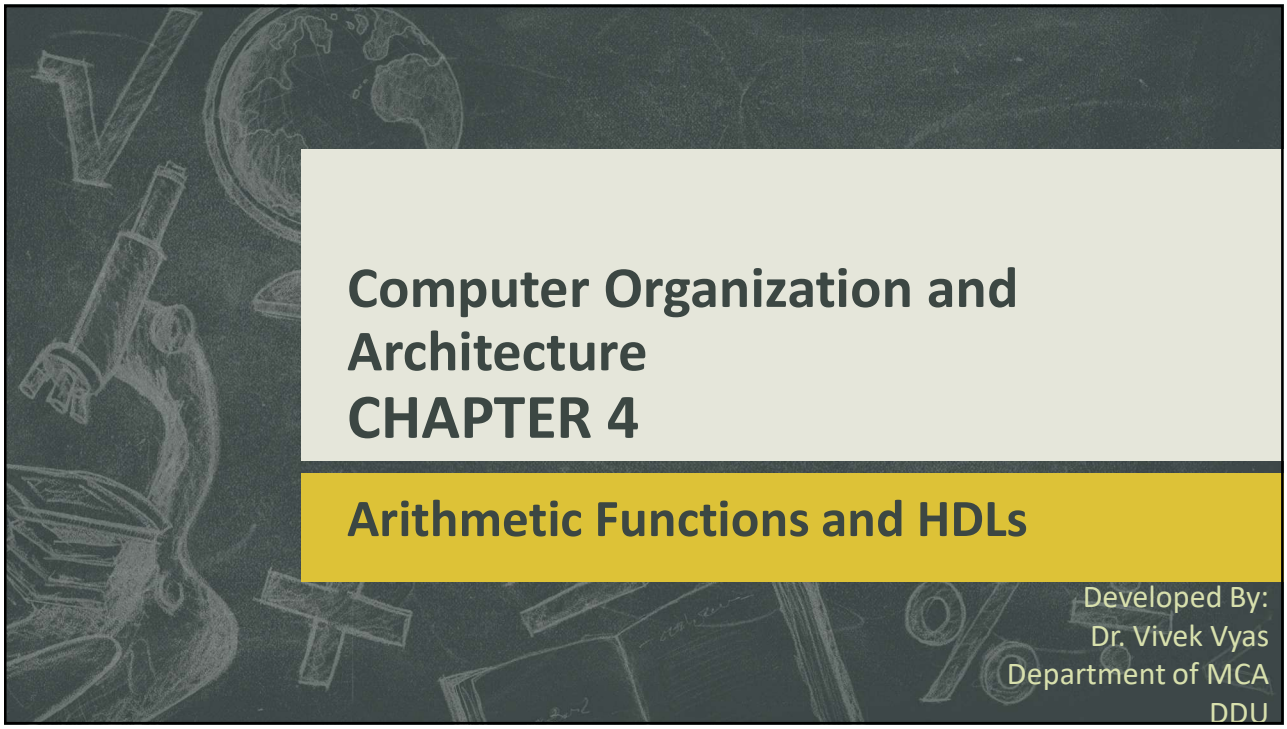


Computer Organization and Architecture
CHAPTER 4

Arithmetic Functions and HDLs

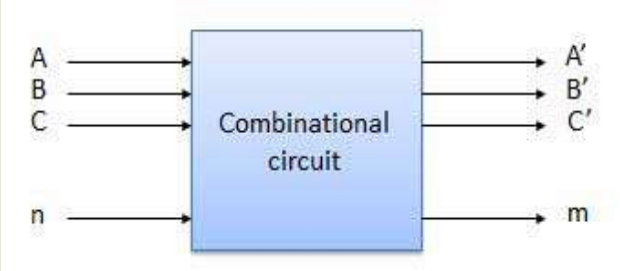
Developed By:
Dr. Vivek Vyas
Department of MCA
DDU

1. ARITHMETIC LOGIC UNIT

- An **arithmetic logic unit (ALU)** is a digital circuit used to perform arithmetic and logic operations. It represents the fundamental building block of the **central processing unit (CPU)** of a computer. Modern CPUs contain very powerful and complex ALUs. In addition to ALUs, modern CPUs contain a control unit (CU).
- Most of the operations of a CPU are performed by one or more ALUs, which load data from input registers. A **register** is a small amount of storage available as part of a CPU. The control unit tells the ALU what operation to perform on that data, and the ALU stores the result in an output register. The control unit moves the data between these registers, the ALU, and memory.

1. ARITHMETIC LOGIC UNIT

- An ALU performs basic arithmetic and logic operations. Examples of arithmetic operations are addition, subtraction, multiplication, and division. Examples of logic operations are comparisons of values such as NOT, AND, and OR.

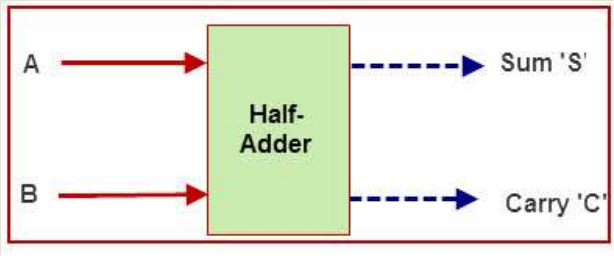


2. BINARY ADDERS

- Combinational circuit that performs the addition of two bits is called a *half adder*.
- One that performs the addition of three bits is called a *full adder*.

❑ Half Adder

- By using half adder, you can design simple addition with the help of logic gates.
- Let's see an addition of single bits.



❑ 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

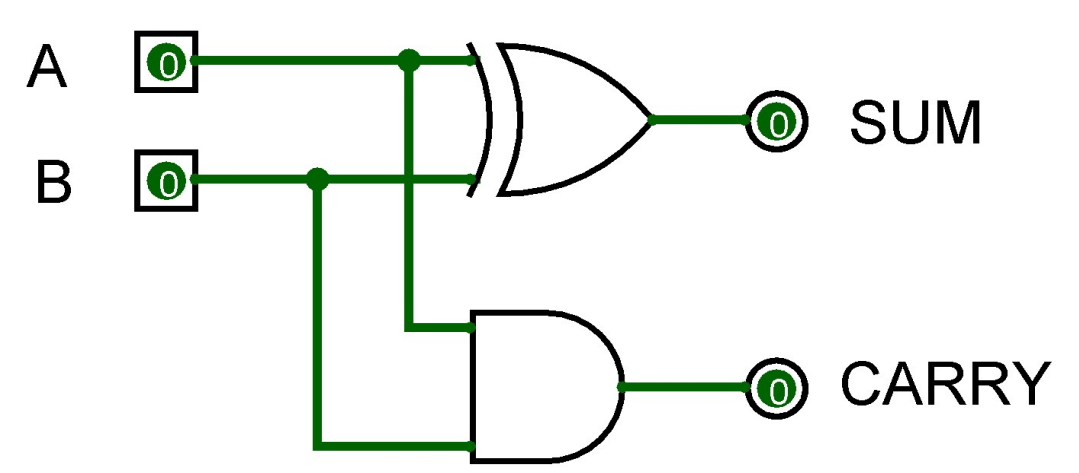
❑ Half Adder

- Half Adder Truth Table

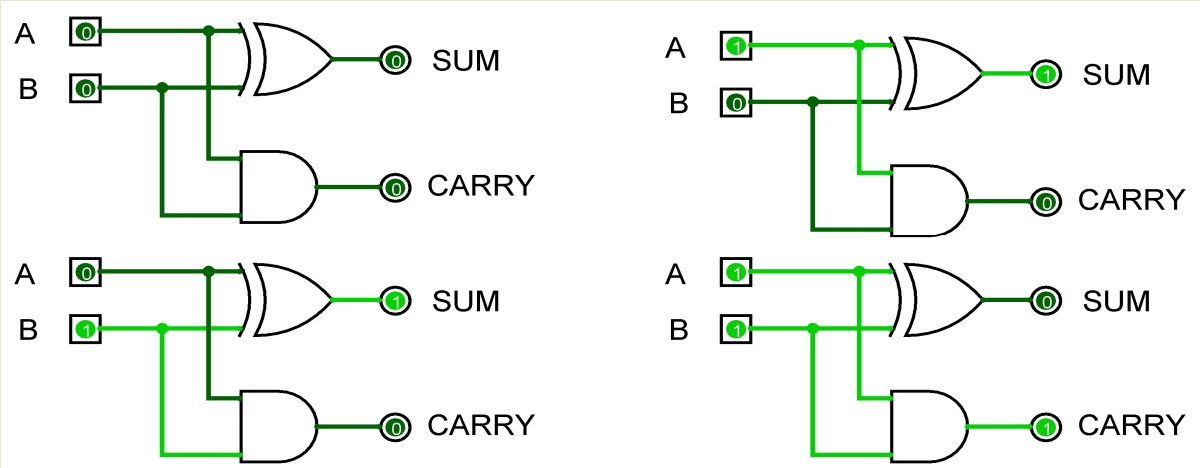
TURTH TABLE			
INPUT		OUTPUT	
A	B	S = A ⊕ B	C = AB
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

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'.

❑ Half Adder

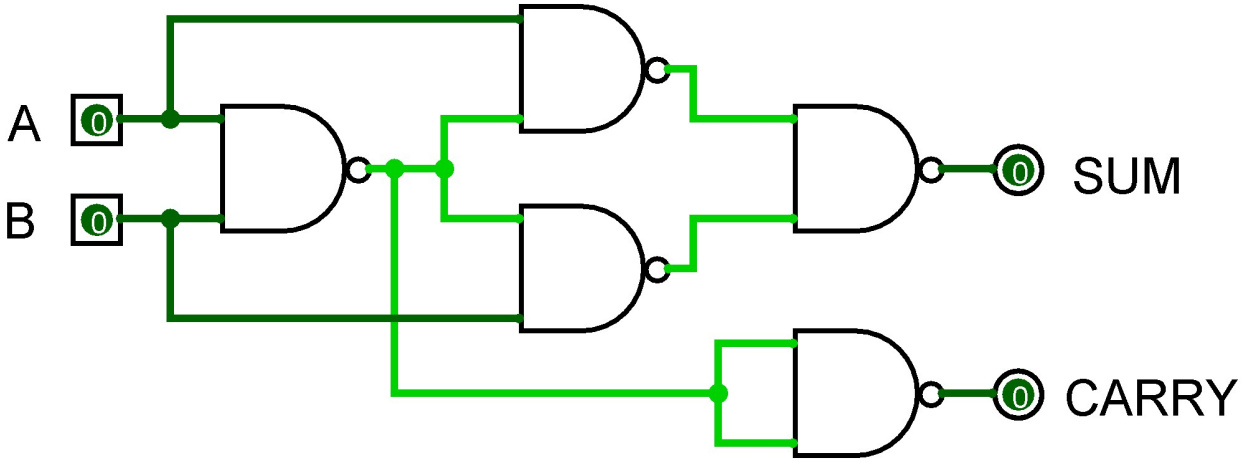


❑ Half Adder



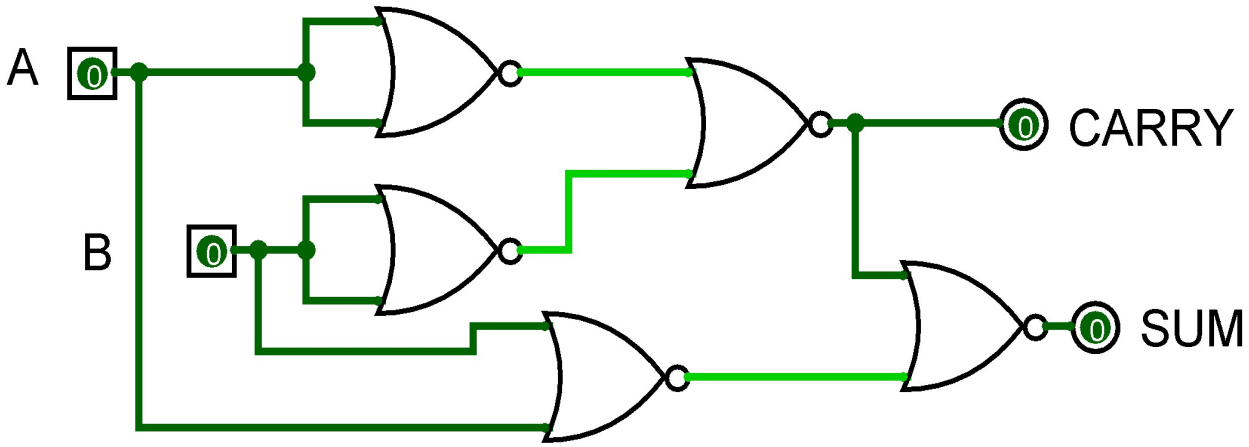
❑ Half Adder

- **Half Adder Using NAND Gate**



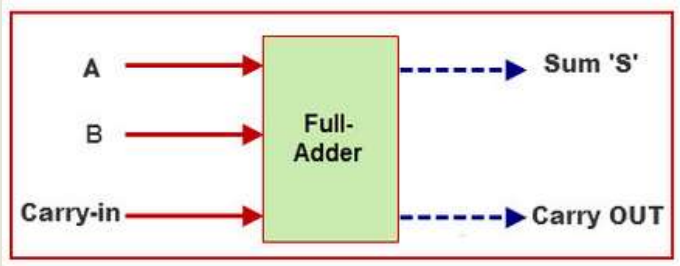
❑ Half Adder

- **Half Adder Using NOR Gate**



❑ 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.



❑ Full Adder

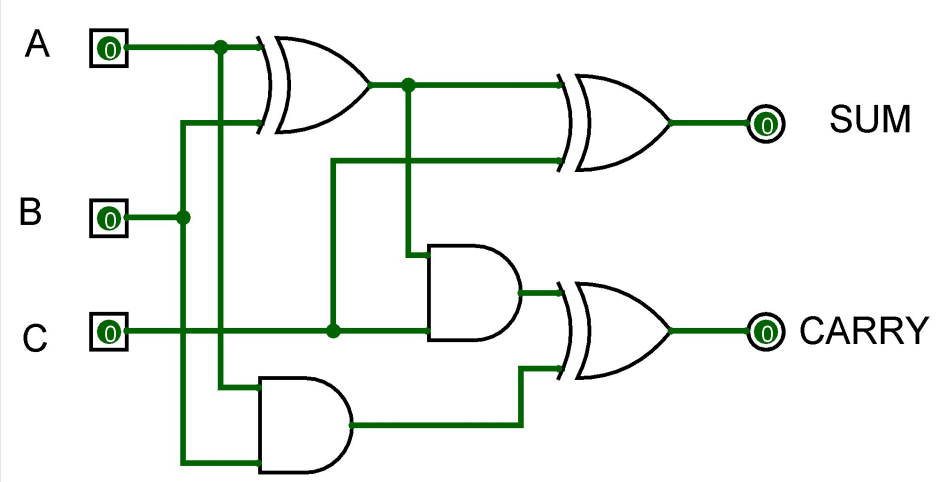
- Full Adder Truth Table

TURTH TABLE				
INPUT			OUTPUT	
A	B	C	S = A ⊕ B	C = AB + BC + AC
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

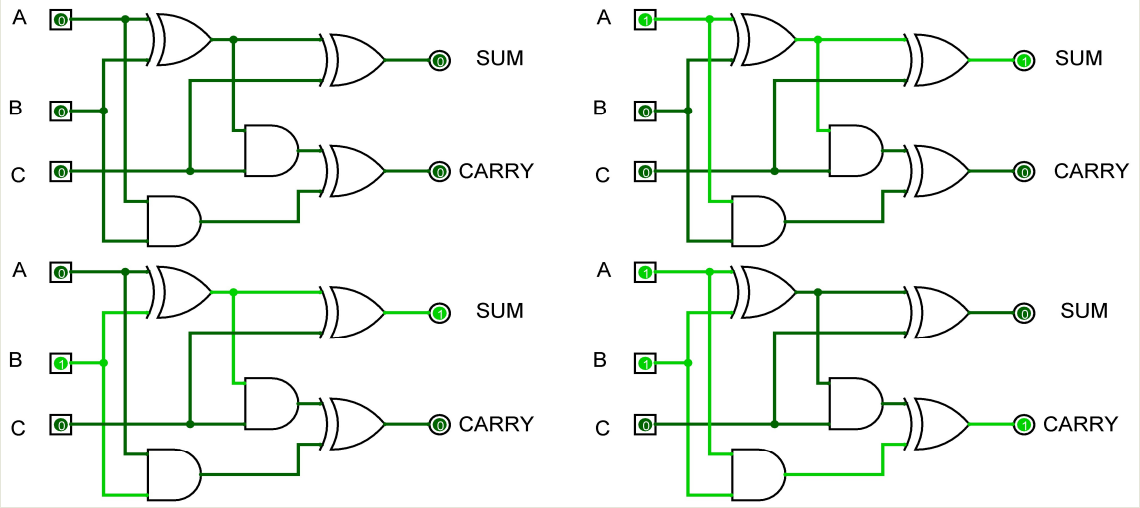
❑ Full Adder

- 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.

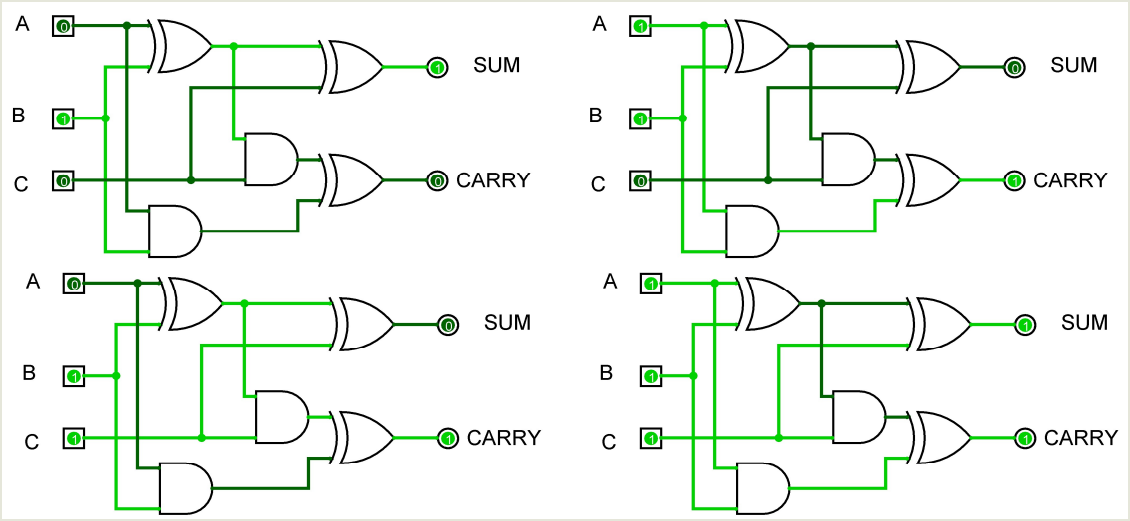
❑ Full Adder



❑ Full Adder

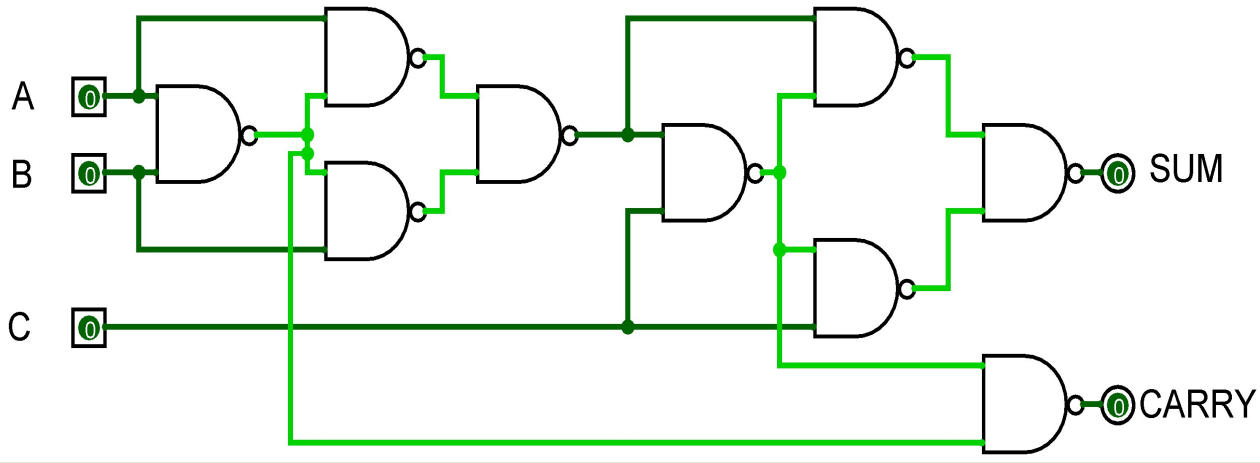


❑ Full Adder



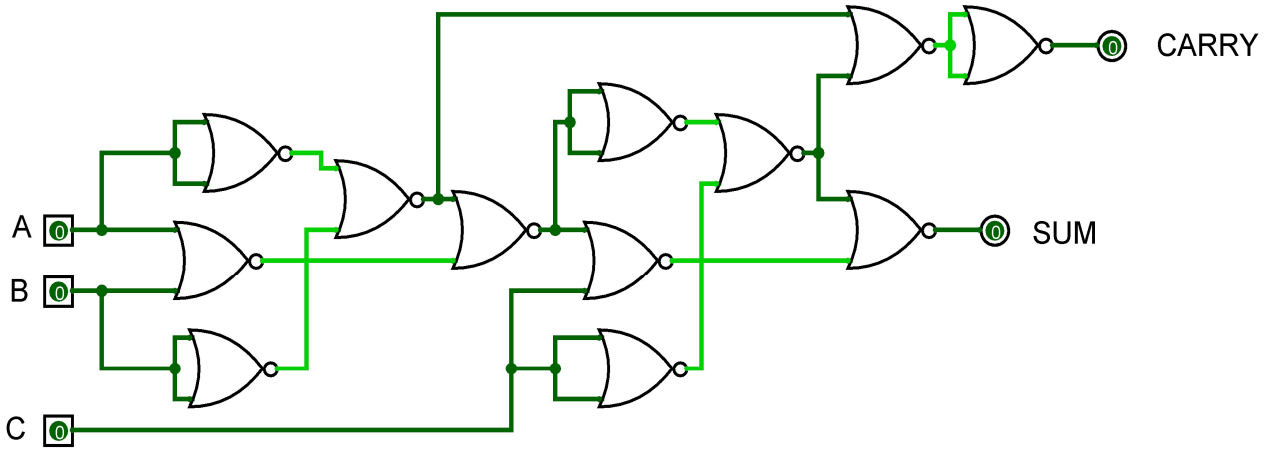
❑ Full Adder

- **Full Adder Using NAND Gate**



❑ Full Adder

- **Full Adder Using NOR Gate**

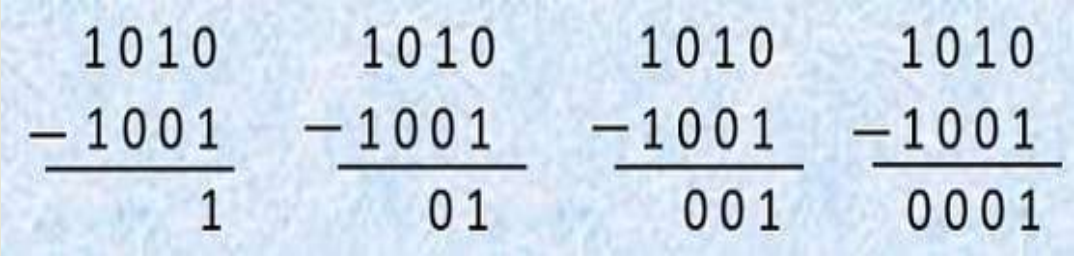


❑ Half Subtractor

- The **Half Subtractor** is a digital circuit which processes the subtraction of two 1-bit numbers.
- The circuit of Half subtractor consists of two inputs and two outputs.
- The word **"HALF"** before the subtractor signifies that it deals with only **two 1-bit numbers**, it has nothing to do with the borrow from the previous stage.

❑ Half Subtractor

- **Procedure of Subtraction**



❑ Half Subtractor

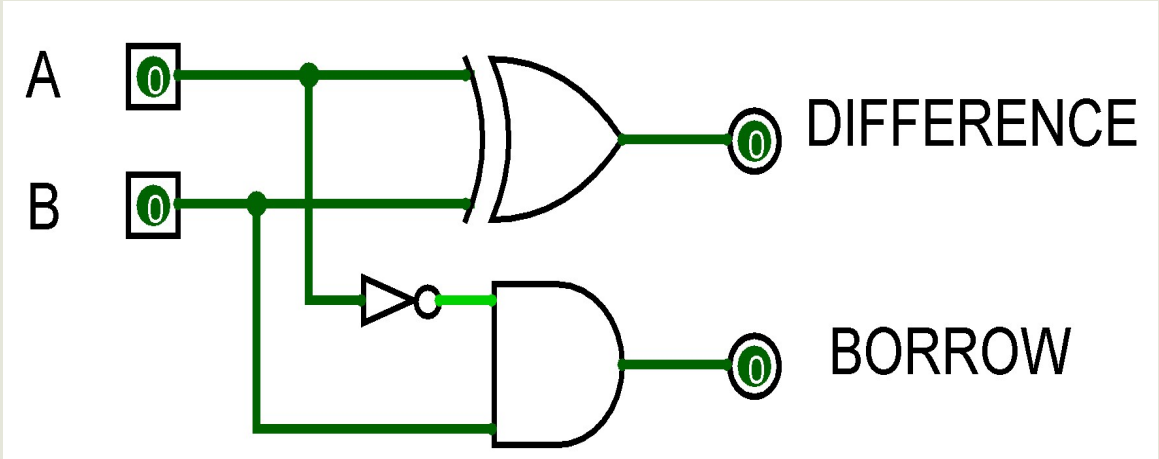
- **Truth Table of Half Subtractor**

TURTH TABLE			
INPUT		OUTPUT	
A	B	D = A ⊕ B	B = AB
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

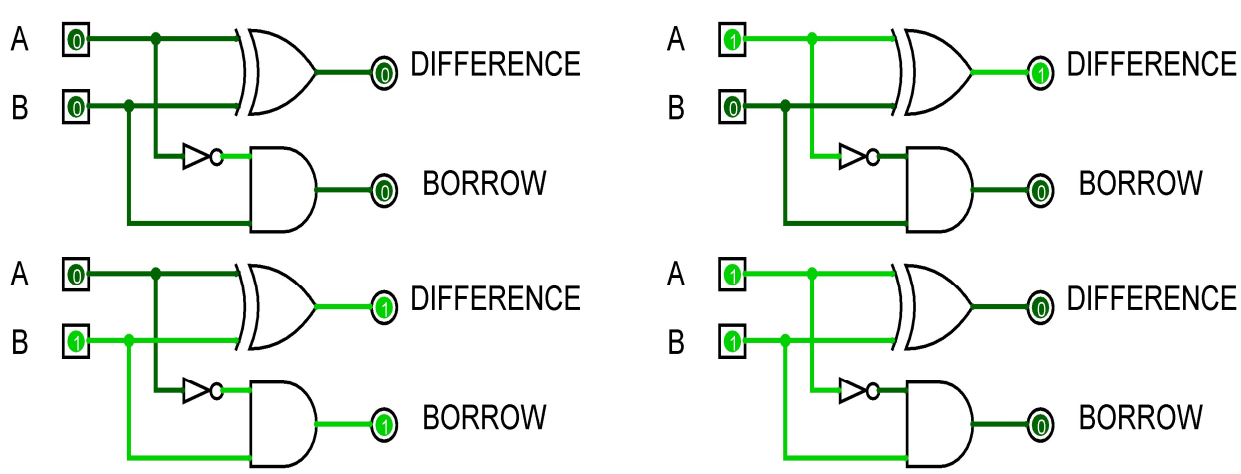
❑ Half Subtractor

- If we observe carefully, it is quite evident that the difference operation performed by half subtractor is exactly similar to the operation of EX-OR gate.
- Thus, we can easily utilize the EX-OR gate for generating difference bit.
- Similarly, the borrow generated by half subtractor can be easily obtained by using the combination of NOT gate and AND gate.

❑ Half Subtractor

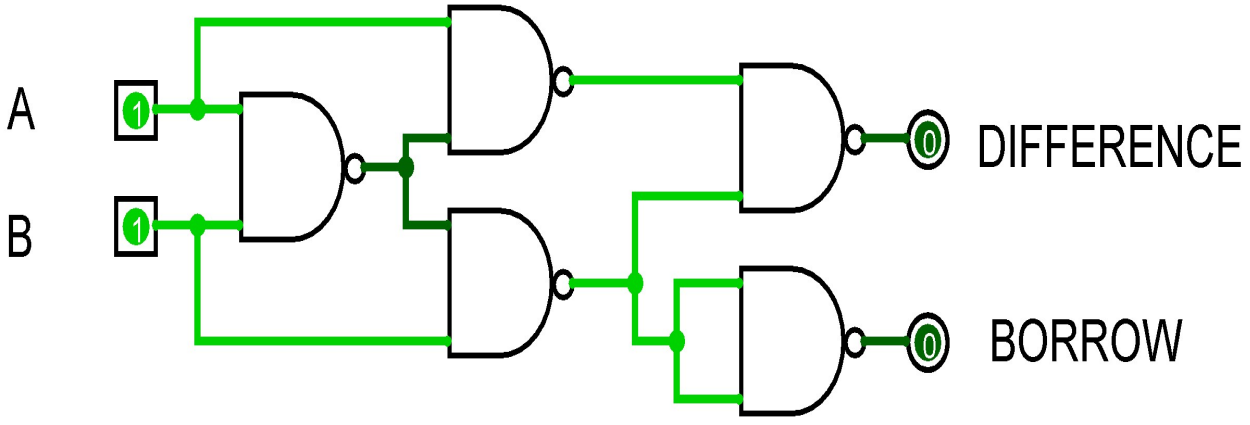


❑ Half Subtractor



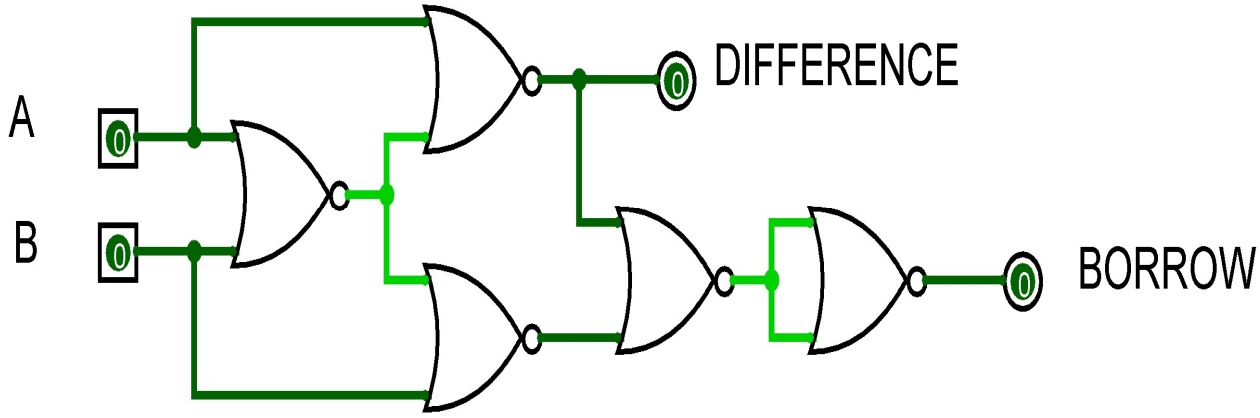
❑ Half Subtractor

- **Half Subtractor using NAND gates**



❑ Half Subtractor

- **Half Subtractor using NOR gates**



❑ Full Subtractor

- A full subtractor is a combinational circuit that performs a subtraction between two bits, taking into account borrow of the lower significant stage.
- This circuit has three inputs and two outputs. The three inputs are A, B and C.
- The two outputs, D and B represent the difference and output borrow, respectively.

❑ Full Subtractor

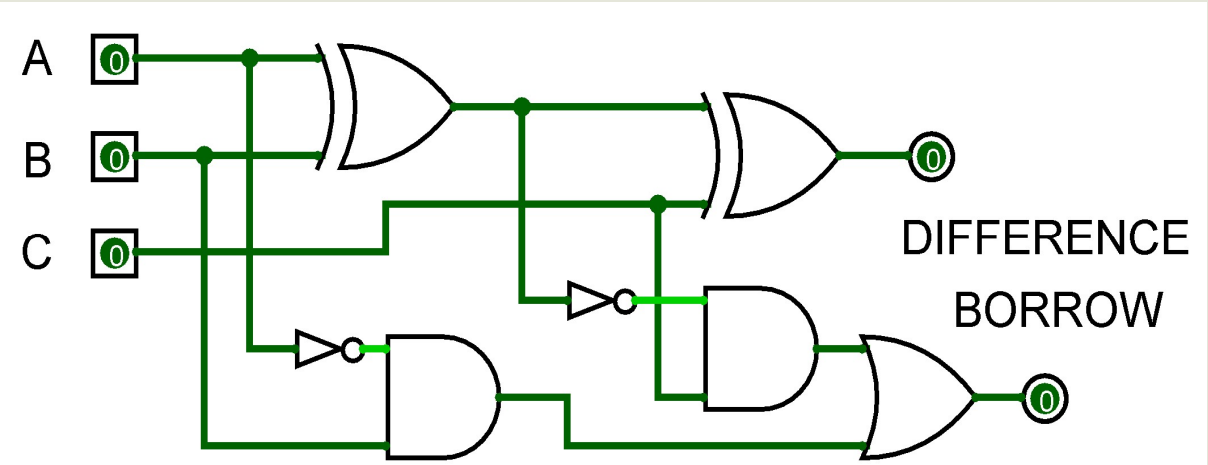
- **Truth Table of Full Subtractor**

INPUT			OUTPUT	
A	B	Bin	D	Bout
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

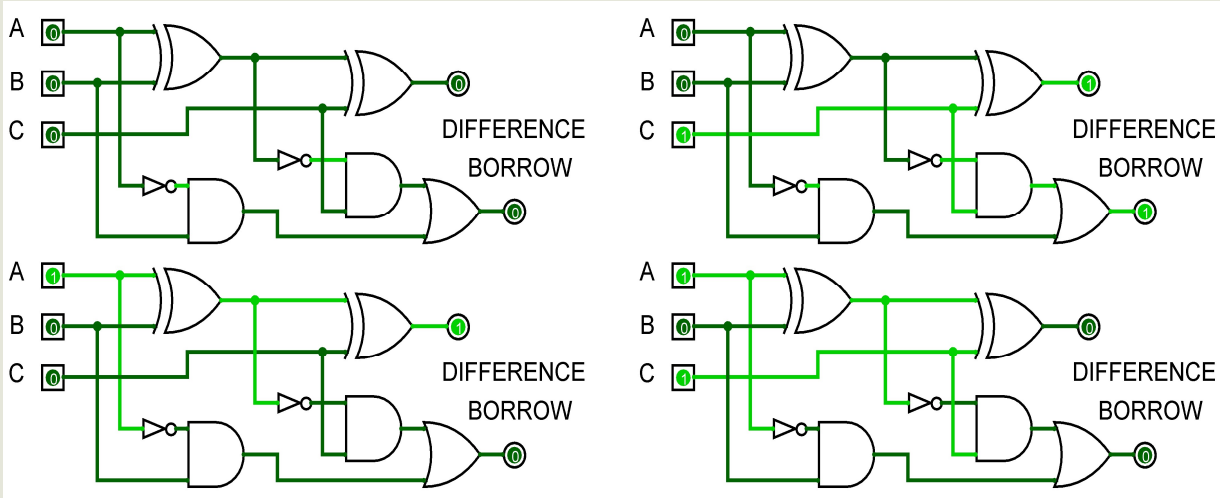
❑ Full Subtractor

- The construction of **full subtractor circuit diagram** involves two half subtractor joined by an OR gate.
- The two borrow bits generated by two separate half subtractor are fed to the OR gate which produces the final borrow bit.
- The final difference bit is the combination of the difference output of the first half adder and the next higher order pair of bits.

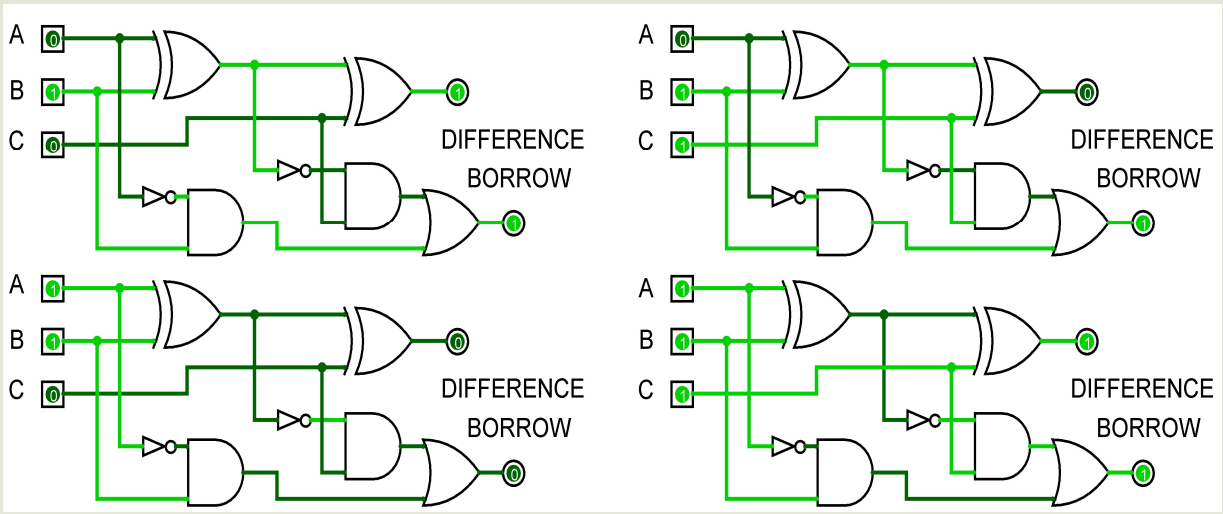
❑ Full Subtractor



❑ Full Subtractor

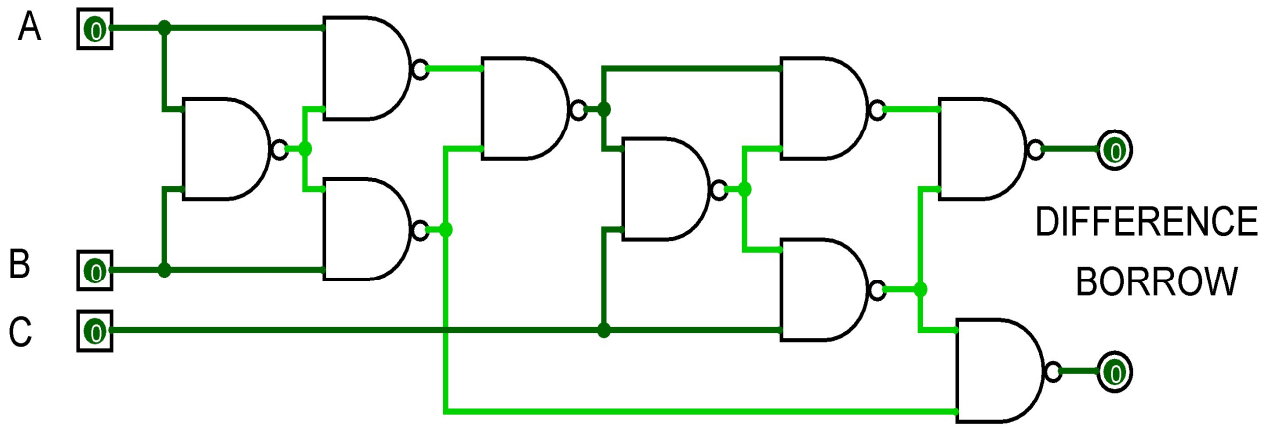


❑ Full Subtractor



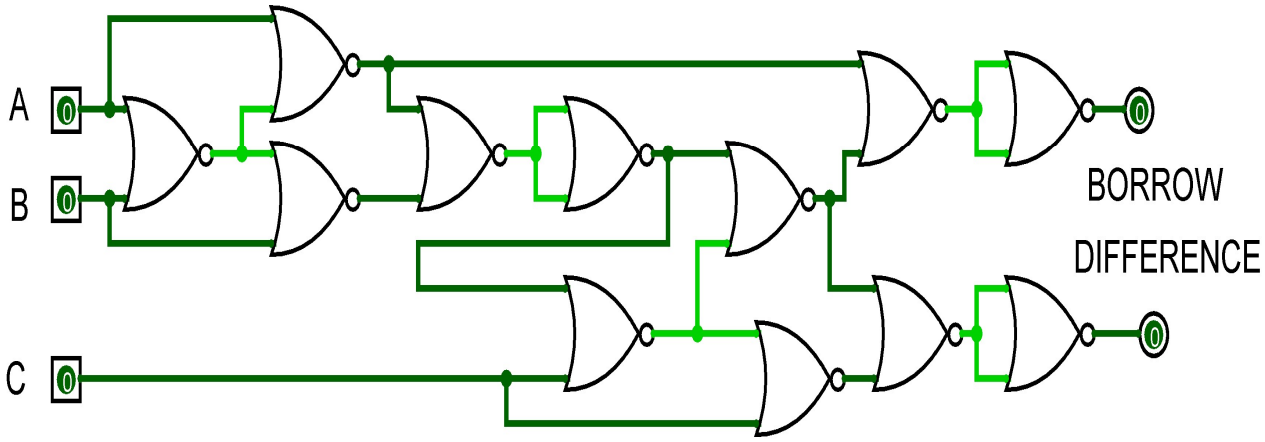
❑ Full Subtractor

- **Full Subtractor using NAND gates**



❑ Full Subtractor

- **Full Subtractor using NOR gates**

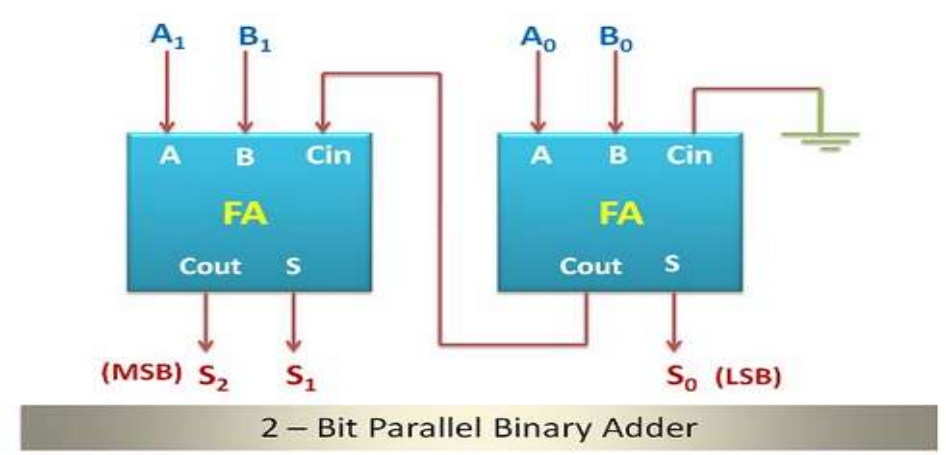


❑ Binary Parallel adder

- The **Parallel binary adder** is a **combinational circuit** consists of various full adders in parallel structure so that when more than 1-bit numbers are to be added, then there can be full adder for every column for the addition.
- The number of full adders in a parallel binary adder depends on the number of bits present in the number for the addition.
- If 4-bits numbers are to be added, then there will be 4-full adder in the parallel binary adder.

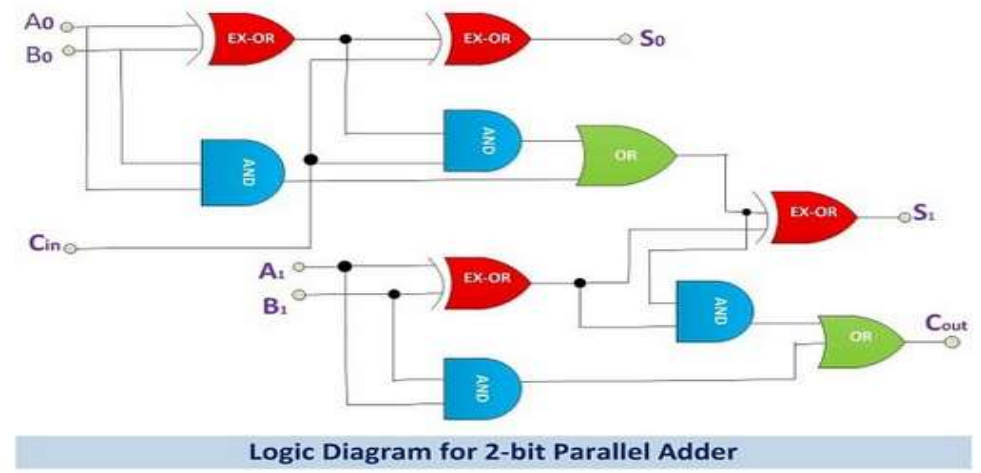
❑ Binary Parallel adder

- **Working of Binary Parallel adder**



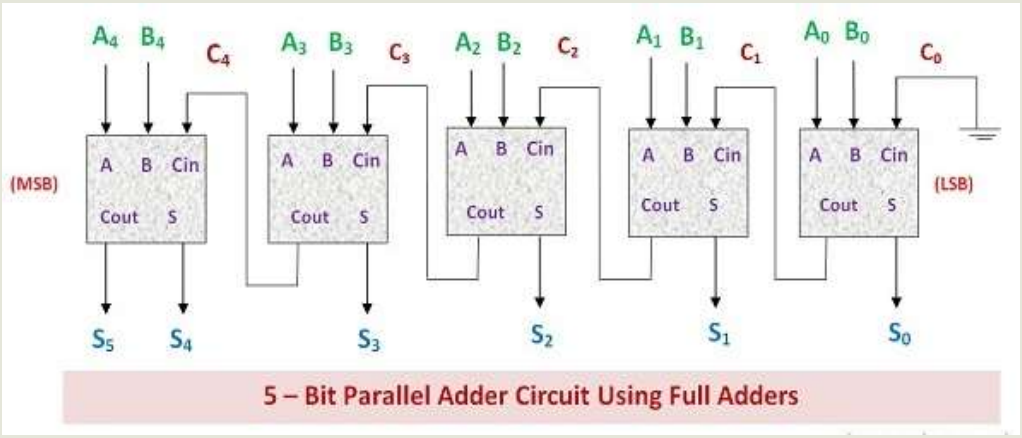
❑ Binary Parallel adder

- **Logic circuit of 2-Bit Parallel Binary Adder**



❑ Binary Parallel adder

- **Working of 5 bit Binary Parallel adder**



5 – Bit Parallel Adder Circuit Using Full Adders

❑ Decimal Adder and BCD Adder

- Decimal Adder – The digital systems handles the decimal number in the form of binary coded decimal numbers (BCD).
- A BCD adder is a circuit that adds two BCD digits and produces a sum digit also in BCD.
- BCD numbers use 10 digits, 0 to 9 which are represented in the binary form 0 0 0 0 to **1 0 0 1,** i.e. each BCD digit is represented as a 4-bit binary number.
- Here, we should note that BCD cannot be greater than 9.
- When we write BCD number say 526, it can be represented as

5	2	6
↓	↓	↓
0 1 0 1	0 0 1 0	0 1 1 0

❑ Decimal Adder and BCD Adder

- If the sum of two number is less then or equal to 9, then the value of BCD sum and binary sum will be same otherwise they will differ by 6(0110 in binary).
- Now, lets move to the table and find out the logic when we are going to add "0110".

❑ Decimal Adder and BCD Adder

- **Truth Table Decimal Adder**
- **/ BCD Adder**

Decimal	Binary Sum					BCD Sum				
	C'	S3'	S2'	S1'	S0'	C	S3	S2	S1	S0
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0
11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0
19	1	0	0	1	1	1	1	0	0	1

❑ Decimal Adder and BCD Adder

We are adding "0110" (=6) only to the second half of the table.

The conditions are:

1. If C' = 1 (Satisfies 16-19)
2. If S3′.S2′ = 1 (Satisfies 12-15)
3. If S3′.S1′ = 1 (Satisfies 10 and 11)

So, our logic is

```
C' + S3'.S2' + S3'.S1' = 1
```

❑ Decimal Adder and BCD Adder