ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

Khoa Điện – Điện tử

Bộ môn Điện tử - Viễn thông

----------o0o----------

BÁO CÁO ĐỒ ÁN

CHỦ ĐỀ: HỆ THỐNG GIỮ XE BẰNG THẺ TỪ VÀ XỬ LÝ ẢNH

GVHD: QUẢN TRỌNG LÊ HOÀNG

Sinh viên thực hiện:	**MSSV**
Lê Hữu Khánh	**1812590**
Bùi Trung Kiên	**1812696**

TPHCM, ngày 29 tháng 5 năm 2022

MỤC LỤC

LỜI MỞ ĐẦU

Ngày nay khoa học công nghệ ngày càng phát triển, vi điều khiển ARM là vi điều khiển ngày càng thông dụng và hoàn thiện hơn, nhưng có thể nói sự xuất hiện của STM đã hỗ trợ cho con người rất nhiều trong lập trình và thiết kế, nhất là đối với những người bắt đầu tìm tòi về vi điều khiển mà không có quá nhiều kiến thức, hiểu biết sâu sắc về vật lý và điện tử. Phần cứng của thiết bị đã được tích hợp nhiều chức năng cơ bản và là mã nguồn mở. Chính vì những lý do như vậy nên STM hiện đang dần phổ biến và được phát triển ngày càng mạnh mẽ trên toàn thế giới.

Với những hiểu biết về các thiết bị điện tử và qua các kiến thức đã được học, nhóm chúng em đã quyết định thực hiện đề tài: **hệ thống giữ xe bằng xử lý ảnh và thẻ từ** với mục đích để tìm hiểu thêm về STM32, xử lý ảnh, làm quen với các thiết bị điện tử và nâng cao hiểu biết cho bản thân. Do kiến thức còn hạn hẹp, thêm vào đó đây là lần đầu chúng em thực hiện đồ án nên chắc chắn không tránh khỏi những thiếu sót hạn chế, vì thế chúng em rất mong có được sự góp ý và nhắc nhở từ thầy để có thể hoàn thiện đề tài của mình.

CHƯƠNG 1: TỔNG QUAN

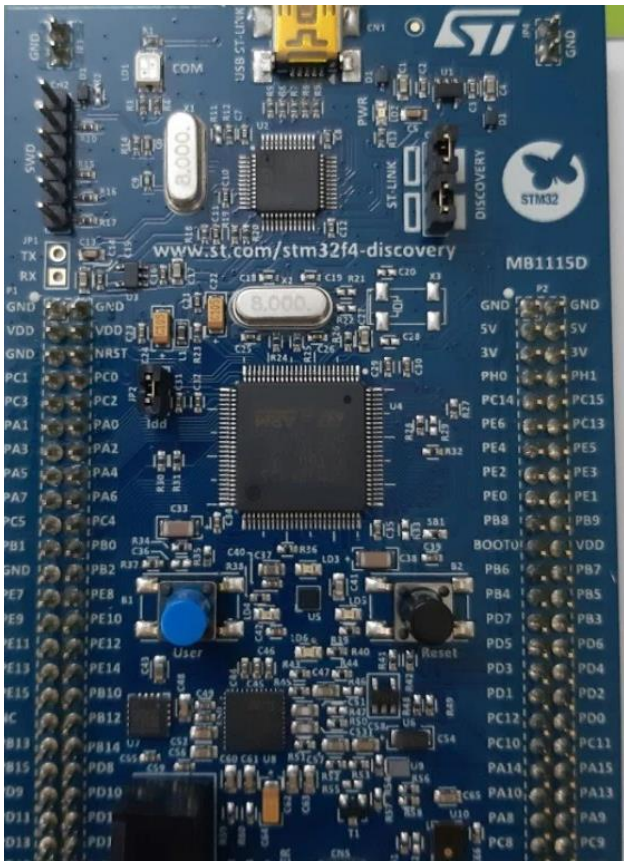
1. Giới thiệu chung về vi xử lý ARM Cortex và dòng vi điều khiển STM32:

ARM Cortex là vi xử lý được sử dụng phỏ biến nhất hiện nay trong lĩnh vực hệ thống nhúng. Tất cả các vi xử lý ARM Cortex đều có 32 bit địa chỉ bộ nhớ, điều này nghĩa là không gian bộ nhớ tối đa có thể lên đến 4GB. Các vi xử lý được ứng dụng rộng rãi trong các điện thoại di động, máy tính bảng và các thiết bị di động khác. ARM có kiến trức RISC, cho phép tiêu hao năng lượng thấp nên là một lựa chọn lý tưởng cho các hệ thống nhúng.

STM32 là chip của ST, dựa trên lõi ARM Cortex là thế hệ mới, thiết lập các tiêu chuẩn mới về hiệu suất, chi phí, ứng dụng cho các thiết bị cần tiêu thụ năng lượng thấp và đáp ứng yêu cầu thời gian thực khắt khe. Với những tính năng nổi bật như: tiêu thụ năng lượng cực thấp, hiệu suất cao và tích hợp một loạt ngoại vi (GPIO, I2C, SPI, ADC, …), nên các dòng STM đã được sử dụng rộng rãi: từ điện tử dân dụng (tivi, đầu máy, máy giặt, …), xe hơi đời mới, game, mobie, laptop, …

2. Giới thiệu chung về Board STM32F411 Discovery:

STM32F411 Discovery là board vi điều khiển dành cho người mới học lập trình nhúng hoặc dành cho những người muốn làm quen với lập trình vi điều khiển 32-bit dòng ARM. Board được tích hợp chip ARM Cortex-M4 cùng với bộ tính toán số thực (FPU), hoạt động với tần số rất cao 168 MHz, tỷ suất DMIPS/MHZ cao 1.25 giúp cho hệ thống có thể đạt được hiệu năng 210 DMIPS, board rất thích hợp cho các ứng dụng với yêu cầu tính toán xử lý nhanh, ví dụ như DSP, điều khiển robot… Với STM32F411 Discovery, người dùng sẽ không cần phải lo lắng và không cần phải bỏ tiền ra để mua mạch nạp đắt tiền như các loại board MCU thông thường. Điểm nổi bật nhất của board là có nhiều tính năng trong khi giá thành rất rẻ.



Hình 1: Module STM32F411VE

Các đặc điểm của Board:

- Tích hợp sẵn mạch nạp ST-LINK/V2
- Nguồn cung cấp cho board: qua USB bus hoặc từ nguồn điện ngoài 5V
- Cấp nguồn cho ứng dụng ngoài: 3V và 5V
- Audio DAC CS43L22 với driver loa lớp D tích hợp
- 8 đèn LED: LD1 (red/green) dùng cho giao tiếp USB, LD2 (red) báo hiệu nguồn 3.3 V đang bật, 4 đèn LED người dùng: LD3 (orange), LD4 (green), LD5 (red) và LD6 (blue), 2 đèn LED cho USB OTG: LD7 (green) VBus và LD8 (red) over-current.
- 2 nút bấm (user và reset)
- USB OTG FS với micro-AB connector

3. Giới thiệu về module thẻ từ RFID RC522:

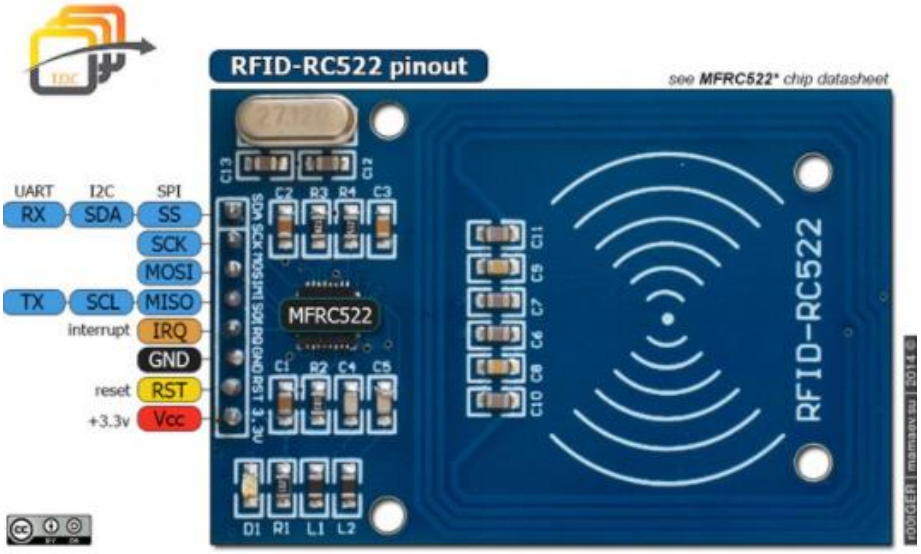
Module RFID RC522 sử dụng IC MFRC522 của NXP được sử dụng để đọc và ghi dữ liệu cho thẻ RFID NFC tần số 13.56Mhz, mạch có thiết kế nhỏ gọn được sử dụng rất phổ biến hiện nay với Arduino hoặc các loại Vi điều khiển khác trong các ứng dụng cần ghi, đọc thẻ RFID NFC. có thể đọc được các loại thẻ có kết nối không dây như NFC, thẻ từ (loại dùng làm thẻ giảm giá, thẻ xe bus, tàu điện ngầm...).

RFID (Radio Frequency Identification) là công nghệ nhận dạng đối tượng bằng sóng vô tuyến. Công nghệ này cho phép nhận biết các đối tượng thông qua hệ thống thu phát sóng radio, từ đó có thể giám sát, quản lý hoặc lưu vết từng đối tượng. Một hệ thống RFID thường bao gồm 2 thành phần chính là thẻ tag (chip RFID chứa thông tin) và đầu đọc (reader) đọc các thông tin trên chip.

Thông tin kỹ thuật:

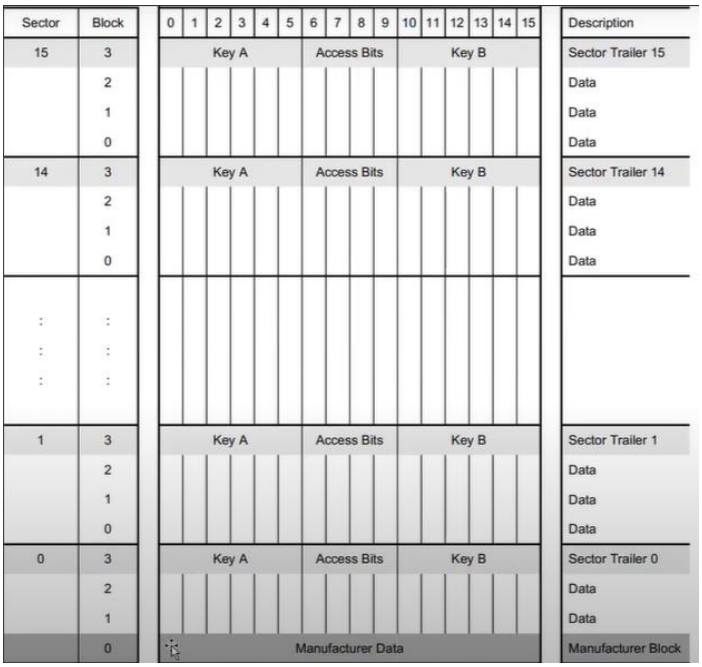
- Nguồn sử dụng: 3.3VDC

- Dòng điện:13~26mA
- Tần số hoạt động: 13.56Mhz
- Khoảng cách hoạt động: 0~60mm（mifare1 card）
- Chuẩn giao tiếp: SPI
- Tốc độ truyền dữ liệu: tối đa 10Mbit/s
- Các loại card RFID hỗ trợ: mifare1 S50, mifare1 S70, mifare UltraLight, mifare Pro, mifare Desfire
- Kích thước: 40mm × 60mm



Hình 2: Board module RFID RC522

Bên trong thẻ từ chia làm 16 sector, trên mỗi sector có 4 block và mỗi block gồm có 16byte. Sector 0 | Block 0 chứa ID của thẻ từ. Block thứ 3 của mỗi sector để lưu sector trailer (khi mã hóa muốn đọc được thẻ mà không có key thì không thể đọc được – tăng cường tính bảo mật cho thẻ).



4. Giới thiệu động cơ Servo SG90S:

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 PPW) với góc quay nằm trong khoảng bất kì từ 0^o - 180^o. Mỗi loại servo có kích thước, khối lượng và cấu tạo khác nhau.

Thông tin kỹ thuật:

- Nguồn sử dụng: 4.8V – 6V DC
- Tốc độ quay: 0.12 giây/60° (4.8V) , 0.1 giây/60° (6V)
- Mômen xoắn: 1.8kg/cm (4.8V) , 2.5kg/cm (6V)
- Góc quay: 180°
- Bánh răng: nhựa
- Kích thước: 22.5 * 11.8 * 30 mm
- Chiều dài dây điện: 175mm
- Trọng lượng: 9g

- Nhiệt độ hoạt động: 0°C ~ 55°C
- Dây cam: Xung
- Dây đỏ: Vcc (4.8V ~ 6V)
- Dây đen: GND / 0V

Hình 3: Động cơ servo SG90

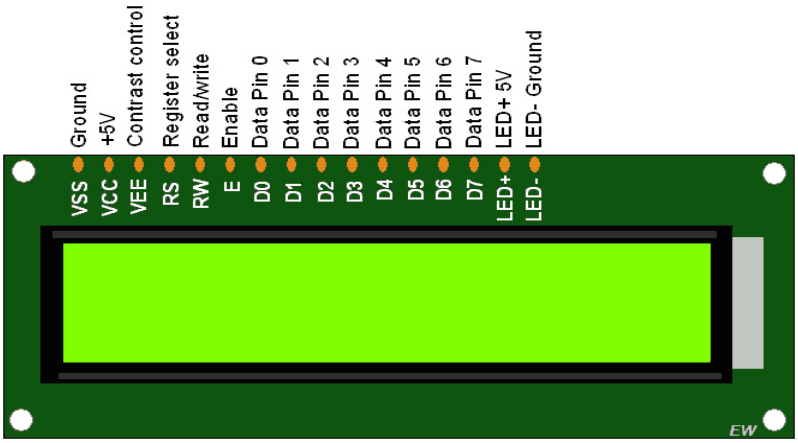
5. Giới thiệu về LCD 16x2 + I2C:

LCD 16×2 được sử dụng để hiển thị trạng thái hoặc các thông số.

- LCD 16×2 có 16 chân trong đó 8 chân dữ liệu (D0 – D7) và 3 chân điều khiển (RS, RW, EN).
- 5 chân còn lại dùng để cấp nguồn và đèn nền cho LCD 16×2.
- Các chân điều khiển giúp ta dễ dàng cấu hình LCD ở chế độ lệnh hoặc chế độ dữ liệu.
- Chúng còn giúp ta cấu hình ở chế độ đọc hoặc ghi.

- Điện áp hoạt động: 2.5-6V DC.
- Hỗ trợ màn hình: LCD1602,1604,2004 (driver HD44780).
- Giao tiếp: I2C.
- Địa chỉ mặc định: 0X27 (có thể điều chỉnh bằng ngắn mạch chân A0/A1/A2).
- Tích hợp Jump chốt để cung cấp đèn cho LCD hoặc ngắt.
- Tích hợp biến trở xoay điều chỉnh độ tương phản cho LCD.

LCD 16×2 có thể sử dụng ở chế độ 4 bit hoặc 8 bit tùy theo ứng dụng ta đang làm.



Hình 4: Cấu hình LCD 16x02

Module I2C LCD 16x2

Hình 5: Module I2C 16x2

LCD có quá nhiều nhiều chân gây khó khăn trong quá trình đấu nối và chiếm dụng nhiều chân trên vi điều khiển. **Module I2C LCD** ra đời và giải quyết vấn đề này

Thay vì phải mất 6 chân vi điều khiển để kết nối với LCD 16×2 (RS, EN, D7, D6, D5 và D4) thì module IC2 bạn chỉ cần tốn 2 chân (SCL, SDA) để kết nối.

Module I2C hỗ trợ các loại LCD sử dụng driver HD44780(LCD 16×2, LCD 20×4, …) và tương thích với hầu hết các vi điều khiển hiện nay.

Ưu điểm

- Tiết kiệm chân cho vi điều khiển.
- Dễ dàng kết nối với LCD.

Thông số kĩ thuật

- Điện áp hoạt động: 2.5-6V DC.
- Hỗ trợ màn hình: LCD1602,1604,2004 (driver HD44780).
- Giao tiếp: I2C.
- Địa chỉ mặc định: 0X27 (có thể điều chỉnh bằng ngắn mạch chân A0/A1/A2).
- Tích hợp Jump chốt để cung cấp đèn cho LCD hoặc ngắt.
- Tích hợp biến trở xoay điều chỉnh độ tương phản cho LCD.

Các lỗi thường gặp khi sử dụng I2C LCD

- Hiển thị một dãy ô vuông.
- Màn hình chỉ in ra một ký tự đầu.
- Màn hình nhấp nháy.

Hình 6: Module LCD 16x2 với đế I2C

6. Giới thiệu cảm biến vật cản hồng ngoại:

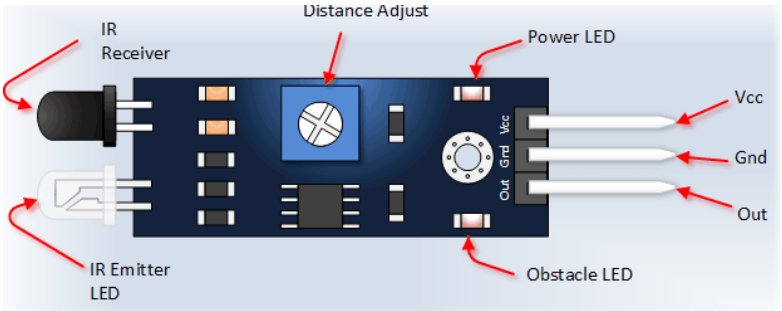
Cảm biến vật cản hồng ngoại có khả năng thích nghi với môi trường, có một cặp truyền và nhận tia hồng ngoại. Tia hồng ngoại phát ra một tần số nhất định, khi phát hiện hướng truyền có vật cản (mặt phản xạ), phản xạ vào đèn thu hồng ngoại, sau khi so sánh, đèn màu xanh sẽ sáng lên, đồng thời đầu cho tín hiệu số đầu ra (một tín hiệu bậc thấp).

Khoảng cách làm việc hiệu quả 2 ~ 5cm, điện áp làm việc là 3.3 V đến 5V. Độ nhạy sáng của cảm biến vật cản hồng ngoại được điều chỉnh bằng chiết áp, cảm biến dễ lắp ráp, dễ sử dụng,….

Có thể được sử dụng rộng rãi trong robot tránh chướng ngại vật, xe tránh chướng ngại vật và dò đường….

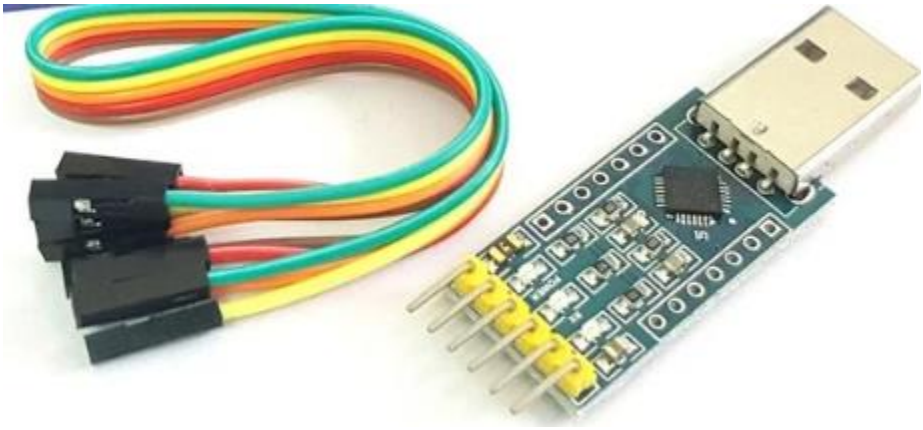
Thông tin kỹ thuật:

- Bộ so sánh sử dụng LM393, làm việc ổn định
- Điện áp làm việc: 3.3V – 5V DC.
- Khi bật nguồn, đèn báo nguồn màu đỏ sáng.
- Lỗ vít 3 mm, dễ dàng cố định, lắp đặt.
- Kích thước: 3.2cm * 1.4cm
- Các mô-đun đã được so sánh điện áp ngưỡng thông qua chiết áp, nếu sử dụng ở chế độ thông thường, xin vui lòng không tự ý điều chỉnh chiết áp.



Hình 7: Cấu hình cảm biến

7. Giới thiệu mạch chuyển USB UART CP2102A:



Mạch được sử dụng để chuyển giao tiếp từ USB sang UART TTL và ngược lại với tốc độ và độ ổn định cao.

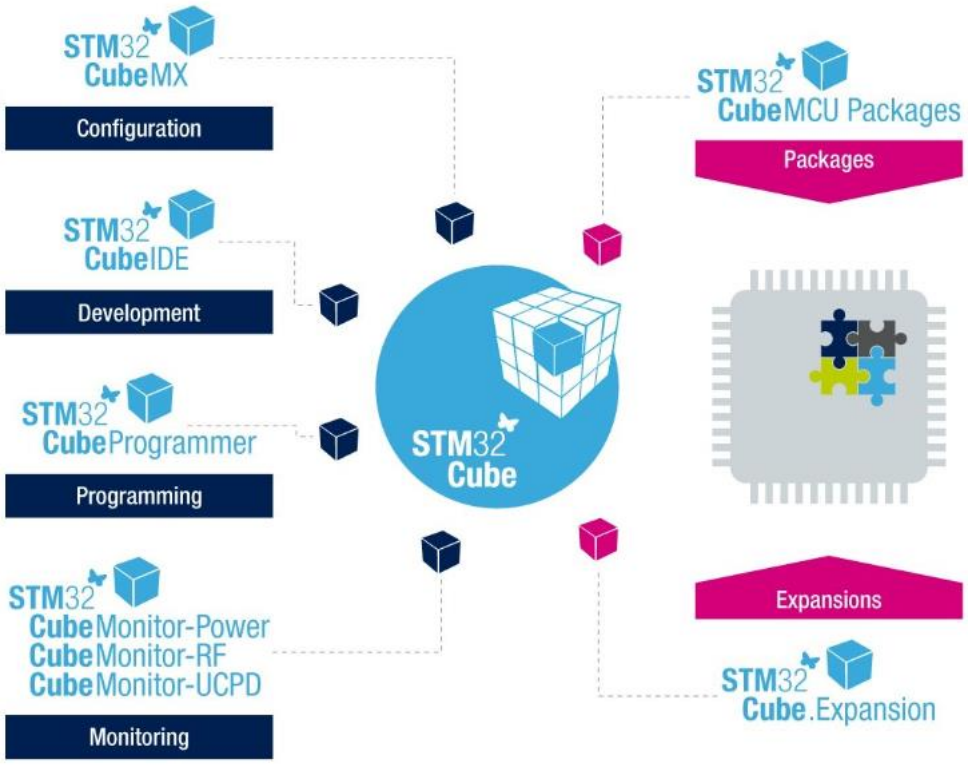
Mô tả chân:

- TXD: chân truyền dữ liệu UART TTL (3.3VDC), dùng kết nối đến chân nhận RX của các module sử dụng mức tín hiệu TTL 3.3~5VDC.
- RXD: chân nhận dữ liệu UART TTL (3.3VDC), dùng kết nối đến chân nhận TX của các module sử dụng mức tín hiệu TTL 3.3~5VDC.
- GND: chân mass hoặc nối đất.
- 5V: Chân cấp nguồn 5VDC từ cổng USB, tối đa 500mA.
- DTR: Chân tín hiệu DTR, thường được dùng để cấp tín hiệu Reset nạp chương trình cho mạch Arduino.
- 3.3V: Chân nguồn 3.3VDC (dòng cấp rất nhỏ tối đa 100mA), không sử dụng để cấp nguồn, thường chỉ sử dụng để thiết đặt mức tín hiệu Logic.

8. Giới thiệu về STM32CubeIDE và ngôn ngữ lập trình cho IDE:

STM32Cube là sự kết hợp của các công cụ phần mềm và các thứ viện phần mềm nhúng. Nó có đầy đủ các công cụ phần mềm hỗ trợ chạy trên máy tính giúp giải quyết tất cả những nhu cầu trong một chu trình phát triển dự án hoàn chỉnh. Các phần mềm nhúng được thiết kế để chạy trên các dòng vi điều khiển STM32 và

các vi xử lý tương ứng với nhiều chức năng khác nhau từ các driver cho từng ngoại vi của vi điều khiển đến những tính năng định hướng ứng dụng nâng cao.



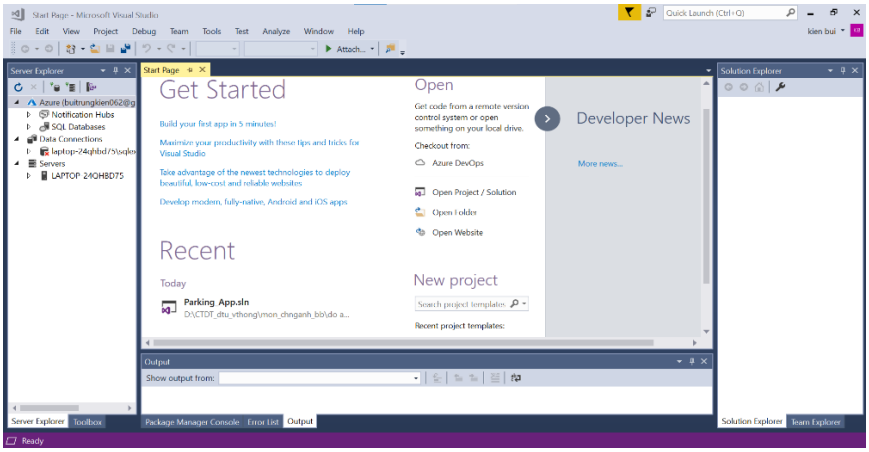
STM32CubeIDE, một môi trường phát triển tích hợp. Dựa trên các giải pháp nguồn mở như Eclipse, GNU C/C++ toolchain. IDE này bao gồm các tính năng báo cáo biên dịch chương trình và các tính năng gỡ lỗi nâng cao. Nó cũng được tích hợp thêm công cụ STM32CubeMX bên trong để tiện cho việc cấu hình và sinh code.

9. Giới thiệu về IDE và ngôn ngữ lập trình cho giao diện người dùng:

C# là một ngôn ngữ lập trình hướng đối tượng cho phép nhà phát triển xây dựng một loạt các ứng dụng an toàn và mạnh mẽ trên .NET Framework. Có thể sử dụng C# để tạo ra các ứng dụng truyền thống Windows, dịch vụ Web XML, thành phần phân phối ứng dụng dưới dạng clientserver, ứng dụng cơ sở dữ liệu…. .NET

Framework là một nền tảng phát triển phổ biến để xây dựng các ứng dụng cho Windows, Windows Store, Windows Phone, Windows Server và Windows Azure. Nền tảng .NET Framework bao gồm ngôn ngữ lập trình C#, Visual Basic, Common Language Runtime và lớp thư viện rộng lớn.

Visual Studio là một trong những công cụ hỗ trợ lập trình nổi tiếng của Microsoft. Visual Studio được viết bằng hai ngôn ngữ là C# và VB+. Đây là hai ngôn ngữ lập trình giúp người dùng tiếp cận việc xây dựng hệ thống một cách nhanh chóng và dễ dàng thông qua Visual Studio.



Từ khi ra đời đến nay, Visual Studio trải qua nhiều phiên bản khác nhau. Mục đích là giúp người dùng lựa chọn tương thích với cấu hình máy mà mình sử dụng. Phiên bản em đang sử dụng là Visual Studio 2017.

* Một số tính năng chính:

- Biên tập mã:

Giống như các IDE khác, Visual Studio gồm một trình soạn thảo mã hỗ trợ cú pháp và hoàn thiện mã bằng cách sử dụng intellisense không chỉ cho các hàm, biến mà còn sử dụng trong cấu trúc ngôn ngữ như truy vấn hoặc vòng điều khiển.

- Trình gỡ lỗi:

Visual Studio có trình gỡ lỗi vừa gỡ lỗi được cấp máy và gỡ lỗi cấp mã

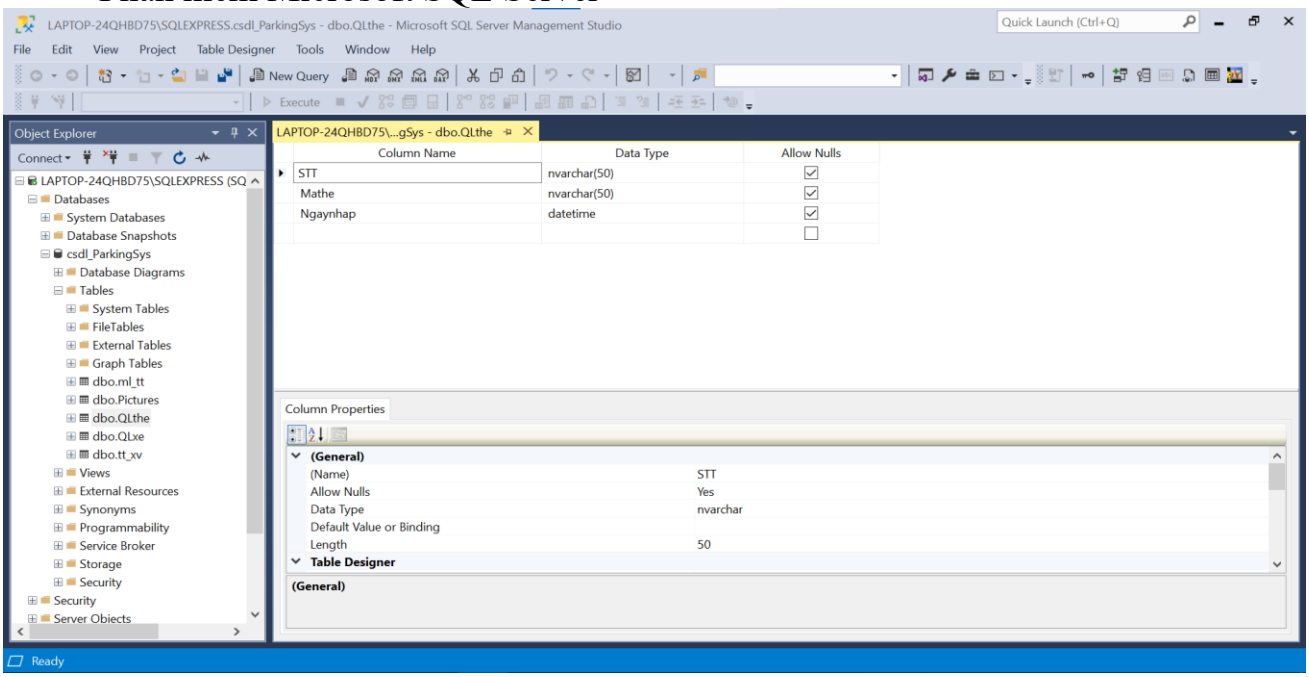
nguồn. Tính năng này hoạt động với cả hai mã quản lý giống như ngôn ngữ máy và có thể sử dụng gỡ lỗi các ứng dụng và viết bằng ngôn ngữ được hỗ trợ bởi Visual Studio.

- Thiết kế:

+ Windows Forms Designer:

Được sử dụng với mục đích xây dựng GUI sử dụng Windows Forms, được bố trí để xây dựng các nút điều khiển bên trong hoặc tạo các thiết bị hoặc cổng kết nối giả lập để kết nối với phần cứng khác. Điều khiển trình bày dữ liệu có thể liên kết với các nguồn dữ liệu như: cơ sở dữ liệu hoặc truy vấn.

Phần mềm Microsoft SQL Server



SQL Server là viết tắt của Structure Query Language Server, nó là một công cụ quản lý dữ liệu được sử dụng phổ biến ở nhiều lĩnh vực. SQL Server có khả năng hỗ trợ một số lượng lớn các quy trình xử lý giao dịch, ứng dụng doanh nghiệp và ứng dụng phân tích trong các công ty hoạt động trong lĩnh vực IT. Cũng giống như các hệ thống quản lý cơ sở dữ liệu qua hệ khác, SQL Server được xây dựng trên lớp SQL – là ngôn ngữ lập trình tiêu chuẩn hoá được quản trị viên cơ sở dữ

liệu (DBAs) và các chuyên gia IT sử dụng để quản lý cơ sở dữ liệu và truy vấn các dữ liệu nằm bên trong.

* Đặc điểm của SQL Server:

- Cho phép chèn, cập nhật, xóa các hàng trong một quan hệ.

- Tạo, sửa đổi, thêm, xóa các đối tượng trong cơ sở dữ liệu.

- Điều khiển việc truy vấn tới cơ sở dữ liệu và các đối tượng của cơ sở dữ liệu để đảm bảo tính bảo mật cơ sở dữ liệu.

- Đảm bảo tính nhất quán và sự ràng buộc cơ sở dữ liệu.

- SQL cung cấp tập lệnh phong phú cho việc truy vấn nên cần hiểu rõ cấu trúc cơ sở dữ liệu của mình.

CHƯƠNG 2: YÊU CẦU HỆ THỐNG

1. ***Tên (Name):***

- Hệ thống giữ xe bằng thẻ từ và xử lý ảnh.

2. ***Mục đích (Purpose):***

- Giảm khối lượng công việc cho người giữ xe.
- Dùng camera kết hợp với xử lí ảnh để lưu trữ hình ảnh về xe và biển số từ đó đưa những dữ liệu cần thiết vào thẻ từ, điều khiển động cơ servo.

3. ***Ngõ vào/ra (Input and output):***

- *Input:* data in RFID card, hình ảnh biển số xe từ camera
- *Output*:
    - Thông tin của người ra vào trên LCD (biển số xe).
    - Dữ liệu và thời gian thực của người ra vào.
    - Còi cảnh báo và tín hiệu điều khiển động cơ Servo

4. ***Trường hợp sử dụng (Use cases):***

- Người quản lý : xem ứng dụng được thiết kế bằng C# để quản lí xe vào ra.

5. ***Chức năng (Functions):***

- Đầu đọc thẻ từ được đặt ở cửa ra vào, lưu biển số từ camera gửi về.
- Thực hiện việc cảnh báo bằng buzzer nếu thẻ sai.
- Hiển thị lên LCD.
- Lưu trữ thông tin xe ra vào.
- Trực tiếp điều khiển động cơ servo đóng/mở bằng nút nhấn.
- Xóa thẻ bằng nút nhấn.

6. ***Hiệu năng (Performance):***

- Hệ thống hoạt động 24/7 (khi được cấp nguồn điện)

7. Giá thành sản xuất (Manufacturing cost): (Khoảng: 1.000.000 VNĐ)

8. Công suất (Power):

- Chính thức: nguồn 220VAC

9. Cài đặt (Installation):

- Thiết lập thông tin thẻ từ vào hệ thống
- Thiết lập xử lý ảnh

CHƯƠNG 3: ĐẶC TẢ HỆ THỐNG

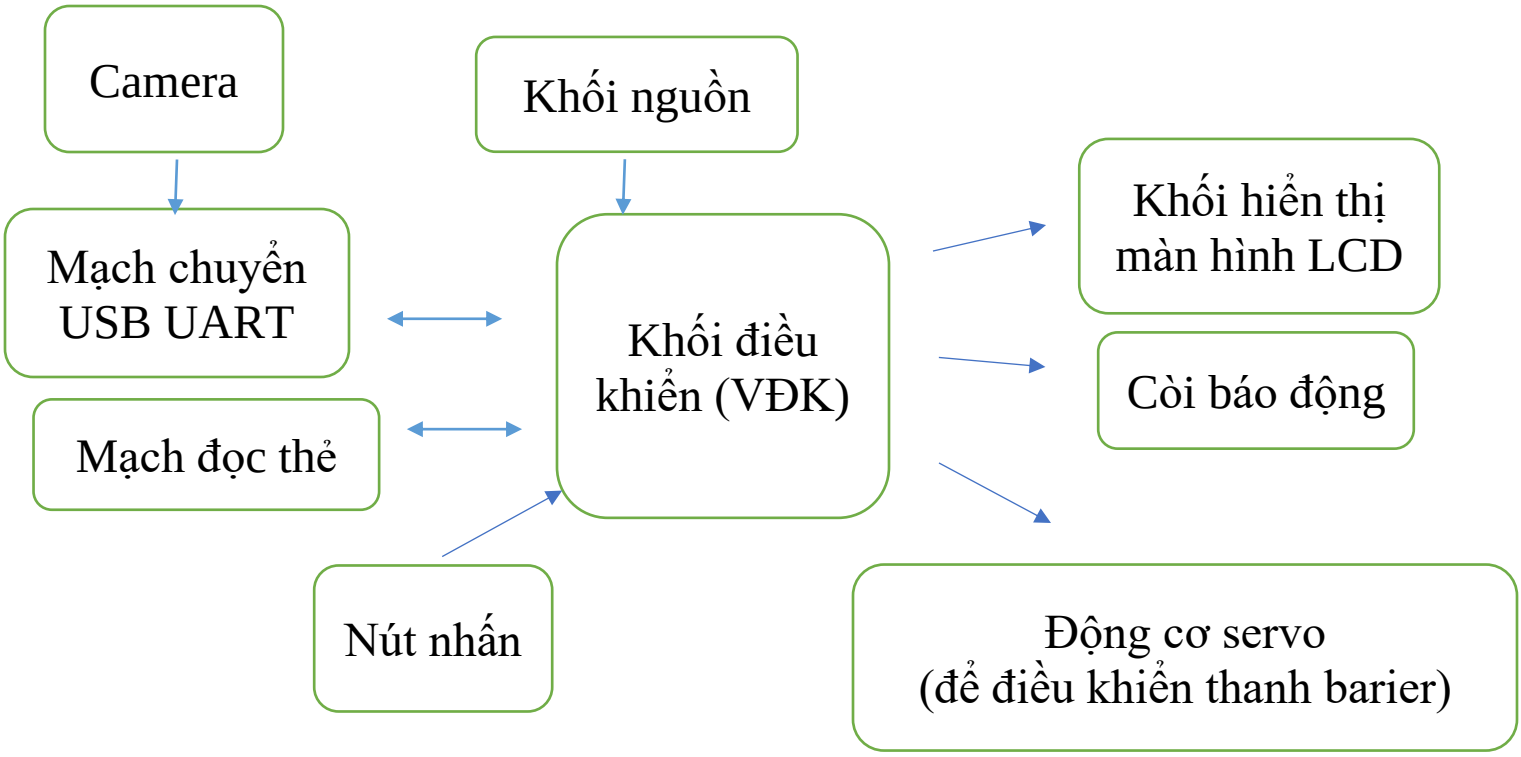
1. ***Nguyên lý hoạt động:***

- Mạch đọc thẻ từ RFID RC522 phát ra sóng điện từ ở tần số 13.56Mhz.
- Đầu vào:
    - Khi người dùng quẹt thẻ từ NFC RFID 13.56MHz vào mạch RC522 thứ nhất, thông qua giao tiếp UART với Vi xử lý, mạch RC522 sẽ đưa 1 tín hiệu yêu cầu gửi thông tin xe vào với vi xử lý.
    - Khi đó Vi xử lí sẽ yêu cầu Camera gửi về biển số xe vào lưu vào trong thẻ từ.
    - Nếu thẻ từ hợp lệ Vi xử lí sẽ gửi tín hiệu điều khiển thanh barier mở .
    - Khi xe đi qua quá cảm biến thì thanh barier tự động được đóng lại.
- Đầu ra:
    - Khi mạch RC522 thứ hai nhận được thẻ từ, mạch RC522 sẽ yêu cầu vi xử lý tiến hành so sánh dữ liệu trong thẻ với dữ liệu đọc được từ camera từ xe đi ra.
    - Nếu so sánh đúng thì thẻ từ sẽ được xóa hết dữ liệu, thanh barier sẽ được mở và khi xe đi quá cảm biến thì thanh barier cũng tự động đóng lại.
    - Tất cả thông tin của xe vào ra đều được hiển thị trên phần mềm được viết bằng C#.
- Có 3 nút nhấn trong đó có 2 nút nhấn riêng biệt để có thể độc lập điều khiển barier, 1 nút nhấn còn lại để xóa thông tin trong thẻ.
- LCD để hiển thị biển số xe vào/ra.

2. ***Môi trường hoạt động (External Enviroment):***

- Hệ thống được ở nơi người giữ xe làm việc.

3. ***Sơ đồ khối hệ thống (System connectivity):***



4. ***Mô tả các khối chính (Module descriptions)***

- Khối nguồn : Cung cấp nguồn điện 5 VDC.
- Mạch đọc thẻ : Đưa về vi xử lý mã thẻ từ và thông tin được lưu trong thẻ.
- Vi điều khiển:
    - Đọc, ghi và xử lý tín hiệu từ mạch đọc thẻ.
    - Giao tiếp I2C với LCD để hiển thị thông tin.
    - Kích hoạt còi báo động.
    - Điều khiển động cơ servo
- Camera: đưa dữ liệu về xe vào/ ra về vi xử lí.

5. ***Phân chia phần cứng và phần mềm (divide hardware/software):***

- Phần cứng :
    - Mạch đọc thẻ từ RFID RC522
    - Vi xử lý: STM32F411VE Discovery
    - Cảm biến vật cản hồng ngoại FM52
    - LCD 16x2 với đế i2c
    - Buzzer
    - Button
    - Động cơ servo
    - Nguồn 5V DC – 2A
    - Điện trở và tụ điện (nhằm chống rung nút nhấn)
    - Mạch chuyển USB UART CP2102
- Phần mềm :
    - Code C trên stm32CubeIDE
    - Code giao diện C# Visual Studio Code

CHƯƠNG 4: THIẾT KẾ, THI CÔNG PHẦN CỨNG

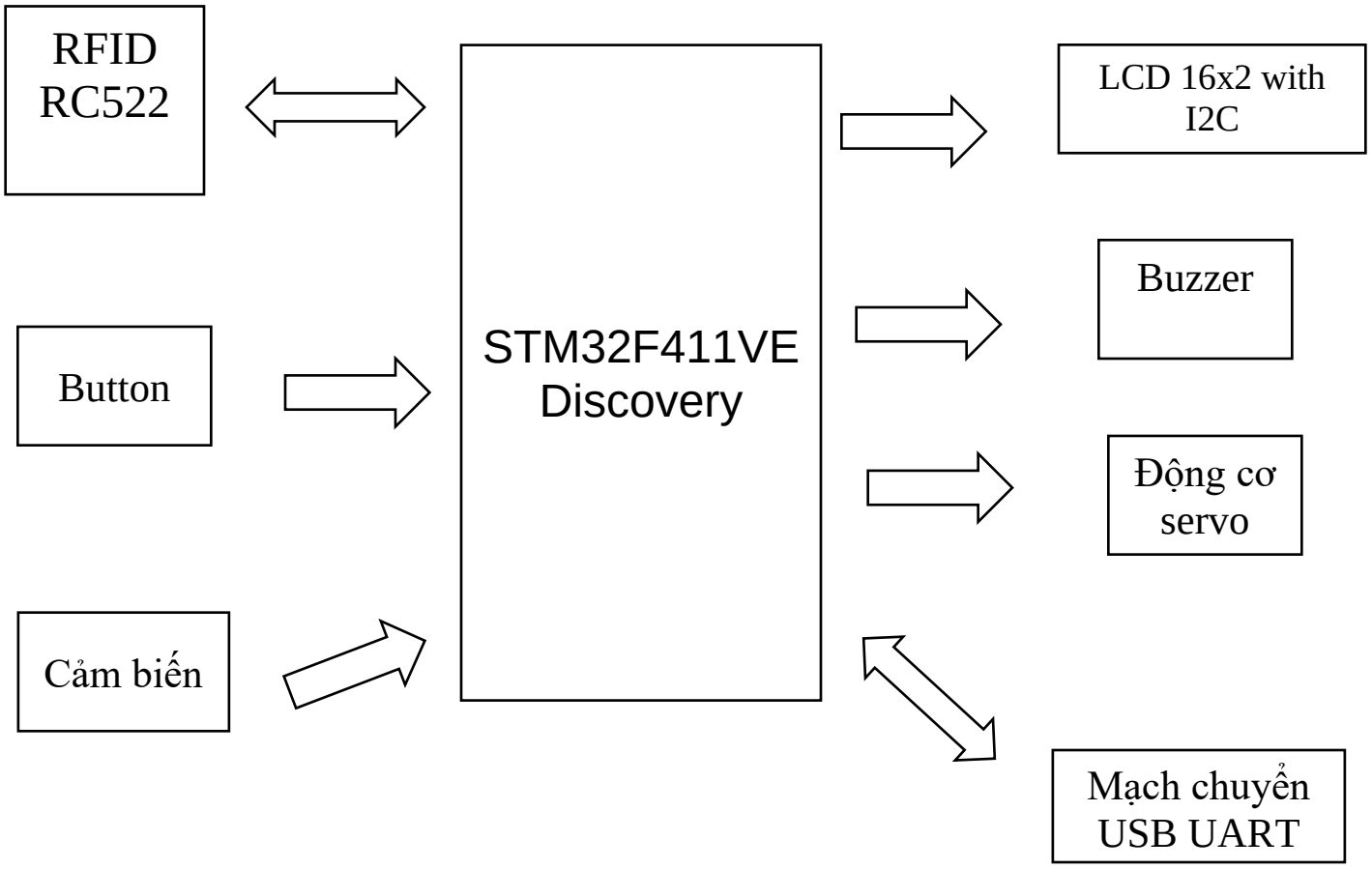
1. Danh sách các linh kiện cần thiết cho dự án:

Tên linh kiện	Số lượng
STM32F411VE Discovery	1
RFID RC-522	2
Cảm biến vật cản hồng ngoại FM52	2
Button	3
LCD 16x2 + I2C	1
Buzzer	1
Nguồn 5V DC – 2A	1
Điện trở 10K	3
Điện trở 470	3
Tụ điện 100nF	3
Động cơ servo	2
Mạch chuyển USB UART CP2102	1

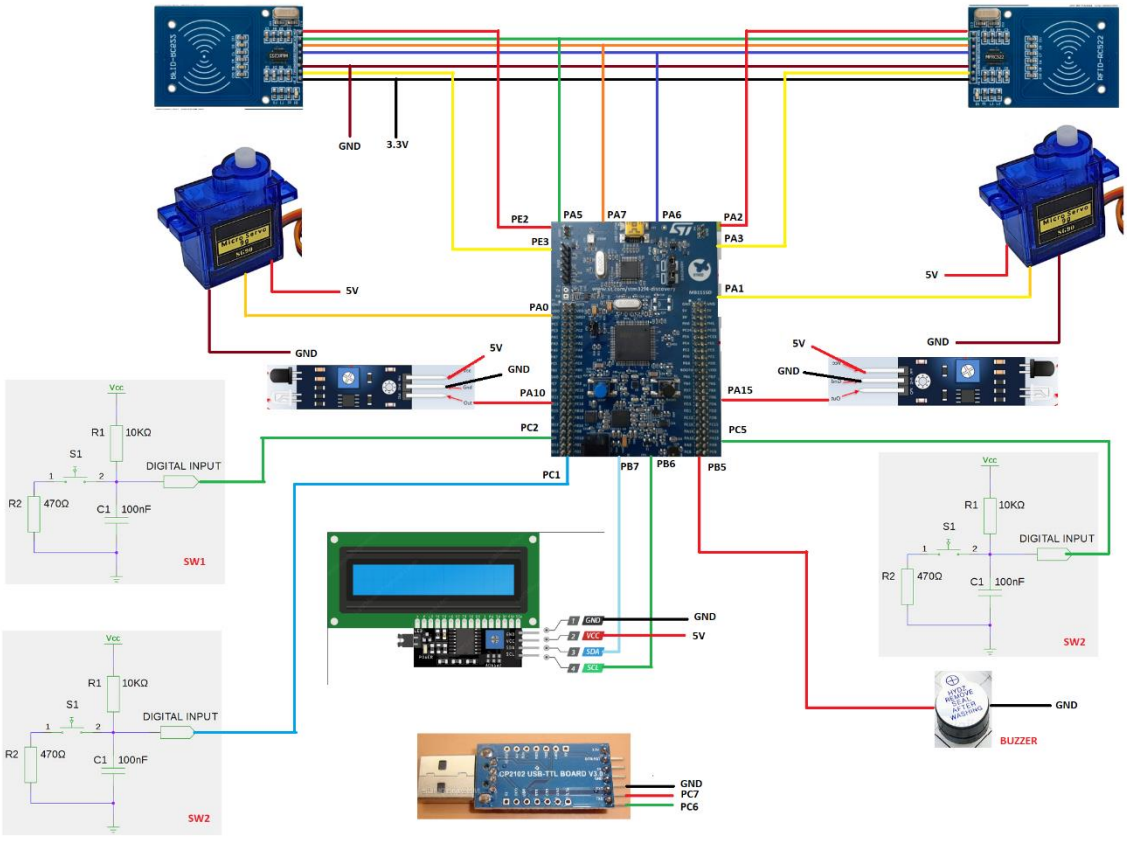
2. Phần cứng của hệ thống:

- Vi điều khiển: STM32F411VE Discovery
- Ngoại vi:
    - RFID RC-522 : giao tiếp SPI với vi xử lý
    - Cảm biến vật cản hồng ngoại FM52, Buzzer, Button và servo giao tiếp i/o Port
    - LCD 16x2 và I2C: giao tiếp I2C với vi xử lý
- Nguồn:
    - Nguồn 5VDC: cấp cho vi điều khiển

3. Sơ đồ khối phần cứng:



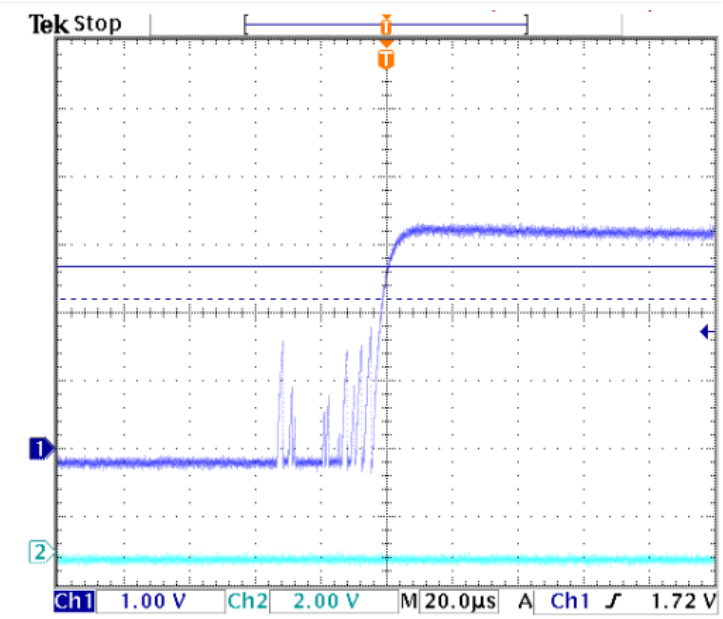
4. Sơ đồ mạch:



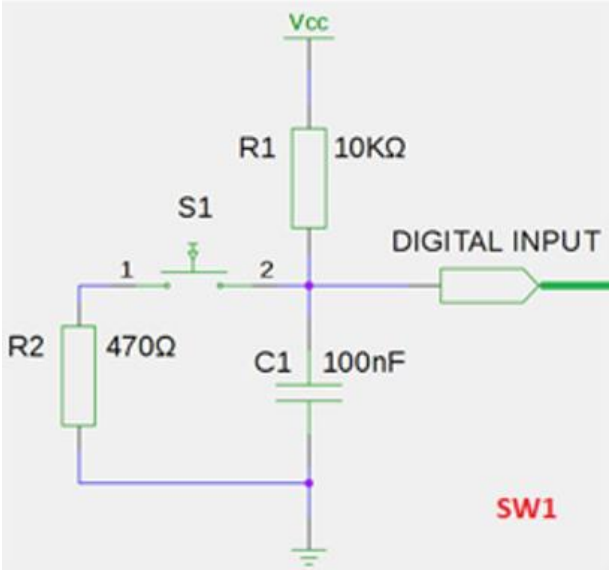
- Mạch chống rung nút nhấn bằng phần cứng:

Khi nhấn nút, ban đầu nó sẽ tiếp xúc với những vùng kim loại khác (có thể gọi là tiếp xúc chưa hoàn toàn), nhưng chỉ trong một khoảng thời gian cực ngắn, cỡ vài micro giây. Quá trình diễn ra dần dần cho đến khi tiếp xúc hoàn toàn.

Do phần cứng phản xạ cực nhanh với các tiếp xúc, nên khi trong quá trình nhấn nút như trên thì phần cứng nó hiểu rằng bạn nhấn công tắc nhiều lần. Đây chính là hiện tượng rung phím bấm.



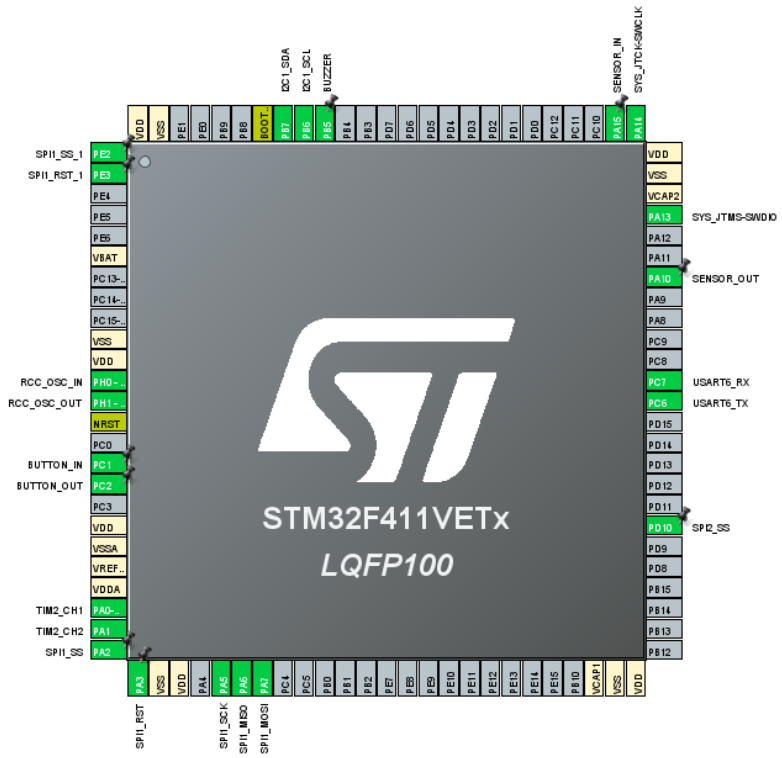
Để khắc phục hiện tượng trên, có 2 cách giải quyết đó là: chống rung bằng phần mềm và chống rung bằng phần cứng. Ở đây, nhóm đã lựa chọn chống rung bằng phần cứng bằng cách sử dụng mạch dưới đây:



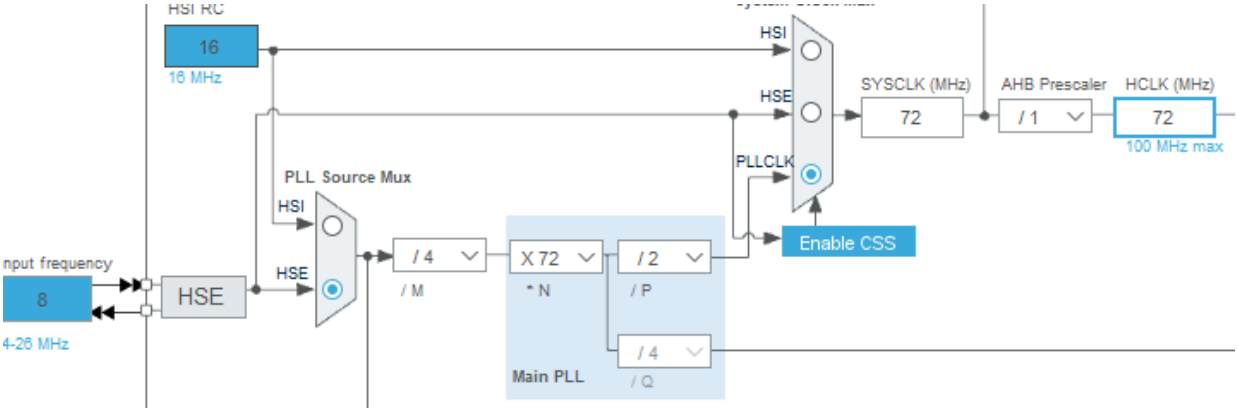
CHƯƠNG 5: THIẾT KẾ, LẬP TRÌNH PHẦN MỀM

1. Lập trình trên STM32CubeIDE:

1.1. Cấu hình trên STM32CubeIDE:



- Sử dụng thạch anh nội: 8MHz
- Clock của hệ thống: 72MHz



1.2. Các Module để giao tiếp với ngoại vi:

- Các module có sẵn của STm32CubeIDE:
    - Module gpio: cấu hình và khởi tạo giao thức kết nối I/O port, ngắt ngoài (EXTI) (dành cho Button, Buzzer, chân SS và RST của RC522, SG90, cảm biến)
    - Module i2c: cấu hình và khởi tạo giao thức kết nối I2C (dành cho LCD16x02 + i2c)
    - Module spi: cấu hình và khởi tạo giao thức kết nối SPI (dành cho các chân SCK, MISO, MOSI của RC522)
    - Module tim: cấu hình các ngắt (phục vụ cho hàm delay_ms và điều khiển động cơ)
    - Module usart: cấu hình và khởi tạo giao thức kết nối UART (dành cho mạch chuyển USB UART)
    - Một số 1 module dùng cho việc xử lí dữ liệu: stdio.h, math.h, string.h
- Các module thêm vào:
    - Module mfrc522: bao gồm các hàm dùng để kết nối và giao tiếp với RC522
    - Module i2c-lcd: bao gồm các hàm dùng để kết nối và giao tiếp với LCD

1.3. Module main:

- Các hàm con:
    - Hàm delay_ms: tạo hàm delay theo đơn vị ms

```
void delay_ms(uint16_t ms)
{
    __HAL_TIM_SET_COUNTER(&htim11, 0);
    while(__HAL_TIM_GET_COUNTER(&htim11) < (ms * 10) );
}
```

Như cấu hình bên trên: Clock nội = 72MHz

⇨ Tần số vào của Timer = 72MHz / Prescaler (7200) = 10 000

⇨ 1 Tick = 1/10 000

⇨ Thời gian đếm hết (ms*10) tick = ms*10/10 000 = ms*10^{-3} (s) = ms (ms)

- Hàm Buzzer_Announ: tạo hàm bật rồi tắt loa trong khoảng thời gian đơn vị là ms

```
void Buzzer_Announ(uint32_t time)
{
    HAL_GPIO_WritePin(BUZZER_GPIO_Port, BUZZER_Pin, GPIO_PIN_SET);
    delay_ms(time);
    HAL_GPIO_WritePin(BUZZER_GPIO_Port, BUZZER_Pin, GPIO_PIN_RESET);
}
```

- Hàm display: hiển thị biển số xe lên LCD

```
void display(uint8_t n) //IN --> n = 4; OUT --> n = 5;
{
    uint8_t x;
    if((char)numData[9] == '-')
        x = 9;
    else
        x = 10;
    for(int j = 0; j < x; j++)
    {
        lcd_put_cur((n - 4), j + n);
        lcd_send_data((char)numData[j]);
    }
}
```

Nếu n = 4 ta sẽ ghi data lên dòng IN: nhằm thể hiển biển số xe vào còn nếu n = 5 ta sẽ ghi data lên dòng OUT: nhằm thể hiện biển số xe ra.

- Hàm CompareData: để so sánh 2 chuỗi dữ liệu (nếu return MI_OK tức là hai chuỗi này giống nhau còn return MI_ERR là 2 chuỗi này khác nhau)

```
uint8_t CompareData()
{
    uint8_t x;
    if((char)numData[9] == '-')
        x = 9;
    else
        x = 10;
    for (int i = 0; i < x; i++)
    {
        if (data[i] != numData[i])
        {
            return MI_ERR;
        }
    }
    return MI_OK;
}
```

- Hàm clear_numData: đưa toàn bộ chuỗi dữ liệu về chuỗi 0 để ghi vào

Block của thẻ từ.

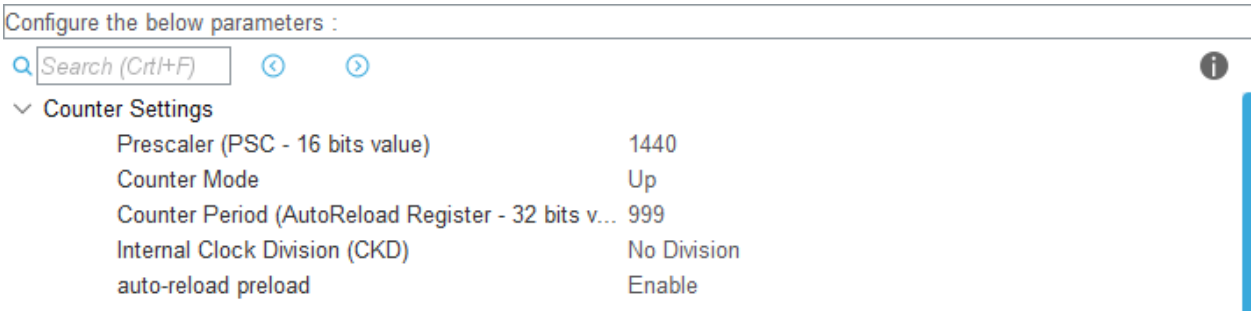
```
void clear_numData()
{
    for(int i = 0; i < 10; i++)
        numData[i] = 0;
}
```

- Hàm Check: kiểm tra và cho phép đọc và ghi vào thẻ.

```
uint8_t Check(status_t string)
{
    switch(string)
    {
        case IN:
        {
            if (MFRC522_Request(PICC_REQALL, serNum) != MI_OK)                      return MI_ERR; //Yeu cau tim Card nam trong vung hoat dong.
            if (MFRC522_Anticoll(serNum) != MI_OK)                                 return MI_ERR; //Chong trung the
            if (MFRC522_Select(serNum) != MI_OK)                                   return MI_ERR; //Lua ch?n Card de giao tiep_doc data tu card
            if (MFRC522_AuthState(PICC_AUTHENT1A, 4, defaultKeyA, serNum) != MI_OK) return MI_ERR; //Lenh nay de xac thuc cho phep giao tiep voi Block 4
            if (MFRC522_Read(4, data) != MI_OK)                                    return MI_ERR; //Doc va ghi gia tri vao mang data
            return MI_OK;
        }
        case OUT:
        {
            if (MFRC522_Out_Request(PICC_REQALL, serNum) != MI_OK)                      return MI_ERR;
            if (MFRC522_Out_Anticoll(serNum) != MI_OK)                                 return MI_ERR;
            if (MFRC522_Select_Out(serNum) != MI_OK)                                   return MI_ERR;
            if (MFRC522_AuthState_Out(PICC_AUTHENT1A, 4, defaultKeyA, serNum) != MI_OK) return MI_ERR;
            if (MFRC522_Read_Out(4, data) != MI_OK)                                    return MI_ERR;
            return MI_OK;
        }
        return MI_ERR;
    }
}
```

1.4. Điều khiển Servo:

PWM: điều chế độ rộng xung hay điều chế thời gian xung là 1 trong những phương pháp để điều khiển Servo. Để điều khiển quay, ta phải cấp xung PWM có tần số là 50Hz. Đối với SG90 thì độ rộng xung 1ms để quay góc 0º, 1,5ms để quay 90º và 2ms để quay 180º.



Vì tần số khi đi vào Timer2 bây giờ đang là 72MHz nên khi cài đặt thông số Prescaler = 1440 thì tần số lúc này là: 72 Mhz/1440 = 50 KHz => timer 2 phát 1 xung trong 1/50 KHz = 0.00002s. Trên thực tế ta muốn tạo f = 50 Hz, là cần

phát 1 xung trong 1/50 Hz = 0.02s tức là timer2 phát 0.02/0.00002 = 1000 xung thì mới tạo được tần số f = 50Hz. Khi cài đặt Counter Period = 999 vì timer2 đếm từ 0 đến 999 thì sẽ tràn.

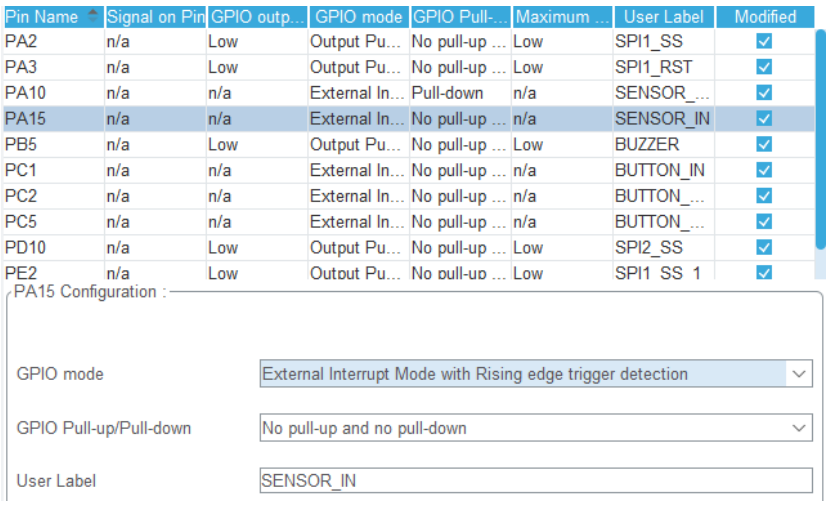
Để thay đổi được góc quay của Servo, ta sẽ đưa giá trị từ 0 - 999 vào thanh ghi CCRx của timer2 bằng câu lệnh: `htim2.Instance->CCRx = X;` .

```
#define  OpenOut()          htim2.Instance->CCR1 = 118;
#define  OpenIn()           htim2.Instance->CCR2 = 30;
#define  CloseOut()         htim2.Instance->CCR1 = 65;
#define  CloseIn()          htim2.Instance->CCR2 = 80;
```

1.5. Sử dụng ngắt: khi có sự kiện ngắt xảy ra, chương trình sẽ nhảy vào hàm HAL_GPIO_EXTI_Callback để thực thi

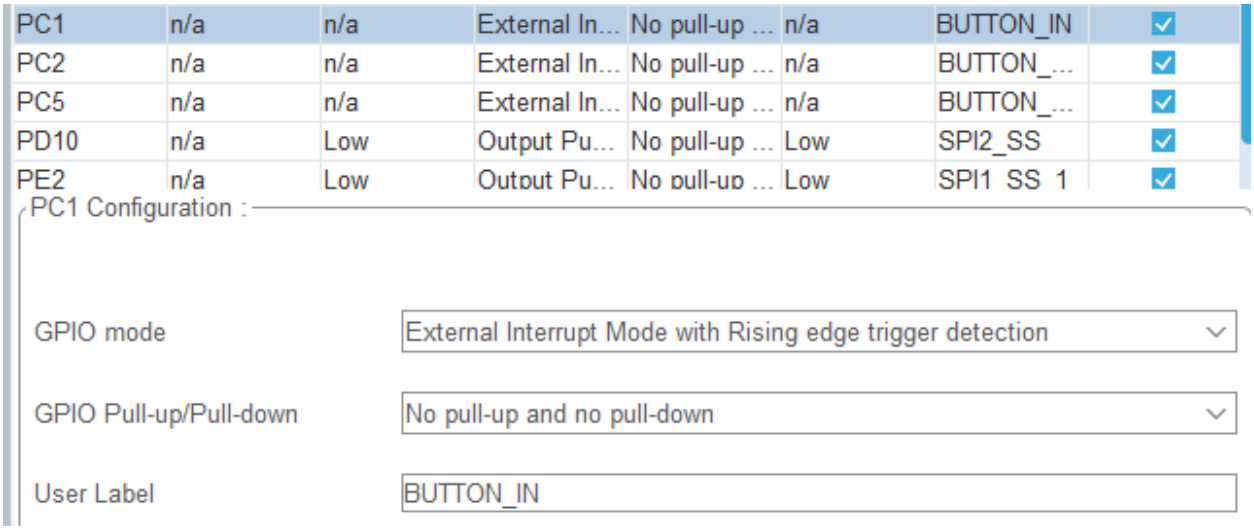
```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    switch (GPIO_Pin)
    {
    case BUTTON_RST_Pin:
    {
        clear = 1;
        break;
    }

    case BUTTON_OUT_Pin: {
        OpenOut();
        break;
    }
    case BUTTON_IN_Pin: {
        OpenIn();
        break;
    }
    case SENSOR_OUT_Pin: {
        CloseOut();
        break;
    }
    case SENSOR_IN_Pin: {
        CloseIn();
        break;
    }
    }
}
```

- Sử dụng ngắt với cảm biến vật cản:



Vì khi có vật cản xuất hiện trên Cảm biến thì chân Out sẽ đi ra mức 0, GPIO mode để Rising nghĩa là khi chân PA15 chuyển từ 0 xuống 1 sẽ xảy ra ngắt. Khi đó chương trình sẽ nhảy vào trình phục vụ ngắt và thực thi lệnh trong case: SENSOR_OUT_Pin.

- <u>Sử dụng ngắt với nút nhấn:</u>

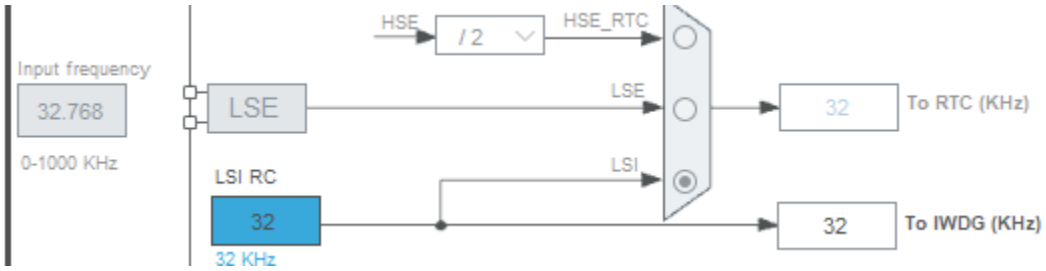


Vì khi có vật cản xuất hiện trên Cảm biến thì chân Out sẽ đi ra mức 0, GPIO mode để Rising nghĩa là khi chân PA15 chuyển từ 0 xuống 1 sẽ xảy ra ngắt. Tương tự ngắt với cảm biến, khi nút nhấn được nhấn thì ngay lập tức sẽ xảy ra ngắt và chương tình sẽ nhảy vào trình phục vụ ngắt.

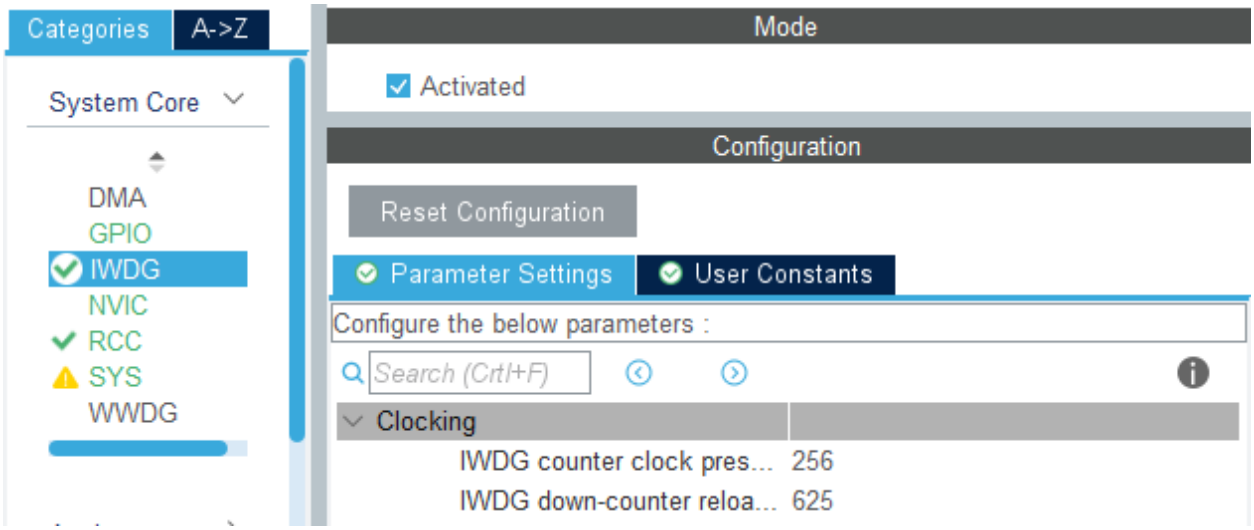
1.6. Watchdog Timer:

Đây là 1 loại Timer rất hay được sử dụng trong thực tế, chức năng chủ yếu là kiểm tra vi điều khiển có bị treo hay hoạt động sai không. Từ đó đưa ra lệnh Reset và làm cho chương trình chạy lại từ đầu.

Trong STM32 Watchdog Timer có 2 loại là: Independent Watchdog Timer và Window Watchdog Timer, ở đây ta sử dụng IWDG. Đây là bộ timer sử dụng nguồn xung LSI vì vậy chúng có thể hoạt động ngay cả khi nguồn clock của chương trình chính không hoạt động. Khi ngắt IWDG xảy ra, MCU sẽ lập tức bị reset mà không cần làm gì cả.



Ở đây chương trình có LSI = 32KHz.



Thời gian để reset chương trình nếu không reload lại giá trị của bộ đếm là: T_{reset} = 256 * 625 / 32KHz = 5s.

Vì chương trình sử dụng khá nhiều hàm While nên dễ bị treo chương trình nên ta sẽ đặt `HAL_IWDG_Init(&hiwdg);` vào sau hàm While ở main. Sau mỗi lần gọi lại câu lệnh này, chương trình sẽ đặt lại giá trị đếm xuống. Nếu sau khi gọi câu lênh này, chương trình chạy quá 5s mà thì chương trình sẽ bị reset ngay lạp tức.

```
  while (1)
  {
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
      //Check UID card
      HAL_IWDG_Init(&hiwdg);
      if (Check(IN) == MI_OK)
      {
```

1.7. Chương trình main:

Trong vòng lặp While(1) của chương trình main ta chia 3 nhiệm vụ: kiểm tra thẻ vào, kiểm tra thẻ ra và xóa thẻ.

Chương trình kiểm tra thẻ vào/ra:

```
HAL_IWDG_Init(&hiwdg);
if (Check(IN) == MI_OK)
{
    if(data[0] != 48)
    {
        lcd_put_cur(0,4);
        lcd_send_string("Error");
        Buzzer_Announ(500);
        lcd_put_cur(0,0);
        lcd_send_string("IN:                ");
        continue;
    }
    else
    {
        HAL_UART_Transmit(&huart6, (uint8_t*) "I", 1, 300);
        while(HAL_UART_Receive(&huart6,(uint8_t*)  &c, 1, 300) != HAL_OK)
        {
            if(c != "D" && c != "E")        continue;
            else                            break;
        }
        switch(c)
        {
            case 'E':
            {
                Buzzer_Announ(500);
                continue;
            }
```

```
            case 'D':
            {
                while(HAL_UART_Receive(&huart6, (uint8_t*)&numData, 10, 1000) != HAL_OK);
                MFRC522_Write(4, (uint8_t*) &numData);
                display(4);
                delay_ms(1000);
                lcd_put_cur(0,0);
                lcd_send_string("IN:                ");
                clear_numData();
                OpenIn();
                delay_ms(700);
                continue;
            }
            continue;
        }
    }
```

Chương trình đầu tiên sẽ kiểm tra thẻ vào bằng hàm Check(IN) nếu có thì nó sẽ kiểm tra nội dung trong thẻ. Vì trong hàm Check(IN) có câu lệnh đọc dữ liệu block 4 từ thẻ từ vào mảng data, nên nếu có dữ liệu thì data[0] khác 0 còn nếu không có dữ liệu thì data[0] = 0. Nếu thẻ đã có dữ liệu, trên màn hình LCD tại hàng IN: sẽ hiển thị chữ ERROR và buzzer bật cảnh báo trong 500ms, sau đó tiếp tục kiểm tra tiếp.

Nếu trong thẻ không có dữ liệu thì Vi xử lý sẽ thông qua giao tiếp UART gửi kí tự “I” lên máy tính để tiến hành quá trình xử lý và gửi về biển số xe. Nếu máy tính gửi lại kí tự “E” là đọc không thành công và buzzer bật cảnh báo trong 500ms. Còn nếu nhận lại được kí tự “D”, chương trinh sẽ chờ nhận các ký tự của biển số xe sau đó hiện lên LCD dòng IN:.

Chương trình xóa thẻ:

```
while(clear)
{
    lcd_clear();
    lcd_put_cur(0, 0);
    lcd_send_string("-----CLEAR-----");
    if (Check(IN) == MI_OK)
    {
        while(MFRC522_Write(4, (uint8_t*) "0000000000") != MI_OK);
        lcd_clear();
        lcd_put_cur(0,0);
        lcd_send_string("IN: ");
        lcd_put_cur(1, 0);
        lcd_send_string("OUT: ");
        clear = 0;
    }
}
```

Sau khi nhận được ngắt từ nút nhấn, chương trình sẽ nhảy vào vòng lặp while(clear) có nhiệm vụ xóa nội dung trong thẻ. Chương trình sẽ hiển thị lên LCD chuỗi kí tự "----CLEAR----" và chờ khi nào có thẻ cần xóa thì sẽ thoát ra khỏi vòng lặp này và trở về vòng lặp chính.

2. Lập trình trên giao diện bằng C# trên Visual Studio Code:

Giải thuật xử lý ảnh và hệ thống quản lý dữ liệu

B1: Khởi động ứng dụng, thực hiện thao tác kết nối đầu đọc RFID trên ứng dụng.

B2: Thực hiện việc quẹt thẻ ở ngõ vào, thông qua chương trình C được lập trình trong bộ vi xử lý STM32F411VE Discovery tín hiệu sẽ truyền về máy tính thông qua chuẩn giao tiếp UART tiếp tục xử lý. Chương trình C# được lập trình để tạo ứng dụng quản lý sẽ ra lệnh Webcam chụp ảnh. Hình ảnh này sẽ được lưu trữ và xử lý trong chương trình C#.

B3: Sau khi có hình ảnh từ Webcam, chương trình tiến hành quá trình tách biển số xe ra khỏi hình. Bằng việc sử dụng tệp tin lưu trữ dữ liệu huấn luyện nhận dạng biển số xe tỉ lệ 33x25 (output-hv-33-x25.xml), chương trình sẽ tiến hành xử lý và trả biển số xe với kích thước ảnh 400x400 pixel.

B4: Từ biển số xe ta tiến hành lấy đường viền ảnh. Vì biển số xe màu trắng còn ký tự là màu đen, nên khi lấy đường viền ta sẽ dễ dàng có được đường viền bao quanh ký tự để phục vụ cho bước tiếp theo là cắt ký tự.

B5: Từ các đường viền này ta sẽ cắt riêng lẻ từng ký tự biển số xe. Do đề tài này em nhắm đến nhận dạng biển số xe ô tô nên sẽ có 9 ký tự.

B6: Với những mẫu ký tự cắt ra, đưa chúng vào ký tự nhận dạng có sẵn

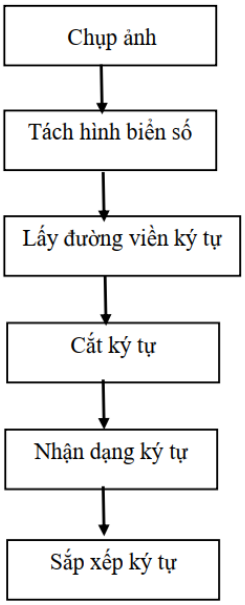
trong thư viện Emgu.CV. Từ đó ta sẽ chuyển từ dạng tương tự (hình ảnh) sang dạng số (mã ASCII).

B7: Mặc dù nhận dạng được ký tự nhưng thứ tự rất lộn xộn. Do đó cần sắp xếp lại chúng dựa vào vị trí cắt ở B5. Và cuối cùng là những chữ số, chữ cái được sắp xếp hợp lý như hình ảnh biển số đã chụp ban đầu.

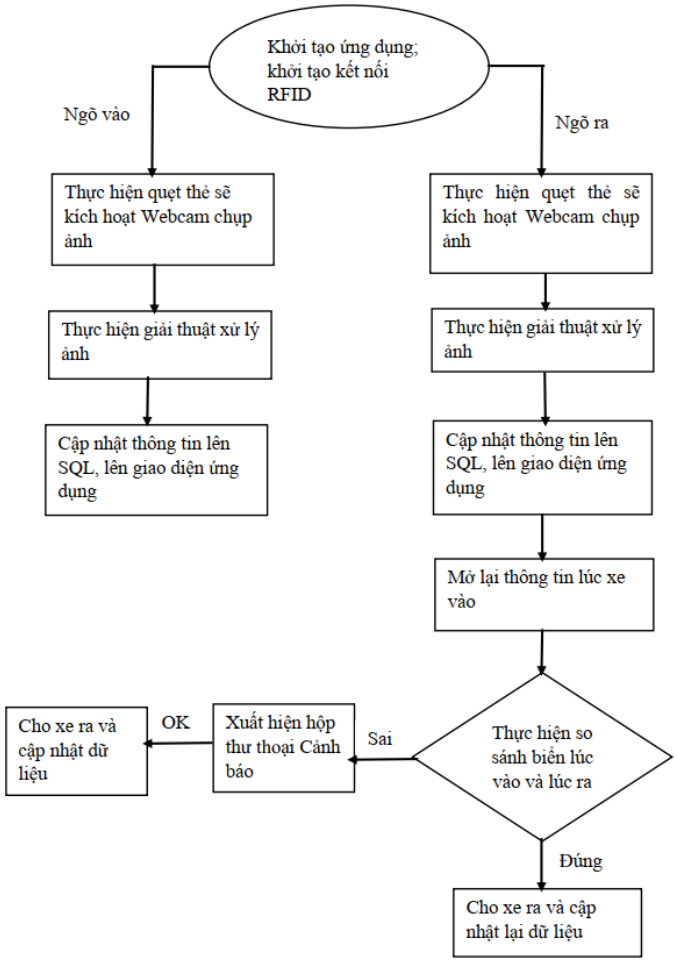
Lúc này hệ thống sẽ quản lý ba loại ảnh: một là loại ảnh thô chụp từ Webcam, hai là loại ảnh nhận dạng ký tự - những ký tự nhận dạng được trong ảnh này sẽ được đóng khung màu xanh lá, ba là loại ảnh xử lý - có màu đen trắng. Thông tin ba loại ảnh sẽ được cập nhật trên hệ thống SQL để quản lý. Ngoài ra những thông tin về trạng thái thẻ, biển số xe, thời gian xe lúc vào cũng sẽ được hiển thị trên giao diện ứng dụng để người dùng tiện theo dõi.

Đối với xe lúc ra, giải thuật xử lý ảnh tương tự với giải thuật xử lý ảnh lúc xe vào (từ B2 đến B7). Tuy nhiên về mặt hệ thống quản lý dữ liệu có chút khác biệt. Sau khi xử lý ảnh và lấy thông tin lúc xe ra thì hệ thống sẽ có chương trình lấy lại thông tin lúc xe vào để đối chiếu. Thông tin này sẽ được hiển thị tại bảng biểu mục Mở lại thông tin xe vào và các picturebox và textbox ở mục Nhận diện biển số xe vào. Khi đối chiếu xong những dữ liệu trong mục Thông tin xe vào cũng sẽ được cập nhật lại để thẻ xe trở lại về trạng thái rảnh còn dùng cho những xe khác.

Sơ đồ khối giải thuật xử lý ảnh:



Sơ đồ khối giải thuật hệ thống quản lý dữ liệu:



Giải thích chương trình

Thực hiện khởi tạo các biến cho đầu đọc thẻ RFID, thực hiện kết nối hệ thống dữ liệu SQL. Thực hiện cấu hình cổng kết nối serialport. Thực hiện khởi tạo nhận diện ảnh, công cụ nhận diện văn bản Tesseract.

Khi Form được load lên hay khởi động ứng dụng thì chúng ta sẽ kiểm tra trạng thái cổng truyền thông serialport. Nếu cổng ở trạng thái đóng thì chúng ta nhấp vào dấu mũi tên ở ô comboBoxRFID chọn COM3 và nhấn button Kết nối. Ngoài ra những dữ liệu ở dataGridView như ở Thông tin xe vào, Mở lại thông tin xe vào và Quản lý xe sẽ được kết nối với SQL.

Khi đóng Form hay tắt ứng dụng thì serialport sẽ tự động ngắt.

```
namespace Parking_App
{
public partial class Form1 : Form
{
#region Khoitao
// khai bao bien cho dau doc the RFID
string InputData = string.Empty;
delegate void SetTextCallback(string text);
bool RF = false;
// tao chuoi knoi SQL
SqlConnection Connect = new SqlConnection(@"Data Source =
.\SQLEXPRESS;Initial   Catalog=csdl_ParkingSys;User
ID=sa;Password=dhbkk18");
private string portcom, baudrate;

#endregion

public Form1()
{
InitializeComponent();
this.portcom = "COM3";
this.baudrate = "115200";
}
#region Define
List<Image<Bgr, byte>> PlateImagesList = new
List<Image<Bgr, byte>>();
Image Plate_Draw;
List<string> PlateTextList = new List<string>();
```

```
List<Rectangle> listRect = new List<Rectangle>();
PictureBox[] box = new PictureBox[12];

public TesseractProcessor full_tesseract = null;
public TesseractProcessor ch_tesseract = null;
public TesseractProcessor num_tesseract = null;
private string m_path = Application.StartupPath +
@"\data\";
private List<string> lstimages = new List<string>();
private const string m_lang = "eng";

//int current = 0;
Capture capture = null;

#endregion

private void Form1_Load(object sender, EventArgs e)
{

if (serialPort1.IsOpen == false)
{
serialPort1.PortName = portcom;
serialPort1.BaudRate = int.Parse(baudrate);
serialPort1.Open();
}
// TODO: This line of code loads data into the
'csdl_ParkingSysDataSetforPA.tt_xv' table. You can move, or
remove it, as needed.
this.tt_xvTableAdapter.Fill(this.csdl_ParkingSysDataSetforP
A.tt_xv);
// TODO: This line of code loads data into the
'csdl_ParkingSysDataSetforPA_mltt.ml_tt' table. You can
move, or remove it, as needed.
this.ml_ttTableAdapter.Fill(this.csdl_ParkingSysDataSetforP
A_mltt.ml_tt);
// TODO: This line of code loads data into the
'csdl_ParkingSysDataSetforPA.QLxe' table. You can move, or
remove it, as needed.
this.qLxeTableAdapter.Fill(this.csdl_ParkingSysDataSetforPA
```

```
.QLxe);


// load cac Port dang su dung len Combobox
string[] port = SerialPort.GetPortNames();
comboBoxRFID.Items.AddRange(port);

Connect.Open();    // mo ket noi CSDL

LoadDataCarIn();   // qly thong tin xe vao
LoadDataCar();     // update du lieu xe vao ra

try
{
capture = new Emgu.CV.Capture();
}
catch { }

timer1.Enabled = true;

full_tesseract = new TesseractProcessor();
bool succeed = full_tesseract.Init(m_path, m_lang, 3);
if (!succeed)
{
MessageBox.Show("Tesseract initialization failed.
The application will exit.");
Application.Exit();
}
full_tesseract.SetVariable("tessedit_char_whitelist"
, "ABCDEFHKLMNPRSTVXY1234567890").ToString();

ch_tesseract = new TesseractProcessor();
succeed = ch_tesseract.Init(m_path, m_lang, 3);
if (!succeed)
{
MessageBox.Show("Tesseract initialization failed.
The application will exit.");
Application.Exit();
}
```

```
ch_tesseract.SetVariable("tessedit_char_whitelist"
, "ABCDEFHKLMNPRSTUVXY").ToString();

num_tesseract = new TesseractProcessor();
succeed = num_tesseract.Init(m_path, m_lang, 3);
if (!succeed)
{
MessageBox.Show("Tesseract initialization failed.
The application will exit.");
Application.Exit();
}
num_tesseract.SetVariable("tessedit_char_whitelist"
, "1234567890").ToString();


m_path = System.Environment.CurrentDirectory + "\\";
string[] ports = SerialPort.GetPortNames();
for (int i = 0; i < box.Length; i++)
{
box[i] = new PictureBox();
}
}

private void Form1_Close(object sender, FormClosedEventArgs
e)
{
// dong cac ket noi
if (serialPort1.IsOpen)
{
serialPort1.Close();
}
Connect.Close();
}
```

Chương trình con kết nối RFID:

```
#region ketnoiRFID
private void btnRFID_Click(object sender, EventArgs e)
{
if (comboBoxRFID.Text == "")
```

```
{
MessageBox.Show("Vui long chon ket noi", "Thong bao"
, MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
else
{
if (!serialPort1.IsOpen)
{
serialPort1.PortName = comboBoxRFID.Text;
serialPort1.Open();
MessageBox.Show("Ket noi thanh cong", "Thong bao"
, MessageBoxButtons.OK, MessageBoxIcon.Information);

comboBoxRFID.Enabled = false;
}
else
{
serialPort1.Close();

comboBoxRFID.Enabled = true;
}
}
}
#endregion
```

Chương trình con quản lý thông tin xe vào:

```
#region quan ly thong tin xe vao
public void LoadDataCarIn()
{
string sqlSelectCarIn = "SELECT ROW_NUMBER () OVER
(ORDER BY [Giovao] DESC) STT, *FROM
[csdl_ParkingSys].[dbo].[tt_xv]";
SqlCommand cmd = new SqlCommand(sqlSelectCarIn, Connect);
DataTable dt = new DataTable();
SqlDataAdapter da = new SqlDataAdapter(cmd);
da.Fill(dt);
dataGridViewTTXV.Invoke((MethodInvoker)(() =>
dataGridViewTTXV.DataSource = dt));
}
```

```
#endregion
```

Chương trình con mở lại thông tin xe vào, sử dụng khi quẹt thẻ ở lối ra:

```
#region mo lai thong tin de doi chieu
public void LoadDataMLTT()
{
string sqlSelectMLTT = "SELECT ROW_NUMBER () OVER
(ORDER BY [Bienso] DESC) STT, *FROM
[csdl_ParkingSys].[dbo].[ml_tt]";
SqlCommand cmd = new SqlCommand(sqlSelectMLTT, Connect);
DataTable dt = new DataTable();
SqlDataAdapter da = new SqlDataAdapter(cmd);
da.Fill(dt);
dataGridViewMLTT.Invoke((MethodInvoker)(() =>
dataGridViewMLTT.DataSource = dt));
}
#endregion
```

Chương trình con quản lý xe vào ra, khi quẹt thẻ ngõ ra thì dataGridViewQLxe sẽ bao gồm thông tin xe lúc vào và thông tin xe lúc ra:

```
#region quanlydlieuxevaora
public void LoadDataCar()
{
string sqlSelectCar = "SELECT ROW_NUMBER () OVER
(ORDER BY [Giora] DESC) STT, *FROM
[csdl_ParkingSys].[dbo].[QLxe]";
SqlCommand cmd = new SqlCommand(sqlSelectCar, Connect);
DataTable dt = new DataTable();
SqlDataAdapter da = new SqlDataAdapter(cmd);
da.Fill(dt);
dataGridViewQLxe.Invoke((MethodInvoker)(() =>
dataGridViewQLxe.DataSource = dt));
}
#endregion
```

Chương trình con xử lý quẹt thẻ và nhận dạng biển số. Sau khi quẹt thẻ tín hiệu sẽ truyền từ đầu đọc RFID về bộ vi xử lý STM32F411VE Discovery. Sau đó bộ vi xử lý sẽ truyền tín hiệu cho máy tính qua chuẩn giao tiếp UART. Nếu tín hiệu truyền

là “I” thì thông tin sẽ được truy vấn ở ngõ vào, nếu là “O” thông tin sẽ được truy vấn ở ngõ ra.

```
private void sp1_datarecieviedhandle(object sender,
SerialDataReceivedEventArgs e)
{
InputData += serialPort1.ReadExisting();

if (InputData == "I")
{
pictureBox_CarOut.Image = null;
ProcessIN(InputData);
InputData = "";
}
if (InputData == "O")
{
pictureBox_CarIn.Image = null;
ProcessOUT(InputData);
InputData = "";
}
}
```

Hàm ProcessIN trong chương trình con xử lý quẹt thẻ và nhận dạng biển số ở ngõ vào. Hàm này xử lý các công việc sau:
- Chụp ảnh và nhận diện biển số. Ảnh được lưu lại gồm 3 loại. Thứ nhất là ảnh thô chụp từ Webcam máy tính. Ảnh này được lưu trong folder in – đường dẫn có ghi trong đoạn code. Thứ hai là ảnh nhận diện ký tự. Những ký tự nhận diện được được đóng khung màu xanh lá. Ảnh này được lưu vào folder bs_in – đường dẫn có ghi trong đoạn code. Thứ ba là ảnh xử lý những ký tự nhận dạng được. Ảnh này được lưu vào folder kt_in – đường dẫn có ghi trong đoạn code.
- Thực hiện định dạng tên ảnh.
- Thực hiện điền thông tin vào các ô textbox ngõ vào và cập nhật dữ liệu tại dataGridViewTTXV như trạng thái thẻ, biển số xe, thời gian xe vào.

```
private void ProcessIN (string text)
{
if (this.txtID.InvokeRequired)
{
SetTextCallback d = new SetTextCallback(ProcessIN);
this.Invoke(d, new object[] { text });
}
```

```
else this.txtID.Text = text;
string mathe = text.Trim();

#region chup anh va nhan dien bien so
pictureBox_CarIn.Image =
(Bitmap)pictureBox_WC.Image.Clone();
this.Invoke((MethodInvoker)delegate
{
pictureBox_CarIn.Image.Save(@"D:\CTDT_dtu_vthong\mon_chngan
h_bb
\do an_HK212\thvien_anh\in\" + text_PlateIn.Text
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg");

});
bool nhdang_in = false; // kiem tra nhan dang bien so vao
Image temp1;
string temp2, temp3;
Reconize(@"D:\CTDT_dtu_vthong\mon_chnganh_bb
\do an_HK212\thvien_anh\in\"
+DateTime.Now.ToString("_yyyyMMdd_HHmmss")
+ ".jpg", out temp1, out temp2, out temp3);

if (temp2.Length != 9 && temp2.Length != 10)
{
text_PlateIn.Invoke((MethodInvoker)(() =>
text_PlateIn.Text = "Error"));
nhdang_in = false;
serialPort1.Write("E");
pictureBox_CarIn.Image = null;
}
else
{
text_PlateIn.Invoke((MethodInvoker)(() =>
text_PlateIn.Text = temp2));
nhdang_in = true;
serialPort1.Write("D");
serialPort1.Write(temp2);

}
```

```
pictureBox_PlateIn.Image.Save(@"D:\CTDT_dtu_vthong\mon_chng
anh_bb
\do an_HK212\thvien_anh\bs_in\" + text_PlateIn.Text
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg");
pictureBox_PlateIn2.Image.Save(@"D:\CTDT_dtu_vthong\mon_chn
ganh_bb
\do an_HK212\thvien_anh\kt_in\" + text_PlateIn.Text
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg");
#endregion


string sql_ispicture = "INSERT INTO Pictures(bso, picdate)
VALUES('"
+ text_PlateIn.Text + "', '"
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + "')";
SqlCommand cmd_ispicture = new SqlCommand(sql_ispicture,
Connect);
cmd_ispicture.ExecuteNonQuery();

#region lay thong tin xe vao
txtIDvao.Text = mathe;
txtTimeVao.Invoke((MethodInvoker)(() =>
txtTimeVao.Text = DateTime.Now.ToString("yyyy/MM/dd
HH:mm:ss")));
string sql_iscarin = "INSERT INTO tt_xv(Mathe, Bienso,
Giovao)
VALUES( '" + mathe + "', '" + text_PlateIn.Text + "', '"
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + "')";
SqlCommand cmd_carin = new SqlCommand(sql_iscarin,
Connect);
cmd_carin.ExecuteNonQuery();
LoadDataCarIn();
#endregion

}
```

Hàm ProcessOUT trong chương trình con xử lý quẹt thẻ và nhận dạng biển số ở ngõ ra. Hàm này xử lý các công việc sau:

- Chụp ảnh và nhận diện biển số. Ảnh được lưu lại gồm 3 loại. Thứ nhất là ảnh thô chụp từ Webcam máy tính. Ảnh này được lưu trong folder in – đường dẫn có ghi trong đoạn code. Thứ hai là ảnh nhận diện ký tự. Những ký tự nhận diện được được đóng khung màu xanh lá. Ảnh này được lưu vào folder bs_in – đường dẫn có ghi trong đoạn code. Thứ ba là ảnh xử lý những ký tự nhận dạng được. Ảnh này được lưu vào folder kt_in – đường dẫn có ghi trong đoạn code.

- Thực hiện việc mở lại thông tin lúc xe vào để đối chiếu. Thông này sẽ được hiển thị tại dataGridViewMLTT trên giao diện ứng dụng. Ngoài ra, thông tin ảnh ngõ vào cũng sẽ được tái hiện ở ô pictureBox_PlateIn và pictureBox_PlateIn2 và các ô textbox của Biển số vào, Trạng thái thẻ và Thời gian vào.

- Thực hiện việc so sánh thông tin xe vào và thông tin xe ra. Thông tin xe ra sẽ được hiển thị tại dataGridViewQLxe. Nếu thông không khớp thì xuất hiện hộp thoại cảnh báo. Ngoài ra thì dữ liệu trong Thông tin xe vào sẽ được xóa để thẻ xe ở trạng thái rảnh để sử dụng cho xe khác.

```
private void ProcessOUT(string text)
{
if (this.txtID.InvokeRequired)
{
SetTextCallback d = new SetTextCallback(ProcessOUT);
this.Invoke(d, new object[] { text });
}
else this.txtID.Text = text;
string mathe = text.Trim();
pictureBox_CarOut.Image =
(Bitmap)pictureBox_WC.Image.Clone();
this.Invoke((MethodInvoker)delegate
{
pictureBox_CarOut.Image.Save(@"D:\CTDT_dtu_vthong\mon_chnga
nh_bb
\do an_HK212\thvien_anh\out\" + text_PlateOut.Text
+  DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg");
```

```
});

Image yyy = pictureBox_CarOut.Image;
bool nhdang_out = false; // kiem tra nhan dang bien so ra
Image temp1r;
string temp2r, temp3r;
ReconizeOut(@"D:\CTDT_dtu_vthong\mon_chnganh_bb
\do an_HK212\thvien_anh\out\" + text_PlateOut.Text
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg"
, out temp1r, out temp2r, out temp3r);
yyy = temp1r;
if (temp2r.Length != 9 && temp2r.Length != 10 )
{
text_PlateOut.Invoke((MethodInvoker)(() =>
text_PlateOut.Text = "Error!"));
nhdang_out = false;
serialPort1.Write("E");
pictureBox_CarOut.Image = null;
}
else
{
text_PlateOut.Invoke((MethodInvoker)(() =>
text_PlateOut.Text = temp2r));
nhdang_out = true;
serialPort1.Write("D");
serialPort1.Write(temp2r);
}

pictureBox_PlateOut.Image.Save(@"D:\CTDT_dtu_vthong\mon_chn
ganh_bb
\do an_HK212\thvien_anh\bs_out\" + text_PlateOut.Text
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg");
pictureBox_PlateOut2.Image.Save(@"D:\CTDT_dtu_vthong\mon_ch
nganh_bb
\do an_HK212\thvien_anh\kt_out\" + text_PlateOut.Text
+ DateTime.Now.ToString("_yyyyMMdd_HHmmss") + ".jpg");

#region Mo lai thong tin luc xe vao de kiem tra
string sql_mltt = "INSERT INTO ml_tt(Mathe, Bienso, Giovao)
```

```
SELECT
Mathe, Bienso, Giovao FROM tt_xv WHERE Bienso = '"
+ text_PlateOut.Text + "'";
SqlCommand cmd_mltt = new SqlCommand(sql_mltt, Connect);
cmd_mltt.ExecuteNonQuery();
LoadDataMLTT();

string sql_mltt_again = @"SELECT * FROM ml_tt";
SqlDataAdapter da_mltt_again = new
SqlDataAdapter(sql_mltt_again, Connect);
DataSet ds_mltt_again = new DataSet();
da_mltt_again.Fill(ds_mltt_again, "ml_tt");
txtIDvao.DataBindings.Clear();
txtIDvao.Invoke((MethodInvoker)(() =>
txtIDvao.DataBindings.Add(
"Text", ds_mltt_again, "ml_tt.Mathe")));
text_PlateIn.DataBindings.Clear();
text_PlateIn.Invoke((MethodInvoker)(() =>
text_PlateIn.DataBindings.Add("Text", ds_mltt_again,
"ml_tt.Bienso")));
txtTimeVao.DataBindings.Clear();
txtTimeVao.Invoke((MethodInvoker)(() =>
txtTimeVao.DataBindings.Add(
"Text", ds_mltt_again, "ml_tt.Giovao")));

pictureBox_PlateIn.Image = new Bitmap(Image.FromFile(
@"D:\CTDT_dtu_vthong\mon_chnganh_bb\do
an_HK212\thvien_anh\bs_in\"
+ text_PlateIn.Text + txtTimeVao.Text + ".jpg"));
pictureBox_PlateIn2.Image = new Bitmap(Image.FromFile(
@"D:\CTDT_dtu_vthong\mon_chnganh_bb\do
an_HK212\thvien_anh\kt_in\"
+ text_PlateIn.Text + txtTimeVao.Text + ".jpg"));
#endregion

if (text_PlateIn.Text == text_PlateOut.Text)
{
txtIDra.Text = mathe;
txtTimeRa.Invoke((MethodInvoker)(() => txtTimeRa.Text =
```

```
DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss")));
string sqlInsert_carout = "INSERT INTO QLxe(Mathe, Bienso,
Giovao,
Giora) VALUES('" + mathe + "', '" + text_PlateOut.Text +
"', '"
+ txtTimeVao.Text + "', '" + txtTimeRa.Text + "')";
SqlCommand cmd_carout = new SqlCommand(sqlInsert_carout,
Connect);
cmd_carout.ExecuteNonQuery();
LoadDataCar();
string sqlDelete_carin = "DELETE FROM
[csdl_ParkingSys].[dbo].[tt_xv]
WHERE Bienso = '" + text_PlateIn.Text + "'";
SqlCommand cmd_carin = new SqlCommand(sqlDelete_carin,
Connect);
cmd_carin.ExecuteNonQuery();
LoadDataCarIn();
string sqlDelete_pics = "DELETE FROM
[csdl_ParkingSys].[dbo].[Pictures]
WHERE bso = '" + text_PlateIn.Text + "'";
SqlCommand cmd_pics = new SqlCommand(sqlDelete_pics,
Connect);
cmd_pics.ExecuteNonQuery();
}
if (text_PlateOut.Text != text_PlateIn.Text)
{
DialogResult ktra = MessageBox.Show("Bien so xe khong
trung.
\n Ban co muon cho xe ra?", "Thong bao",
MessageBoxButtons.OKCancel
, MessageBoxIcon.Error);
if (ktra == DialogResult.OK)
{
txtIDra.Text = "Khong hop le";
txtTimeRa.Invoke((MethodInvoker)(() => txtTimeRa.Text =
DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss")));
string sqlInsert_carout = "INSERT INTO QLxe(Bienso, Giovao,
Giora)
VALUES( '" + text_PlateIn.Text + "', '" + txtTimeVao.Text +
```

```
"', '"
+ txtTimeRa.Text + "')";
SqlCommand cmd_carout = new SqlCommand(sqlInsert_carout,
Connect);
cmd_carout.ExecuteNonQuery();
LoadDataCar();
string sqlDelete_carin = "DELETE FROM
[csdl_ParkingSys].[dbo].[tt_xv] WHERE Bienso = '"
+ text_PlateIn.Text + "'";
SqlCommand cmd_carin = new SqlCommand(sqlDelete_carin,
Connect);
cmd_carin.ExecuteNonQuery();
LoadDataCarIn();
string sqlDelete_pics = "DELETE FROM
[csdl_ParkingSys].[dbo].[Pictures] WHERE bso = '"
+ text_PlateIn.Text + "'";
SqlCommand cmd_pics = new SqlCommand(sqlDelete_pics,
Connect);
cmd_pics.ExecuteNonQuery();
}
}
}
```

Chương trình xử lý ảnh ngõ vào:

```
public void ProcessImageIn(string urlImage)
{
PlateImagesList.Clear();
PlateTextList.Clear();
Bitmap imagein = new Bitmap(pictureBox_CarIn.Image);
FindLicensePlate(imagein, out Plate_Draw);

}
```

Thực hiện việc xoay ảnh:

```
public static Bitmap RotateImage(Image image, float angle)
{
if (image == null)
throw new ArgumentNullException("image");
```

```
PointF offset = new PointF((float)image.Width / 2,
(float)image.Height / 2);

//create a new empty bitmap to hold rotated image
Bitmap rotatedBmp = new Bitmap(image.Width, image.Height);
rotatedBmp.SetResolution(image.HorizontalResolution
, image.VerticalResolution);

//make a graphics object from the empty bitmap
Graphics g = Graphics.FromImage(rotatedBmp);

//Put the rotation point in the center of the image
g.TranslateTransform(offset.X, offset.Y);

//rotate the image
g.RotateTransform(angle);

//move the image back
g.TranslateTransform(-offset.X, -offset.Y);

//draw passed in image onto graphics object
g.DrawImage(image, new PointF(0, 0));

return rotatedBmp;
}
```

Xử lý ảnh từ mã nguồn mở, mã đó gồm 3000 ảnh biển số xe đã được training:

```
public void FindLicensePlate(Bitmap image, out Image
Plate_Draw)
{
Plate_Draw = null;
Image<Bgr, byte> frame;
bool isface = false; // neu isface dung thi thuc hien ko
thi ko thuc hien
Image dst = image;
Bitmap src;
//cong nghe Haarcascade dan xuat den file co ma nguon mo,
ma trong do gom rat nhieu file da duoc training (3000 anh
bien so xe)
```

```
HaarCascade haar = new HaarCascade(Application.StartupPath
+
"\\output-hv-33-x25.xml");
for (float i = 0; i <= 20; i += 3)
{
for (float s = -1; s <= 1 && s + i != 1; s += 2)
{
//src bang hinh anh bmp tra ve la anh xoay
src = RotateImage(dst, i * s);
PlateImagesList.Clear();
frame = new Image<Bgr, byte>(src);
using (Image<Gray, byte> grayframe = new Image<Gray,
byte>(src))
{
//phat hien cho nao la bien so xe trong 1 image
//1.1 la do scale anh
//8: vi du nhin duoc so 1 thi no dem xung quanh no co
8 so tro len de lam 1 vung bien so xe
var faces = grayframe.DetectHaarCascade(haar, 1.1, 8
, HAAR_DETECTION_TYPE.DO_CANNY_PRUNING, new Size(0, 0))[0];
//khoanh cung bien so xe
foreach (var face in faces)
{
Image<Bgr, byte> tmp = frame.Copy();//copy tu anh
da duoc lam de nhin
tmp.ROI = face.rect;//xuat khung hcn khoanh vung bien so
frame.Draw(face.rect, new Bgr(Color.Red), 2);//khung dc
son mau do va do dai bang 2
PlateImagesList.Add(tmp);//them anh da dc khoanh vung
bien so xe vao danh sach
isface = true;
}

if (isface)
{
Image<Bgr, byte> showing = frame.Clone();//tao 1 bang copy
rieng cua frame
Plate_Draw = (Image)showing.ToBitmap();//chuyen sang bitmap
if (PlateImagesList.Count > 1)
```

```
{
for (int k = 1; k < PlateImagesList.Count; k++)
{
if (PlateImagesList[0].Width <
PlateImagesList[k].Width)
{
PlateImagesList[0] = PlateImagesList[k];
}
}
}
PlateImagesList[0] = PlateImagesList[0].Resize(400, 400,
Emgu.CV.CvEnum.INTER.CV_INTER_LINEAR);//giam bo nho
return;
}
}
}
}
}
```

Chương trình Reconize có nhiệm vụ tìm tập các điểm liên tục tạo thành một đường cong có cùng màu sắc hoặc giá trị cường độ; làm mịn đường cong; lọc và sắp xếp lý tự nhận dạng được; thực hiện phóng đại.

```
private void Reconize(string link, out Image hinhbienso,
out string bienso, out string bienso_text)
{
for (int i = 0; i < box.Length; i++)
{
this.Controls.Remove(box[i]);
}
//gan gtri ban dau
hinhbienso = null;
bienso = "";
bienso_text = "";

ProcessImageIn(link);
//neu co bien so xe thi...
if (PlateImagesList.Count != 0)
{
Image<Bgr, byte> src = new Image<Bgr,
```

```
byte>(PlateImagesList[0].ToBitmap());
Bitmap grayframe;
Bitmap color;
int c = clsBSoft.IdentifyContours(src.ToBitmap(), 50, false
, out grayframe, out color, out listRect);
pictureBox_PlateIn.Image = color;
hinhbienso = Plate_Draw;
pictureBox_PlateIn2.Image = grayframe;
Image<Gray, byte> dst = new Image<Gray, byte>(grayframe);
grayframe = dst.ToBitmap();

string zz = "";
// lọc và sắp xếp số
// Library đc training các chữ cáivà nó sẽ scale đến khi
phù hợp
List<Bitmap> bmp = new List<Bitmap>();
List<int> erode = new List<int>();
List<Rectangle> up = new List<Rectangle>();
List<Rectangle> dow = new List<Rectangle>();
int up_y = 0, dow_y = 0;
bool flag_up = false;

int di = 0;

if (listRect == null) return;

for (int i = 0; i < listRect.Count; i++)
{
Bitmap ch = grayframe.Clone(listRect[i],
grayframe.PixelFormat);
int cou = 0;
full_tesseract.Clear();
full_tesseract.ClearAdaptiveClassifier();
string temp = full_tesseract.Apply(ch);
while (temp.Length > 3)
{
Image<Gray, byte> temp2 = new Image<Gray, byte>(ch);
temp2 = temp2.Erode(2);
ch = temp2.ToBitmap();
```

```
full_tesseract.Clear();
full_tesseract.ClearAdaptiveClassifier();
temp = full_tesseract.Apply(ch);
cou++;
if (cou > 10)
{
listRect.RemoveAt(i);
i--;
di = 0;
break;
}
di = cou;
}
}

for (int i = 0; i < listRect.Count; i++)
{
for (int j = i; j < listRect.Count; j++)
{
if (listRect[i].Y > listRect[j].Y + 100)
{
flag_up = true;
up_y = listRect[j].Y;
dow_y = listRect[i].Y;
break;
}
else if (listRect[j].Y > listRect[i].Y + 100)
{
flag_up = true;
up_y = listRect[i].Y;
dow_y = listRect[j].Y;
break;
}
if (flag_up == true) break;
}
}

for (int i = 0; i < listRect.Count; i++)
{
```

```
if (listRect[i].Y < up_y + 50 && listRect[i].Y > up_y - 50)
{
up.Add(listRect[i]);
}
else if (listRect[i].Y < dow_y + 50 && listRect[i].Y >
dow_y - 50)
{
dow.Add(listRect[i]);
}
}

if (flag_up == false) dow = listRect;

for (int i = 0; i < up.Count; i++)
{
for (int j = i; j < up.Count; j++)
{
if (up[i].X > up[j].X)
{
Rectangle w = up[i];
up[i] = up[j];
up[j] = w;
}
}
}
for (int i = 0; i < dow.Count; i++)
{
for (int j = i; j < dow.Count; j++)
{
if (dow[i].X > dow[j].X)
{
Rectangle w = dow[i];
dow[i] = dow[j];
dow[j] = w;
}
}
}

int x = 0;
```

```
int c_x = 0;

for (int i = 0; i < up.Count; i++)
{
Bitmap ch = grayframe.Clone(up[i], grayframe.PixelFormat);
Bitmap o = ch;
string temp;
if (i < 2)
{
temp = clsBSoft.Ocr(ch, false, full_tesseract,
num_tesseract
, ch_tesseract, true); // nhan dien so
}
else
{
temp = clsBSoft.Ocr(ch, false, full_tesseract,
num_tesseract
, ch_tesseract, false); // nhan dien chu
}

zz += temp;
c_x++;
}
bienso = zz + "-";
zz += "\r\n";
for (int i = 0; i < dow.Count; i++)
{
Bitmap ch = grayframe.Clone(dow[i], grayframe.PixelFormat);
string temp = clsBSoft.Ocr(ch, false, full_tesseract
, num_tesseract, ch_tesseract, true); // nhan dien so
zz += temp;
bienso += temp;
}
bienso_text = zz;
bienso = bienso.Replace("\n", "");
bienso = bienso.Replace("\r", "");
if (bienso.Length == 9) bienso += "-";
}
}
```

Tương tự ngõ vào, ta cũng có các chương trình con xử lý ảnh ở ngõ ra:

```
public void ProcessImageOut(string urlImage)
{
PlateImagesList.Clear();
PlateTextList.Clear();
Bitmap imageout = new Bitmap(pictureBox_CarOut.Image);
FindLicensePlate(imageout, out Plate_Draw);
}

private void ReconizeOut(string link, out Image hinhbienso,
out string bienso
, out string bienso_text)
{
for (int i = 0; i < box.Length; i++)
{
this.Controls.Remove(box[i]);
}
//gan gtri ban dau
hinhbienso = null;
bienso = "";
bienso_text = "";

ProcessImageOut(link);
//neu co bien so xe thi...
if (PlateImagesList.Count != 0)
{
Image<Bgr, byte> src = new Image<Bgr
, byte>(PlateImagesList[0].ToBitmap());
Bitmap grayframe;
Bitmap color;
int c = clsBSoft.IdentifyContours(src.ToBitmap(), 50, false
, out grayframe, out color, out listRect);
pictureBox_PlateOut.Image = color;
hinhbienso = Plate_Draw;
pictureBox_PlateOut2.Image = grayframe;
Image<Gray, byte> dst = new Image<Gray, byte>(grayframe);
grayframe = dst.ToBitmap();
```

```
string zz = "";
// lọc và sắp xếp số
// Library đc training các chữ cáivà nó sẽ scale đến khi
phù hợp
List<Bitmap> bmp = new List<Bitmap>();
List<int> erode = new List<int>();
List<Rectangle> up = new List<Rectangle>();
List<Rectangle> dow = new List<Rectangle>();
int up_y = 0, dow_y = 0;
bool flag_up = false;

int di = 0;

if (listRect == null) return;

for (int i = 0; i < listRect.Count; i++)
{
Bitmap ch = grayframe.Clone(listRect[i],
grayframe.PixelFormat);
int cou = 0;
full_tesseract.Clear();
full_tesseract.ClearAdaptiveClassifier();
string temp = full_tesseract.Apply(ch);
while (temp.Length > 3)
{
Image<Gray, byte> temp2 = new Image<Gray, byte>(ch);
temp2 = temp2.Erode(2);
ch = temp2.ToBitmap();
full_tesseract.Clear();
full_tesseract.ClearAdaptiveClassifier();
temp = full_tesseract.Apply(ch);
cou++;
if (cou > 10)
{
listRect.RemoveAt(i);
i--;
di = 0;
break;
}
```

```
di = cou;
}
}

for (int i = 0; i < listRect.Count; i++)
{
for (int j = i; j < listRect.Count; j++)
{
if (listRect[i].Y > listRect[j].Y + 100)
{
flag_up = true;
up_y = listRect[j].Y;
dow_y = listRect[i].Y;
break;
}
else if (listRect[j].Y > listRect[i].Y + 100)
{
flag_up = true;
up_y = listRect[i].Y;
dow_y = listRect[j].Y;
break;
}
if (flag_up == true) break;
}
}

for (int i = 0; i < listRect.Count; i++)
{
if (listRect[i].Y < up_y + 50 && listRect[i].Y > up_y - 50)
{
up.Add(listRect[i]);
}
else if (listRect[i].Y < dow_y + 50 && listRect[i].Y >
dow_y - 50)
{
dow.Add(listRect[i]);
}
}
```

```
if (flag_up == false) dow = listRect;

for (int i = 0; i < up.Count; i++)
{
for (int j = i; j < up.Count; j++)
{
if (up[i].X > up[j].X)
{
Rectangle w = up[i];
up[i] = up[j];
up[j] = w;
}
}
}
for (int i = 0; i < dow.Count; i++)
{
for (int j = i; j < dow.Count; j++)
{
if (dow[i].X > dow[j].X)
{
Rectangle w = dow[i];
dow[i] = dow[j];
dow[j] = w;
}
}
}

int x = 0;
int c_x = 0;

for (int i = 0; i < up.Count; i++)
{
Bitmap ch = grayframe.Clone(up[i], grayframe.PixelFormat);
Bitmap o = ch;
string temp;
if (i < 2)
{
temp = clsBSoft.Ocr(ch, false, full_tesseract,
num_tesseract
```

```
, ch_tesseract, true); // nhan dien so
}
else
{
temp = clsBSoft.Ocr(ch, false, full_tesseract,
num_tesseract
, ch_tesseract, false); // nhan dien chu
}

zz += temp;
c_x++;
}
bienso = zz + "-";
zz += "\r\n";
for (int i = 0; i < dow.Count; i++)
{
Bitmap ch = grayframe.Clone(dow[i], grayframe.PixelFormat);
string temp = clsBSoft.Ocr(ch, false, full_tesseract
, num_tesseract, ch_tesseract, true); // nhan dien so
zz += temp;
bienso += temp;
}
bienso_text = zz;
bienso = bienso.Replace("\n", "");
bienso = bienso.Replace("\r", "");
if (bienso.Length == 9) bienso += "-";
}
}
```

Chương trình xử lý WEBCAM:

```
class WEBCAM
{
private bool DeviceExist = false;
private FilterInfoCollection videoDevices;
private VideoCaptureDevice videoSource = null;
public List<string> list_cam = new List<string>();
public Bitmap bitmap;
public Image image;
public string status = "Non!";
```

```
public string pb = "";

public static List<string> get_all_cam()
{
List<string> ls = new List<string>();
try
{
FilterInfoCollection videoDevices = new
FilterInfoCollection(
FilterCategory.VideoInputDevice);
if (videoDevices.Count == 0)
throw new ApplicationException();
foreach (FilterInfo device in videoDevices)
{
ls.Add(device.Name);
}
}
catch (ApplicationException)
{
ls.Clear();
}
return ls;
}
public void refesh()
{
try
{
videoDevices = new FilterInfoCollection(
FilterCategory.VideoInputDevice);
list_cam.Clear();
if (videoDevices.Count == 0)
throw new ApplicationException();

DeviceExist = true;
foreach (FilterInfo device in videoDevices)
{
list_cam.Add(device.Name);
}
}
```

```
catch (ApplicationException)
{
DeviceExist = false;
list_cam.Clear();
list_cam.Add("No capture device on your system");
}
}
public void Start(int index)
{
refesh();
if (DeviceExist)
{
videoSource = new VideoCaptureDevice(
videoDevices[index].MonikerString);
videoSource.NewFrame += new
NewFrameEventHandler(video_NewFrame);
CloseVideoSource();
videoSource.DesiredFrameSize = new Size(640, 480);
//videoSource.DesiredFrameRate = 10;
videoSource.Start();
status = "run";
}
}
public void Stop()
{
if (videoSource.IsRunning)
{
CloseVideoSource();
status = "stop";
}
}
private void video_NewFrame(object sender,
NewFrameEventArgs eventArgs)
{
bitmap = (Bitmap)eventArgs.Frame.Clone();
image = bitmap;
}
private void CloseVideoSource()
{
```

```
if (!(videoSource == null))
if (videoSource.IsRunning)
{
videoSource.SignalToStop();
videoSource = null;
}
}
public void Save()
{
string path = Application.StartupPath + @"\data\" +
"aa.bmp";
image.Save(path);
}
public void put_picturebox(string name)
{
pb = name;
}
}
```

Thực hiện cấu hình cho App.config:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
<configSections>
</configSections>
<connectionStrings>
<add
name="Parking_App.Properties.Settings.csdl_ParkingSysConnec
tionStringforPA"
connectionString="Data Source=.\SQLEXPRESS;Initial
Catalog=csdl_ParkingSys;
User ID=sa;Password=dhbkk18"
providerName="System.Data.SqlClient" />
</connectionStrings>
<startup useLegacyV2RuntimeActivationPolicy="true">
<supportedRuntime version="v4.0"
sku=".NETFramework,Version=v4.6.1" />
</startup>
</configuration>
```

CHƯƠNG 6: KẾT LUẬN

Sau khi hoàn thành đề tài đồ án lần này, về cơ vản nhóm đã được các mục tiêu sau:

- Biết được cách thức giao tiếp và sử dụng vi điều khiển STM32.
- Biết cách sử dụng phần mềm STM32CubeIDE để viết phần mềm điều khiển STM.
- Biết cách sử dụng và lập trình ngôn ngữ C#.
- Biết tạo giao diện người dùng bằng WindowForm và liên kết với SQL.

Ưu điểm:

- Nhóm làm việc ăn ý, hiệu quả.
- Sản phẩm hoạt động khá đúng yêu cầu đề ra.

Khuyết điểm:

- Chưa hoàn thiện 100% sản phẩm, mới dừng lại ở dạng mô hình và còn đi dây phức tạp, độ thẩm mĩ chưa cao.
- Phần xử lý biển số xe vẫn chưa được nhạy.

CHƯƠNG 7: TÀI LIỆU THAM KHẢO

STM32 Arm Cortex MCUs - 32-bit Microcontrollers - STMicroelectronics

MFRC522 - Arduino Reference

https://www.slideshare.net/trongthuy2/luan-van-mo-hinh-dieu-khien-va-giam-sat-bai-giu-xe-o-to-tu-dong

https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf

https://www.st.com/resource/en/user_manual/dm00148985-discovery-kit-with-stm32f411ve-mcu-stmicroelectronics.pdf

https://tapit.vn/giao-tiep-stm32f103c8t6-voi-lcd-16x2-thong-qua-module-i2c/

https://tapit.vn/chuan-giao-tiep-spi-tren-stm32f4/

http://iot47.com/giao-tiep-voi-module-the-tu-rfid-rc522/

https://vidieukhien.xyz/2018/04/22/bai-14-stm32f4-ung-dung-spi-su-dung-cong-nghe-rfid-rc522/

https://dothanhspyb.com/bai-2-ket-noi-va-doc-csdl-sql-server-len-windows-form-voi-visual-studio-2015/

https://laptrinhvb.net/bai-viet/devexpress/Chuong-trinh-nhan-dang-bien-so-xe-bang--Csharp--(Demo-version)/3a4500923205f566.html