Arithmetic Circuits

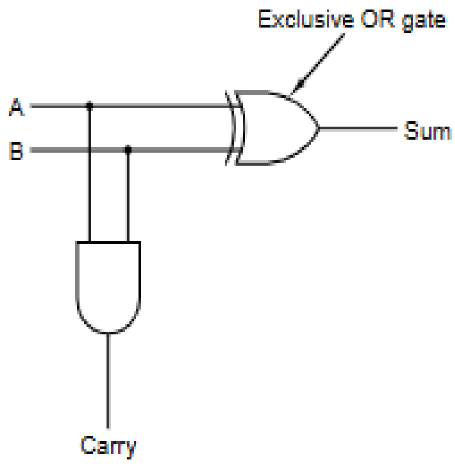
Half Adder and Full Adder

Arithmetic circuits are logic circuits which are capable of performing simple arithmetic operations such as addition.

Adders: A hardware circuit for performing addition is an essential part of every CPU; Below is a truth-table for 1-bit addition along with a circuit known as the half adder;

There are two outputs present: the sum of the inputs, A and B , and the carry to the next (leftward) position

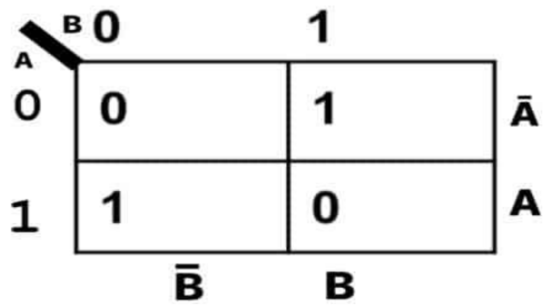
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Designing Half Adder

With the help of the Truth Table, We can design a Karnaugh Map or K-Map for Half Adder to obtain a Boolean Expression.

Karnaugh Map for Sum:

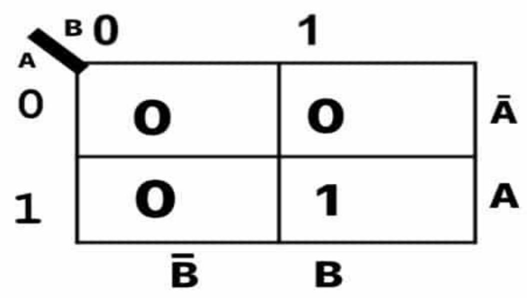


By solving this,

S= A'B + AB'

$S = A \oplus B$

Karnaugh Map for Carry:



By looking at the K-map, We can conclude;

$C = A \cdot B$

This Boolean expression helps us to design a half adder with an XOR Gate and AND gate.

The operation of Half Adder is limited because it can only add two-bit binary digits. But in practical applications, we need to add three or more bits.

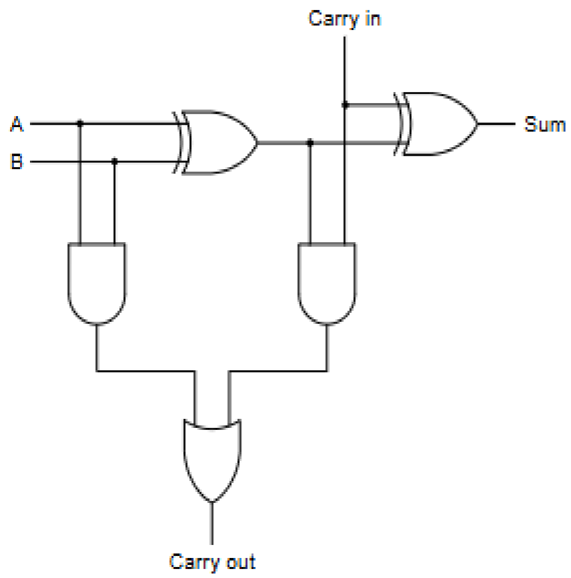
This inability of the circuit puts a limitation on its use. Half-adders were used in early microprocessors and basic digital circuits.

The Full Adder

The truth table and circuit for the full adder will be given next.

A	B	Carry in	Sum	Carry 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

(a)



(b)

How does the Full Adder Work?

From inspection, a full adder is built up from two half adders

The Sum output line is 1 if an odd number of A, B and the Carry in are 1

The Carry out is 1 if either A and B are both 1 (left input to the OR gate) or exactly one of them is 1 and the Carry in bit is also 1.

Designing Full Adder

With the help of the Truth Table, We can design a Karnaugh Map or K-Map for Full Adder to obtain a Boolean Expression.

Karnaugh Map for Sum of Full Adder:

	$\bar{B}\bar{D}$	$\bar{B}D$	BD	$B\bar{D}$	
$\bar{A}$	0	1	0	1	0
A	1	0	1	0	1
	00	01	11	10	

By Solving this,

Step 1: The Original Expression

The original expression is:

$$S = (\overline{A}\,\overline{B}D) + (A\,\overline{B}\,\overline{D}) + (ABD)$$

The aim is to simplify this using **EX-NOR** and **EX-OR** operations.

Step 2: Grouping Terms

The terms are grouped and factored as:

$$S = D(\overline{A}\,\overline{B} + AB) + \overline{D}(A\,\overline{B} + \overline{A}B)$$

Here:

- $\overline{A}\,\overline{B} + AB$ is a standard **EX-NOR** operation.
- $A\,\overline{B} + \overline{A}B$ is a standard **EX-OR** operation.

Step 3: Logic Operations

The terms are rewritten using the EX-NOR and EX-OR definitions:

- **EX-NOR:** $A \odot B = \overline{A}\,\overline{B} + AB$
- **EX-OR:** $A \oplus B = A\overline{B} + \overline{A}B$

So:

$$S = D \cdot (A \odot B) + \overline{D} \cdot (A \oplus B)$$

Step 4: Changing EX-NOR to EX-OR

The EX-NOR operation ($A \odot B$) can also be rewritten in terms of EX-OR ($A \oplus B$) as:

$$\overline{(A \oplus B)} = A \odot B$$

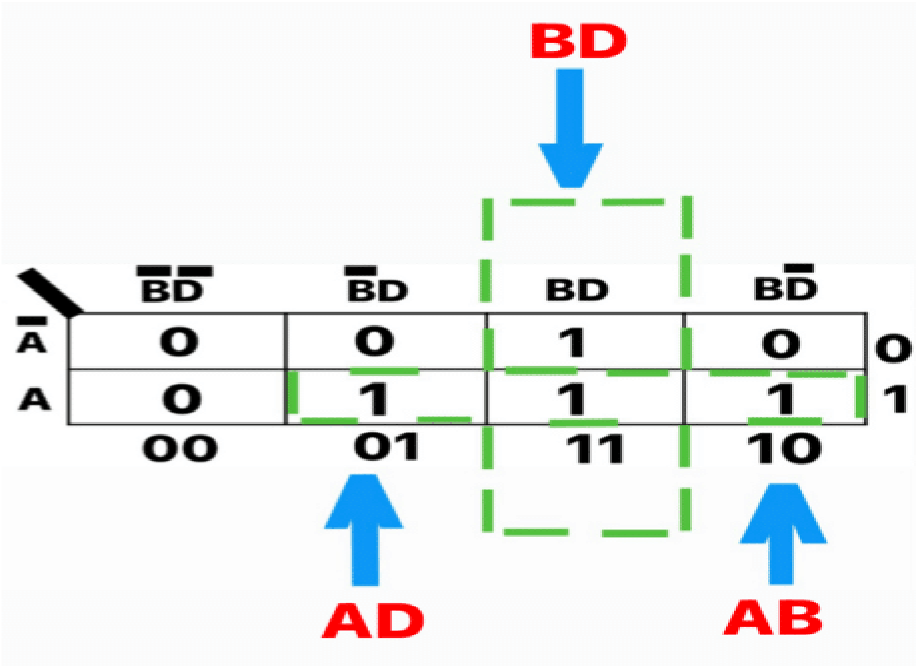
Substituting this back, the equation becomes:

$$S = D \cdot \overline{(A \oplus B)} + \overline{D} \cdot (A \oplus B)$$

the whole equation becomes an EX-OR operation;

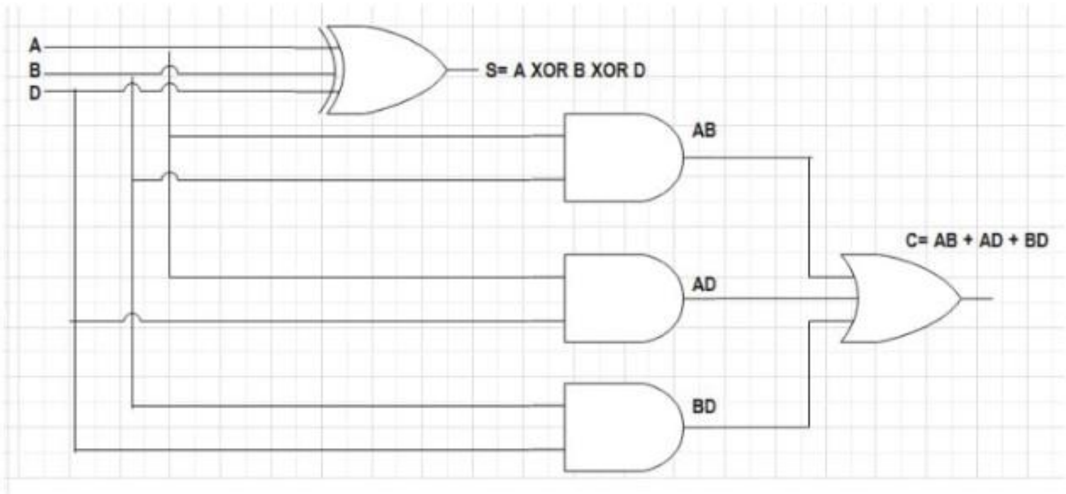
S= D $\oplus$ A $\oplus$ B

Karnaugh Map for Carry:



The equation becomes,

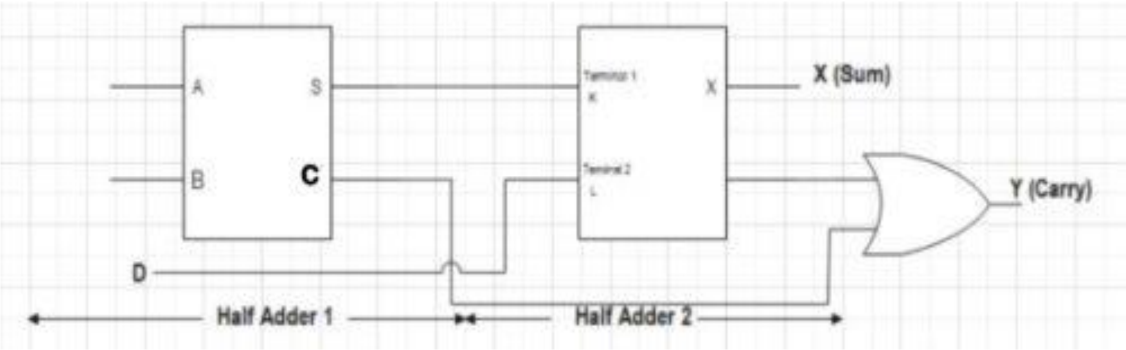
C= AB + AD + BD



Designing Full Adder using two Half adders

The inputs of the first half adder are two single binary digits A and B. The output of the first half adder sum S is fed to the input of the second half adder. The sum output of the second half adder is obtained across X.

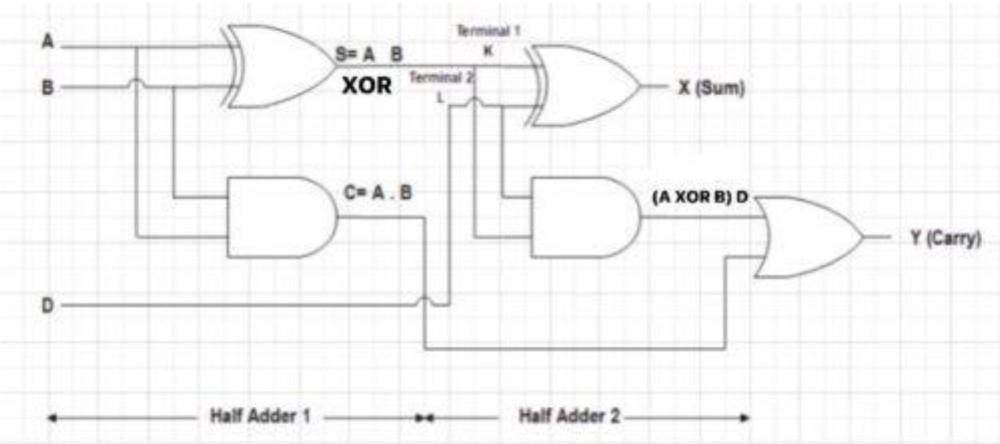
The carry bit D is directly applied across terminal 2 on L of the second half adder. The carry output is obtained across Y of the second half adder.



Operation:

The Boolean expression for Output Sum at S of the first half adder

$S= A \oplus B$



The output across the second half adder for Sum X is a direct EXOR operation of S of the first half adder at input terminal 1 and the carry bit at input terminal 2 of the second half adder

$$S = A \oplus B \oplus D$$

Hence, the expression for the sum of two half adders is the same as the sum of the Full Adder circuit.

The Boolean expression for Output Carry at Y is an OR operation of the output carry of both the first and second-half adders.

$$Y = (A \oplus B)\, D + AB$$

For this, we have to solve the Boolean expression.

By circuit diagram,

Y= (A'B + AB') D + AB

Y= A'BD + AB'D + AB

By inserting the Boolean Law of OR,

1+D= D

Y= A'BD+ AB'D + AB (1+D)

Y= A'BD+ AB'D + AB + ABD

Y= BD (A'+A) + AB'D + AB

By using the Boolean Law of OR,

A+A'= 1

Y= BD + AB’D + AB

By inserting the Boolean Law of OR,

1+D= 1

Y= BD + AB’D + AB (1+D)

Y= BD + AB’D + AB + ABD

Y= BD + AD (B’+B) + AB

By using the Boolean Law of OR,

B+B’= 1

Y= BD + AD + AB

The expression for Carry of two half adders is the same as that of a Full Adder.

Hence, **when two half-adders are connected, they behave as full adders** and perform all the operations of a full adder.

Half Subtractor and Full Subtractor

Subtractor circuits take two binary numbers as input and subtract one binary number input from the other binary number input. Similar to adders, it gives out two outputs, difference and borrow (carry-in the case of Adder). There are two types of subtractors.

1. Half Subtractor
2. Full Subtractor

1) Half Subtractor

The half-subtractor is a combinational circuit which is used to perform subtraction of two bits. It has two inputs, A (minuend) and B (subtrahend) and two outputs Difference and Borrow. The logic symbol and truth table are shown below.

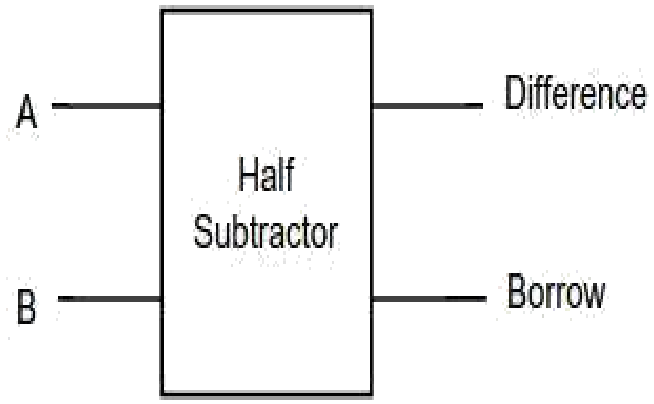
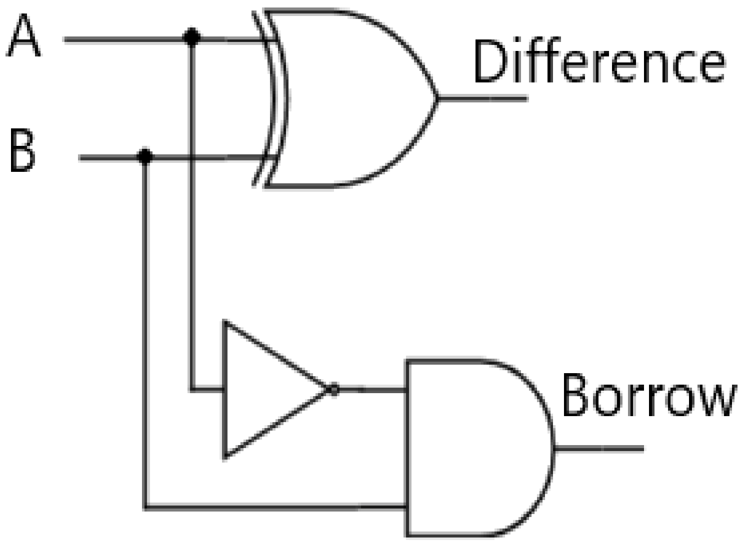


Figure-1:Logic Symbol of Half subtractor

Inputs		Outputs	
A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Figure-2:Truth Table of Half subtractor

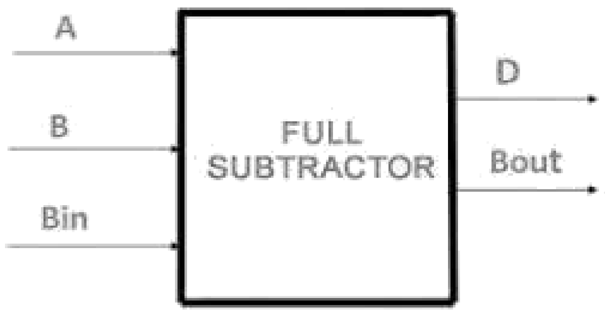


From the above truth table we can find the boolean expression.

Difference = A ⊕ B
Borrow = A' B

2) Full Subtractor

A full subtractor is a combinational circuit that performs subtraction involving three bits, namely A (minuend), B (subtrahend), and Bin (borrow-in) . It accepts three inputs: A (minuend), B (subtrahend) and a Bin (borrow bit) and it produces two outputs: D (difference) and Bout (borrow out). The logic symbol and truth table are shown below.



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

Figure-5:Truth Table of Full subtractor

From the above truth table we can find the boolean expression.

$$D = A \oplus B \oplus Bin$$
$$Bout = A'\ Bin + A'\ B + B\ Bin$$

From the equation we can draw the Full-subtractor circuit as shown in the figure