# EXPERIMENT NO 02

## AIM

To study of Half and Full adder.

## THEORY

Digital computers perform a variety of information-processing tasks. Among the basic functions encountered are various arithmetic operations. The most basic arithmetic operations are, no doubt, the addition and subtraction of binary digits (bit).

## HALF ADDER

The possible operations, when we want to add only two bits, would be the

followings:

0 + 0 = 0 0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 & Carry 1

Above mentioned operation could be performed by a **Half Adder** circuit.

## Truth Table 1: HALF ADDER

We arbitrarily assign symbols **A** and **B** to the two inputs and **S** (for sum) and **Cout** (for Carry) to the two outputs. Truth table for Half Adder as shown below:-

Table 1: Truth Table for Half Adder operation

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | **Output** | |
| **A** | **B** | **S** | **Cout** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

The simplified Boolean function for the two outputs can be written from this truth table as:-

## S = A'.B +A.B'

**Cout = A.B**

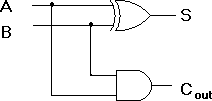
The circuit diagram for the Half Adder to implement above mentioned Boolean function could be quite a few. We will however verify only one.

Figure 1: Half Adder using AND and XOR gates.

## FULL ADDER

We know that in practice, all addition operations must take into account the Carry bit (or digit) from the previous operation. Adders in digital computers also take into account the Carry bit from last operation and add it with the Augend and Addend bits of the present operation to complete the addition operation. The possible operations are:

0 + 0 + 0 (carry) = 0

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

0 + 1 + 1 (carry) = 0 & carry 1 (to be added to next higher digit)

1 + 1 + 0 (carry) = 0 & carry 1 (to be added to next higher digit) 1 + 1 + 1 (carry) = 1 & carry 1 (to be added to next higher digit)

The adder that performs the addition of three bits (two significant bits and a previous carry) is called a **Full Adder**.

As mentioned in the beginning, a full-adder is a combinational circuit that forms the arithmetic sum of three input bits (two significant bits and a previous carry bit) and two output bits. We arbitrarily assign symbols **A** and **B** to the two significant bit inputs and **Cin** for the Carry from the previous lower significant position, and **S** (for sum) and **Cout** (for Carry) to the two outputs.

Truth table for the Full Adder is shown below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | | | **Output** | |
| **A** | **B** | **Cin** | **S** | **Cout** |
| 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 |

Simplified Boolean function for the two outputs can be written from this truth table as:-

**S = A ** **B ** **Bin CC Cout = (A ** **B) C +A.B**

The circuit diagram for the Full Adder is as under:

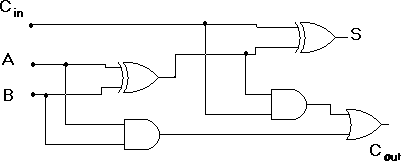


Figure 2: Full Adder comprising of two Half Adders and an OR gate