

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP.HCM
Khoa Khoa học và Kỹ thuật Máy tính



MÁY BÁN HÀNG TỰ ĐỘNG
Báo cáo Kỹ thuật

Đồ án Thiết kế Luận lý với Vi điều khiển STM32

Giảng viên hướng dẫn:
Nguyễn Thành Lộc

Nhóm tác giả:
Nguyễn Hưng Thịnh
Lê Thế Lộc
Trần Doãn Hoàng Lâm

Đơn vị:
Trường Đại học Bách Khoa TP.HCM (HCMUT)

Ngày 22 tháng 12 năm 2025

Tóm tắt nội dung

Báo cáo kỹ thuật này trình bày quá trình thiết kế và hiện thực hệ thống máy bán hàng tự động thông minh sử dụng vi điều khiển STM32F103C8T6. Hệ thống được thiết kế đặc biệt để bán các vật phẩm trong trò chơi (Trang phục đội tuyển SKT trong Liên Minh Huyền Thoại) và có các tính năng như tự động phát hiện khách hàng thông qua cảm biến siêu âm, giao diện LCD tương tác với bàn phím, quản lý kho hàng sử dụng bộ nhớ Flash, xử lý thanh toán toàn diện và chế độ quản trị bảo mật để kiểm soát kho hàng. Dự án minh họa các ứng dụng thực tế của nguyên lý hệ thống nhúng bao gồm thiết kế máy trạng thái hữu hạn, giao tiếp ngoại vi, điều khiển thời gian thực và lưu trữ dữ liệu. Việc hiện thực sử dụng thư viện STM32 HAL và bao gồm các trình điều khiển tùy chỉnh cho giao tiếp LCD I2C, quét bàn phím ma trận 4x4 và đo khoảng cách siêu âm HC-SR04. Các tính năng chính bao gồm tự động bật nguồn thông qua đo khoảng cách, xử lý giao dịch đa trạng thái với xử lý lỗi và lưu trữ kho hàng dựa trên bộ nhớ Flash.

Mục lục

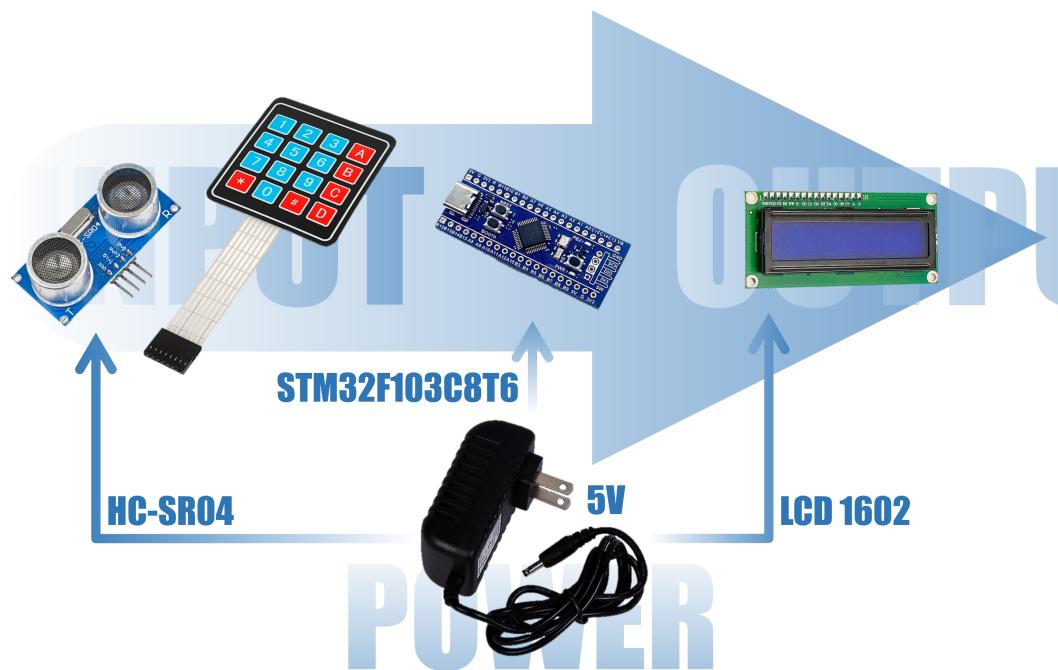
1	Tổng quan Hệ thống	5
1.1	Kiến trúc Hệ thống	5
1.1.1	Lớp Phần cứng	5
1.1.2	Lớp Trình điều khiển	6
1.1.3	Lớp Ứng dụng	6
1.1.4	Lớp Giao diện Người dùng	6
1.2	Triết lý Thiết kế	6
1.2.1	Tính Mô-đun và Phân tách Mối quan tâm	7
1.2.2	Kiến trúc Hướng Trạng thái	7
1.2.3	Vận hành An toàn	7
1.2.4	Thiết kế Lấy Người dùng làm Trung tâm	7
1.3	Yêu cầu Hệ thống	8
1.4	Luồng Vận hành Hệ thống	8
2	Thành phần Phần cứng	10
2.1	Vi điều khiển - STM32F103C8T6	10
2.1.1	Lý do Lựa chọn	10
2.1.2	Cấu hình Chân	11
2.2	Mô-đun Hiển thị - LCD 16x2 với I2C	11
2.2.1	Giao thức Giao tiếp	11
2.3	Thiết bị Đầu vào - Bàn phím Ma trận 4×4	11
2.3.1	Chiến lược Chống rung	12
2.4	Mô-đun Cảm biến - Cảm biến Siêu âm HC-SR04	12
2.4.1	Cấu hình Bắt đầu vào Bộ định thời	12
2.5	Bộ nhớ - Flash Nội bộ	12
2.5.1	Chiến lược Bảo toàn Dữ liệu	13
2.6	Nguồn điện và Giao diện Lập trình	13
2.6.1	Yêu cầu Nguồn điện	13
2.6.2	Bộ nạp ST-Link V2	13
3	Thiết kế Phần mềm	14
3.1	Kiến trúc Máy trạng thái Hữu hạn	14
3.1.1	Định nghĩa Trạng thái	14
3.1.2	Phân loại Trạng thái	14
3.1.3	Sơ đồ Chuyển đổi Trạng thái	15
3.1.4	Hiện thực FSM	16
3.2	Hệ thống Quản lý Bộ định thời	17
3.2.1	Các loại Bộ định thời	17
3.2.2	Hiện thực Bộ định thời	17
3.2.3	Thực thi Bộ định thời	17
3.2.4	Mẫu Sử dụng Bộ định thời	18
3.3	Cấu trúc Dữ liệu và Quản lý Bộ nhớ	18
3.3.1	Cấu trúc Dữ liệu Sản phẩm	18
3.3.2	Bộ cục Bộ nhớ Flash	18
3.3.3	Biến Toàn cục	19
3.4	Các Thuật toán Chính	19
3.4.1	Thuật toán Phát hiện Khách hàng	19

3.4.2	Thuật toán Xác thực Thanh toán	19
3.4.3	Thuật toán Lưu trữ Kho hàng	20
3.4.4	Thuật toán Xác thực Quản trị	21
3.5	Tổ chức Mã nguồn và Tính Mô-đun	22
3.5.1	Cấu trúc Mô-đun	22
3.5.2	Phụ thuộc Mô-đun	23
4	Hiện thực	24
4.1	Hiện thực Trình điều khiển Ngoại vi	24
4.1.1	Trình điều khiển LCD I2C	24
4.1.2	Trình điều khiển Bàn phím	26
4.1.3	Trình điều khiển Cảm biến Siêu âm	27
4.2	Chi tiết Hiện thực Máy trạng thái	27
4.2.1	Hiện thực các Trạng thái Quan trọng	27
4.3	Quản lý Hiển thị và Xử lý Lỗi	28
4.4	Tối ưu hóa Hiệu năng	28
5	Trình diễn	29
6	Đề xuất cho Công việc Tương lai	30
6.1	Màn hình Lớn hơn	30
6.2	Mở rộng Dung lượng Kho hàng	30
6.3	Hỗ trợ Da ngôn ngữ	30
7	Kết luận	31
7.1	Thành tựu Dự án	31
7.2	Ứng dụng Thực tế	31
7.3	Lời kết	32
8	Thông tin Dự án	33
8.1	Kho lưu trữ Dự án	33
8.2	Tác giả	33
8.2.1	Nhóm Dự án	33
8.2.2	Tổ chức	33
8.2.3	Thông tin Khóa học	34
8.3	Lời cảm ơn	34
8.4	Giấy phép và Sử dụng	34

1 Tổng quan Hệ thống

1.1 Kiến trúc Hệ thống

Hệ thống máy bán hàng tự động tuân theo mẫu thiết kế kiến trúc phân lớp, tách biệt phần trùu tượng hóa phần cứng, logic nghiệp vụ và giao diện người dùng. Hình 1 minh họa kiến trúc tổng thể của hệ thống.



Hình 1: Sơ đồ kiến trúc hệ thống

Kiến trúc bao gồm bốn lớp chính:

1.1.1 Lớp Phần cứng

Lớp này bao gồm tất cả các thành phần vật lý và giao diện phần cứng cấp thấp:

- **Vi điều khiển STM32F103C8T6**: Đơn vị xử lý trung tâm quản lý mọi hoạt động, chạy ở tần số 72 MHz với lõi ARM Cortex-M3
- **Màn hình LCD I2C**: Màn hình ký tự 16x2 kết nối qua bộ mở rộng I2C PCF8574 trên các chân PB6 (SCL) và PB7 (SDA)
- **Bàn phím ma trận 4x4**: Thiết bị đầu vào với các hàng trên PA0-PA3 và các cột trên PA4-PA7
- **Cảm biến siêu âm HC-SR04**: Mô-đun đo khoảng cách sử dụng bộ bắt đầu vào TIM1 để định thời chính xác

- **Bộ nhớ Flash:** Flash nội bộ 64KB với Trang 63 được dành riêng để lưu trữ dữ liệu
- **Đèn báo LED:** LED tích hợp trên PC13 để chỉ báo trạng thái hệ thống

1.1.2 Lớp Trình điều khiển

Các trình điều khiển ngoại vi tùy chỉnh cung cấp sự trừu tượng hóa phần cứng:

- **Trình điều khiển LCD I2C (tv_lcd_i2c.c):** Hiện thực giao tiếp LCD 4-bit qua giao thức I2C bit-banged
- **Trình điều khiển Bàn phím (keypad.c):** Cung cấp chức năng quét, chống rung và ánh xạ ký tự
- **Trình điều khiển Cảm biến (sensor.c):** Xử lý kích hoạt siêu âm và tính toán khoảng cách thông qua bộ bắt thời gian
- **Trình điều khiển I2C (i2c.c):** Hiện thực giao thức I2C dựa trên phần mềm để giao tiếp LCD
- **Quản lý Bộ định thời (timer.c):** Hệ thống định thời phần mềm cho các thời gian chờ và độ trễ trạng thái

1.1.3 Lớp Ứng dụng

Logic nghiệp vụ và quản lý trạng thái:

- **Máy trạng thái hữu hạn (fsm_vm.c):** Logic cốt lõi với 19 trạng thái quản lý luồng giao dịch, xử lý lỗi và chuyển đổi chế độ
- **Quản lý Kho hàng (store.c):** Xử lý cấu trúc dữ liệu sản phẩm, thao tác đọc/ghi Flash và cập nhật tồn kho
- **Mô-đun Quản trị (ADMIN.c):** Xác thực mật khẩu và quản lý bảo mật

1.1.4 Lớp Giao diện Người dùng

Quản lý trình bày và tương tác:

- Định dạng hiển thị LCD và trình bày văn bản căn giữa
- Thông dịch đầu vào bàn phím và ánh xạ lệnh
- Phản hồi trạng thái và hiển thị thông báo lỗi
- Luồng điều hướng đa màn hình

1.2 Triết lý Thiết kế

Thiết kế hệ thống tuân theo một số nguyên tắc chính:

1.2.1 Tính Mô-đun và Phân tách Mối quan tâm

Mỗi thành phần chức năng được hiện thực như một mô-đun độc lập với các giao diện được xác định rõ. Ví dụ, trình điều khiển LCD cung cấp các hàm cấp cao như `lcd_write_string()` mà không để lộ chi tiết giao tiếp I2C. Tính mô-đun này tạo điều kiện thuận lợi cho việc kiểm tra, gỡ lỗi và sửa đổi trong tương lai.

1.2.2 Kiến trúc Hướng Trạng thái

Hệ thống sử dụng máy trạng thái hữu hạn làm cấu trúc điều khiển cốt lõi, cung cấp:

- **Hành vi Dự đoán được:** Mỗi trạng thái có các điều kiện đầu vào, hành động và chuyển đổi đầu ra được xác định
- **Phục hồi Lỗi:** Các trạng thái lỗi cho phép xử lý nhẹ nhàng các đầu vào không hợp lệ và điều kiện thời gian chờ
- **Khả năng Bảo trì:** Thêm các tính năng mới đòi hỏi thêm các trạng thái và chuyển đổi mà không cần cấu trúc lại mã hiện có
- **Gỡ lỗi:** Theo dõi biến trạng thái đơn giản hóa việc phân tích hành vi hệ thống

1.2.3 Vận hành An toàn

Nhiều cơ chế an toàn đảm bảo độ tin cậy của hệ thống:

- Bộ định thời gian chờ ngăn chặn việc chờ đợi vô thời hạn (30 giây cho khách hàng, 60 giây cho quản trị viên)
- Bộ đếm lỗi giới hạn các lỗi liên tiếp (tối đa 5 lần thử)
- Kiểm tra đầu vào từ chối số lượng và số tiền thanh toán không hợp lệ
- Xác minh số ma thuật Flash ngăn chặn việc tải dữ liệu bị hỏng

1.2.4 Thiết kế Lấy Người dùng làm Trung tâm

Giao diện ưu tiên trải nghiệm người dùng thông qua:

- Điều hướng rõ ràng với các ánh xạ phím nhất quán (U/D để điều hướng, # để xác nhận, * để quay lại)
- Phản hồi ngay lập tức cho mọi hành động của người dùng
- Hiển thị văn bản căn giữa để cải thiện khả năng đọc
- Thông báo lỗi cung cấp thông tin với hướng dẫn phục hồi
- Tự động quay lại trạng thái an toàn khi hết thời gian chờ

1.3 Yêu cầu Hệ thống

- FR1: **Phát hiện Khách hàng:** Hệ thống sẽ phát hiện sự hiện diện của khách hàng trong phạm vi 20cm trong 3 giây liên tiếp và tự động bật nguồn
- FR2: **Duyệt Sản phẩm:** Người dùng sẽ điều hướng qua 16 sản phẩm bằng các phím Lên/Xuống với cập nhật hiển thị thời gian thực
- FR3: **Thông tin Sản phẩm:** Hệ thống sẽ hiển thị tên sản phẩm, tên người chơi, số lượng có sẵn và giá
- FR4: **Chọn Số lượng:** Người dùng sẽ nhập số lượng từ 1-9 đơn vị với xác thực và phản hồi lỗi
- FR5: **Xử lý Thanh toán:** Hệ thống sẽ chấp nhận các mệnh giá tiền tệ Việt Nam (5K, 10K, 20K, 50K, 100K, 200K, 500K VND) và tính toán tiền thừa
- FR6: **Quản lý Kho hàng:** Hệ thống sẽ theo dõi mức tồn kho và ngăn chặn bán hàng khi hết hàng
- FR7: **Lưu trữ Dữ liệu:** Dữ liệu kho hàng sẽ tồn tại qua các chu kỳ nguồn sử dụng bộ nhớ Flash
- FR8: **Truy cập Quản trị:** Chế độ quản trị được bảo vệ bằng mật khẩu sẽ cho phép điều chỉnh kho hàng và giá cả
- FR9: **Xử lý Lỗi:** Hệ thống sẽ xử lý các đầu vào không hợp lệ, thời gian chờ và hiển thị các thông báo lỗi thích hợp
- FR10: **Hoàn tất Giao dịch:** Khi thanh toán thành công, hệ thống sẽ cập nhật kho hàng và quay lại lựa chọn sản phẩm

1.4 Luồng Vận hành Hệ thống

Hoạt động hoàn chỉnh của hệ thống tuân theo trình tự sau:

1. Giai đoạn Khởi tạo:

- Bật nguồn và khởi tạo phần cứng
- Tải kho hàng từ bộ nhớ Flash (hoặc khởi tạo mặc định nếu là lần khởi động đầu tiên)
- Cấu hình tất cả các thiết bị ngoại vi (GPIO, I2C, Timer)
- Hiển thị màn hình khởi tạo
- Vào trạng thái nhàn rỗi với giám sát cảm biến

2. Giai đoạn Phát hiện Khách hàng:

- Quét liên tục cảm biến siêu âm mỗi 100ms
- Phát hiện sự hiện diện khi khoảng cách $\leq 20\text{cm}$ trong 3 giây
- Kích hoạt đèn nền và hiển thị thông báo chào mừng

- Chuyển sang chế độ chọn sản phẩm

3. Giai đoạn Chọn Sản phẩm:

- Hiển thị sản phẩm hiện tại (ID, tên, người chơi)
- Chấp nhận các phím Lên/Xuống để điều hướng qua 16 sản phẩm
- Phím # để xem thông tin chi tiết
- Phím R để vào chế độ quản trị (yêu cầu mật khẩu)
- Thời gian chờ không hoạt động 30 giây quay lại trạng thái nhàn rỗi

4. Giai đoạn Mua hàng:

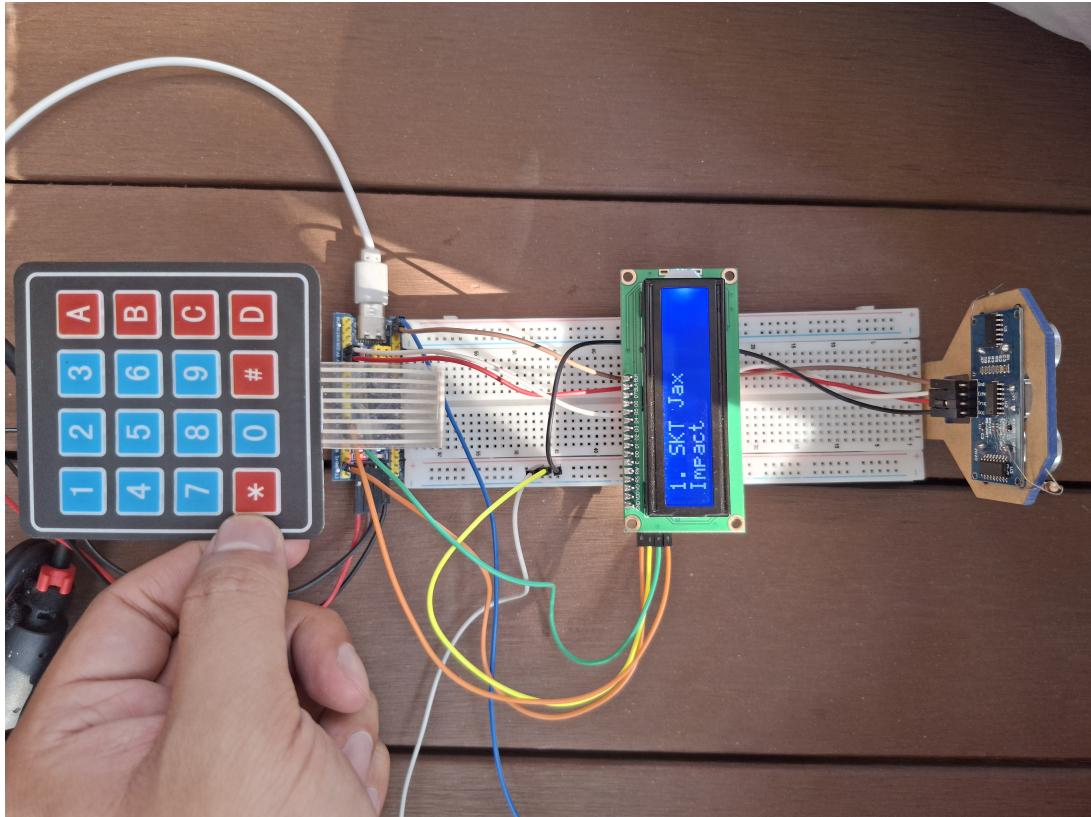
- Hiển thị chi tiết số lượng và giá
- Chấp nhận đầu vào số lượng (1-9 đơn vị)
- Xác thực tình trạng còn hàng
- Tính toán và hiển thị tổng số tiền thanh toán
- Chấp nhận đầu vào thanh toán với xác thực mệnh giá
- Tính toán tiền thừa nếu trả thừa
- Cập nhật kho hàng và lưu vào Flash

5. Giai đoạn Chế độ Quản trị:

- Xác thực với mật khẩu 6 chữ số
- Điều hướng sản phẩm để điều chỉnh
- Sửa đổi số lượng (0-9) và giá (1K-99K VND)
- Lưu thay đổi vào bộ nhớ Flash
- Thời gian chờ không hoạt động 60 giây để bảo mật

2 Thành phần Phần cứng

Phần này cung cấp thông số kỹ thuật chi tiết và chi tiết tích hợp cho tất cả các thành phần phần cứng được sử dụng trong hệ thống máy bán hàng tự động.



Hình 2: Hiện thực phần cứng của hệ thống Máy bán hàng tự động

2.1 Vi điều khiển - STM32F103C8T6

STM32F103C8T6 (thường được gọi là "Blue Pill") đóng vai trò là đơn vị xử lý trung tâm của hệ thống.

2.1.1 Lý do Lựa chọn

STM32F103C8T6 được chọn cho dự án này vì:

- **Sức mạnh Xử lý:** Xung nhịp 72 MHz cung cấp đủ hiệu năng cho các hoạt động thời gian thực
- **Bộ nhớ:** 64 KB Flash chứa được các trình điều khiển HAL và mã ứng dụng; 20 KB RAM đủ cho dữ liệu thời gian chạy
- **Hỗ trợ Ngoại vi:** Tích hợp sẵn I2C, bộ định thời và GPIO đáp ứng mọi yêu cầu giao diện
- **Hệ sinh thái Phát triển:** Hỗ trợ tuyệt vời thông qua STM32CubeIDE và thư viện HAL

- **Hiệu quả Chi phí:** Bo mạch phát triển giá rẻ có sẵn rộng rãi
- **Hỗ trợ Cộng đồng:** Cộng đồng người dùng lớn và tài liệu phong phú

2.1.2 Cấu hình Chân

Các gán chân quan trọng cho dự án này:

Bảng 1: Gán chân STM32F103C8T6

Chân	Chức năng	Mô tả
PA0-PA3	Hàng Bàn phím	Chân đầu ra để quét bàn phím
PA4-PA7	Cột Bàn phím	Chân đầu vào với điện trở kéo lên
PB6	I2C1_SCL	Xung nhịp I2C cho giao tiếp LCD
PB7	I2C1_SDA	Dữ liệu I2C cho giao tiếp LCD
PA8	TIM1_CH1	ECHO cảm biến siêu âm (Bắt đầu vào)
PA9	GPIO Output	TRIG cảm biến siêu âm
PC13	GPIO Output	Đèn báo LED (tích cực mức thấp)
PA13	SWDIO	Dữ liệu I/O Gõ lỗi Serial Wire
PA14	SWCLK	Xung nhịp Gõ lỗi Serial Wire

2.2 Mô-đun Hiển thị - LCD 16x2 với I2C

Hệ thống sử dụng màn hình LCD 16x2 kết hợp với bộ mở rộng I/O I2C PCF8574 để hiển thị thông tin.

2.2.1 Giao thức Giao tiếp

Hệ thống hiện thực giao tiếp I2C bit-banged:

1. **Điều kiện START:** SDA chuyển từ cao xuống thấp trong khi SCL ở mức cao
2. **Khung Địa chỉ:** Gửi địa chỉ slave 7-bit + bit R/W
3. **Truyền Dữ liệu:** Gửi dữ liệu 8-bit với kiểm tra ACK
4. **Điều kiện STOP:** SDA chuyển từ thấp lên cao trong khi SCL ở mức cao

Đối với hoạt động LCD ở chế độ 4-bit:

1. Gửi nibble cao (4 bit) của byte dữ liệu
2. Tạo xung chân Enable
3. Gửi nibble thấp (4 bit) của byte dữ liệu
4. Tạo xung chân Enable

2.3 Thiết bị Đầu vào - Bàn phím Ma trận 4×4

Bàn phím ma trận cung cấp 16 phím được sắp xếp thành 4 hàng và 4 cột, hoạt động dựa trên nguyên tắc quét hàng-cột.

2.3.1 Chiến lược Chống rung

Các công tắc cơ học thể hiện hiện tượng rung, gây ra nhiều chuyển đổi trong một lần nhấn. Hệ thống hiện thực chống rung bằng phần mềm:

```

1 if (HAL_GPIO_ReadPin(GPIOA, col_pins[c]) == GPIO_PIN_RESET) {
2     HAL_Delay(20); // Debounce delay
3
4     if (HAL_GPIO_ReadPin(GPIOA, col_pins[c]) == GPIO_PIN_RESET) {
5         // Key press confirmed
6         while (HAL_GPIO_ReadPin(GPIOA, col_pins[c]) ==
7             GPIO_PIN_RESET);
8         // Wait for release
9         return keymap[r][c];
10    }
11 }
```

Listing 1: Hiện thực Chống rung Bàn phím

2.4 Mô-đun Cảm biến - Cảm biến Siêu âm HC-SR04

HC-SR04 sử dụng phép đo thời gian bay (time-of-flight) để đo khoảng cách.

2.4.1 Cấu hình Bắt đầu vào Bộ định thời

Hệ thống sử dụng Kênh 1 của TIM1 được cấu hình cho chế độ Bắt đầu vào (Input Capture):

- **Chế độ Bắt:** Cả hai cạnh (lên và xuống)
- **Tần số Bộ định thời:** 1 MHz (độ phân giải 1 micro giây)
- **Ngắt:** Được kích hoạt khi có sự kiện bắt

Trình tự do:

1. Bắt lần đầu (cạnh lên): Ghi lại thời gian bắt đầu IC_Val1
2. Bắt lần hai (cạnh xuống): Ghi lại thời gian kết thúc IC_Val2
3. Tính hiệu số: $Difference = IC_Val2 - IC_Val1$
4. Xử lý tràn bộ định thời: Nếu $IC_Val1 > IC_Val2$, cộng thêm chu kỳ bộ định thời
5. Chuyển đổi sang khoảng cách: $Distance = Difference \times 0.034/2$

2.5 Bộ nhớ - Flash Nội bộ

Hệ thống sử dụng trang cuối cùng (Trang 63) của bộ nhớ Flash nội bộ để lưu trữ dữ liệu kho hàng, đảm bảo dữ liệu được bảo toàn khi mất điện.

2.5.1 Chiến lược Bảo toàn Dữ liệu

Cấu trúc dữ liệu kho hàng (44 byte mỗi sản phẩm):

- **Số Ma thuật:** 4 byte (0xDEADBEEF) để xác thực dữ liệu
- **Mảng Sản phẩm:** 16 sản phẩm × 44 byte = 704 byte
- **Tổng Lưu trữ:** 708 byte (vừa trong trang 1 KB)

Khi bật nguồn:

1. Đọc số ma thuật từ Trang 63 của Flash
2. Nếu số ma thuật khớp 0xDEADBEEF: Tải kho hàng từ Flash
3. Nếu số ma thuật không hợp lệ: Khởi tạo kho hàng mặc định và lưu vào Flash

2.6 Nguồn điện và Giao diện Lập trình

2.6.1 Yêu cầu Nguồn điện

- **Lõi STM32:** 3.3V, 50mA
- **Mô-đun LCD:** 5V, 30mA (không đèn nền), 150mA (có đèn nền)
- **Cảm biến HC-SR04:** 5V, 15mA
- **Bàn phím:** Tiêu thụ dòng không đáng kể
- **Tổng Hệ thống:** 200mA ở 5V hoạt động hiển hình

Phân phối nguồn:

- Bộ chuyển đổi ngoài 5V cung cấp cho các đường nguồn breadboard
- Bộ ổn áp 3.3V tích hợp (AMS1117-3.3) cấp nguồn cho STM32
- LCD và cảm biến kết nối trực tiếp với đường nguồn 5V

2.6.2 Bộ nạp ST-Link V2

- **Giao diện:** SWD (Serial Wire Debug)
- **Kết nối:**
 - SWDIO → PA13
 - SWCLK → PA14
 - GND → GND
 - 3.3V → 3.3V (optional for powering during debug)
- **Tính năng:** Programming, debugging, real-time variable monitoring

3 Thiết kế Phần mềm

3.1 Kiến trúc Máy trạng thái Hữu hạn

Cốt lõi của phần mềm máy bán hàng tự động là một máy trạng thái hữu hạn (FSM) với 19 trạng thái riêng biệt quản lý toàn bộ vòng đời giao dịch, xử lý lỗi và các chức năng quản trị. FSM cung cấp hành vi xác định và đơn giản hóa việc gỡ lỗi bằng cách làm cho trạng thái hệ thống rõ ràng tại mọi thời điểm.

3.1.1 Định nghĩa Trạng thái

Hệ thống hiện thực 19 trạng thái riêng biệt, được định nghĩa trong `fsm_vm.h`, bao gồm các trạng thái khởi tạo, giao dịch khách hàng và chế độ quản trị.

3.1.2 Phân loại Trạng thái

19 trạng thái được tổ chức thành các danh mục chức năng:

Trạng thái Khởi tạo (1-2):

- **INIT:** Khởi tạo hệ thống, thiết lập ngoại vi, tải kho hàng từ Flash
- **WAIT_SENSOR:** Trạng thái nhàn rỗi với giám sát cảm biến siêu âm liên tục để phát hiện khách hàng

Trạng thái Giao dịch Khách hàng (3-14):

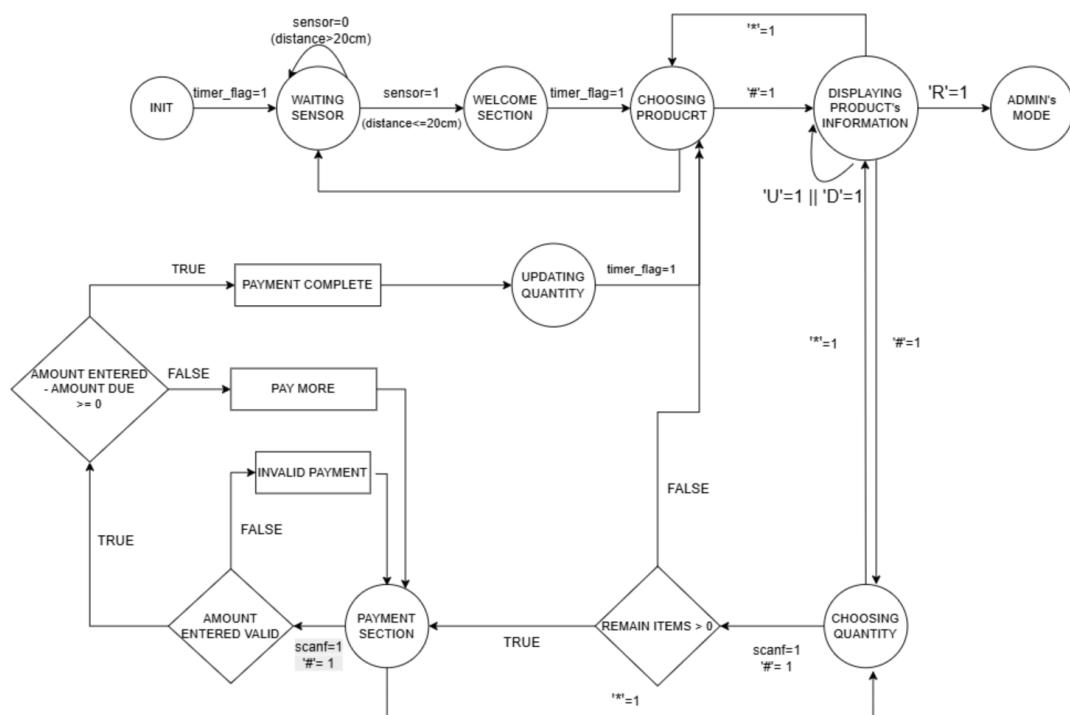
- **WELCOME_SECTION:** Hiển thị thông báo chào mừng trong 3 giây
- **CHOOSING_SKIN:** Duyệt sản phẩm với điều hướng Lên/Xuống
- **DISPLAY_INFO:** Hiển thị thông tin chi tiết sản phẩm (số lượng, giá)
- **CHOOSING_QUANTITY:** Nhập số lượng (1-9)
- **OUT_OF_STOCK_NOTIFICATION:** Cảnh báo khi sản phẩm đã chọn không có sẵn
- **QUANTITY_ERROR:** Xử lý nhập số lượng không hợp lệ với thử lại
- **MAX_ERROR_STATE:** Hủy giao dịch sau 5 lỗi liên tiếp
- **PAYMENT_SHOW_TOTAL:** Hiển thị tổng số tiền phải trả
- **PAYMENT_INPUT:** Chấp nhận các mệnh giá thanh toán
- **PAYMENT_ERROR:** Xử lý số tiền thanh toán không hợp lệ
- **PAYMENT_INFO_WAIT:** Hiển thị số tiền còn lại hoặc tiền thừa
- **THANKS:** Thông báo cảm ơn và cập nhật kho hàng

Trạng thái Chế độ Quản trị (15-19):

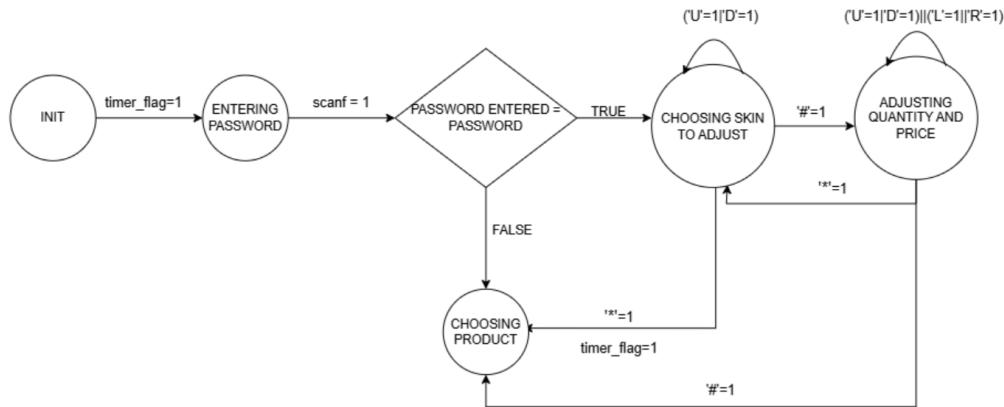
- **ADMIN_MODE:** Xác nhận đăng nhập thành công
 - **CHOOSING_SKIN_TO_ADJUST:** Chọn sản phẩm để sửa đổi
 - **TIMEOUT_ADMIN_MODE:** Tự động đăng xuất sau 60 giây không hoạt động
 - **ADJUST_QUANTITY_AND_PRICE:** Chính sửa kho hàng và giá cả
 - **CONFIRM_EXIT_ADMIN_MODE:** Hộp thoại xác nhận đăng xuất

3.1.3 Sơ đồ Chuyển đổi Trạng thái

Các chuyển đổi trạng thái tuân theo các đường dẫn chính sau:



Hình 3: Sơ đồ chuyển đổi trạng thái FSM (Chế độ Người dùng)

ADMIN's MODE

Hình 4: Sơ đồ chuyển đổi trạng thái FSM (Chế độ Quản trị)

Luồng Giao dịch Bình thường:

INIT → WAIT_SENSOR → WELCOME_SECTION → CHOOSING_SKIN
 → DISPLAY_INFO → CHOOSING_QUANTITY → PAYMENT_SHOW_TOTAL
 → PAYMENT_INPUT → PAYMENT_INFO_WAIT → THANKS → CHOOSING_SKIN

Đường dẫn Phục hồi Lỗi:

- Lỗi số lượng: CHOOSING_QUANTITY → QUANTITY_ERROR → CHOOSING_QUANTITY (hoặc MAX_ERROR_STATE sau 5 lần thử)
- Lỗi thanh toán: PAYMENT_INPUT → PAYMENT_ERROR → PAYMENT_INPUT (hoặc INIT sau 5 lần thử)
- Hết hàng: DISPLAY_INFO → OUT_OF_STOCK_NOTIFICATION → DISPLAY_INFO
- Thời gian chờ: Bất kỳ trạng thái khách hàng nào → trạng thái phục hồi thích hợp sau 30 giây

Đường dẫn Quản trị:

CHOOSING_SKIN (Phím R + mật khẩu) → ADMIN_MODE
 → CHOOSING_SKIN_TO_ADJUST → ADJUST_QUANTITY_AND_PRICE
 → CHOOSING_SKIN_TO_ADJUST hoặc CONFIRM_EXIT_ADMIN_MODE

3.1.4 Hiện thực FSM

FSM được hiện thực sử dụng cấu trúc switch-case trong `fsm_vm.c`. Mỗi trạng thái hiện thực:

1. Hành động đầu vào (cập nhật hiển thị, khởi tạo biến)
2. Xử lý đầu vào (quét bàn phím)
3. Điều kiện chuyển đổi (cờ bộ định thời, đầu vào người dùng, kết quả xác thực)
4. Hành động đầu ra (dọn dẹp, thiết lập bộ định thời)

3.2 Hệ thống Quản lý Bộ định thời

Hệ thống sử dụng cơ chế bộ định thời phần mềm để quản lý độ trễ, thời gian chờ và chuyển đổi trạng thái mà không chặn thực thi.

3.2.1 Các loại Bộ định thời

Năm loại bộ định thời độc lập được hiện thực:

Bảng 2: Các loại Bộ định thời Phần mềm

Bộ định thời	Thời lượng	Mục đích
init_timer	1000ms	Độ trễ khởi tạo trước khi kích hoạt cảm biến
welcome_timer	3000ms	Thời gian hiển thị màn hình chào mừng
timeout_timer	30s / 60s	Thời gian chờ không hoạt động (khách/admin)
message_timer	3000ms	Thời gian hiển thị thông báo tin
sensor_timer	100ms	Khoảng thời gian thăm dò cảm biến siêu âm

3.2.2 Hiện thực Bộ định thời

Mỗi bộ định thời bao gồm ba thành phần:

```

1 // Counter: Decremented each millisecond
2 int welcome_timer_counter = 0;
3
4 // Flag: Set to 1 when counter reaches 0
5 int welcome_timer_flag = 0;
6
7 // Setter function: Initialize counter and clear flag
8 void setWelcomeTimer(int duration) {
9     welcome_timer_counter = duration;
10    welcome_timer_flag = 0;
11 }
```

Listing 2: Cấu trúc Dữ liệu Bộ định thời

3.2.3 Thực thi Bộ định thời

Hàm timerRun() được gọi mỗi 1ms (thường trong ngắt SysTick):

```

1 void timerRun() {
2     if (welcome_timer_counter > 0) {
3         welcome_timer_counter--;
4         if (welcome_timer_counter <= 0) {
5             welcome_timer_flag = 1;
6         }
7 }
```

```

7     }
8     // ... repeat for other timers ...
9 }
```

Listing 3: Hàm Cập nhật Bộ định thời

3.2.4 Mẫu Sử dụng Bộ định thời

Cách sử dụng điển hình trong các trạng thái FSM:

```

1 case WELCOME_SECTION:
2     // State entry: Set timer
3     if (entered_state) {
4         setWelcomeTimer(3000);    // 3 second display
5         lcd_clear();
6         lcd_write_string("WELCOME!");
7     }
8
9     // State execution: Check flag
10    if (welcome_timer_flag == 1) {
11        status = CHOOSING_SKIN; // Transition
12        lcd_clear();
13    }
14    break;
```

Listing 4: Ví dụ Sử dụng Bộ định thời

3.3 Cấu trúc Dữ liệu và Quản lý Bộ nhớ

3.3.1 Cấu trúc Dữ liệu Sản phẩm

Hệ thống kho hàng sử dụng kiểu dữ liệu có cấu trúc cho sản phẩm:

```

1 typedef struct {
2     uint8_t id;           // Product ID (1-16)
3     char skinName[16];   // Product name (e.g., "SKT Jax")
4     char playerName[16]; // Player name (e.g., "Impact")
5     uint32_t quantity;  // Stock level (0-9)
6     uint32_t price;     // Price in VND
7 } Skin;
8
9 // Global inventory array
10 Skin skt_skins[16];
```

Listing 5: Cấu trúc Dữ liệu Sản phẩm

Dấu chân bộ nhớ: 44 byte mỗi sản phẩm \times 16 sản phẩm = 704 byte

3.3.2 Bộ cục Bộ nhớ Flash

Lưu trữ dữ liệu sử dụng bộ nhớ Flash nội bộ:

```

1 // Page 63 address: Last 1KB of 64KB Flash
2 #define FLASH_ADDR_PAGE_63 0x0800FC00
3
4 // Magic number for data validation
5 #define MAGIC_NUMBER          0xDEADBEEF
6
7 // Memory layout:
8 // Offset 0x00: Magic number (4 bytes)
9 // Offset 0x04: Skin array (704 bytes)
10 // Total: 708 bytes

```

Listing 6: Cấu hình Bộ nhớ Flash

3.3.3 Biến Toàn cục

Các biến trạng thái chính được duy trì bởi FSM:

```

1 int status = INIT;                                // Current FSM state
2 uint8_t current_id = 1;                            // Selected product ID
3 uint32_t input_quantity = 0;                      // User-entered quantity
4 uint8_t error_count = 0;                           // Consecutive error counter
5
6 uint32_t total_payable = 0;                        // Total payment required
7 uint32_t money_inserted_current = 0;               // Current denomination
    input
8 uint32_t money_paid_accumulated = 0;              // Total paid so far
9 uint8_t payment_error_count = 0;                   // Payment error counter
10
11 uint32_t detection_start_time = 0;                // Sensor detection
    timestamp

```

Listing 7: Biến Trạng thái Toàn cục

3.4 Các Thuật toán Chính

3.4.1 Thuật toán Phát hiện Khách hàng

Hệ thống giám sát liên tục cảm biến siêu âm. Nếu phát hiện vật thể trong phạm vi 20cm trong 3 giây liên tiếp, hệ thống sẽ chuyển sang trạng thái chào mừng. Cơ chế này giúp lọc nhiễu và đảm bảo sự hiện diện có chủ ý của khách hàng.

3.4.2 Thuật toán Xác thực Thanh toán

Xử lý thanh toán đa mệnh giá:

```

1 // Valid Vietnamese currency denominations
2 int is_valid_money(uint32_t amount) {
3     switch(amount) {
4         case 5000:
5         case 10000:
6         case 20000:

```

```

7     case 50000:
8     case 100000:
9     case 200000:
10    case 500000:
11        return 1;
12    default:
13        return 0;
14    }
15 }
16
17 // Payment processing
18 if (is_valid_money(money_inserted_current)) {
19     money_paid_accumulated += money_inserted_current;
20
21     if (money_paid_accumulated < total_payable) {
22         uint32_t remaining = total_payable -
23             money_paid_accumulated;
24         // Display remaining amount
25     } else {
26         uint32_t change = money_paid_accumulated - total_payable;
27         // Payment complete, display change
28         status = THANKS;
29     }
30 } else {
31     payment_error_count++;
32     if (payment_error_count >= 5) {
33         status = INIT; // Abort transaction
34     } else {
35         status = PAYMENT_ERROR; // Retry
36     }
}

```

Listing 8: Xác thực Thanh toán

3.4.3 Thuật toán Lưu trữ Kho hàng

Đọc/ghi Flash với xác thực:

```

1 void Store_SaveToFlash(void) {
2     // 1. Unlock Flash
3     HAL_FLASH_Unlock();
4
5     // 2. Erase page
6     FLASH_EraseInitTypeDef EraseInitStruct;
7     uint32_t PageError;
8     EraseInitStruct.TypeErase = FLASH_TYPEERASE_PAGES;
9     EraseInitStruct.PageAddress = FLASH_ADDR_PAGE_63;
10    EraseInitStruct.NbPages = 1;
11    HAL_FLASHEx_Erase(&EraseInitStruct, &PageError);
12
13    // 3. Write magic number
14    HAL_FLASH_Program(FLASH_TYPEPROGRAM_WORD,

```

```

15         FLASH_ADDR_PAGE_63 ,
16         MAGIC_NUMBER) ;
17
18     // 4. Write data array
19     uint32_t *pData = (uint32_t*)skt_skins;
20     uint32_t numWords = sizeof(skt_skins) / 4;
21
22     for (uint32_t i = 0; i < numWords; i++) {
23         uint32_t address = FLASH_ADDR_PAGE_63 + 4 + (i * 4);
24         HAL_FLASH_Program(FLASH_TYPEPROGRAM_WORD , address , pData[i]
25             );
26     }
27
28     // 5. Lock Flash
29     HAL_FLASH_Lock();
30 }
31
32 void init(void) {
33     uint32_t stored_magic = *(_IO uint32_t*)FLASH_ADDR_PAGE_63;
34
35     if (stored_magic == MAGIC_NUMBER) {
36         // Valid data exists: Load from Flash
37         uint32_t *pFlashData = (uint32_t*)(FLASH_ADDR_PAGE_63 + 4)
38             ;
39         uint32_t *pRamData = (uint32_t*)skt_skins;
40         uint32_t numWords = sizeof(skt_skins) / 4;
41
42         for (uint32_t i = 0; i < numWords; i++) {
43             pRamData[i] = pFlashData[i];
44         }
45     } else {
46         // First boot: Initialize defaults
47         skt_skins[0] = (Skin){1, "SKT Jax", "Impact", 9, 20000};
48         // ... initialize 15 more products ...
49         Store_SaveToFlash();
50     }
51 }
```

Listing 9: Các hoạt động Bộ nhớ Flash

3.4.4 Thuật toán Xác thực Quản trị

Xác minh mật khẩu với thời gian chờ:

```

1 const char password[] = "070596";
2 #define MAX_INPUT 6
3
4 int admin_log(void) {
5     char input[MAX_INPUT + 1] = {0};
6     int len = 0;
7     uint32_t last_tick = HAL_GetTick();
8 }
```

```

9   lcd_clear();
10  lcd_write_string("ENTER PASSWORD");
11
12  while (1) {
13      // 10-second timeout
14      if (HAL_GetTick() - last_tick > 10000) {
15          return 0; // Timeout
16      }
17
18      char c = Keypad_Scan();
19      if (c == 0) continue;
20
21      last_tick = HAL_GetTick(); // Reset timeout
22
23      if (c >= '0' && c <= '9') {
24          if (len < MAX_INPUT) {
25              input[len++] = c;
26              lcd_write_char('*'); // Masked display
27
28              if (len == MAX_INPUT) {
29                  HAL_Delay(200);
30                  return (strcmp(password, input) == 0) ? 1 : 0;
31              }
32          }
33      } else if (c == '*') { // Backspace
34          if (len > 0) {
35              len--;
36              input[len] = '\0';
37              // Clear last asterisk
38          } else {
39              return 0; // Quick exit
40          }
41      }
42
43      HAL_Delay(100); // Debouncing
44  }
45 }
```

Listing 10: Xác minh Mật khẩu Quản trị

3.5 Tổ chức Mã nguồn và Tính Mô-đun

3.5.1 Cấu trúc Mô-đun

Phần mềm tuân theo kiến trúc mô-đun với sự phân tách rõ ràng các mối quan tâm:

Bảng 3: Các Mô-đun Phần mềm

Mô-đun	Trách nhiệm
main.c	Điểm vào, khởi tạo ngoại vi, vòng lặp chính
fsm_vm.c/h	Logic máy trạng thái hữu hạn, chuyển đổi trạng thái, quy tắc nghiệp vụ
store.c/h	Quản lý dữ liệu kho hàng, hoạt động đọc/ghi Flash
keypad.c/h	Quét bàn phím ma trận, chống rung, ánh xạ ký tự
sensor.c/h	Kích hoạt cảm biến siêu âm, đo khoảng cách
tv_lcd_i2c.c/h	Điều khiển hiển thị LCD, giao tiếp I2C, định dạng văn bản
i2c.c/h	Hiện thực giao thức I2C bit-banged
timer.c/h	Quản lý bộ định thời phần mềm, xử lý thời gian chờ
ADMIN.c/h	Xác thực quản trị, xác minh mật khẩu

3.5.2 Phụ thuộc Mô-đun

Phân cấp phụ thuộc (từ trên xuống dưới):

```
main.c
  - fsm_vm.c
    - store.c (Flash operations)
    - keypad.c (user input)
    - sensor.c (customer detection)
    - tv_lcd_i2c.c (display)
      - i2c.c (communication protocol)
      - timer.c (timeouts)
    - ADMIN.c (authentication)
```

Cấu trúc phân cấp này giảm thiểu sự phụ thuộc vòng tròn và tạo điều kiện thuận lợi cho kiểm thử đơn vị.

4 Hiện thực

4.1 Hiện thực Trình điều khiển Ngoại vi

4.1.1 Trình điều khiển LCD I2C

Trình điều khiển LCD hiện thực chế độ giao tiếp 4-bit qua giao thức I2C bit-banged. Cách tiếp cận này được chọn thay vì I2C phần cứng để duy trì khả năng tương thích với các bo mạch chuyển đổi PCF8574 khác nhau và cung cấp khả năng kiểm soát thời gian tốt hơn.

Các hàm Giao thức I2C: .

```

1 void I2C_Start(void) {
2     // START condition: SDA high to low while SCL high
3     HAL_GPIO_WritePin(I2C_SDA_PORT, I2C_SDA_PIN, GPIO_PIN_SET);
4     HAL_GPIO_WritePin(I2C_SCL_PORT, I2C_SCL_PIN, GPIO_PIN_SET);
5     delay_us(5);
6     HAL_GPIO_WritePin(I2C_SDA_PORT, I2C_SDA_PIN, GPIO_PIN_RESET);
7     delay_us(5);
8     HAL_GPIO_WritePin(I2C_SCL_PORT, I2C_SCL_PIN, GPIO_PIN_RESET);
9 }
10
11 void I2C_Write(uint8_t data) {
12     for (int i = 0; i < 8; i++) {
13         // Write MSB first
14         if (data & 0x80) {
15             HAL_GPIO_WritePin(I2C_SDA_PORT, I2C_SDA_PIN,
16                               GPIO_PIN_SET);
17         } else {
18             HAL_GPIO_WritePin(I2C_SDA_PORT, I2C_SDA_PIN,
19                               GPIO_PIN_RESET);
20         }
21         delay_us(5);
22         HAL_GPIO_WritePin(I2C_SCL_PORT, I2C_SCL_PIN, GPIO_PIN_SET)
23         ;
24         delay_us(5);
25         HAL_GPIO_WritePin(I2C_SCL_PORT, I2C_SCL_PIN,
26                           GPIO_PIN_RESET);
27         data <<= 1;
28     }
29 }
```

Listing 11: Hiện thực I2C Bit-Banged

Các hàm Chế độ LCD 4-Bit: .

```

1 void LCD_Send4Bit(unsigned char Data) {
2     data_MASK &= 0x0F; // Clear upper 4 bits
3     // Map data bits to PCF8574 pins P4-P7
4     data_MASK |= (Data & 0x01) << 4;
```

```

5   data_MASK |= (Data & 0x02) << 4;
6   data_MASK |= (Data & 0x04) << 4;
7   data_MASK |= (Data & 0x08) << 4;
8 }
9
10 void LCD_Send1Byte(unsigned char byte) {
11   LCD_Send4Bit(byte >> 4); // Send upper nibble
12   LCD_Enable(); // Pulse enable
13   LCD_Send4Bit(byte); // Send lower nibble
14   LCD_Enable(); // Pulse enable
15 }
16
17 void LCD_Enable(void) {
18   data_MASK |= LCD_EN; // Set enable bit
19   for(int i=0; i<50; i++) __NOP();
20   PCD8574_write(data_MASK);
21   data_MASK &= ~LCD_EN; // Clear enable bit
22   for(int i=0; i<100; i++) __NOP();
23   PCD8574_write(data_MASK);
24 }
```

Listing 12: Các hàm Giao tiếp LCD

Trình tự Khởi tạo LCD:

```

1 void lcd_init(uint8_t addr) {
2   LCDI2C_ADDR = addr;
3
4   // Power-on delay
5   HAL_Delay(50);
6
7   // 8-bit mode initialization sequence
8   LCD_Send4Bit(0x03);
9   LCD_Enable();
10  HAL_Delay(5);
11  LCD_Enable();
12  HAL_Delay(5);
13  LCD_Enable();
14
15  // Switch to 4-bit mode
16  LCD_Send4Bit(0x02);
17  LCD_Enable();
18
19  // Function set: 4-bit, 2 lines, 5x8 font
20  LCD_Send1Byte(0x28);
21
22  // Display control: Display on, cursor off
23  LCD_Send1Byte(0x0C);
24
25  // Entry mode: Increment cursor, no shift
26  LCD_Send1Byte(0x06);
27 }
```

```

28     lcd_clear();
29 }
```

Listing 13: Khởi tạo LCD

4.1.2 Trình điều khiển Bàn phím

Trình điều khiển bàn phím hiện thực quét hàng với chốt rung:

```

1  char Keypad_Scan(void) {
2      // Key mapping array
3      static const char keymap[4][4] = {
4          {'1', '2', '3', 'U'},
5          {'4', '5', '6', 'D'},
6          {'7', '8', '9', 'L'},
7          {'*', '0', '#', 'R'}
8      };
9
10     for (int r = 0; r < 4; r++) {
11         // Set all rows HIGH
12         HAL_GPIO_WritePin(GPIOA,
13                         GPIO_PIN_0 | GPIO_PIN_1 |
14                         GPIO_PIN_2 | GPIO_PIN_3,
15                         GPIO_PIN_SET);
16
17         // Pull current row LOW
18         HAL_GPIO_WritePin(GPIOA, row_pins[r], GPIO_PIN_RESET);
19         HAL_Delay(1);
20
21         // Read all columns
22         for (int c = 0; c < 4; c++) {
23             if (HAL_GPIO_ReadPin(GPIOA, col_pins[c]) ==
24                 GPIO_PIN_RESET) {
25                 // Key pressed detected
26                 HAL_Delay(20); // Debounce delay
27
28                 // Confirm key still pressed
29                 if (HAL_GPIO_ReadPin(GPIOA, col_pins[c]) ==
30                     GPIO_PIN_RESET) {
31                     // Wait for key release
32                     while (HAL_GPIO_ReadPin(GPIOA, col_pins[c])
33                           == GPIO_PIN_RESET);
34
35                     return keymap[r][c];
36                 }
37             }
38         }
39     }
40     return 0; // No key pressed
41 }
```

Listing 14: Hiện thực Quét Bàn phím

4.1.3 Trình điều khiển Cảm biến Siêu âm

Trình điều khiển cảm biến sử dụng bộ định thời bắt tín hiệu đầu vào để đo thời gian phản hồi của xung siêu âm, từ đó tính toán khoảng cách chính xác.

4.2 Chi tiết Hiện thực Máy trạng thái

4.2.1 Hiện thực các Trạng thái Quan trọng

Trạng thái CHOOSING_SKIN: .

```

1  case CHOOSING_SKIN:
2  {
3      if (timeout_timer_flag == 1) {
4          status = INIT;
5          fsm_init();
6          break;
7      }
8
9      char key = Keypad_Scan();
10     if (key != 0) {
11         setTimeoutTimer(30000); // Reset 30s timeout
12
13        if (key == 'U') {
14            current_id--;
15            if (current_id < 1) current_id = 16; // Wrap around
16            display_current_skin(current_id);
17        }
18        else if (key == 'D') {
19            current_id++;
20            if (current_id > 16) current_id = 1; // Wrap around
21            display_current_skin(current_id);
22        }
23        else if (key == '#') {
24            status = DISPLAY_INFO;
25            lcd_clear();
26            display_skin_detail(current_id);
27        }
28        else if (key == 'R') {
29            int is_admin = admin_log();
30            if (is_admin) {
31                status = ADMIN_MODE;
32                // Display admin success message
33            }
34        }
35    }
36 }
37 break;

```

Listing 15: Trạng thái Chọn Sản phẩm

Trạng thái PAYMENT_INPUT: .

- **Trạng thái Chọn Sản phẩm:** Xử lý phím Lên/Xuống để duyệt danh sách, phím # để xem chi tiết và phím R để vào chế độ quản trị.
- **Trạng thái Thanh toán:** Cộng dồn các mệnh giá tiền được nhập, kiểm tra tính hợp lệ và tính toán tiền thừa.
- **Trạng thái Quản trị:** Cho phép điều chỉnh số lượng và giá cả của từng sản phẩm thông qua giao diện trực quan.

4.3 Quản lý Hiển thị và Xử lý Lỗi

Hệ thống sử dụng các hàm định dạng văn bản để căn giữa nội dung trên màn hình LCD 16x2. Cơ chế xử lý lỗi bao gồm xác thực đầu vào (số lượng, mệnh giá tiền) và giới hạn số lần nhập sai liên tiếp để ngăn chặn hành vi spam hoặc lỗi hệ thống.

4.4 Tối ưu hóa Hiệu năng

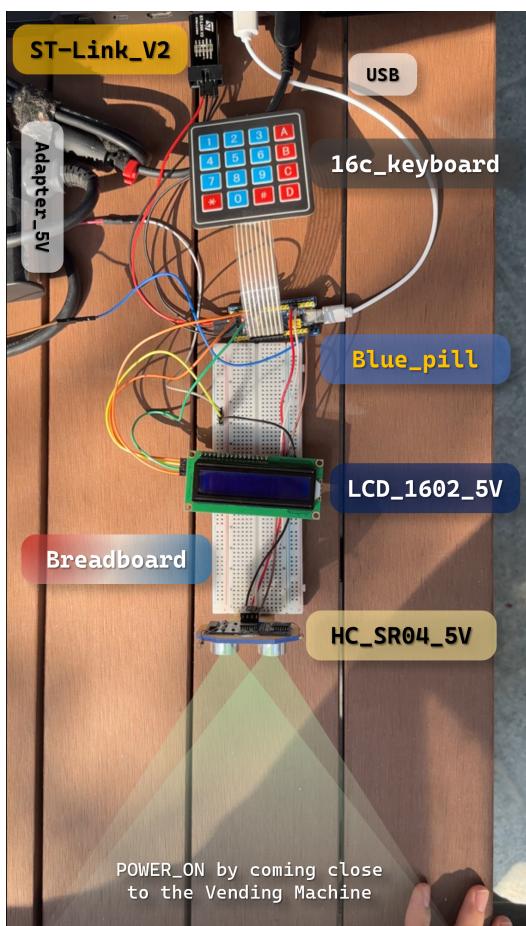
Để đảm bảo hiệu năng thời gian thực, hệ thống áp dụng các kỹ thuật tối ưu hóa như:

- Chỉ cập nhật màn hình LCD khi nội dung thay đổi.
- Sử dụng ngắt SysTick 1ms duy nhất cho tất cả các bộ định thời phần mềm.
- Giảm thiểu số lần ghi vào bộ nhớ Flash bằng cách kiểm tra sự thay đổi dữ liệu trước khi ghi.

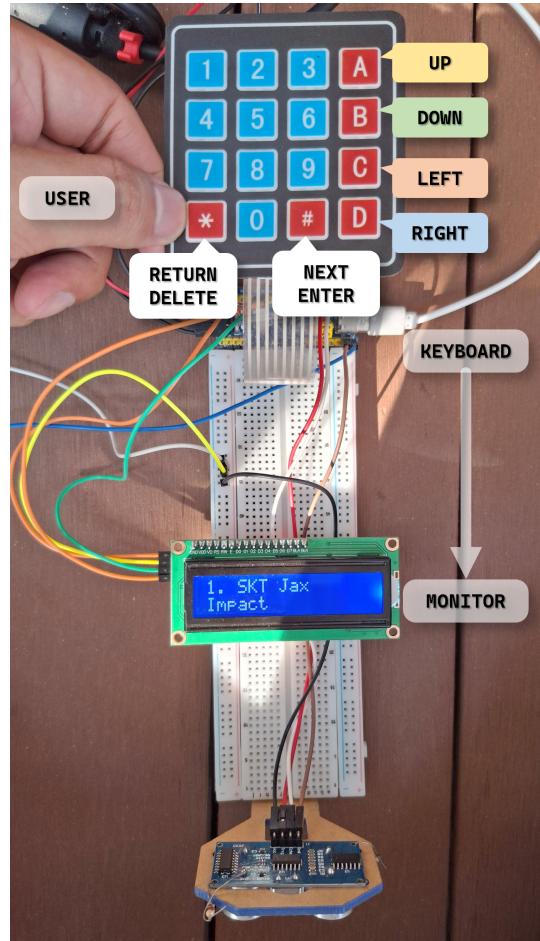
5 Trình diễn

Phần này trình bày các quy trình kiểm thử, kết quả vận hành và phân tích hiệu năng của hệ thống máy bán hàng tự động.

Xem thêm video trình diễn tại: <https://youtu.be/EFUxOFmGWq4?si=UAjL1C0VcYWAQK1N>



(a) Demo vận hành hệ thống - Phần 1



(b) Demo vận hành hệ thống - Phần 2

Hình 5: Trình diễn vận hành Máy bán hàng tự động

Tất cả các đường dẫn lỗi được kiểm thử thành công:

- Đầu vào số lượng không hợp lệ bị từ chối chính xác và nhắc thử lại
- Số tiền thanh toán không hợp lệ kích hoạt thông báo lỗi và cho phép sửa chữa
- Điều kiện hết hàng ngăn chặn mua hàng và hướng dẫn người dùng đến các lựa chọn thay thế
- Các kịch bản thời gian chờ đưa hệ thống về trạng thái an toàn
- Chế độ quản trị tự động đăng xuất hoạt động chính xác sau 60 giây không hoạt động
- Thực thi giới hạn 5 lỗi ngăn chặn vòng lặp thử lại vô hạn

6 Đề xuất cho Công việc Tương lai

While the current implementation successfully demonstrates embedded vending machine concepts, several enhancements could expand functionality and commercial viability.

6.1 Màn hình Lớn hơn

Trạng thái Hiện tại: LCD ký tự 16×2 giới hạn hiển thị thông tin.

Cải tiến Đề xuất:

- Nâng cấp lên LCD đồ họa (128×64 hoặc 320×240 pixel)
- Màn hình màu TFT với giao diện cảm ứng
- Hiển thị hình ảnh sản phẩm, thông tin dinh dưỡng, khuyến mãi
- Hỗ trợ đa ngôn ngữ với các biểu tượng đồ họa

Lợi ích: Cải thiện trải nghiệm người dùng, giảm thời gian làm quen, trình bày thương hiệu tốt hơn, tính năng hỗ trợ truy cập.

6.2 Mở rộng Dung lượng Kho hàng

Trạng thái Hiện tại: 16 sản phẩm bị giới hạn bởi điều hướng hiển thị và bộ nhớ.

Cải tiến Đề xuất:

- EEPROM ngoài (I2C/SPI) cho cơ sở dữ liệu kho hàng mở rộng
- Hỗ trợ 100+ sản phẩm với điều hướng dựa trên danh mục
- Lưu trữ thêm siêu dữ liệu sản phẩm (mô tả, hình ảnh, ngày hết hạn)
- Hiện thực truy vấn kiểu cơ sở dữ liệu cho tìm kiếm sản phẩm

Cách tiếp cận Kỹ thuật: AT24C256 (32KB EEPROM) hoặc lớn hơn, cấu trúc menu phân cấp, hệ thống phân loại sản phẩm.

6.3 Hỗ trợ Đa ngôn ngữ

Trạng thái Hiện tại: Giao diện chỉ tiếng Anh.

Cải tiến Đề xuất:

- Tùy chọn tiếng Việt, tiếng Anh và các ngôn ngữ khác
- Menu chọn ngôn ngữ khi khởi động hoặc qua chế độ quản trị
- Cấu trúc bảng chuỗi cho quản lý dịch thuật hiệu quả
- Hỗ trợ Unicode cho các ký tự không phải ASCII

7 Kết luận

7.1 Thành tựu Dự án

Dự án này đã thiết kế và hiện thực thành công một hệ thống máy bán hàng tự động toàn diện sử dụng vi điều khiển STM32F103C8T6. Hệ thống hoàn chỉnh thể hiện ứng dụng thực tế của các khái niệm hệ thống nhúng và đạt được tất cả các mục tiêu chính:

- **Tích hợp Phần cứng Hoàn chỉnh:** Giao tiếp thành công vi điều khiển với màn hình LCD, bàn phím ma trận, cảm biến siêu âm và bộ nhớ Flash, tạo ra một hệ thống nhúng đầy đủ chức năng.
- **Hiện thực Máy trạng thái Mạnh mẽ:** Phát triển một FSM 19 trạng thái quản lý các luồng giao dịch phức tạp, xử lý lỗi và các chức năng quản trị với hành vi xác định.
- **Lưu trữ Dữ liệu Bền vững:** Hiện thực lưu trữ không bay hơi sử dụng bộ nhớ Flash nội với xác thực số ma thuật, đảm bảo dữ liệu kho hàng tồn tại qua các chu kỳ nguồn.
- **Trình điều khiển Ngoại vi Tùy chỉnh:** Tạo các trình điều khiển hiệu quả cho giao tiếp LCD I2C bit-banged, quét bàn phím chống rung và đo khoảng cách siêu âm dựa trên bộ định thời.
- **Xử lý Lỗi Toàn diện:** Hiện thực xác thực đầu vào, quản lý thời gian chờ, bộ đếm lỗi và cơ chế phục hồi đảm bảo độ tin cậy của hệ thống và hoạt động thân thiện với người dùng.
- **Tự động Phát hiện Khách hàng:** Đạt được hoạt động tự chủ với phát hiện sự hiện diện của khách hàng dựa trên cảm biến siêu âm và lọc nhiễu 3 giây.
- **Quản trị An toàn:** Cung cấp chế độ quản trị được bảo vệ bằng mật khẩu với bảo mật dựa trên thời gian chờ cho quản lý kho hàng.

7.2 Ứng dụng Thực tế

Mặc dù được phát triển như một dự án giáo dục, hệ thống thể hiện các khái niệm áp dụng cho máy bán hàng tự động thương mại và tự động hóa bán lẻ:

- Tự động phát hiện khách hàng giảm tiêu thụ năng lượng
- Theo dõi kho hàng cho phép trí tuệ kinh doanh
- Xử lý lỗi đảm bảo hoạt động liên tục
- Chế độ quản trị hỗ trợ bảo trì tại hiện trường
- Thiết kế mô-đun tạo thuận lợi cho mở rộng tính năng

Kiến trúc có thể được thích ứng cho các kịch bản bán lẻ tự động khác nhau bao gồm máy bán hàng truyền thống, ki-ốt bán vé, hệ thống cho thuê và tủ khóa thông minh.

7.3 Lời kết

Dự án máy bán hàng tự động đã đạt được thành công các mục tiêu thiết kế và hiện thực một hệ thống nhúng toàn diện thể hiện tích hợp phần cứng, điều khiển máy trạng thái, lưu trữ dữ liệu bền vững và tương tác người dùng. Hệ thống hoạt động tin cậy, xử lý lỗi một cách duyên dáng và cung cấp giao diện trực quan cho cả khách hàng và quản trị viên.

Ngoài các thành tựu kỹ thuật, dự án đã cung cấp kinh nghiệm quý báu về phương pháp thiết kế hệ thống nhúng, tích hợp phần cứng-phần mềm và giải quyết vấn đề dưới các ràng buộc tài nguyên. Kiến trúc mô-đun và tài liệu kỹ lưỡng đảm bảo hệ thống có thể phục vụ như một nền tảng cho các cải tiến trong tương lai và mục đích giáo dục.

Các kỹ năng phát triển qua dự án này—lập trình vi điều khiển, giao tiếp ngoại vi, thiết kế hệ thống thời gian thực và gỡ lỗi có hệ thống—có thể áp dụng trực tiếp cho phát triển hệ thống nhúng chuyên nghiệp trong tự động hóa công nghiệp, điện tử tiêu dùng và các ứng dụng IoT.

8 Thông tin Dự án

8.1 Kho lưu trữ Dự án

Mã nguồn dự án, tài liệu và tài nguyên có sẵn tại:

- **Kho lưu trữ GitHub:** <https://github.com/1172005thinh/VendingMachine>
- **Cấu trúc Dự án:** Thư mục VendingMachine/ chứa tất cả các tệp nguồn
- **Tài liệu:** README.md với hướng dẫn thiết lập

8.2 Tác giả

8.2.1 Nhóm Dự án

Dự án này được phát triển bởi ba sinh viên từ Trường Đại học Bách Khoa - ĐHQG-HCM (HCMUT):

Tên	Đóng góp
Nguyễn Hưng Thịnh	<ul style="list-style-type: none"> • Tích hợp phần cứng và thiết kế mạch • Kiểm thử và gỡ lỗi hệ thống • Chuẩn bị tài liệu • Quan lý dự án và phối hợp nhóm • Quay video trình diễn và soạn thảo báo cáo cuối cùng
Lê Thế Lộc	<ul style="list-style-type: none"> • Phát triển trình điều khiển bàn phím • Hệ thống quản lý kho hàng • Các hoạt động bộ nhớ Flash • Thiết kế kiến trúc hệ thống • Phát triển trình điều khiển cảm biến
Trần Doãn Hoàng Lâm	<ul style="list-style-type: none"> • Thiết kế và hiện thực máy trạng thái hữu hạn • Hiện thực hệ thống bộ định thời • Thiết kế giao diện người dùng • Thực hiện, soạn kịch bản trình diễn

8.2.2 Tổ chức

Trường Đại học Bách Khoa - ĐHQG-HCM (HCMUT)

- Khoa Khoa học và Kỹ thuật Máy tính
- Bộ môn Đồ án Thiết kế Luận lý
- Địa chỉ: 268 Lý Thường Kiệt, Quận 10, Thành phố Hồ Chí Minh, Việt Nam
- Website: <https://www.hcmut.edu.vn/>

8.2.3 Thông tin Khóa học

- **Khóa học:** Đồ án Thiết kế Luận lý
- **Năm học:** 2025 - 2026
- **Thời gian Dự án:** Tháng 09/2025 - Tháng 12/2025
- **Ngày Hoàn thành:** Ngày 22 tháng 12 năm 2025

8.3 Lời cảm ơn

Các tác giả xin bày tỏ lòng biết ơn đến:

- Các cố vấn khoa về hướng dẫn các nguyên tắc thiết kế hệ thống luân lý
- Cộng đồng mã nguồn mở về các ví dụ mã và hỗ trợ khắc phục sự cố

8.4 Giấy phép và Sử dụng

Dự án này được phát triển cho mục đích giáo dục như một phần của khóa học tại HCMUT. Mã nguồn và tài liệu được cung cấp cho:

- Tham khảo giáo dục và học tập
- Nghiên cứu và học tập học thuật
- Trình diễn hệ thống nhúng phi thương mại
- Phát triển và cải tiến thêm

Đối với các ứng dụng thương mại hoặc các tác phẩm phái sinh, yêu cầu ghi công thích hợp cho các tác giả gốc và tổ chức.

Báo cáo này được biên soạn vào Ngày 22 tháng 12 năm 2025
Trường Đại học Bách Khoa - DHQG-HCM
Hệ thống Máy bán hàng Tự động - Đồ án Thiết kế Luận lý
