Design and Implementation of a Basic Arithmetic Logic Unit (ALU)

INTROCUCTION

This project aims to design and build a basic Arithmetic Logic Unit (ALU) that performs fundamental arithmetic and logic operations. The ALU can handle binary inputs and execute four arithmetic operations: Addition, Subtraction, Multiplication and Division, and three Logical Operations AND, OR, and NOT. The project is implemented using an ESP32 microcontroller, 0.96-inch OLED display, jumper wires, USB cable and programmed through the Arduino IDE.

COMPONENTS USED

ESP32 Microcontroller

The ESP32 is a powerful and versatile microcontroller developed by Espressif Systems, widely used in IoT, home automation, and embedded applications. It features a dual-core processor (Tensilica Xtensa LX6) running up to 240 MHz, with integrated Wi-Fi and Bluetooth capabilities. The ESP32 supports a variety of peripherals, including multiple GPIO pins, I2C, SPI, UART, PWM, and ADC/DAC for analog input/output. These GPIO pins make it suitable for various IoT applications, including this ALU project.

0.96-inch OLED Display

The OLED display is used to show the results of the arithmetic and logic operations. Its small size and low power consumption make it ideal for microcontroller-based projects. The display communicates with the ESP32 over I2C.

Jumper Wires and USB Cable

Jumper wires are used to make physical connections between the ESP32, the OLED display, and other components. A USB cable is used to upload code from the Arduino IDE to the ESP32.

Arduino IDE

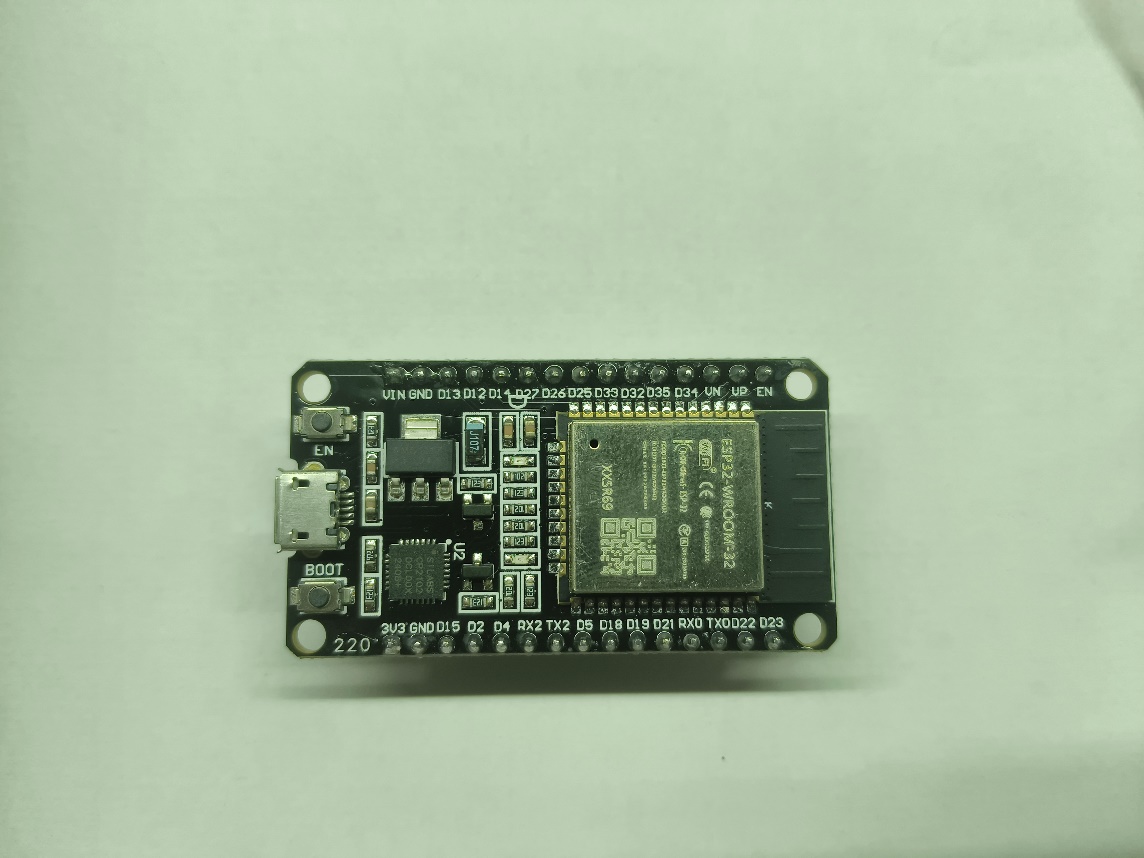
The Arduino IDE (Integrated Development Environment) is a versatile, open-source software platform used primarily for programming and uploading code to Arduino-compatible microcontroller boards, such as the Arduino Uno, ESP32, and many others. Known for its user-friendly interface, the IDE supports languages based on C/C++ and includes built-in libraries to streamline hardware interaction, allowing users to control components like LEDs, sensors, motors, and displays. In this project use used Arduino IDE to program the ESP32 to function as an Arithmetic and Logic Unit (ALU)

PHYSICAL CONNECTIONS

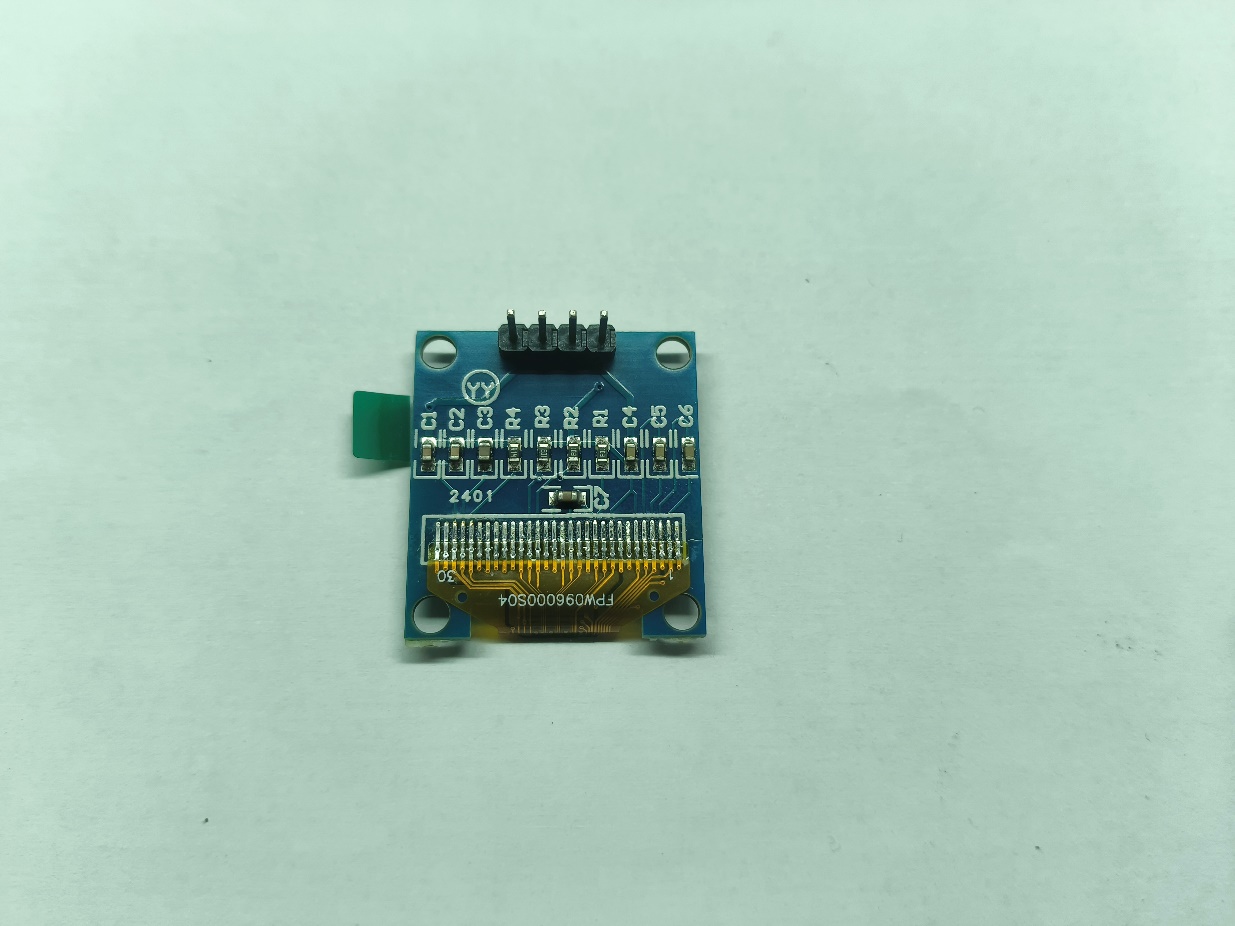
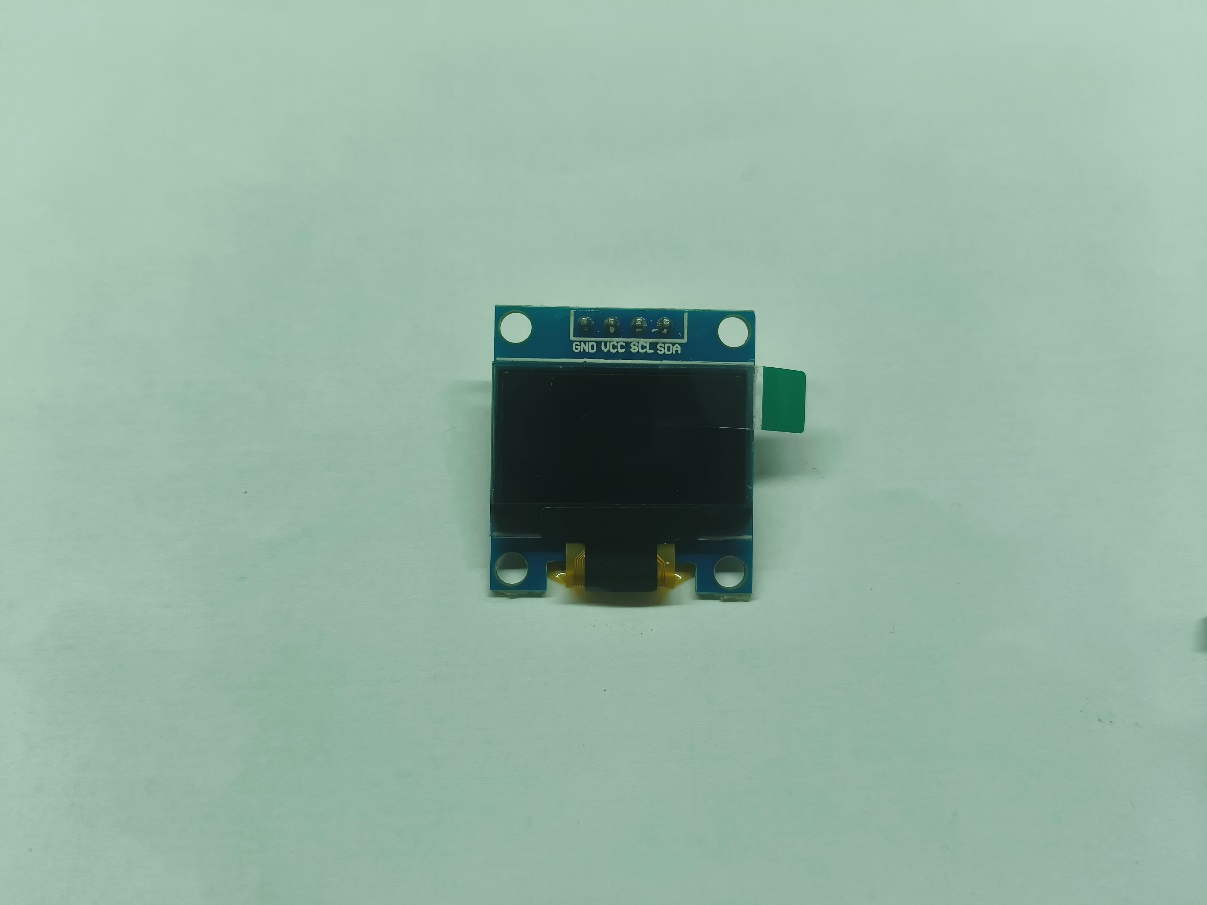
The following connections are made between the ESP32 microcontroller and the 0.96-inch OLED display as seen in the uploaded image.

- VCC (OLED) to 3V3 (ESP32)   
- GND (OLED) to GND (ESP32)  
- SCL (OLED) to GPIO D22 (ESP32)  
- SDA (OLED) to GPIO D21 (ESP32)

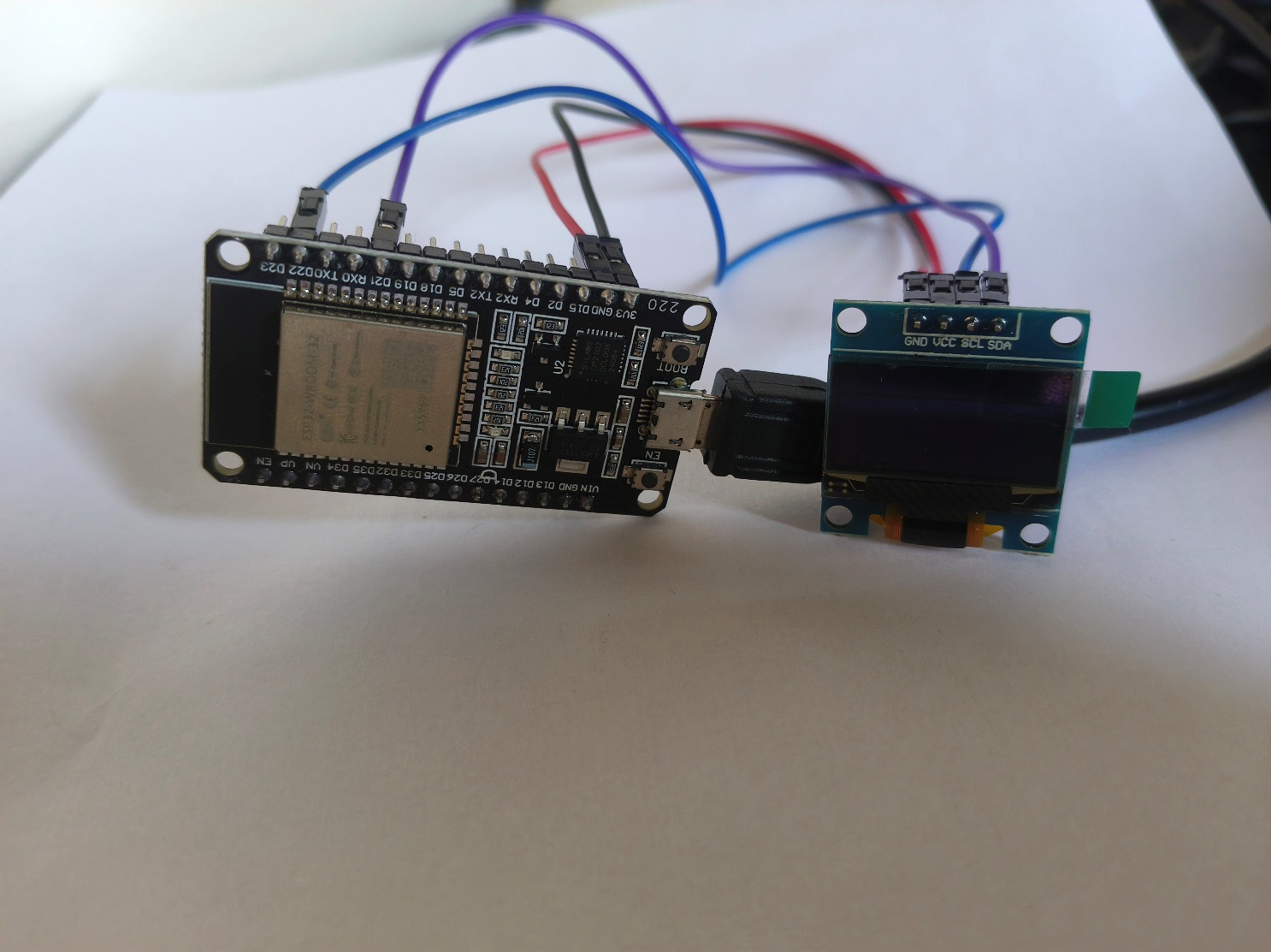
Jumper wires are used to establish these connections. The USB cable connects the ESP32 to a computer, allowing code to be uploaded via the Arduino IDE.



***ESP32 MICROCONTROLLER***



***OLED DISPLAY***

******

***ALU SETUP***

CODE

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, -1);

void setup() {

  // Initialize Serial Communication for receiving commands

  Serial.begin(115200);

  // Initialize OLED Display

  if (!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

    Serial.println(F("SSD1306 allocation failed"));

    for (;;);

  }

  display.clearDisplay();

  display.display();

  // Display opening message

  displayWelcomeMessage();

}

void loop() {

  // Check if data is available in Serial

  if (Serial.available() > 0) {

    String command = Serial.readStringUntil('\n');

    command.trim();  // Remove any extra whitespace

    String inputDisplay, outputDisplay;

    processCommand(command, inputDisplay, outputDisplay);  // Get input and output displays

    displayText(inputDisplay, outputDisplay);  // Display both input and output

  }

}

void processCommand(String command, String &inputDisplay, String &outputDisplay) {

  int num1, num2;

  char operation;

  // Parse input in the format "4 + 5"

  if (sscanf(command.c\_str(), "%d %c %d", &num1, &operation, &num2) == 3) {

    switch (operation) {

      case '+':

        inputDisplay = String(num1) + " + " + String(num2);

        outputDisplay = String(num1 + num2);

        break;

      case '-':

        inputDisplay = String(num1) + " - " + String(num2);

        outputDisplay = String(num1 - num2);

        break;

      case '\*':

        inputDisplay = String(num1) + " \* " + String(num2);

        outputDisplay = String(num1 \* num2);

        break;

      case '/':

        inputDisplay = String(num1) + " / " + String(num2);

        if (num2 != 0) {

          float divResult = (float)num1 / num2;

          outputDisplay = String(divResult, 2);

        } else {

          outputDisplay = "Error";

        }

        break;

      case '&':  // AND operation

        inputDisplay = String(num1) + " AND " + String(num2);

        outputDisplay = toBinaryString(num1 & num2);

        break;

      case '|':  // OR operation

        inputDisplay = String(num1) + " OR " + String(num2);

        outputDisplay = toBinaryString(num1 | num2);

        break;

      default:

        inputDisplay = "Invalid Op";

        outputDisplay = "";

    }

  } else if (sscanf(command.c\_str(), "NOT %d", &num1) == 1) {

    inputDisplay = "NOT " + String(num1);

    outputDisplay = toBinaryString(~num1);

  } else {

    inputDisplay = "Invalid Cmd";

    outputDisplay = "";

  }

}

void displayText(String inputText, String outputText) {

  display.clearDisplay();

  display.setTextSize(1.3);  // Set a smaller text size for better layout

  display.setTextColor(SSD1306\_WHITE);

  // Display the input on the first line

  display.setCursor(0, 5);

  display.print("Input: ");

  display.print(inputText);

  // Display the output on the second line

  display.setCursor(0, 20);  // Move cursor to the next line

  display.print("Output: ");

  display.print(outputText);

  display.display();

}

void displayWelcomeMessage() {

  display.clearDisplay();

  display.setTextSize(2);  // Larger text for welcome message

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(45, 17);  // Center the text

  display.print("ALU");

  display.setCursor(30, 40);

  display.print("Ready!");

  display.display();

}

// Helper function to convert an integer to a binary string

String toBinaryString(int num) {

  String binaryString = "";

  for (int i = 15; i >= 0; i--) {

    binaryString += ((num >> i) & 1) ? '1' : '0';

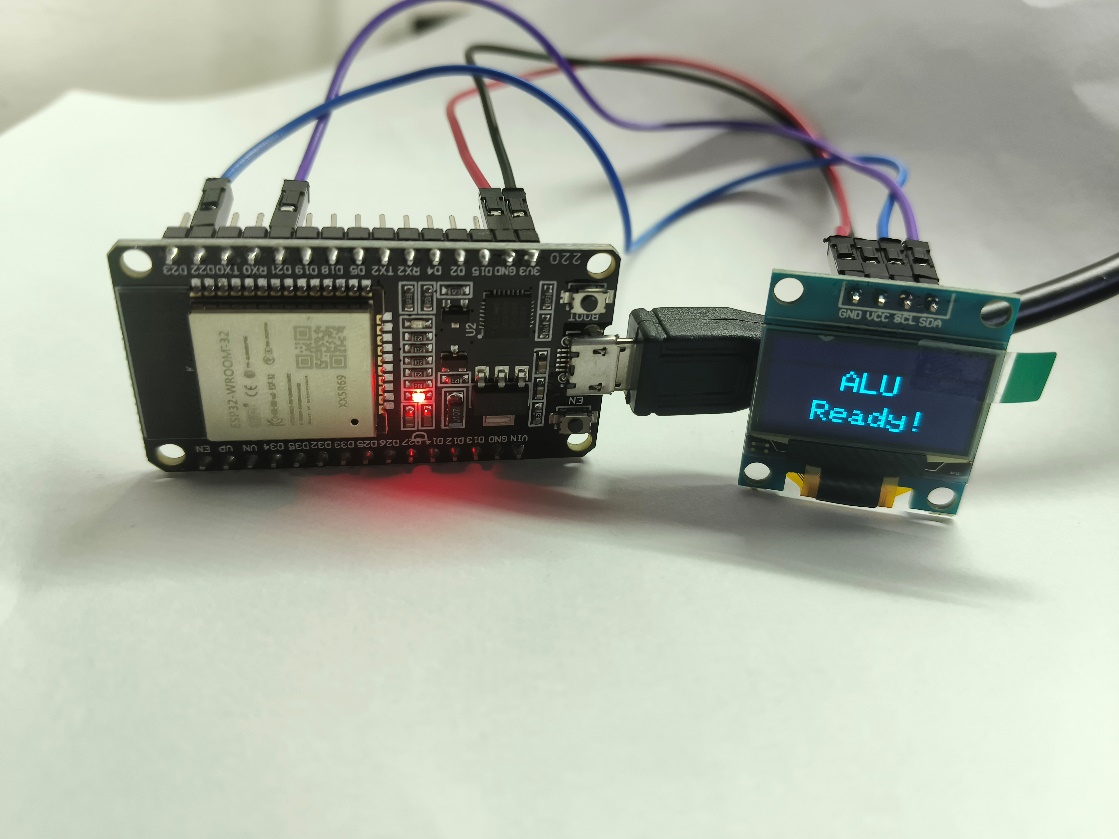
  }

  return binaryString;

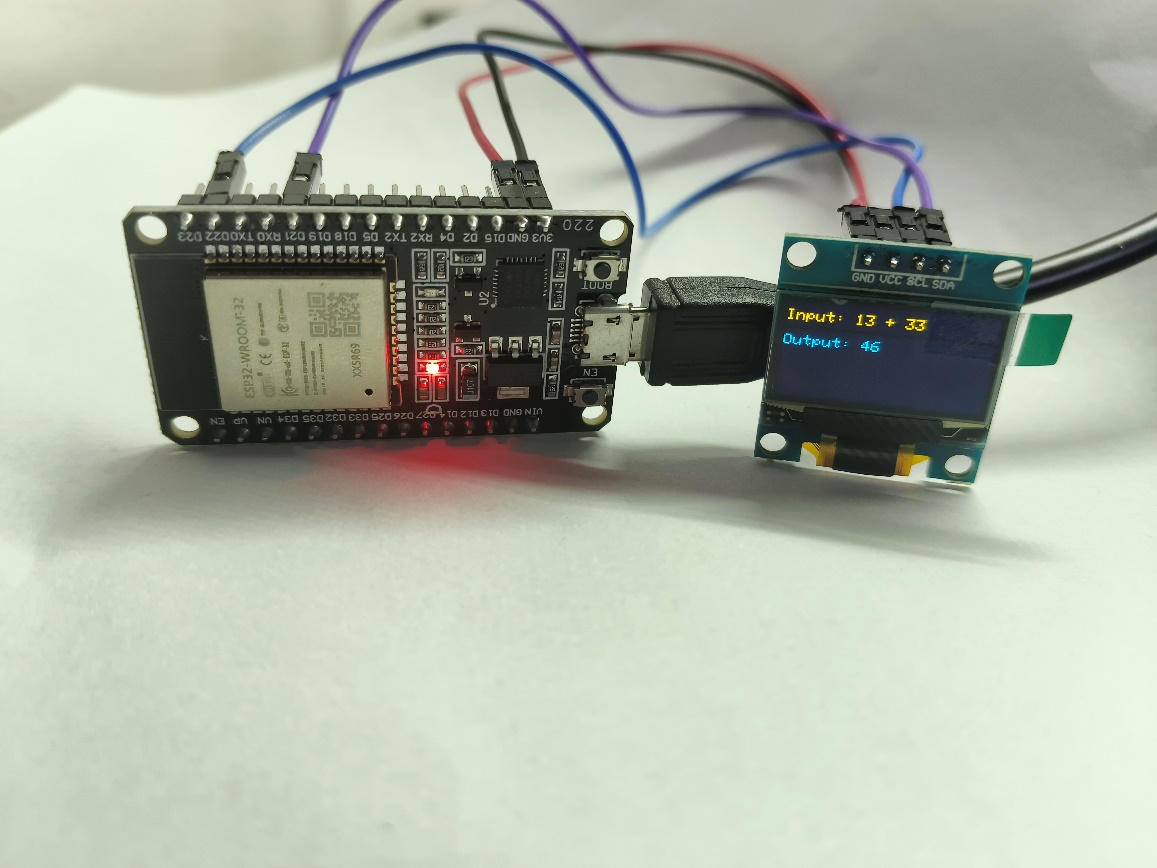
}

RESULTS

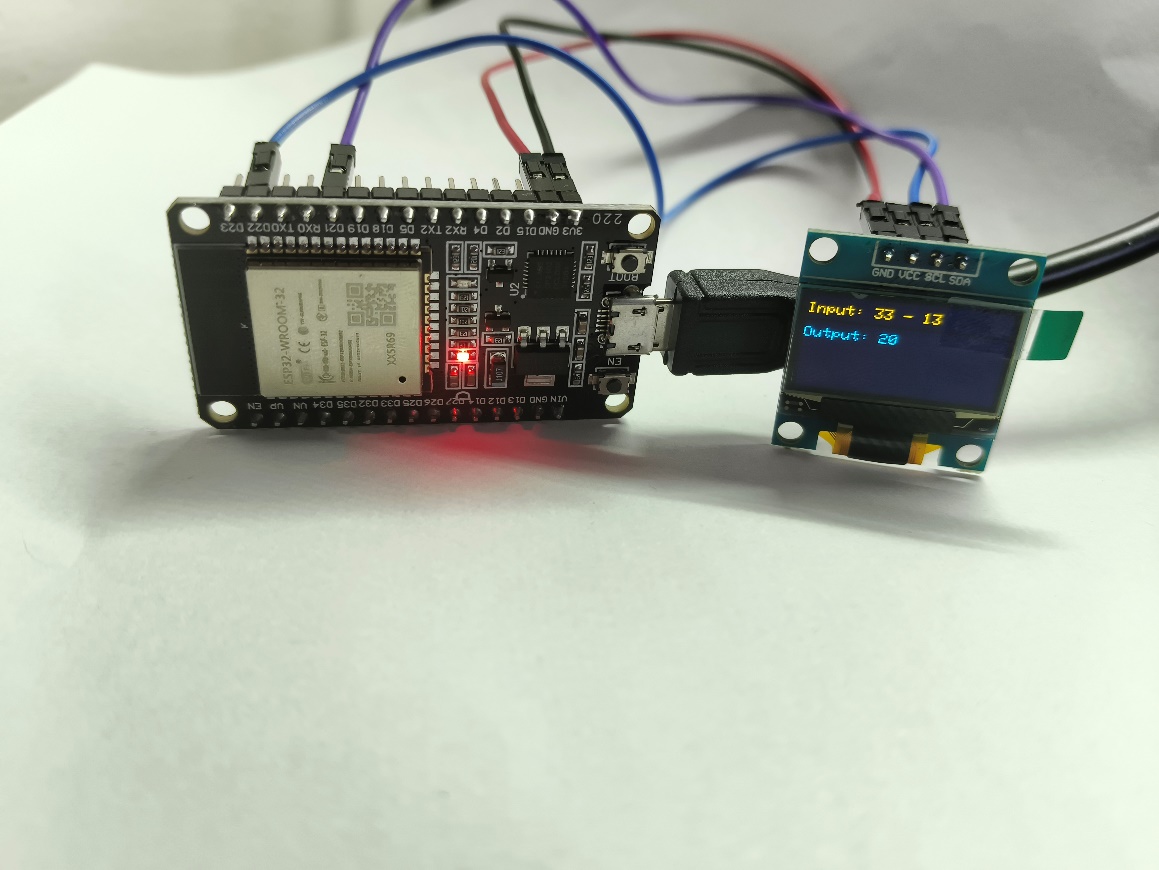
Command input for the arithmetic and logic function are given from the serial monitor using laptop keyboard as the input device.



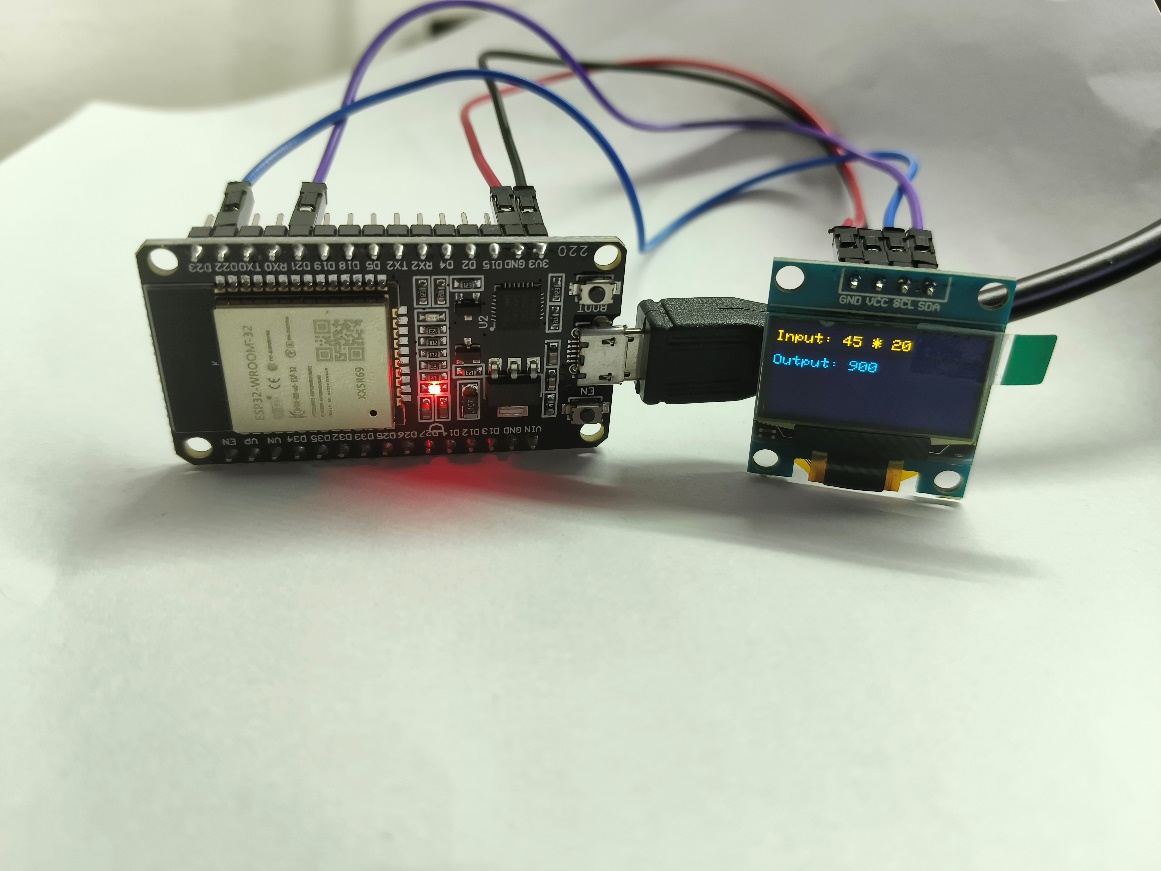
***STARTING UP***



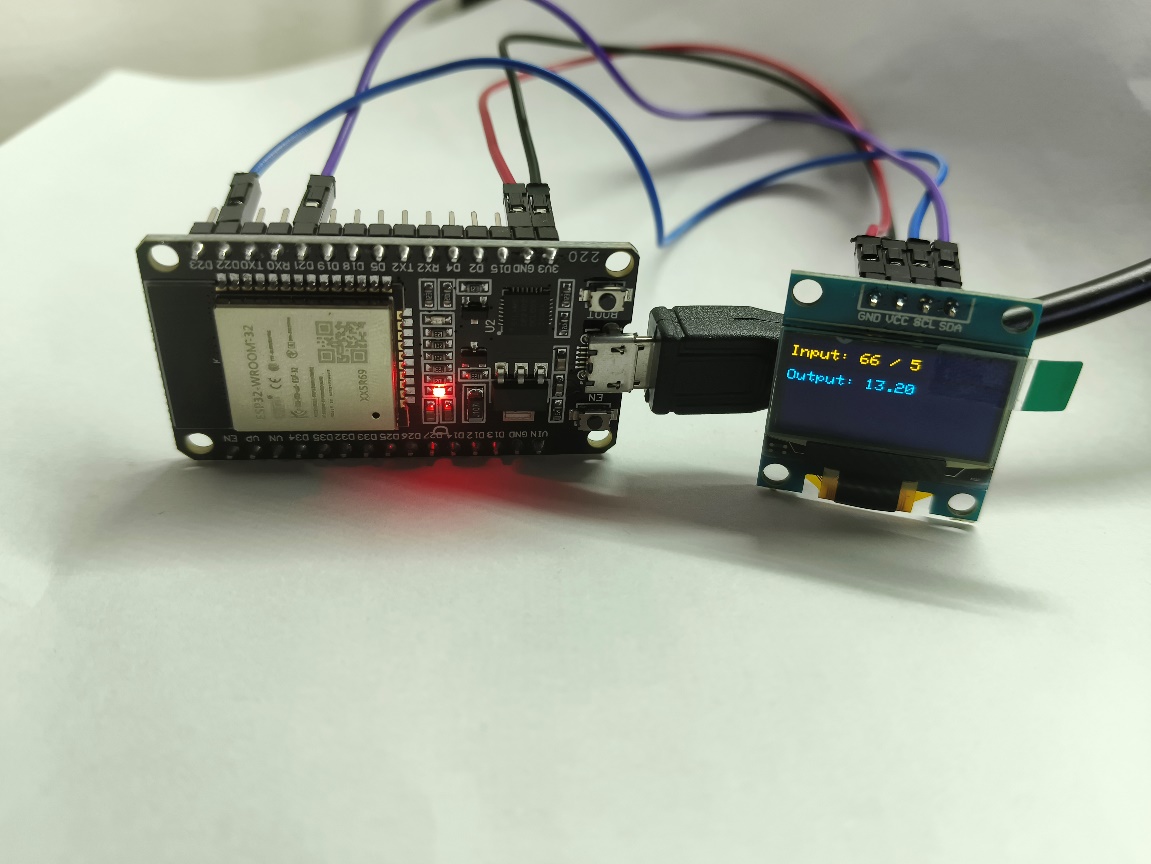
***ARITHMETIC FUNCTION - ADDITION***



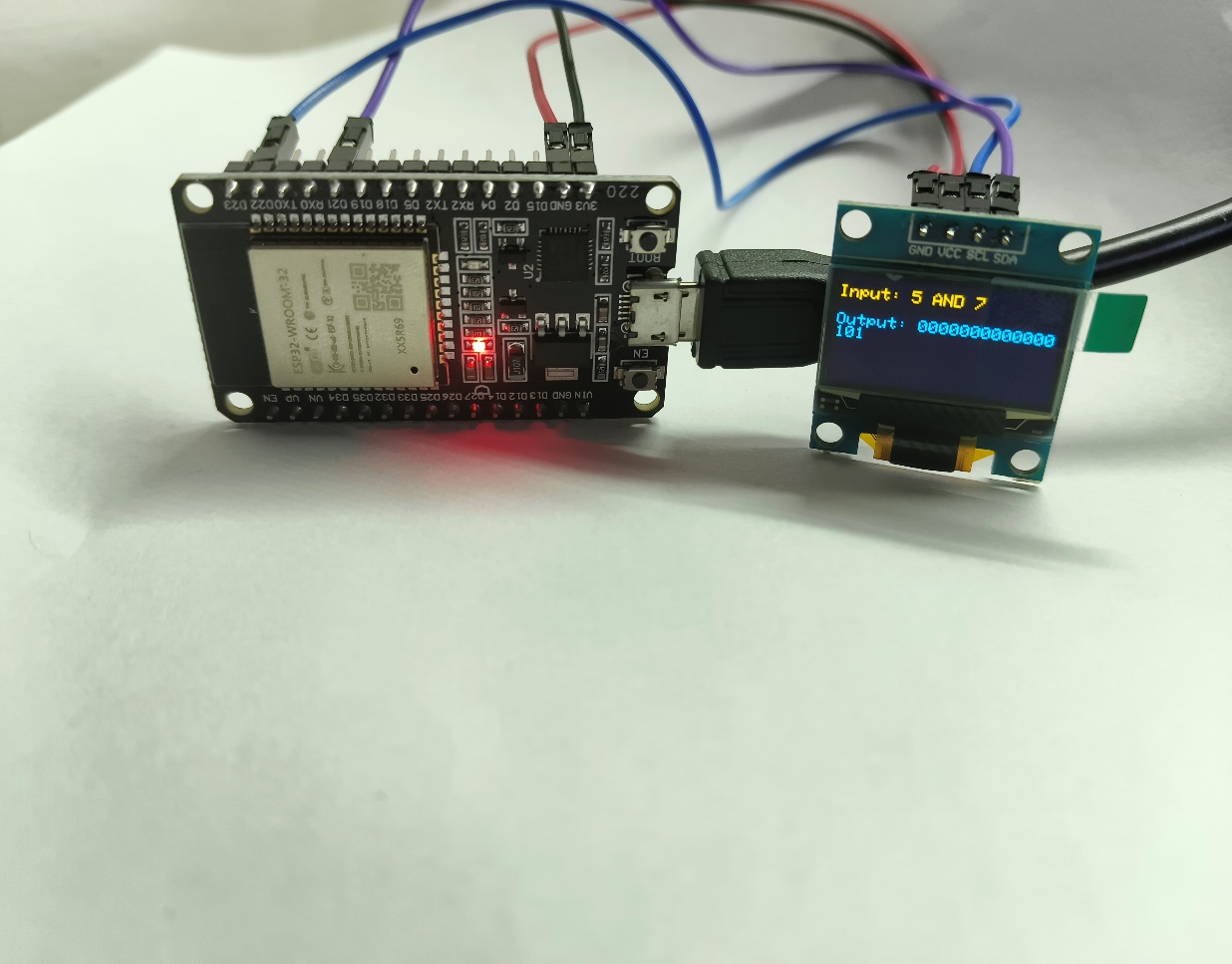
***ARITHMETIC FUNCTION - SUBSTRACTION***



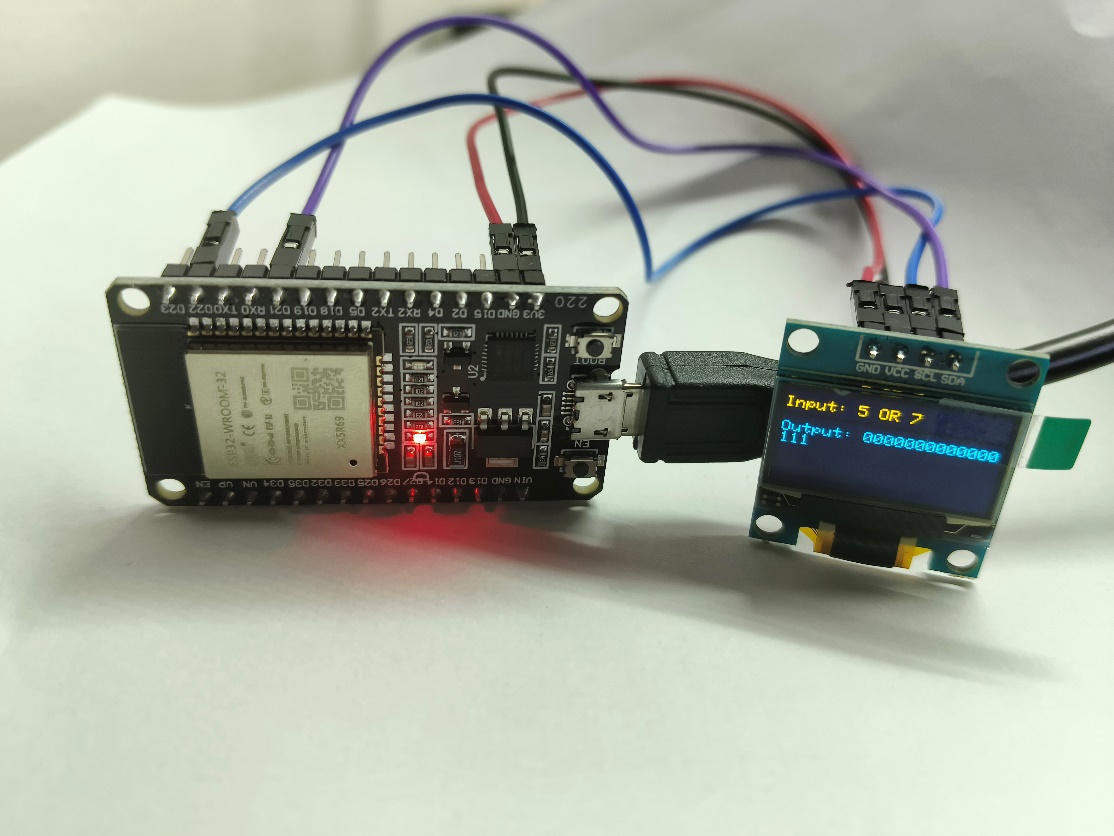
***ARITHMETIC FUNCTION - MULTIPLICATION***



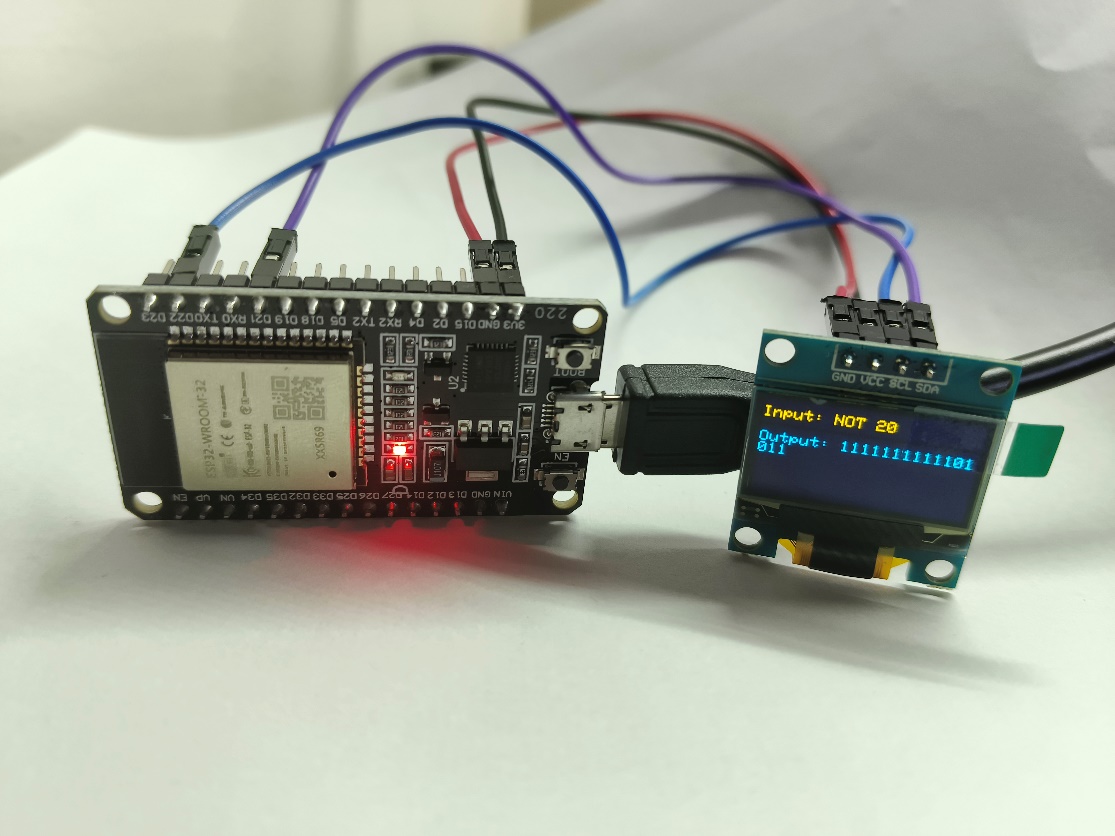
***ARITHMETIC FUNCTION – DIVISION***



***LOGICAL FUNCTION – AND***



***LOGICAL FUNCTION – OR***



***LOGICAL OPERATION – NOT***