

# CSCB58: Logisim-Evolution Reference

Alexander Klemenchuk (last updated by Moshe Gabel, January 2022)

This document is based on a slightly older version of Logisim-evolution, so some of the details and screenshots may be a little off.

## 1 Introduction

Logisim is a powerful logic circuit simulation environment. In CSCB58, we will use Logisim-Evolution (a fork of the original Logisim). To obtain the software, go to the following link:

<https://github.com/logisim-evolution/logisim-evolution/releases>

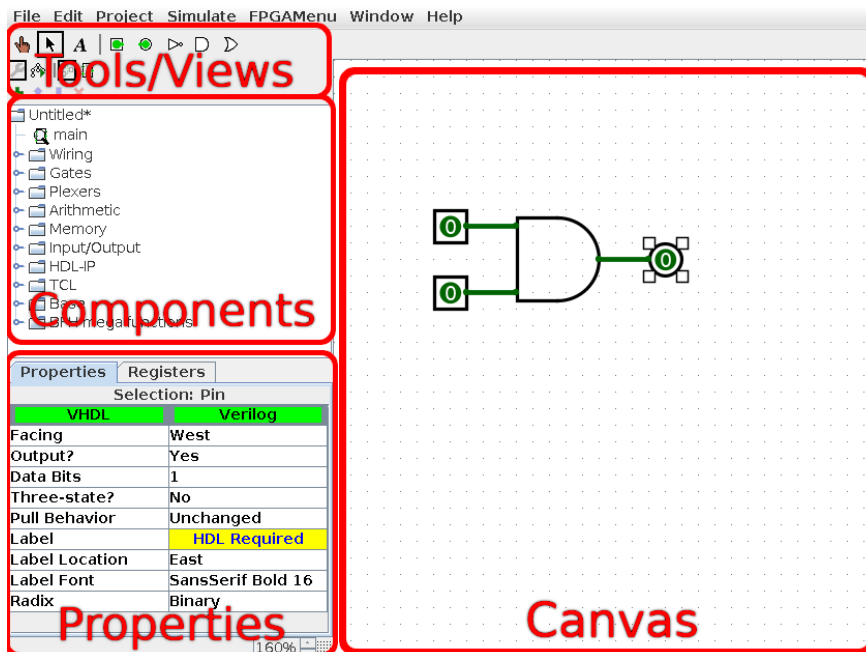
and download one of the installers or the jar file. **Note:** Make sure to use Logisim-Evolution downloaded at the above link. Do NOT use the original Logisim or any other variations of it.

To launch the program, simply double click on `logisim-evolution.jar` on most operating systems with a working Java installation. You might need to perform a `chmod +x logisim-evolution.jar` command to make the file executable. If you have issues with the scaling of the interface on high resolution displays, it may be worth it to try updating Java to version 9 or higher.

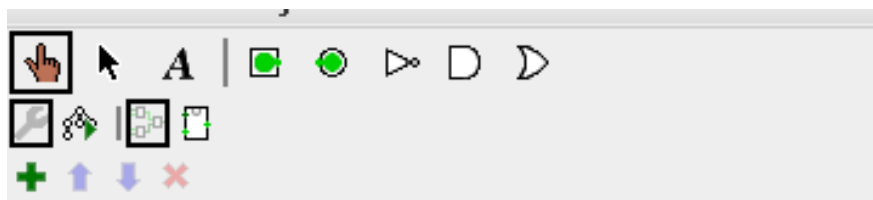
Logisim works with `.circ` files. There is one `.circ` file per “project”. You can add multiple circuits to the same file. Since the software is easy to run on any computer and the files are portable, it’s easy to work from home!

This is a very brief overview of some of the features available. For a tutorial walking through the basics of everything offered by Logisim-Evolution, including how to create and analyze circuits, have a look at the excellent documentation. Find this in Help → Tutorial

## 2 The Interface



## 2.1 Tools and Views



- “Poke” tool: Click on wires to inspect their state, click on most components to change their state.
- “Select” tool: Selects and moves things in the canvas, and manipulates wires/buses. Click and drag from inputs/outputs to create wires/buses. Click to select individual wire segments. Click and drag from an empty region to box-select components. Keep in mind, overlapping wires and inputs will always connect, so be careful! You can also click and drag the end of wire segments to detach them from components. **You can also hold shift while clicking and dragging a component to leave connections behind.**
- Text tool. Click to add text.
- On the top row there are also shortcuts to commonly used components. Like the input pin () and the output pin () This can be customized in **Project** → **Toolbar**.
- The button views the components list, and the button views the simulation hierarchy (this is not important for the course).
- The button views the circuit in the canvas and the button views the circuit symbol so you can customize how it looks as a sub-circuit.
- The button adds a new circuit, move circuits up and down in the list, and deletes a circuit.

## 2.2 Properties

Memory

Properties

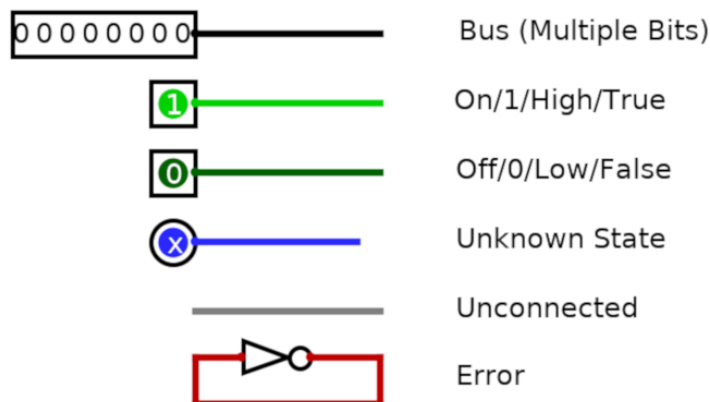
Registers

Selection: Pin

VHDL	Verilog
Facing	East
Output?	No
Data Bits	2
Three-state?	No
Pull Behavior	Unchanged
Label	my_input
Label Location	North
Label Font	Comic Sans MS Bo...
Radix	Binary

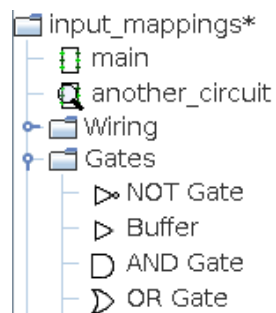
This section of the interface allows you to customize the currently selected component in the canvas. This Pin is configured to have two data bits (bus), is an output pin, and has a customized label.

## 2.3 Wire Coloring



Wires and buses can have many states. You can inspect the state of a wire/bus using the 🖱️ “poke” tool available in the toolbar.

## 2.4 Components



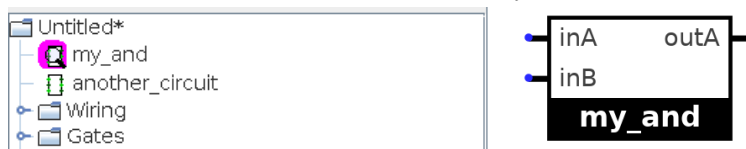
This contains all the circuits in this file as well as all the built-in components.

Double-click on each circuit to view it. To place a component from this list, select it, and then click somewhere in the canvas.

## 2.5 Sub-Circuits (Using a circuit in another circuit)

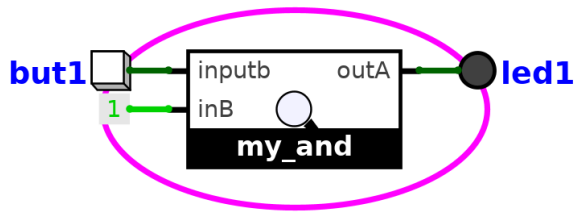
All circuits you make are automatically components, appearing in the components pane at the top of the list. This means you can use a circuit in another circuit by placing it just like any other component.

To add another circuit to the project, either click on the plus button above the components pane, or right click on the folder icon with the name of your file and select “Add Circuit”.



This is the default appearance of a sub-circuit, this can be customized by clicking the “Edit circuit Appearance” button in the Tools/Views pane.

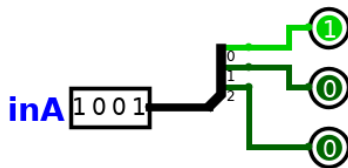
**Note:** Only named Pin input and outputs will be accessible from the circuit that uses the symbol.



Using the 🖱️ “poke/hand” tool from the toolbar you can double click on the center (the magnifying glass) and see what the subcircuit is doing with that input/output! This is a live view, so clocks will keep updating from the outer circuit.

### 3 Buses

Buses are wires that contain multiple bits. Buses are created automatically when wires are connected to components that have multi-bit inputs or outputs. Buses can be split into smaller buses or individual wires with a **Splitter**. Here is an example using both:



Here, I have connected a multi-bit input **Pin** to a splitter, then to three single bit output pins. The input is 4-bits “wide” as I have set the **Data Bits** property to 4.

To edit the value in the input **Pin**, you can use the “Poke” 🖱️ tool, clicking to toggle bits.

Properties	Registers
Selection: Splitter	
VHDL	Verilog
Facing	East
Fan Out	3
Bit Width In	4
Appearance	Left-handed
Bit 0	0 (Top)
Bit 1	1
Bit 2	2 (Bottom)
Bit 3	None

The splitter takes in a 4 bit input as it’s **Bit Width In** is set to 4. It splits it into 3 wires, as it’s **Fan Out** is set to 3. Bit 0 of the input is sent to output 0, 1 to 1, 2 to 2, and bit 4 is sent nowhere. If you send multiple bits to the same output, that output becomes a bus.



**Splitters work both ways! In this example, the inputs and outputs could be switched.**






### 4 Component Descriptions

Far more detailed descriptions are also available in **Help → Library Reference**






The following is a list of the components that bear mentioning.

#### 4.1 Wiring

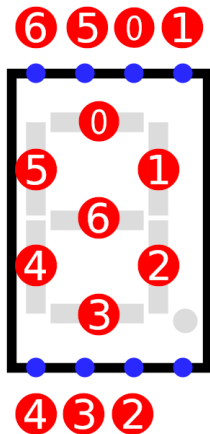
-  **Splitter:** Splits buses into individual wires or smaller buses. Works both ways.
-  **Pin:** Appears in sub-circuit symbols, mappable to inputs/outputs on DE-2 board if input/output respectively. Like most components, has a configurable number of bits.

-  **Clock:** Toggles when you press **Ctrl+T**, can toggle on a timer by going into the **Simulate** dropdown in the menu-bar, **Ticks Enabled** turns this on and **Tick Frequency** controls how fast it ticks. This works on the DE-2 board!
-  **Probe:** Can be attached to a wire to display its state.
-  **Tunnel:** Tunnels with the same label are connected to each other.
-  **Constant:** Outputs a constant value (can be multiple bits on a bus).
-  **Bit Extender:** Pads or sign extends bits on a bus.

## 4.2 Input/Output

-  **Button:** Can be mapped to the switches and buttons on the DE-2 board. Only outputs a 1 when held down with the  “poke” tool.
-  **Dip switch:** Can be mapped to the switches and buttons on the DE-2 board. Is multiple switches in one.
-  **7-Segment Display:** Can be mapped to the 7-segment display on the DE-2 board. More information below.
-  **LED:** Can be mapped to the outputs on the DE-2 board.

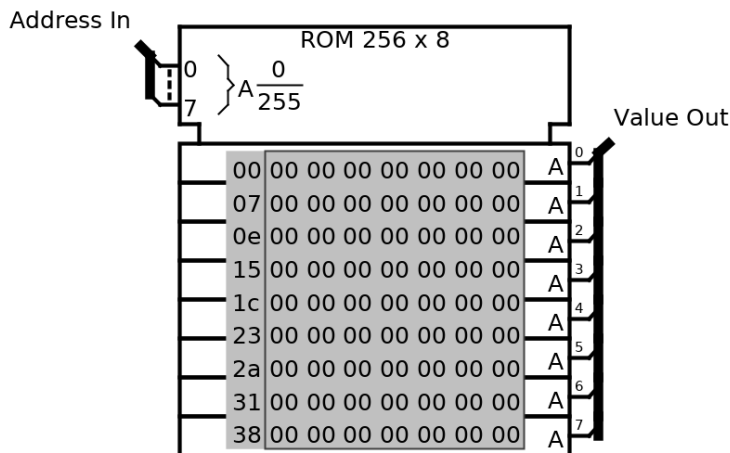
## 4.3 7-Segment Display



Above is the mappings of the pins to segments of the 7-segment component.

The 7-segment display can be mapped to any the displays on the DE-2 board. The decimal point can not be mapped as it is not connected.

## 4.4 ROM



Use the “poke” tool to select a memory cell and type a value in hexadecimal.

You can right click and “Edit Contents” to open a full editor that views the contents as hexadecimal and has options to save it as a file that can be loaded later.

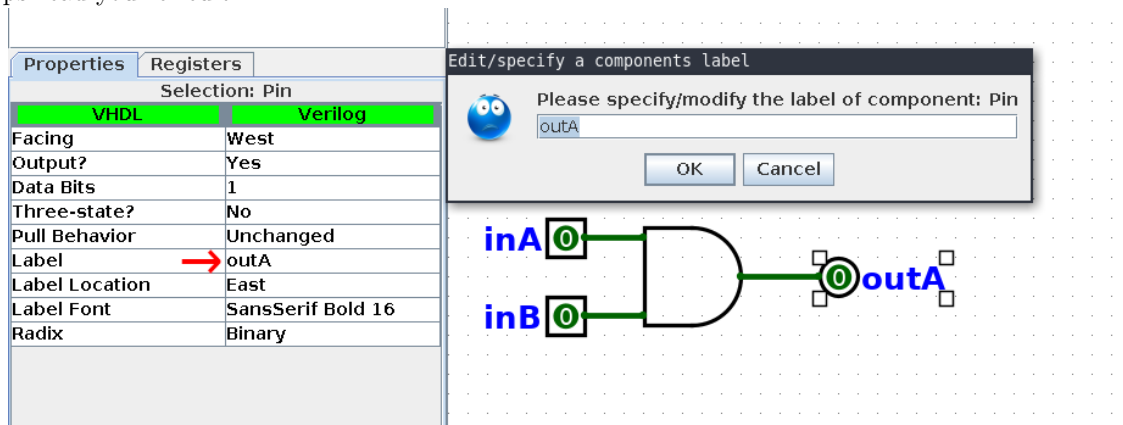
This block works on the DE-2 board!

**Note on a possible bug:** If you move the ROM block before editing it’s contents in the canvas, buggy behavior might happen. To fix this, if you get issues with the ROM block, simply reset the simulation with Simulate → Reset Simulation or hit Ctrl+R

## 5 Best Practices

### 5.1 Naming Inputs/Outputs

Not only will this create more usable circuit symbols, and allow you to program the DE-2 board, but it also helps read your circuit.



To give a component a name, either double click on it, change the Label property in the Properties window, or hit the **Annotate** button when about to program (see later).

### 5.2 Naming Circuits

It’s important to give your circuits/sub-circuits good names.

To rename a circuit, click on the circuit in the components list, and edit it’s **Circuit Name** property.

### 5.3 Sub-Circuit tips

- Use sub-circuits any time you find yourself repeating a pattern in a circuit

- Don't use any other input/output device in a sub-circuit other than `Pin`. They won't show up in outer circuits and are a nuisance when programming the board.