# Week 5 Lab A: Sequential Circuits

## Objectives

Develop understanding and experience of:

1. Latches and Flip-flops to store one bit of data.

## Sequential Circuits

* What is the difference between sequential and combinatorial circuits?

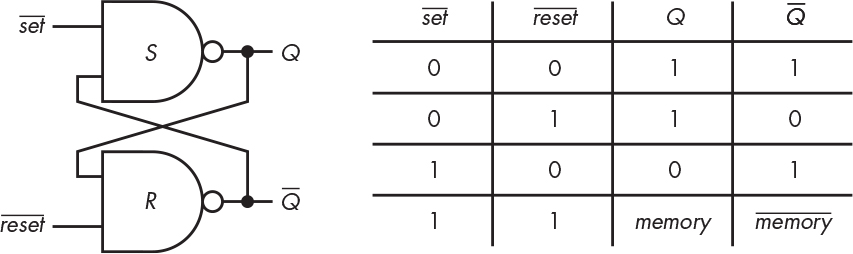
A Combinational Circuit is a type of circuit in which the output is independent of time and only relies on the input present at that instant. A Sequential circuit is a type of circuit output can also depend on previous outputs as well as the current inputs.

Why do we need sequential circuits?

* What does a horizontal line across an input/output represent? For example, A̅.

Means NOT a so output will be inverted.

## Latch



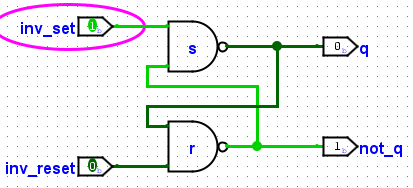


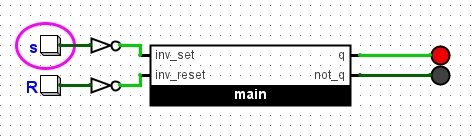
Figure taken from from Steinhart (2019)

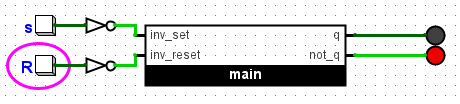
1. Create a latch as shown in the figure (without any NOT gates) and label your input and output pins. Note that the lines above indicate that they are the inverted set and reset signals so your labels might read inv\_Set and inv\_Reset (or NOT\_S and NOT\_R) to indicate that they are inverted. Name the circuit SR\_latch. The next task creates a way to test the circuit.
2. Create a new circuit (named test\_latch) to test the latch within the same overall Logisim Evolution file so that you can use the latch as a sub-circuit.

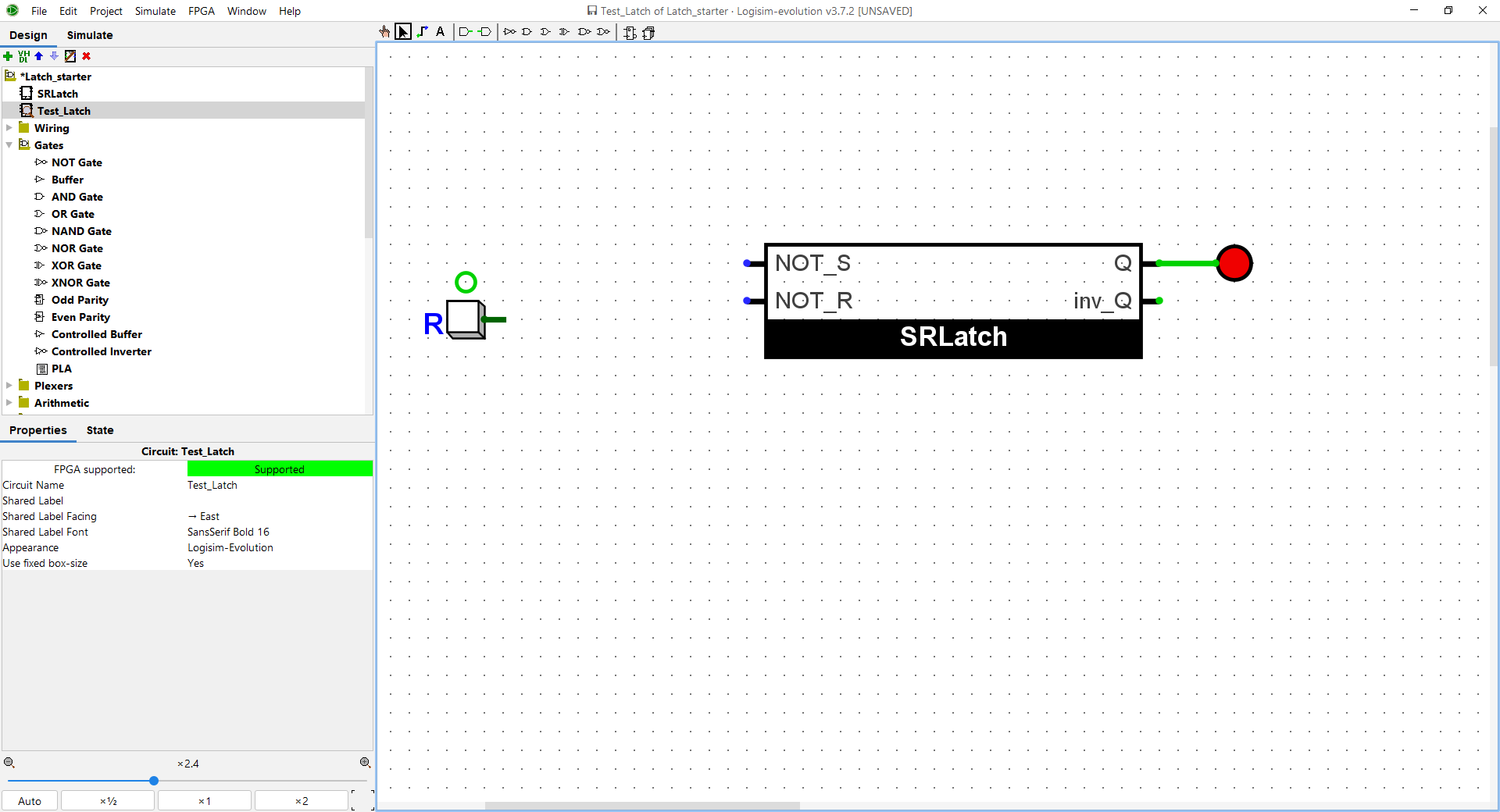
The new circuit should have two buttons (one to set the latch to 1, and the other to reset the latch to zero). Buttons are in the Input/Output section in Logisim Evolution. Note that the signal from the buttons is normally off. When you press a button, it gives a brief signal of 1 (a pulse) along the connected wire while the mouse is pressed.

The signals from the buttons need to be ***inverted*** to connect them to the respective inputs to the latch. Connect an LED from the Q output of the latch. Test the circuit.

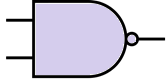
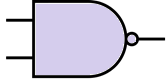
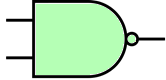
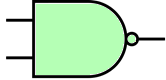
Add images of your latch and the test circuit to this document.







## Flip Flop



*Q*

*Q*

Data

Clock



Figure shows a flip-flop constructed from three latches adapted from Figure 3.9 in Steinhart (2019).

1. Create another circuit in the same project and use the latch you created in task a as a **sub-circuit** to create the flip-flop shown above, do **not** construct it from scratch from NAND gates. The figure above adapted from Steinhart (2019) shows the three separate latches in different colours. For the clock input, you can use a pin, or a clock from the wiring tools in Logisim Evolution.

When using our existing latch, we must cater for one of the NAND gates having 3 inputs, the red input in the figure above, but our existing latch has only two. We want to combine the signals that go in at so that the result will still be the same as a 3-input NAND gate. This needs some thought. You will need an additional logic gate to combine two of the inputs that need to go in at . Remember that a NAND gate is the same as combining multiple inputs with AND, then inverting the result.

The completed flip-flop has two separate inputs, the clock signal and the data to store. This is a positive edge-triggered flip flop and will store the value of the data when the clock changes from zero to one. Test the flip flop by changing the clock signal manually to see when the change to Q happens.

Add some images to this worksheet.

1. Create a Logisim Evolution circuit that uses the built in D Flip-Flop from the memory tools, keep the trigger property as “Rising Edge” to match the flip-flop you created. Use a 1-bit data input and a clock signal. Test the circuit and compare to your flip-flop built above.

## Extension exercises

### Registers

1. Create a 4-bit register from four flip-flops with a common clock and enable signal. Have 4-bit input and output pins so that you can set the value in the register and read the output from it. You will need splitters and to be careful about the order of the bits in the signal.

Use the parallel in/parallel out register from Coates (2020b) as was shown in the lecture slides. Use Logisim Evolution’s built in D Flip-Flops (remember that a flip-flop can store one bit). Each bit from the input should go to its own flip-flop. There should be a clock signal that is connected to **all** flip-flops and a 1-bit input pin that is connected to the enable input on all four flip-flops. Test your register by using the hand icon, remember that the register will change its value on the rising edge of the clock signal, that is, when the clock changes from zero (dark green) to one (bright green).

Read about other ways to connect flip-flops to create registers, e.g., from Coates (2020b).

### Counter

Counting is an essential aspect of computing, so now we will create a binary counter.

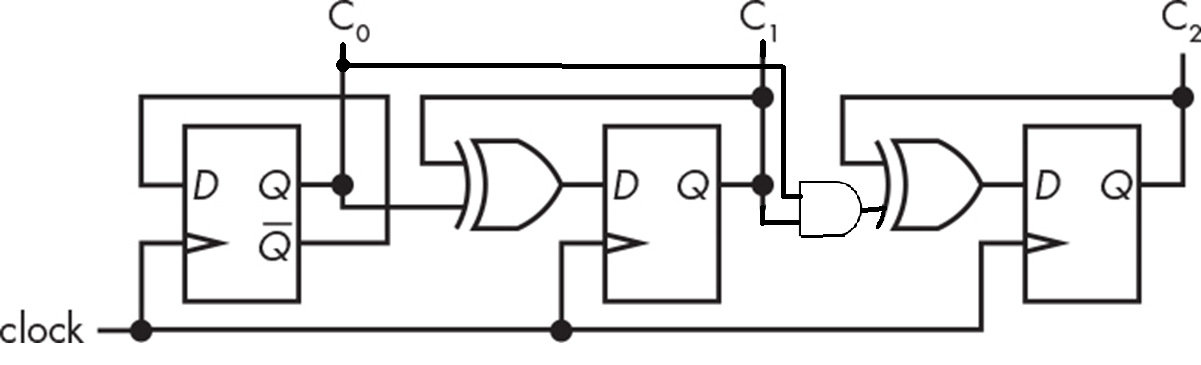


Figure 3-16 adapted from Steinhart (2019) showing a 3-bit synchronous counter.

1. Create a counter based on the figure above adapted from Steinhart (2019). Use the Logisim Evolution built-in D type flip-flop as the sub-circuit. Note the order of the outputs as C0 is the least significant digit (which you would normally expect to be on the right) also these are output bits but are shown at the top that doesn’t match the normal conventions of having output below or to the right. Test your counter.

This was a synchronous counter, that is all the digits update as appropriate on each cycle of the clock. You should make sure that you know the difference between synchronous and asynchronous counters, for example from Coates (2020a) or Steinhart (2019).

1. The built-in flip-flops in Logisim Evolution have an input labelled “R” that is for asynchronous reset, the tip that comes up says “Clear”. That input forces the flip-flop to be zero when the input at that point 1 regardless of the state of the clock.

Add a button to your counter circuit that, when pressed, sets all the digits of the counter to be zero.

1. As a more challenging task, build a feature so that you can specify a particular number that the counter starts from. The flip-flop components that you used to construct your counter have an input labelled “S” with a tip that comes up saying “Preset”. Add additional input to your circuit and use that preset input on the flip-flops to set an initial value of your choice. Think about how to test your circuit.

## References

Coates, E. (2020a) *Digital Counters* [Online] [Accessed on 10th September 2022] <https://learnabout-electronics.org/Digital/dig56.php>

Coates, E. (2020b) *Registers* [Online] [Accessed on 10th September 2022] <https://learnabout-electronics.org/Digital/dig57.php>

Steinhart, J. E. (2019) *The Secret Life of Programs Understand Computers – Craft Better Code* San Francisco: No Starch Press, Inc