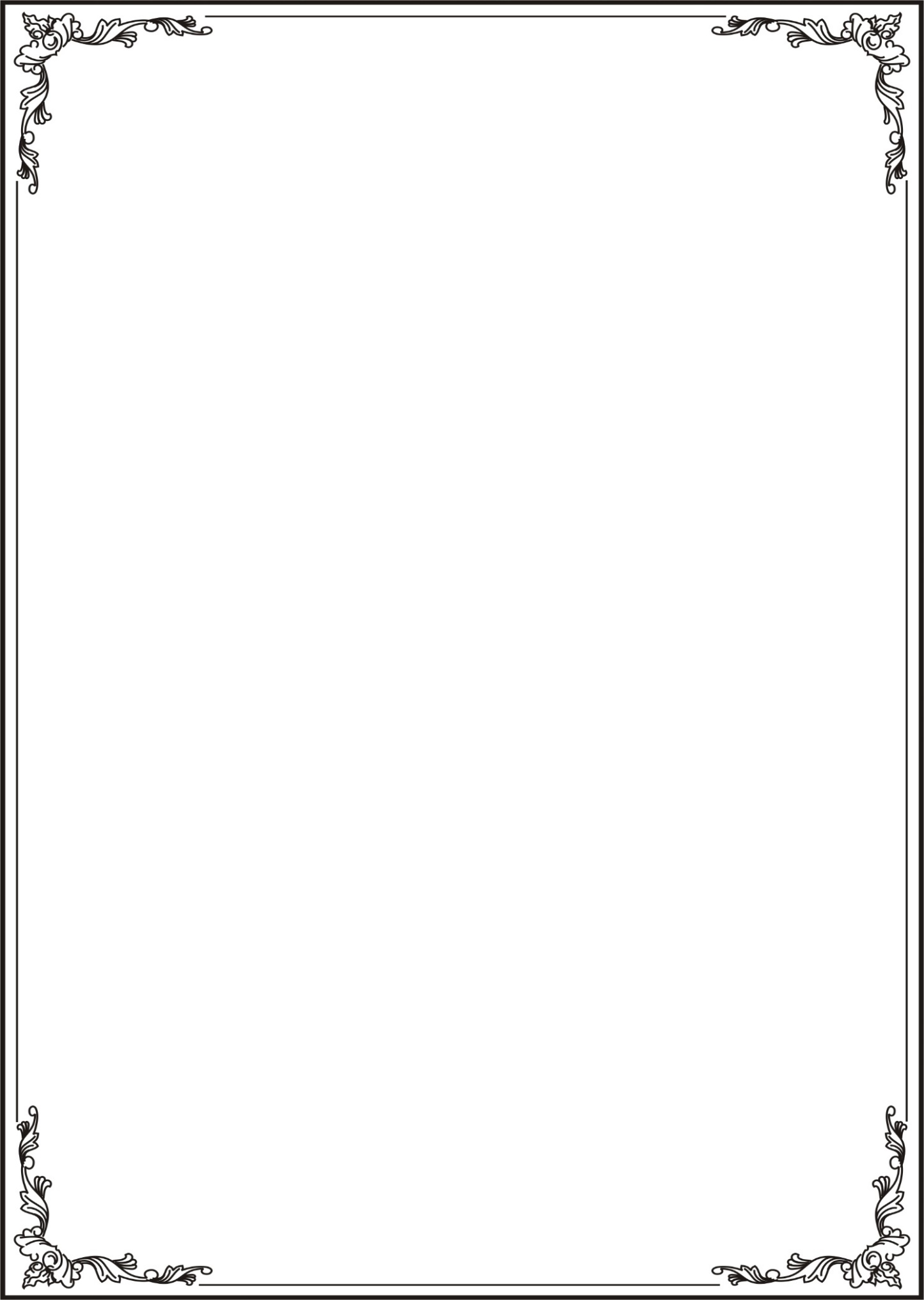
****

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG THÁI NGUYÊN**

**KHOA CÔNG NGHỆ TỰ ĐỘNG HÓA**



**THỰC TẬP TỐT NGHIỆP**

***ĐỀ TÀI:***

*XÂY DỰNG ỨNG DỤNG WINFORM ĐIỀU KHIỂN THIẾT BỊ ĐIỆN GIA DỤNG QUA GIAO THỨC MQTT*

|  |  |
| --- | --- |
| Giáo viên hướng dẫn | : Th.S Trần Tuấn Việt |
| Sinh viên thực hiện | : Phan Ngọc Phong |
| Lớp | : KTĐ ĐT K16A |
|  |  |

# **LỜI NÓI ĐẦU**

Nếu như Cách mạng công nghiệp đầu tiên sử dụng năng lượng nước và hơi nước để cơ giới hóa sản xuất. Cuộc cách mạng lần 2 diễn ra nhờ ứng dụng điện năng để sản xuất hàng loạt. Cuộc cách mạng lần 3 sử dụng điện tử và công nghệ thông tin để tự động hóa sản xuất. Bây giờ, cuộc Cách mạng Công nghiệp Thứ tư đang nảy nở từ cuộc cách mạng lần ba, nó kết hợp các công nghệ lại với nhau, làm mờ ranh giới giữa vật lý, kỹ thuật số và sinh học. Cách mạng***công nghệ 4.0*** tạo điều kiện thuận lợi cho việc tạo ra các nhà máy thông minh, sự phát triển của internet vạn vật giúp tạo ra bản sao ảo của thế giới vật lý, cho phép mọi người ở khắp nơi trên thế giới kết nối với nhau thông qua mạng internet dịch vụ qua các thiết bị di động ở mọi lúc, mọi nơi.

***Công nghiệp 4.0*** thúc đẩy chuyển đổi kỹ thuật số của sản xuất thông qua việc tích hợp các hệ thống và quy trình khác nhau trước đây thông qua các hệ thống máy tính được kết nối với nhau qua chuỗi cung ứng và giá trị. Cách mạng Công nghiệp 4.0 đang báo hiệu một sự thay đổi trong bối cảnh sản xuất truyền thống bao gồm ba xu hướng công nghệ thúc đẩy sự chuyển đổi này: kết nối, thông minh và tự động hóa linh hoạt .

 Trong quá trình học tập em đã nhận được đề tài: *Xây dựng ứng dụng Winform điều khiển thiết bị điện gia dụng qua giao thức MQTT.* Đề tài này đã giúp em hiểu biết hơn về ứng dụng của việc điều khiển từ xa giúp giảm sức lao động

Để hoàn thành đề tài này, ngoài sự nỗ lực của bản thân em xin chân thành gửi lời cảm ơn sâu sắc tới các thầy cô Khoa Công Nghệ Tự Động Hóa trường Đại học Công nghệ thông tin và Truyền thông Thái Nguyên, đặc biệt là thầy **Th.S Trần Tuấn Việt** đã tận tình hướng dẫn, giúp đỡ và tạo điều kiện tốt nhất cho em kể từ khi nhận đề tài tới khi hoàn thành đề tài này.

Em xin chân thành cảm ơn!

# **MỤC LỤC**

[**LỜI NÓI ĐẦU** i](#_Toc73975428)

[**MỤC LỤC** ii](#_Toc73975429)

[**DANH MỤC HÌNH ẢNH** iii](#_Toc73975430)

[**CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI** 1](#_Toc73975431)

[**1.1** **Giới thiệu chung** 1](#_Toc73975432)

[**1.2** **Tổng quan các công nghệ sử dụng** 1](#_Toc73975435)

[*1.2.1* *Công nghệ Bluetooth* 1](#_Toc73975436)

[*1.2.2* *Công nghệ Winform Application* 3](#_Toc73975437)

[*1.2.3* *Công nghệ khí nén* 5](#_Toc73975438)

[*1.2.4* *JavaScript Object Notation* 6](#_Toc73975439)

[*1.2.5* *Hệ quản trị cơ sở dữ liệu SQL Server* 8](#_Toc73975440)

[**CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG** 10](#_Toc73975441)

[**2.1** **Đặc tả hệ thống** 10](#_Toc73975444)

[**2.2** **Phân tích và thiết kế hệ thống** 10](#_Toc73975445)

[*2.2.1* *Sơ đồ khối của hệ thống và chức năng từng khối* 10](#_Toc73975446)

[*2.2.2* *Sơ đồ nguyên lý* 11](#_Toc73975447)

[*2.2.3* *Sơ đồ mạch in PCB* 12](#_Toc73975448)

[*2.2.4* *Lưu đồ thuật toán* 12](#_Toc73975449)

[*2.2.5* *Xây dựng cơ sở dữ liệu* 13](#_Toc73975450)

[*2.2.6* *Xây dựng phần mểm* 15](#_Toc73975451)

[*2.2.7* *Mô phỏng cơ cấu gắp bằng SolidWork* 15](#_Toc73975452)

[**2.3** **Giới thiệu một số công cụ, thiết bị sử dụng** 16](#_Toc73975453)

[*2.3.1* *Esp32* 16](#_Toc73975454)

[*2.3.2* *Visual Studio Code và Platform IO* 20](#_Toc73975455)

[*2.3.3* *Altium Design* 22](#_Toc73975456)

[**CHƯƠNG 3: KẾT QUẢ ĐÁNH GIÁ** 25](#_Toc73975457)

[**3.1** **Các bước sử dụng hệ thống** 25](#_Toc73975459)

[**3.2** **Đánh giá hệ thống** 27](#_Toc73975460)

[*3.2.1* *Ưu điểm* 27](#_Toc73975461)

[*3.2.2* *Nhược điểm* 27](#_Toc73975462)

[**3.3** **Kết luận** 27](#_Toc73975463)

[**Tài liệu tham khảo** 28](#_Toc73975464)

[**Phụ lục** 29](#_Toc73975465)

# **DANH MỤC HÌNH ẢNH**

[*Hình 1. 1 Bluetooth kết nối* 2](#_Toc73975652)

[*Hình 1. 2 Giao diện winform* 4](#_Toc73975653)

[*Hình 1. 3 Dây chuyến sử dụng khí nén* 6](#_Toc73975654)

[*Hình 1. 4 Cấu trúc Json* 7](#_Toc73975655)

[*Hình 1. 5 SSMS Studio* 9](#_Toc73975656)

[Hình 2. 1 Sơ đồ khối hệ thống 10](#_Toc73975660)

[Hình 2. 2 Sơ đồ nguyên lý hệ thống 11](#_Toc73975661)

[Hình 2. 3 Mạch in PCB 12](#_Toc73975662)

[Hình 2. 4 Sơ đồ thuật toán hệ thống 13](#_Toc73975663)

[Hình 2. 5 Module ESP32 17](#_Toc73975664)

[Hình 2. 6 Sơ đồ chân Esp32 19](#_Toc73975665)

[Hình 2. 7 Visual Studio Code 20](#_Toc73975666)

[Hình 2. 8 Platform IO 21](#_Toc73975667)

[Hình 2. 9 Tạo Project 21](#_Toc73975668)

[Hình 2. 10 Điền thông tin project 22](#_Toc73975669)

[Hình 2. 11 Cấu trúc project sau khi tạo xong 22](#_Toc73975670)

[Hình 2. 12 Altium Design 23](#_Toc73975671)

[*Hình 3. 1 Khi gắp vật và kẹp lại* 15](#_Toc73975672)

[*Hình 3. 2 Khi cơ cấu mở ra* 16](#_Toc73975673)

[Hình 3. 3 Tìm cổng COM kết nối BLE 25](#_Toc73975674)

[Hình 3. 4 Mở app điều khiển 26](#_Toc73975675)

[Hình 3. 5 Giao diện điều khiển của app 26](#_Toc73975676)

[*Hình 3. 6 Kết nối mạch điều khiển và mạch relay* 26](#_Toc73975677)

# **CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI**

## **Giới thiệu chung**

Như chúng ta đã biết, hiện nay đại đa số các loại công tắc điện ở nhà. Đều đang dùng loại công tắc cơ đời cũ của các hãng như pana, sino, vanlock,..



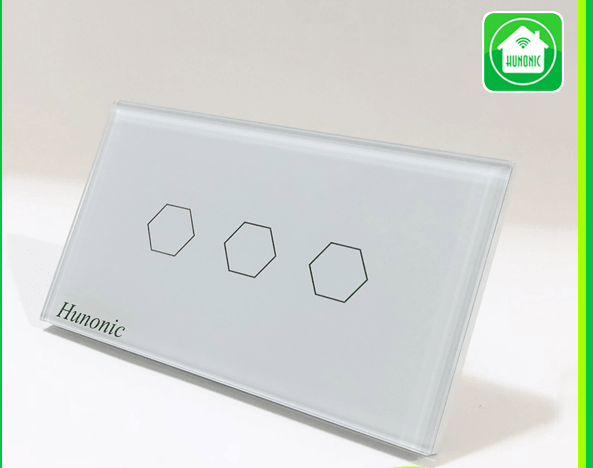
*Hình 1. 1 Công tắc cơ truyền thống*

Đặc điểm loại công tắc này là chúng ta chỉ có thể bật tắt bằng tay và muốn thao tác thì chỉ có cách là lại gần công tắc và bật.

Ngày này, Dù có ở khoảng cách xa nửa vóng trái đất đi chăng nữa. Chúng ta đã có thể nói chuyện và nhìn thấy nhau. Thông qua các thiết bị thông minh như máy tính,…và đặc biệt là điện thoại thông minh.

Vậy với chiếc điện thoại thông minh trên tay, chúng ta có thể bật tắt đèn từ xa khi đã ra khỏi nhà được không?

Chắc chắn là có rồi. Chúng ta sẽ phải sử dụng công tắc thông minh thế hệ mới để thay thế các loại công tắc đời cũ.



*Hình 1. 2 Công tắc thông minh trên thị trường*

Vì vậy em đã chọn đề tài này để tạo ra một thiết bị giúp người dùng có thể điều khiển bật, tắt tại chỗ hay từ xa thông qua internet giúp tăng độ tiện nghi khi sử dụng các thiết bị điện



## **Tổng quan các công nghệ sử dụng**

### *Công nghệ Wifi*

*a, Khái niệm*

Wifi là viết tắtcủa Wireless Fidelity là hệ thống truy cập internet không dây. Loại sóng vô tuyến này tương tự như sóng điện thoại, truyền hình và radio.

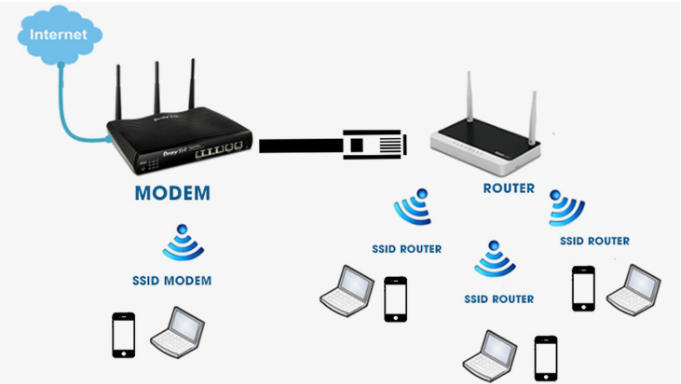


*Hình 1. 3 Công nghệ Wifi*

*b, Nguyên tắc hoạt động của Wifi*

Để tạo được kết nối Wifi nhất thiết phải có [Router](https://www.dienmayxanh.com/thiet-bi-mang-router)(bộ thu phát), Router này lấy thông tin từ mạng Internet qua kết nối hữu tuyến rồi chuyển nó sang tín hiệu vô tuyến và gửi đi, bộ chuyển tín hiệu không dây (adapter) trên các thiết bị di động thu nhận tín hiệu này rồi giải mã nó sang những dữ liệu cần thiết.

Quá trình này có thể thực hiện ngược lại, Router nhận tín hiệu vô tuyến từ Adapter và giải mã chúng rồi gởi qua Internet.



*Hình 1. 4 Hoạt động của Wifi*

*c, Wifi trong IOT*

Xu hướng hỗ trợ đa dạng kết nối dưới một nền tảng chung không giới hạn ở băng thông rộng không dây. Khi Internet of Things (IoT) bắt nhịp, sẽ có nhu cầu lớn đối với các kết nối giữa máy với máy (M2M), đa phần là không dây. Chúng sẽ có nhiều yêu cầu về hiệu năng hơn, sử dụng nhiều trong các trường hợp khác nhau dưới sự bảo trợ của IoT.

Không có công nghệ đơn lẻ nào giải quyết tất cả các yêu cầu thế nên sẽ có một danh sách dài các giao thức IoT không dây. Theo thời gian thì có thể củng cố hơn nữa, nhưng chắc chắn sẽ cần ít nhất một công nghệ mở, được tiêu chuẩn hóa cho một số cấu hình IoT chính. Những cấu hình này thay đổi theo mức độ mà chúng hỗ trợ:

* Công suất cực thấp với công suất vừa phải
* Tầm xa với phạm vi địa phương với phạm vi rất ngắn
* Tốc độ dữ liệu thấp với tốc độ dữ liệu vừa phải
* Độ trễ cực thấp với độ trễ thấp
* Phổ không có giấy phép với phổ được cấp phép

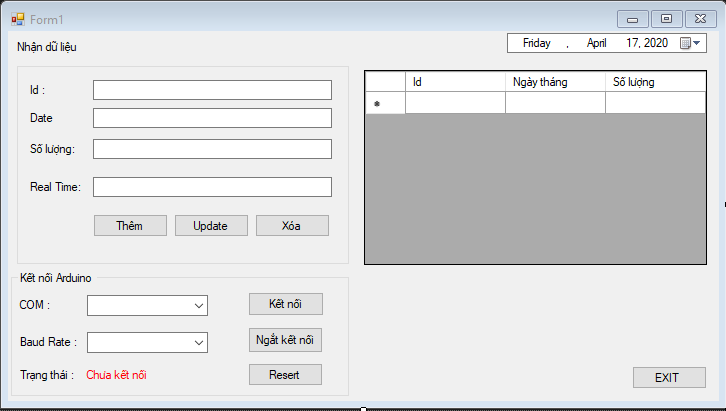
*d, Hạn chế Wifi*

Bên cạnh những ưu điểm, mạng Wifi cũng tồn tại nhiều nhược điểm chưa thể khắc phục như: phạm vi kết nối của mạng Wifi tới thiết bị có giới hạn, đi càng xa router kết nối càng yếu dần đi. Giải pháp cho vấn đề này là trang bị thêm các Repeater hoặc Access point. Tuy nhiên, gặp nhiều khó khăn do giá thành cao. Nhược điểm tiếp theo của mạng Wifi là về vấn đề băng thông, càng nhiều người kết nối vào mạng thì tốc độ truy cập giảm rõ rệt.

### *Công nghệ Winform Application*

*a, Khái niệm*

Windows Forms là một thư viện lớp đồ họa mã nguồn mở và miễn phí được bao gồm như một phần của Microsoft .NET Framework hoặc Mono Framework, cung cấp một nền tảng để viết các ứng dụng khách phong phú cho máy tính để bàn, máy tính xách tay và máy tính bảng.



*Hình 1. 5 Giao diện winform*

*b, Thành phần cấu thành Winform*

***a, Label***

*Thuộc tính:*

* Name: tên của nhãn(ảnh hưởng trực tiếp tới lập trình)
* Text : Ký tự hiển thị cho người sử dụng(Thường đặt giống với chức năng)
* Backcolor: Màu nền của nhãn

*Event:*

* Click: Ấn chuột vào đối tượng
* Mouse move: Di chuột vào đối tượng

***b, Text Box***

*Thuộc tính:*

* Name: tên của textbox
* Text: ký tự nằm trong textbox
* Multiline: cho phép nhập nhiều dòng
* Enable: cho phép/không cho phép nhập ký tự, tương ứng với 2 lựa chọn true/false
* Backcolor: màu nền của textbox

*Event:*

* Mouse: khi người dùng nhấn chuột vào textbox
* Click: khi người dùng ấn chuột vào textbox
* Mouse move: khi người dùng di chuột vào textbox

***c, Button***

*Thuộc tính:*

* Name: tên của nút bấm
* Backcolor: màu nền
* Background image: ảnh nền của button
* Font: kiểu chữ
* Size: kích cỡ
* Text: kiểu chữ trình bày trên button
* Visible: Ẩn/ hiện chữ trên button
* FlatStyle: Kiểu đường viền của điều khiển Button.

*Event:*

* Click: khi người dùng ấn chuột vào đối tượng

***d, Form***

*Thuộc tính:*

* Name: tên của form
* Backcolor: màu nền của form
* Background: ảnh nền của form
* Language: ngôn ngữ trên form
* Icon: biểu tượng
* Showicon: cho phép hiển thị biểu tượng trên form
* Text: tên hiển thị

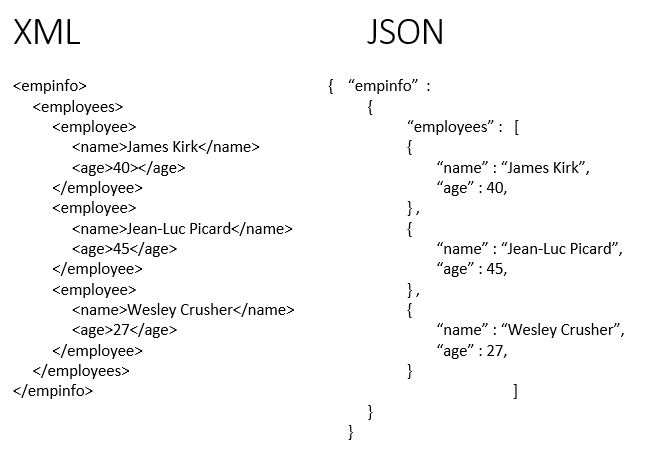
*Event:*

* Load: Nạp hay thực thi khi chạy form

### *JavaScript Object Notation*

*a, Json là gì?*

**JSON** là viết tắt của **J**ava**S**cript **O**bject **N**otation, là một kiểu định dạng dữ liệu tuân theo một quy luật nhất định mà hầu hết các ngôn ngữ lập trình hiện nay đều có thể đọc được. **JSON**là một tiêu chuẩn mở để trao đổi dữ liệu trên web.



*Hình 1. 6 Cấu trúc Json*

*b, Định nghĩa*

Định dạng JSON sử dụng các cặp key – *value* để dữ liệu sử dụng. Nó hỗ trợ các cấu trúc dữ liệu như đối tượng và mảng.

Ta có thể thấy cú pháp của JSON có 2 phần đó là key và value

* Chuỗi JSON được bao lại bởi dấu ngoặc nhọn {}
* Các key, valuecủa JSON bắt buộc phải đặt trong dấu nháy kép {“}, nếu bạn đặt nó trong dấu nháy đơn thì đây không phải là một chuỗi JSON đúng chuẩn. Nếu trường hợp trong value của bạn có chứa dấu nháy kép " thì hãy dùng dấu () để đặt trước nó, ví dụ  "json là gì".
* Nếu có nhiều dữ liệu thì dùng dấu phẩy , để ngăn cách.
* Các key của JSON bạn nên đặt chữ cái không dấu hoặc số, dấu \_ và không có khoảng trắng., ký tự đầu tiên không nên đặt là số.

*c, Cấu trúc chuỗi Json*

Object trong Json được thể hiện bằng dấu ngoặc nhọn {}. Khái niệm Object trong Json cũng khá tương đồng với Object trong Javascript. Tuy nhiên, Object trong Json vẫn có những giới hạn như:

Key: phải luôn nằm trong dấu ngoặc kép, không được phép là biến số.

Value: Chỉ cho phép các kiểu dữ liệu cơ bản: numbers, String, Booleans, arrays, objects, null. Không cho phép function, date, undefined.

Không cho phép dấy phẩy cuối cùng như Object trong Javascript.

## **Giao thức MQTT**

### *Các khái niệm cơ bản*

*a, Khái niệm*

MQTT (Message Queuing Telemetry Transport) là dựa trên giao thức tin nhắn Publish-Subscribe. Sử dụng TCP/IP để truyền dữ liệu. Giao thức được sử dụng phổ biến trong mạng lưới IOT với ưu điểm code đơn giản, gói tin nhẹ sử dụng băng thông nhỏ để truyền dữ liệu thích hợp trong mạng truyền bị giới hạn.

*b, Broker*

Là server cài MQTT server thu thập dữ liệu và giao tiếp các client.

*c, Topic*

Là chủ đề mà Broker tạo ra để Device gửi vào. Nó như là folder chứa dữ liệu vậy nên có thể ở dạng đường dẫn abc/topic1.

*d, Publish*

Là gói tin Device gửi lên Broker. Khi Device Publish dữ liệu vào topic, các Client sẽ nhận được dữ liệu khi đăng kí topic. Gói Publish có thể là gói gửi dữ liệu gửi vào topic hoặc gói cài đặt Retain Message hay LWT Message do cờ Retain, LWT quy định.

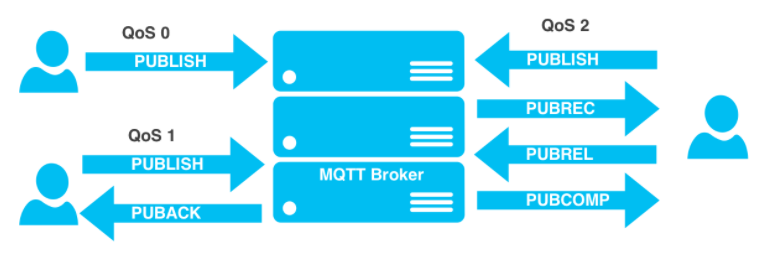
*e, Subcribe*

Là gói đăng kí nhận data hay thông tin từ topic đăng kí(Broker sẽ tạo một topic mới nếu không tìm được topic đăng kí). Khi Client gửi gói tin Subcribe tới Broker để đăng kí một topic, ví dụ topic “T2Tdemo”. Bất cứ khi nào Device gửi gói tin Publish vào topic “T2Tdemo”. Dữ liệu trong gói Publish chuyển về Client. Giống như bạn Subscribe youtube.

*f, QOS*

Chất lượng đường truyền

* Qos 0: Device gửi Publish tới Server không cần quan tâm đến gói tin gửi.
* Qos 1: Device gửi Publish tới Server ,Sau đó nếu Server nhận được gói tin và gửi lại Device gói PUBACK để xác nhận đã nhận được gói Publish từ Device
* Qos 2: Device gửi Publish tới Server , Server nhận gói Publish gửi PUBREC lại Device kèm theo ID đã nhận. Device nhận PUBREC gửi PUBREL kèm ID đó lại Server. Server gửi lại PUBCOMP lại Device.



*Hình 1. 7 Chất lượng đường truyền*

*g, Retain*

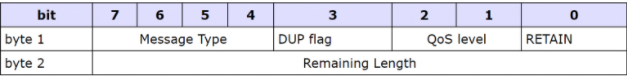
Là cờ báo gói Publish, Device gửi tới Broker cài đặt Retain Message cho topic. Khi Client subcribe vào topic đã cài đặt Retain Message sẽ nhận được ngay Retain Message. Ví dụ “Device online”.

*h, LWT(Last Will and Testament)*

Là cờ báo gói Publish, Device gửi tới Broker topic cài đặt LWT Message. Khi Client Subcribe vào topic đã cài đặt LWT Message sẽ nhận được tin nhắn LWT Message khi Device offline

### *Packet*

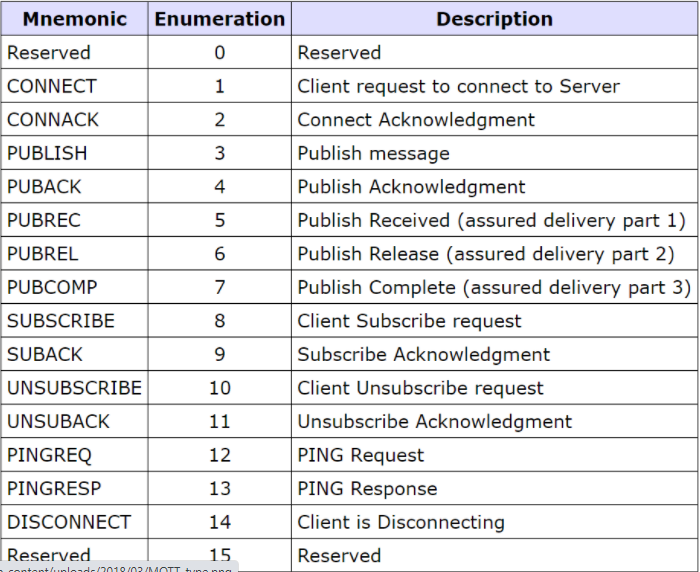
*a, Cấu trúc gói tin giao tiếp*



*Hình 1. 8 Cấu trúc gói tin giao tiếp*

Có thể có nhiều byte được sử dụng sau Remaining Length

Byte 1: Bits 7-4 Là byte nhận biết đây là packet tên là gì. Dựa vào bảng dưới ta có thể thấy rằng có tất cả 14 loại packet khác nhau.



*Hình 1. 9 Bảng tra tên packet*

Bits 0-3 là các flag Qos,  Retain mình đã giới thiệu ở topic [giới thiệu chung](https://vidieukhien.xyz/vi/2018/03/14/mqtt-tutorial-co-ban-den-nang-cao-khai-niem/). Dup flag bật khi Client hoặc Server gửi lại một trong các message sau

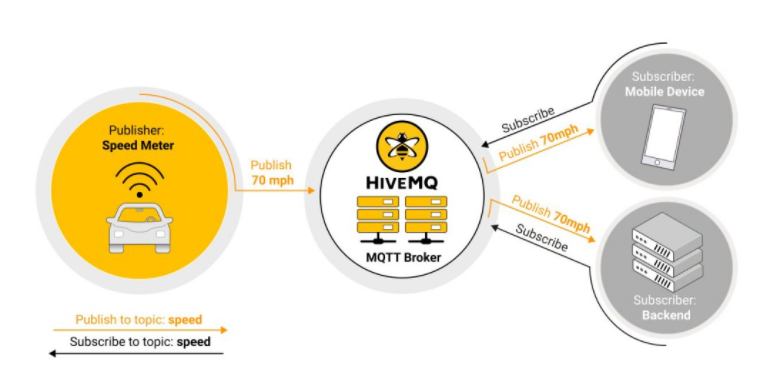
Byte 2: Remain Length: Số byte trong packet. Độ dài này cần được encode chuẩn UTF-8

Encode UTF-8: Trong 1byte: 7 bit đầu dùng để tính kích thước, bit thứ 8 sẽ báo byte có được sử dụng hay không.

### *MQTT Publish & Subscribe*

*a, Mô hình kiến trúc publish / subscribe*

Mẫu publish/subscribe (còn được gọi là pub/sub) cung cấp một thay thế cho kiến ​​trúc máy khách-máy chủ truyền thống. Trong mô hình máy khách-máy khách, máy khách liên lạc trực tiếp với điểm cuối. Mô hình pub / sub tách riêng máy khách gửi tin nhắn (nhà xuất bản-publish) từ máy khách hoặc máy khách nhận tin nhắn (người đăng ký-subscribe ) . Các nhà xuất bản và đăng ký không bao giờ liên lạc trực tiếp với nhau. Trong thực tế, họ thậm chí không nhận thức được rằng cái kia tồn tại. Kết nối giữa chúng được xử lý bởi một thành phần thứ ba (người môi giới-broker) . Công việc của nhà môi giới là lọc tất cả các tin nhắn đến và phân phối chúng một cách chính xác cho người đăng ký. Vì vậy, hãy đi sâu hơn một chút vào một số khía cạnh chung của pub/sub



*Hình 1. 10 Mô hình kiến trúc publish / subscribe*

Khía cạnh quan trọng nhất của pub / sub là sự tách rời của nhà xuất bản tin nhắn từ người nhận (người đăng ký). Sự tách rời này có một số chiều:

* *Phân tách không gian:* Nhà xuất bản và người đăng ký không cần biết nhau (ví dụ: không trao đổi địa chỉ IP và cổng).
* *Thời gian tách rời:* Nhà xuất bản và người đăng ký không cần phải chạy cùng một lúc.
* *Phân tách đồng bộ hóa:* Hoạt động trên cả hai thành phần không cần phải bị gián đoạn trong quá trình xuất bản hoặc nhận.

*b, Khả năng mở rộng*

Pub / Sub quy mô tốt hơn so với cách tiếp cận máy khách-máy chủ truyền thống. Điều này là do các hoạt động trên môi giới có thể được song song hóa cao và các thông điệp có thể được xử lý theo cách hướng sự kiện. Bộ nhớ đệm tin nhắn và định tuyến thông minh của tin nhắn thường là một yếu tố quyết định để cải thiện khả năng mở rộng. Tuy nhiên, mở rộng lên đến hàng triệu kết nối là một thách thức. Mức độ kết nối cao như vậy có thể đạt được với các nút môi giới được phân cụm để phân phối tải trên nhiều máy chủ riêng lẻ hơn bằng cách sử dụng bộ cân bằng tải.

### *Ưu điểm, nhược điểm của MQTT và một số ứng dụng triển khai*

*a, Ưu điểm*

MQTT cho phép hệ thống [SCADA](https://mesidas.com/scada/) của bạn truy cập dữ liệu IoT. MQTT mang lại nhiều ưu điểm vượt trội cho hệ thống của bạn:

* Phân phối thông tin hiệu quả hơn
* Tăng khả năng mở rộng
* Giảm đáng kể tiêu thụ băng thông mạng
* Giảm tốc độ cập nhật xuống giây
* Rất phù hợp cho điều khiển và viễn thám
* Tối đa hóa băng thông đang sử dụng
* Chi phí đầu tư cực kỳ thấp
* Rất an toàn (bảo mật dựa trên sự cấp phép)
* Đã được sử dụng bởi các ngành công nghiệp dầu khí, Amazon, Facebook và các doanh nghiệp lớn khác
* Giảm thiểu thời gian phát triển
* Giao thức publish/subscribe thu thập nhiều dữ liệu hơn với ít băng thông hơn so với giao thức cũ.

*b, Nhược điểm*

* MQTT có chu kỳ truyền chậm hơn so với CoAP.
* Tài nguyên của MQTT hoạt động dựa trên subscribe động, trong khi CoAP sử dụng hệ thống tài nguyên tĩnh – ổn định.
* MQTT không được mã hóa. Thay vào đó, nó sử dụng TLS / SSL để mã hóa bảo mật.
* Rất khó để tạo ra một mạng MQTT có thể mở rộng toàn cầu.

*c, Ứng dụng của MQTT trong các ứng dụng lớn*

* Facebook Messenger: Facebook đã sử dụng các khía cạnh của MQTT trong Facebook Messenger để trò chuyện trực tuyến . Tuy nhiên, không rõ MQTT được sử dụng bao nhiêu hoặc để làm gì.
* IECC Scalable: DeltaRail phiên bản mới nhất của hệ thống kiểm soát hiệu IECC của họ đã sử dụng MQTT cho thông tin liên lạc trong các phần khác nhau của hệ thống và các thành phần khác của hệ thống báo hiệu. Nó cung cấp khung truyền thông cơ bản cho một hệ thống tuân thủ các tiêu chuẩn CENELEC cho các thông tin liên lạc quan trọng về an toàn.
* Amazon Web Services đã công bố Amazon IoT dựa trên MQTT vào năm 2015.
* Các tổ chức không gian địa lý SensorThings API đặc điểm kỹ thuật tiêu chuẩn có một phần mở rộng MQTT trong tiêu chuẩn như một giao thức thông báo bổ sung ràng buộc. Nó đã được chứng minh trong một thí điểm IoT của Bộ An ninh Nội địa Hoa Kỳ.
* Các dịch vụ của Cơ sở hạ tầng thượng nguồn OpenStack được kết nối bằng một bus tin nhắn hợp nhất MQTT với Mosquitto là broker MQTT.
* Adafruit đưa ra một MQTT miễn phí Cloud Service cho thí nghiệm IOT và người học gọi Adafruit IO trong năm 2015.
* Microsoft Azure IoT Hub sử dụng MQTT làm giao thức chính cho các tin nhắn từ xa .
* XIM, Inc. đã ra mắt ứng dụng khách MQTT có tên MQTT Buddy vào năm 2017. Đây là ứng dụng MQTT dành cho Android và iOS , nhưng không phải là F-Droid , người dùng có sẵn bằng tiếng Anh, tiếng Nga và tiếng Trung Quốc.
* Node-RED hỗ trợ các nút MQTT kể từ phiên bản 0.14, để định cấu hình đúng các kết nối TLS.
* Nền tảng tự động hóa phần mềm nguồn mở Home Assistant được bật MQTT và cung cấp bốn tùy chọn cho các broker MQTT.

## *Bunifu framework*

*a, Giới thiệu*

Bunifu frameworklà một thư viện hỗ trợ việc thiết kế cho windows form đơn giản hơn. Với bunifu framework .NET, thiết kế winform chưa bao giờ dễ như thế.



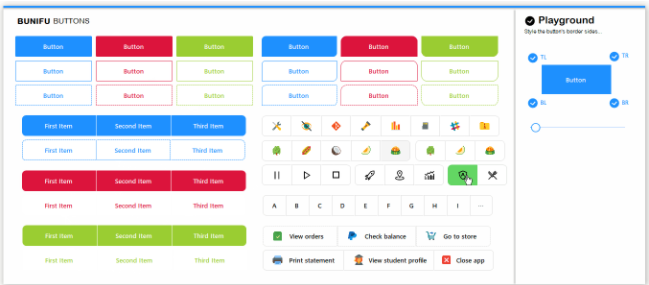
*Hình 1. 11 Bunifi UI*

Với bunifu  bạn có thể làm nhiều thứ dễ dàng hơn như :

* Slide menu hoạt hình (đóng mở)
* Nền winform đẹp hơn với Gradient background - Tô màu cầu vồng cho form (trộn 3 màu)
* Hỗ trợ Flat UI - Thiết kế giao diện phẳng như Metro Design
* Nhiều chức năng khác như progressbar quay tròn,biểu đồ ,rất tiện lợi.

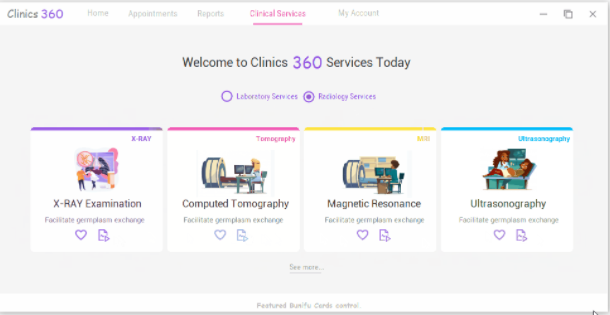
*b, Một số thành phần cơ bản*

***Button Bunifu*** thực sự là một trong những loại nút. Có rất nhiều thứ bạn có thể đạt được với một điều khiển này và dễ dàng tùy chỉnh đến mức bạn sẽ tự hỏi mình đã sống như thế nào nếu không có nó ... Với hơn 12 phong cách thiết kế được tích hợp sẵn, tất cả những gì bạn cần là một chút sáng tạo và bạn đã sẵn sàng để tạo ra một cái gì đó đẹp!



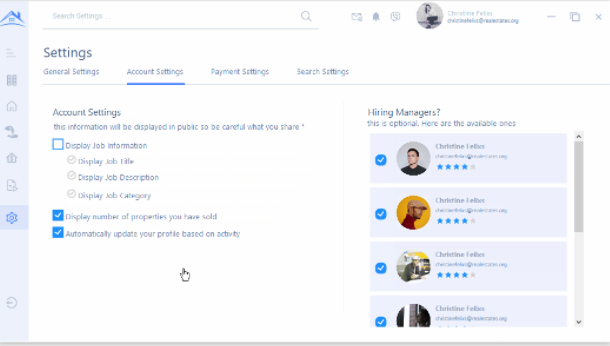
*Hình 1. 12 Bunifi Button*

***Bunifu Cards*** là một điều khiển được thiết kế để bắt chước một thẻ cơ bản và hữu ích khi sắp xếp thông tin trên trang tổng quan của bạn. Bảng điều khiển sau đã sử dụng thẻ Bunifu để phát triển một bố cục thể hiện sự sắp xếp trang nhã của danh sách các dịch vụ được cung cấp.



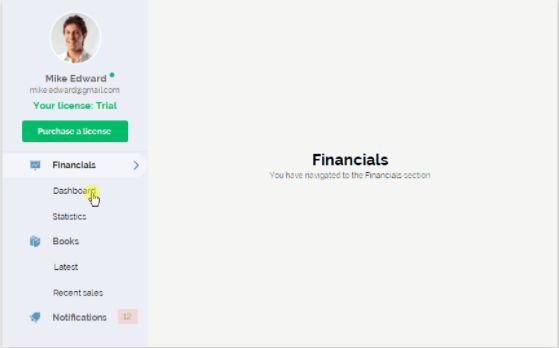
*Hình 1. 13 Bunifu Card*

***Bunifu CheckBox*** hỗ trợ Tristate: Các trạng thái được kiểm tra, không được kiểm tra và không xác định; Tùy chỉnh giao diện người dùng linh hoạt cho các trạng thái đã chọn và không được chọn.



*Hình 1. 14 Bunifu Checkbox*

***Bunifu Pages*** là một điều khiển đa vùng chứa trong Windows Forms độc đáo cho phép dễ dàng phát triển giao diện người dùng với nhiều chế độ xem trong cùng một Biểu mẫu, vùng chứa hoặc điều khiển người dùng tùy chỉnh.



*Hình 1. 15 Bunifu Page*

# **CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG**



## **Đặc tả hệ thống**

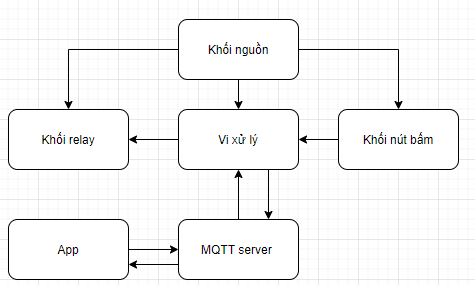
Hệ thống sau khi xây dựng xong phải đảm bảo các yêu cầu thiết kế sau:

* Hệ thống có khả năng điều khiển thông qua môt trường internet
* Kết hợp được kiểu điều khiển truyền thống và điều khiển từ xa qua internet
* Ứng dụng điều khiển trực quan dễ thao tác cho người sử dụng
* Đồng bộ hóa dữ liệu (trạng thái đầu ra) giữa phần mềm và phần cứng
* Tiết kiệm năng lượng cho người dùng
* Tiết kiệm chi phí hoạt động

## **Phân tích và thiết kế hệ thống**

### *Sơ đồ khối tổng quan của hệ thống và chức năng từng khối*

*a, Sơ đồ khối tổng quan của hệ thống*



Hình 2. 1 Sơ đồ khối hệ thống

*b, Chức năng chi tiết từng khối*

*Khối nguồn:* Cung cấp điện áp ổn định tới các khối trong hệ thống, khối nguồn được sử dụng trong hệ thống có điện áp 5 VDC và 3.3 VDC.

*Khối vi điều khiển:* Có chức năng kết nối Wifi nhận dữ liệu từ khối điều khiển Application, tính toán dữ liệu và gửi về khối Relay .Vi điều khiển được sử dụng trong hệ thống là ESP8266

*Khối điều khiển Application:* Là một ứng dụng trên máy tínhcó kết nối internet và MQTT client giúp gửi những thiết lập cần thiết để điều khiển

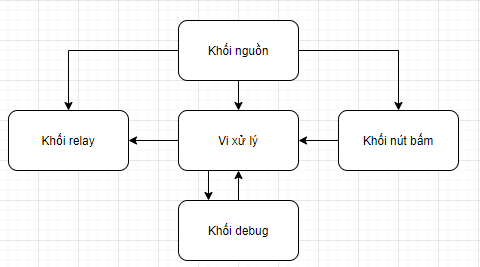
*Khối Relay:* Khi nhận được tín hiệu từ vi điều khiển nó thực hiện đóng / mở theo tín hiệu nhận được.

*Khối nút bấm :* Thực hiện thao tác của người dùng tùy theo mục đích sử dụng như đóng hay mỏ relay thông qua vi điều khiển.

*MQTT server*: MQTT (Message Queuing Telemetry Transport) là một giao thức gởi dạng publish/subscribe sử dụng cho các thiết bị [Internet of Things](/tags/IoT) với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định. Ở đây MQTT làm nơi trung chuyển dữ liệu giữa app và vi điều khiển

### *Thiết kế phần cứng*

*a, Sơ đồ các khối cần thiết trong mạch*



*Khối nguồn:* Cung cấp điện áp ổn định tới các khối trong hệ thống, khối nguồn được sử dụng trong hệ thống có điện áp 5 VDC và 3.3 VDC.

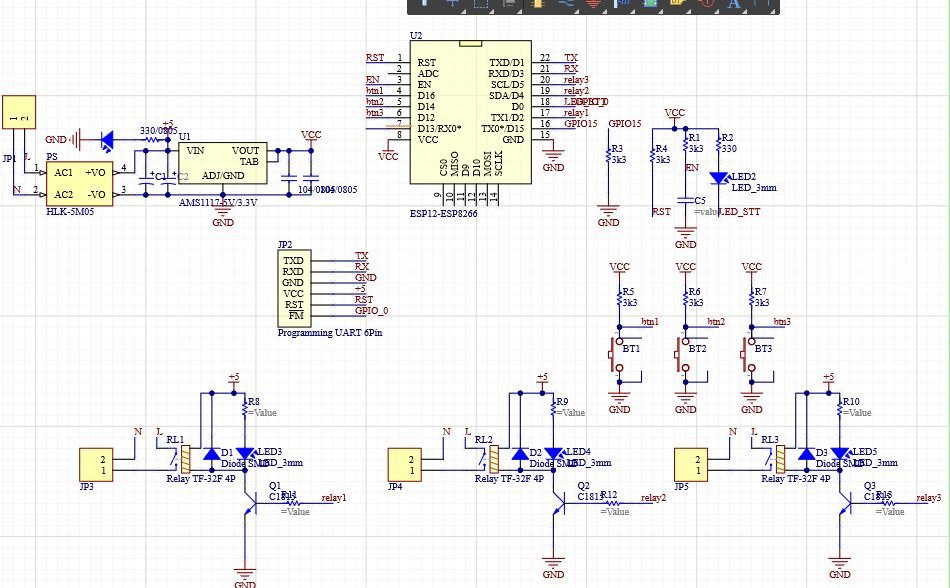
*Khối vi điều khiển:* Có chức năng kết nối Wifi nhận dữ liệu từ khối điều khiển Application, tính toán dữ liệu và gửi về khối Relay .Vi điều khiển được sử dụng trong hệ thống là ESP8266

*Khối điềudebug:* Gồm 6 jum để kết nối với máy tính qua giao thức UART để nạp chương trình và hiển thị lỗi trong quá trình code

*Khối Relay:* Khi nhận được tín hiệu từ vi điều khiển nó thực hiện đóng / mở theo tín hiệu nhận được.

*Khối nút bấm :* Thực hiện thao tác của người dùng tùy theo mục đích sử dụng như đóng hay mỏ relay thông qua vi điều khiển.

*b, Sơ đồ nguyên lý*

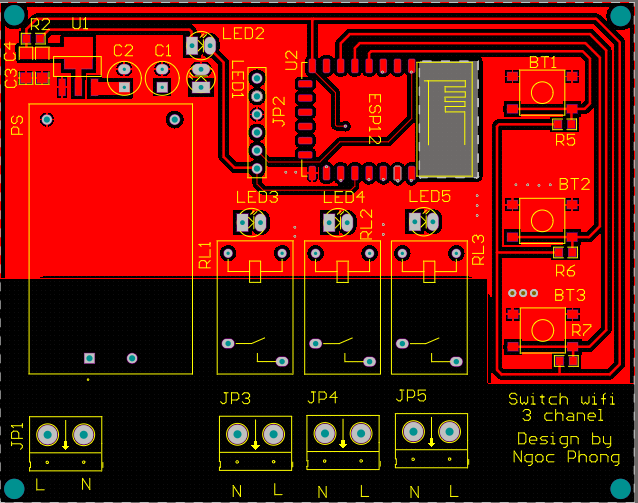


*Hình 2. 2 Sơ đồ nguyên lý*

*Nguyên lý hoạt động tổng quát:* Khi người dùng tác động vào nút bấm thì sẽ cho kết quả ở đầu ra là bật hay tắt các relay (nếu trạng thái bạn đầu là tắt thì đèn báo trên relay sẽ tắt và ngược lại)

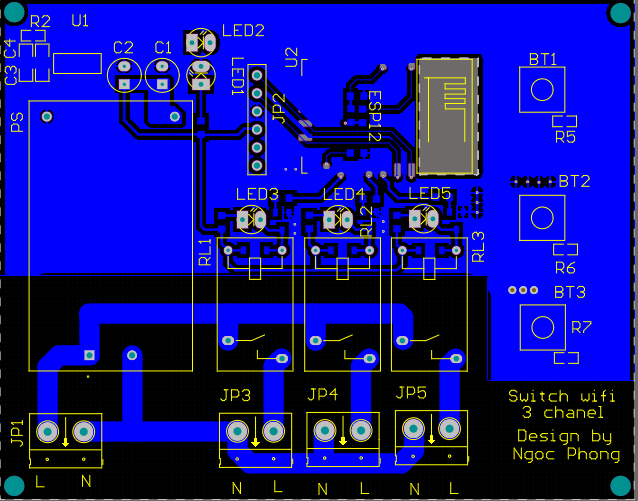
*b, Layout mạch PCB – 3D*

*Top layer:* Gồm các dây điện đi ở mặt trước của mạch

**

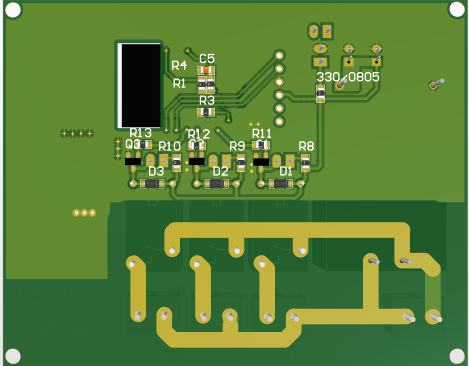
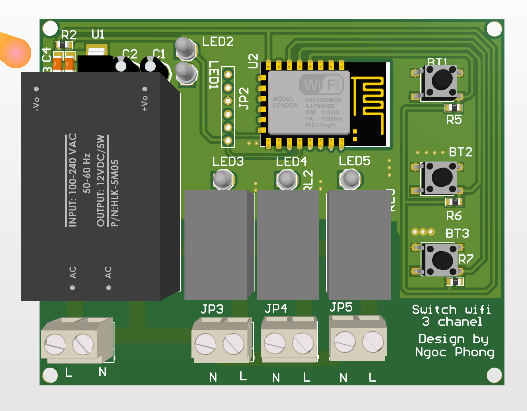
*Hình 2. 3 Mặt top layer*

*Bottom layer:* Gồm các dây điện đi ở mặt sau của mạch



*Hình 2. 4 Mặt bottom layer*

*3D layout mode:*

**

*Hình 2. 5 3D layout*

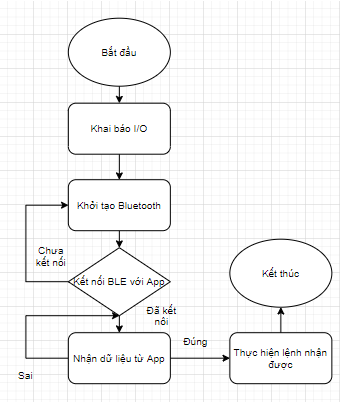
Kích thước mạch

### *Sơ đồ mạch in PCB*

Hình 2. 6 Mạch in PCB

### *Lưu đồ thuật toán*

*a, Lưu đồ toàn hệ thống*

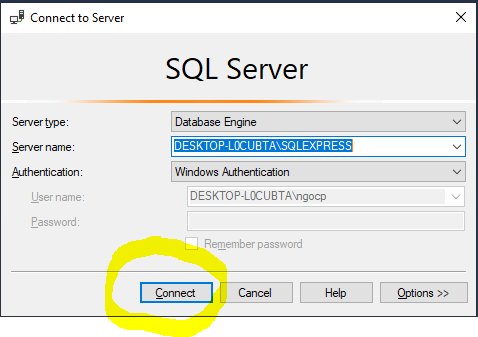
**

Hình 2. 7 Sơ đồ thuật toán hệ thống

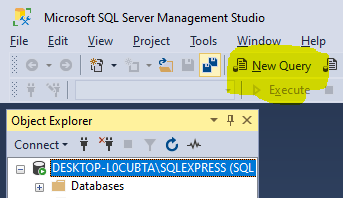
### *Xây dựng cơ sở dữ liệu*

B1: Mở SSMS Studio

B2: Chọn kết nối để connect tới server là máy tính của bạn



B3: Chọn New Query để bắt đầu tạo CSDL



B4: Bắt đầu tạo CSDL bằng các câu lệnh SQL

B5: Cơ sở dữ liệu

create table state\_ctr

(

id int primary key identity,

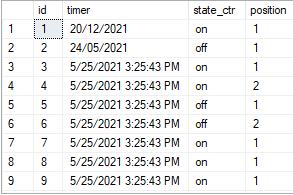
timer nvarchar(255) not null,

state\_ctr nvarchar(255) not null,

position nvarchar(255) not null

)

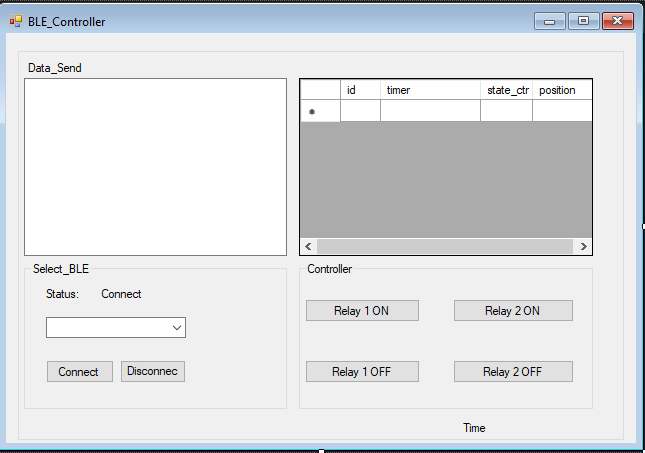
B6: Kết quả CSDL



Mô tả các trường dữ liệu:

* Id: lưu số thứ tự
* Timer: Lưu trữ thời gian thao tác
* State\_ctr: lưu trữ trạng thái điều khiển
* Position: Vị trí relay điều khiển

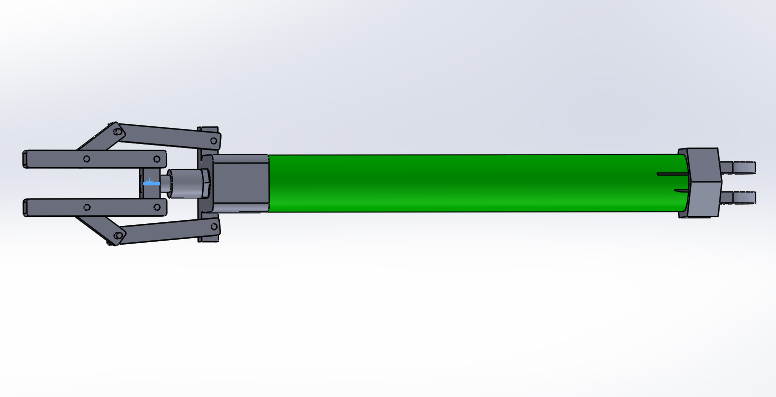
### *Xây dựng phần mểm*



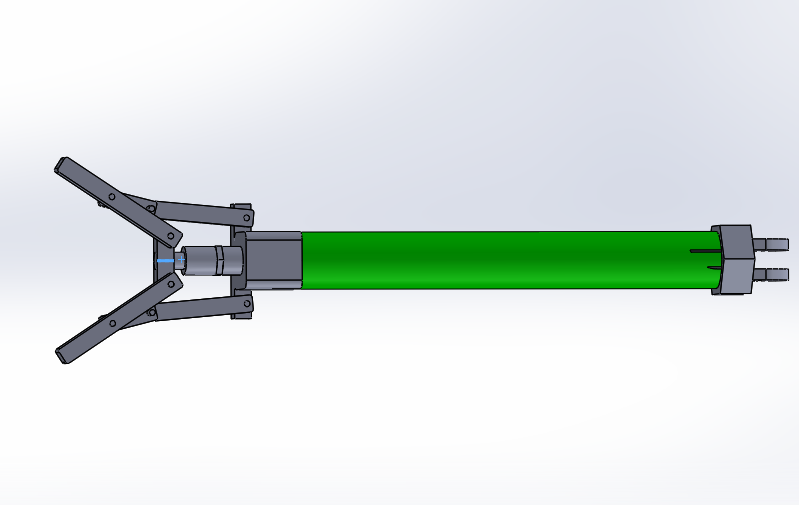
Mô tả phần mềm:

* COM : quét các cổng COM hoạt động trên máy tính(Khi đã kết nối BLE)
* Status : Cho biết phần mềm đã kết nối với cổng COM hay chưa
* Connect : Khi đã chọn cổng COM thì click vào để phần mềm kết nối BLE với VĐK
* Disconnect : Ngắt kết nối phần mềm với VĐK
* Data\_Send: Hiển thị dữ liệu vừa gửi đi
* Datagridview : Hiển thị cơ sở dữ liệu
* Relay 1ON : Gửi tín hiệu bật relay1
* Relay 1ON : Gửi tín hiệu bật relay2
* Relay 2OFF: Gửi tín hiệu tắt relay2
* Relay 2ON : Gửi tín hiệu bật relay2
* Time: Hiển thị thời gian hiện tại

### *Mô phỏng cơ cấu gắp bằng SolidWork*



*Hình 3. 1 Khi gắp vật và kẹp lại*



*Hình 3. 2 Khi cơ cấu mở ra*

*Mô tả:* Khi pittong bị khí nén đẩy từ phía dưới ra thì cơ cấu chấp hành gắn ở đầu xilanh sẽ mở ra còn khi van điện từ đổi hướng thì khí nén sẽ chuyển hướng lên đầu pittong và đẩy pittong xuống cùng với đó là cơ cấu kẹp khép lại gắp lấy vật

## **Giới thiệu một số công cụ, thiết bị sử dụng**

### *Esp32*

*a, Giới thiệu*

ESP32-WROOM-32 là mô đun MCU đa dụng, mạnh mẽ và được sử dụng rộng rãi[trong thiết kế mạch PCB](https://vnpcb.com/dat-mach-pcb) Wifi- Bluetooth, BLE được ứng dụng rất phổ biến cho nhiều ứng dụng về IoT hiện nay. Phạm vi ứng dụng từ mạng sensor tiết kiệm năng lượng đến những ứng dụng với tác vụ phức tạp nhất, như mã hóa âm thanh, âm nhạc trực tuyến đến giải mã MP3.

Lõi của module là họ chip ESP32-D0WDQ6, chip nhúng được thiết kế cho khả năng mở rộng và tùy biến cao. Có đến 2 lõi CPU độc lập có thể điều khiển, tần số clock của CPU có thể được điều chỉnh tử 80MHZ đến 240 Mhz. Người lập trình có thể tắt CPU để sử dụng bộ đồng xử lý công suất thấp  để theo dõi sự thay đổi hoặc vượt ngưỡng của các ngoại vi . ESP32 tích hợp bộ ngoại vi khá phong phú từ cảm biến điện dung, cảm biến Hall, SD card, Ethernet, SPI tốc độ cao, UART, I2S hay I2C.

Việc tích hợp cả Bluetooth, BLE và Wifi đảm bảo cho khả năng ứng đáp ứng nhiều loại ứng dụng khác nhau và module đó sử dụng với ngoại vi, thiết bị nào: wifi cho phép kết nối rộng rãi về mặt vật lý ra Internet qua Wi-fi router, trong khi sử dụng Bluetooth cho phép người dùng thuận tiện khi kết nối với smartphone, hay thiết bị beacon tiết kiệm điện. Ở chế độ ngủ, chíp ESP32 tiêu thụ dòng dưới 5 µA, phù hợp với những thiết kế mạch dùng pin hay thiết bị đeo được. Tốc độ truyền thông cho phép lên đến 150 Mbps, và công suất tín hiệu khoảng 20 dBm trên anten cho phép phạm vi tín hiệu xa. Như vậy module này có thông số kỹ thuật thuộc dạng đầu bảng trên thị trường cũng như hiệu suất, độ tin cậy tốt nhất cho tích hợp, thiết kế ứng dụng điện tử, tự động hóa, đòi hỏi phạm vi hoạt động rộng, tiết kiệm năng lượng cũng như khả năng kết nối đa dạng.

Hệ điều hành chạy được trên ESP32 là FreeRTOS vơi LwIP, TLS 1.2. Hỗ trợ update firmware qua OTA mã hóa, điều này cho phép nhà phát triển sản phẩm có thể nâng cấp phần mềm sản phẩm ngay cả khi thiết bị đang được sử dụng một cách tiết kiệm tiền bạc và nhân lực.

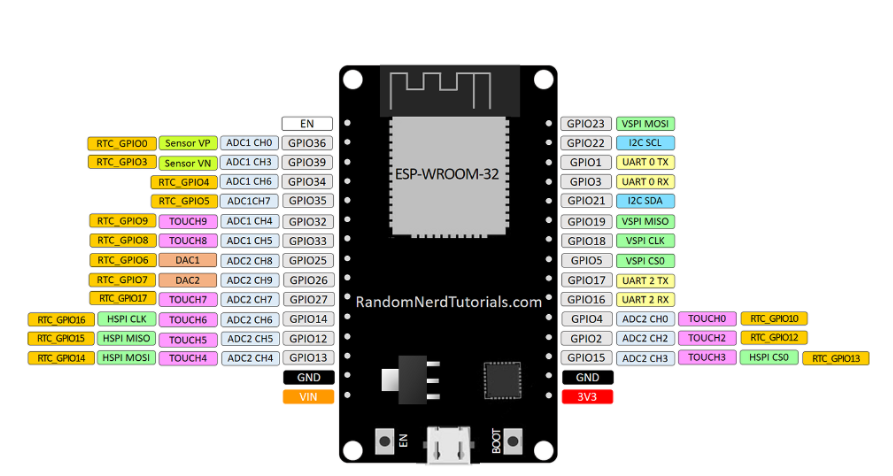


Hình 2. Module ESP32

*b, Thông số kỹ thuật*

|  |  |  |
| --- | --- | --- |
| **Categories** | **Items** | **Specifications** |
| Certification | RF certification | FCC/CE-RED/IC/TELEC/KCC/SRRC/NCC |
| Wi-Fi certification | Wi-Fi Alliance |
| Bluetooth certification | BQB |
| Green certification | RoHS/REACH |
| Test | Reliablity | HTOL/HTSL/uHAST/TCT/ESD |
| Wi-Fi | Protocols | 802.11 b/g/n (802.11n up to 150 Mbps) |
| A-MPDU and A-MSDU aggregation and 0.4 *µ*s guard interval support |
| Frequency range | 2.4 GHz ~ 2.5 GHz |
| BLUETOOTH Hardware | Protocols | Bluetooth v4.2 BR/EDR and BLE specification |
| Radio | NZIF receiver with –97 dBm sensitivity |
| Class-1, class-2 and class-3 transmitter |
| AFH |
| Audio | CVSD and SBC |
| Module interfaces | SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC |
| On-chip sensor | Hall sensor |
| On-board clock | 40 MHz crystal |
| Operating voltage/Power supply | 2.7 ~ 3.6V |
| Operating current | Average: 80 mA |
| Minimum current delivered by power supply | 500 mA |
| Recommended operating tem perature range | –40°C ~ +85°C |
| Package size | (18.00±0.10) mm x (25.50±0.10) mm x (3.10±0.10) mm |

*c, Chức năng các chân*



Hình 2. 9 Sơ đồ chân Esp32

ESP32 có rất nhiều kit phát triển khác nhau nên các bạn lưu ý bài hướng dẫn này chỉ viết cho ESP32 DEVKIT V1 – DOIT loại 30 chân, các kit khác tương tự. Trên kit có tổng cộng 25 chân GPIO tuy nhiên cũng giống với NodeMCU, khi dùng GPIO trên ESP32 DEVKIT cần lưu ý:

GPIO34, 35, 36, 39 chỉ dùng được chức năng INPUT, không dùng được chức năng OUTPUT.

GPIO1, GPIO3: Giống như NodeMCU 2 chân này cũng được nối đến bộ UART0 và ESP32 Dev Kit cũng nạp code thông qua bộ UART này nên tránh sử dụng 2 chân này cho chức năng I/O.

GPIO0, 2, 4, 5, 12, 15 đã được định sẵn mức logic bên trong module ESP32 để phục vụ quá trình nạp code, nên tránh sử dụng các GPIO này.

Ngoài ra đối với những bạn thiết kế mạch sử dụng module ESP32 rời (module tích hợp trên ESP32 Dev Kit) thì ngoài lưu ý ở trên còn lưu ý thêm các chân từ GPIO6 đến GPIO11. Các chân này dùng để giao tiếp SPI với External Flash nên không thể dùng được chức năng I/O và trên ESP32 Dev Kit đã ẩn đi các chân này.

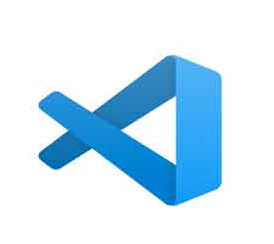
Một lưu ý quan trọng khác cho cả NodeMCU và ESP32 Dev Kit là mức điện áp INPUT tối đa mà các chân GPIO có thể hoạt động được là 3.6V (khác với các board Arduino là 5V) vì vậy nếu tác động mức điện áp lớn hơn 3.6V vào chân INPUT sẽ làm hỏng chân GPIO.

### *Visual Studio Code và Platform IO*

*a, Visual Studio Code*

Là một trình biên tập lập trình code miễn phí dành cho Windows, Linux và macOS, Visual Studio Code được phát triển bởi Microsoft. Nó được xem là một sự kết hợp hoàn hảo giữa IDE và Code Editor.

Visual Studio Code hỗ trợ chức năng debug, đi kèm với Git, có syntax highlighting, tự hoàn thành mã thông minh, snippets, và cải tiến mã nguồn. Nhờ tính năng tùy chỉnh, Visual Studio Code cũng cho phép người dùng thay đổi theme, phím tắt, và các tùy chọn khác.



Hình 2. 10 Visual Studio Code

Một số tính năng của Visual Studio Code:

* Hỗ trợ nhiều ngôn ngữ lập trình
* Hỗ trợ đa nền tảng
* Cung cấp kho tiện ích mở rộng
* Kho lưu trữ an toàn
* Hỗ trợ Web, lưu trữ dữ liệu đa dạng phân cấp

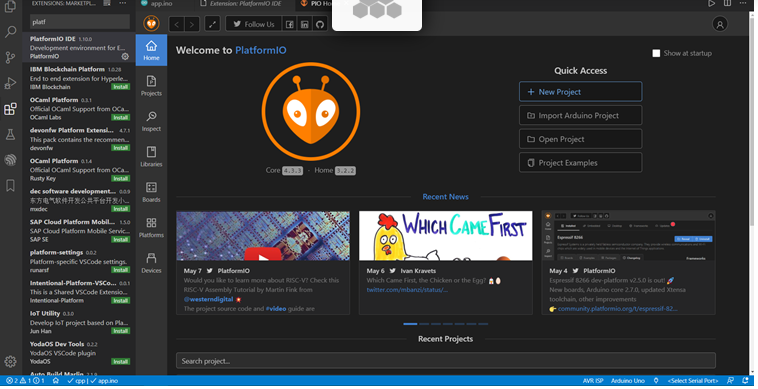
*b, Platform IO*

[PlatformIO](http://platformio.org/) là mội plugin có khả năng lập trình Arduino/ ARM mbed nó hỗ trợ tới hơn 250 board khác nhau.



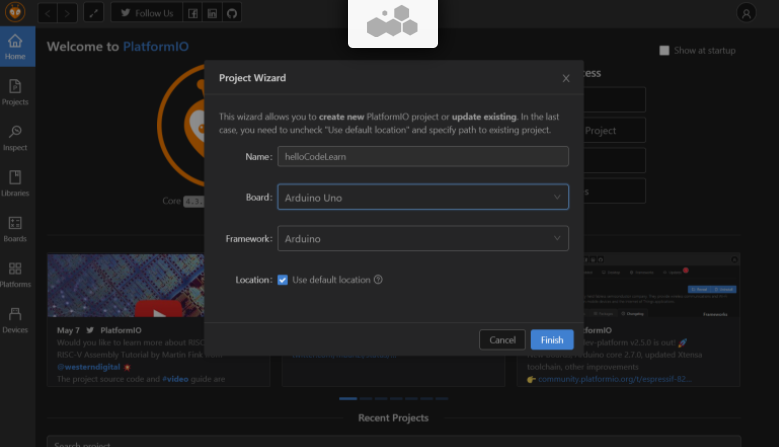
Hình 2. 11 Platform IO

Để bắt đầu với project mới, các bạn chọn New Project:



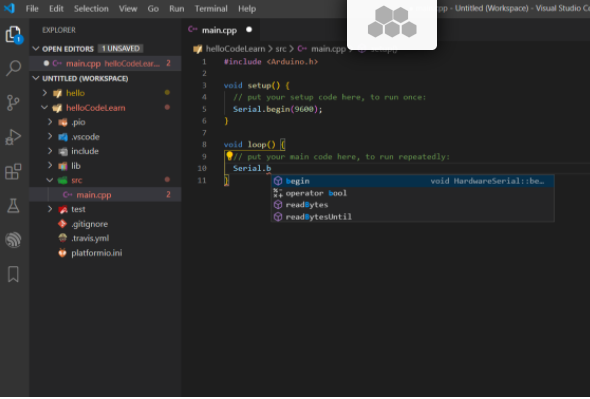
Hình 2. 12 Tạo Project

Điền thông tin vào, nhớ rằng platformio sẽ cấu hình luôn board mạch bạn sử dụng ngay khi khởi tạo project:



Hình 2. 13 Điền thông tin project

Kết quả sau khi tạo project:



Hình 2. 14 Cấu trúc project sau khi tạo xong

### *Altium Design*

*a, Giới thiệu*

Altium Designer trước kia có tên gọi quen thuộc là Protel DXP, là một trong những công cụ vẽ mạch điện tử mạnh nhất hiện nay. Được phát triển bởi hãng Altium Limited. Altium designer là một phần mềm chuyên nghành được sử dụng trong thiết kế mạch điện tử. Nó là một phần mềm mạnh với nhiều tính năng thú vị, tuy nhiên phần mềm này còn được ít người biết đến so với các phần mềm thiết kế mạch khác như orcad hay proteus.



Hình 2. 15 Altium Design

*b, Một số đặc trưng*

Giao diện thiết kế, quản lý và chỉnh sửa thân thiện, dễ dàng biên dịch, quản lý file, quản lý phiên bản cho các tài liệu thiết kế.

Hỗ trợ mạnh mẽ cho việc thiết kế tự động, đi dây tự động theo thuật toán tối ưu, phân tích lắp ráp linh kiện. Hỗ trợ việc tìm các giải pháp thiết kế hoặc chỉnh sửa mạch, linh kiện, netlist có sẵn từ trước theo các tham số mới.

Mở, xem và in các file thiết kế mạch dễ dàng với đầy đủ các thông tin linh kiện, netlist, dữ liệu bản vẽ, kích thước, số lượng…

Hệ thống các thư viện linh kiện phong phú, chi tiết và hoàn chỉnh bao gồm tất cả các linh kiện nhúng, số, tương tự…

 Đặt và sửa đối tượng trên các lớp cơ khí, định nghĩa các luật thiết kế, tùy chỉnh các lớp mạch in, chuyển từ schematic sang PCB, đặt vị trí linh kiện trên PCB.

Mô phỏng mạch PCB 3D, đem lại hình ảnh mạch điện trung thực trong không gian 3 chiều, hỗ trợ MCAD-ECAD, liên kết trực tiếp với mô hình STEP, kiểm tra khoảng cách cách điện, cấu hình cho cả 2D và 3D

Hỗ trợ thiết kế PCB sang FPGA và ngược lại.

*c, Tóm tắt các bước thiết kế mạch trên Altium*

- Đặt ra các yêu cầu bài toán.

-  Lựa chọn linh kiện.

- Thiết kế mạch nguyên lý.

- Lựa chọn các chân linh kiện để chuyển sang mạch in Update mạch nguyên lý sang mạch in.

-  Lựa chọn kích thước mạch in Sắp sếp các vị trí các loại linh kiện  như điện trở , tụ điện, IC...

-  Đặt kích thước các loại dây nối.

- Đi dây trên mạch.

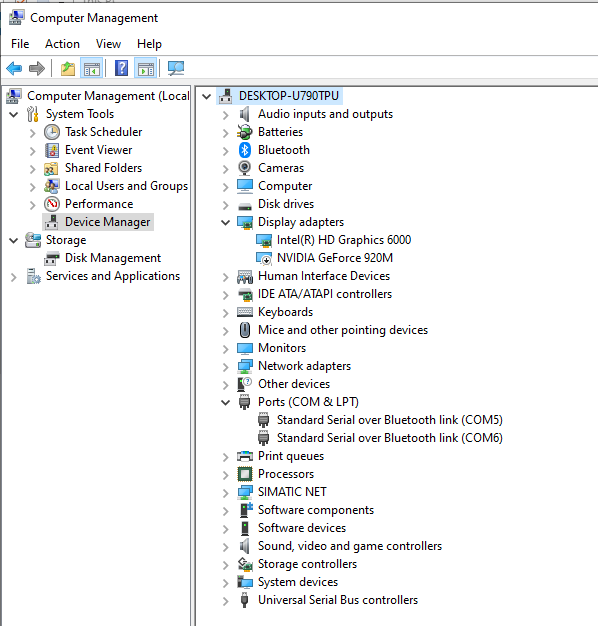
- Kiểm tra toàn mạch.

# **CHƯƠNG 3: KẾT QUẢ ĐÁNH GIÁ**



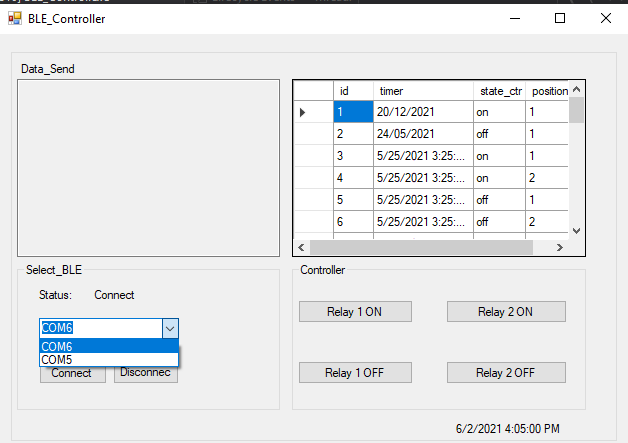
## **Các bước sử dụng hệ thống**

*Bước 1:* Cấp nguồn cho mạch điều khiển và bật kết nối Bluetooth ở máy tính và kết nối với ESP32 cùng với đó kiểm tra cổng COM kết nối BLE ở máy tính ở phần manager



Hình 3. 3 Tìm cổng COM kết nối BLE

*Bước 2:* Mở phần mềm điều khiển C# để chọn cổng COM



Hình 3. 4 Mở app điều khiển

*Bước 3:*Giao diện điều khiển của app cho người dùng thao tác theo ý muốn

Hình 3. 5 Giao diện điều khiển của app

*Bước 4*: Ghép nối mạch điều khiển và mạch relay

*Hình 3. 6 Kết nối mạch điều khiển và mạch relay*

*Bước 5*:

## **Đánh giá hệ thống**

### *Ưu điểm*

Hệ thống đáp ứng được đúng những yêu cầu đã được được đưa ra.

Giúp người sử dụng có thể điều khiển được thiết bị từ xa

Giúp người dùng tiết kiệm điện năng

Tận dụng tối đa khả năng điều khiển của bluetooth

Thay thế được điều khiển truyền thống

Có thể điều khiển bằng máy tính hoặc điện thoại thông minh

### *Nhược điểm*

Thiết bị chỉ sử dụng được khi có kết nối bluetooth và phần mềm điều khiển

Không có khả năng điều khiển bằng tay trên mạch điều khiển

Khoảng cách điều khiển hạn chế (3m - 5m).

## **Kết luận**

Trải qua thời gian nghiên cứu và thực hiện đề tài đã giúp em hiểu ra hơn về vai trò, các ứng dụng của vi điều khiển trong thực tế, cũng như cách lập trình cho vi điều khiển Esp8266, cách thiết kế mạch trên phần mềm Altium.

Ngoài ra qua đề tài này em cũng hiểu rõ hơn về các cách truyền dữ liệu qua Bluetooth và các ứng dụng của lĩnh vực IOT trong đời sống từ đó có những hướng nghiên cứu và phát triển những thiết bị thông minh trong tương lai

# **Tài liệu tham khảo**

[1] <https://github.com/crankyoldgit/IRremoteESP8266>

[2] <https://www.instructables.com/Web-IR-Remote-With-Esp8266-NodeMCU>

[3] <https://www.espressif.com/en/products/socs/esp8266>

# **Phụ lục**

#include <Arduino.h>

#include "BluetoothSerial.h"

#include "ArduinoJson.h"

#include <string.h>

BluetoothSerial bleSerial;

StaticJsonBuffer<300> doc;

//String json = "{\"name\":\"phong\", \"age\": 15, \"add\": \"this is add\"}";

enum ctr\_cmd

{

  CH1\_ON = 1,

  CH1\_OFF = 2,

  CH2\_ON = 3,

  CH2\_OFF = 4

};

void control\_IO(int cmd);

void parseJson(char \*cmd)

{

  JsonObject &data = doc.parseObject(cmd);

  if (!data.success())

  {

    Serial.print("Phong\_hg56 : Faild");

    return;

  }

  String name = data["name"].as<String>();

  Serial.println(name);

  int cmds = data["cmd"];

  Serial.println(cmds);

  control\_IO(cmds);

  doc.clear();

}

char data[200];

void setup()

{

  Serial.begin(115200);

  bleSerial.begin("BLE\_PHONG");

  Serial.println("The device started, now you can pair it with bluetooth!");

  pinMode(GPIO\_NUM\_14, OUTPUT);

  pinMode(GPIO\_NUM\_27, OUTPUT);

}

void loop()

{

  if (bleSerial.available())

  {

    bleSerial.setTimeout(10);

    String msg = bleSerial.readString();

    //Serial.println(msg.c\_str());

    parseJson((char \*)msg.c\_str());

  }

}

void control\_IO(int cmd)

{

  switch (cmd)

  {

  case CH1\_ON:

  {

    Serial.println("relay 1 on");

    digitalWrite(GPIO\_NUM\_14, HIGH);

  }

  break;

  case CH1\_OFF:

  {

    Serial.println("relay 1 off");

    digitalWrite(GPIO\_NUM\_14, LOW);

  }

  break;

  case CH2\_ON:

  {

    Serial.println("relay 2 on");

    digitalWrite(GPIO\_NUM\_27, HIGH);

  }

  break;

  case CH2\_OFF:

  {

    Serial.println("relay 2 off");

     digitalWrite(GPIO\_NUM\_27, LOW);

  }

  break;

  }

}