Bluetooth Device Tracker using ESP32/Arduino UNO

Table of Contents

1. Introduction

This project involves creating a Bluetooth Device Tracker using an ESP32 microcontroller. The tracker detects nearby Bluetooth devices and logs the information, which is displayed on the Serial Monitor. This tool can be used for identifying unauthorized or suspicious devices in the vicinity.

2. Project Objective

- To create a tool that can scan and log nearby Bluetooth devices.
- To provide a user-friendly interface via the Serial Monitor for displaying detected devices.
- To demonstrate the use of the ESP32's built-in Bluetooth capabilities.

3. Components and Requirements

Hardware

- ESP32 Development Board
- USB Cable for programming and power
- Breadboard and jumper wires (optional for better organization)

Software

- Arduino IDE
- Bluetooth Terminal App (for testing)

Libraries

- BluetoothSerial.h (included in the ESP32 board package for Arduino)

4. ESP32 Setup and Configuration

Steps to Setup

1. **Install ESP32 Board Package**:
    - Open the Arduino IDE.
    - Go to **File > Preferences** and add the following URL in the *Additional Boards Manager URLs* field:

arduino

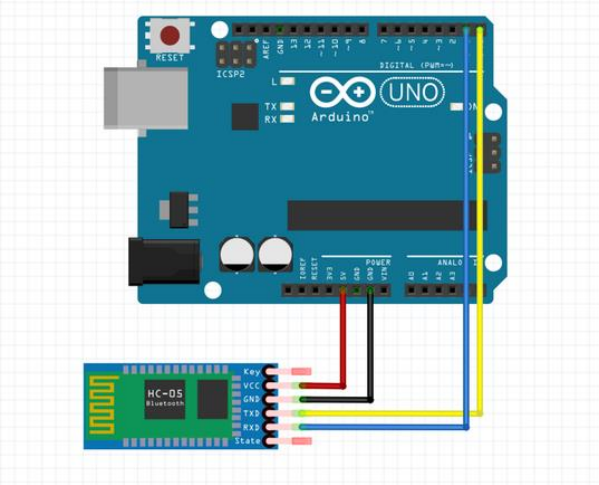
Copy code

https://dl.espressif.com/dl/package_esp32_index.json

    - Go to **Tools > Board > Boards Manager**, search for esp32, and install the latest version.
2. **Select the Correct Board and Port**:
    - Go to **Tools > Board**, and select your ESP32 board.
    - Go to **Tools > Port**, and select the appropriate COM port for your ESP32.
3. **Upload the Code**:
    - Copy and paste the provided code into the Arduino IDE.
    - Click on the **Upload** button to compile and upload the code to your ESP32.

5. Circuit Diagram

The circuit is straightforward since the ESP32/Arduino uno has built-in Bluetooth. No additional wiring is required beyond connecting the board to your computer using a USB cable.



6. Code Explanation

Code Overview

cpp

Copy code

```
#include "BluetoothSerial.h"

BluetoothSerial SerialBT;

void setup() {
  Serial.begin(115200);
  while (!Serial) {
    ; // Wait for Serial to initialize
  }

  if (!SerialBT.begin("ESP32_BT_Tracker")) {
    Serial.println("An error occurred initializing Bluetooth");
  } else {
    Serial.println("Bluetooth initialized successfully");
  }
}

void loop() {
  if (SerialBT.available()) {
    Serial.println("Bluetooth data available");
    String incomingData = "";
    while (SerialBT.available()) {
      incomingData += char(SerialBT.read());
    }
```

```
  Serial.print("Received: ");
  Serial.println(incomingData);
 }

 if (Serial.available()) {
  String outgoingData = "";
  while (Serial.available()) {
   outgoingData += char(Serial.read());
  }
  SerialBT.print(outgoingData);
 }
}
```

Explanation

- **Libraries and Initialization**:
  - The BluetoothSerial.h library is used to manage Bluetooth communication.
  - BluetoothSerial object SerialBT is initialized to handle Bluetooth operations.
- **Setup Function**:
  - Initializes the Serial Monitor at 115200 baud for debugging purposes.
  - Initializes the Bluetooth with a device name "ESP32_BT_Tracker".
  - Prints messages to the Serial Monitor indicating the success or failure of the Bluetooth initialization.
- **Loop Function**:
  - Continuously checks for available Bluetooth data. If available, it reads and displays the data on the Serial Monitor.
  - It also allows sending data from the Serial Monitor to the Bluetooth device.

7. Testing and Verification

Steps to Test

1. **Connect the ESP32** to your computer using a USB cable.
2. **Open the Serial Monitor** in the Arduino IDE and set the baud rate to 115200.

3. **Use a Bluetooth-enabled device** (such as a smartphone) to pair with the ESP32.
4. **Send data** from the Bluetooth device to the ESP32 and observe the output on the Serial Monitor.
5. **Type and send data** from the Serial Monitor and observe the reception on the paired Bluetooth device.

8. Troubleshooting

- **No Output on Serial Monitor**:
  - Ensure the correct COM port is selected.
  - Verify the baud rate matches the Serial.begin() value.
  - Check the USB cable and connections.
- **Bluetooth Not Pairing**:
  - Ensure the ESP32 is in discoverable mode.
  - Verify the device name in the code is correct and unique.
- **Data Not Being Sent or Received**:
  - Test with a different Bluetooth terminal app.
  - Ensure the device is within Bluetooth range.

9. Further Improvements

- Add a display (such as an OLED) to show detected devices without needing a computer.
- Implement filtering to identify suspicious devices based on predefined criteria.
- Enhance the user interface to display more information about each device, such as signal strength and device type.

10. References

- ESP32 Bluetooth Documentation
- Arduino ESP32 Board Installation

This documentation provides a complete overview of the project, making it easier to understand, set up, and maintain. If you have any further questions or need additional assistance, feel free to ask!