# Week 3 Lab Memory

## Objectives

Develop understanding and experience of:

1. Latches and Flip-flops to store one bit of data
2. Storing multiple bits of data – Registers
3. Storing multiple bits of data – Addressable Memory

## Part 1 Latches and Flip-flops

The tutor will talk through some concepts relating to sequential circuits, latches, clock signals and flip-flops. Remember to work on the reinforcement tasks if you have time during the lab session.

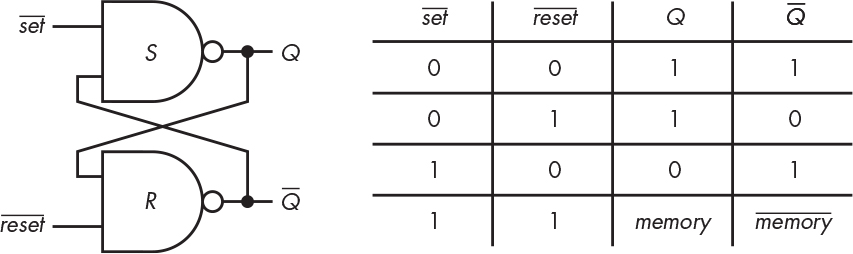


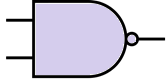
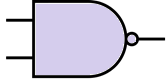
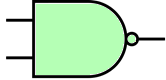
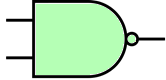
Figure 3.5 from Steinhart (2019)

1. Create a latch as shown in the figure. Make sure that the inputs and outputs are labelled. Note that the lines above indicate that they are the inverted set and reset signals. You might want to label them, inv Set and inv Reset (or Set bar and Reset bar) to indicate that they are inverted. Name the circuit latch. You may want to test the latch as it is.
2. Create a new circuit to test the latch within the same overall Logisim file.

To add a new circuit, right-click on the first folder icon on the left in Logisim and choose add circuit. Give it a suitable name. If you click on your latch whilst in your new (empty) circuit, you will be able to add it in the same way as adding other components.

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. 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.

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.



*Q*

*Q*

Data

Clock



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

1. Use the latch you created in task a as a sub-circuit to create the flip-flop shown, do **not** construct it from scratch from NAND gates. The figure above taken from Steinhart (2019) shows the three separate latches in different colours.

When using our existing latch, we have to cater for one of the NAND gates having 3 inputs, the input labelled in red 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. 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 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.

1. Create a Logisim circuit that uses the built in D Flip-Flop from the memory tools. Use a 1-bit data input, a clock signal and a 1-bit enable signal to determine whether the value stored should update. Test the circuit; keep the enable signal set to 1 and compare to your flip-flop built above.

The tutor will review the work on latches and flip-flops.

## Part 2 Registers

The tutor will introduce concepts of registers, memory and addressing. The exercises here and in part 3 follow the memory levels from nandgame (Kjær, no date). We will now use the Logisim built-in flip-flop component.

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 Logisim’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 when the clock changes from zero (dark green) to one (bright green).
2. Add a Logisim built-in register and another 4-bit output pin for comparison. Set the register to have data bits of 4 and use all the same inputs as your register constructed in the task above, that will include the clock signal. Test your circuit so that you can verify that your register constructed from flip-flops works in the same way as the register built in to Logisim.

The tutor will review the work on registers.

## Part 3 Addressable memory

In this part we are continuing to look at registers and memory, but now we will use the Logisim built-in memory components.

1. Create addressable ROM that is made of two registers and 1-bit address as follows. It should be possible to read from the ROM.
2. Create a new version of the circuit to have four registers and a 2-bit address. Use an input pin with data-bits set to 2 for the address. Think about how to test the circuit systematically to check it is working correctly.
3. Compare your ROM to the ROM component built-in to Logisim as follows.

Create a new circuit in Logisim and add a ROM component from the memory tools. To match the exercise above, set the Address Bit Width to 2 and the Data Bit Width to 4. Connect a 2-bit input pin to the Address (A) input on the ROM and a 4-bit output pin to the Data (D) output.

Test the circuit as follows. When you are in simulation mode (with the hand highlighted) you can click on data at individual addresses to change its value (as you could with the registers). Logisim ROM (and RAM) components allow additional ways to edit the contents as there would normally be more data than this. Right-click on the ROM in simulation mode and you will see options to edit contents, load image and save image (in this context the word image means a file containing hex digits). Select Edit Contents to bring up the Logisim hex editor and key in four hex digits. Click save and save it in the same place as you are using for the Logisim exercises from this lab. Give it a file name of your own choice (I use the file name extension of .dat for data). Open the file you have saved in a text editor (notepad++ or VS Code for example) and you will see a header line and the data you typed. This could be edited here and loaded into the ROM using Load Image. Make sure that you can load data using that method.

The tutor will review the work on registers and memory.

## References

Kjær, O. J. (no date) *NandGame – Build a computer from scratch* [Online] [Accessed on 10th September 2021] <https://nandgame.com>

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