

# Online Laboratory: Ones and Zeros: Digital Combo Logic

## Description and purpose

For this project, you will be using Logisim to create a simple combinational circuit that plays a simple matching game. The circuit requires nothing but basic logic gates, input and output pins, wires, and two LED units.

Logic gates are normally realised using integrated circuits, such as the “74” series. The 7408, as an example, contains four two-input “AND” logic gates on one chip, whereby the digital logic levels of each gate can be manipulated independently by connecting HIGH or LOW signals to the input pins corresponding to each gate, and monitoring the output pins (for example, by using an LED to see if something is HIGH or LOW).

After downloading Logisim (more below), you should familiarize yourself with Logisim by working your way through the built-in tutorial (choose Help -> Tutorial from the menu).

“The Big Bang Theory” sitcom introduced a game based on the classic rock-paper-scissors game that children have played for decades. In the Big Bang Theory (BBT) version, each player chooses one of five options: rock, paper, scissors, lizard or Spock. The winner, if any, is determined by a simple set of rules that rank the “power” of each option in such a way that no single option is a guaranteed winner.

## Objectives

You will implement a simple circuit, named *RockPaperLizardSpock*, that takes inputs indicating the option chosen by each of two players, and sets two LED output components to indicate the winner, based upon those input settings and the rules given below.

## Procedure

Go to the Logisim page (<http://www.cburch.com/logisim/>) and download the software for your own platform (Windows, Mac, Linux, etc).

Then, download this Logisim skeleton file, and save it to your computer: <https://drive.google.com/file/d/15r37Rww-LdQQ9bq85rzo3LPV8tvVkCi3/view?usp=sharing> (when you click on the link you’ll see a down arrow ⤓ top right)

# Rock-Paper-Lizard-Spock Game Circuit

In order to simplify this online laboratory, we will modify the BBT version by eliminating “scissors” as an option (we pretend to not trust scissors to geeks).

The winner of our simplified game is determined according to the [following rules](https://bigbangtheory.fandom.com/wiki/Rock,_Paper,_Scissors,_Lizard,_Spock):

* **rock vs paper** paper covers rock, so paper wins
* **rock vs lizard** rock crushes lizard, so rock wins
* **rock vs Spock** Spock vaporizes rock, so Spock wins
* **paper vs lizard** lizard eats paper, so lizard wins
* **paper vs Spock** paper disproves Spock, so paper wins
* **lizard vs Spock** lizard poisons Spock, so lizard wins

Note that these are commutative (can be turned around); that is, “rock vs paper” is equivalent to “paper vs rock” and so forth. Other combinations, such as rock vs rock, produce no winner.

You will implement a simple circuit, named *RockPaperLizardSpock*, that takes inputs indicating the option chosen by each of two players, and sets two LED output components to indicate the winner, based upon those input settings and the rules given above.

When designing and implementing a circuit like this, where the input values are not naturally viewed as integers (whole numbers), one of the first steps is to decide how to represent the input and output data. In this case, each player must make a choice among four options, so it is natural to represent the choices (in hardware) as two-bit integers (0, 1, 2, and 3). In order to make our marking simpler, we will mandate that you represent the input values using the following mappings:

|  |  |
| --- | --- |
| **Choice** | **Representation** |
| rock | 00 |
| paper | 01 |
| lizard | 10 |
| Spock | 11 |

Therefore, the circuit will take two 2-bit inputs (so 4 input bits altogether for the two players).

As for output, there are three cases: player A wins, player B wins, and no one wins (draw). We will represent those outcomes by 00, 11, and 01, respectively. (There is a reason we defined these in that manner; it actually simplifies the analysis to produce a Boolean function for each output bit.)

For the output to be visually attractive, the circuit must be able to clearly indicate which of the two players has won, or that neither has won. A simple way to do that is to employ an LED component for each player, and to turn on a player’s LED if and only if that player has won. The LED outputs should be set so that the winner, if any, is indicated by a green LED, and non-winners are indicated by red. Note that the LED components must be controlled by using the two output bits set by the circuit, and that both LEDs should be red if there is no winner.

There will be two deliverables for the assignment. The **first deliverable is a Word or PDF file containing a completed truth table corresponding to the operation of the circuit, the Boolean functions for each of the two output bits (using the Sum of Products method), and simplified versions of those functions (including the Karnaugh maps used for the simplification)**.

Please copy this table in to the top of the file, and fill it in with your own details:

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name:** |  | | |
| **ID Number:** |  | | |
| **Degree Name:** |  | | |
| **Assessment:** | EI140 EEE-CA-4 | | |

To make grading easier, you will structure the truth table as below (not all rows are shown). Here, A1 and A0 form the option chosen by player A, B1 and B0 the option chosen by player B, and W1 and W0 the output bits indicating the winner (if any). Note the rows of the table are ordered so that the 4 input bits “count” from 0000 to 1111.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A1** | **A0** | **B1** | **B0** |  | **W1** | **W0** |
| 0 | 0 | 0 | 0 |  | ? | ? |
| 0 | 0 | 0 | 1 |  | ? | ? |
| 0 | 0 | 1 | 0 |  | ? | ? |
| ... | ... | ... | ... |  | ... | ... |
| 1 | 1 | 0 | 1 |  | ? | ? |
| 1 | 1 | 1 | 0 |  | ? | ? |
| 1 | 1 | 1 | 1 |  | ? | ? |

The **second deliverable is a single Logisim file containing your implementation of the circuit**. You can use the skeleton .circ file downloaded earlier as your starting point. Your main circuit must be named *RockPaperLizardSpock*, and must include the following interface elements:

* at the top: four 1-bit inputs, labelled A1, A0, B1, and B0, from left to right
* at the bottom: two 1-bit outputs, labelled W1 and W0, from left to right
* on the right side: two LED components, labeled A Wins and B Wins, from top to bottom

The layout and labelling of the input and output pins is important, because that determines the way Logisim will generate a truth table for your circuit, and we may use that truth table in our marking of these assignments.

You are encouraged but not forced to create sub-circuits in your implementation, since that makes it possible to create a much simpler abstract view of the final design. In my case, I have implemented two sub-circuit blocks that you will see when you download the skeleton file: RPLS takes the options chosen by players A and B and computes the output bits W1 and W0. LEDControl takes W1 and W0 as input and sets the correct input signals for the two LED components. (I actually have two more sub-circuits that are not shown. Those perform computations that are used inside RPLS.)

You are not required to achieve a design that uses a minimal number of gate levels; in some cases, it is better to have more levels in order to create a more modular design.

# What to submit

You submit BOTH your Word or PDF file named *YOURSTUDENTID.docx* or *.pdf* with the truth table/Boolean functions/simplified functions plus Karnaugh maps, AND your Logisim *YOURSTUDENTID.circ* file, to the Blackboard area for EI140 -> Assessment -> EEE -> EEE-CA-4 Online Laboratory Submission.