Faculty of Science, Engineering and Technology

Computer Systems

Overview

This laboratory session will be focused on building a full adder of multiple bits, and working with Flip-Flops. You should spend time reviewing the week's lecture, and watching the tutorial videos linked below.

Purpose:	Building a 4 bit adder and working with Flips Flops.
Task:	
Time:	This lab is due by the start of your week 3 lab.
Assessment:	This lab is worth 1% of your assessment for this unit, and only if demonstrated to your lab demonstrator in the week it is due.

Resources:	▪ Swin tutorials
	  ▪ Full-adder tutorial
	  ▪ Intro to Flip Flops

Submission Details

You must submit the following files to Canvas:

- A document containing all required work as described below.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Instructions

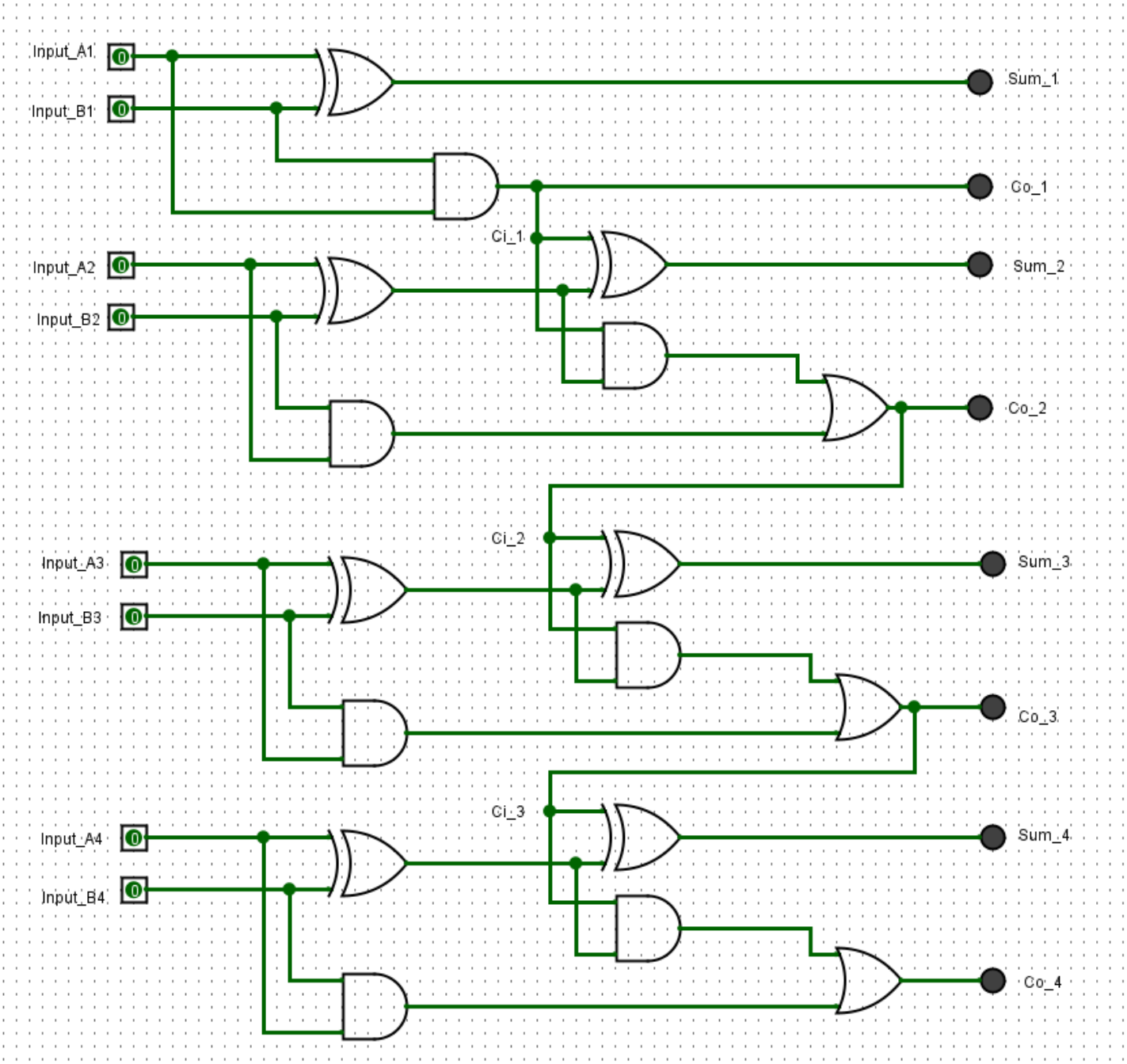
1. Start Logisim and open a new canvas

2. Today you're going to build a 4 bit adder. Before you start, plan it out. Think about the parts you need. Review this week's lecture slides on full-adders, and watch the video linked under resources. Discuss it with your demonstrator if you need to.

HINT: for this task you might want to re-use your half-adder and full-adder from last week's lab.

3. A 4-bit adder sums together two 4-bit binary numbers. Each bit of each number is represented by a binary on/off pin. So, you will need two sets of four input pins.
   3.1. Layout the pins you need for each input bit.
   3.2. While doing this, workout the order of your bits (*Big-endian or Little-endian?).* Use labels to indicate the significance of each bit (i.e., which column, from least significant (2^0 column) to most significant (2^3 column).
4. Now start implementing your adder. Use a half-adder to add the first bits from both binary numbers, and wire up an LED to represent the output bit. We did this last week so it should now be straight forward!
5. Now wire-up a full-adders to add all remaining bits. Remember that a full-adder also adds the carry-in bit from the previous adder. Use an LED to show the sum output for each column.

Once complete, check its correctness by testing and filling out a truth table like the following (over page). Add the circuit screen shot and the table to your submission document:


Input_A1
Input_B1
Sum_1
Co_1
Ci_1
Input_A2
Input_B2
Sum_2
Co_2
Ci_2
Input_A3
Input_B3
Sum_3
Co_3
Ci_3
Input_A4
Input_B4
Sum_4
Co_4

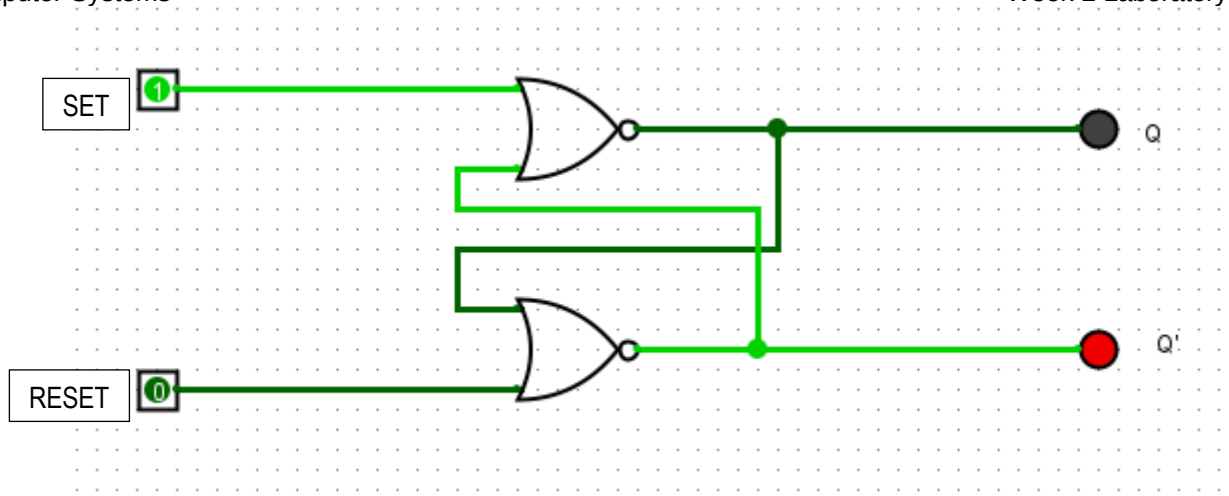
Input A	Input B	Output
0101	**0000**	0101
0101	**0001**	0110
0101	**0010**	0111
0101	**0011**	1000
0101	**0100**	1001
0101	**0101**	1010
0101	**0110**	1011
0101	**0111**	1100
0101	**1000**	1101
0101	**1001**	1110
0101	**1010**	1111
0101	**1011**	0000
0101	**1100**	0001
0101	**1101**	0010
0101	**1110**	0011
0101	**1111**	0100

Part 2: Storing bits with Flip Flops

Any computing hardware that seeks to perform meaningful calculations using bits requires circuitry to store them - that is, circuits that can maintain a given state. In lectures we discussed Flip Flops, which are simple block circuits designed to maintain a particular binary state, and transition between binary states depending on the inputs given.

6. Review this week's lecture slides, and if needed, also take a look at the quick video tutorials linked under resources at the beginning of this lab sheet.
7. Create a clear canvas.
8. Using the lecture slides as a guide, wire up your own R-S Flip Flop using a pair of 2-input NOR gates (**do not use Logisim's S-R Flip-Flop!**). You should have 2 input pins, one for the "*Set*" pin, and one for the "*Reset*", and two output LEDs: *Q* and *Q'* .
9. When you've finished wiring it up, set both input pins to 1. The LEDs should both be dark (assuming you've wired it correctly).

Export your circuit as an image and include it in your submission document.



10. Set the pins *in the following order* and record the states for Q and Q'

Set	Reset	Q	Q'
1	**0**	0	1
1	1	0	0
0	1	1	0
1	1	0	0

11. Describe in a sentence, the behaviour of the circuit when one of the inputs is 1 (but not both) and why this is useful for digital circuit design.

    When either one of the inputs is 1 but not both , the state of outputs would also be 1: if *set* = 1 then *Q'* = 1 and vice versa or when *reset* = 1 then *Q* = 1.

    This is useful for digital circuit design as RS Flip Flops provide a method to hold and store information as a 1-bit memory.

12. What do you notice about the two times you set both inputs to 1. Briefly explain what is happening here and why this is an issue for digital circuit design?

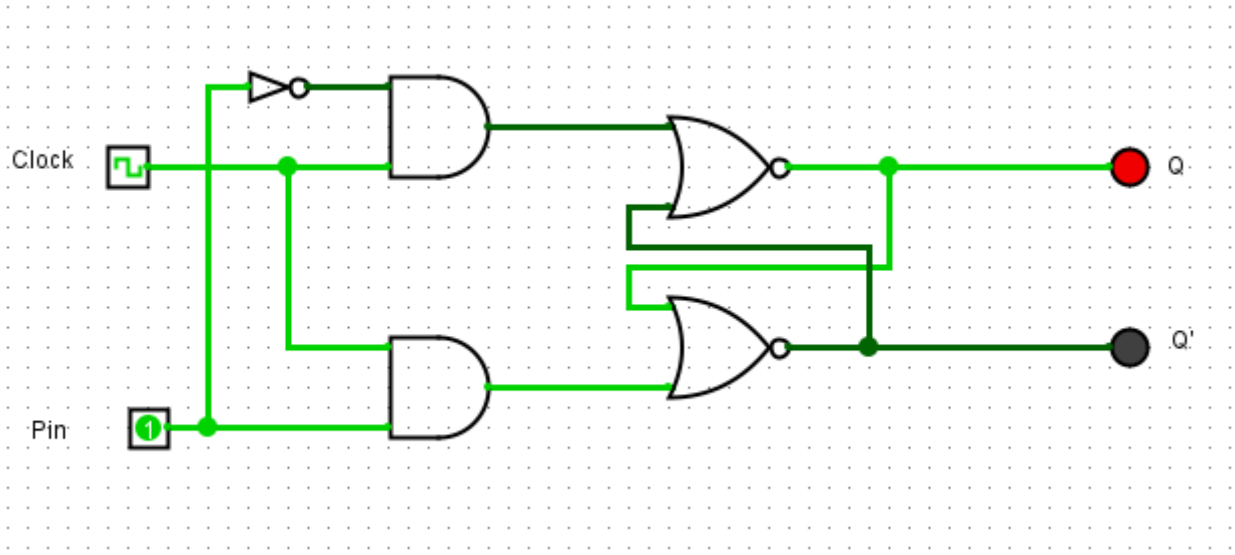
    When both inputs are active it would result in circuit being indeterminant in a NOR-Based RS Flip Flop. This is dangerous and highly undesirable in circuit design.

Discuss 12 and 13 with your lab demonstrator and provide your answer in your submission document, along with the truth table in Step 11.

13. So the unclocked R-S flip flop has issues. Lets talk about the D Flip-Flop then. Review the lecture slides on the D Flip-Flop, and when you feel comfortable, wire up a D Flip Flop using AND gates and NOR gates, with output LEDS labeled Q and Q'.

    - For this you will have only 1 input pin, as well as a clock input. The clock can be pulsed on and off by clicking it with the operation pointer (the finger in the top left of screen), or you can simple enable clock ticking from the menu (under "Simulate").

Export your circuit as an image and include it in your submission document.



14. Explore the behaviour of the D Flip Flop by filling out the following truth table

Clock	Pin	Q	Q'
0	**0**	0	0
0	**1**	0	0
1	**1**	1	0
1	**0**	0	1

15. Briefly explain the behaviour of a D Flip Flop and how it is useful for digital circuit design.

    D Flip Flop has 1 input and whenever the clock is active, the output of Q would change the same as the input pin, while Q' would be opposite. This is extremely useful for digital circuit design as it will store the logic level of the D input, it also acts as both a Reset and Set input and will never be active at the same time.

16. What is the role of the clock? How does it impact the changing of state of Q and Q' ?

    The role of the clock is to synchronize the signal of the input. It impacts the input data, which in turn changes the state of the outputs.

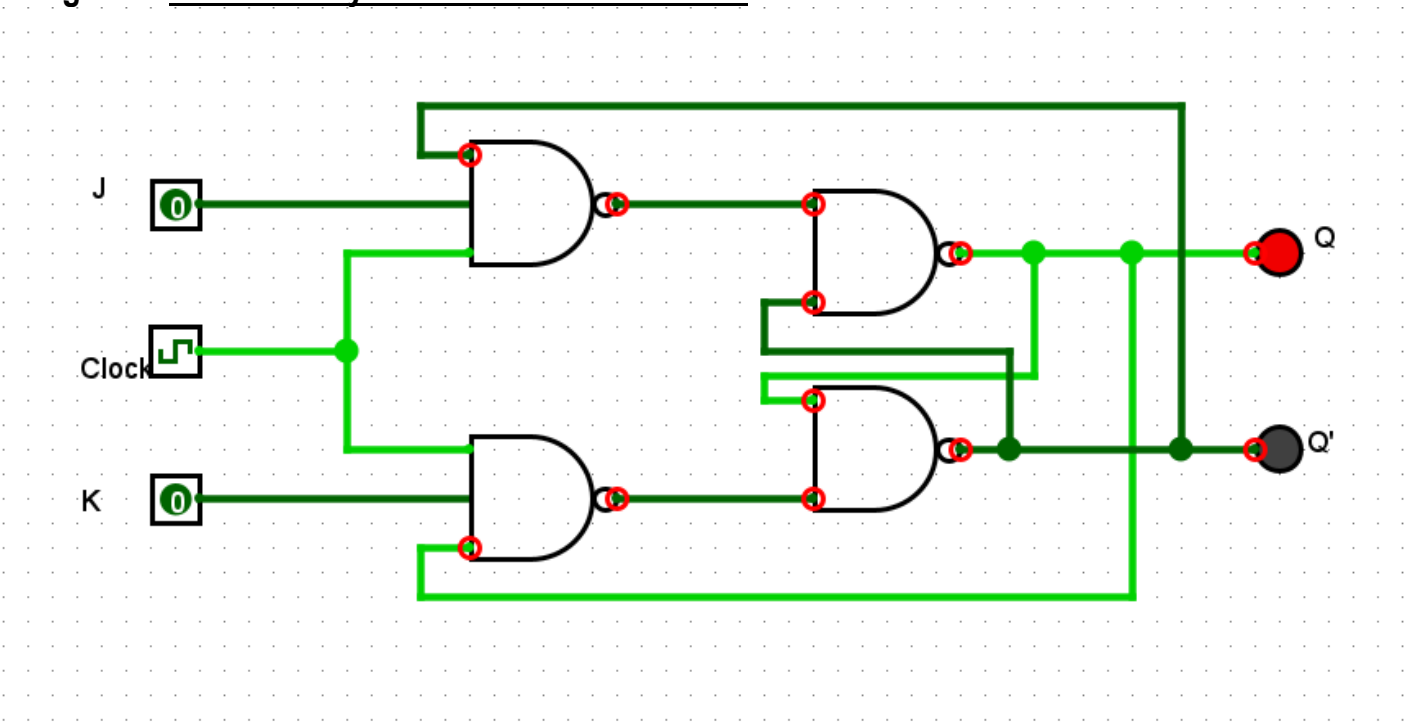
17. Why is it generally preferred over the R-S Flip Flop ?

D Flip Flops are preferred over R-S Flip Flops because it eliminates the indeterminant state as D Flip Flop implements an inverter that guarantees the inputs will never both be the same.

Discuss 15 -17 with your lab demonstrator and provide your answer in your submission document, along with the truth table above.

18. J-K Flip Flops are like your general purpose Flip Flop because they are programmable. Review the slides on JK Flip Flops, and when you're feeling comfortable, wire up a J-K FF using NAND gates (remember we did this in the lecture!). Two of your NAND gates will need to deal with three inputs.

Logisim will not be able to simulate this circuit, but export your completed circuit as an image and include it in your submission document.



19. **Complete and include this truth table for JK Flip Flops in your submission document.**

J	K	Q (when clocked)	Q' (when clocked)
0	0	No change	No change
1	0	1	0
0	1	0	1
1	1	Toggle	Toggle

20. How can a J-K Flip Flop be made to behave like a D Flip Flop?

    To make a J-K Flip Flop behave like a D Flip Flop, we need to drive the *J* and *K* input pins with the *D* input, as well as a NOT gate.

21. How can a J-K Flip Flop be made to behave like a toggle (T Flip Flop)?

    A J-K Flip Flop is made to behave like a toggle when both inputs *J* and *K* are 1.

Discuss these questions with your lab demonstrator and provide your answer in your submission document, along with the truth table in Step 20.

When complete:

- Submit your answers (screen shots, etc) in a single document using **Canvas**
- Show your lab demonstrator your working circuits in class (you must do this to get the 1%). Your lab demonstrator may request you to resubmit if issues exist.