

BỘ GIÁO DỤC & ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH  
KHOA ĐIỆN – ĐIỆN TỬ  
BỘ MÔN ĐIỆN TỬ CÔNG NGHIỆP – Y SINH



# ĐỒ ÁN TỐT NGHIỆP

NGÀNH CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ TRUYỀN THÔNG

Đề Tài:

THIẾT KẾ VÀ THI CÔNG MÔ HÌNH ỨNG DỤNG

IOT VÀO VIỆC ĐIỀU KHIỂN GIÁM SÁT

CÁC THIẾT BỊ ĐIỆN TRONG NHÀ

GVHD: TS. Nguyễn Văn Thái

SVTH: Nguyễn Huỳnh Tâm 16341022

Hình Đông Tịnh 16341024

Tp. Hồ Chí Minh – 01/2018

BỘ GIÁO DỤC & ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH  
KHOA ĐIỆN – ĐIỆN TỬ  
BỘ MÔN ĐIỆN TỬ CÔNG NGHIỆP – Y SINH

# ĐỒ ÁN TỐT NGHIỆP

NGÀNH CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ TRUYỀN THÔNG

Đề Tài:

THIẾT KẾ VÀ THI CÔNG MÔ HÌNH ỨNG DỤNG  
IOT VÀO VIỆC ĐIỀU KHIỂN GIÁM SÁT  
CÁC THIẾT BỊ ĐIỆN TRONG NHÀ

GVHD: TS. Nguyễn Văn Thái

SVTH: Nguyễn Huỳnh Tâm 16341022

Hình Đông Tịnh 16341024

Tp. Hồ Chí Minh – 01/2018

Tp. HCM, ngày 12 tháng 1 năm 2018

## NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên:	Nguyễn Huỳnh Tâm	MSSV:16341022
	Hình Đông Tịnh	MSSV:16341024
Chuyên ngành:	Công nghệ kỹ thuật điện tử truyền thông	Mã ngành: 41
Hệ đào tạo:	Đại học chính quy chuyển tiếp	Mã hệ: 3
Khóa:	2016	Lớp: 163410

I. TÊN ĐỀ TÀI: **THIẾT KẾ VÀ THI CÔNG MÔ HÌNH ỦNG DỤNG IOT VÀO VIỆC ĐIỀU KHIỂN GIÁM SÁT CÁC THIẾT BỊ ĐIỆN TRONG NHÀ.**

### II. NHIỆM VỤ

#### 1. Các số liệu ban đầu:

Cảm ứng điện dung từ tay người dùng .....  
.....  
.....  
.....  
.....

#### 2. Nội dung thực hiện:

- Điều khiển thiết bị thông qua Internet và theo dõi trạng thái thiết bị .....  
- Lưu trữ dữ liệu vào cơ sở dữ liệu .....  
- Xây dựng web server .....  
- Thiết kế thi công mô hình .....  
.....  
.....

III. NGÀY GIAO NHIỆM VỤ: 25/9/2017

IV. NGÀY HOÀN THÀNH NHIỆM VỤ: 10/1/2018

V. HỌ VÀ TÊN CÁN BỘ HƯỚNG DẪN: TS. NGUYỄN VĂN THÁI

CÁN BỘ HƯỚNG DẪN

BM. ĐIỆN TỬ CÔNG NGHIỆP – Y SINH

Tp. HCM, ngày 12 tháng 1 năm 2018

## LỊCH TRÌNH THỰC HIỆN ĐO ÁN TỐT NGHIỆP

Họ tên sinh viên 1: Nguyễn Huỳnh Tâm.....  
Lớp: 163410A..... MSSV: 16341022.....  
Họ tên sinh viên 2: Hình Đông Tịnh .....,  
Lớp: 163410A..... MSSV: 16341024.....  
Tên đề tài: Thiết kế và thi công mô hình ứng dụng IoT vào việc điều khiển giám sát  
các thiết bị điện trong nhà .....,  
.....

<i>Tuần/ngày</i>	<i>Nội dung</i>	<i>Xác nhận GVHD</i>
Tuần 4 (tháng 9)	Tìm hiểu sản phẩm thị trường về công tắc thông minh.	
Tuần 1 (tháng 10)	Tìm hiểu hoạt động của mạch ESP, nguồn, cảm ứng điện dung, mạch công suất.	
Tuần 2+3 (tháng 10)	Thiết kế mạch nguồn, mạch công suất.	
Tuần 4 (tháng 10)	Thiết kế mạch cảm ứng điện dung, mạch ESP	
Tuần 1 (tháng 11)	Kiểm tra mạch, chỉnh sửa thiết kế	
Tuần 2+3 (tháng 11)	Tìm hiểu về web server, cơ sở dữ liệu,...	
Tuần 4 tháng 11	Thiết kế web server, cơ sở dữ liệu,... + chỉnh sửa, thiết kế board mạch	
Tuần 1 (tháng 12)	Thiết lập kết nối ESP và server + chỉnh sửa thiết kế board mạch	
Tuần 2 (tháng 12)	Chỉnh sửa phần cứng và phần mềm	
Tuần 3 (tháng 12)	Tìm hiểu html, css, javascript,...	
Tuần 4 (tháng 12)	Chỉnh sửa giao diện web	
Tuần 1 (tháng 1/2018) + hiện tại	Kiểm tra hệ thống, chỉnh sửa + viết báo cáo.	

GV HƯỚNG DẪN  
(Ký và ghi rõ họ và tên)

## LỜI CAM ĐOAN

- **Tên đề tài:** THIẾT KẾ VÀ THI CÔNG MÔ HÌNH ỦNG DỤNG IOT VÀO VIỆC ĐIỀU KHIỂN GIÁM SÁT CÁC THIẾT BỊ ĐIỆN TRONG NHÀ
- **GVHD:** TS. NGUYỄN VĂN THÁI
- **Họ tên sinh viên 1:** NGUYỄN HUỲNH TÂM  
MSSV: 16341022 Lớp: 163410A  
Số điện thoại liên lạc: 0972797876  
Email: 16341022@student.hcmute.edu.vn
- **Họ tên sinh viên 2:** HÌNH ĐÔNG TỊNH  
MSSV: 16341024 Lớp: 163410A  
Số điện thoại liên lạc: 01678922669  
Email: 16341024@student.hcmute.edu.vn

*“Tôi xin cam đoan khoá luận tốt nghiệp (ĐATN) này là công trình do chính tôi nghiên cứu và thực hiện. Tôi không sao chép từ bất kỳ một bài viết nào đã được công bố mà không trích dẫn nguồn gốc. Nếu có bất kỳ một sự vi phạm nào, tôi xin chịu hoàn toàn trách nhiệm”.*

Người thực hiện đề tài

Nguyễn Huỳnh Tâm

Hình Đông Tịnh

## LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn sâu sắc đến Thầy Nguyễn Văn Thái đã trực tiếp hướng dẫn và tận tình giúp đỡ, tạo điều kiện để chúng em hoàn thành tốt đề tài.

Chúng em xin chân thành cảm ơn Thầy Nguyễn Hữu Trung, giảng viên Khoa Công nghệ thông tin, đã góp ý và chia sẻ nhiều kinh nghiệm quý báu để chúng em có thể thực hiện tốt đề tài.

Chúng em xin gửi lời chân thành cảm ơn các thầy cô trong Khoa Điện-Điện Tử đã tạo những điều kiện tốt nhất cho chúng em trong quá trình thực hiện đồ án.

Chúng em cũng gửi lời đồng cảm ơn đến các bạn lớp 16341 đã chia sẻ trao đổi kiến thức cũng như những kinh nghiệm quý báu giúp đỡ chúng em hoàn thành Đồ án tốt nghiệp này.

Xin chân thành cảm ơn!

Người thực hiện đề tài

Nguyễn Huỳnh Tâm

Hình Đông Tịnh

# MỤC LỤC

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP .....	i
LỊCH TRÌNH THỰC HIỆN ĐỒ ÁN TỐT NGHIỆP .....	ii
LỜI CAM ĐOAN .....	iii
LỜI CẢM ƠN.....	iv
MỤC LỤC .....	v
LIỆT KÊ HÌNH .....	vii
LIỆT KÊ BẢNG .....	x
TÓM TẮT ĐỒ ÁN .....	xi
<b>Chương 1. TỔNG QUAN.....</b>	<b>1</b>
1.1 ĐẶT VẤN ĐỀ .....	1
1.2 MỤC TIÊU .....	2
1.3 NỘI DUNG THỰC HIỆN .....	2
1.4 GIỚI HẠN .....	2
1.5 BỐ CỤC .....	3
<b>Chương 2. CƠ SỞ LÝ THUYẾT .....</b>	<b>4</b>
2.1 NGUYÊN LÝ HOẠT ĐỘNG CỦA CÔNG TẮC .....	4
2.1.1 Điều khiển trực tiếp từ mô hình .....	4
2.1.2 Điều khiển thông qua web .....	4
2.2 GIỚI THIỆU VỀ PHẦN CỨNG.....	5
2.2.1 Mạch cảm ứng điện dung .....	5
2.2.2 Mạch công suất.....	8
2.2.3 Mạch xử lý trung tâm .....	10
2.2.4 Mạch nguồn .....	18
2.3 GIỚI THIỆU VỀ PHẦN MỀM.....	23
2.3.1 Web server.....	23
2.3.2 Cơ sở dữ liệu.....	24
<b>Chương 3. TÍNH TOÁN VÀ THIẾT KẾ .....</b>	<b>26</b>
3.1 GIỚI THIỆU.....	26
3.2 TÍNH TOÁN VÀ THIẾT KẾ HỆ THỐNG .....	27
3.2.1 Thiết kế sơ đồ khái hệ thống .....	27
3.2.2 Tính toán và thiết kế mạch .....	28
<b>Chương 4. THI CÔNG HỆ THỐNG .....</b>	<b>41</b>
4.1 GIỚI THIỆU.....	41
4.2 THI CÔNG HỆ THỐNG .....	41

4.2.1 Thi công bo mạch .....	41
4.2.2 Lắp ráp và kiểm tra.....	51
<b>4.3 ĐÓNG GÓI VÀ THI CÔNG MÔ HÌNH .....</b>	<b>57</b>
4.3.1 Đóng gói bộ điều khiển .....	57
4.3.2 Thi công mô hình.....	58
<b>4.4 LẬP TRÌNH HỆ THỐNG .....</b>	<b>58</b>
4.4.1 Lưu đồ giải thuật của Web Server.....	58
4.4.2 Lưu đồ giải thuật của ESP .....	60
4.4.3 Phần mềm lập trình cho ESP .....	61
4.4.4 Phần mềm lập trình cho Web .....	63
4.4.5 Phần mềm xây dựng cơ sở dữ liệu .....	64
<b>4.5 HƯỚNG DẪN SỬ DỤNG, THAO TÁC.....</b>	<b>65</b>
<b>Chương 5. KẾT QUẢ NHẬN XÉT ĐÁNH GIÁ .....</b>	<b>68</b>
5.1 MẠCH CẢM ỨNG ĐIỆN DUNG .....	68
5.2 MẠCH XỬ LÝ TRUNG TÂM .....	69
5.3 MẠCH CÔNG SUẤT .....	69
5.4 MẠCH NGUỒN.....	70
5.5 KẾT QUẢ MÔ HÌNH .....	71
5.5 KẾT QUẢ PHẦN MỀM .....	71
<b>Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b>	<b>73</b>
6.1 KẾT LUẬN .....	73
6.2 HƯỚNG PHÁT TRIỂN .....	73
<b>TÀI LIỆU THAM KHẢO.....</b>	<b>74</b>
<b>PHỤ LỤC .....</b>	<b>76</b>

## **LIỆT KÊ HÌNH**

Hình 2.1: IC Cảm ứng điện dung AT42QT2120.....	5
Hình 2.2: Sơ đồ chân của AT42QT2120.....	6
Hình 2.3: Triac BTA12 – 600BRG. ....	8
Hình 2.4: Sơ đồ chân của triac BTA12 – 600BRG. ....	8
Hình 2.5: Sơ đồ chân và cấu tạo của MOC3020. ....	9
Hình 2.6: Sơ đồ chân ESP8266. ....	11
Hình 2.7: Sơ đồ nguyên lý cho ESP8266. ....	13
Hình 2.8: Module tích hợp phô biến (Module ESP-12F). ....	14
Hình 2.9: Sơ đồ chân của module ESP-12F. ....	15
Hình 2.10: IC FT232. ....	17
Hình 2.11: IC CP2102. ....	17
Hình 2.12: IC PL-2303. ....	17
Hình 2.13: IC CH340G.....	18
Hình 2.14: IC nguồn LNK3206G. ....	19
Hình 2.15: Sơ đồ khối chức năng của LNK3206G. ....	20
Hình 2.16: Sơ đồ chân của LNK3206G. ....	20
Hình 2.17: IC MC34063.....	21
Hình 2.18: Sơ đồ chân của MC34063. ....	22
Hình 2.19: Sơ đồ khối chức năng của MC34063. ....	22
Hình 2.20: Cách thức giao tiếp với web server. ....	24
Hình 3.1: Sơ đồ khối của hệ thống. ....	27
Hình 3.2: Sơ đồ nguyên lý của khối cảm ứng điện dung. ....	28
Hình 3.3: Sơ đồ khối của mạch ESP-12. ....	30
Hình 3.4: Sơ đồ nguyên lý của mạch reset. ....	31
Hình 3.5: Sơ đồ nguyên lý của mạch nguồn sử dụng nguồn từ cổng USB. ....	31
Hình 3.6: Sơ đồ nguyên lý mạch nạp cho ESP.....	32
Hình 3.7: Sơ đồ nguyên lý mạch điều khiển GPIO. ....	33
Hình 3.8: Sơ đồ nguyên lý của khối xử lý trung tâm. ....	34
Hình 3.9: Sơ đồ nguyên lý của khối công suất.....	37
Hình 3.10: Sơ đồ nguyên lý của mạch 220VAC – 12VDC.....	38
Hình 3.11: Sơ đồ nguyên lý mạch 12VDC-3,3VDC.....	39

Hình 4.1: Ảnh thực tế board mạch của mô hình.....	42
Hình 4.2: Sơ đồ bố trí linh kiện mặt trên của PCB thứ 1 .....	42
Hình 4.3: Sơ đồ đi dây mặt trên của PCB thứ 1 .....	43
Hình 4.4: Sơ đồ bố trí linh kiện mặt dưới của PCB thứ 1 .....	43
Hình 4.5: Sơ đồ đi dây mặt dưới của PCB thứ 1 .....	44
Hình 4.6: Sơ đồ mạch in của PCB thứ 1 .....	44
Hình 4.7: Sơ đồ bố trí linh kiện mặt trên của PCB thứ 2 .....	45
Hình 4.8: Sơ đồ đi dây mặt trên của PCB thứ 2 .....	45
Hình 4.9: Sơ đồ bố trí linh kiện mặt dưới của PCB thứ 2 .....	46
Hình 4.10: Sơ đồ đi dây mặt dưới của PCB thứ 2 .....	46
Hình 4.11: Sơ đồ mạch in của PCB thứ 2 .....	47
Hình 4.12: Sơ đồ bố trí linh kiện mặt trên của PCB thứ 3 .....	47
Hình 4.13: Sơ đồ đi dây mặt trên của PCB thứ 3 .....	48
Hình 4.14: Sơ đồ bố trí linh kiện mặt dưới của PCB thứ 3 .....	48
Hình 4.15: Sơ đồ đi dây mặt dưới của PCB thứ 3 .....	48
Hình 4.16: Sơ đồ mạch in của PCB thứ 3 .....	49
Hình 4.17: Mặt trên khói nguồn .....	51
Hình 4.18: Mặt dưới khói nguồn .....	52
Hình 4.19: Mặt trên khói công suất .....	53
Hình 4.20: Mặt dưới khói công suất .....	53
Hình 4.21: Mặt trên PCB thứ 2 .....	54
Hình 4.22: Mặt dưới PCB thứ 2 .....	55
Hình 4.23: Mặt trên của mạch nạp USB-UART .....	56
Hình 4.24: Mặt dưới của mạch nạp USB-UART .....	56
Hình 4.25: Board công tắc IoT .....	57
Hình 4.26: Thiết kế vỏ hộp bằng phần mềm Solidworks .....	58
Hình 4.27: Lưu đồ của server .....	59
Hình 4.28: Lưu đồ của ESP .....	60
Hình 4.29: Phần mềm Arduino .....	62
Hình 4.30: Phần mềm Visual Studio .....	64
Hình 4.31: Phần mềm SQL Server Management Studio .....	65
Hình 4.32: Các nút nhấn cảm ứng điện dung .....	65

Hình 4.33: Giao diện đăng nhập vào web server .....	66
Hình 4.34: Phòng số 01 .....	66
Hình 4.35: Phòng khách .....	66
Hình 4.36: Giao diện điều khiển và giám sát thiết bị .....	67
Hình 5.1: Mô hình hoàn chỉnh sau thời gian thực hiện. ....	71
Hình 5.2: Giao diện web server. ....	71
Hình 5.3: Giao diện điều khiển thiết bị của web server. ....	72

## **LIỆT KÊ BẢNG**

Bảng 2.1 Mô tả các chân của AT42QT2120 ở chế độ Standalone.....	7
Bảng 2.2: Các chế độ boot của ESP8266 và cấu hình chân GPIO.....	12
Bảng 2.3: Một số module được AI-Thinker sản xuất.....	14
Bảng 3.1: Trạng thái chuyển mạch tự động .....	32
Bảng 3.2: Liệt kê công suất tiêu thụ của một số thiết bị điện gia dụng .....	35
Bảng 3.3: Thông số điện áp và dòng tiêu thụ của các linh kiện trong mô hình .....	38
Bảng 4.1: Danh sách linh kiện.....	49

## TÓM TẮT ĐỒ ÁN

IoT vẫn còn là khái niệm khá mới với nhiều người. Nói một cách đơn giản, IoT là một hệ thống kết nối mọi thứ xung quanh chúng ta lại với nhau qua Internet, như xe hơi, các vật dụng trong nhà và thậm chí những hệ thống lớn phức tạp như đèn giao thông hay các cảm biến thời tiết. Chúng ta có thể điều khiển, quản lý hoặc thu thập các thông tin từ chúng một cách dễ dàng qua các thiết bị cầm tay.

IoT đã và đang làm thay đổi cả thế giới, tại Việt Nam trong khoảng 4 năm trở lại đây, IoT đã từng bước được ứng dụng trong một số lĩnh vực của cuộc sống, tuy nhiên chỉ dừng lại ở mức rời rạc, chưa đồng bộ. Những lợi ích và tiềm năng phát triển của IoT là rất lớn, ở đâu có kết nối Internet ở đó đều có khả năng xuất hiện các thiết bị IoT mang đến giá trị thông qua việc truyền tải và trao đổi thông tin, dữ liệu.

Dù cho sự phát triển của IoT đang tăng lên với tốc độ chóng mặt, nhưng việc điều khiển thiết bị dân dụng vẫn còn dùng công tắc cơ. Công tắc cơ đã trở nên lạc hậu trong thời đại “Internet thế giới phẳng”. Công tắc cần được thay đổi theo cách hiện đại hơn, việc kết hợp 2 ý tưởng công tắc điều khiển và IoT mà đề tài “thiết kế và thi công mô hình ứng dụng IoT vào việc điều khiển giám sát các thiết bị điện trong nhà” ra đời. Trên thị trường Việt Nam hiện nay, ý tưởng công tắc kết nối Internet, công tắc thông minh đã được thực hiện bởi các doanh nghiệp tiêu biểu như LUMI, BKAV,... Đề tài này với mục tiêu đặt ra là đạt được những tính năng mà các công tắc thông minh trên thị trường đang có. Là có thể điều khiển thiết bị từ xa thông qua Internet và điểm nổi bật của mô hình là có thể giám sát trạng thái của thiết bị, có server quản lý và điều khiển thiết bị.

## Chương 1. TỔNG QUAN

### 1.1 ĐẶT VẤN ĐỀ

Mỗi giai đoạn phát triển của lịch sử thế giới đều gắn liền với những cuộc cách mạng về khoa học kỹ thuật. Và ngày nay, cuộc cách mạng Internet of Things đã tạo nên những thay đổi đáng kể cuộc sống của chúng ta ở hiện tại và trong tương lai. Với sự phát triển của Internet, Smartphone và đặc biệt là các thiết bị cảm biến, Internet of Things (IoT) đang trở thành xu hướng mới của thế giới. IoT là một mạng lưới các vật thể được gắn các cảm biến hoặc hệ thống điện tử đặc biệt cho phép chúng kết nối với nhau để thu thập và trao đổi dữ liệu. Các vật thể trong mạng lưới này có thể được kết nối với mạng Internet cho mục đích điều khiển và giám sát từ xa. Việc chúng ta vào nhà, mở cửa, đèn sẽ tự động sáng ở chỗ ta đang đứng, điều hòa sẽ tự động điều chỉnh nhiệt độ, nhạc sẽ tự động bật lên,... những điều chỉ có trong phim khoa học viễn tưởng mà chúng ta thường xem, đang dần trở thành hiện thực với công nghệ IoT.

Trong cuộc sống thường nhật, chúng ta đã quá quen thuộc với việc bật tắt các thiết bị bằng công tắc thông thường. Với cuộc sống bô bô bè ngày nay, chúng ta bị chi phối bởi nhiều thứ. Việc chúng ta ra khỏi nhà mà quên tắt đèn, điều hòa là chuyện không hiếm gặp. Với công tắc thông thường, khi chúng ta rời khỏi nhà mà vẫn quên tắt các thiết bị trong nhà. Để tắt các thiết bị thì chỉ cách quay trở lại về nhà rồi tắt chúng. Điều này đôi khi gây ra cho chúng ta nhiều phiền toái.

Để giải quyết vấn đề trên, nhóm đã lựa chọn đề tài: "**THIẾT KẾ VÀ THI CÔNG MÔ HÌNH ỨNG DỤNG IOT VÀO VIỆC ĐIỀU KHIỂN GIÁM SÁT CÁC THIẾT BỊ ĐIỆN TRONG NHÀ**", ứng dụng công nghệ IoT vào đời sống. Giúp chúng ta có thể bật tắt các thiết bị trong nhà ở mọi lúc mọi nơi. Ngoài chức năng bật tắt các thiết bị từ xa, đề tài của nhóm sẽ xây dựng thêm các chức năng giám sát trạng thái điều khiển, bật tắt của thiết bị. Đây là một đề tài không mới, nhiều anh chị khóa trước cũng đã thực hiện. Nhưng vẫn còn nhiều điểm cần cải thiện đó là tốc độ đáp ứng khi điều khiển thiết bị và giao diện điều khiển thiết bị. Vì vậy đề tài của nhóm trọng tâm sẽ thực hiện việc cải thiện tốc độ điều khiển thiết bị lên mức tối đa có thể, xây dựng giao diện điều khiển thiết bị có tính thẩm mỹ và thân thiện với người dùng.

## **1.2 MỤC TIÊU**

Tìm hiểu và thực hiện điều khiển các thiết bị trong nhà như đèn, quạt,... thông qua mạng Internet. Cụ thể là tìm hiểu chip ESP8266 để điều khiển thiết bị thông qua mạng wifi.

Tìm hiểu và tiến hành xây dựng cơ sở dữ liệu, truyền nhận dữ liệu giữa các thiết bị và server.

Xây dựng giao diện web server để điều khiển và giám sát thiết bị.

Thiết kế và thi công mô hình.

## **1.3 NỘI DUNG THỰC HIỆN**

**NỘI DUNG 1:** Tìm hiểu nguyên lý hoạt động và thiết kế mạch điều khiển sử dụng chip ESP8266.

**NỘI DUNG 2:** Thiết kế mạch nguồn và mạch công suất cho mô hình.

**NỘI DUNG 3:** Tìm hiểu và xây dựng cơ sở dữ liệu.

**NỘI DUNG 4:** Thiết kế lưu đồ giải thuật và viết chương trình điều khiển thiết bị, thiết kế giao diện web server và để điều khiển, giám sát trạng thái đóng tắt của thiết bị thông qua Internet.

**NỘI DUNG 5:** Thi công mô hình.

**NỘI DUNG 6:** Thủ nghiệm và điều chỉnh phần cứng cũng như chương trình để mô hình được tối ưu. Đánh giá các thông số của mô hình.

**NỘI DUNG 7:** Viết báo cáo thực hiện.

## **1.4 GIỚI HẠN**

- Mô hình chỉ điều khiển các thiết bị trong nhà có công suất dưới 1000W.
- Mô hình có thể điều khiển tối đa là 4 thiết bị.
- Web server chỉ có thể điều khiển và giám sát trạng thái của 4 thiết bị.

## 1.5 BỘ CỤC

### Chương 1: Tổng Quan

Chương này trình bày các phần: đặt vấn đề, lý do chọn đề tài, mục tiêu, nội dung thực hiện, giới hạn và bối cảnh của đồ án.

### Chương 2: Cơ Sở Lý Thuyết

Chương này trình bày quy trình hoạt động của mô hình. Cụ thể:

- Trình bày về lý thuyết về mạch nguồn, nguyên tắc hoạt động của mạch công suất, chip ESP8266 và mạch cảm ứng điện dung.
- Trình bày lý thuyết về web server và cơ sở dữ liệu. Phương thức truyền nhận dữ liệu giữa thiết bị và web server.

### Chương 3: Tính Toán và Thiết Kế

Trình bày các phương án thiết kế, tính toán các thông số và từ đó tiến hành thiết kế mạch nguồn, mạch công suất, mạch xử lý trung tâm, mạch cảm ứng. Thiết kế cơ sở dữ liệu, xây dựng giao diện web server.

### Chương 4: Thi Công Hệ Thống

Trong chương này trình bày quá trình thực hiện thi công các board mạch, các phần điều khiển và phần cứng mô hình của đề tài.

Sau đó lắp ráp một hệ thống mô hình hoàn chỉnh đã xây dựng trước đó, để từ đó xây dựng nên lưu đồ giải thuật và viết chương trình điều khiển cho hệ thống.

Sau khi có được mô hình và chương trình điều khiển, tiến hành cho hệ thống hoạt động, tìm lỗi và khắc phục lỗi nếu có.

### Chương 5: Kết Quả, Nhận Xét và Đánh Giá

Chương này trình bày kết quả mô hình, những yêu cầu đạt được và chưa đạt được bao gồm cả về phần cứng và phần mềm. Từ đó đưa ra nhận xét, đánh giá và tìm cách khắc phục.

### Chương 6: Kết luận và Hướng phát triển

Trình bày tóm tắt những việc làm được và chưa làm được của cả mô hình. Nêu hướng phát triển của sản phẩm trong tương lai.

## **Chương 2. CƠ SỞ LÝ THUYẾT**

### **2.1 NGUYÊN LÝ HOẠT ĐỘNG CỦA MÔ HÌNH**

Mô hình được sử dụng để đóng tắt thiết bị điện trong nhà 220VAC. Nguồn điện 220VAC là nguồn cung cấp cho toàn mô hình, 220VAC sẽ được chuyển đổi thành nguồn 12VDC, trước khi hạ xuống thành nguồn 3,3VDC cấp cho các board mạch trong mô hình hoạt động. Công tắc có 4 phím cảm ứng, để điều khiển 4 thiết bị. Quá trình hoạt động như sau:

#### **2.1.1 Điều khiển trực tiếp từ mô hình**

Khi người dùng chạm vào bề mặt phím cảm ứng điện dung của mô hình, tín hiệu điều khiển được đưa tới bộ xử lý trung tâm, sau đó bộ xử lý trung tâm gửi tín hiệu điều khiển đến mạch công suất, cho phép đóng tắt thiết bị tương ứng. Tiếp theo gửi dữ liệu chứa trạng thái của thiết bị lên server thông qua Internet, cụ thể là mạng wifi. Server sẽ tiếp nhận dữ liệu vừa gửi tới, kế đến tiến hành xử lý và lưu dữ liệu vào cơ sở dữ liệu. Cập nhật trạng thái hiện tại của thiết bị trên web. Người dùng có thể truy cập vào web để biết được trạng thái hiện tại của thiết bị.

#### **2.1.2 Điều khiển thông qua web**

Người dùng truy cập vào web, server sẽ trả về trạng thái hiện tại của thiết bị lên web. Người dùng có thể lựa chọn thiết bị cần điều khiển. Khi người dùng tác động vào web, server sẽ gửi dữ liệu điều khiển về bộ xử lý trung tâm. Bộ xử lý trung tâm nhận dữ liệu, sau đó tiến hành tác động để đóng mở thiết bị tương ứng và đồng thời server sẽ cập nhật trạng thái của thiết bị vừa được tác động.

## 2.2 GIỚI THIỆU VỀ PHẦN CỨNG

### 2.2.1 Mạch cảm ứng điện dung

#### a. *Tổng quan [1]*

Cảm ứng điện dung là công nghệ cảm ứng dựa trên những thay đổi của điện tích trên màn hình khi tay người chạm nhẹ vào. Công nghệ này ra đời như một sự kế tục của các công nghệ cảm ứng trước đó như cảm ứng hồng ngoại, cảm ứng sóng âm bể mặt và cảm ứng điện trở.

Ưu điểm của công nghệ này là người dùng không cần tác động mạnh lên lớp cảm ứng, chỉ cần chạm nhẹ.

Về bản chất, cảm ứng điện dung có thể chia thành 2 loại: Một là cảm ứng đơn điểm, chỉ nhận được tối đa 1 chạm trong quá trình thao tác, hai là cảm ứng đa điểm (multi-touch), nghĩa là cho phép người dùng thực hiện được nhiều thao tác chạm cùng một lúc.

#### b. IC cảm ứng điện dung AT42QT2120 [2]



Hình 2.1: IC Cảm ứng điện dung AT42QT2120.

AT42QT2120 sử dụng nguồn từ 1,8 ~ 5,5V, chỉ cần chạm vào lớp đồng tương ứng với kênh cảm ứng điện dung thì kênh cảm ứng điện dung sẽ hoạt động tương tự như tác động vào nút nhấn.

Khi có điện dung thay đổi trên lớp đồng (nối với ngõ vào của IC AT42QT2120) thì ngõ ra tương ứng của IC sẽ xuất ra 1 mức điện áp.

**Thông số kỹ thuật:**

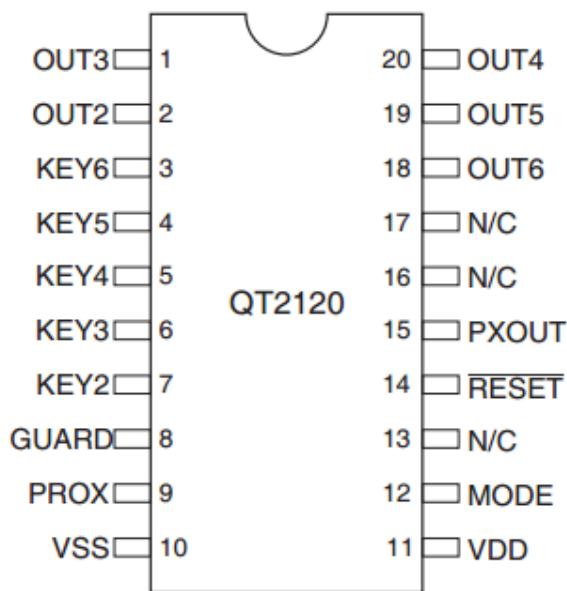
- Có 5 kênh cảm ứng, tối đa 12 kênh khi giao tiếp thông chuẩn I2C.
- Điện áp làm việc từ 1,8 ~ 5,5V.
- Dòng tiêu thụ khi hoạt động khoảng 0,55mA.
- Kênh cảm ứng, có thể được tác động từ lớp nhựa có độ dày lên đến 10mm, lớp thủy tinh có độ dày lên đến 20mm.

AT42QT2120 có 2 chế độ làm việc: chế độ Comms và chế độ Standalone.

Với chế độ Comms: AT42QT2120 có thiết lập tối đa 12 kênh cảm ứng điện dung thông qua chuẩn giao tiếp I2C. Ở chế độ này, phải cần 1 vi điều khiển để giao tiếp với AT42QT2120.

Với chế độ Standalone: AT42QT2120 hoạt động độc lập, có thể thiết lập tối đa 5 kênh cảm ứng điện dung.

AT42QT2120 có dãy điện áp hoạt động rộng, dòng tiêu thụ thấp, có 5 kênh cảm ứng đáp ứng hơn cả yêu cầu của đề tài là 4 kênh, có thể mở rộng lên thành 12 kênh dễ dàng cho việc mở rộng và phát triển đề tài với quy mô lớn. Vì thế nhóm quyết định chọn AT42QT2120 để xây dựng mạch cảm ứng điện dung.



Hình 2.2: Sơ đồ chân của AT42QT2120.

Bảng 2.1 Mô tả các chân của AT42QT2120 ở chế độ Standalone

STT	Tên	Kiểu chân	Mô tả
1	KEY6	I/O	Ngõ vào cảm ứng điện dung kênh thứ 6
2	KEY5	I/O	Ngõ vào cảm ứng điện dung kênh thứ 5
3	KEY4	I/O	Ngõ vào cảm ứng điện dung kênh thứ 4
4	KEY3	I/O	Ngõ vào cảm ứng điện dung kênh thứ 3
5	KEY2	I/O	Ngõ vào cảm ứng điện dung kênh thứ 2
6	GUARD	I/O	Kênh bảo vệ (kênh thứ 1)
7	PROX	I/O	Kênh này được sử dụng như 1 cảm biến tiệm cận (kênh 0)
8	VSS	P	Nối với mass
9	VDD	P	Nối với nguồn
10	MODE	I	Dùng để lựa chọn chế độ hoạt động Chế độ Comms: nối chân MODE với VSS Chế độ Standalone: nối chân MODE với VDD
11	N/C	OD	Ở chế độ Standalone, chân này không sử dụng
12	RESET	I	Dùng để reset AT42Q2120, reset tích cực mức thấp
13	PXOUT	OD	Ngõ ra cực máng hở của kênh cảm biến tiệm cận
14	N/C	OD	Ở chế độ Standalone, chân này không sử dụng
15	N/C	OD	Ở chế độ Standalone, chân này không sử dụng
16	OUT6	I/O	Ngõ ra tín hiệu của kênh cảm ứng điện dung thứ 6
17	OUT5	I/O	Ngõ ra tín hiệu của kênh cảm ứng điện dung thứ 5
18	OUT4	I/O	Ngõ ra tín hiệu của kênh cảm ứng điện dung thứ 4
19	OUT3	I/O	Ngõ ra tín hiệu của kênh cảm ứng điện dung thứ 3
20	OUT2	I/O	Ngõ ra tín hiệu của kênh cảm ứng điện dung thứ 2

**Chú thích kiểu chân:**

- I: Chỉ có thể là chân ngõ vào (Input)
- I/O: Có thể là chân ngõ vào hoặc ngõ ra (Input/Output)
- P: Chân nguồn (Power)
- OD: Chân loại cực máng hở (Open Drain)

## 2.2.2 Mạch công suất

Mạch xử lý trung tâm sử dụng điện áp 3,3VDC, có công suất thấp nên không thể đóng tắt được thiết bị điện 220VAC. Để điều khiển được thiết bị, mạch xử lý trung tâm cần gửi tín hiệu điều khiển đến một mạch điện khác, đó là mạch công suất.

Có nhiều loại mạch dùng để đóng cắt thiết bị điện 220VAC, nhưng thông dụng nhất là mạch công suất dùng relay và mạch công suất sử dụng triac.

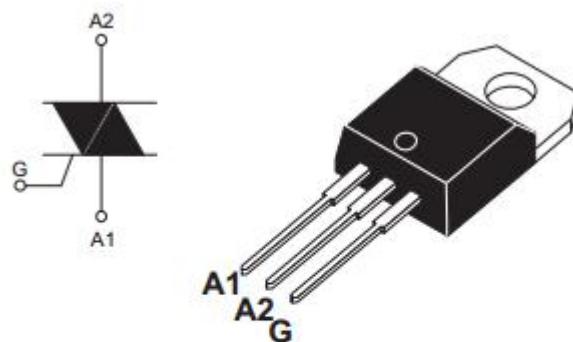
Việc sử dụng mạch công suất dùng relay có 2 nhược điểm lớn mà người dùng cần quan tâm là gây ồn và dễ gây nhiễu cho các thiết bị xung quanh. Với mạch công suất sử dụng triac, khi đóng cắt thiết bị sẽ không gây ra tiếng ồn, hạn chế phát sinh tia lửa điện như việc đóng cắt bằng relay nên giảm thiểu được việc gây nhiễu cho các thiết bị xung quanh. Với những ưu điểm trên nên nhóm quyết định sử dụng mạch công suất dùng triac để đóng tắt thiết bị điện 220VAC.

### a. Triac BTA12 – 600BRG [3]



Hình 2.3: Triac BTA12 – 600BRG.

Triac BTA12 – 600BRG là linh kiện bán dẫn có 3 cực lớp, làm việc như 2 thyristor mắc song song ngược chiều, có thể dẫn điện theo hai chiều.



**TO-220AB Insulated  
(BTA12)**

Hình 2.4: Sơ đồ chân của triac BTA12 – 600BRG.

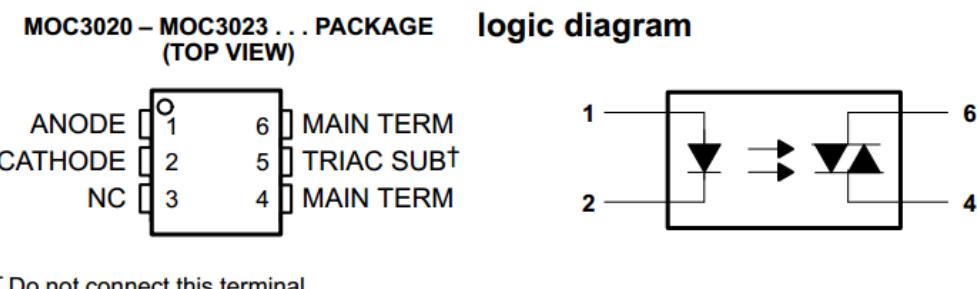
Triac BTA12 được sử dụng trong các ứng dụng điều chỉnh điện áp xoay chiều, điều khiển động cơ, điều khiển đóng tắt thiết bị điện,...

### Thông số kỹ thuật:

- Điện áp cực đại chịu được: 600V.
- Dòng điện thuận cực đại: 12A.
- Điện áp điều khiển mở van: 1,3V
- Dòng điều khiển mở van: 50mA
- Nhiệt độ làm việc: - 40°C ~ 125°C.

### b. Opto Coupler MOC3020 [4]

Opto MOC3020 là một linh kiện bán dẫn cấu tạo gồm 1 bộ phát quang và một cảm biến quang tích hợp trong 1 khối bán dẫn. Bộ phát quang là 1 diode phát quang dùng để phát ra ánh sáng kích cho các cảm biến quang (triac) dẫn.



† Do not connect this terminal  
NC – No internal connection

Hình 2.5: Sơ đồ chân và cấu tạo của MOC3020.

MOC3020 được dùng để cách ly giữa các khối chênh lệch nhau về điện áp hay công suất như khối có công suất nhỏ với khối điện áp lớn, có thể dùng để chống nhiễu cho các mạch cầu H, ngõ ra PLC, chống nhiễu cho các thiết bị đo lường.

Nguyên lý hoạt động: khi có dòng điện đủ lớn đi qua 2 đầu led trong opto, sẽ làm cho led phát sáng. Khi led phát sáng làm thông 2 cực của triac, mở cho dòng điện chạy qua.

Thông số kỹ thuật:

#### Emitter:

- Dòng duy trì tối đa: 60mA.
- Điện áp ngược tối đa: 3V.
- Điện áp thuận đầu vào: 1,5V.

**Detector**

- Điện áp đầu ra tối đa chịu được: 400VAC.
- Dòng tối đa trên triac: 1A.
- Dòng kích LED tối đa: 30mA.
- Dòng duy trì: 100 $\mu$ A.

Mạch xử lý trung tâm sử dụng mức điện áp là 3,3V, để giao tiếp với mạch công suất dùng triac để điều khiển thiết bị điện 220VAC. Cần sử dụng opto để mạch xử lý trung tâm có thể điều khiển mạch công suất. Với các đặc tính trên, opto MOC3020 hoàn toàn đáp ứng được các yêu cầu sử dụng, nên nhóm quyết định sử dụng MOC3020.

**2.2.3 Mạch xử lý trung tâm**

Mạch xử lý trung tâm sẽ nhận tín hiệu từ mạch cảm ứng điện dung, gửi tín hiệu yêu cầu bật tắt thiết bị tương ứng đến mạch công suất.

**a. ESP8266 [5][6][7]**

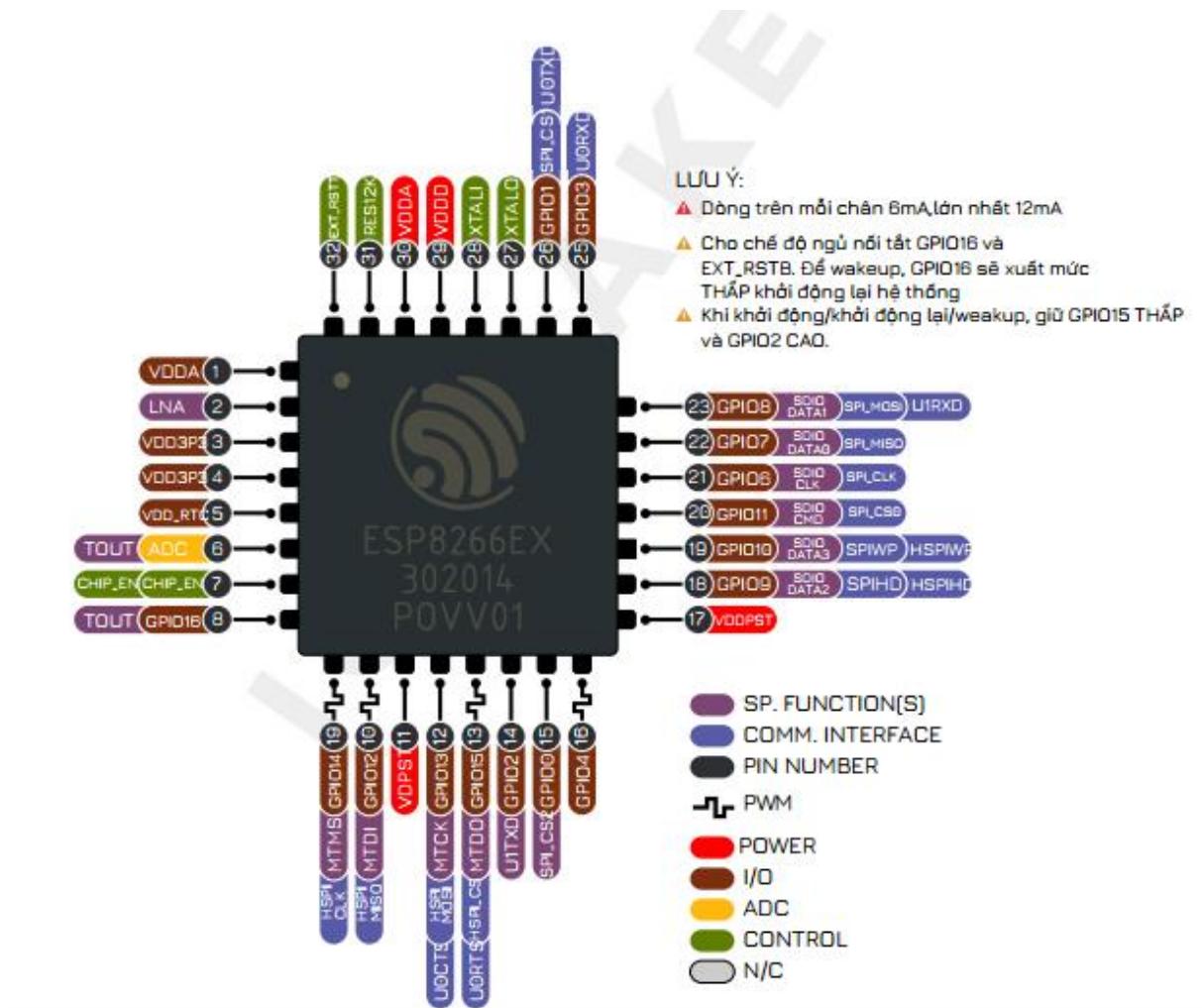
ESP8266 là dòng chip tích hợp Wi-Fi 2,4GHz có thể lập trình được, giá thành thấp được sản xuất bởi công ty bán dẫn Trung Quốc: Espressif Systems.

Được phát hành đầu tiên vào tháng 8 năm 2014, đóng gói đưa ra thị trường dạng Module ESP-01, được sản xuất bởi bên thứ 3: AI-Thinker. Có khả năng kết nối Internet qua mạng Wi-Fi một cách nhanh chóng và sử dụng rất ít linh kiện đi kèm. Với giá cả có thể nói là rất rẻ so với tính năng và khả năng ESP8266 có thể làm được.

ESP8266 có một cộng đồng các nhà phát triển trên thế giới rất lớn, cung cấp nhiều module lập trình mã nguồn mở giúp nhiều người có thể tiếp cận và xây dựng ứng dụng rất nhanh.

Hiện nay tất cả các dòng chip ESP8266 trên thị trường đều mang nhãn ESP8266EX, là phiên bản nâng cấp của ESP8266.

## Sơ đồ chân



Hình 2.6: Sơ đồ chân ESP8266.

## Thông số kỹ thuật

- 32-bit RISC CPU : Tensilica Xtensa LX106 chạy ở xung nhịp 80 MHz.
- Hỗ trợ Flash ngoài từ 512KiB đến 4MiB.
- 64KBytes RAM thực thi lệnh.
- 96KBytes RAM dữ liệu.
- 64KBytes boot ROM.
- Chuẩn wifi IEEE 802.11 b/g/n, Wi-Fi 2.4 GHz.
- Tích hợp TR switch, balun, LNA, khuếch đại công suất và matching network.
- Hỗ trợ WEP, WPA/WPA2, Open network.
- Tích hợp giao thức TCP/IP.
- Hỗ trợ nhiều loại anten.

- Nguồn cấp từ 2,5VDC ~ 3,6VDC.
- Dòng điện khi hoạt động trung bình khoảng 80mA.
- Dòng I/O tối đa: 12mA.
- 16 chân GPIO.
- Hỗ trợ SDIO 2.0, UART, SPI, I<sup>2</sup>C, PWM, I<sup>2</sup>S.
- 1 ADC 10-bit.
- Dải nhiệt độ hoạt động rộng : -40°C ~ 125°C.

Do không hỗ trợ bộ nhớ Flash nên các board sử dụng ESP8266 phải gắn thêm Flash bên ngoài, để ESP8266 có thể đọc chương trình ứng dụng với chuẩn SPI hoặc SDIO.

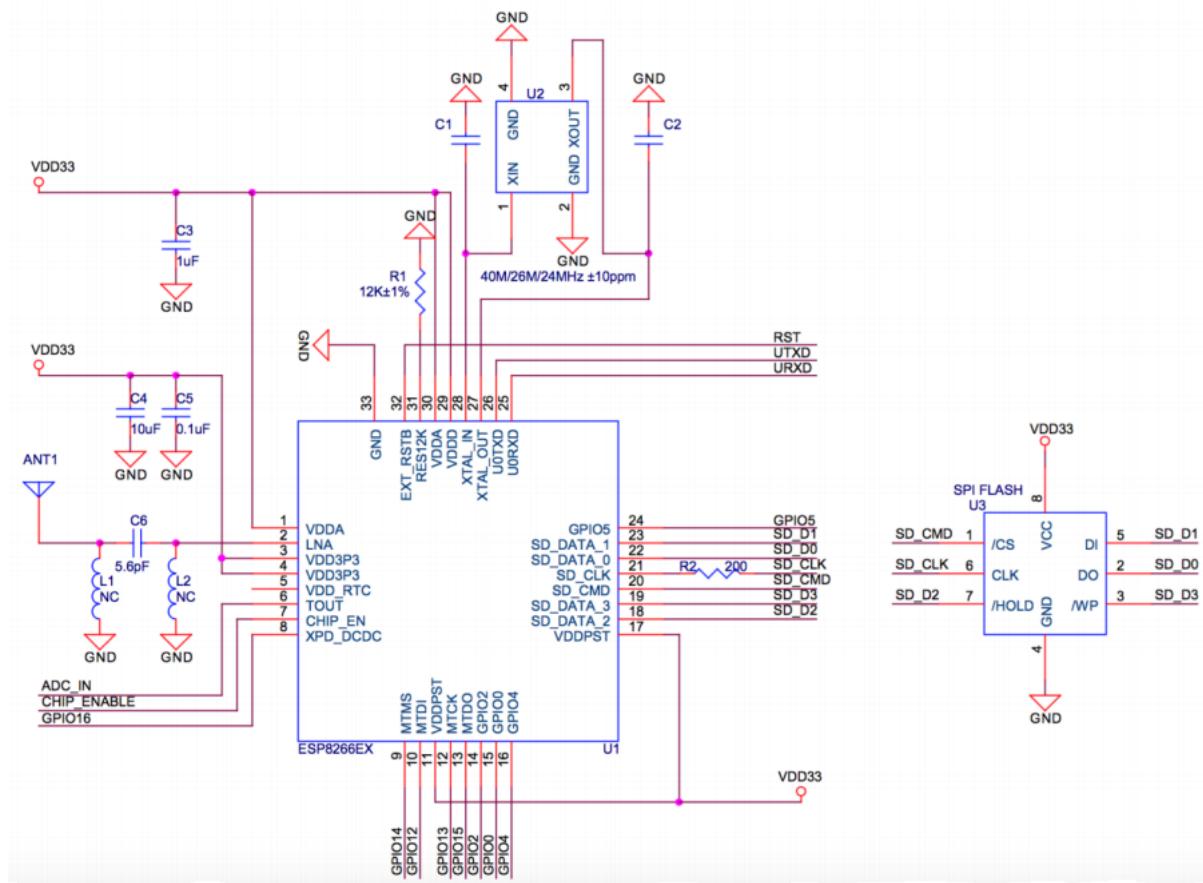
### Các chế độ Boot của ESP8266

Bảng 2.2: Các chế độ boot của ESP8266 và cấu hình chân GPIO

MTDO	GPIO0	GPIO2	Mode	Description
L	L	H	UART	Download code from UART
L	H	H	Flash	Boot from SPI Flash
H	x	x	SDIO	Boot from SD card

Chân MTD0 là chân GPIO15 của ESP8266. Ta có thể kết nối với điện trở kéo lên hoặc kéo xuống, dùng nút nhấn,... trên board để tạo tín hiệu High/Low cho các chân để chọn bộ nhớ chứa code trên board mà ESP8266 có thể đọc vào và thực thi (ví dụ như SPI Flash, SD Card). Ngoài ra ESP8266 còn có chế độ cho phép nạp code ứng dụng từ máy tính thông qua UART và lưu vào bộ nhớ SPI Flash trên board. Chế độ này dùng để nạp code mới cho các board ESP8266.

Để ESP8266 hoạt động, tất nhiên phải cần vài linh kiện cần thiết. Trong đó phần khó nhất là anten. Đòi hỏi phải được sản xuất, kiểm tra với các thiết bị hiện đại. Do đó trên thị trường xuất hiện nhiều module và board mạch phát triển để người dùng có thể phát triển các ứng dụng.



Hình 2.7: Sơ đồ nguyên lý cho ESP8266.

Ngoài trừ module ESP-WROOM-02 được phát triển bởi chính Espressif cho mục đích nghiên cứu các tính năng của ESP8266, các module ứng dụng phổ biến hiện nay của ESP8266 đều được phát triển bởi công ty AI-Thinker.

Hiện tại có khá nhiều module khác nhau cho ESP8266 được sản xuất bởi công ty AI-Thinker. Đặc điểm khác nhau giữa các module này bao gồm:

- Loại anten được sử dụng (PCB anten, chip anten hoặc gắn anten ngoài).
- Dung lượng chip Flash SPI trên board.
- Kích thước board của module.
- Có gắn khung nhôm chống nhiễu hoặc không.
- Số lượng GPIO đưa ra chân kết nối.

Hiện tại AI-Thinker sản xuất 14 loại module cho ESP từ module ESP-01 đến ESP-14.

Bảng 2.3: Một số module được AI-Thinker sản xuất

Tên	Số chân	Pitch	LEDs	Antenna	Shielded	Dimensions
ESP-01	6	0.1"	Yes	PCB	No	14.3 x 24.8
ESP-02	6	0.1"	No	U-FL	No	14.2 x 14.2
ESP-03	10	2mm	No	Ceramic	No	17.3 x 12.1
ESP-04	10	2mm	No	None	No	14.7 x 12.1
ESP-05	3	0.1"	No	U-FL	No	14.2 x 14.2
ESP-06	11	misc	No	None	Yes	14.2 x 14.7
ESP-07	14	2mm	Yes	Ceramic+U-FL	Yes	20.0 x 16.0
ESP-08	10	2mm	No	None	Yes	17.0 x 16.0
ESP-09	10	misc	No	None	No	10.0 x 10.0
ESP-10	3	2mm	No	None	No	14.2 x 10.0
ESP-11	6	0.05"	No	Ceramic	No	17.3 x 12.1
ESP-12	14	2mm	Yes	PCB	Yes	24.0 x 16.0
ESP-12E	20	2mm	Yes	PCB	Yes	24.0 x 16.0
ESP-12F	20	2mm	Yes	PCB	Yes	24.0 x 16.0
ESP-13	16	1.5mm	No	PCB	Yes	18.0 x 20.0
ESP-14	22	2mm	No	PCB	Yes	24.3 x 16.2

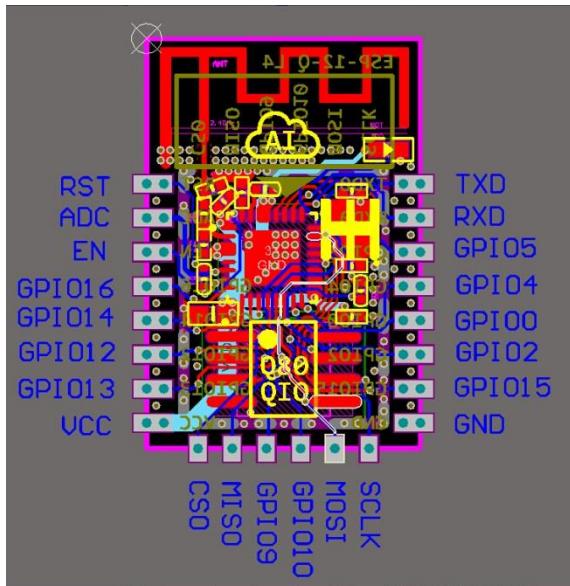
Ở thị trường Việt Nam có 3 module được sử dụng phổ biến và rộng rãi, đó là: ESP-01, ESP-07 và ESP-12F.



Hình 2.8: Module tích hợp phổ biến (Module ESP-12F).

Theo yêu cầu đề tài của nhóm đề ra sẽ sử dụng 4 nút cảm ứng điện dung ngõ vào tương ứng để điều khiển 4 thiết bị ngõ ra, vì thế số lượng chân GPIO để kết nối ít nhất

là 8. Chính vì vậy nhóm quyết định lựa chọn module ESP-12F. Với các đặc điểm, thông số kỹ thuật phù hợp với yêu cầu.



Hình 2.9: Sơ đồ chân của module ESP-12F.

#### **Đặc điểm kỹ thuật của module ESP-12F:**

- Sử dụng PCB anten on-board.
- Module đưa ra 11 chân GPIO, 2 chân Tx, Rx cho UART, các chân cho SPI, chân RST để reset chip, 1 chân ADC.
- Dung lượng SPI Flash là 4MByte.
- Có thể hàn jumper để cắm dây vào các board khác hoặc hàn trực tiếp lên board ứng dụng.

#### **b. IC CH340G USB-UART [8] [9]**

Để mạch xử lý trung tâm có thể hoạt động theo ý muốn, ta cần phải viết chương trình (code) cho nó. Khi viết chương trình xong, để mạch xử lý trung tâm có thể hoạt động theo những yêu cầu mà chương trình đã viết. Ta cần phải nạp chương trình vào mạch xử lý trung tâm, bằng cách sử dụng mạch nạp.

ESP8266 hỗ trợ cho người dùng 3 chế độ nạp (boot mode): nạp chương trình từ máy tính thông qua UART, nạp chương trình SPI Flash thông qua SPI và nạp chương trình từ SD Card thông qua SDIO. Với 3 chế độ trên, chế độ nạp thông qua SPI và SDIO gây bất tiện cho người dùng, để nạp chương trình người dùng cần phải lưu chương trình vào SPI Flash hay SD Card. Còn với chế độ nạp thông qua UART, người dùng có thể nạp chương trình trực tiếp từ máy tính, ta có thể thấy rõ sự linh hoạt vượt

trội ở chế độ này so với 2 chế độ còn lại. Để thực hiện chế độ nạp này, ta cần thiết kế mạch chuyển đổi từ USB của máy tính sang UART để nạp chương trình cho ESP8266.

**UART (Universal Asynchronous serial Reveiver and Transmitter)** là bộ truyền nhận nối tiếp không đồng bộ. Khái niệm UART dùng để chỉ thiết bị phần cứng (hardware), không phải chuẩn giao tiếp. UART cần phải kết hợp với một thiết bị chuyển đổi mức điện áp để tạo ra một chuẩn giao tiếp như chuẩn RS232 (máy tính sẽ sử dụng cổng COM để truyền nhận dữ liệu bằng UART thông qua chuẩn RS232). UART thường dùng để truyền nhận dữ liệu giữa 2 hay nhiều thiết bị khác nhau (có thể là giữa vi điều khiển với vi điều khiển, giữa vi điều khiển với máy tính, vi điều khiển với module sim,...). Ở kiểu truyền nối tiếp bất đồng bộ bao gồm 1 đường truyền dữ liệu và 1 đường nhận dữ liệu, không có tín hiệu xung clock nên gọi là bất đồng bộ. Để có thể truyền nhận dữ liệu, yêu cầu cả bên truyền và bên nhận phải tự tạo xung clock có cùng tần số, gọi là tốc độ truyền dữ liệu (baud).

#### **Các thông số cơ bản trong việc truyền nhận UART:**

**Baund rate (tốc độ baud):** số bit được truyền trong 1 giây. Thông số này phải được cài đặt giống nhau ở thiết bị truyền và nhận.

**Frame (khung truyền):** Khung truyền quy định về số bit trong mỗi lần truyền.

**Start bit:** là bit đầu tiên được truyền trong 1 frame. Báo hiệu cho thiết bị nhận có một gói dữ liệu sắp được truyền đến. Bit này là bit bắt buộc phải có.

**Data:** dữ liệu cần truyền. Bit có trọng số nhỏ nhất (LSB) được truyền trước, sau đó đến bit có trọng số lớn nhất (MSB).

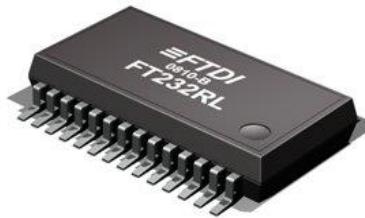
**Parity bit:** là bit dùng để báo hiệu số lượng bit có giá trị bằng 1 (hệ nhị phân) trong một nhóm bit cho trước là một số chẵn hay số lẻ. Sử dụng parity bit là cách phát hiện lỗi đơn giản nhất.

**Stop bit:** là 1 hoặc các bit báo cho thiết bị rằng gói dữ liệu đã gửi xong. Thiết bị nhận sẽ tiến hành kiểm tra khung truyền nhằm đảm bảo tính chính xác của dữ liệu. Bit này là bit bắt buộc phải có.

Các máy tính ngày nay, không hỗ trợ cổng COM, vì thế để máy tính có thể truyền nhận dữ liệu bằng UART, ta cần phải chuyển giao tiếp USB của máy tính sang UART.

Để thiết kế mạch nạp cho ESP8266, ta cần phải sử dụng IC chuyển đổi USB-UART. Trên thị trường hiện nay, có khá nhiều IC hỗ trợ công việc này như: FT232, PL2303, CP2102, CH340G,...

FT232 được đánh giá cao về tính năng lẫn độ ổn định nhưng giá thành lại cao. IC đã được tích hợp hầu hết các linh kiện cần thiết nên việc thiết kế mạch sẽ đơn giản hơn.



Hình 2.10: IC FT232.

CP2102 có tính năng tương đương FT232, giá thành thấp hơn FT232 một chút. Nhưng do cách đóng gói IC ở dạng QFN (chân gầm) nên gây khó khăn trong việc hàn thủ công.



Hình 2.11: IC CP2102.

PL2303 có giá thành tương đối thấp, tuy nhiên khi thiết kế cần phải thiết kế thêm nguồn xung.



Hình 2.12: IC PL-2303.

CH340G có giá thành thấp hơn cả PL2303, IC đóng gói chân dạng SO nên việc hàn thủ công không gặp nhiều khó khăn.



Hình 2.13: IC CH340G.

Nhóm quyết định chọn CH340G vì các đặc tính, yêu cầu phù hợp với đè tài và giá thành thấp.

#### **Đặc điểm của CH340G:**

- Tốc độ của thiết bị phù hợp với đặc điểm của USB phiên bản 2.0.
- Giao diện nối tiếp theo hai chiều, thiết lập bộ thu phát đệm các tín hiệu, hỗ trợ tốc độ truyền thông tin liên lạc khác nhau từ 50bps đến 2Mbps.
- Hỗ trợ tín hiệu MODEM thông thường như RTS DTR, DCD, RI, DSR và CTS.
- Có thể bổ sung thêm thiết bị chuyển đổi như RS232, RS485, RS422,...
- Hỗ trợ hoàn toàn các cổng giao tiếp hồng ngoại IrDA SIR và tốc độ truyền tín hiệu từ 2400 bps đến 11520 bps.
- Thông qua USB chuyển đổi giao tiếp nối tiếp, CH340G chỉ phù hợp ở mức tương đối với các chương trình ứng dụng.
- Hỗ trợ giao tiếp với vi điều khiển ở 2 mức điện áp 5V và 3,3V.

#### **2.2.4 Mạch nguồn**

Để mô hình có thể hoạt động, buộc phải nguồn cấp cho các mạch. Mạch nguồn đóng vai trò quan trọng trong mô hình. Để cấp nguồn cho mô hình, nhóm sử dụng nguồn 220VAC cho qua cầu diode để chỉnh lưu thành nguồn DC, tiếp đến cho qua mạch hạ áp dùng IC LNK3206G để hạ xuống thành 12VDC, sau đó cho 12VDC cho

qua tiếp mạch hạ áp dùng IC MC34063 để hạ xuống còn 3,3VDC để cấp cho mạch cảm ứng điện dung và mạch xử lý trung tâm.

### a. IC nguồn LNK3206G [10]

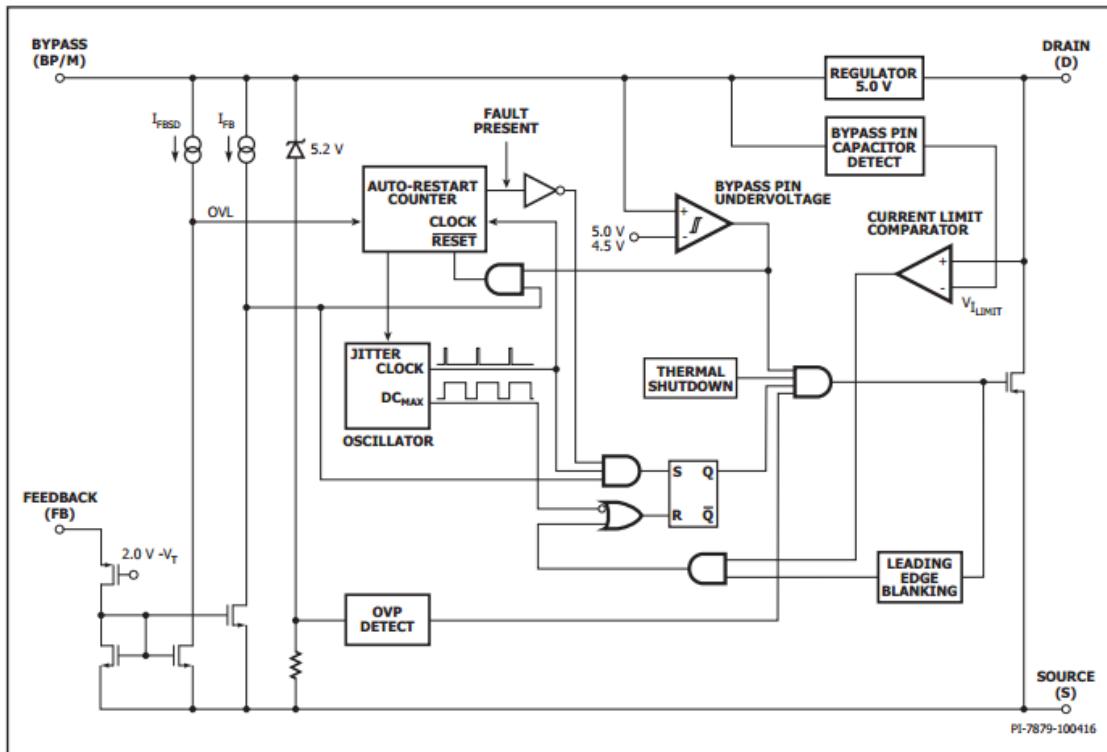


Hình 2.14: IC nguồn LNK3206G.

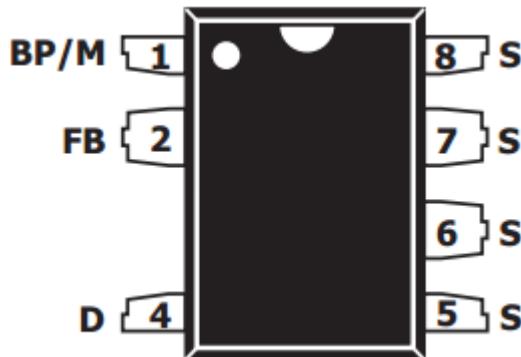
Từ 220VAC qua cầu diode, chỉnh lưu thành nguồn DC để hạ xuống thành 12VDC, thì cần phải sử dụng IC ổn áp có dãy điện áp đầu vào lớn. LNK3206G đáp ứng được các yêu cầu trên.

#### Thông số kỹ thuật:

- Dãy điện áp đầu vào rộng từ 85 ~ 265VDC.
- Dòng ngõ ra tối đa 360mA.
- Tần số đóng cắt 66 kHz.
- Nhiệt độ khi hoạt động: - 40°C ~ 150°C.
- Điện áp đầu ra có thể điều chỉnh, tối là 50V, tối đa hoá liên tục nguồn ra.
- Giảm nhiễu EMI.
- Ứng dụng trong các mạch chuyển đổi DC – DC: buck, buck-boost và flyback.



Hình 2.15: Sơ đồ khái niệm chức năng của LNK3206G.



Hình 2.16: Sơ đồ chân của LNK3206G.

#### Chức năng các chân của LNK3206G:

- Chân D (DRAIN): chân này sẽ nối với nguồn cấp cho MOSFET bên trong IC. Cung cấp dòng để khởi động và ổn định trạng thái khi hoạt động.
- Chân BP/M (BYPASS): chân BP/M có nhiều chức năng, chân này nối với một tụ bypass để bên trong IC tạo ra một nguồn 5V. Chân này có thể quyết định giới hạn dòng ngõ ra, giới hạn của dòng phụ thuộc vào điện dung của tụ được thêm vào. Chân BP/M còn có chức năng bảo vệ quá áp đầu ra với các mạch điện bên ngoài.

- Chân FB (FEEDBACK): trong quá trình hoạt động, chuyển mạch của MOSFET được điều khiển bởi chân FB, việc chuyển mạch bị chấm dứt khi có một dòng điện lớn hơn  $I_{FB}$  ( $49\mu A$ ) được đưa vào chân FB.
- Chân S (SOURCE): chân này sẽ nối với nguồn cấp cho MOSFET bên trong IC, đây cũng là chân mass cho các chân BP/M và chân FB.

### b. IC nguồn MC34063 [11]

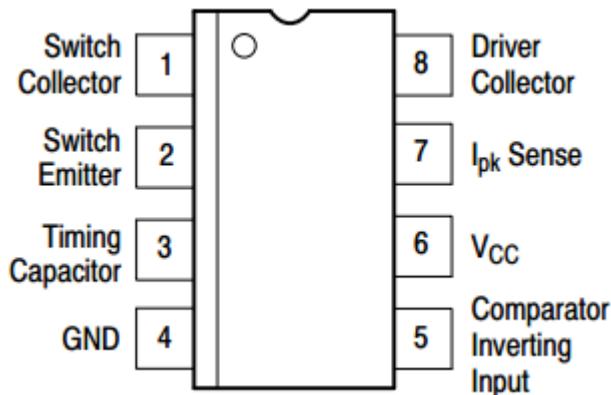
MC34063 là IC nguồn điều khiển PWM đa năng với dòng ra trực tiếp có thể lên tới 1,5A. Với cấu trúc của MC34063, có thể thiết kế thành các mạch nguồn DC-DC không cách ly thông dụng như: mạch buck, mạch boost.



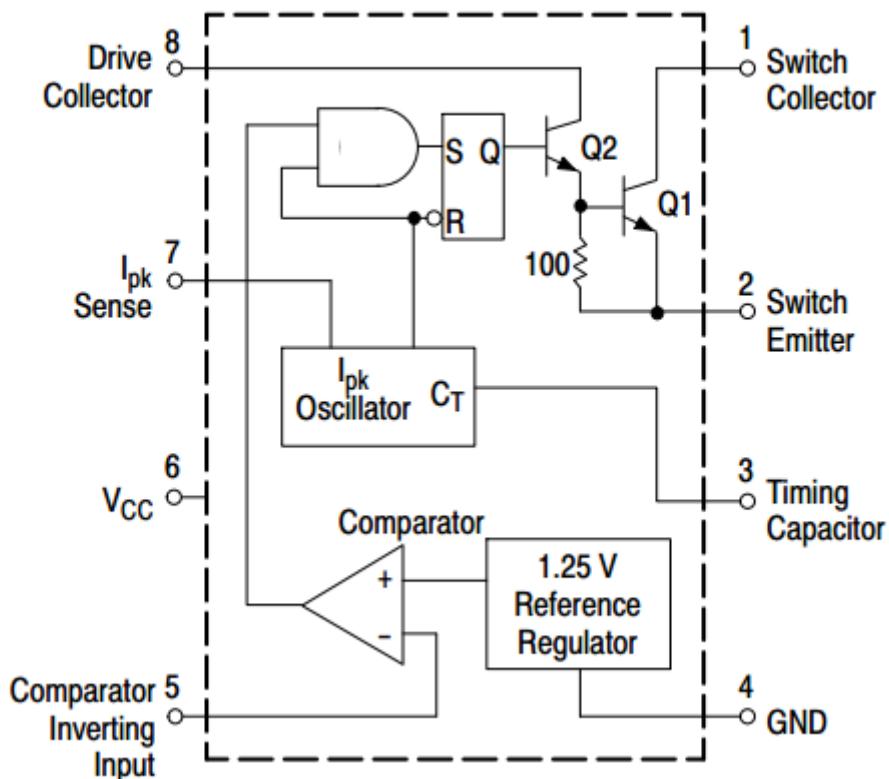
Hình 2.17: IC MC34063.

#### Thông số kỹ thuật:

- Điện áp đầu vào từ 3 ~ 40V.
- Điện áp đầu ra từ 1,25 ~ 40V.
- Tần số đóng cắt từ 80 ~ 100 kHz.
- Dòng ngõ ra tối đa 1,5A.



Hình 2.18: Sơ đồ chân của MC34063.



Hình 2.19: Sơ đồ khối chức năng của MC34063.

Cấu trúc của MC34063 gồm:

Khối tạo xung vuông.

Khối tạo dao động.

Khối so sánh điện áp với mức áp mẫu 1,25V.

Với các đặc điểm trên, phù hợp với yêu cầu của đè tài, nên nhóm quyết định sử dụng MC34063 để xây dựng mạch nguồn cho các khối trong mô hình.

## 2.3 GIỚI THIỆU VỀ PHẦN MỀM

Khi người dùng chạm vào bề mặt phím cảm ứng điện dung của mô hình, tín hiệu điều khiển được đưa tới bộ xử lý trung tâm, sau đó bộ xử lý trung tâm gửi tín hiệu điều khiển đến mạch công suất, cho phép đóng tắt thiết bị tương ứng. Tiếp theo gửi dữ liệu chứa trạng thái của thiết bị lên server thông qua Internet, cụ thể là mạng wifi. Server sẽ tiếp nhận dữ liệu vừa gửi tới, kế đến tiến hành xử lý và lưu dữ liệu vào cơ sở dữ liệu. Cập nhật trạng thái hiện tại của thiết bị trên web. Khi người dùng tác động vào web, server sẽ gửi dữ liệu điều khiển về bộ xử lý trung tâm. Bộ xử lý trung tâm nhận dữ liệu, sau đó tiến hành tác động để đóng mở thiết bị tương ứng và đồng thời server sẽ cập nhật trạng thái của thiết bị vừa được tác động. Để có thể lưu được trạng thái của thiết bị trên web, ta cần xây dựng web server, cơ sở dữ liệu và hiểu được giao thức truyền dữ liệu từ bộ xử lý trung tâm lên web server.

### 2.3.1 Web server

#### a. Khái niệm [12]

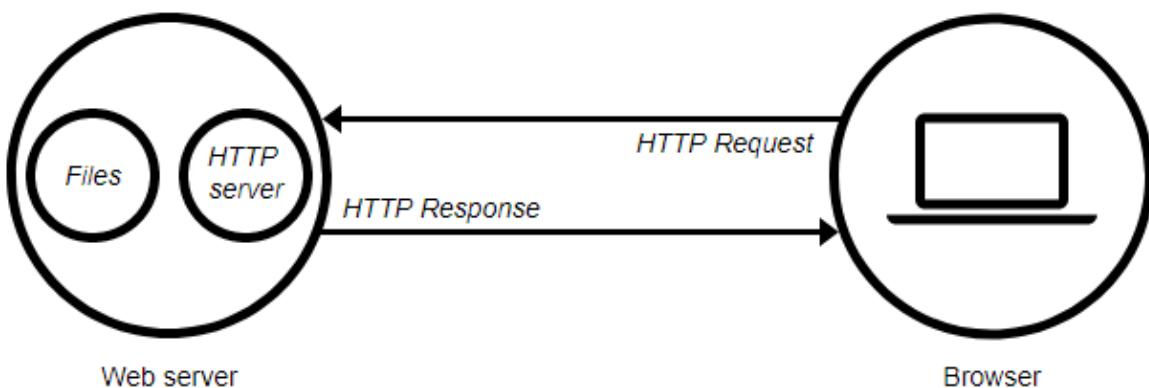
Web server có thể là phần cứng hoặc phần mềm, hoặc cả hai.

Ở khía cạnh phần cứng, web server là một máy tính lưu trữ các file thành phần của một website (các tài liệu, hình ảnh,...) và có thể phân phát chúng tới thiết bị của người dùng. Web server kết nối tới Internet và có thể truy cập tới thông qua một tên miền.

Ở khía cạnh phần mềm, web server điều khiển người sử dụng web truy cập tới các file được lưu trữ trên một HTTP server (máy chủ HTTP). HTTP server là một phần mềm hiểu được các địa chỉ web (URL) và giao thức trình duyệt web (HTTP).

#### b. Cách giao tiếp với Web server [12]

Khi một trình duyệt cần một file lưu trữ trên một web server, trình duyệt sẽ yêu cầu (request) file đó thông qua HTTP. Khi một yêu cầu gửi tới đúng web server (phần cứng), HTTP server (phần mềm) sẽ gửi file được yêu cầu cũng thông qua HTTP.



Hình 2.20: Cách thức giao tiếp với web server.

Web server hỗ trợ giao thức HTTP (Giao thức truyền phát siêu văn bản). HTTP là cách truyền các siêu văn bản giữa hai máy tính.

HTTP cung cấp các quy tắc rõ ràng, về cách client và server giao tiếp với nhau:

Chỉ client có thể tạo ra các HTTP request tới các server. Các server chỉ có thể phản hồi HTTP request của client.

Khi yêu cầu một file thông qua HTTP, client phải cung cấp URL của file đó.

Web server phải trả lời mọi HTTP request.

Trên web server, HTTP server chịu trách nhiệm xử lý và trả lời các request đã được client gửi đến:

- Khi nhận một request, HTTP server sẽ kiểm tra xem URL được yêu cầu có khớp với một file hiện có không.
- Nếu có, web server gửi nội dung file trả lại client. Nếu không, một application server sẽ tạo ra file cần thiết.
- Nếu không thể xử lý, web server trả lại một thông điệp lỗi cho client.

### 2.3.2 Cơ sở dữ liệu

#### a. Khái niệm [13]

Cơ sở dữ liệu là một hệ thống các thông tin có cấu trúc, được lưu trữ trên các thiết bị lưu trữ nhằm thỏa mãn yêu cầu khai thác thông tin đồng thời của nhiều người sử dụng hay nhiều chương trình ứng dụng chạy cùng lúc với những mục đích khác nhau.

**b. Phân loại cơ sở dữ liệu [14]**

Cơ sở dữ liệu dạng file: dữ liệu được lưu trữ dưới dạng các file có thể là text, ascii,...

Cơ sở dữ liệu quan hệ: dữ liệu được lưu trữ trong các bảng dữ liệu gọi là các thực thể, giữa các thực thể này có mối liên hệ với nhau. Các hệ quản trị hỗ trợ cơ sở dữ liệu quan hệ như: MS SQL server, Oracle, MySQL,...

Cơ sở dữ liệu hướng đối tượng: dữ liệu cũng được lưu trữ trong các bảng dữ liệu nhưng các bảng có bổ sung thêm các tính năng hướng đối tượng như lưu trữ thêm các hành vi, nhằm thể hiện hành vi của đối tượng. Các hệ quản trị có hỗ trợ cơ sở dữ liệu hướng đối tượng: MS SQL server, Oracle, Postgres,...

Cơ sở dữ liệu bán cấu trúc: dữ liệu được lưu dưới dạng XML, với định dạng này thông tin mô tả về đối tượng thể hiện trong các tag. Cơ sở dữ liệu này có nhiều ưu điểm do lưu trữ được hầu hết các loại dữ liệu khác nhau nên cơ sở dữ liệu bán cấu trúc là hướng mới trong nghiên cứu và ứng dụng.

## Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

### 3.1 GIỚI THIỆU

Mô hình với kích thước nhỏ gọn là mục tiêu mà nhóm hướng đến. Vì vậy nhóm sẽ thiết kế mô hình thành 3 bản PCB được kết nối với nhau thông qua rào cǎm, bao gồm các khối sau:

Khối nguồn sử dụng nguồn 220VAC để cấp cho thiết bị điện được điều khiển thông qua mô hình, nguồn 220VAC sẽ được chỉnh lưu bằng cầu diode thành nguồn DC. Và được hạ áp xuống thành 12VDC thông qua IC LNK3206G, tiếp tục cho qua mạch hạ áp sử dụng IC MC34063, hạ xuống thành 3,3VDC cấp cho mạch xử lý trung tâm và mạch cảm ứng điện dung.

Khối công suất sử dụng opto MOC3020 và triac BTA12 để điều khiển 4 thiết bị sử dụng nguồn xoay chiều 220VAC.

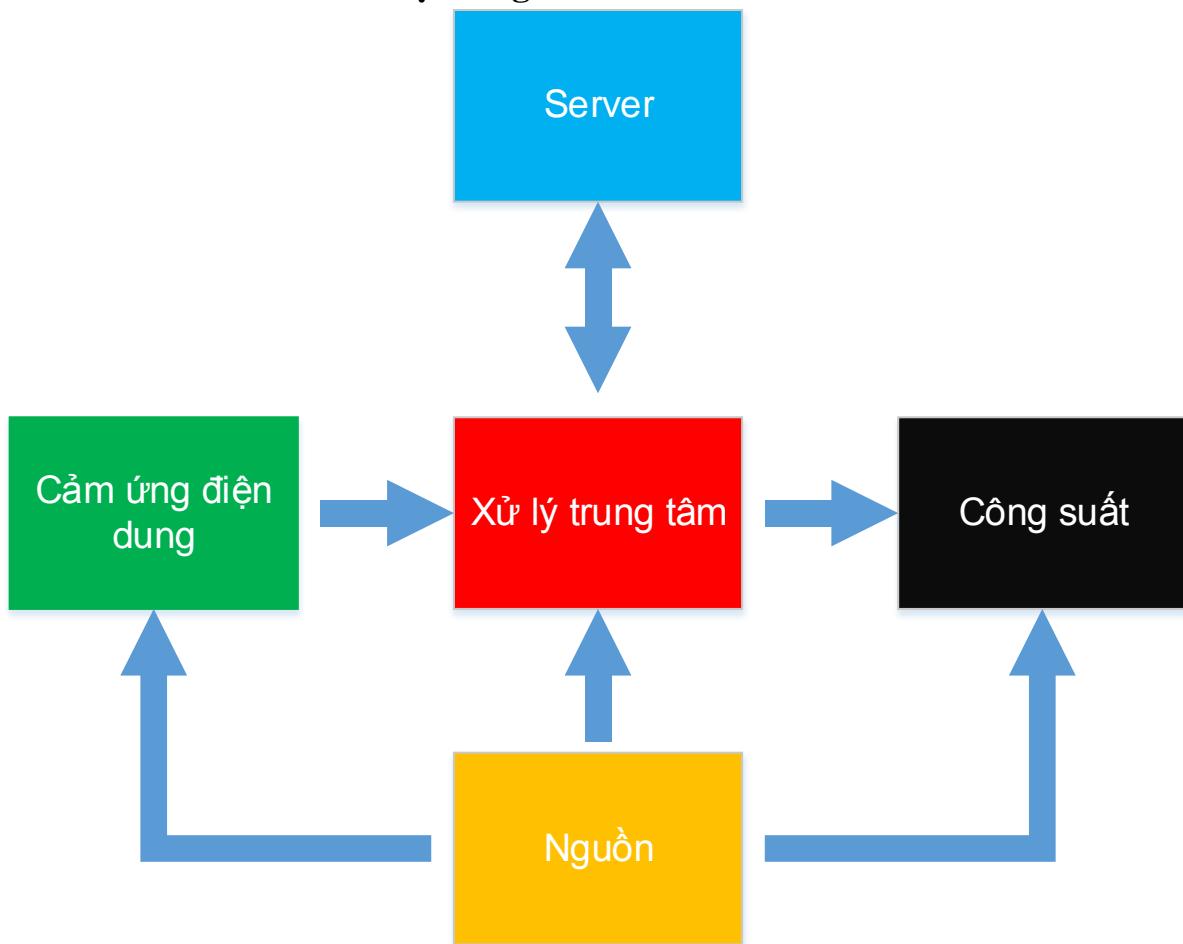
Khối cảm ứng điện dung sử dụng IC AT42QT2120 với 4 phím cảm ứng, để điều khiển 4 thiết bị.

Khối xử lý trung tâm, sử dụng ESP8266 module ESP-12F, đóng vai trò là thiết bị kết nối với wifi truyền nhận dữ liệu từ server. Thiết kế mạch nạp cho ESP8266 sử dụng IC CH340G.

Một PCB sử dụng IC CH340G để làm mạch nạp cho ESP8266. Một PCB tích hợp khối xử lý trung tâm và khối cảm biến điện dung. Một PCB sử dụng điện áp 220VAC, tích hợp khối nguồn và khối công suất.

### 3.2 TÍNH TOÁN VÀ THIẾT KẾ HỆ THỐNG

#### 3.2.1 Thiết kế sơ đồ khái hệ thống



Hình 3.1: Sơ đồ khái của hệ thống.

**Khối cảm ứng điện dung:** khi người dùng chạm vào phím cảm ứng, tín hiệu tác động sẽ gửi sang khối xử lý trung tâm.

**Khối xử lý trung tâm:** nhận tín hiệu tác động từ khối cảm ứng điện dung, xử lý sau đó xuất tín hiệu điều khiển thiết bị cho khối công suất thực thi, kể đến gửi dữ liệu lên khối server. Khối xử lý trung tâm có thể nhận dữ liệu yêu cầu điều khiển thiết bị từ server, sau đó gửi tín hiệu đến khối công suất. Để điều khiển thiết bị mà server yêu cầu thông qua sự tác động của người dùng.

**Khối công suất:** nhận tín hiệu từ khối xử lý trung tâm, để điều khiển thiết bị.

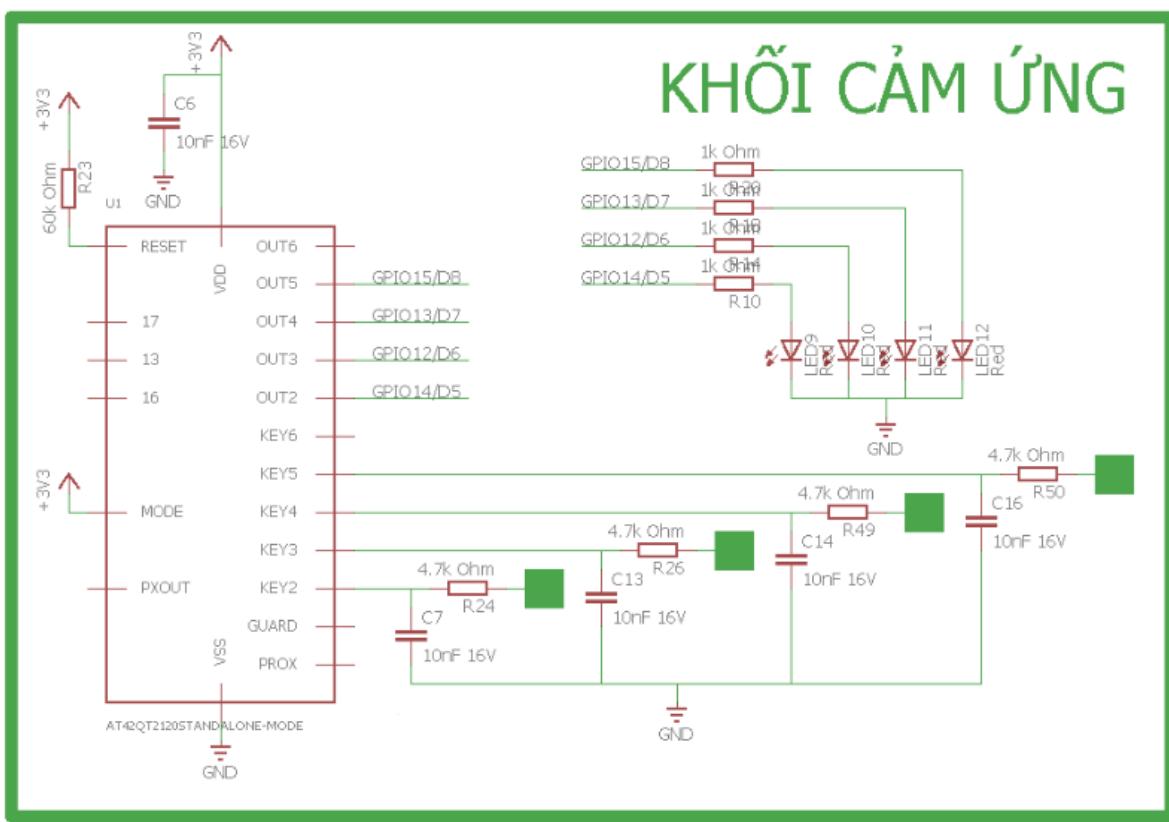
**Khối server:** nhận dữ liệu điều khiển từ khối xử lý trung tâm, tiến hành lưu trạng thái của thiết bị vào cơ sở dữ liệu. Để người dùng có thể truy cập, biết được lịch sử đóng/tắt của thiết bị. Người dùng có thể sử dụng web server, tác động và điều khiển thiết bị theo như mong muốn. Khi tác động thì server sẽ tiến hành gửi dữ liệu điều

khiến xuống khỏi xử lý trung tâm, khỏi xử lý trung tâm nhận được dữ liệu sẽ truyền dữ liệu điều khiển thiết bị tương ứng sang khối công suất để tiến hành đóng tắt thiết bị.

### 3.2.2 Tính toán và thiết kế mạch

#### a. Thiết kế khối cảm ứng điện dung

Sử dụng IC AT42QT2120, khi người dùng chạm vào 4 lớp đồng kết nối với 4 chân KEY, từ KEY2 đến KEY5. 4 chân này sẽ nhận tín hiệu và sẽ xuất ra các tín hiệu tương ứng với 4 chân OUT, từ OUT2 đến OUT5, tín hiệu của các chân này sẽ đưa đến 4 chân GPIO của ESP8266.



Hình 3.2: Sơ đồ nguyên lý của khối cảm ứng điện dung.

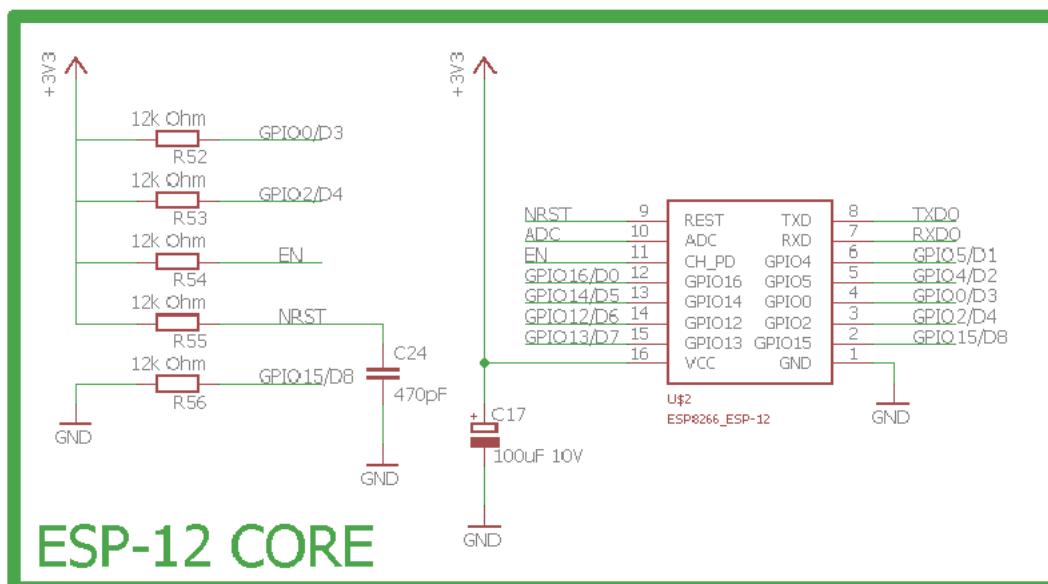
Điện trở  $R_{23} = 60\text{k}\Omega$  được nối chân reset và đưa lên nguồn, để IC cảm ứng điện dung có thể hoạt động mà không bị reset, giá trị của trở  $R_{23}$  được sử dụng theo yêu cầu của nhà sản xuất. Tụ  $C_6 = 100\text{nF}$ , dùng để lọc nhiễu nguồn cấp cho IC cảm ứng điện dung, giá trị của tụ  $C_6$  được tham khảo yêu cầu của nhà sản xuất. Các led D9, D10, D11, D12 dùng để báo hiệu khi người dùng chạm vào nút cảm ứng điện dung, led sẽ sáng giữ trong giây lát sau khi người dùng chạm vào. Các trở  $R_{10}, R_{14}, R_{18}, R_{20}$  dùng để hạn dòng cho các led D9, D10, D11, D12. Các trở  $R_{24}, R_{26}, R_{49}, R_{50}$

được nhà sản xuất yêu cầu sử dụng với giá trị từ  $4,7\text{k}\Omega \sim 20\text{k}\Omega$  với mục đích để chống nhiễu cho các nút cảm ứng, ở đây nhóm sử dụng các điện trở trên với giá trị là  $4,7\text{k}\Omega$ . Các tụ C7, C13, C14, C16 các tác dụng điều chỉnh độ nhạy của nút cảm ứng. Qua quá trình thử nghiệm, với giá trị của các tụ trên là  $10\text{nF}$ , sẽ tạo ra độ nhạy phù hợp khi sử dụng.

### b. Khối xử lý trung tâm

Khối xử lý trung tâm sử dụng ESP8266 phiên bản ESP-12F. Khối này được chia nhỏ thành các mạch sau:

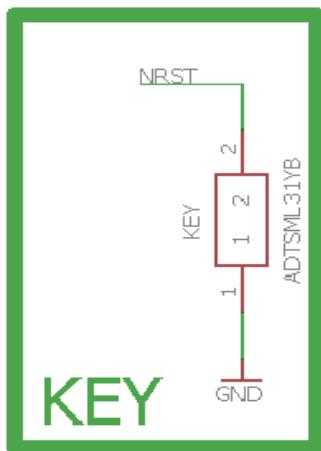
Mạch ESP-12: ESP8266 làm bộ xử lý trung tâm, với 1 tụ lọc nguồn 1000uF trước ngõ vào giúp mạch hoạt động ổn định hơn. Khi ESP khởi động, dòng khởi động có thể lên tới  $I_{max} = 300mA$ , do đó tụ còn có tác dụng là tránh sốc điện lúc ESP khởi động. Tụ 470pF dùng để chống nhiễu cho nút reset. Một số chân GPIO được nối trở kéo lên hoặc kéo xuống là để giúp ESP hoạt động ổn định.



Hình 3.3: Sơ đồ khối của mạch ESP-12.

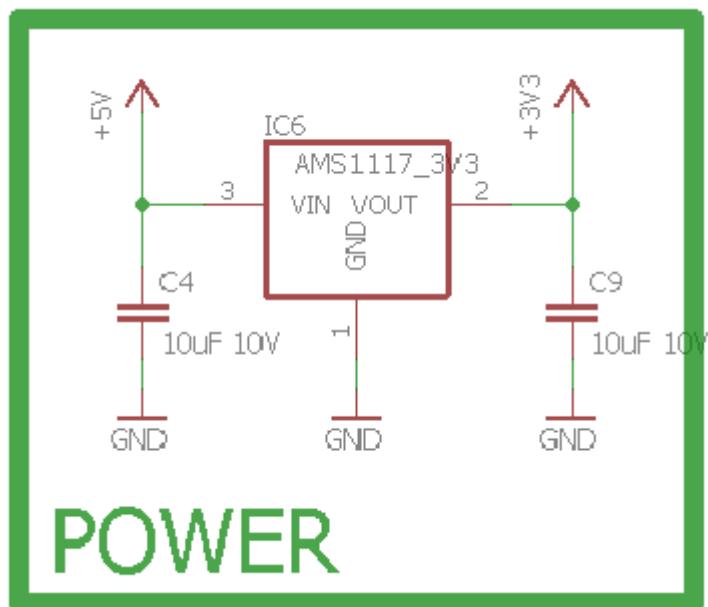
Các giá trị thông số của các điện trở và tụ trên mạch ESP-12, được tham khảo theo sơ đồ nguyên lý của module “NodeMCU ESP12” [1i].

Mạch reset dùng để reset toàn mạch.



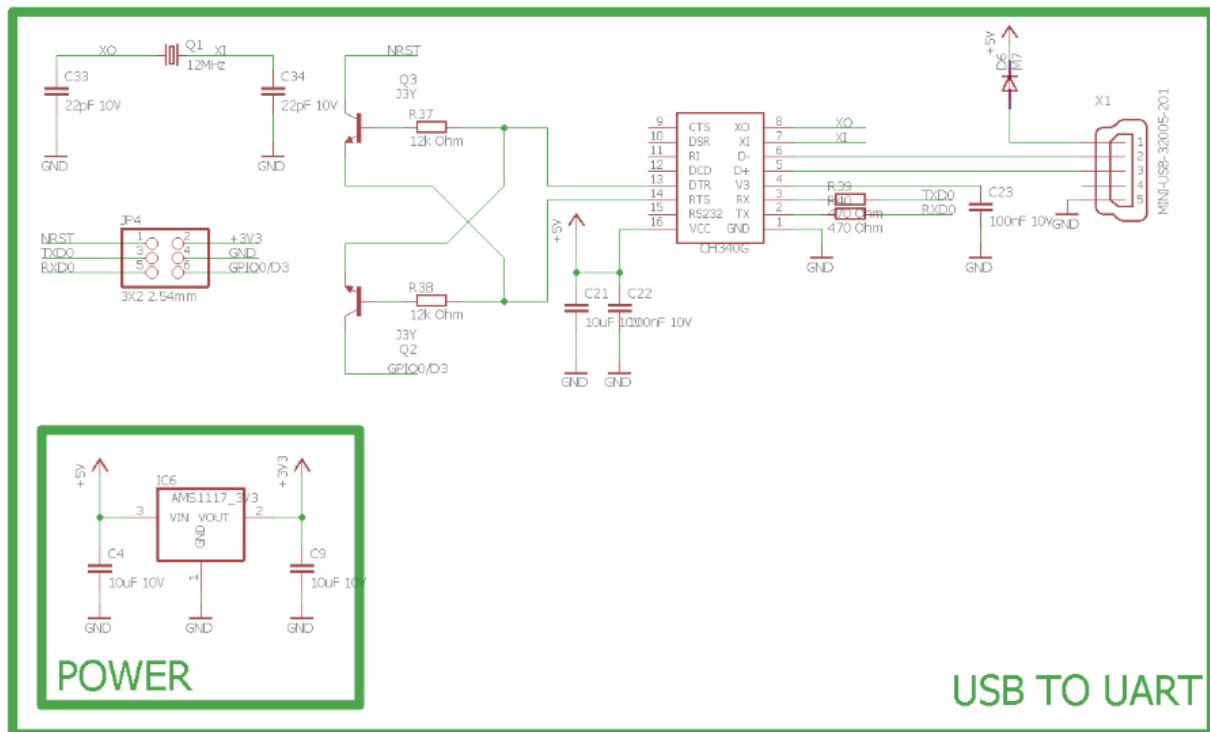
Hình 3.4: Sơ đồ nguyên lý của mạch reset.

Mạch nguồn cho ESP khi ESP sử dụng nguồn từ cổng USB máy tính: ESP hoạt động ở mức điện áp 3,3V (dãy điện áp có thể hoạt động là từ 2,5 ~ 3,6VDC). Vì vậy cần có mạch hạ áp từ nguồn 5V (của cổng USB) về 3,3V sử dụng IC nguồn AMS1117- 3,3V. Mạch sử dụng 2 tụ 10uF dùng để lọc ở đầu vào và cả đầu ra, giúp ESP hoạt động ổn định. Mục đích của mạch nguồn này, là dùng để cấp nguồn cho ESP8266 khi ta tiến hành nạp chương trình từ máy tính cho ESP thông qua cổng USB.



Hình 3.5: Sơ đồ nguyên lý của mạch nguồn sử dụng nguồn từ cổng USB.

Mạch nạp cho ESP từ cổng USB: mạch này được xây dựng dùng để nạp chương cho ESP thông qua cổng USB của máy tính. Chế độ nạp được sử dụng ở mạch này là chế độ nạp thông qua UART.



Hình 3.6: Sơ đồ nguyên lý mạch nạp cho ESP.

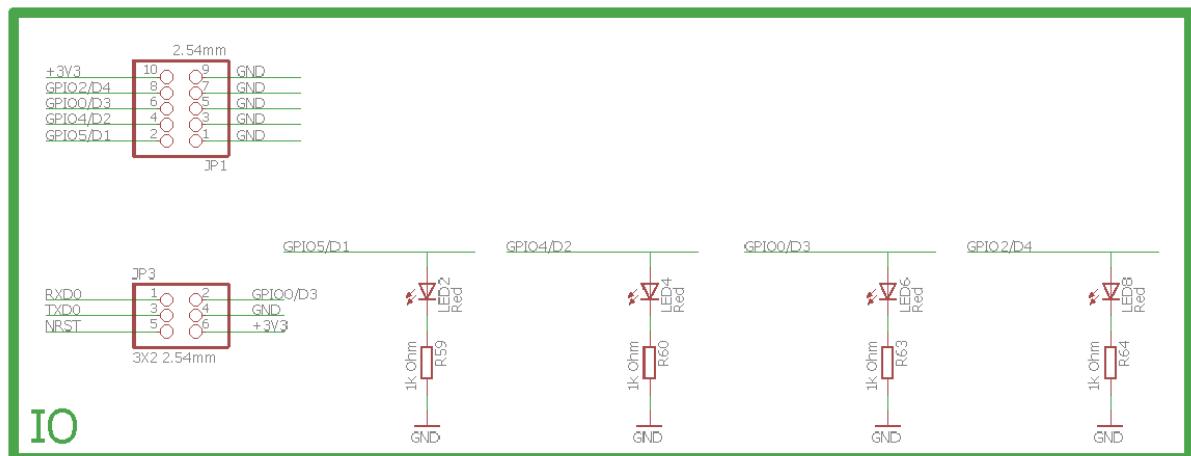
Mạch nạp sử dụng CH340G, sử dụng nguồn 5V từ cổng USB của máy tính, 2 tụ C21, C22 có tác dụng lọc nhiễu cho CH340G, thạch anh 12MHz dùng để cấp xung clock cho CH340G. Kết nối các chân DTR, RTS của CH340G với transistor J3Y để tạo nên mạch tự động chuyển chế độ cho ESP giữa trạng thái chạy chương trình chính và trạng thái nạp chương trình cho ESP.

Bảng 3.1: Trạng thái chuyển mạch tự động

<b>DTR</b>	<b>RTS</b>	<b>RST</b>	<b>GPIO0</b>
1	1	1	1
0	0	1	1
1	0	0	1
0	1	1	0

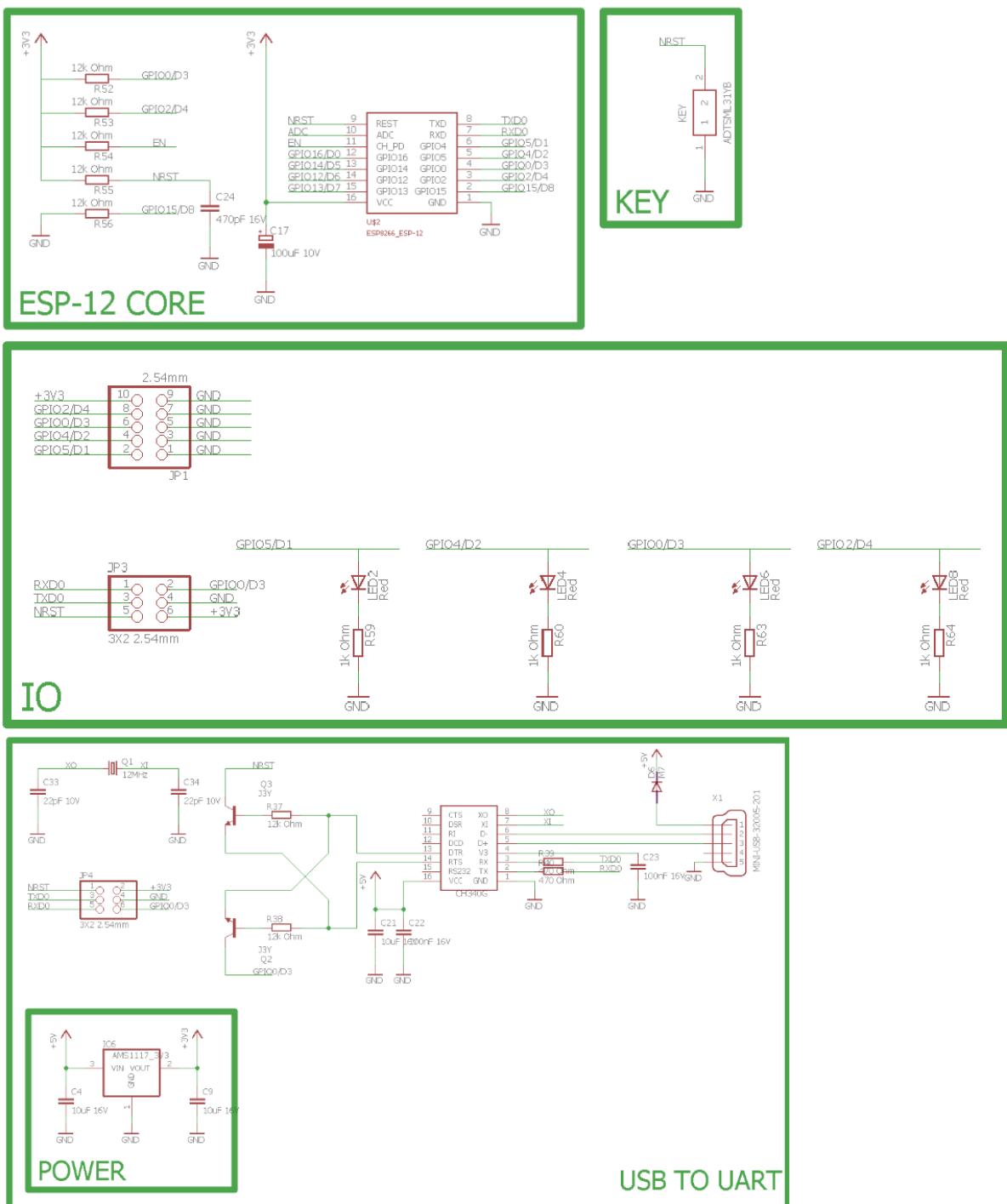
Các giá trị thông số của linh kiện trên mạch nạp cho ESP, được tham khảo theo sơ đồ nguyên lý của module “NodeMCU ESP12” [1i].

Mạch điều khiển: các chân GPIO dùng để xuất các tín hiệu, các led đơn dùng để hiển thị trạng thái cho các chân output dùng làm tín hiệu điều khiển cho mạch công suất. Các trở R59, R60, R63, R64 dùng để hạn dòng cho các led D2, D4, D6, D8.



Hình 3.7: Sơ đồ nguyên lý mạch điều khiển GPIO.

## Sơ đồ của khối xử lý trung tâm



Hình 3.8: Sơ đồ nguyên lý của khối xử lý trung tâm.

*c. Thiết kế khói công suất*

Khói công suất nhận tín hiệu điều khiển từ khói xử lý trung tâm, để điều khiển các thiết bị điện trong nhà. Mạch dùng opto kết hợp với triac công suất. Để lựa chọn linh kiện phù hợp, ta tiến hành khảo sát công suất tiêu thụ các thiết bị trong gia đình

Bảng 3.2: Liệt kê công suất tiêu thụ của một số thiết bị điện gia dụng

STT	Tên thiết bị	Công suất thiết bị
1	Tivi LED Sony 32 inches	69W
2	Tủ lạnh Panasonic NR-BJ176 152 lít	97 - 130W
3	Máy giặt Toshiba AW-E920LV 8,2Kg	410W
4	Bóng đèn huỳnh quang 1,2m	36W
5	Bóng đèn huỳnh quang 0,6m	18W
6	Sạc Laptop	65-85W
7	Màn hình vi tính LCD-17 inches	35W
8	Quạt	55W
9	Máy lạnh 1,5 HP	1200W
10	Quạt thông gió	25W
11	Điện thoại bàn	9W
12	Thiết bị mạng modem	10W
13	Máy tính xách tay	160W
14	Cục sạc iPhone	5W
15	Quạt bàn FUNA	67W
16	Cục sạc Laptop DELL	65W
17	Loa SoundMax 2.1	18W
18	Loa Karaoke JBL RM 10 II	50-400W
19	Đầu DVD ARIRANG AR3600	30W
20	Ampli FUJIYAMA DA-3600N	240W
21	Quạt đứng Asia Vina D16011	55W
22	Nồi cơm SHARP KSH-317V	600W
23	Đèn sợi đốt Rạng Đông	25-60W

24	Camera hồng ngoại	15W
----	-------------------	-----

Dựa vào bảng trên ta có thể thấy máy lạnh là thiết bị có công suất cao nhất 1200W. Với điện áp sử dụng trong hộ gia đình từ lưới điện 220VAC. Ta có công thức tính dòng hiệu dụng như sau:

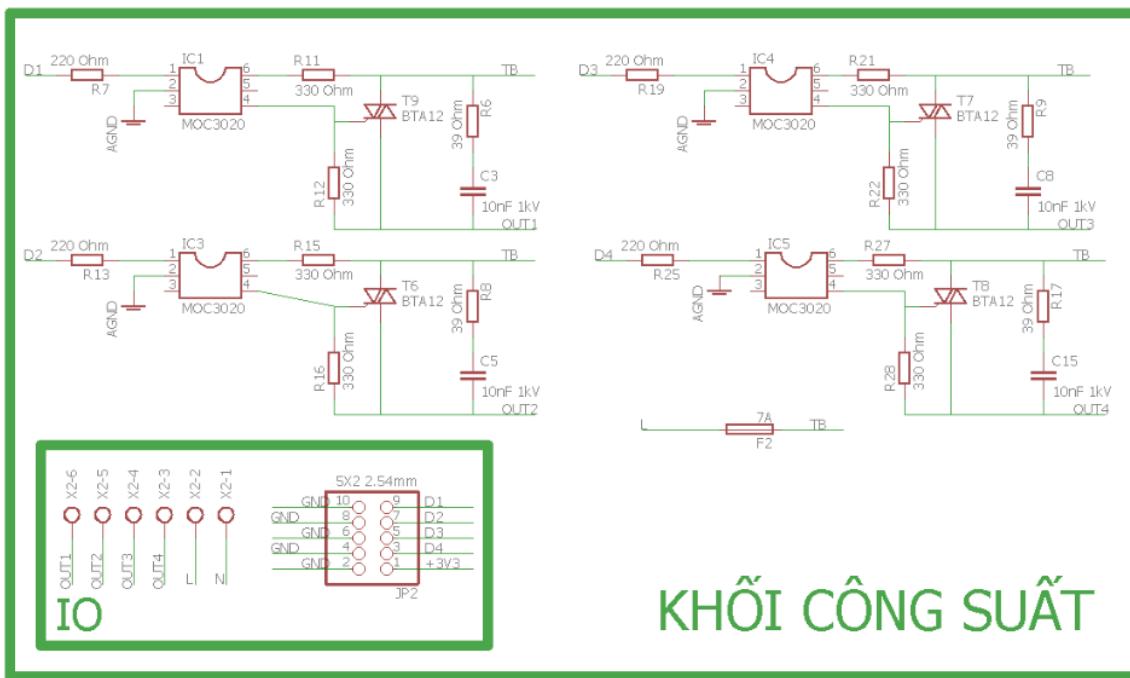
$$I_{hd} = P/U_{hd} = 1200/220 = 5,45A.$$

Như vậy dòng tải thiết bị cao nhất gần 6A, lựa chọn triac có dòng định mức lớn hơn 6A, điện áp ngược chịu trên 220VAC là phù hợp. Opto có áp đầu ra bên phần detector cần lớn hơn 220VAC, cần quan tâm tới áp và dòng kích của opto. Do khói xử lý trung dùng ESP-12F, điện áp các chân ngõ ra khoảng 3,3V cùng với dòng xuất ra mỗi chân chỉ tối đa là 12mA.

#### Lựa chọn triac cần chú ý tới các thông số:

- Dòng điện định mức ( $I_{st} > 1,25I_{td}$ ).
- Điện áp ngược lớn nhất mà triac chịu được khi ngưng dẫn.
- Dòng điều khiển  $I_g$  (mA).
- Nhiệt độ hoạt động.

Triac BTA12 có dòng định mức là 12A, điện áp ngược chịu được là 600V. Opto MOC3020 có áp đầu ra chịu được tối đa là 400V, điện áp để kích opto tối thiểu là 3V, dòng kích tối thiểu là 10mA. Như vậy triac BTA12 và opto MOC3020 đáp ứng được yêu cầu của đề tài.



## KHÔI CÔNG SUẤT

Hình 3.9: Sơ đồ nguyên lý của khối công suất.

Các trở R7, R13, R19, R25 dùng để hạn dòng cho photodiode của MOC3020.

Điện áp của các chân output của ESP8266 là 3,3V, lựa chọn trở hạn dòng là photodiode là  $220\Omega$ . Nên dòng kích cho opto là  $I_{FT} = 3,3/220 = 15mA$ , thoả yêu cầu để kích cho opto.

Các trở R11, R15, R21, R27 để hạn dòng vào triac bên trong MOC3020. Dòng vào tối đa của triac bên trong MOC3020 là 1A, lựa chọn trở hạn dòng vào là  $330\Omega$ . Điện áp đặt vào sẽ là điện áp của thiết bị cần điều khiển là 220VAC. Nên dòng vào triac bên trong MOC3020 là  $220/330 \approx 667mA$ , thoả yêu cầu mà nhà sản xuất đặt ra.

Mạch RC (R6 và C3, R8 và C5, R9 và C8, R17 và C15) có tác dụng chống nhiễu, chống hiện tượng tự kích với tải cảm, tải có công suất lớn cũng như nguồn đang sử dụng có tải cảm tham gia. Giá trị điện trở và điện dung của mạch RC được tham khảo theo datasheet của MOC3020 [2i].

Các trở R12, R16, R22, R28 dùng để tạo xung kích chuẩn cho triac BTA12.

L và N là đầu dây 220VAC đưa vào công tắc, OUT1, OUT2, OUT3, OUT4 kết nối với các thiết bị sử dụng nguồn 220VAC. Các chân D1, D2, D3, D4 nhận tín hiệu từ các chân GPIO của ESP.

Trên thực tế, mạch công suất chỉ hoạt động ổn định với công suất 1000W trở xuống, xấp xỉ 220VAC, dòng < 5A. Do kích thước, bố trí cũng như khả năng tản nhiệt mạch.

#### d. Thiết kế khối nguồn

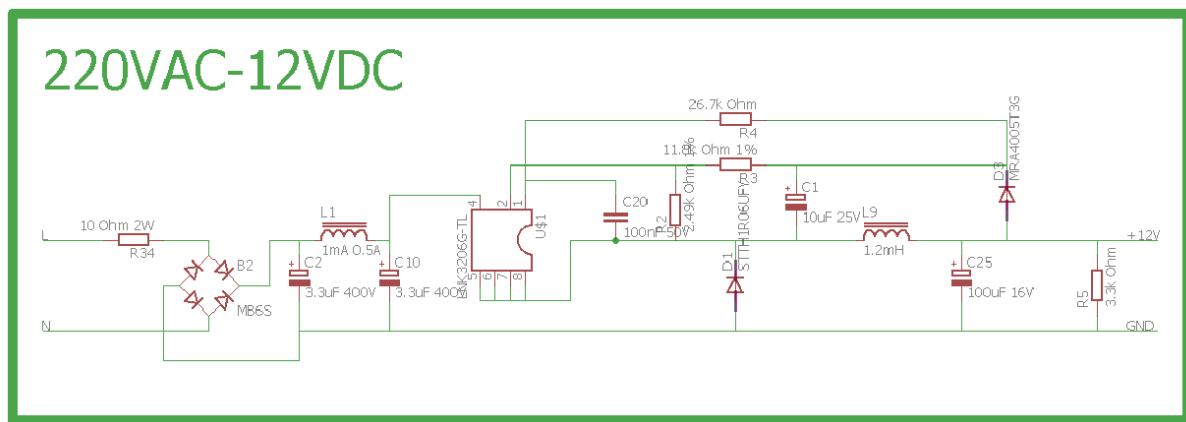
Mô hình sử dụng nguồn 220VAC, mạch nguồn cần được chuyển về điện áp DC để cấp cho các mạch điện khác trong mô hình hoạt động.

Bảng 3.3: Thông số điện áp và dòng tiêu thụ của các linh kiện trong mô hình

Linh kiện	VCC (V)	I <sub>dm</sub> (mA)
ESP8266	3,3	300
IC AT42QT2120	3,3	0,55mA
4 Led nối với 4 nút cảm ứng	3,3	80mA

Tổng dòng tiêu thụ sơ bộ là 381mA. Các linh kiện sử dụng điện áp 3,3V, vì vậy ta cần phải thiết kế bộ nguồn 220VAC hạ áp thành 3,3V – 500mA là phù hợp. Phương án thiết kế nguồn của nhóm như sau:

Điện áp 220VAC sẽ được chỉnh lưu thành nguồn DC, rồi được hạ xuống thành 12VDC thông qua IC LNK3206G. Theo như tài liệu hướng dẫn thiết kế của nhà sản xuất [3i], khi dùng LNK3206G xây dựng mạch hạ áp xuống thành 12VDC thì dòng ngõ ra của mạch tối đa là 225mA.

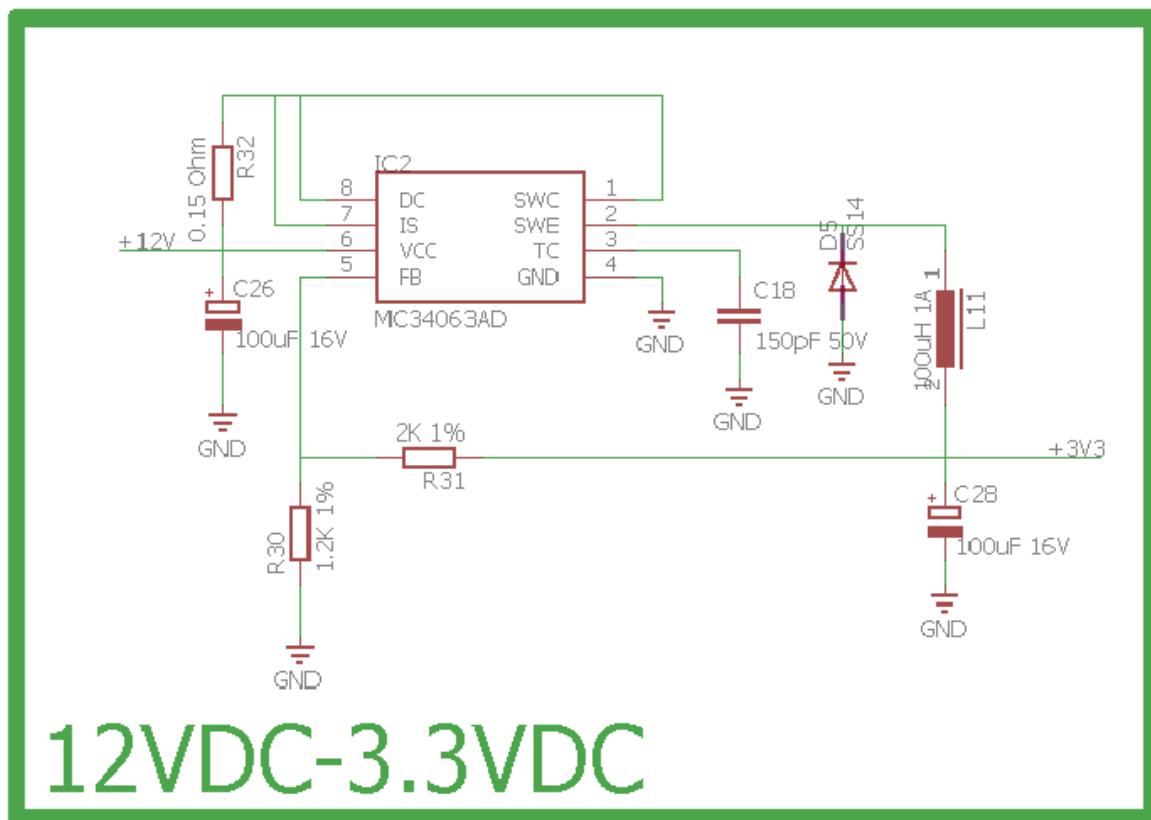


Hình 3.10: Sơ đồ nguyên lý của mạch 220VAC – 12VDC.

Công suất của mạch được thiết kế sẽ là  $P = 12 \times 0,225 = 2,7\text{W}$ . Với  $P > 1\text{W}$ , theo tài liệu của nhà sản xuất, giá trị của R34 được khuyên dùng là  $8,2\Omega$ , 1W. Lựa chọn giá trị R34 là  $10\Omega$ , 2W là tương đối đạt yêu cầu. R34 sẽ đóng vai trò bảo vệ mạch. Cầu diode MB6S dùng để chỉnh lưu nguồn 220VAC thành nguồn AC, điện áp tối đa đặt

vào MB6S là 600VAC. Nên việc chọn MB6S là phù hợp. Cuộn cảm L1 và 2 tụ C2, C10 sẽ tạo thành bộ lọc nhiễu EMI, giá trị của L1 được khuyến nghị là từ 470uH ~ 2,2mH, C2 và C1 là từ 2uF trở lên. Chọn L1 = 1mH, C2 = C1 = 3,3uF là hợp lý. Tụ C20 dùng với giá trị 100nF, để dòng ngõ ra đạt tối đa có thể. Diode D1 phải là diode có thời gian phục hồi ngược nhanh, theo yêu cầu của nhà sản xuất thì thời gian phục  $t_{RR} \leq 35\text{ns}$ , với diode D1 là diode STTH1R06,  $t_{RRmax} = 25\text{ns}$ , đáp ứng đúng yêu cầu của nhà sản xuất. Tụ C25 có chức năng hạn chế điện áp gợn đầu ra. Điện trở R4 dùng để giảm công suất đầu vào không tải và tăng hiệu suất với điều kiện có tải. Điện áp rơi trên D1 và D3 là như nhau. Cuộn cảm L9 đóng vai trò không thể thiếu trong mạch buck. Điện áp trên C1 được quyết định bởi cặp trở R2 và R6, các giá trị của R2 và R6 được chọn sao cho điện áp đầu ra tại chân FB là 2V. Điện trở R5 sẽ đóng vai trò tải trong trường hợp chưa có tải nối với các chân điện áp ngõ ra 12V.

Điện áp 12VDC sẽ được hạ áp xuống thành 3,3VDC thông qua IC MC34063. Nguồn 3,3VDC sẽ được cấp cho các mạch điện trong mô hình hoạt động.



Hình 3.11: Sơ đồ nguyên lý mạch 12VDC-3,3VDC.

Tụ C26 = 100uF dùng để lọc nguồn đầu vào cấp cho MC34063, điện trở R32 = 0,15Ω để giới hạn dòng điện tải ở ngõ ra. C18 = 150pF dùng để quyết định tần số của mạch dao động. Cuộn cảm L11 dùng để ổn dòng. Diode D5 dùng để lấy dòng chúa trong cuộn dây L cấp cho tải khi xung ở chu ngắt. Tụ C18, cuộn dây L11 và tụ C28 = 100uF tăng mức ổn định cho điện áp ngõ ra. Tín hiệu hồi tiếp trả về qua cầu phân áp R30 và R31. Điện trở R31, R30 có tác dụng điều chỉnh điện áp ngõ ra.

$$V_o = 1,25 \times (1 + R31/R30) \approx 3,3V.$$

## Chương 4. THI CÔNG HỆ THỐNG

### 4.1 GIỚI THIỆU

Mô hình chia làm 2 phần thi công: phần cứng và phần mềm.

#### Phần cứng gồm:

Thi công 2 PCB kích thước 40mm x 69mm, 1 PCB kích thước 40mm x 20mm. Mỗi PCB đều được gia công 2 lớp (Top và Bottom). Sắp xếp linh kiện và đi dây hợp lý. Mạch đã được thi công hoàn tất.

Thi công hộp đựng mô hình phù hợp với board mạch. Đã hoàn thành, board mạch được đặt cố định trong hộp. Kích thước hộp là 74x45x40mm.

Kiểm tra thiết bị: diễn ra xuyên suốt trong quá trình thi công, với 2 lần thi công phần cứng và 2 lần thiết kế hộp. Sản phẩm cuối cùng đã đạt gần 90% yêu cầu đề ra.

#### Phần mềm gồm:

Lập trình cho ESP8266 xử lý yêu cầu của người dùng, điều khiển đóng/tắt thiết bị và truyền nhận dữ liệu từ server. Đã hoàn thành cơ bản, thời gian đáp ứng tương đối nhanh.

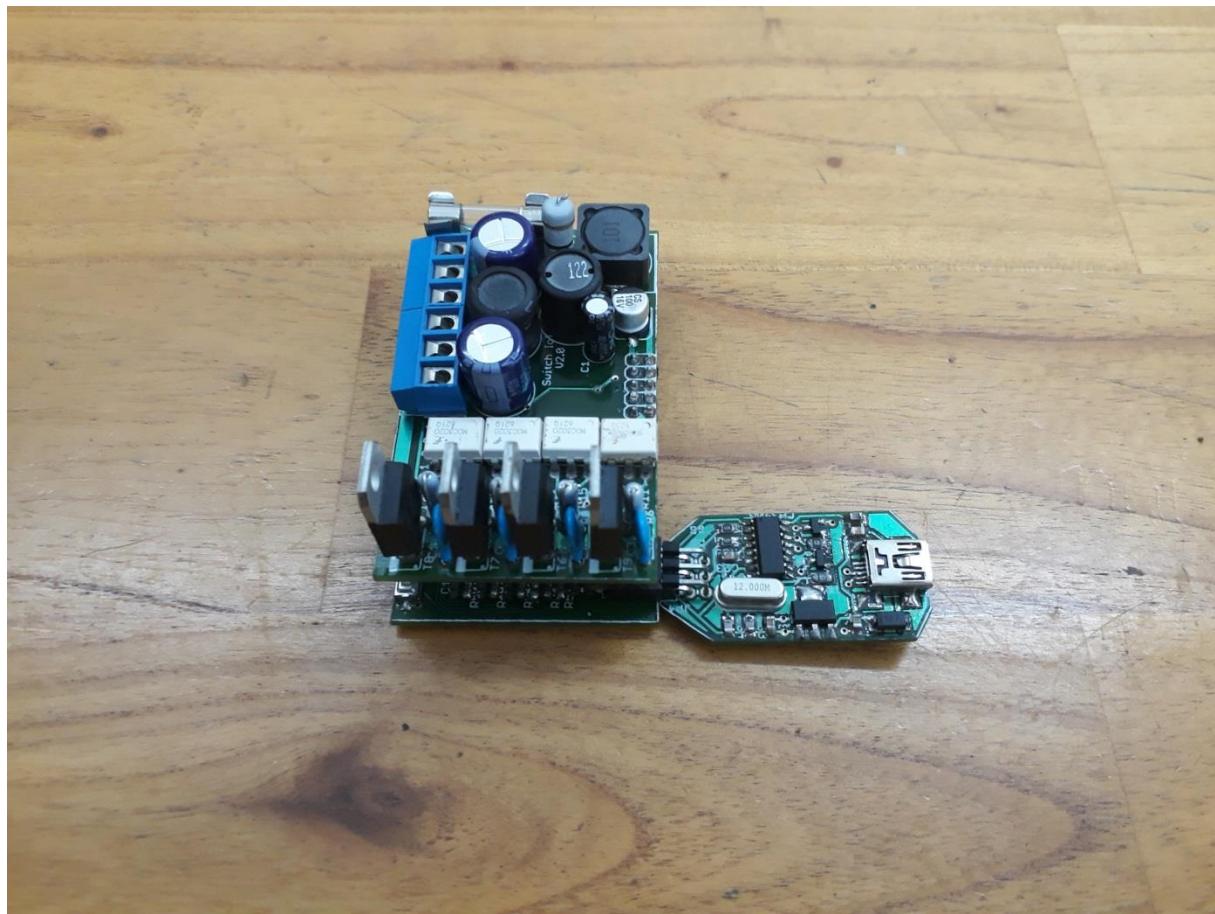
Xây dựng server, cơ sở dữ liệu, giao diện web. Đã hoàn thành cơ bản, server hoạt động ổn định, xây dựng cơ sở dữ liệu cơ bản, giao diện web đúng như yêu cầu đề ra.

Mô hình đã hoàn thành được yêu cầu đề ra, đã kết nối truyền nhận dữ liệu với server, có thể lưu trạng thái đóng/tắt các thiết bị trên web, có giao diện quản lý trên web. Hệ thống hoạt động đáp ứng cơ bản, vẫn còn nhiều ở phần cứng, cấu trúc cơ sở dữ liệu chưa tối ưu.

### 4.2 THI CÔNG HỆ THỐNG

#### 4.2.1 Thi công board mạch

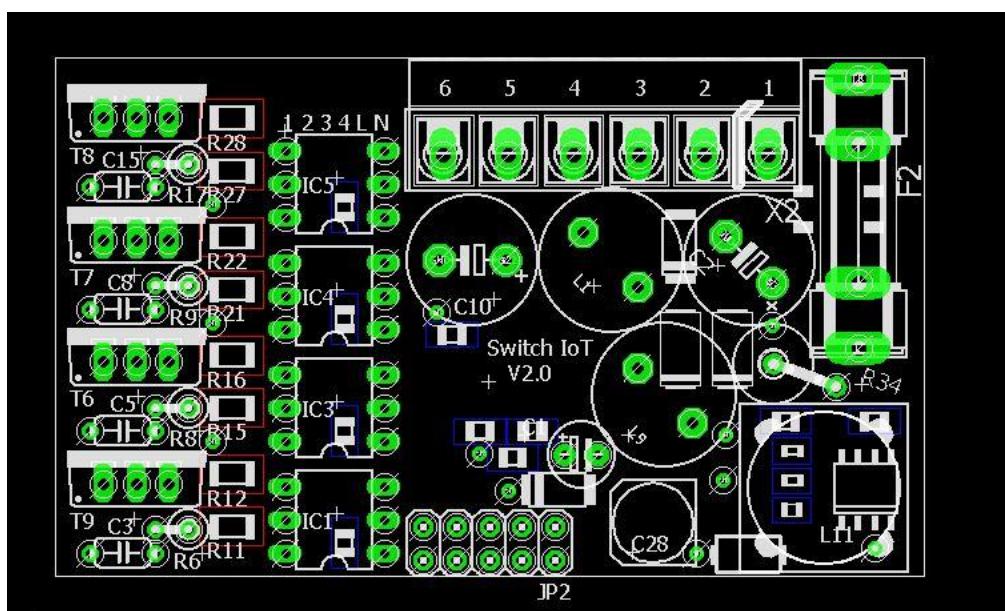
Thi công board mạch gồm 3 PCB. Một PCB chứa mạch nguồn và mạch công suất, có 2 đầu vào của dây 220VAC, có 4 dây ra nối các thiết bị sử dụng điện 220VAC, có rào để kết nối với PCB thứ 2, gồm 4 chân tín hiệu nhận dữ liệu điều khiển từ ESP và 2 chân nguồn 3,3VDC. PCB thứ 3 chứa mạch nạp chương trình cho ESP, được kết nối PCB thứ 2.



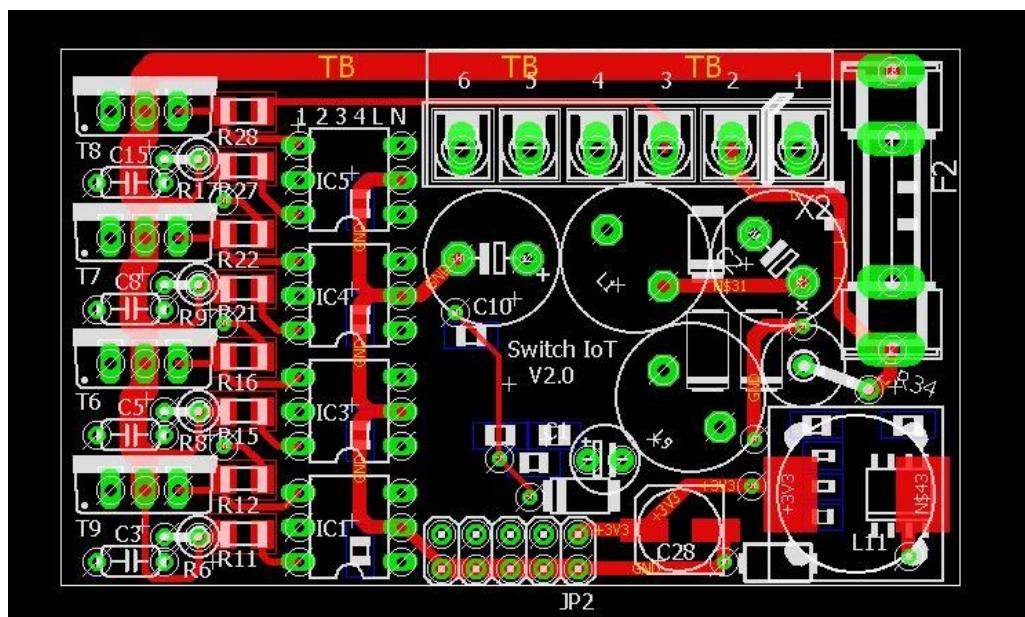
Hình 4.1: Ảnh thực tế board mạch của mô hình.

PCB thứ 3 là mạch nạp chương trình, chỉ sử dụng khi cần nạp chương trình mới vào ESP. Khi mô hình hoạt động, thì không cần kết nối PCB thứ 3.

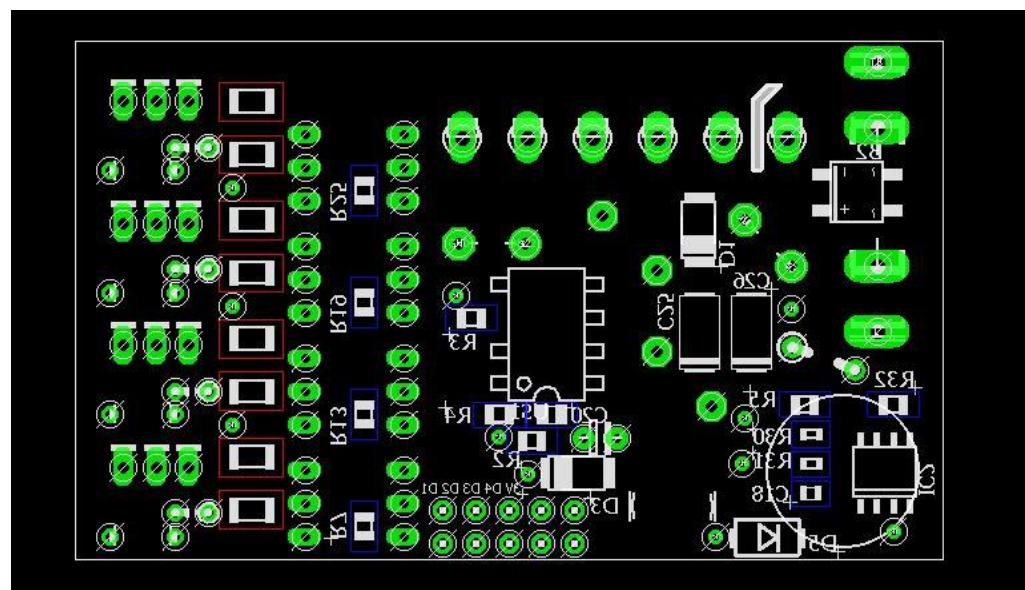
Mạch in của PCB thứ 1:



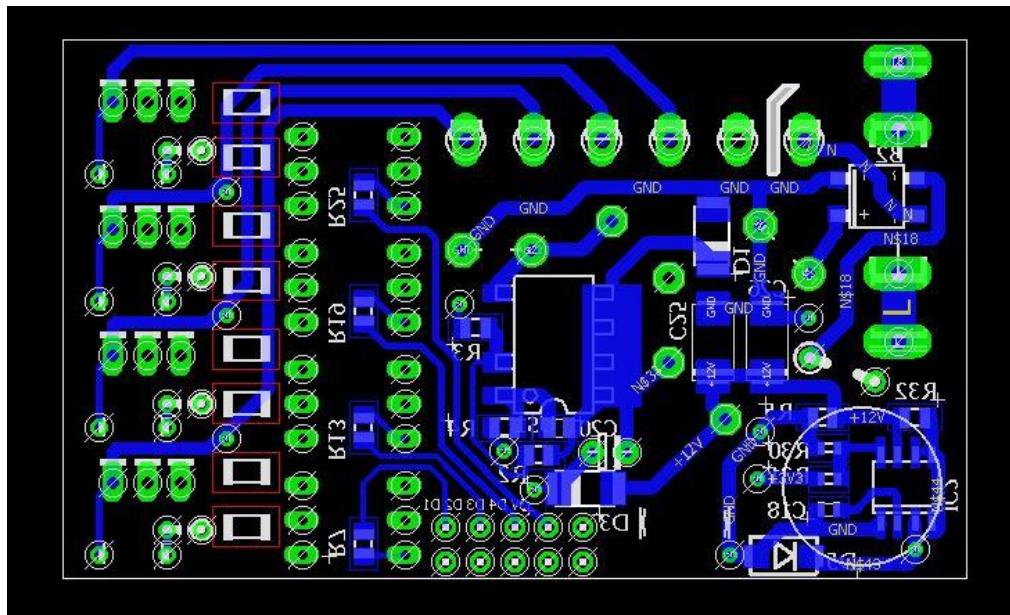
Hình 4.2: Sơ đồ bố trí linh kiện mặt trên của PCB thứ 1.



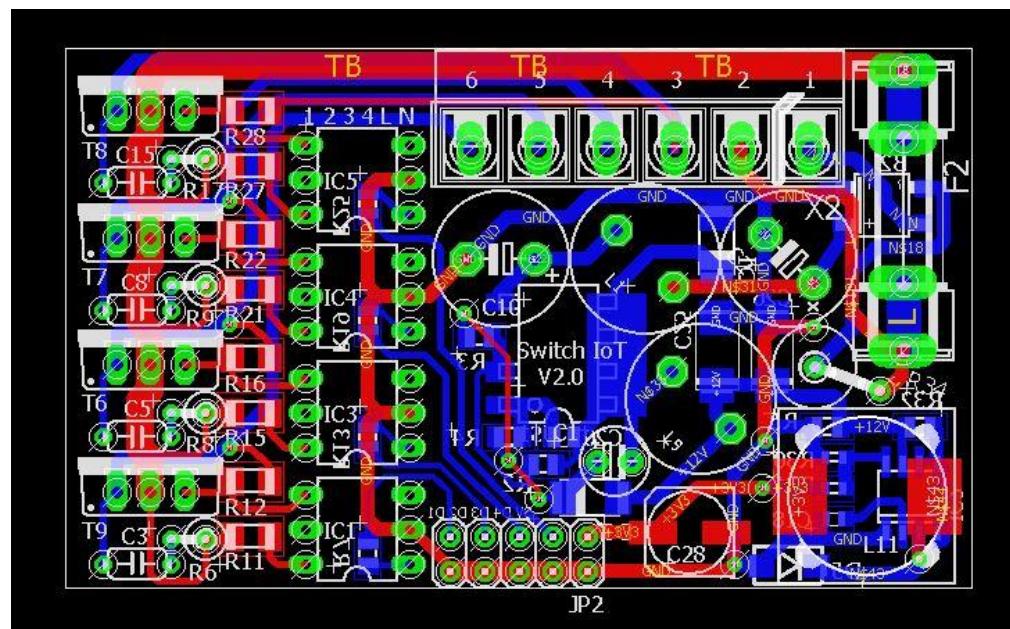
Hình 4.3: Sơ đồ đi dây mặt trên của PCB thứ 1.



Hình 4.4: Sơ đồ bố trí linh kiện mặt dưới của PCB thứ 1.

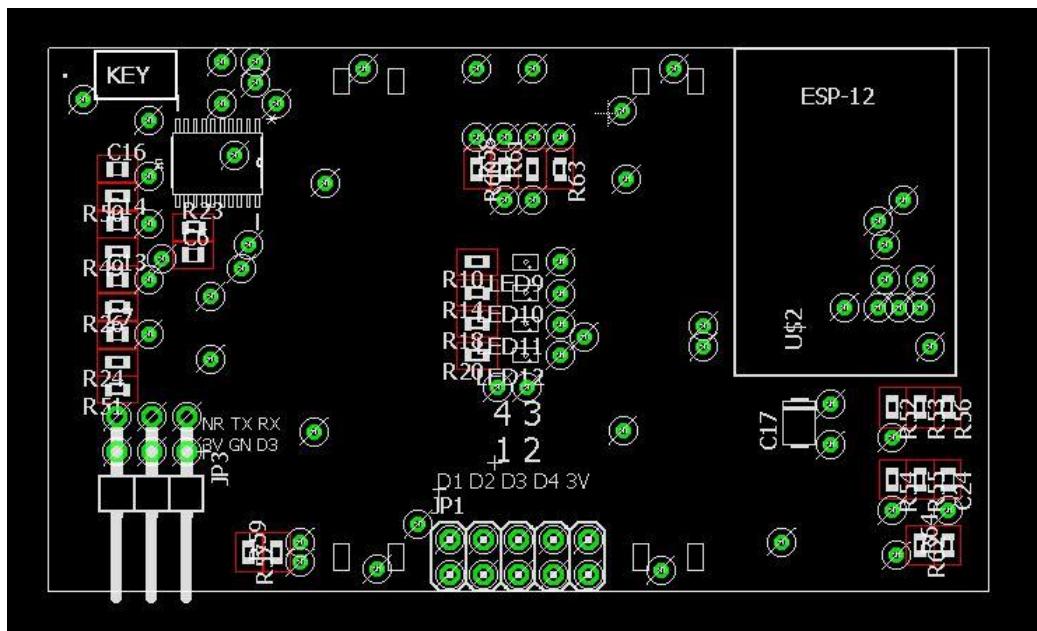


Hình 4.5: Sơ đồ đi dây mặt dưới của PCB thứ 1.

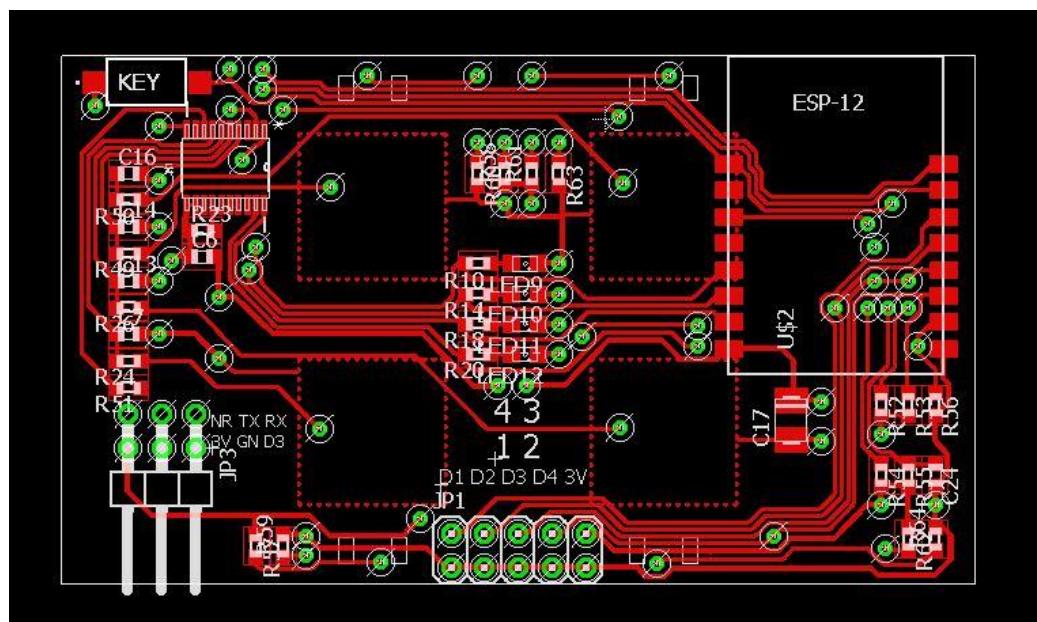


Hình 4.6: Sơ đồ mạch in của PCB thứ 1.

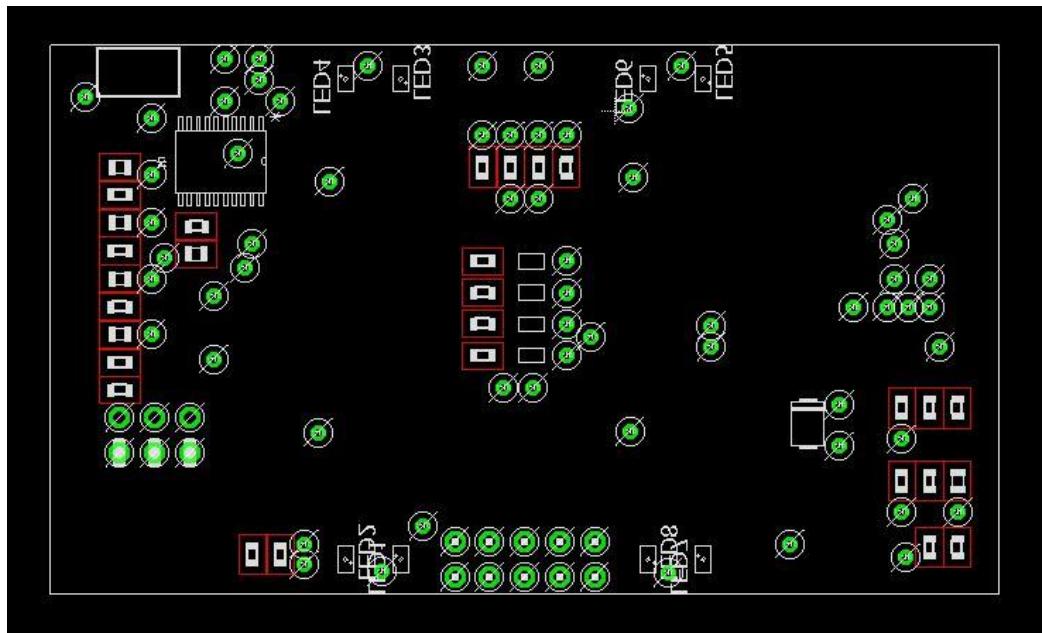
Mạch in của PCB thứ 2:



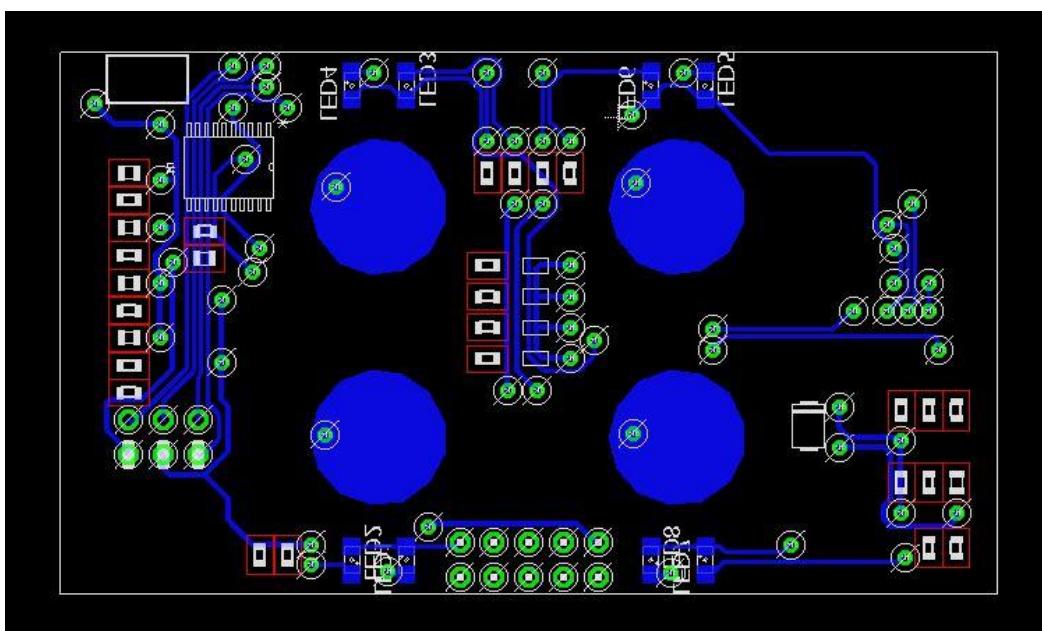
Hình 4.7: Sơ đồ bố trí linh kiện mặt trên của PCB thứ 2.



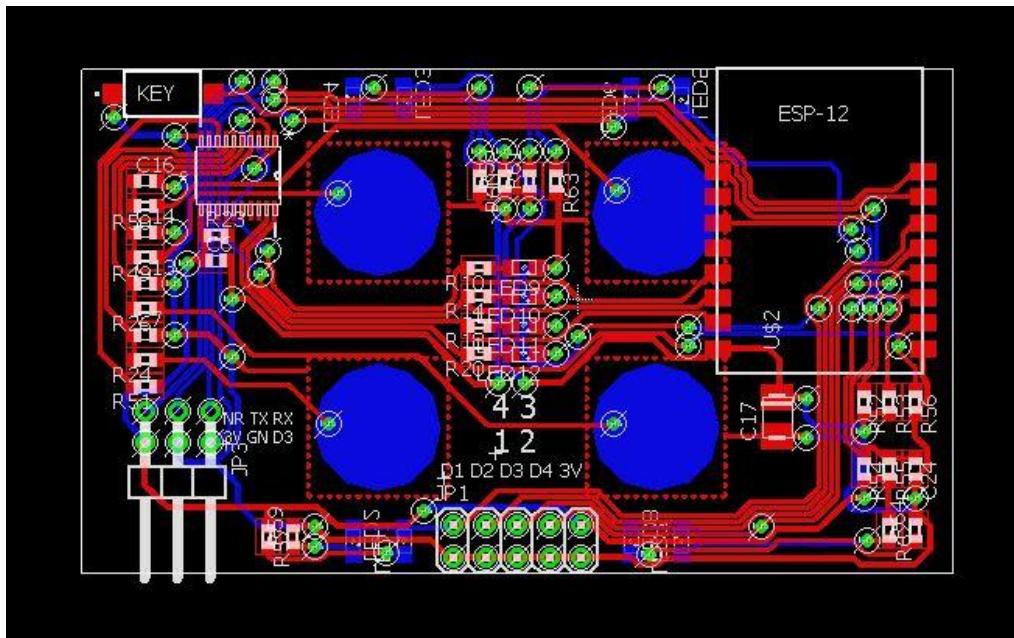
Hình 4.8: Sơ đồ đi dây mặt trên của PCB thứ 2.



Hình 4.9: Sơ đồ bố trí linh kiện mặt dưới của PCB thứ 2.

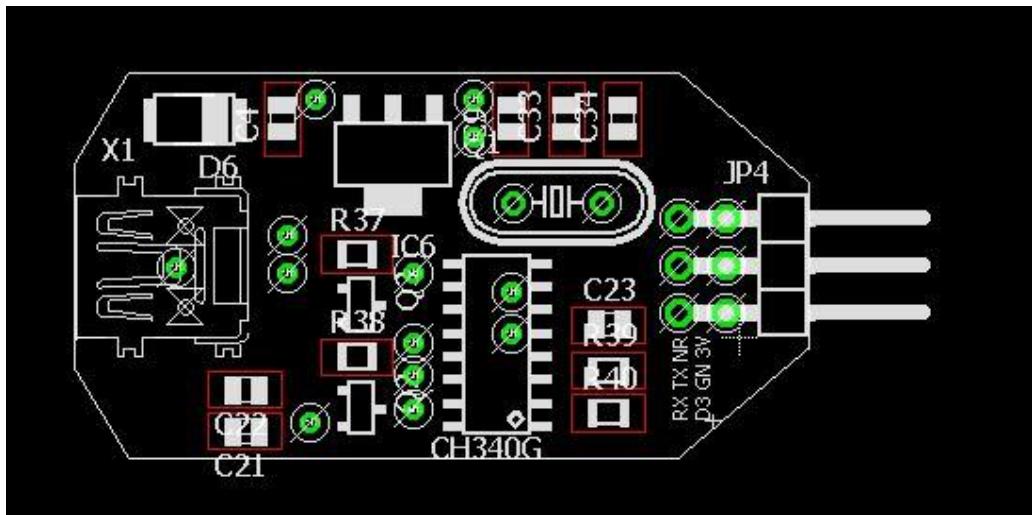


Hình 4.10: Sơ đồ đi dây mặt dưới của PCB thứ 2.

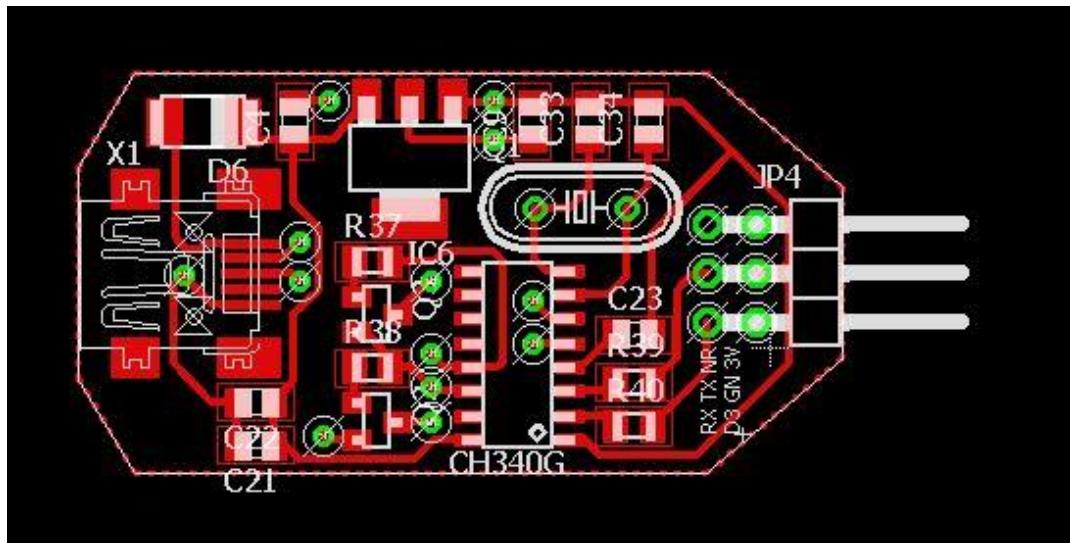


Hình 4.11: Sơ đồ mạch in của PCB thứ 2.

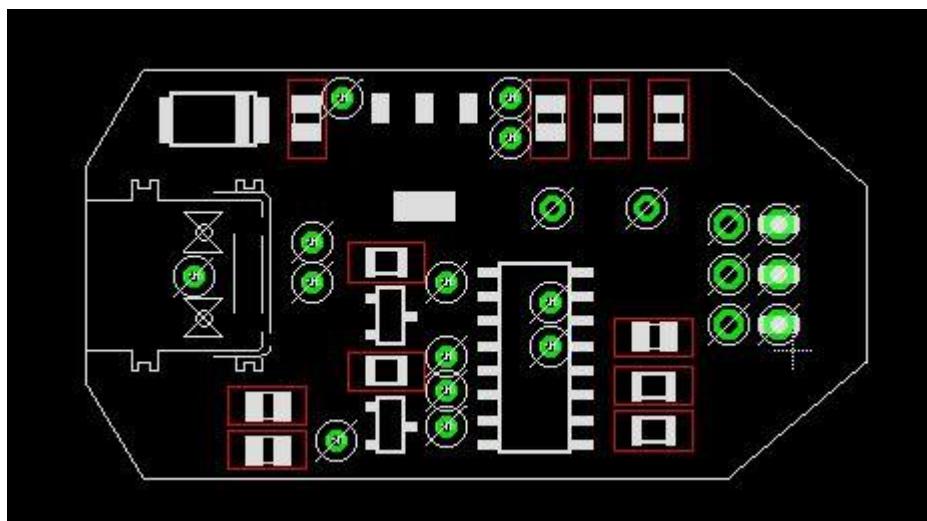
Mạch in của PCB thứ 3:



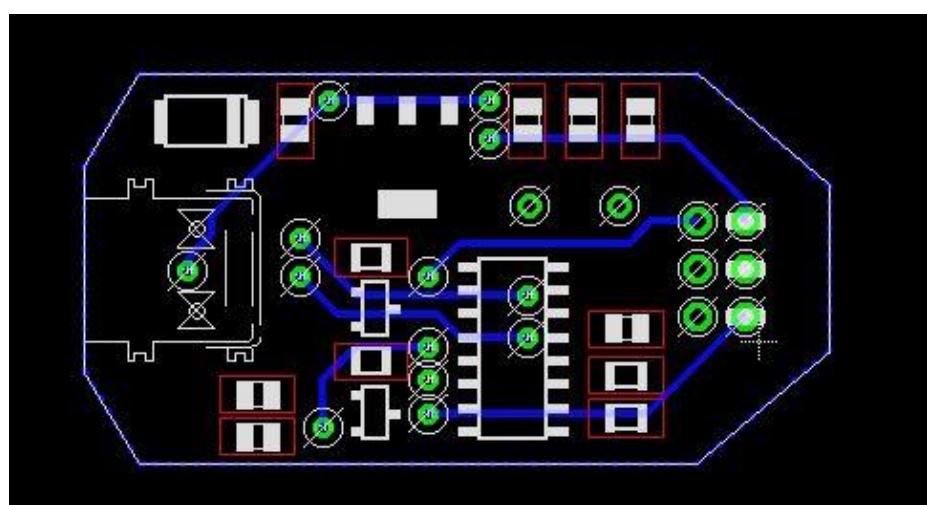
Hình 4.12: Sơ đồ bố trí linh kiện mặt trên của PCB thứ 3.



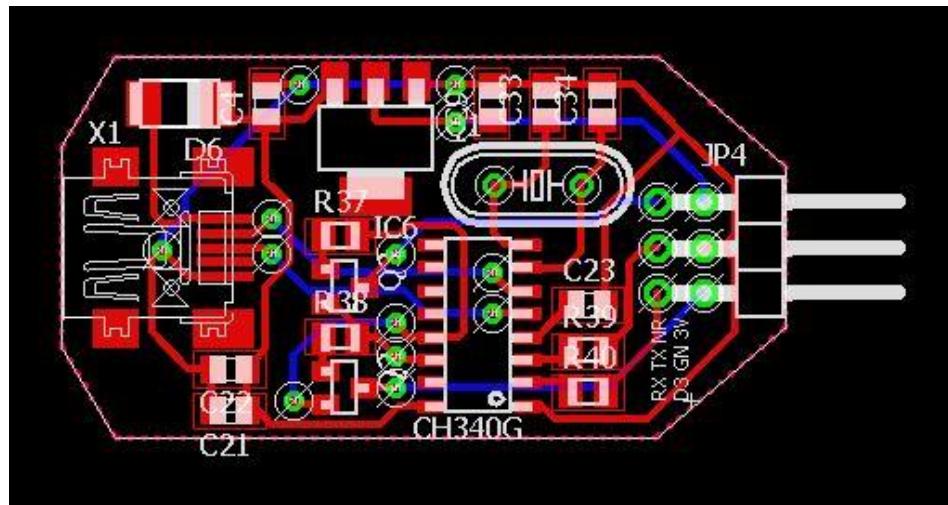
Hình 4.13: Sơ đồ đi dây mặt trên của PCB thứ 3.



Hình 4.14: Sơ đồ bố trí linh kiện mặt dưới của PCB thứ 3.



Hình 4.15: Sơ đồ đi dây mặt dưới của PCB thứ 3.



Hình 4.16: Sơ đồ mạch in của PCB thứ 3.

Bảng 4.1: Danh sách linh kiện

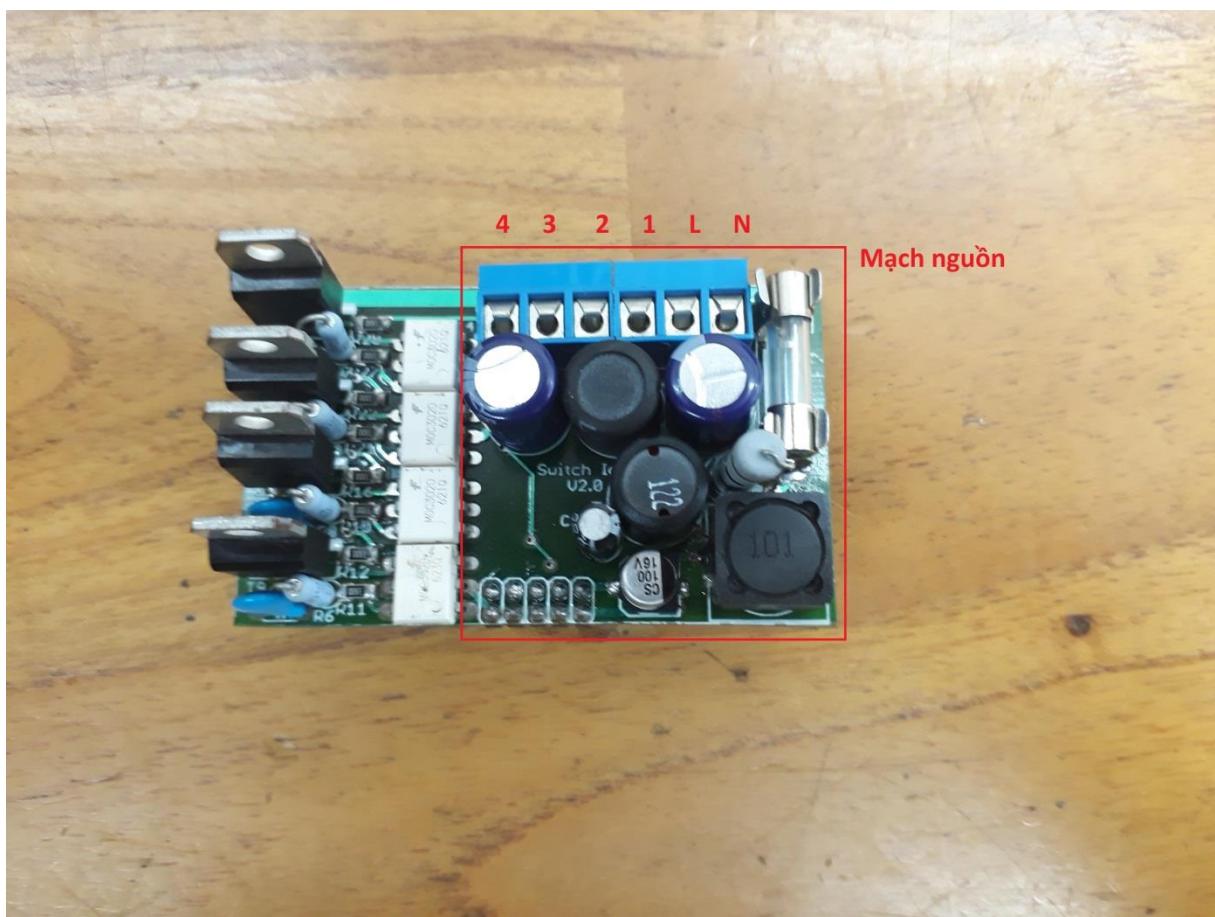
STT	Tên linh kiện	Giá trị	Dạng vỏ	Chú thích
1	Điện trở	0,15 Ω, 1/8W	R0805	Sai số 1%
2	Điện trở	1,2 kΩ, 1/8W	R0603	Sai số 1%
3	Điện trở	10 Ω, 2W	R0603	Sai số 1%
4	Điện trở	11,8 kΩ, 1/8W	R0805	Sai số 1%
5	Điện trở	12 kΩ, 1/8W	R0805	Sai số 1%
6	Điện trở	1 kΩ, 1/8W	R0603	Sai số 1%
7	Điện trở	2,49 kΩ, 1/8W	R0805	Sai số 1%
8	Điện trở	220 Ω, 1/8W	R0805	Sai số 1%
9	Điện trở	26,7 kΩ, 1/8W	R0805	Sai số 1%
10	Điện trở	2 kΩ, 1/8W	R0603	Sai số 1%
11	Điện trở	3,3 kΩ, 1/8W	R0805	Sai số 1%
12	Điện trở	330, 1/8W	M1206	Sai số 1%
13	Điện trở	39, 1/8W	R0805	Sai số 1%
14	Điện trở	4,7 kΩ, 1/8W	R06030	Sai số 1%
15	Điện trở	470 Ω, 1/8W	R0805	Sai số 1%
16	Điện trở	60 kΩ, 1/8W	R0603	Sai số 1%
17	Cuộn cảm	1,2 mH, 3A	R0603	
18	Cuộn cảm	100 uH, 3A	0912	
19	Cuộn cảm	1 mH, 3A	0912	

20	Tụ điện	100 nF, 50V	C0805	
21	Tụ điện	100 uF, 50V	C/6032	
22	Tụ điện	10 nF, 25V	C0603	
23	Tụ điện	10 uF, 25V	C0805	
24	Tụ điện	150 pF, 50V	C0603	
25	Tụ điện	3,3 uF, 400V	E5-10,5	
26	Tụ điện	470pF, 16V	C0603	
27	Thạch anh	12 MHz	HC49/S	
28	Hàng rào đực 5x2		DIP	
29	Hàng rào cái 5x2		DIP	
30	Hàng rào đực 3x2		DIP	
31	Hàng rào cái 3x2		DIP	
32	Nút nhấn 2 chân	12VDC, 50mA	SMD	
33	IC AMS117	3,3V	SOT223	IC ỗn áp
34	IC AT42QT2120		QT2120	IC cảm ứng điện dung
35	Triac BTA12	12A, 600V	TO220BV	
36	Led đơn màu đỏ	3,3V, 10mA	SML0805	
37	IC CH340G		SOIC16	IC chuyển USB-UART
38	ESP8266 12-F		ESP12	Module wifi
39	Cầu chì	7A	SHK20L	
40	Transistor J3Y	40V, 0.5A	SOT23	
41	IC LNK3206G	225mA	SMD	IC ỗn áp
42	Diode M7	1000V, 1A	DO-214AC	
43	Cầu diode MB6S	600V, 0.5A	SOIC-4	
44	IC MC34063	1,5A	SOIC-8	IC ỗn áp
45	USB Mini Type-B		32005-201	Cổng USB
46	Opto MOC3020	400V	DIL06	

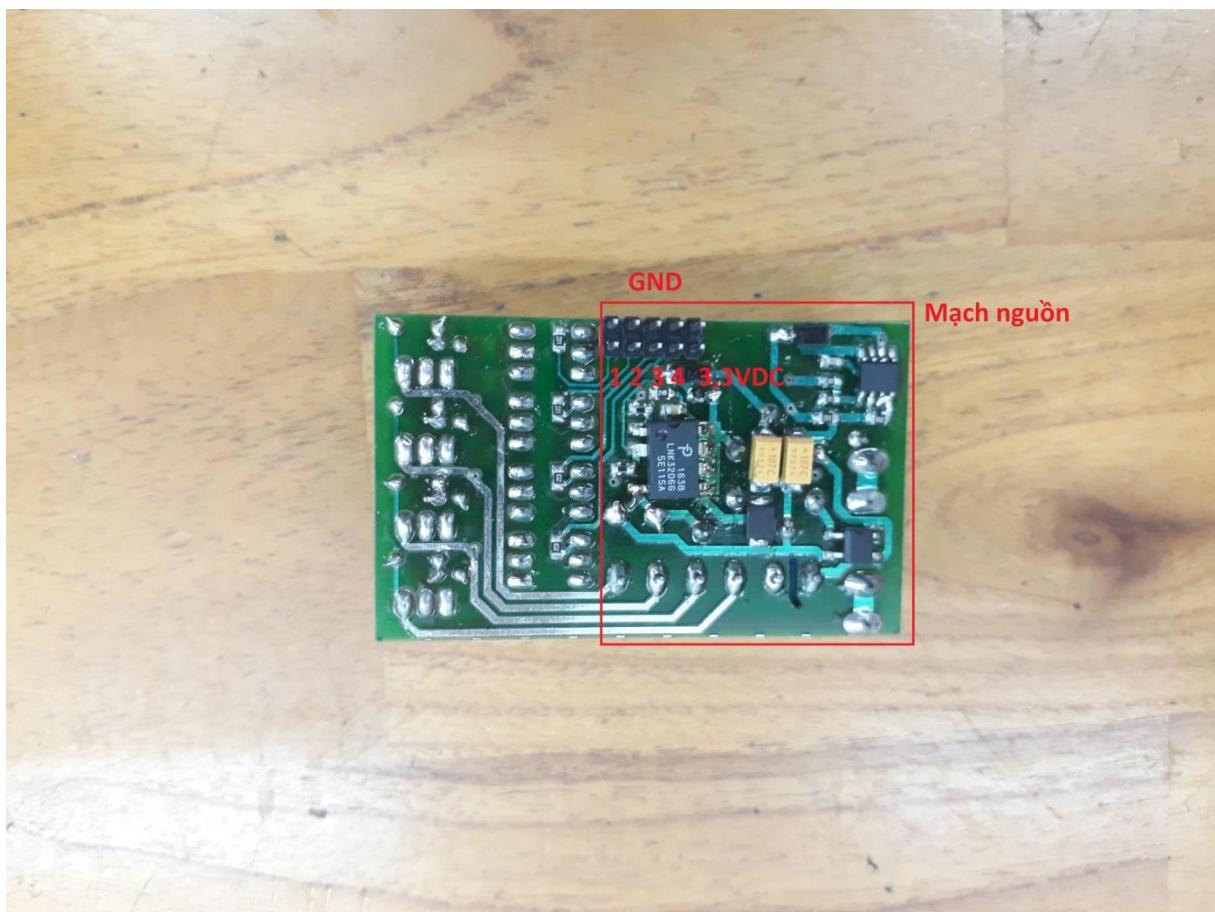
47	Diode MRA4007	1000V, 1A	DO-214AC	
48	Diode SS14	40V, 1A	DO-214BA	
49	Diode STTH1R06	600V, 1A	DO-214AC	

#### 4.2.2 Lắp ráp và kiểm tra

##### a. Lắp ráp mạch nguồn



Hình 4.17: Mặt trên khối nguồn.

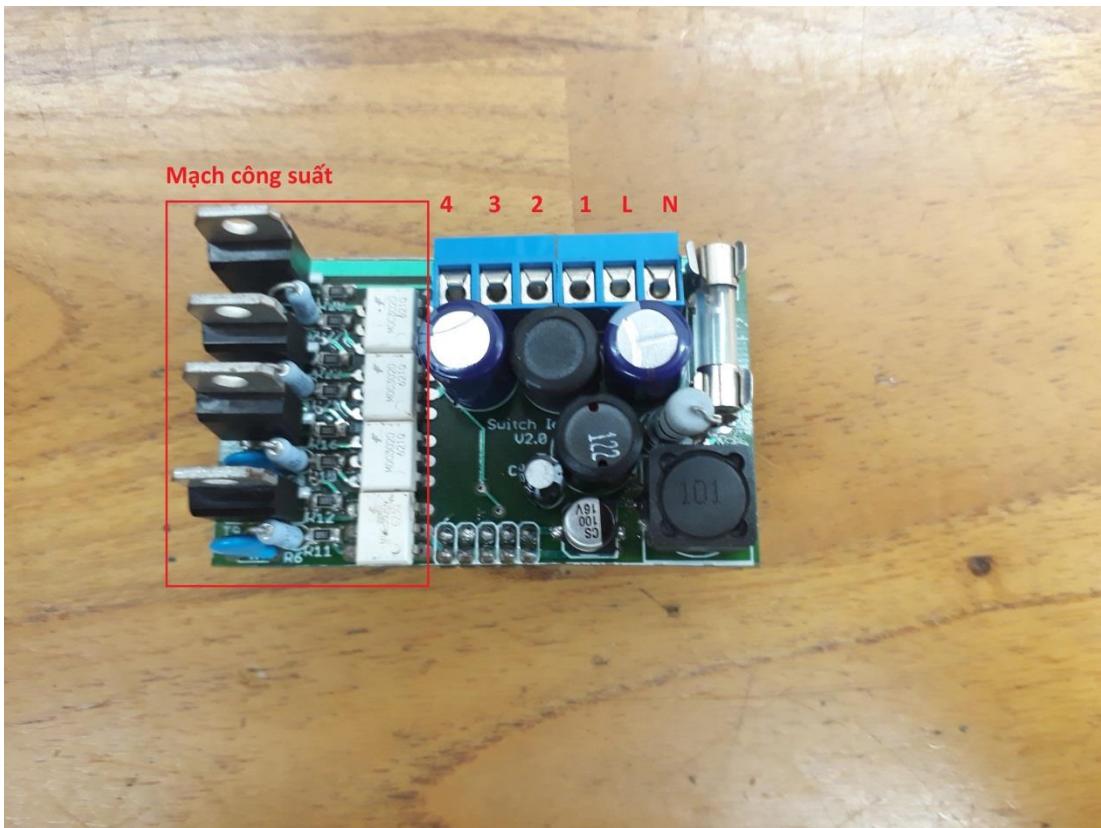


Hình 4.18: Mặt dưới khối nguồn.

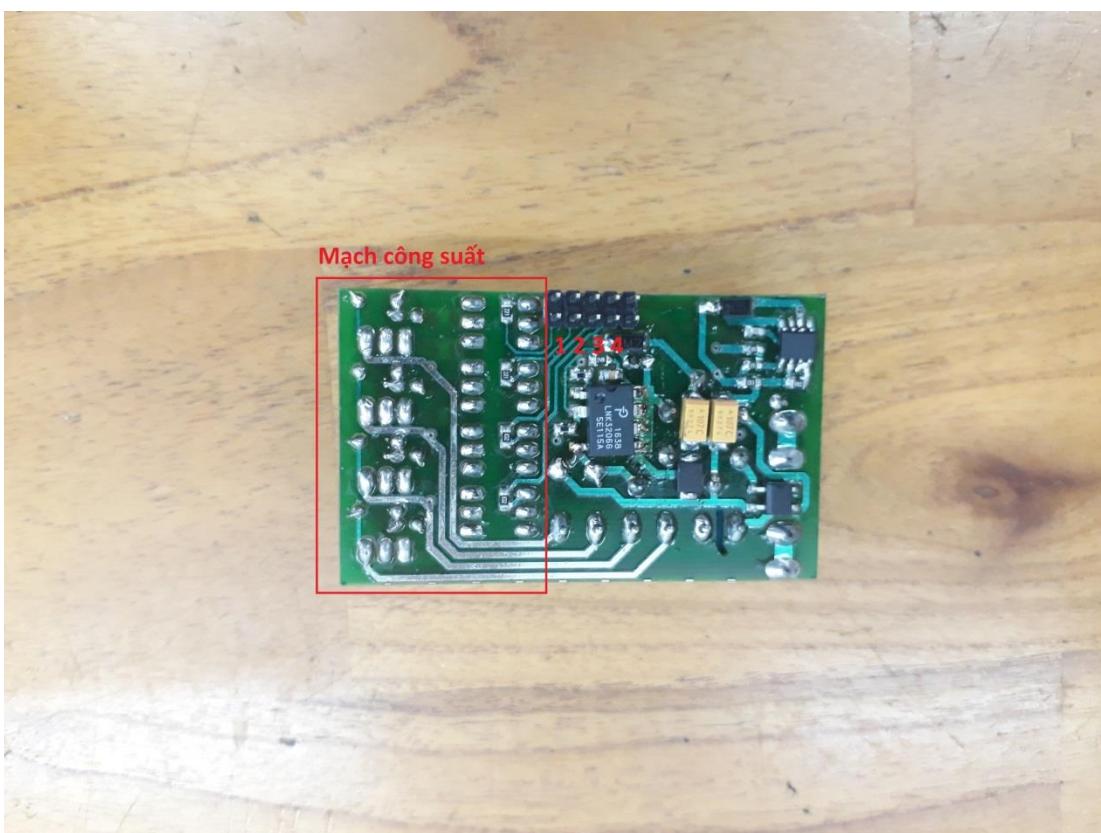
Kích thước module nguồn vào khoảng 40x40mm. Sử dụng mạch in 2 lớp, lớp trên đặt các linh kiện xuyên lõi, lớp dưới đặt các linh kiện dán. Kiểm tra khối nguồn bằng các dây 220VAC vào L và N. Kiểm tra ngõ ra (GND – 3,3V) nếu đạt 3,3V +- 5% thì đạt yêu cầu về áp, dùng trở công suất kiểm tra khả năng chịu dòng của nguồn, nếu dòng khoảng 400 ~ 500mA mà áp ra 3,3V không thay đổi thì đạt yêu cầu.

#### b. Lắp ráp mạch công suất

Mạch công suất gồm 4 MOC3020 và 4 TRIAC BTA12 nhận tín hiệu từ 4 chân 1, 2, 3, 4 ở mặt dưới và đóng tắt thiết bị tương ứng tại 4 ngõ ra 1, 2, 3, 4 ở mặt trên.



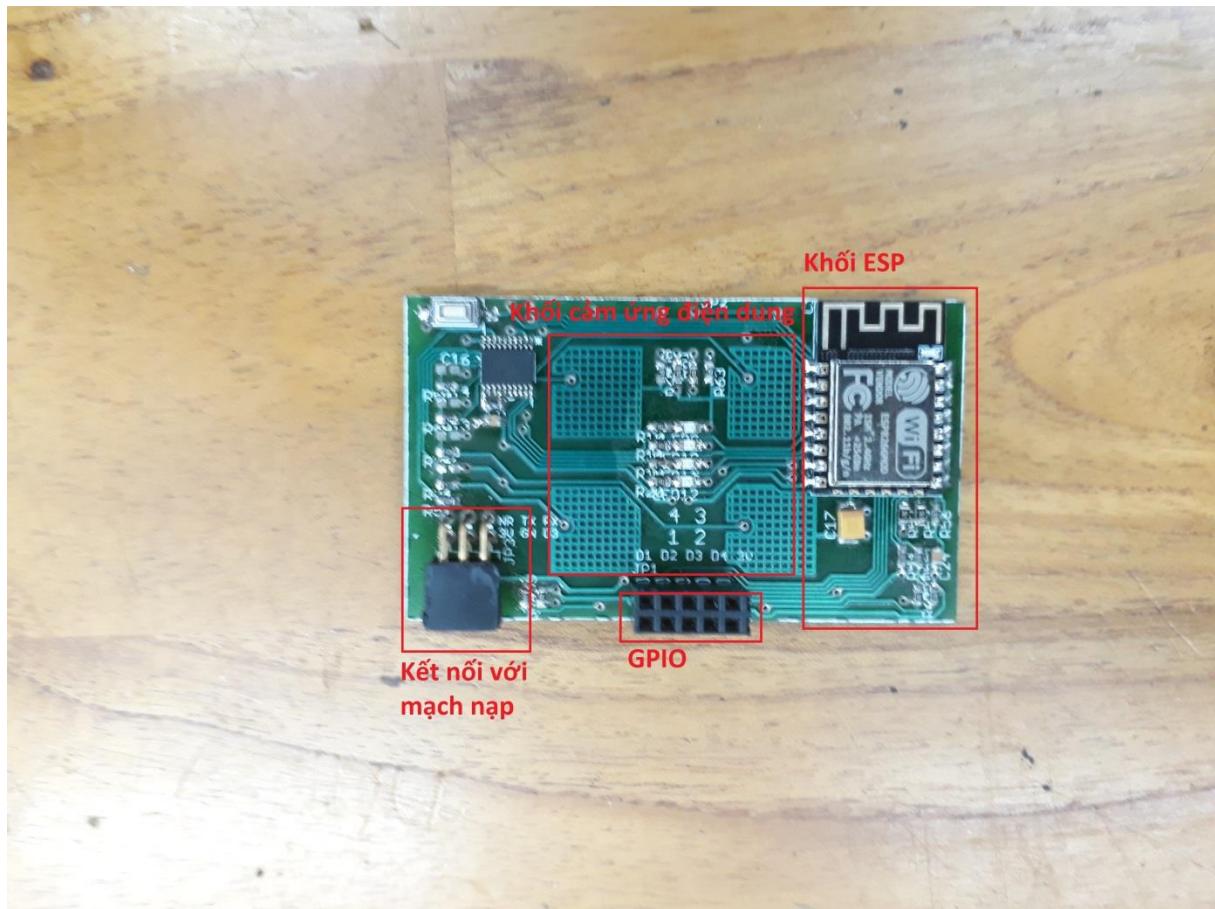
Hình 4.19: Mặt trên khối công suất.



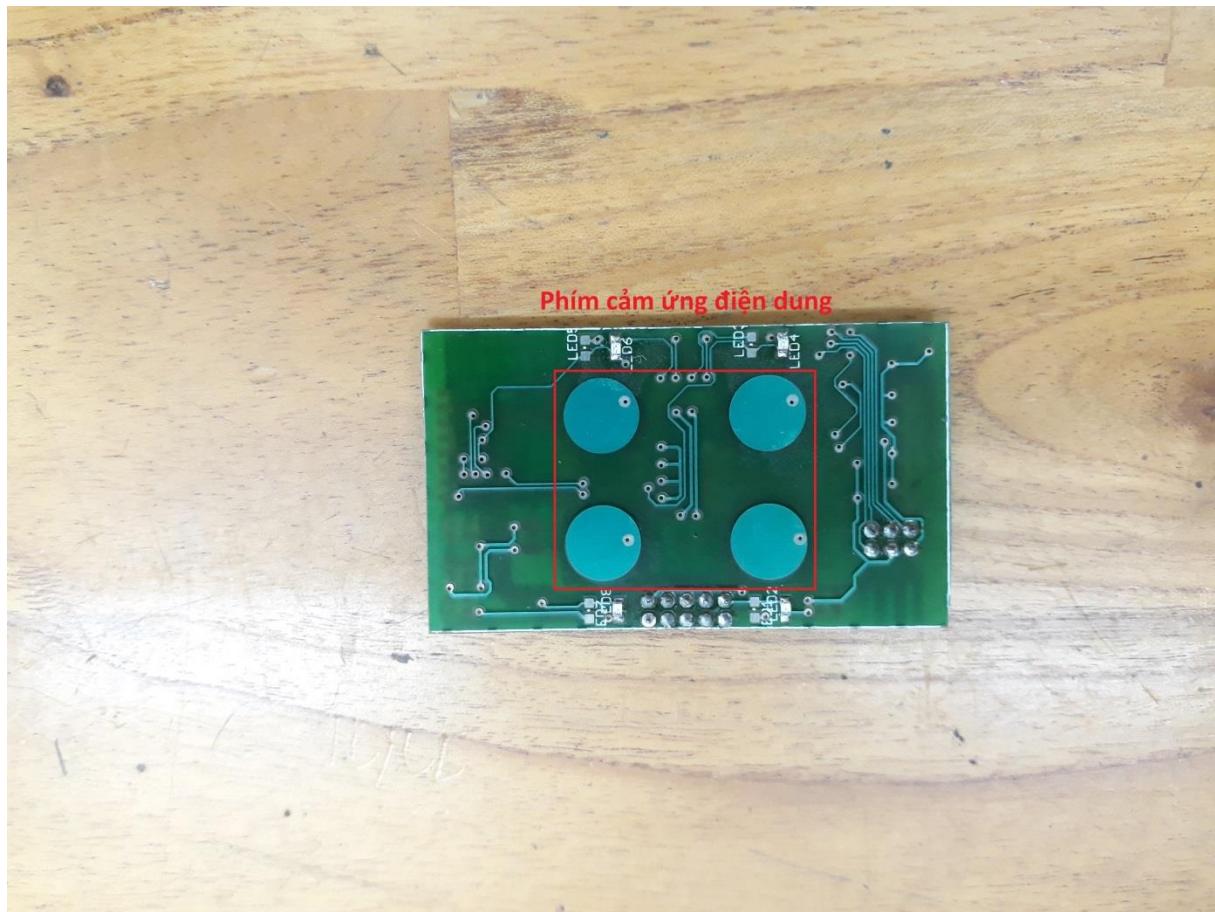
Hình 4.20: Mặt dưới khối công suất.

Kiểm tra mạch kết nối dây 220V nguồn và kết nối dây với các thiết bị sử dụng điện 220V. Các đầu dây còn lại của thiết bị nói chung với dây nguồn (dây N). Kích các chân tín hiệu 1, 2, 3, 4 xem mạch hoạt động tương ứng không, kiểm tra nhiều có tác động làm thiết bị tự kích với tải cảm như quạt... Nếu mạch đóng tắt tốt, không bị tự kích với các loại tải thiết bị thì mạch đạt yêu cầu.

### c. Lắp ráp mạch xử lý trung tâm và mạch cảm ứng điện dung



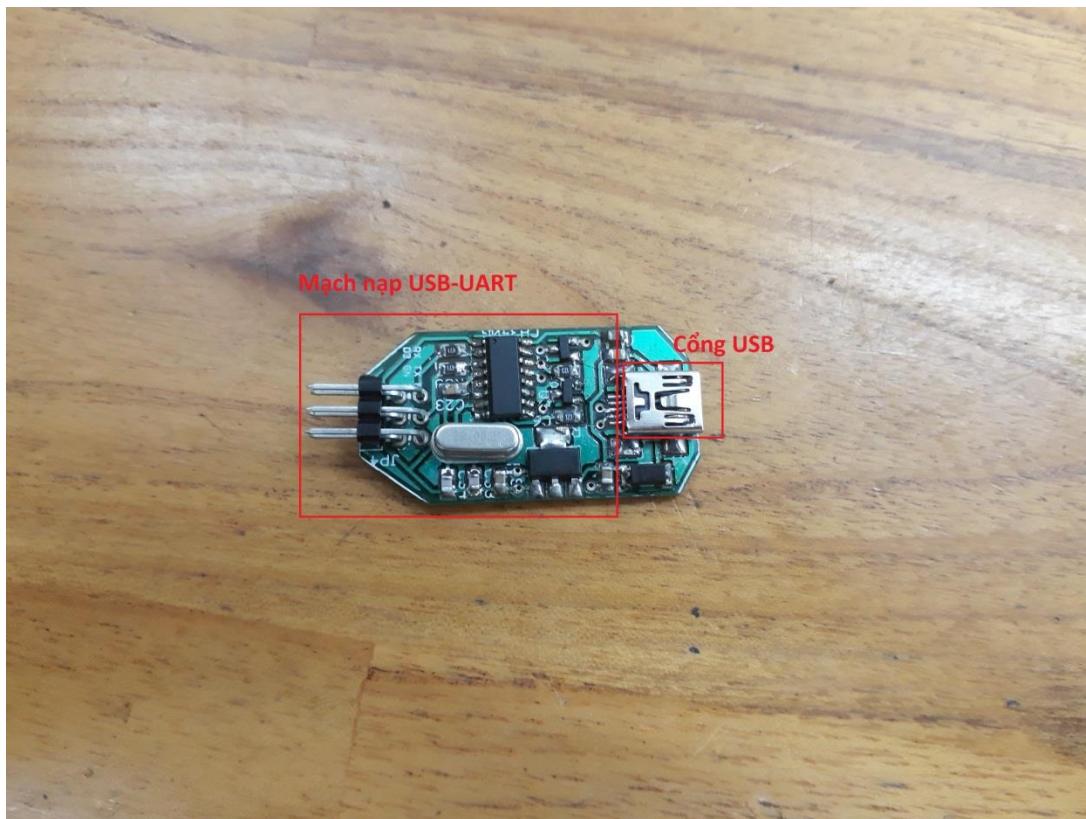
Hình 4.21: Mặt trên PCB thứ 2.



Hình 4.22: Mặt dưới PCB thứ 2.

Kiểm tra cấp nguồn 3,3V từ PCB 1, ESP có hoạt động, nạp chương trình cho ESP, ESP có nhận chương trình, kiểm tra KEY nhấn có nhận tín hiệu và có nhạy hay không, nếu có thì mạch đạt yêu cầu.

d. Lắp ráp mạch nạp chương trình cho ESP



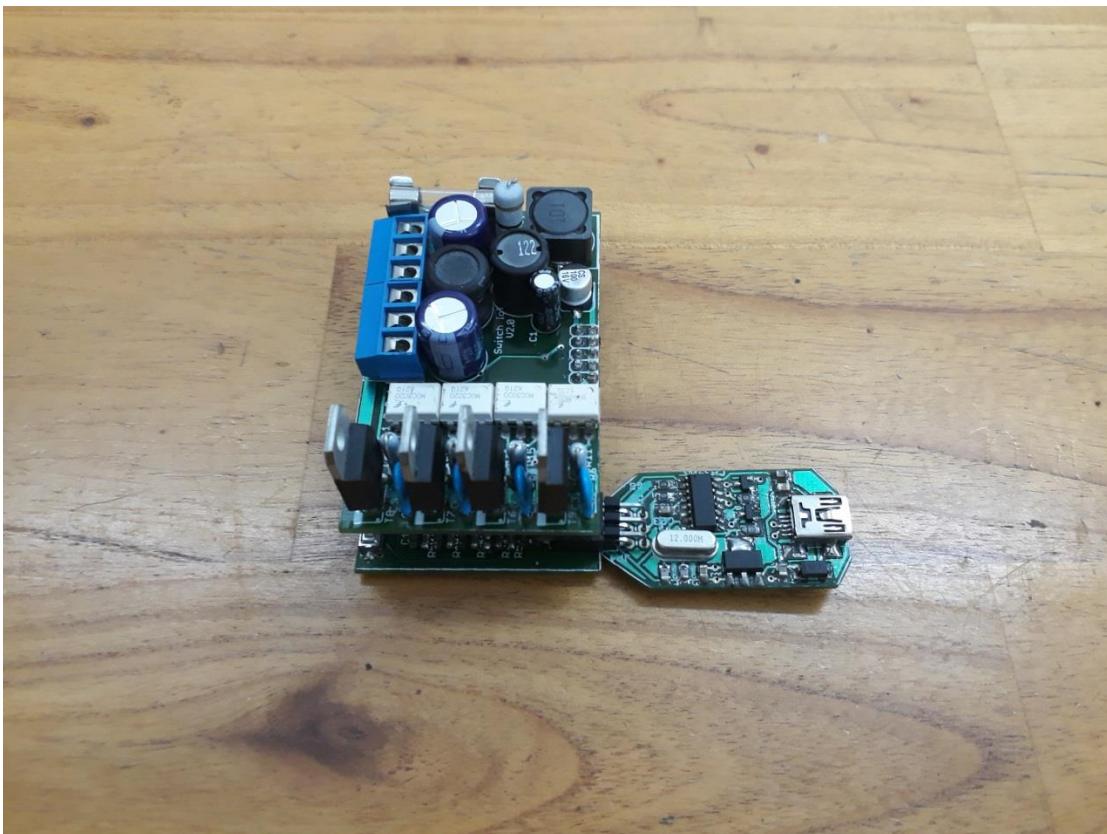
Hình 4.23: Mặt trên của mạch nạp USB-UART.



Hình 4.24: Mặt dưới của mạch nạp USB-UART.

Kết nối mạch nạp với PCB thứ 2 (mạch xử lý trung tâm và mạch cảm ứng điện dung). Kết nối với máy tính, xem máy tính có nhận cổng COM hay không, nếu có thì đạt yêu cầu.

#### e. Lắp ráp toàn mạch



Hình 4.25: Board công tắc IoT.

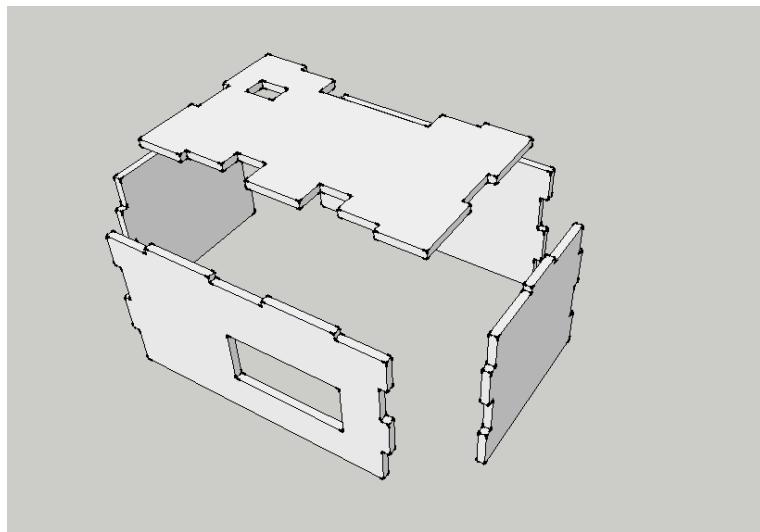
Khi cho board hoạt động thì không cần gắn mạch nạp, mô hình sẽ trở nên nhỏ gọn hơn. Cáp nguồn và kết nối với thiết bị, kiểm tra tổng thể toàn mạch, hoạt động đúng yêu cầu.

### 4.3 ĐÓNG GÓI VÀ THI CÔNG MÔ HÌNH

#### 4.3.1 Đóng gói bộ điều khiển

Board mạch được đóng hộp bởi 1 khối mica lắp ghép vào nhau, thiết kế vỏ với kích thước nhỏ gọn để đặt vừa trong đế âm tường.

Thiết kế hộp có khoảng trống đưa dây 220VAC và dây các thiết bị, board mạch ổn định, có lỗ thông thoát khí, tăng tính tản nhiệt khi board hoạt động.



Hình 4.26: Thiết kế vỏ hộp bằng phần mềm Solidworks.

#### 4.3.2 Thi công mô hình

Mô hình gồm 4 công tắc điều khiển 4 bóng đèn nhỏ sử dụng điện áp 220VAC, công tắc được gắn như các công tắc thiết bị trong gia đình có đế âm tường. Mô hình là 1 khối mica kích thước 74x45x40cm. Các công tắc và đèn được gắn ở phía trước, việc đi dây sẽ ở phía sau mô hình.

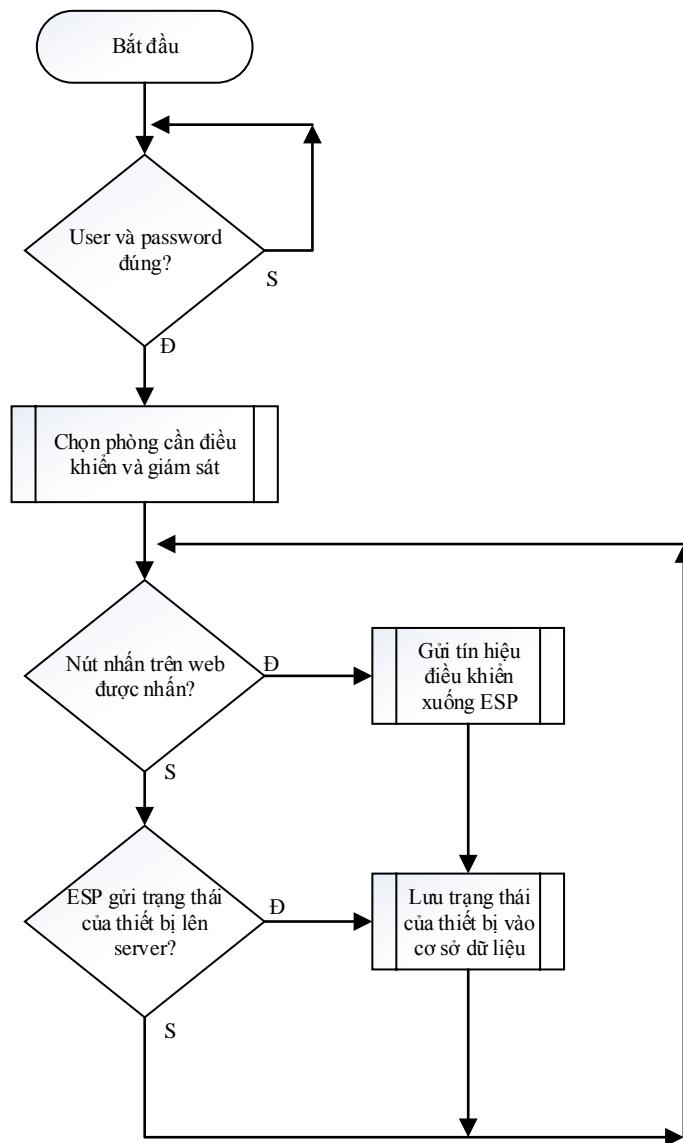
### 4.4 LẬP TRÌNH HỆ THỐNG

#### 4.4.1 Lưu đồ giải thuật của Web Server

##### a. Yêu cầu

Người dùng có thể điều khiển đóng tắt, giám sát trạng thái của các thiết bị thông qua web server.

*b. Lưu đồ của web server*



Hình 4.27: Lưu đồ của server.

Người dùng truy cập vào web, đăng nhập đúng user và password. Sau đó chọn phòng cần điều khiển và giám sát. Khi người dùng nhấn vào nút điều khiển trên web, thì web sẽ gửi tín hiệu điều khiển xuống ESP, thiết bị tương ứng được điều khiển. Sau đó web lưu trạng thái của thiết bị vừa được điều khiển vào cơ sở dữ liệu. Khi ESP gửi trạng thái của thiết bị lên web, web cũng sẽ lưu trạng thái của thiết bị vào cơ sở dữ liệu.

#### 4.4.2 Lưu đồ giải thuật của ESP

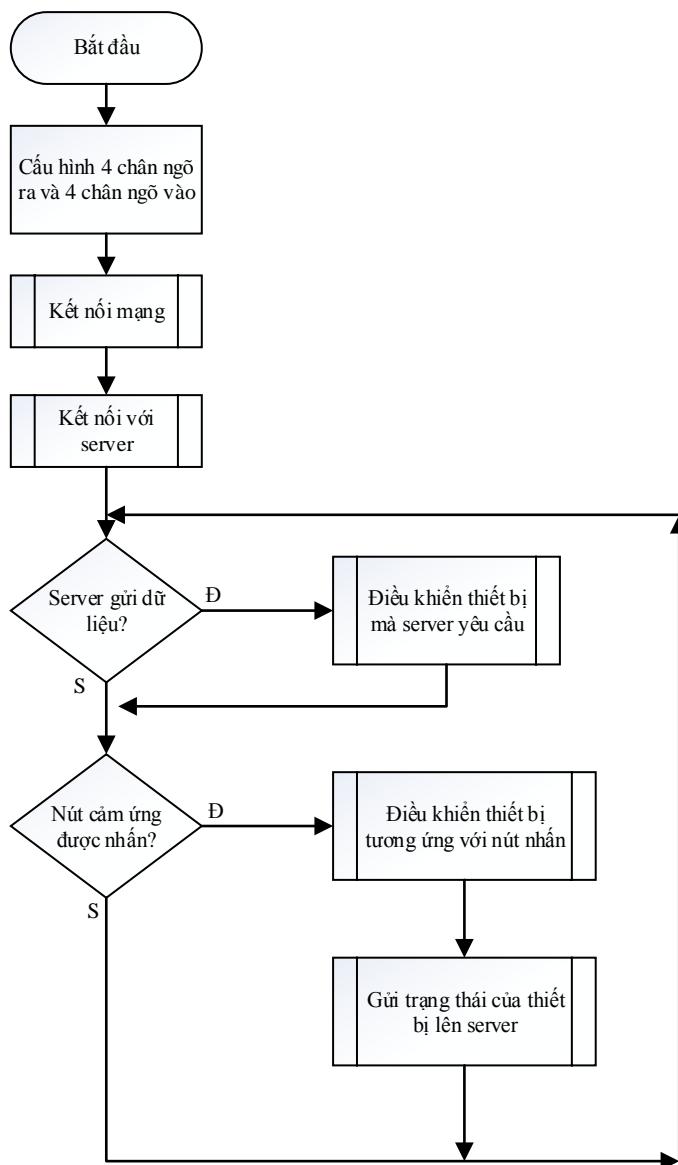
##### a. Yêu cầu điều khiển

ESP sẽ thực hiện 2 nhiệm vụ:

Khi người dùng tác động vào nút nhấn cảm ứng, thì thiết bị tương ứng sẽ được bật lên, sau đó gửi dữ liệu trạng thái của thiết bị vừa được tác động lên server.

Khi người dùng tác động từ web server, để điều khiển thiết bị thì ESP sẽ nhận dữ liệu từ web server. Điều khiển đóng tắt thiết bị mà web server yêu cầu.

##### b. Lưu đồ của ESP



Hình 4.28: Lưu đồ của ESP.

ESP sẽ kết nối với mạng wifi, tiến hành kết nối với server. Sau khi kết nối xong, sẽ thực hiện tuần hoàn. Chờ nếu có dữ liệu từ server gửi xuống, thì sẽ điều khiển thiết

bị mà server mong muốn. Nếu người dùng tác động vào nút nhấn, thì ESP sẽ tiến hành gửi tín hiệu sang mạch công suất để điều khiển thiết bị, sau đó tiến hành gửi dữ liệu lên server. Server sẽ lưu trạng thái của thiết bị.

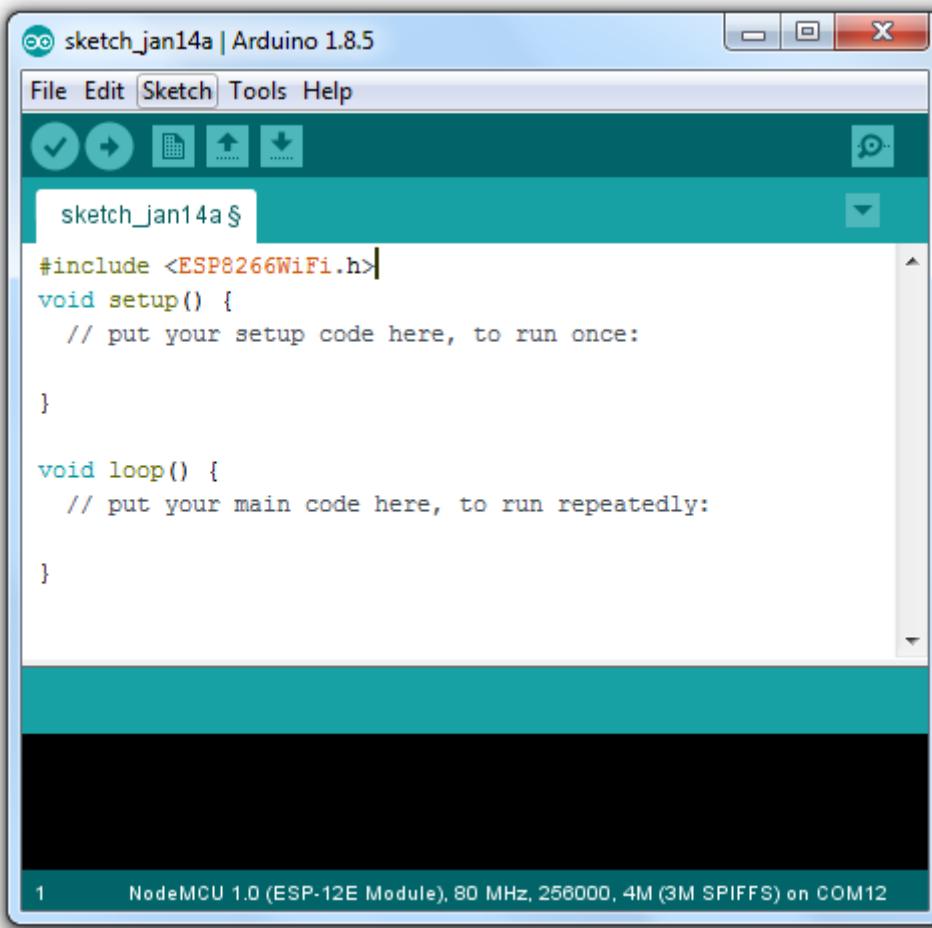
#### 4.4.3 Phần mềm lập trình cho ESP

Để viết chương trình cho ESP8266 Espressif hiện đã hỗ trợ 3 nền tảng SDK (Software Development Kit – Gói phát triển phần mềm) độc lập là: **NONOS SDK**, **RTOS SDK** và **Arduino**. Cả 3 đều có những ưu điểm riêng phù hợp với từng ứng dụng nhất định. Hiện nay Arduino đang được sử dụng rộng rãi bởi tính dễ sử dụng, kiến trúc phần mềm tốt và tận dụng được nhiều thư viện cộng đồng.

**Arduino** là một IDE tích hợp sẵn editor, compiler, programmer và đi kèm với nó là các firmware có bootloader, các bộ thư viện được xây dựng sẵn và dễ dàng tích hợp.

Ngôn ngữ sử dụng là C/C++. Tất cả đều opensource và được đóng góp, phát triển hàng ngày bởi cộng đồng. Triết lý thiết kế và sử dụng của Arduino giúp cho người mới, không chuyên rát dễ tiếp cận, các công ty, hardware dễ dàng tích hợp. Tuy nhiên, với trình biên dịch C/C++ và các thư viện chất lượng được xây dựng bên dưới thì mức độ phổ biến ngày càng tăng và hiệu năng thì không hề thua kém các trình biên dịch chuyên nghiệp cho chip khác.

Đại diện cho Arduino ban đầu là chip AVR, 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, và RISC-V với năng lực phần cứng và phần mềm đi kèm mạnh mẽ hơn nhiều.



Hình 4.29: Phần mềm Arduino.

### **Đặc điểm của Arduino:**

Arduino che dấu đi sự phức tạp của điện tử bằng các khái niệm đơn giản, che đi sự phức tạp của phần mềm bằng các thủ tục ngắn gọn. Việc setup output cho 1 MCU bằng cách setup thanh ghi rõ ràng phức tạp đến độ người chuyên cũng phải lật datasheet ra xem, nhưng với Arduino thì chỉ cần gọi 1 hàm.

Bởi vì tính phổ biến và dễ dùng, với các thư viện được tích hợp sẵn. Ta chỉ cần quan tâm đến tính năng sản phẩm mà bỏ qua các tiêu tiết (protocol, datasheet ...) Nên giúp người không chuyên dễ dàng tiếp cận và làm ra các sản phẩm tuyệt vời mà không cần phải biết nhiều về điện tử.

Chính vì không quan tâm nhiều đến cách thức hoạt động của các Module đi kèm, nên đa phần người dùng sẽ khó xử lý được khi có các vấn đề phát sinh ngoài tầm của thư viện.

Các module prototype làm sẵn cho Arduino có độ bền không cao, mục tiêu đơn giản hóa quá trình làm sản phẩm.

### Các lợi ích khi sử dụng Arduino

Thiết kế IDE tốt, có thể dễ dàng tích hợp nhiều loại compiler, nhiều loại hardware mà không hề giảm hiệu năng. Ví dụ: Arduino gốc cho AVR, nhưng có nhiều phiên bản cho STM32, PIC32, ESP8266, ESP32... tận dụng tối đa các thư viện sẵn có.

Các thư viện được viết dựa trên lớp API trên cùng, nên đa số các thư viện cho Arduino có thể dùng được cho tất cả các chip. Diễn hình là Arduino cho ESP8266 có thể tận dụng trên 90% các thư viện cho Arduino khác.

Trình biên dịch cho Arduino là C/C++, khi biên dịch ESP8266 non-os SDK và ESP8266 Arduino cùng dùng chung trình biên dịch, hiệu năng không hề thua kém.

Cách tổ chức các thư viện C/C++ theo dạng OOP giúp phân lớp, kế thừa và quản lý cực kỳ tốt cho các ứng dụng lớn. Các MCU ngày càng mạnh mẽ và ứng dụng cho nó sẽ ngày càng lớn. Các mô hình quản lý code đơn giản trước đây (thuần C) sẽ khó. Các project cho Arduino đều opensource, ta dễ dàng lấy nó và đưa vào sản phẩm với chất lượng tốt và học hỏi được nhiều từ cách thức thiết kế chương trình.

Arduino chú trọng tính đa nền tảng, module hóa cao, phù hợp với các ứng dụng từ phức tạp tới cực kỳ phức tạp. Các ứng dụng kiểu này rất phổ biến trong thực tế. Nếu không dùng C++, hoặc arduino mà gặp vấn đề về overcontrol thì nên thử qua Arduino. Tiết kiệm được rất nhiều thời gian cho việc tập trung vào tính năng sản phẩm.

### Arduino cho ESP8266

Với Arduino ta có thể viết 1 Sketch sử dụng các thư viện và hàm tương tự của Arduino cho ESP8266.

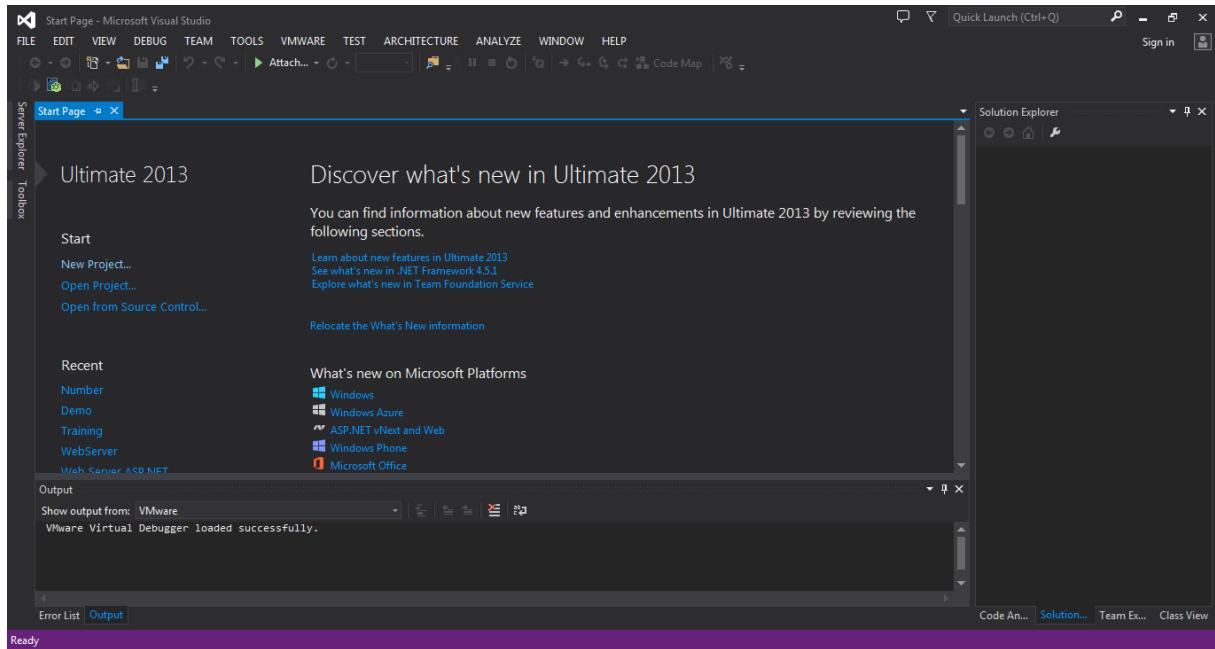
ESP8266 Arduino core đi kèm với thư viện kết nối Wifi hỗ trợ TCP, UDP và các ứng dụng HTTP, mDNS, SSDP, DNS Servers. Ngoài ra còn có thể thực hiện cập nhật OTA, sử dụng Filesystem dùng bộ nhớ Flash hay SD-card, điều khiển servo, ngoại vi SPI, I2C, ...

#### 4.4.4 Phần mềm lập trình cho Web

Nhóm sử dụng phần mềm Visual Studio để lập trình cho Web.

Visual Studio là IDE (Integrated Development Environments) có thể được sử dụng để xây dựng các dự án liên quan đến giải pháp phần mềm, ứng dụng và giao diện

người dùng, đồ họa. Hỗ trợ lập trình với nhiều ngôn ngữ như: C/C++, Visual Basic, C#, HTML, CSS, Javascript,...

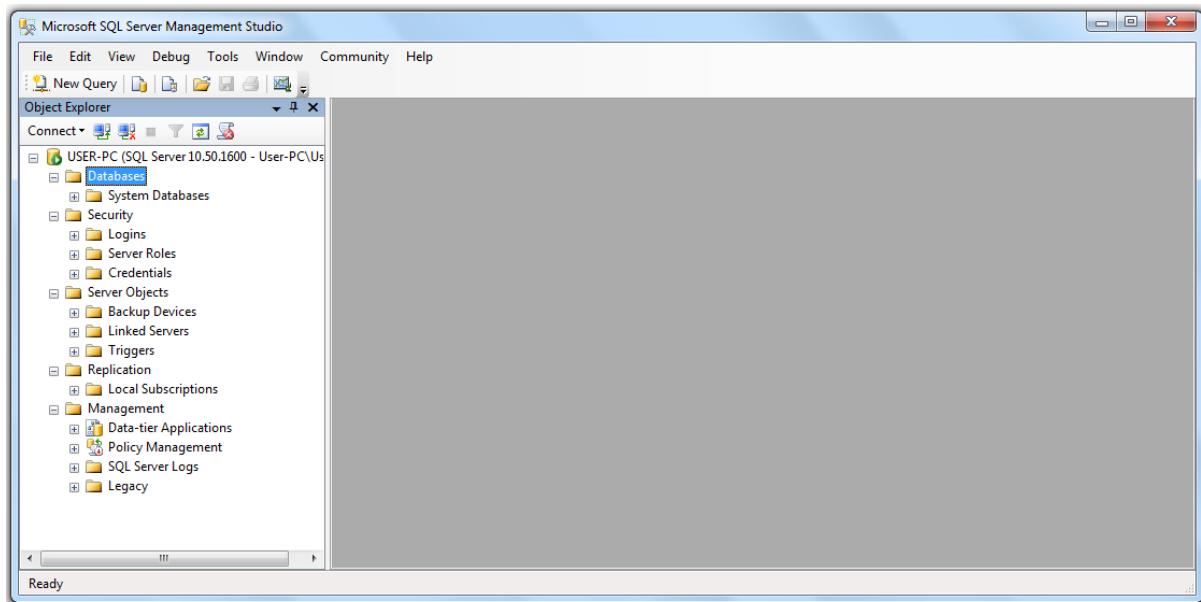


Hình 4.30: Phần mềm Visual Studio.

#### 4.4.5 Phần mềm xây dựng cơ sở dữ liệu

Nhóm sử dụng phần mềm Microsoft SQL Server để xây dựng cơ sở dữ liệu cho web server.

Microsoft SQL Server Management Studio là một công cụ trực quan dùng để quản lý SQL Server, người dùng có thể thực hiện được các tương tác với cơ sở dữ liệu (database) bằng câu lệnh hoặc trên giao diện người. SQL Server Management Studio được thiết kế đơn giản và dễ sử dụng nhưng người dùng cần có thời gian nhất định để làm quen với nó.

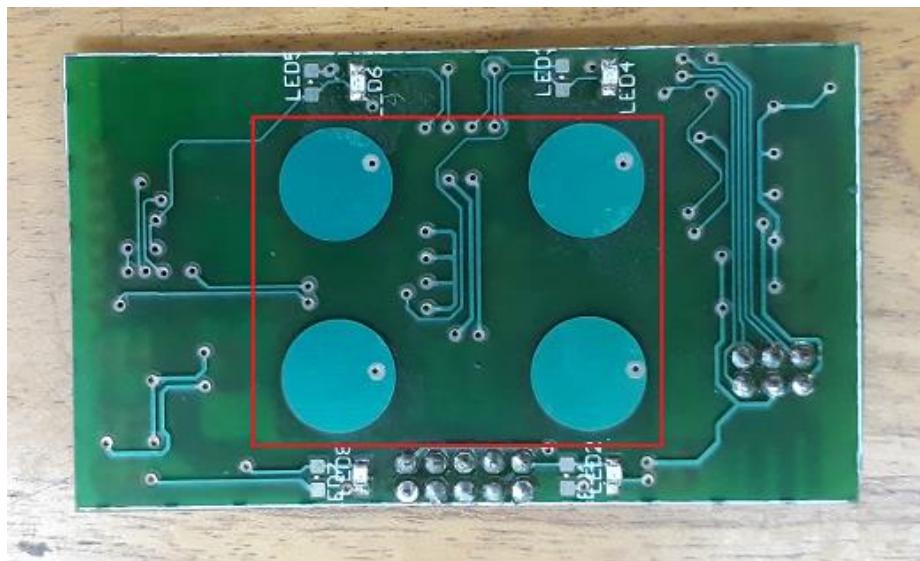


Hình 4.31: Phần mềm SQL Server Management Studio.

#### 4.5 HƯỚNG DẪN SỬ DỤNG, THAO TÁC

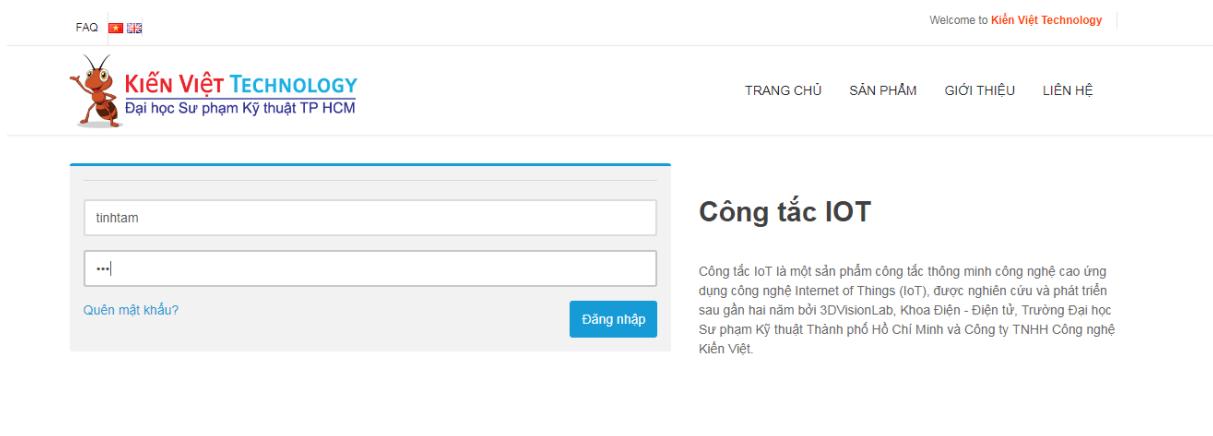
**Bước 1:** Kết nối công tắc IoT với các thiết bị cần điều khiển, tiến hành cấp nguồn 220VAC cho mô hình.

Sau khi thực hiện bước này, người dùng có thể điều khiển thiết bị trực tiếp thông qua các nút nhấn cảm ứng điện dung. Khi chạm vào nút nhấn điện dung thì thiết bị tương ứng sẽ được bật lên. Để tắt thiết bị người dùng cần chạm vào nút nhấn một lần nữa, dữ liệu trạng thái của thiết bị sẽ được gửi lên server và lưu trữ trên database.



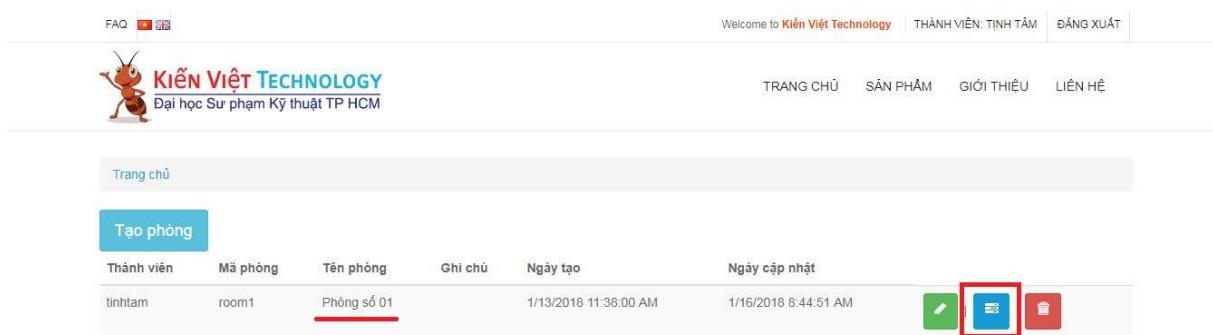
Hình 4.32: Các nút nhấn cảm ứng điện dung.

**Bước 2:** Truy cập vào web “tinhtam.baigiai.vn”, người dùng điền chính xác user và password. Với user: tinhtam, password: 123.



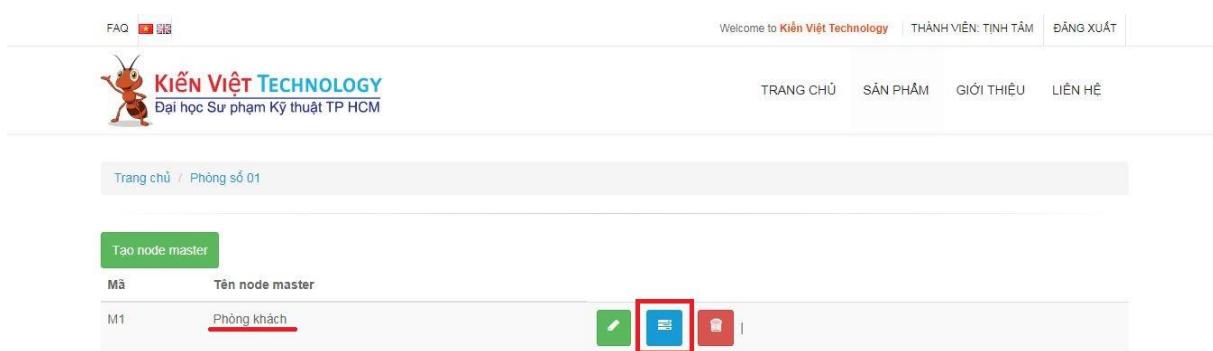
Hình 4.33: Giao diện đăng nhập vào web server.

Sau khi đăng nhập vào web, người dùng tiến hành chọn phòng cần điều khiển và giám sát. Ở đây có 1 phòng được tạo là phòng số 01. Click vào biểu tượng “ngăn tủ màu xanh” để chọn phòng số 01.



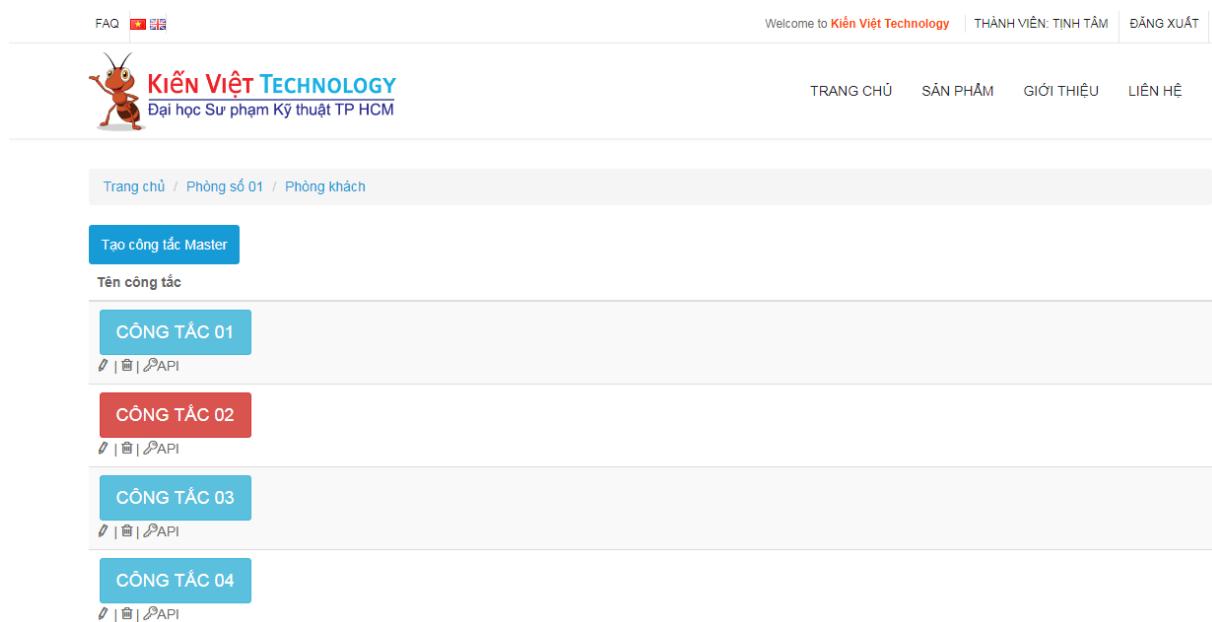
Hình 4.34: Phòng số 01.

Trong trang này, chọn vào phòng khách. Click vào biểu tượng “ngăn tủ màu xanh” để chọn phòng khách.



Hình 4.35: Phòng khách.

Tới đây người dùng đã vào trang điều khiển và giám sát thiết bị của phòng khách. Nút màu xanh tương ứng với trạng thái tắt của thiết bị, nút màu đỏ tương ứng với trạng thái bật của thiết bị. Người dùng tác động vào các nút nhấn trên web để điều khiển thiết bị. Trạng thái của các nút nhấn sẽ đồng bộ với trạng thái của thiết bị được điều khiển.



Hình 4.36: Giao diện điều khiển và giám sát thiết bị.

## Chương 5. KẾT QUẢ NHẬN XÉT ĐÁNH GIÁ

Mô hình được thực hiện từ tháng 9/2017 cho đến nay, trong quá trình thực hiện đã đạt được những kết quả sau:

### 5.1 MẠCH CẢM ỨNG ĐIỆN DUNG

Kết quả đạt được:

**Ưu điểm:**

- Mạch hoạt động ổn định, đạt những yêu cầu cơ bản, nút nhấn cảm ứng điện dung có thể tác động thông qua mặt mica, mặt kiếng với độ dày khoảng 15mm.
- Hiểu được cách hoạt động của cảm ứng điện dung, biết thêm nhiều linh kiện hỗ trợ chức năng cảm ứng điện dung như các dòng IC của Microchip, TI, ST,...
- Hiểu được cách thức hoạt động của IC cảm ứng điện dung AT42QT2120.

**Nhược điểm:**

- Khoảng cách tác động nút nhấn cảm ứng điện dung còn hạn chế, yêu cầu đề ra là 25mm nhưng khoảng cách tối đa chỉ đạt 15mm.
- Mạch hoạt động đôi khi không ổn định do nhiều tác động từ mạch nguồn, mạch công suất.
- Nút cảm ứng điện dung thiết kế với kích thước nhỏ, nên gây khó khăn cho người dùng khi tác động.

**Hướng phát triển:**

- Thiết kế cách đi dây, đặt mạch cảm ứng hợp lý hơn, cách xa và hạn chế tác động từ các nguồn gây nhiễu, xây dựng lồng Faraday chống nhiễu cho mạch.
- Có thể thay thế các loại chip khác, để nâng cao chất lượng của mạch cảm ứng điện dung (tăng khoảng cách tác động, tăng độ nhạy).

## 5.2 MẠCH XỬ LÝ TRUNG TÂM

Kết quả đạt được:

### **Ưu điểm:**

- Board mạch được thiết kế nhỏ gọn, hoạt động khá ổn định.
- Thiết kế và thi công hoàn thành mạch nạp có chức năng tự động chuyển chế độ hoạt động của ESP, giữa chế độ chạy chương trình và chế độ tiếp nhận chương trình mới khi người dùng tiến hành nạp. Mạch nạp được thiết kế riêng thành 1 pcb, nên khả năng sử dụng trở nên linh hoạt, tiết kiệm được chi phí, thay vì cứ 1 ESP sẽ có 1 mạch nạp tích hợp.
- Mạch xử lý trung tâm ESP xử lý được tín hiệu từ mạch cảm ứng điện dung, hoàn thành được yêu cầu cơ bản đề ra.
- Hiểu rõ hơn nguyên lý hoạt động, cách thức nạp của ESP.

### **Nhược điểm:**

- Mạch xử lý trung tâm ESP, đôi khi bị mất kết nối với wifi.

### **Hướng phát triển:**

- Thiết kế thêm phần chống nhiễu cho ESP bằng các mạch lọc, lồng Faraday,...
- Có thể thay thế ESP phiên bản mới hơn ESP32, có nhiều chân GPIO, tích hợp thêm điều khiển qua Bluetooth, RF. Tìm hiểu thêm các IC phục vụ cho các project IoT với giá thành hợp lý và chất lượng hơn so với ESP.

## 5.3 MẠCH CÔNG SUẤT

Kết quả đạt được:

### **Ưu điểm:**

- Mạch hoạt động đúng yêu cầu đề ra, đóng tắt thiết bị không gây ra tiếng ồn, ít bị nhiễu, không xảy ra hiện tượng tự kích của triac.
- Thiết kế mạch tương đối nhỏ gọn.
- Hiểu rõ nguyên lý hoạt động của mạch, các linh kiện được thêm vào với mục đích giảm nhiễu, chống hiện tượng tự kích.

### **Nhược điểm:**

- Mạch công suất dễ gây ra nhiễu cho mạch xử lý trung tâm.

**Hướng phát triển:**

- Thiết kế mạch có kích thước nhỏ gọn hơn, dùng linh kiện dán nhưng vẫn đảm bảo được các thông số kỹ thuật

**5.4 MẠCH NGUỒN**

Kết quả đạt được:

**Ưu điểm:**

- Mạch nguồn được thiết kế với kích thước tương đối nhỏ gọn 40x40mm, điện áp đầu ra đạt yêu cầu 3,3V, dòng cấp đạt khoảng 400mA.
- Thiết kế mạch nguồn với kích thước đường mạch phù hợp với áp và dòng đi qua, tạo được khoảng cách chống nhiễu, tản nhiệt cho mạch.
- Năm được nguyên lý và cơ chế hoạt động của các IC nguồn LNK3206G, MC34063.
- Hiểu thêm một số cách thức chống nhiễu, lọc nguồn cho mạch dùng mạch lọc nhiễu EMI, tụ bảo vệ, cuộn cảm lọc đầu ra,...
- Biết được nhiều loại nguồn chuyển đổi từ AC – DC, DC – DC.

**Nhược điểm:**

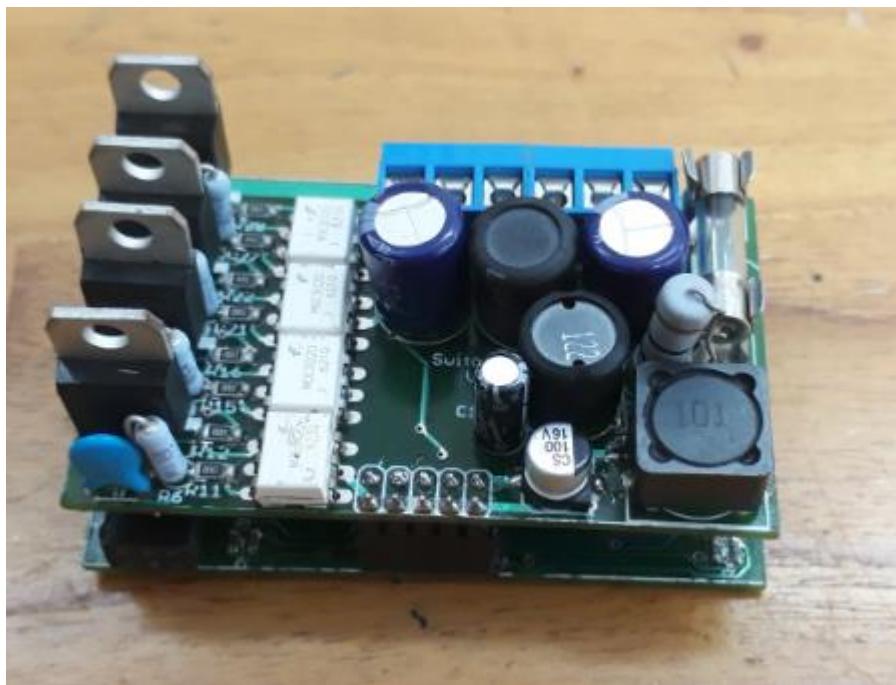
- Tuy có nhiều linh kiện lọc nhiễu, song với nguồn 220VAC không được sạch làm cho mạch vẫn có nhiều gai nhiễu ở đầu ra, ảnh hưởng tới cả hệ thống đặc biệt là phần mạch cảm ứng điện dung.
- Do không cách lưu hoàn toàn, nên vẫn ra hiện tượng rò điện.

**Hướng phát triển:**

- Tìm hiểu thiết kế mạch chống nhiễu tối ưu hơn.
- Đóng hộp cách ly nguồn với các phần khác trong mạch, để chống nhiễu và tránh rò điện.

## 5.5 KẾT QUẢ MÔ HÌNH

Hình ảnh kết quả đạt được sau thời gian thực hiện:



Hình 5.1: Mô hình hoàn chỉnh sau thời gian thực hiện.

Sau 2 lần thi công, cuối cùng nhóm cũng đã hoàn thành được mô hình. Kết quả sau khi kiểm tra, đạt được 90% yêu cầu đề ra.

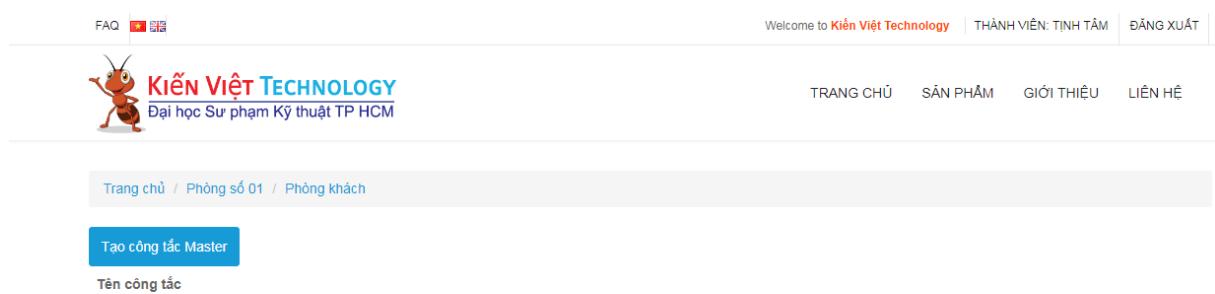
## 5.5 KẾT QUẢ PHẦN MỀM

Sau 4 tháng thực hiện đồ án, nhóm đã biết được một số kiến thức nền tảng về việc lập trình web server: HTML, CSS, Javascript,...

Lập trình kết nối ESP với server trên phần mềm Arduino, giúp ESP có thể nhận dữ liệu điều khiển từ server để điều khiển và gửi tín hiệu phản hồi lại cho server.

Thiết kế được giao diện web server để điều khiển thiết bị, giám sát trạng thái đóng/tắt trên web server.

Hình 5.2: Giao diện web server.



Hình 5.3: Giao diện điều khiển thiết bị của web server.

Đề tài thực hiện là sự liên kết giữa điện tử và công nghệ thông tin đưa việc điều khiển các thiết bị lên tầm cao hơn, linh hoạt và tiện tích hơn.

Đến thời điểm hiện tại, nhóm đã hoàn thành được 90% mục tiêu đã đặt ra. Mô hình hoạt động tương đối ổn định.

## Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 6.1 KẾT LUẬN

Nhóm đã thi công được mô hình, có chức năng tương tự như các công tắc thông minh trên thị trường. Có server quản lý điều khiển và giám sát trạng thái đóng tắt của thiết bị. Thiết kế mô hình tương đối nhỏ gọn phù hợp với hệ thống điện trong gia đình. Tạo được server quản lý và lưu trữ dữ liệu. Xây dựng giao diện web thân thiện với người dùng.

Phần cứng đạt hơn 90% yêu cầu đề ra song vẫn còn nhiều tác động, thiết kế hộp mang tính tượng trưng, chưa phù hợp với thực tế. Phần mềm đạt được 90% dự tính ban đầu, đáp ứng được những yêu cầu chính đề ra xây dựng được server, cơ sở dữ liệu, truyền và nhận dữ liệu giữa thiết bị và server. Chưa xây dựng được các tính năng phát triển cho sản phẩm như: hẹn giờ, thiết lập bối cảnh bật tắt thiết bị,...

Nhìn chung mô hình đạt được những yêu cầu chính đặt ra. Khả năng ổn định hệ thống phụ thuộc vào nguồn điện cấp 220VAC và tốc độ truy cập của mạng, công tắc tính thẩm mỹ chưa cao, chưa xây dựng tính bảo mật hệ thống.

### 6.2 HƯỚNG PHÁT TRIỂN

Mô hình hiện tại chỉ thực hiện được việc đóng tắt thiết bị, có thể phát triển thêm phần điều chỉnh điện áp thiết bị, ứng dụng điều chỉnh độ sáng tối đèn, điều khiển tốc độ động cơ,... Xây dựng thêm mạch kiểm tra tải kết nối với công tắc có còn hoạt động hay không thông qua cảm biến dòng.

Phần web server cần xây dựng thêm nhiều chức năng như hẹn giờ đóng tắt thiết bị, thiết lập bối cảnh sinh hoạt dựa vào lựa chọn thiết bị bật tắt. Xây dựng hệ thống bảo mật cho web server.

## TÀI LIỆU THAM KHẢO

### Nguồn Internet:

[1] “Cảm ứng điện dung là gì?”.

Link: <https://fptshop.com.vn/tin-tuc/danh-gia/cam-ung-dien-dung-la-gi-35632>

[2] “Datasheet của AT42QT2120.

Link: [ww1.microchip.com/downloads/en/DeviceDoc/doc9634.pdf](http://ww1.microchip.com/downloads/en/DeviceDoc/doc9634.pdf)

[3] “Datasheet của BTA12”.

Link: [www.st.com/resource/en/datasheet/t12xx.pdf](http://www.st.com/resource/en/datasheet/t12xx.pdf)

[4] “Datasheet của MOC3020”. Link:

<http://www.alldatasheet.com/datasheet-pdf/pdf/176764/FAIRCHILD/MOC3020M.html>

[5] “IoT dành cho người mới bắt đầu”. Link: <https://iotmakervn.github.io/iot-starter-book/>

[6] “Hướng dẫn sử dụng ESP8266 trong các ứng dụng Internet Of Things”. Link:

<http://htelelectronics.vn/huong-dan-su-dung-esp8266-trongcac-ung-dung-internet-of-things/>

[7] “Datasheet của ESP8266”. Link:

[https://cdn-shop.adafruit.com/product-files/2471/0A-ESP8266\\_Datasheet\\_EN\\_v4.3.pdf](https://cdn-shop.adafruit.com/product-files/2471/0A-ESP8266_Datasheet_EN_v4.3.pdf)

[8] “Giao tiếp UART”. Link: <http://www.hocavr.com/index.php/lectures/uart>

[9] “Datasheet của CH340G”. Link:

<https://cdn.sparkfun.com/datasheets/Dev/Arduino/Other/CH340DS1.PDF>

[10] “Datasheet LNK3206G”. Link:

[https://ac-dc.power.com/sites/default/files/product-docs/linkswitch-tn2\\_family\\_datasheet.pdf](https://ac-dc.power.com/sites/default/files/product-docs/linkswitch-tn2_family_datasheet.pdf)

[11] “Datasheet của MC34063”. Link:

<https://www.onsemi.com/pub/Collateral/MC34063A-D.PDF>

[12] “Web server là gì?”. Link: <https://techmaster.vn/posts/34420/web-server-la-gi>

[13] “Tìm hiểu cơ sở dữ liệu là gì?”. Link:

<https://freetuts.net/tim-hieu-co-so-du-lieu-la-gi-va-he-quan-tri-cSDL-mysql-168.html>

[14] "Có những loại database nào?". Link: <https://hostingviet.vn/co-so-du-lieu-database-la-gi>

[15] "Datasheet của AMS1117 – 3,3V". Link:

[www.advanced-monolithic.com/pdf/ds1117.pdf](http://www.advanced-monolithic.com/pdf/ds1117.pdf)

[16] "Lập trình web". Link: [webcoban.vn/](http://webcoban.vn/)

[17] "Diễn đàn điện tử Việt Nam. Link: [www.dientuvietnam.net/](http://www.dientuvietnam.net/)

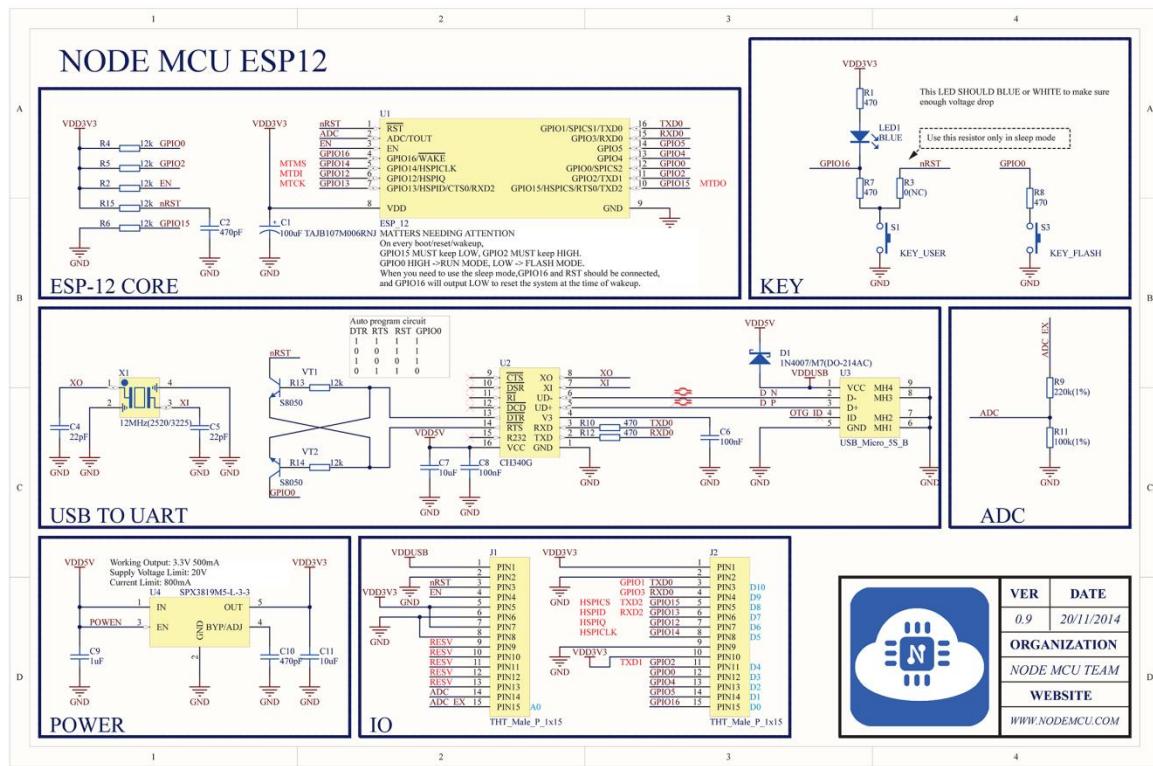
[18] "Datasheet của các linh kiện khác". Link: <http://www.alldatasheet.com>

### **Sách tham khảo**

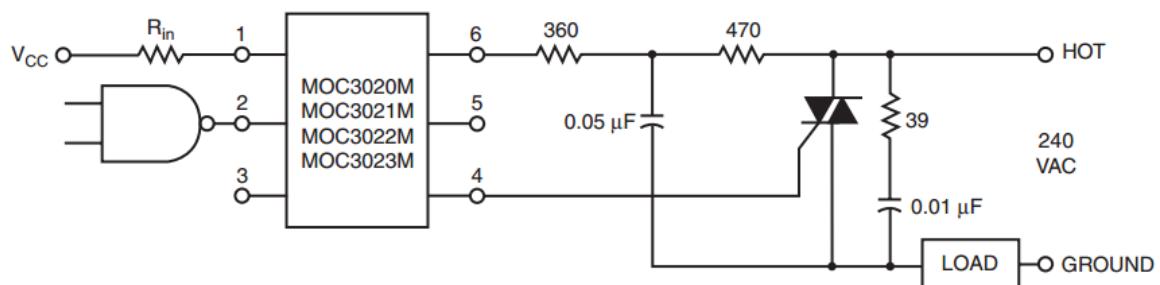
[19] Hoàng Ngọc Văn, "Giáo trình điện tử công suất" – ĐH Sư phạm kỹ thuật TP Hồ Chí Minh.

## PHỤ LỤC

[1i] Sơ đồ nguyên lý của module NodeMCU ESP12



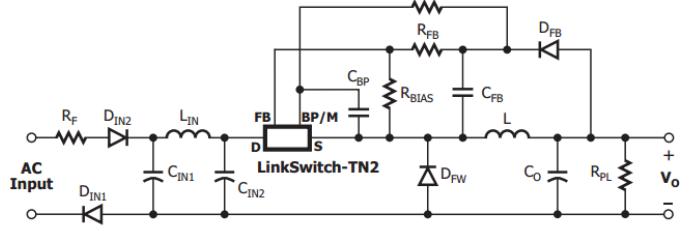
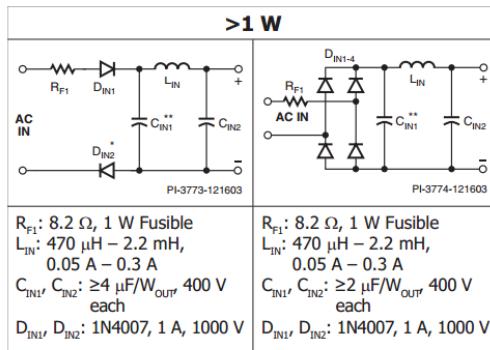
[2i] Sơ đồ mạch RC của MOC3020



## [3i] Tài liệu thiết kế mạch nguồn hạ áp LNK3206G

$V_{OUT}$	$I_{OUT(MAX)}$	Inductor			LNK320X	Mode	Diode $t_{RR}$	$R \times FB$	$V_z$
		$\mu H$	T <sub>RMS(mA)</sub>	Tokin					
12	≤63 80	2000 2400	88 84	– –	LNK3202	MDCM CCM	≤ 75 ns ≤ 35 ns	11.8 k	11 V
	85 120 160	870 870 1500	152 151 167	SBC4-152-251 RFB0810-152	LNK3204	MDCM MDCM CCM	≤ 75 ns ≤ 75 ns ≤ 35 ns		
	175 225	870 870	265 236	– –	LNK3205	MDCM CCM	≤ 75 ns ≤ 35 ns		
	175 225	680 1200	336 364	SBC6-681-431 RFB0810-681 RFB1010-122	LNK3206	MDCM CCM	≤ 75 ns ≤ 35 ns		

Other Standard Components

 $R_{BIAS}$ : 2.49 kΩ, 1%, 1/8 W $C_{BP}$ : 0.1 μF, 50 V Ceramic $C_{FB}$ : 10 μF, 1.25 ×  $V_o$  $D_{FB}$ : 1N4005GP $R_z$ : 470 Ω to 2 kΩ, 1/8 W, 5%

## [4i] Chương trình cho ESP8266:

```
#include <ESP8266WiFi.h>
#include "ESP8266WebServer.h"
#include "ESP8266HTTPClient.h"
#include "time.h"
//Khai bao bien
String ESP_name = "M1";
String ESP_CT[4] = {"SS01", "SS02", "SS03", "SS04"};
String link_M1 = "http://tinhtam.baigiai.vn/api/CongTacMaster?ma=M1";
String link_SS11 = "http://tinhtam.baigiai.vn/api/CongTacSlave?ma=SS11";
String link_SS22 = "http://tinhtam.baigiai.vn/api/CongTacSlave?ma=SS22";
String link_POST_M = "http://tinhtam.baigiai.vn/api/CongTacMaster";
String link_POST_S = "http://tinhtam.baigiai.vn/api/CongTacSlave";
//String link_POST_S = "http://plugiot.baigiai.vn/api/CongTacSlave";

int httpCode = 0;
int CodeLag = 0;
const char* ssid = "DASAN";
const char* password = "12345678";
char* host = "192.168.1.102";
const int port = 2019;
IPAddress local_IP(192,168,2,1);
IPAddress gateway(192,168,2,1);
IPAddress subnet(255,255,255,0);
#define MAXSCC 4
```

```

String jsData =
""; // {ID:'0', MaCamBien:'null', GiaTri:'0', ThoiGian:'20/12/2017
10:15:00', UserID:'0', CamBien:'null'}";
// Cac bien JSON
//=====
String MessageSV = ""; //Du lieu nhan tu SERVER
String MessageSV_M1 = "";
String MessageSV_M1_old = "";
String MessageSV_SS11 = "";
String MessageSV_SS11_old = "";
String MessageSV_SS22 = "";
String MessageSV_SS22_old = "";
String Message[MAXSC] = {""};
String MessageClient[MAXSC] = {""}; //Du lieu nhan ve tu cac SSxx
WiFiClient client;
WiFiClient AP_Client[MAXSC];
WiFiServer AP_Server(local_IP, 2018);
//bien xu ly thoi gian
unsigned long time3 = 0, time2 = 0, time1 = 0;
time_t now;
struct tm * timeinfo;
String nam, thang, ngay, gio, phut, giay, sthoigian="";
//Bien xu ly trang thai nhan
String z1="1", z2="2", z3="3", z4="4";
boolean led1 = false, led2 = false, led3 = false, led4 = false;
boolean L1=false, L2=false, L3=false, L4=false;
boolean WEBsend = false; //trang thai du lieu gui tu WEB
boolean WEBLed = false; //
//Khai bao nguyen mau ham
void joinWF();
void createWF();
void joinSV();
void createSV();
void AP_checkSTA(void);
void sendSV(void);
String receiveSV(void);
void sendClient(String data);
void receiveClient(void);
int Post_data(String link, String data);
String Get_data(String Link);
String Str2JSON(String data); // Ham nay chuyen kieu #M1,SS1x,1 sang kieu
JSON
String JSON2Str(String data); // Ham nay thuc hien sau ham GET_data de
chuyen JSON thanh &M1,SS1x,1
void XLString(String data); //kiem tra gia tri nhan ve tu server de dieu
khien thiet bi
void getTime(void);
String thoigian(void);
void control1(void);
void control2(void);
void control3(void);
void control4(void);
void KT_NutNhan(void);
//ham chinh chuong trinh
void setup() {
    pinMode(5, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(2, OUTPUT);
    pinMode(0, OUTPUT);

    pinMode(12, INPUT_PULLUP);

```

```

pinMode(13, INPUT_PULLUP);
pinMode(14, INPUT_PULLUP);
pinMode(15, INPUT_PULLUP);
attachInterrupt(14, control1, RISING);
attachInterrupt(12, control2, RISING);
attachInterrupt(15, control3, RISING);
attachInterrupt(13, control4, RISING);

Serial.begin(115200);
WiFi.mode(WIFI_AP_STA);
joinWF();
createWF();
// joinSV();
createSV();
client.println(ESP_name + " Xin chao!");
getTime();
sthogian = thoigian();
jsData = "";
time3 = time2 = time1 = millis();
}

void loop() {
    time1 = millis();
    if((unsigned long)(time1 - time2) >= 15000){
        time2 = millis();
        if(WiFi.status() != WL_CONNECTED){
            joinWF();
        }
        // joinSV();
    }
    // MessageSV_SS11 = Get_data(link_SS11);
    // Serial.println(MessageSV_SS11);
}
if((unsigned long)(time1 - time3) >= 1000){
    time3 = millis();
    sthogian = thoigian();
}
AP_checkSTA();
receiveClient();
//Doan lenh gui va nhan du lieu tu SERVER
sendSV(link_POST_S); //Gui theo dang socket
//Nhan du lieu va gui ve Clien
MessageSV_SS11 = Get_data(link_SS11);
if(MessageSV_SS11_old != MessageSV_SS11){
    MessageSV_SS11_old = MessageSV_SS11;
    MessageSV = JSON2Str(MessageSV_SS11);
    if(MessageSV.length() != 0){
        Serial.println("Gui Clients: "+ MessageSV);
        sendClient(MessageSV);
        MessageSV = "";
    }
}
MessageSV_SS22 = Get_data(link_SS22);
if(MessageSV_SS22_old != MessageSV_SS22){
    MessageSV_SS22_old = MessageSV_SS22;
    MessageSV = JSON2Str(MessageSV_SS22);
    if(MessageSV.length() != 0){
        Serial.println("Gui Clients: "+ MessageSV);
        sendClient(MessageSV);
        MessageSV = "";
    }
}
//=====

```

```

//Nhận dữ liệu từ Server M1 và xử lý
MessageSV_M1 = Get_data(link_M1);
delay(1);
if(MessageSV_M1_old != MessageSV_M1){
    MessageSV_M1_old = MessageSV_M1;
    MessageSV_M1 = Json2Str(MessageSV_M1);
    XLString(MessageSV_M1);
}
KT_NutNhan();
delay(1);
}

void joinWF(){
unsigned int ilag = 0;
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
    ilag++;
    if(ilag > 20){
        ilag = 0;
        break;
    }
}
if(WiFi.status() == WL_CONNECTED){
    Serial.print("\nDia chi IP: ");
    Serial.println(WiFi.localIP());
}
else{
    Serial.println("\nKhong the ket noi");
}
}

void createWF(){
WiFi.softAPConfig(local_IP, gateway, subnet);
delay(500);
WiFi.softAP("M1","12345678");
delay(500);
Serial.println(WiFi.softAPIP());
Serial.println("Tao AP_"+ESP_name+" xong");
}

void joinSV(){
unsigned int ilag = 0;
while(!client.connect(host, port)){
    Serial.print(".");
    delay(100);
    ilag++;
    if(ilag > 30){
        ilag = 0;
        break;
    }
}
if(client.connected()){
    Serial.println("Ket noi " + String(host) + " thanh cong");
}
else{
    Serial.println("Khong the ket noi");
}
}

void createSV(){
AP_Server.begin();
AP_Server.setNoDelay(true);
Serial.println("Tao SV_"+ESP_name+" xong");
}

```

```

}

void AP_checkSTA(void){
    if (AP_Server.hasClient()){ //kiem tra xem co note nao ket noi
        for(uint8_t i = 0; i < MAXSC; i++){ //do cac phan tu trong mang
APClient
        //Neu Client co hoac dang ket noi thi bo qua kiem tra
        if (!AP_Client[i] || !AP_Client[i].connected()){
            // Kiem tra Note da tung ket noi chua
            if(AP_Client[i]){
                AP_Client[i].stop();
            }
            // Kiem tra Note da ket noi toi server
            if(AP_Client[i] == AP_Server.available()){
                Serial.println("Ket noi moi: " + String(i));
                //lay luon IP cua clients
            }
            // tiep tuc do tim
            continue;
        }
    }
    //Neu da het khoan trong ket noi thi huy phien hien tai
    WiFiClient APClient = AP_Server.available();
    APClient.stop();
}
else{
    //Serial.println("Khong co ket noi moi.");
}
}

//Nhan du lieu tu Socket server
String receiveSV(void){
    String data = "";
    if(client.connected()){
        while(client.available()){
//            MessageSV = client.readStringUntil('\r'); //ESP_name = M2 mau:
&M1,SS1x,1
            data = client.readStringUntil('\r'); //ESP_name = &M1,SS1x,1
            if(data.length() != 0){
                if(data.charAt(2) == '1'){
                    Serial.println("M1 nhan dc tu server:" + data);
                }
                else{
                    Serial.println("Du lieu sai dia chi:" + data);
                }
            }
            client.flush();
        }
    }
    return data;
}
void sendClient(String data){
    for(uint8_t i = 0; i < MAXSC; i++){
        if(AP_Client[i].connected()){
            Message[i] = data;
            if(Message[i].length() != 0){
                AP_Client[i].println(Message[i]);
                Serial.println("Gui cho " + String(i) + ":" + Message[i]);
                Message[i] = "";
            }
        }
    }
}
}

```

```

void sendSV(String link){
    String ESP_SS = "";
    for(uint8_t i = 0; i < MAXSC; i++){
        if(MessageClient[i].length() != 0){
            CodeLag = 0; // kiem tra ma tra ve
            ESP_SS = MessageClient[i];
            ESP_SS.remove(8);
            //Chuyen du lieu tu #M1,SS1x,1 sang kieu JSON roi moi gùi
            MessageClient[i] = Str2JSon(MessageClient[i]);

            Serial.println("Gui len SERVER");
            CodeLag = Post_data(link_POST_S,MessageClient[i]);
            Serial.println(MessageClient[i]);
            MessageClient[i] = "";
        }

        //kiem tra neu server tra ve oke thi xoa du lieu cu
    }
}
}

void receiveClient(void){
    for(uint8_t i = 0; i < MAXSC; i++){
        if (AP_Client[i] && AP_Client[i].connected() &&
AP_Client[i].available()){
            while(AP_Client[i].available()){
                MessageClient[i] = AP_Client[i].readStringUntil('\r'); //luu du
lieu vao bien tam
                AP_Client[i].flush(); // xoa du lien tai phien hien tai
                Serial.println("Client " + String(i) + ": " + MessageClient[i]);
            }
        }
    }
}

int Post_data(String link, String data){
    httpCode = 0;
    if(WiFi.status() == WL_CONNECTED){
        HTTPClient http;
        // http.begin("http://tinhtam.baigiai.vn/api/CamBiens");
        http.begin(link);
        http.addHeader("Content-Type", "application/json");
        httpCode = http.POST(data); //Ney tra ve 201 hoac 200 laf gui thanh
cong
        Serial.println(httpCode);
        http.end(); //Close connection
    }
    else{
        Serial.println("Khong the ket noi server");
    }
    return httpCode;
}

String Get_data(String link){
    httpCode=0;
    String payload = "";
    if(WiFi.status() == WL_CONNECTED){
        HTTPClient http;
        http.begin(link); //Dia chi nhan du lieu
//        http.begin("http://tinhtam.baigiai.vn/api/CongTacSlave?ma=S11");
        //Dia chi nhan du lieu
        httpCode = http.GET();
        //Serial.println(httpCode);
        if (httpCode > 0){


```

```

        payload = http.getString(); //Gia tri tra ve
    }
    http.end(); //Close connection
}
return payload;
}

//Thuc hien sau ham GET >> JSON2Str >> XLString
void XLString(String data){ //"/&M1,SSxx,xxxx"
    if(data.charAt(0) == '&'){ //Kiem tra phai la chuoi dieu khien khong?
        String MyString = data;
        String MyLed1 = String(led1);
        String MyLed2 = String(led2);
        String MyLed3 = String(led3);
        String MyLed4 = String(led4);
        String MyLed = MyLed1+MyLed2+MyLed3+MyLed4;
        MyString.remove(0,9); // bo &M1,SSxx, cua &M1,SSxx,xxxx
        if(!(MyString.equalsIgnoreCase(MyLed))){ //Co du lieu dc gui tu WEB
            Serial.println("Led 1");
            control1();
        }
        if(String(MyString.charAt(1)) != MyLed2){
            Serial.println("Led 2");
            control2();
        }
        if(String(MyString.charAt(2)) != MyLed3){
            Serial.println("Led 3");
            control3();
        }
        if(String(MyString.charAt(3)) != MyLed4){
            Serial.println("Led 4");
            control4();
        }
    }
}
}

//khu vuc cap nhat thoi gian
void getTime(){
    configTime(7 * 3600, 0, "pool.ntp.org", "time.nist.gov");
    Serial.println("Cho lay thoi gian");
    while (!time(nullptr)) {
        Serial.print(".");
        delay(1000);
    }
    time(&now);
    timeinfo = localtime(&now);
    Serial.println(asctime (timeinfo));
}

String thoigian(){
    String nowtime = "";
    time(&now);
    //Chuyen Giay lây dc thanh câu trúc date time
    timeinfo = localtime(&now);

    nam = String((timeinfo->tm_year) + 1900);
    if((timeinfo->tm_mon) < 9){
        thang = "0"+ String(timeinfo->tm_mon +1);
    }
    else{
        thang = String(timeinfo->tm_mon +1);
    }
}

```

```

    }
    if((timeinfo->tm_mday) < 10){
        ngay = "0"+ String(timeinfo->tm_mday);
    }
    else{
        ngay = String(timeinfo->tm_mday);
    }
    if((timeinfo->tm_hour) < 10){
        gio = "0"+ String(timeinfo->tm_hour);
    }
    else{
        gio = String(timeinfo->tm_hour);
    }
    if((timeinfo->tm_min) < 10){
        phut = "0"+ String(timeinfo->tm_min);
    }
    else{
        phut = String(timeinfo->tm_min);
    }
    if((timeinfo->tm_sec) < 10){
        giay = "0"+ String(timeinfo->tm_sec);
    }
    else{
        giay = String(timeinfo->tm_sec);
    }
    nowtime =  ngay + "-" + thang + "-" + nam + " " + gio + ":" + phut + ":" +
+ giay;
    return nowtime;
}
void control1()
{
    L1 = led1;
    led1 = !led1;
}
void control2()
{
    L2 = led2;
    led2 = !led2;
}
void control3()
{
    L3=led3;
    led3 = !led3;
}
void control4(){
    L4=led4;
    led4 = !led4;
}
void KT_NutNhan(){
    boolean Flag = false;
    if(L1 != led1){
        L1 = led1;
        Flag = true;
        digitalWrite(5, led1);
    }
    if(L2 != led2){
        L2 = led2;
        Flag = true;
        digitalWrite(2, led2);
    }
    if(L3 != led3){

```

```

L3 = led3;
Flag = true;
digitalWrite(0, led3);
}
if(L4 != led4){
L4 = led4;
Flag = true;
digitalWrite(4, led4);
}
if(Flag){
jsData =
"\\"0,tinhtam,room1,M1,, "+String(led1)+String(led2)+String(led3)+String(led4
)+"+", "+sthoigian+"";
}

if(WEBsend){ //True la gui tu web
WEBsend = false;
Serial.println("From WEB");
}
else{
Post_data(link_POST_M,jsData);
}
Serial.println(jsData);
Flag = false;
}
}

String Str2Json(String data){
// #M1,SSxx,xxxx
String jsString = "";
String ESP_send = data;
String ESP_Name = "";
if(data.length() != 0){
ESP_send.remove(0,1); // M1,SSxx,xxxx
jsString = "\\"0,tinhtam,room1,"+ESP_send+", "+ sthoigian+"";
}
return jsString;
}
String JSON2Str(String data){
// dang tra ve: "1,room1,M1,SSxx,xxxx" vd: "1,tinhtam,room1,M1,S11,010" //
"1,tinhtam,room1,M1,,0001"
String MyESP = data;
String MyCT = "";
int index = 0;
MyESP.remove(0,17); //M1,,0001"
index = MyESP.lastIndexOf('\'');
MyESP.remove(index); //M1,,0001
MyCT = MyESP;
index = MyESP.lastIndexOf(',');
MyCT.remove(0,index +1); //0001
MyESP.remove(index+1); //M1,
if(data.charAt(1) == '1'){
if(MyESP == "M1,,"){
MyESP = "&M1,SSxx," + MyCT;
Serial.println("M1: "+MyESP);
}
else{
MyESP += MyCT;
MyESP = "&" + MyESP;
Serial.println("SS: "+MyESP);
}
}
return MyESP;
}

```

}

**[5i] Chương trình cho web server:**

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <meta http-equiv="refresh" content="5">
    <title>Công tắc IOT</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.0/jquery.min.js"></sc
ript>
    <!-- CORE CSS -->
    <link href="/assets1/plugins/bootstrap/css/bootstrap.min.css"
rel="stylesheet" type="text/css" />
    <!-- THEME CSS -->
    <link href="/assets1/css/essentials.css" rel="stylesheet"
type="text/css" />
    <link href="/assets1/css/layout.css" rel="stylesheet" type="text/css"
/>

    <!-- PAGE LEVEL SCRIPTS -->
    <link href="/assets1/css/header-1.css" rel="stylesheet" type="text/css"
/>
    <link href="/assets1/css/color_scheme/blue.css" rel="stylesheet"
type="text/css" id="color_scheme" />
</head>
<body class="smoothscroll enable-animation">
    <!-- Top Bar -->
    <div id="topBar">
        <div class="container">
            <!-- right -->
            <ul class="top-links list-inline pull-right">
                <li class="text-welcome hidden-xs">Welcome to <strong
style="color: #ff4500">Kiến Việt Technology</strong></li>
                <li>
                    <a href="/Profiles/Edit/tinhtam">Thành viên: Tịnh
T&#226;m</a></li>
                    <li><a href="/Home/DangXuat">Đăng xuất</a></li>
                </li>
            </ul>
            <!-- left -->
            <ul class="top-links list-inline">
                <li><a href="#">FAQ</a></li>
                <li>
                    
                    
                </li>
            </ul>
        </div>
    </div>
    <!-- /Top Bar -->
    <div id="header" class="sticky clearfix">
        <!-- TOP NAV -->
        <header id="topNav">
            <div class="container">
                <!-- Mobile Menu Button -->
                <button class="btn btn-mobile" data-toggle="collapse" data-
target=".nav-main-collapse">

```

```

        <i class="fa fa-bars"></i>
    </button>
    <!-- Logo -->
    <a class="logo pull-left" href="/ThanhVien/Index"
target=_parent">
        
    </a>
    <div class="navbar-collapse pull-right nav-main-collapse
collapse">
        <nav class="nav-main uppercase">
            <ul id="topMain" class="nav nav-pills nav-main">
                <li>
                    <a href="/ThanhVien/Index"
target=_parent">
                        Trang Chủ
                    </a>
                </li>
                <li>
                    <a href="#" target=_blank">
                        Sản Phẩm
                    </a>
                </li>
                <li>
                    <a href="#" target=_blank">
                        Giới Thiệu
                    </a>
                </li>
                <li>
                    <a href="#" target=_blank">
                        Liên Hệ
                    </a>
                </li>
            </ul>
        </nav>
    </div>
</header>
<!-- /Top Nav -->
</div>
<p></p>
<div class="container">
<ol class="breadcrumb">
    <li><a href="/ThanhVien/index">Trang chủ</a></li>
    <li><a href="/ThanhVien/ChiTietPhong?ma=room1">Phòng số
01</a></li>
    <li><a class="active">Phòng khach</a></li>
</ol>
<button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#Modal_congtacmaster">Tạo công tắc Master</button>
<div class="modal fade" id="Modal_congtacmaster">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal"
aria-hidden="true">&times;</button>
                <h4 class="modal-title">Tạo công tắc Master </h4>
            </div>
            <form method="post" action="#" id="frm_taocontacmaster">
                <div class="modal-body">
                    <div id="err"></div>

```

```

        <div class="form-group">
            <label class="control-label">Tên công tắc</label>
            <input type="hidden" class="form-control"
id="ma_nodemaster" name="ma_nodemaster" value="M1" />
            <input type="text" class="form-control"
id="tencongtaclmaster" name="tencongtaclmaster" />
        </div>
    </div>
    <div class="modal-footer">
        <button type="submit" class="btn btn-primary"
id="login_btn"><i class="icon-save"></i>Lưu thông tin</button>
        <button type="button" class="btn btn-default" data-
dismiss="modal">Đóng</button>
    </div>
</form>
</div><!-- /.modal-content -->
</div><!-- /.modal-dialog -->
</div><!-- /.modal -->
<table class="table table-striped" style="width:100%">
    <thead>
        <tr>
            <th>Tên công tắc</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>
                <button rel="3" id="btnmaster3" class="btn_
master
btn btn-info btn-lg uppercase ">C&#244;ng tắc 01</button>
                <br/>
                <a href="/ThanhVien/SuaCongTacMaster?ma=3"><i
class="icon-pencil"></i></a> |
                <a href="" onclick="XoaCongTacMaster(3);return
false"><i class="icon-trash"></i></a> |
                <a href="/api/CongTacMaster?ma=M1"
target="_blank"><i class="icon-key"></i>API</a>
            </td>
        </tr>
        <tr>
            <td>
                <button rel="4" id="btnmaster4" class="btn_
master
btn btn-info btn-lg uppercase ">C&#244;ng tắc 02</button>
                <br/>
                <a href="/ThanhVien/SuaCongTacMaster?ma=4"><i
class="icon-pencil"></i></a> |
                <a href="" onclick="XoaCongTacMaster(4);return
false"><i class="icon-trash"></i></a> |
                <a href="/api/CongTacMaster?ma=M1"
target="_blank"><i class="icon-key"></i>API</a>
            </td>
        </tr>
        <tr>
            <td>
                <button rel="7" id="btnmaster7" class="btn_
master
btn btn-info btn-lg uppercase ">C&#244;ng tắc 03</button>
                <br/>
                <a href="/ThanhVien/SuaCongTacMaster?ma=7"><i
class="icon-pencil"></i></a> |
                <a href="" onclick="XoaCongTacMaster(7);return
false"><i class="icon-trash"></i></a> |
            </td>
        </tr>
    </tbody>
</table>

```

```

        <a href="/api/CongTacMaster?ma=M1"
target="_blank"><i class="icon-key"></i>API</a>
        </td>
    </tr>
    <tr>
        <td>
            <button rel="8" id="btnmaster8" class="btn_master
btn btn-info btn-lg uppercase ">C&#244;ng tăc 04</button>
            <br/>
            <a href="/ThanhVien/SuaCongTacMaster?ma=8"><i
class="icon-pencil"></i></a> |
            <a href="" onclick="XoaCongTacMaster(8);return
false"><i class="icon-trash"></i></a> |
            <a href="/api/CongTacMaster?ma=M1"
target="_blank"><i class="icon-key"></i>API</a>
            </td>
        </tr>
    </tbody>
</table>
<hr />
<button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#Modal_NodeSlave">Tạo node slave</button>
<div class="modal fade" id="Modal_NodeSlave">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal"
aria-hidden="true">&times;</button>
                <h4 class="modal-title">Tạo node slave</h4>
            </div>
            <form method="post" action="#" id="frm_taonodeslave">
                <div class="modal-body">
                    <div id="err"></div>
                    <input type="hidden" class="form-control"
id="ma_nodemaster" name="ma_nodemaster" value="M1" />
                    <div class="form-group">
                        <label class="control-label">Mã node slave</label>
                        <input type="text" class="form-control"
id="manodelslave" name="manodelslave" />
                    </div>
                    <div class="form-group">
                        <label class="control-label">Tên node slave</label>
                        <input type="text" class="form-control"
id="tennodeslave" name="tennodeslave" />
                    </div>
                </div>
                <div class="modal-footer">
                    <button type="submit" class="btn btn-primary"
id="login_btn"><i class="icon-save"></i>Lưu thông tin</button>
                    <button type="button" class="btn btn-default" data-
dismiss="modal">Đóng</button>
                </div>
            </form>
        </div><!-- /.modal-content -->
    </div><!-- /.modal-dialog -->
</div><!-- /.modal -->
<table class="table table-striped">
    <thead>
        <tr>
            <th>Mã</th>
            <th>Tên node slave</th>

```

```

        </tr>
    </thead>
    <tbody>
        <tr>
            <td>SS11</td>
            <td>Phòng khach 1</td>
            <td>
                <a href="/ThanhVien/SuaNodeSlave?ma=SS11"><i
class="icon-pencil2 btn btn-success"></i></a> |
                <a href="/ThanhVien/ChiTietNodeSlave?ma=SS11"><i
class="icon-tasks btn btn-primary"></i></a> |
                <a href="" onclick="XoaNodeSlave('SS11'); return
false"><i class="icon-trash btn btn-danger"></i></a>
            </td>
        </tr>
        <tr>
            <td>SS22</td>
            <td>Phòng Bếp</td>
            <td>
                <a href="/ThanhVien/SuaNodeSlave?ma=SS22"><i
class="icon-pencil2 btn btn-success"></i></a> |
                <a href="/ThanhVien/ChiTietNodeSlave?ma=SS22"><i
class="icon-tasks btn btn-primary"></i></a> |
                <a href="" onclick="XoaNodeSlave('SS22'); return
false"><i class="icon-trash btn btn-danger"></i></a>
            </td>
        </tr>
    </tbody>
</table>
<script>
$(document).ready(function () {
    $("#frm_taonodeslave").submit(function () {
        if ($("#manodelslave").val() == '') {
            alert("Mã node slave không được bỏ trống!");
            return false;
        }
        if ($("#tennodelslave").val() == '') {
            alert("Tên node slave không được bỏ trống!");
            return false;
        }
        var form_data = {
            tennodelslave: $("#tennodelslave").val(),
            manodelslave: $("#manodelslave").val(),
            ma_nodemaster: $("#ma_nodemaster").val(),
        };
        $.ajax({
            url: '/ThanhVien/TaoNodeSlave',
            type: 'POST',
            async: true,
            data: form_data,
            success: function (msg) {
                if (msg == "ok")
                    window.location.reload();
                else {
                    alert(msg)
                }
            }
        });
        return false;
    });
});

```

```

</script>
<script>
    $(document).ready(function () {
        $("#frm_taocontacmaster").submit(function () {
            if ($("#tencongtacmaster").val() == '') {
                alert("Tên công tác không được bỏ trống!");
                return false;
            }
            var form_data = {
                tencongtacmaster: $("#tencongtacmaster").val(),
                ma_nodemaster: $("#ma_nodemaster").val(),
            };
            $.ajax({
                url: '/ThanhVien/TaoCongTacMaster',
                type: 'POST',
                async: true,
                data: form_data,
                success: function (msg) {
                    if (msg == "ok")
                        window.location.reload();
                    else {
                        alert(msg)
                    }
                }
            });
            return false;
        });
    });
</script>
<script>
    function XoaNodeSlave(ma) {
        if (confirm("Bạn có muốn xóa không?")) {
            var form_data = {
                ma_nodeslave: ma,
            };
            $.ajax({
                url: '/ThanhVien/XoaNodeSlave',
                type: 'POST',
                async: true,
                data: form_data,
                success: function (msg) {
                    if (msg == "ok")
                        window.location.reload();
                    else {
                        alert(msg);
                    }
                }
            });
        }
    }
</script>
<script>
    function XoaCongTacMaster(ma) {
        if (confirm("Bạn có muốn xóa không?")) {
            var form_data = {
                ma_congtacmaster: ma,
            };
            $.ajax({
                url: '/ThanhVien/XoaCongTacMaster',
                type: 'POST',
                async: true,

```

```

        data: form_data,
        success: function (msg) {
            if (msg == "ok")
                window.location.reload();
            else {
                alert(msg);
            }
        }
    });
}
</script>
<script>

$(document).ready(function () {
    $(".btn_master").click(function () {
        var id = $(this).attr('rel');
        ClickCongTacMaster(id);
    });
});
///////////////////////////////
function ClickCongTacMaster(ma) {
    var form_data = {
        ma_congtacMaster: ma,
    };
    $.ajax({
        url: '/ThanhVien/ClickCongTacMaster',
        type: 'POST',
        async: true,
        data: form_data,
        success: function (msg) {
            if (msg == "ok0") {
                $("#btnmaster" + ma).removeClass('btn-danger');
                //$("#btnmaster" + ma).addClass('btn-primary');
                //window.location.reload();
            }
            else if (msg == "ok1") {
                $("#btnmaster" + ma).addClass('btn-danger');
                // $("#btnmaster" + ma).removeClass('btn-primary');
            }
            else {
                alert(msg);
            }
        }
    });
}
//function changecl()
//{
//    $("#btndlave").removeClass('btn-danger');
//}
</script>

</div>
<!-- /container -->
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>

<!-- FOOTER -->
<footer id="footer">

```

```

<div class="container">
    <div class="row">
        <div class="col-md-6">
            <!-- Footer Logo -->
            

            <!-- Small Description -->
            <p>
                Công ty TNHH Công nghệ Kiến Việt <br />
                258/7 Trần Hưng Đạo, Quận 1, <br /> TP. Hồ Chí Minh, Việt Nam
            </p>

            <!-- Contact Address -->
            <address>
                <ul class="list-unstyled">
                    <li class="footer-sprite address">
                        </li>
                    <li class="footer-sprite phone">
                        0902 80 7576<br />
                    </li>
                    <li class="footer-sprite email">
                        sale@kienviettech.vn
                        <br />
                        <br />
                        <a href="http://online.gov.vn/CustomWebsiteDisplay.aspx?DocId=23008" target="_blank">
                            
                            </a>
                            <br />
                            <br />
                            MST: 0312635982 <br />
                            Cấp lần đầu tiên vào ngày 22/01/2014
                        </li>
                    </ul>
                </address>
                <!-- /Contact Address -->
            </div>

            <div class="col-md-2">
                <!-- Links -->
                <h4 class="letter-spacing-1">SITEMAP</h4>
                <ul class="footer-links list-unstyled">
                    <li>
                        <a href="/ThanhVien/Index" target="_parent">
                            Trang Chủ
                        </a>
                    </li>
                    <li>
                        <a href="#" target="_blank">
                    </li>
                </ul>
            </div>
        </div>
    </div>

```

```

        Sản Phẩm
    </a>
</li>
<li>
    <a href="#" target="_blank">
        Giới Thiệu
    </a>
</li>
<li>
    <a href="#" target="_blank">
        Liên Hệ
    </a>
</li>
</ul>
<!-- /Links -->

</div>

<div class="col-md-4">

    <!-- Newsletter Form -->
    <h4 class="letter-spacing-1 uppercase">
        Giữ Kết Nối
    </h4>
    <p>
        Giữ kết nối với hộp thư của chúng tôi để luôn nhận
        được những thông tin mới nhất.
    </p>

    <div id="UpdatePanel3">

        <div class="input-group">
            <span class="input-group-addon"><i class="fa fa-envelope"></i></span>
            <input name="txbNhanBanTin" type="text"
                id="txbNhanBanTin" placeholder="Type your email..." class="form-control"
                required" />
            <span class="input-group-btn">
                <input type="submit" name="btnNhanBanTin"
                    value="Kết nối" id="btnNhanBanTin" class="btn btn-primary" />
            </span>
        </div>

        </div>
        <!-- /Newsletter Form -->
        <!-- Social Icons -->
        <div class="margin-top-20">
            <a
                href="https://www.facebook.com/kienviettechnology" target="_blank"
                class="social-icon social-icon-border social-facebook pull-left"
                data-toggle="tooltip" data-placement="top"
                title="Facebook">

                <i class="icon-facebook"></i>
                <i class="icon-facebook"></i>
            </a>
            <a href="https://youtu.be/6Mq7uqlLlbe"
                target="_blank">

```

```

        class="social-icon social-icon-border social-
youtube pull-left"
        title="YouTube">
            <i class="icon-youtube"></i>
            <i class="icon-youtube"></i>
        </a>

        <a href="#" class="social-icon social-icon-border
social-twitter pull-left" data-toggle="tooltip" data-placement="top"
title="Twitter">
            <i class="icon-twitter"></i>
            <i class="icon-twitter"></i>
        </a>

        <a href="https://plus.google.com/+KiếnViệtTech"
class="social-icon social-icon-border social-gplus pull-left" data-
toggle="tooltip" data-placement="top" title="Google plus">
            <i class="icon-gplus"></i>
            <i class="icon-gplus"></i>
        </a>

        <a href="#" class="social-icon social-icon-border
social-linkedin pull-left" data-toggle="tooltip" data-placement="top"
title="Linkedin">
            <i class="icon-linkedin"></i>
            <i class="icon-linkedin"></i>
        </a>

    </div>
    <!-- /Social Icons -->

</div>

</div>

</div>

<div class="copyright">
    <div class="container">
        <ul class="pull-left nomargin list-inline mobile-block">
            <li>
                &copy;
                Bản quyền thuộc Công ty TNHH Công nghệ Kiến Việt
            </li>
            <li>&bull;</li>
            <li>
                Được phát triển bởi Công ty TNHH Công nghệ Kiến
                Việt
            </li>
        </ul>
        <ul class="pull-right nomargin list-inline mobile-block">
            <li>
                Chính sách bảo mật thông tin
            </li>
        </ul>
    </div>
</div>
</footer>

```

```
<!-- /FOOTER -->
</div>
<!-- /wrapper -->
<!-- SCROLL TO TOP -->
<a href="#" id="toTop"></a>
<!-- PRELOADER -->
<div id="preloader">
    <div class="inner">
        <span class="loader"></span>
    </div>
</div>
<!-- /PRELOADER -->
<!-- JAVASCRIPT FILES -->
<script type="text/javascript">var plugin_path = '../' +
'assets1/plugins/' ;</script>
<script src="/assets1/plugins/jquery/jquery-2.2.3.min.js"></script>
<script src="/assets1/js/scripts.js"></script>
</body>
</html>
```