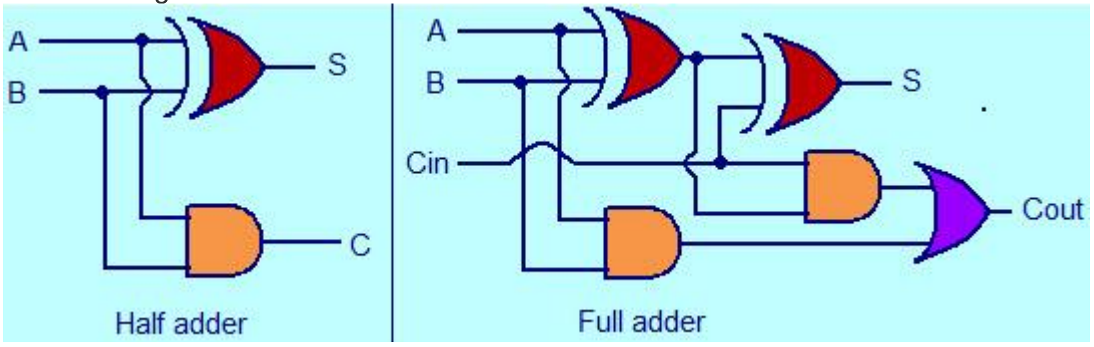
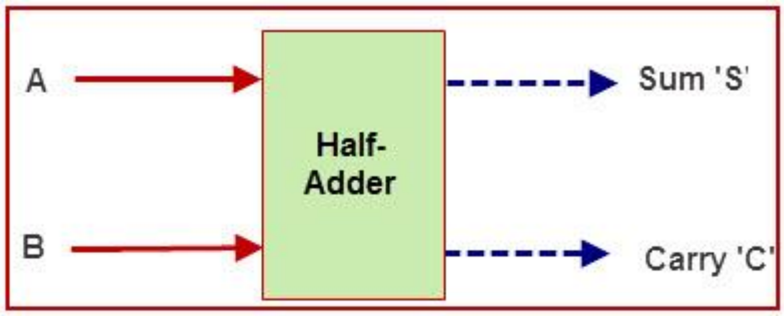
HALF ADDER AND FULL ADDER

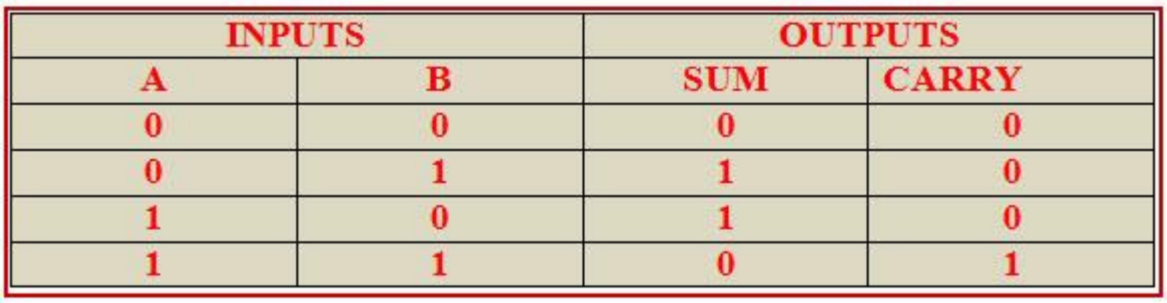
An adder is a digital logic circuit in electronics that implements addition of numbers. In many computers and other types of processors, adders are used to calculate addresses, similar operations and table indices in the ALU and also in other parts of the processors. These can be built for many numerical representations like excess-3 or binary coded decimal. Adders are classified into two types: half adder and full adder. The half adder circuit has two inputs: A and B, which add two input digits and generate a carry and sum. The full adder circuit has three inputs: A and C, which add the three input numbers and generate a carry and sum. This article gives brief information about half adder and full adder in tabular forms and circuit diagrams.



Half Adder

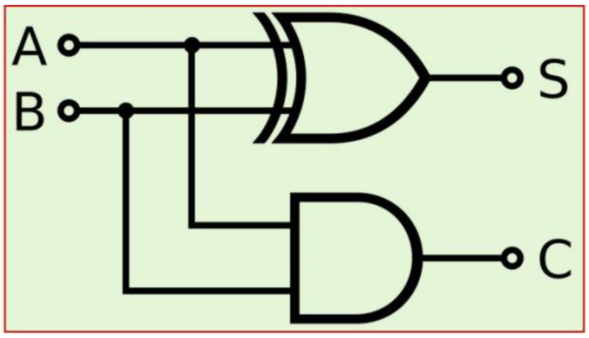


0+0 = 0
0+1 = 1
1+0 = 1
1+1 = 10
These are the least possible single-bit combinations. But the result for 1+1 is 10, the sum result must be re-written as a 2-bit output. Thus, the equations can be written as
0+0 = 00
0+1 = 01
1+0 = 01
1+1 = 10
The output '1'of '10' is carry-out. 'SUM' is the normal output and 'CARRY' is the carry-out.



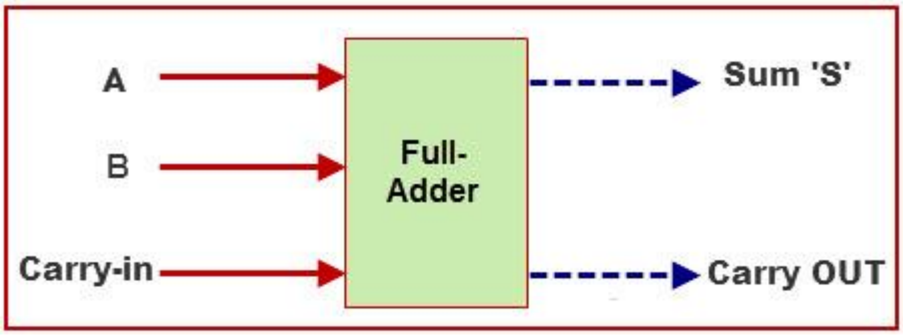
INPUTS		OUTPUTS	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Now it has been cleared that 1-bit adder can be easily implemented with the help of the XOR Gate for the output 'SUM' and an AND Gate for the 'Carry'. When we need to add, two 8-bit bytes together, we can be done with the help of a full-adder logic. The half-adder is useful when you want to add one binary digit quantities. A way to develop a two-binary digit adders would be to make a truth table and reduce it. When you want to make a three binary digit adder, do it again. When you decide to make a four digit adder, do it again. The circuits would be fast, but development time is slow.



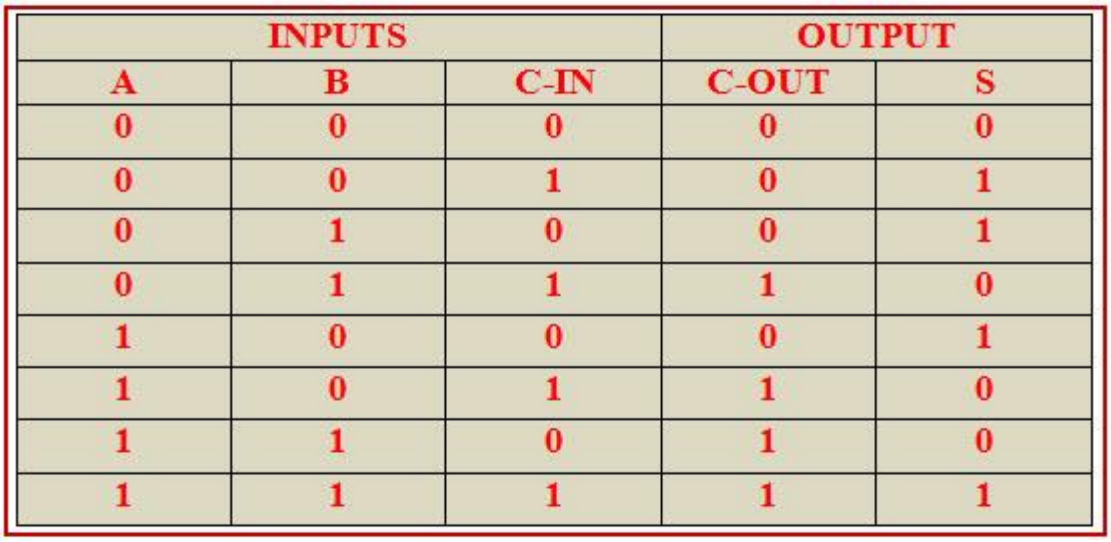
The simplest expression uses the exclusive OR function: Sum=A XOR B. An equivalent expression in terms of the basic AND, OR, and NOT is: SUM=A'.B + A.B' = ((NOT A) AND B) OR (A AND (NOT B)).

Full Adder



This adder is difficult to implement than a half-adder. The difference between a half-adder and a full-adder is that the full-adder has three inputs and two outputs, whereas half adder has only two inputs and two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. When a full-adder logic is designed, you string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next.

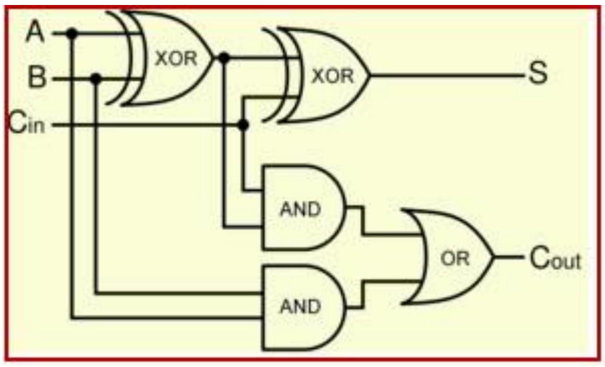
The output carry is designated as C-OUT and the normal output is designated as S.



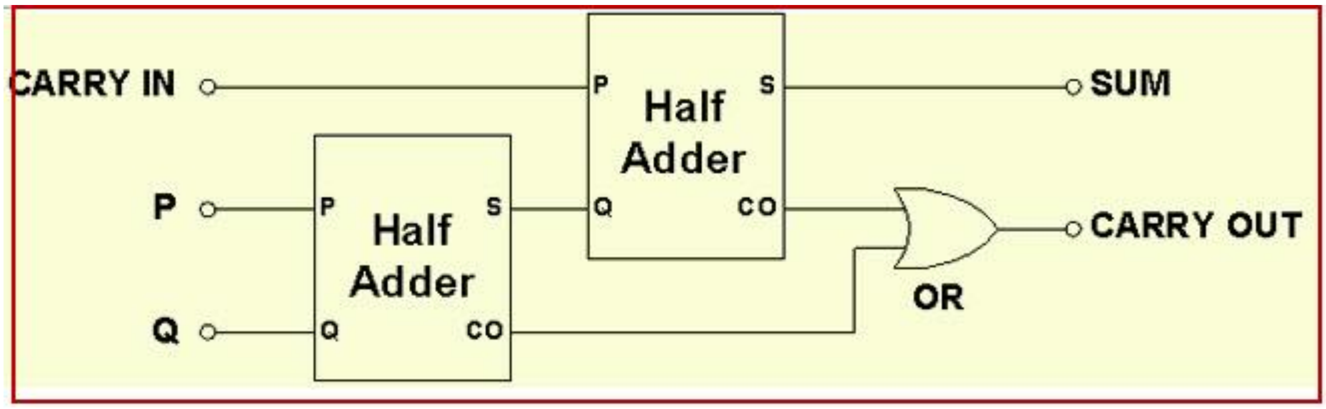
INPUTS			OUTPUT	
A	B	C-IN	C-OUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

With the truth-table, the full adder logic can be implemented. You can see that the output S is an XOR between the input A and the half-adder, SUM output with B and C-IN inputs. We take C-OUT will only be true if any of the two inputs out of the three are HIGH.

So, we can implement a full adder circuit with the help of two half adder circuits. At first, half adder will be used to add A and B to produce a partial Sum and a second half adder logic can be used to add C-IN to the Sum produced by the first half adder to get the final S output.



If any of the half adder logic produces a carry, there will be an output carry. So, COUT will be an OR function of the half-adder Carry outputs. Take a look at the implementation of the full adder circuit shown below. The implementation of larger logic diagrams is possible with the above full adder logic a simpler symbol is mostly used to represent the operation. Given below is a simpler schematic representation of a one-bit full adder.



Made by Rohnak Agarwal, 18.11.2019
Source : elprocus.com