# Week 5 Lab B: Registers

## Objectives

Develop understanding and experience of:

1. Using a register to store a value.

## Registers

* What is a clock signal?

It alternates 0 and 1.

* Why do some components need a clock signal?

To synchronise the components.

## Use registers.

1. Create the 8-bit counter as described in the week 4 lecture (and shown below).

You can set your counter to tick automatically. Use the simulate menu and “Auto Tick Enabled” or Ctrl+K to switch ticks on and off. The Auto Tick Frequency is the number of ticks per second.

A screenshot of a computer

Description automatically generated

1. Open your traffic light simulation from last week, if you completed it. Add a register and addition so that it will automatically cycle through the stages for the traffic lights.

## Extension exercises

These exercises are a little more challenging but show that we can now build complex components with what we have covered.

## Fibonacci Sequence

The Fibonacci Sequence is formed by starting with 0 and 1 and forming each following term by adding the previous two.

The first few terms are as follows:

0, 1, 1, 2, 3, 5, 8, 13, 21

Create a circuit that calculates the Fibonacci sequence. Use two registers and an addition component as well as an output pin. You will need to copy the total at each step to one register and copy the previous total to the other register. Decide how many bits you want to cater for and check the output against an online source. You will have to manually set one of the registers to store 1 to start your simulation.

### Create a clock

For this exercise you should create a clock that shows the time on a digital display. This task is fairly challenging so you should build up in stages. The output would display the time in hours, minutes and seconds for example 10.43.17.

In the Input/Output tools in Logisim Evolution you will find a hex digit display. Use registers that hold 4 bits so that you can feed them directly to a hex digit display. For this exercise you will be using binary coded decimal (BCD) as you will not allow the digits to get to a, b etc and carry to the next register as in decimal. You might want to research binary coded decimal as well.

1. Create a counter using a 4-bit register. Add gates and wiring so that the register resets to zero as soon as it gets to hex a (decimal 10).
2. Add a second register (to the left) that only adds 1 when the first register is reset to zero. You can control this using the clock input to the register. This would create a counter that displays values from 0 to 99. Although you are using a hex digit display, we are interpreting the digits in decimal.
3. To make the second register be part of the seconds, it needs to reset when it gets to 6 and the minutes needs to go up.
4. The minutes will be built in a similar way to the seconds.
5. You could choose to make a 12-hour or 24-hour clock.

Make sure that all of your changes to add to hours, minutes and seconds work correctly so that you don’t gain or lose a second each time. You will also want to simulate it with a high frequency of ticks.