2

Digital logic design

William W.-Y. Hsu
Department of Computer Science and Engineering
Department of Environmental Biology and Fisheries Science
National Taiwan Ocean University

CONTENTS

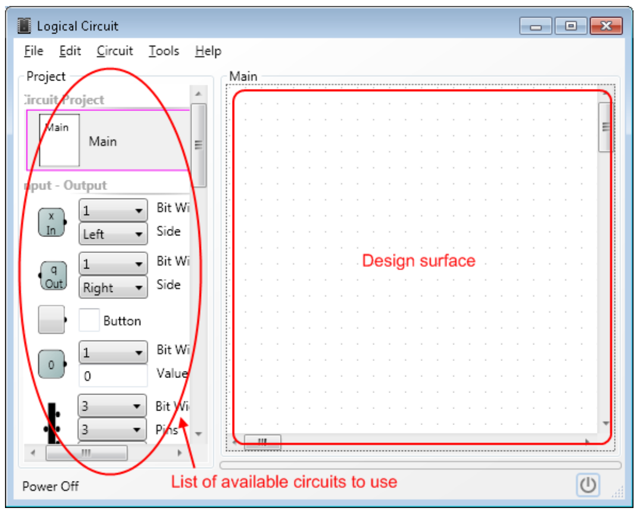
2.1 Introduction

LogicCircuit is free, open source educational software for designing and simulating digital logic circuits. Intuitive graphical user interface, allows you to create unrestricted circuit hierarchy with multi bit buses, debug circuits behavior with oscilloscope, and navigate running circuits hierarchy. It can be downloaded from `http://www.logiccircuit.org/download.html`.

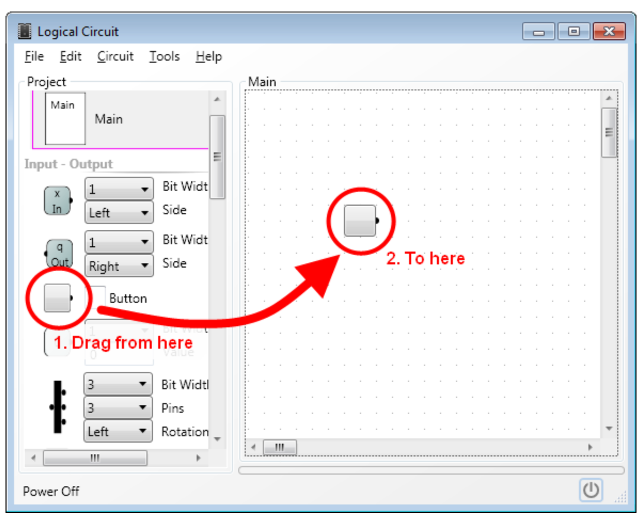
2.2 Lab Procedures

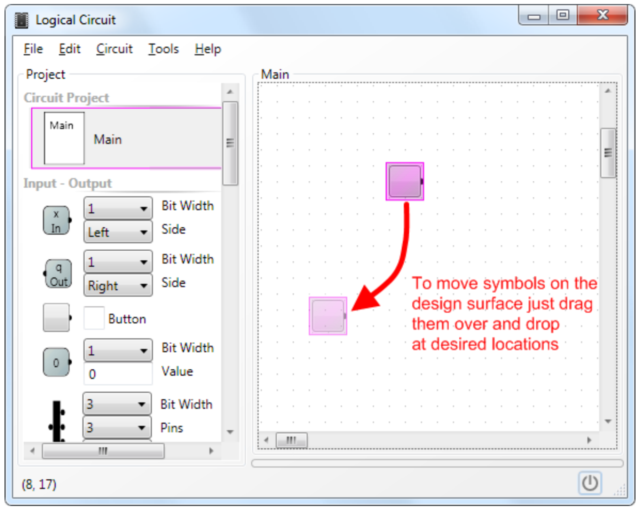
2.2.1 Editing circuits

When you first open LogicCircuit it will be one empty logical circuit visible on the right pane of the program window named “Main”.

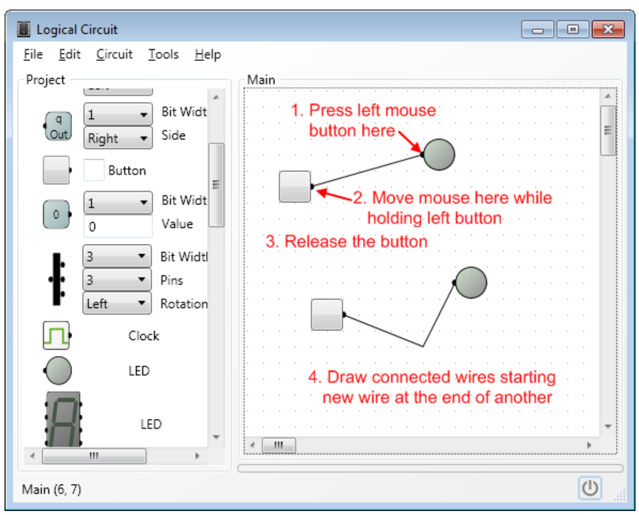


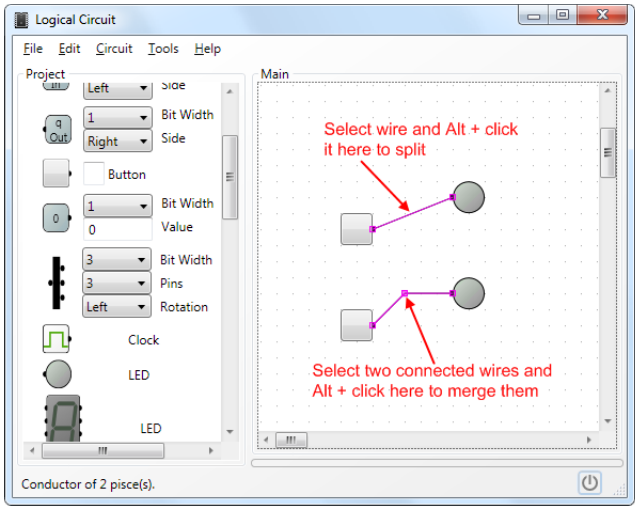
To create you first circuit just drag and drop any item (for example a button) from the left pane to the design surface.



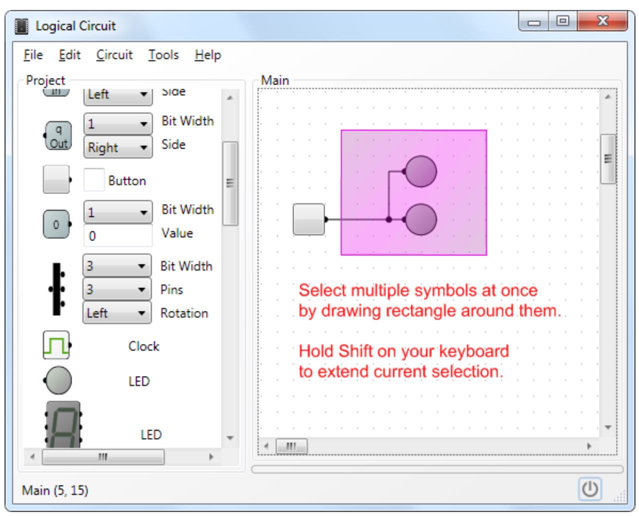


To wire your circuits connect output pin of one circuit with one or more input pins of another one. All pins are bold black dots on the edges of circuit symbols. All you need to do is just draw wires with your mouse.



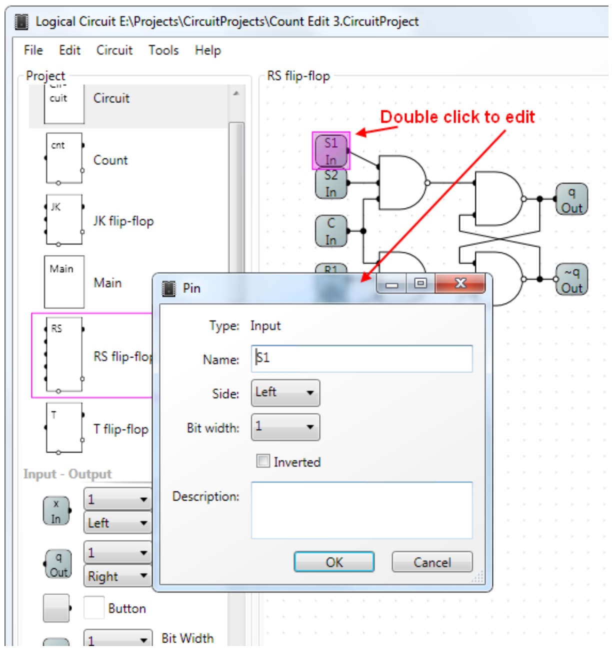


You can select and move multiple symbols on your circuit. Hold `Ctrl` key on your keyboard and click items on the design surface to select or unselect them. Click wire while holding `Shift` key to select entire "conductor". There are bunch of selection commands in the `Edit` menu.



When you moving symbols over the diagram wires connected to them will be sticking to the pins. If you want to move the symbols without wires hold Shift while dropping at the desired location.

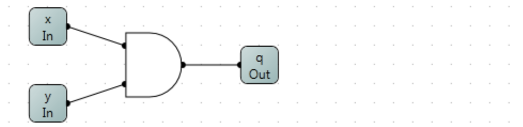
To edit properties of the symbol on the design surface double click it and the property dialog will pop up. However double clicking symbol of logical circuit will open it on the design surface.

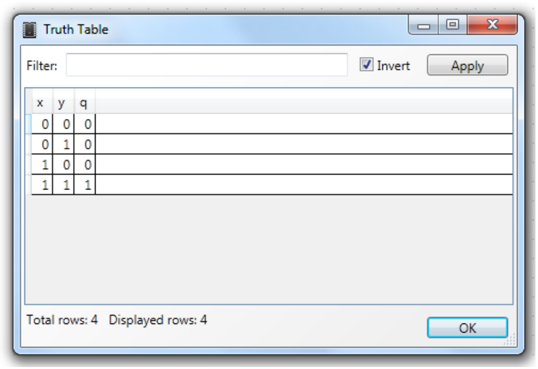


Now you have your first circuit ready and can try to power it up and see how it works.

2.2.2 Truth table

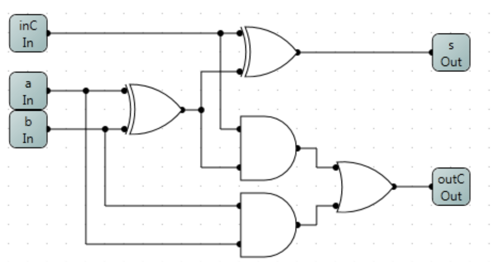
You can build truth table of your sub circuit if it has input and output pins and does not contain any states holding elements like flip-flops and RAM. In other words you circuit is a pure function. To build truth table click menu `Circuit/Truth Table`.

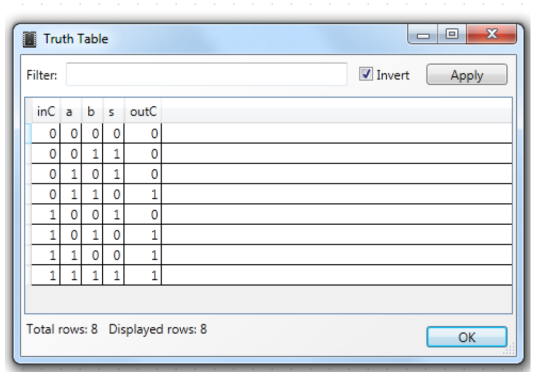




x	y	q
0	0	0
0	1	0
1	0	0
1	1	1

In the popped up dialog you'll see a table which columns are named like input and output pins. All the rows of the table are all the permutations of input and result of the outputs. Here is another example of truth table of *full adder*.





inC	a	b	s	outC
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

So as you can imagine number of rows should be equal to two in power of sum of number of bits on all inputs. For example if you circuit like the one above has 3 input pins of one bit each then $2^3 = 8$ the truth table will be 8 rows long. Therefore it is fairly easy to build very big truth table by just increasing bit width of input pins. The dialog however will not show more than 4096 rows because it is unmanageable number of rows. But you can use filter to ensure correctness of you circuit.

Filter is just an expression that applied to each row in the truth table to determine if the row needs to be included in the resulting truth table. In these expressions you can use names of input and output pins and check if their values are satisfy some conditions. For example for the first truth table which is just AND gate the filter expression can be: $q = x\&y$. Please note the invert check box right next to Filter text box in the top of the dialog. If this box is checked then the result of the expression is inverted and hence the above expression been inverted will result in empty table. This is indicating correct behavior of the circuit. If you uncheck it then all the rows will be visible in the table. Please note that there is two numbers in the bottom of the dialog showing total number of rows in the truth table and number displayed in the dialog

2.3 Lab Questions

1. (20, In course demonstration) Construct and simulate the circuit in Figure 2.1 by tabulating the truth table in Section 2.2.2.
2. (30, In course demonstration) Construct a *J-K flip flop* using the software (Figure 2.2).
3. (30) Implement a 1-bit *full adder* using the software, and display the truth table (Figure 2.3).
4. (20) Try to combine 2 1-bit *full adder*s creating a 2-bit full adder.
5. (20) $\star$ Combine the 2 2-bit *full adder* into a 4-bit full

adder, creating a adder of this type is called a ripple carry adder.

6. Reading: Ripple carry adders takes more clock cycles to compute. Try looking up for the *carry lookahead adder*s.

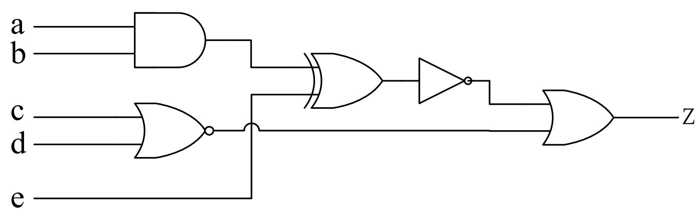


FIGURE 2.1
A digital logic circuit. This circuit has 5 inputs and 1 output.

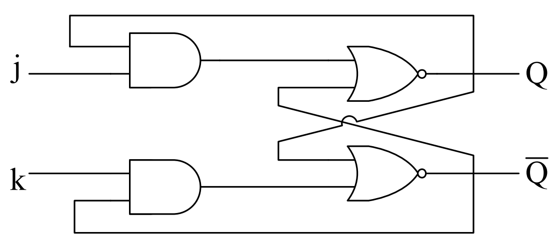


FIGURE 2.2
A J-K flip flop. This is an implementation of a J-K flip flop without clock signals.

2.4 Lab Report

Your lab report is due in class or no later than 1 week after the lab.

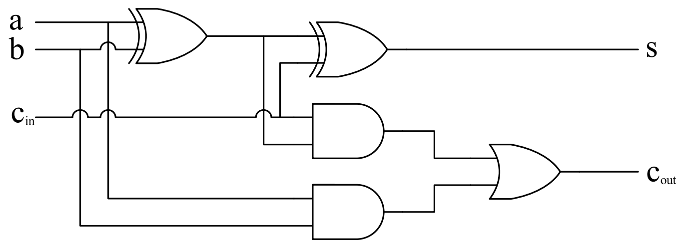


FIGURE 2.3
Full adder circuit. This is a 1-bit full adder.