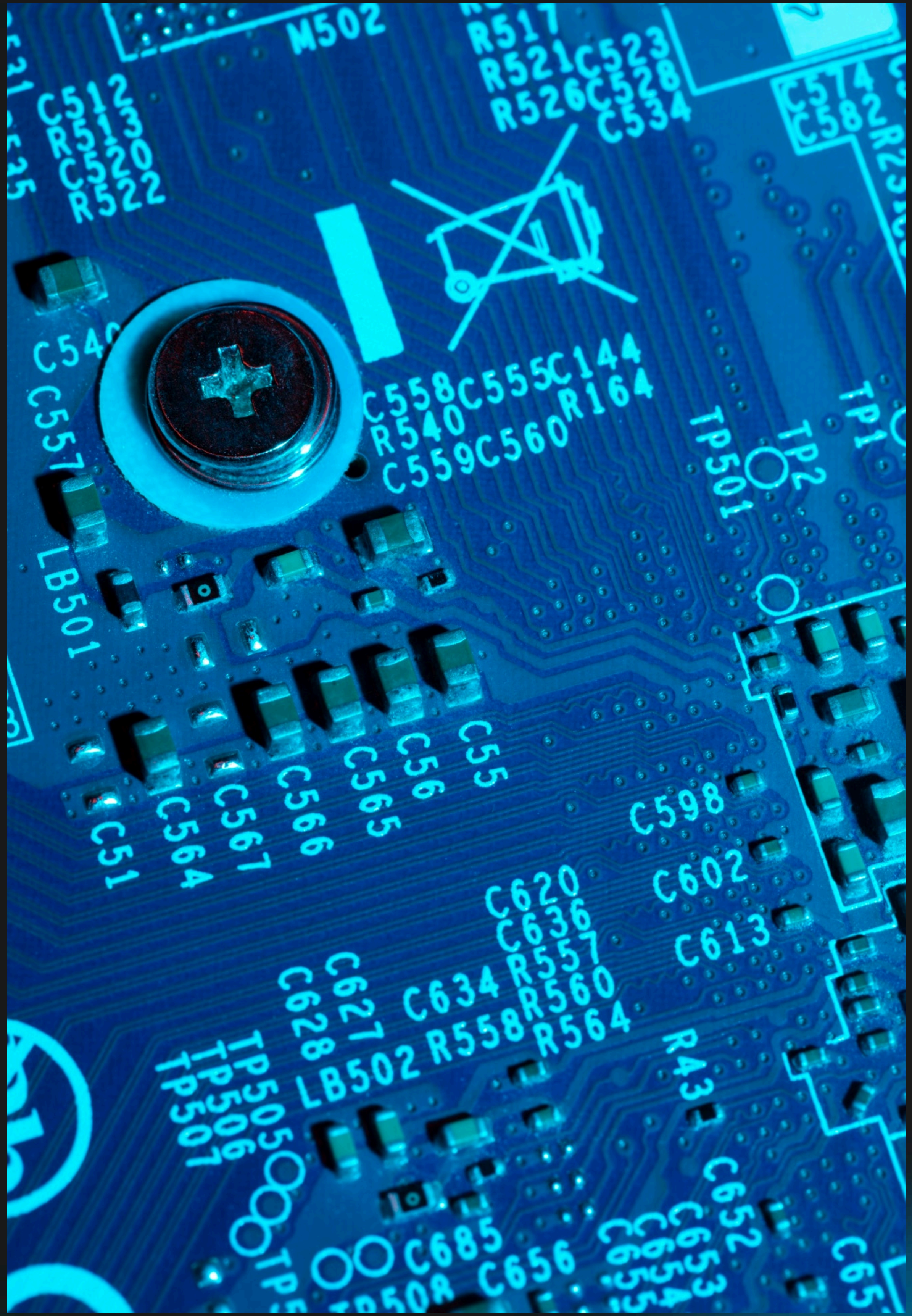
□ □ □

Binary Adder with Arduino

A Guide for Advanced Beginners

Introduction

Welcome to the *Binary Adder* with Arduino Project. In this presentation, we will explore the fundamental concepts of binary addition and demonstrate how it can be achieved using Arduino microcontrollers.



Understanding Binary Addition

Binary addition is a fundamental operation in digital computing. It involves adding together **binary numbers** to produce a sum. Each digit in a binary number represents a power of 2, and the addition follows similar rules to decimal addition.

How the Binary Base Works ❓

The binary base uses only the digits 0 and 1. Each position in a binary number represents a power of 2, similar to how each position in a decimal number represents a power of 10. For example, the binary number 1010 represents:

$$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 8 + 0 + 2 + 0 = 10$$

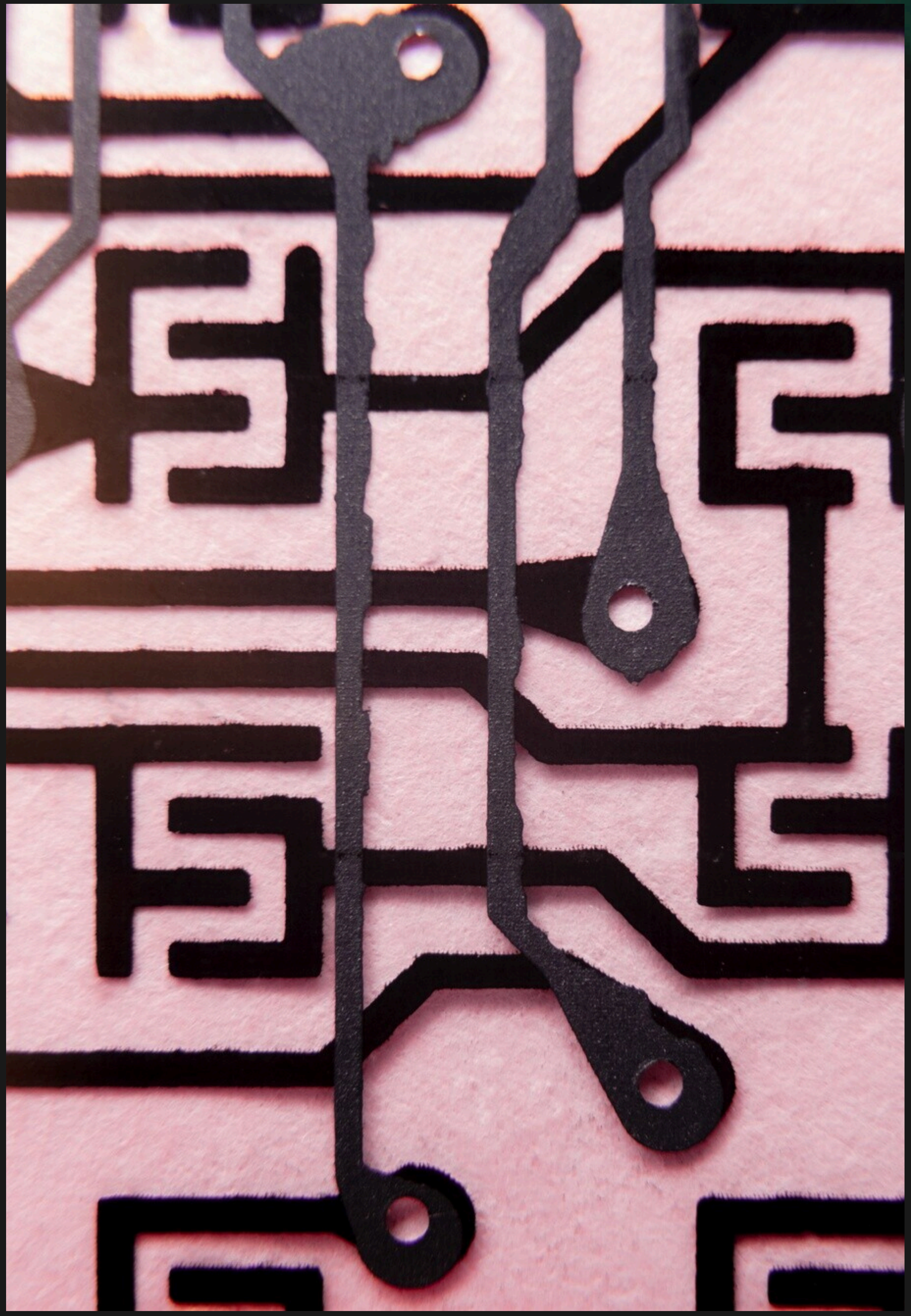
How to Add Binary Numbers ➕

To add binary numbers, you must add each column of bits, similar to decimal addition;
But remembering that (1 + 1 = 10) in binary. The result can have a carry that must be added to the next column.

```
  0101 (5 in decimal)
+ 0110 (6 in decimal)
 --------
  1011 (11 in decimal)
```

Binary Addition in Arduino

Arduino microcontrollers can perform binary addition using simple **logic operations**. By manipulating the individual bits of binary numbers, Arduino can add them together to produce accurate results. This process involves utilizing digital pins and bitwise operators.



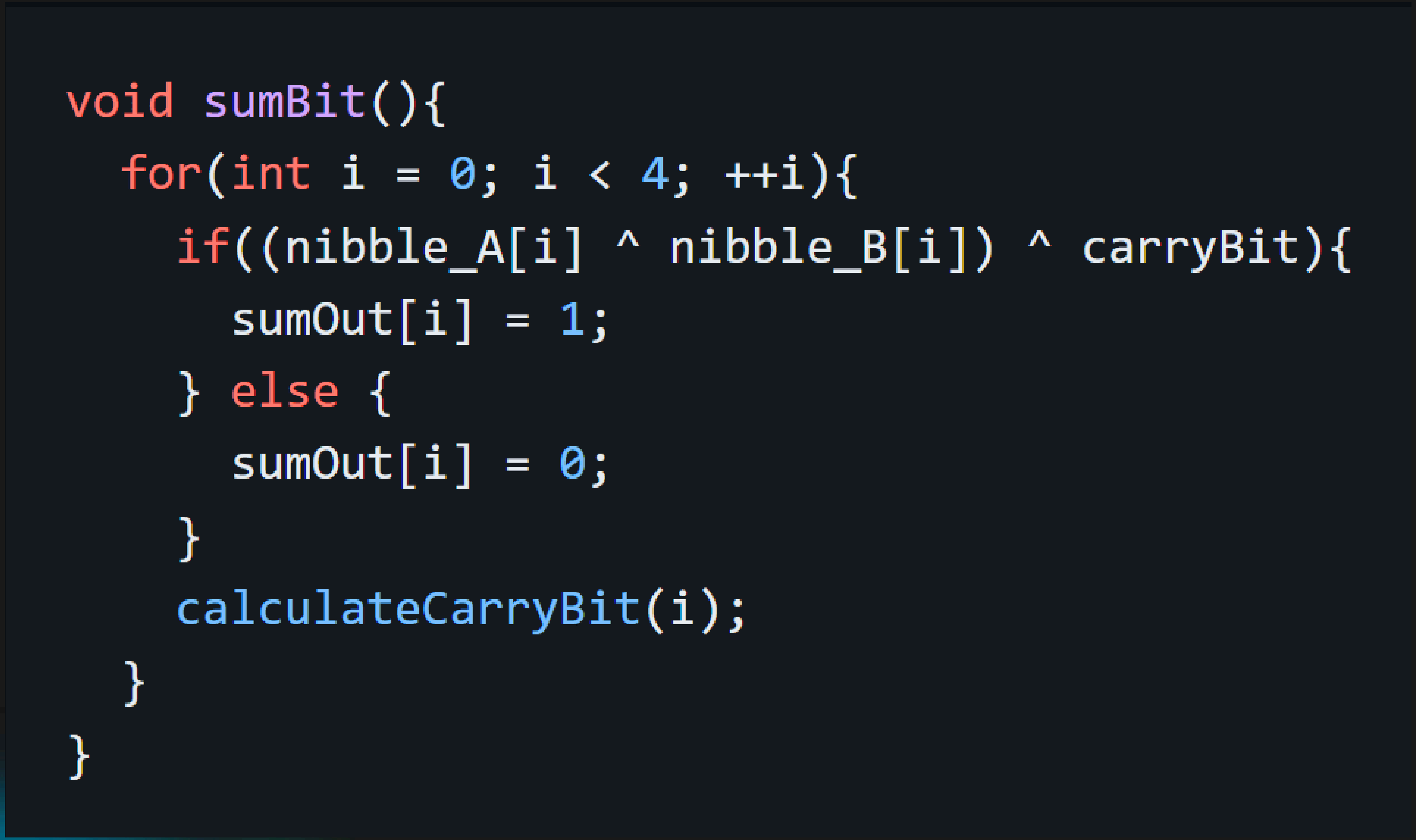
Bitwise Operators in Arduino

Arduino supports bitwise operators such as **AND**, **OR**, and **XOR** which are essential for manipulating individual bits in binary addition. These operators allow precise control over the binary addition process, making it efficient.

Implementing Binary Addition Algorithm

To implement binary addition with Arduino, we need to develop a **step-by-step algorithm** that performs the addition of two binary numbers. This algorithm will involve iterating through each bit and applying the appropriate logic operations.

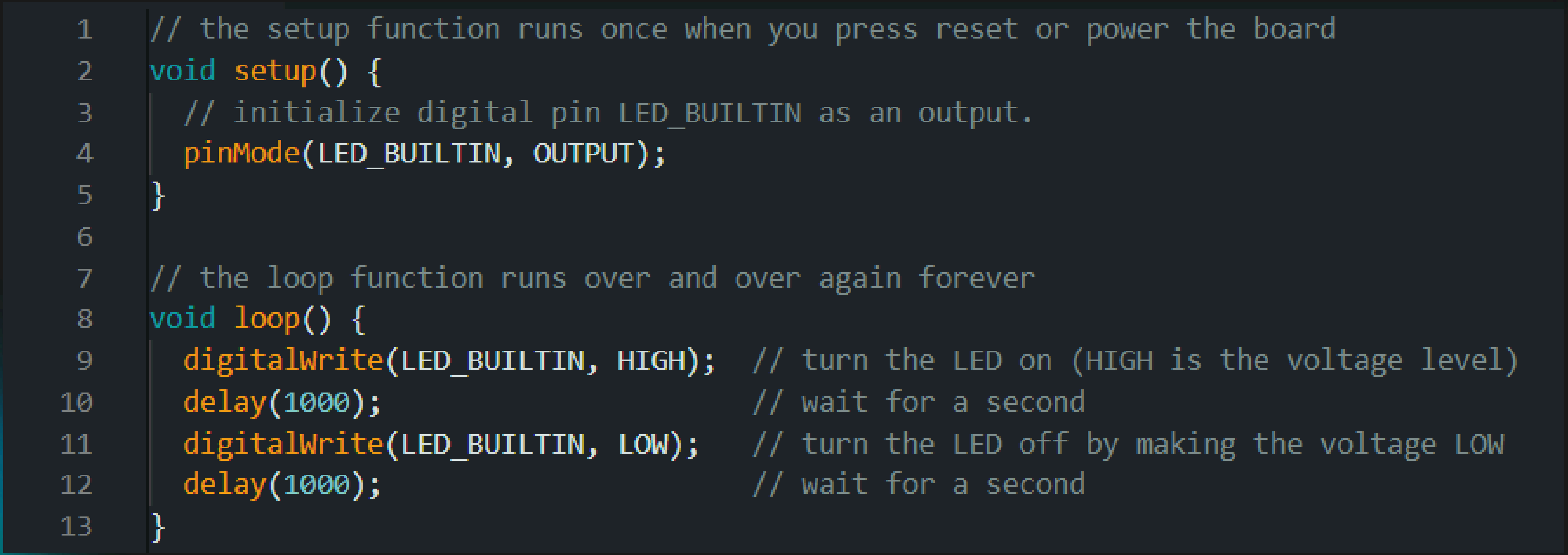
Addition Algorithm C++



```cpp
void sumBit(){
  for(int i = 0; i < 4; ++i){
    if((nibble_A[i] ^ nibble_B[i]) ^ carryBit){
      sumOut[i] = 1;
    } else {
      sumOut[i] = 0;
    }
    calculateCarryBit(i);
  }
}
```

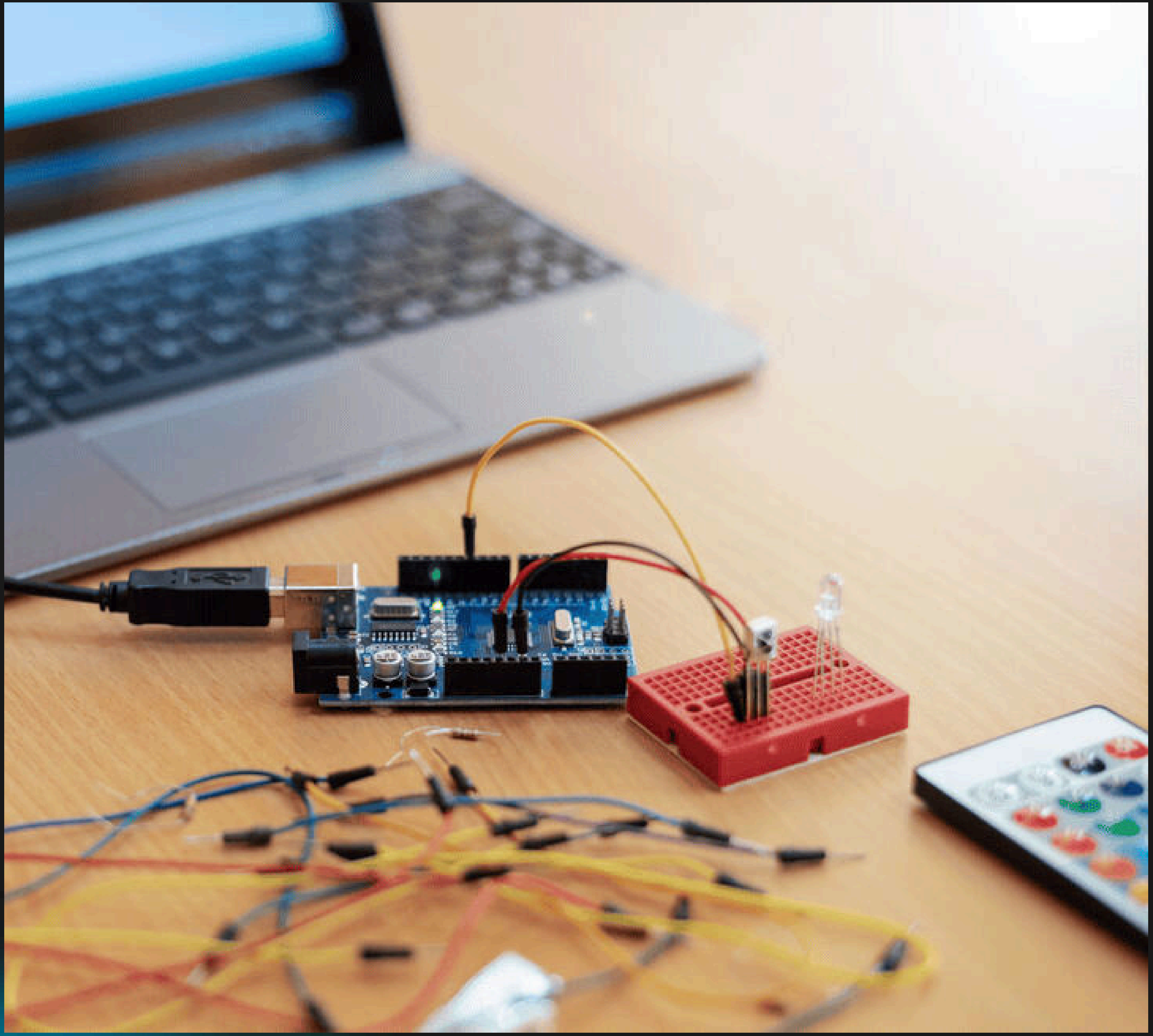
Arduino Code Example

Let's dive into a practical example by writing an **Arduino sketch** that implements the binary addition algorithm.



```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```



Testing and Debugging

Before implementing the binary addition code, it's crucial to thoroughly **test and debug** the functionality.

Choose a Simulator

Choosing a good simulator is crucial for efficient binary addition in Arduino. Optimization techniques play a key role in enhancing performance and minimizing resource usage.

TINKERCAD AUTODESK® TINKERCAD®

MultisimLive

Conclusion

In conclusion, this Begginer's Advanced guide has provided a detailed understanding of binary addition and its implementation with Arduino. We have explored the fundamental concepts, practical examples, and potential simulation, paving the way for further exploration and innovation in this field.

Thanks!

Do you have any questions?
luizgustavodiaspulzoficial@gmail.com
+55 51 99227-0465
github.com/LgZika