ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN
THIẾT KẾ HỆ THỐNG NHÚNG
CE224.P13

Giảng viên hướng dẫn: ***Trần Ngọc Đức***
Sinh viên thực hiện:

Dương Hiển Gia Khang 22520610

TP. Hồ Chí Minh, 30 tháng 12 năm 2024

MỤC LỤC

I. Wired Connections

1. Giao thức UART

a. Khái quát về giao thức UART

b. Giao tiếp giữa ESP8266 NODEMCU và STM32F407VET6 thông qua UART

2. Giao thức I2C

a. Khái quát về giao thức I2C

b. Giao tiếp giữa SHT31 và STM32F407VET6 thông qua I2C

3. Giao thức SPI

a. Khái quát về giao thức SPI

b. Giao tiếp giữa ILI9341 và STM32F407VET6 thông qua SPI cho chức năng cảm biến (touch)

II. Wireless Connections

1. Wifi (Web Openweathermap.com)

2. Protocol

CHƯƠNG 4: SOFTWARE DESIGN AND IMPLEMENTATION

A. Đọc cảm biến SHT31

B. Hiển thị screen

CHƯƠNG 5: XÂY DỰNG SƠ ĐỒ VẬN HÀNH

A. Sơ đồ vận hành tính năng hệ thống

ASMD

B. Sơ đồ vận hành của màn hình hiển thị

CHƯƠNG 6: SẢN PHẨM HOÀN THIỆN

A. Hình ảnh sản phẩm thực tế

B. Source code

C. Ưu điểm và nhược điểm

D. Cải tiến sản phẩm

CHƯƠNG 1. TỔNG QUAN

A. Mục tiêu đề tài

Xây dựng hệ thống dự báo thời tiết thông minh:

- Thiết kế và triển khai một hệ thống nhúng có khả năng thu thập dữ liệu thời tiết từ môi trường thực tế và từ nguồn dữ liệu trực tuyến.
- Đảm bảo an toàn Hệ thống phải có khả năng phát hiện và phản ứng kịp thời trước các tình huống bất ngờ trên đường, đảm bảo an toàn cho hành khách và người tham gia giao thông khác.

Áp dụng công nghệ IoT và hệ thống nhúng:

- Kết nối hệ thống với Internet để truy xuất dữ liệu thời tiết từ API, đảm bảo thông tin được cập nhật liên tục và chính xác.
- Sử dụng các thiết bị và công nghệ hiện đại, kết hợp giữa phần cứng và phần mềm để tối ưu hiệu suất và tính ứng dụng của hệ thống.

Nâng cao kiến thức:

- Biết vận dụng kiến thức đã học về lập trình STM32 để thực hiện đồ án.

.

B. Nội dung thực hiện

Để quá trình thực hiện đồ án đảm bảo đúng tiến độ, nhóm đã tiến hành các nội dung sau:

- Tìm hiểu về các loại VDK phù hợp: Qua tìm hiểu, nhóm biết được trên thị trường hiện có các loại thông dụng là: Arduino, ESP32, PIC, 8051, STM32. Nhóm đã quyết định chọn VDK STM32F407VET6 vì đây là một vi điều khiển 32 bit dễ sử dụng, giá cả hợp lý và hỗ trợ mạnh mẽ cũng như phổ biến trong nhiều ứng dụng nhúng khác nhau.
- Tìm hiểu về nguyên lý hoạt động của các linh kiện: Tra cứu và đọc datasheet để hiểu nguyên lý của các linh kiện và lựa chọn ra linh kiện sẽ dùng cho đồ án.
- Phát triển thuật toán: Tìm hiểu các nguồn trên các website và các bài toán mẫu, dựa vào đó thay đổi code cho phù hợp với dòng VDK đang dùng và các linh kiện khác.
- Tìm hiểu về phần mền để nhúng: lựa chọn STM32CubeIDE, vì trình biên dịch này là khá nhanh, mạnh, được nhiều người sử dụng, hỗ trợ simulation, hỗ trợ nhiều loại debugger, tối ưu hóa cao cho các ứng dụng có bộ xử lý lõi ARM.
- Kiểm thử và điều chỉnh: thử nghiệm bằng testboard, kiểm tra khả năng hoạt động của linh kiện và code nhúng để tránh các hiện tượng hư mạch hay hư linh kiện, đảm bảo sản phẩm đầu ra phải chạy được và hiển thị đầy đủ chức năng.
- Hàn mạch: Chuẩn bị PCB đục lỗ, dây đồng, hàng rào bảo vệ để tiến hành hàn. Đảm bảo hàn đúng kỹ thuật, an toàn nhằm đem đến một sản phẩm vừa đầy đủ vừa tinh gọn.
- Đóng gói: bao bọc PCB và các linh kiện khác bằng lớp formex trắng phủ xung quanh để sản phẩm trong giống một thiết bị được đóng gói bên ngoài thị trường.

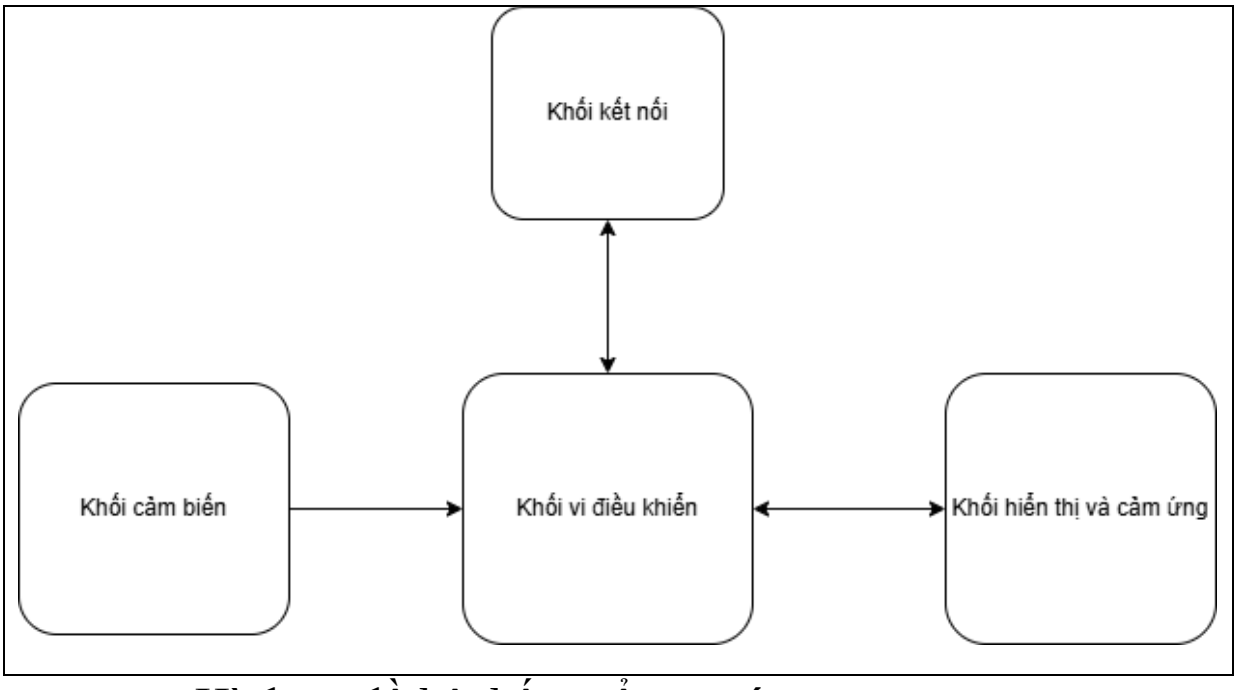
CHƯƠNG 2. THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG

A Các yêu cầu hệ thống:

- Yêu cầu API:
  - Đúng đường link API
  - Đảm bảo API key đã được kích hoạt
- Yêu cầu Wifi:
  - Có tín hiệu Wifi chuẩn
  - Đảm báo đúng yêu mật khẩu và tên Wifi
- Yêu cầu UART:
  - Đảm bảo baudrate của STM32 và ESP32 đồng nhất .
  - Đảm bảo các thông số Word lenght,Parity và Stop bits là giống nhau
  - Đảm bảo Clock hệ thống là đồng nhất
  - Chung GND giữa STM32 và ESP32

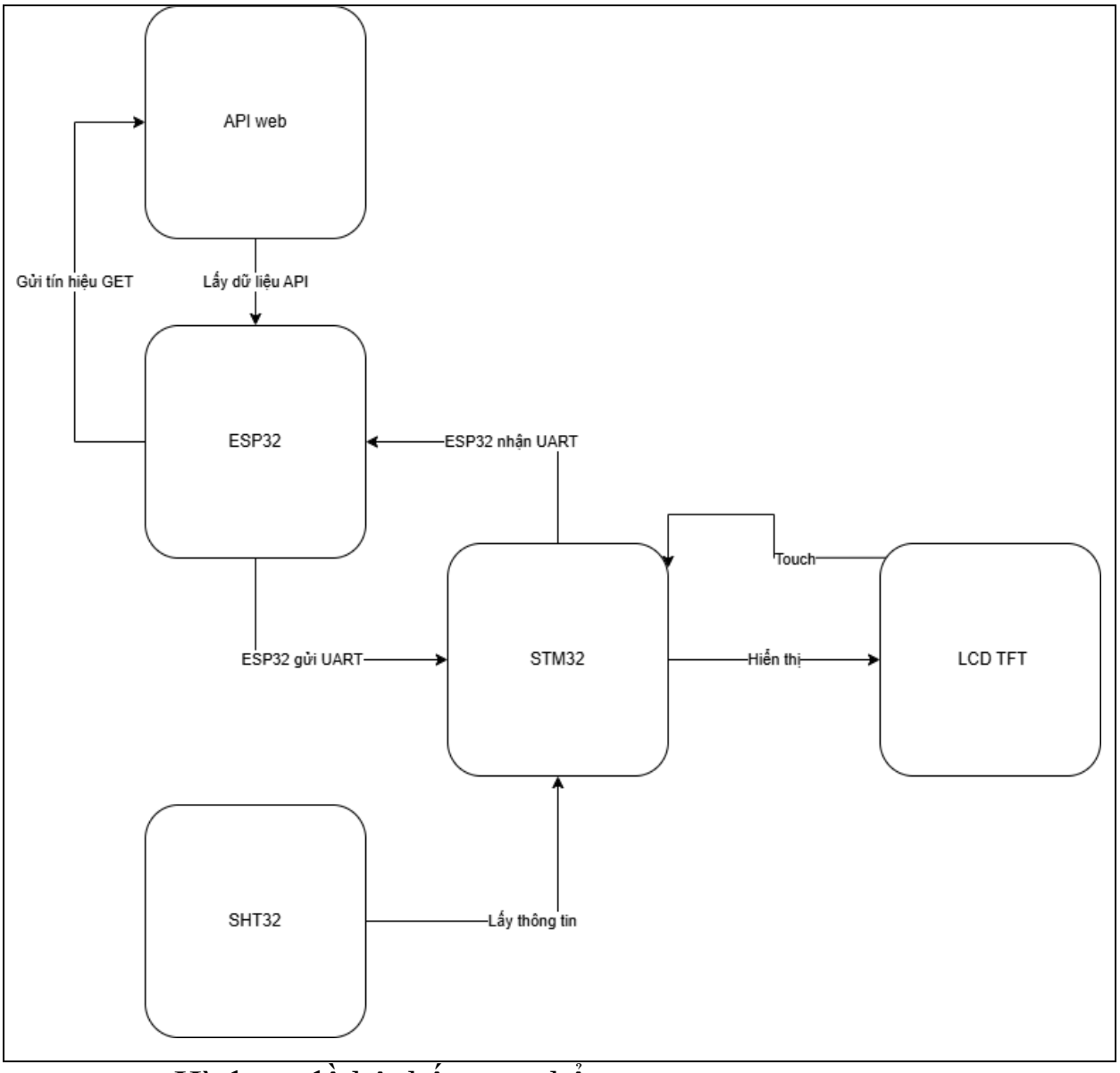
B Sơ đồ hệ thống:

I .Sơ đồ hệ thống tổng quát:



Hình sơ đồ hệ thống tổng quát

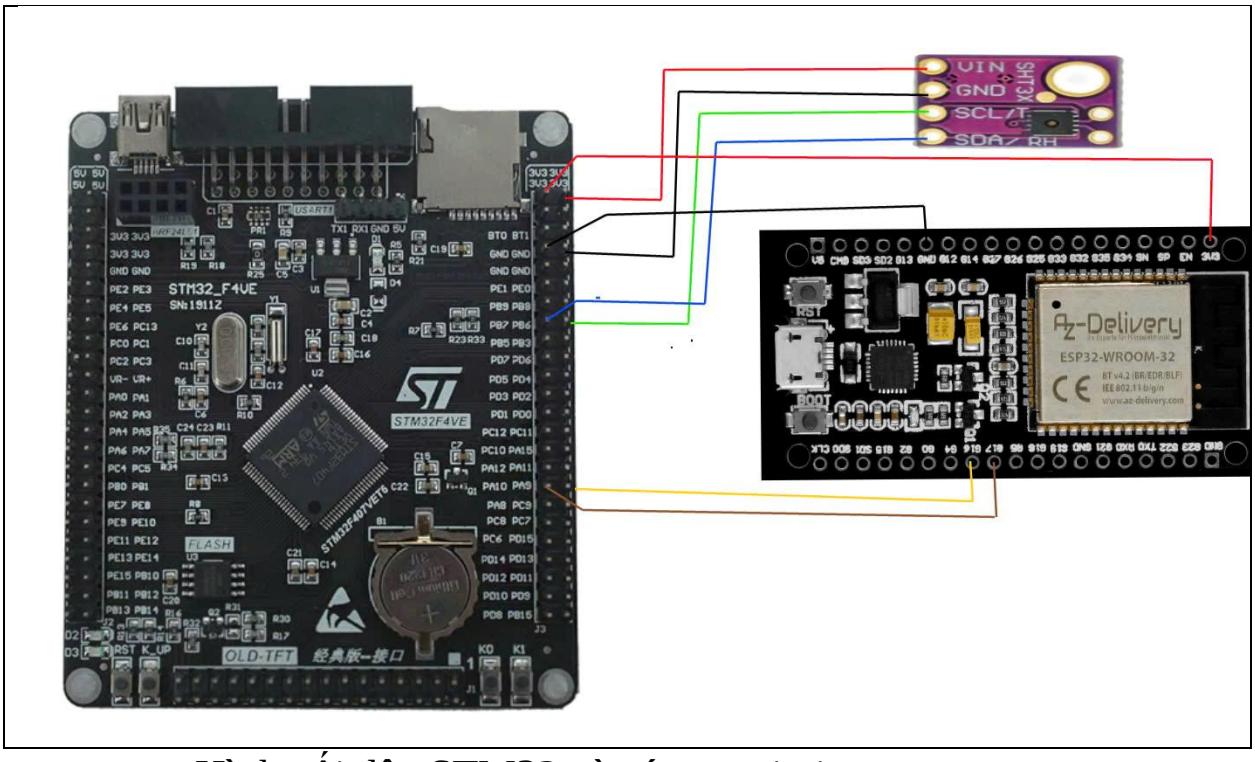
II .Sơ đồ hệ thống cụ thể:



Hình sơ đồ hệ thống cụ thể

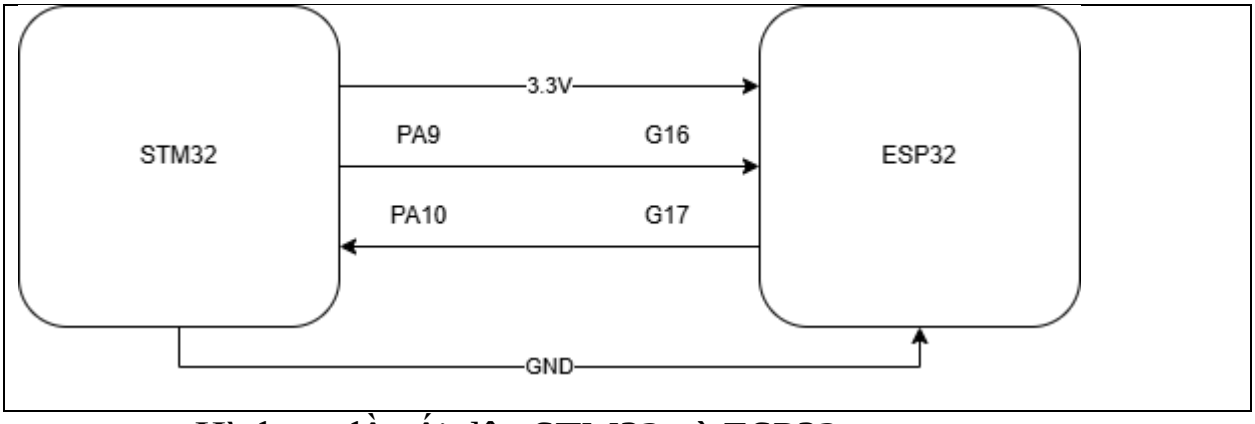
C Sơ đồ nối dây:

I.Sơ đồ dây tổng thể:

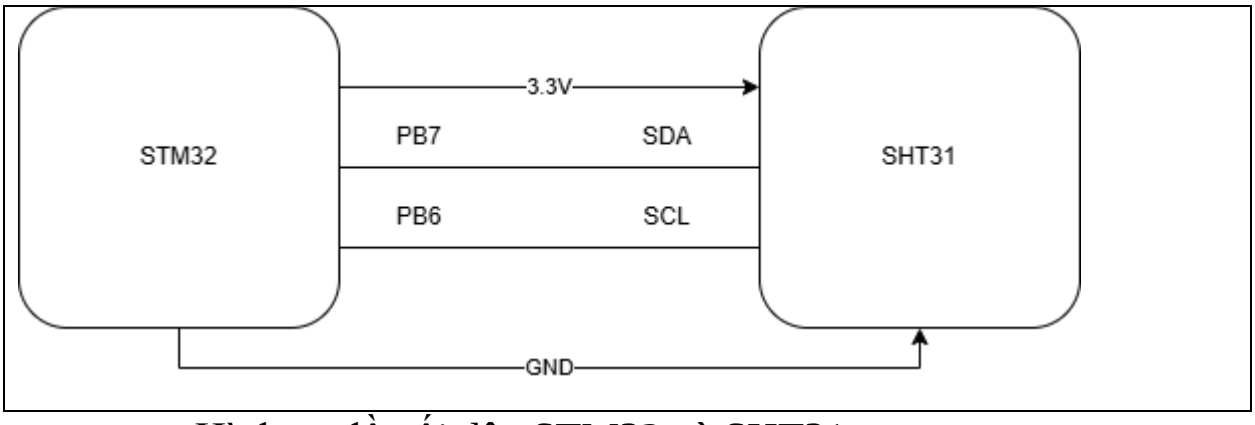


Hình nối dây STM32 và các ngoại vi

II.Sơ đồ nối dây cụ thể:



Hình sơ đồ nối dây STM32 và ESP32



Hình sơ đồ nối dây STM32 và SHT31

CHƯƠNG 3: MÔ TẢ CÔNG NGHỆ

A. Hardware Tech

I. Processor - STM32F407VET6

1. Tổng quan về vi điều khiển STM32F407VET6

STM32F407VET6 là một trong những vi điều khiển 32-bit hiệu năng cao thuộc dòng STM32F4 của STMicroelectronics. Nó được xây dựng dựa trên kiến trúc ARM Cortex-M4 với clock mã là 168Mhz, cung cấp hiệu năng xử lý mạnh mẽ, khả năng kết nối linh hoạt và khả năng tiêu thụ điện năng thấp, được ứng dụng rộng rãi trong các dự án nhúng đòi hỏi hiệu suất cao, như điều khiển động cơ, giao diện người máy, IoT, và nhiều ứng dụng khác.

Thông số của STM32F407VET6:

Bộ nhớ:

- 512KB bộ nhớ Flash.
- 192KB SRAM.

Clock, reset và quản lý nguồn:

- Sử dụng thạch anh ngoài từ 4Mhz -> 16Mhz.
- Thạch anh nội RC ở 8Mhz hoặc 40kHZ.
- Sử dụng thạch anh ngoài 32.768kHZ được sử dụng cho RTC (real time clock).
- Power on reset (POR), power down reset (PDR), programmable voltage detector (PVD).
- Điện áp hoạt động ở mức từ 1.8V -> 3.6V.
- Cấp nguồn ở chân Vbat bằng pin để hoạt động đồng bộ RTC và lưu trữ data khi mất nguồn cấp chính.

ADC : 2 bộ ADC 12bit với 10 kênh cho mỗi bộ.

- Giá trị chuyển đổi từ 0V -> 3.6V.
- Lấy mẫu 1 hoặc nhiều kênh.
- Có cảm biến nhiệt độ nội.

DMA : bộ chuyển đổi giúp tăng tốc độ xử lí.

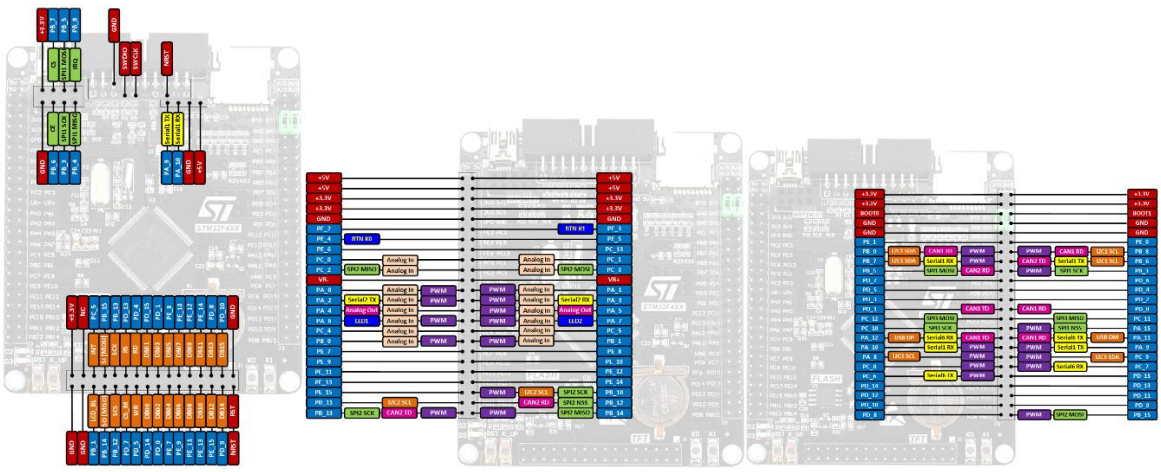
- 7 kênh DMA.
- Hỗ trợ DMA cho ADC,I2C,SPI,UART.

- 7 timer.
- 3 timer 16 bit hỗ trợ các mode IC/OC/PWM.
- 1 timer 16 bit hỗ trợ điều khiển động cơ.
- 2 watdog timer để bảo vệ và kiểm tra lỗi.
- 1 systick timer 24bit đếm xuống dùng cho các ứng dụng như delay.

Hỗ trợ 9 kênh giao tiếp gồm :

- 3 bộ I2C.
- 4 bộ USART, 2 UART.
- 3 SPIS.
- 1 FSMC.
- 2 bộ giao tiếp CAN.
- USB 2.0 full speed.
- 1 bộ giao tiếp SDIO/MMC.

2. Sơ đồ và chức năng các chân của vi điều khiển STM32F407VET6



Các chân của STM32F407VET6 có thể được chia thành các nhóm chính sau:

1. **Chân nguồn:**
    - **VCC:** Nguồn điện dương cho vi điều khiển.

- **GND:** Nguồn điện âm (mass).

2. **Chân tín hiệu:**
   - **GPIO (General Purpose Input/Output):** Các chân có thể được cấu hình làm ngõ vào hoặc ngõ ra để tương tác với các thiết bị ngoại vi khác.
   - **UART (Universal Asynchronous Receiver-Transmitter):** Dùng để truyền nhận dữ liệu tuần tự.
   - **SPI (Serial Peripheral Interface):** Giao tiếp nối tiếp đồng bộ.
   - **I2C (Inter-Integrated Circuit):** Giao tiếp nối tiếp đồng bộ tốc độ thấp.
   - **USB:** Giao tiếp nối tiếp tốc độ cao, dùng để kết nối với máy tính hoặc các thiết bị USB khác.
   - **CAN (Controller Area Network):** Giao tiếp công nghiệp, thường dùng trong các hệ thống ô tô.
   - **ADC (Analog-to-Digital Converter):** Chuyển đổi tín hiệu tương tự thành tín hiệu số.
   - **DAC (Digital-to-Analog Converter):** Chuyển đổi tín hiệu số thành tín hiệu tương tự.
   - **Timer:** Tạo xung, đo thời gian, điều khiển PWM.
   - **Và nhiều loại chân khác:** Ethernet, SDIO, ...
3. **Chân đặc biệt:**
   - **Reset:** Đặt lại vi điều khiển về trạng thái khởi động.
   - **Boot:** Chọn chế độ khởi động.
   - **Clock:** Cung cấp xung nhịp cho vi điều khiển.

II. Wifi Module - ESP8266 NODEMCU

1. Tổng quan về ESP8266 NODEMCU

ESP32 là một con chip cung cấp kết nối Wi-Fi và Bluetooth (ở một số mẫu)

cho các thiết bị nhúng – nói cách khác, cho các thiết bị IoT. Mặc dù về mặt kỹ thuật, ESP32 chỉ là con chip, nhưng các mô-đun và bo mạch phát triển chứa con chip này thường được nhà sản xuất gọi là “ESP32”.

ESP8266 NodeMCU đi kèm với mô-đun ESP-12E chứa chip ESP8266 có bộ vi xử lý Tensilica Xtensa 32-bit LX106 RISC. Bộ vi xử lý này hỗ trợ RTOS và hoạt động ở tần số xung nhịp có thể điều chỉnh từ 80MHz đến 160 MHz.

NodeMCU có 128 KB RAM và 4MB bộ nhớ Flash để lưu trữ dữ liệu và chương trình. Sức mạnh xử lý cao của nó với Wi-Fi / Bluetooth và các tính năng Điều hành Ngủ sâu tích hợp khiến nó trở nên lý tưởng cho các dự án IoT.

NodeMCU có thể được cấp nguồn bằng giắc cắm Micro USB và chân VIN (Chân nguồn cung cấp bên ngoài). Nó hỗ trợ giao diện UART, SPI và I2C.

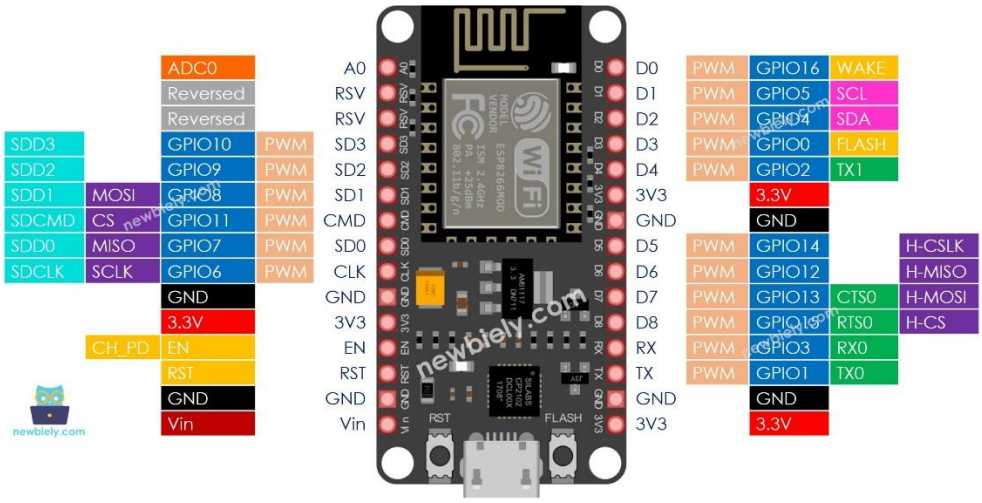
Các mô hình phổ biến nhất của NodeMCU là Amica (dựa trên khoảng cách chân hẹp tiêu chuẩn) và LoLin có khoảng cách chân rộng hơn và bo mạch lớn hơn

Một vài đặc điểm của ESP8266 NODEMCU:

- Bộ vi điều khiển: CPU RISC 32-bit Tensilica Xtensa LX106
- Điện áp hoạt động: 3.3V
- Điện áp đầu vào: 7-12V
- Bộ nhớ Flash: 4 MB
- SRAM: 64 KB
- Tốc độ đồng hồ: 80 MHz
- USB-TTL dựa trên CP2102 được bao gồm trên bo mạch, cho phép Plug n Play
- Cổng nạp giao tiếp: Micro || Type-C
- Tương thích các chuẩn wifi : 802.11 b/g/n

- Hỗ trợ: Wi-Fi Direct (P2P), soft-AP
- Tích hợp TCP/IP protocol stack
- Tích hợp TR switch, balun, LNA, power amplifier và mạng lưới phù hợp
- Tích hợp bộ nhân tần số, ổn áp, DCXO and power management units
- Công suất đầu ra +25 dBm ở chế độ 802.11b.

2. Sơ đồ và chức năng các chân của ESP8266 NODEMCU



ESP8266 NodeMCU Pinout

Các nhóm chân chính và chức năng

Chân nguồn:

- **VCC:** Nguồn điện dương cho module, thường là 3.3V.
- **GND:** Nguồn điện âm (mass).
- **3V3:** Điện áp 3.3V, dùng để cấp nguồn cho các thiết bị ngoại vi.
- **5V:** Điện áp 5V, có thể dùng để cấp nguồn cho một số thiết bị ngoại vi.

Chân GPIO (General Purpose Input/Output):

- **D0, D1, D2, ...:** Các chân đa năng, có thể được cấu hình làm ngõ vào hoặc ngõ ra để điều khiển các thiết bị ngoại vi như LED, cảm biến, động

cơ, ...

- **A0:** Chân analog, dùng để đọc giá trị tương tự từ các cảm biến.

Chân giao tiếp:

- **TX, RX:** Dùng để giao tiếp với máy tính hoặc các thiết bị khác qua cổng UART.
- **SCL, SDA:** Dùng để giao tiếp với các thiết bị I2C.
- **SPI:** Giao tiếp SPI để kết nối với các thiết bị như flash, cảm biến.

Chân đặc biệt:

- **RST:** Reset module.
- **CH_PD:** Chip power down, dùng để tắt nguồn chip.
- **GPIO0, GPIO2:** Dùng để vào chế độ flash firmware.

III. Sensor - SHT31

1. Tổng quan về cảm biến SHT31

Cảm biến SHT31 là một trong những cảm biến đo độ ẩm và nhiệt độ kỹ thuật số được ưa chuộng nhất hiện nay. Với kích thước nhỏ gọn, độ chính xác cao và giao tiếp I2C đơn giản, SHT31 được ứng dụng rộng rãi trong nhiều lĩnh vực, từ nông nghiệp, khí tượng đến các thiết bị gia dụng thông minh.

SHT31 hỗ trợ hai địa chỉ I2C riêng biệt: 0x44 và 0x45. Điều này cho phép sử

dụng hai mô-đun SHT31 trên cùng một bus hoặc tránh xung đột địa chỉ với thiết bị khác trên bus. Chân AD xác định địa chỉ I2C của mô-đun. Chân này có điện trở kéo xuống tích hợp. Do đó, khi chân AD không được kết nối, địa chỉ I2C mặc định là 0x44 và khi kết nối nó với tín hiệu điện áp cao, địa chỉ I2C trở thành 0x45.

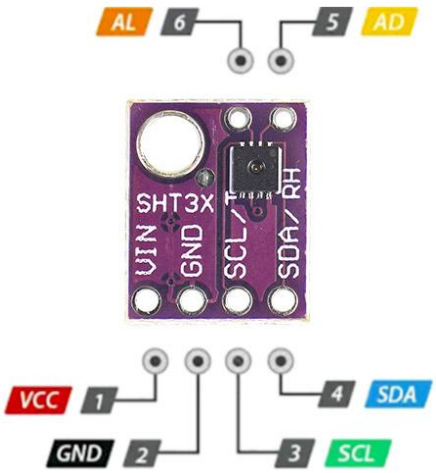
Nguyên lý hoạt động:

1. **Đo độ ẩm:** Phần tử cảm biến độ ẩm hấp thụ hơi nước trong không khí. Khi có dòng điện chạy qua, điện trở của phần tử này sẽ thay đổi tương ứng với lượng hơi ẩm hấp thụ.
2. **Đo nhiệt độ:** Điện trở nhiệt thay đổi giá trị theo nhiệt độ.
3. **Chuyển đổi tín hiệu:** Mạch điện tử bên trong cảm biến sẽ chuyển đổi các giá trị điện trở thành tín hiệu số và truyền đi qua giao diện I2C.

Thông số kỹ thuật điển hình:

- **Điện áp hoạt động:** 2.15V - 5.5V (tiêu chuẩn 3.3V)
- **Dải đo độ ẩm:** 0% - 100% RH
- **Dải đo nhiệt độ:** -40°C - 125°C
- **Độ phân giải:** 14 bit
- **Giao tiếp:** I2C với tốc độ liên lạc lên tới 1 MHz
- **Độ chính xác:** ± 2% rh và ± 0,3 ° C
- **Kích thước:** 11 x 13 mm

2. Sơ đồ và chức năng các chân của SHT31



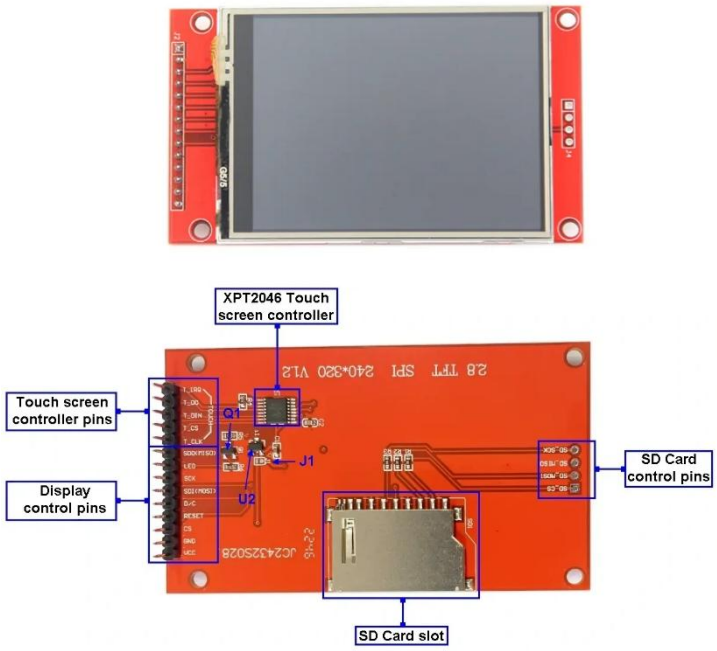
Cảm biến SHT31 có 6 chân chính:

- **VCC:** Cấp nguồn cho cảm biến, thường là 2.5V đến 5.5V.
- **GND:** Chân nối đất.
- **SCL:** Dây đồng hồ của giao tiếp I2C.
- **SDA:** Dây dữ liệu của giao tiếp I2C.
- AD: Chân chọn địa chỉ I2C.
- AL: Chân đầu ra cảnh báo.

IV. Display - ILI9341

1. Tổng quan về ILI9341

ILI9341 TFT Display Module



ILI9341 là một loại tích hợp mạch (IC) điều khiển màn hình TFT (Thin-Film Transistor) với độ phân giải 240x 320 pixel rất phổ biến. ILI9341 hỗ trợ chế độ hiển thị đầy đủ màu sắc, 8 màu và chế độ ngủ để kiểm soát nguồn điện chính xác bằng phần mềm. Nó được sử dụng rộng rãi trong các dự án điện tử, đặc biệt là các thiết bị nhúng, để hiển thị thông tin đồ họa màu sắc.

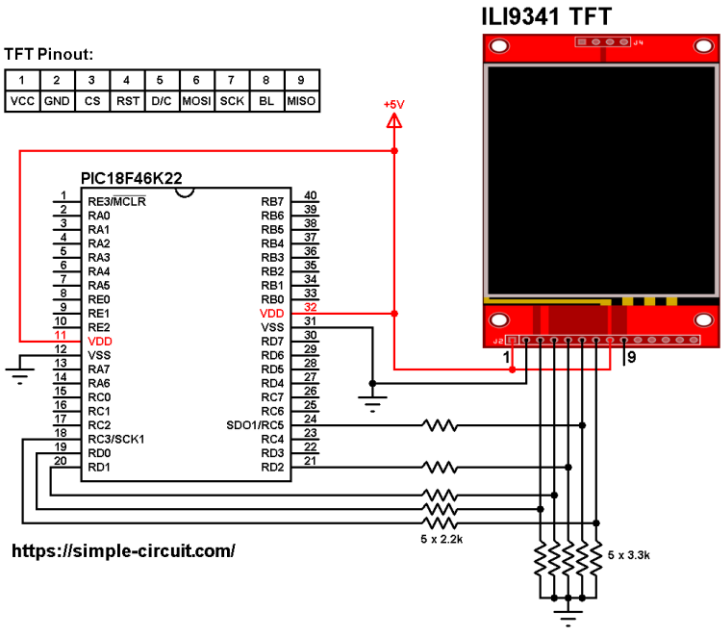
Các thông số kỹ thuật:

- **Độ phân giải:** Tùy thuộc vào loại màn hình và kích thước, độ phân giải có thể khác nhau (ví dụ: 240x320, 320x480).
- **Số màu:** Hỗ trợ 16-bit màu (65.536 màu) tới 18-bit màu (262k màu).
- **Điện áp hoạt động:** Thường là 3.3V hoặc 5V.
- **Giao tiếp:** SPI hoặc 8080.
- **Kích thước màn hình:** Tương thích với nhiều kích thước màn hình khác nhau.
- **Khả năng màn hình cảm ứng:** Hầu hết các màn hình ILI9341 đều có màn hình cảm ứng điện trở với bộ điều khiển màn hình cảm ứng tích

hợp (XPT2046).

- **Khe cắm thẻ SD:** Một số mô-đun hiển thị ILI9341 đi kèm với khe cắm thẻ SD, điều này giúp loại bỏ nhu cầu về mô-đun thẻ SD bên ngoài.

2. Sơ đồ chân và chức năng các chân của ILI9341



- **VCC:** Cấp nguồn cho IC, thường là 3.3V hoặc 5V.
- **GND:** Chân nối đất.
- **RS (Register Select):** Chọn chế độ đọc/ghi dữ liệu.
- **RW (Read/Write):** Chọn chế độ đọc hoặc ghi dữ liệu (một số IC không có chân này).
- **CS (Chip Select):** Chọn chip khi giao tiếp với nhiều thiết bị trên cùng một bus.
- **RD (Read):** Chân đọc dữ liệu.
- **WR (Write):** Chân ghi dữ liệu.
- **RESET:** Chân reset IC về trạng thái ban đầu.
- **SCL, SDA:** Dùng cho giao tiếp I2C (nếu có).
- **MOSI, MISO, SCK:** Dùng cho giao tiếp SPI (nếu có).

- **Các chân điều khiển khác:** Dùng để điều chỉnh độ sáng, màu sắc, chế độ hiển thị.

B. Connection Tech

I. Wired Connections

1. Giao thức UART

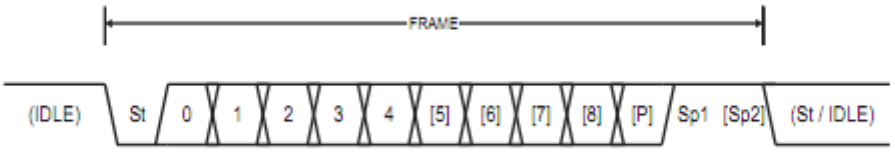
a. Khái quát về giao thức UART

UART (Universal Asynchronous Receiver-Transmitter) là một giao thức truyền thông nối tiếp không đồng bộ, được sử dụng rộng rãi để trao đổi dữ liệu giữa các thiết bị điện tử. Nói một cách đơn giản, UART cho phép các thiết bị khác nhau "nói chuyện" với nhau bằng cách truyền dữ liệu tuần tự, bit sau bit.

Một khung dữ liệu UART bao gồm các phần sau:

- **Bit bắt đầu:** Luôn có giá trị 0, đánh dấu bắt đầu một khung dữ liệu mới.
- **Các bit dữ liệu:** Mang thông tin cần truyền.
- **Bit chẵn lẻ (tùy chọn):** Dùng để kiểm tra lỗi.
- **Bit dừng:** Luôn có giá trị 1, đánh dấu kết thúc một khung dữ liệu.

Figure 19-4. Frame Formats



St	Start bit, always low.
(n)	Data bits (0 to 8).
P	Parity bit. Can be odd or even.
Sp	Stop bit, always high.
IDLE	No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.

b. Giao tiếp giữa ESP8266 NODEMCU và STM32F407VET6 thông qua UART

- **ESP8266 NodeMCU:**
  - TX: Kết nối với RX của STM32
  - RX: Kết nối với TX của STM32
  - GND: Kết nối chung
- **STM32F407VET6:**
  - TX: Kết nối với RX của ESP8266
  - RX: Kết nối với TX của ESP8266
  - GND: Kết nối chung

Cấu hình phần cứng

- **ESP8266 NodeMCU:**
  - **Tốc độ baud:** Cấu hình trong code Arduino hoặc các môi trường lập trình khác.
  - **Chân TX, RX:** Thường là các chân cố định, có thể tham khảo datasheet.
- **STM32F407VET6:**
  - **Tốc độ baud:** Cấu hình trong phần mềm lập trình STM32 (ví dụ: STM32CubeIDE).
  - **GPIO:** Chọn các chân GPIO tương ứng với TX và RX.
  - **Cấu hình đồng hồ:** Đảm bảo đồng hồ hệ thống hoạt động ổn định.
  - **Cấu hình USART:** Khởi tạo USART, cấu hình tốc độ baud, số bit dữ liệu, bit dừng và bit chẵn lẻ.

2. Giao thức I2C

a. Khái quát về giao thức I2C

I2C (Inter-Integrated Circuit) là một giao thức truyền thông nối tiếp đồng bộ, thường

được sử dụng để kết nối các thiết bị điện tử ở khoảng cách gần với nhau. Nó được phát triển bởi Philips Semiconductors và sử dụng hai dây tín hiệu chính:

- **SCL (Serial Clock Line):** Dây đồng hồ, xác định tốc độ truyền dữ liệu.
- **SDA (Serial Data Line):** Dây dữ liệu, truyền nhận dữ liệu.

Cách thức hoạt động

1. **Khởi tạo:** Thiết bị Master bắt đầu giao tiếp bằng cách kéo thấp SDA khi SCL ở mức cao (điều kiện bắt đầu).
2. **Gửi địa chỉ:** Master gửi địa chỉ của thiết bị Slave muốn giao tiếp.
3. **Xác nhận:** Slave nhận được địa chỉ sẽ trả lời một bit xác nhận.
4. **Truyền dữ liệu:** Master gửi dữ liệu và Slave nhận dữ liệu.
5. **Kết thúc:** Master kết thúc giao tiếp bằng cách tạo điều kiện kết thúc (giữ SDA cao khi thay đổi SCL từ thấp lên cao).

b. Giao tiếp giữa SHT31 và STM32F407VET6 thông qua I2C

Kết nối phần cứng:

- **Kết nối các chân:**
  - **VCC:** Nối chung với nguồn cấp của cả hai chip.
  - **GND:** Nối chung với ground của cả hai chip.
  - **SDA:** Nối chân SDA của SHT31 với chân SDA tương ứng trên STM32.
  - **SCL:** Nối chân SCL của SHT31 với chân SCL tương ứng trên STM32.
- **Kéo điện trở:** Để đảm bảo hoạt động ổn định của bus I2C, cần kéo điện trở lên các chân SDA và SCL. Giá trị điện trở thường là 4.7kΩ hoặc 10kΩ.

Cấu hình STM32:

- **Khởi tạo I2C:** Sử dụng thư viện HAL của STM32 để khởi tạo giao diện I2C, bao gồm cấu hình tốc độ truyền, địa chỉ thiết bị, và các thông số khác.
- **Gửi lệnh đọc/ghi:** Viết các hàm để gửi lệnh đọc hoặc ghi dữ liệu đến SHT31. Lệnh sẽ bao gồm địa chỉ của SHT31 và các register cần truy cập.

- **Xử lý dữ liệu:** Sau khi nhận được dữ liệu từ SHT31, tiến hành xử lý để tính toán giá trị độ ẩm và nhiệt độ thực tế.

Lập trình firmware:

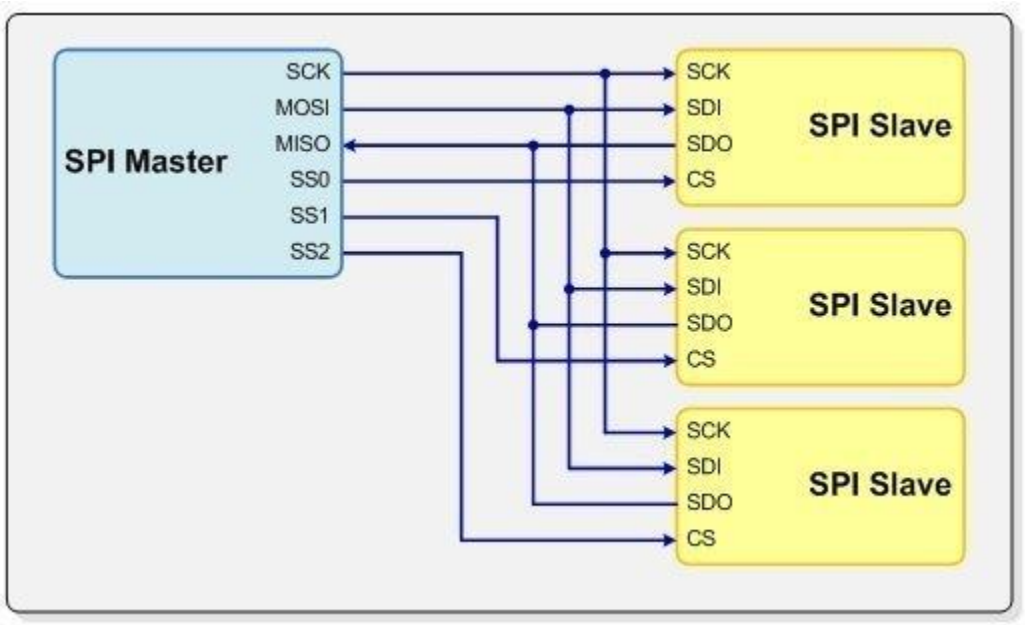
- **Sử dụng thư viện:** Có thể sử dụng các thư viện hỗ trợ sẵn để đơn giản hóa quá trình giao tiếp I2C.
- **Viết mã:** Viết code để thực hiện các tác vụ sau:
    - Khởi động I2C
    - Gửi lệnh kích hoạt đo
    - Đọc dữ liệu từ SHT31
    - Tính toán giá trị độ ẩm và nhiệt độ
    - Hiển thị kết quả trên màn hình LCD hoặc truyền dữ liệu qua UART.

3. Giao thức SPI
    a. Khái quát về giao thức SPI

SPI (Serial Peripheral Interface) là một giao thức truyền thông nối tiếp đồng bộ, thường được sử dụng trong các hệ thống nhúng để trao đổi dữ liệu tốc độ cao giữa các thiết bị. Nó cho phép một thiết bị chủ (master) giao tiếp với nhiều thiết bị phụ (slave) trong một cấu trúc bus đơn giản.

Cấu trúc của SPI:

- **MOSI (Master Out Slave In):** Dây dẫn dữ liệu từ master đến slave.
- **MISO (Master In Slave Out):** Dây dẫn dữ liệu từ slave đến master.
- **SCK (Serial Clock):** Dây xung đồng hồ để đồng bộ hóa quá trình truyền nhận dữ liệu.
- **SS (Slave Select):** Dây chọn slave, cho phép master chọn một slave cụ thể để giao tiếp.



Nguyên lý hoạt động:

1. **Khởi tạo:** Master tạo xung nhịp trên SCL và kéo thấp SDA để bắt đầu một giao tiếp.
2. **Gửi địa chỉ:** Master gửi địa chỉ 7 hoặc 10 bit của slave mà nó muốn giao tiếp.
3. **Xác nhận:** Slave nhận được địa chỉ của mình sẽ kéo thấp SDA để xác nhận.
4. **Truyền dữ liệu:** Master gửi dữ liệu đến slave hoặc nhận dữ liệu từ slave. Sau mỗi byte dữ liệu, slave sẽ kéo thấp SDA để xác nhận.
5. **Kết thúc:** Master kết thúc giao tiếp bằng cách thả SDA.

b. Giao tiếp giữa ILI9341 và STM32F407VET6 thông qua SPI cho chức năng cảm biến (touch)

SPI là một giao thức truyền thông nối tiếp đồng bộ, được sử dụng để giao tiếp giữa các thiết bị ngoại vi với vi điều khiển. Trong trường hợp này, SPI sẽ được sử dụng để truyền lệnh và dữ liệu từ STM32F407VET6 đến ILI9341, cũng như nhận dữ liệu cảm ứng từ ILI9341 về.

Kết nối phần cứng, các chân:

- **SCK:** Xung đồng hồ
- **MOSI:** Dữ liệu gửi từ STM32 đến ILI9341
- **MISO:** Dữ liệu nhận từ ILI9341 về STM32
- **DC:** Chọn chế độ dữ liệu hoặc lệnh
- **CS:** Chọn thiết bị (trong trường hợp này là ILI9341)
- **RST:** Reset màn hình
- **TOUCH:** Dây tín hiệu cảm ứng (nếu ILI9341 có tích hợp cảm ứng)

Cấu hình phần mềm STM32

- **Khởi tạo SPI:** Cấu hình tốc độ, chế độ hoạt động, và các thông số khác của giao tiếp SPI.
- **Gửi lệnh:** Gửi các lệnh điều khiển màn hình (ví dụ: khởi tạo màn hình, vẽ hình, hiển thị văn bản) đến ILI9341.
- **Đọc dữ liệu cảm ứng:** Đọc dữ liệu tọa độ từ ILI9341 khi có sự kiện chạm.

Quá trình hoạt động

- **Khởi tạo:** Khởi tạo giao tiếp SPI trên STM32.Gửi các lệnh khởi tạo cho ILI9341 để thiết lập các thông số như độ phân giải, màu sắc, hướng màn hình.
- **Hiển thị:** Gửi các lệnh vẽ hình, hiển thị văn bản đến ILI9341.
- **Đọc cảm ứng:** Đọc dữ liệu tọa độ từ ILI9341 khi có sự kiện chạm. Xử lý dữ liệu cảm ứng để thực hiện các hành động tương ứng.

II. Wireless Connections

1. Wifi (Web openweathermap.com)

Wifi là một công nghệ cho phép các thiết bị điện tử kết nối với nhau và Internet mà không cần sử dụng dây cáp. Thay vào đó, Wi-Fi sử dụng sóng vô tuyến để truyền dữ liệu.

Nhờ có Wi-Fi, các ứng dụng thời tiết trên các thiết bị có thể kết nối với các dịch vụ thời tiết như OpenWeatherMap.com để tải về dữ liệu thời tiết mới nhất.

OpenWeatherMap.com là một nền tảng cung cấp dữ liệu thời tiết trực tuyến miễn phí, được sử dụng rộng rãi bởi các nhà phát triển, ứng dụng và trang web trên toàn thế giới. Nói cách khác, đây là một "kho" dữ liệu thời tiết khổng lồ mà bất kỳ ai cũng có thể truy cập và sử dụng để xây dựng các ứng dụng hoặc dịch vụ liên quan đến thời tiết.

OpenWeatherMap.com cung cấp :

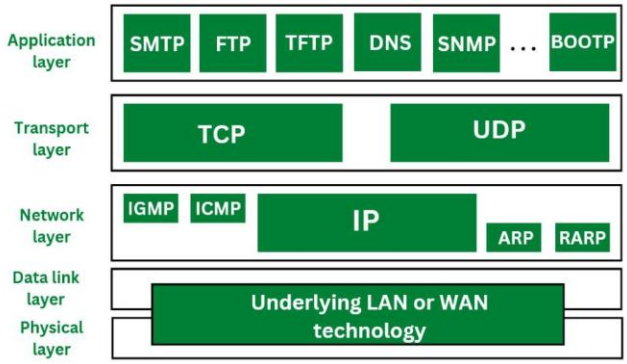
- **Dữ liệu thời tiết đa dạng:** Bạn có thể truy cập thông tin về nhiệt độ, độ ẩm, tốc độ gió, lượng mưa, áp suất khí quyển, và nhiều thông số thời tiết khác tại bất kỳ vị trí nào trên thế giới.
- **Dự báo thời tiết:** Không chỉ cung cấp thông tin thời tiết hiện tại, OpenWeatherMap còn cho phép bạn truy cập dự báo thời tiết trong nhiều ngày tới.
- **Bản đồ thời tiết:** Bạn có thể xem các bản đồ trực quan hiển thị các thông tin về nhiệt độ, lượng mưa, gió trên một khu vực rộng lớn.
- **API miễn phí:** OpenWeatherMap cung cấp một API (Giao diện lập trình ứng dụng) miễn phí, cho phép các nhà phát triển dễ dàng tích hợp dữ liệu thời tiết vào các ứng dụng của mình.

2. Protocol

Giao thức(protocol) trong lĩnh vực công nghệ thông tin có thể hiểu đơn giản là một **tập hợp các quy tắc, quy định và định dạng** được thống nhất, nhằm đảm bảo việc truyền thông và trao đổi dữ liệu giữa các thiết bị, phần mềm hoặc hệ thống khác nhau diễn ra một cách hiệu quả và chính xác.

Các loại giao thức phổ biến

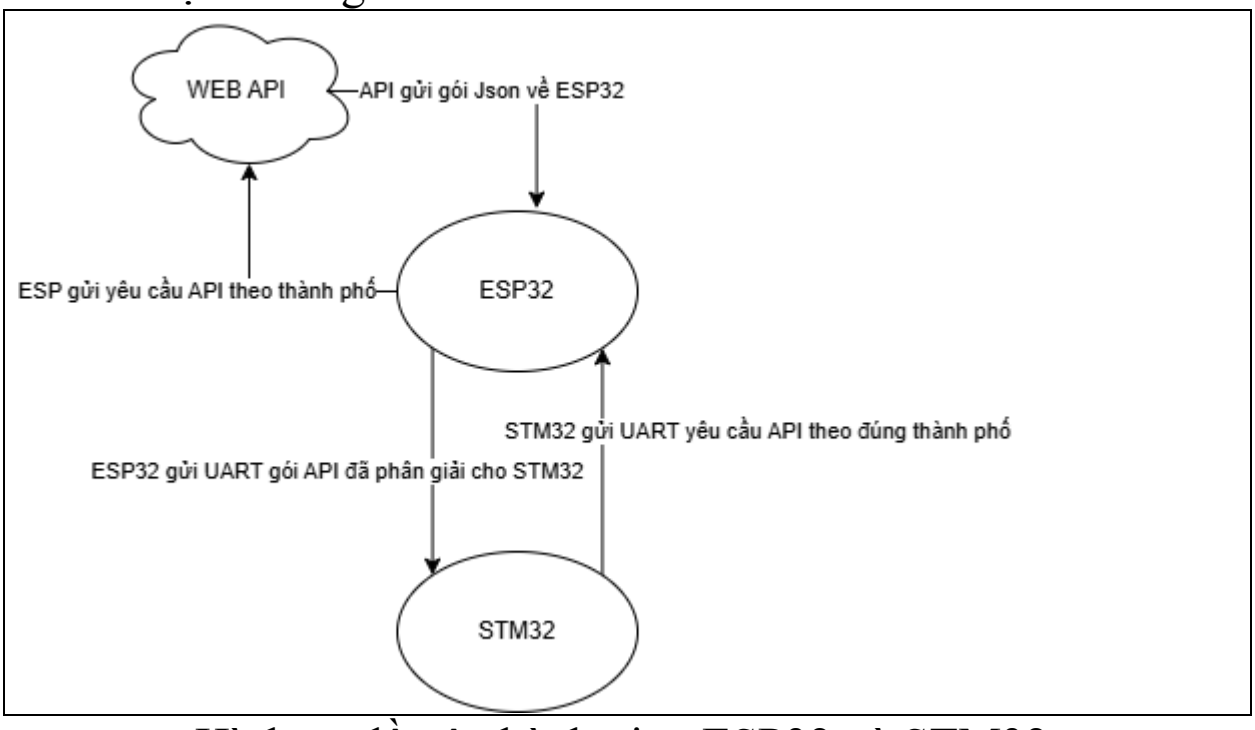
- **Giao thức truyền dữ liệu:**
  - **TCP/IP:** Giao thức truyền thông cơ bản của Internet.
  - **HTTP:** Giao thức truyền tải siêu văn bản, sử dụng để truy cập các trang web.
  - **FTP:** Giao thức truyền tệp.
  - **SMTP:** Giao thức đơn giản để chuyển thư.
  - **POP3, IMAP:** Giao thức lấy thư.
- **Giao thức mạng cục bộ:**
  - **Ethernet:** Giao thức mạng cục bộ phổ biến nhất.
  - **Wi-Fi:** Giao thức không dây.
  - **Bluetooth:** Giao thức không dây tầm ngắn.
- **Giao thức cấp thấp:**
  - **I2C:** Giao thức truyền thông nối tiếp đơn giản.
  - **SPI:** Giao thức truyền thông nối tiếp tốc độ cao.
  - **UART:** Giao thức truyền thông nối tiếp không đồng bộ.



CHƯƠNG 4: MÔ TẢ CÔNG NGHỆ

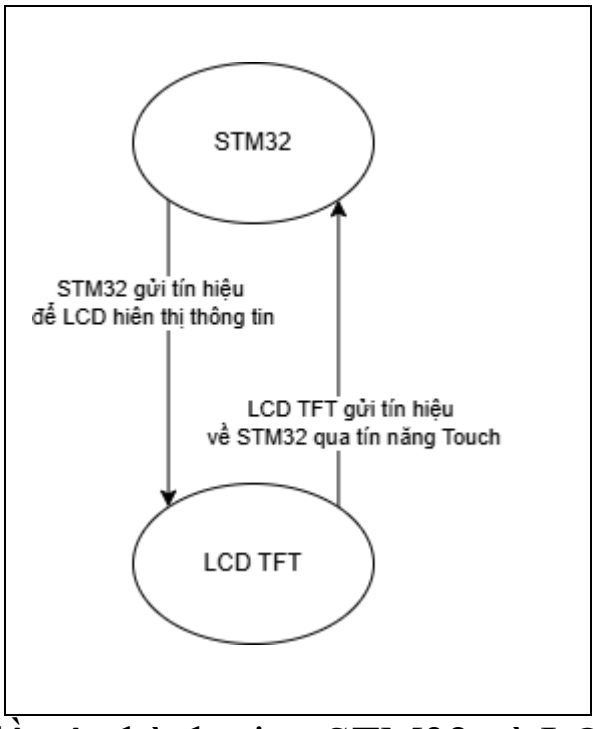
A Sơ đồ vận hành tính năng hệ thống:

I Sơ đồ vận hành giữa ESP32 và STM32:



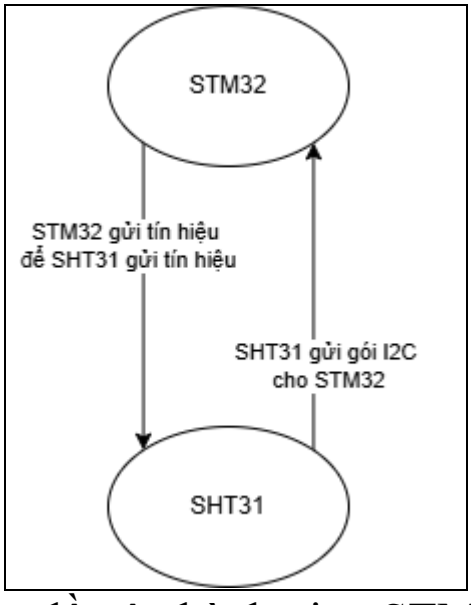
Hình sơ đồ vận hành giữa ESP32 và STM32

II Sơ đồ vận hành giữa STM32 và LCD TFT:



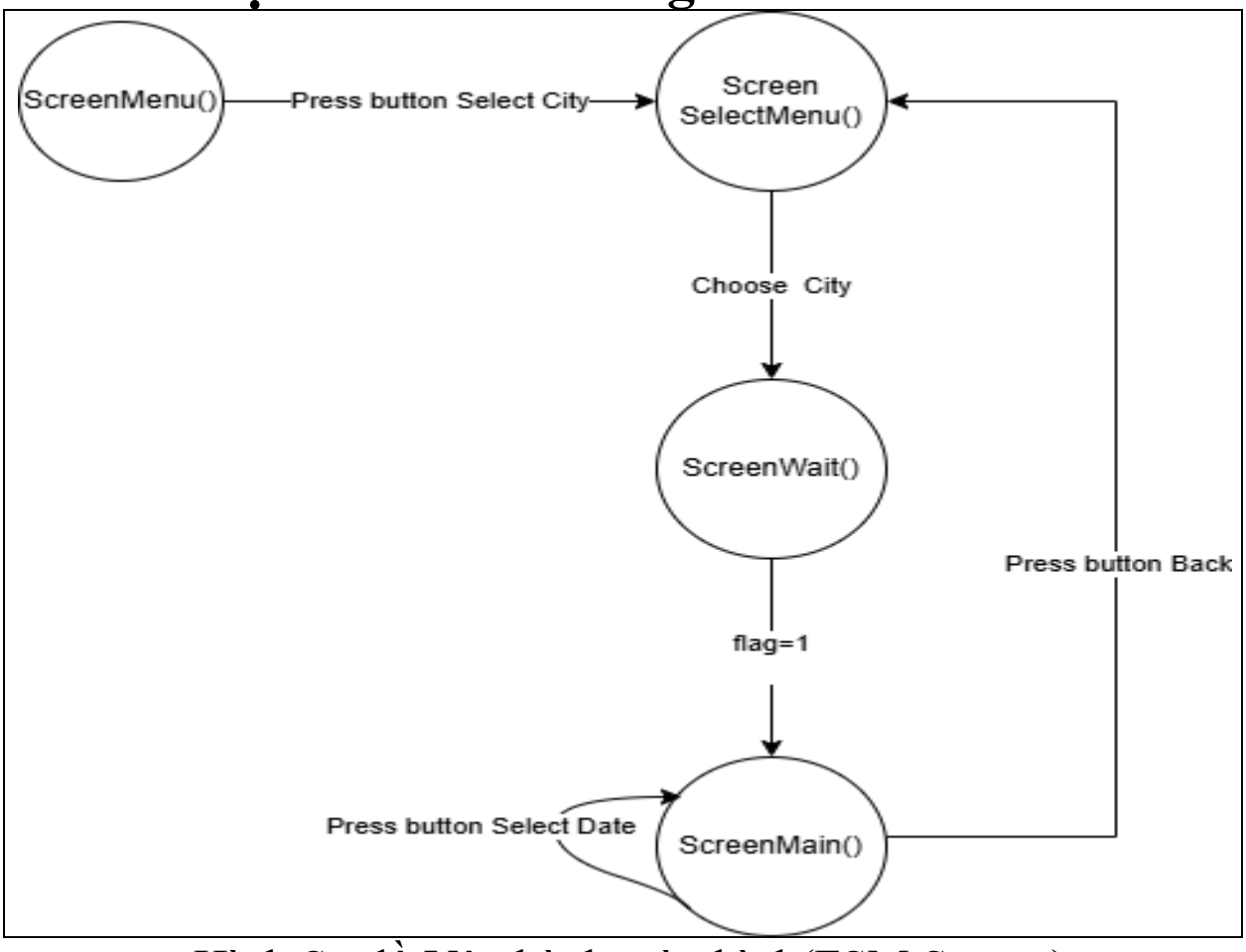
Hình sơ đồ vận hành giữa STM32 và LCD TFT

III Sơ đồ vận hành giữa STM32 và SHT31:



Hình sơ đồ vận hành giữa STM32 và SHT31

B Sơ đồ vận hành tính năng màn hình:



Hình Sơ đồ Vận hành màn hình(FSM Screen)

CHƯƠNG 4: SOFTWARE DESIGN AND IMPLEMENTATION

A. Đọc cảm biến SHT31

Sử dụng hàm **void SHT31_Read_Temp_Hum(float** *temperature, **float** *humidity) để đọc cảm biến SHT31.

- Gửi lệnh đo

HAL_I2C_Master_Transmit(&hi2c1, SHT31_ADDRESS, cmd, 2, HAL_MAX_DELAY);

- Đọc dữ liệu

HAL_I2C_Master_Receive(&hi2c1, SHT31_ADDRESS, data, 6, HAL_MAX_DELAY);

- Tính toán nhiệt độ

uint16_t raw_temp = (data[0] << 8) | data[1];

*temperature = -45 + (175 * (**float**)raw_temp / 65535.0);

- Tính toán độ ẩm

uint16_t raw_humidity = (data[3] << 8) | data[4];

*humidity = 100 * (**float**)raw_humidity / 65535.0;

B. Hiển thị screen, cảm ứng

Sử dụng các hàm và hằng số cụ thể của một thư viện đồ họa LCD để hiển thị màn hình.

lcdFillRGB();

- Dòng này tô màu nền của màn hình LCD,thường được sử dụng để xóa màn hình hoặc thay đổi màu nền.

lcdFillRoundRect(60, 25,120, 60, 10, COLOR_GREEN);

- Vẽ một hình chữ nhật tròn góc màu xanh lá cây (COLOR_GREEN) trên màn hình.
- Các tham số:
    - 60, 25: Tọa độ x, y của góc trên bên trái hình chữ nhật.
    - 120, 60: Chiều rộng và chiều cao của hình chữ nhật.
    - 10: Bán kính của góc tròn.

lcdSetCursor(x,y);

- Đặt vị trí con trỏ hiển thị văn bản tại tọa độ x và y trên màn hình.Các tọa độ này thường được tính toán để văn bản được hiển thị ở vị trí mong muốn trên màn hình.

lcdSetTextFont(&Font16);

- Thiết lập font chữ cho văn bản là Font16, tức là sử dụng font chữ có kích thước 16 pixel.

lcdSetTextColor(COLOR_, COLOR_);

- Thiết lập màu chữ và màu nền của chữ.Nếu sử dụng màu nền chữ khác với màu nền màn hình, văn bản sẽ được hiển thị với hiệu ứng trong suốt.

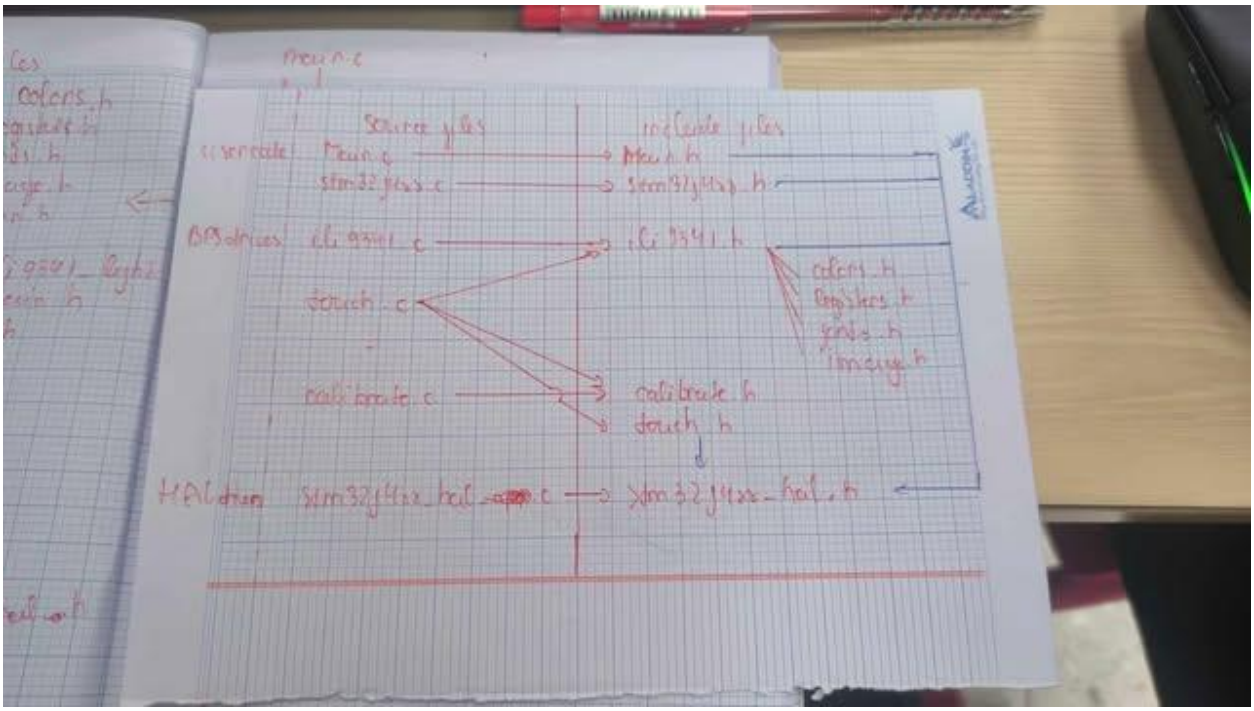
lcdPrintf("WEATHER \r\n");

- Hiển thị chuỗi văn bản "WEATHER" và xuống dòng (\r\n) tại vị trí con trỏ đã đặt.Hàm lcdPrintf() tương tự như hàm printf() trong ngôn ngữ C, cho phép định dạng và hiển thị chuỗi văn bản trên màn hình LCD.

Hàm TouchGetCalibratedPoint(&x, &y);

- thường được sử dụng trong lập trình vi điều khiển để lấy tọa độ (x, y) của một điểm chạm trên màn hình cảm ứng.

C. Sơ đồ các thư viện **A khan lười**



CHƯƠNG 5: SẢN PHẨM HOÀN THIỆN

A. Hình ảnh các màn hình thực tế:

		
ScreenMenu	ScreenSelectMenu()	ScreenWait()

 ScreenMain(Day1)	 ScreenMain(Day2)	 ScreenMain(Day6)

B Video demo sản phẩm:

Link Youtube

B. source code:

Link Drive

C. Ưu diểm và nhược điểm:

Ưu điểm:

- Lấy API nhanh, không bị lỗi gói tin truyền về
- Chuyển giữa các màn hình mượt, không bị lỗi màn hình.

Nhược điểm:

- Màn hình còn sơ xài, không có hình ảnh minh họa thành phố