
Orange
Digital Center

# Orange Digital Center

# Digital Design and FPGA Flow

## Assignment 1
## Week 1

**Created By: Ahmed Mostafa Gomaa**
**Submitted to :**
**Dr. Tamer Saleh**
**Eng. Asmaa El-sayed**
**Eng. Nourhan**

# Table of Contents

**1. Introduction ..... 3**

**2. 4 bit Carry Lock ahead ..... 3**

2.1. Introduction ..... 3
2.2. Design of the 4-Bit Carry Lookahead Adder ..... 3
2.3. How the 4-Bit CLA Works ..... 4
2.4. Advantages of the 4-Bit CLA ..... 4
2.5. Disadvantages of the 4-Bit CLA ..... 4
2.6. Applications of the 4-Bit CLA ..... 4
2.7. Conclusion ..... 4
2.8. RTL Viewer ..... 5

**3. 4 bit Ripple Carry ..... 5**

3.1. Introduction ..... 5
3.2. Design of the 4-Bit Ripple Carry Adder (RCA) ..... 5
3.3. How the 4-Bit Ripple Carry Adder Works ..... 5
3.4. Advantages of the 4-Bit RCA ..... 6
3.5. Disadvantages of the 4-Bit RCA ..... 6
3.6. Applications of the 4-Bit RCA ..... 6
3.7. Conclusion ..... 6
3.8. RTL Viewer ..... 6

**4. BCD – Excess 3 ..... 7**

4.1. Introduction ..... 7
*4.1.1. What is BCD? ..... 7*
*4.1.2. What is Excess-3? ..... 7*
4.2. Purpose of BCD to Excess-3 Conversion ..... 7
4.3. Advantages of Excess-3 Code ..... 7
4.4. Applications of Excess-3 Code ..... 7
4.5. Conclusion ..... 7

**5. Half Adder ..... 8**

5.1. Introduction ..... 8
5.2. Purpose of a Half Adder ..... 8
5.3. Applications of Half Adder ..... 8
5.4. Limitations of Half Adder ..... 8
5.5. Circuits and Truth Table ..... 8
5.6. RTL Viewer ..... 9
5.7. Conclusion ..... 9

**6. Full Adder ..... 9**

6.1. Introduction ..... 9
6.2. Purpose of a Full Adder ..... 9
6.3. Applications of Full Adder ..... 9
6.4. Advantages of Full Adder ..... 10
6.5. Circuits and Truth Table ..... 10
6.6. RTL Viewer ..... 10
6.7. Conclusion ..... 10

**7. Full Adder From Half Adder ..... 11**

7.1. Description ..... 11
7.2. Circuits and Truth Table ..... 11
7.3. RTL Viewer ..... 11
7.4. Key Points ..... 11

**8. Referance ..... 12**

# 1. Introduction

In digital electronics, adders are fundamental components used to perform arithmetic operations in processors and other computational circuits. Two common types of adders are the 4-bit Carry Lookahead Adder (CLA), and the 4-bit Ripple Carry Adder (RCA), each with distinct approaches to handling carry propagation. The Ripple Carry Adder is a simpler design where the carry output from each full adder stage propagates sequentially to the next, leading to a delay that increases linearly with the number of bits. While easy to implement, this design suffers from slower performance due to the cumulative carry propagation delay. On the other hand, the Carry Lookahead Adder employs a more advanced technique to predict carry signals in parallel, significantly reducing the delay by eliminating the need for sequential carry propagation. This makes the CLA faster and more efficient for high-speed applications, albeit at the cost of increased circuit complexity. Both adders have their unique advantages and trade-offs, making them suitable for different use cases in digital design. This introduction explores the principles, operation, and key differences between these two 4-bit adder architectures.

# 2. 4 bit Carry Lock ahead

## 2.1. Introduction

The **4-bit Carry Lookahead Adder (CLA)** is a high-speed adder circuit designed to minimize the delay caused by carry propagation in multi-bit addition. Unlike the Ripple Carry Adder (RCA), which propagates the carry sequentially through each bit, the CLA computes carry signals in parallel, significantly reducing the overall delay. This makes the CLA particularly useful in high-performance computing systems where speed is critical.

## 2.2. Design of the 4-Bit Carry Lookahead Adder

The CLA is built using two main components:

1. **Generate (G):** A carry is generated when both input bits are 1 $G_i = A_i \,.\, B_i$
2. **Propagate (P):** A carry is propagated if at least one of the input bits is 1 $P_i = A_i \oplus B_i$ .
3. **Carry Lookahead Logic:**
   The carry for each bit is computed using the generate and propagate signals from previous bits. The carry for the $i^{th}$ bit ($C_i$) is given by:
   $C_i = G_{i-1} + (P_{i-1} \cdot C_{i-1})$

- These equations are implemented using AND-OR logic gates to compute the carry signals in parallel.

4. **Sum Calculation:**
   The sum for each bit ($S_i$) is calculated using the propagate signal and the carry-in:

$$Si = P_i \oplus C_i$$

## 2.3. How the 4-Bit CLA Works

1. **Input Stage:**
   - The 4-bit inputs A ($A_3A_2A_1A_0$) and B ($B_3B_2B_1B_0$) are fed into the circuit.
   - The generate (G*i*) and propagate (Pi) signals are computed for each bit.
2. **Carry Computation:**
   - The carry signals ($C_1$,$C_2$,$C_3$,$C_4$) are computed in parallel using the carry lookahead logic. This eliminates the need to wait for the carry to ripple through each bit, as in the RCA.
3. **Sum Calculation:**
   - The sum bits ($S_0$,$S_1$,$S_2$,$S_3$) are computed using the XOR operation between the propagate signals and the corresponding carry-in signals.
4. **Output Stage:**
   - The final 4-bit sum ($S_3S_2S_1S_0$) and the carry-out ($C_4$) are produced as outputs.

## 2.4. Advantages of the 4-Bit CLA

- **Speed:** The parallel computation of carry signals reduces the overall delay, making the CLA faster than the RCA.
- **Scalability:** The CLA can be extended to larger bit widths (e.g., 16-bit, 32-bit) by grouping 4-bit CLAs and using additional lookahead logic.

## 2.5. Disadvantages of the 4-Bit CLA

- **Complexity:** The CLA requires more hardware (additional gates) compared to the RCA, increasing the design complexity and cost.

- **Power Consumption:** The increased number of gates leads to higher power consumption.

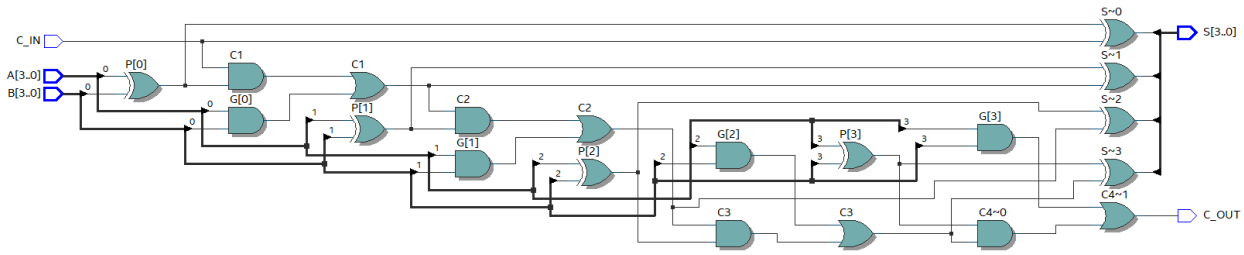
## 2.6. Applications of the 4-Bit CLA

The 4-bit CLA is widely used in high-speed arithmetic logic units (ALUs), microprocessors, and digital signal processors (DSPs), where fast addition is essential for performance.

## 2.7. Conclusion

In summary, the 4-bit Carry Lookahead Adder is a powerful and efficient design that overcomes the limitations of the Ripple Carry Adder by computing carry signals in parallel. While it is more complex to implement, its speed and scalability make it a preferred choice for high-performance digital systems.

## 2.8. RTL Viewer




# 3. 4 bit Ripple Carry

## 3.1. Introduction

The 4-bit Ripple Carry Adder (RCA) is one of the simplest and most fundamental digital circuits used to perform binary addition. It is widely employed in arithmetic logic units (ALUs), microprocessors, and other digital systems where basic addition operations are required. The RCA operates by adding two 4-bit binary numbers along with a carry-in bit, producing a 4-bit sum and a carry-out bit. Despite its simplicity, the RCA serves as a foundational building block for understanding more complex adder designs.

## 3.2. Design of the 4-Bit Ripple Carry Adder (RCA)

The 4-bit Ripple Carry Adder (RCA) is constructed by connecting four full adders (FA) in series. Each full adder is responsible for adding two corresponding bits from the input numbers along with a carry-in from the previous stage. The design is modular and straightforward, making it easy to understand and implement. Below, we break down the design and operation of the 4-bit RCA.

## 3.3. How the 4-Bit Ripple Carry Adder Works

The RCA is constructed by cascading four **full adders (FA)** in series. Each full adder takes three inputs: two bits from the input numbers ($A_i$ and $B_i$) and a carry-in ($C_{i-1}$) from the previous stage. It produces two outputs: a sum bit ($S_i$) and a carry-out ($C_i$), which propagates to the next full adder. The carry signal "ripples" through each stage, hence the name **Ripple Carry Adder**.

**Step-by-Step Operation**

1. **Input Stage:**
   - The 4-bit inputs A ($A_3A_2A_1A_0$) and B ($B_3B_2B_1B_0$) are fed into the circuit.
   - The carry-in ($C_0$) is typically set to 0 unless there is an external carry.
2. **Bit-wise Addition:**
   - Each full adder computes the sum and carry for its corresponding bit position.
   - The carry-out from one full adder becomes the carry-in for the next full adder.
3. **Output Stage:**
   - The sum bits ($S_3S_2S_1S_0$) are produced as the final result.
   - The carry-out from the last full adder ($C_3$) is the final carry-out.

## 3.4. Advantages of the 4-Bit RCA

- **Low Complexity:** The RCA requires minimal hardware, making it cost-effective and easy to implement.
- **Scalability:** It can be extended to handle larger bit-widths by simply adding more full adders.

## 3.5. Disadvantages of the 4-Bit RCA

- **Slow Operation:** The sequential propagation of the carry signal results in a delay that grows linearly with the number of bits, making the RCA slower for large inputs.
- **Performance Limitation:** The ripple carry delay limits its use in high-speed applications.

## 3.6. Applications of the 4-Bit RCA

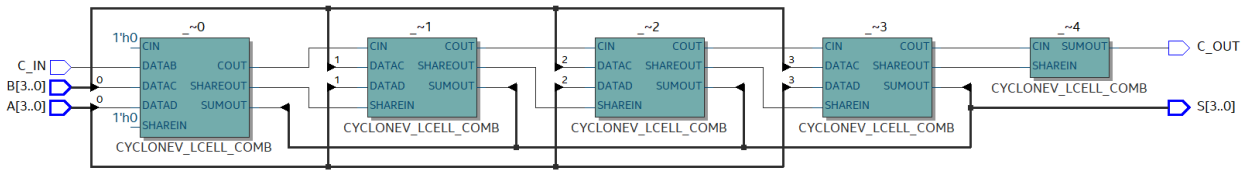
The 4-bit RCA is commonly used in:

- Basic arithmetic circuits.
- Educational tools for teaching digital logic design.
- Low-speed or low-power applications where simplicity is more critical than speed.

## 3.7. Conclusion

In summary, the 4-bit Ripple Carry Adder is a straightforward and widely used circuit for binary addition. While it is not the fastest adder design, its simplicity and modularity make it an essential concept in digital electronics and a stepping stone to understanding more advanced adder architectures like the Carry Lookahead Adder (CLA).

## 3.8. RTL Viewer



# 4. BCD – Excess 3

## 4.1. Introduction

### 4.1.1. What is BCD?

BCD (Binary-Coded Decimal) is a class of binary encodings where each digit of a decimal number is represented by a fixed number of binary bits, usually 4 bits.For example, the decimal number 5 is represented as 0101 in BCD.

### 4.1.2. What is Excess-3?

- **Excess-3 (XS-3)** is a self-complementing binary code used to represent decimal digits. It is derived by adding 3 (binary 0011) to the BCD representation of a digit.
- For example, the decimal number 5 in BCD is 0101. In Excess-3, it becomes 0101 + 0011 = 1000.

## 4.2. Purpose of BCD to Excess-3 Conversion

Excess-3 code is used in digital systems for arithmetic operations, as it simplifies subtraction and complements operations.
This documentation explains the design and implementation of a circuit to convert BCD to Excess-3.

## 4.3. Advantages of Excess-3 Code

- **Self-Complementing:** The 9's complement of a number can be easily obtained by inverting the bits.
- **Simplifies Subtraction:** Excess-3 makes subtraction easier by converting it into addition.

## 4.4. Applications of Excess-3 Code

- **Arithmetic Operations:** Excess-3 simplifies subtraction and complement operations in digital systems.
- **Error Detection:** Excess-3 is a self-complementing code, making it useful for error detection in arithmetic circuits.
- **Digital Displays:** Used in devices like calculators and digital clocks.

## 4.5. Conclusion

The BCD to Excess-3 converter is a fundamental circuit in digital electronics, enabling efficient arithmetic operations and error detection. By following the design and implementation steps outlined in this documentation, you can build a circuit to convert BCD inputs to Excess-3 outputs. This circuit is widely used in applications requiring decimal arithmetic and digital displays.

# 5. Half Adder

## 5.1. Introduction

A Half Adder is a basic digital circuit used to perform the addition of two single-bit binary numbers. It is the simplest form of an adder and serves as a building block for more complex arithmetic circuits like Full Adders and Ripple Carry Adders.

## 5.2. Purpose of a Half Adder

The Half Adder is designed to:

- Add two single-bit binary numbers.
- Produce a **sum** and a **carry** as outputs.

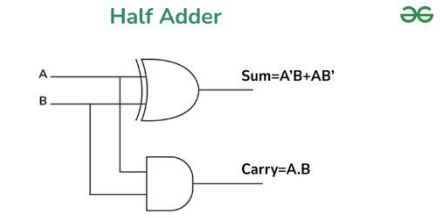
## 5.3. Applications of Half Adder

1. **Building Block for Full Adder:**
   - Half Adders are used to construct **Full Adders**, which can handle carry-in inputs.
2. **Arithmetic Logic Units (ALUs):**
   - Half Adders are used in ALUs to perform basic arithmetic operations.
3. **Digital Counters:**
   - Half Adders are used in digital counters to increment values.
4. **Simple Addition Circuits:**
   - Half Adders are used in circuits where only two single-bit numbers need to be added.

## 5.4. Limitations of Half Adder

1. **No Carry-In Input:**
   - The Half Adder cannot handle a carry-in from a previous addition, making it unsuitable for multi-bit addition.
2. **Limited Functionality:**
   - It can only add two single-bit binary numbers.

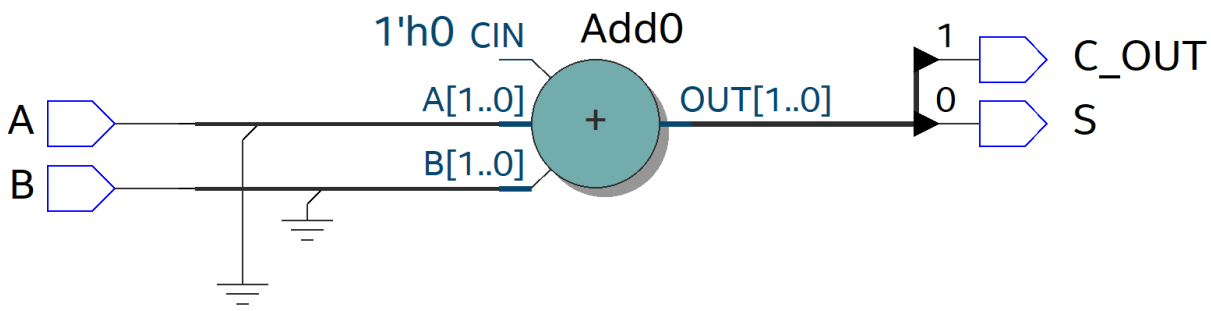
## 5.5. Circuits and Truth Table




Truth Table

| A | B | S | C |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

## 5.6. RTL Viewer




## 5.7. Conclusion

The **Half Adder** is a fundamental digital circuit used to add two single-bit binary numbers. It produces a **sum** and a **carry** as outputs, which are computed using an **XOR gate** and an **AND gate**, respectively. While it is simple and easy to implement, its functionality is limited to single-bit addition. However, it serves as a critical building block for more complex arithmetic circuits like **Full Adders** and **Ripple Carry Adders**.

# 6. Full Adder

## 6.1. Introduction

A **Full Adder** is a digital circuit used to perform the addition of three single-bit binary numbers. It is an extension of the **Half Adder** and can handle a **carry-in** from a previous addition, making it suitable for multi-bit addition.

## 6.2. Purpose of a Full Adder

The Full Adder is designed to:

- Add three single-bit binary numbers (two inputs and a carry-in).
- Produce a **sum** and a **carry-out** as outputs.

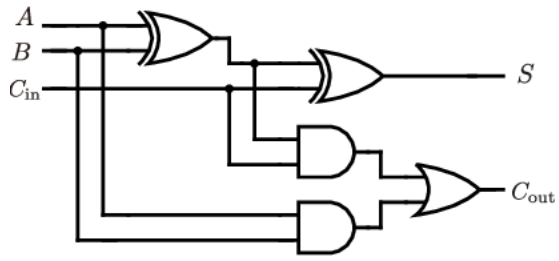
## 6.3. Applications of Full Adder

- **Building Block for Multi-Bit Adders:**
  - Full Adders are used to construct **Ripple Carry Adders** and **Carry Lookahead Adders** for multi-bit addition.
- **Arithmetic Logic Units (ALUs):**
  - Full Adders are used in ALUs to perform arithmetic operations.
- **Digital Signal Processing:**
  - Full Adders are used in digital signal processing circuits for fast addition.
- **Microprocessors:**
  - Full Adders are used in microprocessors for arithmetic and logic operations.

## 6.4. Advantages of Full Adder

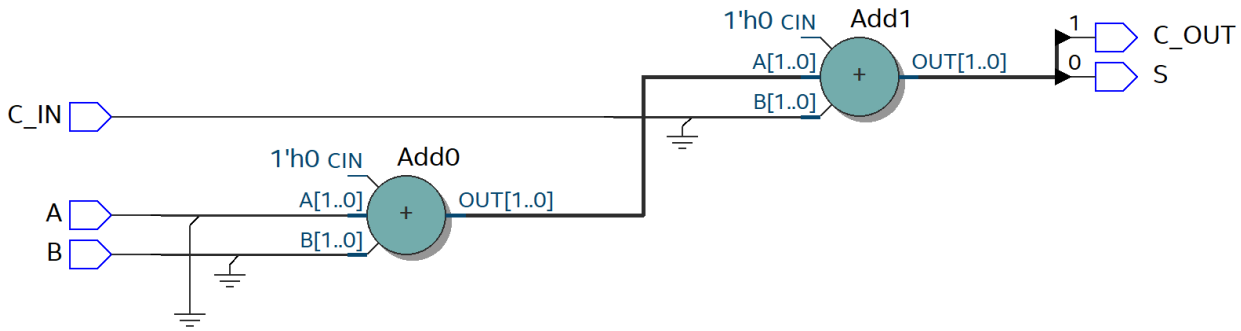
1. **Handles Carry-In:**
   - The Full Adder can handle a carry-in from a previous addition, making it suitable for multi-bit addition.
2. **Scalability:**
   - Multiple Full Adders can be cascaded to perform addition on larger binary numbers.

## 6.5. Circuits and Truth Table




| Inputs | | | Outputs | |
|---|---|---|---|---|
| $A$ | $B$ | $C_{in}$ | $S$ | $C_{out}$ |
| 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 |

## 6.6. RTL Viewer




## 6.7. Conclusion

The **Full Adder** is a fundamental digital circuit used to add three single-bit binary numbers. It produces a **sum** and a **carry-out** as outputs, which are computed using **XOR** and **AND-OR** logic, respectively. The Full Adder is a critical building block for multi-bit addition circuits and is widely used in digital systems for arithmetic operations.

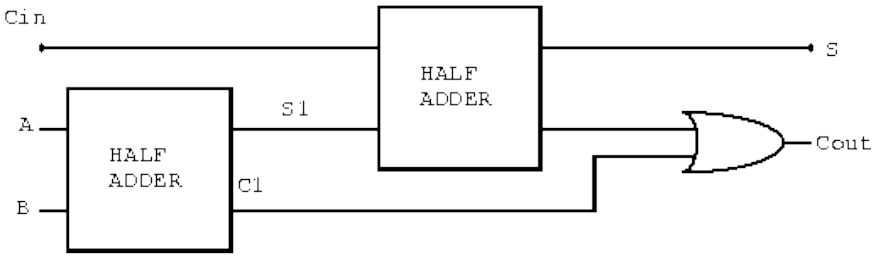
# 7. Full Adder From Half Adder

## 7.1. Description

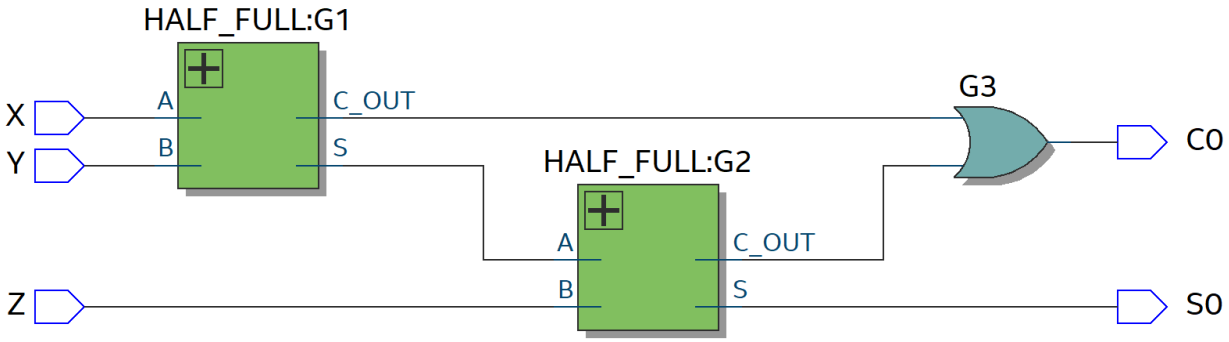
A **full adder** is a digital circuit that adds three binary inputs (A, B, and a carry-in, Cin) and produces two outputs: a sum (S) and a carry-out (Cout). It is a fundamental building block in arithmetic logic units (ALUs) and is used to perform binary addition in processors and other digital systems.
A full adder can be constructed using **two half adders** and an additional OR gate.

## 7.2. Circuits and Truth Table

| $C_1$ | $X_1$ | $Y_1$ | $Z_1$ | $C_2$ |
|---|---|---|---|---|
| 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 |




## 7.3. RTL Viewer




## 7.4. Key Points

1. **Half Adder vs. Full Adder**:
   - A half adder can only add two bits and does not account for a carry-in.
   - A full adder can add three bits (A, B, and Cin) and is used in multi-bit addition.
2. **Applications**:
   - Full adders are used in ripple-carry adders, carry-lookahead adders, and other arithmetic circuits.

3. **Scalability**:
    - Multiple full adders can be cascaded to create adders for larger bit-widths (e.g., 8-bit, 16-bit, etc.).

# 8. Referance


1. "Design and Analysis of High-Speed Adders Using Carry Lookahead Technique" https://www.ijert.org
2. "Design and Analysis of High-Speed Adders Using Carry Lookahead Technique" https://www.ijert.org
3. "Ripple Carry Adder" https://www.geeksforgeeks.org
4. "Performance Analysis of Ripple Carry Adder and Carry Lookahead Adder" https://www.ijarcs.info
5. "BCD to Excess-3 Converter" https://www.tutorialspoint.com
6. "Design and Implementation of BCD to Excess-3 Code Converter" https://www.ijert.org
7. "Half Adder and Full Adder" https://www.electronics-tutorials.ws
8. "Design and Analysis of Half Adder Using Quantum-Dot Cellular Automata" https://www.ijirset.com
9. "Full Adder" https://www.geeksforgeeks.org
10. "Design and Implementation of Full Adder Using CMOS Technology" https://www.ijert.org
11. "Full Adder Using Half Adders" https://www.allaboutcircuits.com
12. "Design of Full Adder Using Half Adder" https://www.ijcaonline.org