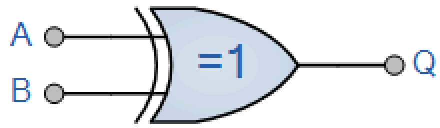


Exclusive-OR Gate Tutorial

The Exclusive-OR logic function is a very useful circuit that can be used in many different types of computational circuits

In the previous tutorials, we saw that by using the three principal gates, the AND Gate, the OR Gate and the NOT Gate, we can build many other types of logic gate functions, such as a NAND Gate and a NOR Gate or any other type of digital logic function we can imagine.

But there are two other types of digital logic gates which although they are not a basic gate in their own right as they are constructed by combining together other logic gates, their output Boolean function is important enough to be considered as complete logic gates. These two "hybrid" logic gates are called the **Exclusive-OR (Ex-OR) Gate** and its complement the **Exclusive-NOR (Ex-NOR) Gate**.

Previously, we saw that for a 2-input OR gate, if A = "1", **OR** B = "1", **OR BOTH** A + B = "1" then the output from the digital gate must also be at a logic level "1" and because of this, this type of logic gate is known as an Inclusive-OR function. The logic gate gets its name from the fact that it *includes* the case of Q = "1" when both A and B = "1".

If however, an logic output "1" is obtained when **ONLY** A = "1" or when **ONLY** B = "1" but **NOT** both together at the same time, giving the binary inputs of "01" or "10", then the output will be "1". This type of gate is known as an Exclusive-OR function or more commonly an Ex-Or function for short. This is because its boolean expression *excludes* the "**OR BOTH**" case of Q = "1" when both A and B = "1".

In other words the output of an Exclusive-OR gate **ONLY** goes "HIGH" when its two input terminals are at "**DIFFERENT**" logic levels with respect to each other.

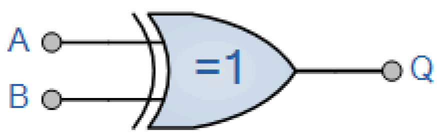
An odd number of logic "1's" on its inputs gives a logic "1" at the output. These two inputs can be at logic level "1" or at logic level "0" giving us the Boolean expression of: $Q = (A \oplus B) = \overline{A}.B + A.\overline{B}$

The **Exclusive-OR Gate** function, or **Ex-OR** for short, is achieved by combining standard logic gates together to form more complex gate functions that are used extensively in building arithmetic logic circuits, computational logic comparators and error detection circuits.

The two-input "Exclusive-OR" gate is basically a modulo two adder, since it gives the sum of two binary numbers and as a result are more complex in design than other basic types of logic gate. The truth table, logic symbol and implementation of a 2-input Exclusive-OR gate is shown below.

The Digital Logic "Exclusive-OR" Gate

2-input Ex-OR Gate

<table>
<tr><th>Symbol</th><th colspan="3">Truth Table</th></tr>
<tr><td rowspan="5">

2-input Ex-OR Gate</td><td>B</td><td>A</td><td>Q</td></tr>
<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>0</td></tr>
<tr><td>Boolean Expression $Q = A \oplus B$</td><td colspan="3">A OR B but NOT BOTH gives Q</td></tr>
</table>

Giving the Boolean expression of: $Q = A\overline{B} + \overline{A}B$

The truth table above shows that the output of an Exclusive-OR gate ONLY goes "HIGH" when both of its two input terminals are at "DIFFERENT" logic levels with respect to each other. If these two inputs, A and B are both at logic level "1" or both at logic level "0" the output is a "0" making the gate an "odd but not the even gate". In other words, the output is "1" when there are an odd number of 1's in the inputs.

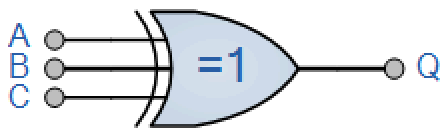
This ability of the *Exclusive-OR gate* to compare two logic levels and produce an output value dependent upon the input condition is very useful in computational logic circuits as it gives us the following Boolean expression of:

$$Q = (A \oplus B) = \overline{A}.B + A.\overline{B}$$

The logic function implemented by a 2-input Ex-OR is given as either: "A OR B but NOT both" will give an output at Q. In general, an Ex-OR gate will give an output value of logic "1" ONLY when there are an **ODD** number of 1's on the inputs to the gate, if the two numbers are equal, the output is "0".

Then an Ex-OR function with more than two inputs is called an "odd function" or modulo-2-sum (Mod-2-SUM), not an Ex-OR. This description can be expanded to apply to any number of individual inputs as shown below for a 3-input Ex-OR gate.

3-input Ex-OR Gate

<table>
<tr><th>Symbol</th><th colspan="4">Truth Table</th></tr>
<tr><td rowspan="9">

3-input Ex-OR Gate</td><td>C</td><td>B</td><td>A</td><td>Q</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>Boolean Expression $Q = A \oplus B \oplus C$</td><td colspan="4">"Any ODD Number of Inputs" gives Q</td></tr>
</table>

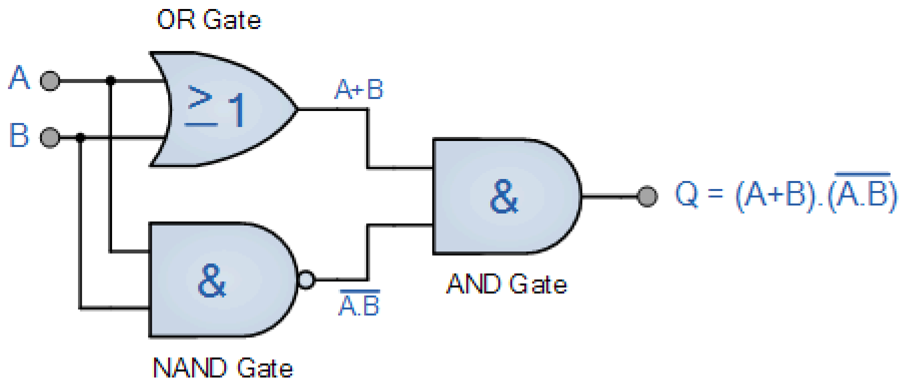
Giving the Boolean expression of: $Q = A\overline{B}\,\overline{C} + \overline{A}B\overline{C} + \overline{A}\,\overline{B}C + ABC$

The symbol used to denote an Exclusive-OR odd function is slightly different to that for the standard Inclusive-OR Gate. The logic or Boolean expression given for a logic OR gate is that of logical addition which is denoted by a standard plus sign.

The symbol used to describe the Boolean expression for an **Exclusive-OR** function is a plus sign, (+) within a circle (O). This exclusive-OR symbol also represents the mathematical "direct sum of sub-objects" expression, with the resulting symbol for an *Exclusive-OR* function being given as: ($\oplus$).

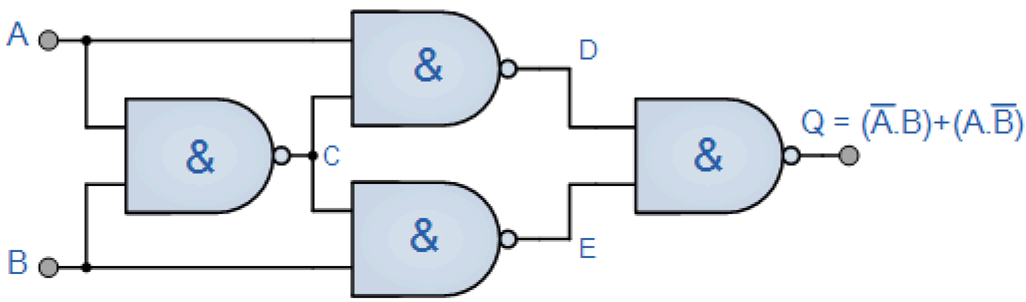
We said previously that the Ex-OR function is not a basic logic gate but a combination of different logic gates connected together. Using the 2-input truth table above, we can expand the Ex-OR function to: $(A+B).(\overline{A.B})$ which means that we can realise this new expression using the following individual gates.

Ex-OR Gate Equivalent Circuit



One of the main disadvantages of implementing the Ex-OR function above is that it contains three different types logic gates OR, NAND and finally AND within its design. One easier way of producing the Ex-OR function from a single gate is to use our old favourite the NAND gate as shown below.

Ex-OR Function Realisation using NAND gates



NAND Gate Realisation

Exclusive-OR Gates are used mainly to build circuits that perform arithmetic operations and calculations especially **Adders** and **Half-Adders** as they can provide a "carry-bit" function or as a controlled inverter, where one input passes the binary data and the other input is supplied with a control signal.

Commonly available digital logic Exclusive-OR gate IC's include:

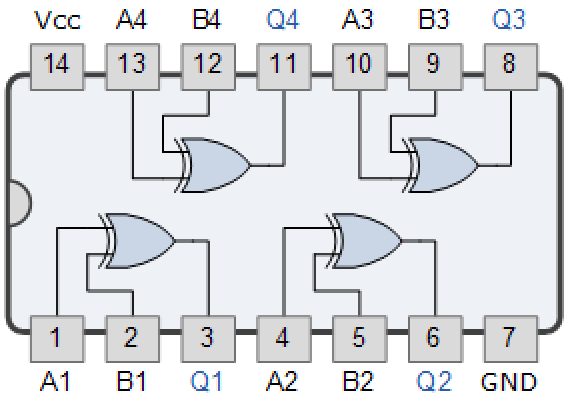
TTL Logic Ex-OR Gates

74LS86 Quad 2-input

CMOS Logic Ex-OR Gates

CD4030 Quad 2-input

7486 Quad 2-input Exclusive-OR Gate



The **Exclusive-OR** logic function is a very useful circuit that can be used in many different types of computational circuits. Although not a basic logic gate in its own right, its usefulness and versatility has turned it into a standard logical function complete with its own Boolean expression, operator and symbol. The *Exclusive-OR Gate* is widely available as a standard quad two-input 74LS86 TTL gate or the 4030B CMOS package.

One of its most commonly used applications is as a basic logic comparator which produces a logic "1" output when its two input bits are not equal. Because of this, the exclusive-OR gate has an inequality status being known as an odd function. In order to compare numbers that contain two or more bits, additional exclusive-OR gates are needed with the 74LS85 logic comparator being 4-bits wide.

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic *Exclusive-NOR* gate known commonly as the Ex-NOR Gate function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

57 Comments

Join the conversation

Your Name

Email Address

Write your comment here

SUBMIT

S Srushti

A very well explanation ✍️👍......... thanks a lot to author,helped me for clearing mine all basic concepts.

Posted on October 19th 2018 | 3:36 pm

Reply

---

a alvine llavu

Thanks to the author, i missed my class but after reading this i have understood everything, thanks alot

Posted on October 13th 2018 | 3:40 am

Reply

S Succoth

Great

Posted on September 23rd 2018 | 5:00 pm

Reply

Dom Dev

This website gives me great joy. Thank you

Posted on September 23rd 2018 | 4:06 pm

Reply

B

Biplob

digital systems by Ronald j. tocci book page no. 144-145 here shows that XOR and XNOR gates have only 2 input. so how can possible it as a 3 input???????? which is error ?????????

Posted on September 03rd 2018 | 5:53 pm

Reply

W

Wayne Storr

Neither is an error, I give you more information about Ex-OR and Ex-NOR gates than your book, and all for free.

Posted on September 03rd 2018 | 8:08 pm

Reply

D

Datha

S why don't u 🤔 give an easy method

Posted on September 18th 2018 | 1:59 am

Reply

A

Abhishek majhi

It is a assum tutorial

Posted on August 30th 2018 | 2:25 pm

Reply

S Shinde pravin p

Vnice

Posted on August 10th 2018 | 5:24 am

Reply

P P.mohan kuma

Good

Posted on June 25th 2018 | 7:07 pm

Reply

B Bipin Raj

Thanks for your help.

Posted on June 23rd 2018 | 3:42 pm

Reply

---

h haile

good how it implemented that

Posted on May 06th 2018 | 1:49 pm

Reply

View More