**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**

**----oOo----**

**Text, logo

Description automatically generated**

**VI ĐIỀU KHIỂN VÀ ỨNG DỤNG**

**BÀI BÁO CÁO 11**

**NHÓM 8**

Giảng viên: **Phạm Quang Trí**

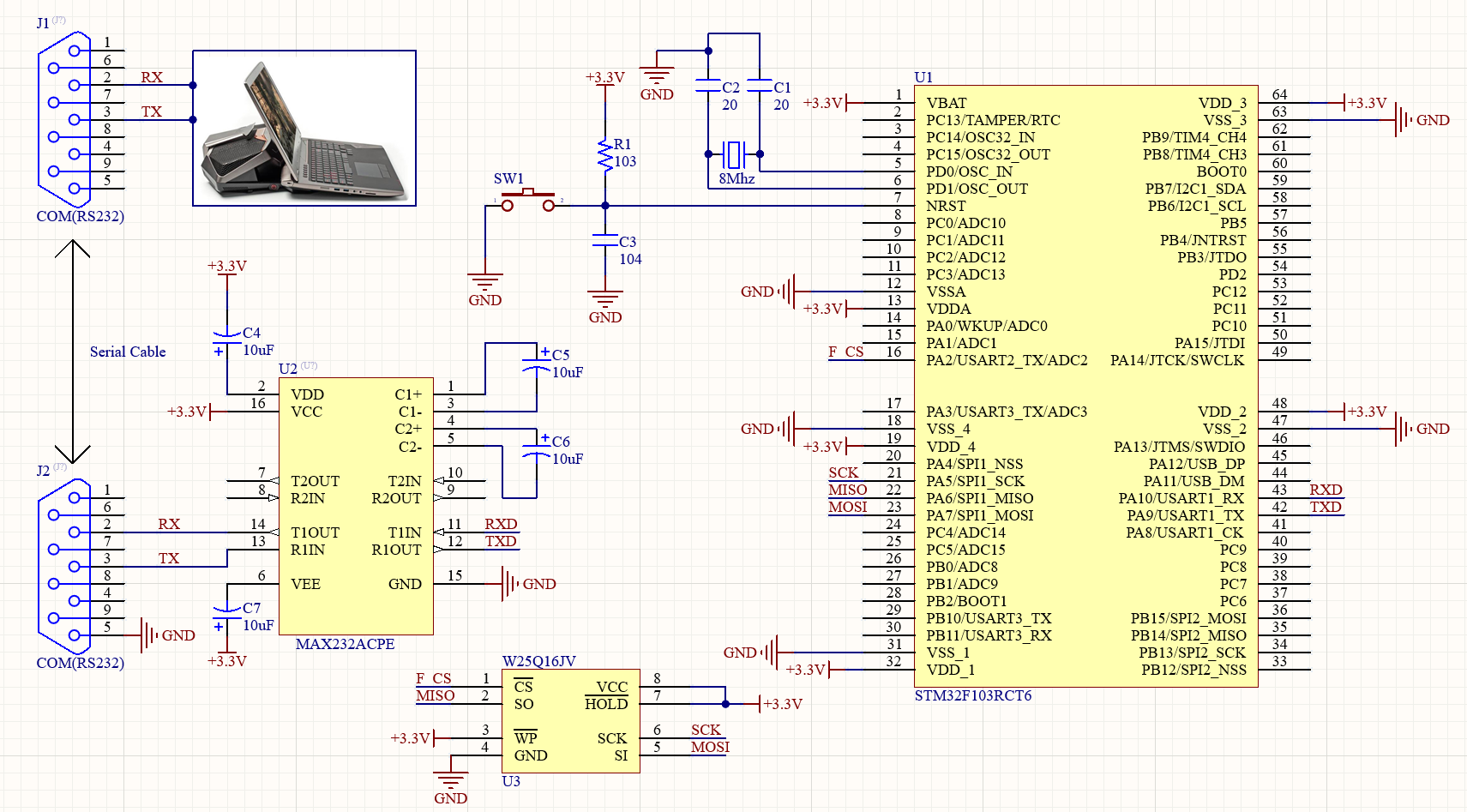
Sinh viên:

Nguyễn Ngọc Huy\_21115981

**TP.HCM 2023**

**Yêu cầu 1**: **Trong quyển báo cáo, dựa vào yêu cầu bài tập đã chọn, phải vẽ sơ đồ kết nối phần cứng của toàn bộ hệ thống sử dụng vi điều khiển STM32.**

* Sơ đồ kết nối phần cứng



**Yêu cầu 2: Trong quyển báo cáo, trình bày rõ ràng lưu đồ giải thuật và mã nguồn của chương trình điều khiển**

* Lưu đồ giải thuật

A

Khởi tạo 3 biến địa chỉ 'add\_h', 'add\_m', 'add\_l' để lưu 3 phần địa chỉ 'add'

Điều khiển cấm vi mạch nhớ hoạt động

Khởi tạo mảng 'data\_send' để lưu dữ liệu sẽ được truyền qua SPI

Delay 1ms

Điều khiển cho phép vi mạch nhớ hoạt động

Khởi tạo biến 'code' và gán giá trị 0x06, 0x06 là mã lệnh để bắt đầu quá trình ghi dữ liệu

Truyền 4 byte từ mảng 'data\_send' qua giao tiếp SPI

Tách địa chỉ 'add' thành 3 phần: byte cao (add\_h), byte giữa (byte\_m) ,byte thấp (add\_l)

Truyền dữ liệu từ mảng data với kích thước size thông qua giao tiếp SPI

Đặt giá trị 0x02 vào byte đầu tiên của mảng 'data\_send', 02 là 1 mã lệnh chỉ việc ghi dữ liệu vào vị trí cụ thể

Điều khiển cấm vi mạch nhớ hoạt động

Đặt giá trị của 'add\_h', 'add\_m', và 'add\_l' vào các byte tiếp theo của mảng 'data\_send'. Để cung cấp địa chỉ cụ thể trên thiết bị ngoại vi mà dữ liệu sẽ được ghi vào.

Delay 1ms

Điều khiển cho phép vi mạch nhớ hoạt động

Gửi một byte dữ liệu là mã lệnh từ biến 'code' đến thiết bị ngoại vi thông qua giao tiếp SPI

A

B

Khởi tạo 3 biến địa chỉ 'add\_h', 'add\_m', 'add\_l' để lưu 3 phần địa chỉ 'add'

Khởi tạo mảng 'data\_send' để lưu dữ liệu sẽ được truyền qua SPI

Điều khiển cấm vi mạch nhớ hoạt động

Delay 1ms

Tách địa chỉ 'add' thành 3 phần: byte cao (add\_h), byte giữa (byte\_m) ,byte thấp (add\_l)

Đặt giá trị 0x03 vào byte đầu tiên của mảng 'data\_send', 03 là 1 mã lệnh chỉ việc đọc dữ liệu từ vị trí cụ thể

Đặt giá trị của 'add\_h', 'add\_m', và 'add\_l' vào các byte tiếp theo của mảng 'data\_send'. Để cung cấp địa chỉ cụ thể trên thiết bị ngoại vi mà dữ liệu sẽ được đọc.

Điều khiển cho phép vi mạch nhớ hoạt động

Truyền 4 byte từ mảng 'data\_send' qua giao tiếp SPI

Đọc dữ liệu từ thiết bị ngoại vi thông qua giao tiếp SPI. Dữ liệu đọc được sẽ được lưu vào 'data' với kích thước 'size'.

B

* Mã nguồn chương trình

#include "main.h"

SPI\_HandleTypeDef hspi1;

UART\_HandleTypeDef huart1;

void write\_data(uint32\_t add, uint8\_t \*data, uint8\_t size)

{

uint8\_t add\_h, add\_m, add\_l;

uint8\_t data\_send[4];

uint8\_t code = 0x06;

add\_h = (add & 0xff0000) >> 16;

add\_m = (add & 0x00ff00) >> 8;

add\_l = add & 0xff;

data\_send[0] = 0x02;

data\_send[1] = add\_h;

data\_send[2] = add\_m;

data\_send[3] = add\_l;

HAL\_GPIO\_WritePin(F\_CS\_GPIO\_Port, F\_CS\_Pin, 0);

HAL\_SPI\_Transmit(&hspi1, &code, 1, 10);

HAL\_GPIO\_WritePin(F\_CS\_GPIO\_Port, F\_CS\_Pin, 1);

HAL\_Delay(1);

HAL\_GPIO\_WritePin(F\_CS\_GPIO\_Port, F\_CS\_Pin, 0);

HAL\_SPI\_Transmit(&hspi1, data\_send, 4, 10);

HAL\_SPI\_Transmit(&hspi1, data, size, 200);

HAL\_GPIO\_WritePin(F\_CS\_GPIO\_Port, F\_CS\_Pin, 1);

HAL\_Delay(1);

}

void read\_data(uint32\_t add, uint8\_t \*data, uint8\_t size)

{

uint8\_t add\_h, add\_m, add\_l;

uint8\_t data\_send[4];

add\_h = (add & 0xff0000) >> 16;

add\_m = (add & 0x00ff00) >> 8;

add\_l = add & 0xff;

data\_send[0] = 0x03;

data\_send[1] = add\_h;

data\_send[2] = add\_m;

data\_send[3] = add\_l;

HAL\_GPIO\_WritePin(F\_CS\_GPIO\_Port, F\_CS\_Pin, 0);

HAL\_SPI\_Transmit(&hspi1, data\_send, 4, 10);

HAL\_SPI\_Receive(&hspi1, data, size, 200);

HAL\_GPIO\_WritePin(F\_CS\_GPIO\_Port, F\_CS\_Pin, 1);

HAL\_Delay(1);

}

void SystemClock\_Config(void);

static void MX\_GPIO\_Init(void);

static void MX\_SPI1\_Init(void);

static void MX\_USART1\_UART\_Init(void);

uint8\_t \*data\_send = "The tranquil sea glistened, kissed by the golden sun, creating a mesmerizing and peaceful scene.\n";

uint8\_t data\_re[98];

int main(void)

{

HAL\_Init();

SystemClock\_Config();

MX\_GPIO\_Init();

MX\_SPI1\_Init();

MX\_USART1\_UART\_Init();

write\_data(90000, data\_send, 98);

read\_data(90000, data\_re, 98);

while (1)

{

HAL\_UART\_Transmit(&huart1, &data\_re[0], 98, 20);

HAL\_Delay(1500);

}

}

**Yêu cầu 3: Nạp chương trình vào kit thí nghiệm STM32 và thực hiện cho chạy thử trên phần cứng trong phòng thực hành. Quay video clip minh chứng kết quả thực hiện, tải lên Youtube duy nhất 1 video clip ( trong trường hợp minh chứng có nhiều video clip nhỏ thì sinh viên phải tự ghép lại thành 1 video clip tổng hợp ) và ghi liên kết vào báo cáo.**

Liên kết video: