Programming Arduino microcontroller

ARDUINO®

Software Introduction :

- This program is installed on an Arduino board and can add two 4-bit numbers.

- It reads the number input bits and a sum signal from the board's digital pins. The program does the bit-by-bit addition of two numbers based on the sum signal. The sum bits are calculated using XOR operations, while the carry bits are calculated using AND and OR operations.

- The digital output pins then receive the results of the sum, including the final carry bit. Basic principles of digital binary arithmetic are demonstrated by this system, which is useful for applications requiring the addition of small binary numbers.

Development I :

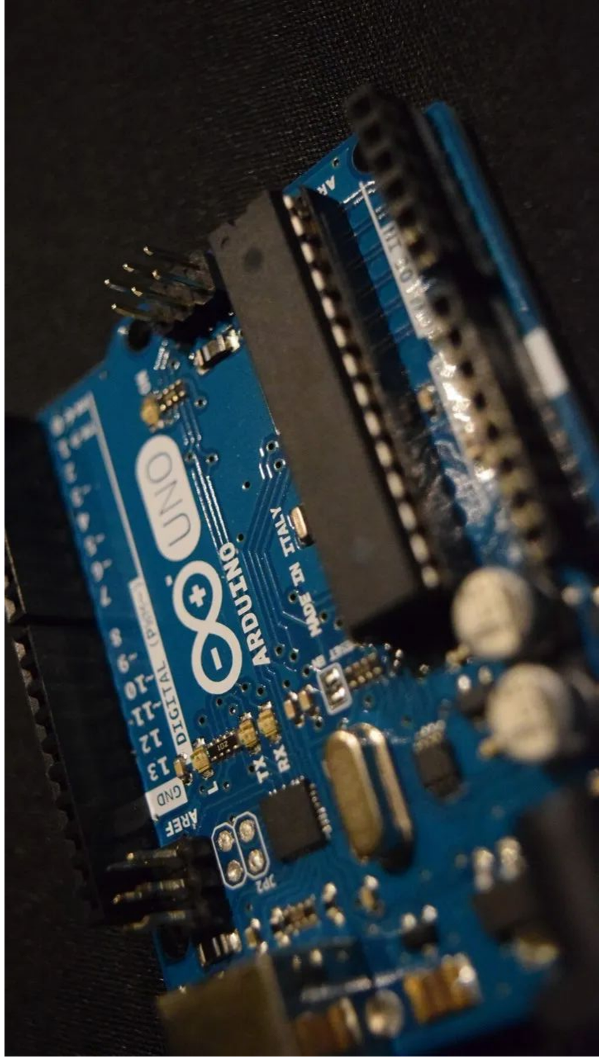
- **Program structure**

The code is structured in three main sections: the definition of global variables, the configuration of the input and output pins in the setup() function, and the main addition logic in the loop() function.

- **Definition of variables**

Global variables are defined at the start of the program. These include input pins for the number bits, an input pin for the sum signal, a variable to store the bits of the numbers to be summed, and variables to store the result of the sum and the carry bit.

The global variables are defined at the start of the program. These include input pins for the number bits, an input pin for the sum signal, a variable for storing the bits of the numbers to be summed, and variables for storing the sum result and the carry bit.



Development II :

- **Sum and Carry functions**

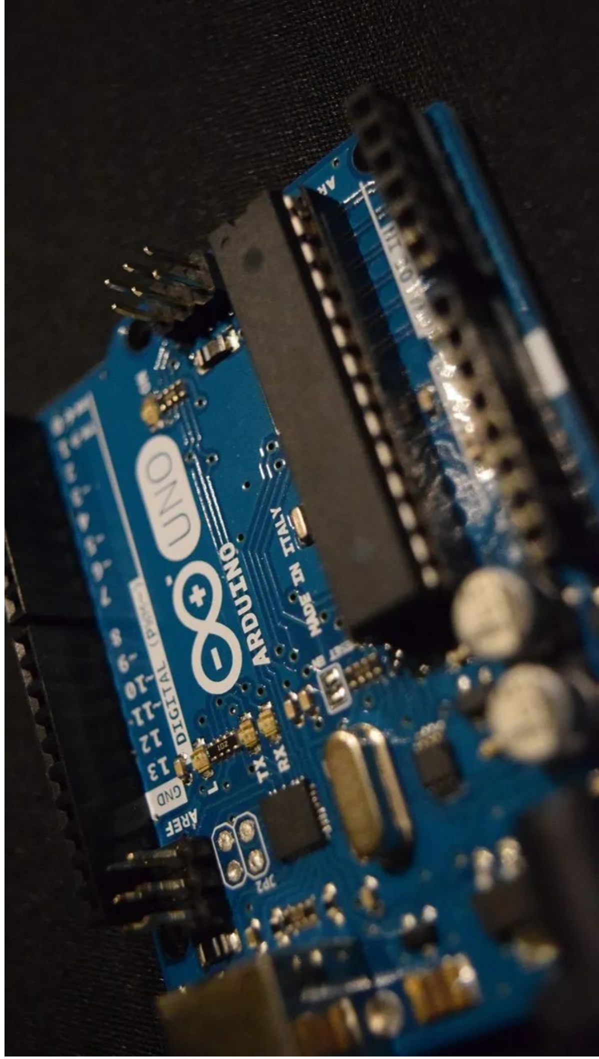
Two auxiliary functions are defined to perform the sum and carry calculation operations:

sumbit(int b1a, int b2a, int cBit): Calculates the sum bit of two input bits with a carry bit.

sumCarryBit(int b1a, int b2a, int cBit): Calculates the carry bit resulting from the sum of two input bits with a carry bit.

- **Conclusion**

This program is an example of how binary sum logic can be implemented in a microcontroller environment such as Arduino. It demonstrates the use of basic logical operations to perform the addition of binary numbers and the importance of the carry bit in arithmetic operations. This concept can be expanded to larger bit-width numbers and to other arithmetic operations such as subtraction, multiplication and division.



Survey of future activities

- **Expansion to Larger Bit Width Numbers**

To increase the capacity of the adder to handle numbers of 8, 16 or more bits.

Modify the code to read more input pins.

Implement the sum and carry logic for more bits.

Ensure that results and carry bits are properly managed and displayed.

- **Binary Subtraction implementation**

Develop auxiliary functions to perform subtraction with carry bits (borrow bits).

Add logic to select between addition and subtraction based on a control signal.

Bugs fixed :

- **Incorrect reading of input pins**

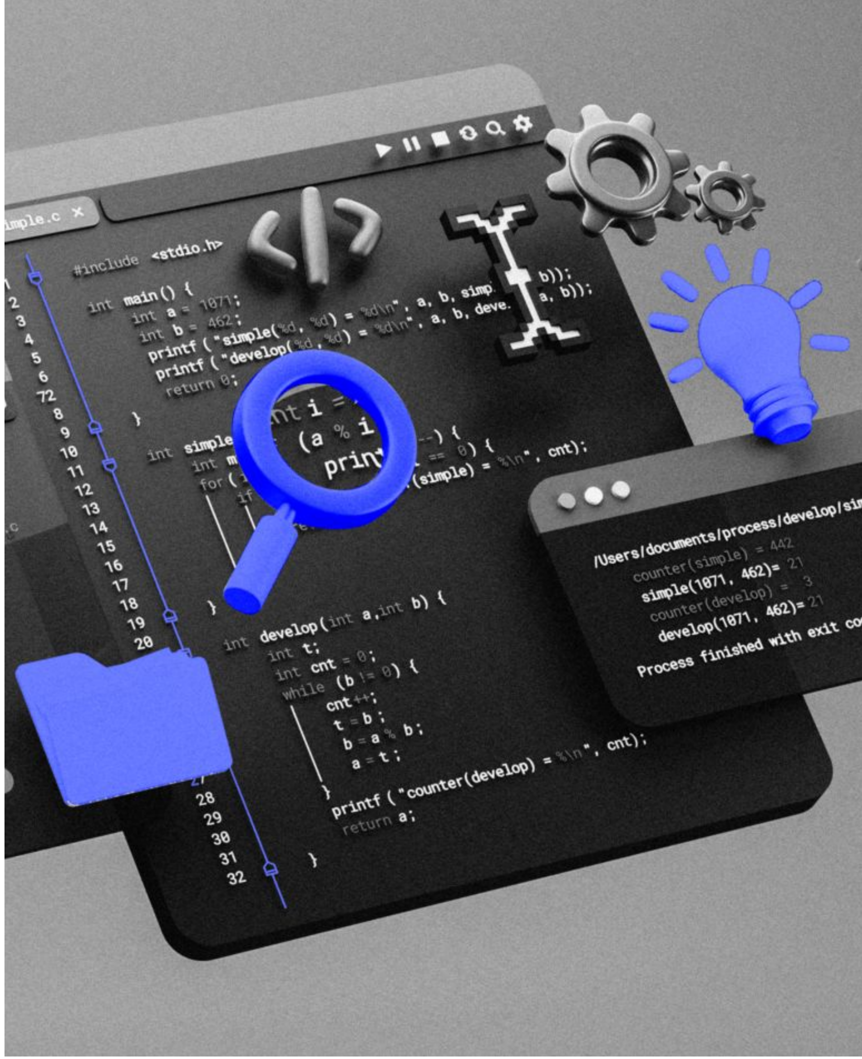
Initially, the binary number input pins were read incorrectly, causing incorrect sum results.

The correct configuration of the input pins was checked using pinMode(). In addition, it was ensured that the pins were correctly connected to the binary input values.

- **Output pin update failure**

The output pins for the result bits and the carry bit were not updating as expected.

The logic in the main loop was adjusted to ensure that the output pin values were updated after each sum operation.



Bugs to be resolved :

- **Synchronization conditions**

In cases where the input signals change rapidly, the adder may not synchronize the input bits correctly, resulting in incorrect sums.

Implement a synchronization or debounce mechanism to ensure that the input values are stable before performing the sum.

- **Performance with Higher Bit Width Numbers**

Summing numbers with more than 4 bits can cause slowdowns and incorrect results due to the increased complexity of the sum and carry operations.

Revise and optimize the code to efficiently manage the sum of larger bit-width numbers. Consider using more advanced programming techniques and appropriate data structures.