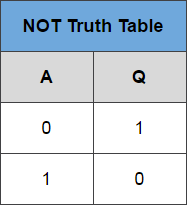
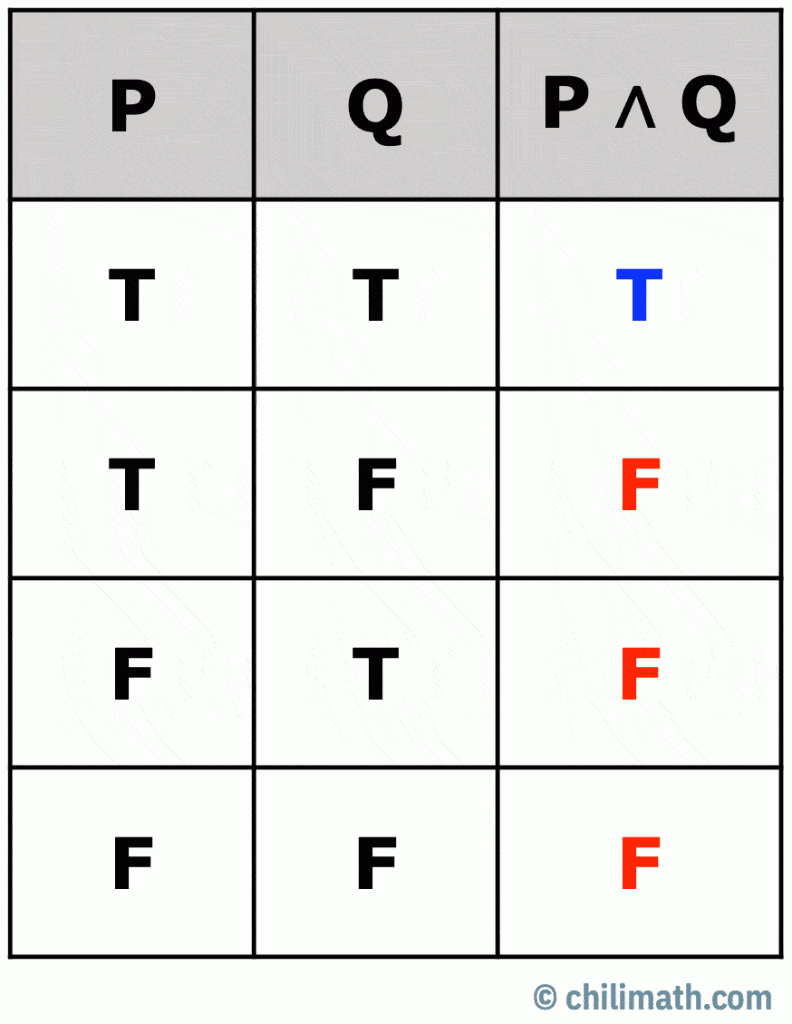
In this project the code is divided into two parts: The first part is where the functionality is defined while in the second the functionality is auto tested using various test cases.

Let us look at the gates

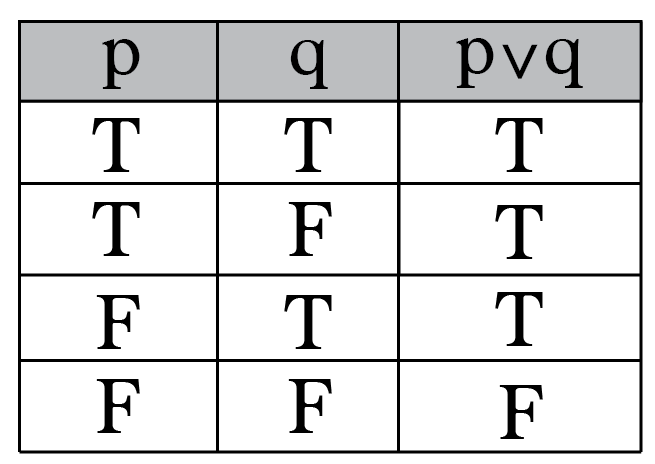
1. Not: It is a very basic of all gates. The gate inverts the input and returns it as the output. The operation function defines the functionality of the input by returning the inverted result of the input.



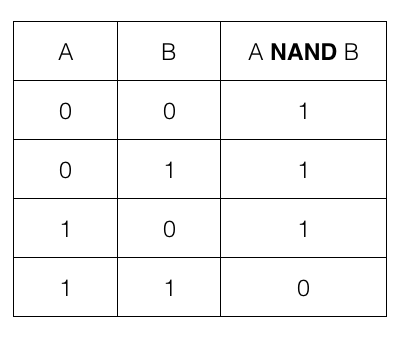
1. AND: This block is among the fundamental blocks for logic gates. This block acts like a multiplier. When the input to both the terminals is true, its then the output of this gate is True. In rest all cases it is false.



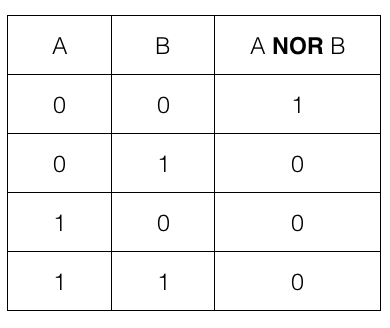
1. OR: this block like the and block is fundamental. It acts like an addition block. If either of the input or both the inputs are true. The output is true. The only case the gate gives a negative output is when both inputs are false.



1. NAND: This is a composite block and is a mixture of AND and NOT gates. The operation is AND operation followed by the NOT operation on the output of the AND. The behaviour of NAND is invert of a AND gate and the resultant output is FALSE when both the input are true. Rest in all cases the output is True.



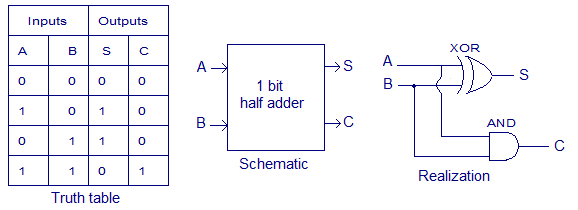
1. NOR: This is a composit of OR and NOT. The operation is OR followed by and NOT. The resultant behaviour is the output of NOR is true when both the input are false. In rest all cases the output is False.



The test code does the test for following conditions and if the test passes for all values of truth table, that’s when the overall test passes.

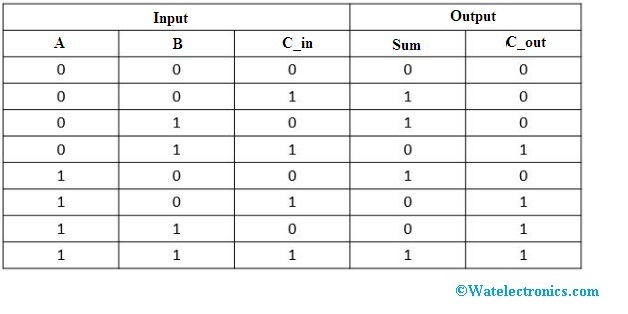
ALU: Arithmetic and logical unit. In this block we can perform complex operations of addition and subtraction on binary numbers. The basic building block of a ALU is the Half adder.

The half adder takes 2 inputs and returns a sum and a carry. The carry can be the cascaded into a Full adder that can than daisy chain to form a full adder combination.

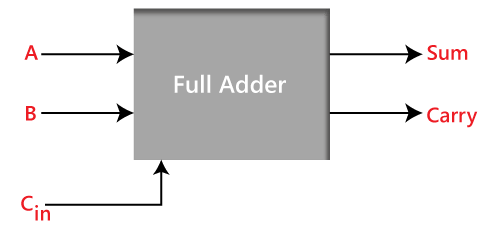


The following is the truth table, and symbol for half adder.

The case of a full adder is a bit more complex as it inputs a carry into its computation, thus 3 inputs, A,B and carry. To give the resultant sum and carry.



Truth table for a full adder



Symbol for the full adder.

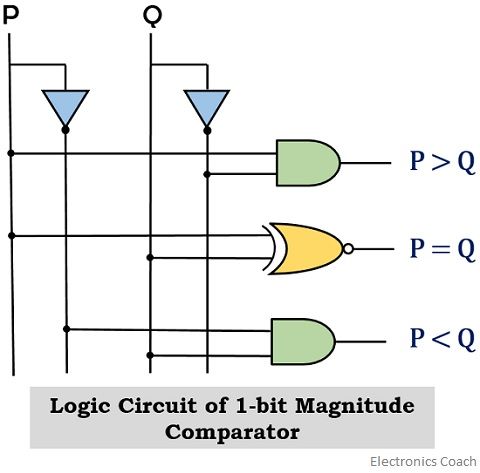
The C\_in is a carry from the previous stage of computation.

Now to perform a Full 8-bit calculation we use 8 full adder and the c\_in for the first bit is set to 0. Thus, we have daisy chain of 8 Full adders to give a full 8-bit calculation.

The similar circuit can be used to create a 8 bit subtractor.

To achieve this, the second input is inverted and the c\_in is set to 1 for the first Full adder. This is done so that the circuit can perform a 2’s complete subtraction.

A comparator is a single system that outputs the three possible results based on the input. Less than, greater than and equal to. The following is the implementation of the comparator



In case of a multi bit comparison, the single bit comparator is cascaded and the compounded output of all the comparators is used to compare the two numbers.