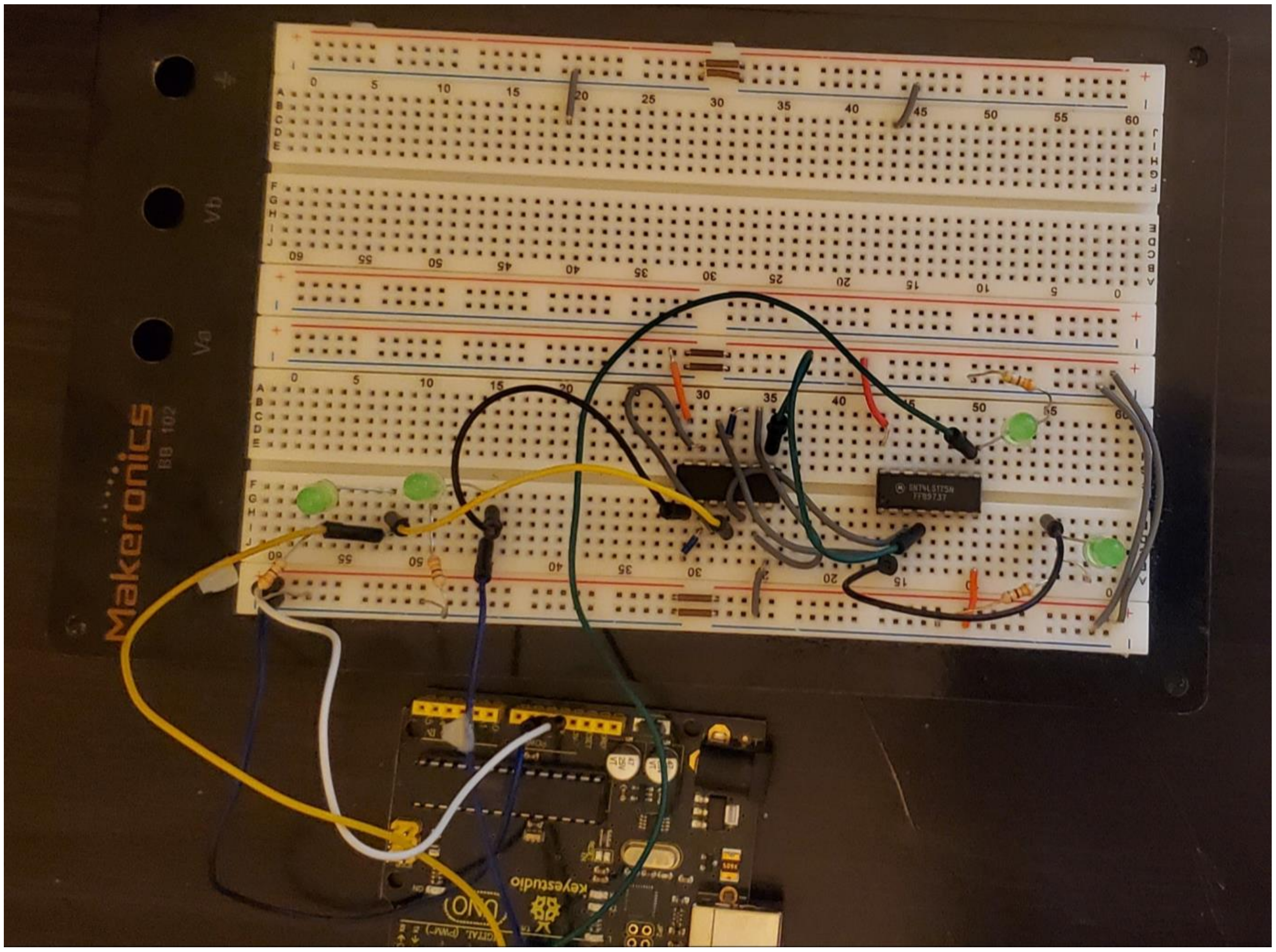
Sequential Building Blocks

ECE2300L Module 4 Report



Kyler Martinez

November 4th, 2020

Introduction

The objectives of the lab are to learn the basic functions and types of sequential logic building blocks. These building blocks include latches and flip flops. The skills acquired in the lab are to identify the various types of latches and discuss the advantages and disadvantages of certain types, including racing conditions. Other skills include being able to identify the different flip flops and how they differ among themselves and the latches. Finally, these sequential logic circuits will be built and simulated to verify their behavior and analyze the feedback paths.

Activity 4.1 — Latches

a. Using CircuitVerse the SR latch in figure 1 and the following table was completed.

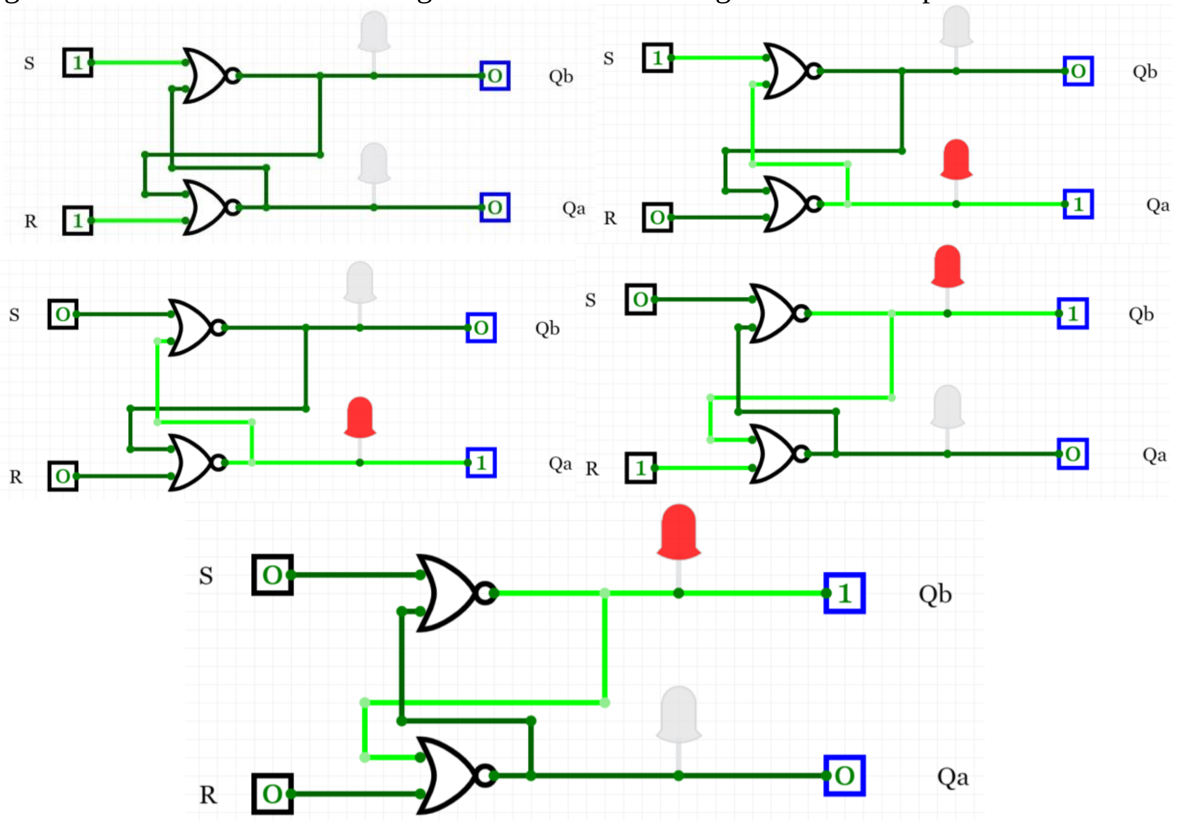


Figure 1: SR Latch in CircuitVerse

Step	Qa	Qb	Explanation
S =1, R=1	0	0	Initially both outputs are 0, and when the logic 1 is applied to the inputs the NOR gates produces an output of 0 which results in the outputs being at logic level 0.
S =1, R=0	1	0	When R is changed to zero, the output of the lower NOR gate becomes logic 1 with (R + Qb$^-$)’ which influences the upper gate to be at logic 0.
S =0, R=0	1	0	When S is put to 0, the output of the upper NOR gate is (S+Qa$^-$)’ which is equal to 0. The output is the same as the previous step thus no change in the lower NOR gate.
S =0, R=1	0	1	When R is put to 1, the output of the lower gate becomes 0 which causes the output of the upper NOR gate to be 1, creating a steady state of Q_a = 0 and Q_b = 1.
S =0, R=0	0	1	When R is put to 0, the output of the lower NOR gate is the same which does not cause a change in the upper gate. Resulting in the same output values as the previous step.

Note: x$^-$ is used to denote the previous state of the variable before any changes at this stage.

b. Two SR latch circuits were created in CircuitVerse with one gate having a slighter lower delay than the other. The inputs were both tied to logic level one and then changed to logic level the results were recorded.

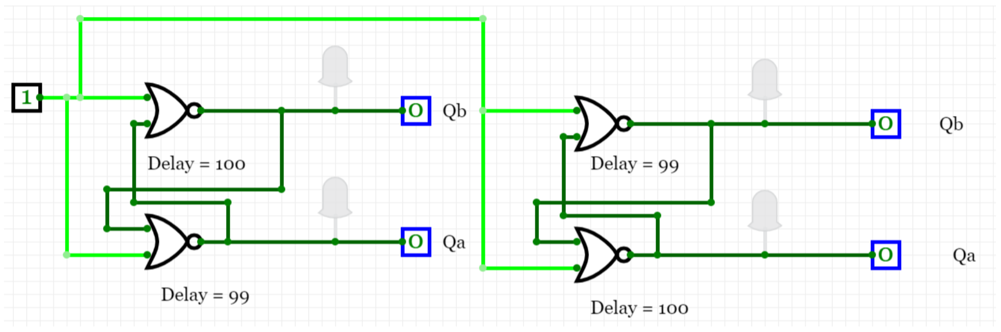


Figure 2: SR Latch Circuits Tied To 1 With Delays Noted

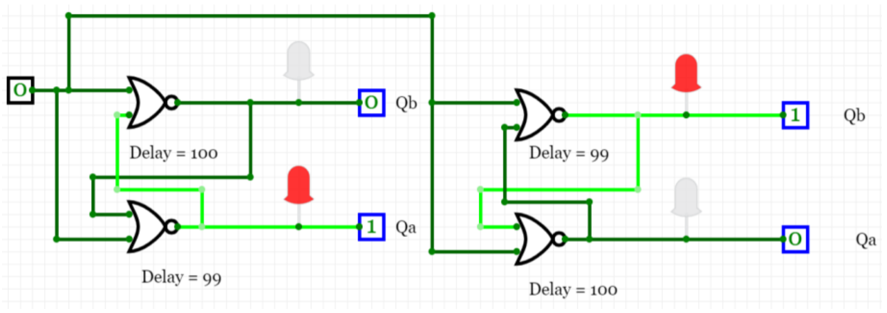


Figure 3: SR Latch Circuits Tied To 0 With Delays Noted

When there is a delay the gate with a lower delay will output first and when the output is set to logic level zero, the gate that has the lower delay will output logic level 1 first and then feed this output into the input of the gate with the greater delay. This ultimately allows for Qa to be at logic level 1 in the circuit where the lower NOR gate is at a lower delay and the other circuit to have Qb at logic level one where the upper NOR gate has a lower delay.

c.

Often having S and R at logic one is an invalid input combination since it is typically stated that they must be compliments of each. The input combination produces the outputs of 0, which is completely valid for the SR latch. However it is not desirable to use this input since the step after setting S and R to logic one can produce the racing effect seen in figure 3 if the outputs are both set to 0 at the same time. Often, we cannot know the exact delay of each gate to compensate for this we result in the situation where the designer would not know the exact output of the SR latch. One way to avoid this is to intentionally create a delay in a gate however this would defeat the purpose of the issue arising from two gates with ideally no delay.

d.

Characteristic Table

S	R	Qa	Qb
0	0	Hold Previous Values	
0	1	0	1
1	0	1	0
1	1	0	0

e.
The circuit in figure 4 is called a gated D latch and is an alternative version of an SR latch that removes some issues of the SR latch while keeping the memory capability.

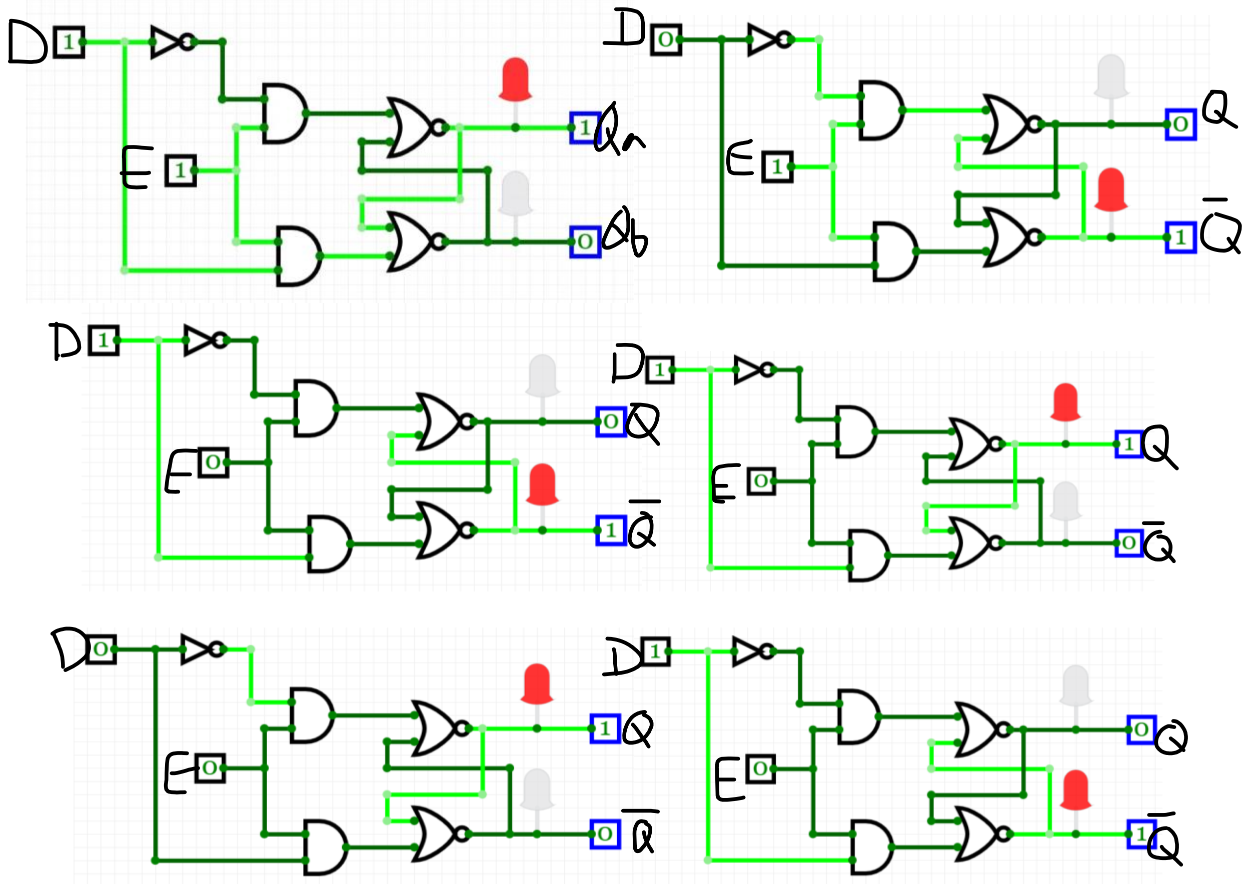


Figure 4: Gated D Latch

The purpose of the E input is to essentially enable the latch since when E = 0, the outputs will not change, and the latch essentially hold onto the outputs. This is caused by the AND gates that are connected to each input of the NOR gates. If E = 1, then the latch would work as normal and alter the data. The purpose of the inverter is to make the inputs of the NOR gates be complemented. This eliminates the possibility of both inputs being at logic level 1 and creating the racing issue discussed previously. However, it prevents the user from making both the open inputs of the NOR gate the same and producing 0 for both outputs. When the E input is at logic 1, Qa will follow the logic level of D and Qb will be the compliment of Qa. This leads to the notation running as Q and Q’ as opposed to Qa and Qb.

f.

<u>Characteristic Table</u>

D	E	Qa (Q)	Qb (Q)’
x	0	Hold Previous Values	
0	1	0	1
1	1	1	0

Activity 4.2 — D Flip Flop

A D Flip Flop was constructed in CircuitVerse along with a timing diagram to explore the behavior of the D Flip Flop. The Flip Flop was configured to be negative-edge triggered. To accomplish this, an NOT gate was added to the clock input of the DFF.

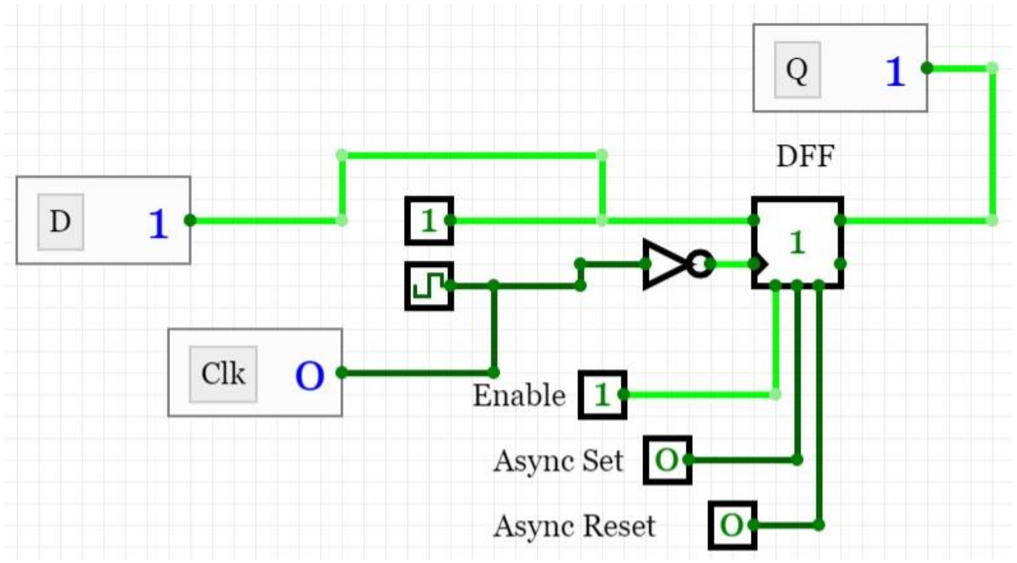


Figure 5: Negative Edge D Flip Flop Circuit In CircuitVerse

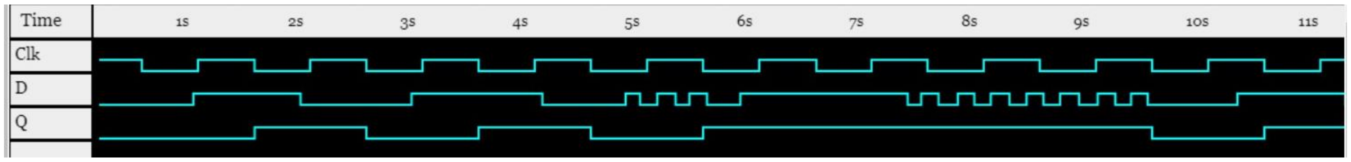


Figure 6: Time Diagram

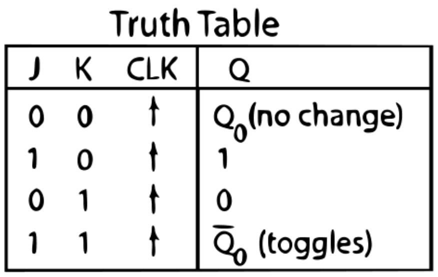
From the diagram we can see that Q copies D only at the time when the clock is switching from high to low. It is not apparent until the time between 1 and 2 seconds where Q comes to logic high. It maintains logic high until sometime in between 2 and 3 seconds where Q copies the value of D which has been at logic low for some time. At any time where the clock is not changing from logic level one to logic level zero, the value of Q maintains its previous state until the opportunity to change is presented. This helps prevent glitches from appearing in our output since the memory can only change at specific time points and these can either be infrequent or frequent depending on the period of the clock source. It does not fully prevent glitches since a glitch can momentarily occur when Q is reacting to an input. If we consider the section between ~7.5s to ~9.5s to be a glitch then we can see that the glitch managed to be represented in the Q output the entire interval due to lining up with the clock source. Two of the glitches between 4.5s to 5.5s were not seen in the output, however the last one did get represented in the memory state.

Activity 4.3 — Flip Flop Conversion

a.

D Flip Flop Excitation Table

Q	Q^+	D
0	0	0
0	1	1
1	0	0
1	1	1



Truth Table

J	K	CLK	Q
0	0	↑	Q_0(no change)
1	0	↑	1
0	1	↑	0
1	1	↑	$\bar{Q}_0$ (toggles)

b.

J	K	Q	Q^+	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

J \ KQ	00	01	11	10
0	0	1	0	0
1	1	1	0	1

D = K'Q + JQ'

c.

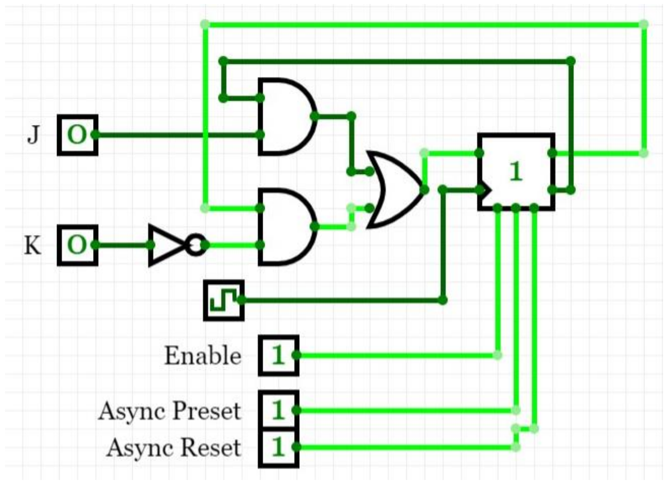


Figure 7: CircuitVerse Schematic Of Circuit.

d. A the circuit shown in figure 7 using a SN74LS175 and an SN74LS00. Various input combinations were tested in the video demo.

Video demo link: https://youtu.be/pIA9VU_S5ts

e. The flip flop was run with it configured as a JK flip flop with JK tied together. A push button was used to deliver power to them.

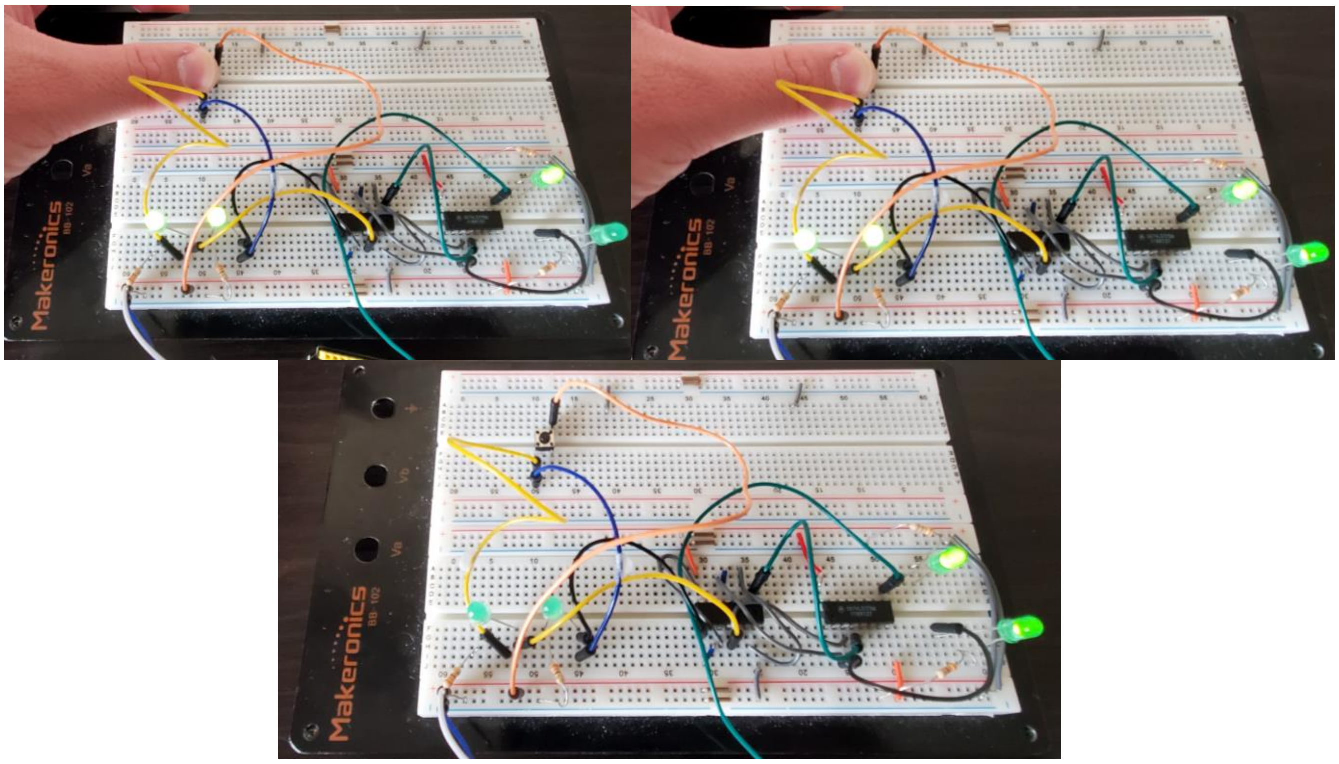
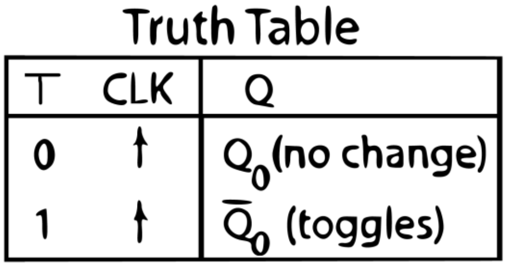


Figure 8: Pictures Of Tied Together Flip Flop At Various Moments

When the inputs are tied together and at logic level 1, Q will take on the opposite state of the previous state when triggered. When the inputs are at logic level 0, Q will retain its previous state. This behavior is that of a T flip flop since when T is at logic 1, Q will take the opposite state of the previous state and at logic 0, will retain its memory.

f.
Using the truth table of the T flip flop and the excitation table of a D flip flop, the conversion of tying the inputs together can be validated.

T	Q	Q^+	J	K
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	0	1	1


Truth Table

T	CLK	Q
0	$\uparrow$	Q_0 (no change)
1	$\uparrow$	$\bar{Q}_0$ (toggles)

From the truth table we can see that by typing the inputs of J and K together we can achieve the effect of a T flip flop using a JK flip flop, converted from a D flip flop.

Exercise 4.1 — Combinational vs. Sequential Logic

The difference between a combinational and sequential circuit is that combinational logic circuits has outputs that only depend on the possible combinations of the outputs and produces and allows for only one possible output combination per input combination. However sequential logic circuits have outputs or components that depend on the previous state of the circuit and can result in situations where multiple outputs can be generated for one input combination depending on the previous state of the circuit. This creates the sequence feature of a sequential circuit.

Exercise 4.2 — Another Look at the Mux

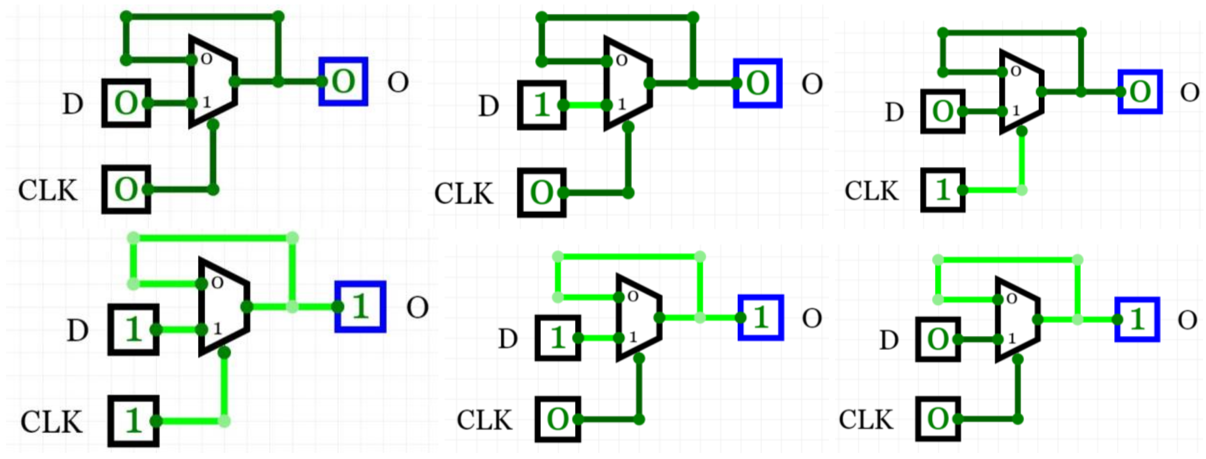


Figure 9: Multiplexor From CircuitVerse At Various Input & Sequence Combinations

a.

Characteristic Table

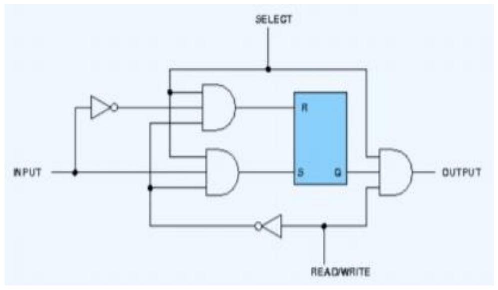
D	CLK	O
x	0	Hold Previous Value
0	1	0
1	1	1

When CLK is at logic level 1 the output is allowed to change and matches the logic level of D. When CLK is at logic level 0, the multiplexor will follow the logic level of the output.

b.

The multiplexor with one of the inputs following the output of the multiplexor emulates the gated D latch since the selector acts as the enable input fo the gated D latch, and the other input, also marked as D, acts as the D input where one of the outputs, the only output for the multiplexor, follows the logic level of D.

Exercise 4.3 - Memory Cell



a. For the logic diagram on the right, to write a 0 or 1 into the memory cell, the input would need to be set to the proper logic value first and then select input set to logic one. This enables the two AND gates and sets the output of the SR latch. Then the logic level of the read/write input needs to be set to logic level 1 to output the memory cell. This enables the output AND gate and allows the user to see the stored memory of the SR latch. If the read/write input is at logic level 0 then the memory cell can be altered. If the select input is at 0 before the read/write is set to 1, then the memory cell will not be altered until it is set to logic 1. If the select input is at 0 and the read/write pin is at 1, then the memory cell's data will not be displayed, but the memory cell's data will not be altered.

b.
For my functional table I added an additional output of the memory cell (SR latch) to properly describe the behavior of the circuit.

Characteristic Table

Input	Select	Read/Write	Memory Cell	Output
x	0	0	Holds Previous Value	0
0	1	0	0	0
1	1	0	1	0
x	0	1	Holds Previous Value	0
x	1	1	Holds Previous Value	Memory Cell Logic Level

c.
The cell circuit show was created in circuit verse and is the basic state with logic level one stored in the memory cell. It is outputting the memory cell.

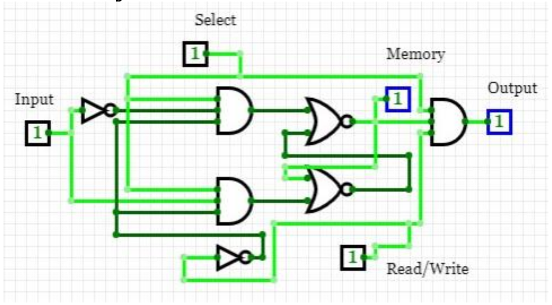


Figure 10: Memory Cell Circuit Designed In CircuitVerse

If the memeory cell is not selected , the memeory is not affected and the output reads logic level zero, even with the input and read/write inputs changing.

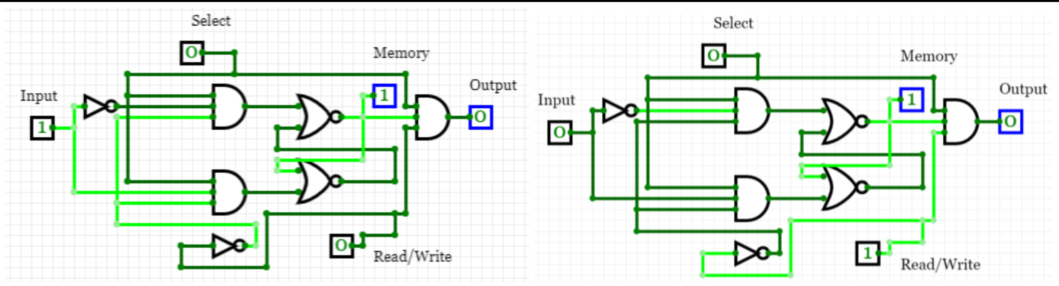


Figure 11: Circuit Output When Select Is At Logic Low (Not Selected)

If the memeory cell is set to write, the input is set to logic zero, the memory will only change to the value of the input if the memeory cell is selected. If the memeory cell is not selected the memeory will retian it's data.

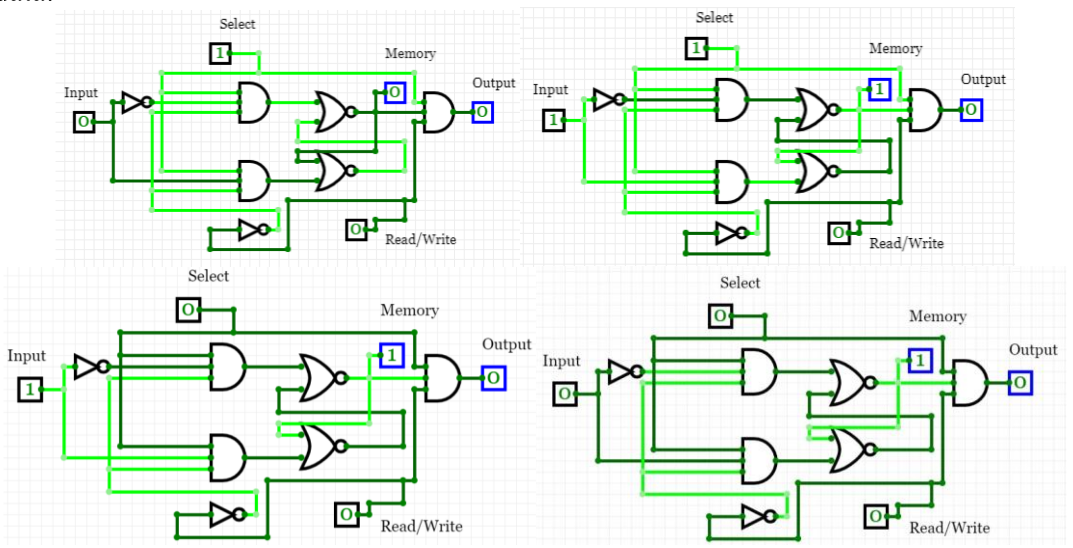


Figure 12: Circuit Output When The Memory Cell Is At Write For Various Conditions

When the cell i selected for read, the output will show the logic level of the memory if also the select is at logic level high. However if the select input is at logic level 0, the cell is not slected, then the output will be 0 regardless of the memeory cell. The input has no effect on the circuit.

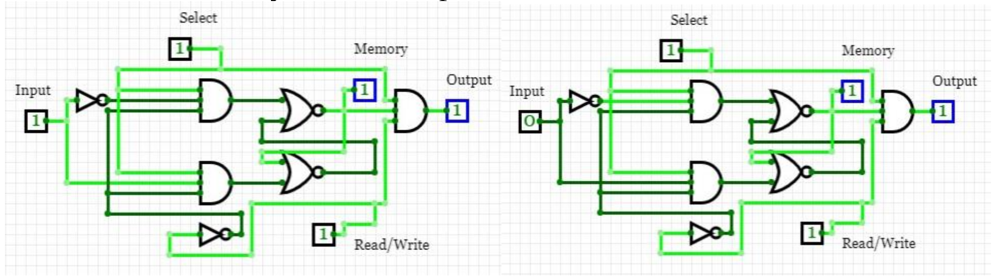


Figure 13: Circuit Output When The Memory Cell Is Set At Read For Various Conditions

Overall, for the memory cell to write, the select input must be at logic level 1, and the read and write input at logic 0. For the cell to read, the read/write pin must be at logic level 1 and the select pin must be at logic level one to show the logic level of the memory cell in the output.

Exercise 4.4 - Latch vs. Flip Flop

a.
The difference between a flip flop and a latch is what triggers the memory saving capability of it, latches are level sensitive and flip flops are edge sensitive. A latch will store a bit of information when there is a change in the logic level of the input, or inputs, but will retain its memory if there is no change. However, flip flops will store information when a clock signal changes from either logic level 1 to logic level 0 or from logic level 0 to logic level 1 depending on the configuration of the latch. The memory will not change if the input is not at an edge state proper for the circuit configuration.

b.

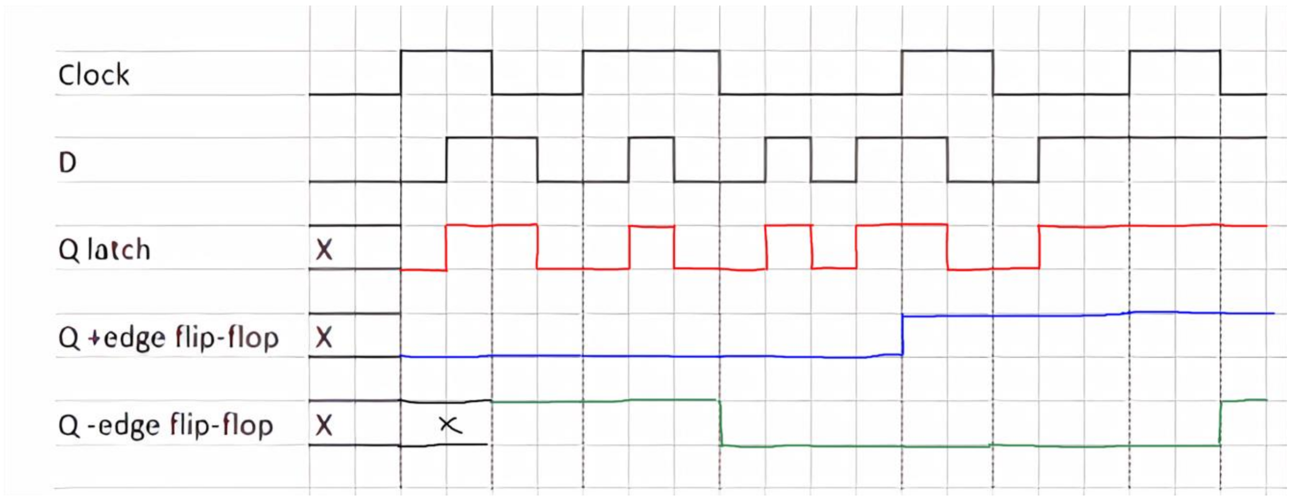


Figure 14: Time Diagram For Flip Flops & Latch

Exercise 4.5 - Flip Flop Applications

a.

The circuit shown below uses a D Flip Flop with an inverted Q output as the input. At the positive edge of the clock cycle the output Q changed from logic 0 to logic 1 and logic 1 to logic 0. This is because after each clock cycle the input for the next cycle is the inverse of Q which allow for the flipflop to store different values. If the input was not inverted, the output Q would always stay at either logic level low or logic level high, depending on the initial state of Q. Q changes at the positive edge of the clock cycle since the Flip Flop is designed to be edge level sensitive and the flip flop is configured for positive edge triggering.

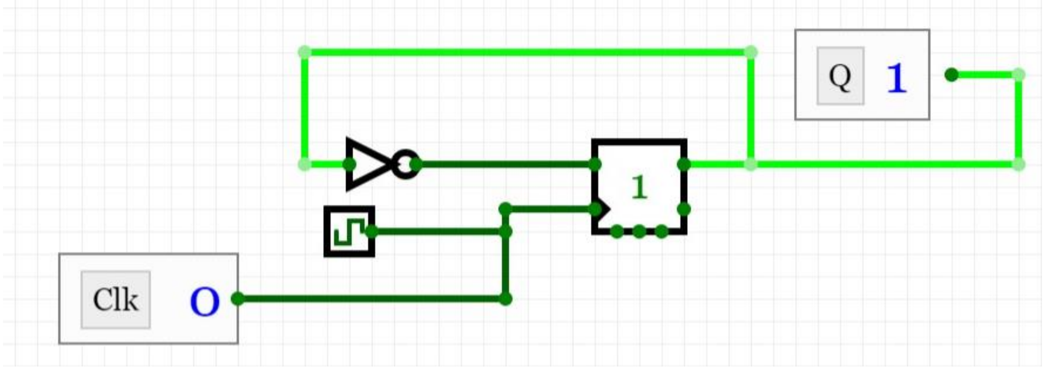


Figure 15: CircuitVerse Circuit Of Flip Flop Application

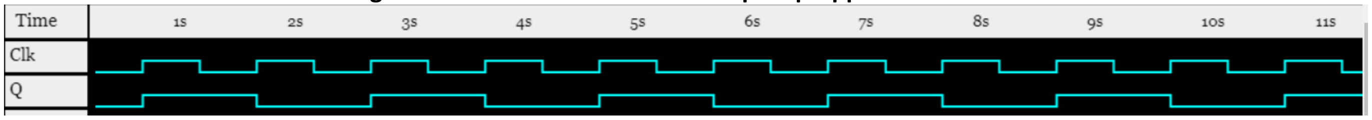


Figure 16: Timing Diagram

b.

If the clock frequency were 10Hz, the period would be .1 seconds. Since the output Q can only change once a period due to being triggered by the positive edge, Q will be at logic level high for one period of Clk and at logic level zero for one period of Clk. This in turn makes the period of the Q output .2 seconds and a frequency of 5 Hz.