TINKERING LAB

PROJECT-4

Wireless Communication over WIFI / Bluetooth between ESP32 Microcontrollers

IIT Ropar

Indian Institute of Technology, Ropar

Group Members:

2021MEB1258 – Aabhas Bhadauria

2021MEB1262 – Abhishek Meena

2021MEB1265 – Akshat Chouhan

2021MEB1277 – Bommiditha Jyothsnavi

2021MEB1303 – Neetesh Kumar Meena

2021MEB1333 – Umang Raj

Introduction:

The ESP32 is an Internet of Things platform that utilizes a microcontroller and includes built-in Wi-Fi and Bluetooth modules. Just like any other microcontroller board, it has communication protocols that facilitate the transmission and reception of data.

With Wi-Fi, the ESP32 boards can connect to the same wireless network, allowing them to exchange data with each other over longer distances. On the other hand, Bluetooth enables the ESP32 boards to communicate with each other over shorter distances, typically within a range of 10 meters or less.

The environmental conditions output will be displayed on I2C SSD1306 OLED display.

Objectives:

Our Project objective is to create a system of wireless communication with WIFI/Bluetooth with ESP32 controllers.

In our project we will be trying to build a communication between two ESP32 microcontrollers. The devices will use Bluetooth for communication over the working area.

Materials Required:

A) **Breadboard**
B) **Jumper Wires**
C) **Sensor for Soil Moisture**

How Does a Soil Moisture Sensor Work?

The soil moisture sensor operates in a straightforward manner.

The fork-shaped probe with two exposed conductors acts as a variable resistor (similar to a potentiometer) whose resistance varies with the soil's moisture content.

- The more water in the soil, the better the conductivity and the lower the resistance.
- The less water in the soil, the lower the conductivity and thus the higher the resistance.

The sensor produces an output voltage according to the resistance, which by measuring we can determine the soil moisture level.

D) **ESP32 Development boards:**

The ESP32 development board is a relatively small, low-power, easily obtainable, complete and breadboard-friendly microcontroller board, with built-in Wi-Fi and Bluetooth.

It comes integrated with an antenna and RF balun, power amplifier, low-noise amplifiers, filters and a power management module. The board is robust and is capable of functioning reliably in temperatures ranging from far below freezing point up to a little above boiling point.

E) **12C SSD 1306 OLED display:**

The SSD1306 OLED display can be used with ESP32 to display sensor data, show network parameters, or display insights that are received from a server. The environmental conditions output will be displayed on 12C SS1306 OLED display.

Code for Sender:

```
#include <WiFi.h>

// Replace with your network credentials
const char* ssid = "POCO F4";
const char* password = "87654321";

// IP address of receiver ESP32
IPAddress receiverIP(192, 168, 141, 202);
// Port number to send data to receiver
```

```
uint16_t receiverPort = 8888;

// Pin connected to moisture sensor
int moisturePin = 34;

WiFiClient client;

void setup() {
_Serial.begin(115200);

_// Connect to Wi-Fi network
_Serial.println();
_Serial.println();
_Serial.print("Connecting to ");
_Serial.println(ssid);

_WiFi.begin(ssid, password);

_while (WiFi.status() != WL_CONNECTED) {
__delay(1000);
__Serial.print(".");
_}

_Serial.println("");
_Serial.println("WiFi connected");
_Serial.println("IP address: ");
_Serial.println(WiFi.localIP());
}

void loop() {
_// Read moisture sensor data
_int moistureValue = analogRead(moisturePin);

_// Send moisture data to receiver
_if (client.connect(receiverIP, receiverPort)) {
__Serial.print("Sending moisture data: ");
__Serial.println(moistureValue);

__client.print(moistureValue);

__client.stop();
_} else {
__Serial.println("Connection failed");
_}

_delay(5000); // Wait for 5 seconds
}
```
Code for receiver:

```
#include <WiFi.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Replace with your network credentials
const char* ssid = "POCO F4";
const char* password = "87654321";

// Port number to listen for incoming data
uint16_t receiverPort = 8888;

// Initialize LCD
LiquidCrystal_I2C lcd(0x27, 16, 2);

WiFiServer server(receiverPort);

void setup() {
_Serial.begin(115200);
_lcd.init();
_lcd.backlight();

_// Connect to Wi-Fi network
_Serial.println();
_Serial.println();
_Serial.print("Connecting to ");
_Serial.println(ssid);

_WiFi.begin(ssid, password);

_while (WiFi.status() != WL_CONNECTED) {
__delay(1000);
__Serial.print(".");
_}

_Serial.println("");
_Serial.println("WiFi connected");
_Serial.println("IP address: ");
_Serial.println(WiFi.localIP());

_// Start server
_server.begin();
}

void loop() {
_// Check if client has connected
_WiFiClient client = server.available();
_if (client) {
```

```
__Serial.println("Client connected");

__// Read moisture data from client
__String data = client.readStringUntil('\n');
__int moistureValue = data.toInt();

__Serial.print("Received moisture data: ");
__Serial.println(moistureValue);

__// Print moisture data to LCD
__lcd.setCursor(0,0);
__lcd.print("Moisture: ");
__lcd.print(moistureValue);

__client.stop();
_}
}
```

Applications:

The ESP32 supports several different wireless communication protocols. Each protocol has its advantages and disadvantages and one can be more suitable than the other depending on the application.

Bluetooth Low Energy (BLE):

Bluetooth Low Energy, BLE for short, is a power-conserving variant of Bluetooth. BLE's primary application is short-distance transmission of small amounts of data (low bandwidth).

Unlike Bluetooth, which is always on, BLE remains in sleep mode constantly except for when a connection is initiated. This makes it consume very low power. BLE consumes approximately 100x less power than Bluetooth (depending on the use case).

- Low power consumption
- Short distance transmission
- Low bandwidth (small amounts of data)
- Ideal for exchanging small amounts of data periodically
- Supports point-to-point, broadcast, and mesh network

Bluetooth Classic:

Bluetooth is a wireless technology standard used for exchanging data between fixed and mobile devices over short distances. It is optimized for continuous data streaming, while BLE is optimized for short burst data transmission. It consumes approximately x100 more power than BLE.

- Short distance transmission

- Optimized for continuous data streaming

<u>ESP:</u>

ESP is a connectionless communication protocol developed by Espressif that features short packet transmission. This protocol enables multiple devices to talk to each other in an easy way.

- Fast communication protocol
- Up to 250-byte payload can be carried
- Encrypted and unencrypted communication
- Range communication (220 meters in open field accordingly to our experiments)