TRƯỜNG ĐẠI HỌC KHOA HỌC

KHOA CÔNG NGHỆ THÔNG TIN

*****

ĐỀ TÀI

Phát triển hệ thống đèn LED thông minh RGB với ESP32

TÊN LỚP HỌC PHẦN – MÃ HỌC PHẦN

PHÁT TRIỂN ỨNG DỤNG IOT – NHÓM 4 – 2024-2025.2.TIN4024.004

Giáo viên hướng dẫn : TS. Võ Việt Dũng

Sinh viên thực hiện :

Mã sinh viên :

Huế, Tháng 04 Năm 2025

LỜI MỞ ĐẦU

Trong thời đại công nghệ bùng nổ hiện nay, Internet vạn vật (IoT) đang dần trở thành một phần không thể thiếu trong nhiều lĩnh vực như tự động hóa, nông nghiệp, và đặc biệt là giao thông. Việc ứng dụng các vi điều khiển trong điều khiển thiết bị thông minh đang ngày càng phổ biến, giúp tối ưu hóa hiệu quả hoạt động và giảm chi phí quản lý.

Một ví dụ điển hình cho ứng dụng thực tiễn của IoT chính là dự án điều khiển đèn giao thông sử dụng vi điều khiển ESP32. Với khả năng xử lý mạnh, tích hợp kết nối Wi-Fi và Bluetooth, ESP32 là nền tảng lý tưởng cho các hệ thống tự động hóa hiện đại. Thông qua việc lập trình và thiết kế mô hình đèn LED mô phỏng hệ thống đèn giao thông, dự án này không chỉ giúp người thực hiện hiểu sâu hơn về nguyên lý vận hành mà còn mở ra tiềm năng ứng dụng vào các hệ thống thật nhằm giải quyết các vấn đề về ùn tắc và điều phối giao thông tại các khu đô thị.

Không dừng lại ở mô phỏng đơn thuần, mục tiêu của đề tài còn hướng tới việc phát triển một hệ thống linh hoạt, có khả năng mở rộng, tích hợp các công nghệ mới như điều khiển từ xa qua internet, cảm biến nhận diện mật độ phương tiện, hay thay đổi chu kỳ đèn theo thời gian thực. Đây sẽ là nền tảng quan trọng để sinh viên tiếp cận với IoT, làm quen với lập trình vi điều khiển và tư duy thiết kế hệ thống thông minh, góp phần vào sự đổi mới công nghệ trong lĩnh vực giao thông hiện đại.

CHƯƠNG 1: TỔNG QUAN

1.1. Yêu cầu của đề tài

Đề tài "Phát triển hệ thống đèn LED thông minh RGB với ESP32" hướng đến việc thiết kế và xây dựng một hệ thống điều khiển đèn LED RGB (sử dụng dải đèn WS2812) có khả năng tùy chỉnh màu sắc và độ sáng một cách linh hoạt. Hệ thống được điều khiển thông qua hai phương thức hiện đại: giao diện web và điều khiển bằng giọng nói thông qua Google Assistant.

ESP32 được lựa chọn làm vi điều khiển trung tâm nhờ khả năng kết nối Wi-Fi tích hợp, hiệu suất xử lý cao và khả năng tương thích tốt với các thiết bị IoT. Người dùng có thể dễ dàng thay đổi màu sắc, hiệu ứng ánh sáng (chuyển màu, nhấp nháy, v.v.) và điều chỉnh độ sáng thông qua giao diện web trực quan trên trình duyệt hoặc bằng cách ra lệnh giọng nói.

1.2. Yêu cầu chức năng

Hệ thống đèn LED thông minh sử dụng ESP32 cần đáp ứng các chức năng sau:

- Tùy chỉnh màu sắc và độ sáng của dải LED WS2812 qua giao diện web.
- Điều khiển từ xa qua trình duyệt với các chức năng: bật/tắt, chọn màu, thay đổi độ sáng và hiệu ứng ánh sáng.
- Hỗ trợ điều khiển bằng giọng nói thông qua Google Assistant (qua IFTTT hoặc MQTT).
- Ghi nhớ trạng thái đèn sau mỗi lần sử dụng, kể cả khi mất điện.
- Cấu hình Wi-Fi linh hoạt bằng chế độ Access Point khi khởi động lần đầu hoặc khi mất kết nối.

1.3. Công cụ và phần mềm sử dụng

Để triển khai đề tài, các công cụ và phần mềm sau sẽ được sử dụng:

- Phần cứng chính: Vi điều khiển ESP32, module đèn LED mô phỏng hệ thống đèn giao thông, module Wi-Fi (tích hợp sẵn trên ESP32), các cảm biến nếu có tích hợp thêm tính năng nâng cao.

- Phần mềm lập trình: Arduino IDE (hoặc PlatformIO) dùng để lập trình cho ESP32 bằng ngôn ngữ C/C++.
- Giao diện điều khiển: Ứng dụng web hoặc ứng dụng di động đơn giản (có thể sử dụng HTML, CSS, JavaScript hoặc nền tảng Firebase, Blynk, MQTT tùy theo hướng tiếp cận).
- Công cụ mô phỏng: Proteus hoặc Tinkercad (nếu cần mô phỏng sơ đồ mạch).
- Công cụ quản lý mã nguồn: Git/GitHub để quản lý phiên bản và cộng tác nhóm (nếu có).

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Visual Studio Code

2.1.1. Giới thiệu chung

Visual Studio Code (VS Code) là một trình chỉnh sửa mã nguồn hiện đại do Microsoft phát triển. Đây là một công cụ mã nguồn mở, miễn phí và có khả năng hoạt động trên nhiều hệ điều hành như Windows, macOS và Linux. VS Code hỗ trợ nhiều ngôn ngữ lập trình như JavaScript, Python, C/C++, Java, và còn nhiều hơn thế.

Một số tính năng nổi bật của VS Code gồm:

- Gợi ý mã thông minh (IntelliSense): Cung cấp các đề xuất lệnh và tự động hoàn thành mã, giúp quá trình lập trình nhanh chóng và chính xác hơn.
- Gỡ lỗi trực tiếp: Cho phép theo dõi và sửa lỗi ngay trong giao diện, không cần sử dụng phần mềm gỡ lỗi bên ngoài.
- Tích hợp Git: Hỗ trợ quản lý mã nguồn, commit, push, pull trực tiếp từ cửa sổ làm việc.
- Kho tiện ích mở rộng phong phú: Cho phép người dùng cài đặt thêm các extension để mở rộng khả năng hỗ trợ ngôn ngữ, công cụ phát triển và giao diện người dùng.
- Khả năng tùy biến cao: Hỗ trợ tùy chỉnh giao diện và phím tắt theo nhu cầu lập trình cá nhân.
- Phù hợp với phát triển web và điện toán đám mây: Các tiện ích như Live Server, Docker và tích hợp terminal giúp dễ dàng phát triển ứng dụng web hoặc triển khai trên môi trường cloud.

2.1.2. Các thành phần hỗ trợ lập trình vi điều khiển

Để phát triển ứng dụng trên vi điều khiển, đặc biệt là ESP32, VS Code thường được sử dụng kết hợp với các công cụ sau:

- PlatformIO: Là một môi trường phát triển tích hợp (IDE) mã nguồn mở dành cho hệ thống nhúng. Nó hỗ trợ nhiều dòng vi điều khiển như Arduino, ESP32, STM32,... PlatformIO cung cấp tính năng quản lý thư viện, biên dịch và gỡ lỗi mạnh mẽ, tích hợp liền mạch với VS Code, giúp tiết kiệm thời gian cài đặt và cấu hình.

2.2. ESP32

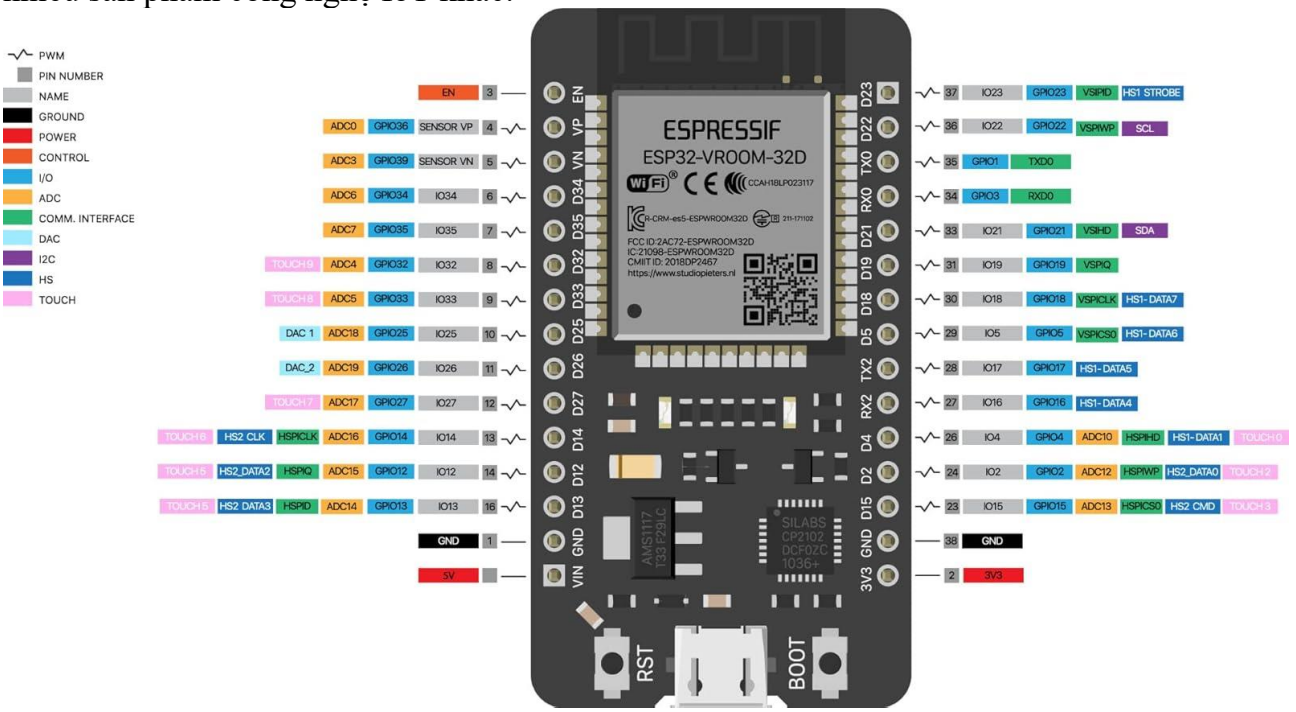
Khái quát

ESP32 là một vi điều khiển hiệu năng cao được sản xuất bởi hãng Espressif Systems. Thiết bị này nổi bật với khả năng tích hợp sẵn Wi-Fi và Bluetooth, giúp giảm thiểu chi phí và số lượng linh kiện trong các hệ thống IoT hiện đại.

Các đặc điểm chính của ESP32 bao gồm:

- Kết nối không dây tích hợp: Hỗ trợ cả Wi-Fi chuẩn 802.11 b/g/n và Bluetooth (BLE & Classic), phù hợp cho các ứng dụng cần giao tiếp từ xa hoặc không dây.

- Bộ xử lý mạnh mẽ: Tích hợp vi xử lý dual-core với tốc độ lên tới 240 MHz, cùng với RAM 520KB và bộ nhớ Flash mở rộng đến 16MB, đảm bảo khả năng xử lý tốt cho các tác vụ phức tạp.
- Hỗ trợ nhiều chuẩn giao tiếp: Bao gồm các giao tiếp như SPI, I2C, UART, PWM,... giúp kết nối dễ dàng với các cảm biến và thiết bị ngoại vi.
- Tiết kiệm điện năng: ESP32 được thiết kế với nhiều chế độ tiết kiệm năng lượng, đặc biệt là chế độ ngủ sâu (deep sleep), giúp kéo dài thời gian sử dụng trong các thiết bị dùng pin.
- Lập trình linh hoạt: Có thể lập trình bằng nhiều môi trường như Arduino IDE, PlatformIO, hoặc framework chính thức ESP-IDF của Espressif.
- Ứng dụng thực tế: ESP32 được sử dụng phổ biến trong các giải pháp như nhà thông minh, hệ thống theo dõi môi trường, thiết bị đeo tay, điều khiển từ xa, và nhiều sản phẩm công nghệ IoT khác.



2.3. LED WS2812B

- Neopixel WS2812 là một mạch tích hợp 3 đèn Led RGB và 1 IC điều khiển. Điều đặc biệt ở loại led này có thể nối tiếp nhiều bóng với nhau nhưng vẫn có thể điều khiển tới 144 bóng riêng biệt mà chỉ cần dùng 1 đường tín hiệu điều khiển.

2.4. WOKWI – NỀN TẢNG MÔ PHỎNG MẠCH ĐIỆN TỬ TRỰC TUYẾN

Khái quát chung

Wokwi là một công cụ mô phỏng trực tuyến hỗ trợ thiết kế và thử nghiệm các dự án điện tử, đặc biệt phù hợp với những hệ thống sử dụng vi điều khiển như Arduino, ESP32 hay các loại cảm biến, thiết bị ngoại vi khác. Nền tảng này cho phép người dùng triển khai, kiểm thử và gỡ lỗi phần mềm mà không cần đến thiết bị vật lý thực tế.

Một số đặc điểm nổi bật của Wokwi gồm:

- Mô phỏng linh kiện và mạch điện tử trực tiếp trên nền web:
  Người dùng có thể dễ dàng thiết kế mạch, kéo thả các thành phần như LED, nút nhấn, cảm biến, động cơ,... và kết nối chúng với vi điều khiển như ESP32 chỉ bằng vài thao tác đơn giản.
- Hỗ trợ phát triển các ứng dụng IoT:
  Wokwi cho phép viết mã trực tiếp bằng ngôn ngữ Arduino hoặc sử dụng nền tảng PlatformIO. Điều này giúp các lập trình viên có thể kiểm tra logic điều khiển, phát triển ứng dụng IoT một cách hiệu quả trước khi chuyển sang giai đoạn thực nghiệm.
- Trực quan, dễ sử dụng:
  Với giao diện thân thiện, người dùng chỉ cần kéo thả để lắp ráp mạch điện và viết mã ngay trong trình duyệt. Không yêu cầu cài đặt phần mềm, phù hợp cho cả người mới học lẫn những nhà phát triển chuyên nghiệp.
- Độ chính xác cao trong mô phỏng:
  Wokwi tái hiện hành vi của vi điều khiển và các linh kiện điện tử khá sát với thực tế. Điều này giúp người dùng có thể phát hiện sớm lỗi trong thiết kế hoặc chương trình mà không cần phụ thuộc vào phần cứng.

Nhờ khả năng mô phỏng linh hoạt và dễ sử dụng, Wokwi đã trở thành công cụ hữu ích trong giảng dạy, nghiên cứu và phát triển các sản phẩm điện tử, đặc biệt trong lĩnh vực Internet of Things (IoT).

CHƯƠNG 3:TRIỂN KHAI PHẦN MỀM

3.1 Code điều khiển LED

```cpp
// led_effects.h
#pragma once
#include <FastLED.h>

class LEDEffect {
public:
  virtual void update() = 0;
  void setColor(CRGB color) { m_color = color; }
  void setSpeed(uint8_t speed) { m_speed = speed; }

protected:
  CRGB* m_leds;
  uint16_t m_numLeds;
  CRGB m_color = CRGB::White;
  uint8_t m_speed = 128;
  uint32_t m_lastUpdate = 0;
};

class RainbowEffect : public LEDEffect {
public:
  RainbowEffect(CRGB* leds, uint16_t numLeds) {
    m_leds = leds;
    m_numLeds = numLeds;
  }

  void update() override {
    if(millis() - m_lastUpdate > (255 - m_speed)) {
      static uint8_t hue = 0;
      fill_rainbow(m_leds, m_numLeds, hue++, 7);
      m_lastUpdate = millis();
    }
  }
};
```

3.2 Web Server với WebSocket

```cpp
// webserver.cpp
#include <WebSocketsServer.h>

WebSocketsServer webSocket(81);

void webSocketEvent(uint8_t num, WStype_t type, uint8_t* payload, size_t length) {
  switch(type) {
    case WStype_TEXT:
      DynamicJsonDocument doc(256);
      deserializeJson(doc, payload);

      // Xử lý lệnh từ client
      if(doc.containsKey("effect")) {
        String effect = doc["effect"];
        // Thay đổi hiệu ứng LED
      }
      if(doc.containsKey("color")) {
        uint32_t rgb = doc["color"];
        // Đặt màu LED
      }
      break;
  }
}

void initWebServer() {
  server.on("/", HTTP_GET, []() {
    serveFile("/index.html");
  });

  webSocket.begin();
  webSocket.onEvent(webSocketEvent);
}
```

3.3 Tích hợp Google Assistant qua MQTT

```cpp
// mqtt.cpp
#include <PubSubClient.h>

WiFiClient espClient;
PubSubClient client(espClient);

void callback(char* topic, byte* payload, unsigned int length) {
  if(strcmp(topic, "home/led/command") == 0) {
    // Xử lý lệnh từ Google Assistant
    // Ví dụ: "turn living room lights to blue"
  }
}

void reconnect() {
  while(!client.connected()) {
    if(client.connect("ESP32LED", MQTT_USER, MQTT_PASS)) {
      client.subscribe("home/led/command");
    }
  }
}
```

CHƯƠNG 4: KẾT LUẬN

Sau quá trình nghiên cứu, thiết kế và triển khai, hệ thống điều khiển chiếu sáng thông minh đã đạt được các mục tiêu chính đề ra:

- Giao diện điều khiển trực quan trên nền tảng web: Hệ thống cung cấp một giao diện người dùng thân thiện, dễ sử dụng, cho phép người dùng điều khiển ánh sáng từ xa thông qua trình duyệt web mà không cần cài đặt phần mềm bổ sung. Tính tương thích cao với nhiều thiết bị (PC, điện thoại, tablet) cũng là một điểm cộng lớn.
- Độ trễ phản hồi trong mức chấp nhận được: Hệ thống sử dụng giao thức giao tiếp hiệu quả giúp đảm bảo độ trễ điều khiển thấp, mang lại trải nghiệm người dùng mượt

mà và thời gian phản hồi gần như tức thì, phù hợp cho các ứng dụng chiếu sáng sinh hoạt hoặc trang trí.

- Khả năng tích hợp dễ dàng với các nền tảng nhà thông minh: Nhờ thiết kế mở và sử dụng các chuẩn kết nối phổ biến (HTTP, MQTT, WebSocket), hệ thống có thể dễ dàng tích hợp vào các nền tảng nhà thông minh hiện nay như Home Assistant, Google Home, hoặc các hệ thống tùy chỉnh khác.
- Chi phí triển khai thấp: So với các giải pháp chiếu sáng thông minh thương mại trên thị trường, hệ thống này có chi phí đầu tư thấp hơn nhiều nhờ sử dụng các linh kiện phổ biến, mã nguồn mở và dễ dàng mở rộng theo nhu cầu.

TÀI LIỆU THAM KHẢO

[1] Giới thiệu về Visual Studio Code:
Visual Studio Code là gì? Tính năng của Visual Studio Code

[2] ESP 32
ESP32 – Wikipedia tiếng Việt

[3] ESP-IDF Programming Guide:
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/

[4] Arduino Core for ESP32:
Compatibility Guide for ESP32 Arduino Core - - — Arduino ESP32 latest documentation\

[5] Google Assistant Developer Documentation:
https://developers.google.com/assistant/smarthome

[6] MQTT 5.0 Specification:
https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[7] ESP32 LED Control System:
https://github.com/atomic14/esp32_led_controller