TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG ĐIỆN – ĐIỆN TỬ

BÁO CÁO BÀI TẬP LỚN
HỆ THỐNG NHÚNG
VÀ THIẾT KẾ GIAO TIẾP NHÚNG

Đề tài: Hệ thống điều khiển tàu điện con thoi chạy nội bộ trong sân bay

Giảng viên hướng dẫn: TS. Phạm Văn Tiến

Nhóm sinh viên thực hiện:

Tên sinh viên	MSSV	Mã lớp
Đinh Văn Quang	20193066	145609
Phạm Đức Cương	20192732	145609
Hoàng Thế Đang	20192742	145609
Trần Hữu Đông	20192761	145609

Hà Nội, 12-2023

LỜI NÓI ĐẦU

Thời gian gần đây, các Hệ thống nhúng – Thời gian thực được quan tâm nhiều hơn ở Việt Nam, và trên thế giới thì các hệ thống này đã và đang được phát triển mạnh mẽ và là xu hướng thịnh hành ở các nước Công nghiệp vì những lợi ích to lớn, thiết thực mà nó mang lại. Chương trình học môn Hệ thống nhúng và thiết kế giao tiếp nhúng là một phần quan trọng giúp hiểu rõ quy trình thiết kế, đánh giá hệ thống nhúng. Trong báo cáo này, nhóm chúng em triển khai thiết kế và mô phỏng hệ thống điều khiển tàu điện con thoi chạy nội bộ trong sân bay. Từ nhu cầu thực tiễn và lợi ích của xã hội ta thấy hệ thống điều khiển tàu điện con thoi là rất cần thiết trong các sân bay. Chính vì vậy, nhóm chúng em quyết định chọn đề tài mô phỏng hệ thống điều khiển tàu điện con thoi với hệ thống đón trả khách tự động và điều khiển tăng giảm tốc khi đến trạm . Cho ra kết quả hoạt động đúng với yêu cầu bài toán. Chúng em sẽ đi trình bày cụ thể những gì chúng em đã làm được thông qua 6 chương.

Nhóm chúng em xin chân thành cảm ơn thầy Phạm Văn Tiến đã tận tâm hướng dẫn nhóm chúng em trong quá trình thực hiện bài tập lớn cũng như hoàn hiện báo cáo này.

Nội dung

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

CHƯƠNG 1. XÁC ĐỊNH CHỈ TIÊU KỸ THUẬT

1.1 Yêu cầu kỹ thuật:

- Đảm bảo tàu đạt được vận tốc tối đa và duy trì tốc độ ổn định trong một khoảng thời gian cụ thể.
- Tàu phải giảm tốc độ khi tiếp cận bến để đảm bảo dừng đúng vị trí.
- Hệ thống cần có giao diện người-máy để hành khách có thể tương tác và nhận thông báo.
- Tích hợp với hệ thống điều khiển trung tâm để quản lý lịch trình và theo dõi hoạt động của tàu.
- Tàu phải duy trì độ ổn định khi di chuyển và khi thực hiện các chuyển động như mở/đóng cửa.

Các chỉ tiêu định lượng:

- Chỉ Tiêu 1: Chính Xác Vị Trí Dừng

Định lượng: Sai số vị trí khi dừng tại bến, đo bằng đơn vị độ hoặc mét.

Mục Tiêu: Sai số không quá ± 0.5 mét để đảm bảo hành khách lên, xuống tàu dễ dàng.

- Chỉ Tiêu 2: Thời Gian Đỗ Đúng

Định lượng: Thời gian tàu dừng tại bến, đo bằng giây.

Mục Tiêu: Thời gian dừng tại bến không quá 20 giây để đảm bảo lịch trình.

- Chỉ Tiêu 3: Tốc Độ Đóng Mở Cửa

Định lượng: Thời gian cửa mở và đóng, đo bằng giây.

Mục Tiêu: Thời gian mở/đóng cửa không quá 5 giây để đảm bảo hiệu quả.

- Chỉ Tiêu 4: Độ Chính Xác Của Hệ Thống Cảm Biến Hồng Ngoại

Định lượng: Sai số của cảm biến hồng ngoại, đo bằng đơn vị mét.

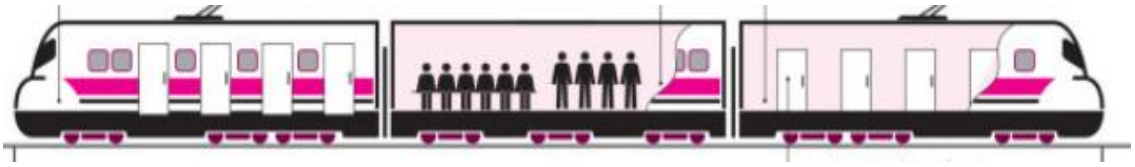
Mục Tiêu: Sai số không quá ± 0.2 mét để đảm bảo nhận diện chính xác.

1.2 Yêu cầu công nghệ:

- Dễ sử dụng
- An toàn cho người dùng và thiết bị
- Tiết kiệm điện năng
- Dễ dàng bảo trì, nâng cấp

CHƯƠNG 2. MÔ HÌNH HOÁ HỆ THỐNG

2.1. Mô tả cấu trúc và hoạt động của hệ thống



Hình 2. 1: Hình ảnh tàu thực tế

Cấu trúc và hoạt động của hệ thống

- Phần mềm điều khiển: Dùng để giao tiếp, tương tác với hệ thống điều khiển động cơ; theo dõi quá trình hoạt động của hệ thống
- Hệ thống đóng, mở cửa: Kết hợp với cảm biến hồng ngoại và chỉ hoạt động khi tàu ở trạng thái dừng, đỗ tại bến. Khi đó nếu cảm biến nhận diện được người, cửa sẽ mở ra, ngược lại nếu trong 1 thời gian nếu không nhận diện được thì cửa sẽ đóng.
- Cảm biến hồng ngoại: Dùng để nhận diện đối tượng. Được sử dụng kết hợp với hệ thống đóng, mở cửa.
- Hệ thống động cơ : Bao gồm điều khiển động cơ và động cơ. Hệ thống điều khiển để điều khiển chiều quay, tốc độ của động cơ tương ứng với các trạng thái của tàu: tăng tốc, giảm tốc, duy trì tốc độ, dừng, di chuyển tới bến 1, di chuyển về bến 2.
- Nguồn điện: Cung cấp năng lượng cho các thành phần khác hoạt động: Hệ thống đóng, mở cửa; Hệ thống động cơ; Màn hình hiển thị; Cảm biến hồng ngoại.
- Màn hình hiển thị: Hiển thị trạng thái hiện tại của tàu: dừng, tăng tốc, giảm tốc, duy trì tốc độ, đóng cửa, mở cửa.

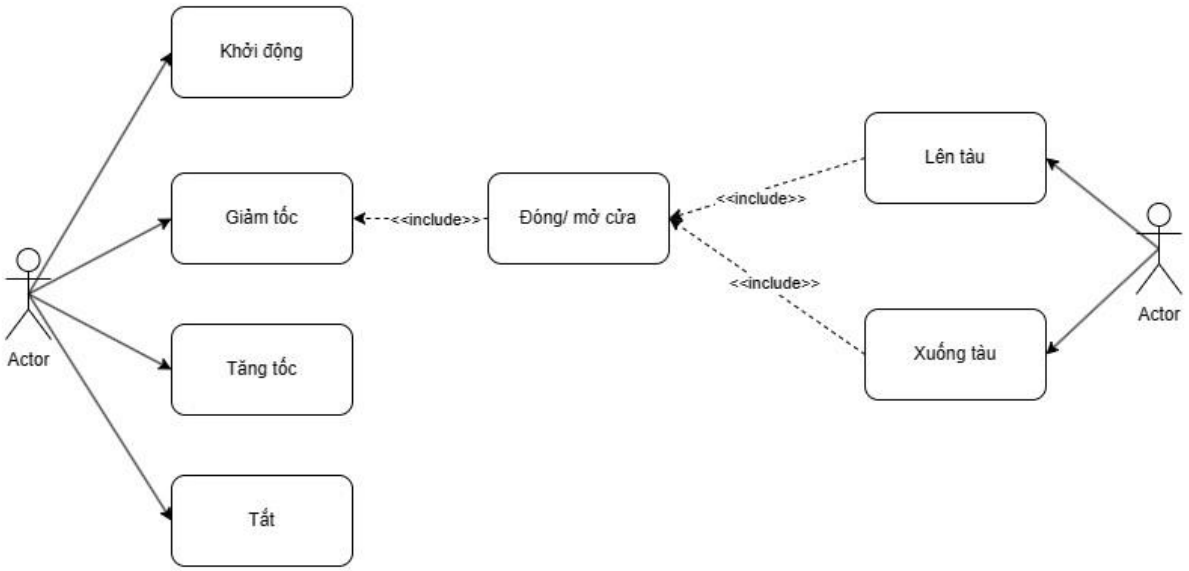
2.2. Nguyên lý làm việc:

- Người điều khiển: Trước hành trình, người điều khiển phải khởi động hệ thống thông qua phần mềm quản lý. Việc này bao gồm kích hoạt chức năng tự động hóa và kiểm soát các thao tác khởi đầu của tàu. Người điều khiển có nhiệm vụ theo dõi tiến trình hoạt động đảm bảo rằng hệ thống hoạt động một cách an toàn và hiệu quả.
- Hệ thống tàu: Tại bến 1, hệ thống tàu tự động mở cửa để đón hành khách. Cửa tàu sau đó tự động đóng lại sau khoảng thời gian nhất định, nhưng nếu có người tiến tới, cửa sẽ mở lại để chờ đến khi không còn người. Khi tàu bắt đầu hành trình, nó tăng tốc đến vận tốc tối đa và duy trì ổn định. Khi tiếp cận bến 2, tàu giảm tốc độ và dừng đúng vị trí, mở cửa để hành khách lên, xuống và sau đó đóng lại di chuyển ngược về bến 1. Quá trình này lặp đi lặp lại và hệ thống sử dụng cảm biến và bộ điều khiển tự động để tương tác với môi trường và hành khách.

2.3. Mô hình hóa hệ thống

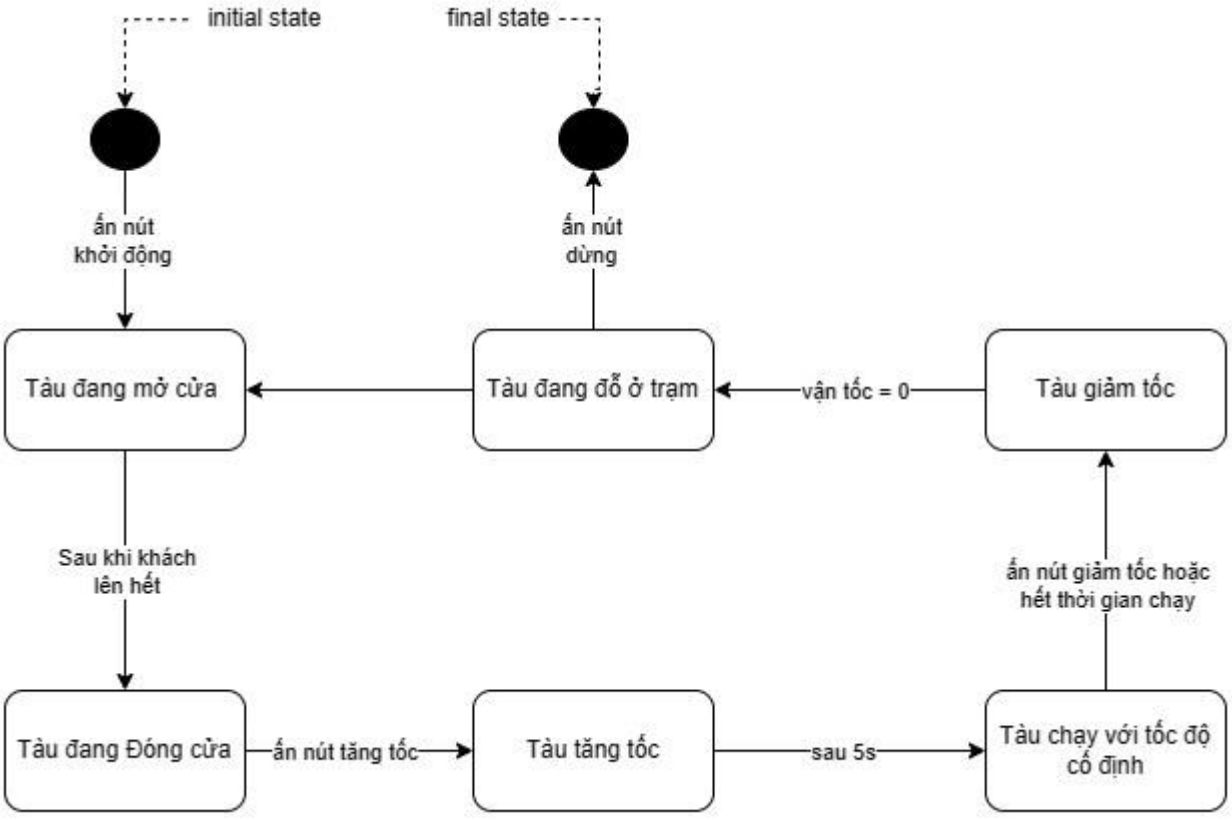
2.3.1. Mô hình hoá bằng UML

- UseCase



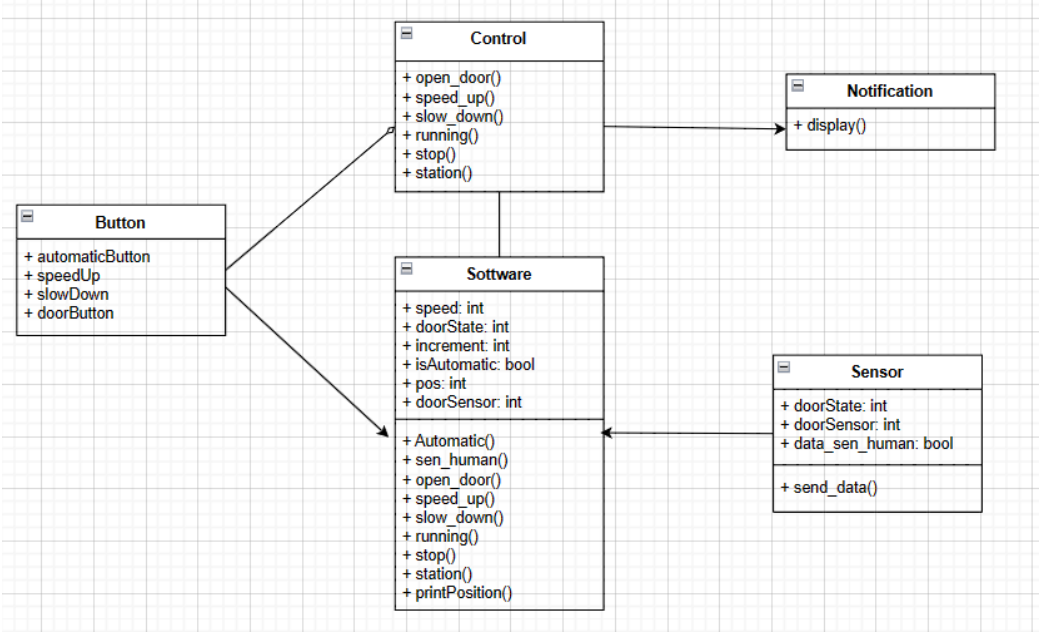
Hình 2. 2: Sơ đồ UseCase

- State Diagram



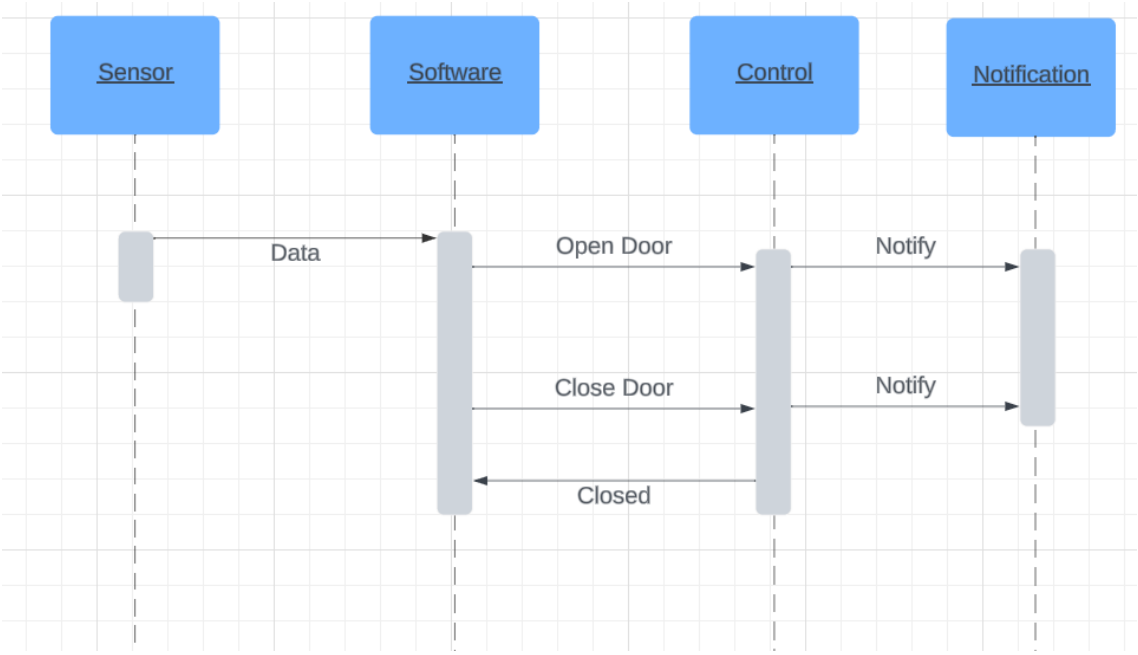
Hình 2. 3: State Diagram của hệ thống

- Class Diagram

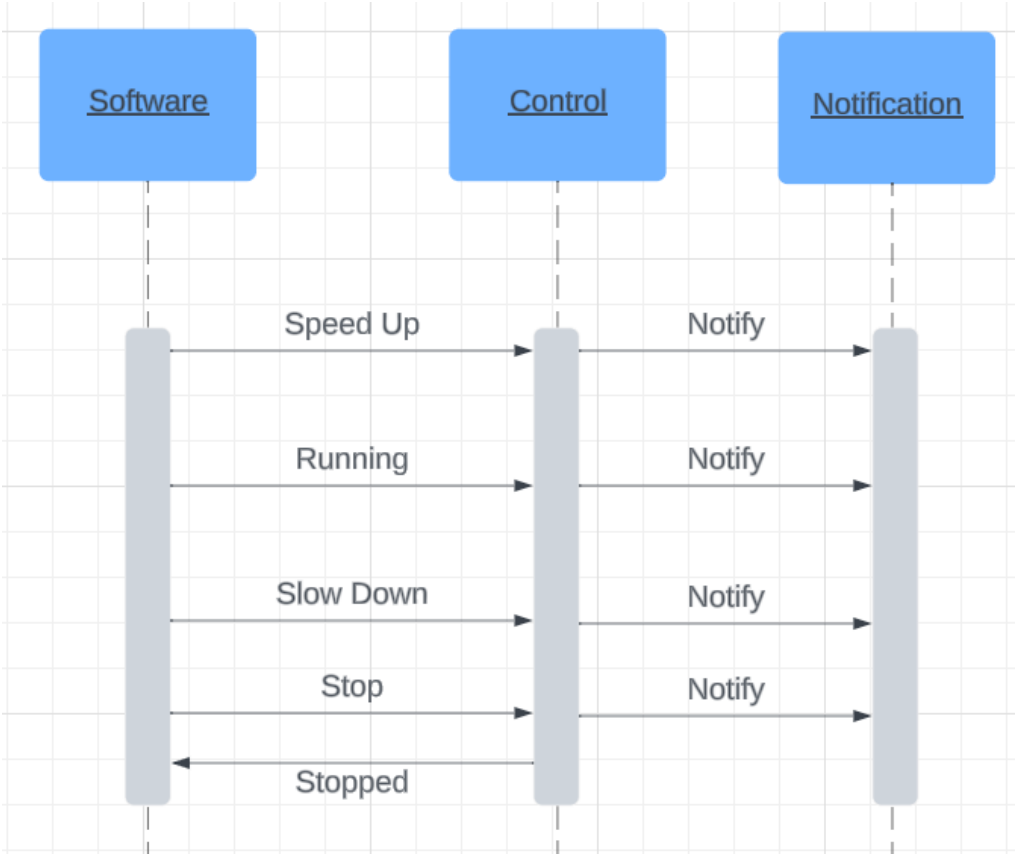


Hình 2. 4: Class Diagram của hệ thống

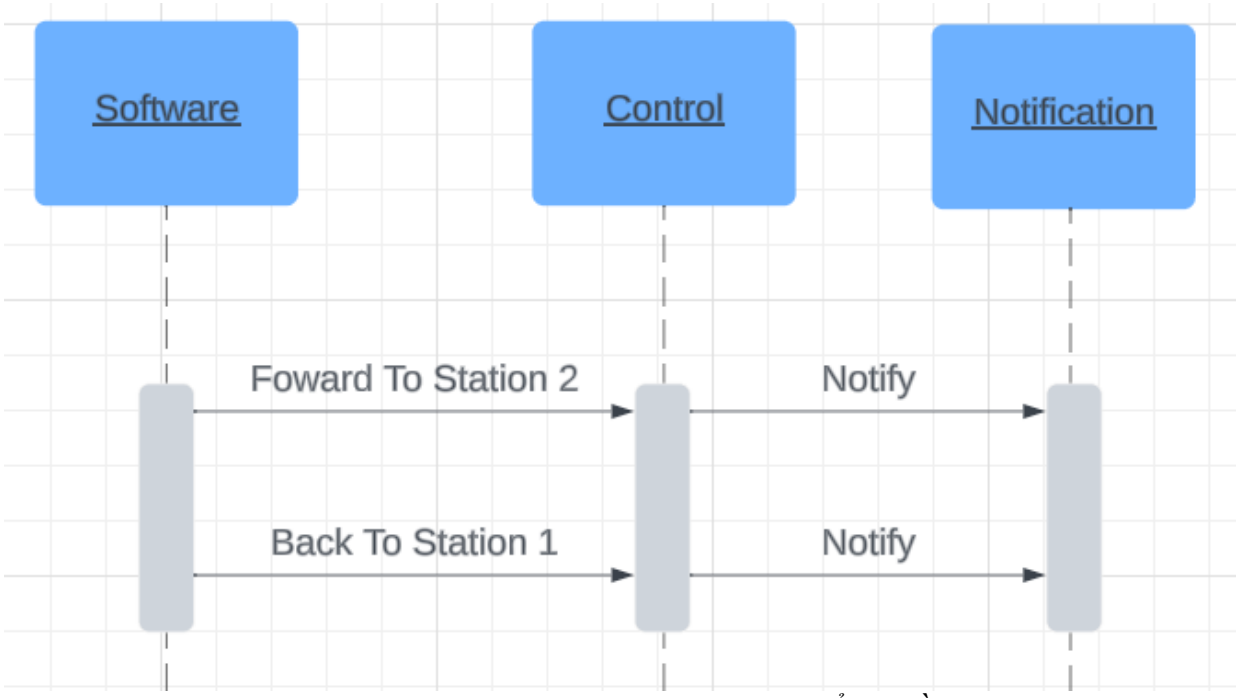
- Sequence Diagram



Hình 2. 5: Sequence Diagram Đóng, Mở cửa



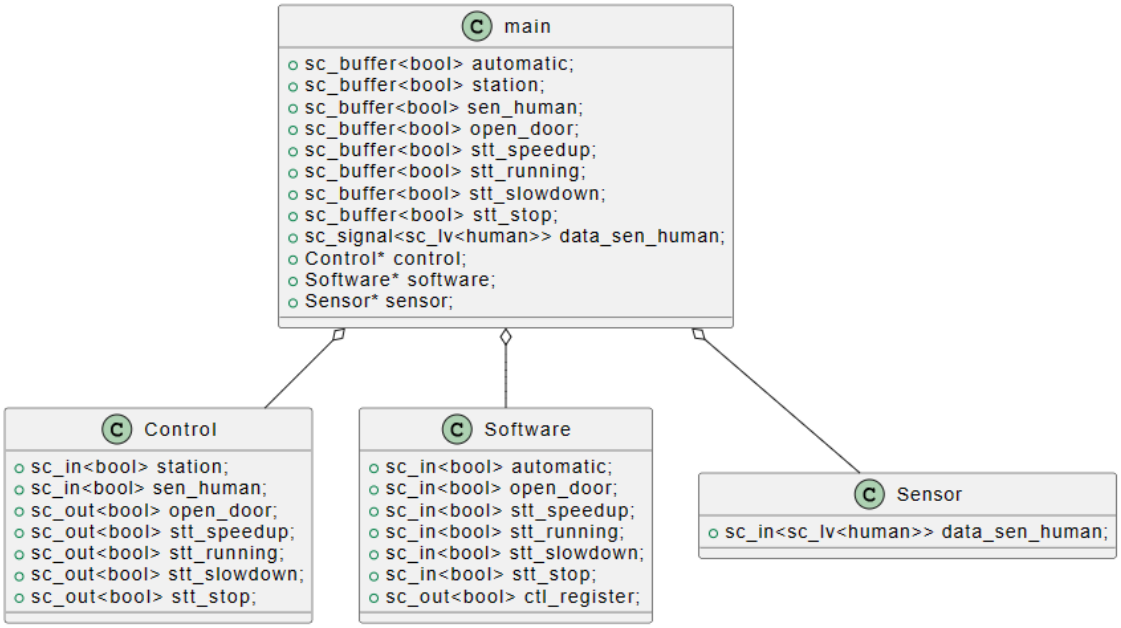
Hình 2. 6: Sequence Diagram Điều khiển trạng thái



Hình 2. 7: Sequence Diagram Di chuyển 2 đầu

2.3.2. Mô hình hoá bằng SystemC

SystemC là một thư viện các lớp và macro C++ dùng để mô tả các hệ thống có các sự kiện xảy ra đồng thời, nên có thể dùng để mô tả phần cứng. Hình 1.2 dùng để mô tả các thành phần của quá trình mô hình hoá bằng SystemC,



Hình 2. 8: Mô hình hoá bằng SystemC

- Class main: Chứa các tín hiệu đầu vào, đầu ra của hệ thống phục vụ cho việc mô phỏng
- Class control: Chứa tín hiệu của phần điều khiển
- Class software: Chứa tín hiệu của phần mềm
- Class Sensor: Chứa dữ liệu cảm biến hồng ngoại

Bảng 2. 1: Tín hiệu đầu vào, đầu ra của Control

Tên	Input/Output	Mô tả
state	input	Vị trí bến
sen_human	input	Cảm biến hồng ngoại
open_door	output	Tín hiệu mở cửa
stt_speedup	output	Tín hiệu tăng tốc
stt_running	output	Tín hiệu đang chạy
stt_slowdown	output	Tín hiệu giảm tốc
stt_stop	output	Tín hiệu dừng

Bảng 2. 2: Tín hiệu đầu vào, đầu ra của Software

Tên	Input/Output	Mô tả

automatic	input	Tín hiệu chạy hệ thống tự động
open_door	input	Tín hiệu mở cửa
stt_speedup	input	Tín hiệu tăng tốc
stt_running	input	Tín hiệu đang chạy
stt_slowdown	input	Tín hiệu giảm tốc
stt_stop	input	Tín hiệu dừng
ctl_register	output	Đưa tín hiệu vào khối thanh ghi

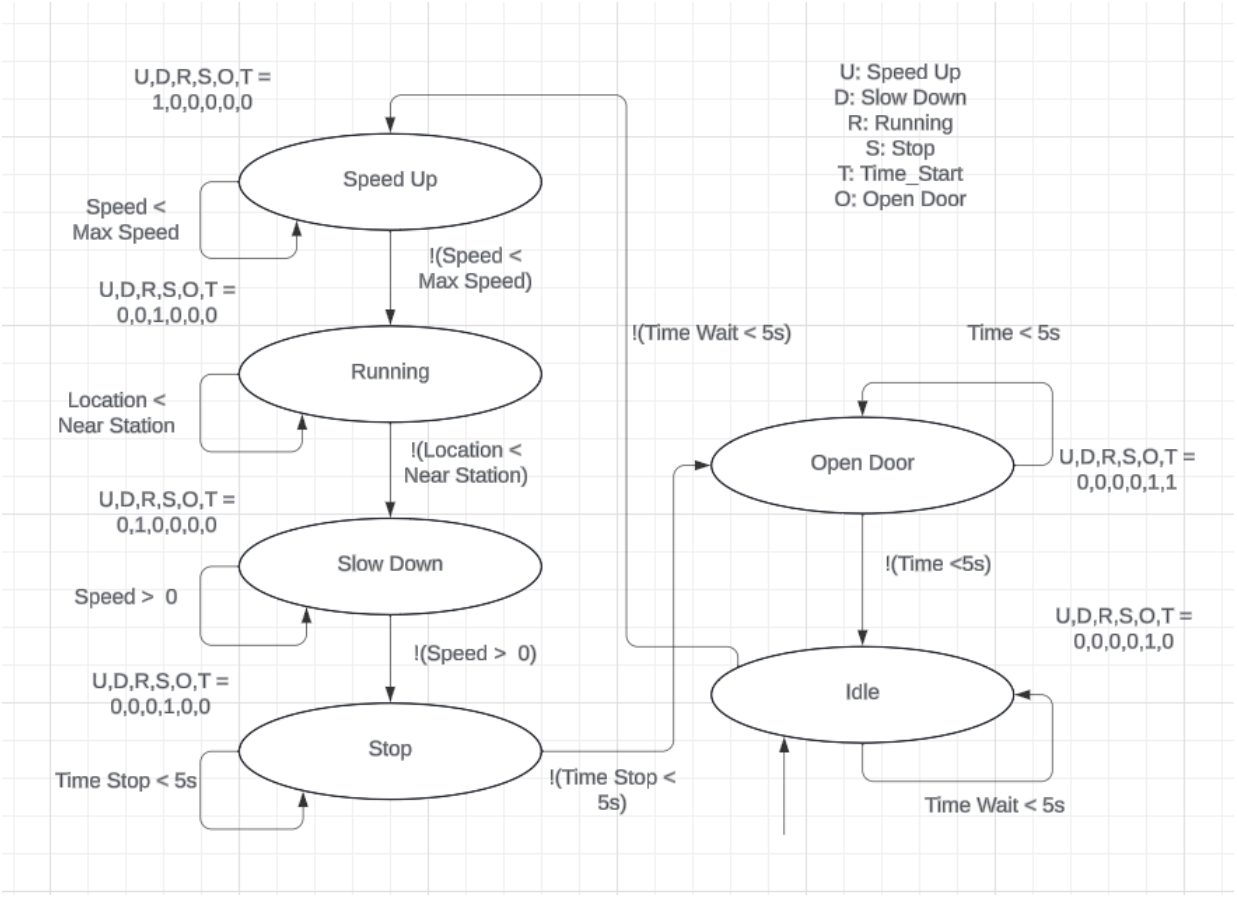
Bảng 2. 3: Tín hiệu đầu vào, đầu ra của cảm biến hồng ngoại

Tên	Input/Output	Mô tả
data_sen_human	input/output	Tín hiệu từ cảm biến hồng ngoại

2.3.3. Mô hình hoá bằng FSM

Máy trạng thái hữu hạn (Finite State Machine - FSM) hoặc Máy tự động trạng thái hữu hạn (finite-state automaton FSA), hoặc là máy tự động hữu hạn, hoặc gọi đơn giản là máy trạng thái, là một mô hình toán học. Nó là một máy trừu tượng luôn có trạng thái nằm trong tổng hữu hạn các trạng thái tại bất kỳ thời điểm nào. Máy trạng thái hữu hạn có thể chuyển từ trạng thái này sang trạng thái khác để phù hợp với đầu vào; sự thay đổi này được gọi là quá trình chuyển đổi. Máy trạng thái hữu hạn được xác định bởi danh sách các trạng thái của nó, trạng thái khởi đầu, và các điều kiện cho từng sự chuyển đổi trạng thái.

Ở dự án này, chúng em sử dụng 6 trạng thái: Open Door, Idle, Speed Up, Slow Down, Running và Stop. Cùng các sự kiện, đầu vào và điều kiện chi tiết ở Hình 2.9:



Hình 2. 9: Mô hình hoá bằng FSM

Mô tả chi tiết:

- Bắt đầu hệ thống, bắt đầu ở trạng thái mở cửa (Open Door): Nếu thời gian chờ để kết thúc trạng mở cửa < 5 giây hoặc nếu có người vào thì vẫn duy trì trạng thái mở cửa; còn nếu không thì chuyển sang trạng thái chờ (Idle).
- Tại trạng thái chờ (Idle), nếu thời gian chờ < 5 giây thì vẫn giữ trạng thái; còn nếu không chuyển sang trạng thái tăng tốc (Speed Up(.
- Sau đó tiến tới trạng thái tăng tốc (Speed Up): nếu tốc độ chưa đạt được tốc độ ngưỡng quy định (Max Speed) thì vẫn tiếp tục tăng, còn nếu đã đạt tốc độ ngưỡng quy định thì chuyển sang chế độ duy trì tốc độ (Running).
- Khi đang ở trạng thái duy trì tốc độ (Running) nếu vị trí hiện tại vẫn chưa tới điểm gần bến thì vẫn duy trì trạng thía, còn nếu không chuyển sang trạng thái giảm tốc độ (Slow Down).
- Tại trạng thái giảm tốc độ (Slow Down), nếu tốc độ hiện tại vẫn chưa giảm tới 0 thì vẫn tiếp tục giảm, còn nếu đã tới 0 chuyển sang trạng thái dừng (Stop).
- Tại trạng thái dừng (Stop), nếu thời gian dừng < 5 giây thì vẫn giữ trạng thái, sau 5 giây chuyển sang trạng thái mở cửa (Open Door).

CHƯƠNG 3. THIẾT KẾ HỆ THỐNG

3.1. Tìm hiểu link kiện

3.1.1. Board mạch ESP8266 NodeMCU Lua V3 CH340:

ESP8266 là một vi mạch (chip) WiFi mạnh mẽ được sản xuất bởi Espressif Systems, cho phép kết nối internet thông qua Wi-Fi. Đây là một công cụ quan trọng trong phát triển các ứng dụng IoT (Internet of Things) và dự án điều khiển từ xa. ESP8266 tích hợp sẵn Wi-Fi và cung cấp giao diện để gửi và nhận dữ liệu qua mạng Wi-Fi, đồng thời có thể được lập trình để thực hiện nhiều chức năng khác nhau, từ điều khiển đèn LED đến thu thập dữ liệu từ cảm biến và gửi lên đám mây.

NodeMCU ESP8266 là một nền tảng phần cứng dựa trên vi mạch ESP8266 của Espressif Systems. Được xây dựng trên board mạch, NodeMCU cung cấp một cách tiếp cận dễ dàng hơn để phát triển các ứng dụng IoT và các dự án điện tử. Với tính năng hỗ trợ firmware được xây dựng sẵn và các chức năng như cổng USB tích hợp, GPIOs, ADCs, NodeMCU là một công cụ linh hoạt và thuận tiện để kết nối Wi-Fi, điều khiển thiết bị và xây dựng các ứng dụng IoT.

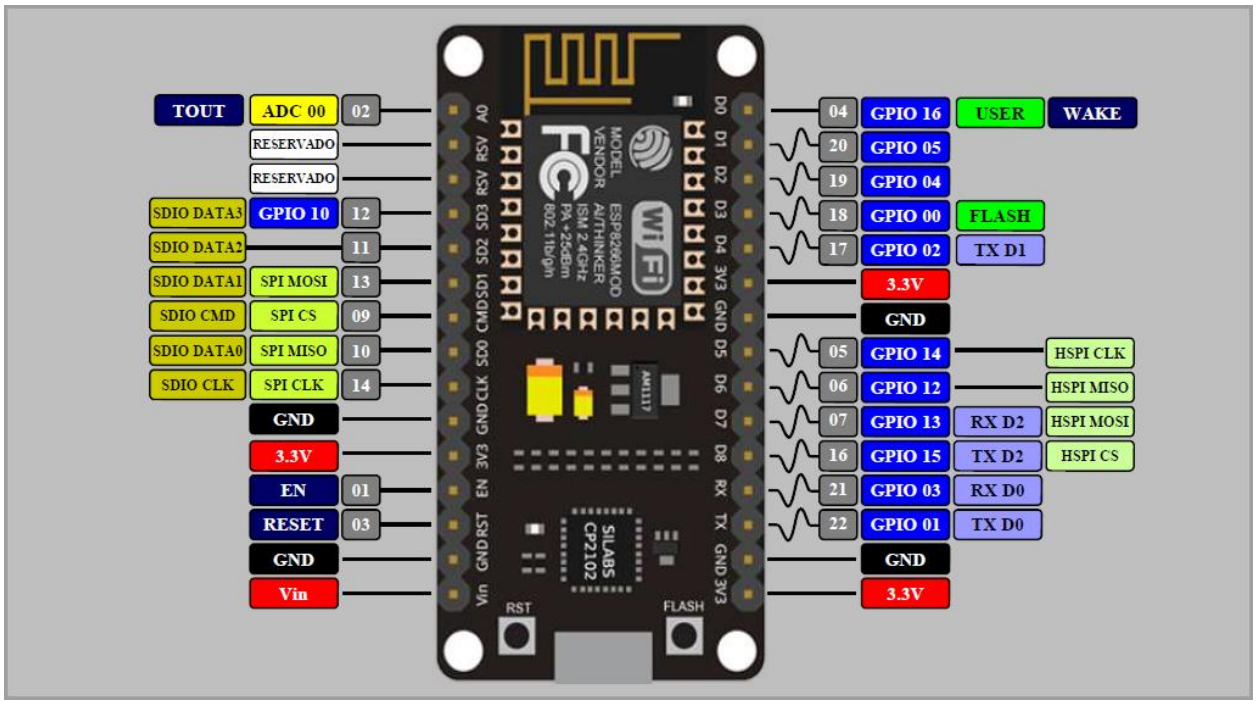
Kit thu phát Wifi ESP8266 NodeMCU Lua V3 CH340 là phiên bản NodeMCU sử dụng IC nạp CH340 từ Lolin với bộ xử lý trung tâm là module Wifi SoC ESP8266, kit có thiết kế dễ sử dụng và đặc biệt là có thể sử dụng trực tiếp trình biên dịch của Arduino để lập trình và nạp code, điều này khiến việc sử dụng và lập trình các ứng dụng trên ESP8266 trở nên rất đơn giản.

Thông số kĩ thuật:

- IC chính: ESP8266 Wifi SoC.
- Phiên bản firmware: NodeMCU Lua
- Chip nạp và giao tiếp UART: CH340
- GPIO tương thích hoàn toàn với firmware Node MCU.
- Cấp nguồn: 5VDC MicroUSB hoặc Vin.
- GIPO giao tiếp mức 3.3VDC
- Số chân I/O: 17 chân GPIO
- Kết nối mạng: WiFi 802.11 b/g/n
- Giao diện mạng: TCP/IP
- Đồng hồ thời gian thực (RTC): không tích hợp
- Bộ nhớ trong: 4MB
- Tích hợp Led báo trạng thái, nút Reset, Flash.
- Tương thích hoàn toàn với trình biên dịch Arduino.
- Kích thước: 59 x 32mm
- Hỗ trợ các giao thức: MQTT, CoAP, HTTP/HTTP

Hình 3. 1 ESP8266

Sơ đồ chân ESP8266 NodeMCU:



Hình 3. 2 Sơ đồ chân ESP8266 NodeMCU

Các chân của kit và các loại giao tiếp mà chúng sử dụng:

17 chân GPIO	Được sử dụng để đọc hoặc điều khiển các tín hiệu kỹ thuật số.
1 chân Analog Input	Được sử dụng để đo giá trị analog từ cảm biến hoặc thiết bị tương tự
2 chân giao tiếp UART	NodeMCU ESP8266 có chân TX và RX cho việc truyền và nhận dữ liệu qua UART, giúp kết nối với các thiết bị hoặc module khác sử dụng giao tiếp này.
4 đầu ra PWM	4 chân PWM để điều khiển tốc độ động cơ hoặc độ sáng của đèn LED.
2 giao tiếp SPI và 1 giao tiếp I2C	2 giao tiếp SPI và một giao tiếp I2C để kết nối các cảm biến và thiết bị ngoại vi khác.
Giao tiếp I2S	Một giao tiếp I2S

Bảng 3. 1 Các chân của kit và các loại giao tiếp

3.1.2. Màn Hình Oled 0.96 Inch Giao Tiếp I2C

Trong hệ thống này, màn hình OLED sẽ được sử dụng để hiển thị các thông số như tốc độ vòng quay của trục, trạng thái của tàu, khoảng cách từ tàu tới ga để có thể điều khiển động cơ tăng tốc, giảm tốc một cách tự động.

Hình 3. 3 Màn Hình Oled 0.96 Inch

3.1.3. Động cơ vàng 2 trục DC:

Động cơ DC có gắn giảm tốc 2 trục và hoạt động với dải điện áp từ 3-12V.

Số vòng/1 phút: 125 vòng/phút tại 3V và 208 vòng/phút tại 5V.

Hình 3. 4: Động cơ giảm tốc vàng 2 trục

3.1.4. Module cảm biến hồng ngoại LM393

Cảm biến có khả năng nhận biết vật cản ở môi trường với một cặp LED thu phát hồng ngoại để truyền và nhận dữ liệu hồng ngoại. Tia hồng ngoại phát ra với tần số nhất định, khi có vật cản trên đường truyền của LED phát nó sẽ phản xạ vào LED thu hồng ngoại, khi đó LED báo vật cản trên module sẽ sáng, khi không có vật cản, LED sẽ tắt.

Với khả năng phát hiện vật cản trong khoảng 2 ~ 30cm và khoảng cách này có thể điều chỉnh thông qua chiết áp trên cảm biến cho thích hợp với từng ứng dụng cụ thể như: xe dò line, xe tránh vật cản, …

Thông số kỹ thuật:

- IC so sánh: LM393
- Điện áp: 3.3V - 6VDC
- Dòng tiêu thụ:

  + Vcc = 3.3V: 23 mA

  + Vcc = 5.0V: 43 mA
- Góc hoạt động: 35°
- Khoảng cách phát hiện: 2 ~ 30 cm
- LED báo nguồn và LED báo tín hiệu ngõ ra
- Mức logic ngõ ra:

+ Mức thấp - 0V: khi có vật cản

+ Mức cao - 5V: khi không có vật cản

- Kích thước: 3.2cm x 1.4cm

Hình 3. 5: Module cảm biến hồng ngoại LM393

3.1.5. Động cơ Servo SG90:

Servo là một dạng động cơ điện đặc biệt. Không giống như động cơ thông thường cứ cắm điện vào là quay liên tục, servo chỉ quay khi được điều khiển (bằng xung PPM) với góc quay nằm trong khoảng bất kì từ 0o - 180o. Mỗi loại servo có kích thước, khối lượng và cấu tạo khác nhau. Có loại thì nặng chỉ 9g (chủ yếu dùng trên máy bay mô mình), có loại thì sở hữu một momen lực bá đạo (vài chục Newton/m), hoặc có loại thì khỏe và nhông sắc chắc chắn, ...

Động cơ Servo được thiết kế những hệ thống hồi tiếp vòng kín. Tín hiệu ra của động cơ được nối với một mạch điều khiển. Khi động cơ quay, vận tốc và vị trí sẽ được hồi tiếp về mạch điều khiển này. Nếu có bất kỳ lý do nào ngăn cản chuyển động quay của động cơ, cơ cấu hồi tiếp sẽ nhận thấy tín hiệu ra chưa đạt được vị trí mong muốn. Mạch điều khiển tiếp tục chỉnh sai lệch cho động cơ đạt được điểm chính xác. Các động cơ servo điều khiển bằng liên lạc vô tuyến được gọi là động cơ servo RC (radio-controlled).

Trong thực tế, bản thân động cơ servo không phải được điều khiển bằng vô tuyến, nó chỉ nối với máy thu vô tuyến trên máy bay hay xe hơi. Động cơ servo nhận tín hiệu từ máy thu này.

Thông số kỹ thuật:

- Khối lượng: 9g
- Kích thước: 22.2x11.8.32 mm

- Momen xoắn: 1.8kg/cm
- Tốc độ hoạt động: 60 độ trong 0.1 giây
- Điện áp hoạt động: 4.8V(~5V)
- Nhiệt độ hoạt động: 0 ºC – 55 ºC
- Kết nối dây màu đỏ với 5V, dây màu nâu với mass, dây màu cam với chân phát xung của vi điều khiển. Ở chân xung cấp một xung từ 1ms-2ms theo để điều khiển góc quay theo ý muốn.

Hình 3. 6: Động cơ Servo SG90

3.1.6. Cảm biến tốc độ Encoder V2:

Mạch cảm biến tốc độ Encoder V2 có thiết kế dễ lắp đặt được sử dụng với dĩa Encoder mica đi kèm có 20 lỗ, tương thích với Encoder V1, nếu bạn muốn có số xung nhiều hơn có thể thiết kễ dĩa Encoder lớn hơn với số lỗ lớn hơn.

Thông số kỹ thuật:

- Điện áp sử dụng: 3.3~5VDC
- Dòng sử dụng: 15mA
- Mức tín hiệu xuất ra: Digital TTL
- Khoảng cách giữa hai mắt phát và thu: 5mm

Nguyên tắc hoạt động của mạch bao gồm 1 mắt phát và 1 mắt thu hồng ngoại đặt cách nhau qua 1 ke hở, khi ánh sáng từ mắt phát đi được tới mắt thu (xuyên qua lỗ của dĩa encoder) thì sẽ có tín hiệu mức cao (5v) phát ra khỏi chân out, khi bị che lại thì chân out phát ra tín hiệu mức thấp (0v).

➔ Số vòng/phút với đĩa quay 20 lỗ: RPM =

➔ Công thức tính tốc độ của trục quay: $v = \frac{\text{Số xung}}{20} \times 2 \times Bán\ kính\ đĩa\ quay \times \pi$(m/s)

ENCODER

Hình 3. 7: Cảm biến tốc độ Encoder V2

3.1.7. Cảm biến khoảng cách siêu âm Ultrasonic Sensor (HC-SR04)

Do đề tài này chỉ dừng lại ở mức làm mô hình cỡ nhỏ nên cảm biến đo khoảng cách em chọn là HC-SR04.

Hình 3. 8: Cảm biến siêu âm HC-SR04

Thông số kỹ thuật:

- Số model: HC - SR04 CS100A.
- Điện áp hoạt động: 3.3- 5VDC.
- Dòng tĩnh: <2mA.
- Góc cảm ứng: <15 độ
- Tần số phát sóng: 40Khz
- Khoảng cách đo: từ 2 - 450cm (4.5m) (khoảng cách xa nhất đạt được ở điều khiện lý tưởng với không gian trống và bề mặt vật thể bằng phẳng, trong điều kiện bình thường

cảm biến cho kết quả chính xác nhất ở khoảng cách <100cm).

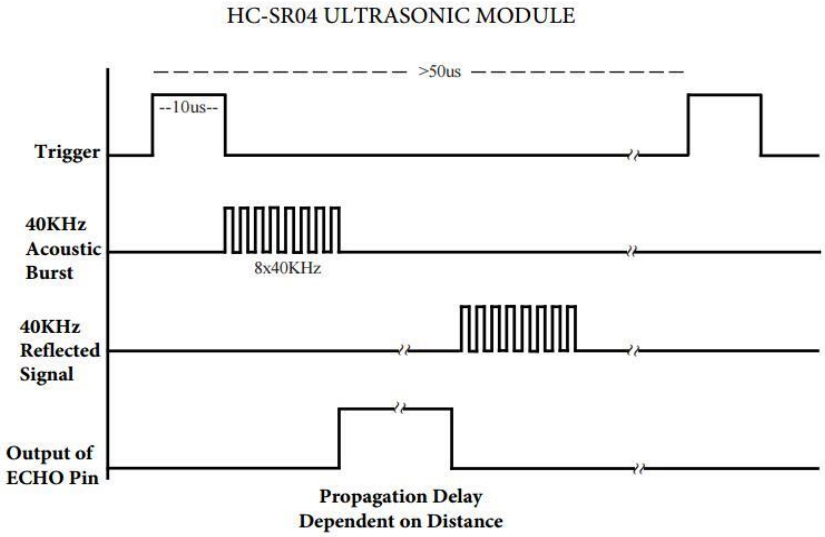
- Độ chính xác: 0,1 cm sai số 1%. (khoảng cách xa nhất đạt được ở điều khiện lý tưởng với không gian trống và bề mặt vật thể bằng phẳng, trong điều kiện bình thường cảm biến cho kết quả chính xác nhất ở khoảng cách <100cm).
- Nhiệt độ hoạt động: -40 đến +85 độ.
- Kích thước: 43mm x 20mm x 17mm
- Cổng đầu ra: GPIO.

Cảm biến siêu âm HC-SR04 là cảm biến cơ bản mà chúng ta sử dụng để xác định khoảng cách của một vật thể. Theo một cách khác, chúng ta có thể nói rằng nó được sử dụng để đo khoảng cách của vật thể với một điểm tham chiếu cụ thể. Về cơ bản nó có 4 chân;

- Trig pin
- Echo pin
- GND pin
- VCC pin (+5V)

Nguyên lý hoạt động:

Cảm biến siêu âm HC-SR04 hoạt động bằng cách sử dụng sóng siêu âm để đo khoảng cách từ chính cảm biến đến đối tượng mà nó hướng đến. Đầu tiên, cảm biến phát ra sóng siêu âm từ bộ phát siêu âm bên trong. Sóng siêu âm này không thể nghe được đối với tai người vì nó có tần số cao hơn tầm nghe được của con người. Khi sóng này va vào một đối tượng, nó sẽ bị phản xạ và trở về cảm biến.



Hình 3. 9: 100 HC-SR04 timing diagram

Cảm biến HC-SR04 có 2 chân TRIGGER và ECHO riêng biệt, khi chân MODE để trống, HC-SR04 sẽ sử dụng cả 2 chân chức năng TRIGGER và ECHO cho việc điều khiển hoạt động của cảm biến. Từ biểu đồ thời gian của chế độ hoạt động ta thấy để đo khoảng cách, đầu tiên ta phát 1 xung cỡ 5 us từ chân TRIG, sau đó cảm biến sẽ tạo ra xung HIGH ở chân ECHO cho đến khi nhận được xung phản xạ ở chân này. Chiều rộng của xung sẽ bằng với thời gian sóng siêu âm được phát từ cảm biến quay trở lại

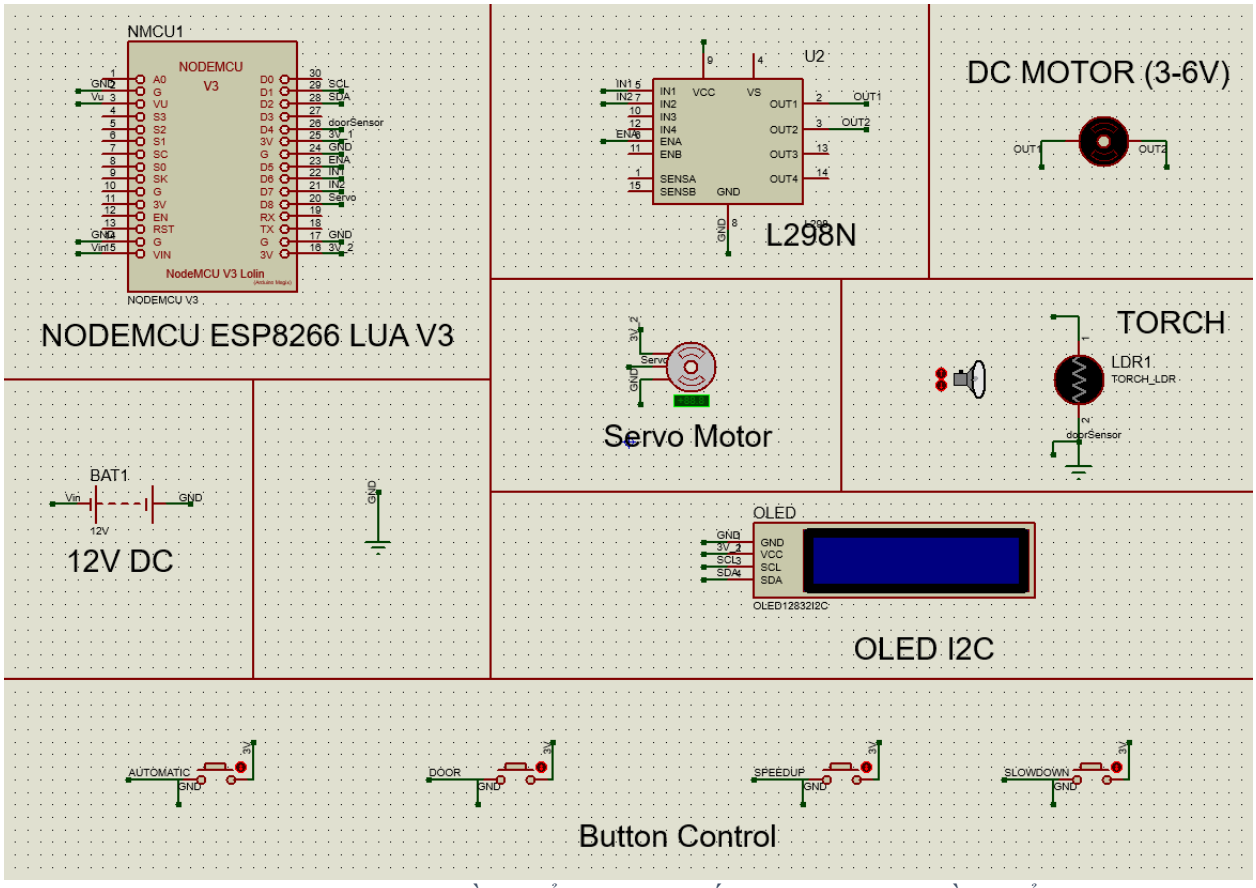
Tốc độ của âm thanh trong không khí tương đương 29,412 us/cm =3.108 m/s. Khi có khoảng cách từ cảm biến tới vật cản được tính bằng công thức:

$$d = \frac{Thời\ gian\ từ\ lúc\ phát\ đến\ lúc\ thu\ (\mu s)}{29{,}412}$$

3.2 Thiết kế mạch

3.2.1 Vẽ và mô phỏng trên phần mềm Proteus

a) Mạch điều khiển động cơ L298N kết hợp với mạch điều khiển cánh cửa tự động sủ dụng đông cơ Servo SG90



Hình 3. 10 Mô phỏng mạch điều khiển động cơ kết hợp với mạch điều khiển cánh cửa tự động

Mạch điều khiển cánh cửa:

Trong hệ thống tàu con thoi tự động trong sân bay, tàu được thiết kế đóng mở cửa tự động, kết hợp nhiều loại cảm biến để có thể đóng mở tự động. Nhưng vì đề tài chỉ trong mức thực hiện mô hình cỡ nhỏ, nên nhóm chúng em đã tìm hiểu và đưa ra một mô hình tổng quát nhỏ gọn:

Mô hình mạch kết hợp giữa kit NodeMCU Esp8266 Lua V3, cảm biến hồng ngoại LM393, động cơ Servo SG90 và màn hình OLED I2C.

Các chân kết nối:

- OLED I2C: 2 chân VCC và GND nối với các chân cấp nguồn của kit Wifi, 2 chân SCL và SDA lần lượt nối với 2 chân giao tiếp I2C D1 và D2.
- Cảm biến hồng ngoại LM393: ngoài 2 chân cấp nguồn và nối đất, 1 chân OUT của cảm biến nối với chân GPIO D4 của kit
- Động cơ Servo SG90: 2 chân cấp nguồn và nối đất, 1 chân nhận tín hiệu nối với GPIO D8 của kit.

Nguyên lý hoạt động:

- Ngay khi tàu dừng, tức tàu đang trong trạng thái STOP, cánh cửa sẽ mở tự động để đón và trả khách, khi hết thời gian đón trả khách, cửa sẽ tự động đóng.
- Trong khi cửa đóng, nếu cảm biến phát hiện có người hoặc vật đi qua hoặc nằm giữa cánh cửa (cảm biến trả về giá trị 1 khi không có vật cản và 0 khi có vật cản), cửa sẽ tự động mở lại và đợi một khoảng thời gian tiếp rồi mới đóng.
- Có thể điều khiển cửa đóng mở bằng nút bấm DOOR khi tàu đang trong trạng thái dừng STOP.

Mạch điều khiển động cơ:

- Mạch điều khiển động cơ bao gồm: NodeMCU Esp8266 Lua V3, module mạch cầu điều khiển động cơ L298N, động cơ vàng giảm tốc 2 trục, nguồn 1 chiều 12V và màn hình OLED I2C.

Các chân kết nối:

- OLED I2C: 2 chân VCC và GND nối với các chân cấp nguồn của kit Wifi, 2 chân SCL và SDA lần lượt nối với 2 chân giao tiếp I2C D1 và D2.

- Module L298N: 2 chân cấp nguồn (12V) và nối đất được nối với chân VU và GND của kit Wifi, 3 chân ENA (dùng để điều khiển xung PWM tức điều khiển tốc độ quay của động cơ), IN1 và IN2 (dùng để điều khiển chiều quay của động cơ) lần lượt kết nối với các chân D5, D6 và D7 của kit Wifi

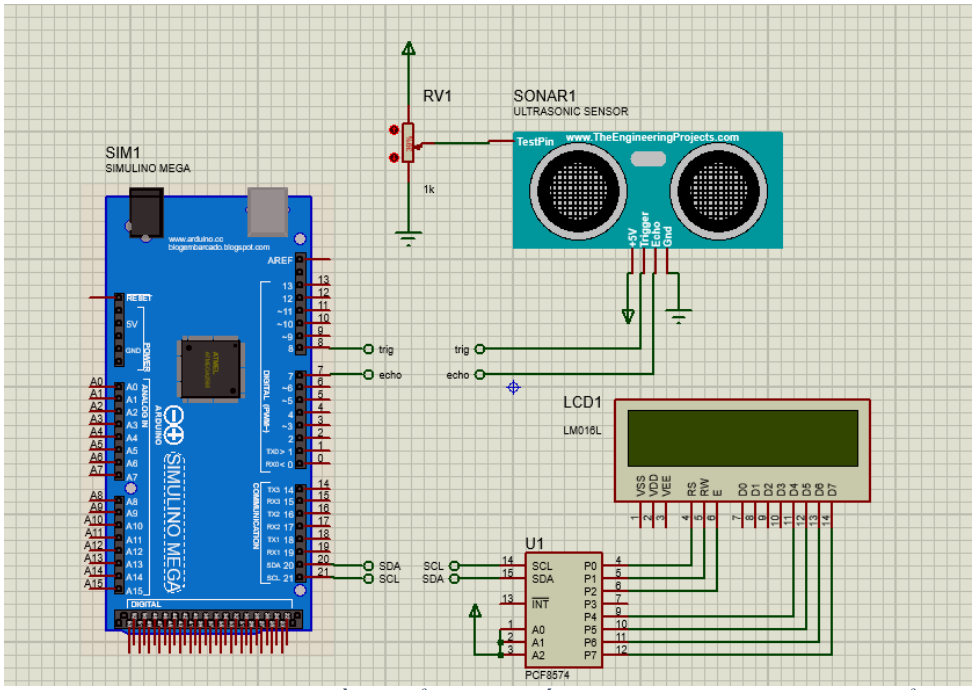
- Động cơ vàng giảm tốc 2 trục: 2 chân được nối với 2 chân OUT1, OUT2 của module L298N

Nguyên lý hoạt động:

- Màn hình OLED sẽ hiển thi trạng thái khi tăng, giảm tốc và khi dừng, tốc độ động cơ.

- Sử dụng các nút SPEEDUP, SLOWDOWN để tăng giảm vận tốc

b) Mô phỏng mạch điều khiển cảm biến khoảng cách HC-SR04 hiển thị LCD I2C
Trong phần tìm hiểu này em sử dụng ARDUINO MEGA 2560 để thực hiện mô phỏng.



Hình 3. 11 Mô phỏng mạch điều khiển cảm biến khoảng cách HC-SR04 hiển thị LCD I2C

Code thực hiện mô phỏng:

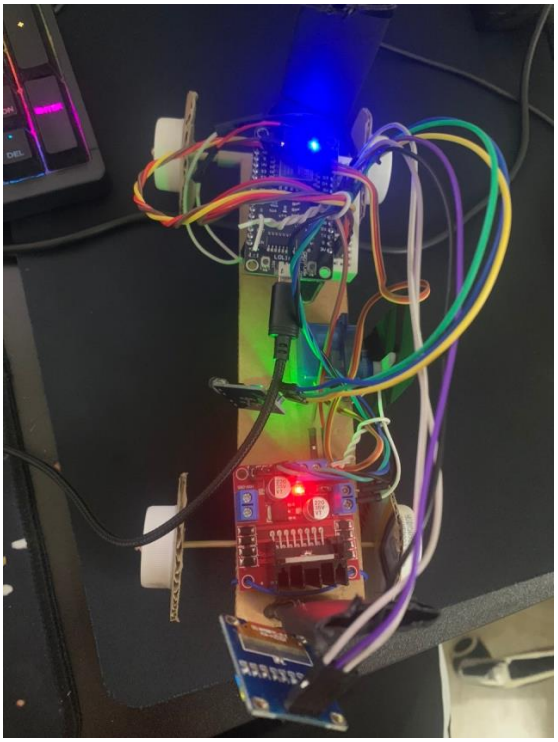
```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
const int trig = 8;
const int echo = 7;
void setup(){
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Distance:");
  lcd.setCursor(13, 0);
  lcd.print(" cm");
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
}
```

```
void loop(){
  unsigned long duration;
  int distance;
  digitalWrite(trig, 0);
  delayMicroseconds(2);
  digitalWrite(trig, 1);
  delayMicroseconds(5);
  digitalWrite(trig, 0);
  duration = pulseIn(echo, HIGH);
  distance = int(duration / 2 / 29.412);
  lcd.setCursor(10, 0);
  lcd.print(distance);
  if (distance < 10) {
    lcd.setCursor(11, 0);
    lcd.print("  ");
  }
  else if (distance < 100) {
    lcd.setCursor(12, 0);
    lcd.print(' ');
  }
    else if (distance < 1000) {
    lcd.setCursor(13, 0);
    lcd.print(' ');
  }
  delay(200);
}
```

Hình 3. 12 Code mạch điều khiển cảm biến khoảng cách HC-SR04

Trong bài này nhóm em mới chỉ thực hiện mô phỏng cảm biến để áp dụng trong việc giảm tốc tàu khi sắp tới các trạm đón trả khách. Nhóm dự tính sau này sẽ áp dụng công việc đó vào mô hình thực tế.

3.2.2 Mạch thực tế:



Hình 3. 13 Mạch điều khiển động cơ kết hợp mạch điều khiển cánh cửa tự động

CHƯƠNG 4. THIẾT KẾ PHẦN MỀM

Trong phần này chúng em sẽ lập trình giao tiếp trình điều khiển động cơ L298N với ESP8266NodeMCU. Chúng em sẽ điều khiển on/off hệ thống và tăng giảm tốc độ động cơ, đóng mở cửa tự động.

4.1. Điều khiển động cơ

Với nguyên lý hoạt động của hệ thống mô phỏng và các sơ đồ use case, FSM ở trên, thiết kế phần mềm điều khiển động cơ DC bằng module điều khiển động cơ L298N và NodeMCU ESP8266.

Trước tiên, kết nối các chân điều khiển của L298N với ESP8266:

```
int doorSensor = 2;
int ENA = 14;
int IN1 = 12;
int IN2 = 13;
```

Trong hàm setup(), chúng ta sẽ cấu hình các chân điều khiển kết nối ở trên thành các chân đầu ra và nút nhấn là các chân đầu vào bằng hàm pinMode(), thiết lập các chân đầu vào ở trạng thái Thấp để ban đầu động cơ tắt:

```
pinMode(ENA, OUTPUT);
pinMode(IN1, OUTPUT);
pinMode(IN2, OUTPUT);
digitalWrite(IN1, LOW);
digitalWrite(IN2, LOW);
```

Có hai hàm xử lý logic tăng và giảm tốc độ là speedUp() và slowDown()

```
void speedUp() {
  status = TrainState::SPEEDUP;
  if(position == Position::STATION1) {
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
  } else {
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
  }
  printMessage("Dang tang toc...");
  for (speed = 0; speed <= 255; speed++) {
    analogWrite(ENA, speed);
    Blynk.virtualWrite(V2, speed);
    delay(15);
  }
}

void slowDown() {
  printMessage("Dang giam toc...");
  for (speed ; speed >= 0; speed = speed - 10) {
    Blynk.virtualWrite(V2, speed);
    analogWrite(ENA, speed);
    delay(15);
    if(speed == 0) {

      break;
    }
  }
```

Điều chỉnh tốc độ động cơ bằng cách tăng giảm tín hiệu PWM trên chân ENA. Tín hiều PWM sẽ có giá trị từ 0-255. Qua nhiều lần mô phỏng và chạy thử mô hình, vì là động cơ DC cỡ nhỏ chúng em nhận thấy rằng khi đặt tín hiệu PWM ở giá trị tối đa là 255 thì hệ thống tàu điện chạy với tốc độ hài hòa, khi giảm tín hiệu xuống cỡ 155 thì nó chạy rất chậm, mô phỏng đúng với cơ chế của tàu khi về trạm đến đón/trả khách. Chính vì vậy mới có thông số như trên.

Hàm closeDoor(), openDoor(), interupt(): để đóng mở của tàu và xử lý đặc biệt khi có sự cố trong quá trình đóng cửa

```
void openDoor () {
  if(doorState == 1) {
    myservo.attach(15);
    printMessage("Dang mo cua");
    for(pos = 180; pos>=1; pos-=1) {
      myservo.write(pos);
      delay(15);
    }
    status = TrainState::STOP;
    doorState = 0;
    Blynk.virtualWrite(V0, 0);
    printMessage("Cua da mo");
  }
}

void interupt () {
  printMessage("Dang mo cua");
  for(pos; pos>=1; pos-=1) {
    myservo.write(pos);
    delay(15);
  }
  status = TrainState::STOP;
  printMessage("Cua da mo");
  delay(5000);
  closeDoor();
}
```

```
void closeDoor () {
  if(doorState == 0) {
    printMessage("Dang dong cua");
    for(pos = 0; pos < 180; pos += 1){
      myservo.write(pos);
      delay(15);
      value = digitalRead(doorSensor);
      if(value == 0){
        break;
      }
    }
    if(value == 0) {
      interupt();
    } else {
      printMessage("Cua da dong");
      doorState = 1;
      Blynk.virtualWrite(V0, 1);
      myservo.attach(16);
      speedUp();
    }
  }
```

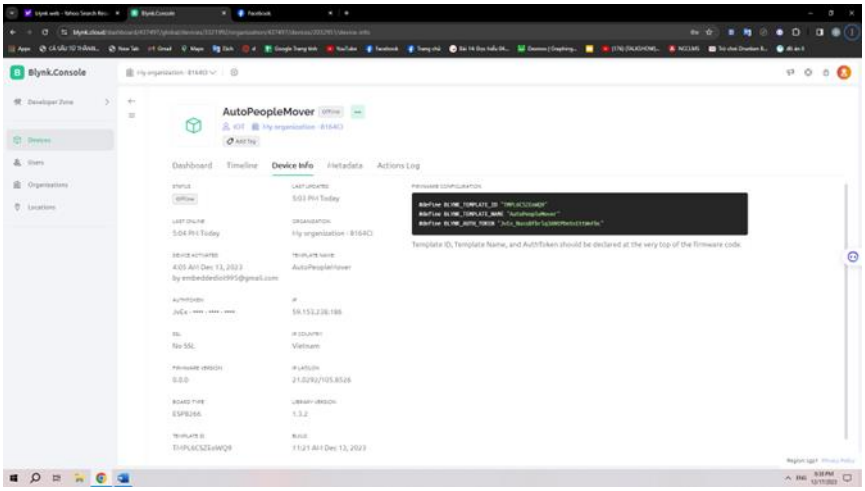
4.2. Lập trình điều khiển sử dụng Blynk

Blynk là một nền tảng IoT cung cấp các công cụ kết nối, quản lý và điều khiển các thiết bị IoT từ xa thông qua mạng Internet. Điểm nổi bật của Blynk là sự dễ dàng và nhanh chóng trong việc sáng tạo và quản lý các ứng dụng Iot, phù hợp cho cả những người mới bắt đầu và những nhà phát triển chuyên nghiệp.

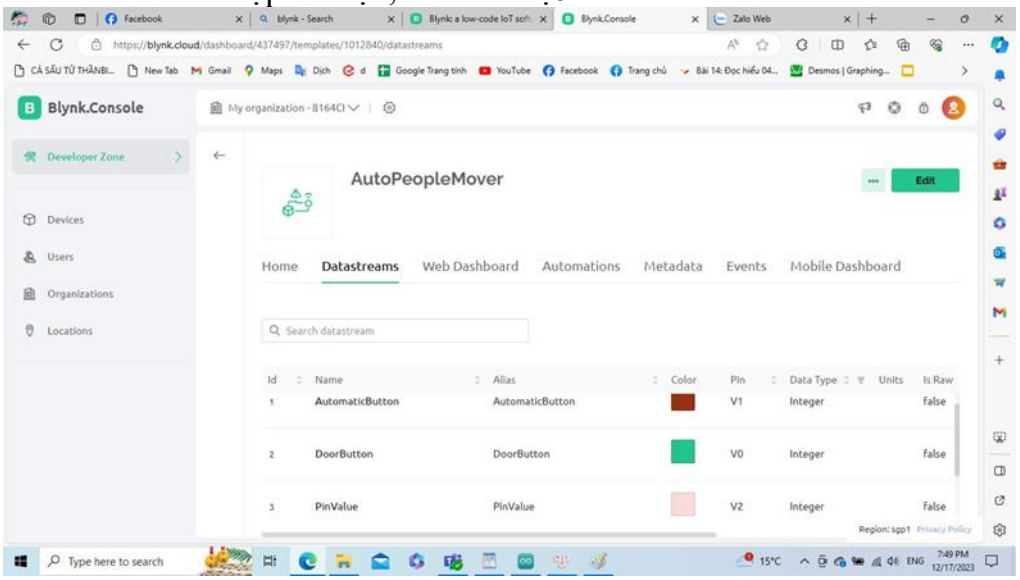
Blynk cung cấp ứng dụng di động và API cho phép người dùng kết nối và điều khiển các thiết bị IoT bằng cách sử dụng các cảm biến và các tín hiệu đầu vào. Giao diện người

dùng được tùy chỉnh linh hoạt để điều khiển thiết bị IoT theo cách tùy chỉnh và tạo ra cách hành động và tương tác phức tạp thông qua mã code.

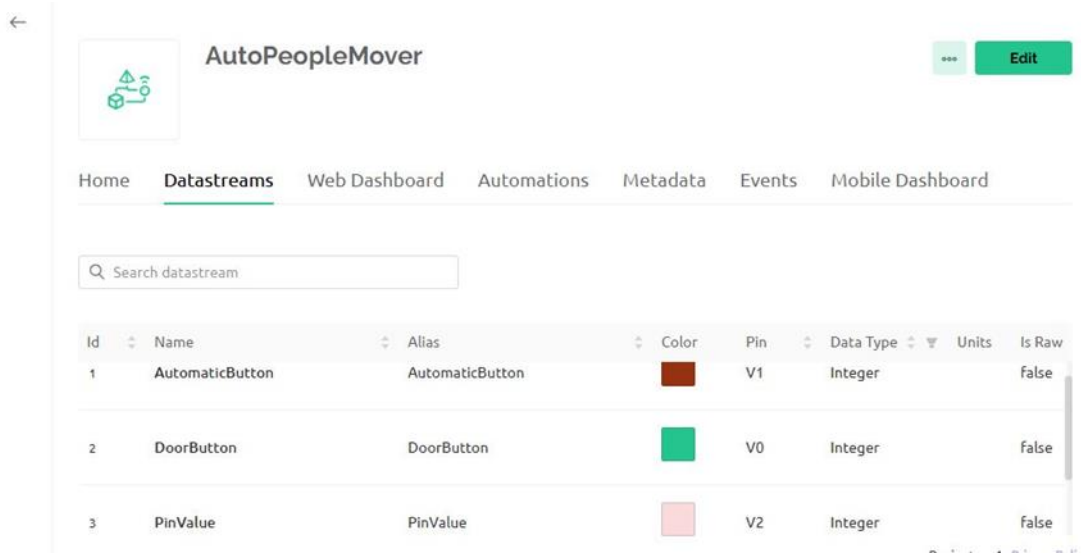
Cấu hình phần mềm: Teamplate ID, Device Name and AuthToken phải được khai báo ở đầu chương trình code.



Thiết lập Datasteams: thiết lập dữ liệu, kiểu dữ liệu



Thiết lập Web Dashboard: Thiết kế nút trên web để điều khiển hệ thống.



Thiết lập điều khiển trên app. Các nút có các chức năng dưới đây:

1) AutomaticButton: tàu chạy tự động
2) DoorButton : đóng mở cửa theo logic đã code Blynk
3) PinValue: tốc độ của tàu (thay đổi 0-255)
4) SpeedUp: tăng tốc độ
5) SlowDown: Giảm tốc độ

Cấu hình phần mềm, kết nối Wifi

```
#define BLYNK_TEMPLATE_ID "TMPL6CSZEoWQ9"
#define BLYNK_TEMPLATE_NAME "AutoPeopleMover"
#define BLYNK_AUTH_TOKEN "JvEx_NaxsBfbrlq3ANtPDeXxEttWeFbc"

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SerialESP8266wifi.h>
#include <BlynkSimpleEsp8266.h>
#include <Servo.h>

char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "abcd";
char pass[] = "1234567889";
```

Thư viện:

- ‘wire.h’: thư viện adruino cho giao tiếp I2C
- ‘servo’ : thư viện điều khiển động cơ servo
- ‘SerialESP8266wifi.h’ : Thư viện tùy chỉnh để liên lạc nối tiếp với mô-đun WiFi ESP8266
- ‘Adafruit_SSD1306.h’: Thư viện cho màn hình OLED
- ‘BlynkSimpleEsp8266.h’: Thư viện kết nối Arduino với nền tảng đám mây Blynk

- Lớp enum

```
enum class TrainState {
  STOP,
  SPEEDUP,
  RUNNING,
  SLOWDOWN
};

enum class Position {
  STATION1,
  STATION2
};
```

Hai lớp enum được định nghĩa: TrainState và Position. Chúng được dùng để thể hiện trạng thái của tàu (STOP, SPEEDUP, RUNNING, SLOWDOWN) và vị trí hiện tại (STATION1, STATION2).

- Blynk Function

1. BLYNK_WRITE(V3)
   Mục đích: điều chỉnh tăng tốc độ tàu

```
BLYNK_WRITE(V3)
{
  int buttonState = param.asInt();
  if (buttonState == HIGH)
  {
    if (speed < 255 - increment)
    {
      speed += increment;
    }
    else
    {
      speed = 255;
    }
    if(position == Position::STATION1) {
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    } else {
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    }
    analogWrite(ENA, speed);
    Blynk.virtualWrite(V3, speed);
```

Định nghĩa hàm trên để xử lý sự kiện khi giá trị của nút kết nối với pin V3 trong ứng dụng Blynk thay đổi. Nếu giá trị là HIGH (được bật) hãy tăng tốc độ theo giá trị gia tăng đã xác định (15), nếu tốc độ chạm ngưỡng tối đa, cập nhập trạng thái của người điều khiển động cơ dựa trên vị trí tàu (trạm 1 hoặc trạm 2) .

2. BLYNK_WRITE(V4)
   Mục đích: điều chỉnh giảm tốc độ tàu

```
BLYNK_WRITE(V4)
{
  int buttonState = param.asInt();
  if (buttonState == HIGH)
  {
    if (speed > increment)
    {
      speed -= increment;
    }
    else
    {
      speed = 0;
    }
    if(position == Position::STATION1) {
      digitalWrite(IN1, LOW);
      digitalWrite(IN2, HIGH);
    } else {
      digitalWrite(IN1, HIGH);
      digitalWrite(IN2, LOW);
    }
    analogWrite(ENA, speed);
    Blynk.virtualWrite(V3, speed);
```

Định nghĩa hàm trên để xử lý sự kiện khi giá trị của nút kết nối với pin V4 trong ứng dụng Blynk thay đổi. Nếu trạng thái là HIGH(được bật) hãy giảm tốc độ theo giá trị đã xác định (15), nếu tốc độ giảm tới ngưỡng tối thiểu, hãy cập nhập trạng thái của người điều khiển động cơ dựa trên vị trí tàu.

3. BLYNK_WRITE(V1)
   Mục đích: Chuyển đổi chế độ tự động

```
BLYNK_WRITE(V1)
{
  int buttonState = param.asInt();
  if (buttonState == HIGH)
  {
    isAutomatic = true;
  }
  else
  {
    isAutomatic = false;
  }
}
```

Định nghĩa hàm trên để xử lý sự kiện khi giá trị của nút kết nối với pin V1 trong ứng dụng Blynk thay đổi. Nếu trạng thái là HIGH (được nhấn nút, trong Blynk tiện ích nút được gửi trạng thái là HIGH khi được gán nhãn), nếu isAutomatic = true thì tàu sẽ hoạt động ở chế độ tự động.

4. BLYNK_WRITE(V0)
   Mục đích: Mở hoặc đóng cửa tàu dựa trện tình trạng hiện tại

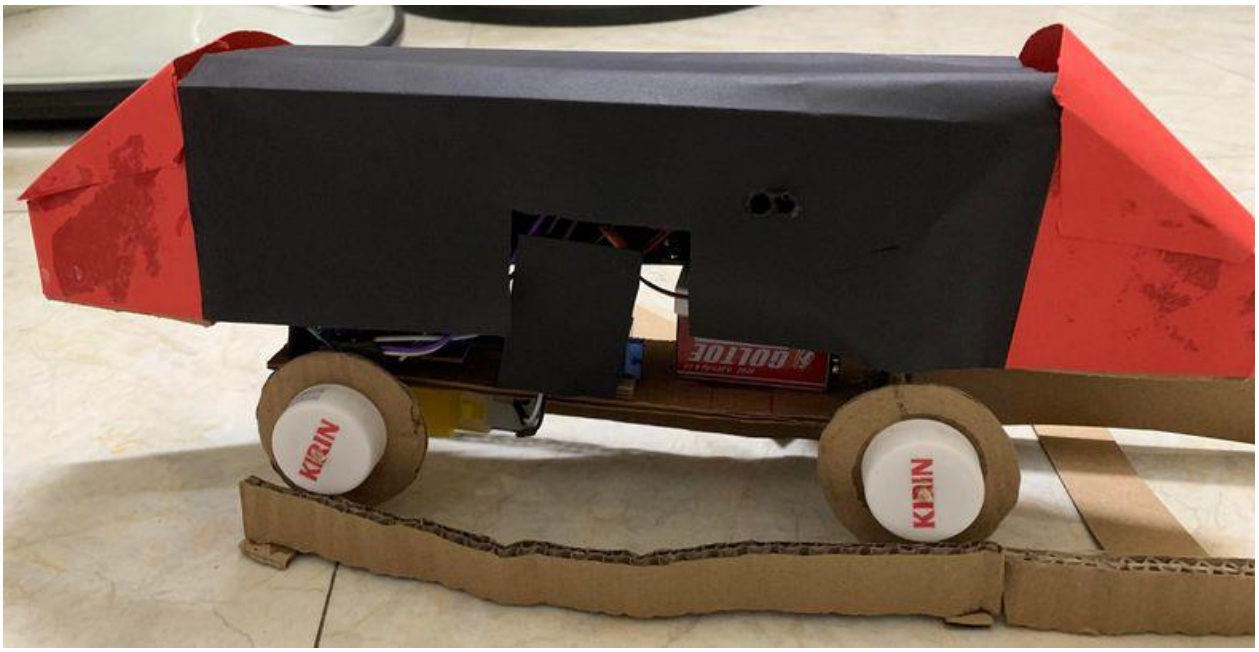
```
BLYNK_WRITE(V0)
{
  int buttonState = param.asInt();
  if(status == TrainState::STOP) {
    if (buttonState == HIGH)

    {
      closeDoor();
    }
  else
    {
      openDoor();
    }
  }
```

Định nghĩa hàm trên để xử lý sự kiện khi giá trị của nút kết nối với pin V0 trong ứng dụng Blynk thay đổi. Nếu tàu đang đứng yên và tôi muốn nhấn nút, nó sẽ đóng cửa. Nếu tàu đang đứng yên và tôi không nhấn nút, nó sẽ mở cửa. Nếu tàu đang chạy hoặc đang tăng tốc, nó sẽ không làm gì.

CHƯƠNG 5. TRIỂN KHAI, THỬ NGHIỆM TOÀN HỆ THỐNG

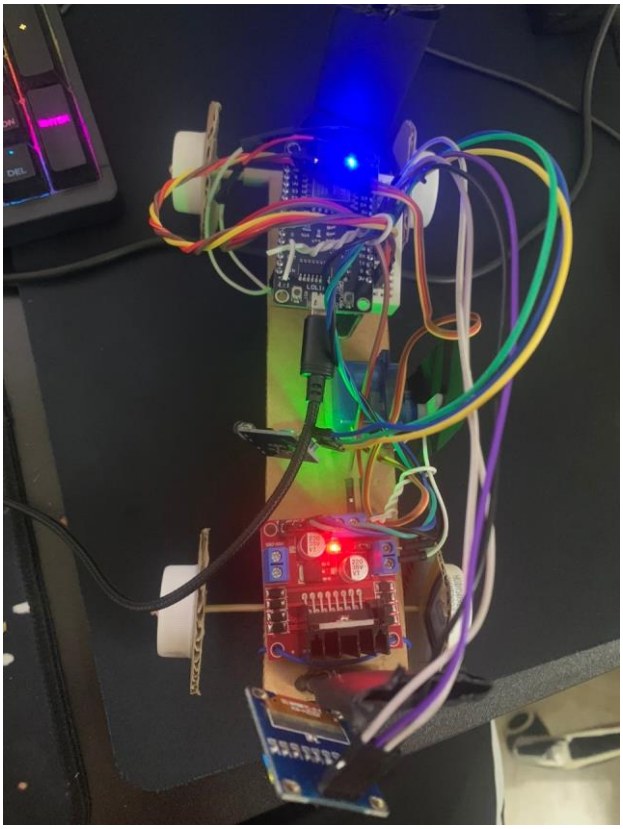
5.1. Mô hình tàu điện con thoi



Hình 5.1 mô hình tàu điện con thoi

Mô hình bao gồm:

- Mạch mô phỏng

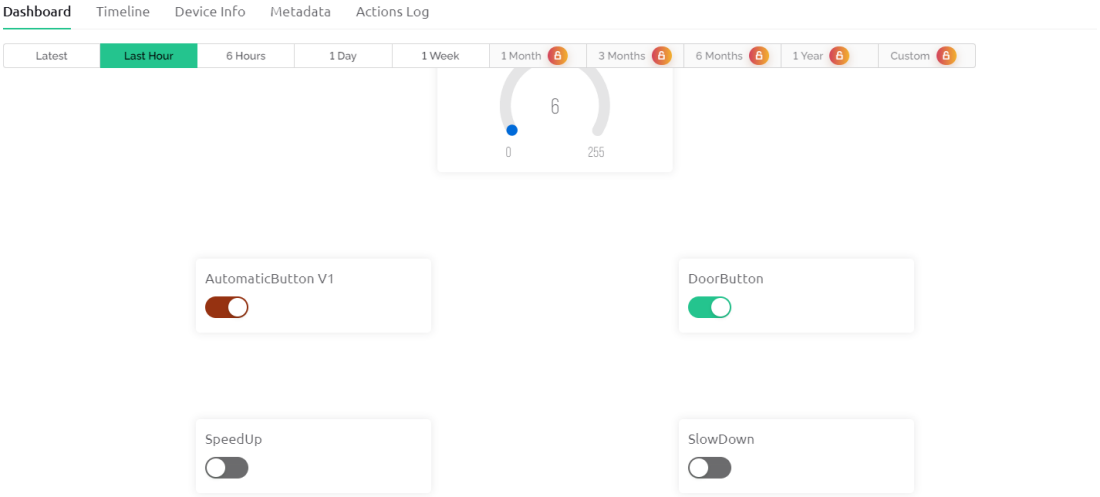


Hình 5.2 mạch mô phỏng hoàn chỉnh

- Khi nạp code vào thì màn LCD sẽ hiện thị như sau:

Hình 5.3 led hiển thị khi chạy chương trình

5.2. App Blynk điều khiển hệ thống



Hình 5.4 App blynk điều khiển hệ thống

Sau khi mở app Blynk, thao tác ấn nút automaticButtonV1 thì tàu chuyển động đồng thời led hiện chữ đang mở cửa như hình sau

Hình 5.5 Tàu đang mở cửa

Trong quá trình vận chuyển người điều khiển có thể chỉnh tăng tốc hoặc giảm tốc độ bằng nút SpeedUp và SlowDown. Mỗi lần bấm sẽ tăng, giảm 15 đơn vị

Hình 5.6 Tàu đang tăng tốc

Hình 5.7 Tàu đang giảm tốc

CHƯƠNG 6. ĐÁNH GIÁ KẾT QUẢ VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

6.1. Đánh giá kết quả

Sau quá trình tìm hiểu và thực hiện đề tài, nhóm chúng em có các đánh giá như sau:

- Mạch chạy ổn định, các phím điều khiển hoạt động đúng yêu cầu như nguyên lý đã đề ra.
- Do không làm mạch in nên các dây cắm trên board trắng còn phức tạp, gây khó khăn trong quá trình kiểm tra và hoạt động của mạch.
- Các thành viên trong nhóm đều có ý thức tìm hiểu, tham gia vào đề tài.

Tuy có nhiều hạn chế về kiến thức và thời gian nhưng về cơ bản nhóm đã hoàn thành các yêu cầu cơ bản của đề tài trong thời gian quy định.

6.2. Định hướng phát triển

Ngoài những chức năng đã phát triển được ở trên, trong tương lai nhóm chúng em sẽ tiếp tục tìm hiểu và tích hợp thêm một số chức năng như:

- Đặt cảm biến khoảng cách tại các trạm, từ đó việc tăng giảm tốc độ sẽ phụ thuộc vào cảm biến khoảng cách khi phát hiện đã gần về trạm hay chưa.
- Chuyển các bo mạch từ board trắng thành mạch in để tín hiệu, dòng điện chạy ổn định hơn, thuận tiện cho việc lắp đặt và sử dụng.

KẾT LUẬN

Nhờ vào sự giảng dạy, hướng dẫn của Thầy Phạm Văn Tiến và qua quá trình tìm hiểu, chúng em đã hiểu biết thêm được nhiều kiến thức để thiết kế một hệ thống nhúng hoàn chỉnh, từ các bước thiết kế, các công cụ thiết kế tổng quan, các công cụ mô phỏng, kết nối mạch phần cứng và phần mềm,…

Trong khoảng thời gian của kì học, chúng em cũng đã cố gắng tìm hiểu về các kiến thức của môn học. Tuy nhiên, do môn học có rất nhiều kiến thức, mang tính hệ thống trong khi kiến thức và kĩ năng của chúng em còn hạn chế, nhiều phần kiến thức chúng em chưa hiểu sâu nên sản phẩm làm ra chưa được hoàn thiện và còn nhiều điểm thiếu sót, cần phát triển thêm. Chúng em sẽ cố gắng nhiều hơn nữa để nắm bắt tốt các phần kiến thức còn yếu, và để phát triển hệ thống của mình tốt và ngày càng hoàn thiện hơn nữa.

Nhóm chúng em xin chân thành cảm ơn thầy đã nỗ lực hết mình giúp đỡ chúng em!

TÀI LIỆU THAM KHẢO

Interface L298N DC Motor Driver Module with ESP8266 NodeMCU

ESP8266 NodeMCU Asynchronous Web Server using Arduino IDE and ESPAsyncWebServer library

Auto Door

DC Motor Control With Blynk

ESP8266EX - Datasheet

Cảm biến khoảng cách HC-SR04