Full Subtractor

The Half Subtractor is used to subtract only two numbers. To overcome this problem, a full subtractor was designed. The full subtractor is used to subtract three 1-bit numbers A, B, and C, which are minuend, subtrahend, and borrow respectively. The full subtractor has three input states and two output states i.e., diff and borrow.

Block diagram

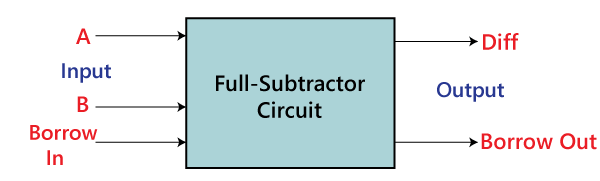


Figure below shows the truth table of the full subtractor.

1) When the inputs minuend (A) =0, Subtrahend (B) =0 and Borrow in (Bin) = 0 then the difference (D) = 0 and Borrow out (Bo) =0

2) When the inputs minuend (A) =0, Subtrahend (B) =0 and Borrow in (Bin) = 1 then the difference (D) = 1 and Borrow out (Bo) =1

3) When the inputs minuend (A) =0, Subtrahend (B) =1 and Borrow in (Bin) = 0 then the difference (D) = 1 and Borrow out (Bo) =1

4) When the inputs minuend (A) =0, Subtrahend (B) =1 and Borrow in (Bin) = 1 then the difference (D) = 0 and Borrow out (Bo) =1

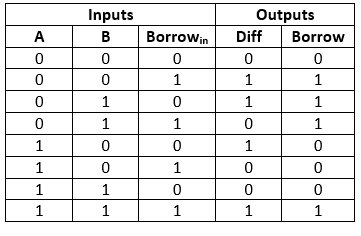
5) When the inputs minuend (A) =1, Subtrahend (B) =0 and Borrow in (Bin) = 0 then the difference (D) = 1 and Borrow out (Bo) =0

6) When the inputs minuend (A) =1, Subtrahend (B) =0 and Borrow in (Bin) = 1 then the difference (D) = 0 and Borrow out (Bo) =0

7) When the inputs minuend (A) =1, Subtrahend (B) =1 and Borrow in (Bin) = 0 then the difference (D) = 0 and Borrow out (Bo) =0

8) When the inputs minuend (A) =1, Subtrahend (B) =1 and Borrow in (Bin) = 1 then the difference (D) = 1 and Borrow out (Bo) =1

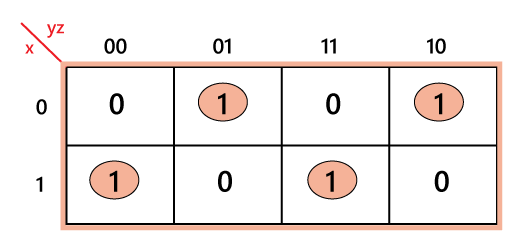
Truth Table



In the above table,

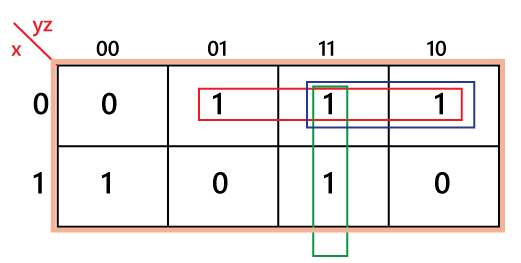
* 'A' and' B' are the input variables. These variables represent the two significant bits that are going to be subtracted.
* 'Borrowin' is the third input which represents borrow.
* The 'Diff' and 'Borrow' are the output variables that define the output values.
* The eight rows under the input variable designate all possible combinations of 0 and 1 that can occur in these variables.

The SOP form can be obtained with the help of K-map as:



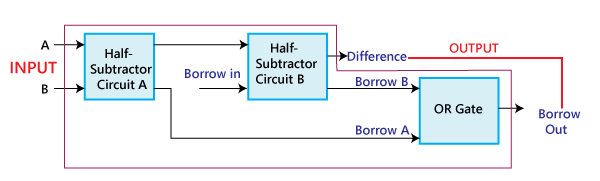
**Diff=xy' z'+x' y' z+xyz+x'yz'**

**=x XOR y XOR z**



**Borrow=x' z+x' y+yz**

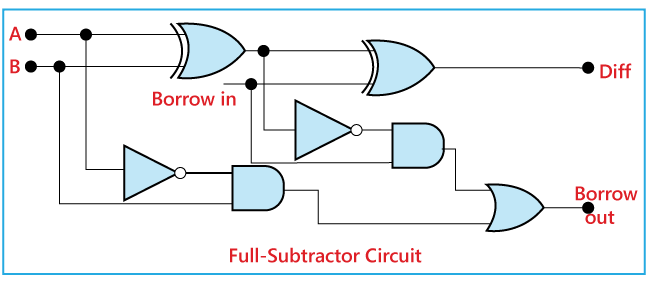
Construction of Full Subtractor Circuit:



**The above block diagram describes the construction of the Full subtractor circuit**. In the above circuit, there are two half adder circuits that are combined using the OR gate. The first half subtractor has two single-bit binary inputs A and B. As we know that, the half subtractor produces two outputs, i.e., 'Diff' and 'Borrow'. The 'Diff' output of the first subtractor will be the first input of the second half subtractor, and the 'Borrow' output of the first subtractor will be the second input of the second half subtractor. The second half subtractor will again provide 'Diff' and 'Borrow'. The final outcome of the Full subtractor circuit is the 'Diff' bit. In order to find the final output of the 'Borrow', we provide the 'Borrow' of the first and the second subtractor into the OR gate. The outcome of the OR gate will be the final carry 'Borrow' of full subtractor circuit.

The MSB is represented by the final 'Borrow' bit.

The full subtractor logic circuit can be constructed using the **'AND', '**[**XOR', and NOT gate**](https://circuitdigest.com/electronic-circuits/and-gate-circuit-working) with an [**OR gate**](https://circuitdigest.com/electronic-circuits/or-gate-circuit).



The actual logic circuit of the full subtractor is shown in the above diagram. The full subtractor circuit construction can also be represented in a Boolean expression.

Diff:

* Perform the XOR operation of input A and B.
* Perform the XOR operation of the outcome with 'Borrow'. So, the difference is (A XOR B) XOR 'Borrowin' which is also represented as:  
  (A ⊕ B) ⊕ 'Borrowin'

Borrow:

* Perform the 'AND' operation of the inverted input A and B.
* Perform the 'XOR' operation of input A and B.
* Perform the 'OR' operations of both the outputs that come from the previous two steps. So the 'Borrow' can be represented as:  
  A'.B + (A ⊕ B)'