Introduction to Logic Gates

Introduction

Logic gates are used when designing logical circuits for machines. Logic gates usually take in two inputs, which can be represented with 1 and 0, true and false, or on and off. While two inputs are traditional, some diagrams will draw gates with multiple inputs to save space. In reality this would be done with multiple copies of the gate. Logic gates always have a single output give in the same format as their inputs. Logic gates can be broken down into three categories: basic gates, negative gates, and exclusive gates.

Logic gates are used to perform various calculations and run through computations. They may seem simple but when used together they can create things like calculators and computer memory. Each logic gate has a visual representation, and using these representations, larger scale diagrams can be made.

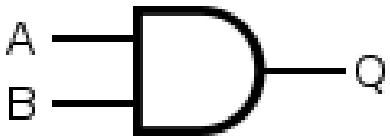
Basic Gates

There are two main basic gates, the AND gate and the OR gate. These two gates, along with NOT gates which will be explained later, can make any other gate explained in this guide.

AND Gate

The AND gate takes two inputs and returns true when both inputs are true. The primary use of this gate is to make sure multiple conditions are met. In essence, if this gate were a door, it would not open unless you have all required keys.

The AND gate's diagram and output is shown below:



A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate

The OR gate takes two inputs and returns true when an input is true. The primary use of this gate is to make sure you meet at least one requirement. In essence, if this gate were a door, any of its keys would open it.

The OR gate's diagram and output is shown below:

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

Negative Gates

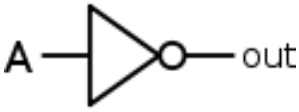
All negative gates follow the basic idea of a not gate, which is "reverse the output." There are four gates that fall under the category of negative gates, the NOT, NAND, NOR, and NXOR. When drawing diagrams for negative gates, a circle is used to indicate the negation.

NOT Gate

The NOT gate is the only gate to not follow the two input one output rule. This gate takes one input and then reverses it.

The NOT gate's diagram and output is shown below:

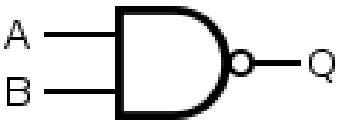
NAND Gate



A	OUT
0	1
1	0

The NAND gate is the inverse of the AND gate. It is simply comprised of an AND gate with a NOT gate at the end, however it is easier and faster to represent it as its own entity. This gate works very similarly to the AND gate, however it is used to make sure that at least one of multiple conditions is not met.

The NAND gate's diagram and output is shown below:



A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

The NOR gate is the inverse of the OR gate. As one would predict, it is made by placing a NOT gate at the end of an OR gate. This gate works similarly to the OR gate, however this gate makes sure absolutely none of the conditions are met.

The NOR gate's diagram and output is shown below:

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

Exclusive Gates

Exclusive is an adjective that can be applied to the OR and NOR gates to make XOR and XNOR. Exclusive or works more like the English definition of or. With computers, OR says "Is this or that true?" and responds with "yes" if at least one is true. XOR, on the other hand asks "Is only this or only that true?"

XOR Gate

This means that the XOR gate returns true when only one input is true. An easy way to think about this would be as a light switch. If there are two light switches, and one is in the on position, the light should be on. However if you flip the switch for the other light switch, you still expect the light to turn off. An XOR gate makes that happen.

There is no simple textual representation for an XOR gate. This is because, as mentioned earlier, it can be made using AND, OR, and NOT gates.

The XOR gate's diagram and output is shown below:

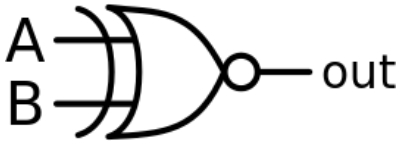
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

NXOR Gate

The XNOR gate, while seeming like an odd concept, is actually very useful in circuits. Since the XOR gate return true when the inputs are different, one true and one false, the XNOR gate returns true when the inputs match. This means that XNOR is unique in that it does not matter whether the input is true or false, so long as both inputs match.

There is no simple textual representation for an XNOR gate. This is because, as mentioned earlier, it can be made using AND, OR, and NOT gates.

The XOR gate's diagram and output is shown below:



A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1