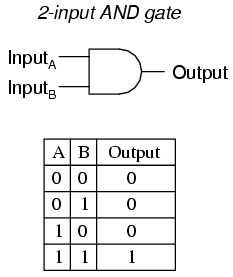
**P4 – C arry out Boolean logic operations**

**Introduction**

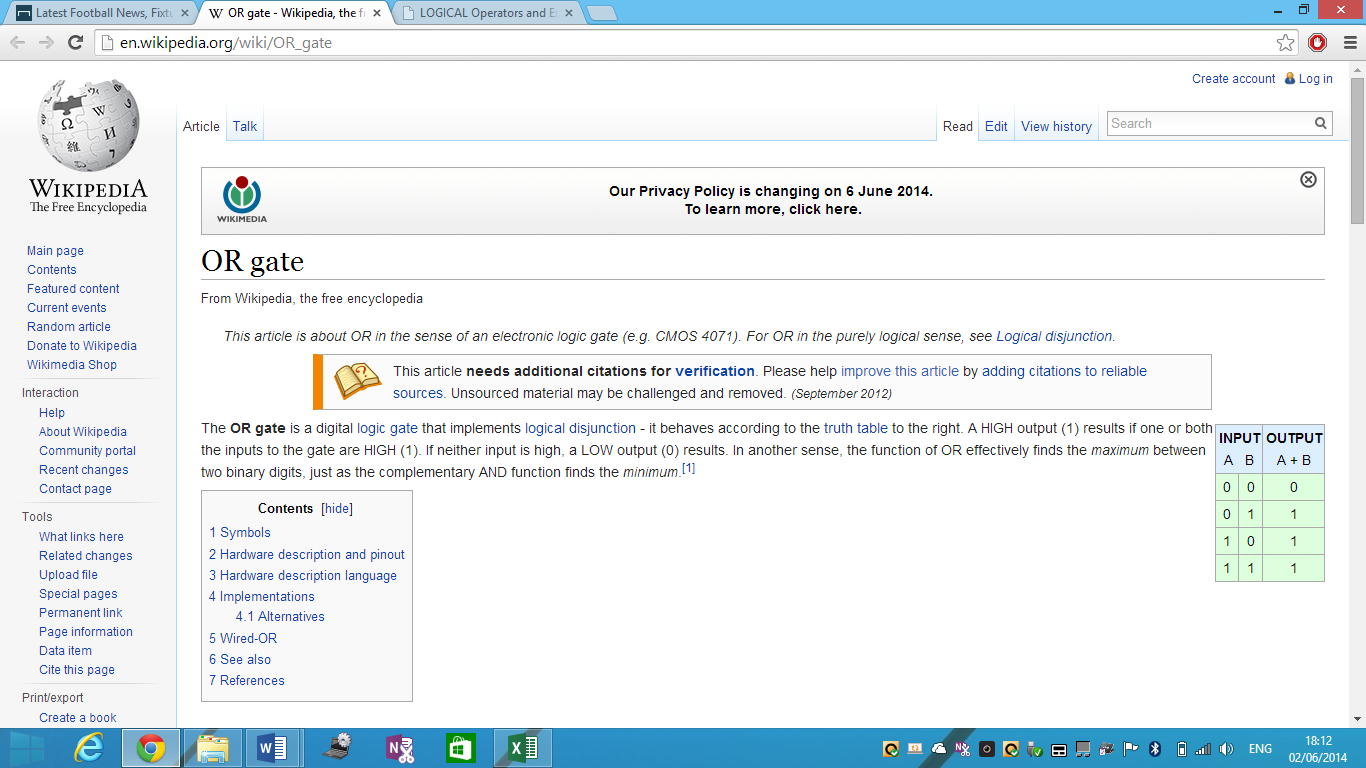
Logical operators are used in case of arithmetic, relational operators and logical operators. If these expressions are involved, they would be used followed by the logical operator. They are five logical operators, but I am going to be talking about the main three:

* AND Gate
* OR Gate
* NOT Gate
* NAND
* NOR
* Exclusive OR
* Half Adder
* Adder

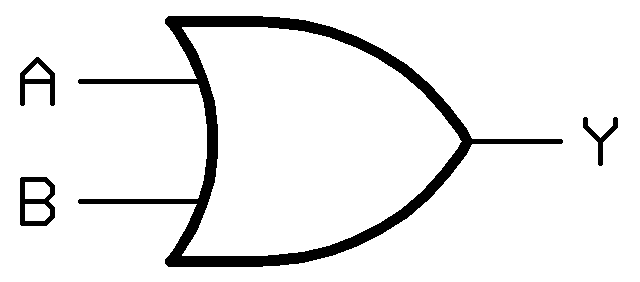
**AND Gate**

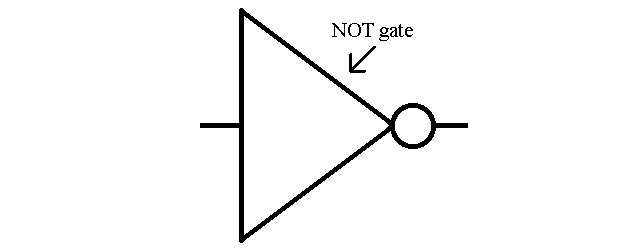
The **AND Gate** involves either if the two numbers are high. A high number is one and a low number is zero. Therefore, if both numbers are one, it would be an AND result. Symbols are represented for each gate and all of them are different. As you can see for figure 1.3, it shows how AND Gate works. The symbol is also present. Both of the inputs are inputted and if they are the same, they would be high (1). The table shows how it works. As shown above, any binary code is 4-digits, which is why there is four digits.

**Fig 1.3**

**OR Gate**

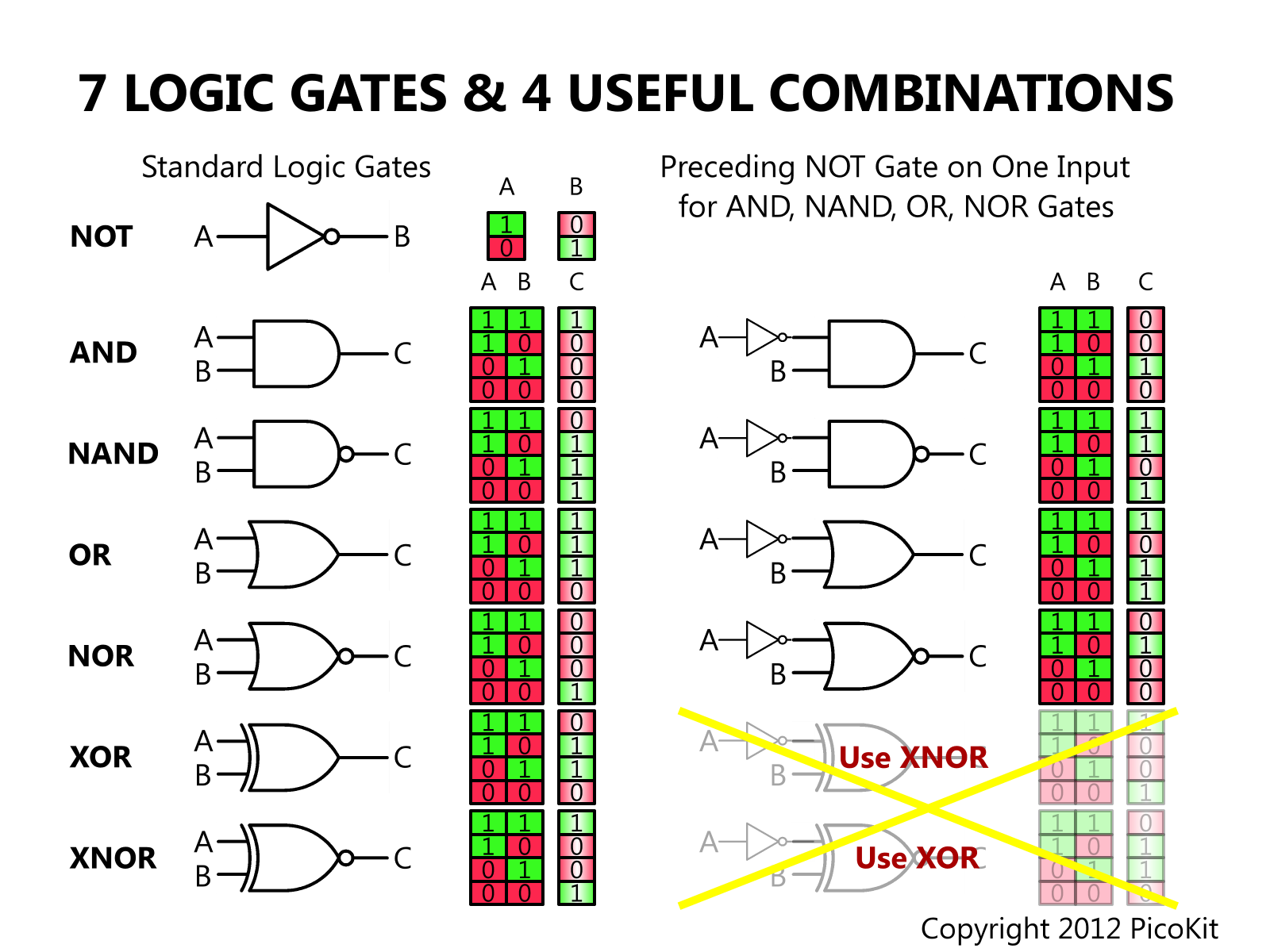
The **OR Gate** is different from AND Gate. Each gate acts similar, but the output and symbols are different. The OR Gate consists of the same adding as AND Gate, but they are different when they both are added up. An example could be that if A = zero and B = one the output would be one. In addition, if both of the numbers are high (1), the output would be zero. Both of the images show how OR Gate looks like and how they operate.

**Fig 1.4**

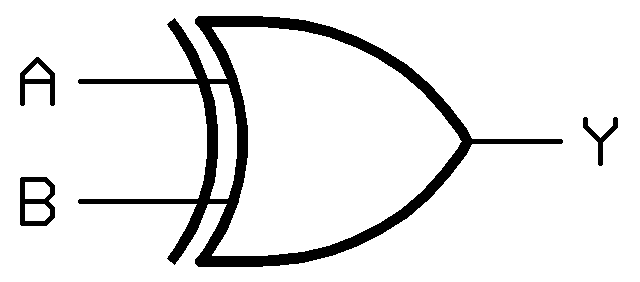
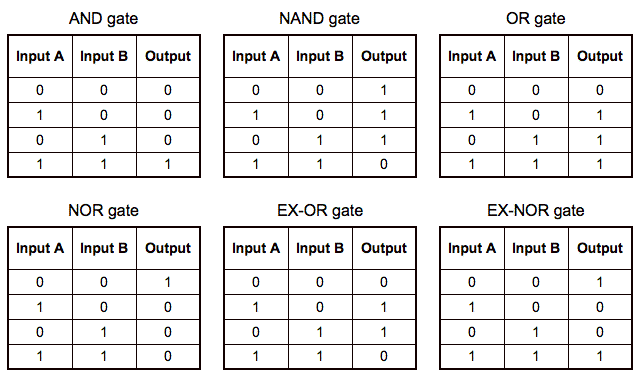


**NOT Gate**

The NOT Gateoperates differently. This is the one that is different from OR Gate and ‘AND’ Gate. The gate sends out the one as an output and if it receives zero, it puts that as an input. This gate can be used on integrated circuits.

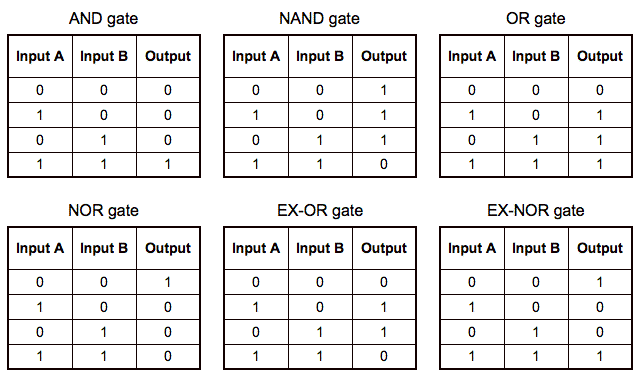


**XOR Gate**

Exclusive OR is another additional logical operation that operates which output is TRUE or FALSE. It can be written as ‘XOR Gate’. The figure 1.6 shows how exclusive OR looks like. Inputs and outputs are inserted within the Exclusive OR and it can tell whether it is false or true.

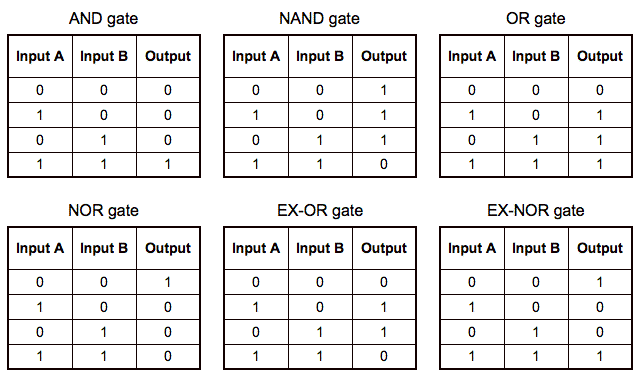
The truth table shows how it works. If both in A and B, zeros are inserted, the outcome will be 0. If 1 is inserted in A and 0 is inserted within B, the outcome will be 1 and vice versa. If 1 is inserted within both of them, the outcome will be 0. This is the table of Exclusive OR.

**NAND Gate**



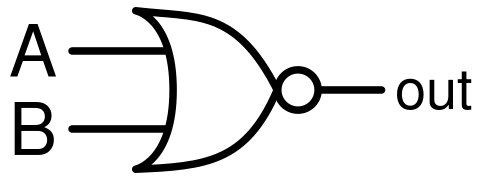
A NAND gate only allows it operates if one of the inputs are one and the other is operate, zero. If both inputs were the same, 0+0 and 1+1, they would result in either one or zero shown on the table.

This is equivalent to NOT and ‘AND’ gate

**NOR Gate**

NOR gate, as any other gate, behaves according to the truth table. In addition, this can be seen as an AND gate. Any input that is the opposite e.g. HIGH (1) and LOW (0). This would equal to zero the answer. However, if both of them were HIGH, they would be zero, vice versa.

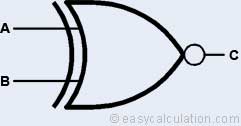
The image shows how the NOR gate operates. It looks a bit like the AND gate, but it is different.



**XNOR Gate**

An XNOR gate is different to NOR gate. This, as any other gate, works according to the truth table. The two inputs that are inserted is HIGH (1) and LOW (1). An example could be that if both are LOW, the output would be HIGH (1). This behaves the opposite of XNOR for the truth table. They look different as well. The picture below shows how this gate differs from NOR Gate.

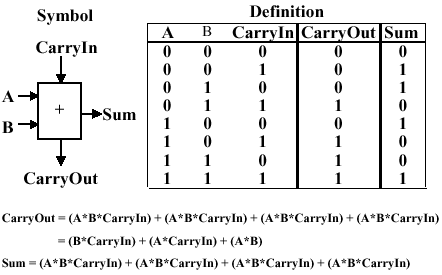
|  |  |  |
| --- | --- | --- |
| **XNOR Truth Table** | | |
| **Input** | | **Output** |
| **A** | **B** |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



*All of these gates that I’ve mentioned so far are used in a circuit. Some are connected with each other for the circuit to be complete. These are gates. Gates need inputs for the gate to be open or closed. This is why these are connected with the truth table. Each gate, as shown above, is different to each gate. They have different inputs for the gate to open.*

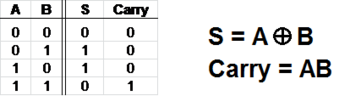
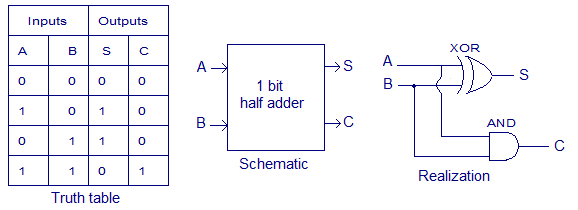
**Adder**

An adder is a combination of two digital circuits that is used to adding the two together. A typical adder adds more columns to the truth table. The truth table contains A (input), B (output), and the added information is sum bit (S) and carry bit in and out (symbolised by C). They is another type of adder which is half adder. Adders are used for arithmetic logic unit(s), and some processors. Some combination of circuits could be ‘AND gate’, and ‘XOR’ gate. The picture shows the truth table of the full adder. The half adder has a different truth table.



**Half Adder**

As the name states, the adder is different from the full adder. The full adder consists of the A and B (input and output), sum bit (S) and carry bit ©. This is very simple, but it may have drawbacks of it being a half adder. As you can see the table below, the half adder consists of only two inputs (A and B). As you can see, the input and output is important for the half adder. Putting in zero for both would equal the carry to zero and the sum. However, if both of them are 1, the sum would be zero, but they would be a carry for it.

[](http://www.circuitstoday.com/wp-content/uploads/2012/03/half-adder-truth-table-schematic-realization.png)