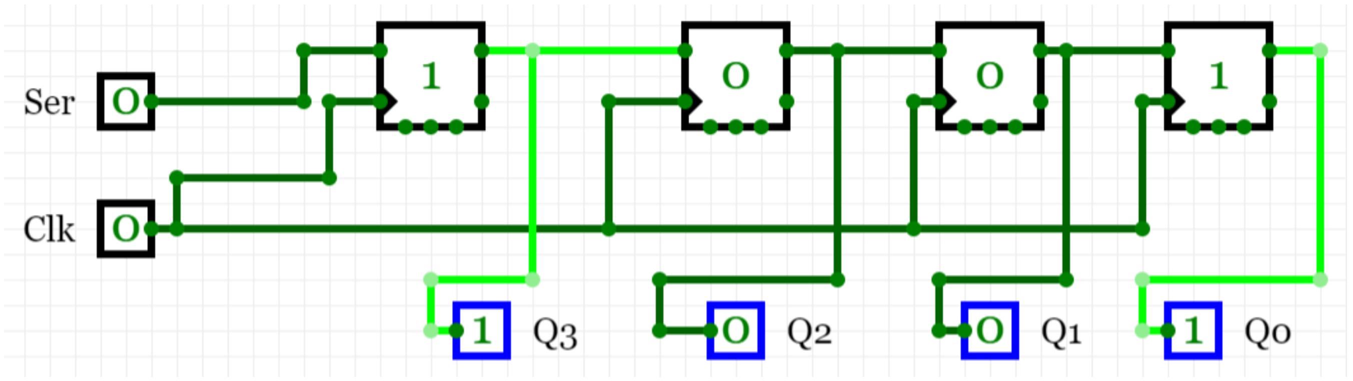


Figure 2: 4-Bit Shift Register

With Q = 1001, if we set Ser = 1, then Q will become 1100 come the following clock cycle, this is due to the fact that the bits move right essentially and the most significant bit is replaced with 1.

If Q = 0110, decimal 6, to get Q to be half its value, decimal 3, we would need to have Ser be equal to logic level zero to shift Q to be $(0011)_2$ before the next clock cycle.

If Q is a negative number in 2's complement, then to divide the number by 2, Ser would need to be put at logic level. If Ser is put at logic level 0, the number would be changed into a positive number if it is put at logic level 1, then the number will remain negative. $(1010)_2$, -6 in decimal, would result in $(1101)_2$ when shifted which is -3 in decimal, showing the proper operation.

c. A 4-Bit load shift register, shown in figure 3, has a function table of. Q^- denotes the previous value before the clock cycle.

Inputs (Set Before Active Clock Edge		Outputs (Values After Active Clock Edge
Shift/Load	Ser	$Q_3Q_2Q_1Q_0$
0	X	$D_3D_2D_1D_0$
1	0	$0Q_3^-Q_2^-Q_1^-$
1	1	$1Q_3^-Q_2^-Q_1^-$

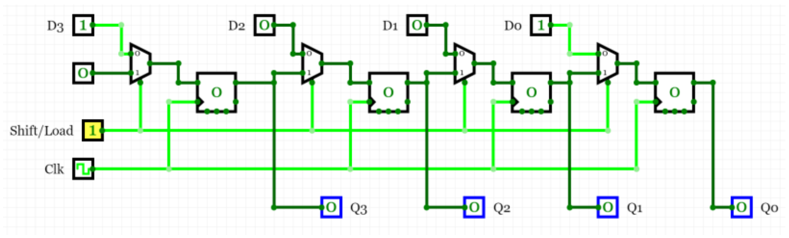


Figure 3: 4-bit Shift/Load' Register

d.

Label	S_Lb	Ser	D	Q
BitWidth	1	1	4	4
1	0	0	0100	xxxx
2	1	1	0000	0100
3	1	1	0000	1010
4	1	0	0000	1101
5	1	1	0000	0110
6	1	0	0000	1011

Figure 4: CircuitVerse Test Bench

Note: Case 6 is only used to verify the result and the shift that comes as a result of case 6 is not necessary for the task given.

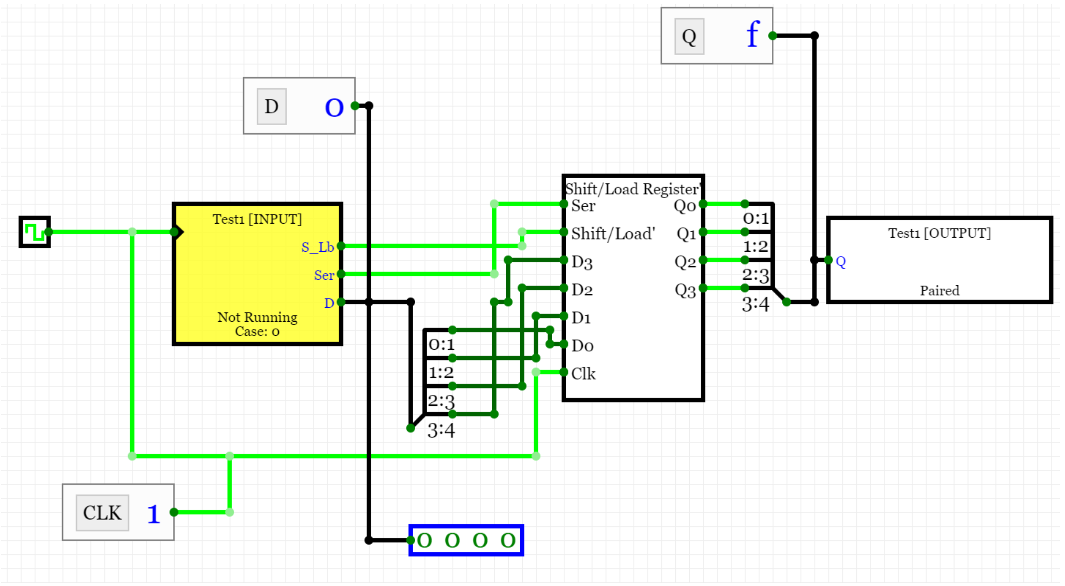


Figure 5: CircuitVerse Testbench Circuit

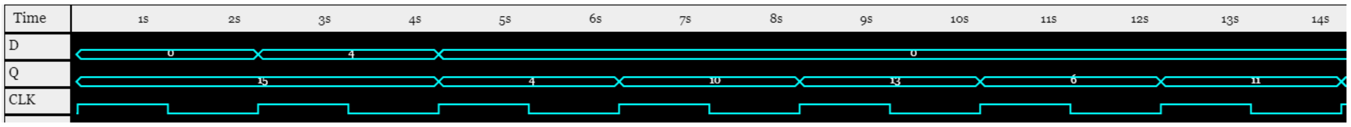


Figure 6: Timing Diagram of Circuit In Figure 5

Note: The timing diagram presents 0 for D for t<2 since the testbench had not received a rising edge while the test bench was running. That portion of the timing diagram can be disregarded.

From the timing diagram, we can see that the output Q is set to $(0100)_2$ or 4 in decimal at the rising edge at ~4s. From there we see that after 4 clock cycles after $(0100)_2$ have been loaded into the register, the output Q becomes decimal 11 or $(1011)_2$, which is the expected output from our description.

e.

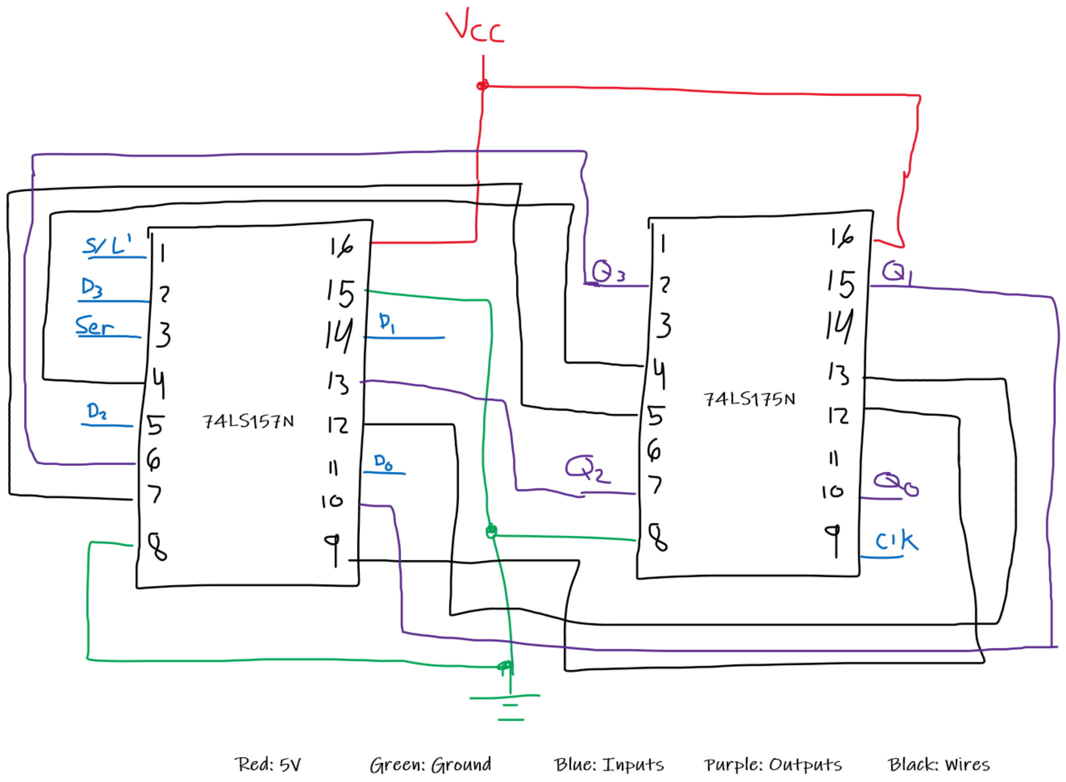


Figure 7: Wiring Diagram With Legend For Colored Wires

f.

Video Demo: https://youtu.be/B7gBg_ayeSU

Activity 5.2 — Binary Counters

a.

Mode	Present Count State Q_1Q_0	Next Count State D_1D_0
0	00	01
0	01	10
0	10	11
0	11	00
1	00	11
1	01	00
1	10	01
1	11	10

$D_0 = Q_0'$ $\qquad$ $D_1 = \text{Mode} \oplus Q_1 \oplus Q_0$

b.

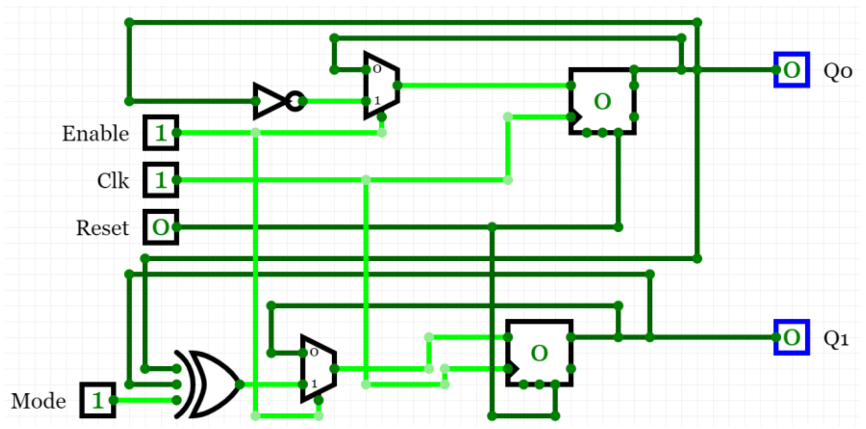
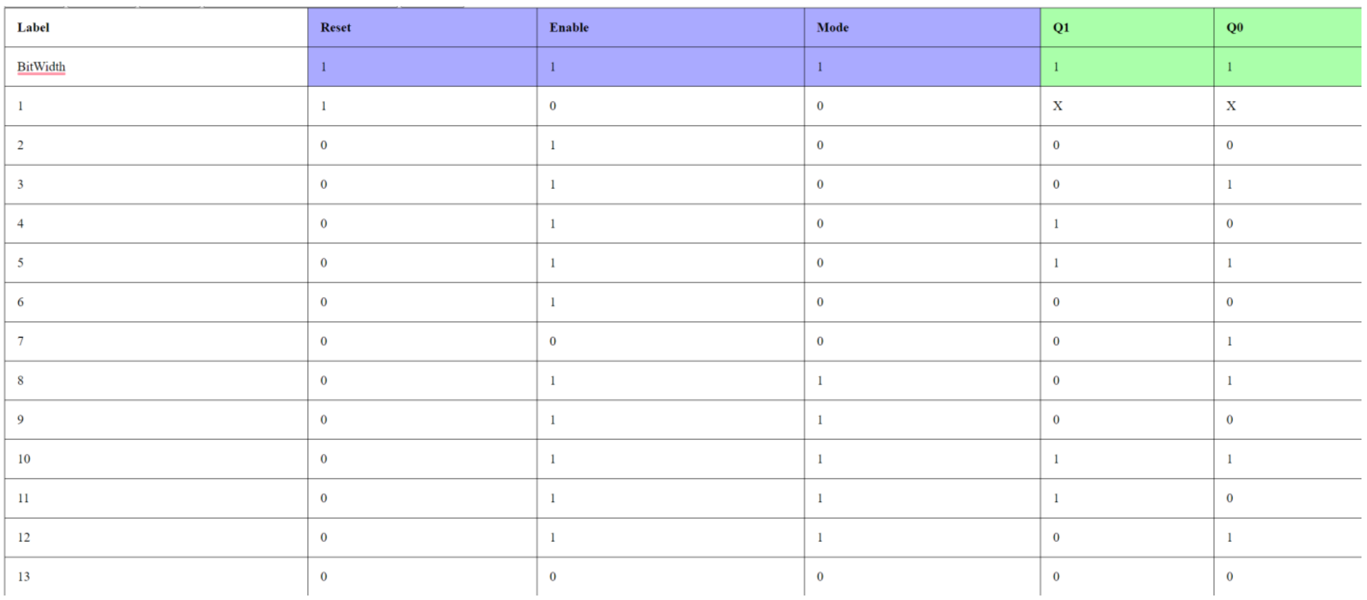


Figure 8: 2-Bit Binary Counter Circuit In CircuitVerse

c.



Label	Reset	Enable	Mode	Q1	Q0
BitWidth	1	1	1	1	1
1	1	0	0	X	X
2	0	1	0	0	0
3	0	1	0	0	1
4	0	1	0	1	0
5	0	1	0	1	1
6	0	1	0	0	0
7	0	0	0	0	1
8	0	1	1	0	1
9	0	1	1	0	0
10	0	1	1	1	1
11	0	1	1	1	0
12	0	1	1	0	1
13	0	0	0	0	0

Figure 9: Testbench For 2-Bit Counter

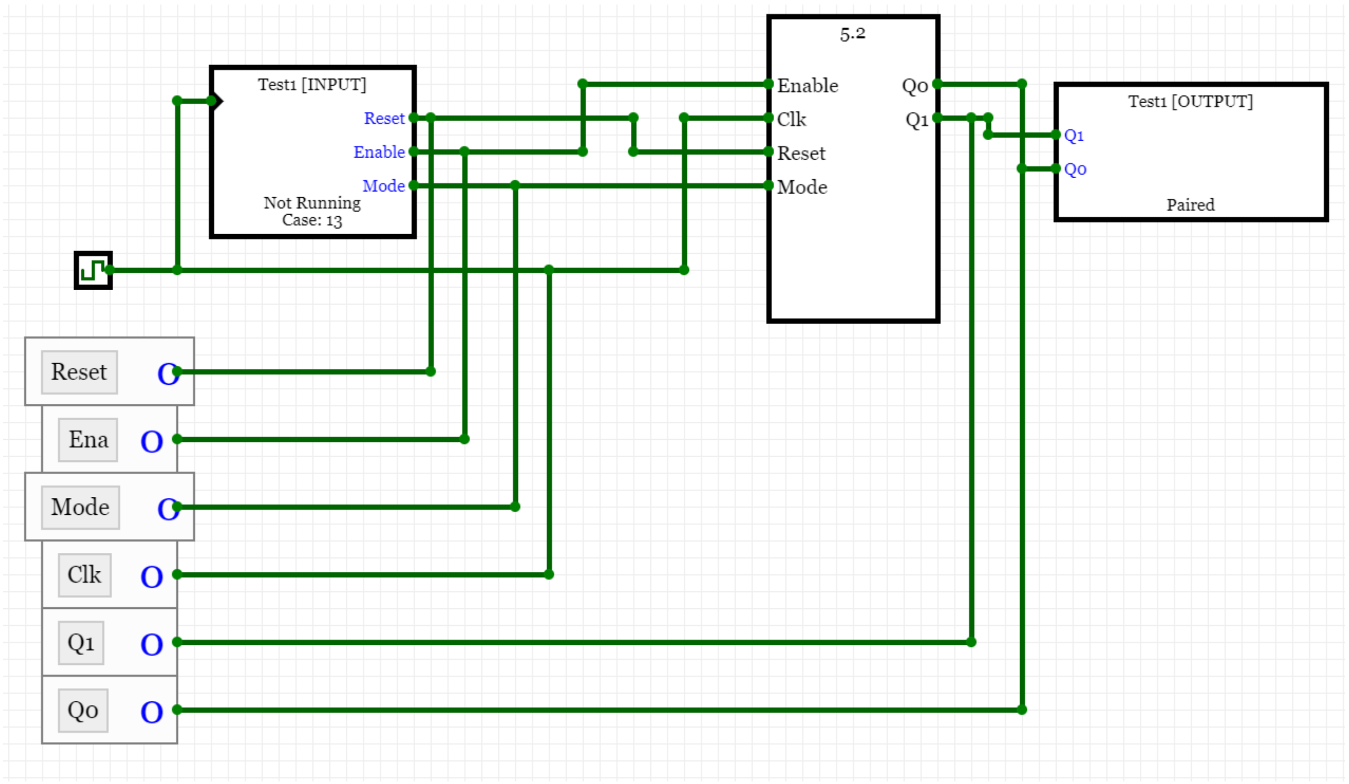


Figure 10: Testbench Circuit

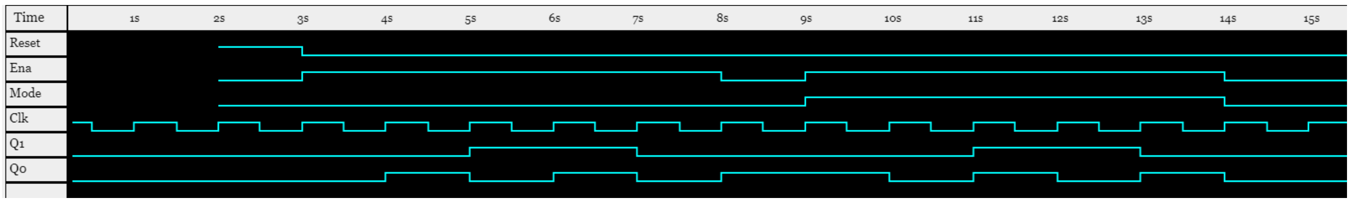


Figure 11: Waveform From Testbench Circuit

From the waveform we can see the circuit initially reset, and then for a few clock cycles, the counter adds up and then holds the previous value. The circuit then decrements the value for a few cycles.

Exercise 5.1 — Linear Feedback Shift Register

a.

In the figure below, I created a Linear Feedback Shift Register with an additional AND gate and input ‘Switch’ that I added to imitate the functionality of a switch since I wanted to pre-load the circuit with $(0001)_2$ and allow for the clock to pass through when I was ready. In the timing diagram, we can see that the register cycles through a sequence of values for the outputs and eventually returns to our initial value after going through all possible combinations except for $(0000)_2$ since decimal 0 causes the circuit to remain at a standstill..

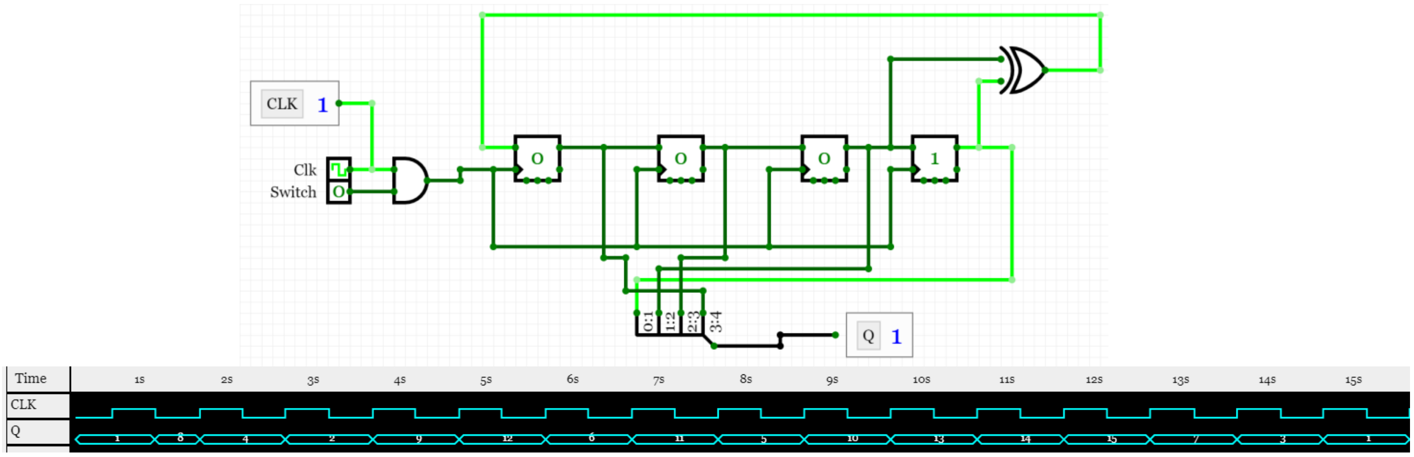


Figure 12: Linear Feedback Shift Register & Timing Diagram

b.

To correct the stalling of the register when the outputs are $(0000)_2$, I wanted to make a solution that provided as little intrusion as possible to prevent disturbing the ordinary function of the circuit. I only wanted to stop the ordinary function of the circuit if the outputs were $(0000)_2$. I created a bit that would contain whether or not the outputs were $(0000)_2$ or not and accomplished this by putting all the Q’ outputs through a 4-input AND gate. I then used a 2-1 Multiplexor with the AND gate’s output as the selection and ‘1’ input while the XOR gate was the ‘0’ input and the multiplexor’s output became the feedback output. The multiplexor allows for the circuit to carry its ordinary function when Q does not equal $(0000)_2$. If Q equals $(0000)_2$, then a 1 will be shifted into the register and the circuit will exit the stalemate. In the figures below, we can see the initial Q input to be decimal 0 and the mux has an output of 1 that’s ready to be shifted in. The circuit then goes through the sequence of outputs and repeats.

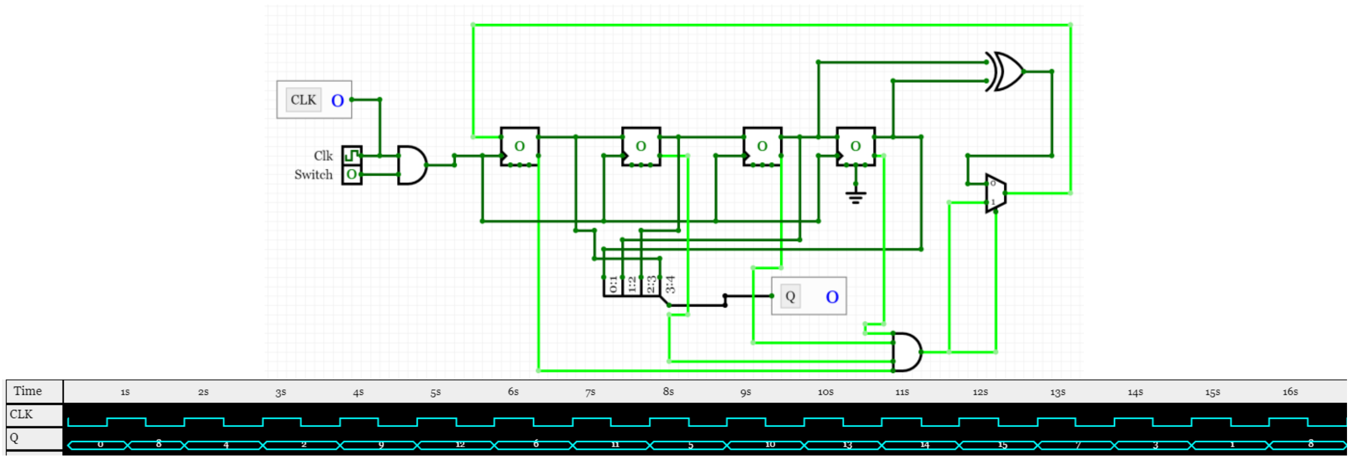


Figure 13: Linear Feedback Shift Register With Correction Logic & Timing Diagram

Exercise 5.2 — Modulo-n Counters

a.

4-Bit Counter Function Table

Note: The clock and input D, composed of $D_3D_2D_1D_0$, are implied in the table. Q^- denotes the previous state of Q.

Load	Enable	Q ($Q_3Q_2Q_1Q_0$)	Q^+ ($Q_3Q_2Q_1Q_0$)
X	0	$Q_3Q_2Q_1Q_0$	$Q_3^-Q_2^-Q_1^-Q_0^-$
0	1	0000	0001
0	1	0001	0010
0	1	0010	0011
0	1	0011	0100
0	1	0100	0101
0	1	0101	0110
0	1	0110	0111
0	1	0111	1000
0	1	1000	1001
0	1	1001	1010
0	1	1010	1011
0	1	1011	1100
0	1	1100	1101
0	1	1101	1110
0	1	1110	1111
0	1	1111	0000
1	1	XXXX	$D_3D_2D_1D_0$

b.

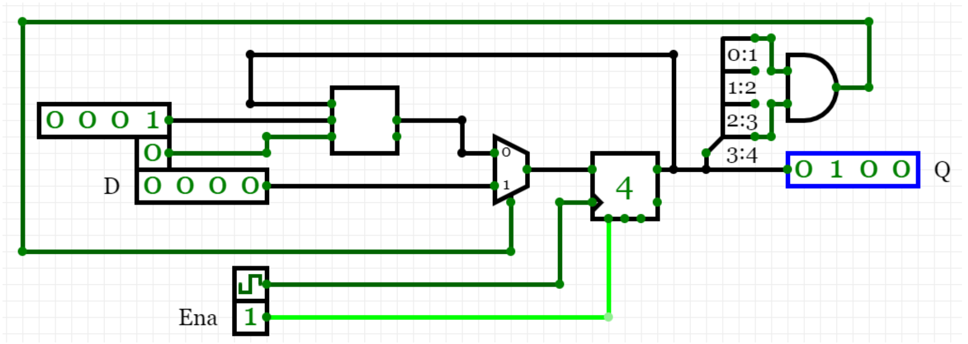


Figure 14: Mod-10 Counter Diagram

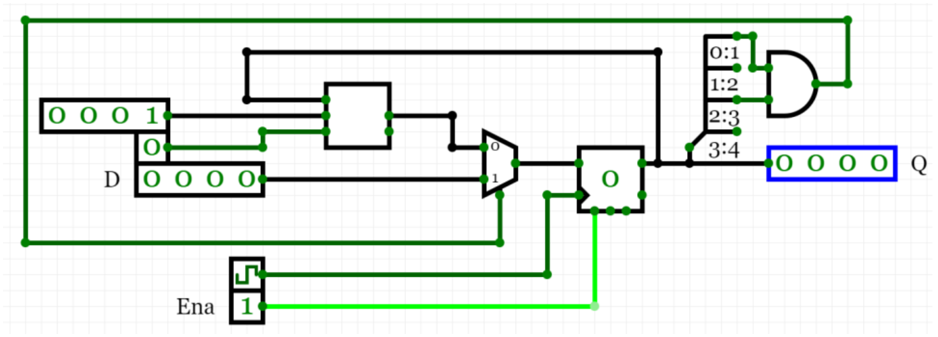


Figure 15: Mod-6 Counter Diagram

c.

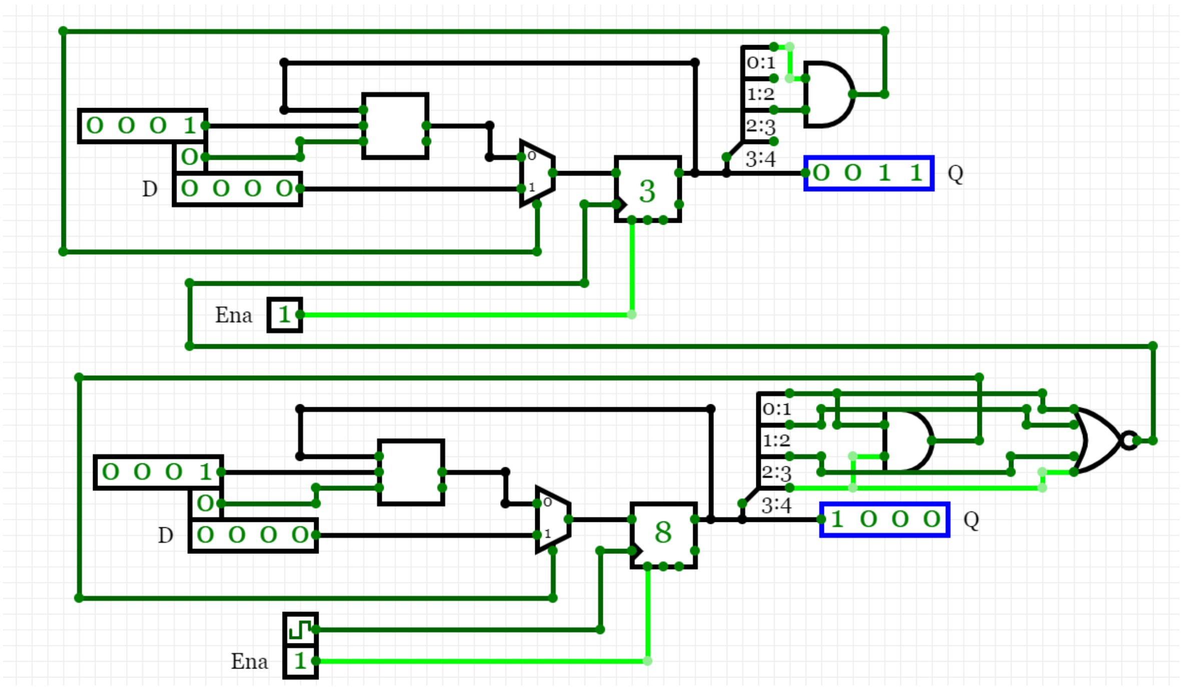


Figure 16: Mod 60 Counter Imitating A Clock

The NOR gate was added in order to put the two counter circuit in sync with each other since the clock of the mod-6 counter will get a rising edge when output of the mod-6 counter is $(0000)_2$. This causes the mod-6 counter to increment after the mod-10 counter increments a 9 value.

Exercise 5.3 – Serial Adder

a.

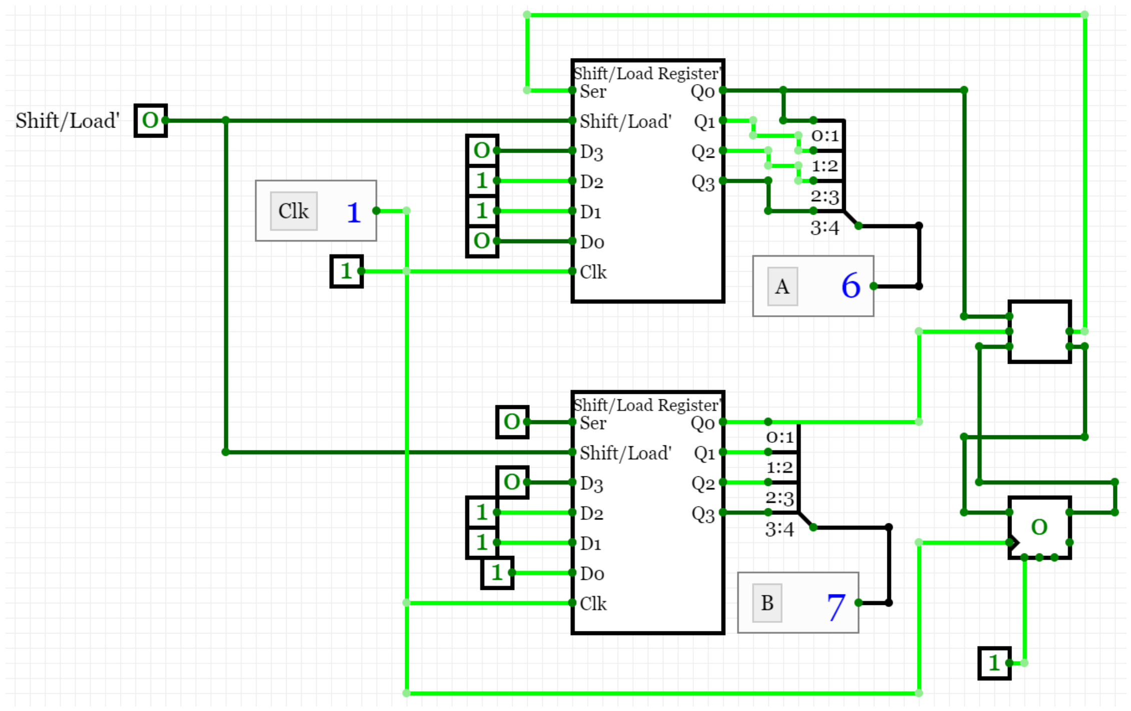


Figure 17: Serial Adder

b.

Loading A = $(0110)_2$ and B = $(0111)_2$ after 4 clock cycles A = $(1101)_2$ which is equal to the initial A value plus B.

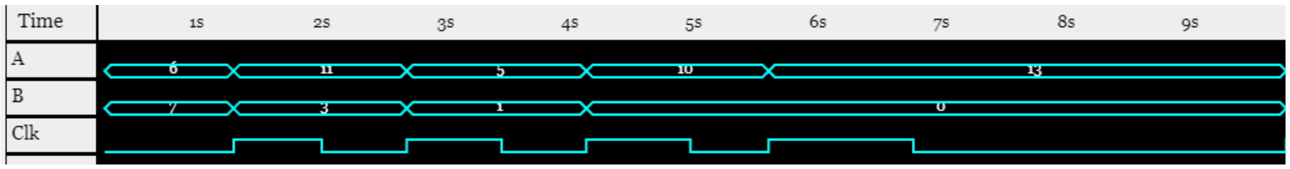


Figure 18: Timing Diagram