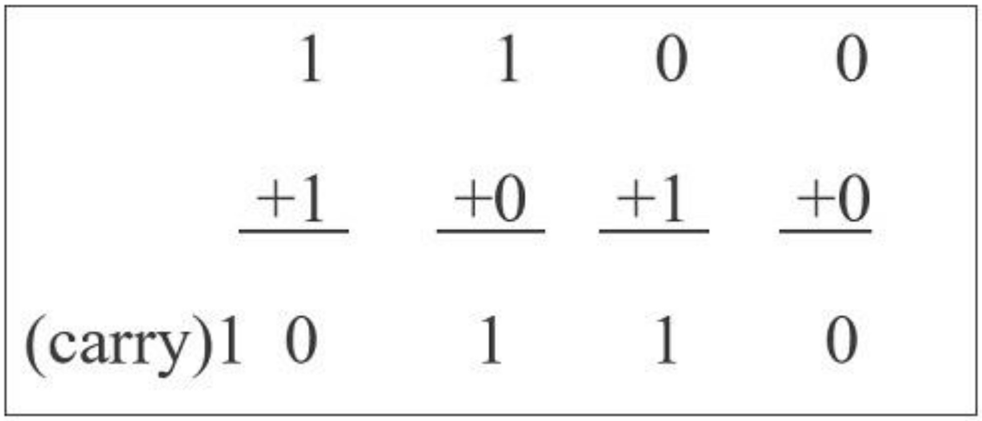
Binary Adder and Subtractor

Binary Addition Circuits

Addition and Subtraction are two basic Arithmetic Operations that must be performed by any Digital Computer. If both these operations can be properly implemented, then Multiplication and Division tasks become easy (as multiplication is repeated addition and division is repeated subtraction).

Consider the operation of adding two binary numbers, which is one of the fundamental tasks performed by a digital computer. The four basic addition operations two single bit binary numbers are:

- 0 + 0 = 0
- 1 + 0 = 1
- 0 + 1 = 1
- 1 + 1 = (Carry)1 0



	1	1	0	0
	+1	+0	+1	+0
(carry)1	0	1	1	0

In the first three operations, each binary addition gives sum as one bit , i.e., either 0 or 1. But for the fourth addition operation (where the inputs are 1 and 1), the result consists of two binary digits. Here, the lower significant bit is called as the ‘Sum Bit’, while the higher significant bit is called as the ‘Carry Bit’.

For single bit additions, there may not be an issue. The problem may arise when we try to add binary numbers with more than one bit.

The logic circuits which are designed to perform the addition of two binary numbers are called as Binary Adder Circuits. Depending on how they handle the output of the '1+1' addition, they are divided into:

- Half Adder
- Full Adder

Let us take a look at the binary addition performed by various adder circuits.

Half Adder

A logic circuit used for adding two 1-bit numbers or simply two bits is called as a Half Adder circuit. This circuit has two inputs and two outputs. The inputs are the two 1-bit binary numbers (known as Augend and Addend) and the outputs are Sum and Carry.

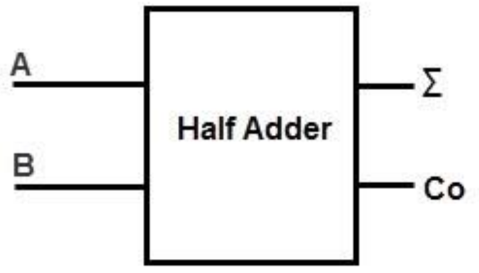
The following image shows the block diagram of Half Adder.

INPUT		**OUTPUT**	
A	**B**	**Sum**	**Carry**
0	0	0	0
0	1	**1**	0
1	0	**1**	0
1	1	0	**1**

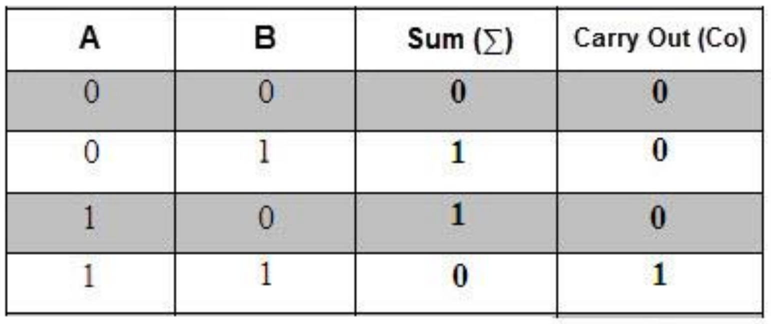
The truth table of the Half Adder is shown in the following table.

If we observe the 'Sum' values in the above truth table, it resembles an Ex-OR Gate. Similarly, the values for 'Carry' in the above truth table resembles an AND Gate.

So, to properly implement a Half Adder, you need two Logic Gates: an XOR gate for 'Sum' Output and an AND gate for 'Carry' output. The following image shows the Logic Diagram of a Half Adder.

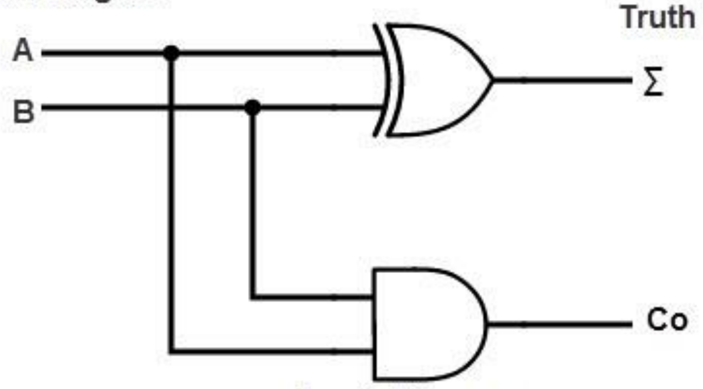


Block Diagram



A	B	Sum (∑)	Carry Out (Co)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Truth Table



Logic Diagram

In the above half adder circuit, inputs are labeled as A and B. The 'Sum' output is labeled as summation symbol (∑) and the Carry output is labeled with C_O.

Half adder is mainly used for addition of augend and addend of first order binary numbers i.e., 1-bit binary numbers. We cannot add binary numbers with more than one bit as the Half Adder cannot include the 'Carry' information from the previous sum.

Due to this limitation, Half Adder is practically not used in many applications, especially in multi-digit addition. In such applications, carry of the previous digit addition must be added along with two bits; hence it is a three bit addition.

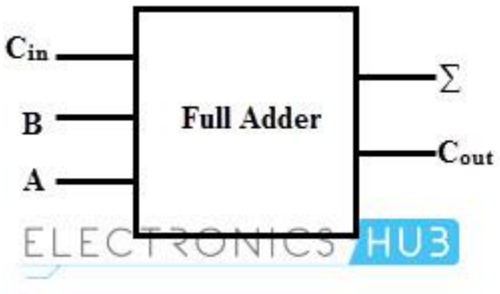
Full Adder

A Full Adder is a combinational logic circuit which performs addition on three bits and produces two outputs: a Sum and a Carry. As we have seen that the Half Adder cannot respond to three inputs and hence the full adder is used to add three digits at a time.

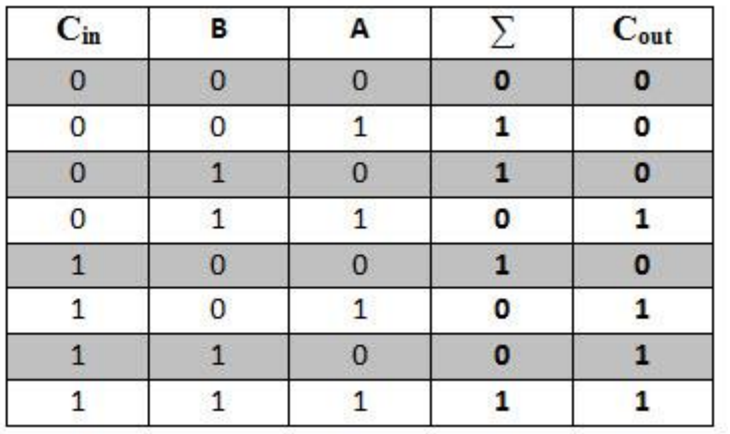
It consists of three inputs, of which two are input variables representing the two significant bits to be added, whereas the third input terminal is the carry from the previous addition. The two outputs are a Sum and Carry outputs.

INPUT			**OUTPUT**	
A	**B**	**C_{IN}**	**Sum**	**C_{OUT}**
0	0	0	0	0
0	0	1	**1**	0
0	1	0	**1**	0
0	1	1	0	**1**
1	0	0	**1**	0
1	0	1	0	**1**
1	1	0	0	**1**
1	1	1	**1**	**1**

The following image shows a block diagram of a Full Adder where the inputs are labelled as A, B and C_{IN}, while the outputs are labelled as ∑ and C_{OUT}.



Block Diagram of Full Adder



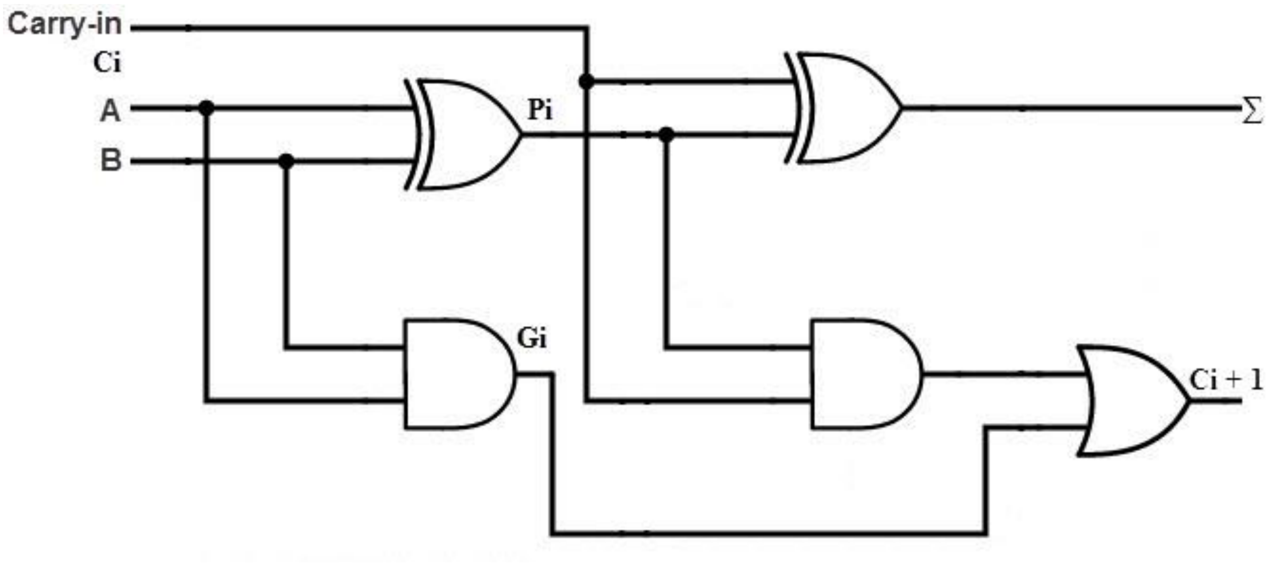
C_{in}	B	A	∑	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth Table

Coming to the truth table, the following table shows the truth table of a Full Adder.

From the above truth table, we can obtain the Boolean Expressions for both the Sum and Carry Outputs. Using those expressions, we can build the logic circuits for Full Adder. But by simplifying the equations further, we can derive at a point that a Full Adder can be easily implemented using two Half Adders and an OR Gate.

The following image shows a Full Adder Circuit implemented using two Half Adders and an OR Gate. Here, A and B are the main input bits, C_{IN} is the carry input, ∑ and C_{OUT} are the Sum and Carry Outputs respectively.

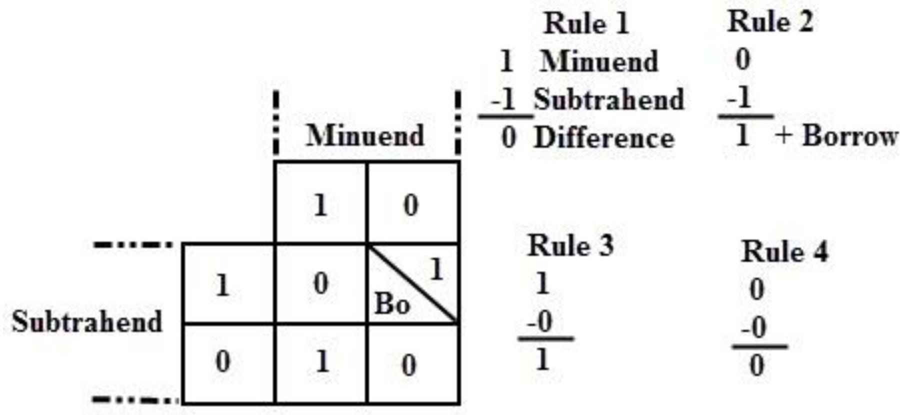


Binary Subtraction Circuits

Another basic arithmetic operation to be performed by Digital Computers is the Subtraction. Subtraction is a mathematical operation in which one integer number is deducted from another to obtain the equivalent quantity. The number from which other number is to be deducted is called as 'Minuend' and the number subtracted from the minuend is called 'Subtrahend'.

Similar to the binary addition, binary subtraction is also has four possible basic operations. They are:

- 0 – 0 = 0
- 0 – 1 = (Borrow)1 1
- 1 – 0 = 1
- 1 – 1 = 0



The above figure shows the four possible rules or elementary operations of the binary subtractions. In all the operations, each subtrahend bit is deducted from the minuend bit.

But in the second rule, minuend bit is smaller than the subtrahend bit, hence 1 is borrowed to perform the subtraction. Similar to the adder circuits, basic subtraction circuits are also of two types:

- Half Subtractor
- Full Subtractor

Half Subtractors

A Half Subtractor is a multiple output Combinational Logic Circuit that does the subtraction of two 1-bit binary numbers. It has two inputs and two outputs. The two inputs correspond to the two 1-bit binary numbers and the two outputs corresponds to the Difference bit and Borrow bit (in contrast to Sum and Carry in Half Adder).

The following image shows the block diagram of a Half Subtractor.

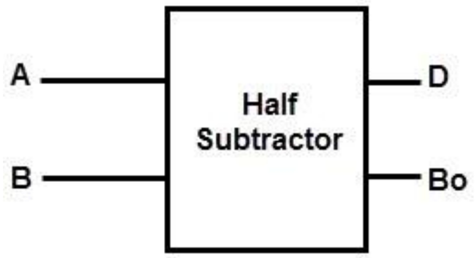
Image

Following table shows the truth table of a Half Subtractor.

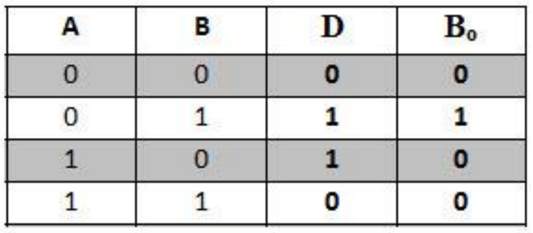
INPUT		**OUTPUT**	
A	**B**	**Difference**	**Borrow**
0	0	0	0
0	1	**1**	**1**
1	0	**1**	0
1	1	0	0

From the above truth table, we can say that the ‘Difference’ output of the Half Subtractor is similar to an XOR output (which is also same as the Sum output of the Half Adder). Thus, the Half Subtraction is also performed by the Ex-OR gate with an AND gate with one inverted input and one normal input, requiring to perform the Borrow operation.

The following image shows the logic circuit of a Half Subtractor.

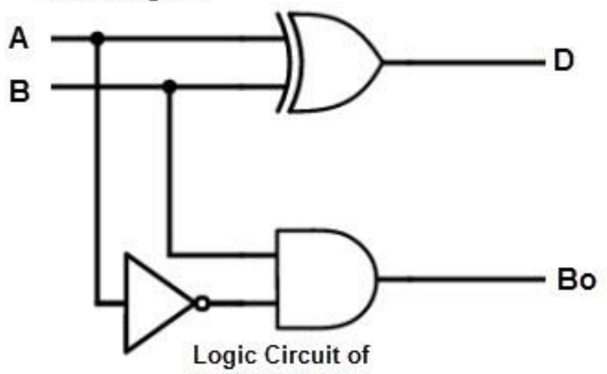


Block Diagram



A	B	D	B_o
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Truth Table



Logic Circuit of Half Subtractor

This circuit is similar to that of the Half Adder with only difference being the minuend input i.e., A is complemented before applied at the AND gate to implement the borrow output.

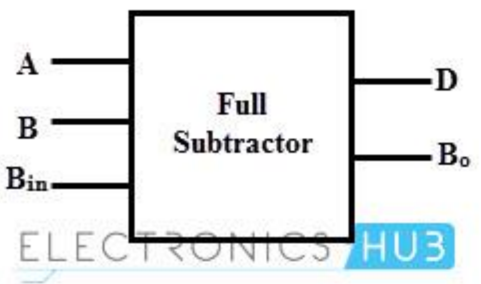
In case of multi-digit subtraction, subtraction between the two digits must be performed along with borrow of the previous digit subtraction, and hence a subtractor needs to have three inputs, which is not possible with a Half Subtractor. Therefore, a half subtractor has limited set of applications and strictly speaking, it is not used in practice.

Full Subtractor

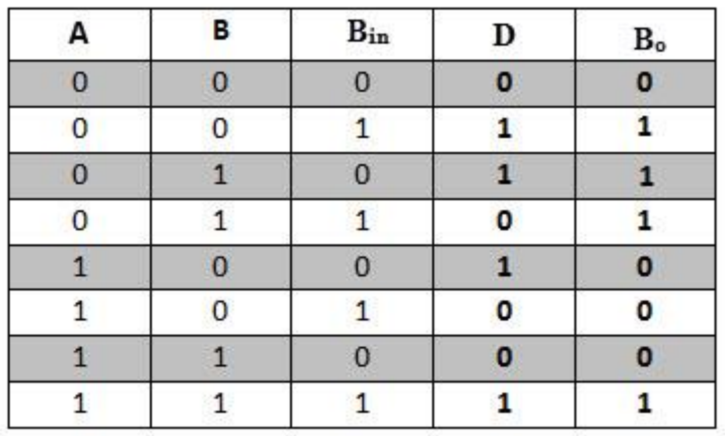
A Full Subtractor is a combinational logic circuit which performs a subtraction between the two 1-bit binary numbers and it also considers the borrow of the previous bit i.e., whether 1 has been borrowed by the previous minuend bit.

So, a Full Subtractor has three inputs, in which two inputs corresponding to the two bits to be subtracted (minuend A and subtrahend B), and a borrow bit, usually represented as B_{IN}, corresponding to the borrow operation. There are two outputs, one corresponds to the difference D output and the other Borrow output B_O.

The following image shows the block diagram of a full subtractor.



Block Diagram of Full Subtractor



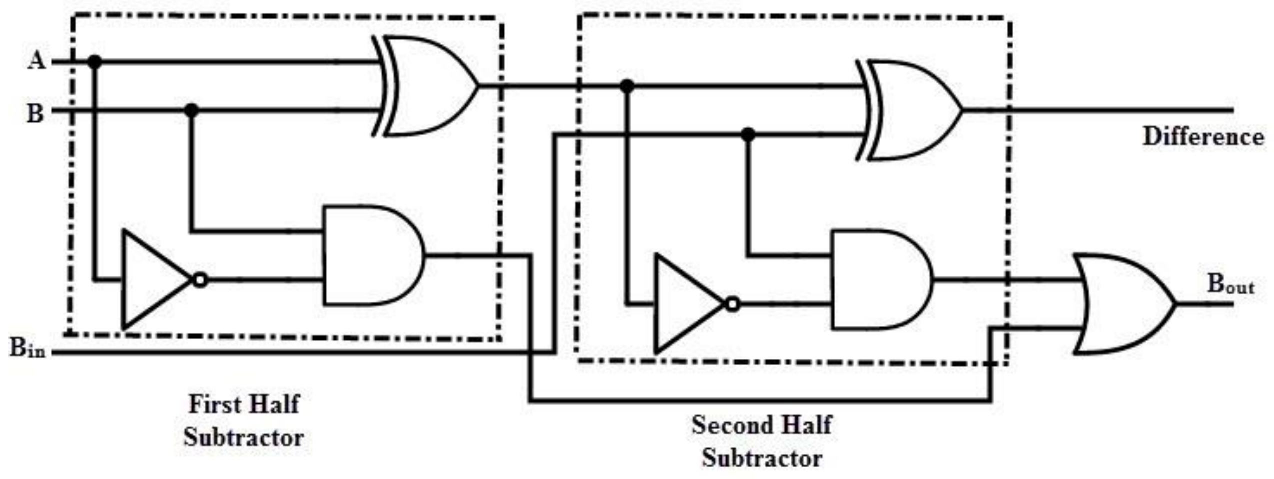
A	B	B_{in}	D	B_o
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Truth Table

The following table shows the truth table of a Full Subtractor.

INPUT			**OUTPUT**	
A	**B**	**B_{IN}**	**D**	**B_{OUT}**
0	0	0	0	0
0	0	1	**1**	**1**
0	1	0	**1**	**1**
0	1	1	0	**1**
1	0	0	**1**	0
1	0	1	0	0
1	1	0	0	0
1	1	1	**1**	**1**

By deriving the Boolean expression for the full subtractor from above truth table, we get the expression that tells that a full subtractor can be implemented with half subtractors with OR gate as shown in figure below.



By comparing the adder and subtractor circuits and truth tables, we can observe that the output D in the full subtractor is exactly same as the output S of the full adder. And the only difference is that input variable A is complemented in the full subtractor.

Therefore, it is possible to convert the full adder circuit into full subtractor by simply complementing the input A before it is applied to the gates to produce the final borrow bit output Bo.