|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | | | | |  |
|  | **ĐẠI HỌC BÁCH KHOA HÀ NỘI TRƯỜNG ĐIỆN – ĐIỆN TỬ**  **---****---**    **BÁO CÁO MÔN HỌC**  **LÝ THUYẾT MẠCH & LÝ THUYẾT THÔNG TIN**  **(Mã học phần ET2050 & ET2072)**  ***ĐỀ TÀI: SMART PARKING***  ***Giảng viên: Nguyễn Hữu Phát*** | | | | |  |
|  | | ***Thực hiện: Nhóm 8*** | | |  | |
| **STT** | **Họ và Tên** | **MSSV** |
| 1 | Nguyễn Trần Trí Dũng | 20213691 |
| 2 | Đỗ Thị Thúy Nga | 20213720 |
| 3 | Đỗ Trang Đức Duy | 20213692 |
| 4 | Kim Minh Quang | 20213726 |
| 5 | Nguyễn Thị Minh Tâm | 20210766 |
| ***Hà Nội, 07/2023*** | | | | | | |
|  |  | | | | |  |

MỤC LỤC

Contents

[LỜI MỞ ĐẦU: 7](#_Toc140524107)

[I. Giới thiệu 8](#_Toc140524122)

[1.1 Ý tưởng: 8](#_Toc140524123)

[1.2 Khảo sát: 8](#_Toc140524124)

[1.3 Thị trường: 10](#_Toc140524131)

[II. Tiêu chí thiết kế 10](#_Toc140524135)

[2.1 Chỉ tiêu chức năng 10](#_Toc140524136)

[2.2 Chỉ tiêu phi chức năng 11](#_Toc140524137)

[III. Cấu tạo 11](#_Toc140524138)

[IV. Dữ liệu 14](#_Toc140524141)

[4.1 Sơ đồ khối: 14](#_Toc140524142)

[4.2 Sơ đồ mạch: 15](#_Toc140524144)

[V. Triển khai và thử nghiệm sản phẩm: 18](#_Toc140524146)

[5.1 Code và giải thích: 18](#_Toc140524147)

[5.2 Sản phẩm: 24](#_Toc140524152)

[VI. Kết luận: 25](#_Toc140524154)

# LỜI MỞ ĐẦU:

# Ngày nay, nước ta số lượng ô tô và xe máy ngày càng tăng nhanh chóng dẫn

# đến việc khó khăn trong việc đỗ xe của người sử dụng. Thực trạng này đỏi

# hỏi sự ra đời của các bãi đỗ xe thông minh. Lấy ý tưởng từ thực tế hiện tại,

# nhóm chúng em cùng tìm hiểu và thực hiện đề tài Smart Parking. Bài báo

# cáo là toàn bộ quá trình nghiên cứu và phát triển đề tài “Smart Parking”.

# Chúng em xin gửi lời cảm ơn chân thành đến giảng viên hướng dẫn - thầy

# Nguyễn Hữu Phát đã dẫn đăt và truyền đạt những kỹ năng, kiến thức cho

# chúng em trong suốt thời gian học vừa qua. Lý thuyết mạch và Lý thuyết

# thông tin là hai bộ môn thú vị và giúp ích rất nhiều cho tương lai của sinh

# viên Điện tử chúng em. Qua việc thực hiện đề tài, nhóm đã tiếp thu thêm

# được một lượng kiến thức không hề nhỏ. Chúng em đã cố gắng hoàn thành

# báo cáo một cách chỉn chu nhất. Nếu có sai sót, kính mong thầy xem xét và

# góp ý để bài báo cáo của chúng em được hoàn thiện và tốt hơn.

# !

# 

# Giới thiệu

# 1.1. Ý tưởng:

**What:** Tình trạng ùn tắc bãi đỗ xe, việc đỗ xe tốn nhiều thời gian và nhân lực

**Why:** Tiết kiệm thời gian gửi xe, hạn chế những sai sót và tiện lợi cho người dân

**Who:** Mọi đối tượng sử dụng phương tiện cá nhân có nhu cầu gửi xe

**When:** Khi tình trạng các bãi đỗ xe quá tải, hoặc khi có quá nhiều gửi xe cùng lúc

**Where:** Các trung tâm thương mại; công ty, trường học; khu dân cư đông đúc, chung cư

**How:** Cần để giảm tình trạng ùn tắc, thuận tiện hơn trong việc tìm vị trí đỗ xe

# Khảo sát:

# Chúng em đã đi khảo sát nhiều người về một số vấn đề liên quan:

# 

# 

# 

# - Như vậy hiện nay, người dân ai cũng có phương tiện cá nhân riêng nhằm phục vụ mục đích đi làm, đi học, đi chơi, và việc phải gửi xe tại các nơi công cộng đã là chuyện bình thường với bao người. Tuy nhiên cũng có thể thấy vẫn có khá nhiều vấn đề bất cập được chỉ ra trên bản khảo sát tại các bãi đỗ xe thông thường.

# - Từ đó, Smart Parking ra đời, đáp ứng các nhu cầu của người dân về một bãi đỗ xe tiện lợi, thân thiện, đồng thời giảm thiểu ùn tắc giao thông. Mặc dù đây vẫn còn là

# một khái niệm mới với đa số người dân, song với tính thực tiễn của Smart Parking mà trong tương lai sẽ được nhiều người biết tới và tìm đến hơn

# 1.3 Thị trường:

# Một số mô hình đỗ xe hiện có:

# 

# Thiết kế bãi đỗ xe thông minhTop 15 hệ thống phần mềm giữ xe thông minh được ưa chuộng

## 

# Tiêu chí thiết kế

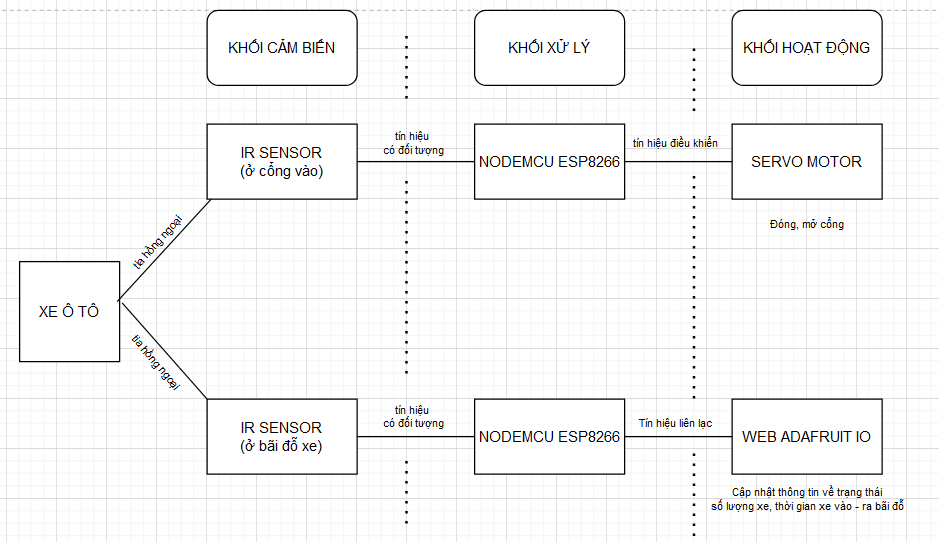
# Chỉ tiêu chức năng

* IR Sensor:
  + - Input: Tia hồng ngoại phát ra từ vật thể, trong bán kính 20cm
    - Output: Tín hiệu gửi đến ESP 8266.
* NodeMCU ESP 8266:
  + - Input: Tín hiệu từ IR sensor
    - Output:

- Tín hiệu điều khiển: Biểu thị trạng thái chiếm chỗ của từng chỗ đỗ xe, mở và đóng cổng.

- Tín hiệu liên lạc: Cung cấp thông tin cập nhật theo thời gian thực về trạng thái lấp đầy của từng chỗ đỗ xe.

* Servo Motor
  + - Input: Tín hiệu từ NodeMCU ESP 8266
    - Ouput: Đóng, mở cổng
* Sơ đồ khối quan hệ Input – Output:



# Chỉ tiêu phi chức năng

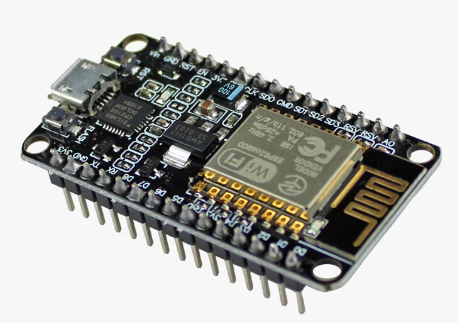
## 

|  |  |
| --- | --- |
| **Năng lượng** | NodeMCU ESP 8266: 3,3V  IR Sensor: 2.5-6V DC  Servo Motor: 3-6V DC  Góc quay: 180° |
| **Tiêu chuẩn** | An toàn: HTTPS, ISO 27001  Các giao thức: TLS/SSL, IPv6; LoRaWAN |
| **Môi trường:** | Bãi đỗ xe ngoài trời, trong nhà, các khu đô thị |
| **Chi phí** | ~300.000VND |
| **Ngoại quan cơ khí:** | Số lượng cổng ra/vào: 1/1  Độ rộng cửa ra vào: 50mm  Kích thước vị trí đỗ xe: 50mmx50mm  Đường kính bãi đỗ xe: 400mm  Diện tích: 1,2x10^5 mm^2  Vật liệu: bìa  Số lượng chỗ đỗ xe: 3 |

# Cấu tạo

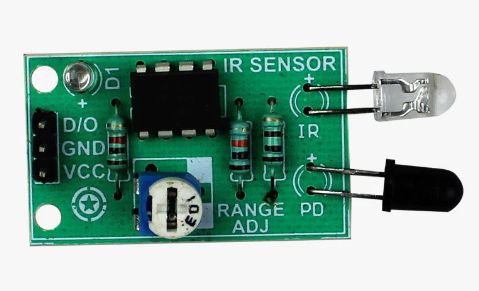
# Hardware:

# [NodeMCU ESP8266](https://quartzcomponents.com/products/nodemcu-development-board):



* IC chính: ESP8266
* Phiên bản firmware: NodeMCU Lua
* Chip nạp và giao tiếp UART: CP2102.
* GPIO tương thích hoàn toàn với firmware Node MCU.
* Cấp nguồn: 5VDC MicroUSB hoặc Vin.
* GIPO giao tiếp mức 3.3VDC
* Tích hợp Led báo trạng thái, nút Reset, Flash.
* Tương thích hoàn toàn với trình biên dịch Arduino.
* Kích thước: 25 x 50 mm

[**IR Senso**](https://quartzcomponents.com/products/ir-line-follower-sensor-module)**r ( 5 cái):**



- Cảm biến vật cản hồng ngoại có khả năng thích nghi với môi trường, có một cặp truyền và nhận tia hồng ngoại. Tia hồng ngoại phát ra một tần số nhất định, khi phát hiện hướng truyền có vật cản (mặt phản xạ), phản xạ vào đèn thu hồng ngoại, sau khi so sánh, đèn sẽ sáng lên, đồng thời đầu cho tín hiệu số đầu ra (một tín hiệu bậc thấp).. Độ nhạy sáng của cảm biến vật cản hồng ngoại được điều chỉnh bằng chiết áp, cảm biến dễ lắp ráp, dễ sử dụng,….

* Điện áp làm việc: 3.3V – 5V DC
* Chất liệu sản phẩm: Nhựa
* Lỗ vít 3 mm, dễ dàng cố định, lắp đặt.
* Kích thước: 3.2cm \* 1.4cm
* Khoảng cách: 2 ~ 5cm
* Đèn hiển thị màu vàng

[**Servo Motor**](https://quartzcomponents.com/products/tower-pro-sg90-servo-9-gms-mini-micro-servo-motor)**(2 cái):**

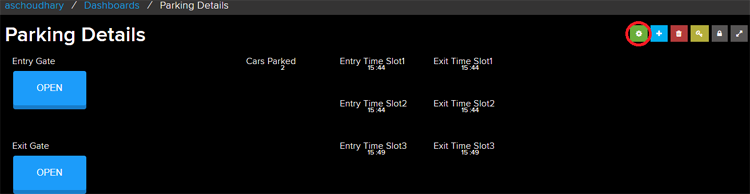


* Động cơ Servo RC Analog quay liên tục
* Điện áp hoạt động: 4.8V đến 6.0V
* Chiều dài chì: 200mm
* Tốc độ tối đa @ 6 V: 130 RPM
* Tốc độ tối đa @ 4.8 V: 100 RPM
* Mô men xoắn @ 6 V: 1.5kg/m
* Mô men xoắn @ 4.8 V: 1.3kg/m
* Hệ thống dây điện:
  + Cam cho tín hiệu RC
  + Đỏ for VCC (bộ nguồn dương)
  + Nâu for GND (bộ nguồn âm)
* Trọng lượng: 9g.
* Kích thước: 23.2mm x 12.5mm x 22mm.

**Online Services: Adafruit IO**

- Adafruit.io là một dịch vụ lưu trữ và trao đổi dữ liệu trực tuyến được phát triển bởi công ty Adafruit Industries. Adafruit.io được tạo ra để hỗ trợ việc kết nối và thu thập dữ liệu từ các thiết bị Internet of Things (IoT) một cách dễ dàng và thuận tiện, sau đó gửi dữ liệu lên trên server để người dung theo dõi trực tiếp theo thời gian thực.

- Adafruit.io giao tiếp với các thiết bị IOT thông qua 1 số giao thức bao gồm MQTT (Message Queuing Telemetry Transport) và RESTful API.



Giao diện người dùng Adafruit IO

# IV. Dữ liệu

# 4.1 Sơ đồ khối:

# 

# 4.2 Sơ đồ mạch:

# 

**Servo động cơ Tower Pro SG90**:

- Có 3 chân và các chân này được đánh số như sau:

1. Chân GND (Ground): Đây là chân mát đất, kết nối với mát đất (0V) của nguồn cung cấp.

2. Chân VCC (Power): Đây là chân nguồn, kết nối với nguồn cung cấp có điện áp từ 4.8V đến 6V. Lưu ý rằng nguồn cung cấp phải có đủ dòng điện để đáp ứng yêu cầu của servo.

3. Chân Signal (Tín hiệu): Đây là chân tín hiệu, kết nối với vi điều khiển hoặc bộ điều khiển servo thông qua tín hiệu PWM (Pulse Width Modulation). Tín hiệu PWM này sẽ điều khiển vị trí của servo.

Chân GND là chân đầu tiên ở phía trái (khi nhìn từ phía trước) và chân Signal là chân thứ ba ở phía bên phải



- Hoạt động:

* Servo động cơ Tower Pro SG90 dựa vào nguyên tắc điều khiển hồi quy, trong đó các tín hiệu điều khiển định hướng và điều khiển vị trí được áp dụng để giữ cho servo đưa ra góc quay nhất định.
* Servo động cơ SG90 chứa một bộ điều khiển điện tử và một bộ cơ học. Bộ cơ học bao gồm một bộ truyền động, potentiometer (biến trở), và một hệ thống cơ cấu phản hồi âm (feedback loop). Quá trình hoạt động cơ bản của servo:
* Tín hiệu điều khiển: Người dùng hoặc bộ điều khiển đưa ra tín hiệu PWM tới chân Signal của servo. Tín hiệu này có chu kỳ và độ rộng xung biến đổi. Độ rộng xung thay đổi từ 1ms đến 2ms, và chu kỳ xung thường là khoảng 20ms.
* Điều khiển vi điều khiển: Bộ điều khiển điện tử bên trong servo phân tích tín hiệu PWM nhận được. Dựa vào độ rộng xung, nó xác định góc quay cần thiết để đưa trục servo về vị trí mong muốn.
* Điều khiển cơ học: Bộ điều khiển điện tử chuyển đổi tín hiệu điều khiển thành các tín hiệu điện thế và dòng điện để điều khiển động cơ bên trong servo. Động cơ servo là động cơ DC, và điện áp và dòng điện điều khiển sẽ xác định tốc độ và hướng quay của động cơ.
* Cơ cấu phản hồi âm: Khi trục servo quay, bộ cơ học trong servo cũng chuyển động, làm thay đổi giá trị của potentiometer. Potentiometer là một thiết bị điện trở có thể điều chỉnh, và giá trị điện trở của nó phụ thuộc vào góc quay
* Lặp lại quá trình điều khiển: Bộ điều khiển điện tử liên tục so sánh giá trị điện trở của potentiometer với giá trị góc quay cần đạt được từ tín hiệu điều khiển. Nếu sự khác biệt giữa hai giá trị này, bộ điều khiển điện tử điều chỉnh điện áp và dòng điện tới động cơ để đưa servo đến vị trí mong muốn. Quá trình này tiếp tục cho đến khi servo đạt được vị trí và góc quay mong muốn.
* Nhờ cơ cấu phản hồi âm của potentiometer mà servo có khả năng duy trì vị trí và độ chính xác cao, giúp nó trở thành một giải pháp tốt cho các ứng dụng yêu cầu điều khiển chính xác góc quay.

**IR Sensor:**

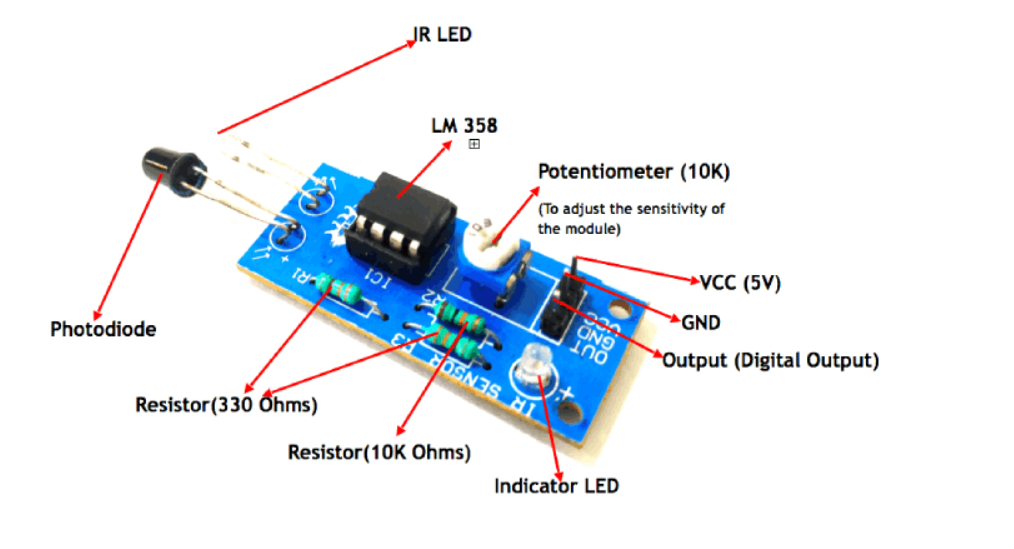
- Mô-đun cảm biến hồng ngoại bao gồm một đèn LED phát sóng hồng ngoại (IR) để phát ra ánh sáng IR, ánh sáng này sau đó sẽ được nhận bởi một đèn LED thu sóng hồng ngoại nếu nó được phản chiếu bởi bất kỳ vật thể nào ở phía trước nó.

- 3 chân của Mô-đun:

Mô-đun có ba chân, trong đó hai chân được sử dụng để cung cấp nguồn cho cảm biến và chân thứ ba cung cấp đầu ra cao / thấp dựa trên khoảng cách của vật thể ở phía trước nó. Mô-đun cảm biến theo dõi đường cũng có một điều chỉnh cơ khí để điều chỉnh độ nhạy của cảm biến.

- Mô-đun cảm biến hồng ngoại này bao gồm một op-amp LM358, một đèn LED hồng ngoại, photodiode, một potentiometer 10K, đèn chỉ báo, 2 resistor 330 ohms và 10K ohms.

- Sơ đồ chân mô-đun cảm biến IR Line Follower:



- Cách hoạt động của mô-đun cảm biến IR:

* Trường hợp 1: Khi một vật thể (không phải bề mặt màu đen) được đặt trước đèn LED hồng ngoại

Trong tình huống này, sóng hồng ngoại phát ra từ đèn LED IR sẽ chiếu vào vật thể (được đặt trong đường truyền sóng IR) và được phản chiếu lại. Photodiode được đặt sao cho nó cảm nhận sóng IR đã phản chiếu. Khi nó cảm nhận sóng IR, trở kháng của nó giảm và nó hoạt động như một mạch ngắn. Dòng điện bắt đầu chảy vào R2 và điện áp xuất hiện ở terminal không đảo. Nếu điện áp ở terminal không đảo lớn hơn điện áp ở terminal đảo, op-amp sẽ cho ra đầu ra cao (5V) và đèn LED bắt đầu sáng. (Lưu ý: điện áp ở terminal đảo có thể thay đổi bằng cách điều chỉnh trở kháng của potentiometer. Điều này được thực hiện để thay đổi độ nhạy của cảm biến. Nếu giá trị potentiometer là 0, thậm chí trước khi photodiode cảm nhận sóng IR, đầu ra của op-amp sẽ là cao. Điều này có nghĩa là cảm biến có độ nhạy cao.)

* Trường hợp 2: Khi không có vật thể nào được đặt trước đèn LED hồng ngoại.

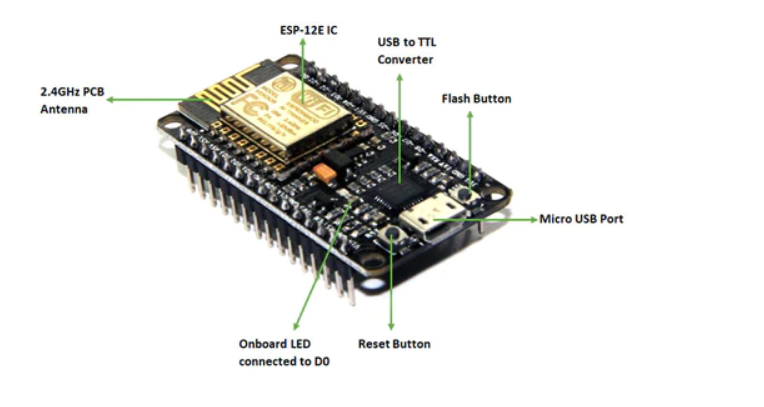
Trong trường hợp này, sóng IR phát ra từ đèn LED IR sẽ không đến photodiode, vì không có vật thể nào có thể phản chiếu lại sóng IR. Do đó, photodiode sẽ cung cấp trở kháng tối đa đến nguồn cấp (Vcc). Vì dòng đi qua resistor là tối thiểu, điện áp tại terminal không đảo sẽ là tối thiểu. Vì điện áp tại terminal không đảo nhỏ hơn điện áp tại terminal đảo, đầu ra từ Op-amp sẽ là giá trị thấp (0V).

**NodeMCU ESP8266:**

- NodeMCU ESP8266 là một firmware mã nguồn mở dựa trên Lua và bo phát triển đặc biệt dành cho các ứng dụng dựa trên IoT (Internet of Things). Nó bao gồm firmware chạy trên SoC (System on Chip) Wi-Fi ESP8266 của Espressif Systems và phần cứng dựa trên module ESP-12. Nó cũng có thể được lập trình bằng Arduino IDE và có thể hoạt động như WiFi Hotspot hoặc kết nối với mạng WiFi.

- Các chân của NodeMCU ESP8266:

* NodeMCU có một chân Analog Input, 16 chân Digital I/O và hỗ trợ giao tiếp chuẩn như SPI, UART, và I2C. NodeMCU có 128 KB RAM và 4MB bộ nhớ Flash để lưu trữ dữ liệu và chương trình.
* Vi xử lý: Tensilica 32-bit RISC CPU Xtensa LX106
* Điện áp hoạt động: 3.3V
* Điện áp đầu vào: 7-12V
* Chân Digital I/O (DIO): 16
* Chân Analog Input (ADC): 1
* UARTs: 1
* SPIs: 1
* I2Cs: 1
* Bộ nhớ Flash: 4 MB
* SRAM: 64 KB
* Tốc độ xung nhịp: 80 MHz



- Hoạt động:

* Bo phát triển NodeMCU được lập trình dễ dàng bằng Arduino IDE, sử dụng cáp micro USB để kết nối với máy tính.

# V. Triển khai và thử nghiệm sản phẩm:

# 5.1 Code và giải thích:

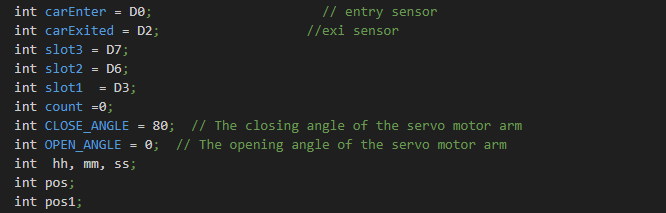
# 

# Đây là phần khai báo thư viện. Thư viện ESP8266WiFi được sử dụng để kết nối với mạng WiFi, Servo để điều khiển servo motor, NTPClient để đồng bộ thời gian từ NTP server, WiFiUdp để kết nối UDP cho NTPClient, Adafruit\_MQTT và Adafruit\_MQTT\_Client để kết nối và giao tiếp với Adafruit IO thông qua MQTT.

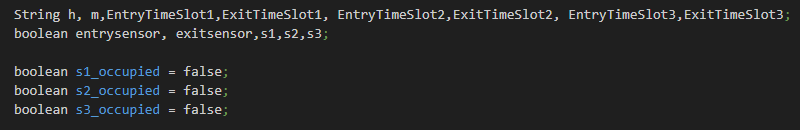
# Đây là phần cấu hình kết nối WiFi và MQTT. Bạn cần điền tên và mật khẩu WiFi của mạng bạn muốn kết nối. Các hằng số MQTT\_SERV, MQTT\_PORT, MQTT\_NAME và MQTT\_PASS xác định địa chỉ MQTT server, cổng MQTT, tên người dùng MQTT và mật khẩu MQTT của Adafruit IO. Bạn cần điều chỉnh các giá trị này để phù hợp với tài khoản Adafruit IO của bạn.

# 

Đây là khởi tạo các đối tượng cho UDP (dùng cho NTPClient), NTPClient để đồng bộ thời gian từ NTP server, và hai đối tượng Servo để điều khiển servo motor.



Đây là khai báo các biến và cổng I/O cho các cảm biến, servo motor và các biến khác sử dụng trong chương trình.



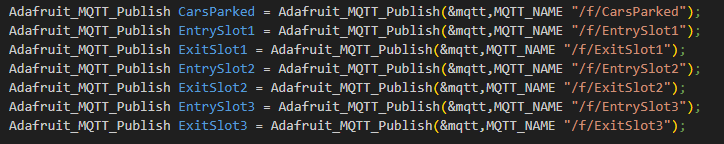
Đây là khai báo các biến dùng cho việc lưu trữ thông tin thời gian và trạng thái của các cảm biến và khe đỗ xe.



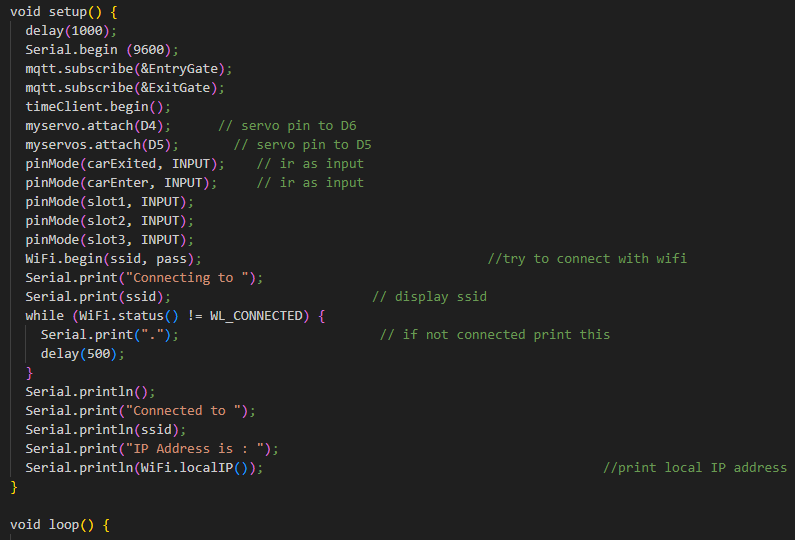
Đây là khởi tạo đối tượng MQTT Client, sử dụng đối tượng WiFiClient để kết nối với MQTT server của Adafruit IO.



Đây là khởi tạo các đối tượng Subscription MQTT để theo dõi các thay đổi trạng thái của EntryGate và ExitGate từ Adafruit IO.



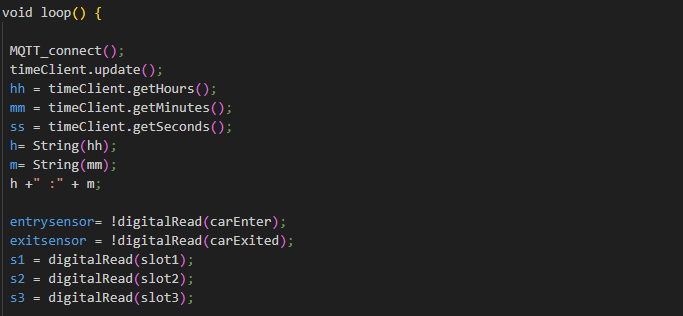
Đây là khởi tạo các đối tượng Publish MQTT để gửi dữ liệu đến Adafruit IO về số xe đỗ, thời gian vào/ra của từng khe đỗ.



Đây là hàm setup(), được gọi một lần duy nhất khi chương trình bắt đầu. Trong hàm

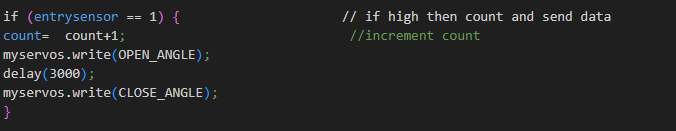
này:

* Đặt một khoảng thời gian chờ để đảm bảo module ESP8266 khởi động hoàn toàn.
* Khởi động giao tiếp Serial.
* Đăng ký các đối tượng Subscription MQTT để theo dõi EntryGate và ExitGate từ Adafruit IO.
* Bắt đầu đồng bộ thời gian với NTP server.
* Gắn servo motor vào các cổng D4 và D5 của ESP8266.
* Thiết lập các cổng I/O như cảm biến vào hoặc ra.
* Kết nối WiFi bằng cách sử dụng tên và mật khẩu được định nghĩa trước.
* In ra thông tin về quá trình kết nối.



Đây là hàm loop(), nó được thực thi liên tục sau khi hàm setup() kết thúc. Trong hàm loop():

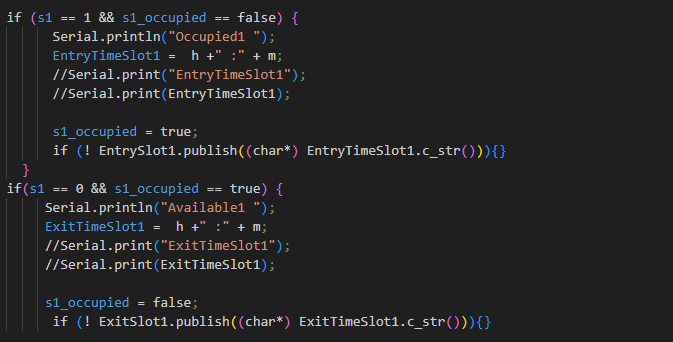
* Gọi hàm MQTT\_connect() để duy trì kết nối với MQTT server.
* Cập nhật thời gian từ NTP server.
* Đọc trạng thái của cảm biến (entrysensor, exitsensor, s1, s2, s3) và lưu trữ vào các biến tương ứng.
* Thực hiện xử lý điều khiển servo motor dựa trên trạng thái của các cảm biến và thông tin từ Adafruit IO thông qua giao thức MQTT.



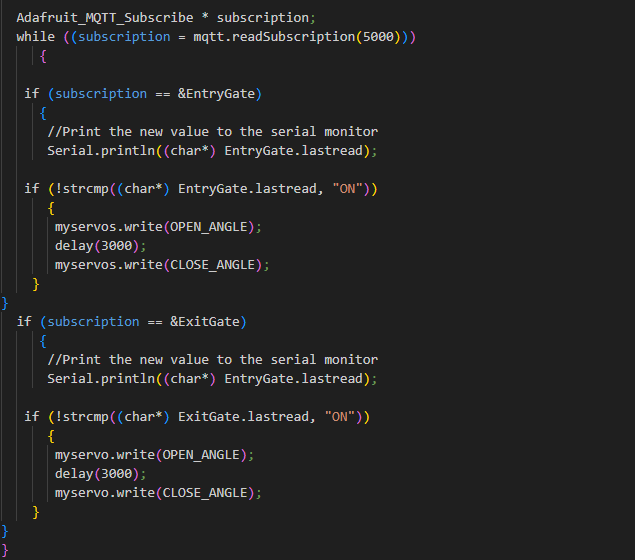
Nếu cảm biến entrysensor nhận được tín hiệu HIGH (1), tức là có một xe vào bãi đỗ, thì biến count sẽ được tăng lên 1 đơn vị (đếm số xe đỗ) và servo myservos sẽ di chuyển đến vị trí mở cổng (OPEN\_ANGLE) trong 3 giây, sau đó di chuyển đến vị trí đóng cổng (CLOSE\_ANGLE).



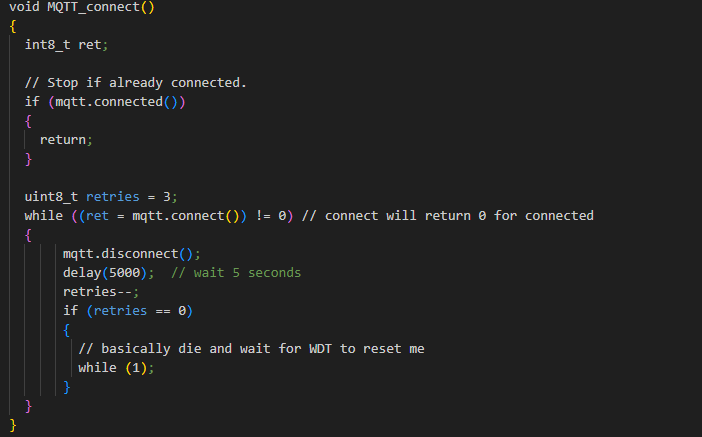
Tương tự, nếu cảm biến exitsensor nhận được tín hiệu HIGH (1), tức là có một xe rời khỏi bãi đỗ, thì biến count sẽ được giảm đi 1 đơn vị (đếm số xe đỗ) và servo myservo sẽ di chuyển đến vị trí mở cổng (OPEN\_ANGLE) trong 3 giây, sau đó di chuyển đến vị trí đóng cổng (CLOSE\_ANGLE).



Nếu khe đỗ s1 có tín hiệu HIGH (1) và trạng thái s1\_occupied là false (chưa có xe đỗ), thì sẽ được hiển thị thông báo "Occupied1" trên Serial Monitor và gửi thông tin thời gian vào khe đỗ s1 lên Adafruit IO thông qua EntrySlot1. Nếu khe đỗ s1 có tín hiệu LOW (0) và trạng thái s1\_occupied là true (đã có xe đỗ), thì sẽ được hiển thị thông báo "Available1" trên Serial Monitor và gửi thông tin thời gian ra khỏi khe đỗ s1 lên Adafruit IO thông qua ExitSlot1.  
Tương tự, các khối lệnh sau đó (s2 và s3) được sử dụng để xử lý thông tin từ các khe đỗ s2 và s3.

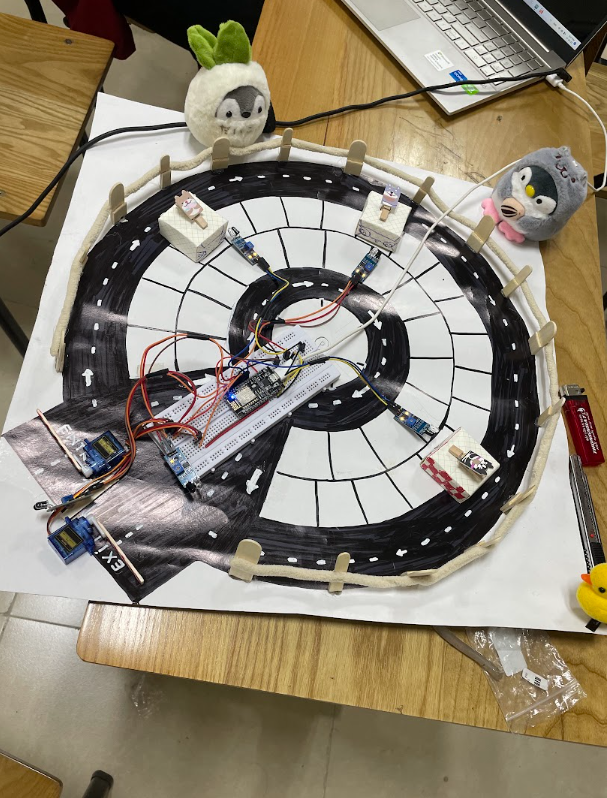


Trong vòng lặp này, chương trình đọc các thông báo từ Adafruit IO thông qua giao thức MQTT. Nếu nhận được thông báo từ EntryGate là "ON", servo myservos sẽ di chuyển đến vị trí mở cổng (OPEN\_ANGLE) trong 3 giây, sau đó di chuyển đến vị trí đóng cổng (CLOSE\_ANGLE). Tương tự, nếu nhận được thông báo từ ExitGate là "ON", servo myservo sẽ di chuyển đến vị trí mở cổng (OPEN\_ANGLE) trong 3 giây, sau đó di chuyển đến vị trí đóng cổng (CLOSE\_ANGLE).



Đây là hàm MQTT\_connect(), được sử dụng để duy trì kết nối với MQTT server của Adafruit IO. Nó sẽ thử kết nối và truy cập đến 3 lần nếu không thành công, và nếu không kết nối được sau 3 lần thì sẽ đặt chương trình vào một vòng lặp vô tận, dừng tất cả hoạt động.

# 5.2 Sản phẩm:



# Video chạy sản phẩm: https://drive.google.com/file/d/1pXqHPZxUNZP0XqTcD8pstuUeDyhzrhQv/view?usp=drive\_link

# VI. Kết luận:

# - Trên đây là bài báo cáo về đề tài “Smart Parking” của nhóm chúng em. Về mặt ý tưởng, chúng em đã xây dựng một bãi đỗ xe thông minh tiện lợi, giảm thiểu chi phí về các thiết bị cũng như con người. Sản phẩm tuy còn đơn giản và quá trình thực hiện còn nhiều thiếu sót nhưng về cơ bản bãi đỗ xe đã có được các tính năng nổi bật như: Kiểm tra bãi còn vị trí trống hay không, xác định được vị trí trống, cảm biến có hoạt động để mở hay đóng cổng, dữ liệu được gửi về server để theo dõi theo thời gian thực,... - Rất mong nhận được sự đánh giá và chỉ dẫn từ thầy để sản phẩm ngày một hoàn thiện hơn trong tương lai.