**Báo cáo hướng dẫn**

**Kết nối ESP32 với Raspberry Pi 4 qua Bluetooth**

Người thực hiện: Nguyễn Bá Việt Hoàng - B21DCDT100

Dự án này nhằm mục đích lập trình kết nối ESP32 với Raspberry Pi 4 qua giao thức Bluetooth để truyền dữ liệu giả lập rung động.

**Giới Thiệu**

Dự án này tập trung vào việc thiết lập kết nối Bluetooth giữa vi điều khiển ESP32 và Raspberry Pi 4. ESP32 được lập trình để giả lập dữ liệu rung động (gia tốc) và gửi đến Raspberry Pi 4 qua giao thức Bluetooth Low Energy (BLE). Raspberry Pi 4 chạy một script Python để nhận và xử lý dữ liệu này, đồng thời kiểm tra các giá trị rung động để phát hiện các trường hợp bất thường. Dự án có thể ứng dụng trong các hệ thống giám sát rung động hoặc IoT.

**Thông Số Kỹ Thuật**

| Thành phần | Thông tin |

|----------------|--------------------------------------|

| MCU | ESP32 |

| Nguồn vào | 5V |

| Kết nối | Bluetooth Low Energy (BLE) |

**Danh Sách Linh Kiện**

| Tên linh kiện | Số lượng | Ghi chú |

|--------------------------|----------|-------------------------------------------------------|

| ESP32 | 1 | Vi điều khiển chính, hỗ trợ BLE |

| Raspberry Pi 4 | 1 | Nhận và xử lý dữ liệu qua BLE |

| Dây nguồn USB | 2 | Cấp nguồn 5V cho ESP32 và Raspberry Pi 4 |

**Lập Trình Firmware**

Ngôn ngữ lập trình:

- ESP32: C++ (sử dụng Arduino framework và thư viện BLE).

- Raspberry Pi 4: Python (sử dụng thư viện `bleak` để nhận dữ liệu BLE).

Mô tả code:

- Code ESP32:

- Khởi tạo BLE Server trên ESP32 với UUID dịch vụ và đặc tính xác định trước.

- Giả lập dữ liệu rung động (gia tốc) ngẫu nhiên từ 0.0 đến 5.0 g.

- Gửi dữ liệu rung động qua BLE Characteristic dưới dạng chuỗi (VD: "VIBRATION:ERROR:2.45").

- Kích hoạt thông báo (notify) để gửi dữ liệu định kỳ đến Raspberry Pi 4.

- Hiển thị cảnh báo trên Serial Monitor nếu giá trị rung vượt ngưỡng 2.5 g.

- Code Raspberry Pi 4 (Python):

- Nhận dữ liệu từ ESP32 qua BLE sử dụng thư viện `bleak`.

- Hiển thị giá trị rung động nhận được và kiểm tra nếu giá trị vượt ngưỡng 2.5 g để in cảnh báo.

- Ví dụ code Python:

```python

import asyncio

from bleak import BleakClient

SERVICE\_UUID = "4fafc201-1fb5-459e-8fcc-c5c9c331914b"

CHARACTERISTIC\_UUID = "beb5483e-36e1-4688-b7f5-ea07361b26a8"

VIBRATION\_THRESHOLD = 2.5

async def run():

async with BleakClient("ESP32\_Fake\_MPU6050") as client:

print("Ket noi thanh cong voi ESP32")

while True:

data = await client.read\_gatt\_char(CHARACTERISTIC\_UUID)

data\_str = data.decode()

print(f"Du lieu nhan duoc: {data\_str}")

value = float(data\_str.split(":")[-1])

if value > VIBRATION\_THRESHOLD:

print("CANH BAO: Rung bat thuong!")

await asyncio.sleep(2)

loop = asyncio.get\_event\_loop()

loop.run\_until\_complete(run())

```

**Các bước nạp code:**

1. ESP32:

- Sử dụng Arduino IDE để biên dịch và nạp code.

- Cài đặt thư viện `ESP32 BLE Arduino` trong Arduino IDE.

- Kết nối ESP32 với PC qua cáp USB và chọn board ESP32 Dev Module.

- Nạp code [maincode.c](maincode.c) vào ESP32.

2. Raspberry Pi 4

- Cài đặt thư viện `bleak` bằng lệnh: `pip install bleak`.

- Chạy script Python trên Raspberry Pi 4 để nhận dữ liệu BLE.

---

**Cách Sử Dụng**

1. Khởi động hệ thống:

- Cấp nguồn 5V cho ESP32 và Raspberry Pi 4 qua cáp USB.

- Đảm bảo ESP32 đã nạp code và Raspberry Pi 4 đã cài đặt script Python.

2. Kết nối Bluetooth:

- Mở terminal trên Raspberry Pi 4 và chạy script Python.

- ESP32 sẽ tự động quảng bá tín hiệu BLE với tên thiết bị "ESP32\_Fake\_MPU6050".

- Raspberry Pi 4 sẽ tìm và kết nối với ESP32 qua UUID dịch vụ.

3. Nhận và xử lý dữ liệu:

- ESP32 gửi dữ liệu rung động giả lập mỗi 2 giây qua BLE.

- Raspberry Pi 4 nhận dữ liệu và hiển thị trên terminal.

- Nếu giá trị rung vượt ngưỡng 2.5 g, một thông báo cảnh báo sẽ được in ra.

4. Kiểm tra và debug:

- Mở Serial Monitor trên Arduino IDE (baud rate 115200) để kiểm tra trạng thái kết nối và dữ liệu gửi đi từ ESP32.

- Trên Raspberry Pi 4, kiểm tra terminal để đảm bảo dữ liệu được nhận đúng.