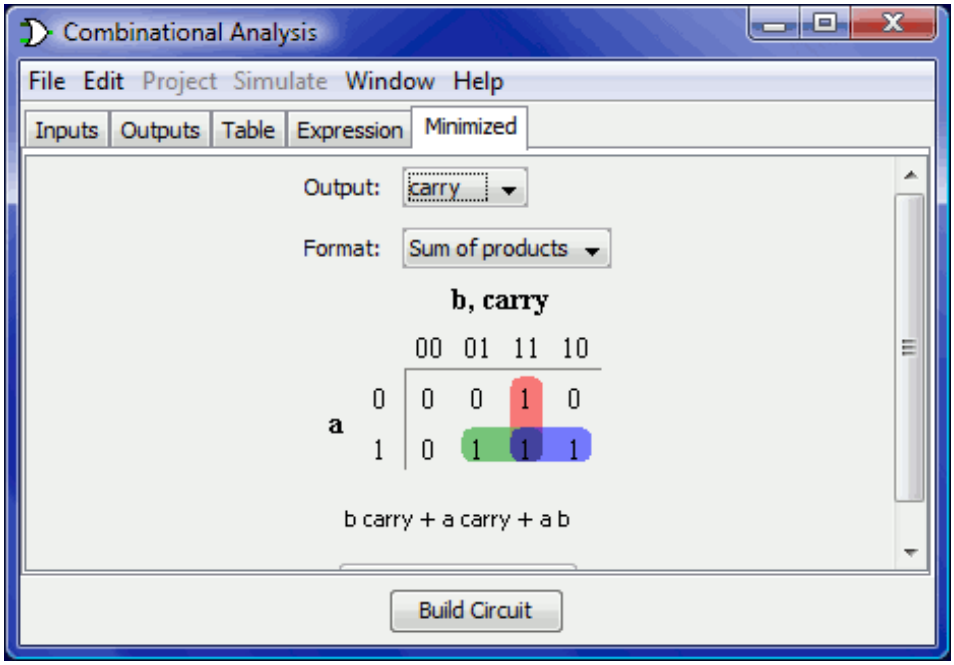
Combinational analysis



All circuits fall into one of two well-known categories: In a **combinational circuit**, all circuit outputs are a strict *combination* of the current circuit inputs, whereas in a **sequential circuit**, some outputs may depend on past inputs (the *sequence* of inputs over time).

The category of combinational circuits is the simpler of the two. Practitioners use three major techniques for summarizing the behavior of such circuits.

- logic circuits
- Boolean expressions, which allow an algebraic representation of how the circuit works
- truth tables, which list all possible input combinations and the corresponding outputs

The *Combinational Analysis* module of Logisim allows you to convert between these three representations in all directions. It is a particularly handy way of creating and understanding circuits with a handful of one-bit inputs and outputs.

Next: Opening Combinational Analysis.

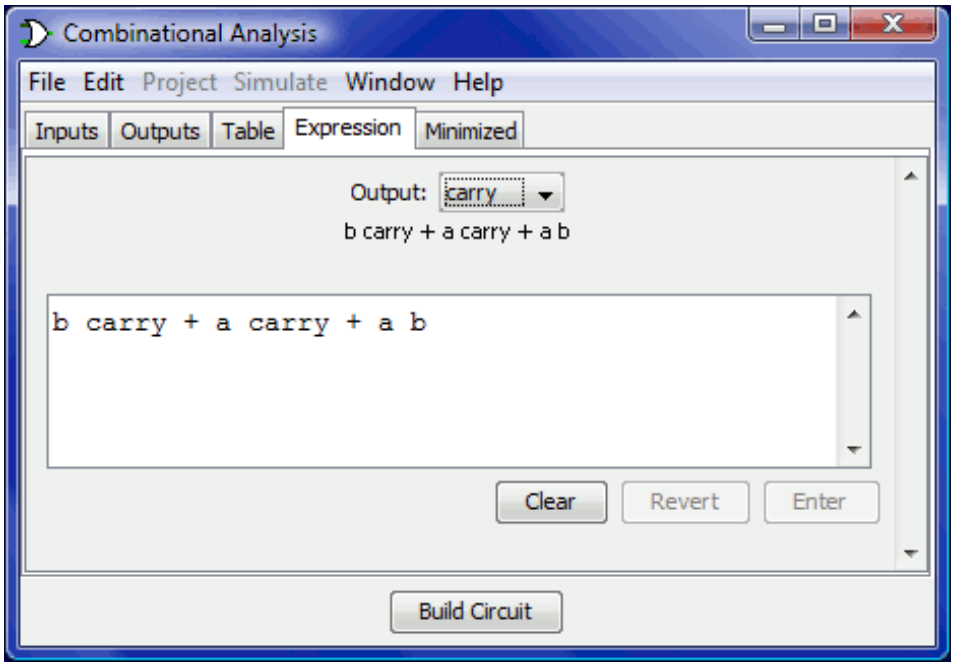
Creating expressions

For each output variable, the Combinational Analysis window maintains two structures - the relevant column of the truth table, and a Boolean expression - specifying how each output relates to its input. You can edit either the truth table or the expression; the other will automatically change as necessary to keep them consistent.

As we will see on the next page, the Boolean expressions are particularly useful because the Combinational Analysis window will use these when told to build a circuit corresponding to the current state.

You can view and edit the expressions using the window's last two tabs, the Expression tab and the Minimized tab.

The Expression tab



The Expression tab allows you to view and edit the current expression associated with each output variable. You can select the output expression you want to view and edit using the selector labeled "Output:" at the pane's top.

Just below the selector will appear the expression formatted in a particularly common notation, where an OR is represented as addition, an AND is represented as multiplication, and a NOT is denoted with a bar above the portion affected by the NOT.

The text pane below this displays the same information in ASCII form. Here, a NOT is represented with a tilde ('~').

You can edit the expression in the text pane and click the Enter button to make it take effect; doing this will also update the truth table to

make it correspond. The Clear button clears the text pane, and the Revert button changes the pane back to representing the current expression.

Note that your edited expression will be lost if you edit the truth table.

In addition to multiplication and addition standing for AND and OR, an expression you type may contain any of C/Java logical operators, as well as simply the words themselves.

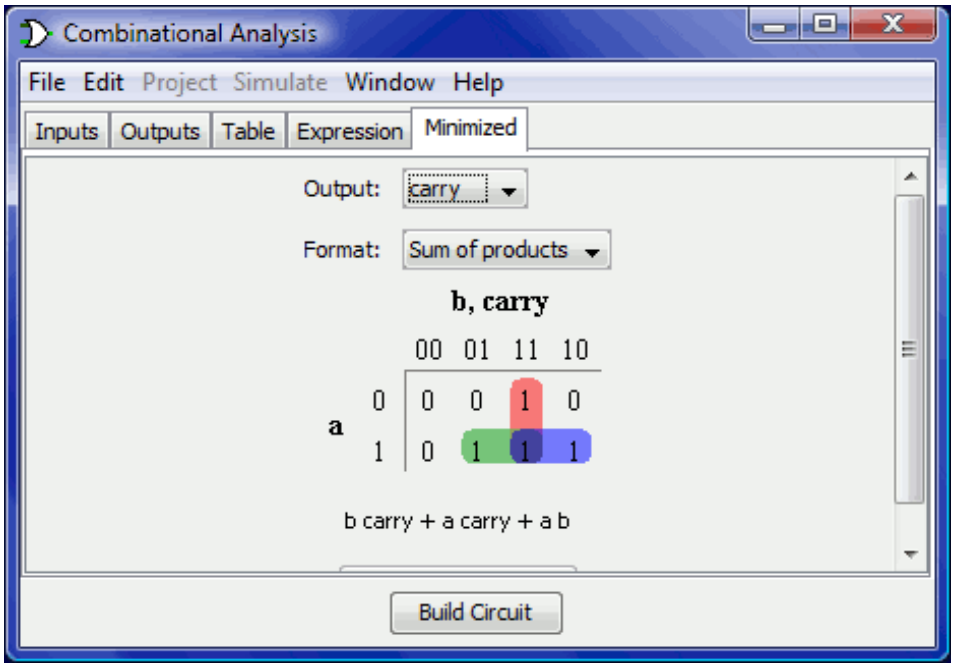
highest precedence	`~ ! '`	**NOT**
	(none) `& &&`	**AND**
	`^`	**XOR**
lowest precedence	`+ | ||`	**OR**

The following examples are all valid representations of the same expression. You could also mix the operators.

```
a' (b + c)
!a && (b || c)
NOT a AND (b OR c)
```

In general, parentheses within a sequence of ANDs (or ORs or XORs) do not matter. (In particular, when Logisim creates a corresponding circuit, it will ignore such parentheses.)

The Minimized tab



The final tab displays a minimized expression corresponding to a column of the truth table. You can select which output's minimized expression you want to view using the selector at top, and you can indicate whether you want to derive a sum-of-products expression or a product-of-sums expression using the selector below.

If there are four or fewer inputs, a Karnaugh map corresponding to the variable will appear below the selector. You can click the Karnaugh map to change the corresponding truth table values. The Karnaugh map will also display the currently selected terms for the minimized expression as solid semitransparent rounded rectangles.

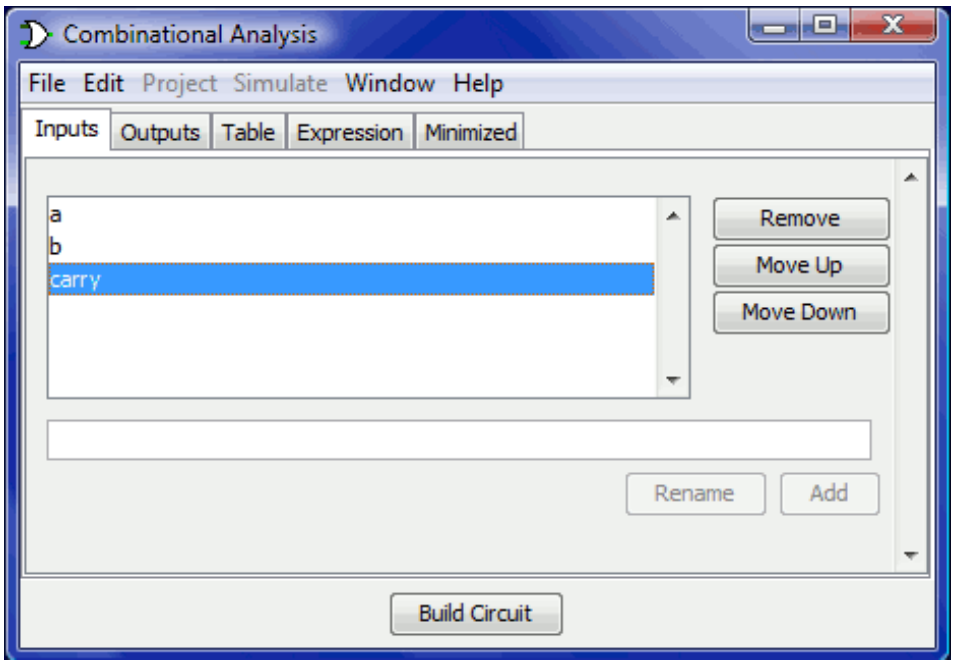
Below this is the minimized expression itself, formatted as in the Expression tab's display. If there are more than four inputs, the Karnaugh map will not appear; but the minimized expression will still be computed. (Logisim uses the Quine-McCluskey algorithm to compute the minimized expression. This is equivalent to a Karnaugh map, but it applies to any number of input variables.)

The Set As Expression button allows you to select the minimized expression as the expression corresponding to the variable. This will generally not be necessary, as edits to the truth table result in using the minimized expression for the changed column; but if you enter an expression through the Expression tab, this can be a convenient way to switch to the corresponding minimized expression.

Next: Generating a circuit.

Editing the truth table

On opening the Combinational Analysis window, you will see that it consists of five tabs.



This page describes the first three tabs, Inputs, Outputs, and Table. The next page of the guide describes the last two tabs, Expression and Minimized.

The Inputs and Outputs tabs

The Inputs tab allows you to view and edit the list of inputs. To add new inputs, type it in the field at the pane's bottom, and click Add. If you want to rename an existing input, select it in the list in the pane's upper left region; then type the name and click Rename.

To remove an input, select it from the list and click Remove. You can also reorder the inputs (which affects the order of columns in the truth table and in the generated circuit) using the Move Up or Move Down buttons on an input.

All actions affect the truth table immediately.

The Outputs tab works in exactly the same way as the Inputs tab, except of course it works with the list of outputs instead.

The Table tab

The only item under the Table tab is the current truth table, diagrammed in the conventional order, with inputs constituting the columns on the left and outputs constituting the columns on the right.

You can edit the current values appearing in the output columns by

clicking on the value of interest. The values will cycle through 0, 1, and *x* (representing a "don't care"). As we'll see on the next page, any don't-care values allow the computation of minimized expressions some flexibility.

You can also navigate and edit the truth table using the keyboard. And you can copy and paste values using the clipboard. The clipboard can be transferred to any application supporting tab-delimited text (such as a spreadsheet).

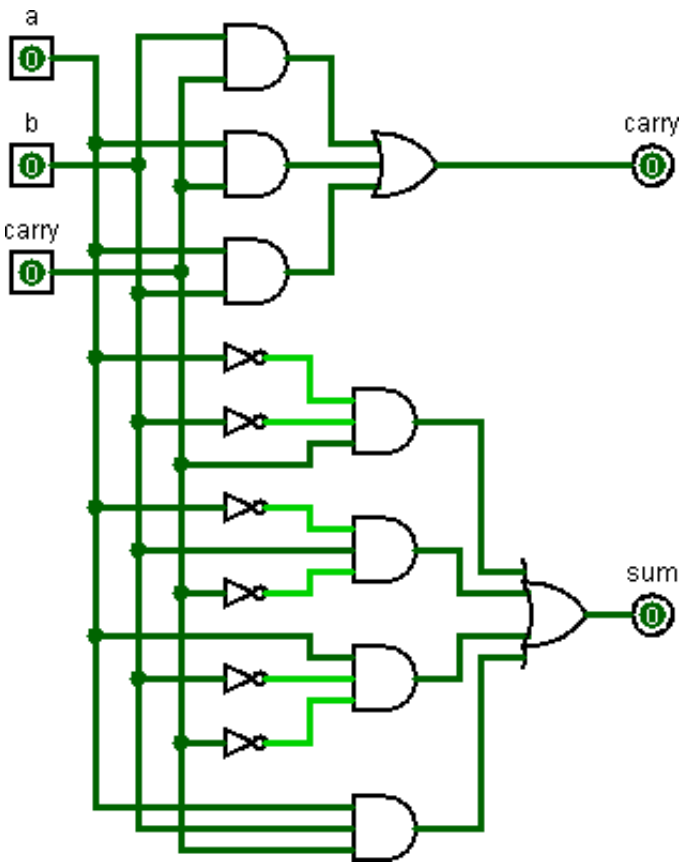
If the truth table is based on an existing circuit, you may see some pink squares in the output columns with "!!" in them. These correspond to errors that occurred while calculating the value for that row - either the circuit seemed to be oscillating, or the output value was an error value (which would be pictured as a red wire in the Logisim circuit). Hovering your mouse over the entry should bring up a tool tip describing which type of error it was. Once you click on the error entry, you will be in the 0-1-*x* cycle; there is no way to go back.

Next: Creating expressions.

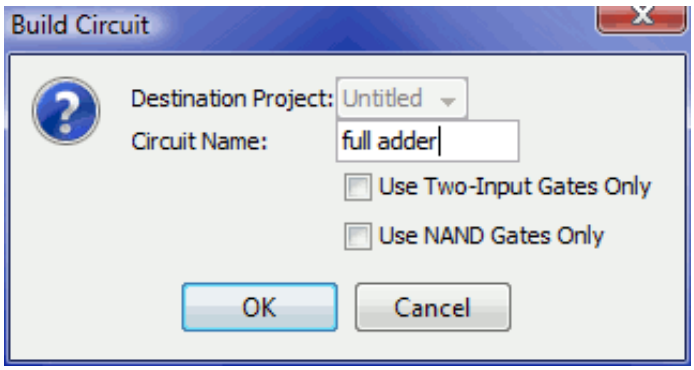
Logisim
a graphical tool for designing and simulating logic circuits

Generating a circuit

The Build Circuit button will construct a circuit whose gates correspond to the currently chosen expressions for each output. The circuit's inputs and outputs will be displayed in top-down order corresponding to how they appear under the Inputs and Outputs tabs. Generally speaking, the constructed circuit will be attractive; and, indeed, one application of Logisim's Combinational Analysis module is to beautify poorly drawn circuits. Still, as with any automatic formatting, it will not express the structural details that a human-drawn circuit would.



When you click the Build Circuit button, a dialog box will appear prompting you to choose which project where you want the circuit and the name you wish to give it.



If you type the name of an existing circuit, then that circuit will be replaced (after Logisim prompts you to confirm that you really want to do this).

The Build Circuit dialog includes two options. The Use Two-Input Gates Only option specifies that you want all gates constructed to have two inputs. (NOT gates, of course, constitute an exception to this rule.) The Use NAND Gates Only option specifies that you would like it to translate the circuit into one using only NAND gates. You can select both options if you want to use only two-input NAND gates.

Logisim cannot construct a NAND-only circuit for an expression containing any XOR operators. This option will therefore be disabled if any outputs' expressions contain XORs.

Next: *User's Guide*.

Opening Combinational Analysis

The bulk of the Combinational Analysis module is accessed through a single window of that name allowing you to view truth tables and Boolean expressions. This window can be opened in two ways.

Via the Window menu

Select Combinational Analysis, and the current Combinational Analysis window will appear. If you haven't viewed the window before, the opened window will represent no circuit at all.

Only one Combinational Analysis window exists within Logisim, no matter how many projects are open. There is no way to have two different analysis windows open at once.

Via the Project menu

From a window for editing circuits, you can also request that Logisim analyze the current circuit by selecting the Analyze Circuit option from the Project menu. Before Logisim opens the window, it will compute Boolean expressions and a truth table corresponding to the circuit and place them there for you to view.

For the analysis to be successful, each input must be attached to an input pin, and each output must be attached to an output pin. Logisim will only analyze circuits with at most eight of each type, and all should be single-bit pins. Otherwise, you will see an error message and the window will not open.

In constructing Boolean expressions corresponding to a circuit, Logisim will first attempt to construct a Boolean expressions corresponding exactly to the gates in the circuit. But if the circuit uses some non-gate components (such as a multiplexer), or if the circuit is more than 100 levels deep (unlikely), then it will pop up a dialog box telling you that deriving Boolean expressions was impossible, and Logisim will instead derive the expressions based on the truth table, which will be derived by quietly trying each combination of inputs and reading the resulting outputs.

After analyzing a circuit, there is no continuing relationship between the circuit and the Combinational Analysis window. That is, changes to the circuit will not be reflected in the window, nor will changes to the Boolean expressions and/or truth table in the window be reflected in the circuit. Of course, you are always

free to analyze a circuit again; and, as we will see later, you can replace the circuit with a circuit corresponding to what appears in the Combinational Analysis window.

Limitations

Logisim will not attempt to detect sequential circuits: If you tell it to analyze a sequential circuit, it will still create a truth table and corresponding Boolean expressions, although these will not accurately summarize the circuit behavior. (In fact, detecting sequential circuits is *provably impossible*, as it would amount to solving the Halting Problem. Of course, you might hope that Logisim would make at least some attempt - perhaps look for flip-flops or cycles in the wires - but it does not.) As a result, the Combinational Analysis system should not be used indiscriminately: Only use it when you are indeed sure that the circuit you are analyzing is indeed combinational!

Logisim will make a change to the original circuit that is perhaps unexpected: The Combinational Analysis system requires that each input and output have a unique name that conforming to the rules for Java identifiers. (Roughly, each character must either a letter or a digit, and the first character must be a letter. No spaces allowed!) It attempts to use the pins' existing labels, and to use a list of defaults if no label exists. If an existing label doesn't follow the Java-identifier rule, then Logisim will attempt to extract a valid name from the label if at all possible.

Incidentally, the ordering of the inputs in the truth table will match their top-down ordering in the original circuit, with ties being broken in left-right order. (The same applies to the ordering of outputs.)

Next: Editing the truth table.