HỌC VIỆN KỸ THUẬT MẬT MÃ

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP MÔN LẬP TRÌNH DRIVER

LẬP TRÌNH DRIVER I2C CHO MÀN HÌNH LCD 1602 TRÊN LINUX

Khoa: Công nghệ thông tin

Chuyên ngành: Kỹ thuật phần mềm nhúng và di động

Người hướng dẫn

Triệu Vũ Anh Quân

Học viện Kỹ thuật mật mã

Nhóm sinh viên thực hiện:

Phạm Văn Dũng	CT040308
Phạm Thị Phương Anh	CT040401
Lại Phương Thảo	CT040445
Lại Phương Thảo	CT040445

Nhóm 02m

Hà Nội - 2023

MỤC LỤC

DANH SÁCH HÌNH VẼ

DANH SÁCH BẢNG

DANH MỤC TỪ VIẾT TẮT

ADC Analog to Digital Converter. 4

DMA Direct Memory Accecss. 4, 9, 10

I^2C Inter-Integrated Circuit. i, ii, 4, 5, 9, 10, 13, 14

SPI Serial Peripheral Interface. 4, 5

UART Universal Asynchronous Receiver-Transmitter. i, ii, 4, 5, 7, 8, 9, 14

USART Universal Synchronous Asynchronous Receiver-Transmitter. ii, 4, 8

USB Universal Serial Bus. 5, 7, 9

VĐK vi điều khiển. i, iii, 1, 2, 3, 4, 5, 6, 7

LỜI NÓI ĐẦU

Các hệ thống nhúng đã hiện diện trong hầu hết các hoạt động của con người trong cuộc sống ngày nay, từ các thiết bị điện đơn giản đến các thiết bị phức tạp. Để phát triển các hệ thống nhúng như vậy yêu cầu sự hỗ trợ nhất định phần cứng. Dòng vi điều khiển STM32 sử dụng lõi ARM đã và đang được sử dụng rộng rãi trong việc phát triển các hệ thống nhúng nói trên do hỗ trợ nhiều loại ngoại vi thông dụng như GPIO, TWI, SPI, ADC và có nhiều thư viện hỗ trợ từ ARM Holdings cũng như ST Microelectronics.

Với mong muốn tìm hiểu sâu hơn về hệ sinh thái ARM và dòng vi điều khiển STM32 dựa trên lõi ARM Cortex M3, nhóm chúng em quyết định thực hiện bài tập lớn: "**Lập trình vi xử lý bằng chip STM32F103C8T6 theo yêu cầu: Lập trình giao tiếp UART để Viết ứng dụng điều khiển thực hiện gửi chuỗi ký tự "Ho va ten cua sinh vien"lên PC, sau đó nhận ký tự được gửi từ PC rồi hiển thị ký tự nhận được lên LCD1602**". Báo cáo bao gồm các chương và bố cục sau:

Chương 1: Cơ sở lý thuyết

Chương 2: Thực hiện đề tài

Chương 3: Kết luận

Hà Nội, ngày 06 tháng 6 năm 2022

Nhóm sinh viên thực hiện

Chương 1: CƠ SỞ LÝ THUYẾT

1.1 Tổng quan về vi xử lý ARM và vi điều khiển (VĐK) STM32

1.1.1 Vi xử lý ARM

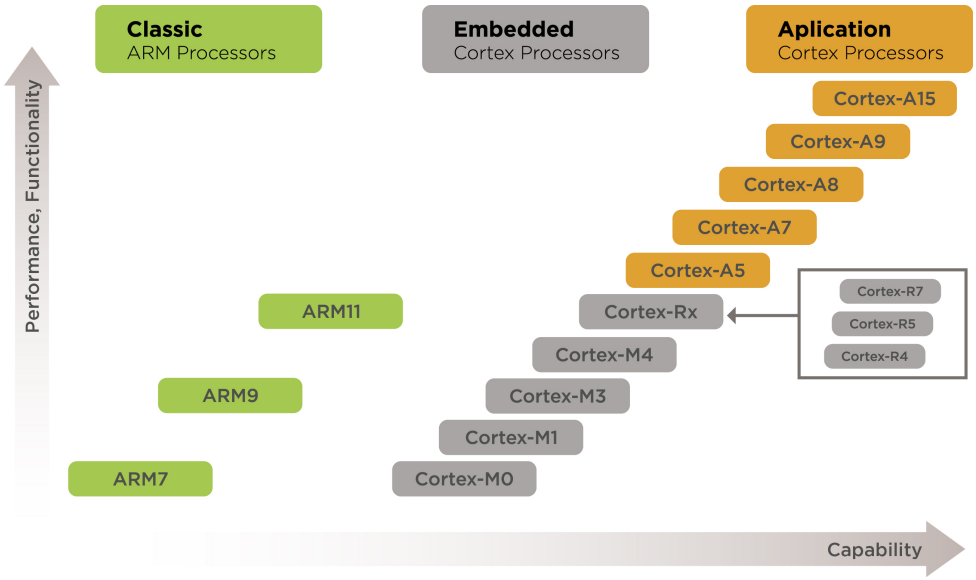
ARM ban đầu là Acorn RISC Machine, là một họ kiến trúc tính toán tập lệnh rút gọn (RISC) cho các bộ xử lý máy tính, được cấu hình cho các môi trường khác nhau. Arm Holdings phát triển kiến trúc và cấp phép cho các công ty khác, những người thiết kế các sản phẩm của riêng họ triển khai một trong những kiến trúc đó bao gồm các hệ thống trên chip (SoC) và các mô đun trên hệ thống (SoM) kết hợp bộ nhớ, giao diện, radio, …Nó cũng thiết kế các lõi thực hiện tập lệnh này và cấp phép cho các thiết kế này cho một số công ty kết hợp các thiết kế cốt lõi đó vào các sản phẩm của riêng họ.

Các bộ xử lý có kiến trúc RISC thường yêu cầu ít bóng bán dẫn hơn các bộ xử lý có kiến trúc tính toán tập lệnh rút gọn (RISC), giúp giảm chi phí, tiêu thụ điện năng và tản nhiệt. Những đặc điểm này là mong muốn đối với các thiết bị nhẹ, di động, chạy bằng pin bao gồm cả điện thoại thông minh, máy tính xách tay và máy tính bảng. Đối với các siêu máy tính tiêu thụ một lượng điện lớn, ARM cũng có thể là một giải pháp tiết kiệm năng lượng.

ARM Holdings định kỳ phát hành bản cập nhật cho kiến trúc này. Các phiên bản của kiến trúc này từ ARMv3 đến ARMv7 hỗ trợ không gian địa chỉ 32 bit và số học 32 bit, hầu hết các kiến trúc đều có các chỉ thị (lệnh) có độ dài cố định 32 bit. Phiên bản Thumb hỗ trợ tập lệnh có độ dài thay đổi, hỗ trợ cả hai lệnh 32 bit và 16 bit để cải thiện mật độ mã. Một số lõi cũ hơn cũng có thể cung cấp thực thi phần cứng cho mã byte Java. Được phát hành năm 2011, kiến trúc ARMv8-A đã hỗ trợ thêm cho không gian địa chỉ 64 bit và số học 64 bit với tập lệnh và độ dài cố định 32 bit.

Bộ vi xử lý ARM được thiết kế để hoạt động hiệu năng nhất có thể, chỉ chấp nhận các lệnh có thể được thực hiện trong một chu kỳ bộ nhớ. Với hơn 100 tỷ bộ

xử lý ARM được sản xuất đến năm 2017, ARM là kiến trúc tập lệnh được sử dụng rộng rãi nhất.



Hình 1.1: Các dòng vi xử lý lõi ARM

1.1.2 Lõi ARM Cortex-M3

Dòng ARM Cortex là một bộ xử lý thế hệ mới đưa ra một kiến trúc chuẩn cho nhu cầu đa dạng về công nghệ. Không giống như các chip ARM khác, dòng Cortex là một lõi xử lý hoàn thiện, đưa ra một chuẩn CPU và kiến trúc hệ thống chung. Dòng Cortex có ba phân nhánh chính [2]:

- Cấu hình A: cho các ứng dụng Application, yêu cầu cao chạy trên các hệ điều hành mở và phức tạp như Linux, Android.
- Cấu hình R: cho các ứng dụng thời gian thực Real Time.
- Cấu hình M: cho các ứng dụng vi điều khiển (VĐK) Microcontroller.

Bộ vi xử lý ARM Cortex-M3 là bộ vi xử lý ARM đầu tiên dựa trên kiến trúc ARMv7-M và được thiết kế đặc biệt để đạt được hiệu suất cao trong các ứng dụng nhúng cần tiết kiệm năng lượng và chi phí, chẳng hạn như các vi điều khiển, hệ thống cơ ô tô, hệ thống kiểm soát công nghiệp và hệ thống mạng không dây. Thêm vào đó là việc lập trình được đơn giản hóa đáng kể giúp kiến trúc ARM trở thành một lựa chọn tốt cho ngay cả những ứng dụng đơn giản nhất.

Bảng 1.1: Các dòng vi điều khiển STM32 và lõi ARM tương ứng

STM32 series	ARM CPU core(s)
F0	Cortex-M0
C0, G0, L0	Cortex-M0+
F1, F2, L1	Cortex-M3
F3, F4, G4, L4, L4+	Cortex-M4
WB, WL	Cortex-M4, Cortex-M0+
F7	Cortex-M7
H7	Cortex-M7, Cortex-M4
H5, L5, U5, WBA	Cortex-M33

1.1.3 Dòng vi điều khiển STM32

STM32 là dòng vi điều khiển dựa trên một số lõi ARM Cortex-M 32 bit khác nhau[1]. STMicroelectronics lấy bản quyền các vi xử lý được thiết kế bởi ARM Holdings với rất nhiều lựa chọn cấu hình và chọn ra các cấu hình cho từng thiết kế vi điều khiển của họ.

STM32F103 thuộc họ F1 với lõi ARM Cortex M3 là vi điều khiển 32 bit, tốc độ tối đa là 72Mhz. Giá thành tương đối rẻ so với các loại vi điều khiển có chức năng tương tự. Mạch nạp cũng như công cụ lập trình khá đa dạng và dễ sử dụng.

1.2 Mạch phát triển STM32F103C8T6 - Blue Pill

1.2.1 Vi điều khiển STM32F103C8T6

Vi điều khiển STM32F103C8T6 thuộc dòng VĐK STM32F1xx của ST Microelectronics với nhân ARM 32-bit Cortex M3 72MHz với các thông số quan trọng trong bảng 1.2.

Bảng 1.2: Các thông số quan trọng của VĐK STM32F103C8T6

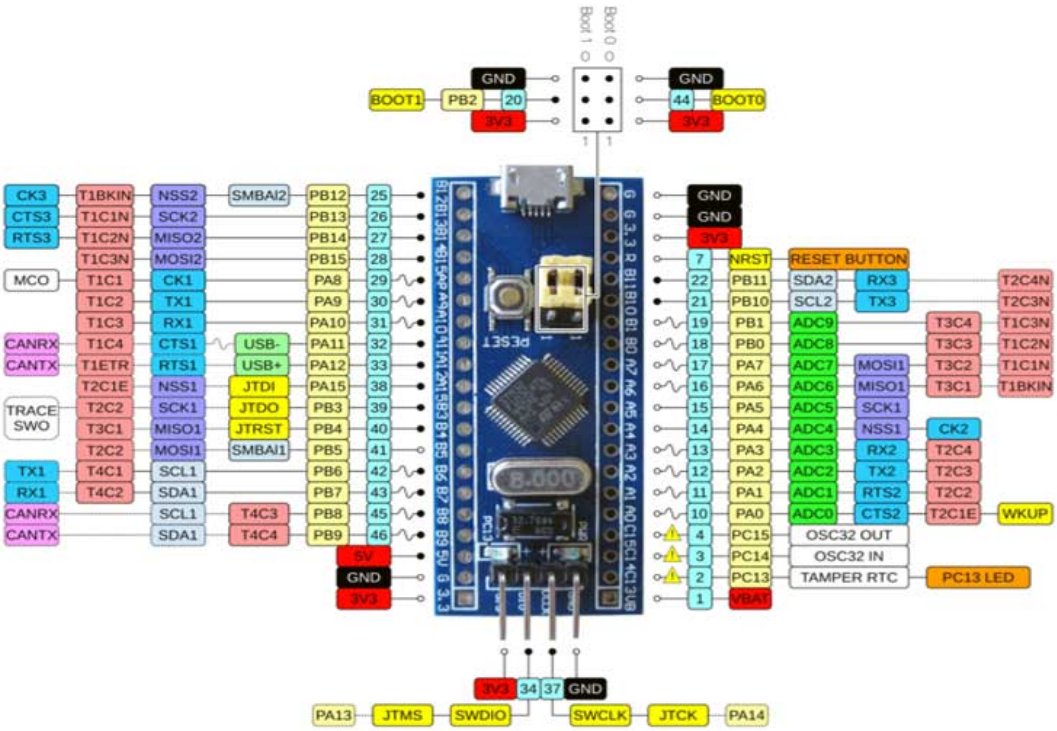
Bộ nhớ	
Bộ nhớ SRAM	20k bytes
Bộ nhớ Flash	64k bytes
Clock, reset và quản lý nguồn	
Điện áp hoạt động	2.0v - 3.6v
Thạch anh ngoại chính	4MHz - 20 MHz
Thạch anh RTC	32.768Hz
Ngoại vi quản lý điện áp	POR, PDR, PVD
Chế độ	Ngủ, chờ, ngừng hoạt động

VĐK STM32F103C8T6 có đa dạng các ngoại vi cho ứng dụng nhúng:

- 2 bộ ADC 12 bit với 9 kênh cho mỗi bộ
  - Khoảng giá trị chuyển đổi từ 0 – 3.6V.
  - Lấy mẫu nhiều kênh hoặc 1 kênh.
  - Có cảm biến nhiệt độ nội.
- Hỗ trợ 9 kênh giao tiếp bao gồm:
  - 2 bộ I^2C(SMBus/PMBus)
  - 3 bộ USART(ISO 7816 interface, LIN, IrDA capability, modem control)
  - 2 bộ SPI (18 Mbit/s)
  - 1 bộ CAN interface (2.0B Active)
  - USB 2.0 full-speed interface
- 7 Timer cho các ngoại vi:
  - 3 timer 16 bit hỗ trợ các mode IC/OC/PWM.
  - 1 timer 16 bit hỗ trợ để điều khiển động cơ với các mode bảo vệ như ngắt input, dead-time..
  - 2 watchdog timer.
  - 1 SysTick timer 24 bit đếm xuống dùng cho các ứng dụng như hàm Delay.
- 7 kênh DMA cho các ngoại vi như ADC, I^2C, SPI, UART.

1.2.2 Mạch phát triển Blue Pill

Blue Pill là một mạch phát triển đơn giản cho vi điều khiển STM32F103C8. Blue Pill ra đầy đủ chân của vi điều khiển cùng với cổng giao tiếp USB, các chân gỡ lỗi SWD và các chân thiết lập khởi động.



Hình 1.2: Ra chân của mạch phát triển Blue Pill

Về chi tiết, mạch phát triển Blue Pill bao gồm:

- VĐK STM32F103C8T6.
- Led báo trạng thái nguồn, led PC13, nút nhấn reset.
- IC chuyển đổi điện áp từ 5 xuống 3.3v cung cấp cho vi điều khiển.
- Cổng Micro USB kết nối tới ngoại vi USB của vi điều khiển và cấp nguồn.
- Thạch anh ngoại 8MHz.
- Thạch anh ngoại thời gian thực 32KHz.
- Ra chân đầy đủ tất cả GPIO và các giao tiếp I^2C, CAN, UART, SPI, USB, ...

1.3 Lập trình cho mạch phát triển Blue Pill

1.3.1 Môi trường lập trình cho Blue Pill

Mỗi vi điều khiển dòng ARM lại có kiểu kiến trúc khác nhau. Đối với STM32 nói chung và STM32F103C8 được sử dụng trong Blue Pill, bộ công cụ lập trình GCC được sử dụng là arm-none-eabi-, đối với các ngôn ngữ rust, bộ công cụ được sử dụng là thumbv7m-none-eabi.

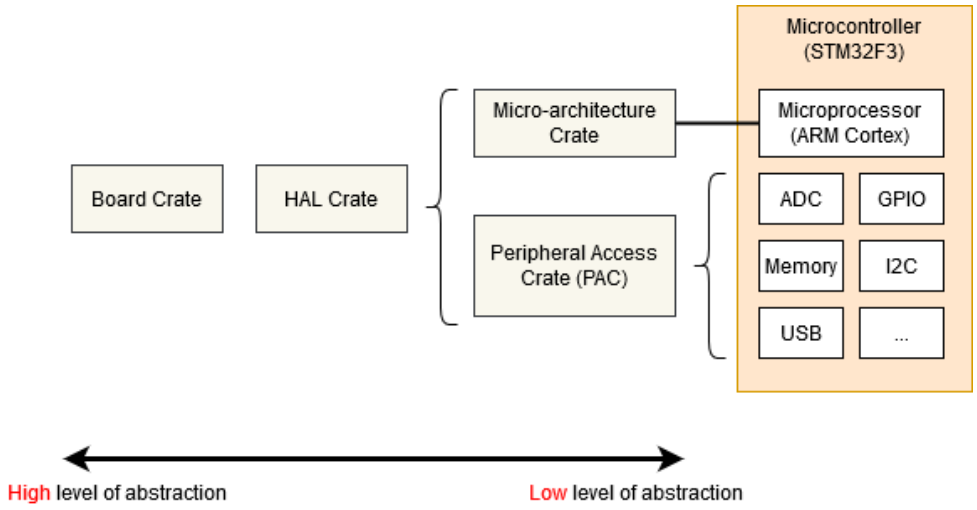
Để lập trình cho vi điều khiển STM32, các công cụ soạn thảo nên có những gợi ý cho người dùng, giúp người dùng có thể dễ dàng biên dịch và nạp cho thiết bị vi điều khiển. Một số phần mềm có thể sử dụng bao gồm:

- STM32CubeIDE và STM32CubeMX: Phần mềm chính hãng từ ST giúp thiết lập trước các thông số và soạn thảo phần mềm cho STM32. Có thể kết hợp với STM32CubeProg giúp nạp chương trình cho vi điều khiển.
- μVision IDE: Công cụ soạn thảo và gỡ lỗi hỗ trợ trực tiếp cho ARM.
- Công cụ soạn thảo khác như VSCode, Vim, Notepad++, ...: Các công cụ này thường không có sẵn môi trường phát triển cho STM32, cần thiết lập thêm qua các công cụ khác.

1.3.2 Các thư viện lập trình

Các thư viện lập trình sẽ giúp đơn giản hóa quá trình phát triển phần mềm cho các vi điều khiển. Các thư viện hỗ trợ lập trình đã có sẵn và rất đa dạng, đang được sử dụng rộng rãi như thư viện từ ST, thư viện Arduino hay từ Rust.

Các thư viện được chia thành các mức khác nhau, từ cấp thấp đến cấp cao. Ở mức PAC, các thanh ghi của các thành phần ngoại vi được tích hợp vào thư viện giúp người lập trình có thể tương tác trực tiếp với các thành ghi. Tại mức HAL, các ngoại vi của vi điều khiển được trừu tượng hóa thành các kiểu dữ liệu của ngôn ngữ lập trình, giúp người lập trình có thể lập trình vi điều khiển mà không cần hiểu chuyên sâu về các thanh ghi. Đối với mức Board, các thành phần mức HAL được tổng hợp lại, chỉnh sửa để hỗ trợ cho việc lập trình từ các mạch phát triển:



Hình 1.3: Các mức hỗ trợ của thư viện lập trình

1.3.3 Nạp chương trình

Sau khi đã phát triển chương trình cho STM32, chương trình sau khi biên dịch cần được nạp vào VĐK để chạy. Mạch phát triển Blue Pill hỗ trợ 2 phương thức nạp phần mềm phổ biến là UART qua ngoại vi UASRT1 sử dụng các thiết bị USB to UART và qua giao thức SWD sử dụng các thiết bị JLink, STLink hay SMSIS-DAP. Có một số phần mềm hỗ trợ nạp bằng cả 2 giao thức này:

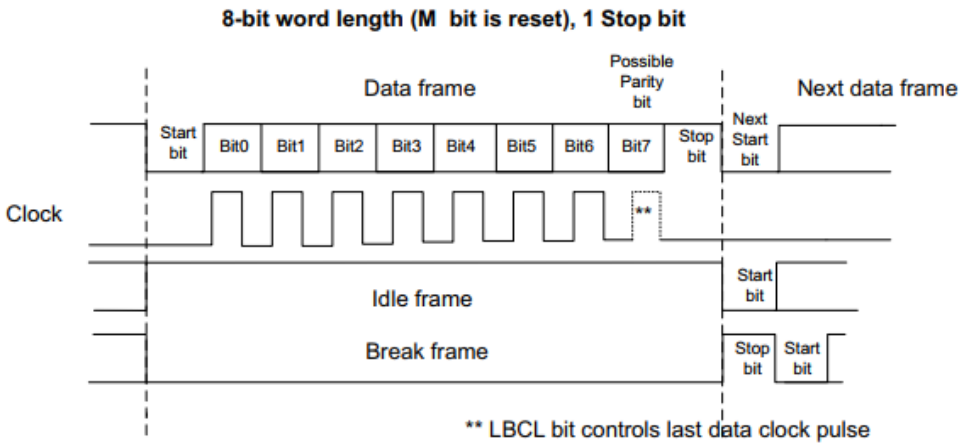
- STM32CubeProgrammer: Đây là công cụ chính hãng để nạp phần mềm cho các sản phẩm vi điều khiển từ ST. Công cụ này hỗ trợ nhiều chuẩn giao tiếp khác nhau cùng nhiều chức năng nâng cao như đọc, ghi, xác thực chương trình hay tự động hóa quá trình nạp phần mềm.

- Probe-rs: Probe-rs là một bộ công cụ mới giúp gỡ lỗi cho thiết bị nhúng, giúp nạp chương trình cho các vi điều khiển ARM, RISC-V sử dụng các chuẩn nạp khác nhau như CMSIS-DAP, STLink, JLink. Ngoài ra, probe-rs còn hỗ trợ các chuẩn gỡ lỗi như GDB và Microsoft DAP giúp gỡ lỗi qua Visual Studio Code hoặc qua giao diện dòng lệnh.

1.4 Giao tiếp UART

UART là một ngoại vi cơ bản và thường dùng trong các quá trình giao tiếp với các module như: Xbee, Wifi, Bluetooth…Khi giao tiếp UART kết hợp với các IC giao tiếp như MAX232CP, SP485EEN… thì sẽ tạo thành các chuẩn giao tiếp RS232, RS485. Đây là các chuẩn giao tiếp thông dụng và phổ biến trong công nghiệp từ trước đến nay.

Giao tiếp USART yêu cầu chân GND chung, chân Tx, chân Rx và chân CLK tùy chọn. Khi sử dụng chân UART_CLK thì giao tiếp UART sẽ trở thành giao tiếp đồng bộ và không dùng sẽ là chuẩn giao tiếp không đồng bộ:



Hình 1.4: Giap tiếp USART

Ưu điểm của giao tiếp UART không đồng bộ: tiết kiệm chân vi điều khiển (2 chân), là ngoại vi mà bất kì một vi điều khiển nào cũng có, có khá nhiều module, cảm biến dùng UART để truyền nhận data với vi điều khiển.

Nhược điểm của loại ngoại vi này là tốc độ khá chậm, tốc độ tối đa tùy thuộc vào từng dòng, quá trình truyền nhận dễ xảy ra lỗi nên trong quá trình truyền nhận cần có các phương pháp để kiểm tra (thông thường là truyền thêm bit hoặc byte kiểm tra lỗi). UART không phải là một chuẩn truyền thông, khi muốn nó là một chuẩn truyền thông hoặc truyền data đi xa, cần phải sử dụng các IC thông dụng để tạo thành các chuẩn giao tiếp đáng tin cậy như RS485 hay RS232....

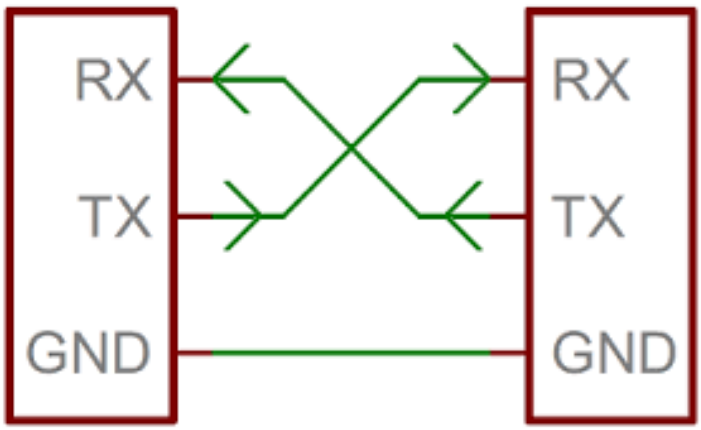
STM32F103C8 có 3 bộ UART với nhiều mode hoạt động với một số tính năng nổi bật như sau:

- Đầy đủ các tính năng của bộ giao tiếp không đồng bộ.
- Điều chỉnh baud rate bằng lập trình và tốc độ tối đa lên đến 4.5Mb/s.
- Độ dài được lập trình là 8 hoặc 9 bit.
- Cấu hình bit stop hỗ trợ là 1 hoặc 2.
- Có chân clock nếu muốn chuyển giao tiếp thành đồng bộ.

- Cấu hình sử dụng 1 dây hoặc 2 dây.
- Có bộ DMA nếu muốn đẩy cao thời gian truyền nhận.
- Bit cho phép truyền nhận riêng biệt.

Thông thường sẽ dùng ngắt nhận UART để nhận dữ liệu vì sử dụng ngắt sẽ tiện lợi, không tốn thời gian chờ cũng như mất dữ liệu. Các tốc độ thường dùng để giao tiếp với máy tính: 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, 115200.

Thực hiện kết nối giao tiếp UART bằng cách kết nối chân GND, Tx, Rx của thiết bị 1 với chân GND, Rx, Tx của thiết bị 2 (Hình 1.5).



Hình 1.5: Kết nối chân giao tiếp UART

Một số phần mềm giao tiếp với máy tính bao gồm: picocom(linux), Hercules_3-2-5, teraterm, SerialOscilloscope-v1.5... Một số module dùng để giao tiếp với máy tính: CP2102 USB 2.0, module PL2303, module FT232RL, CH340G hoặc sử dụng mạch CMSIS-DAP tích hợp cả SWD và UART.

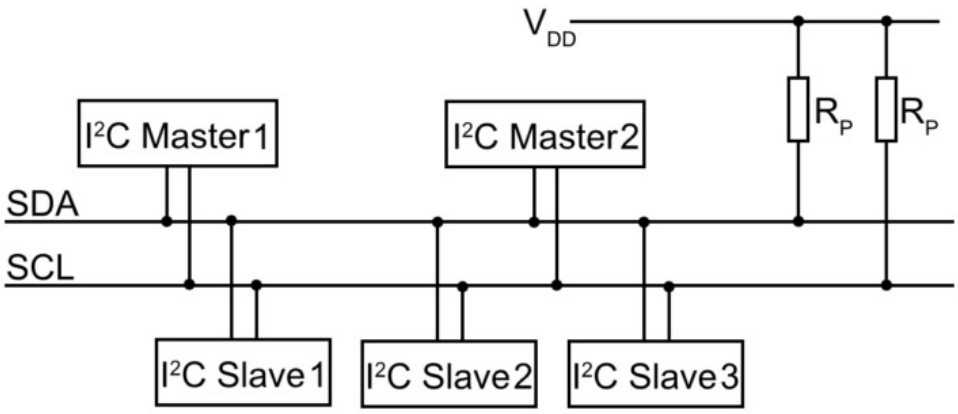
1.5 Giao tiếp I^2C

Inter-Integrated Circuit (I^2C) là một loại bus nối tiếp đồng bộ hai chiều với 2 dây tín hiệu. STM32F103C8T6 có hai bộ I^2C có thể lập trình được:

- 2 chế độ Master và Slave
- Có thể lập trình địa chỉ cho chế độ Slave, có chế độ kiểm tra start/stop.

- Hỗ trợ chế độ địa chỉ 7 bit và 10 bit.
- Hỗ trợ hai chuẩn tốc độ 100Khz và 400 Khz.
- Có bộ lọc nhiễu Analog.
- Tích hợp mode DMA.
- Có các cờ báo trạng thái: Nhận, truyền, kết thúc chuyển đổi, báo lỗi.
- Có các ngắt như: Ngắt buffer truyền, nhận; ngắt sự kiện, ngắt báo lỗi.
- Quá trình truyền data tùy thuộc vào mode cấu hình của I^2C là master hoặc slave, ở chế độ 10 bit địa chỉ hay 7 bit địa chỉ (phổ biến hơn).

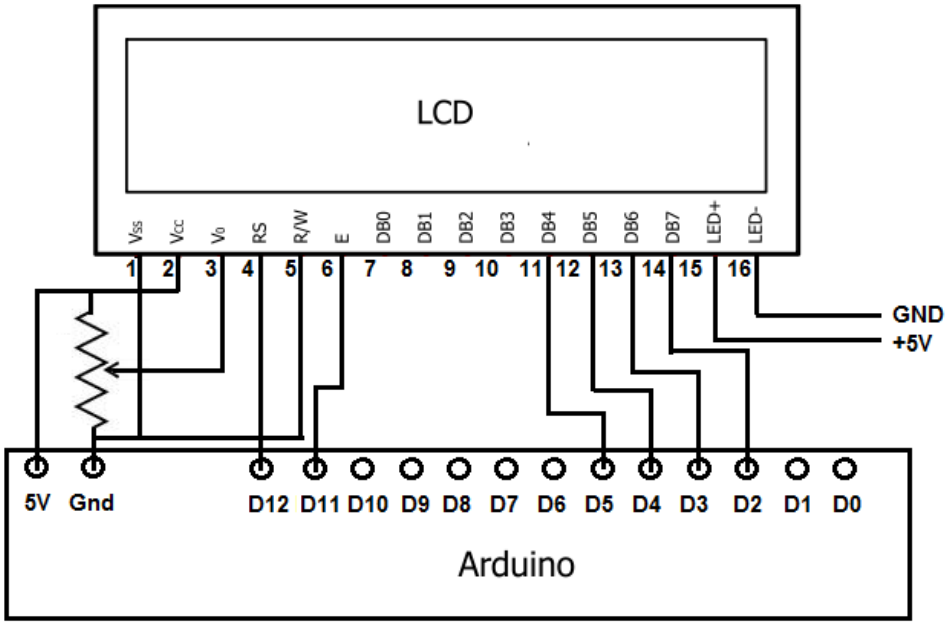
Kết nối I^2C được thực hiện bằng cách kết nối các chân SDA và CLK của các thiết bị lại với nhau (Hình 1.6).



Hình 1.6: Sơ đồ kết nối giao tiếp I^2C

1.6 Module LCD1602 sử dụng ic HD44780

HD44780 là một ic điều khiển màn hình ma trận điểm (dot-matrix) giúp hiển thị các ký tự chữ, số và các ký tự. Nó có thể được thiết lập để điều khiển với chế độ 4 bit hoặc 8 bit. Tất cả các chức năng như bộ nhớ hiển thị, bộ sinh ký tự hiển thị và các thành phần điều khiển màn hình được tích hợp sẵn trong một ic, vì vậy việc điều khiển màn hình ma trận điểm có thể được thực hiện một cách dễ dàng.



Hình 1.7: Màn hình LCD1602 HD44780

Một số chức năng có thể kể đến bao gồm:

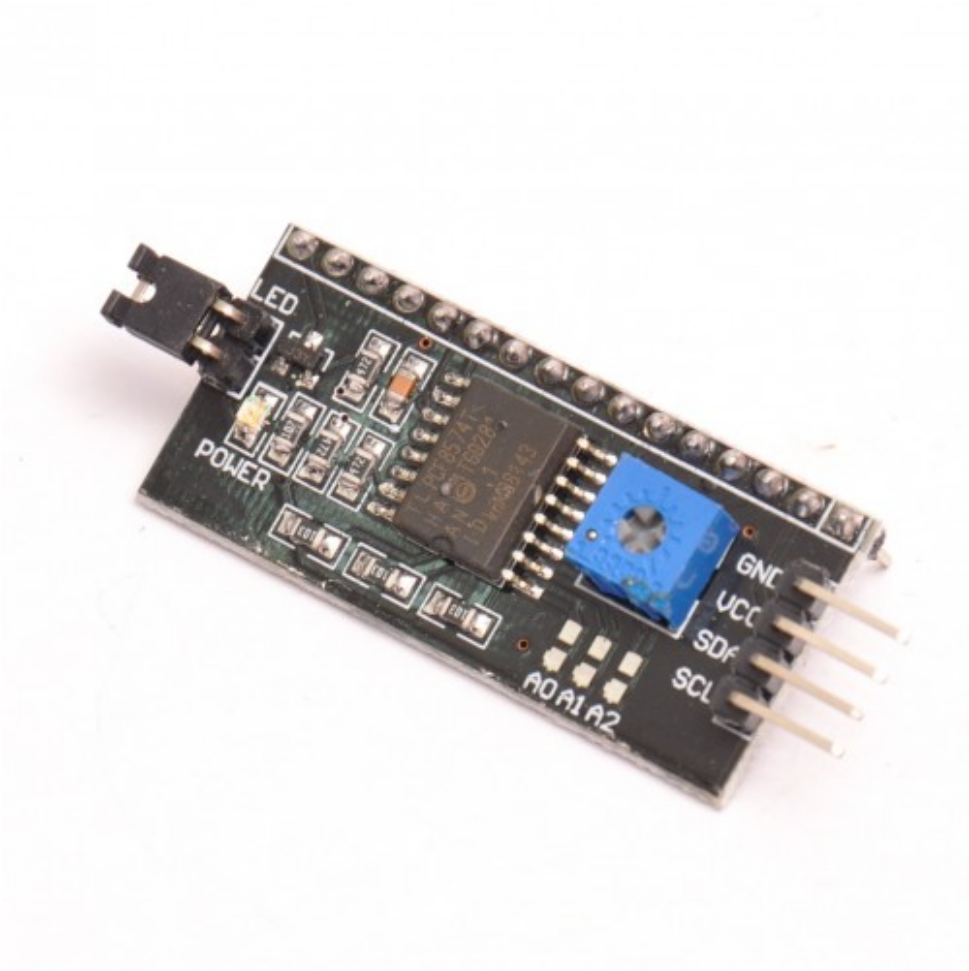
- Điện áp sử dụng: 2.V7-5.5V
- Bộ nhớ tối đa: 80 ký tự
- Có một số chức năng như: Xóa màn hình, quay về đầu, bật tắt hiển thị, bật tắt con trỏ, bật tắt nháy con trỏ, di chuyển con trỏ, di chuyển màn hình.
- Đối với chế độ 4 bit, thực hiện ghi lần lượt 4 bit thấp và 4 bit cao cho một ký tự.

Hình 1.8: Bảng ký tự điều khiển màn hình LCD1602

1.7 Module mở rộng I/O PCF8574

Khi kết nối tới màn hình LCD1602, số lượng chân I/O cần thiết cho chế độ 4 bit là 6, và 10 tương ứng với chế độ 8 bit. Để tiết kiệm chân điều khiển và đơn giản hóa quá trình điều khiển, các module mở rộng chân sẽ là giải pháp tối ưu.

PCF8574 là một module mở rộng vào ra 8 bit sử dụng giao tiếp I^2C được sử dụng rộng rãi với 8 địa chỉ có thể thay đổi được. Với mỗi byte được ghi vào module qua giao tiếp I^2C, các bit tương ứng sẽ được bật tắt giúp điều khiển LCD1602.



Hình 1.9: Module PCF8574 cho màn hình LCD1602

Chương 2: KẾT LUẬN

2.1 Kết quả đạt được

Qua quá trình nghiên cứu các thuật toán, kiến thức liên quan đến vi điều khiển STM32F103C8T6, module I^2C điều khiển màn hình LCD1602, giao thức UART và giao tiếp I^2C, nhóm chúng em đã đạt được môt số kết quả như sau:

- Tìm hiểu giao thức và điều khiển thành công việc đọc và ghi dữ liệu qua giao tiếp UART.
- Tìm hiểu, và ghi được dữ liệu qua giao tiếp I^2C
- Làm chủ được giao tiếp I^2C và giao tiếp UART.
- Hiểu cơ bản phương pháp lập trình nhúng trên nền ARM.

2.2 Hướng phát triển và đề xuất

Bên cạnh những kiến thức thu được cho bản thân, nhóm chúng em cũng những phương hướng phát triển tiếp theo cho đề tài này như sau:

- Thực hiện xử lý một số thao tác nâng cao như di chuyển giữa các ký tự khi nhập, xử lý phím Backspace và phím Delete.
- Cho phép người dùng thao tác và điều khiển các chức năng của màn hình LCD1602 như điều khiển con trỏ, hiển thị phông chữ trực quan hơn, ...
- Thêm module thời gian thực để hiển thị thời gian trên màn hình.

TÀI LIỆU THAM KHẢO

[1] ST Microelectronics. STM32 Microcontrollers (MCUs). Truy cập lần cuối 12/06/2023.

[2] TS. Nguyễn Đào Trường and ThS. Lê Đức Thuận. *Giáo trình lập trình ARM cơ bản*.

PHỤ LỤC: MÃ NGUỒN CHƯƠNG TRÌNH

Mã nguồn chương trình được đăng tải trực tuyến tại đường dẫn: `https://github.com/dungph/litle-prompt`