What is Digital Logic? Why's it important?

What is digital?
Electronic circuits can be divided into two categories: analog and digital. Analog signals can take any shape and be an infinite number of possible values. Digital signals have a very defined, discrete set of possible values; usually only TWO possible values. Many electronic systems will use a combination of analog and digital circuits, but at the heart of most computers and other everyday electronics are discrete, digital circuits.

What is digital logic?
We humans are (mostly) logical beings. When we make decisions and act upon them, logic is working in the background. Logic is the evaluating of one or more inputs, weighing them against any number of outcomes, and deciding a path to follow. Just like we apply logic to make all of our decisions, computers use digital logic circuits to make theirs.

Where do we see it?
There is logic in all circuits. A connection to a power source is an example of an AND. Only if both power AND ground are connected will the circuit allow electricity to run through it. Where else? Everywhere! The traffic light in a city uses logic when pedestrians push the walk button. The button starts a process that uses an AND logic gate. If someone pressed the walk button AND there is a red light for traffic where the pedestrian will be walking, the walk signal will activate. If both conditions of this AND statement are not true, the walk signal will never be illuminated.

Inputs and Outputs
A digital logic circuit uses digital inputs to make logical decisions, which produces digital outputs. Every logic circuit requires at least one input, before it can produce any kind of output.

Digital logic inputs and outputs are usually **binary**. In other words they can only be one of two possible values. There are a number of ways to represent binary input/output values: 1/0 is the least verbose and most common way. However, you may also see them in a boolean representation like TRUE/FALSE or HIGH/LOW. When the values are represented at a hardware level, they might be given actual voltage levels: 0V [Volts] is a 0, while a higher voltage - usually 3V or 5V - is used to represent a 1.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td>5V (volts)</td>
<td>0V</td>
</tr>
</tbody>
</table>
Logic Gates

To make their logical decisions, a computer will use any combination of three fundamental logic functions: AND, OR, and NOT.

- **AND** - The AND function produces a TRUE output if and only if all of its inputs are also TRUE.
- **OR** - An OR will prove TRUE if any (one or more) of its inputs are also TRUE.
- **NOT** - The NOT operator has only one input. The NOT operator **negates** its input, which means the output will be the opposite of the input.

Each of those functions can be realized using logic gates. Logic gates are what we use to create digital logic circuits. They're the building blocks of computers and other electronics. They take one or more inputs, perform a specific function (AND, OR, NOT, etc.) on them, and then produce an output based on that function.

Logic gates all have specific circuit symbols, just like resistors, capacitors, and inductors. And, just like standard electronic component symbols, logic gate can be diagrammed in a schematic by connecting their input and output lines together.

![Logic Gates Diagram](image)

Truth Tables

The 'logic' of a logic gate or function can be represented in a number of ways including truth tables, venn diagrams and boolean algebra. Among those, truth tables are the most common. Truth tables are a complete listing of every possible input combination, and the output they produce.

All inputs are arranged on the left side of the table, while the output is on the right. Every row of the table represents one possible input combination.

For example, an AND gate has two inputs and one output, so the truth table should have two columns for input and one column for output. Two input values means there are four possible input combinations: 0/0, 0/1, 1/0, and 1/1. So, the AND gate's truth table will have four rows, one for each combination of inputs:

<table>
<thead>
<tr>
<th>Input A</th>
<th>Input B</th>
<th>Output Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
LogicBlocks Fundamentals

There are six separate LogicBlocks, which can be divided into three categories: logic gates, input blocks and utility blocks.

Logic Gate Blocks
There are three different logic gate LogicBlocks: AND, OR, and NOT - the fundamental logic gates. They're shaped (as much as possible) like their circuit diagram equivalents, and also labeled with their gate name.

![Logic Gate Blocks Diagram](image)

Each gate has one output, a male header (the pointy one). The female headers represent the inputs; the AND and OR gates have two inputs, while the NOT gate has just one.

The logic gate LogicBlocks each have an LED, which represents their output status. An illuminated LED represents a logic gate producing a 1, while an unlit LED means the gate is producing a 0.

Input Blocks
The input blocks are what you'll use to supply LogicBlocks with digital inputs. They have a switch to set the output of the Input block to either 0 or 1. The LED on the Input block displays what value the block is sending out.

![Input Block](image)

The input blocks have a single, male, output header. This should be plugged into the female inputs of the gate LogicBlocks.

Utility Blocks (and cables)
Key among the utility boards is the Power block. Each Logic gate requires power to operate, and this board is there to supply it. The Power block has a single, female, input header, and should be plugged into the output of the last logic gate. You'll only need ONE power block to supply power to an entire digital logic circuit.

![Power Block](image)

There's also a Splitter LogicBlock, which simply divides one input into two outputs. It doesn't perform any operation on the signal, just passes it through. This'll come in handy when you create more complicated digital logic circuits which require one input to run into two or more logic gates.

Also lumped into this category is the feedback cable. This cable will usually be used in conjunction with the splitter block, for more advanced LogicBlock circuits.
The Rules
There's really only one rule when it comes to playing with LogicBlocks: When connecting an output of one block to the input of another, make sure each of the three pins match up. Each of the inputs and outputs consist of three pins: Power (“+”), Ground (“-“), and Signal (“->”).

You'll also need to make sure there's always one (and only one) power block connected, somehow, to your LogicBlock circuit.

All inputs of a logic block (or cable) should have the output of another block (or cable) attached to them. If an input is left floating (not connected), the gate won't know if the input is a 1 or a 0, and, as a result, the output will be unreliable.

Response times
Logic gates form their output decisions faster than any human could ever detect (we're talking nanoseconds). So to make the process more visible, we've slowed the LogicBlocks down so you can see the process they go through. If you've got a long line of logic blocks, this delay will allow you to more visually see how the outcome of one block affects the input of another.

The delay is on the order of half a second. Just long enough to see what's going on.
The Blocks In-depth

Input LogicBlock

The Input Blocks make the LogicBlocks world go 'round. These small rectangular blocks have one male connector, which plugs into the female input pins of logic gate blocks.

Each Input Block has a two-position switch, which you'll use to set the block as either 1 or 0 (TRUE or FALSE, ON or OFF).

A green LED on the Input Block represents the value it outputs. An illuminated LED indicates that the block is set as 1. If the LED is off, the output value is 0.

AND Gate LogicBlock

The AND LogicBlock is a two-input, single-output AND gate. The block is in a D shape, much like the AND gate circuit symbol. The two female input headers are located on either side of the AND Block. You can connect either an Input Block or the output from another Gate Block to these inputs. The output of the AND Block is located at the middle-top. This male header can be plugged into either the input of another gate or a Power Block.

Each AND Gate Block has a single blue LED, which represents the output of the gate. If the LED is on, then that means the AND Gate is producing a 1 (TRUE, ON) at the output. If the LED is off, then the AND Gate's output is 0.

Here's a truth table for the AND Block:

<table>
<thead>
<tr>
<th>Input A</th>
<th>Input B</th>
<th>Output Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

And the circuit symbol for an AND gate, which you'll get very used to seeing:

```
A   Y
\_\_\_
   B
```

The boolean equation symbol for AND is the centered dot (·). For example, the gate above could be represented by the equation: \( A \cdot B = Y \)
OR Gate LogicBlock

The OR LogicBlock is a two-input, single-output OR gate. The block is shaped like and OR gate circuit symbol - a convex arc on the output side, and a concave arc on the input side. The two female inputs are located on either side of the OR Block. You can connect either an Input Block or the output from another Gate Block to these inputs. The output of the OR Block is located at the middle-top of the Block. This male connector can be plugged into either the input of another gate, or a Power Block.

Each OR Gate Block has a single yellow LED, which represents the output of the gate. If the LED is on, that means the OR Gate is producing a 1 (TRUE, ON) at the output. If the LED is off, then the OR Gate's output is 0.

Here's the truth table for a 2-input/1-output OR gate:

<table>
<thead>
<tr>
<th>Input A</th>
<th>Input B</th>
<th>Output Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

And the circuit symbol:

![OR Gate Circuit Symbol]

We can represent digital logic using boolean equations. The boolean equation symbol for OR is the plus sign (+). Here's the boolean equation for the OR circuit above: \( A + B = Y \)

NOT Gate LogicBlock

The NOT LogicBlock is a single-input, single-output NOT Gate. The Block comes in a trapezoid shape (it's as close as we could get to the triangle-shaped NOT gate circuit symbol). The female input connector is located on the side with the larger edge. You can connect either an Input Block or the output from another Gate Block to this input. The output of the NOT Block is located on the smaller edge. This male connector can be plugged into either the input of another gate or a Power Block.

Each NOT Block has a single red LED, which represents the output of the gate. If the LED is on, then that means the NOT Gate is producing a 1 (TRUE, ON) at the output. IF the LED is off, then the NOT gate's output is 0.
A NOT gate is often called an inverter because it inverts any signal it receives. A 0 turns into a 1, and a 1 turns into a 0. An inverter's truth table looks something like this:

<table>
<thead>
<tr>
<th>Input A</th>
<th>Output Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The circuit symbol for a NOT gate is an isosceles triangle with a circle at the apex:

When writing boolean equations, the NOT operation is usually indicated by a bar over any variables it inverts. For example the boolean equation for the circuit above would simply be:

\[ \overline{A} = Y \]

**Power LogicBlock**

No electronic circuits can work without some form of power. The LogicBlocks can work at voltages between 2 and 5V, and they consume very little current.

The Power Block runs on a single 20mm coin cell battery, which optimally provides around 3V, and should last for quite a while.

The Power Block has a single female connector, which should be connected to the output of a Gate LogicBlock. Only one Power Block should be connected in any circuit.

The Power Block is the final piece to any LogicBlock circuit. It should be connected to the output of the very last Logic Block gate.

**Splitter LogicBlock and Feedback Cable**

The Splitter Block allows the output of one Block to feed into the inputs of two separate Blocks.

For more advanced circuits you'll want to spread inputs to multiple gates, or add feedback. This is where both the Splitter Block and Feedback Cable come in handy. Often the Feedback Cable will be connected to one of the two Splitter Block outputs.

The Feedback Cable is made of three separate wires - colored red, yellow, and black. Red should always be connected to the + pin, black to the - pin, yellow will pass the signal through.