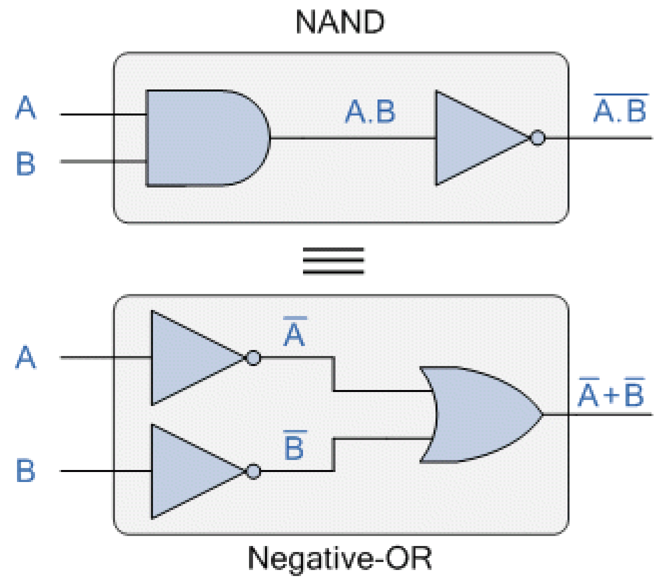
ELEC-DIGIE-S5



DeMorgan’s Theorem

DeMorgan´s Theorem and Laws can be used to to find the equivalency of the NAND and NOR gates

As we have seen previously, Boolean Algebra uses a set of laws and rules to define the operation of a digital logic circuit with “0’s” and “1’s” being used to represent a digital input or output condition. Boolean Algebra uses these zeros and ones to create truth tables and mathematical expressions to define the digital operation of a logic AND, OR and NOT (or inversion) operations as well as ways of expressing other logical operations such as the XOR (Exclusive-OR) function.

While George Boole’s set of laws and rules allows us to analyise and simplify a digital circuit, there are two laws within his set that are attributed to **Augustus DeMorgan** (a nineteenth century English mathematician) which views the logical NAND and NOR operations as separate NOT AND and NOT OR functions respectively.

But before we look at **DeMorgan’s Theory** in more detail, let’s remind ourselves of the basic logical operations where A and B are logic (or Boolean) input binary variables, and whose values can only be either “0” or “1” producing four possible input combinations, 00, 01, 10, and 11.

Truth Table for Each Logical Operation

Input Variable	Output Conditions

A	B	AND	NAND	OR	NOR
0	0	0	1	0	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	0	1	0

The following table gives a list of the common logic functions and their equivalent Boolean notation where a "." (a dot) means an AND (product) operation, a "+" (plus sign) means an OR (sum) operation, and the complement or inverse of a variable is indicated by a bar over the variable.

Logic Function	Boolean Notation
AND	A.B
OR	A+B
NOT	$\overline{A}$
NAND	$\overline{A.B}$
NOR	$\overline{A+B}$

DeMorgan's Theory

DeMorgan's Theorems are basically two sets of rules or laws developed from the Boolean expressions for AND, OR and NOT using two input variables, A and B. These two rules or theorems allow the input variables to be negated and converted from one form of a Boolean function into an opposite form.

DeMorgan's first theorem states that two (or more) variables NOR´ed together is the same as the two variables inverted (Complement) and AND´ed, while the second theorem states that two (or more) variables NAND´ed together is the same as the two terms inverted (Complement) and OR ´ed. That is replace all the OR operators with AND operators, or all the AND operators with an OR operators.

DeMorgan's First Theorem

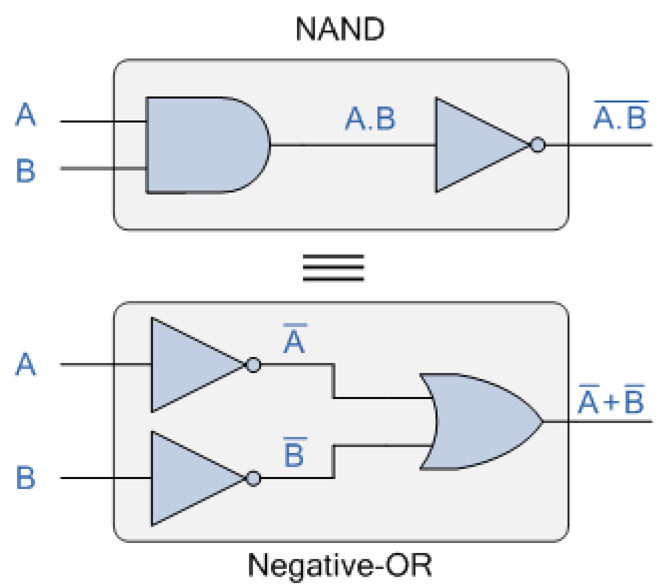
DeMorgan's First theorem proves that when two (or more) input variables are AND'ed and negated, they are equivalent to the OR of the complements of the individual variables. Thus the equivalent of the NAND function and is a negative-OR function proving that $\overline{A.B} = \overline{A}+\overline{B}$ and we can show this using the following table.

Verifying DeMorgan's First Theorem using Truth Table

Inputs		Truth Table Outputs For Each Term				
B	A	A.B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$
0	0	0	**1**	1	1	**1**
0	1	0	**1**	0	1	**1**
1	0	0	**1**	1	0	**1**
1	1	1	**0**	0	0	**0**

We can also show that $\overline{A.B} = \overline{A}+\overline{B}$ using logic gates as shown.

DeMorgan's First Law Implementation using Logic Gates



The top logic gate arrangement of: $\overline{A.B}$ can be implemented using a NAND gate with inputs A and B. The lower logic gate arrangement first inverts the two inputs producing $\overline{A}$ and $\overline{B}$ which become the inputs to the OR gate. Therefore the output from the OR gate becomes: $\overline{A}+\overline{B}$

Thus an OR gate with inverters (NOT gates) on each of its inputs is equivalent to a NAND gate function, and an individual NAND gate can be represented in this way as the equivalency of a NAND gate is a negative-OR.

DeMorgan's Second Theorem

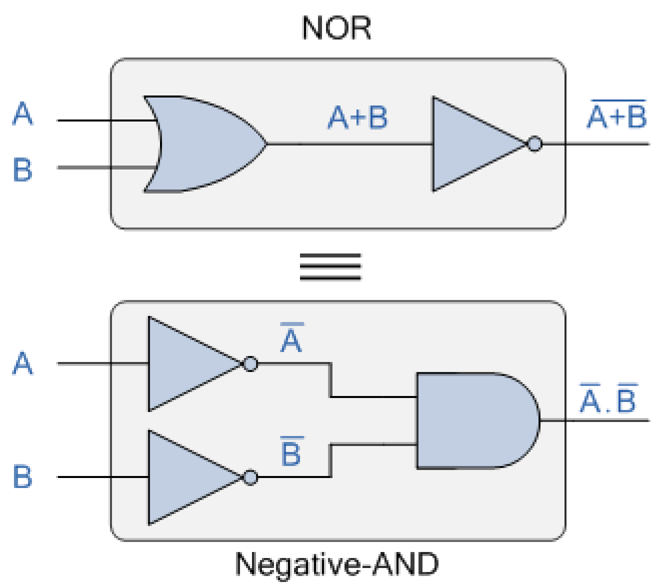
DeMorgan's Second theorem proves that when two (or more) input variables are OR'ed and negated, they are equivalent to the AND of the complements of the individual variables. Thus the equivalent of the NOR function and is a negative-AND function proving that $\overline{A+B} = \overline{A}.\overline{B}$ and again we can show this using the following truth table.

Verifying DeMorgan's Second Theorem using Truth Table

Inputs		Truth Table Outputs For Each Term				
B	A	A+B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	0	**1**	1	1	**1**
0	1	1	**0**	0	1	**0**
1	0	1	**0**	1	0	**0**
1	1	1	**0**	0	0	**0**

We can also show that $\overline{A+B} = \overline{A}.\overline{B}$ using logic gates as shown.

DeMorgan's Second Law Implementation using Logic Gates



The top logic gate arrangement of: $\overline{A+B}$ can be implemented using a NOR gate with inputs A and B. The lower logic gate arrangement first inverts the two inputs producing $\overline{A}$ and $\overline{B}$ which become the inputs to the AND gate. Therefore the output from the AND gate becomes: $\overline{A}.\overline{B}$

Thus an AND gate with inverters (NOT gates) on each of its inputs is equivalent to a NOR gate function, and an individual NOR gate can be represented in this way as the equivalency of a NORgate is a negative-AND.

Although we have used DeMorgan's theorems with only two input variables A and B, they are equally valid for use with three, four or more input variable expressions, for example:

For a 3-variable input

$$\overline{A.B.C} = \overline{A}+\overline{B}+\overline{C}$$

and also

$$\overline{A+B+C} = \overline{A}.\overline{B}.\overline{C}$$

For a 4-variable input

$$\overline{A.B.C.D} = \overline{A}+\overline{B}+\overline{C}+\overline{D}$$

and also

$$\overline{A+B+C+D} = \overline{A}.\overline{B}.\overline{C}.\overline{D}$$

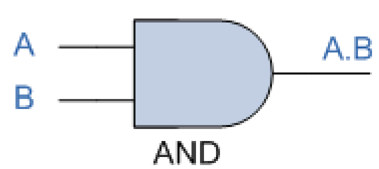
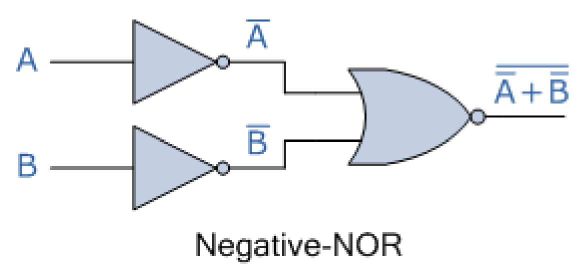
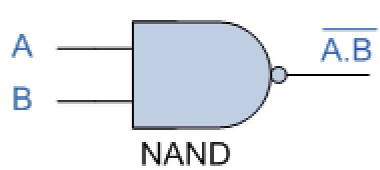
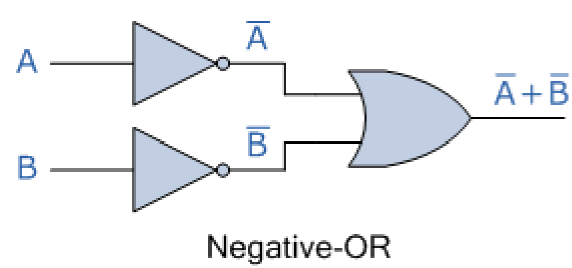
and so on.

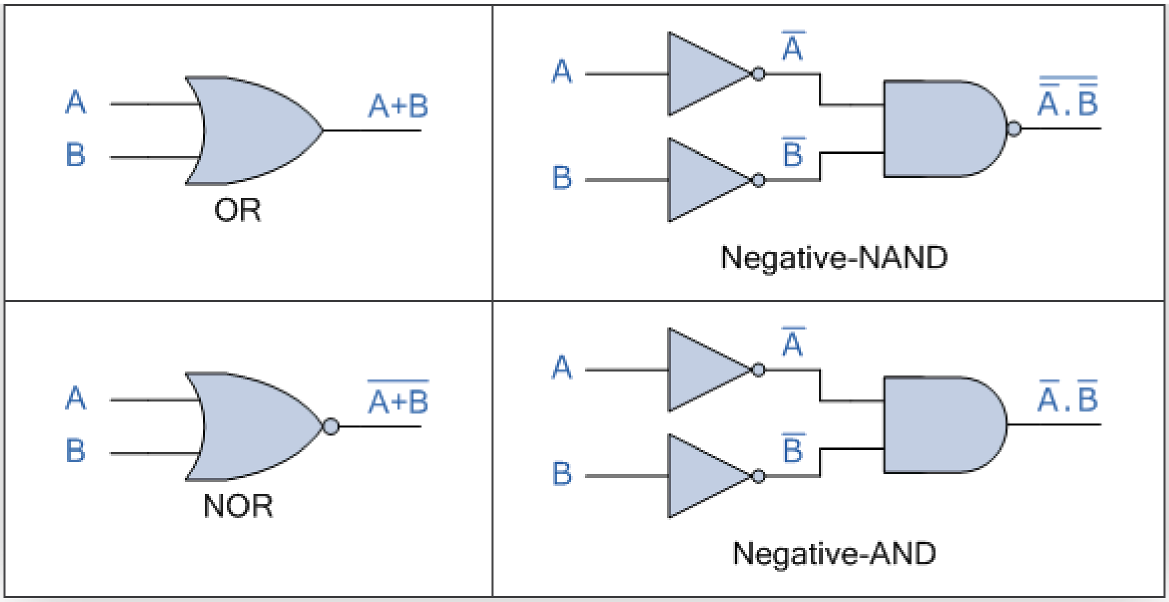
DeMorgan's Equivalent Gates

We have seen here that DeMorgan's Theorems replace all of the AND (.) operators with OR (+) and vice versa and then complements each of the terms or variables in the expression by inverting it, that is 0's to 1's and 1's to 0's before inverting the entire function.

Thus to obtain the DeMorgan equivalent for an AND, NAND, OR or NOR gate, we simply add inverters (NOT-gates) to all inputs and outputs and change an AND symbol to an OR symbol or change an OR symbol to an AND symbol as shown in the following table.

DeMorgan's Equivalent Gates

<table>
<tr><th>Standard Logic Gate</th><th>DeMorgan's Equivalent Gate</th></tr>
<tr><td>
</td><td>
</td></tr>
<tr><td>
</td><td>
</td></tr>
</table>



Then we have seen that the complement of two (or more) AND'ed input variables is equivalent to the OR of the complements of these variables, and that the complement of two(or more) OR'edvariables is equivalent to the AND of the complements of the variables as defined by *DeMorgan*.