BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM

KHOA ĐIỆN - ĐIỆN TỬ

MÔN HỌC: HỆ ĐIỀU HÀNH THỜI GIAN THỰC

BÁO CÁO CUỐI KỲ

MÔ HÌNH BÃI GIỮ XE SỬ DỤNG CÔNG NGHỆ RFID

GVHD: Ts. Phan Văn Ca

SVTH:	**MSSV**
Nguyễn Lê Huy	**21119077**
Nguyễn Quốc Thắng	**21119131**
Hoàng Hữu Thiều	**21119132**

Lớp thứ 7, Tiết 8-11

Mã lớp: RTOS345264

Tp. Hồ Chí Minh, tháng 12 năm 2024

MỤC LỤC

1. PHẦN MỞ ĐẦU

Hiện nay, ngành kỹ thuật đang không ngừng phát triển mạnh mẽ và đóng vai trò quan trọng trong việc thúc đẩy sự tiến bộ của xã hội. Với sự đầu tư lớn về nguồn lực trí tuệ và tài chính, ngành kỹ thuật đang đạt được nhiều thành tựu nổi bật, bao gồm nghiên cứu, chế tạo và lắp ráp các thiết bị hiện đại. Tại Việt Nam, chuyên ngành kỹ thuật công nghiệp nhận được sự quan tâm đặc biệt và đầu tư đáng kể nhằm thúc đẩy sự phát triển bền vững.

Việc ứng dụng công nghệ tiên tiến không chỉ mang lại sự tiện lợi mà còn giúp cải thiện hiệu quả công việc một cách nhanh chóng và chính xác, thay thế cho các phương pháp thủ công truyền thống. Chính vì những lợi ích to lớn đó, nhóm chúng em đã quyết định lựa chọn đề tài: **"Hệ thống bãi giữ xe sử dụng công nghệ RFID"**.

1.1. Lí do chọn đề tài

Trong thời buổi khoa học kỹ thuật phát triển như hiện tại, yêu cầu của người dùng càng ngày được nâng cao. Chẳng hạn như khách hàng muốn thay thế những công việc tay chân bằng những công cụ hiện đại, đỡ tiêu hao nhân lực và thời gian. Nắm bắt được điều đó, nhóm chúng em quyết định chọn đề tài: "Hệ thống bãi giữ xe sử dụng công nghệ RFID". Để tối ưu hóa sự tiện lợi của hệ thống và giảm thiểu khối lượng công việc cho người dùng.

1.2. Yêu cầu đề tài

- **Yêu cầu cho hệ thống:**

Một màn hình thể hiện trạng thái vào ra cho khách, 2 module RFID quét thẻ tương đương 2 lối vào ra, 1 web quản lý thông tin khách hàng, 2 cửa tự động.

- **Quản lý xe ra/vào tự động:**

+ Nhận diện xe ra/vào thông qua thẻ RFID gắn trên xe hoặc do người dùng mang theo.

+ Ghi nhận thời gian xe vào và ra khỏi bãi gửi.

+ Tự động mở/đóng cổng kiểm soát khi xác thực thành công.

- **Quản lý thông tin khách hàng:**

+ Chia ra làm 2 tệp khách hàng: khách vãng lai và khách đăng ký thành viên.

+ Đối với khách vãng lai: chỉ kiểm tra thông tin thông qua UID của thẻ RFID và hình ảnh được chụp.

+ Đối với khách đăng ký thành viên: thêm/xóa thông tin thành viên trên hệ thống (tên, số phòng, thời gian đăng ký, thời gian hết hạn, trạng thái ra/vào hiện tại).

+ Cập nhật và chỉnh sửa thông tin khách hàng khi cần thiết.

- **Tính phí gửi xe:**

+ Tự động tính toán chi phí gửi xe dựa trên thời gian thực tế.

+ Gửi thông báo về chi phí cho khách hàng (qua màn hình hiển thị LCD và web quản lý).

- **Cảnh báo và bảo mật:**

+ Hiển thị hình ảnh xe lúc vào và lúc ra để so sánh.

+ Ghi lại nhật ký xe vào/ra để kiểm tra khi cần.

- **Tích hợp và mở rộng:**

+ Hỗ trợ tích hợp với các hệ thống khác như ứng dụng di động hoặc cổng thanh toán trực tuyến (ví dụ: Momo, ZaloPay).

+ Khả năng mở rộng quy mô bãi giữ xe (thêm đầu đọc RFID, mở rộng cơ sở dữ liệu).

1.3. Tính cấp thiết của đề tài

Đề tài **"Hệ thống bãi giữ xe sử dụng RFID "** mang tính cấp thiết cao trong đời sống xã hội hiện đại. Hệ thống này không chỉ đáp ứng nhu cầu quản lý phương tiện đi lại của các tổ chức, cơ quan mà còn giúp giải quyết nhiều vấn đề thực tiễn. Nó có khả năng thay thế các công việc tay chân, giảm thiểu sức lao động, đồng thời hạn chế tình trạng ùn tắc tại các cổng ra vào của công ty, trường học, trung tâm thương mại, và nhiều địa điểm khác.

Bằng việc tích hợp các khối ngoại vi, vi điều khiển cùng các giao thức giao tiếp hiện đại, nhóm chúng em đã phát triển thành công một hệ thống quét thẻ hỗ trợ con người quản lý dễ dàng hơn, đảm bảo tính bảo mật cao và nâng cao hiệu quả trong công tác quản lý nhân sự.

2. CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu linh kiện

2.1.1. Thẻ RFID là gì ?

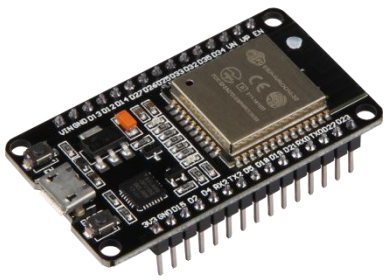
Hình 1. thẻ RFID

RFID (Radio Frequency Identification) là công nghệ sử dụng sóng vô tuyến để tự động nhận diện, theo dõi và quản lý các đối tượng thông qua thẻ nhận diện. Thẻ RFID được thiết kế với một dải băng từ tính ở mặt sau, cho phép thiết bị đọc truy cập dữ liệu lưu trữ bên trong.

- **Ăng-ten tích hợp:** Sử dụng sóng vô tuyến (RFID) để giao tiếp với thiết bị đọc và truyền tải dữ liệu.
- **Chip:** Lưu trữ một chuỗi số nhận diện duy nhất, đảm bảo không trùng lặp với bất kỳ thẻ nào khác.
- **Chất liệu:** Thẻ thường được làm từ nhựa hoặc các vật liệu lót khác để bảo vệ ăng-ten và chip từ.

2.1.2. ESP 32 là gì?

Trên thị trường hiện nay, ESP32 là dòng vi điều khiển giá rẻ, tiêu thụ năng lượng thấp, tích hợp hỗ trợ Wi-Fi và Bluetooth dual-mode. ESP32 được trang bị bộ vi xử lý Tensilica Xtensa LX6, có các phiên bản lõi đơn và lõi kép. Thiết bị tích hợp nhiều thành phần như công tắc ăng-ten, RF balun, bộ khuếch đại công suất, bộ khuếch đại thu nhiễu thấp và module quản lý năng lượng.



Hình 2. ESP32

Bộ xử lý:

- **CPU:** Bộ vi xử lý Xtensa 32-bit LX6 với tùy chọn lõi đơn hoặc lõi kép, hoạt động ở tần số lên đến 240 MHz (160 MHz với ESP32-S0WD và ESP32-U4WDH), đạt hiệu suất tối đa 600 MIPS (200 MIPS với ESP32-S0WD/ESP32-U4WDH).
- **Bộ đồng xử lý:** Hỗ trợ co-processor ULP (Ultra Low Power) cho chế độ tiêu thụ điện năng thấp.
- **Hệ thống xung nhịp:** Bao gồm CPU Clock, RTC Clock và Audio PLL Clock.
- **Bộ nhớ:** 448 KB ROM và 520 KB SRAM.
- **Kết nối không dây:** Wi-Fi 802.11 b/g/n và Bluetooth v4.2 (BR/EDR, BLE).
- **GPIO:** Có 34 pad GPIO vật lý.

Bảo mật:

- **Hỗ trợ chuẩn bảo mật IEEE 802.11:** Bao gồm WFA, WPA/WPA2 và WAPI.
- **Secure Boot:** Chức năng khởi động an toàn.
- **Mã hóa dữ liệu:** Hỗ trợ mã hóa flash.
- **OTP:** Tích hợp bộ nhớ OTP 1024-bit, trong đó 768-bit dành cho người dùng.
- **Tăng tốc phần cứng:** Hỗ trợ các thuật toán mã hóa AES, SHA-2, RSA và elliptic curve cryptography.

Quản lý năng lượng:

- **Bộ ổn áp nội:** Sử dụng bộ ổn áp nội với điện áp rơi thấp.
- **Miền nguồn riêng:** Cung cấp miền nguồn độc lập cho RTC.
- **Tiết kiệm năng lượng:** Dòng điện ở chế độ deep sleep chỉ 5 μA.
- **Khả năng kích hoạt:** Hỗ trợ kích hoạt trở lại từ ngắt GPIO, timer, đo ADC và ngắt cảm ứng điện dung.

2.1.3. Module LCD 20x4 I2C

LCD 20x4 là loại màn hình tinh thể lỏng nhỏ dùng để hiển thị chữ hoặc số trong bảng mã ASCII. Mỗi ô của Text LCD bao gồm các chấm tinh thể lỏng, các

chấm này kết hợp với nhau theo trình tự "ẩn" hoặc "hiện" sẽ tạo nên các kí tự cần hiển thị và mỗi ô chỉ hiển thị được một kí tự duy nhất.

LCD 20x4 nghĩa là loại LCD có 4 dòng và mỗi dòng chỉ hiển thị được 20 kí tự. Đây là loại màn hình được sử dụng rất phổ biến trong các loại mạch điện.

Thông số kĩ thuật của LCD 20x4:

- Điện áp: 5V
- Ngõ giao tiếp: 16 chân
- Màu sắc: xanh lá hoặc xanh dương
- Module hỗ trợ giao tiếp với vi điều khiển: LCD I2C 20x4

Hình 3. Module LCD 20X4.

2.1.4. Servo là gì ?

Servo là động cơ có thể điều chỉnh góc xoay (0-180 độ hoặc tùy thuộc vào loại servo). Nó nhận tín hiệu điều khiển từ **chân PWM** của ESP32 để thay đổi góc quay.

Hình 4. Servo.

Thông số cơ bản của servo:

- **Nguồn hoạt động**: 4.8-6V (nên cấp nguồn riêng nếu sử dụng nhiều servo).
- **Tín hiệu điều khiển**: PWM (Pulse Width Modulation).
- **Góc điều chỉnh**: Từ 0° đến 180°

2.2. Phần mềm lập trình điều khiển

2.2.1 Phần mềm lập trình điều khiển Arduino và kết nối với ESP32

Arduino IDE là một phần mềm mã nguồn mở, chủ yếu được sử dụng để vế và biên dịch các loại mã vào module Arduino.Trong đó IDE là viết tắt của Môi trường phát triển tích hợp, và đây cũng chính là một phần mềm được giới thiệu một cách chính thức bởi Arduino.cc, hầu hết phần tất cả các module arduino đều tương thích với phần mềm này.

Arduino IDE có phiên bản cho tất cả các hệ điều hành ở thời điểm hiện tại như MAC, Windows, Linux và chạy trên nền tảng Java đi kèm với các chức năng và

lệnh có sẵn đóng vai trò quan trọng để gỡ lỗi, chỉnh sửa và biên dịch mã trong các môi trường.

Có rất nhiều các loại module Arduino như Arduino Uno, Arduino Mega, Arduino Leonardo, Arduino Micro và nhiều module khác.

Mỗi module chứa một bộ vi điều khiển trên bo mạch được lập trình và chấp nhận thông tin dưới dạng mã.

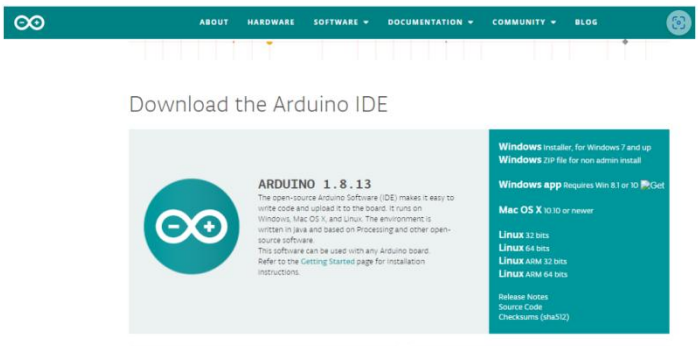
Mã chính, còn được gọi là sketch, được tạo trên nền tảng IDE sẽ tạo ra một file Hex, sau đó được chuyển và tải lên trong bộ điều khiển trên bo.

Môi trường IDE chủ yếu chứa hai phần cơ bản: Trình chỉnh sửa và Trình biên dịch, phần đầu sử dụng để viết mã được yêu cầu và phần sau được sử dụng để biên dịch và tải mã lên module Arduino.

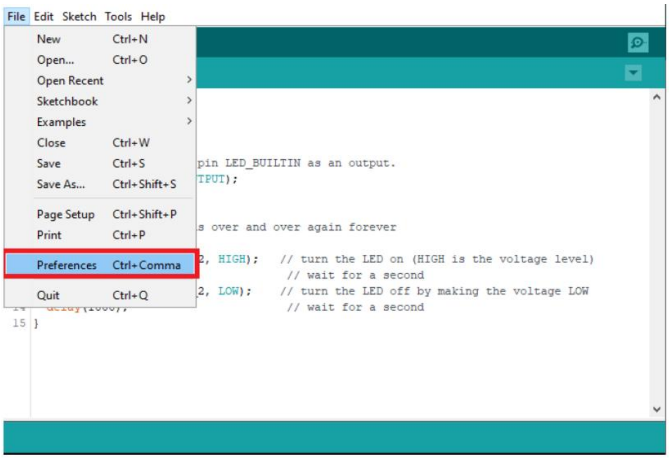
Môi trường này hỗ trợ cả ngôn ngữ C và C ++.

Để kết nối với ESP32 cần thao tác như sau :

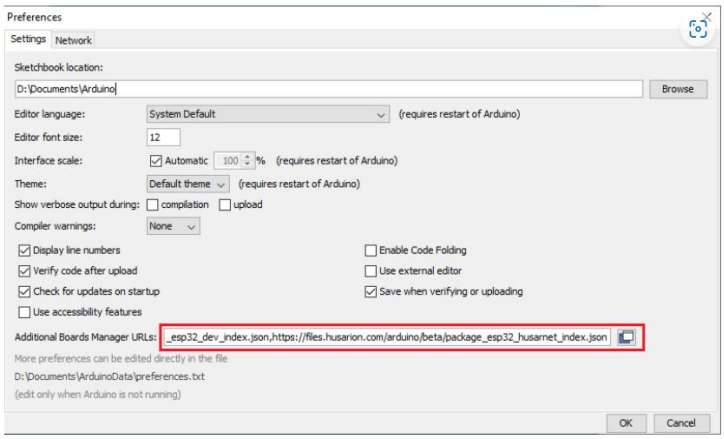
Bước 1 : Download the Arduino IDE



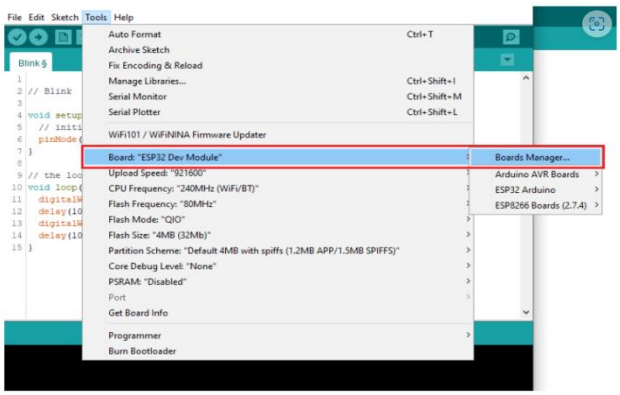
Bước 2 : *Chọn File > Preferences*



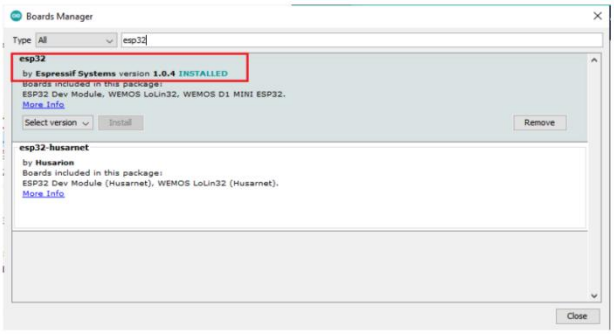
Bước 3 : Nhập ***https://dl.espressif.com/dl/package_esp32_index.json*** vào như trong hình > nhấn OK



Bước 4 : Mở Boards Manager > Tools > Board



Bước 5 : Tìm và cài đặt trình ESP32 như trong hình



2.2.2. Giao tiếp với ESP32

HTTP qua TCP Socket với yêu cầu POST từ thiết bị

Giao thức HTTP (HyperText Transfer Protocol) là một giao thức ứng dụng được sử dụng rộng rãi để trao đổi dữ liệu giữa client (ví dụ: ESP32) và server (web server). HTTP dựa trên giao thức TCP/IP để thiết lập kết nối và truyền dữ liệu.

Client: Thiết bị gửi yêu cầu HTTP (ESP32 trong trường hợp này).
Server: Dịch vụ hoặc máy chủ xử lý yêu cầu HTTP và trả về phản hồi (API server, web server, v.v.).

→ Đề tài chúng em sử dụng ESP32 gửi dữ liệu UID lên database (MongoDB) và từ đó gửi lên web server có thể giao tiếp với nhau thông qua giao thức HTTP.

Phân tích giao thức HTTP trong code:

1) Phương thức POST:

API nhận dữ liệu thông qua POST request. Điều này được thể hiện ở đoạn code

```
client.println("POST /card/parkinout HTTP/1.1");
```

Phương thức này thường được sử dụng để gửi dữ liệu đến server, ví dụ: dữ liệu thẻ RFID.

2) Tiêu đề HTTP (HTTP Headers):

Các thông tin như loại nội dung (Content-Type) và độ dài của dữ liệu (Content-Length) được khai báo:

```
client.println("Content-Type: application/x-www-form-urlencoded");
client.print("Content-Length: ");
client.println(postData.length());
```

3) Nội dung (Body) của HTTP Request:

Dữ liệu UID của thẻ RFID được gửi dưới dạng x-www-form-urlencoded (dữ liệu dạng form):

```
String postData = "card_id=" + uid;
```

```
client.println(postData);
```

4) Kết nối TCP:

HTTP hoạt động dựa trên giao thức TCP, được khởi tạo với server qua đối tượng WiFiClient:

```
if (client.connect(serverIP, serverPort)) {
```

5) Gửi dữ liệu qua giao thức HTTP:

ESP32 mở một kết nối TCP với server tại địa chỉ IP (192.168.200.42) và cổng (3000), rồi gửi dữ liệu HTTP.

Kết luận:

Giao thức được sử dụng trong code này là HTTP thông qua TCP socket. Server mà ESP32 kết nối sẽ xử lý yêu cầu HTTP POST để nhận dữ liệu RFID UID từ thiết bị.

2.3. Giới thiệu sơ lược về FreeRTOS

2.3.1. Giới thiệu về FreeRTOS

FreeRTOS được phát triển bởi Real Time Engineers Ltd, sáng lập và sở hữu bởi Richard Barry, là một hạt nhân hệ điều hành thời gian thực cho hệ thống nhúng, nó hỗ trợ lên tới được 35 nền tảng vi điều khiển.

FreeRTOS là một hệ điều hành hệ thống nhúng rất phù hợp cho nghiên cứu, học tập về các kỹ thuật công nghệ đơn giản, cũng như việc phát triển mở rộng tiếp các thành phần cho hệ điều hành hành (bổ sung modules, driver, thực hiện porting).

2.3.2. Đặc điểm của hệ điều hành FreeRTOS

Được thiết kế một cách nhỏ gọn (4.3 Kbytes sau khi biên dịch trên ARM7) và dễ sử dụng, phần lớn và chủ yếu được viết bằng ngôn ngữ lặp trình C dể dễ dàng port và bảo trì mỗi khi có lỗi, nó cũng có tích hợp thêm một số hợp ngữ .

FreeRTOS cung cấp cho người sử dụng các phương thức đa luồng, mutexes, semaphores và đồng hồ, và chế độ không đánh dấu được cung cấp cho các ứng dụng sử dụng nguồn năng lượng thấp.

2.3.3. Cách sử dụng hệ điều hành FreeRTOS

FreeRTOS.h: File này nhằm định hướng cho hệ điều hành xem và sử dụng các chức năng như thế nào. Kiểm tra xem FreeRTOSconfig.h đã định nghĩa các ứng dụng macro phụ thuộc vào từng chương trình một cách rõ ràng hay chưa. Nếu hàm hoặc macro nào muốn sử dụng cần được đặt lên 1, ngược lại đặt ở 0.

Task.h: Gồm năm phần là các macro và các định nghĩa; Các task tạo API; Các task điều khiển API; Các task tiện ích; Bộ lập lịch

List.h: Trong file list.h, FreeRTOS định nghĩa các cấu trúc, các macro và các hàm phục vụ cho các tiện ích về danh sách. Chức năng của file là tạo mới, thêm, bớt các tác vụ vào danh sách các task đang chạy (running), sẵn sàng (ready), khoá (block), treo (suppend).

croutine.h: Tạo ra các hàm và các macro liên quan đến task và queue

Khởi tạo các task, khởi tạo cho các thiết bị ngoại vi như màn hình, thiết lập ngắt…

Tạo các task ứng dụng: điều khiển led và task đọc giá trị ADC hiển thị lên màn hình.

2.3.4. Các phương tiện có thể sử dụng như

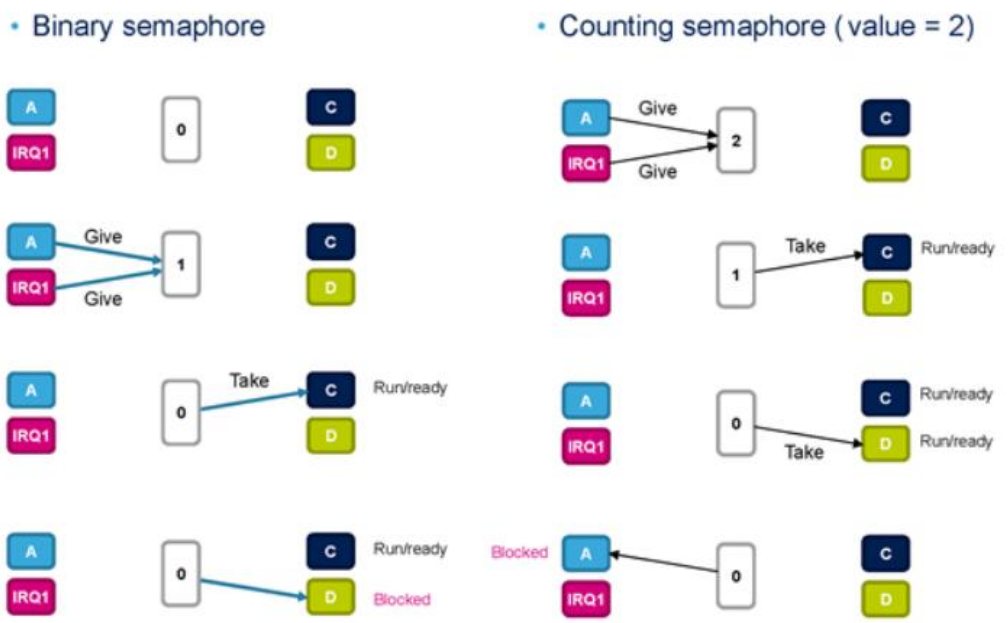
Trong hệ điều hành FreeRTOS, mỗi task như là một chương trình hoạt động theo một cách riêng biệt, và để liên lạc giữa các task này, chúng ta có thể sử dụng một số phương tiện cụ thể như queue,semaphores,...

Queue:

- Có ***khả năng chứa một số hữu hạn*** của các phần tử được khai báo, ngoài ra nếu nó còn đủ chỗ trống trong vùng heap thì nó vẫn sẽ được khởi tạo và chạy một cách bình thường.
- Nó có ***tính linh hoạt rất cao*** bởi nó không được sở hữu bởi một task cụ thể nào cả, nó còn có thể được truy cập từ nhiều task khác nhau. Thông thường khi một queue được ghi nó có thể được ghi từ nhiều task khác nhau, và được đọc ở một task nào đó trong vùng heap .
- ***Hoạt động theo cơ chế FIFO (first in first out)*** , việc đọc sẽ được thực hiện ở cuối queue, và việc ghi có thể ở đầu hoặc đuôi. Một item khi được ghi vào queue thì giá trị của nó được sao chép lại, nghĩa là ta có thể free sau đó, hoặc sử dụng biến cục bộ cho việc ghi.
- ***Block khi đọc queue*** . Khi có một task yêu cầu đọc queue nếu đang trống thì nó sẽ tự đi vào chế độ block và chờ, và nó sẽ thoát khỏi chế độ block khi có một task khác hoặc một task khác có độ ưu tiên cao hơn vào chương trình, và nó cũng sẽ đi đến ready nếu thời gian chờ kết thúc.
- ***Block khi ghi queue***. Khi một task ra lệnh ghi vào queue, nó sẽ tiến vào chế độ Block. Nếu queue đang đầy task sẽ thoát ra khỏi Block khi queue có chổ trống trở lại, hoặc kết thúc thời gian chờ. Trong trường hợp task đang chờ để ghi vào task thì task nào sở hữu độ ưu tiên cao hơn thì task đó sẽ được ghi trước và task kia sẽ chờ lâu hơn vì có độ ưu tiên thấp hơn.

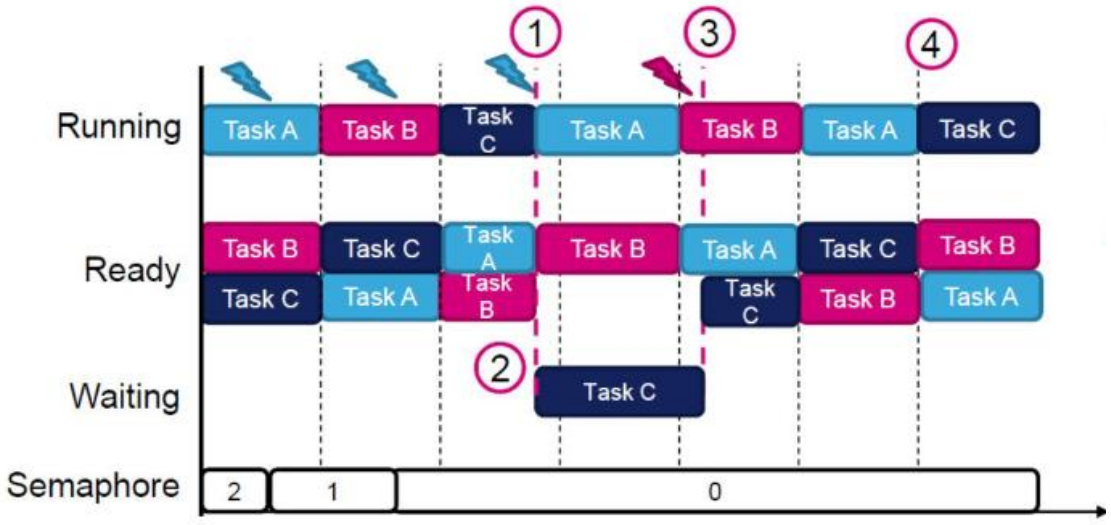
Semaphores :

- Semaphores được dùng để đồng bộ hóa giữa các task, cụ thể là kết nối và ngắt giữa các task, semaphores có 2 dạng chính thường gặp: Binary semaphore, Counting semaphores.
- Đối với Binary semaphore có duy nhất 1 token và sử dụng để đồng bộ một hoạt động.
- Counting semaphore sẽ có nhiều token và đồng bộ nhiều action khác nhau.



Hình 5 . Mô hình hoạt động của 2 dạng semaphore

Ví dụ minh họa :



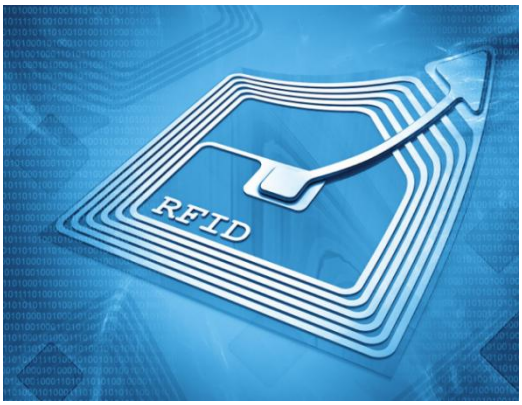
Hình 6. Ví dụ về semaphore

Ở ví dụ trên có 3 task A,B,C và chúng có cùng độ ưu tiên, có 2 semaphore cùng 2 token.

- Task C sẽ gọi hàm osSemaphoreWait() nhưng semaphore token đã được take bởi task A và task B.
- Trạng thái của task C sẽ được chuyển sang hàng chờ và Task A đã được bắt đầu. Task B thì đang trong trạng thái ready.
- Task B gọi hàm osSemaphoreRelease(). Trạng thái của task C sẽ được chuyển sang ready. Vì tất cả các task có cùng độ ưu tiên nên task B sẽ tiếp tục thực thi.
- Cuối cùng là Task C sẽ được thực hiện.

2.4. Công nghệ RFID

2.4.1. Đánh giá tổng quan



Hình 7. Công nghệ RFID

Ở trên thị trường hiện nay đã và đang có rất nhiều sản phẩm, hệ thống ứng dụng công nghệ RFID đặc biệt là trong lĩnh vực quản lý. Cụ thể là quản lý nhân sự , quản lý sinh viên học sinh, quản lý nhà xe… Chúng được sử dụng trong nhiều hệ thống quản lý như vậy vì những ưu điểm sau : [1].

- Tính bảo mật cao
- Tiện nghi và hiện đại
- Cài đặt dễ dàng
- Dễ sử dụng
- Đem lại tính thẩm mỹ

2.4.2. Đặc điểm của hệ thống RFID

Sử dụng công nghệ không dây thu phát sóng radio, không sử dụng tia sáng như mã vạch (Bar Code hay QR Code). Do đó, thông tin của dữ liệu có thể được truyền tải mà không cần tiếp xúc vật lý nào cả.

RFID có thể đọc những thông tin xuyên qua các môi trường, vật liệu như: bê tông, tuyết, sương mù, băng đá, sơn và các điều kiện môi trường thách thức khác mà mã vạch và các công nghệ khác không thể phát huy hiệu quả.

Dải tần số thường được hệ thống sử dụng là 125Khz hoặc 900Mhz.

2.4.3. Thành phần cấu tạo của hệ thống

a. Phần cứng của hệ thống

- **Ăng- ten :** gắn trong thẻ sử dụng song vô tuyến (RFID) để giao tiếp với đầu đọc thẻ và truyền dữ liệu.
- **Chip :** tạo ra một dãy số riêng biệt cho mỗi thẻ, và dãy số đặc biệt không trùng với các thẻ khác.
- **Đầu đọc thẻ :** Đây là thiết bị được cấu tạo từ một hoặc nhiều ăng-ten phát ra sóng vô tuyến và nhận lại tín hiệu từ thẻ
- **Máy chủ :** là nơi mà máy chủ và hệ thống phần mềm được thiết lập và phân tích xử lý những dữ liệu được thu thập được từ các thẻ RFID

b. Phần mềm hệ thống

Các phần mềm có thể hỗ trợ (ERP, MES, PLM, SCM) : việc tích hợp RFID với các phần mềm quản lý hỗ trợ sẽ giúp nâng cao khả năng phân tích dữ liệu cũng như xử lý, từ đó đưa ra các quyết định trong vấn đề chiến lược kinh doanh.

c. Nguyên Lý hoạt động của hệ thống

Nghe thì thấy có vẻ đây là một hệ thống phức tạp, nhưng thật ra đây lại là một hệ thống rất đơn giản : đầu đọc thẻ RFID được đặt cố định ở một vị trí để đọc dữ liệu và nó phát ra một làn sóng vô tuyến ở một tần số nhất định để có thể phát hiện ra các thiệt bị phát xung quoanh vị trí đó.

Khi thẻ RFID đi vào vùng nhận tín hiệu do đầu đọc thẻ RFID phát ra, nó sẽ thu nhận và phát lại tín hiệu bằng một dãi mã số riêng để có thể phân biệt, từ đó đầu đọc có thể biết RFID nào đang trong vùng nhận tín hiệu.

2.5. Các chuẩn giao tiếp sử dụng trong đề tài

2.5.1. Chuẩn giao tiếp SPI

Giao thức giao tiếp I2C (Inter-Integrated Circuit) là một phương thức liên kết được sử dụng để kết nối và truyền dữ liệu giữa các linh kiện điện tử trong mạch điều khiển. I2C bao gồm hai dây truyền thông chính: một dây dành cho dữ liệu (SDA - Serial Data) và một dây dành cho xác định đồng bộ dữ liệu (SCL - Serial Clock). Điều này cho phép nhiều thiết bị kết nối với nhau thông qua cùng một bus, giúp tiết kiệm chân và tối ưu hóa việc truyền thông giữa chúng.

2.5.2. Chuẩn giao tiếp I2C

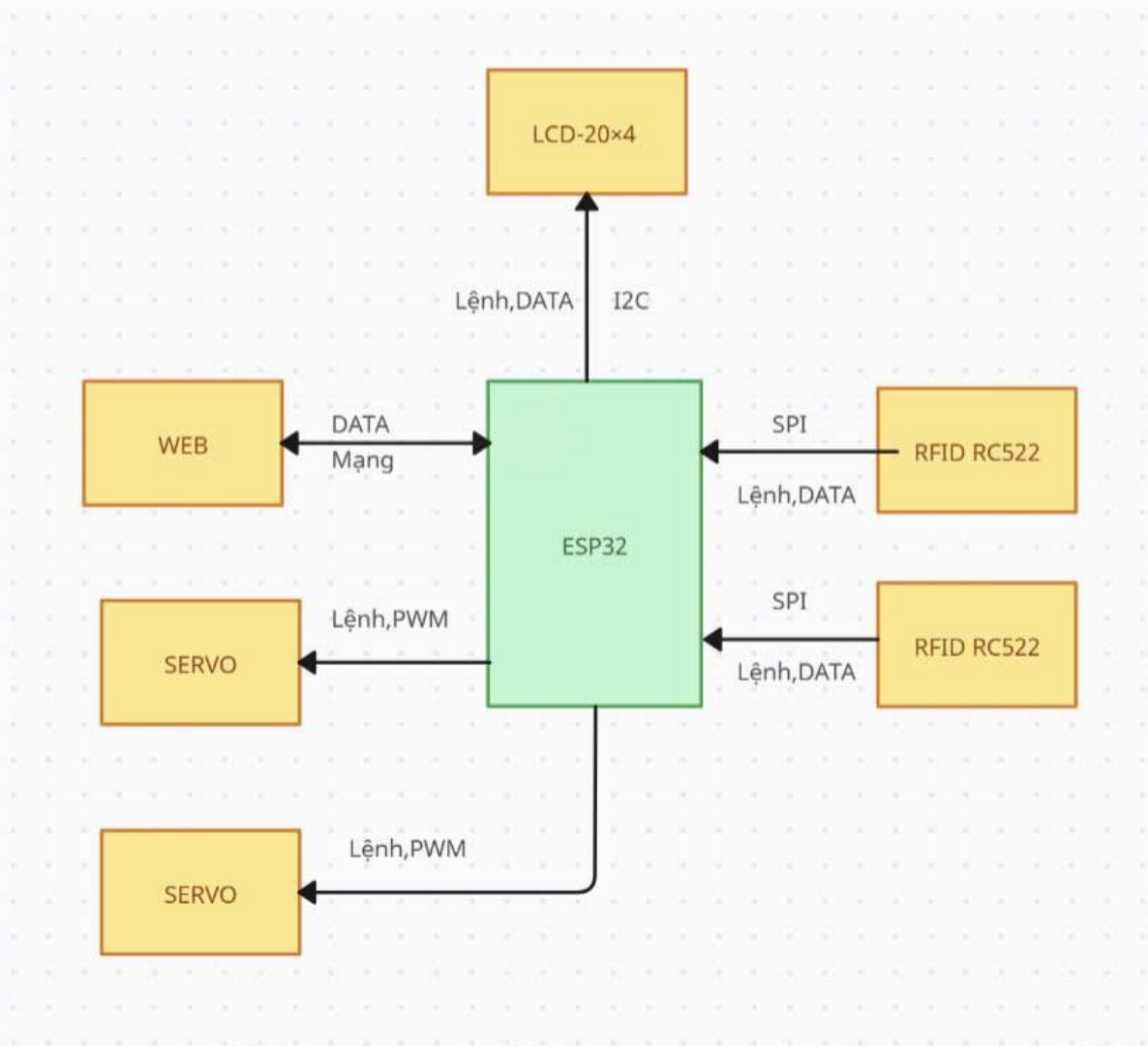
Giao thức SPI (Serial Peripheral Interface) là một giao thức truyền thông đồng bộ được sử dụng trong việc kết nối giữa các linh kiện điện tử trong mạch điều khiển. Nó bao gồm ba hoặc bốn dây kết nối: dây clock (SCLK) xác định tốc độ truyền dữ liệu, dây chip select (CS) để chọn thiết bị cần truyền thông, và hai dây dành cho dữ liệu truyền và nhận (MISO - Master In Slave Out, MOSI - Master Out Slave In). SPI cho phép truyền dữ liệu với tốc độ cao, cho phép một thiết bị điều khiển (Master) giao tiếp với nhiều thiết bị khác (Slaves) trên cùng một bus. Nó thường được sử dụng trong các ứng dụng yêu cầu tốc

2.5.3. Chuẩn giao tiếp UART

UART (Universal Asynchronous Receiver-Transmitter) là một giao thức truyền thông giữa các thiết bị được sử dụng rộng rãi trong các ứng dụng nhúng và máy tính. Giao thức này được sử dụng để truyền dữ liệu giữa các thiết bị điện tử như vi điều khiển, cảm biến, module truyền thông và các thiết bị khác. UART là một giao thức truyền thông phần cứng dùng giao tiếp nối tiếp không đồng bộ và có thể cấu hình được tốc độ.

UART truyền dữ liệu không đồng bộ, có nghĩa là không có tín hiệu đồng hồ để đồng bộ hóa đầu ra của các bit từ UART truyền đến việc lấy mẫu các bit bởi UART nhận. Thay vì tín hiệu đồng hồ, UART truyền thêm các bit start và stop vào gói dữ liệu được chuyển. Khi được cấu hình đúng cách, UART có thể hoạt động với nhiều loại giao thức nối tiếp khác nhau liên quan đến việc truyền và nhận dữ liệu nối tiếp.

2.6. Sơ đồ khối của hệ thống



Hình 8. Sơ đồ khối của hệ thống điểm danh, gửi dữ liệu lên Web

- Hệ thống bao gồm:

+ Vi điều khiển ESP32 (38 pins)
+ 2xModule đọc thẻ RFID RC522 giao tiếp SPI

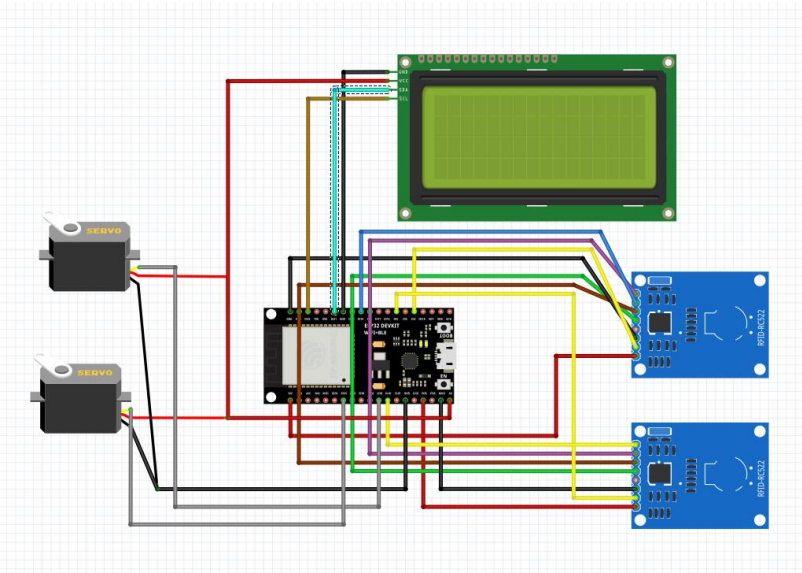
+ Web
+ 2x servo
+ 1x LCD 20x4 I2C

- Cách thức hoạt động:

- Các module giao tiếp bằng giao thức I2C được kết nối thông qua module I2C, có thể kết nối lên tới 8 modules.
- Mỗi khi ESP32 hoạt động (hay được cấp nguồn), VĐK sẽ thực hiện kết nối với Wifi được cấu hình sẵn trong Source code, Wifi được sử dụng để truyền nhận thông tin giữa ESP32 với WEB.
- Module đọc thẻ RC522 thực hiện đọc dữ liệu của thẻ được đưa vào
- Màn hình LCD được sử dụng để hiển thị giao diện để hiển thị các thông tin cần thiết như thông tin thẻ được quét, chế độ làm việc,...
- Servo được điều khiển bằng tín hiệu PWM từ ESP32 để thay đổi góc của servo, ví dụ như mở cổng, hoặc thay đổi các trạng thái khác dựa trên dữ liệu từ thẻ RFID.

3. THIẾT KẾ CHI TIẾT

3.1. Thiết kế chi tiết phần cứng

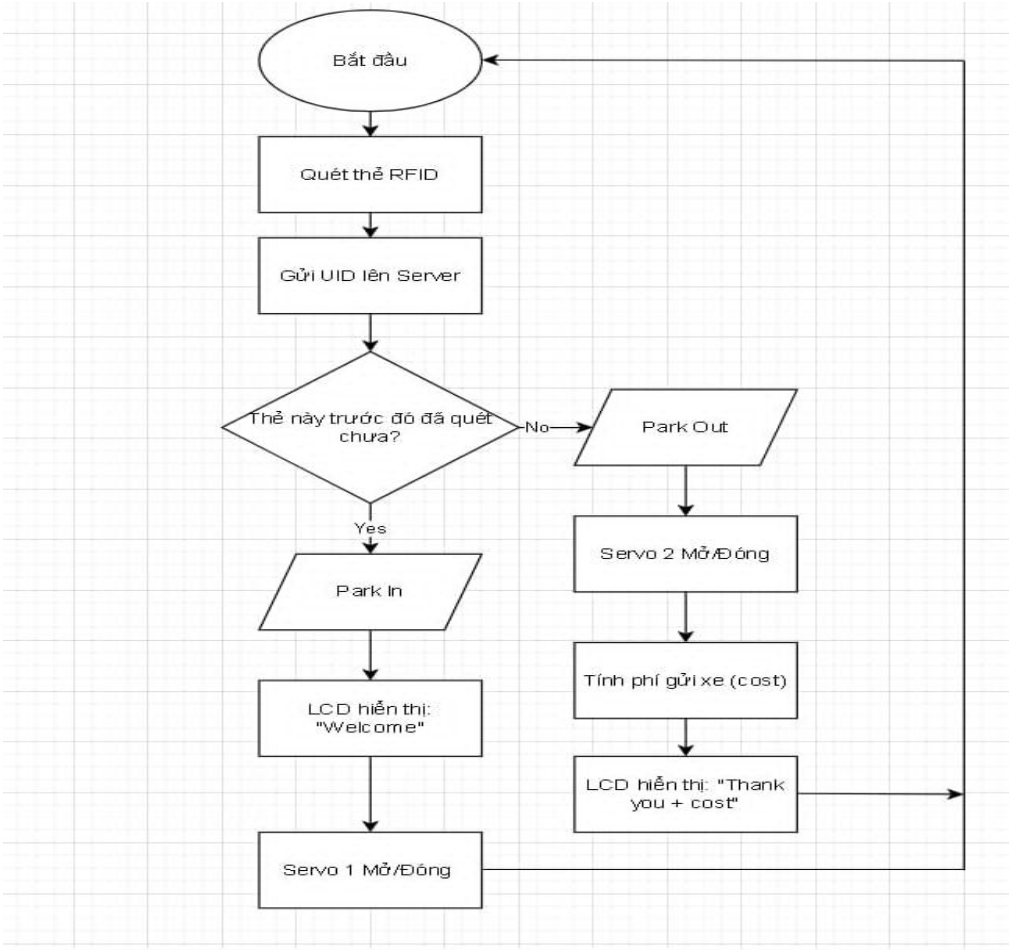


Hình 9 . Sơ đồ khối của hệ thống

ESP32	RC522(1)	RC522(2)	LCD	SERVO (1)	SERVO (2)
22	X	X	SCL	X	X
21	X	X	SDA	X	X
19	MISO	MISO	X	X	X
23	MOSI	MOSI	X	X	X
GND	GND	GND	GND	GND	GND
3V3	3V3	3V3	X	X	X
18	SCK	SCK	X	X	X
32	X	X	X	X	IN
27	X	X	X	IN	X
5V	X	X	5V	5V	5V
4	RST	X	X	X	X
14	X	RST	X	X	X

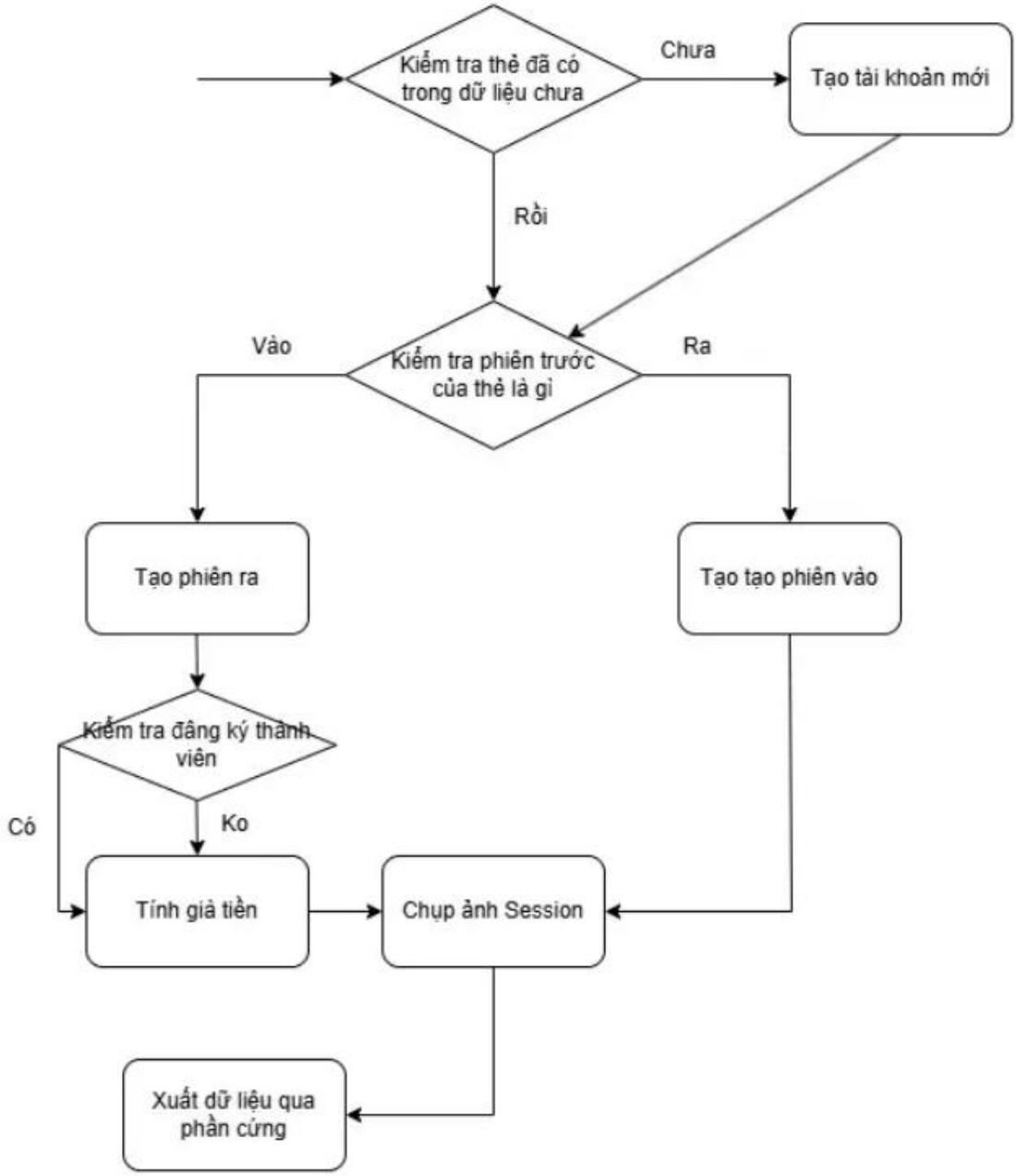
3.2. Thiết kế phần mềm

3.2.1. Lập trình thiết kế



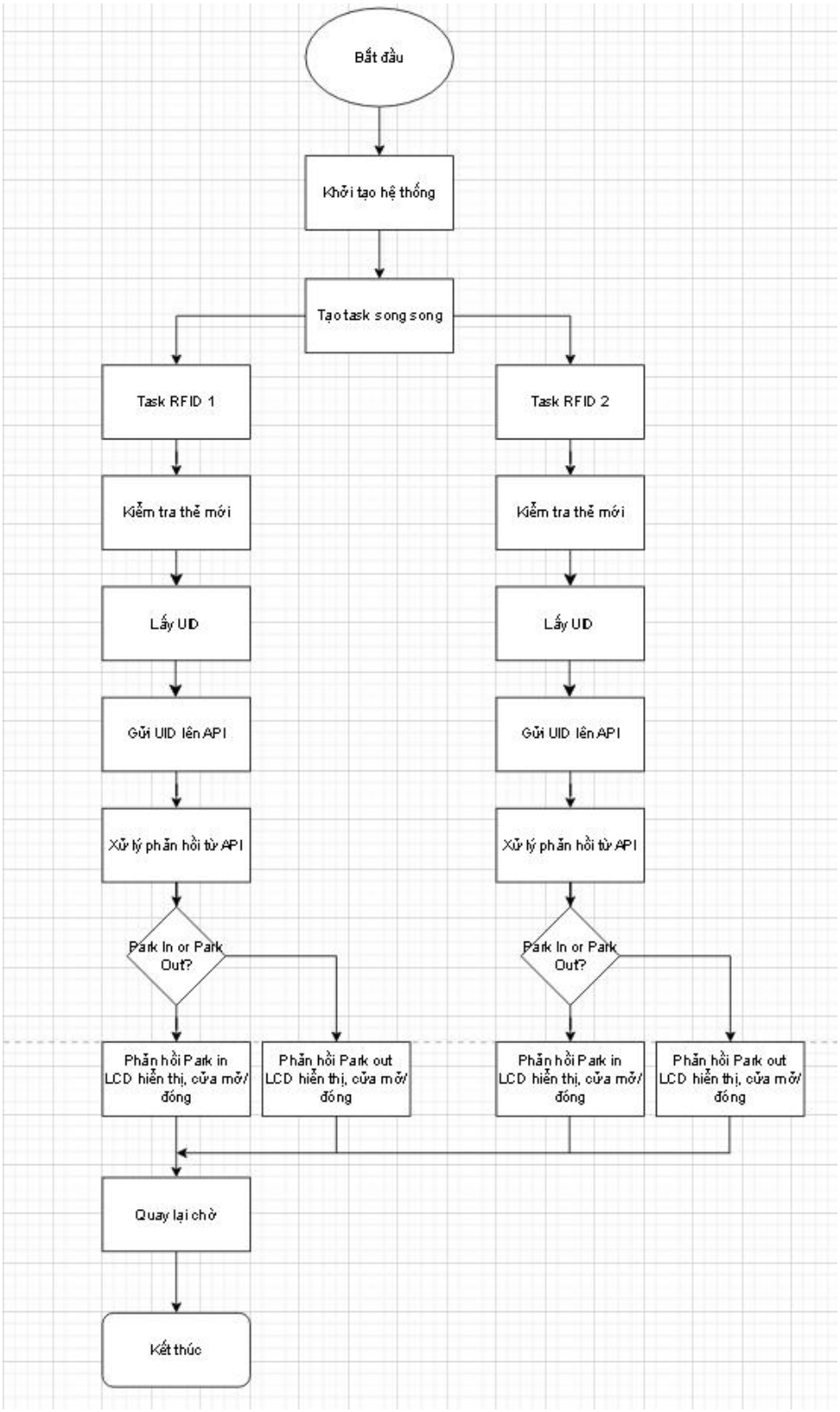
Hình 10. Lưu đồ giải thuật hoạt động của phần cứng gửi dữ liệu lên web

3.2.2. Lập trình trên Arduino IDE + Lưu đồ giải thuật khi không có FreeRTOS



Hình 11. Lưu đồ giải thuật hoạt động khi đưa dữ liệu lên WEB

3.2.3. Lưu đồ giải thuật khi có FreeRTOS



Hình 12. Lưu đồ giải thuật hoạt động của hệ thống điểm danh với FreeRTOS

Tài liệu tham khảo

1. Nguyễn Minh Cường, *Đề tài nghiên cứu công nghệ RFID*, Tài liệu E-BOOK, Đề tài Nghiên cứu công nghệ RFID - Tài liệu, ebook, giáo trình (doc.edu.vn) (Tham khảo về RFID) [truy cập ngày 29/12/2023]
2. *RTOS cơ bản phần 1*, Học ARM, RTOS cơ bản phần 1 | Học ARM (hocarm.org) (Tham khảo về RTOS) [truy cập ngày 29/12/2023]
3. (8) Ha Huynh - YouTube (Tham khảo về định nghĩa cũng như hiểu rõ và sâu hơn về lý thuyết