

## Acknowledgement:

Antian Wang prepared this part of the tutorial. Part of the content was inspired and adapted from UC Berkeley CS61C Great Ideas of Computer Architecture (Machine Structures) Lab 5 logisim evolution.

## Version info

Manual Version	Software Version	Author	Memo	Month/Year
V1.0	V3.8.0	Antian Wang	Initial setup	Dec. 2023.

## Software download:

Logisim evolution can be downloaded from <https://github.com/logisim-evolution/logisim-evolution>

Go to the release link (<https://github.com/logisim-evolution/logisim-evolution/releases>) if you don't want to build the software by yourself.

According to the info from <https://github.com/logisim-evolution/logisim-evolution#download>, please download your installation package accordingly.

logisim-evolution-<version>-1\_amd64.deb: Debian package (also suitable for Ubuntu and derivatives),

logisim-evolution-<version>-1.x86\_64.rpm: Package for Fedora/Redhat/CentOS/SuSE Linux distributions,

logisim-evolution-<version>-amd64.snap: The Snap archive for all supported Linux distributions (also available in Snapcraft store),

logisim-evolution-<version>-aarch64.msi: Installer package for Microsoft Windows for Arm processors,

logisim-evolution-<version>-amd64.msi: Installer package for Microsoft Windows for Intel processors,

logisim-evolution-<version>-aarch64.dmg: macOS package for Apple processors,

logisim-evolution-<version>-x86\_64.dmg: macOS package for Intel processors (also runs on Apple processors in simulation).

In most of the cases:

Windows user, please use logisim-evolution-<version>.msi

Mac user, please use logisim-evolution-<version>.dmg

Linux user, please use logisim-evolution-<version>-1.x86\_64.rpm.

Please notice that Logisim Evolution is an open-source education software. Some of your computers may prohibit installing it.

For Windows user:

After seeing the warning sign, Please click **MORE INFO**, and you will see the install button.

For Mac users:

You need to open it in the downloaded path for the first usage. Since it is an open-source software, Apple may not allow you to open it by default. You need to follow the steps shown below to open the software when you try to open it. Click open if there is a pop-up window following.



Figure 1 MacOS warning.

### Apple can't check app for malicious software

Using an app that can't be checked for malicious software might harm your Mac or compromise your privacy. If you're certain that the app you want to use is from a trustworthy source, you can override your Mac security settings to open it.


1. In the Finder  on your Mac, locate the app you want to open.  
Don't use Launchpad to locate the app.
2. Control-click the app icon, then choose Open from the shortcut menu.
3. Click Open.  
The app is saved as an exception to your security settings, and you can open it at any time by double-clicking it, just as you do with any registered app.

Figure 2 MacOS open unknown software.

<https://support.apple.com/guide/mac-help/apple-cant-check-app-for-malicious-software-mchleab3a043/mac>

For Linux user:

Logisim evolution need java 16 or newer support.

Download the rpm file.

Get sudo previlage

```
sudo alien -i xxxx.rpm
```

### Known issue for Windows

Note that there may be an issue with logisim evolution, making some download files to .circ extension. You can fix it using the method here.

<https://github.com/logisim-evolution/logisim-evolution/issues/1471#issuecomment-1332917265>

"Removing the registry key HKEY\_CLASSES\_ROOT\MIME\Database\Content Type\application/octet-stream\Extension instantly resolves the issue"

Known issue for MAC:

Known issue for Linux:

### Tutorial accompanied with the software

In Menu->Help->tutorial, it includes the completed list of features for Logisim evolution. This tutorial only discusses the features that are useful for this lab.

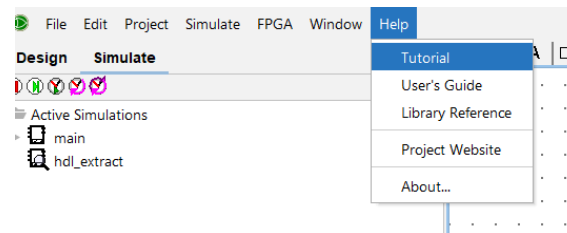


Figure 3 Access software tutorial.

## Background

Logisim Evolution is the newer version of Logisim, which stopped its development in 2014. The Logisim evolution has a better GUI interface and more advanced functionality compared with Logisim. It provides the best of logic simulation and allows you understand the course content from ECE 2010 and ECE 2090. We use it for the ECE 2090 as the software for the lab's simulation part.

The tutorial includes all needed features within the Logisim evolution that are helpful for the ECE 2010 & ECE 2090 course study.

Note that some of the features of logisim evolution are out of the scope of ECE 2010 and ECE 2090 and possibly extend to other ECE courses.

You can explore these features by yourself for your interest, but we only use the gate-level digital logic feature of the software in this lab, including the final project.

## Logisim Evolution usage

### GUI interface

The GUI interface of the Logisim Evolution should look like follows. You can place whatever amount logic gates you want within the scheme on the right.

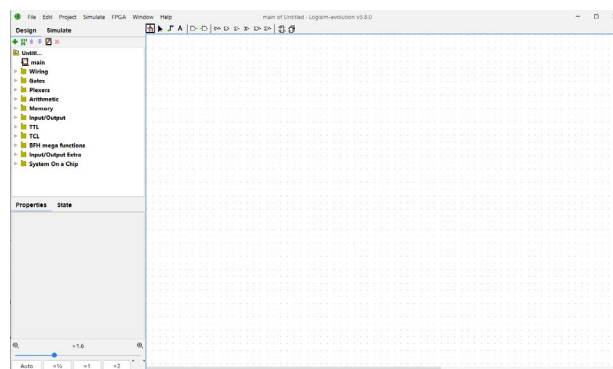


Figure 4 Logisim Evolution GUI interface.

## GUI components

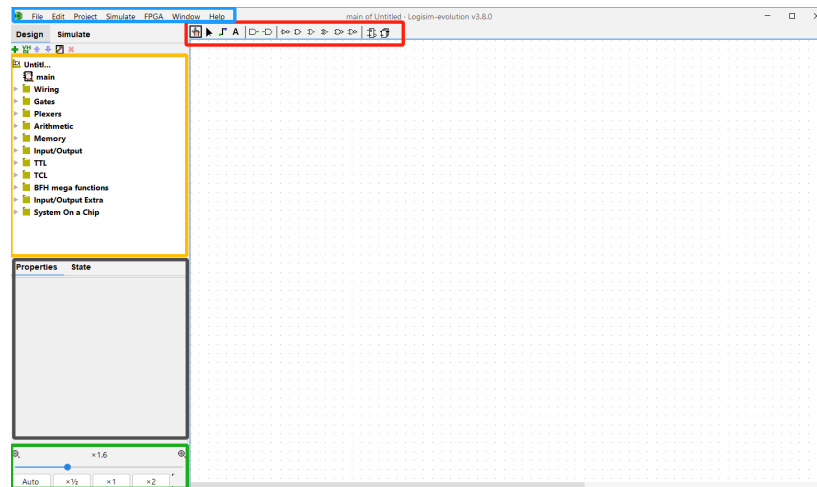


Figure 5 Logisim Evolution GUI region.

## Basic usage shortcut

The red part gives you the shortcut for typical editing tools, input/output pins, and simple logic gates.

When you approach the short cut, there will be a brief hint & short cut key beneath showing its meaning in case you are not familiar with them.

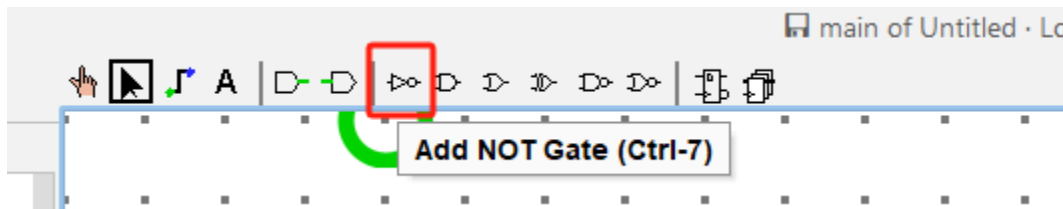


Figure 6 Shortcut toolbar.

## Component library

The orange part shows the components that you can place it to the schematic on the right.

You can click the folder to expand them by clicking the triangle before the folder, and find the needed components.

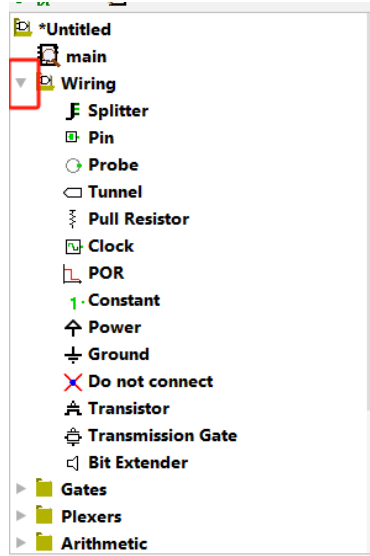


Figure 7 Expand component folder.

### Property & state editor

The black part will show more detailed properties of the components you place that you could possibly edit.

For logic gates, you can use the property is negating one or multiple inputs to its complement. It avoids adding one additional inverter, and ease the wiring difficulty. Below is one simple example.

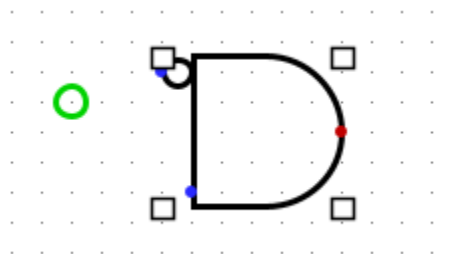


Figure 8 AND gate input negate.

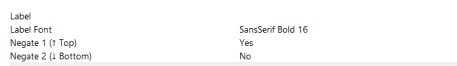


Figure 9 Invert negate property setting.

### Zoom in and zoom out

The green part allows you to zoom in and zoom out the circuit diagram. You can also scroll up or down using your mouse by pressing Ctrl for Windows users, ??? for Mac user, and ??? for Linux user to zoom in and zoom out.

If you happen to zoom in too much, you can re-center the design by clicking the “target ” shown as below.

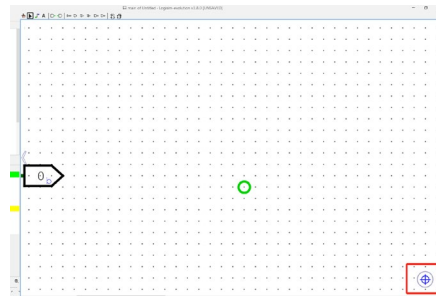


Figure 10 Recenter the schematic

And you can see the complete design within the scheme.

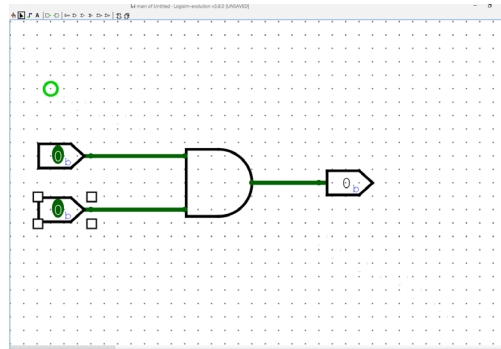


Figure 11 Schematic after recenter.

Basic project wise setting.

The blue part is similar to other GUI-based software that allow you save, edit the project.

### Toggle Input and Output

Input & output pins

The most commonly used IO in the lab is input and output pins.

There are two ways of finding two types of them, from the top of the shortcut toolbar, as shown below.

**You are required to use input pins for your combinational input.**

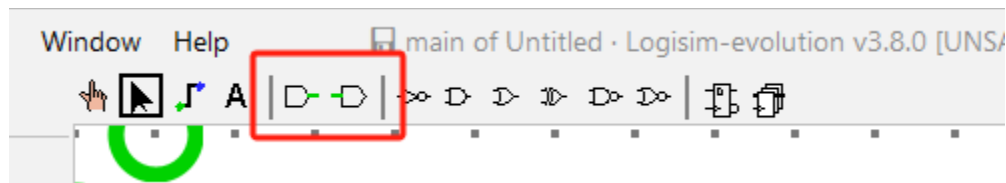


Figure 12 IO pin in shortcut toolbar.

Or find it within the wiring section, and configure it to needed function.

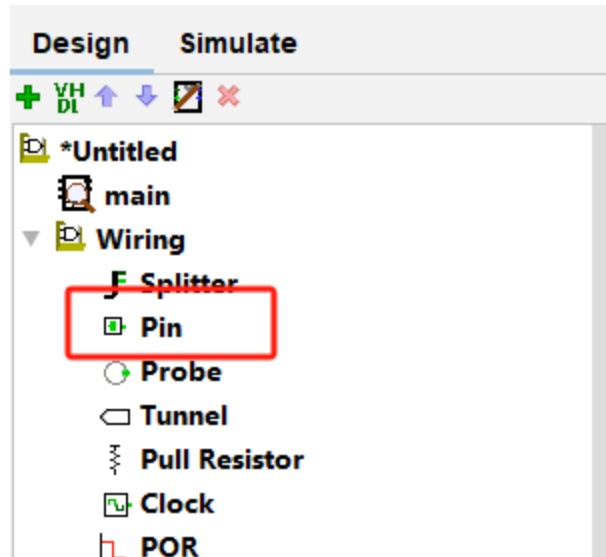


Figure 13 Pin in wiring folder.

Once you place the pin to the canvas, it shows the following property to configure.

Properties	State
<b>Pin (80,230)</b>	
FPGA supported:	Supported
Facing	→ East
Output?	No
Data Bits	1
Three-state?	No
Pull Behavior	Unchanged
Label	HDL Required
Label Font	SansSerif Bold 16
Radix	Binary
Reset value:	0x0
Appearance	Arrow shapes

Figure 14 Pin's property.

### Facing

You can change the facing of the components by your arrow keys on your keyboard or configure here within the property. Below shows the different facing of the input pins, allowing you to make the design compact and ordered.

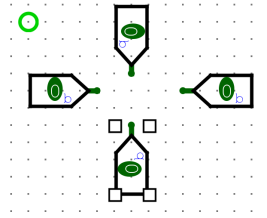


Figure 15 IO pins with different facing.

### Output configuration.

You can directly select the pins as output from the shortcut bar, but you can also configure it from input to output using the drop down menu.

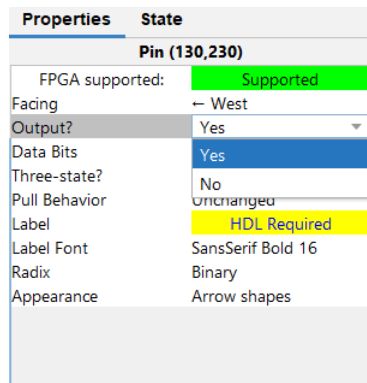


Figure 16 Configure input pin to output pin.

### Data bit width selection

You can also configure the data bits width by simply change the data bits property.

Below is what we look like for 5-bit input, please be cautious about the

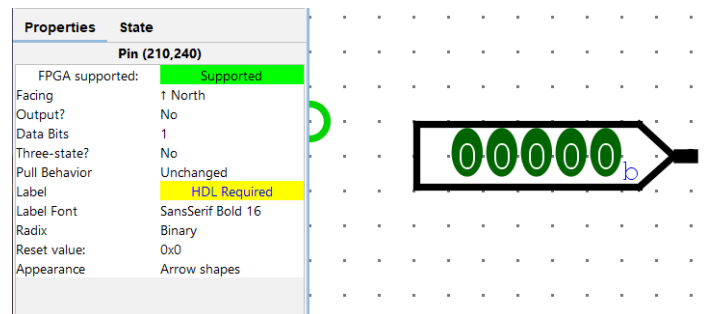


Figure 17 Configure to multi-bit pin.

### Radix

Using different radix may help you identify the value for multi-bit data, especially helpful for arithmetic logic designs.

Below is what it looks like by configuration the input pin to an unsigned decimal and connecting it to an output pin represented in binary you can change the value by clicking the input pins.



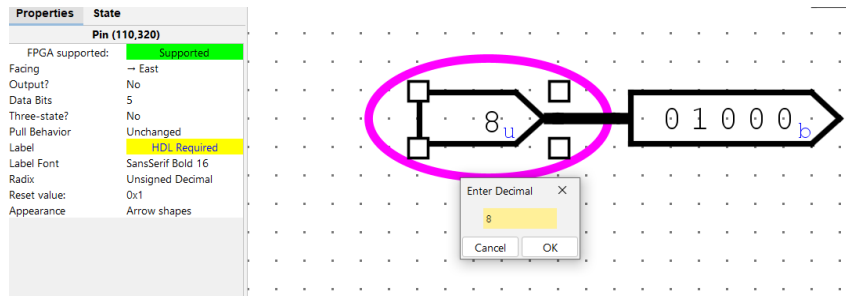


Figure 18 Using non-binary radix pin.

## DIP Switches

In most of the design, you can use DIP switches circuit input, located under input/output folder for convenient logic inputs. Below is the 8-bit DIP switch, you can configure it to whatever bit width you want.

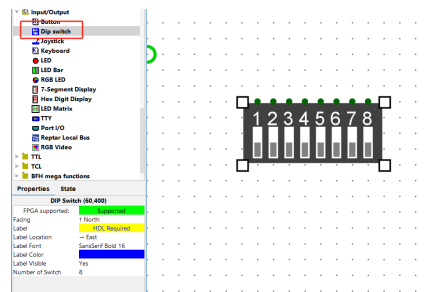
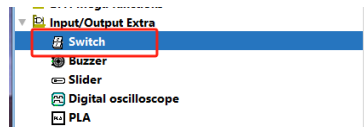


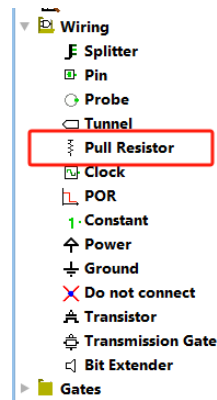
Figure 19 DIP switch.

## Switch

In input-output extra, there exists “switch”, shown as below, different from the DIP switch above. However, this switch is mostly close to what you need to wire if you build an actual circuit.



In such setting, you need to use a new components named “pull resistor”



If you configure the “pull resistor ” pulled to one as configured below, you need to do the following connection.

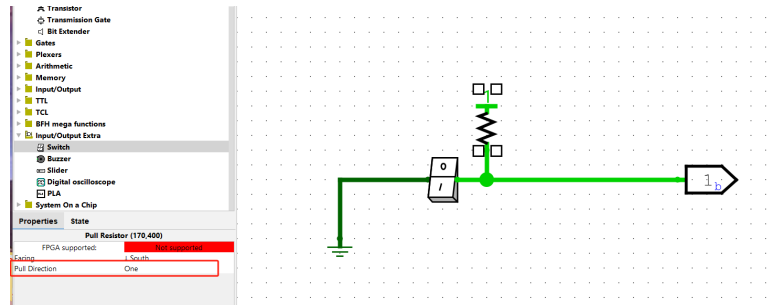


Figure 20 Pull up resistor connection

In such a way, you can produce logic 0 & logic 1 by clicking the switch. If you configure the pull resistor to logic 0, then you need to use the following connection:

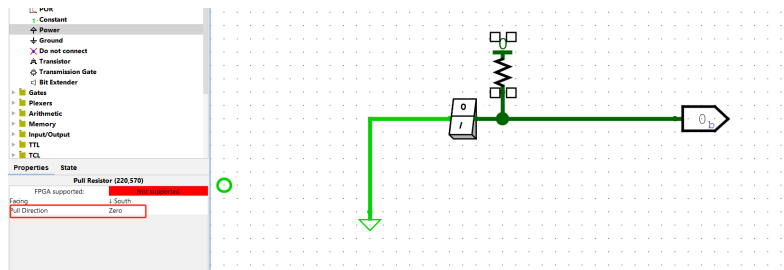


Figure 21 Pull down resistor connection

## LEDs

LED is another simple way to evaluate the output logic value. Here in Logisim Evolution, you can use the LED also RGB LED based on your application choice.

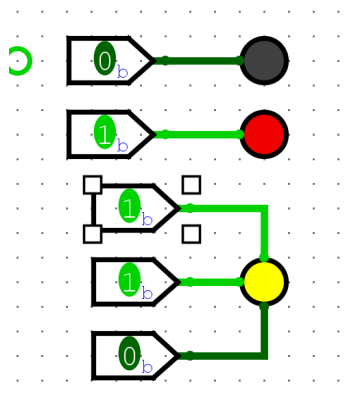


Figure 22 2-color LED and RGB LED.

### Other input-output components

There are other interesting in-out components available for your use to make your design more interactive, like LED bar, LED matrix. You can explore it by yourselves.

### Logic gate connection and configuration

#### Find the logic gate

You can find the logic gate at the top of short-cut toolbars. You may also find more possible logic gates under gate folder

#### Configure the logic gate input number

You can change the input numbers by changing the number of inputs, as 4-input AND gate shown on the right.

#### Configure the size of the logic gate

You can change the gate size to narrow to save some space in the schematics shown on the left.

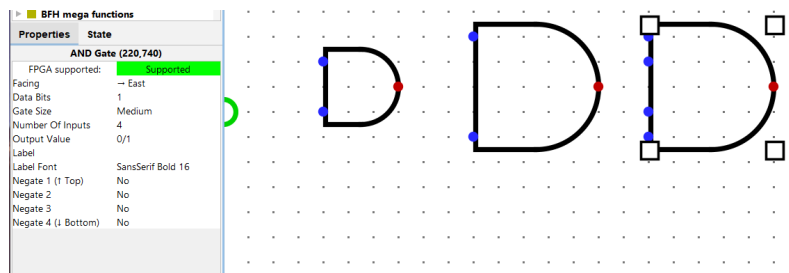





Figure 23 And gate of different size and input pins.

### Verify the functionality

- Place the input/output/And gate on the schematic: Input  Output  And gate 

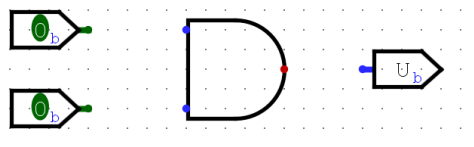




Figure 24 Connect and gates with two input pins and output pin.

- Connect the pins together by 
- Use the  to toggle the pin's value.

You can then verify the functionality of 2-INPUT AND gate.

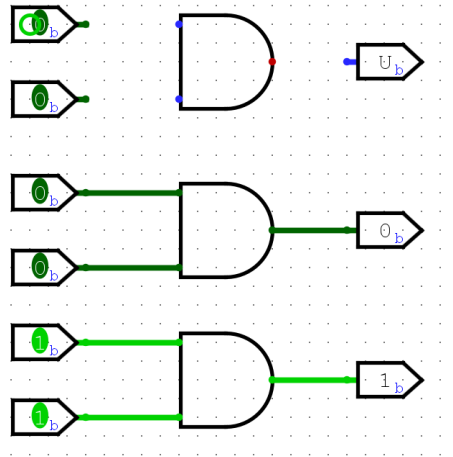


Figure 25 Step of the connection.

## TTL IC configuration

One of the goal for the lab is to familiar with 74 series logic chips. In Logisim Evolution, these gates are available under TTL folder, with its IC number and corresponding functionality. It differs significantly with Logisim where only custom 74 libraries are available, and need to load it every time.

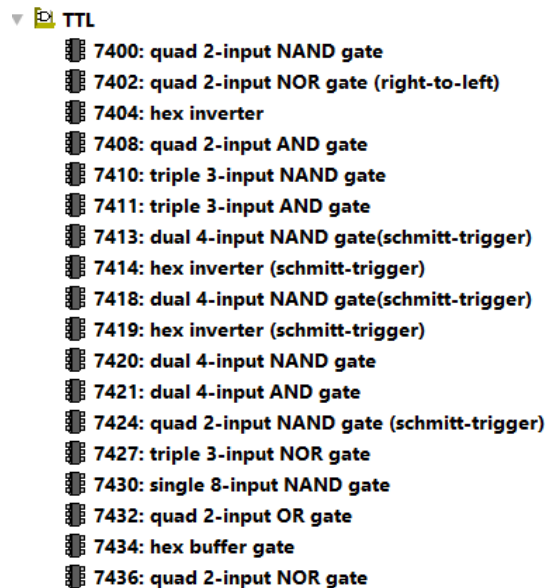


Figure 26 TTL 74 series IC folder

## Find the IC in the menu

Here I use the one of 74 series TTL gate available, 7400 as example.

## Place IC on the schematic

Drag the gate to the schematic, and you will see the following:

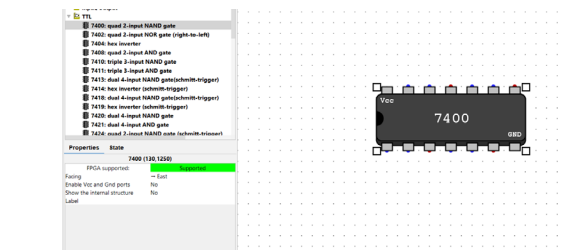


Figure 27 Place IC on schmetic.

## Configure the IC

There are two configurations that we typically exist with most of the 74 series gates. Enable Vcc and Gnd ports and show the internal structure. My preference is not to enable the Vcc and Gnd ports, and enable the show of the internal structure for clarity and simplicity.

Below is what it looks like when the internal structure is shown. the internal structure allows you to connect the IC without looking up the pin layout from the datasheet, making the design and connection faster.

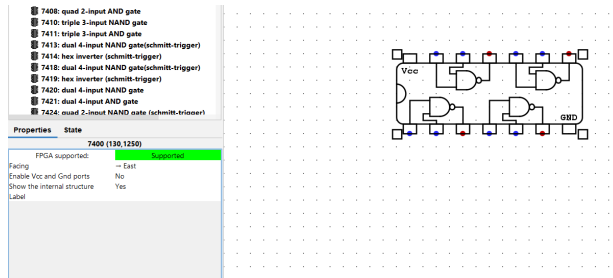


Figure 28 IC configuration with internal structure visible.

## Wiring features

The part of this copy from CS61C Lab 5.

### Splitter

Splitters allow you to take a multi-bit value and split it up into smaller parts, or (despite the name) combine multiple values that are one or more bits into a single value.

You can find it under wiring folder. Please notice that splitter is bidirectional.

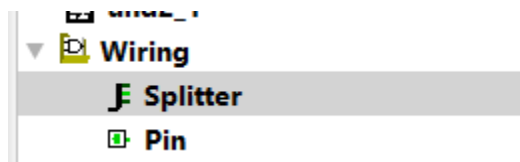
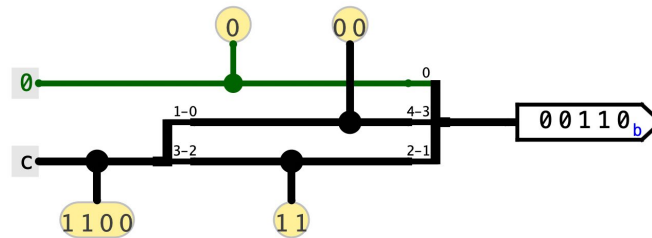


Figure 29 Location of splitter.

Here, we split the 4-bit binary number **1100** into **11** and **00**, swap their positions, and combine them with **0** to create the final 5 bit number **00110**:

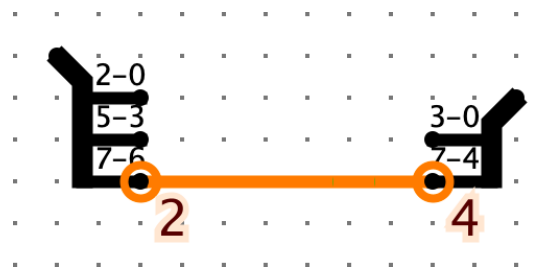


Click on a splitter (using the **Select** tool) to get its attributes in the sidebar (lower left). You can configure attributes like the number of arms on your splitter and the bits present on each arm. For the circuit above, the left and right splitters' attributes look like this:

Properties	State	Properties	State
Selection: Splitter		Selection: Splitter	
VHDL	Verilog	VHDL	Verilog
Facing	East	Facing	West
Fan Out	2	Fan Out	3
Bit Width In	4	Bit Width In	5
Appearance	Centered	Appearance	Centered
Spacing	2	Spacing	2
Bit 0	0 (Top)	Bit 0	0 (Top)
Bit 1	0 (Top)	Bit 1	2 (Bottom)
Bit 2	1 (Bottom)	Bit 2	2 (Bottom)
Bit 3	1 (Bottom)	Bit 3	1
		Bit 4	1

Notice that there's an attribute called **Facing**. You can use this to rotate your splitter. In the above example, the splitter on the right is facing **West** while the splitter on the left is facing **East**.

If you see an error wire that is orange, this means that your bit width in does not match your bit width out. Make sure that if you're connecting two components with a wire, you correctly set the bit width in that component's menu.



### Tunnel

A tunnel allows you draw an "invisible wire" to bind two points together. Tunnels are grouped by case-sensitive labels give to a wire. They are used to connect wires like so:

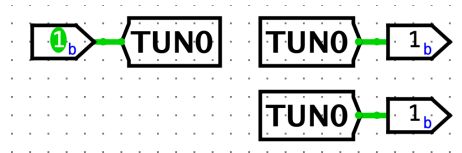


Figure 30 Tunnel example

which effectively is:

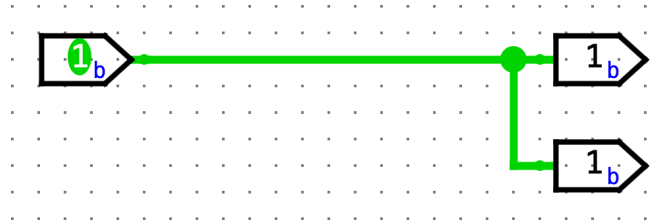


Figure 31 Equivalent tunnel connection

Here shows how you change the label of the tunnel.

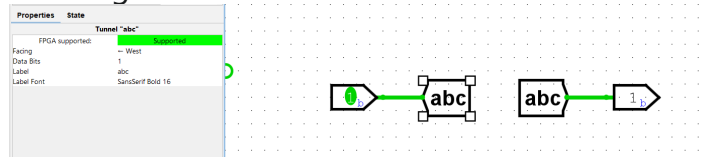
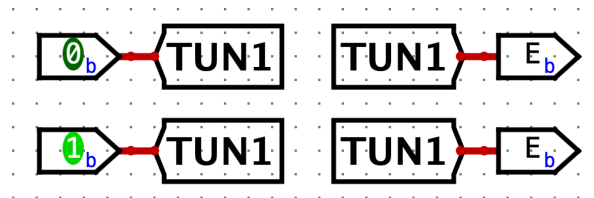
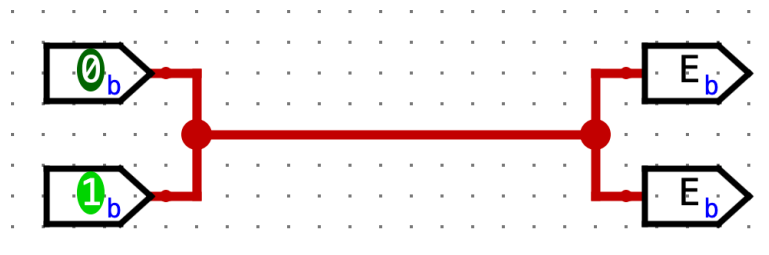


Figure 32 Change the label of tunnel.

Some care should be taken as to which wires are connected with tunnels to which other wires, such as in this case:



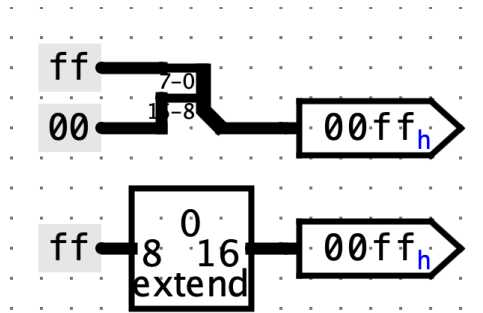
which effectively is:



We *strongly* recommend you use tunnels with Logisim, because they make your circuits look much cleaner (less wiring spaghetti), which means easier debugging.

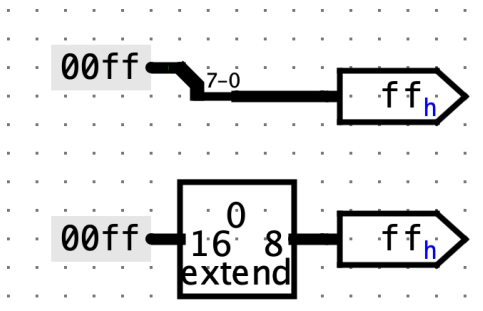
### Extenders

When changing the width of a wire, you should use a bit extender for clarity. For example, consider the following implementation of zero-extending an 8-bit wire into a 16-bit wire:



Compared to the splitter, the extender is easier to understand at a glance. This becomes especially helpful when working with complex circuits.

Additionally, consider the case of throwing out bits. Despite its name, an extender can also perform this operation:



While the splitter is more minimalistic, it's also slightly harder to read.

#### Wire connection warning

Logisim evolution gives a very interactive color representation of connections. Please refer to the graph and table below for details.

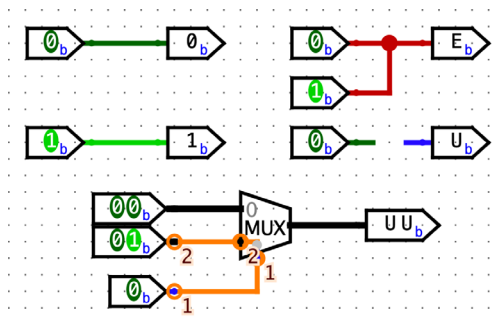


Table 1 Logisim Evolution wire color and meanings.

Color	Meaning
Dark green	1-bit wire has a value of 0
Bright green	1-bit wire has a value of 1



Color	Meaning
Black	Multi-bit wire (many components have bit width attributes which can be configured in the attributes menu on the bottom left)
Red (values with EEEE) Blue (values with UUUU)	The wire has multiple values on it (in this case, a 0 and 1 from the 2 inputs). Also, remember that a big circle appears at wire junctions.
Orange	The wire is floating (i.e. has no known value)

### Combinational Analysis:

In windows -> combinational analysis, the Logisim evolution provide excellent access for combinational logic synthesis from truth table or logic expression to actual combinational gates.

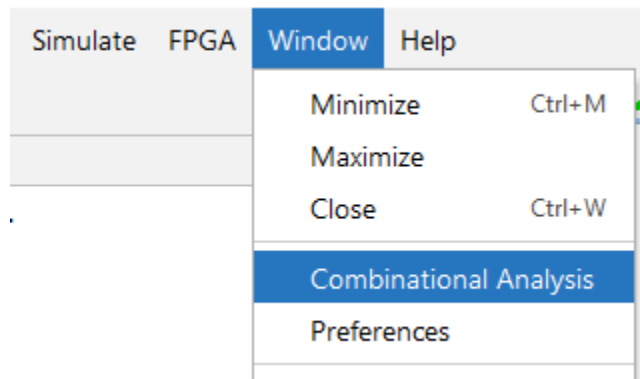


Figure 33 Location of combinational analysis

Once you type in your logic variables, you are allowed to change the truth table or expression or K-map.

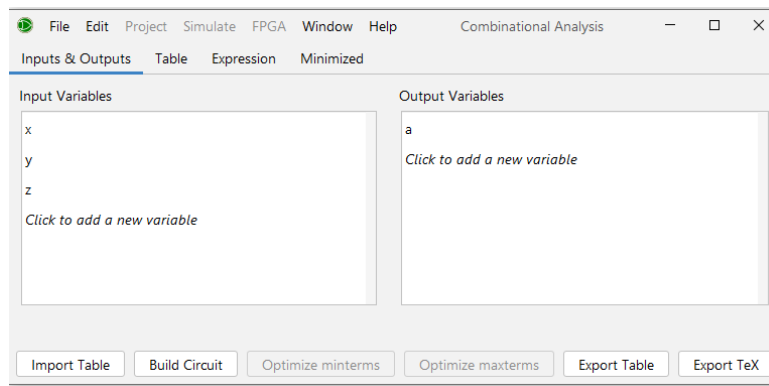
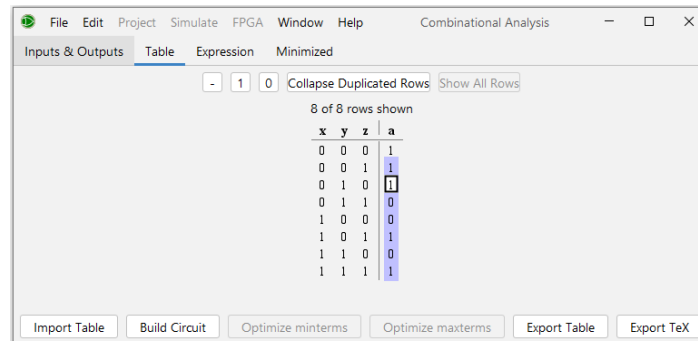


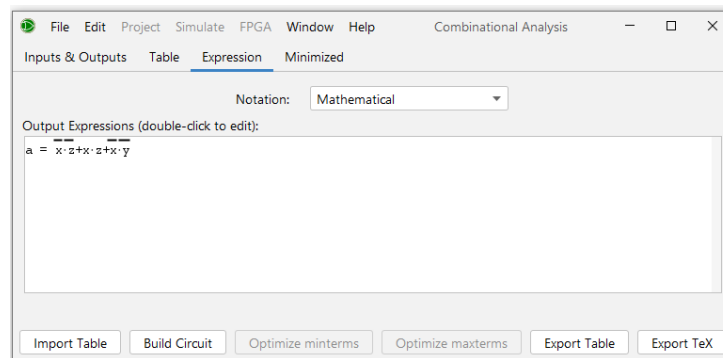
Figure 34 Way to add variables



8 of 8 rows shown

x	y	z	a
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Figure 35 Craft desired logic function.

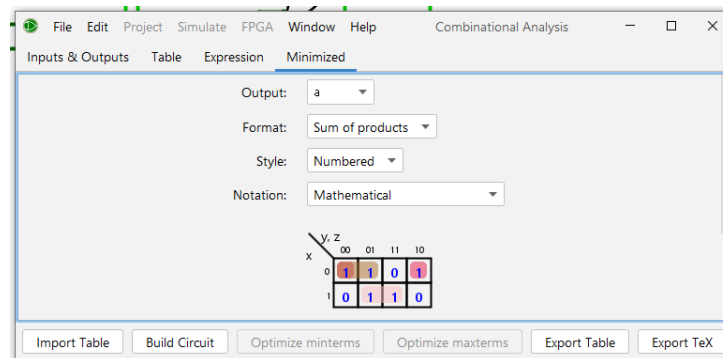


Notation: Mathematical

Output Expressions (double-click to edit):

a =  $x \cdot z + x \cdot z + x \cdot y$

Figure 36 Expression result.



Output: a

Format: Sum of products

Style: Numbered

Notation: Mathematical

	00	01	11	10
0	1	1	0	1
1	0	1	1	0

Figure 37 K-map result.

Then you click build circuit, you can then customize how your circuit is built, and name the circuit.

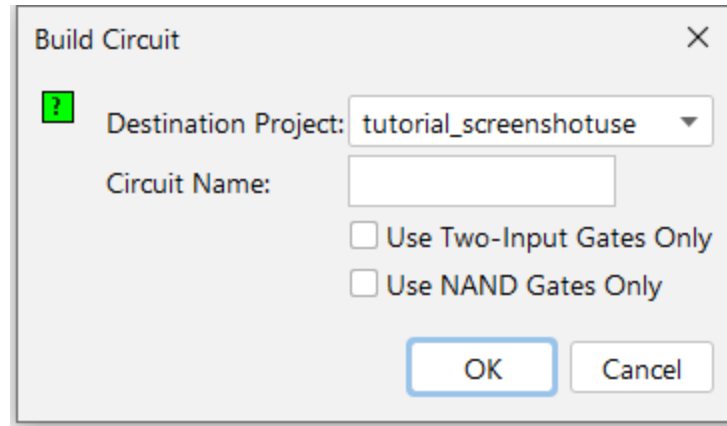
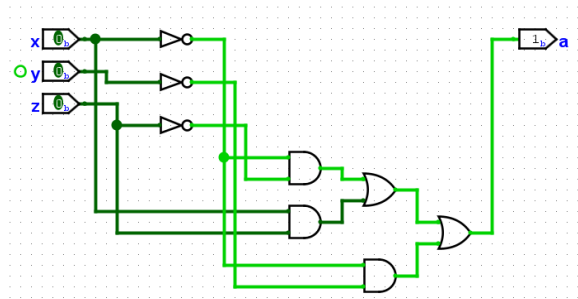


Figure 38 Extract the built digital logic.

The software gives the tool to do some of the logic designs, while you still need to learn and understand the underlying principles to pass your lecture's exam.



#### Multiple schematics in one .circ file:

For practical usage, you may need multiple digital logics saved within one .circ file to avoid multiple .circ files being saved for one lab. Also you may also want to use the macro to use it for larger designs in circuit design. Such an approach is similar to what you did in software programming using the concept of modular design by instantiating the functions (i.e., square function) to avoid writing everything in the main() function.

#### Add a new scheme

You can add new schemes by clicking the + sign here. It allows you to include multiple possible independent circuits within one circuit file, reducing the possible number of files within your lab or class folders.

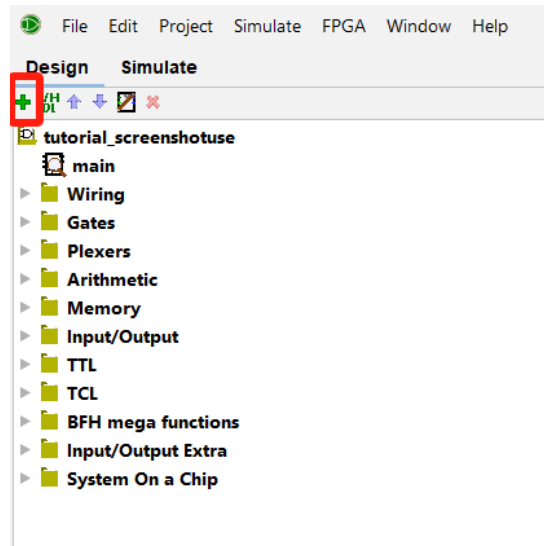


Figure 39 Location of add a new subcircuit.

Then you are asked to name the new circuit, here let's say and 2 for 2-input and gates.

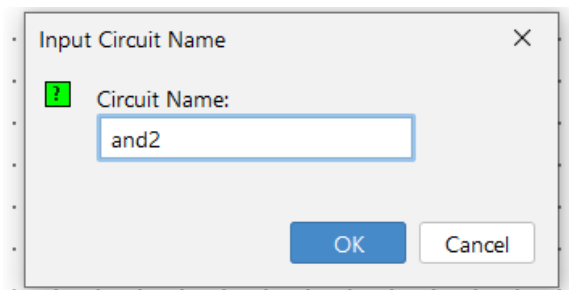


Figure 40 Name the circuit.

Then you will see there is an and2 scheme beneath the default main scheme. It is not necessary to have the top level under the "main" circuit.

Then let's have another 2-input and gate here. You have to make the inputs using input pins and outputs using output pins to allow the connectivity to and from the subcircuit.

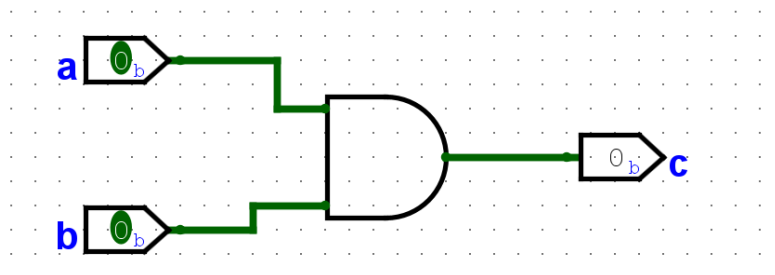
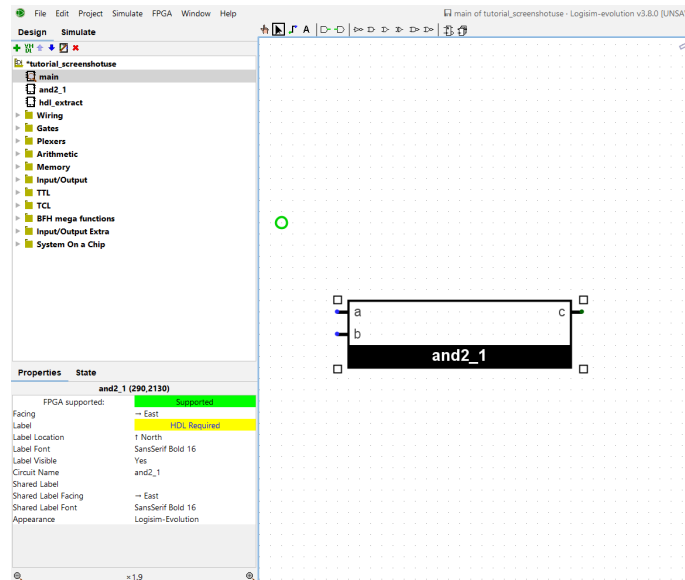


Figure 41 2 input and gate with label on IOs.

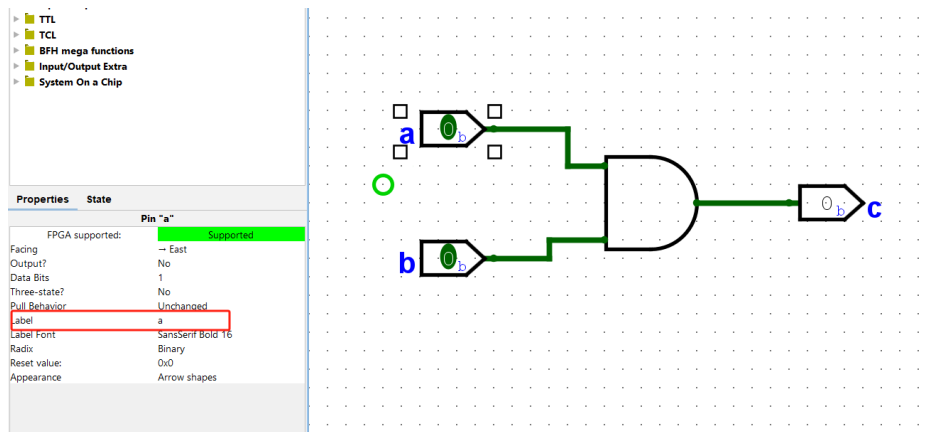
#### Subcircuit/Macro

Go back to main circuit, drag the and2\_1 to the main circuit as below.

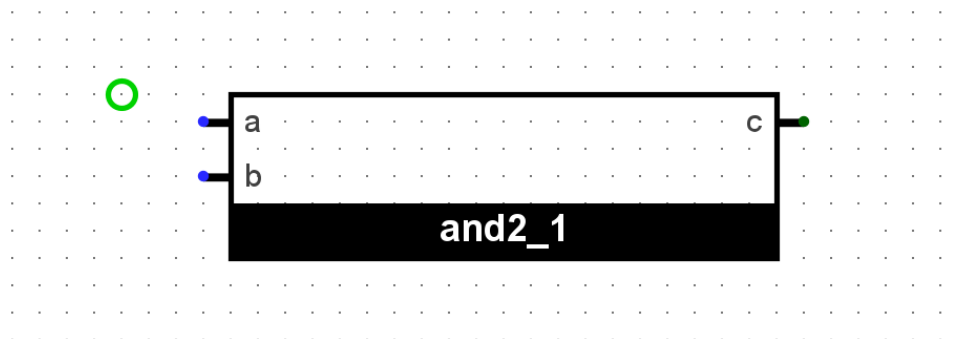


In default, the left part will be input pins, and right part will be output pins. Please make sure your input and output pins are ordered in a good manner.

It is highly recommended to edit the input pins and output pins by changing the label property as shown below.



So that there will be clear labels shown on the macro. You can then connect it with the input and output pins to verify its functionality.



You can also further rename and label the subcircuit by yourself.

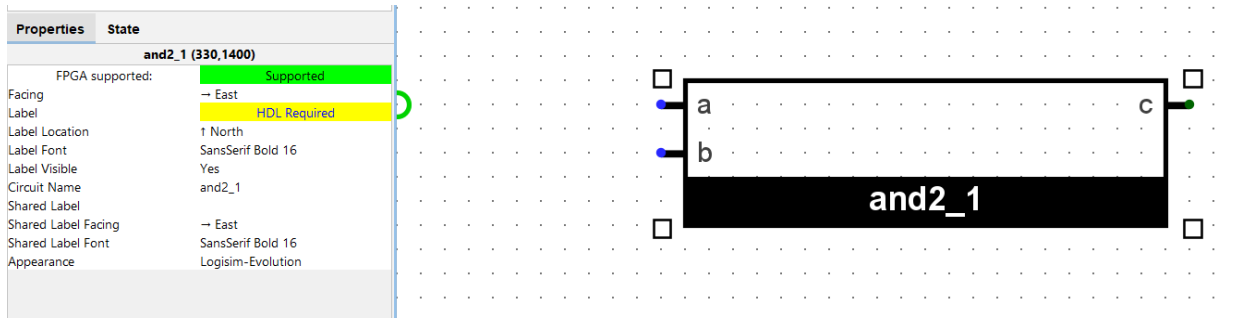


Figure 42 Pack 2-input AND gate as a macro.

## Memory

In this lab course, we have the experience of using memory to serve as cache. In the memory folder, you can see the available devices that can serve as memory.

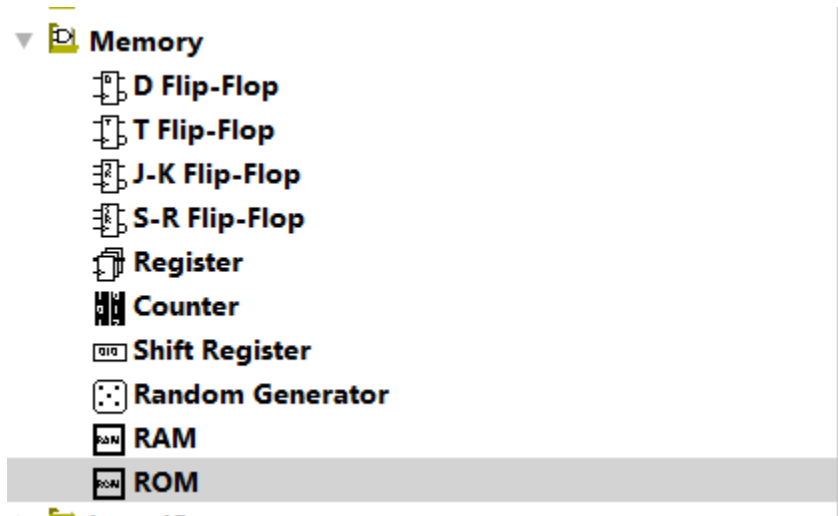


Figure 43 Location of ROM/RAM.

Here I only discuss the ROM used in the lab, and RAM is similar to ROM but with write enable, out enable, and clock pins as shown below. Note that the numbers circled in red are address of the ROM/RAM, not the data.

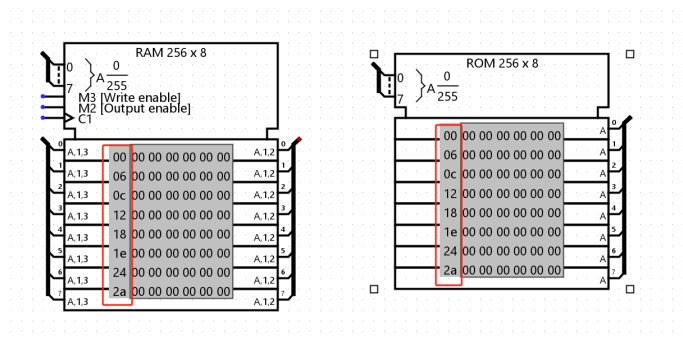


Figure 44 RAM & ROM, addresses are circled in red.

Once you have place the ROM/RAM, you can see the stored data shown in the grey table. You can configure the data bits(in default it is 8 bit) or data storage depth (in default it is 256) to the value you want by changing the address bit width and data bit width.

Properties	State
<b>ROM (370,2430)</b>	
FPGA supported:	Supported
Address Bit Width	8
Data Bit Width	8
Line size	Single
Allow misaligned?	No
Contents	(click to edit)
Label	HDL Required
Label Font	SansSerif Bold 16
Label Visible	No
Appearance	Logisim-Evolution

Figure 45 Property of ROM.

You can directly edit the content by right click the grey box or click the contents in properties. In default, the editing is in HEX.

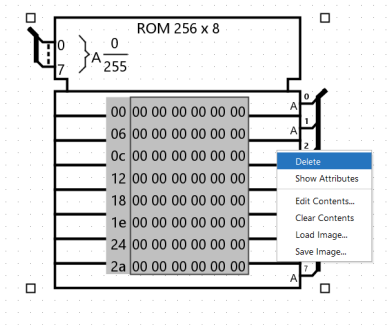


Figure 46 Edit the content of ROM.

As shown in the figure below, I change the content in address 0 to 0x0f.

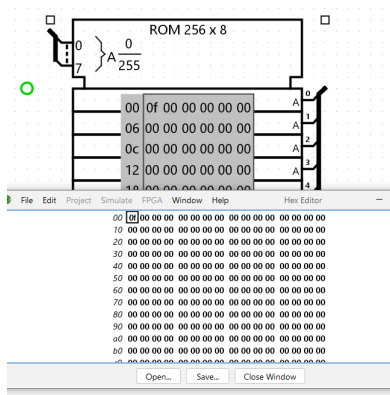


Figure 47 Editing window of ROM.

## Sequential logic

Another important category of digital logic is sequential logic. For a sequential logic, you need a clock signal connected to observe its behavior.

### Clock signal

You can find the clock signal under the wiring folder.

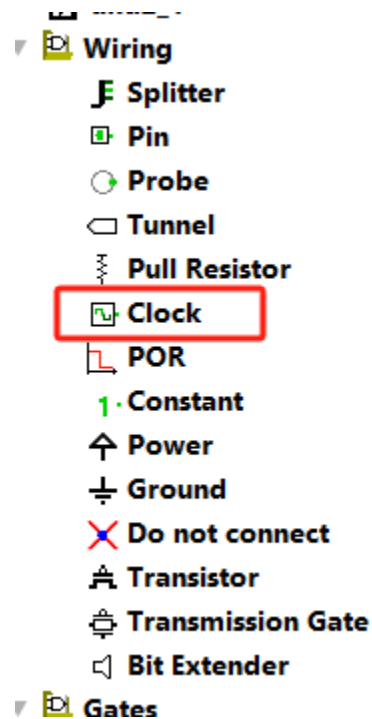


Figure 48 Location of clock signal.

Then you can connect your clock signal to a sequential logic, here shown an example for D flip flop:

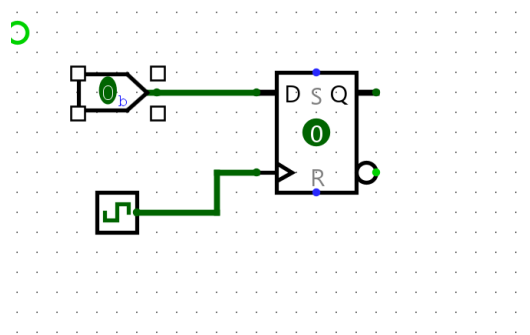


Figure 49 Example connection of clk to D flip-flop.

You can see the internal state of the D flipflop, and change it if needed.

Similarly for sequential logic chips, like 74161, once you make the internal structure visible, you can also observe the internal state and possibly change it as shown below:



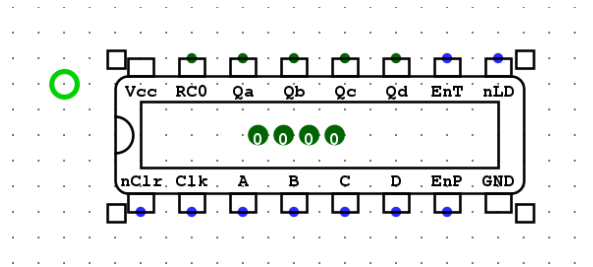


Figure 50 Interior of integrated counter.

You can toggle the clock signal as you toggle combinational pins. However, it may not be efficient for large circuits.

### Auto tick clock signal

You can then go to simulate, to set the auto-tick frequency to set the ticking frequency. Typically, around 4 Hz is the best practice. After you set the frequency and enable auto tick, you could possibly see all the clock signal in the current scheme will tickling according to the frequency.

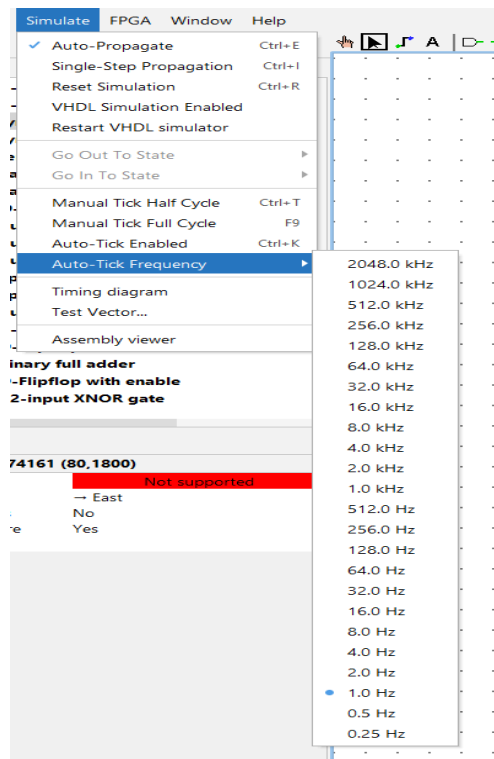


Figure 51 Change the simulation clock frequency.

You can also observe the tickling pattern from the clock signal.



Figure 52 Frequency of operating clock signal.

One of the issue of the clock signal is its icon will not change based on the orientation. Please always mindful to ensure you actually connected to the clock signal pin.

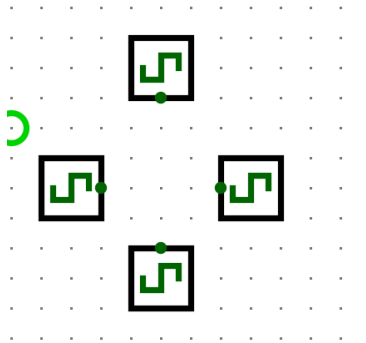


Figure 53 Clock signal with different directions.

## Digital Oscilloscope

The digital oscilloscope located under input/output extra gives you the initial taste of verifying functionality of digital waveform. Here the oscilloscope is limited to binary waveform, which could be extend to representation in decimal/hexadecimal in real EDA software.

You need to connect the oscilloscope with the clock signal, and then connect the oscilloscope with the signal you want to observe. You can also easily configure the number of inputs and number of states shown by modifying corresponding properties. It is super useful to observe the state transition for sequential digital logic. Below show case a simple connection of digital oscilloscope with integrated counter.

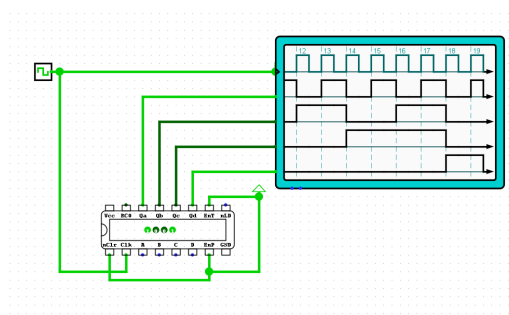
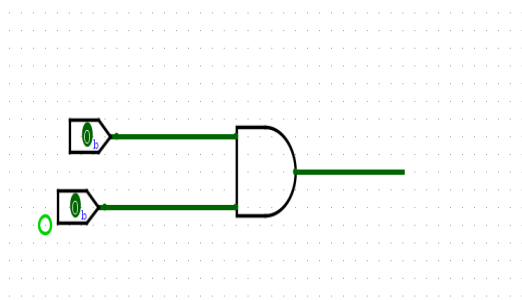


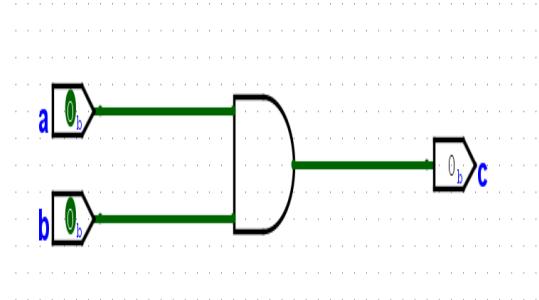
Figure 54 Digital Oscilloscope

### Common errors:

- No output/input pin provided in simulation, you need to label them to help you and others know the pin's corresponding functionality, especially when they are ordered unnaturally.

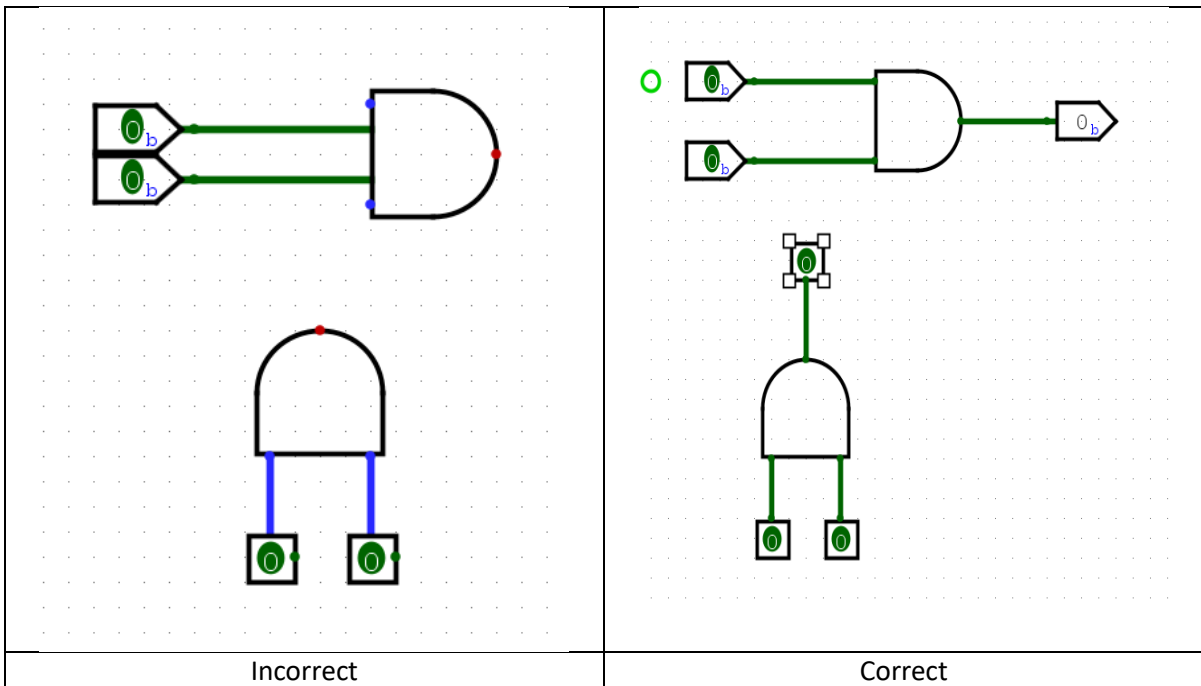


Incorrect

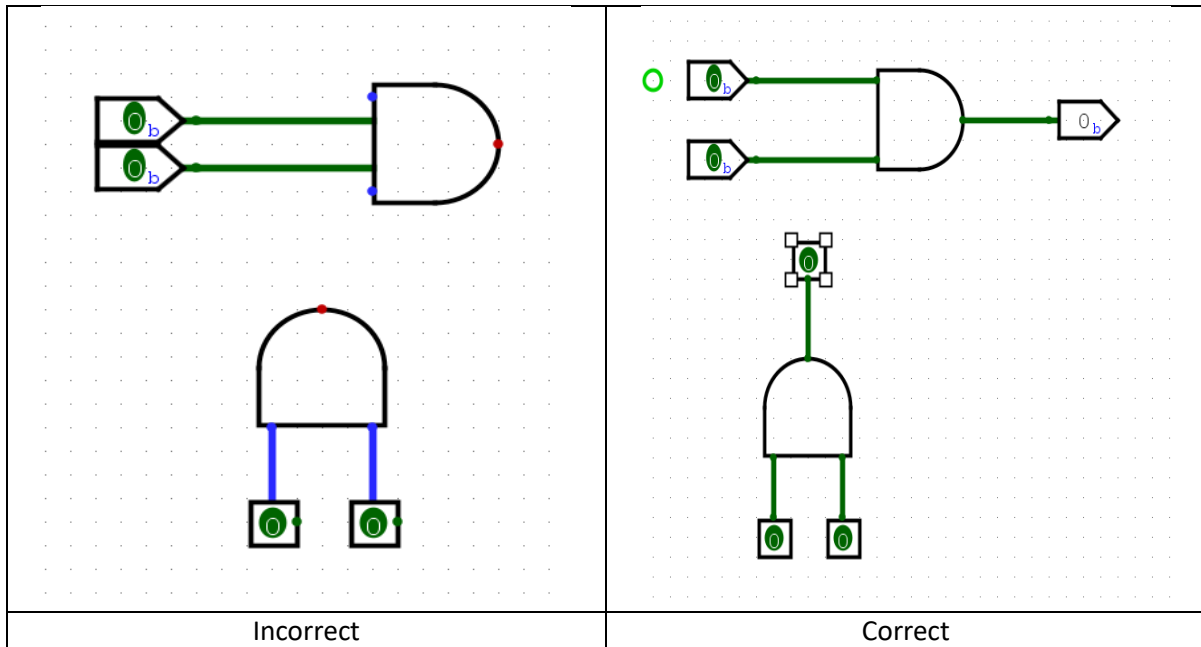


Correct

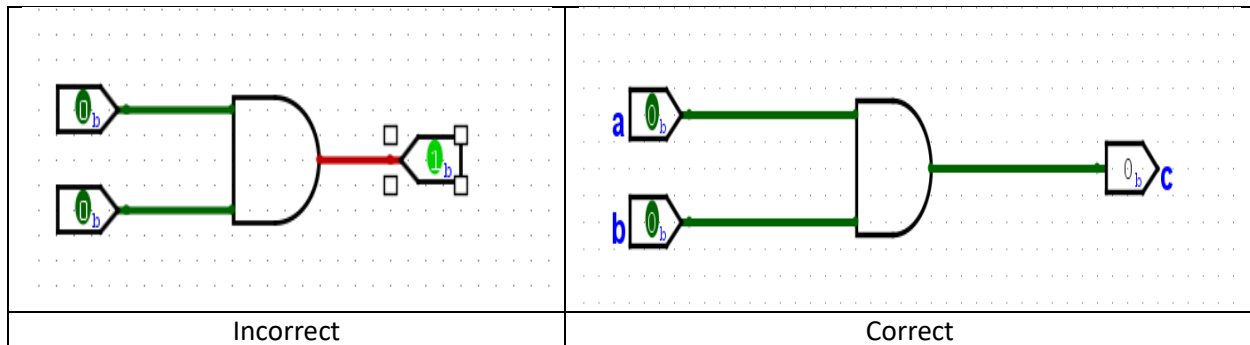
- Wire is not properly connected to the gate/desired pins.



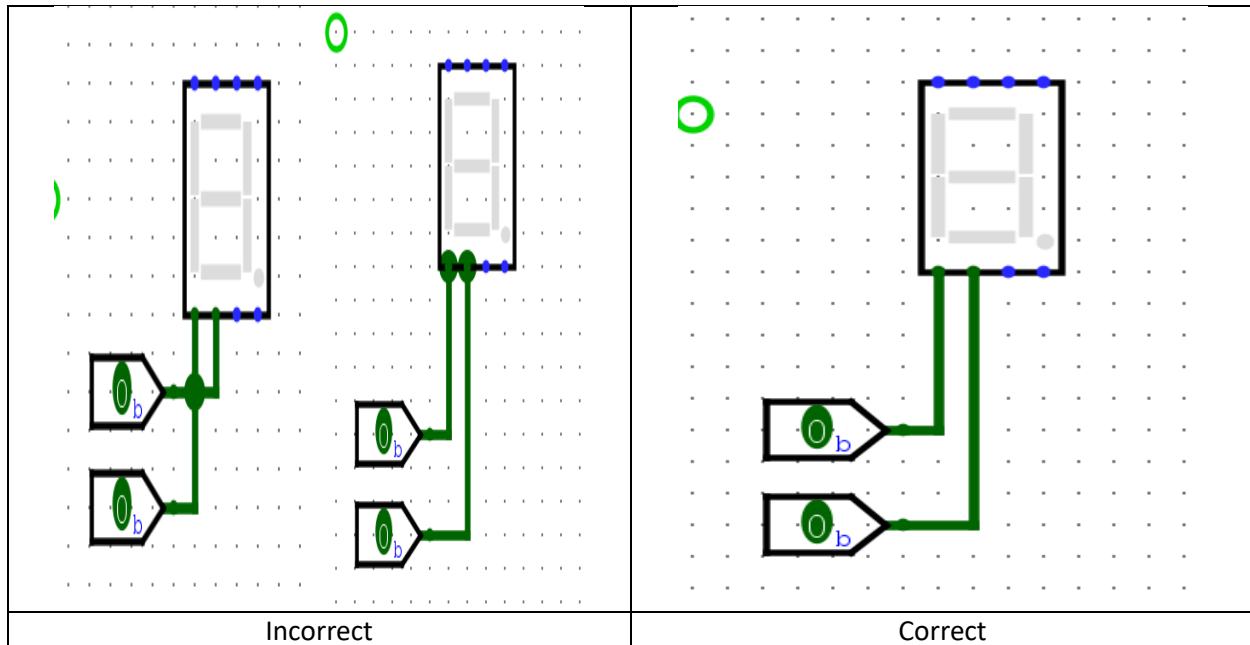
- Wrong circuit in the simulation.



Use input pins as output



Connect multiple wires together without justification



No necessary labels

