ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

BÁO CÁO BÀI TẬP LỚN

THIẾT KẾ MÁY BÁN HÀNG TỰ ĐỘNG CƠ BẢN SỬ DỤNG STM32

GVHD: Nguyễn Phan Hải Phú

Lớp: L04 - Nhóm: 10

Họ và tên	MSSV	Nhiệm vụ
Trịnh Thế Hải	2310885	IR sensor + Nút Nhấn + Vỏ hộp
Lê Vĩnh Phúc	2312685	LCD + Motor + Vỏ hộp
Huỳnh Bảo Anh	2210060	LCD + Motor + Vỏ hộp
Huỳnh Tấn Đạt	2210675	IR sensor + Nút nhấn + Vỏ hộp

TP. HỒ CHÍ MINH, 12/2025

MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU

1.1. Đặt vấn đề (Lý do chọn đề tài):

Xu hướng tự động hóa toàn cầu ngày càng phát triển nhanh chóng nhờ vào AI, do đó mà các thiết bị phục vụ tự động cũng ngày càng trở nên phổ biến. Để kịp chạy theo xu thế, nhóm quyết định lựa chọn đề tài máy bán hàng tự động (Vending Machine) như một ví dụ điển hình vì nó đóng vai trò quan trọng trong đời sống hiện đại nhờ khả năng hoạt động liên tục 24/7, tiết kiệm diện tích, giảm chi phí nhân công bán hàng và mang lại sự tiện lợi tối đa cho người tiêu dùng tại các khu vực công cộng như trường học, bệnh viện, hay nhà ga.

Ngoài đi theo xu hướng, nhóm còn có nhu cầu tìm hiểu sâu và ứng dụng thực tế dòng vi điều khiển hiện đại STM32 vào các hệ thống nhúng vì STM32 là dòng vi điều khiển mạnh mẽ, đa năng và được sử dụng rộng rãi trong công nghiệp.

Xuất phát từ thực tế đó, nhóm quyết định chọn đề tài "Thiết kế và triển mô hình Máy bán hàng tự động sử dụng vi điều khiển STM32". Đề tài này không chỉ giúp giải quyết bài toán mô phỏng một hệ thống thực tế mà còn là cơ hội để nhóm củng cố kiến thức về lập trình nhúng và thiết kế phần cứng.

1.2. Mục tiêu của đề tài:

Thiết kế và chế tạo được phần khung vỏ cho máy bán hàng.

Xây dựng được các chức năng cơ bản vận hành ổn định:

- Nhận tiền: Hệ thống ghi nhận tín hiệu nạp tiền từ người dùng.
- Chọn món: Người dùng lựa chọn sản phẩm mong muốn qua giao diện LCD và tương tác nút nhấn.
- Trả hàng: Dựa vào món hàng người dùng đã chọn, hoạt động chính xác để đưa đúng sản phẩm đã chọn ra khay lấy đồ.
- Thối tiền:Tính toán thực hiện quy trình trả lại tiền thừa.

Vận hành được dòng vi điều khiển STM32 cụ thể là STM32F103C8T6

Ứng dụng được các kỹ thuật lập trình ngoại vi trong khi giao tiếp với STM32

- GPIO: Đọc trạng thái nút nhấn và hiển thị lên LCD
- Ngắt : Xử lý tín hiệu đầu vào tức thời, đảm bảo hệ thống không bị treo khi chờ đợi người dùng.
- Timer/PWM: Tạo xung điều khiển động cơ quay lò xo đủ thời gian để đẩy hàng.

1.3. Giới hạn và Phạm vi đề tài:

Do giới hạn về thời gian thực hiện và kinh phí, đề tài được thực hiện dưới dạng mô hình prototype với các phạm vi cụ thể như sau

- Quy mô mô hình: Đây là mô hình thử nghiệm, nhóm chỉ tập trung vào giải thuật điều khiển và nguyên lý hoạt động hơn là độ bền của sản phẩm.
- Số lượng sản phẩm: Hệ thống được thiết kế để chứa và quản lý 4 loại sản phẩm khác nhau.
- Phương thức thanh toán: máy bán hàng của nhóm sẽ không tích hợp module đọc tiền thật do chi phí cao và phức tạp về phần cứng. Thay vào đó, việc thanh toán được giả lập bằng các nút nhấn. Ví dụ như nhấn nút một lần sẽ cộng thêm 5 đồng, nếu để quá lâu không chọn sản phẩm hệ thống ngắt trong hệ thống sẽ tự động đếm để ngắt tránh trường hợp hệ thống bị treo.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Tổng quan về Vi điều khiển STM32F103C8T6:

STM32 là dòng chip 32bit của hãng STMicroelectronics sử dụng công nghệ lõi ARM Cortex mạnh mẽ, hiệu năng tốt nhưng vẫn giữ được giá thành rẻ, phù hợp với đa số các công ty hiện nay. Lý do nhóm chọn STM32 để thực hiên dự án thay vì arduino vì STM32 có khả năng xử lý các tác vụ phức tạp hơn, linh hoạt hơn và có giá thành rẻ hơn, tuy phần lập trình sẽ khó hơn arduino do không có thư viện tích hợp sẵn, tuy nhiên nhờ khó khăn đó mà giúp nhóm học hỏi được nhiều kỹ năng lập trình hơn từ đó vận hành được các vi điều khiển khác phức tạp hơn. STM32 sử dụng kiến trúc Harvard của Cortex-M3 cho phép truy xuất đồng thời mã lệnh và dữ liệu, giúp tăng tốc độ xử lý thực thi lệnh. Ngoài ra, bộ điều khiển ngắt lồng nhau (NVIC) được tích hợp sâu giúp việc xử lý các sự kiện thời gian thực trở nên nhanh chóng và chính xác.
Vi điều khiển STM32F103C8T6 hoạt động với tốc độ xung clock là 72MHz, Flash 64KB, RAM 20KB. Nhìn qua các thông số ta có thể thấy hiệu suất hoạt động của STM32 vượt trội hơn nhiều so với các arduino hiện tại từ đó củng cố hơn cho quyết định chọn STM32 để thực hiện dự án này của nhóm.

2.2. Các chuẩn giao tiếp và ngoại vi:

Trong hệ thống máy bán hàng, việc phát hiện người dùng nhấn nút (chọn món/nạp tiền) cần độ phản hồi tức thì.

- GPIO (General Purpose Input/Output): Các chân vi điều khiển được cấu hình là Input Pull-up (kéo lên nguồn). Trạng thái bình thường là mức 1 (High), khi nhấn nút sẽ xuống mức 0 (Low).
- Ngắt ngoài hay EXTI (External Interrupt): Thay vì để CPU liên tục kiểm tra trạng thái nút nhấn, đề tài sử dụng ngắt cạnh xuống (Falling Edge).

Cơ chế hoạt động là khi nút nhấn được tác động, điện áp tại chân GPIO chuyển từ cao xuống thấp. Sự thay đổi này kích hoạt bộ NVIC, CPU sẽ tạm dừng chương trình chính để nhảy vào hàm phục vụ ngắt (ISR) xử lý ngay yêu cầu cộng tiền ngay lập tức để tránh trường hợp người dùng bấm mà máy không nhận, kết thúc quá trình đó mới chuyển qua quá trình tiếp diễn ra tương tự với hai nút nhấn còn lại là chọn món và trả hàng. Điều này giúp hệ thống không bao giờ bỏ sót thao tác của người dùng.

2.3. Các linh kiện phần cứng:

2.3.1 Màn hình LCD 1602:

Máy bán hàng của nhóm sử dụng màn hình LCD 16x2 tức màn hình gồm 2 dòng và mỗi dòng hiển thị được 16 ký tự, màn hình LCD phục vụ biểu diễn thông tin số tiền nạp, sản phẩm chọn và trạng thái trả hàng của máy do người dùng giao tiếp vật lý với nút nhấn
Để tiết kiệm chân GPIO cho vi điều khiển STM32, LCD được kết nối ở chế độ 4-bit tức chỉ dùng các chân dữ liệu D4-D7. Vi điều khiển sẽ gửi dữ liệu 8-bit chia làm 2 lần với 4 bit cao trước, 4 bit thấp sau.

2.3.2 Động cơ giảm tốc DC:

Để máy bán hàng có thể đẩy hàng ra ngoài, nhóm sử dụng động cơ giảm tốc DC gắn với lò xo xoắn. Motor sử dụng động cơ điện một chiều gắn với một hộp số giảm tốc, nhóm chọn sử dụng động cơ giảm tốc là vì các động cơ DC thường có tốc độ rất cao (vài nghìn vòng/phút) nhưng lực kéo (mô-men xoắn) yếu. Do đó, động cơ có gắn hộp số giúp giảm tốc độ quay xuống mức phù hợp đồng thời tăng mô-men xoắn lên nhiều lần từ đó tạo lực kéo lớn để thắng được ma sát và trọng lượng của sản phẩm khi xoay lò xo để có thể đẩy được sản phẩm ra ngoài.

2.3.3 Driver động cơ L293D:

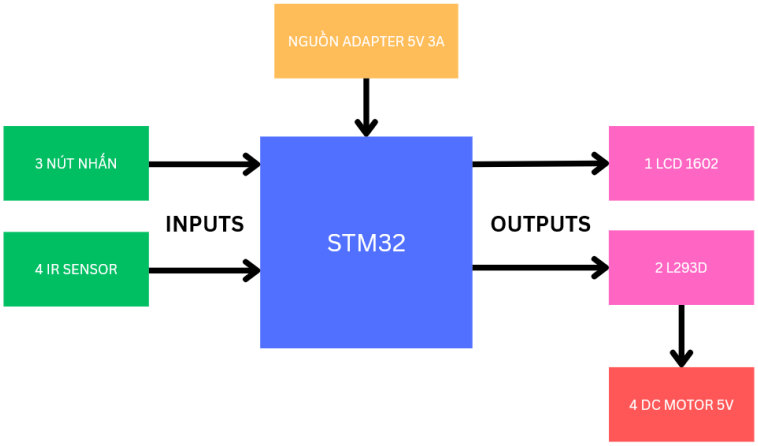
Vi điều khiển STM32 hoạt động ở mức điện áp 3.3V và dòng điện ngõ ra rất nhỏ (khoảng 20mA), vì thế không thể chạy trực tiếp động cơ DC vì cần 5V-12V và dòng hàng trăm mA đê hoạt động. Driver sẽ nhận tín hiệu điều khiển dòng nhỏ từ STM32 và đóng mở nguồn dòng lớn từ nguồn ngoài để cấp cho động cơ qua đó giúp cách ly về mặt điện áp, ngăn dòng ngược sinh ra từ cuộn dây động cơ phá hỏng vi điều khiển.

2.3.4 Cảm biến hồng ngoại (IR Sensor):

Để xác định xem sản phẩm có trong khay lấy hàng hay không, hệ thống sử dụng module cảm biến vật cản hồng ngoại. 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. Tín hiệu ngõ ra của module sử dụng IC so sánh (LM393). Khi phát hiện vật cản, chân OUT thường sẽ chuyển trạng thái (ví dụ từ mức 1 xuống mức 0 - Active LOW). Vi điều khiển đọc tín hiệu này để dừng động cơ và trừ tiền trong tài khoản.

CHƯƠNG 3: THIẾT KẾ VÀ THI CÔNG HỆ THỐNG

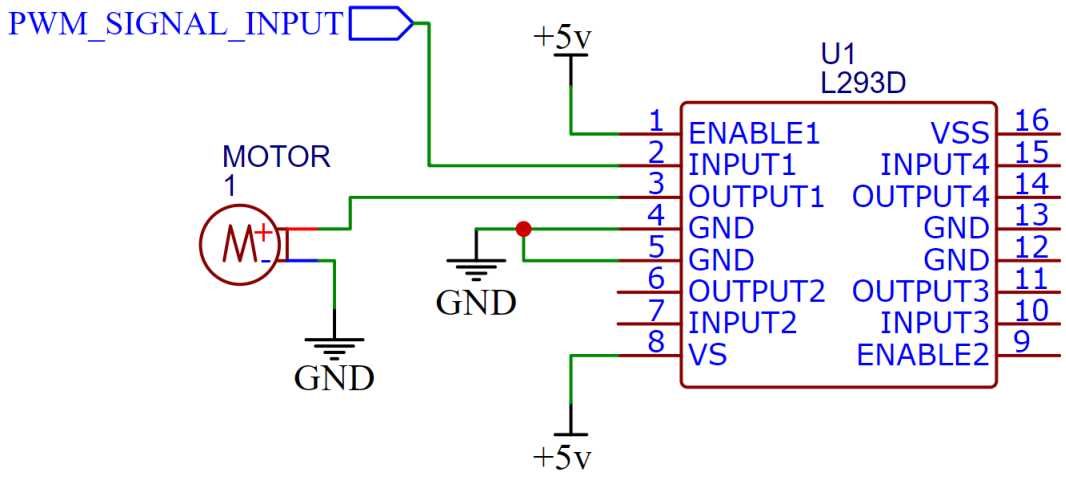
3.1. Sơ đồ khối hệ thống :



-Block diagram cho hệ thống-

3.2. Schematic cho máy bán hàng tự động:

Đây là mạch điều khiển động cơ DC sử dụng IC L293D theo cấu hình đơn hướng tức chỉ quay một chiều.



-Sơ đồ mạch kết nối của mỗi motor-

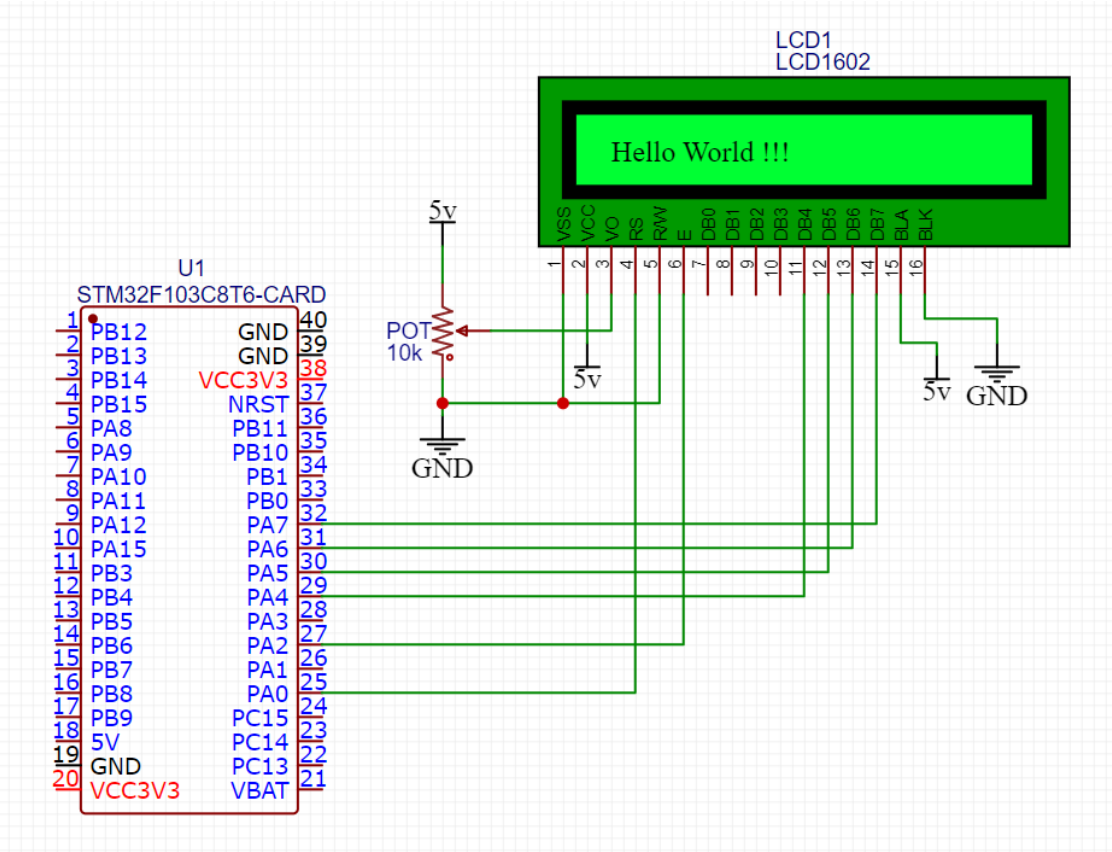
Đối với ngõ ra: Đầu ra tức chân 3 (OUTPUT1) của L293D sẽ được nối vào cực dương của motor, đây là chân cấp nguồn cho động cơ. Khi vi điều khiển điều khiển chân này lên mức cao , dòng điện sẽ chạy từ chân 3 vào động cơ. Cực âm của động cơ được nối trực tiếp xuống đất (GND) thay vì nối vào chân OUTPUT2 của IC. Cách nối này làm mạch trở thành mạch điều khiển đơn cực. Động cơ chỉ có thể quay theo một chiều duy nhất (khi OUTPUT1 có điện) hoặc dừng lại (khi OUTPUT1 mất điện).

Đối với ngõ vào: Đầu vào chân 2 (INPUT1) của L293D sẽ được nối vào PWM_SIGNAL_INPUT, đây là chân nhận tín hiệu điều khiển từ vi điều khiển STM32.Tín hiệu này là xung PWM. Tốc độ động cơ sẽ phụ thuộc vào độ rộng xung (duty cycle) của tín hiệu này. Chân 1 (ENABLE1) của L293D nối vào nguồn +5V, chân Enable sẽ đóng vai trò như công tắc tổng cho kênh 1, chân được nối cứng lên nguồn +5V nghĩa là kênh này luôn luôn được kích hoạt (Always ON). Khi đó IC luôn sẵn sàng nhận tín hiệu từ chân INPUT1 để lái động cơ.

Đối với nguồn cấp: chân 16 (VSS) sẽ được nối vào nguồn +5V, đây là nguồn nuôi cho phần logic bên trong IC để IC hiểu được tín hiệu 0V/5V từ vi điều khiển, tuy không thể hiện trong hình trên nhưng trong sơ đồ mạch đầy đủ nhóm vẫn thực hiện nối, chân 8 (VS) cũng được nối vào nguồn +5V, còn đây là nguồn cấp động lực cho động cơ tức nguồn chịu tảiLưu ý: Vì nhóm sử dụng động cơ giảm tốc với công suất nhỏ nên trong sơ đồ này, VSS và VS đang dùng chung nguồn +5V.Chân 4, 5 (GND) nối vào GND tức nối đất chung cho cả mạch điều khiển và mạch động lực.

Nguyên lý hoạt động: Khi tín hiệu PWM_INPUT_SIGNAL ở mức cao, IC L293D mở thông mạch bên trong, nối nguồn VS (Chân 8) sang OUTPUT1 (Chân 3). Dòng điện sẽ đi từ nguồn

(+5V) đến chân 8 và chân 3 và cực dương của motor, cực âm của motor nối xuống GND từ đó động cơ quay. Khi tín hiệu PWM_INPUT_SIGNAL ở mức thấp, IC ngắt điện tại OUTPUT1, không có dòng điện chạy qua động cơ từ đó động cơ dừng quay.

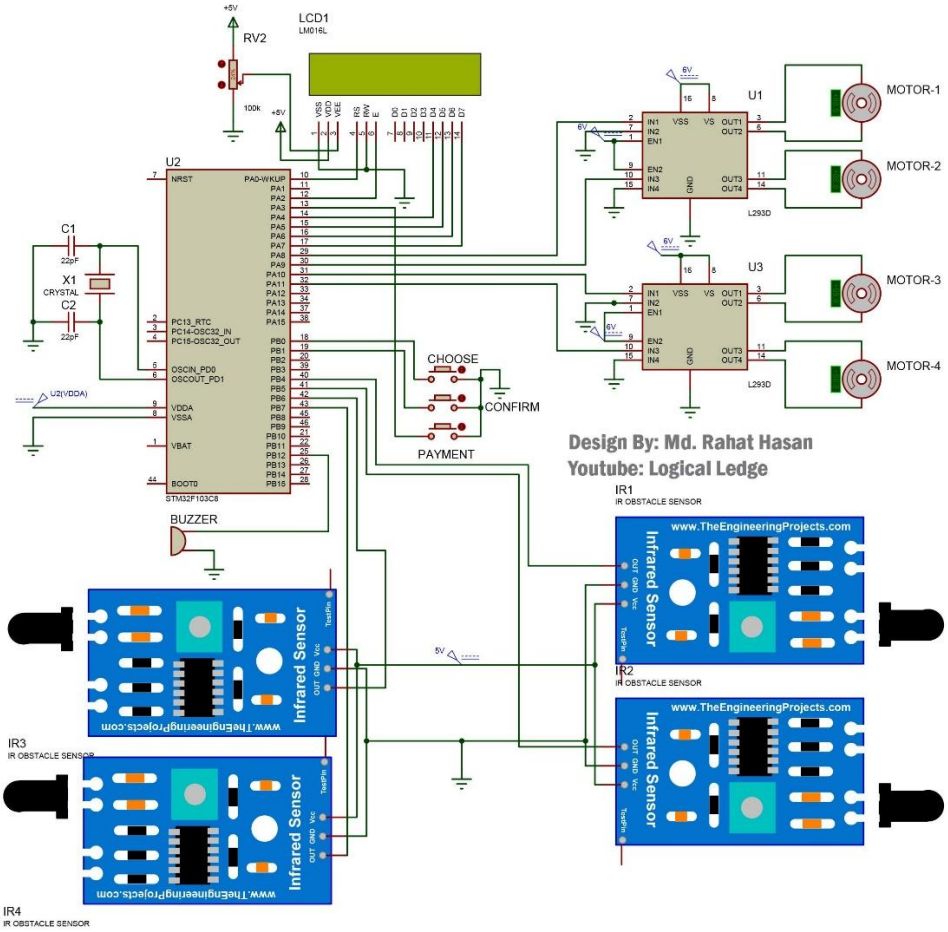


-Sơ đồ mạch kết nối vi điều khiển STM32 với LCD-

Đối với khối nguồn và màn hình : Chân 1 (VSS) nối vào GND, nối đất chung của hệ thống. Chân 2 (VCC) nối vào nguồn +5V, nguồn nuôi cho màn hình LCD. Chân 3 (VO) nối vào chân giữa biến trở 10k, hai đầu còn lại của biến trở nối 5V và GND để điều chỉnh độ tương phản của chữ. Chân 15 (BLA) nối +5V cấp nguồn cho đèn nền màn hình, tương tự chân 16 (BLK) nối vào GND cấp vào cực âm của đèn nền. Kết nối giúp đèn nền sáng liên tục .

Đối với khối điều khiển : Chân 4 (RS) nối vào PA0 để chọn thanh ghi. Chân 5 (RW) nối vào GND để cố định chế độ chỉ ghi vào LCD vì STM32 chỉ cần ghi dữ liệu của người dùng chọn ra màn hình chứ không cần đọc trạng thái từ màn hình về, việc nối đất giúp tiết kiệm được 1 chân GPIO của STM32.Chân 6 (E) nối vào PA2, chân cho phép chốt dữ liệu. Khi có một xung, LCD sẽ nhận dữ liệu đang có trên các chân data để xử lý.

Đối với khối dữ liệu : Như đã nếu ở trên, LCD chỉ sử dụng 4 chân dữ liệu cao D4-7 để giao tiếp 4-bit. Thay vì gửi 8 bit cùng lúc, STM32 sẽ chia 1 byte thành 2 phần.LCD sẽ tự động ghép lại thành 1 byte hoàn chỉnh.



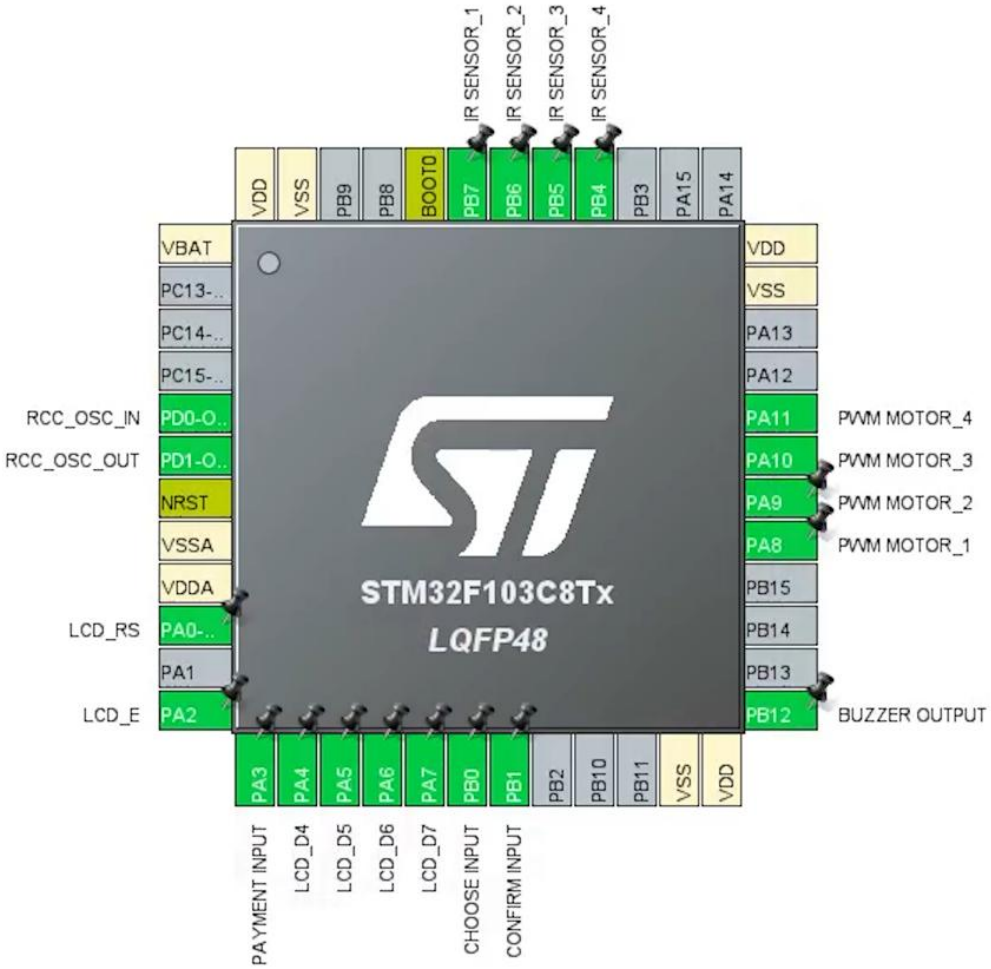
-Sơ đồ mạch đầy đủ-

Đối với khối MCU : mạch dao động sử dụng thạch anh ngoại (Crystal X1) kết hợp với 2 tụ gốm 22pF (C1, C2) để tạo dao động ổn định cho vi điều khiển hoạt động chính xác theo thời gian thực.

Đối với khối đầu vào : Để tương tác với người dùng, hệ thống sử dụng 3 nút nhấn kết nối với các chân GPIO, cụ thể là PB0, PB1 và PA3 để thực hiện các chức năng nạp tiền để thanh toán, chọn món và xác nhận, các chân GPIO này cần được cấu hình là Input Pull-up (Điện trở kéo lên). Ở trạng thái bình thường (nhả nút) → mức 1, ở rạng thái tích cực (nhấn nút) → Mức 0.

Đối với khối động cơ : hệ thống điều khiển 4 động cơ DC (tương ứng 4 lò xo đẩy hàng) thông qua 2 IC đệm dòng L293D . L293D (U1) sẽ điều khiển motor 1 và motor 2. Nhận tín hiệu điều khiển từ PA8 và PA9 tương tự với hai motor còn lại sẽ được điều khiển từ U2 và nhận tín hiệu từ PA10 và PA11

Đối với khối cảm biến : Để đảm bảo có hàng , hệ thống bố trí 4 cảm biến vật cản hồng ngoại (IR1-4) tại cửa ra của mỗi khay hàng. Các chân tín hiệu (OUT) của cảm biến được nối vào PB4-7, khi chưa có hàng cảm biến xuất mức 1, khi có hàng cảm biến phát hiện vật cản, xuất mức 0. Từ đó giúp vi điều khiển nhận biết để xác nhận giao dịch thành công và cho chạy động cơ.



-Cấu hình chân cho vi điều khiển STM32F103C8T6-

3.3. Thiết kế mô hình cơ khí (Vỏ hộp):

3.3.1 Vật liệu và Kết cấu:

- Vật liệu chính: Sử dụng bìa carton cứng. Đây là vật liệu nhẹ, chi phí thấp, dễ gia công cắt gọt và định hình, rất phù hợp cho việc xây dựng mô hình kiểm chứng nguyên lý .
- Phương pháp gia công: Cắt thủ công theo kích thước thiết kế và liên kết các chi tiết bằng keo nóng để đảm bảo độ kết dính nhanh và chắc chắn.

3.3.2 Kích thước tổng thể:

- Chiều rộng: 35 cm
- Chiều dài (sâu): 25 cm
- Chiều cao: 25 cm

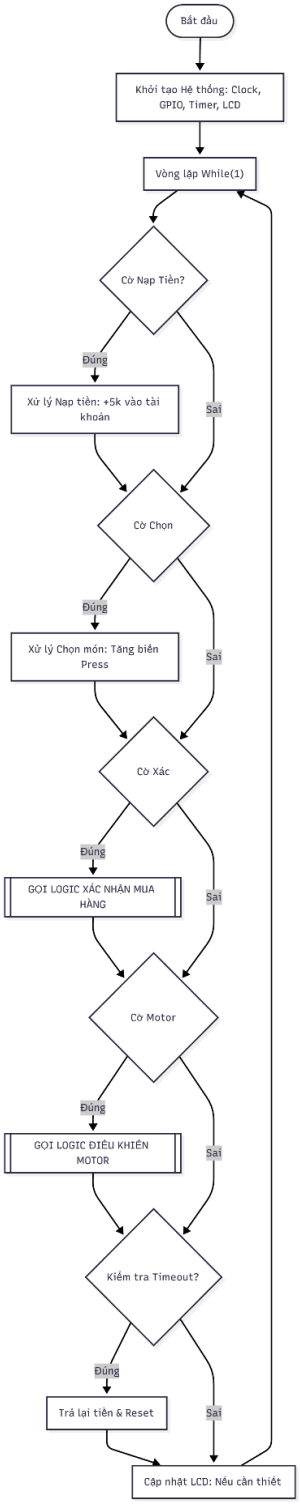
3.3.3 Bố trí linh kiện:

- Khu vực Điều khiển (Mặt trên hộp):

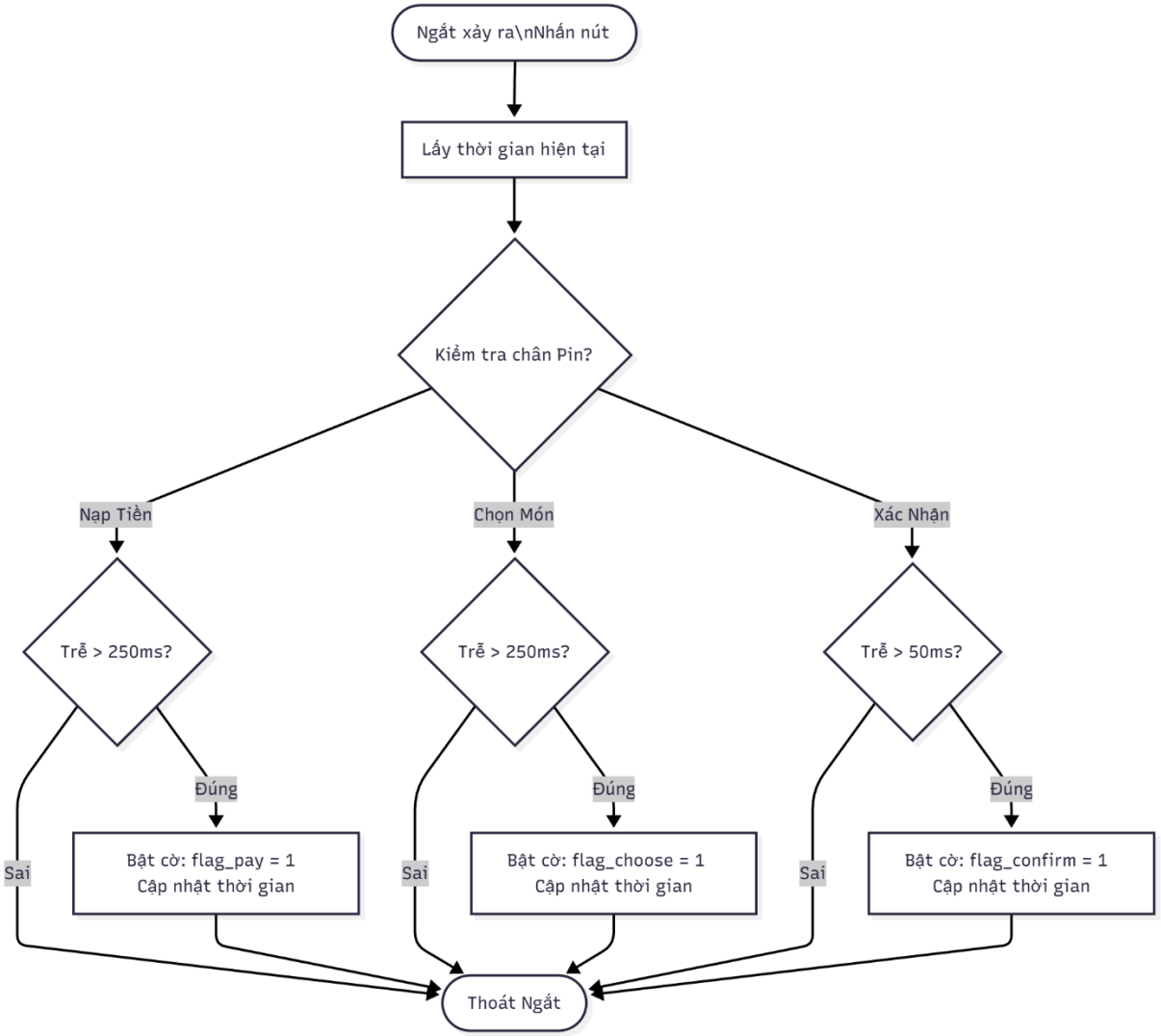
    - Đây là giao diện tương tác chính với người dùng (User Interface).
    - Bố trí: Vi điều khiển trung tâm STM32F103C8T6, Màn hình LCD 1602, 03 Nút nhấn chức năng (Nạp tiền, Chọn, Xác nhận) và Biến trở chỉnh độ tương phản. Việc đặt STM32 ở đây cũng giúp dễ dàng kết nối USB để nạp code và sửa lỗi.
- Khoang chứa hàng (Bên trong hộp):
    - Không gian bên trong được chia thành 4 luồng chứa hàng tương ứng với 4 loại sản phẩm.
    - Cơ cấu chấp hành: 4 Động cơ giảm tốc được gắn cố định vào thành hộp, trục động cơ nối với lò xo xoắn để đẩy sản phẩm.
    - Cảm biến: Tại vị trí đầu ra của mỗi lò xo, bố trí 1 cảm biến hồng ngoại (IR Sensor) hướng vào vị trí đặt sản phẩm để phát hiện trạng thái còn hàng hay hết hàng.
- Khu vực Mạch điện & Kết nối (Mặt sau hộp): Toàn bộ hệ thống dây dẫn, mạch khuếch đại công suất động cơ (Driver L293D) được đấu ra sau/ trên hộp carton.

CHƯƠNG 4: GIẢI THUẬT VÀ CHƯƠNG TRÌNH

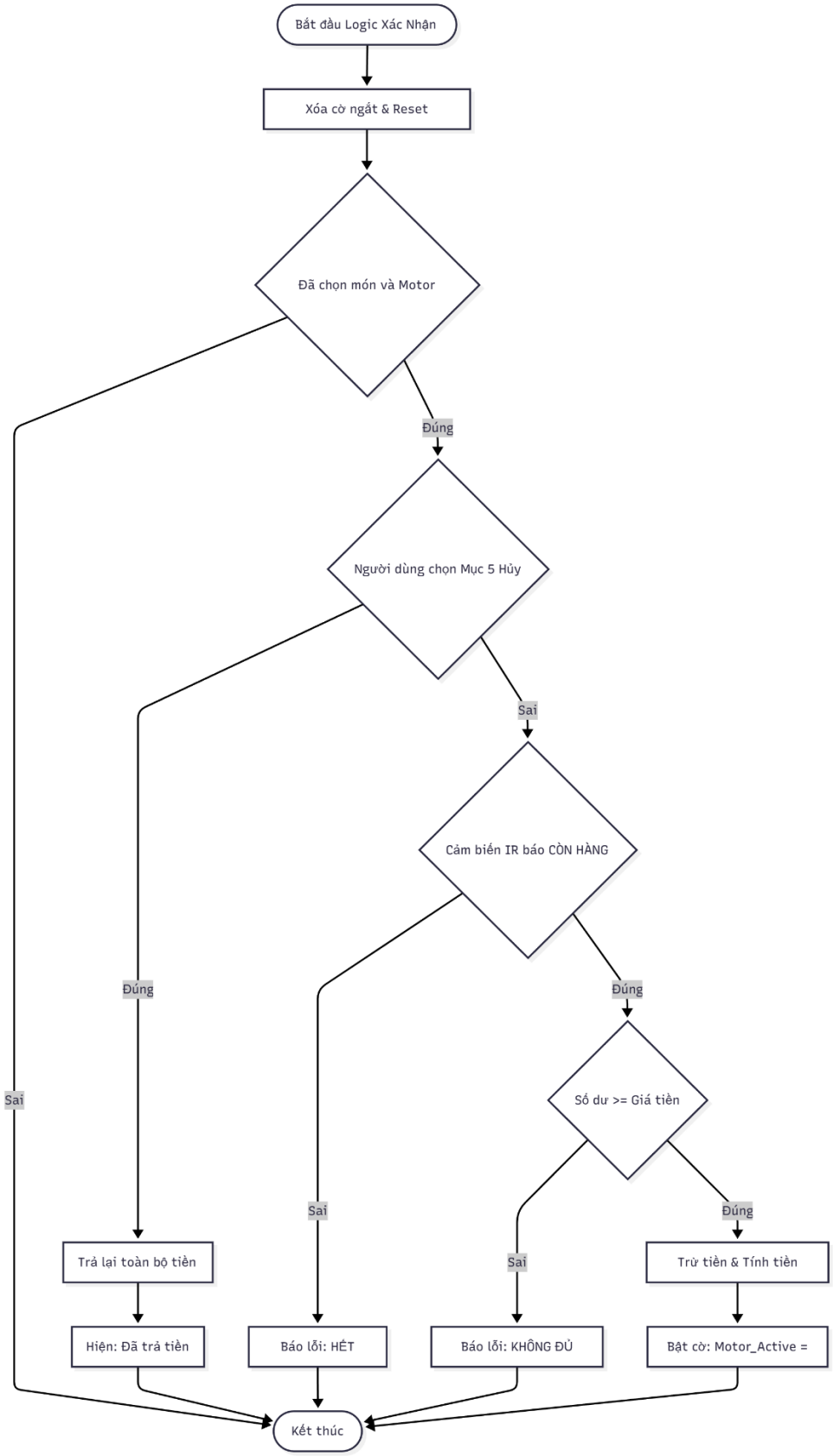
4.1. Lưu đồ giải thuật (Flowchart):



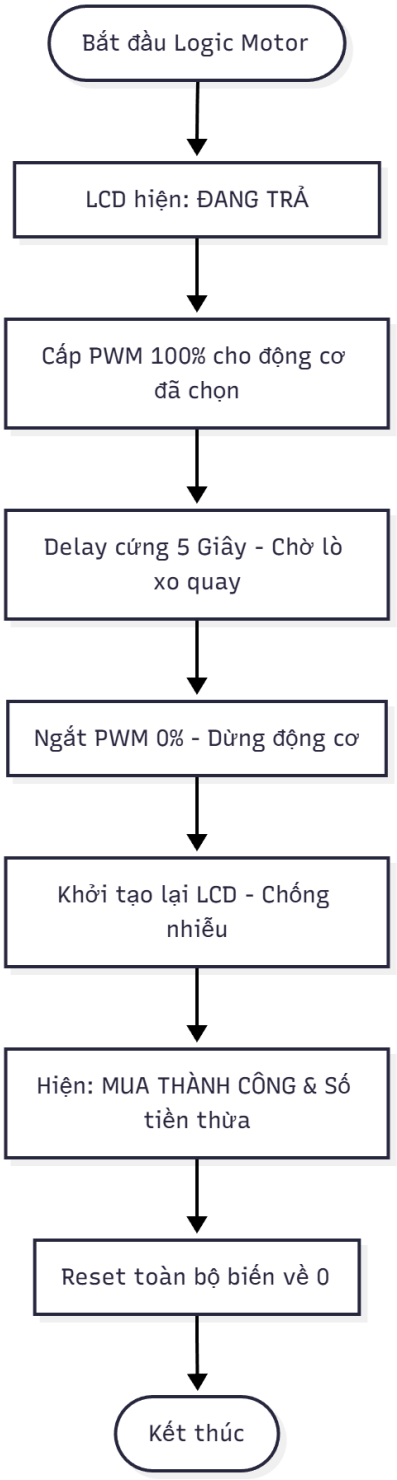
-Lưu đồ tổng quát-



-Lưu đồ giải thuật ngắt-



-Lưu đồ logic mua hàng-



-Lưu đồ điều khiển động cơ-

4.2. Giải thích mã nguồn (Source Code):

4.2.1 Module LCD:

Đây là module mô tả hoạt động của thư viện "lcd.h" được nhóm tích hợp vào module chính

```
#define RS_BIT 0 // Register select bit
#define EN_BIT 2 // Enable bit
#define BL_BIT 3 // Backlight bit
#define D4_BIT 4 // Data 4 bit
#define D5_BIT 5 // Data 5 bit
#define D6_BIT 6 // Data 6 bit
#define D7_BIT 7 // Data 7 bit

#define LCD_ROWS 2 // Number of rows on the LCD
#define LCD_COLS 16 // Number of columns on the LCD
```

Đầu tiên, module sẽ định nghĩa các chân thành tên cho gọi và dễ sửa chữa trong code hơn

```
// Define global variable for backlight state
uint8_t backlight_state = 1;

void lcd_write_nibble(uint8_t nibble, uint8_t rs) {
  uint8_t data = nibble << D4_BIT;
  data |= rs << RS_BIT;
  data |= backlight_state << BL_BIT; // Include backlight state in data
  data |= 1 << EN_BIT;
  GPIOA->ODR=data;
  HAL_Delay(1);
  data &= ~(1 << EN_BIT);
  GPIOA->ODR=data;
}
```

Tiếp theo, module sẽ dùng biến backlight_state để lưu trạng thái đèn nền của màn hình (1 là đèn bật) để tránh trường hợp màn hình LCD bị tắt do không có đèn nền. Sau đó thực hiện dịch nibble tức 4 bit dữ liệu cần gửi sang trái (<<) để đưa 4 bit dữ liệu vào đúng vị trí các chân D4-D7 . Sau đó thực hiện chèn trạng thái đèn nền, bit chọn thanh ghi vào và bit enable vào data . Sau khi enable thành công, toàn bộ dữ liệu kèm bit EN=1 được xuất ra thanh ghi ODR, sau khoảng trễ 1ms, bit EN được xóa về 0 và xuất lại ra thanh ghi. Hành động này sẽ tạo ra một cạnh tại chân enable, kích hoạt bộ điều khiển của LCD chốt dữ liệu từ các đường bus vào bộ nhớ đệm để xử lý.

```
void lcd_send_data(uint8_t data) {
  uint8_t upper_nibble = data >> 4;
  uint8_t lower_nibble = data & 0x0F;
  lcd_write_nibble(upper_nibble, 1);
  lcd_write_nibble(lower_nibble, 1);
}
```

Đoạn này thực hiện tách 8 bit thành 4 bit cao và 4 bit thấp bằng cách dịch 4 bit cao bằng phương pháp dịch bit và AND với 1. Sau đó truyền lần lượt xuống dữ liệu đã tách xuống LCD nhờ set RS = 1 theo thứ tự từ nibble cao đến thấp

```
void lcd_init() {
  HAL_Delay(50);
  lcd_write_nibble(0x03, 0);
  HAL_Delay(5);
  lcd_write_nibble(0x03, 0);
  HAL_Delay(1);
  lcd_write_nibble(0x03, 0);
  HAL_Delay(1);

  lcd_write_nibble(0x02, 0);
  lcd_send_cmd(0x28);
  lcd_send_cmd(0x0C);
  lcd_send_cmd(0x06);
  lcd_send_cmd(0x01);
  HAL_Delay(2);
}
```

Phần này để khởi tạo LCD hoạt động ổn định, dùng lệnh lcd_write_nibble(0x02, 0) để vào chế độ 4 bit, sau đó cấu hình cmd 2 dòng, ma trận điểm 5x8, bật màn hình và tắt con trỏ gạch dưới kí tự và thiết lập chế độ tự động dịch con trỏ sang phải sau mỗi lần ghi.

```
void lcd_write_string(char *str) {
  while (*str) {
    lcd_send_data(*str++);
  }
}
```

Hàm sử dụng vòng lặp để cấu hình gửi chuỗi kí tự lên LCD

```
void lcd_set_cursor(uint8_t row, uint8_t column) {
    uint8_t address;
    switch (row) {
        case 0:
            address = 0x00;
            break;
        case 1:
            address = 0x40;
            break;
        default:
            address = 0x00;
    }
    address += column;
    lcd_send_cmd(0x80 | address);
}
```

Cấu hình để định vị vị trí con trỏ tức tọa độ hàng cột của con trỏ trên LCD

```
void lcd_clear(void) {
        lcd_send_cmd(0x01);
    HAL_Delay(2);
}
```

Hàm này dùng để xóa màn hình LCD

```
void lcd_backlight(uint8_t state) {
  if (state) {
```

```
    backlight_state = 1;
  } else {
    backlight_state = 0;
  }
}
```

Và cuối cùng hàm này dùng để cập nhật trạng cho đèn nền qua biến backlight_state

4.2.2 Module chính :

```
#include "main.h"

/* Private includes ----------------------------------------------------------
---*/
/* USER CODE BEGIN Includes */
#include <stdio.h>

#include <string.h>

#include "lcd.h"

#include <stdlib.h>
```

Phần này khai báo các thư viện sử dụng với thư viện *<stdio.h>* là thư viện chuẩn nhập và xuất dữ liệu lên LCD, thư viện*<string.h>*là thư viện xử lý các chuỗi kí tự in lên LCD, thư viện *"lcd.h"* là thư viện do nhóm tạo ở module LCD trên và thư viện *<stdlib.h>*. để cung cấp các hàm chuyển đổi dữ liệu, cấp phát bộ nhớ động, hoặc tạo số ngẫu nhiên.

```
char *text = "CHOOSE PRODUCT";
char *text_cancel = "CANCEL PURCHASE?";
char *text_no_money = "NOT ENOUGH MONEY";
char *text_refund = "MONEY RETURNED";
char *text_insert_coin = "VUI LONG NAP TIEN";

int prices[4] = {5, 10, 15, 10};

// --- CAC BIEN TOAN CUC ---
volatile int total_money = 0;
volatile int press = 0;
volatile int change = 0;

// Bien dieu khien Motor
volatile int motor_active = 0;
volatile int selected_motor = 0;

// Cac "Co" (Flags) bao hieu tu ngat
volatile uint8_t flag_pay_pressed = 0;
volatile uint8_t flag_choose_pressed = 0;
volatile uint8_t flag_confirm_pressed = 0;
```

```
// Bien thoi gian
uint32_t last_activity_tick = 0;
uint32_t last_lcd_refresh = 0;

// Chong rung rieng biet
volatile uint32_t time_pay_press = 0;
volatile uint32_t time_choose_press = 0;
volatile uint32_t time_confirm_press = 0;

char lcd_line1[20];
char lcd_line2[20];
```

Đoạn này khai báo các biến cần sử dụng với char khai báo kiểu kí tự, **volatile** là khai báo biến có sử dụng ngắt hoặc thanh ghi để thay đổi, int và uint là khai báo kiểu số nguyên có dấu và không dấu.

```
// --- 1. HAM XU LY NGAT (Chi bat co) ---
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
  uint32_t current_time = HAL_GetTick();

  // Nut NAP TIEN (PA3)
  if (GPIO_Pin == PAYMENT_INPUT_Pin) {
    if ((current_time - time_pay_press) > 250) {
      flag_pay_pressed = 1;
      time_pay_press = current_time;
    }
  }
```

Nhóm thực hiện cấu hình ngắt ngoài cho GPIO_Pin bằng HAL_GPIO_EXTI_Callback tức mỗi khi nhấn thì STM32 sẽ đưa về hàm này để chạy.
Sau đó dùng uint32_t current_time = HAL_GetTick() để lấy kết quả thời gian hiện tại. Sau đó lấy kết quả thời gian này trừ cho thời gian lần nhất gần nhất nếu >250ms thì mới chấp nhận lần ấn mới và thực hiện lệnh kế tiếp vì tránh trường hợp khi ta ấn nút sẽ tạo rung phím khi cho STM32 hiểu lầm là nhiều lần nhấn, vì thế nhóm ràng buộc điều kiện thời gian giữa mỗi lần nhấn để chống rung phím khi nhấn, khi nhấn thành công cờ báo trả về 1.

```
TIM_HandleTypeDef htim1;

/* USER CODE BEGIN PV */
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------
---*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM1_Init(void);
```

Đoạn này để khai báo cấu hình xung Clock, GPIO và Timer. Ở đây sử htim1 gọi lại cấu hình timer khi cần kích xung PWM.

```
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_1);
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_2);
```

```
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_3);
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_4);
```

Đây là cách nhóm kích xung PWM trên 4 kênh của Timer1 vì nhóm sử dụng 4 động cơ để đẩy hàng

```
lcd_init();
last_activity_tick = HAL_GetTick();
last_lcd_refresh = HAL_GetTick();
```

Nhóm thực hiện lấy hai mốc thời gian lần cuối bấm nút và thời gian refresh lần cuối để phục vụ cho các mục đích đằng sau.

```
// 1. Xu ly NAP TIEN
if (flag_pay_pressed == 1) {
  flag_pay_pressed = 0;
  last_activity_tick = HAL_GetTick();

  if (press == 0 && motor_active == 0) {
    if (total_money < 100) {
      total_money += 5;
      update_lcd = 1;
    }
  }
}
```

Đầu tiên là phần xử lý nạp tiền, khi có cờ báo ấn nút nạp tiền, hệ thống sẽ cập nhật và xóa cờ hiện tại để cho các lần ấn sau, sau đó cập nhật thời gian ấn lần cuối bằng lệnh HALT, hệ thống chỉ chấp nhật lệnh nạp tiền thêm khi máy đang ở trạng thái chờ press = 0 và motor_active = 0 tức không có motor nào đang xoay. Hệ thống cho phép nạp tối đa là 100 đồng.

```
// 2. Xu ly CHOOSE
    if (flag_choose_pressed == 1) {
        flag_choose_pressed = 0;
        last_activity_tick = HAL_GetTick();

        if (motor_active == 0) {
            if (total_money == 0) {
                 lcd_clear();
                 lcd_set_cursor(0,0); lcd_write_string(text_insert_coin);
                 HAL_Delay(1000);
                 update_lcd = 1;
            } else {
                press++;
                if (press > 5) press = 1;
                update_lcd = 1;
            }
        }
    }
```

Tương tự xử lý chọn máy cũng phải ở trạng thái chờ và motor không xoay, sau đó kiểm tra người dùng đã nạp tiền chưa, nếu chưa thì thông báo dòng thông báo insert_coin trong vòng 1s để đọc rồi cập nhật lại trạng thái LCD, nếu nhấn nút cập nhật trạng thái LCD, sau lần nhấn

thứ 5 sẽ quay lại lần nhấn đầu tiên vì chỉ có 4 món hàng ứng với 4 motor nhóm đã thiết kế và chức năng cancel sản phẩm.

```
// 3. Xu ly CONFIRM
    if (flag_confirm_pressed == 1) {
        flag_confirm_pressed = 0;
        last_activity_tick = HAL_GetTick();

        if (press == 0 || motor_active == 1) {
            // Khong lam gi
        }
        else if (press == 5) { // Cancel
            change = total_money;
            total_money = 0;
            press = 0;
            lcd_clear();
            lcd_set_cursor(0,0); lcd_write_string(text_refund);
            char temp[10]; sprintf(temp, "%d", change);
            lcd_set_cursor(1,0); lcd_write_string(temp);
            HAL_Delay(2000);
            update_lcd = 1;
        }
        else if (press >= 1 && press <= 4) { // Mua SP
            int price = prices[press-1];
            int stock = 0; // Mac dinh la het hang

            // [CAP NHAT IR] Kiem tra hang theo tung san pham (IR: 0=Con,
1=Het)
            if (press == 1) {
                if (HAL_GPIO_ReadPin(IR_SENSOR_1_GPIO_Port,
IR_SENSOR_1_Pin) == GPIO_PIN_RESET) stock = 1;
            }
            else if (press == 2) {
                if (HAL_GPIO_ReadPin(IR_SENSOR_2_GPIO_Port,
IR_SENSOR_2_Pin) == GPIO_PIN_RESET) stock = 1;
            }
            else if (press == 3) {
                if (HAL_GPIO_ReadPin(IR_SENSOR_3_GPIO_Port,
IR_SENSOR_3_Pin) == GPIO_PIN_RESET) stock = 1;
            }
            else if (press == 4) {
                if (HAL_GPIO_ReadPin(IR_SENSOR_4_GPIO_Port,
IR_SENSOR_4_Pin) == GPIO_PIN_RESET) stock = 1;
            }

            if (stock == 0) {
                lcd_clear();
                lcd_set_cursor(0,0); lcd_write_string("HET HANG");
                HAL_Delay(1500);
                update_lcd = 1;
            } else if (total_money < price) {
                lcd_clear();
                lcd_set_cursor(0,0); lcd_write_string(text_no_money);
                HAL_Delay(1500);
                update_lcd = 1;
            } else {
```

```
                // === MUA THANH CONG ===
                change = total_money - price;
                total_money = 0;

                selected_motor = press;
                motor_active = 1;
                update_lcd = 0;
            }
        }
    }
```

Tương ở trên, phần này cũng kiểm tra trạng thái hệ thống và motor trước khi bắt đầu, sau đó người dùng bắt đấu nhấn, sẽ chia ra các trường hợp sau:

- Đầu tiên là press = 5 tức người dùng thực hiện cancel, máy sẽ thối lại tiền qua biến change và reset lại phím nhấn và số tiền hiện đã nạp
- Nếu press nằm trong khoảng 1 tới 4, tức người dùng đã chọn sản phẩm, hệ thống sẽ kiểm tra hàng qua biến stock (1 là có hàng) qua, biến này được set phụ thuộc vào GPIO_PIN_RESET của STM32 với cảm biến vật cản IR(0 là có hàng là set stock lên 1 và ngược lại). Sau khi đã kiểm tra có hàng, hệ thống sẽ so sánh với giá sản phẩm thông quá **if** (total_money < price), nếu giá thấp hơn giá sản phẩm sẽ thông báo cho người dùng. Sau khi có hàng và đủ tiền, hệ thống sẽ tiếp tục tính tiền thối, reset lại tổng tiền và bật tín hiệu cho motor chạy ứng với trạng thái nút ấn tương ứng.

```
// B. LOGIC MOTOR (BLOCKING MODE)
    // ====================================================
    if (motor_active == 1) {
        lcd_clear();
        lcd_set_cursor(0,0); lcd_write_string("DANG TRA HANG...");

        // Bat Motor
        if (selected_motor == 1) TIM1->CCR1 = 65535;
        else if (selected_motor == 2) TIM1->CCR2 = 65535;
        else if (selected_motor == 3) TIM1->CCR3 = 65535;
        else if (selected_motor == 4) TIM1->CCR4 = 65535;

        HAL_Delay(5000); // Chay 5 giay tuyet doi

        // Tat Motor
        TIM1->CCR1 = 0; TIM1->CCR2 = 0; TIM1->CCR3 = 0; TIM1->CCR4 = 0;

        // Khoi phuc LCD
        lcd_init();
        lcd_clear();
        lcd_set_cursor(0,0); lcd_write_string("MUA THANH CONG");
        sprintf(lcd_line2, "Change: %d", change);
        lcd_set_cursor(1,0); lcd_write_string(lcd_line2);
        HAL_Delay(3000);

        // Reset
        press = 0;
```

```
        motor_active = 0;
        selected_motor = 0;
        change = 0;
        update_lcd = 1;

        // Reset flags de tranh xung dot sau khi delay dai
        last_activity_tick = HAL_GetTick();
        flag_pay_pressed = 0; flag_choose_pressed = 0; flag_confirm_pressed
= 0;
    }
```

Sau khi cờ motor được bật, phần này sẽ xử lý tiếp hoạt động của motor. Đầu tiên hệ thống sẽ xóa màn hình LCD và hiện thông báo để người dùng biết máy đang hoạt động bằng dòng "DANG TRA HANG" lên LCD, tránh trường hợp người dùng hiểu lầm là máy treo. Sau đó hệ thống sẽ cấp điện áp lên thanh ghi TIMx->CCRx quyết định độ rộng xung của xung PWM, ở đây nhóm thực hiện kích tối đa (65535) để thắng lực ma sát lò xo. Sau đó, nhóm sử dụng hàm HAL_Delay để chỉnh thời gian chạy ra sản phẩm của motor. Sau khi hết thời gian trả hàng, hệ thống sẽ khôi phục lại LCD tránh nhiễu và cập nhật dòng "MUA THANH CONG" và số tiền thừa (change) lên LCD trong 3s tiếp theo. Sau quá trình đó, hệ thống sẽ reset lại toàn bộ hệ thống và các biến để tiếp tục cho lần mua tiếp theo.

```
// --- C. LOGIC TIMEOUT (10s) ---
    if (total_money > 0 && motor_active == 0 && (HAL_GetTick() -
last_activity_tick > 10000)) {
        change = total_money;
        total_money = 0;
        press = 0;
        lcd_clear();
        lcd_set_cursor(0,0); lcd_write_string("TIMEOUT!");
        sprintf(lcd_line2, "Tra lai: %d", change);
        lcd_set_cursor(1,0); lcd_write_string(lcd_line2);
        HAL_Delay(3000);
        update_lcd = 1;
        last_activity_tick = HAL_GetTick();
    }
```

Đoạn này xử lý time_out tránh việc hệ thống bị treo, hệ thống kiểm tra thời gian trôi qua kể từ lần thao tác cuối cùng (last_activity_tick). Nếu vượt quá 10 giây và trong tài khoản vẫn còn số dư (total_money > 0), hệ thống sẽ tự động kích hoạt quy trình hoàn tiền băng cách hủy bỏ các lựa chọn sản phẩm đang dang dở sau đó chuyển toàn bộ số dư hiện tại sang biến tiền trả lại (change).
Hiển thị thông báo "TIMEOUT" và số tiền được trả lại trên màn hình LCD trong 3 giây và reset hệ thống về trạng thái chờ ban đầu , sẵn sàng cho lượt khách hàng tiếp theo

```
// --- D. HIEN THI LCD DINH KY (500ms) ---
    if ((update_lcd || (HAL_GetTick() - last_lcd_refresh > 500)) &&
motor_active == 0) {
        lcd_clear();
        if (press == 0) {
```

```
            sprintf(lcd_line1, "Total: %d", total_money);
            lcd_set_cursor(0,0); lcd_write_string(lcd_line1);
            lcd_set_cursor(1,0); lcd_write_string(text);
        } else if (press == 5) {
            sprintf(lcd_line1, "Total: %d", total_money);
            lcd_set_cursor(0,0); lcd_write_string(lcd_line1);
            lcd_set_cursor(1,0); lcd_write_string(text_cancel);
        } else {
            sprintf(lcd_line1, "Total: %d", total_money);
            sprintf(lcd_line2, "%c: Price: %d", 'A' + (press-1),
prices[press-1]);
            lcd_set_cursor(0,0); lcd_write_string(lcd_line1);
            lcd_set_cursor(1,0); lcd_write_string(lcd_line2);
        }
        update_lcd = 0;
        last_lcd_refresh = HAL_GetTick();
    }
  }
}
```

Để đảm bảo giao diện người dùng luôn phản hồi chính xác và tránh hiện tượng nhấp nháy màn hình, nhóm thực hiện cập nhật LCD được thực hiện dựa trên cơ chế kết hợp giữa sự kiện và định kỳ

Điều kiện để cập nhật màn hình là khi có cờ update_lcd (tức khi người dùng nhấn nút) hoặc sau mỗi chu kỳ 500ms. Tuy nhiên, tác vụ này sẽ bị chặn nếu hệ thống đang trong quá trình trả hàng (motor_active == 1) để ưu tiên hiển thị trạng thái hoạt động của động cơ.

Trên LCD sẽ hiển thị như sau

- Dòng 1: Luôn hiển thị tổng số dư hiện tại trong máy (Total: …).
- Dòng 2: Thay đổi linh hoạt tùy theo biến trạng thái biến press.
    - Nếu press = 0: Hiển thị lời chào mặc định.
    - Nếu press = 1..4: Hiển thị tên sản phẩm (A, B, C,D) kèm theo đơn giá tương ứng lấy từ mảng prices[ ].
    - Nếu press = 5: Hiển thị tùy chọn hủy và hoàn tiền.

Các đoạn sau phần này là khởi tạo phần cứng cho hệ thống với các hàm khởi tạo MX_..._Init có nhiệm vụ thiết lập chế độ hoạt động cho vi điều khiển trước khi đi vào vòng lặp chính:

- Xung nhịp hệ thống: Cấu hình bộ PLL để vi điều khiển hoạt động ổn định tại tần số 72MHz.
- GPIO:
    - Các chân kết nối LCD được cấu hình là Output Push-Pull.
    - Các chân nút nhấn được cấu hình là Input Interrupt (Ngắt cạnh xuống) tích hợp điện trở Pull-up, đảm bảo bắt tín hiệu nhạy và chính xác.
    - Các chân cảm biến hồng ngoại được cấu hình là Input thường.
    - Timer: Timer 1 được khởi tạo ở chế độ PWM với độ phân giải 16-bit, cung cấp 4 kênh điều khiển độc lập cho 4 động cơ lò xo.

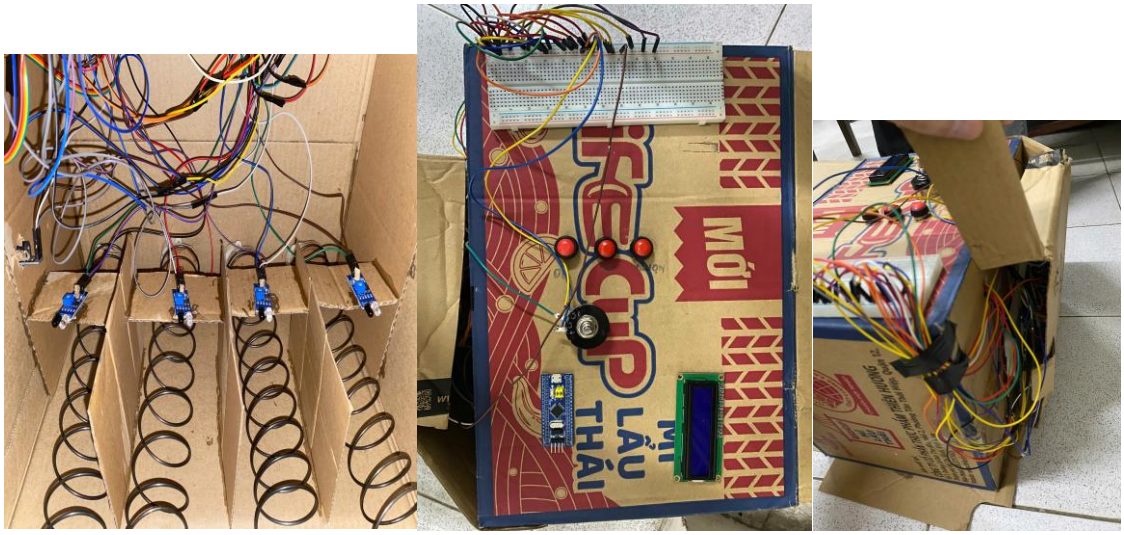
CHƯƠNG 5: KẾT QUẢ VÀ ĐÁNH GIÁ

5.1. Kết quả thi công thực tế

Sau quá trình thiết kế và lắp ráp, nhóm đã hoàn thiện mô hình máy bán hàng tự động với đầy đủ các khối chức năng theo yêu cầu.

Mô tả phần cứng hoàn thiện:

- Khung vỏ: Được gia công từ Carton, đảm bảo độ cứng vững vừa phải để chứa các linh kiện và chịu được rung động khi động cơ hoạt động.
- Bố trí mạch điện:
    - Mạch xử lý trung tâm (STM32) và mạch công suất (L293D) được bố trí riêng biệt để hạn chế nhiễu.
    - Các mối nối dây nút nhấn và cảm biến đã được gia cố và cách điện kỹ lưỡng để tránh hiện tượng tín hiệu treo (floating) hoặc chập chờn.
- Giao diện người dùng: Màn hình LCD 1602 và 3 nút nhấn (Nạp tiền, Chọn, Xác nhận) được đặt ở trên thuận tiện cho thao tác.
- Cơ cấu chấp hành: Bốn động cơ giảm tốc được gắn chặt vào khung, kết nối với lò xo để đẩy sản phẩm. Mỗi khay hàng đều có bố trí cảm biến hồng ngoại (IR) để giám sát thực tế.



-Hình ảnh sản phẩm thực tế ở ba góc độ-

5.2. Kiểm tra các test cases

Để đánh giá độ ổn định của hệ thống, nhóm đã tiến hành kiểm thử theo các kịch bản (Test Cases) bao phủ các trường hợp sử dụng thông thường và các trường hợp ngoại lệ.

Bảng 5.1: Kết quả kiểm thử chức năng

STT	Tên kịch bản (Test Case)	Thao tác thực hiện	Kết quả mong đợi	Kết quả thực tế	Đánh giá
1	Khởi động hệ thống	Cấp nguồn cho mạch.	LCD hiển thị "CHOOSE PRODUCT", Total: 0. Các biến trạng thái được reset.	LCD hiển thị đúng, hệ thống vào trạng thái chờ (Idle).	Đạt
2	Nạp tiền & Chọn món	Nhấn nút Nạp tiền 3 lần (+15) và nhấn nút Chọn sản phẩm C.	LCD cập nhật Total: 15. LCD hiển thị "SAN PHAM C" và "Price: 15".	LCD cập nhật số dư và tên sản phẩm chính xác, không bị nhấp nháy.	Đạt
3	Mua hàng thành công	(Tiếp theo case 2) Nhấn nút Xác nhận. (Điều kiện: IR không bị che - Còn hàng).	LCD báo "DANG TRA HANG...", Motor 3 quay trong 5s. Sau đó báo "MUA THANH CONG", thối lại 0.	Motor quay ổn định đủ vòng. LCD hiển thị đúng trạng thái và tiền thối.	Đạt
4	Hết hàng (IR Sensor)	Chọn Sản phẩm A. Dùng vật cản che mắt cảm biến IR 1 (hoặc để hở tùy logic IR). Nhấn Xác nhận.	LCD báo "HET HANG" hoặc "PRODUCT NOT AVAILABLE". Motor không quay, tiền không bị trừ.	Hệ thống nhận diện đúng tín hiệu từ cảm biến IR và ngăn chặn giao dịch.	Đạt
5	Không đủ tiền	Nạp 5. Chọn sản phẩm C. Nhấn Xác nhận.	LCD báo "NOT ENOUGH MONEY". Giữ nguyên số dư, không quay motor.	Cảnh báo hiển thị đúng, hệ thống bảo toàn số tiền đã nạp.	Đạt
6	Hủy giao dịch (Cancel)	Nạp 10. Nhấn nút Chọn đến mục "CANCEL PURCHASE". Nhấn Xác nhận.	LCD báo "MONEY RETURNED", hiển thị số tiền hoàn lại là 10. Lcd reset lại ban đầu.	Hệ thống hoàn trả đúng số tiền đã nạp và reset về trạng thái chờ.	Đạt
7	Tự động hủy (Timeout)	Nạp tiền nhưng không thao tác gì thêm trong vòng 10 giây.	Hệ thống tự động kết thúc phiên, báo "TIMEOUT", hoàn tiền và reset.	Đồng hồ đếm ngược (HAL_GetTick) hoạt động chính xác, tự reset sau 10s.	Đạt

STT	Tên kịch bản (Test Case)	Thao tác thực hiện	Kết quả mong đợi	Kết quả thực tế	Đánh giá
8	Kiểm tra chống rung	Nhấn nút Chọn liên tục hoặc nhấn rất nhanh.	Hệ thống chỉ nhận 1 lần nhấn cho mỗi thao tác dứt khoát, không bị nhảy cóc (ví dụ từ A nhảy sang C).	Bộ lọc chống rung (250ms) hoạt động hiệu quả, loại bỏ tín hiệu nhiễu.	Đạt

5.3. NHẬN XÉT VÀ ĐÁNH GIÁ

Dựa trên quá trình thiết kế và kết quả vận hành thực tế, nhóm đưa ra những đánh giá sau:

Ưu điểm:

- Hệ thống hoạt động ổn định: Giải quyết được triệt để vấn đề nhiễu tín hiệu (Noise) thường gặp khi kết hợp vi điều khiển với động cơ DC. Màn hình LCD không bị treo (Freeze) hay hiển thị ký tự lạ nhờ cơ chế khởi tạo lại (Re-init) và quét định kỳ.
- Thuật toán xử lý nút nhấn tối ưu: Việc sử dụng ngắt (Interrupt) kết hợp với kỹ thuật dùng cờ (Flag) và bộ đếm thời gian riêng biệt cho từng nút giúp thao tác người dùng rất mượt mà, độ trễ thấp và không bị hiện tượng "nhấn 1 nhận 2" (Double click).
- Logic nghiệp vụ chặt chẽ: Các trường hợp ngoại lệ như hết hàng, thiếu tiền, quên thao tác (timeout) đều được xử lý và thông báo rõ ràng trên màn hình, tránh gây nhầm lẫn cho người dùng.
- Thiết kế phần cứng an toàn: Việc sử dụng nguồn riêng cho động cơ và vi điều khiển (thông qua Driver L293D) giúp bảo vệ chip STM32 khỏi dòng ngược và sụt áp.

CHƯƠNG 6: KẾT LUẬN & HƯỚNG PHÁT TRIỂN

6.1 Kết luận :

Sau quá trình nghiên cứu, thiết kế và thi công, nhóm thực hiện đã hoàn thành đề tài "Thiết kế mô hình Máy bán hàng tự động sử dụng vi điều khiển STM32". Hệ thống đã đáp ứng được các mục tiêu cơ bản đề ra ban đầu:

Về mặt phần cứng:

- Xây dựng thành công mô hình cơ khí với 4 khay chứa sản phẩm hoạt động độc lập.
- Kết nối và điều khiển thành công các khối ngoại vi quan trọng: Động cơ DC Màn hình LCD 1602, Cảm biến vật cản hồng ngoại và hệ thống nút nhấn điều khiển.
- Mạch hoạt động ổn định ở mức điện áp 5V (động lực) và 3.3V (vi xử lý).

Về mặt phần mềm

- Ứng dụng thành công vi điều khiển STM32F103 trên nền tảng thư viện HAL.
- Xây dựng được giải thuật điều khiển theo mô hình Máy trạng thái (State Machine), giúp hệ thống hoạt động logic, tuần tự qua các bước: Chờ → Nạp tiền →Chọn món → Kiểm tra kho →Trả hàng.

- Xử lý hiệu quả bài toán thời gian thực: Sử dụng Ngắt (Interrupt) để bắt tín hiệu nút nhấn nhạy bén và Timer/PWM để điều khiển tốc độ động cơ chính xác.

Về tính năng:

- Máy thực hiện trơn tru quy trình bán hàng: Nhận tiền (giả lập), chọn món, kiểm tra cảm biến trả hàng và tự động tính toán trả lại tiền thừa.
- Tích hợp các tính năng an toàn: Tự động hoàn tiền (Timeout) nếu người dùng quên thao tác, và kiểm tra lỗi hết hàng bằng cảm biến.

6.2 Hướng phát triển đề tài

Do giới hạn về thời gian và kinh phí, mô hình hiện tại vẫn dừng lại ở mức độ Prototype (mẫu thử nghiệm). Để phát triển thành sản phẩm thương mại hoàn chỉnh, nhóm đề xuất các hướng nâng cấp sau:

Tích hợp Internet of Things (IoT):

- Sử dụng thêm module ESP8266/ESP32 để kết nối máy bán hàng với Internet.
- Xây dựng Web Server/Dashboard để người quản lý có thể giám sát doanh thu, số lượng hàng tồn kho và trạng thái lỗi của máy từ xa theo thời gian thực.

Thanh toán điện tử (Digital Payment):

- Thay thế việc giả lập nút nhấn nạp tiền bằng tính năng quét mã QR Code (VietQR). Hệ thống sẽ giao tiếp với API ngân hàng để xác nhận giao dịch chuyển khoản thành công trước khi nhả hàng.
- Tích hợp module RFID/NFC để hỗ trợ thanh toán bằng thẻ thành viên nội bộ (ví dụ trong trường học, công ty).

Nâng cấp giao diện người dùng (UI/UX):

- Thay thế màn hình LCD 1602 đơn sắc bằng màn hình cảm ứng màu (HMI hoặc LCD TFT) để hiển thị hình ảnh sản phẩm trực quan và sinh động hơn.
- Phát triển Mobile App cho phép người dùng đặt hàng trước trên điện thoại và đến máy quét mã để lấy hàng.

7. TÀI LIỆU THAM KHẢO

[1] Datasheet STM32F103C8T6, STMicroelectronics.

[2] Tài liệu hướng dẫn sử dụng STM32CubeIDE và thư viện HAL.

[3] Các bài giảng môn Thiết kế hệ thống nhúng - ĐHBK TP.HCM.

[4] DIY Vending Machine Using STM32 Blue Pill | STM32f103C8T6 Mircrocontroller Used | : 8 Steps - Instructables

[5] https://mecsu.vn/ho-tro-ky-thuat/cam-bien-hong-ngoai-ir-sensor-cau-tao-nguyen-ly-hoat-dong.nR0

[6] https://linhkienx.com/tin-tuc/pwm-la-gi-ung-dung-trong-dieu-khien-dong-co-dc-va-cach-chon-mach-phu-hop

[7] https://arduinokit.vn/giao-tiep-i2c-lcd-arduino/

[8] https://www.studocu.vn/vn/document/university-of-information-technology/vi-xu-ly-vi-dieu-khien/chuong-gioi-thieu-stm32/97900319