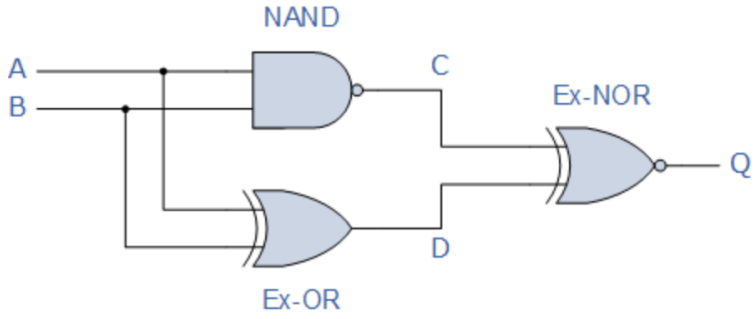
In this Boolean Algebra examples tutorial we will see that the Laws of Boolean Algebra can be used to identify unnecessary logic gates within a digital logic design reducing the number of gates required saving on both power consumption and cost.

We have seen throughout this section that digital logic functions can be defined and displayed as either a Boolean Algebra expression or as a logic gate truth table. So here are a few examples of how we can use **Boolean Algebra** to simplify larger digital logic circuits.

Boolean Algebra Examples No1

Construct a Truth Table for the logical functions at points C, D and Q in the following circuit and identify a single logic gate that can be used to replace the whole circuit.



First observations tell us that the circuit consists of a 2-input NAND gate, a 2-input EX-OR gate and finally a 2-input EX-NOR gate at the output. As there are only 2 inputs to the circuit labelled A and B, there can only be 4 possible combinations of the input (2^2) and these are: 0-0, 0-1, 1-0 and finally 1-1. Plotting the logical functions from each gate in tabular form will give us the following truth table for the whole of the logic circuit below.

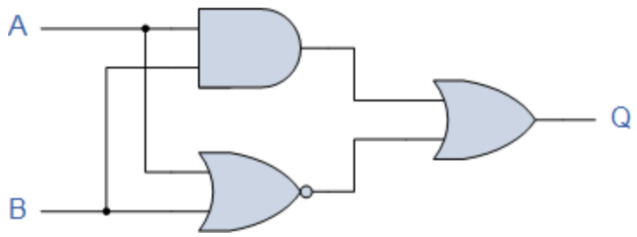
Inputs		Output at		
A	B	C	D	Q
0	0	1	0	0
0	1	1	1	1
1	0	1	1	1
1	1	0	0	1

From the truth table above, column C represents the output function generated by the NAND gate, while column D represents the output function from the Ex-OR gate. Both of these two output expressions then become the input condition for the Ex-NOR gate at the output.

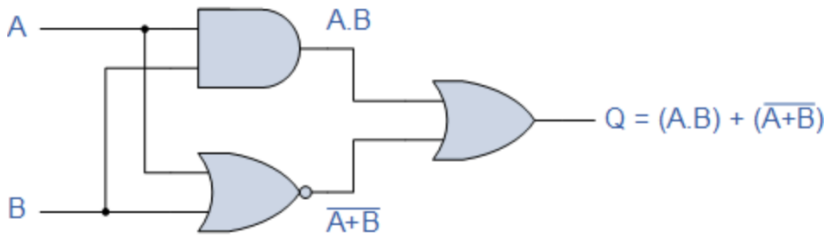
It can be seen from the truth table that an output at Q is present when any of the two inputs A or B are at logic 1. The only truth table that satisfies this condition is that of an OR Gate. Therefore, the whole of the above circuit can be replaced by just one single **2-input** OR Gate.

Boolean Algebra Examples No2

Find the Boolean algebra expression for the following system.



The system consists of an AND Gate, a NOR Gate and finally an OR Gate. The expression for the AND gate is A.B, and the expression for the NOR gate is $\overline{A+B}$. Both these expressions are also separate inputs to the OR gate which is defined as $\overline{A+B}$. Thus the final output expression is given as:



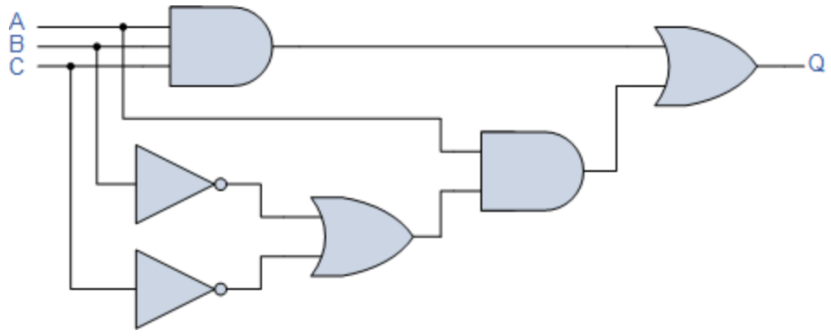
The output of the system is given as Q = (A.B) + ($\overline{A+B}$), but the notation $\overline{A+B}$ is the same as the De Morgan´s notation $\overline{A}.\overline{B}$, Then substituting $\overline{A}.\overline{B}$ into the output expression gives us a final output notation of Q = (A.B)+($\overline{A}.\overline{B}$), which is the Boolean notation for an Exclusive-NOR Gate as seen in the previous section.

Inputs		Intermediates		Output
B	A	A.B	$\overline{A+B}$	Q
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

Then, the whole circuit above can be replaced by just one single Exclusive-NOR Gate and indeed an Exclusive-NOR Gate is made up of these individual gate functions.

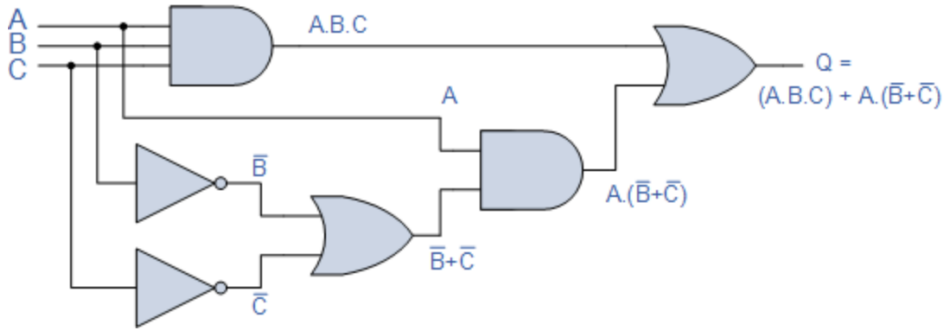
Example No3

Find the Boolean algebra expression for the following system.



This system may look more complicated than the other two to analyse but again, the logic circuit just consists of simple AND, OR and NOT gates connected together.

As with the previous Boolean examples, we can simplify the circuit by writing down the Boolean notation for each logic gate function in turn in order to give us a final expression for the output at Q.



The output from the 3-input AND gate is only at logic "1" when **ALL** the gates inputs are HIGH at logic level "1" (A.B.C). The output from the lower OR gate is only a "1" when one or both inputs B or C are at logic level "0". The output from the 2-input AND gate is a "1" when input A is a "1" and inputs B or C are at "0". Then the output at Q is only a "1" when inputs A.B.C equal "1" or A is equal to "1" and both inputs B or C equal "0", A.($\overline{B}+\overline{C}$).

By using "**de Morgan's theorem**" inputs B and input C cancel out as to produce an output at Q they can be either at logic "1" or at logic "0". Then this just leaves input A as the only input needed to give an output at Q as shown in the table below.

Inputs			Intermediates					Output
C	B	A	A.B.C	$\overline{B}$	$\overline{C}$	$\overline{B}+\overline{C}$	A.($\overline{B}+\overline{C}$)	Q
0	0	0	0	1	1	1	0	0
0	0	1	0	1	1	1	1	1
0	1	0	0	0	1	1	0	0
0	1	1	0	0	1	1	1	1
1	0	0	0	1	0	1	0	0
1	0	1	0	1	0	1	1	1
1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	1

Then we can see that the entire logic circuit above can be replaced by just one single input labelled “A” thereby reducing a circuit of six individual logic gates to just one single piece of wire, (or Buffer). This type of circuit analysis using *Boolean Algebra* can be very powerful and quickly identify any unnecessary logic gates within a digital logic design thereby reducing the number of gates required, the power consumption of the circuit and of course the cost.