



Guide to Being a Logisim User

Beginner's tutorial

Libraries and attributes

Subcircuits

Creating circuits

Using subcircuits

Editing subcircuit appearance

Debugging subcircuits

Logisim libraries

Wire bundles

Combinational analysis

Menu reference

Memory components

Logging

Command-line verification

Application preferences

Project options

Value propagation

JAR libraries

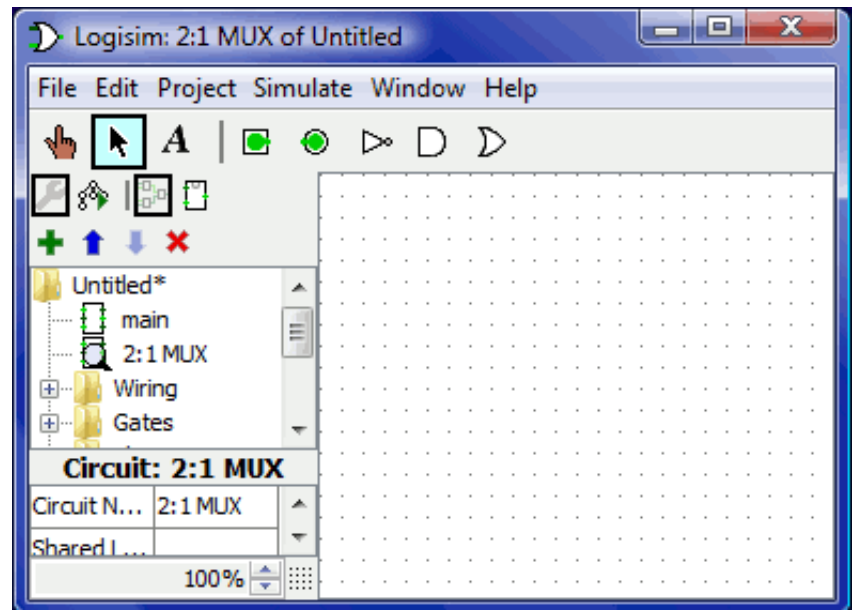
About the program

Library Reference

Creating circuits

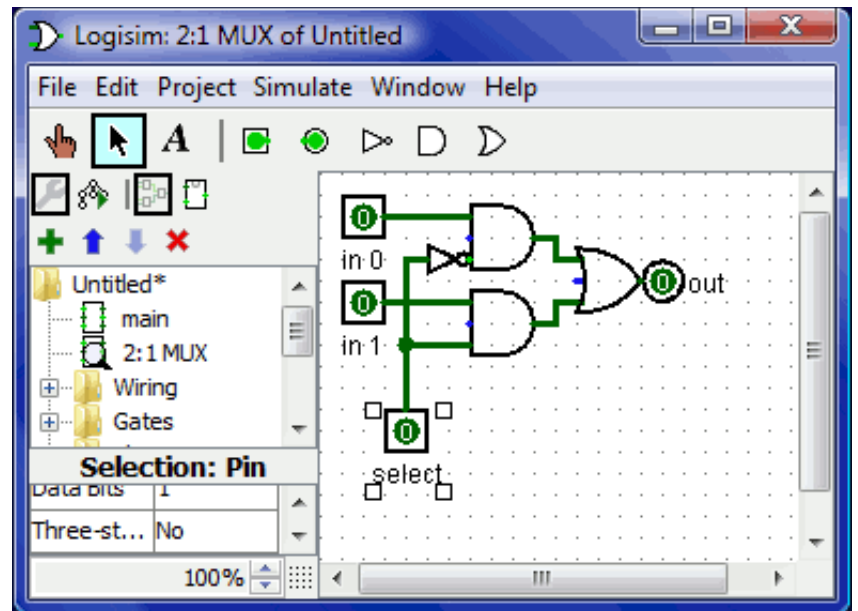
Every Logisim project is actually a library of circuits. In its simplest form, each project has only one circuit (called "main" by default), but it is easy to add more: Select Add Circuit... from the Project menu, and type any name you like for the new circuit you want to create.

Suppose we want to build a 2-to-1 multiplexer named "2:1 MUX." After adding the circuit, Logisim will look like this.



In the explorer pane, you can now see that the project now contains two circuits, "main", and "2:1 MUX." Logisim draws a magnifying glass over the icon of the circuit currently being viewed; the current circuit name also appears in the window's title bar.

After editing the circuit to appear like a 2:1 multiplexer, we might end up with the following circuit.



Next: [Using subcircuits.](#)

Logisim

*a graphical tool for designing
and simulating logic circuits*

Guide to Being a Logisim
User

Beginner's tutorial

Libraries and attributes

Subcircuits

Creating circuits

Using subcircuits

Editing subcircuit
appearance

Debugging subcircuits

Logisim libraries

Wire bundles

Combinational analysis

Menu reference

Memory components

Logging

Command-line verification

Application preferences

Project options

Value propagation

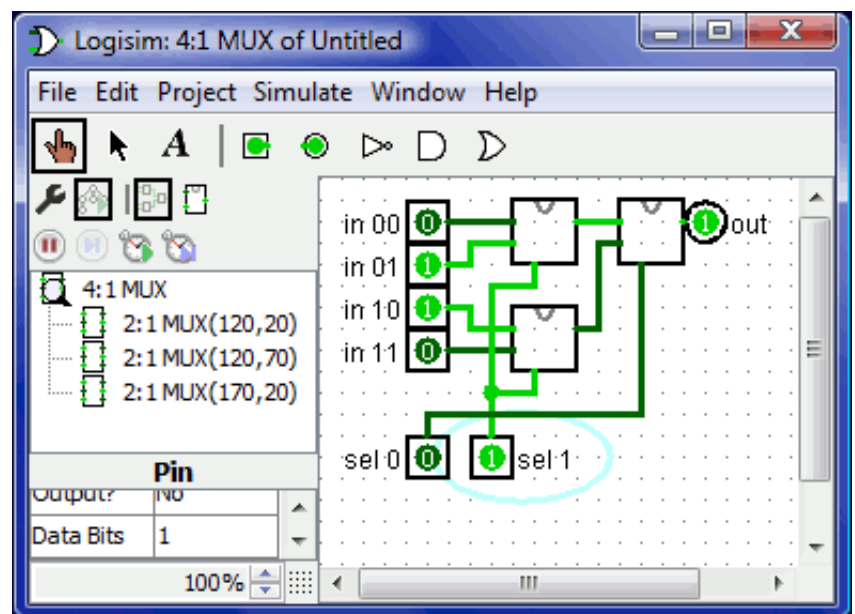
JAR libraries

About the program

Library Reference

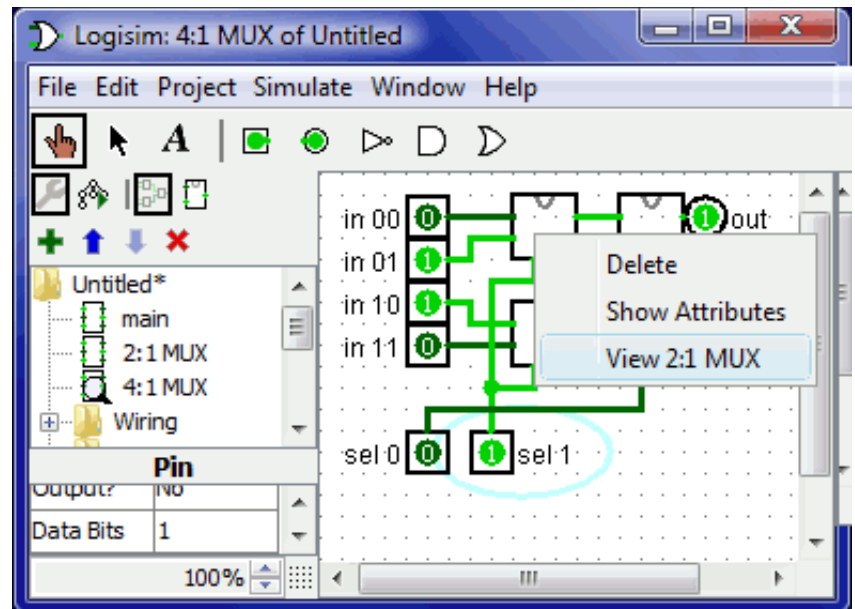
Debugging subcircuits

As you test larger circuits, you will likely find bugs. To nail down what's going wrong, exploring what's going on in the subcircuits while running the overall circuit can help. To enter the subcircuit's state, you can use any of three different techniques. The most straightforward is probably to view the simulation hierarchy by clicking the second icon in the explorer pane's upper toolbar (8), or by selecting "View Simulation Tree" from the Project menu. This switches the explorer pane so that it shows the hierarchy of subcircuits being simulated.



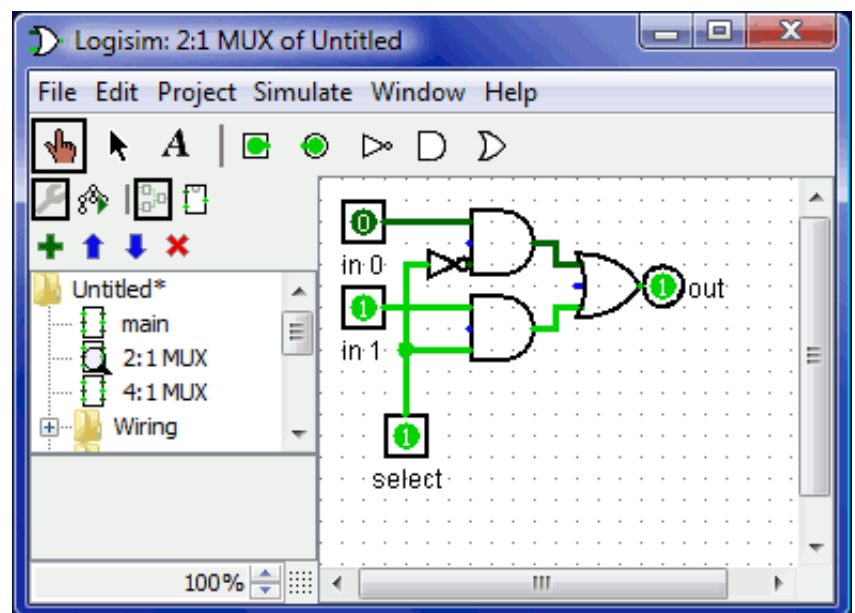
Double-clicking an element in this hierarchy will display what is happening inside that subcircuit.

The second way you can enter a subcircuit is to bring up its popup menu by right-clicking or control-clicking it, and then choosing the View option.



And the third way is to first ensure the Poke Tool is selected and then click the subcircuit you want to enter; a magnifying glass will appear over the subcircuit's center, and double-clicking the magnifying glass will enter the subcircuit's state.

In any case, once you enter the subcircuit, you'll see that the pins' values in the subcircuit match the values being sent through them from the containing circuit.



While in the subcircuit, you are allowed to alter the circuit. If the changes affect any of the subcircuit's outputs, they are propagated into the containing circuit. One exception: The subcircuit inputs are determined based on the values coming into the circuit from the supercircuit, so it doesn't make sense to toggle those values. If you attempt to poke a subcircuit's input, a dialog will pop up asking, "The pin is tied to the supercircuit state. Create a new circuit state?" Clicking No will cancel the toggle request, while

clicking Yes will create a copy of the viewed state, divorced from the outer circuit, with the input pin toggled.

Once you have completed viewing and/or editing, you can return to the parent circuit either by double-clicking the parent circuit in the explorer pane, or via the Go Out To State submenu of the Simulate menu.

Next: [Logisim libraries](#).



Guide to Being a Logisim User

Beginner's tutorial

Libraries and attributes

Subcircuits

Creating circuits

Using subcircuits

Editing subcircuit appearance

Debugging subcircuits

Logisim libraries

Wire bundles

Combinational analysis

Menu reference

Memory components

Logging

Command-line verification

Application preferences

Project options

Value propagation

JAR libraries


About the program

Library Reference


Editing subcircuit appearance

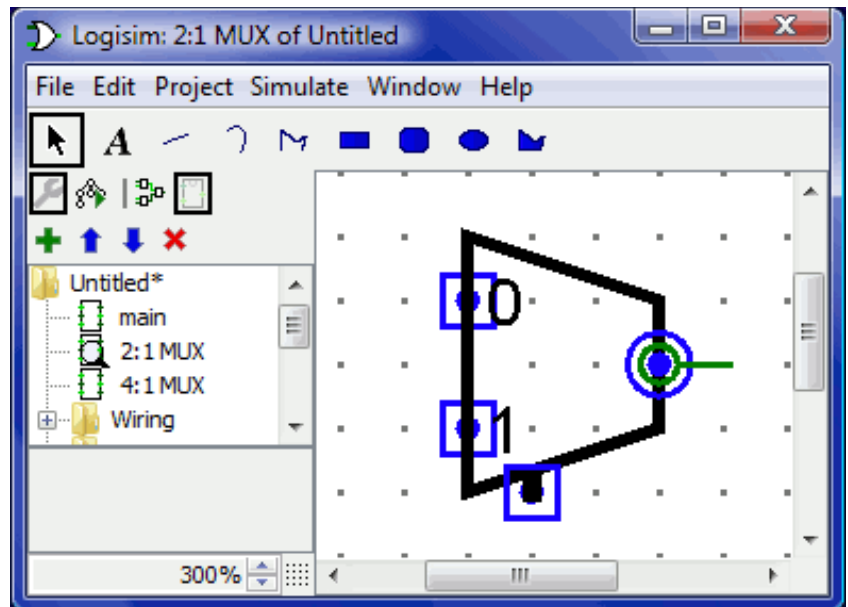
Default appearance

By default, when a subcircuit is placed within a larger circuit, it is drawn as a rectangle with a notch indicating the north end of the subcircuit's layout. Pins will be placed on the rectangle's border based on their facing: Pins that face east in the layout (and typically appear on the west side of the layout) will be placed on the rectangle's west side, according to their top-down ordering in the layout. Pins that face south in the layout (typically toward the north side of the layout) will be placed on the rectangle's north side, according to the left-to-right ordering in the layout.

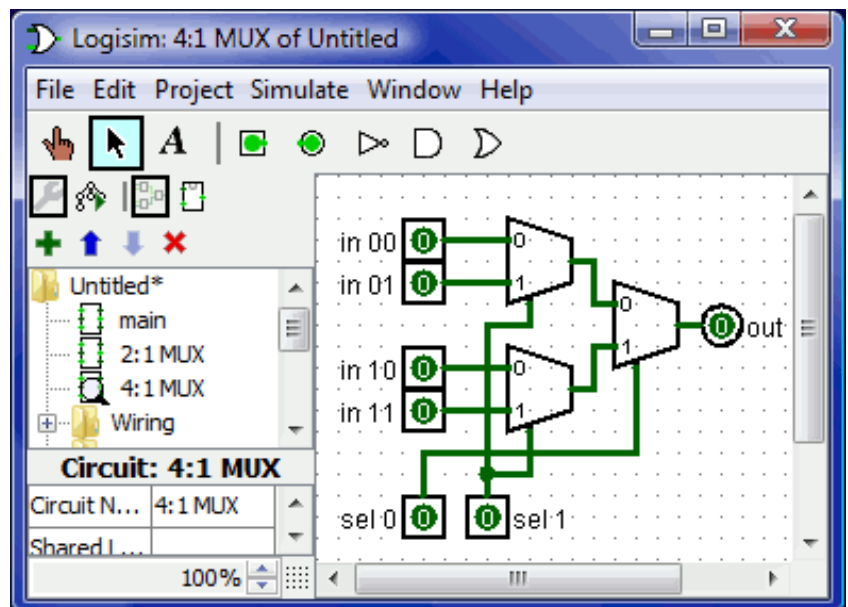
The default rectangle can optionally include some letters that will appear in the middle of the rectangle. To specify this, select the selection tool () and click the background of the circuit's layout. This will show the circuit attributes in the attribute table, including the Shared Label, Shared Label Facing, and Shared Label Font attributes. The value of the Shared Label attribute will be drawn in the rectangle's center; the Shared Label Facing attribute customizes which direction the text is drawn, and of course the Shared Label Font attribute customizes the font used.

Customized appearance

The default appearance is very usable, and indeed Logisim existed for many years with no other option. If, however, you prefer that the subcircuit be drawn differently, you can select "Edit Circuit Appearance" from the Project menu, and Logisim's interface will switch from its regular layout-editing interface to an interface for drawing the circuit's appearance. (You can also click the far-right icon () in the explorer pane's upper toolbar.) Below, we are editing the 2:1 multiplexer's appearance so that it is drawn with the usual trapezoid rather than a rectangle.



With the appearance for the 2:1 multiplexer drawn as above, the layout for the 4:1 multiplexer would then appear as the following.



The appearance editor is like a traditional drawing program, but there are a few special symbols for indicating how the drawing works when placed into a circuit's layout. These special symbols cannot be removed.

- The green circle with a line coming out of it, which we'll call the "anchor." There is exactly one anchor in each subcircuit appearance. Each component in a circuit has a single point identifying its location; a user sees this when creating a new component: The mouse click identifies just a single location, and the component is placed relative to that (usually with the primary output at the mouse's location) The anchor identifies the mouse's location relative to the overall drawing when the subcircuit is created.


The anchor also identifies the appearance's facing, as indicated by the direction the anchor's line points from its circle. When placing the subcircuit into a layout, the user can change the subcircuit's facing; the anchor's facing indicates in which direction the appearance is oriented. In our example, the anchor is facing east, and each instance of the subcircuit in the 4:1 multiplexer is also facing east, so they are all drawn in the same orientation as the 2:1 multiplexer's appearance.


- The blue circles and squares with dots in them are the subcircuit's "ports." There are exactly as many ports as there are input and output pins in the circuit. Ports corresponding to inputs are drawn as squares, while ports corresponding to outputs are drawn as circles. Each port indicates how a wire connecting into the circuit will correspond to an input or output pin within the layout.


When you select a port, Logisim will indicate the corresponding pin by popping up a miniature diagram of the layout in the window's bottom right corner, with the corresponding pin(s) drawn in blue. This does not happen when all ports are selected.


The toolbar contains tools for adding additional shapes, as listed below with descriptions of how the shift and alt key modifies the tool behavior. In addition, clicking or dragging the mouse with the control key pressed regularly snaps the mouse position to the nearest grid point.


 Select, move, copy, and paste shapes.


 Add or edit text.

 Create a line segment. Shift-drag keeps the line's angle at a multiple of 45°.

 Create a quadratic Bezier curve. For the first drag, where you specify the curve's endpoints, shift-drag keeps the endpoints at an angle that is a multiple of 45°. Then you click to indicate the control point's location; shift-click ensures the curve is symmetric, while alt-click draws the curve through the control point.

 Create a sequence of connected lines, whose vertices are indicated by a succession of clicks. Shift-clicking ensures that the angle between the previous vertex and the current one is a multiple of 45°. Double-click or press the Enter key to complete the shape.

 Create a rectangle through dragging from one corner to the opposite corner. Shift-drag to create a square, and alt-drag to create the rectangle starting from the center.

 Create a rectangle with rounded corners through dragging from one corner to the opposite corner. Shift-drag to create a

square, and alt-drag to create the rectangle starting from the center.

- Create an oval through dragging from one corner of its bounding box to the opposite corner. Shift-drag to create a circle, and alt-drag to create the oval starting from the center.
- ▮ Create an arbitrary polygon, whose vertices are indicated by a succession of clicks. Shift-clicking ensures that the vertex is at a 45° angle from the previous one. Double-click, press the Enter key, or click the starting vertex to complete the shape.

Next: [Debugging subcircuits](#).



Guide to Being a Logisim User

[Beginner's tutorial](#)

[Libraries and attributes](#)

[Subcircuits](#)

[Creating circuits](#)

[Using subcircuits](#)

[Editing subcircuit appearance](#)

[Debugging subcircuits](#)

[Logisim libraries](#)

[Wire bundles](#)

[Combinational analysis](#)

[Menu reference](#)

[Memory components](#)

[Logging](#)

[Command-line verification](#)

[Application preferences](#)

[Project options](#)

[Value propagation](#)

[JAR libraries](#)

[About the program](#)

[Library Reference](#)

Logisim libraries

Every Logisim project is automatically a library that can be loaded into other Logisim projects: Just save it into a file and then load the library within another project. All of the circuits defined in the first project will then be available as subcircuits for the second. This feature allows you to reuse common components across projects and to share favorite components with your friends (or students).

Each project has a designated "main circuit," which can be changed to refer to the current circuit via the Set As Main Circuit option in the Project menu. The *only* significance of this is that the main circuit is the one that is displayed when you first open the project. The default name of the circuit in a newly created file ("main") has no significance at all, and you can feel free to delete or rename that circuit.

With a loaded Logisim library, you are allowed to view circuits and manipulate their states, but Logisim will prevent you from altering the circuits' design and other data stored within the file.

If you want to alter a circuit in a loaded Logisim library, then you need to open it separately within Logisim. As soon as you save it, the other project should automatically load the modified version immediately; but if it does not, you can right-click the library folder in the explorer pane and select Reload Library.

Next: [User's Guide](#).



Guide to Being a Logisim User

Beginner's tutorial

Libraries and attributes

Subcircuits

Creating circuits

Using subcircuits

Editing subcircuit appearance

Debugging subcircuits

Logisim libraries

Wire bundles

Combinational analysis

Menu reference

Memory components

Logging

Command-line verification

Application preferences

Project options

Value propagation

JAR libraries

About the program

Library Reference

Subcircuits

As you build circuits that are more and more sophisticated, you will want to build smaller circuits that you can use multiple times as a module nested within larger circuits. In Logisim, such a smaller circuit that is used in a larger circuit is called a **subcircuit**.

If you're familiar with computer programming, you're familiar with the subprogram concept, whether it's called a *subroutine*, *function*, *method*, or *procedure* in your favored language. The subcircuit concept is analogous to this, and it serves the same purpose: To break a large job into bite-sized pieces, to save the effort of defining the same concept multiple times, and to facilitate debugging.

[Creating circuits](#)

[Using subcircuits](#)

[Editing subcircuit appearance](#)

[Debugging subcircuits](#)

[Logisim libraries](#)

Next: [Creating circuits](#).



Guide to Being a Logisim User

Beginner's tutorial

Libraries and attributes

Subcircuits

Creating circuits

Using subcircuits

Editing subcircuit appearance

Debugging subcircuits

Logisim libraries

Wire bundles

Combinational analysis

Menu reference

Memory components

Logging

Command-line verification

Application preferences

Project options

Value propagation

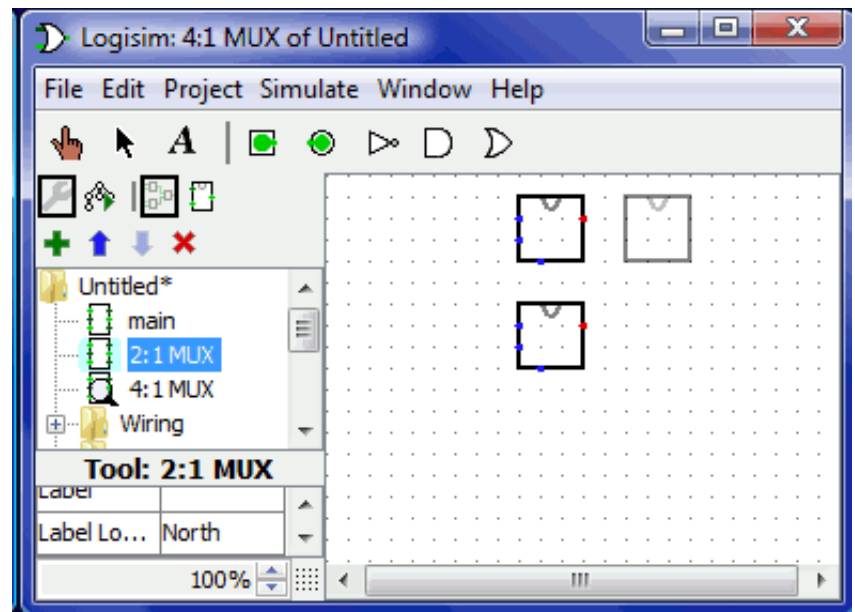
JAR libraries

About the program

Library Reference

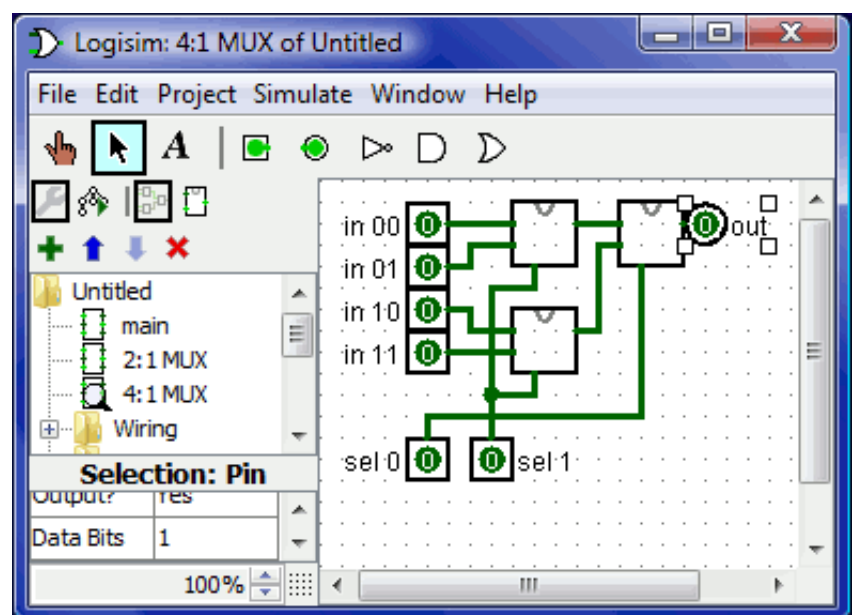
Using subcircuits

Now suppose we want to build a 4-to-1 multiplexer using instances of our 2-to-1 multiplexer. Of course, we would first create a new circuit, which we'll call "4:1 MUX." To add 2-to-1 multiplexers into our circuit, we click the 2:1 MUX circuit *once* in the explorer pane to select it as a tool, and then we can add copies of it, represented as boxes, by clicking within the canvas.



If you were to double-click the 2:1 MUX circuit in the explorer pane, then the window would switch to editing the 2:1 MUX circuit instead.

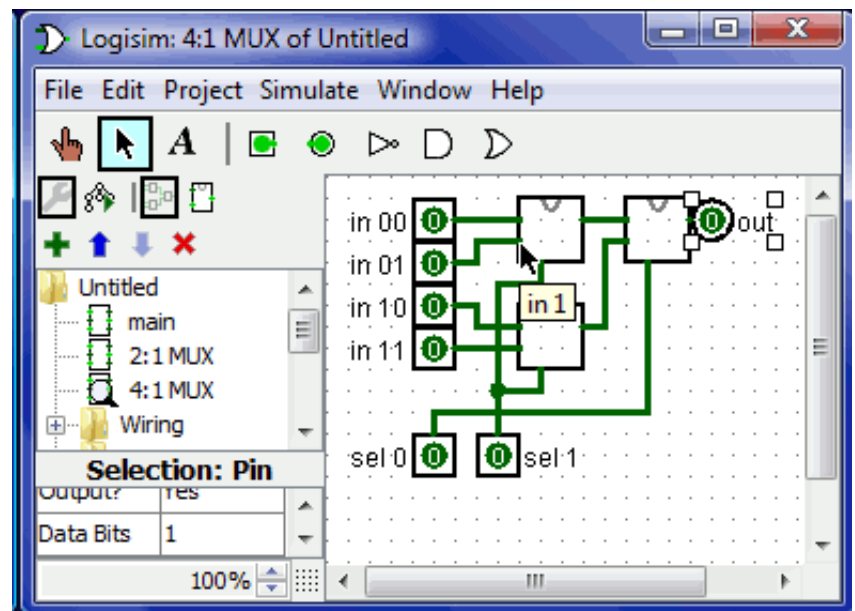
After building up the circuit, we end up with the following.



Our circuit for a 4-to-1 multiplexer uses three copies of the 2-to-1

multiplexer, each drawn as a box with pins along the side. The pins on this box correspond to the input and output pins in the 2:1 MUX circuit. The two pins on the west side of the box correspond to the two pins that face east in the 2:1 MUX circuit; the pin on the box's east side corresponds to the 2:1 MUX's west-facing pin (which happens to be an output pin); and the pin on the box's south side corresponds to the 2:1 MUX's north-facing pin. The order of the two pins on the box's west side correspond to the same top-down ordering from the subcircuit's design. (If there were several pins on the box's north or south side, they would correspond to the same left-right order in the subcircuit.)

If the pins in the subcircuit's layout have labels associated with them, then Logisim will display that label in a **tip** (that is, a temporary text box) when the user hovers the mouse over the corresponding location of the subcircuit component. (If you find these tips irritating, you can disable them via the [Preferences window's Layout tab](#).)



Several other components will display these tips, too: For some of the pins of a built-in [flip-flop](#), for example, hovering over it explains what that pin does.

Incidentally, every pin to a circuit must be either an input or an output. Many manufactured chips have pins that behave as an input in some situations and as an output in others; you cannot construct such chips within Logisim (at least, in the current version).

Logisim will maintain different state information for all subcircuits appearing in a circuit. For example, if a circuit contains a flip-flop, and that circuit is used as a subcircuit several times, then each subcircuit's flip-flop will have its own value when simulating the larger circuit.

Now that we have the 4-to-1 multiplexer defined, we can now use it in other circuits. Logisim has no limits on how deeply circuits can be nested - though it will object to nesting circuits within themselves!

Note: There's nothing wrong with editing a circuit that is being used as a subcircuit; in fact, this is quite common. Be aware, though, that any changes to a circuit's pins (adding, deleting, or moving them) will rearrange them also in the containing circuit. Thus, if you change any pins in a circuit, you will also need to edit any circuits using it as a subcircuit.

Next: [Editing subcircuit appearance](#).