|  |
| --- |
| ­  TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  **VIỆN ĐIỆN TỬ - VIỄN THÔNG**  **-----□□&□□-----**  logo_128  **BÁO CÁO**  **KỸ THUẬT VI XỬ LÝ**  **ĐỀ TÀI: SMART HOME CONTROL**  **Giảng viên hướng dẫn:** TS, Hàn Huy Dũng  **Mã lớp :**123019  **Nhóm sinh viên :** Vũ Khắc Trung 20174294  Nguyễn Quang Huy 20180101  Lê Thành Công 20183491  Văn Hải Dương 20183512  Nguyễn Hoàng Phú 20180154  Hà Nội, 6/2021 |

**LỜI NÓI ĐẦU**

Vi xử lý hay bộ vi xử lý là một linh kiện điện tử máy tính được chế tạo từ các tranzitor thu nhỏ tích hợp lên trên một vi mạch tích hợp đơn. Khối xử lý trung tâm (CPU) là một bộ vi xử lý phổ biến ngoài ra nhiều thành phần khác trong máy tính cũng có bộ vi xử lý riêng của nó, ví dụ trên card màn hình chúng ta cũng có một bộ vi xử lý.

Trước khi xuất hiện các bộ vi xử lý, các CPU được xây dựng từ các mạch tích hợp cỡ nhỏ riêng biệt, mỗi mạch tích hợp chỉ chứa khoảng vài chục tranzitor. Do đó, một CPU là một bảng mạch gồm hàng ngàn hay hàng triệu vi mạch tích hợp. Ngày nay, công nghệ tích hợp đã phát triển, một CPU có thể tích hợp một hoặc vài vi mạch tích hợp cỡ lớn, mỗi vi mạch tích hợp cỡ lớn chứa hàng ngàn hoặc hàng triệu tranzitor. Nhờ đó [công suất tiêu thụ](https://vi.wikipedia.org/w/index.php?title=C%C3%B4ng_su%E1%BA%A5t_ti%C3%AAu_th%E1%BB%A5&action=edit&redlink=1" \o "Công suất tiêu thụ (trang chưa được viết)) và giá thành của bộ vi xử lý đã giảm đáng kể.

[Sự tiến hóa](https://vi.wikipedia.org/wiki/S%E1%BB%B1_ti%E1%BA%BFn_h%C3%B3a) của các bộ vi xử lý một phần nhờ vào việc chạy theo [Định luật Moore](https://vi.wikipedia.org/wiki/%C4%90%E1%BB%8Bnh_lu%E1%BA%ADt_Moore" \o "Định luật Moore) và hiệu suất của nó tăng lên một cách ổn định sau hàng năm. Định luật này phát biểu rằng sự phức tạp của một mạch tích hợp sẽ tăng lên gấp đôi sau mỗi chu kỳ 18 tháng. Và thực tế, sự phát triển của các bộ vi xử lý đã bám sát định luật này từ những [năm 1970](https://vi.wikipedia.org/w/index.php?title=N%C4%83m_1970&action=edit&redlink=1" \o "Năm 1970 (trang chưa được viết)). Nhờ đó, từ [máy tính mẹ](https://vi.wikipedia.org/w/index.php?title=M%C3%A1y_t%C3%ADnh_m%E1%BA%B9&action=edit&redlink=1" \o "Máy tính mẹ (trang chưa được viết)) (*mainframe computer*) lớn nhất cho đến các [máy tính xách tay](https://vi.wikipedia.org/wiki/M%C3%A1y_t%C3%ADnh_x%C3%A1ch_tay" \o "Máy tính xách tay) hiện nay đều sử dụng một bộ vi xử lý rất nhỏ tại trung tâm của chúng.

Lý thuyết kĩ thuật vi xử lý thông qua bộ vi xử lý 8086 và các mạch liên quan của Intel đã cung cấp cho chúng em các khái niệm chủ yếu về một hệ vi xử lý 16 bit: cấu trúc và nguyên tắc hoạt động của một hệ vi xử lý gồm CPU và các mạch phụ trợ, tập lệnh của bộ vi xử lý, cách lập trình bằng hợp ngữ, các phương thức điều khiển việc vào/ra dữ liệu trong hệ vi xử lý và cuối cùng là cách thực hiện một số phối ghép cơ bản trong hệ vi xử lý.

Để hiểu rõ hơn về những lý thuyết đã học trong kĩ thuật vi xử lý chúng em tiến hành thực hiện đề tài thiết kế hệ thống điều khiển thiết bị gia đình sử dụng vi điều khiển. Đồng thời từ việc thực hiện đề tài giúp chúng em nâng cao khả năng làm việc nhóm, khả năng thiết kế, thực hành và rút ra những kinh nghiệm thực tiễn trong quá trình thiết kế một sản phẩm kĩ thuật. Chi tiết thiết kế sản phẩm được chúng em trình bày rõ hơn ở phần sau.

Cuối cùng, chúng em xin chân thành cảm ơn thầy Hàn Huy Dũng đã tận tình hướng dẫn chúng em trong suốt quá trình học tập và thực hiện đề tài. Tuy chúng em đã rất cố gắng để hoàn thành báo cáo nhưng không thể tránh khỏi những sai sót. Chúng em rất mong nhận được sự nhận xét đánh giá, góp ý từ thầy để bài báo cáo của chúng em được hoàn thiện hơn.

**MỤC LỤC**

[**DANH MỤC HÌNH ẢNH** i](#_Toc78534334)

[**DANH MỤC BẢNG BIỂU** ii](#_Toc78534335)

[**CHƯƠNG 1. TÌM HIỂU ĐỀ TÀI** 1](#_Toc78534336)

[1.1 Đặt vấn đề 1](#_Toc78534337)

[1.2 Mục tiêu và phạm vi đề tài 1](#_Toc78534338)

[**CHƯƠNG 2. XÁC ĐỊNH YÊU CẦU HỆ THỐNG** 3](#_Toc78534339)

[2.1 Khảo sát các sản phẩm tương tự có trên thị trường 3](#_Toc78534340)

[2.2 Yêu cầu chức năng 4](#_Toc78534341)

[2.3 Yêu cầu phi chức năng 4](#_Toc78534342)

[**CHƯƠNG 3. LẬP KẾ HOẠCH** 6](#_Toc78534343)

[3.1 Phân tích nhân lực 6](#_Toc78534344)

[3.2 Phân chia công việc cho các thành viên 6](#_Toc78534345)

[3.3 Timeline của đề tài 7](#_Toc78534346)

[**CHƯƠNG 4. THIẾT KẾ VÀ LỰA CHỌN PHƯƠNG ÁN TỐI ƯU CHO TỪNG KHỐI** 9](#_Toc78534347)

[4.1 Sơ đồ khối hệ thống 9](#_Toc78534348)

[4.2 Khối xử lý trung tâm 10](#_Toc78534349)

[4.2.1 Lựa chọn vi xử lý 10](#_Toc78534350)

[4.2.2 Sơ đồ nguyên lý của khối vi xử lý 11](#_Toc78534351)

[4.4 Khối relay 14](#_Toc78534352)

[4.4.1 Lựa chọn relay 14](#_Toc78534353)

[4.4.2 Sơ đồ kết nối relay 15](#_Toc78534354)

[4.5 Khối nguồn 16](#_Toc78534355)

[4.5.1 Lựa chọn các giá trị của nguồn 16](#_Toc78534356)

[4.5.2 Lựa chọn loại nguồn tối ưu 16](#_Toc78534357)

[4.5.3 Sơ đồ mạch nguồn 3V3 18](#_Toc78534358)

[4.6 Khối hiển thị 19](#_Toc78534359)

[**CHƯƠNG 5. THIẾT KẾ HỆ THỐNG** 21](#_Toc78534360)

[5.1 Thiết kế phần cứng 21](#_Toc78534361)

[5.1.1 Thiết kế PCB khối 1 22](#_Toc78534362)

[5.1.2 Thiết kế PCB khối 2 23](#_Toc78534363)

[5.2 Thiết kế phần mềm 25](#_Toc78534364)

[5.2.1 Công nghệ sử dụng 25](#_Toc78534365)

[5.2.2 Thiết kế chương trình 26](#_Toc78534366)

[**CHƯƠNG 6. TEST VÀ KIỂM THỬ** 33](#_Toc78534367)

[**CHƯƠNG 7. KẾT LUẬN** 35](#_Toc78534368)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 36](#_Toc78534369)

[**PHỤ LỤC** 37](#_Toc78534370)

# **DANH MỤC HÌNH ẢNH**

[Hình 4. 1 Sơ đồ khối hệ thống 9](#_Toc78534371)

[Hình 4. 2 ESP32 10](#_Toc78534372)

[Hình 4. 3 ESP 8266 10](#_Toc78534373)

[Hình 4. 4 Sơ đồ nguyên lý khối vi điều khiển 12](#_Toc78534374)

[Hình 4. 5 Sơ đồ kết nối chân IN, OUT của vi điều khiển 12](#_Toc78534375)

[Hình 4. 6 Sơ đồ nguyên lý mạch nút nhấn 13](#_Toc78534376)

[Hình 4. 7 Sơ đồ khối LED báo 14](#_Toc78534377)

[Hình 4. 8 Relay HUIKE 6P 3A 14](#_Toc78534378)

[Hình 4. 9 Relay SONGLE 5P 10A 14](#_Toc78534379)

[Hình 4. 10 Sơ đồ kết nối khối relay 15](#_Toc78534380)

[Hình 4. 11 Adapter nguồn 5V -1A 17](#_Toc78534381)

[Hình 4. 12 Module nguồn 5V-1A 17](#_Toc78534382)

[Hình 4. 13 Module nguồn HiLink 5V-1A 17](#_Toc78534383)

[Hình 4. 14 Sơ đồ mạch nguồn 5V 18](#_Toc78534384)

[Hình 4. 15 IC nguồn AMS1117 19](#_Toc78534385)

[Hình 4. 16 Sơ đồ kết nối IC nguồn 19](#_Toc78534386)

[Hình 4. 17 Mô hình Các thành phần của Blynk 20](#_Toc78534387)

[Hình 5. 1 Schematic của khối thứ 1 22](#_Toc77330569)

[Hình 5. 2 Mạch PCB của khối thứ 1 23](#_Toc77330570)

[Hình 5. 3 Hình ảnh 3D khối thứ 1 23](#_Toc77330571)

[Hình 5. 4. Schematic của khối mạch 24](#_Toc77330572)

[Hình 5. 5 Mạch PCB của khối thứ 2 (top viewer) 25](#_Toc77330573)

[Hình 5. 6 Mạch PCB của khối thứ 2 (bottom viewer) 25](#_Toc77330574)

[Hình 5. 7: Giao diện Blynk 31](#_Toc77330575)

# **DANH MỤC BẢNG BIỂU**

[Bảng 2. 1 Thông tin về các sản phẩm công tắc thông minh trên thị trường 3](#_Toc77330640)

[Bảng 3. 1 Phân tích năng lực của các thành viên 6](#_Toc77330645)

[Bảng 3. 2 Phân chia công việc theo các thành viên 6](#_Toc77330646)

[Bảng 3. 3 Timeline của đề tài 7](#_Toc77330647)

[Bảng 4. 1 So sánh ESP32 và ESP8266 12](#_Toc77330652)

[Bảng 4. 2 : So sánh hai loại Relay HUIKE và SONGLE 16](#_Toc77330653)

[Bảng 4. 3 Yêu cầu về các thiết bị dùng nguồn 18](#_Toc77330654)

[Bảng 4. 4 So sánh các loại nguồn 21](#_Toc77330655)

[Bảng 5. 1 Danh sách linh kiện 24](#_Toc77330656)

# **CHƯƠNG 1. TÌM HIỂU ĐỀ TÀI**

1. Đặt vấn đề

Cùng với sự phát triển nhanh chóng của xã hội ngày nay, những nhu cầu về một cuộc sống tiện nghi, văn minh đang ngày càng lớn. Đi kèm với đó, sự phát triển mạnh mẽ của các ngành khoa học kỹ thuật và kỹ thuật điều khiển tự động, giúp cung cấp các giải pháp công nghệ, đặc biệt là IoT, ngày càng hiện đại, đa dạng và có khả năng ứng dụng thực tiễn vào cuộc sống nhằm cải thiện chất lượng sống của con người. Khi công nghệ ngày càng phát triển mạnh, đi kèm với đó là các vấn đề liên quan đến cải thiện chất lượng sống cũng ngày càng được quan tâm, và một trong số đó là việc quản lý điện năng trong gia đình, trong đó vấn đề đơn giản đó là việc kiểm soát việc bật tắt các thiết bị điện trong gia đình. Có rất nhiều tình huống xảy ra như khi chúng ta quên tắt một thiết bị trong gia đình, mà khi ta nhớ ra thì lại đang không có tại nhà để tắt đi các thiết bị đó, hay như việc chúng ta muốn tắt các thiết bị khác trong gia đình trong khi chúng ta đang ở một khu vực khác trong gia đình mà chúng ta không muốn đến tận nơi để thực hiện công việc này, … rất nhiều tình huống xảy ra liên quan đến việc bật tắt các thiết bị. Từ đó, nhu cầu về một giải pháp công nghệ giúp con người có thể kiểm soát việc hoạt động của các thiết bị trong gia đình từ xa. Bên cạnh đó, sự phát triển mạnh mẽ của các loại thiết bị điện tử ngày nay như smartphone, laptop, … và sự phủ sóng ngày càng lớn của internet, các loại công tắc thông minh được phát triển nhằm giúp con người có thể điều khiển được khả năng bật tắt của các thiết bị từ xa thông qua mạng internet. Xuất phát từ những lý do trên, nhóm chúng em đã quyết định thực hiện đề tài “Smarthome Control” nhằm đưa ra một giải pháp để phát triển các công tắc thông minh.

1. Mục tiêu và phạm vi đề tài

Với đề tài “Smarthome Control”, nhóm chúng em thực hiện thay thế công tắc thông minh thay cho công tắc cơ truyền thống, điều này giúp quản lý thiết bị từ xa, sử dụng thiết bị hiệu quả với việc có thể hẹn giờ bật tắt các thiết bị trong gia đình. Bởi hiện nay, hầu như mọi người đều có cho mình các thiết bị điện tử thông minh như smartphone, laptop…Vì vậy với việc phát triển thêm giải pháp đi kèm như “Smarthome Control” sẽ giúp tối ưu hóa giá trị sử dụng của các thiết bị điện tử thông minh đồng thời cũng có thể giúp chúng ta quản lý năng lượng tiêu thụ của các thiết bị điện trong gia đình tốt hơn. Mục tiêu của đề tài đặt ra đó là phát triển một sản phẩm có khả năng thực hiện được các chức năng cơ bản đã được nêu phía trên. Bên cạnh đó sản phẩm vẫn có thể đảm bảo được việc bật tắt trực tiếp thiết bị.

Nội dung đề tài gồm có 6 phần chính như sau: phần một là tìm hiểu đề tài, phần hai là xác định yêu cầu hệ thống, phần ba là lập kế hoạch, phần bốn là thiết kế sơ đồ khối và thiết kế chi tiết các khối và lựa chọn phương án tối ưu cho từng khối, phần năm là test mạch và kiểm thử, phần sáu là hoàn thiện sản phẩm và cuối cùng là bàn giao sản phẩm.

# **CHƯƠNG 2. XÁC ĐỊNH YÊU CẦU HỆ THỐNG**

2.1 Khảo sát các sản phẩm tương tự có trên thị trường

Trên thị trường hiện nay, các sản phẩm công tắc thông minh đang được phát triển rất nhanh với nhiều đặc điểm kỹ thuật nổi bật. Bảng 2.1 dưới đây sẽ đưa ra các thông tin về một số loại công tắc thông minh có trên thị trường.

**Bảng 2. 1 Thông tin về các sản phẩm công tắc thông minh trên thị trường**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sản phẩm**  **Danh mục** | Công tắc thông minh kèm ổ cắm Hunonic | Công tắc thông minh Javis | Công tắc thông minh Mi Aquara | Công tắc đèn điện tử Sonoff 4CH R2 | Công tắc wifi cảm ứng Tuya 4 nút |
| Thương hiệu | Hunonic | Javis | MI Aqara | Sonoff | Tuya |
| Giá thành | 550.000-690.000đ | 830.000đ | 588.000đ | 653.000đ | 497.000đ |
| Kích thước | 12 x 7.2 x 3.5 cm | 12x7.2x3.6 cm | 8.6 x 8.6 x 4.3 cm | 18x 10 x 12 cm | 11.5x 7.5x3.5 cm |
| Dòng tải tối đa | 10A | 10A | 16A | 16A | 10A |
| Điện áp | AC 90-250V, tần số 50-60Hz | AC 100V-250, tần số 50-60Hz | AC 100 - 220V, tần số 50Hz. | AC 90-240V, tần số 50-60Hz | AC 110-250V, tần số 50-60Hz |
| Kết nối | Wifi 2.4GHz, 3G, 4G | Wifi 2.4GHz | Wifi 2.4Ghz | Wifi 2.4Ghz | Wifi 2.4Ghz IEEE 802.11b/g/n. |
| Ứng dụng điều khiển | Hunonic | Javis Home | Không dây ZigBee, tương thích với bộ sản phẩm Home Kit | EweLink | Tuya và Smartlife |
| Số kênh điều khiển (số thiết bị mà sản phẩm có thể điều khiển) | 2 kênh | 3 kênh | 2 kênh | 4 kênh | 4 kênh |
| Ưu điểm | **Thiết bị có khả năng bật tắt bằng cảm ứng chạm, có thể điều khiển các thiết bị điện trong nhà ở bất cứ nơi đâu chỉ cần có sóng Wifi, 3G, 4G**. **Ngoài ra có thể điều khiển công tắc thông minh qua giọng nói nhờ Alexa và Google Assistant** | Thiết bị có khả năng bật tắt bằng cảm ứng chạm, công tắc kết nối wifi, **có thể điều khiển từ xa qua app Javis Home bằng tiếng việt,** **điều khiển bằng giọng nói qua Google Home và tiếng Anh qua Alexa,**hoạt động một cách độc lập không cần bộ điều khiển trung tâm.  Máy chủ Cloud của công tắc được đặt tại Việt Nam, giúp **cho tốc độ tải dữ liệu luôn được đảm bảo nhanh chóng** và an toàn | Thiết bị có thể điều khiển đèn từ 3W-1500W trở lên  **Khi nhiệt độ vượt mức cho phép khoảng 60 độ C, sản phẩm sẽ tự động ngắt nguồn điện để đảm bảo an toàn**. Ngoài ra thể**kết nối với điện thoại thông minh hoặc các thiết bị đến từ thương hiệu Xiaomi** | Thiết bị có thể điều khiển 4 kênh giúp tiết kiệm chi phí, điều khiển **thông qua APP EweLink**. Hoặc **sử dụng giọng nói để điều kiện**hoạt động của công tắc trên nền tảng Amazon Alexa, Google Nest và cả Google Home. D**ễ dàng hẹn giờ bật/tắt cho hệ thống đèn** có công suất dao động từ 2200W đến 3500W. | Thiết bị có thể điều khiển 4 kênh giúp tiết kiệm chi phí, sản phẩm có khả năng hỗ trợ người dùng **kết nối với Google Home, Amazon Alexa, IFTTT để điều khiển bằng giọng nói, sản phẩm có h**ệ thống đèn led báo trạng thái hoạt động chuẩn xác,**khá tiết kiệm điện năng.** |
| Khuyết điểm | Sản phẩm chỉ có 2 kênh mà giá thành cao. Công suất tối đa chỉ là 150W/kênh | Khó lắp đặt và sửa chữa. Sử dụng qua Google home sẽ chậm hơn vài giây so với app của hãng | Sản phẩm chỉ có 2 kênh mà giá thành cao. | Tuy nhiên máy chủ Cloud của công tắc lại đặt tại Trung Quốc khiến sản phẩm hoạt động không ổn định. Thiết kế lớp mặt nạ nhựa bên ngoài hơi thô. | Bộ **4 nút bấm có độ mượt không cao**và tín hiệu đường truyền đôi khi không ổn định. |

Dựa vào việc tìm hiểu một số loại sản phẩm đã có trên thị trường được trình bày trong bảng 2.1 trên, nhóm đã đưa các yêu cầu chức năng và phi chức năng cần có cho sản phẩm, các yêu cầu này sẽ được trình bày kĩ trong phần 2.2 và 2.3.

* 1. Yêu cầu chức năng

Sản phẩm có chức năng bật tắt các thiết bị điện trong nhà thông qua điều khiển nút nhấn cảm ứng. Người dùng có thể sử dụng tính năng này bằng cách thao tác trực tiếp với nút nhấn cảm ứng hoặc điều khiển thông qua giao diện app được cài đặt trên điện thoại của người dùng. Người dùng có thể biết được trạng thái hiện tại của các thiết bị trong nhà thông qua đèn báo trên nút nhấn hoặc tại giao diện app. Sản phẩm được kỳ vọng có khả năng điều khiển được từ 4 thiết bị trở lên. Sản phẩm còn có chức năng cài đặt cho thiết bị để kết nối vào mạng wifi. Khi người dùng nhấn 1 nút nhấn (tổ hợp nút nhấn) cụ thể nào đó, người dùng hoặc công nhân lắp đặt có thể truy cập vào thiết bị để thay đổi thông tin cần thiết như tên wifi, mật khẩu wifi. Sản phẩm có chức năng lưu các thông tin cần thiết như tên wifi, mật khẩu wifi vào EEPROM để tránh bị mất thông tin khi thiết bị reset do sự cố nguồn điện. Sản phẩm có khả năng hẹn giờ để bật tắt thiết bị. Ngoài ra sản phẩm cần có phương án dự phòng mở rộng trong tương lai khi kết nối với bộ điều khiển trung tâm, trợ lý ảo, voice control.

* 1. Yêu cầu phi chức năng

Sản phẩm có khả năng hoạt động trong điều kiện môi trường bên ngoài có nhiệt độ trong khoảng từ 10˚C-40˚C, độ ẩm cao có thể lên đến 90%, phù hợp với khí hậu nhiệt đới gió mùa. Do mạng điện dân dụng tại Việt Nam là 220V 50Hz, điện áp lắp đặt đầu vào duy nhất của thiết bị 220V AC 50Hz, điện áp đầu ra thiết bị 220V AC 50Hz. Bên cạnh đó, sản phẩm cần được thiết kế nhỏ gọn, có kích thước khoảng 15 x10 x5 cm, có tính tích hợp và tính linh động cao, dễ dàng lặp đặt, sửa chữa và nâng cấp sau này. Giao diện phần mềm đơn giản, giúp người dùng nhanh làm quen và dễ sử dụng. Giá thành sản phẩm dưới 500.000đ.

# **CHƯƠNG 3. LẬP KẾ HOẠCH**

1. Phân tích nhân lực

Nhằm phát triển đề tài “Smarthome control” dựa trên những kiến thức đã học trong bộ môn Kỹ thuật vi xử lý cùng những kiến thức liên quan cần được tìm hiểu để thực hiện đề tài, vì vậy nhóm đã thảo luận và đưa ra các điểm mạnh và điểm yếu của từng thành viên như trong Bảng 3.1

**Bảng 3. 1 Phân tích năng lực của các thành viên**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **Điểm mạnh** | **Điểm yếu** |
| **1** | Lê Thành Công | * Điện tử tương tự * Lập trình Arduino * Teamwork tốt |  |
| **2** | Văn Hải Dương | * Lập trình C * Thiết kế cơ khí | Teamwork |
| **3** | Nguyễn Quang Huy | * Điện tử số * Lập trình C * Teamwork tốt |  |
| **4** | Nguyễn Hoàng Phú | * Từng làm việc với arduino * Thiết kế mạch | Teamwork |
| **5** | Vũ Khắc Trung | * Thiết kế mạch * Lập trình Arduino, C * Layout PCB * Teamwork tốt | Hạn chế về thời gian |

1. Phân chia công việc cho các thành viên

Dựa trên việc phân tích các điểm mạnh và điểm yếu của các thành viên trong nhóm như trong Bảng 3.1, nhóm đã phân công cụ thể công việc cho các thành viên như được đề cập trong Bảng 3.2

**Bảng 3. 2 Phân chia công việc theo các thành viên**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nhóm công việc** | **Công việc cụ thể** | **Nhân lực** | **Ghi chú** |
| **Phân tích đề tài** | Tìm hiểu thị trường, các sản phẩm đã có | cả nhóm |  |
| Định hướng sản phẩm, đưa ra mô hình giải quyết bài toán |  |
| Phân tích các yêu cầu |  |
| **Thiết kế phần cứng** | Thiết kế sơ đồ khối | cả nhóm | Sau khi thiết kế từ tổng quan đến chi tiết cần đưa ra được schematic của khối, các linh kiện sẽ sử dụng, datasheet của linh kiện |
| Thiết kế khối nguồn và điều khiển | Trung + Công |
| Thiết kế khối xử lý trung tâm | Trung + Dương + Phú |
| Thiết kế nút nhấn cảm ứng | Huy + Phú |
| Thiết kế thư viện của linh kiện và layout | Trung |
| **Lập trình phần mềm** | Đọc nút nhấn, kết nối với app Blynk | Công + Huy | Lập trình bám sát vào các yêu cầu chức năng đã đặt ra |
| Điều khiển led và relay | Dương + Phú |
| Trình khởi động, đọc ghi dữ liệu và EEPROM | Trung |
| **Hoàn thiện sản phẩm** | Đặt gia công PCB | Trung |  |
| Tìm mua linh kiện | Huy |  |
| Thiết kế vỏ | Dương |  |
| Nạp code test hoàn thiện | cả nhóm |  |

1. Timeline của đề tài

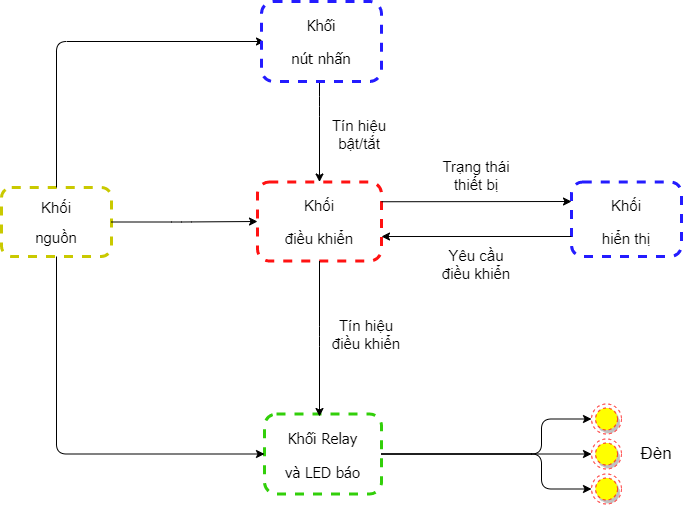
Khi đã xác định các công việc của thể cho từng thành viên, nhóm đã đưa lên timeline cho dự án như trong bảng 3.3

**Bảng 3. 3 Timeline của đề tài**

|  |  |  |  |
| --- | --- | --- | --- |
| **Công việc** | **Thời gian bắt đầu** | **Thời gian kết thúc** | **Thành viên thực hiện** |
| **1. Tìm hiểu các sản phẩm trên thị trường** |  |  |  |
| 1.1 Tìm hiểu các sản phẩm công tắc thông minh trên thị trường | 24/4/2021 | 25/4/2021 | Phú, Huy |
| 1.2 Phân tích các thông số kỹ thuật của các công tắc thông minh đã tìm hiểu | 24/4/2021 | 25/4/2021 | Dương, Công |
| **2. Định hướng sản phẩm, đưa ra mô hình giải quyết bài toán** |  |  |  |
| 2.1 Định hướng sản phẩm | 25/4/2021 | 26/4/2021 | Cả nhóm |
| 2.1 Đưa ra mô hình giải quyết bài toán | 25/4/2021 | 26/4/2021 | Cả nhóm |
| **3, Xác định yêu cầu kỹ thuật** |  |  |  |
| 3.1 Yêu cầu chức năng | 26/4/2021 | 2/5/2021 | Trung, Dương |
| 3.2 Yêu cầu phi chức năng | 26/4/2021 | 2/5/2021 | Trung |
| 3.3 Xác định sơ đồ khối của hệ thống | 26/4/2021 | 2/5/2021 | Trung |
| 3.4 Lên danh sách các linh kiện cần dùng | 26/4/2021 | 3/5/2021 | Công, Phú |
| **4, Tìm hiểu về các linh kiện** |  |  |  |
| 4.1 Tìm hiểu vi xử lý ESP32 và ESP8266 | 4/5/2021 | 8/5/2021 | Trung, Công |
| 4.2 Tìm hiểu về nút nhấn TTP223 | 4/5/2021 | 8/5/2021 | Huy |
| 4.3 Tìm hiểu các loại cung cấp nguồn | 4/5/2021 | 8/5/2021 | Dương |
| 4.4 Tìm hiểu về relay | 4/5/2021 | 8/5/2021 | Phú |
| **5. Lựa chọn tối ưu cho từng khối** |  |  |  |
| 5.1 Lựa chọn linh kiện cho khối vi xử lý | 10/5/2021 | 15/5/2021 | Trung |
| 5.2 Lựa chọn linh kiện cho khối nguồn | 10/5/2021 | 15/5/2021 | Trung |
| 5.3 Xác định các yêu cầu kỹ thuật của nút nhấn | 10/5/2021 | 15/5/2021 | Huy |
| 5.4 Lựa chọn linh kiện cho khối relay | 10/5/2021 | 15/5/2021 | Trung |
| **6. Thiết kế schematic và PCB** |  |  |  |
| 6.1 Thiết kế schematic | 15/5/2021 | 31/5/2021 | Huy |
| 6.2 Thiết kế PCB | 15/5/2021 | 31/5/2021 | Trung |
| **7. Lập trình** |  |  |  |
| 7.1 Đọc nút nhấn, kết nối với app Blynk | 15/5/2021 | 20/6/2021 | Trung |
| 7.2 Điều khiển LED và relay | 15/5/2021 | 20/6/2021 | Huy, Dương |
| 7.3 Trình khởi động, đọc ghi dữ liệu vào EPROM | 15/5/2021 | 20/6/2021 | Công, Phú |
| **8. Thiết kế sản phẩm** |  |  |  |
| 8.1 Đặt mua linh kiện | 15/5/2021 | 22/6/2021 | Trung |
| 8.2 Nạp code, test sản phẩm, sửa chữa sản phẩm | 20/6/2021 | 7/7/2021 | Cả nhóm |
| 8.3 Hoàn thiện sản phẩm, báo cáo | 10/7/2021 | 14/7/2021 | Cả nhóm |
| 8.4 Quay video demo | 14/7/2021 | 15/7/2021 | Trung |

# **CHƯƠNG 4. THIẾT KẾ VÀ LỰA CHỌN PHƯƠNG ÁN TỐI ƯU CHO TỪNG KHỐI**

1. Sơ đồ khối hệ thống



**Hình 4. 1 Sơ đồ khối hệ thống**

Hình 4.1 thể hiện tổng quan về hệ thống. Hệ thống bao gồm 5 khối chính:

* Khối nguồn có nhiệm vụ chuyển đổi điện áp đầu vào và cung cấp điện năng phù hợp cho hệ thống
* Khối điều khiển: thực hiện xử lý thông tin nhận được từ khối nút nhấn và yêu cầu nhận được từ server, từ đó đưa ra tín hiệu điều khiển tới khối relay và led báo.
* Khối nút nhấn có nhiệm vụ nhận tương tác trực tiếp từ người dùng và gửi tín hiệu đó đến khối điều khiển
* Khối relay và LED báo: thực hiện nhận tín hiệu từ khối điều khiển và thực hiện điều khiển các thiết bị điện.
* Khối hiển thị: nhận tín hiệu về trạng thái của thiết bị từ khối điều khiển và đưa thông tin đến giao diện app, từ đó nhận lại yêu cầu điều khiển từ người dùng và gửi lại thông tin về khối điều khiển.

1. Khối xử lý trung tâm

### ***4.2.1 Lựa chọn vi xử lý***

Dựa theo yêu cầu chức năng của sản phẩm, ta có thể xác định các yêu cầu cho khối xử lý trung tâm như sau: có thể kết nối wifi, tối thiểu có 12 GPIO (6 INPUT và 6 OUTPUT), có tích hợp sẵn antenna trên chip và tiết kiệm năng lượng (dòng tiêu thụ <300mA).

Dựa trên các yêu cầu trên, nhóm đã lựa chọn hai phương án vi điều khiển đó là ESP32 và ESP8266. Một số đặc điểm so sánh giữa hai phương án được thể hiện trong bảng 4.1

A close-up of a hard drive

Description automatically generated with low confidence

**Hình 4. 2 ESP32**

A picture containing text, electronics, circuit

Description automatically generated

**Hình 4. 3 ESP 8266**

**Bảng 4. 1 So sánh ESP32 và ESP8266**

|  |  |  |
| --- | --- | --- |
| **Vi điều khiển**  **Tiêu chí** | ESP32 | ESP8266 |
| Giá thành | 98.000 vnđ | 48.000 vnđ |
| Số GPIO khả dụng | 34 | 12 |
| Dòng tiêu thụ | Dòng duy trì 40mA, dòng cho mỗi đầu ra 28mA | Tại công suất phát cao nhất 170mA |
| Băng tần | 2.4GHz | 2.4GHz |
| Tần số thạch anh | 40MHz | 26MHz |

Dựa vào Bảng 4.1, sau khi xác định và tối ưu các yêu cầu cần thiết, nhóm lựa chọn MCU ESP8266 với 3 đặc điểm: giá thành rẻ, đủ số lượng GPIO yêu cầu, dòng tiêu thụ thấp. Ngoài ra, vi điều khiển được thiết kế dạng module tích hợp sẵn thạch anh, chip flash, antenna và chống nhiễu cho mạch nên sản phẩm có hoạt động ổn định.

***4.2.2 Sơ đồ nguyên lý của khối vi xử lý***

Trong khối vi xử lý sẽ bao gồm khối vi điều khiển, các điện trở treo, các tụ lọc nhiễu. Hình 4.4 và 4.5 thể hiện hai các khối trên schematic của khối vi điều khiển, điện trở treo, các tụ lọc nhiễu và sơ đồ kết nối các chân IN, OUT của thiết bị.

Diagram, text, schematic

Description automatically generated

**Hình 4. 4 Sơ đồ nguyên lý khối vi điều khiển**

Table

Description automatically generated

**Hình 4. 5 Sơ đồ kết nối chân IN, OUT của vi điều khiển**

Các tụ C0, C1, C2, C3 có nhiệm vụ lọc nhiễu đầu vào cho vi điều khiển. Các trở R0-R4 có nhiệm vụ enable, flash mode, trở cheo chân reset.

Các chân GPIO 16,14,12,1,3,5 lần lượt được chọn làm các chân từ OUT1 đến OUT6 để nhận tín hiệu từ nút nhấn cảm ứng.

Các chân GPIO 13,10,15,2,0,4 lần lượt được chọn làm các chân từ IN1 đến IN6 để gửi tín hiệu điều khiển cho khối các LED báo và 6 relay.

**4.3 Khối nút nhấn và khối LED báo**

Nút nhấn có nhiệm vụ nhận tín hiệu trực tiếp từ người dùng và gửi lên vi điều khiển. Nút nhấn cần phải gửi tín hiệu chính xác, có độ bền từ 10-20 năm, đồng thời có tính thẩm mỹ và đem lại cảm giác sang trọng và hiện đại. Vì vậy nhóm trực tiếp lựa chọn phương pháp thiết kế nút nhấn cảm ứng thay cho nút nhấn cơ học thông thường vì 2 ưu điểm: không hạn chế số lần nhấn, có tính thẩm mỹ và sự sang trọng. Nhóm quyết định chọn nút nhấn TTP223 (thông số được đinh kèm trong danh mục tham khảo) đáp ứng được các yêu cầu trên. Bên cạnh đó khối LED báo chính là các đèn LED đi cùng với khối nút nhấn giúp hiển thị trạng thái của thiết bị. Nhóm chọn loại đèn LED SMD 0805 (thông số được đính kèm trong danh mục tham khảo). Hinh 4.6 và 4.7 dưới đây thể hiện schematic của hai khối nút nhấn và led báo.

Diagram, schematic

Description automatically generated

Nút nhấn

**Hình 4. 6** **Sơ đồ nguyên lý mạch nút nhấn**

Diagram

Description automatically generated with medium confidence

**Hình 4. 7 Sơ đồ khối LED báo**

4.4 Khối relay

### ***4.4.1 Lựa chọn relay***

Sản phẩm có chức năng chính là điều khiển bật tắt các thiết bị điện dân dụng sử dụng điện áp 220VAC. Hiện nay có 2 giải pháp phổ biến cho cách thức điều khiển điện áp 220V AC là sử dụng thyristor và relay. Trong đó đối với yêu cầu đầu ra chỉ có 2 trạng thái là bật và tắt và tần số thay đổi trạng thái thấp thì giải pháp sử dụng relay được ưu tiên sử dụng với ưu điểm giá thành rẻ, giải pháp điều khiển đơn giản. Khối relay có nhiệm vụ nhận tín hiệu điều khiển từ vi điều khiển và điều khiển trạng thái của các relay tương ứng.

**Bảng 4. 2 So sánh hai loại Relay HUIKE và SONGLE**

|  |  |  |
| --- | --- | --- |
| **Tên** | Relay HUIKE 6P | Relay SONGLE 5P |
| **Hình ảnh** | A picture containing text  Description automatically generated  **Hình 4. 8 Relay HUIKE 6P 3A** | A picture containing text  Description automatically generated  **Hình 4. 9 Relay SONGLE 5P 10A** |
| **Dòng chịu tải** | 3A | 10A |
| **Điện áp đóng** | 5V | 5V |
| **Dòng điện đóng** | 35mA | 50mA |
| **Kích thước** | 15x10x11mm | 18x15x16mm |

Sau khi khảo sát về công suất của các thiết bị điện trong nhà, nhóm nhận thấy các thiết bị thường có công suất từ 200-300W tương đương 1-1.5A. Từ thông số khảo sát, nhóm quyết định lựa chọn relay HUIKE do kích thước nhỏ gọn, tiết kiệm năng lượng đồng thời vẫn đáp ứng các yêu cầu về dòng chịu tải.

Đối với các thiết bị điện có công suất lớn, trong bản cải tiến, nhóm sẽ nâng cấp với phần đế chỉ có 1 relay để đủ không gian chứa các loại relay chịu tải công suất lớn.

### ***4.4.2 Sơ đồ kết nối relay***

Để thực hiện đóng cắt relay, ta cần đặt 1 điện áp 5V với dòng điện 35mA vào 2 đầu cuộn dây của relay. Trong khi đó dòng điện tối đa của mỗi GPIO của vi điều khiển là 12mA. Để hỗ trợ việc đóng cắt cuộn dây và đảm bảo cho hoạt động của vi điều khiển, sản phẩm sử dụng 1 transistor NPN cho mỗi relay.

Để tránh nhiễu do hiện tượng tự cảm của cuộn dây gây nên, 1 diode được mắc song song với cuộn dây để triệt tiêu điện áp ngược của hiện tượng tự cảm giữa cực B của BJT và GPIO của vi điều khiển được mắc 1 trở hạn dòng cho vi điều khiển.

Diagram, schematic

Description automatically generated

**Hình 4. 10 Sơ đồ kết nối khối relay**

Ngoài ra, để quá trình thi công và lắp đặt thiết bị được thuận lợi, đảm bảo an toàn điện và thẩm mỹ, sản phẩm được thiết kế thêm các terminal tại ngõ vào và ngõ ra, giúp công nhân thi công có thể đấu nối dây thông qua tua vít.

4.5 Khối nguồn

### ***4.5.1 Lựa chọn các giá trị của nguồn***

Đầu vào của khối nguồn là điện áp 220VAC-50Hz. Nhóm đã thống kê các yêu cầu cần thiết của khối nguồn để lựa chọn phương án khối nguồn sao cho phù hợp.

**Bảng 4. 3 Yêu cầu về các thiết bị dùng nguồn**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Thiết bị** | **Số lượng** | **Điện áp yêu cầu** | **Dòng tiêu thụ tối đa** |
| 1 | TTP223 | 6 | 3 - 5.5 V | 10mA |
| 2 | Relay HUIKE 5V-3A | 6 | 4-5V | 50mA |
| 3 | ESP8266 | 1 | 2.8-3.6V | 160mA |
| 4 | Led, BJT | 6 |  | 5mA |

Tổng dòng tiêu thụ tối đa: 600mA.

Để thiết bị có hoạt động ổn định, ta lựa chọn nguồn có công suất gấp 150% công suất tối đa.

Suy ra dòng tải tối đa của nguồn bằng 600mA \* 150% = 900mA ~ 1 A

Dựa trên bảng thống kê các yêu cầu về nguồn cấp cho các thành phần, nhóm lựa chọn loại nguồn AC-DC có đầu ra 5V-1A và IC nguồn AMS1117-3V3 để cung cấp nguồn 3V3 đồng thời bảo vệ điện áp cho ESP8266.

Ngoài ra còn yêu cầu hệ thống tụ lọc nhiễu thấp tần, nhiễu cao tần.

### ***4.5.2 Lựa chọn loại nguồn tối ưu***

Dựa theo yêu cầu đã lựa chọn ở trên, nhóm đã tìm được 3 loại nguồn đạt yêu cầu đề ra:

1. Adapter nguồn 5V 1A
2. Module nguồn 5V 1A
3. Module nguồn HiLink 5V 1A



**Hình 4. 11 Adapter nguồn 5V -1A**



**Hình 4. 12 Module nguồn 5V-1A**

A picture containing text, electronics

Description automatically generated

**Hình 4. 13 Module nguồn HiLink 5V-1A**

**Bảng 4. 4 So sánh các loại nguồn**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STT** | **Loại nguồn** | **Độ ổn định** | **Kích thước** | **Độ an toàn** | **Giá thành** | **Ghi chú** |
| 1 | Adapter nguồn 5V 1A | 60% | 65x40x25mm | Tốt | 40.000 | Kích thước quá lớn |
| 2 | Module nguồn 5V 1A | 60% | 33x22x18mm | Trung Bình | 50.000 | Độ an toàn kém, dễ chập cháy |
| 3 | HiLink 5V 1A | 80% | 38x23x18mm | Tốt | 65.000 | Độ ổn định và an toàn cao |

Dựa vào bảng 4.4 và các yêu cầu về nguồn điện bên trên, nhóm quyết định sử dụng nguồn HiLink 5V 1A để làm nguồn cung cấp điện cho các khối trong mạch.

***Diagram, schematic

Description automatically generated***

**Hình 4. 14 Sơ đồ mạch nguồn 5V**

### ***4.5.3 Sơ đồ mạch nguồn 3V3***

Ngoài nguồn VCC = 5V, sản phẩm yêu cầu nguồn cấp 3V3 – 200mA cho chip ESP8266 và khối nút nhấn. Ở đây nhóm sử dụng IC AMS1117 để chuyển từ nguồn điện áp cung cấp 5V(VCC) thành nguồn điện áp 3.3V để cùng cấp có ESP8266.

Diagram

Description automatically generated

**Hình 4. 15 IC nguồn AMS1117**

Diagram, schematic

Description automatically generated

**Hình 4. 16 Sơ đồ kết nối IC nguồn**

4.6 Khối hiển thị

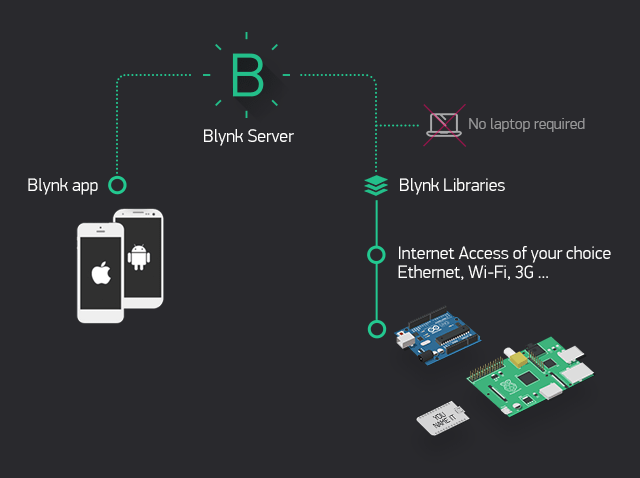
Blynk là một nền tảng có ứng dụng iOS, Android cho phép điều khiển Arduino, Raspberry Pi, ESP8266. Bạn có thể xây dựng ứng dụng điều khiển bằng cách kéo, thả các Widget. Blynk được thiết kế cho IoT, nó có thể điều khiển phần cứng từ xa, hiển thị dữ liệu cảm biến, lưu trữ dữ liệu.

Nhóm sử dụng Blynk vì các yếu tố sau:

* Dễ tiếp cận và tìm hiểu
* Đẹp và dễ tinh chỉnh
* Dễ dàng thử nghiệm nhanh chóng

Các thành phẩn của khối hiển thị bao gồm:

* Blynk App: cho phép tạo các giao diện từ Widget có sẵn
* Blynk Server: truyền tải thông tin giữa Smartphone và thiết bị. Blynk Server có thể là 1 đám mây của Blynk hoặc có thể cài đặt trên máy cá nhân. Có thể cài đặt trên Raspberry Pi.
* Blynk Libraries: thư viện cung cấp kết nối phần cứng đến server, xử lý các lệnh đến và đi.



**Hình 4. 17 Mô hình Các thành phần của Blynk**

# **CHƯƠNG 5. THIẾT KẾ HỆ THỐNG**

5.1 Thiết kế phần cứng

Sản phẩm được chia thành 2 khối lớn khi thiết kế PCB:

* Khối 1 bao gồm các thành phần có chiều cao lớn như: khối nguồn, khối relay được thiết kế để nằm trong đế âm điện.
* Khối 2 bao gồm các thành phần có chiều cao nhỏ (vi điều khiển, cổng nạp), các tụ lọc, trở treo và thành phần tương tác bề mặt (nút nhấn cảm ứng, led báo) được thiết kế nằm trong phần mặt nạ của thiết bị.

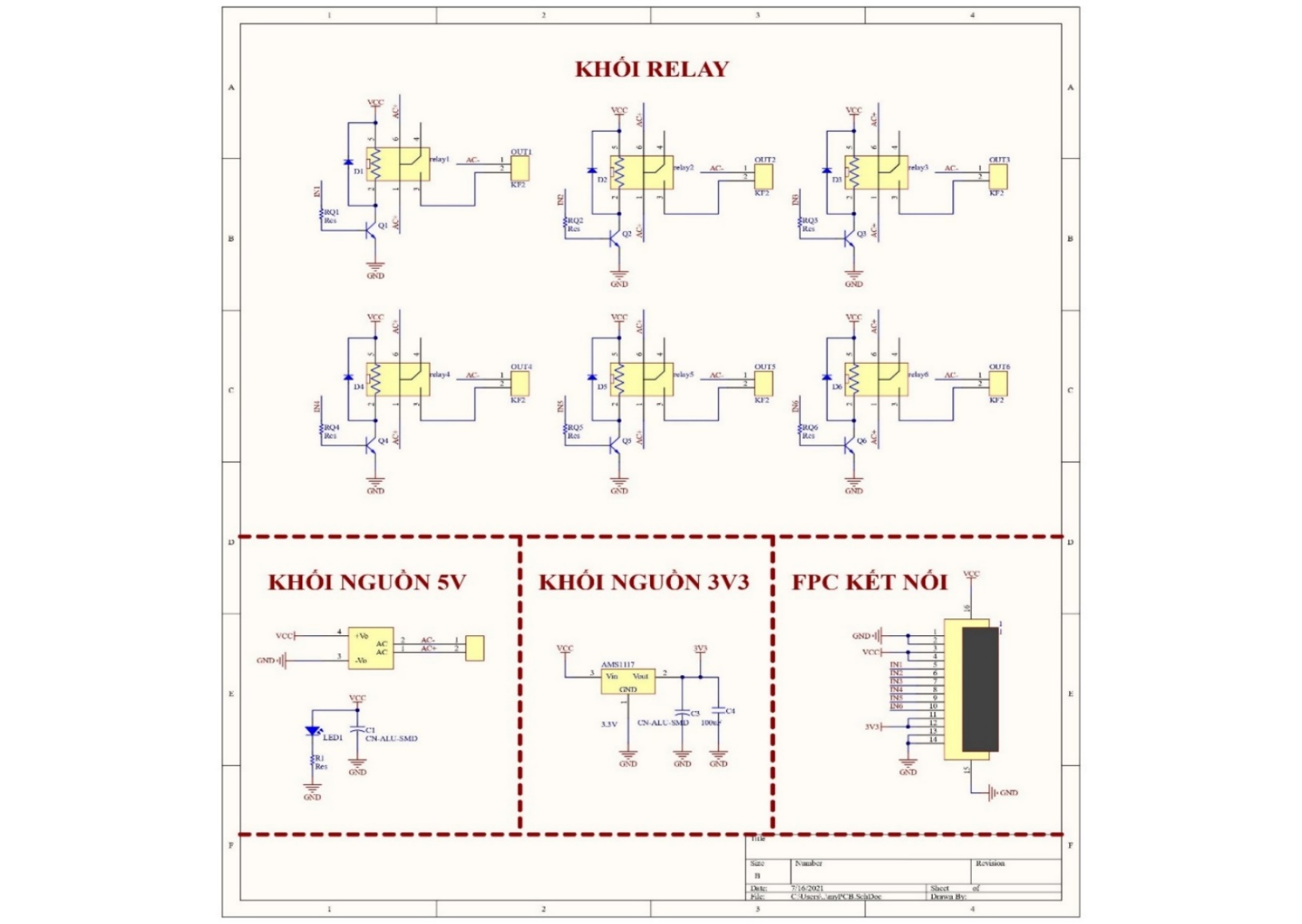
Bảng 5.1 sau đây sẽ tổng hợp các linh kiện sử dụng trong đề tài này

**Bảng 5. 1 Danh sách linh kiện**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên sản phẩm** | **Số lượng** | **Đơn giá (VNĐ)** |
| 1 | ESP8266 | 1 | 48.000 |
| 2 | LED SMD 0805 | 14 | 2.200 |
| 3 | Relay HUIKE 6P 3A | 6 | 6.500 |
| 4 | HiLink 5V 1A | 1 | 75.000 |
| 5 | Nút nhấn TTP 223 | 6 | 2.500 |
| 6 | Tụ CN ALU SMD | 2 | 1000 |
| 7 | Tụ 100uF | 1 | 1000 |
| 8 | Tụ 47uF | 1 | 1000 |
| 9 | Tụ 0.1uF | 2 | 800 |
| 10 | Tụ 1uF | 7 | 500 |
| 11 | Tụ 22pF | 6 | 1000 |
| 12 | Trở 10K | 4 | 500 |
| 13 | Trở 300R | 1 | 500 |
| 14 | Trở 100R | 4 | 500 |
| 15 | IC ASM 1117-3v3 | 1 | 1.500 |
| 16 | Đầu nối FPC | 3 | 2.000 |
| 17 | J3Y NPN Tranzitor | 6 | 300 |
| 18 | KF2-5.08 | 7 | 1.500 |
|  | Tổng tiền |  | 243000 |

### ***5.1.1 Thiết kế PCB khối 1***

Hình 5.1 biểu thị schematic của các khối Relay, khối nguồn 5V, khối nguồn 3V3, và FPC kết nối, mạch PCB và mô hình 3D của khối được thể hiện trong hình 5.2 và 5.3

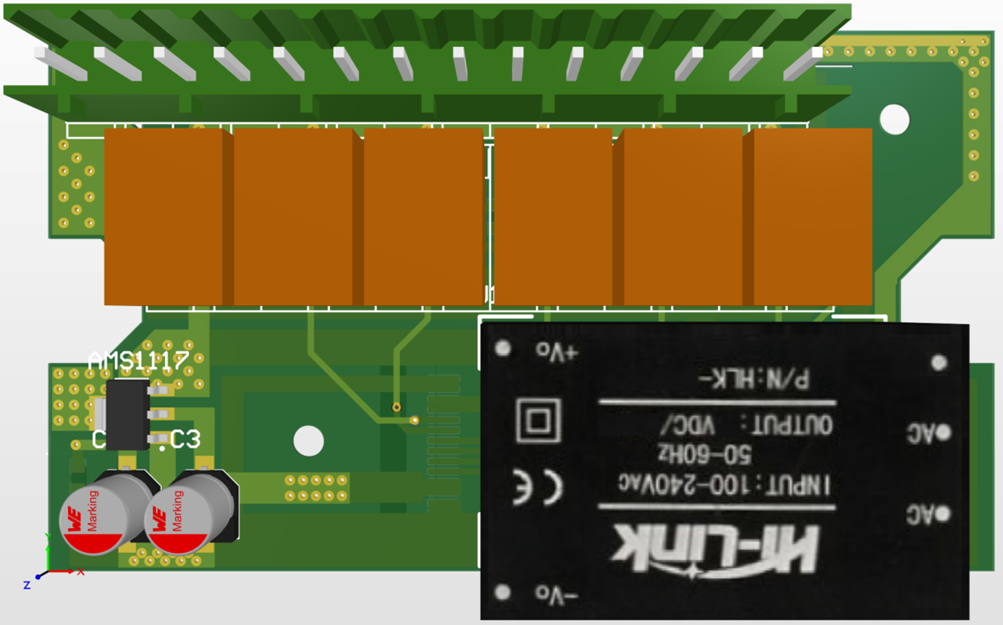


**Hình 5. 1 Schematic của khối thứ 1**

A picture containing text, electronics, circuit

Description automatically generated

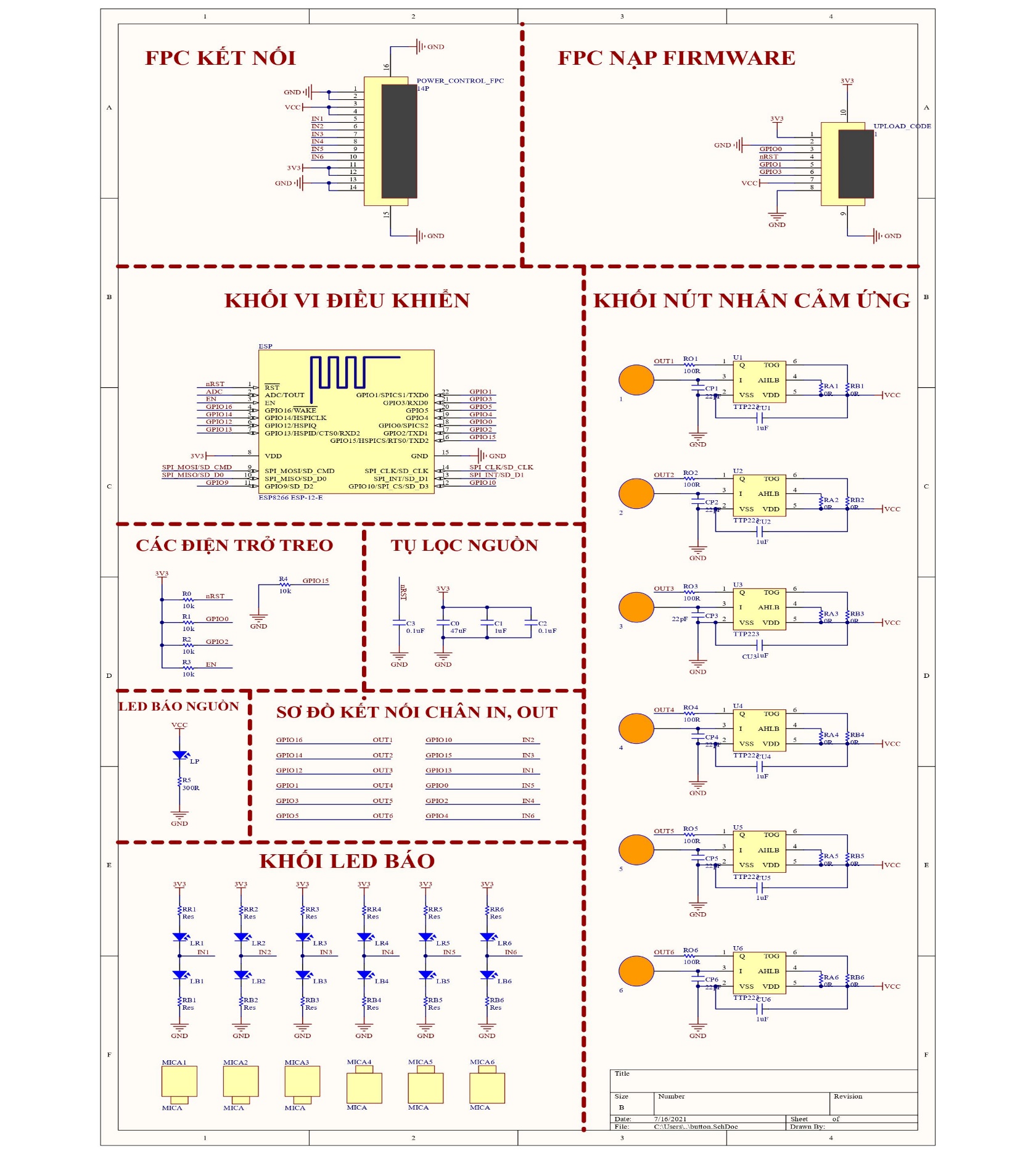
**Hình 5. 2 Mạch PCB của khối thứ 1**



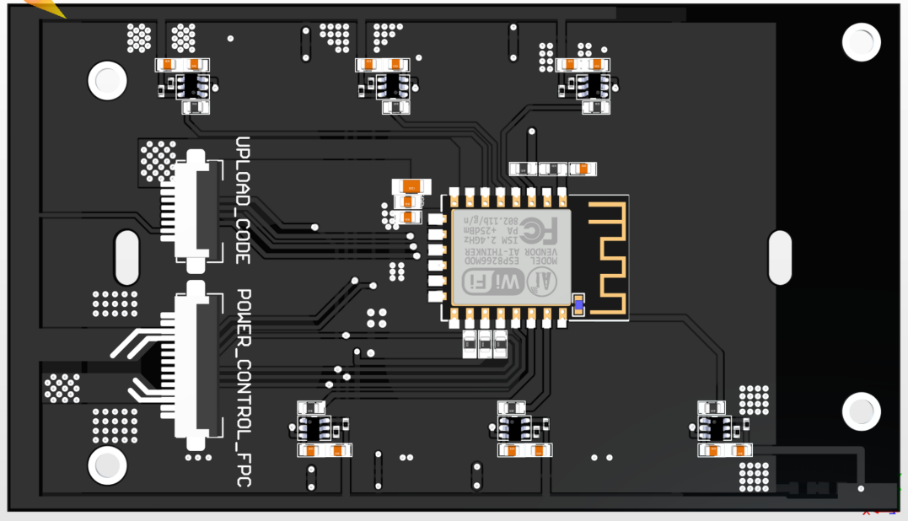
**Hình 5. 3 Hình ảnh 3D khối thứ 1**

### ***5.1.2 Thiết kế PCB khối 2***

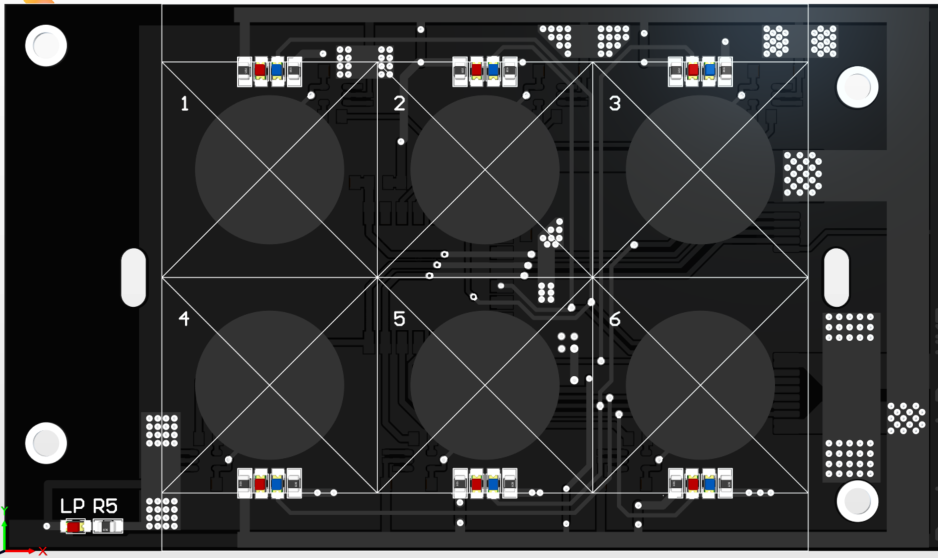
Đối với khối thứ 2 bao gồm có các khối FPC kết nối, khối FPC nạp firmware, khối vi điều khiển, khối nút nhấn cảm ứng, khối led báo, ngoài ra còn các tụ lọc nguồn, các điện trở treo. Từ đó nhóm đã xây dựng schemactic của khốí thứ 2 như mô tả trong hình 5.4. Hình 5.5 và 5.6 thể hiện mạch PCB của khối thứ 2.



**Hình 5. 4. Schematic của khối mạch thứ 2**



**Hình 5. 5 Mạch PCB của khối thứ 2 (top viewer)**



**Hình 5. 6 Mạch PCB của khối thứ 2 (bottom viewer)**

5.2 Thiết kế phần mềm

### ***5.2.1 Công nghệ sử dụng***

Arduino là một nền tảng mã nguồn mở bao gồm cả phần mềm và phần cứng được thiết kế giúp làm việc với các board mạch điện tử trở nên dễ dàng hơn bao giờ hết.

Arduino che dấu đi sự phức tạp của điện tử bằng cách đơn giản hóa các khái niệm, che đi sự phức tạp của phần mềm bằng các thủ tục ngắn gọn thay phải làm việc xuống mức cấp thấp của hardware, nhưng với Arduino thì chỉ cần gọi 1 hàm.

Bởi vì tính phổ biến và dễ dùng cùng với các vô vàn thư viện được tích hợp sẵn để làm việc với rất nhiều module phần cứng khác nhau, bạn chỉ cần quan tâm đến tính năng sản phẩm mà bỏ qua các khái niệm phức tạp (protocol, datasheet …​) từ đó dễ dàng tiếp cận và làm ra các sản phẩm tuyệt vời mà không cần phải biết nhiều về điện tử.

Arduino bao gồm một phần mềm lập trình là Arduino IDE và một tập hợp rất nhiều các board mạch Arduino có thể lập trình được bằng phần mềm này với các biến thể khác nhau. Ban đầu phần lớn các board này đều dựa trên các chip họ AVR của Atmel sản xuất, nhưng sau này có rất nhiều nhà sản xuất sử dụng các chip khác nhau như ARM, PIC, STM32 gần đây nhất là ESP8266, ESP32… với năng lực phần cứng và phần mềm đi kèm mạnh mẽ hơn nhiều cũng phát hành các thư viện giúp làm việc được với Arduino như các board Arduino chính chủ.

### ***5.2.2 Thiết kế chương trình***

***5.2.2.1 Khai báo các thư viện, hằng và các biến có trong chương trình***

//---------Khai bao thu vien dung trong chuong trinh-------//

#include <ESP8266WiFi.h>

#include <DNSServer.h>

#include <WiFiClient.h>

#include <EEPROM.h>

#include <ESP8266WebServer.h>

#include <BlynkSimpleEsp8266.h>

//--------Khai bao cac chan GPIO cua ESP8266--------//

#define OUT1 13

#define OUT2 10

#define OUT3 15

#define OUT4 2

#define OUT5 0

#define OUT6 4

#define IN1 16

#define IN2 14

#define IN3 12

#define IN4 1

#define IN5 3

#define IN6 5

//-------Khai bao cac bien trong chương trinh-------//

boolean restoreConfig();

boolean checkConnection();

void startWebServer();

void setupMode();

String makePage(String title, String contents);

String urlDecode(String input);

void readResetButton1();

const IPAddress apIP(192, 168, 1, 1);

const char\* apSSID = "ESP8266\_SETUP";

boolean settingMode;

String ssidList; // Bien liet ke cac mang wifi xung quanh

bool state1=0, state2 =0, state3=0, state4=0, state5=0, state6=0;

bool norState1 = 1, norState2 = 1, norState3 = 1, norState4 = 1, norState5 = 1, norState6 = 1;

int SETTING\_HOLD\_TIME = 5000; //thoi gian bam nut de chuyen trang thai

long int settingTimeout;

DNSServer dnsServer;

ESP8266WebServer webServer(80); //khoi tao server cho esp8266 voi port 80 cho http

***5.2.2.2 Đồng bộ dữ liệu với server Blynk***

//------------Ket noi voi Blynk---------//

BLYNK\_WRITE(V1)

{

changeState1();

}

BLYNK\_WRITE(V2)

{

changeState2();

}

BLYNK\_WRITE(V3)

{

changeState3();

}

BLYNK\_WRITE(V4)

{

changeState4();

}

BLYNK\_WRITE(V5)

{

changeState5();

}

BLYNK\_WRITE(V6)

{

changeState6();

}

***5.2.2.3 Thao tác với bộ nhớ EEPROM***

//----------Doc duu lieu trong EEPROM va ket noi voi mang wifi--------------

boolean restoreConfig() {

String ssid = "";

String pass = "";

String auth = "";

if (EEPROM.read(0) != 0) { //neu du lieu doc ra tu EEPROM khac 0 thi doc du lieu

for (int i = 0; i < 32; ++i) { //32 o nho dau tieu la chua ten mang wifi SSID

ssid += char(EEPROM.read(i));

}

for (int i = 32; i < 96; ++i) { //o nho tu 32 den 96 la chua PASSWORD

pass += char(EEPROM.read(i));

}

for (int i = 96; i < 128; ++i) { //o nho tu 32 den 96 la chua TOKEN

auth += char(EEPROM.read(i));

}

//WiFi.begin(ssid.c\_str(), pass.c\_str()); //ket noi voi mang WIFI duoc luu trong EEPROM

Blynk.begin(auth.c\_str(),ssid.c\_str(),pass.c\_str());

return true;

}

else {

return false;

}

}

//-----------Kiem tra lai ket noi voi WIFI-----------------------

boolean checkConnection() {

int count = 0;

while ( count < 30 ) {

if (WiFi.status() == WL\_CONNECTED) { //neu ket noi thanh cong thi in ra connected!

return (true);

}

delay(500);

count++;

}

return false;

}

***5.2.2.4 Thao tác nút bấm***

//----------Ham doc trang thai tat ca cac nut bam----------//

void readAllButton(){

readButton1();

readButton2();

readButton3();

readButton4();

readButton5();

readButton6();

}

//----------Ham doc trang thai tung nut bam-------------//

//---------Kich hoat o suon am khi bam nut----------------//

void readButton1(){

if((digitalRead(IN1)-norState1)<0){

norState1 = 0;

changeState1();

Blynk.virtualWrite(V1,state1);

}

else if((digitalRead(IN1)-norState1)>0){

norState1 = 1;

}

}

void readButton2(){

if((digitalRead(IN2)-norState2)<0){

norState2 = 0;

changeState2();

Blynk.virtualWrite(V2,state2);

}

else if((digitalRead(IN2)-norState2)>0){

norState2 = 1;

}

}

void readButton3(){

if((digitalRead(IN3)-norState3)<0){

norState3 = 0;

changeState3();

Blynk.virtualWrite(V3,state3);

}

else if((digitalRead(IN3)-norState3)>0){

norState3 = 1;

}

}

void readButton4(){

if((digitalRead(IN4)-norState4)<0){

norState4 = 0;

changeState4();

Blynk.virtualWrite(V4,state4);

}

else if((digitalRead(IN4)-norState4)>0){

norState4 = 1;

}

}

void readButton5(){

if((digitalRead(IN5)-norState5)<0){

norState5 = 0;

changeState5();

Blynk.virtualWrite(V5,state5);

}

else if((digitalRead(IN5)-norState5)>0){

norState5 = 1;

}

}

void readButton6(){

if((digitalRead(IN6)-norState6)<0){

norState6 = 0;

changeState6();

Blynk.virtualWrite(V6,state6);

}

else if((digitalRead(IN6)-norState6)>0){

norState6 = 1;

}

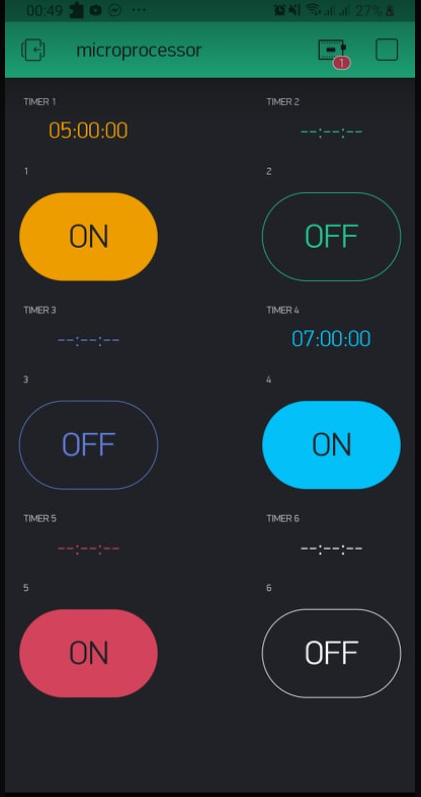
}

***5.2.2.5 Thiết kế giao diện Blynk***

Để tạo giao diện trên App Blynk, đầu tiên ta cần tạo một tài khoản và đăng nhập vào App Blynk. Sau đó nhấn Create New Project để tạo project mới, đặt tên cho project ở phần Project Name là Smarthome Control. Chọn thiết bị ở phần Chose Device là ESP8266. Nhấn Create để tạo project.

Tiếp theo ta thêm các widget ở Widget Box:

* Thêm 6 Button cho 6 relay, đều chọn Mode là SWITCH, Output là VitualPin V1, V2, V3, V4, V5, V6
* Ứng với mỗi Button, thêm một timer có chức năng hẹn giờ bật tắt thiêt bị.



**Hình 5.7 Giao diện Blynk**

# **CHƯƠNG 6. TEST VÀ KIỂM THỬ**

|  |  |
| --- | --- |
| Test Case Identifier: TC-1  Test chế độ Access Point của ESP8266 | |
| Thủ tục kiểm thử | Kết quả |
| Step 1: Chuyển ESP8266 sang chế độ Access Point | Thành công |
| Step 2: Kiểm tra danh sách wifi trên laptop | Danh sách wifi hiện có của laptop có wifi được phát ra bởi ESP8266 |
| Step 3: Kết nối máy tính với wifi trên | Máy tính kết nối với wifi thành công |
| Step 4: Laptop truy cập vào địa chỉ của ESP8266  (192.168.1.1) | Laptop truy cập thành công vào trang web |
| Step 5: Nhập thông số wifi vào lưu vào EEPROM | Người dùng nhập thành công SSID, PASS, AUTH TOKEN vào các ô tương ứng. |

|  |  |
| --- | --- |
| Test Case Identifier: TC-2  Test chức năng xử lý nút bấm vật lý | |
| Thủ tục kiểm thử | Kết quả |
| Step 1: Ấn các nút bấm để thay đổi trạng thái relay | Relay thanh đổi trạng thái tương ứng với khi nút được ấn |

|  |  |
| --- | --- |
| Test Case Identifier: TC-3  Test chức năng điều khiển thiết bị thông qua app Blynk | |
| Thủ tục kiểm thử | Kết quả |
| Step 1: Vào app blynk trên điện thoại | Thành công |
| Step 2: Kiểm tra kết nối giữa esp8266 và server blynk | Kết nối thành công |
| Step 3: Ấn vào các nút trên app và kiểm tra relay | Giá trị relay thay đổi tương ứng với hành động ấn nút |

|  |  |
| --- | --- |
| Test Case Identifier: TC-4  Test chức năng hẹn giờ trên app Blynk | |
| Thủ tục kiểm thử | Kết quả |
| Step 1: Vào app blynk trên điện thoại | Thành công |
| Step 2: Kiểm tra kết nối giữa esp8266 và server blynk | Kết nối thành công |
| Step 3: Hẹn giờ cho button 1  Start time: 17:00:00  Stop time: 17:01:00 | Relay 1 chuyển sang trạng thái ON lúc 17:00:00 và chuyển sang trạng thái OFF lúc 17:01:00 |
| Step 4: Thực hiện step 3 cho các button còn lại | Thành công |

# **CHƯƠNG 7. KẾT LUẬN**

Với mục tiêu ban đầu là thiết kế hệ thống điều khiển công tắc từ xa thông qua mạng internet và các thiết bị thông minh như smartphone, bện cạnh đó là cải thiện chức năng công tắc truyền thống thành nút nhấn cảm ứng khi đó người dùng có thể điều khiển được các công tắc từ xa và sử dụng được nút nhấn cảm ứng thay cho công tắc truyền thống. Từ mục tiêu đó, các thành viên trong nhóm đã cùng nhau tìm hiểu thông tin, cách thức và các kiến thức cần thiết để có thể thiết kế được sản phẩm như yêu cầu. Bên cạnh đó sản phẩm có tiềm năng lớn để mở rộng và nâng cấp sau này.

**Hướng phát triển của đề tài**

Trong thời gian tới, nhóm em sẽ tiếp tục nghiên cứu phát triển đề tài theo hướng sau đây:

* Tăng tính chính xác và ổn định hơn nữa.
* Tích hợp điều khiển khả năng điều khiển thông qua giọng nói.
* Tích hợp nhiều cảm biến hơn nữa phù hợp với nhiều đối tượng người dùng để phục vụ cho cuộc sống và phục vụ trong công nghiệp.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

1. Nguyễn Thị Kim Anh, *Điều khiển thiết bị qua module ESP8266.* Đại học Huế, Đại học Sư Phạm khoa Vật lý, Huế, 2018
2. Mesquita, J., Guimaraes, D., Pereira, C., Santos, F., & Almeida, L. (2018). *Assessing the ESP8266 WiFi module for the Internet of Things.* IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), 2018.
3. Shakthidhar, S., Srikrishnan, P., Santhosh, S., & Sandhya, M. K. (2019). *Arduino and NodeMcu based Ingenious Household Objects Monitoring and Control Environment.* Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), 2019.

[4] *ESP8266EX Datasheet (PDF) - ESPRESSIF SYSTEMS (SHANGHAI) CO., LTD.,* <https://pdf1.alldatasheet.com/datasheet-pdf/view/1148030/ESPRESSIF/ESP8266EX.html>

[5] *SIGNAL RELAY. HK4100F Datasheet,* <https://datasheetspdf.com/pdf/1092439/HUIKE/HK4100F/1>

[6] *TTP223 Datasheet (PDF) - TT Electronics,*

<https://www.alldatasheet.com/datasheet-pdf/pdf/1179637/TTELEC/TTP223.html>

[7] *Top 6 công tắc thông minh tốt nhất 2021,*

<https://vietreview.vn/danh-gia/cong-tac-thong-minh/?fbclid=IwAR3Ocf6-mhA6XKADHh6DHrtbDQiltywE2vBCHPxjnwR6ZChliGxuIOrvq6U>

[8] *SRD05VDCSLC Datasheet,*

<https://www.alldatasheet.com/datasheetpdf/pdf/1131944/SONGLERELAY/SRD05VDCSLC.html>

[9] LED 0805 SMD Datasheet

https://www.alldatasheet.com/datasheet-pdf/pdf/123001/FORMOSA/0805.html

# **PHỤ LỤC**

Mã nguồn của chương trình

|  |
| --- |
| //---------Khai bao thu vien dung trong chuong trinh-------//  #include <ESP8266WiFi.h>  #include <DNSServer.h>  #include <WiFiClient.h>  #include <EEPROM.h>  #include <ESP8266WebServer.h>  #include <BlynkSimpleEsp8266.h>  //--------Khai bao cac chan GPIO cua ESP8266--------//  #define OUT1 13  #define OUT2 10  #define OUT3 15  #define OUT4 2  #define OUT5 0  #define OUT6 4  #define IN1 16  #define IN2 14  #define IN3 12  #define IN4 1  #define IN5 3  #define IN6 5  //-------Khai bao cac bien trong chương trinh-------//  boolean restoreConfig();  boolean checkConnection();  void startWebServer();  void setupMode();  String makePage(String title, String contents);  String urlDecode(String input);  void readResetButton1();  const IPAddress apIP(192, 168, 1, 1);  const char\* apSSID = "ESP8266\_SETUP";  boolean settingMode;  String ssidList; // Bien liet ke cac mang wifi xung quanh  bool state1=0, state2 =0, state3=0, state4=0, state5=0, state6=0; //bien trang thai relay  bool norState1 = 1, norState2 = 1, norState3 = 1, norState4 = 1, norState5 = 1, norState6 = 1; //Bien trang thai nut bam  int SETTING\_HOLD\_TIME = 5000; //thoi gian bam nut de chuyen trang thai  long int settingTimeout;  DNSServer dnsServer;  ESP8266WebServer webServer(80); //khoi tao server cho esp8266 voi port 80 danh cho http  //----------Ham setup----------//  void setup() {  //----------Dat che do cho cac cong GPIO-------------//  pinMode(OUT1, OUTPUT);  pinMode(OUT2, OUTPUT);  pinMode(OUT3, OUTPUT);  pinMode(OUT4, OUTPUT);  pinMode(OUT5, OUTPUT);  pinMode(OUT6, OUTPUT);    pinMode(IN1, INPUT);  pinMode(IN2, INPUT);  pinMode(IN3, INPUT);  pinMode(IN4, INPUT);  pinMode(IN5, INPUT);  pinMode(IN6, INPUT);    EEPROM.begin(512); //  delay(10);  if (restoreConfig()) { //neu co du lieu wifi trong EEPROM thi ket noi va kiem tra ket noi voi wifi do  if (checkConnection()) {  settingMode = false; //set che do cai dat la false de bo qua buoc settingMode  startWebServer(); //khoi tao 1 webserver de truy cap  return;  }  }  settingMode = true; // set che do cai dat lai thanh true  setupMode(); //chuyen sang che do station de vao cai dat cau hinh wifi  }  void loop() {  if (settingMode) { //neu settingMode == 1 thi  dnsServer.processNextRequest();  }  readResetButton1();  readAllButton();  Blynk.run();  webServer.handleClient();  }  //----------Ham doc trang thai tat ca cac nut bam----------//  void readAllButton(){  readButton1();  readButton2();  readButton3();  readButton4();  readButton5();  readButton6();  }  //----------Ham doc trang thai tung nut bam-------------//  //---------Kich hoat o suon am khi bam nut----------------//  void readButton1(){  if((digitalRead(IN1)-norState1)<0){  norState1 = 0;  changeState1();  Blynk.virtualWrite(V1,state1);  }  else if((digitalRead(IN1)-norState1)>0){  norState1 = 1;  }  }  void readButton2(){  if((digitalRead(IN2)-norState2)<0){  norState2 = 0;  changeState2();  Blynk.virtualWrite(V2,state2);  }  else if((digitalRead(IN2)-norState2)>0){  norState2 = 1;  }  }  void readButton3(){  if((digitalRead(IN3)-norState3)<0){  norState3 = 0;  changeState3();  Blynk.virtualWrite(V3,state3);  }  else if((digitalRead(IN3)-norState3)>0){  norState3 = 1;  }  }  void readButton4(){  if((digitalRead(IN4)-norState4)<0){  norState4 = 0;  changeState4();  Blynk.virtualWrite(V4,state4);  }  else if((digitalRead(IN4)-norState4)>0){  norState4 = 1;  }  }  void readButton5(){  if((digitalRead(IN5)-norState5)<0){  norState5 = 0;  changeState5();  Blynk.virtualWrite(V5,state5);  }  else if((digitalRead(IN5)-norState5)>0){  norState5 = 1;  }  }  void readButton6(){  if((digitalRead(IN6)-norState6)<0){  norState6 = 0;  changeState6();  Blynk.virtualWrite(V6,state6);  }  else if((digitalRead(IN6)-norState6)>0){  norState6 = 1;  }  }  //--------------Ham chuyen trang thai cac relay-------------//  void changeState1(){  state1 = !state1;  digitalWrite(OUT1, state1);  }  void changeState2(){  state2 = !state2;  digitalWrite(OUT2, state2);  }  void changeState3(){  state3 = !state3;  digitalWrite(OUT3, state3);  }  void changeState4(){  state4 = !state4;  digitalWrite(OUT4, state4);  }  void changeState5(){  state5 = !state5;  digitalWrite(OUT5, state5);  }  void changeState6(){  state6 = !state6;  digitalWrite(OUT6, state6);  }  //------------Ket noi voi Blynk---------//  BLYNK\_WRITE(V1)  {  changeState1();  }  BLYNK\_WRITE(V2)  {  changeState2();  }  BLYNK\_WRITE(V3)  {  changeState3();  }  BLYNK\_WRITE(V4)  {  changeState4();  }  BLYNK\_WRITE(V5)  {  changeState5();  }  BLYNK\_WRITE(V6)  {  changeState6();  }  //----------Doc duu lieu trong EEPROM va ket noi voi mang wifi--------------  boolean restoreConfig() {  String ssid = "";  String pass = "";  String auth = "";  if (EEPROM.read(0) != 0) { //neu duu lieu doc ra tu EEPROM khac 0 thi doc du lieu  for (int i = 0; i < 32; ++i) { //32 o nho dau tieu la chua ten mang wifi SSID  ssid += char(EEPROM.read(i));  }  for (int i = 32; i < 96; ++i) { //o nho tu 32 den 96 la chua PASSWORD  pass += char(EEPROM.read(i));  }  for (int i = 96; i < 128; ++i) { //o nho tu 32 den 96 la chua TOKEN  auth += char(EEPROM.read(i));  }  //WiFi.begin(ssid.c\_str(), pass.c\_str()); //ket noi voi mang WIFI duoc luu trong EEPROM  Blynk.begin(auth.c\_str(),ssid.c\_str(),pass.c\_str());  return true;  }  else {  return false;  }  }  //-----------Kiem tra lai ket noi voi WIFI-----------------------  boolean checkConnection() {  int count = 0;  while ( count < 30 ) {  if (WiFi.status() == WL\_CONNECTED) { //neu ket noi thanh cong thi in ra connected!  return (true);  }  delay(500);  count++;  }  return false;  }  //-----------------Thiet lap mot WEBSERVER-------------------------------  void startWebServer() {  if (settingMode) { //neu o chua settingMode la true thi thiet lap 1 webserver  webServer.on("/settings", []() {  String s = "<h1>Wi-Fi Settings</h1><p>Please enter your password by selecting the SSID.</p>";  s += "<form method=\"get\" action=\"setap\"><label>SSID: </label><select name=\"ssid\">";  s += ssidList;  s += "</select><br>Password: <input name=\"pass\" length=64 type=\"password\">";  s += "</select><br>Token: <input name=\"auth\" length=64 type=\"text\"><input type=\"submit\"></form>";  webServer.send(200, "text/html", makePage("Wi-Fi Settings", s));  });  webServer.on("/setap", []() {  for (int i = 0; i < 96; ++i) {  EEPROM.write(i, 0); //xoa bo nho EEPROM  }  String ssid = urlDecode(webServer.arg("ssid"));  String pass = urlDecode(webServer.arg("pass"));  String auth = urlDecode(webServer.arg("auth"));  for (int i = 0; i < ssid.length(); ++i) {  EEPROM.write(i, ssid[i]);  }  for (int i = 0; i < pass.length(); ++i) {  EEPROM.write(32 + i, pass[i]);  }  for (int i = 0; i < auth.length(); ++i) {  EEPROM.write(96 + i, auth[i]);  }  String endtext = "";  for (int i = 0; i < auth.length(); ++i) {  EEPROM.write(160 + i, endtext[i]);  }    EEPROM.commit();  String s = "<h1>Setup complete.</h1><p>device will be connected to \"";  s += ssid;  s += "\" after the restart.";  webServer.send(200, "text/html", makePage("Wi-Fi Settings", s));  ESP.restart();  });  webServer.onNotFound([]() {  String s = "<h1>AP mode</h1><p><a href=\"/settings\">Wi-Fi Settings</a></p>";  webServer.send(200, "text/html", makePage("AP mode", s));  });  }  else {  webServer.on("/", []() {  String s = "<h1>STA mode</h1><p><a href=\"/reset\">Reset Wi-Fi Settings</a></p>";  webServer.send(200, "text/html", makePage("STA mode", s));  });  webServer.on("/reset", []() { // kiem tra duong dan"/reset" thi xoa EEPROM  for (int i = 0; i < 96; ++i) {  EEPROM.write(i, 0);  }  EEPROM.commit();  String s = "<h1>Wi-Fi settings was reset.</h1><p>Please reset device.</p>";  webServer.send(200, "text/html", makePage("Reset Wi-Fi Settings", s));  });  }  webServer.begin();  }  //-----------------Che do cai dat wifi cho esp8266----------------------  void setupMode() {  WiFi.mode(WIFI\_STA); //che do hoat dong la May Tram Station  WiFi.disconnect(); //ngat ket noi wifi  delay(100);  int n = WiFi.scanNetworks(); //quet cac mang wifi xung quanh xem co bao nhieu mang  delay(100);  for (int i = 0; i < n; ++i) { //dua danh sach wifi vao list  ssidList += "<option value=\"";  ssidList += WiFi.SSID(i);  ssidList += "\">";  ssidList += WiFi.SSID(i);  ssidList += "</option>";  }  delay(100);  WiFi.mode(WIFI\_AP); // chuyen sang che dong Access point  WiFi.softAPConfig(apIP, apIP, IPAddress(255, 255, 255, 0));  WiFi.softAP(apSSID,"12345678"); //thiet lap 1 open netword WiFi.softAP(ssid, password)  dnsServer.start(53, "\*", apIP);  startWebServer();  }  String makePage(String title, String contents) {  String s = "<!DOCTYPE html><html><head>";  s += "<meta name=\"viewport\" content=\"width=device-width,user-scalable=0\">";  s += "<title>";  s += title;  s += "</title></head><body>";  s += contents;  s += "</body></html>";  return s;  }  String urlDecode(String input) {  String s = input;  s.replace("%20", " ");  s.replace("+", " ");  s.replace("%21", "!");  s.replace("%22", "\"");  s.replace("%23", "#");  s.replace("%24", "$");  s.replace("%25", "%");  s.replace("%26", "&");  s.replace("%27", "\'");  s.replace("%28", "(");  s.replace("%29", ")");  s.replace("%30", "\*");  s.replace("%31", "+");  s.replace("%2C", ",");  s.replace("%2E", ".");  s.replace("%2F", "/");  s.replace("%2C", ",");  s.replace("%3A", ":");  s.replace("%3A", ";");  s.replace("%3C", "<");  s.replace("%3D", "=");  s.replace("%3E", ">");  s.replace("%3F", "?");  s.replace("%40", "@");  s.replace("%5B", "[");  s.replace("%5C", "\\");  s.replace("%5D", "]");  s.replace("%5E", "^");  s.replace("%5F", "-");  s.replace("%60", "`");  return s;  }  //------------Ham chuyen 2 che do Station mode va Access Point khi an giu nut 1 trong 5s----------//  void readResetButton1(){  if(digitalRead(IN1) == 0){  if((settingTimeout + SETTING\_HOLD\_TIME) <= millis()){  settingMode = true; // set che do cai dat lai thanh true  setupMode();  }  }  if(digitalRead(IN1)){  settingTimeout = millis();  }  } |