

## Chương 4: Truyền thông nối tiếp (Ghép nối nối tiếp)

4.1 SPI

4.2 I<sup>2</sup>C

4.3 I<sup>2</sup>S

4.4 UART

4.5 USB

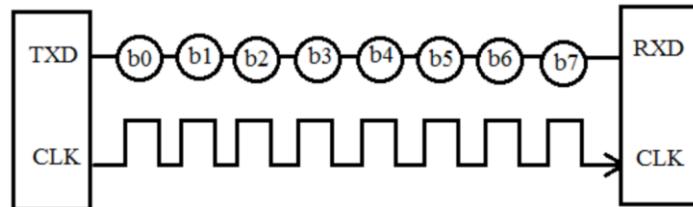
# Giới thiệu

## ❑ Truyền dữ liệu song song

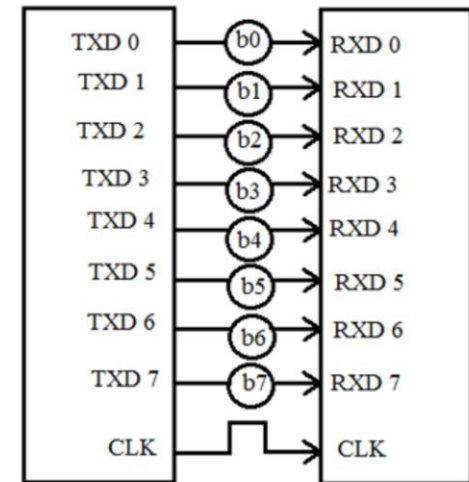
- | Truyền nhận đồng thời nhiều bit dữ liệu. Cần nhiều chân GPIO.
- | Cần có bus điều khiển, địa chỉ, dữ liệu. VD: LCD 1602

## ❑ Truyền dữ liệu nối tiếp

- | Truyền nhận theo từng bit nối tiếp nhau theo chuỗi.
- | Cần có cơ chế xác định chu kỳ/tần số của chuỗi bit.
- | Cần ít chân vào ra.



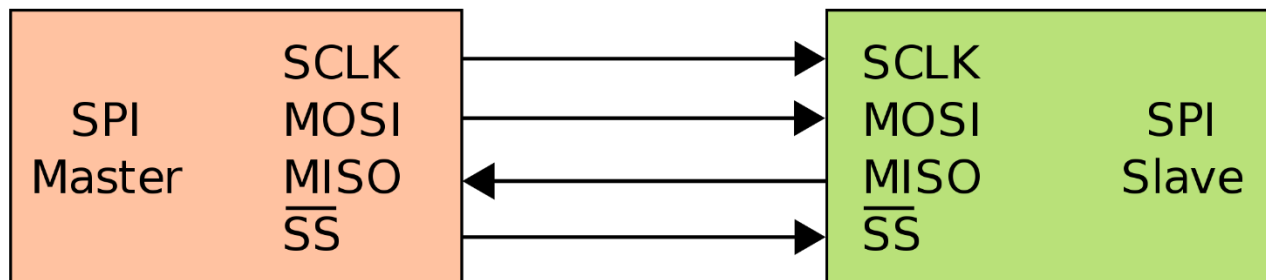
Truyền thông nối tiếp



Truyền thông song song

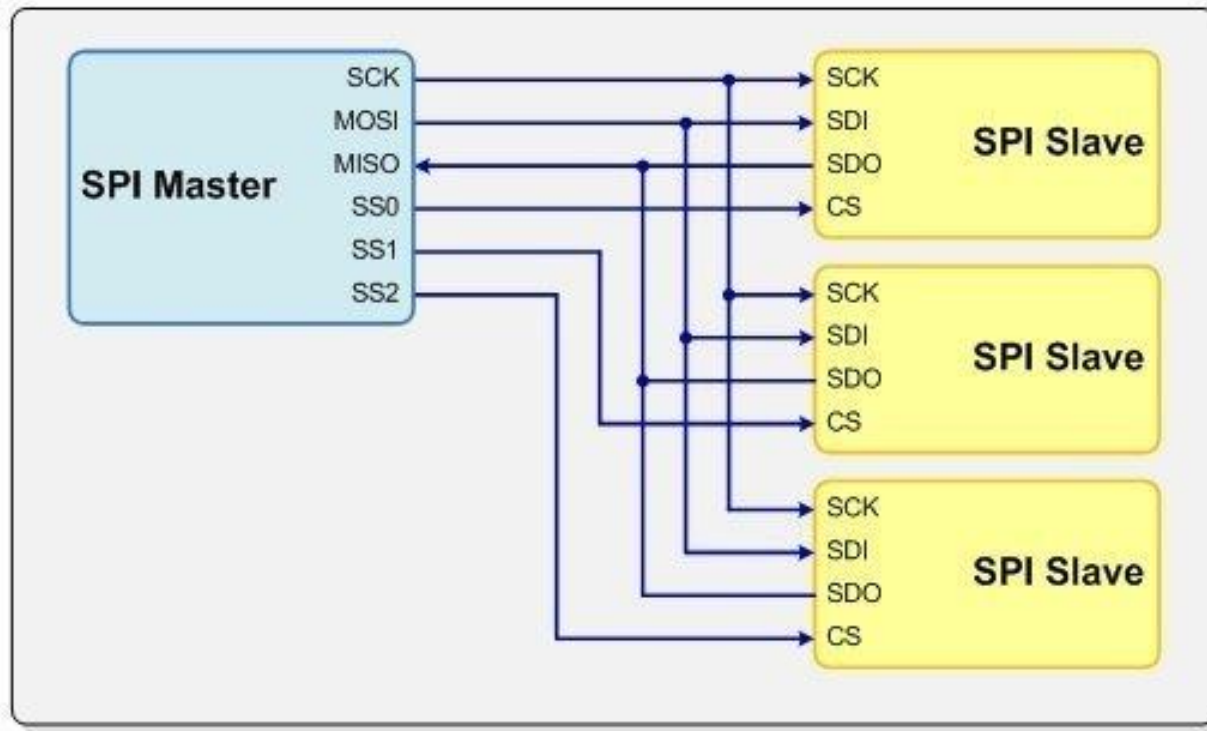
## 4.1 SPI

- ❑ Serial Peripherals Interface, rất thông dụng để ghép nối các ngoại vi tốc độ cao trên cùng một mạch.
  - | Thẻ nhớ, chip nhớ flash memory, MEM sensors, displays,...
- ❑ Là chuẩn truyền nối tiếp tốc độ cao, đồng bộ, hai chiều đồng thời.
- ❑ Thiết bị truyền nhận được chia thành 2 vai: Master và Slave.
- ❑ 4 tín hiệu
  - | MOSI (Master Out Slave In).
  - | MISO (Master In Slave Out).
  - | SCLK (Serial Clock): tạo ra bởi Master
  - | CS hay SS (Chip Select, Slave Select).
- ❑ Một số thiết bị slave đặt tên tín hiệu là SDI, SDO, SCK.



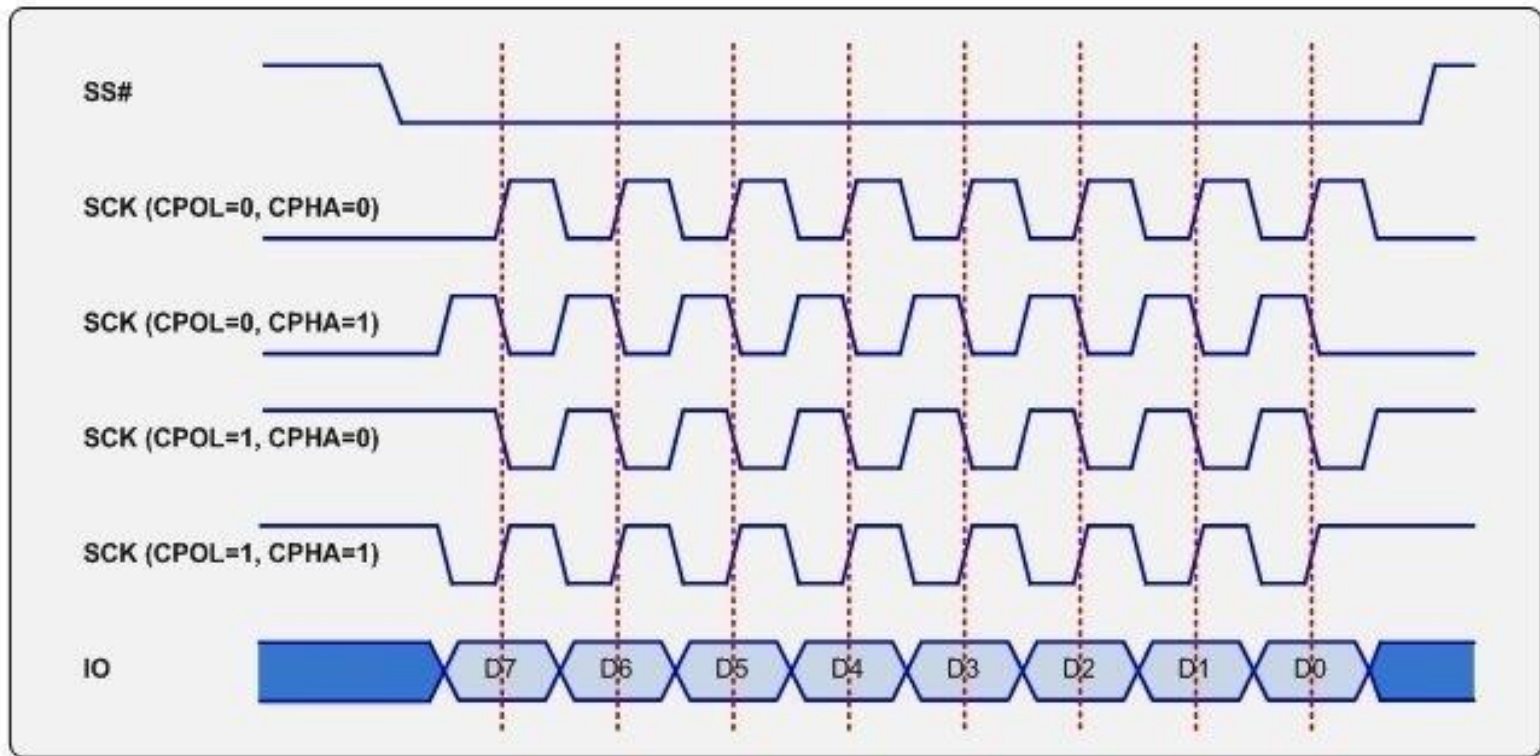
## SPI config

- ❑ Ghép nối một Master và nhiều Slave đồng thời



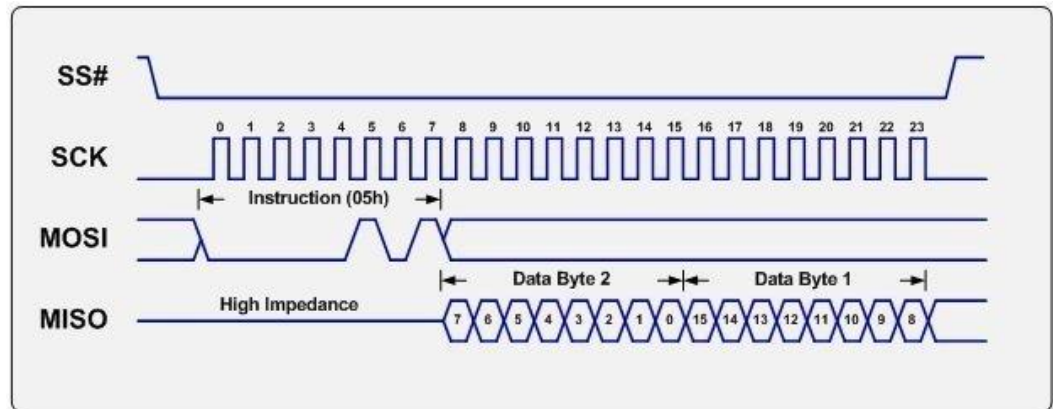
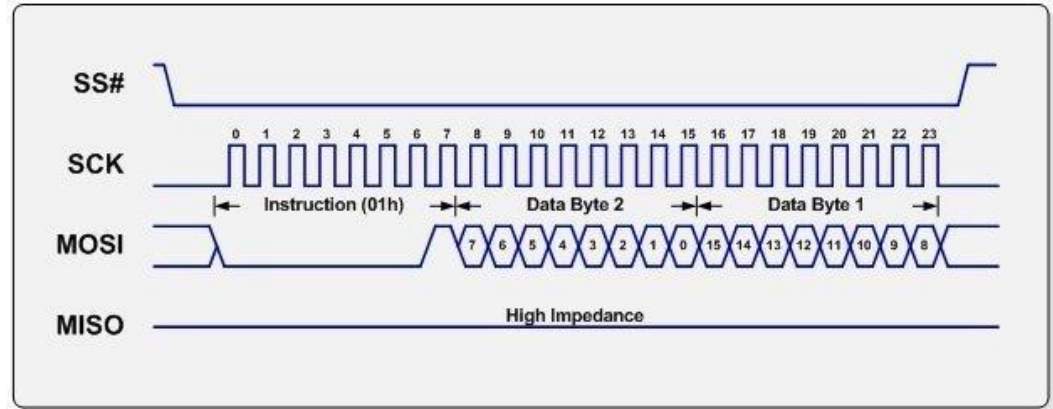
# SPI mode

❑ 4 modes: Mode 0 – Mode 3



# SPI transaction

- ❑ Master sends command
- ❑ Master sends data (write transaction) or
- ❑ Master reads data (read transaction)



# Lập trình với SPI

---

- ❑ Đọc tài liệu UM1725 tìm hiểu các hàm sau

*HAL\_SPI\_Init()*

*HAL\_SPI\_DeInit()*

*HAL\_SPI\_MspltInit()*

*HAL\_SPI\_MspDeInit()*

*HAL\_SPI\_RegisterCallback()*

*HAL\_SPI\_UnRegisterCallback()*

*HAL\_SPI\_Transmit()*

*HAL\_SPI\_Receive()*

*HAL\_SPI\_TransmitReceive()*

*HAL\_SPI\_Transmit\_IT()*

*HAL\_SPI\_Receive\_IT()*

*HAL\_SPI\_TransmitReceive\_IT()*

*HAL\_SPI\_Transmit\_DMA()*

*HAL\_SPI\_Receive\_DMA()*

*HAL\_SPI\_TransmitReceive\_DMA()*

*HAL\_SPI\_Abort()*

*HAL\_SPI\_Abort\_IT()*

*HAL\_SPI\_DMAMPause()*

*HAL\_SPI\_DMAResume()*

*HAL\_SPI\_DMAStop()*

*HAL\_SPI\_IRQHandler()*

*HAL\_SPI\_TxCpltCallback()*

*HAL\_SPI\_RxCpltCallback()*

*HAL\_SPI\_TxRxCpltCallback()*

*HAL\_SPI\_TxHalfCpltCallback()*

*HAL\_SPI\_RxHalfCpltCallback()*

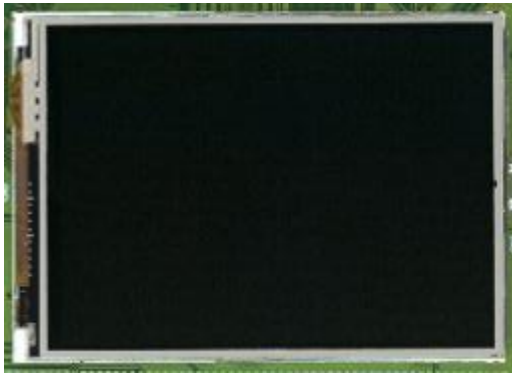
*HAL\_SPI\_TxRxHalfCpltCallback()*

*HAL\_SPI\_ErrorCallback()*

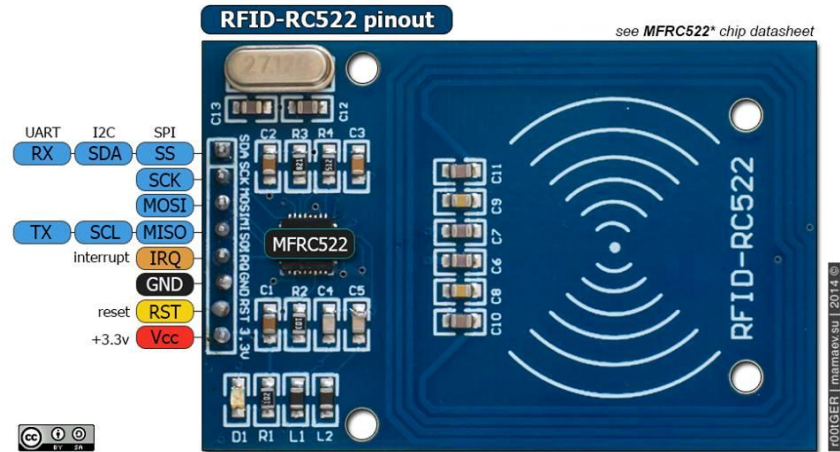
*HAL\_SPI\_AbortCpltCallback()*

# Ví dụ

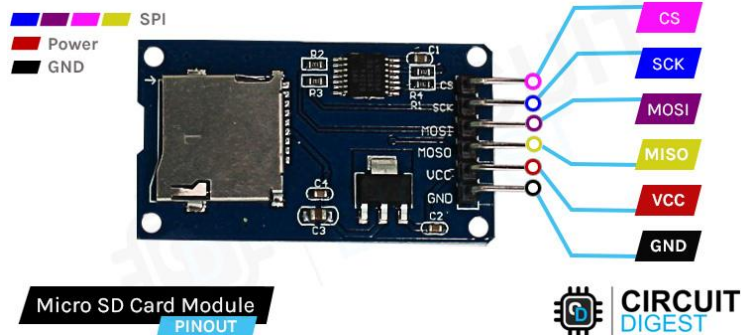
- ❑ Một số linh kiện ghép nối qua chuẩn SPI



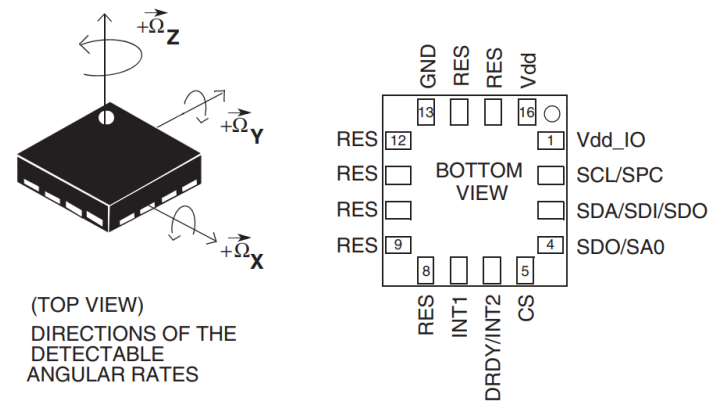
ILI9431 Graphics LCD Controller



RC522 RFID controller



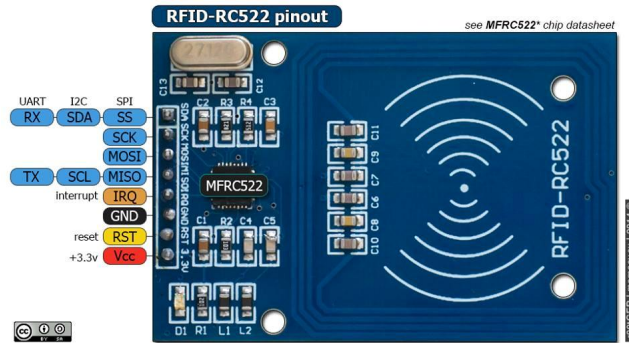
Micro SD card





# Bài tập 1: Ghép nối STM32F429 với RC522

- ❑ RC522: module RFID 13.56 MHz, hỗ trợ SPI, I2C, UART  
→ Module trong hộp làm việc ở chế độ SPI.



RC522 RFID module



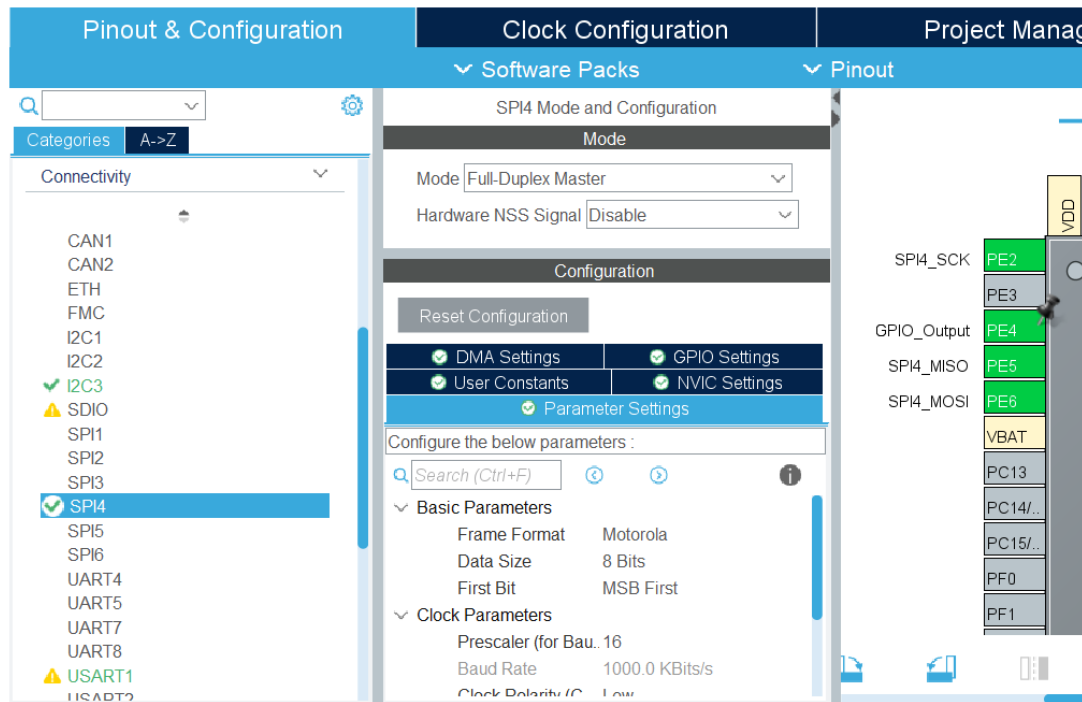
Một số loại tag RFID 13.56 MHz

- ❑ Kết nối RC522 với SPI4 trên STM32F429

SS	→ PE4	cần cấu hình GPIO riêng
SCK	→ PE2 (SPI4_SCK)	cấu hình theo SPI
MISO	→ PE5 (SPI4_MISO)	
MOSI	→ PE6 (SPI4_MOSI)	

# Bài tập: Ghép nối STM32F429 với RC522

- ❑ Tạo project mới, cấu hình các thông số
  - | GPIO: chân PE4 chế độ Output Push-pull
  - | USART1: 115 kbps
  - | SPI4: Full-Duplex Master với các thông số dưới đây



## Bài tập: Ghép nối STM32F429 với RC522

- ❑ Thêm các file tm\_stm32f4\_mfrc522.\* vào project.
- ❑ Thêm mã nguồn trong hàm main().
- ❑ Chạy và kiểm tra kết quả.

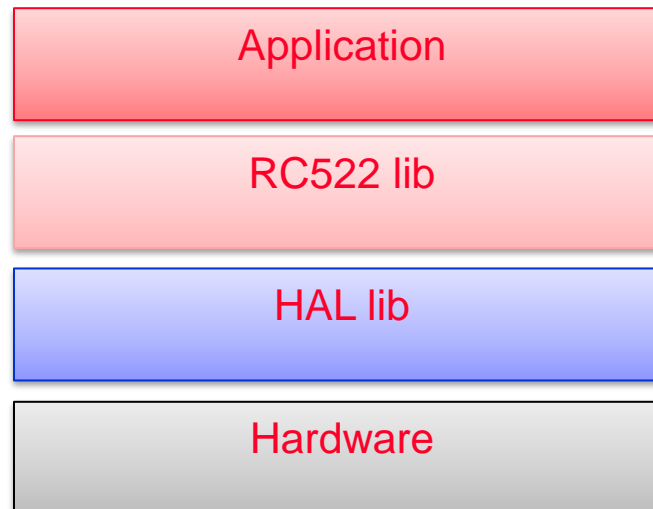
```
char buf[100];
TM_MFRC522_Init();
while (1)
{
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
    uint8_t CardID[5];
    HAL_Delay(100);
    if (TM_MFRC522_Check(CardID) == MI_OK) {
        sprintf (buf, "%s", "RFID Card Found!\r\n");
        HAL_UART_Transmit(&huart1,
                          (const uint8_t*) buf, strlen(buf), 2);
        HAL_Delay(300);
    }
    else{
        sprintf (buf, "%s", "RFID Card Not Found!\r\n");
        HAL_UART_Transmit(&huart1,
                          (const uint8_t*) buf, strlen(buf), 2);
        HAL_Delay(300);
    }
}
```

# Tìm hiểu hoạt động của project RC522

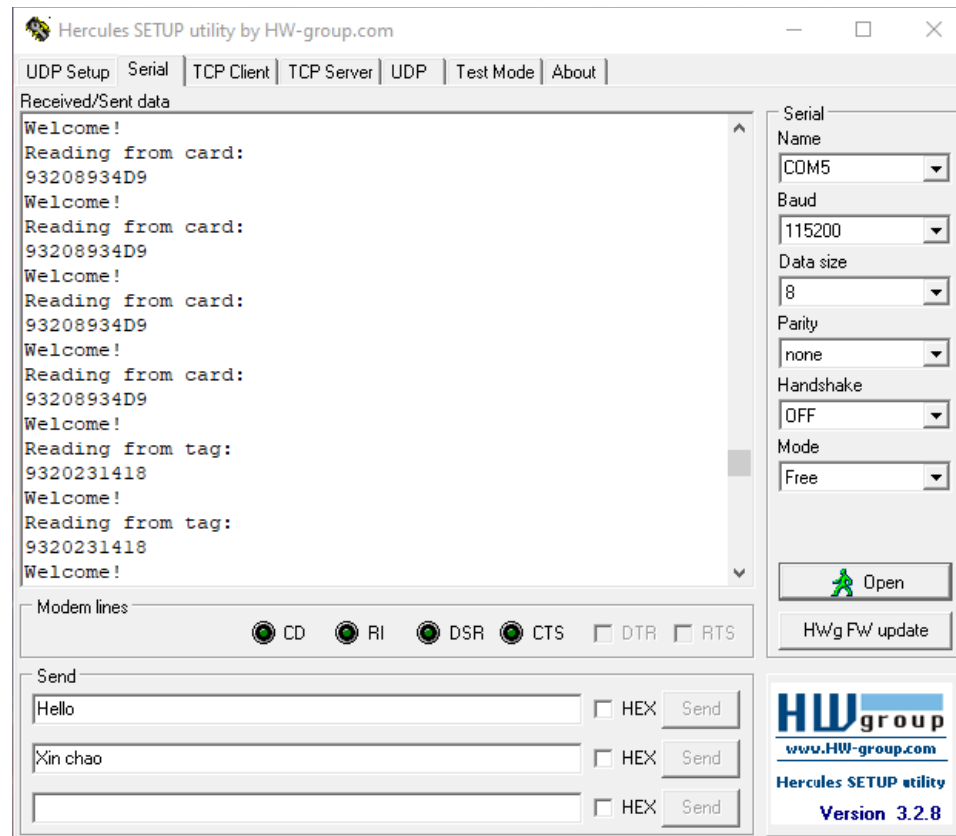
---

- ❑ Hardware: phần cứng gồm STM32F4 + RC522
- ❑ HAL lib (SPI driver): cung cấp hàm điều khiển module ngoại vi SPI trong STM32F4 cho phép gửi/nhận dữ liệu mức thấp qua SPI.
- ❑ RC522 lib (RC522 driver): cung cấp tập hàm giao tiếp mức cao với RC522 để đọc/ghi dữ liệu trên thẻ RFID (thao tác ghi chưa được hỗ trợ).
- ❑ Application: sử dụng RC522 lib cho các chức năng liên quan RFID để xây dựng ứng dụng cuối.



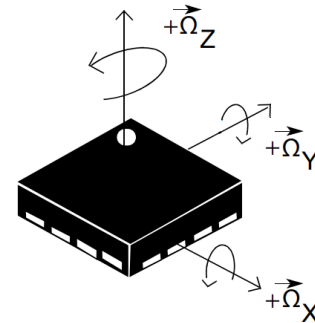
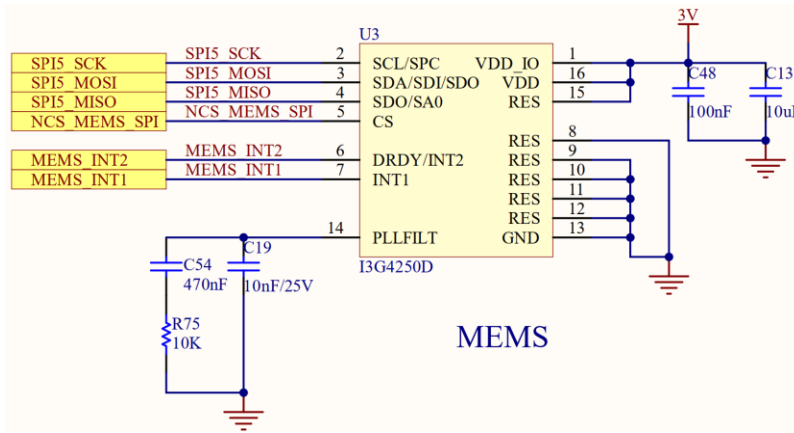
# Bài tập: Ghép nối STM32F429 với RC522

- ❑ Tự thêm code vào project để đọc mã thẻ RFID và gửi về PC qua cổng USART1.

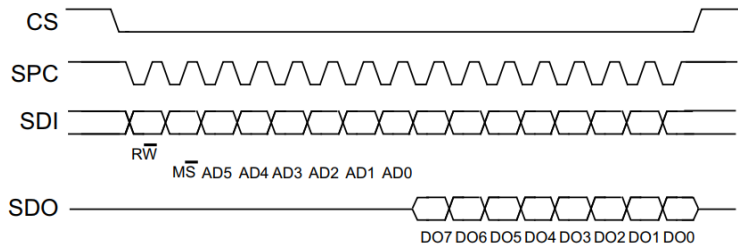
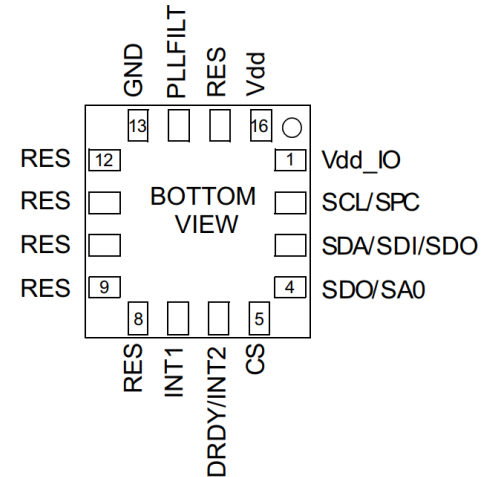


## Bài tập 2: Ghép nối STM32F429 với I3G4250D

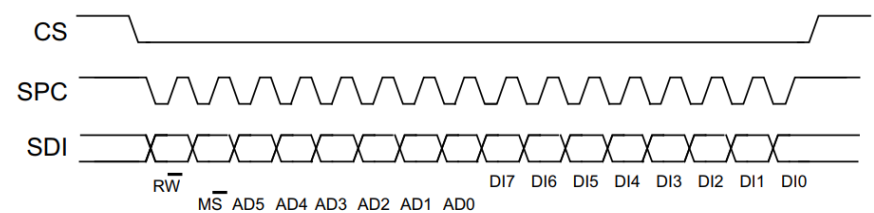
- I3G4250D là cảm biến gia tốc góc có sẵn trên board, nối với STM32F429 qua cổng SPI5.



(TOP VIEW)  
DIRECTIONS OF THE  
DETECTABLE  
ANGULAR RATES



SPI read



SPI write

## Bài tập 2: Ghép nối STM32F429 với I3G4250D

- ❑ Viết chương trình cho STM32F429 để đọc giá trị các gia tốc góc I3G4250D theo 3 trục
  - | Gửi các giá trị gia tốc góc đến PC qua USART1.
  - | Bật LED3 nếu giá trị AngleX > 0.
  - | Bật LED4 nếu AngleY > 0.
  - | Bật cả LED3 và LED4 nếu giá trị AngleZ > 0.

# Bài tập: Ghép nối STM32F429 với I3G4250D

- ❑ Tạo project mới, set CPU clock 180 MHz
- ❑ Cấu hình cổng SPI5 trên board STM32F429

The screenshot displays the STM32CubeMX interface for configuring the SPI5 peripheral. The left sidebar shows the device tree with SPI5 selected. The main window shows the configuration for SPI5, including Mode (Full-Duplex Master), Hardware NSS Signal (Disable), and various parameters like Frame Format (Motorola), Data Size (8 Bits), First Bit (MSB First), Prescaler (16), Baud Rate (5.625 MBits/s), Clock Polarity (CPOL) (Low), Clock Phase (CPHA) (1 Edge), CRC Calculation (Disabled), and NSS Signal Type (Software). The right sidebar shows the pin list with SPI5\_SCK (PF7), SPI5\_MISO (PF8), and SPI5\_MOSI (PF9) highlighted.

Category	Parameter	Value
Mode	Mode	Full-Duplex Master
Mode	Hardware NSS Signal	Disable
Configuration	Reset Configuration	
Configuration	NVIC Settings	✓
Configuration	DMA Settings	✓
Configuration	GPIO Settings	✓
Configuration	Parameter Settings	✓
Configuration	User Constants	✓
Basic Parameters	Frame Format	Motorola
Basic Parameters	Data Size	8 Bits
Basic Parameters	First Bit	MSB First
Clock Parameters	Prescaler (for Baud Rate)	16
Clock Parameters	Baud Rate	5.625 MBits/s
Clock Parameters	Clock Polarity (CPOL)	Low
Clock Parameters	Clock Phase (CPHA)	1 Edge
Advanced Parameters	CRC Calculation	Disabled
Advanced Parameters	NSS Signal Type	Software

Pin	Function
PF3	
PF4	
PF5	
VSS	
VDD	
PF6	
SPI5_SCK	PF7
SPI5_MISO	PF8
SPI5_MOSI	PF9
PF10	
RCC_OSC_IN	PH0/..
RCC_OSC_OUT	PH1/..
	NRST
GPIO_Output	PC0
	PC1
	PC2
	PC3
	VDD
	VSSA
	VREF+



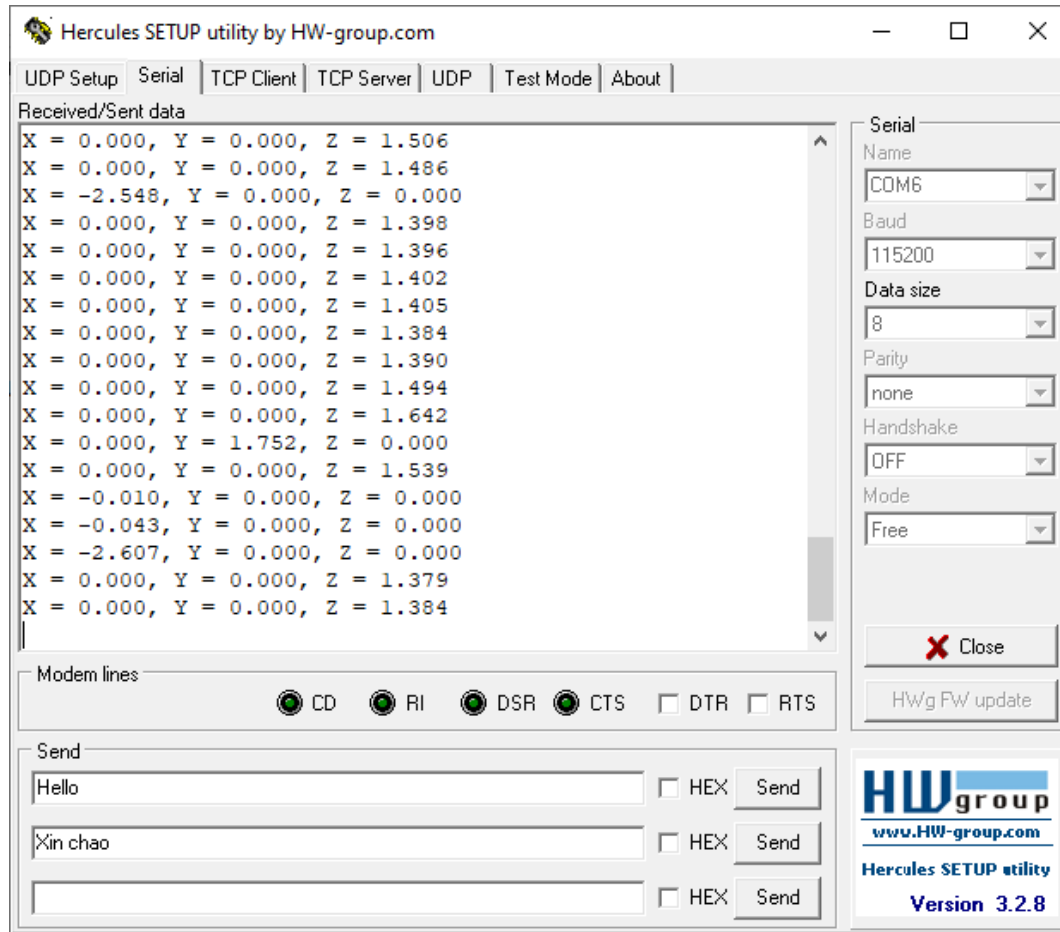
## Bài tập: Ghép nối STM32F429 với I3G4250D

- ❑ Thêm các file thư viện của L3GD20 vào project.
- ❑ Thêm mã nguồn trong hàm main().
- ❑ Chạy và kiểm tra kết quả.

```
95  L3GD20_Init();
96  /* USER CODE END 2 */
97
98  /* Infinite loop */
99  /* USER CODE BEGIN WHILE */
100 while (1)
101 {
102     /* USER CODE END WHILE */
103     /* USER CODE BEGIN 3 */
104     L3GD20_loop();
105     HAL_Delay(1);
106     float X = get_Angle_X(); float Y = get_Angle_Y(); float Z = get_Angle_Z();
107
108     if (X > 0 && Y > 0 && Z > 0) {
109         for (int i = 0; i < 5; i++) {
110             HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_13);
111             HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_14);
112             HAL_Delay(1000);
113         }
114         X = 0; Y = 0; Z = 0;
115     }
116     else {
117         HAL_GPIO_WritePin(GPIOG, GPIO_PIN_13, GPIO_PIN_RESET);
118         HAL_GPIO_WritePin(GPIOG, GPIO_PIN_14, GPIO_PIN_RESET);
119         X = 0; Y = 0; Z = 0;
120     }
121 }
```

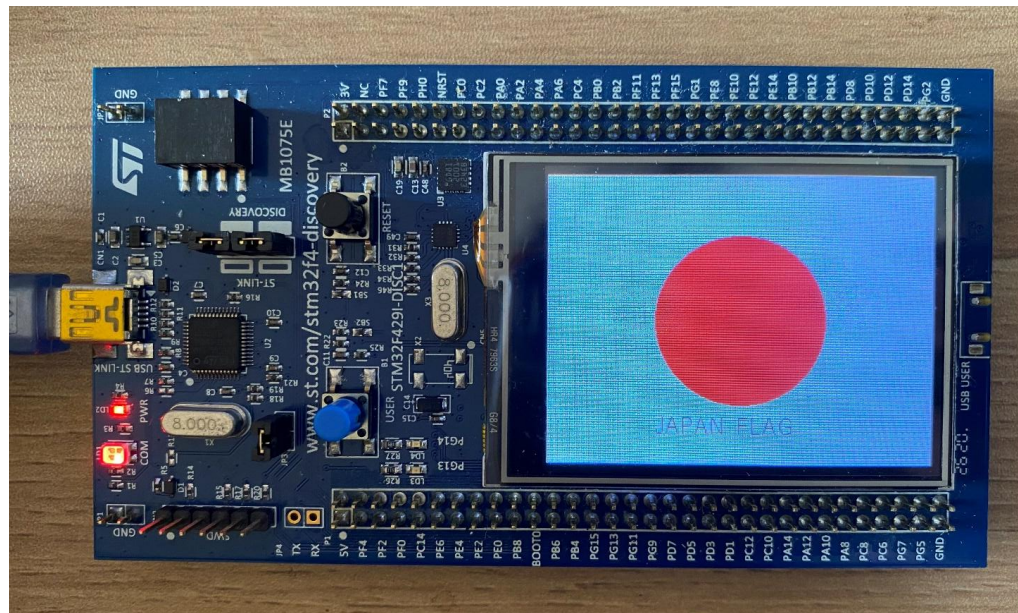
# Bài tập: Ghép nối STM32F429 với I3G4250D

- ❑ Tự làm: thêm code để gửi dữ liệu liên tục về PC



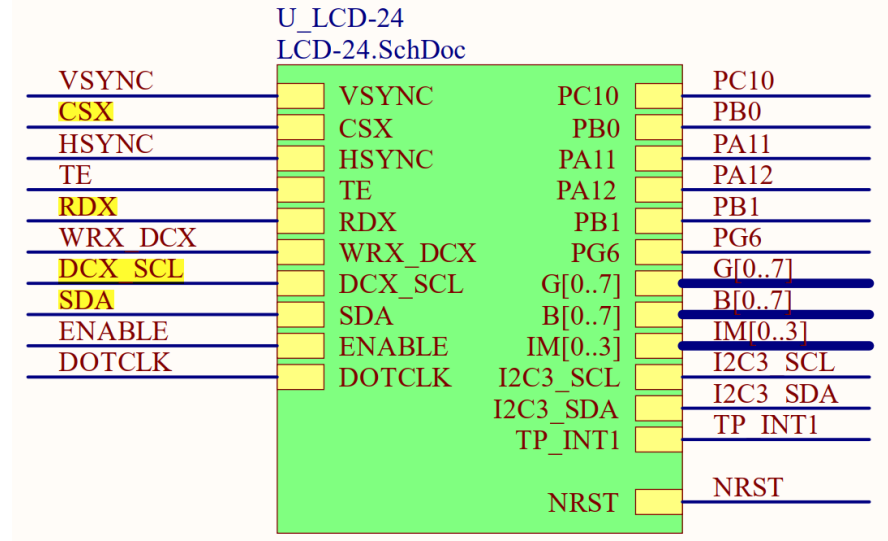
# Bài tập: Ghép nối STM32F429 với built-in touch LCD

- ❑ STM32F429I-DISC có một màn hình cảm ứng.
  - | Màn hình 320x240 điều khiển bởi chip ILI9341 LCD controller (SPI interface).
  - | Touch: điều khiển bởi STMPE811 resistive touch controller (I2C).
- ❑ Code mẫu: ghép nối điều khiển ILI9341 để hiển thị hình lá cờ Nhật Bản lên màn hình.



# Bài tập

- ❑ Tạo project mới, set CPU clock 180 MHz
- ❑ Cấu hình cổng SPI5 trên board STM32F429
- ❑ Kết nối ILI9341 → Board
  - | SS → PC2
  - | DC → PD13
  - | RST → PD12
  - | SCK → PF7
  - | MISO → PF8 (off)
  - | MOSI → PF9



PA4	VSYNC	VSYNC
PC2	CSX	CSX
PC6	HSYNC	HSYNC
PD11	TE	TE
PD12	RDX	RDX
PD13	WRX_DCX	WRX_DCX
PF7	DCX_SCL	DCX_SCL
PF9	SDA	SDA
PF10	ENABLE	ENABLE
PG7	DOTCLK	DOTCLK
PA8	I2C3_SCL	I2C3_SCL
PC9	I2C3_SDA	I2C3_SDA
PA15	TP_INT1	TP_INT1
PA7	ACP_RST	ACP_RST

# Bài tập

## Bài tập

---

- ❑ Thêm các file thư viện của ILI9341 vào project.
- ❑ Thêm mã nguồn trong hàm main().
- ❑ Chạy và kiểm tra kết quả.

```
93  /* Initialize all configured peripherals */
94  MX_GPIO_Init();
95  MX_DMA_Init();
96  MX_SPI5_Init();
97  /* USER CODE BEGIN 2 */
98  ILI9341_Init();
99  ILI9341_FillScreen(WHITE);
100 ILI9341_SetRotation(SCREEN_HORIZONTAL_2);
101 ILI9341_DrawText("JAPAN FLAG", FONT4, 90, 200, BLACK, WHITE);
102 ILI9341_DrawFilledCircle(160, 120, 50, RED);
103 HAL_Delay(1000);
104 /* USER CODE END 2 */
```

## Bài tập

---

❑ Hãy vẽ lá cờ Việt Nam!

## 4.2 I<sup>2</sup>C

---

### ❑ Nhược điểm SPI:

- | Cần nhiều chân tín hiệu.
- | Khó ghép nối với nhiều slave cùng lúc.

➔ Inter-Integrated Circuit (I2C): chuẩn và giao thức thiết kế bởi Philips Semiconductor, cho mục đích trao đổi thông tin giữa các IC trên cùng một PCB.

- | Dùng ít chân.
- | Tốc độ cao.

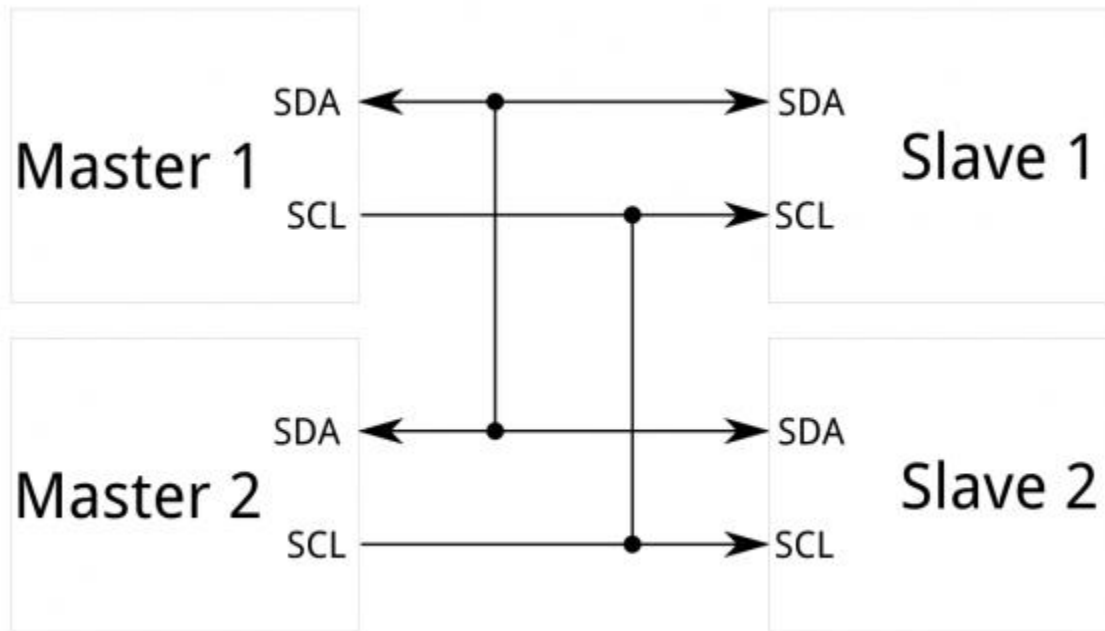
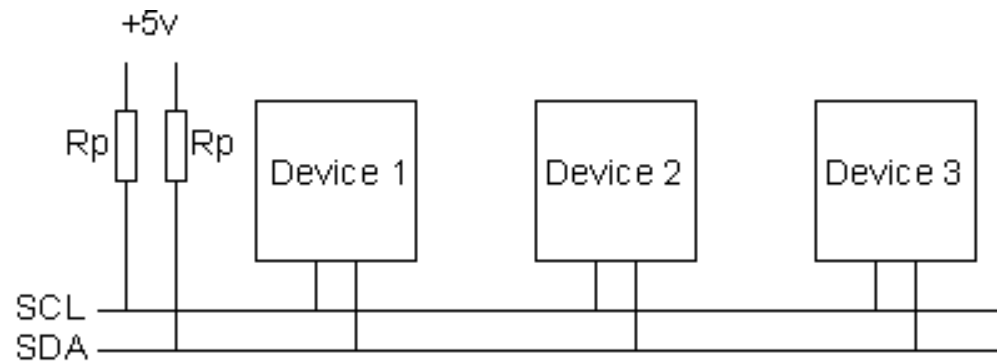
### ❑ Ví dụ ngoại vi dùng I2C: display, sensor,...



# I<sup>2</sup>C bus

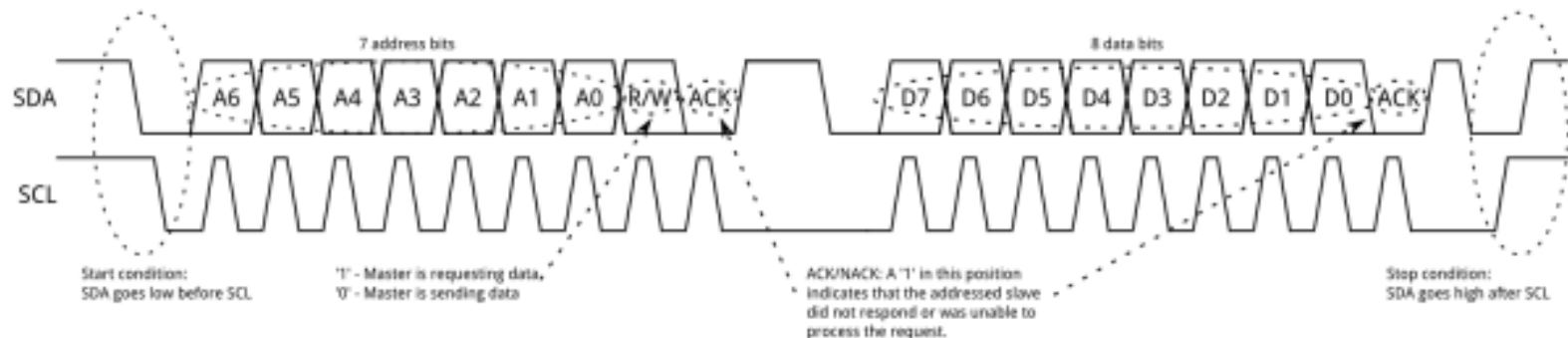
## ❑ Mô hình kết nối

- | SCL: serial clock
- | SDA: serial data



# Giao thức trên I<sup>2</sup>C bus

- ❑ Master khởi tạo phiên truyền
  - | Start
  - | Address frame: 7 hoặc 10 bit địa chỉ + R/W
  - | Data frame: 8 bit (có thể truyền nhiều frame)
  - | Stop
- ❑ Slave có địa chỉ trùng khớp sẽ nhận và thực thi lệnh (đọc/ghi dữ liệu).



# Lập trình với I2C

---

❑ Đọc tài liệu UM1725 tìm hiểu các hàm sau

<i>HAL_I2C_Init()</i>	<i>HAL_I2C_Master_Transmit()</i>	<i>HAL_I2C_IsDeviceReady()</i>
<i>HAL_I2C_DeInit()</i>	<i>HAL_I2C_Master_Receive()</i>	<i>HAL_I2C_Master_Seq_Transmit_IT()</i>
<i>HAL_I2C_MspInit()</i>	<i>HAL_I2C_Slave_Transmit()</i>	<i>HAL_I2C_Master_Seq_Transmit_DMA()</i>
<i>HAL_I2C_MspDeInit()</i>	<i>HAL_I2C_Slave_Receive()</i>	<i>HAL_I2C_Master_Seq_Receive_IT()</i>
	<i>HAL_I2C_Master_Transmit_IT()</i>	<i>HAL_I2C_Master_Seq_Receive_DMA()</i>
<i>HAL_I2C_RegisterCallback()</i>	<i>HAL_I2C_Master_Receive_IT()</i>	<i>HAL_I2C_Slave_Seq_Transmit_IT()</i>
<i>HAL_I2C_UnRegisterCallback()</i>	<i>HAL_I2C_Slave_Transmit_IT()</i>	<i>HAL_I2C_Slave_Seq_Transmit_DMA()</i>
<i>HAL_I2C_RegisterAddrCallback()</i>	<i>HAL_I2C_Slave_Receive_IT()</i>	<i>HAL_I2C_Slave_Seq_Receive_IT()</i>
<i>HAL_I2C_UnRegisterAddrCallback()</i>	<i>HAL_I2C_Master_Transmit_DMA()</i>	<i>HAL_I2C_Slave_Seq_Receive_DMA()</i>
	<i>HAL_I2C_Master_Receive_DMA()</i>	<i>HAL_I2C_EnableListen_IT()</i>
	<i>HAL_I2C_Slave_Transmit_DMA()</i>	<i>HAL_I2C_DisableListen_IT()</i>
	<i>HAL_I2C_Slave_Receive_DMA()</i>	<i>HAL_I2C_Master_Abort_IT()</i>
	<i>HAL_I2C_Mem_Write()</i>	
	<i>HAL_I2C_Mem_Read()</i>	
	<i>HAL_I2C_Mem_Write_IT()</i>	
	<i>HAL_I2C_Mem_Read_IT()</i>	
	<i>HAL_I2C_Mem_Write_DMA()</i>	
	<i>HAL_I2C_Mem_Read_DMA()</i>	

# Lập trình với I2C

---

## ❑ Ví dụ: hàm gửi dữ liệu

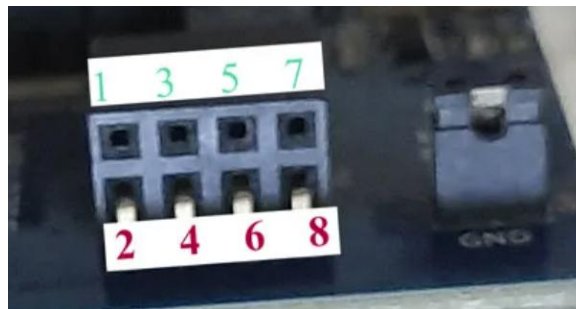
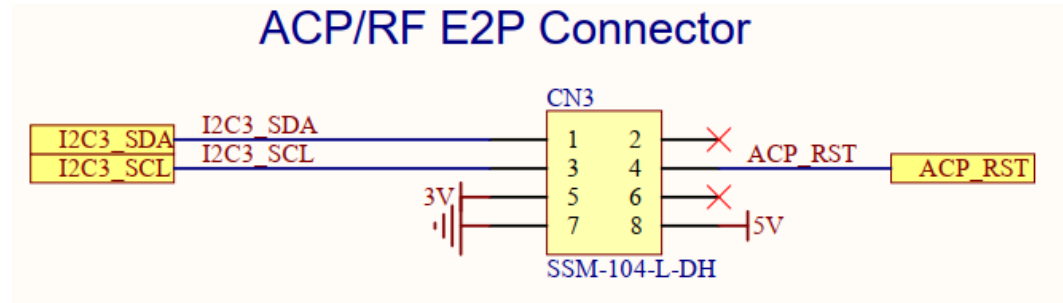
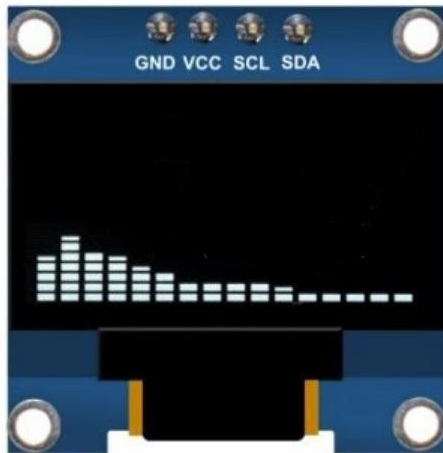
```
HAL_StatusTypeDef HAL_I2C_Master_Transmit (I2C_HandleTypeDef * hi2c,  
                                             uint16_t DevAddress,  
                                             uint8_t * pData,  
                                             uint16_t Size,  
                                             uint32_t Timeout)
```

## ❑ Chú ý:

- | DevAddress: số nguyên 8 bit chứa **địa chỉ slave ở 7 bit cao**, và bit thấp nhất chứa giá trị R/W mã hóa chiều truyền dữ liệu.
  - R/W = 0: master ghi dữ liệu ra slave
  - R/W = 1: master đọc dữ liệu từ slave

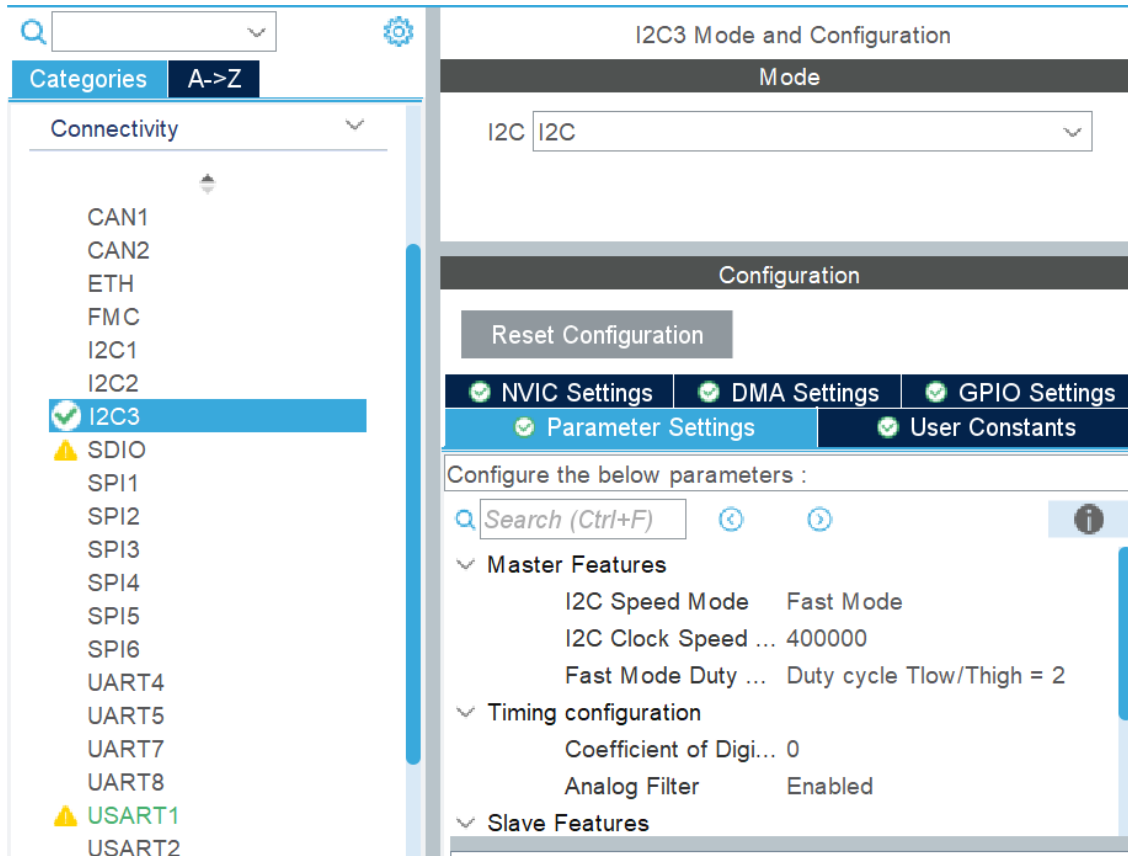
# Bài tập: Ghép nối STM32F429 với OLED SH1106

- ❑ SH1106: module màn hình OLED đơn sắc, độ phân giải 128x64.
- ❑ Giao tiếp theo chuẩn I2C ở địa chỉ 0x3C.
- ❑ Lắp mạch vào cổng CN2 theo đúng thứ tự



# Bài tập: Ghép nối STM32F429 với OLED SH1106

- ❑ Tạo project mới, set CPU clock 180 MHz
- ❑ Cấu hình cổng I2C3 trên board



## Bài tập: Ghép nối STM32F429 với OLED SH1106

- ❑ Thêm các file sh1106.\* và fonts.\* vào project.
- ❑ Thêm mã nguồn trong hàm main().
- ❑ Chạy và kiểm tra kết quả.

```
94  /* Initialize all configured peripherals */
95  MX_GPIO_Init();
96  MX_I2C3_Init();
97  MX_USART1_UART_Init();
98  /* USER CODE BEGIN 2 */
99  char buf[100];
100 int X = 0, Y = 0;
101 SH1106_Init ();
102 sprintf (buf, "%s", "Hello TN66");
103 SH1106_GotoXY (12,10); // goto 10, 10
104 SH1106_Puts(buf, &Font_11x18, 1);
105 SH1106_UpdateScreen(); // update screen
106 HAL_Delay(1000);
107 /* USER CODE END 2 */
```

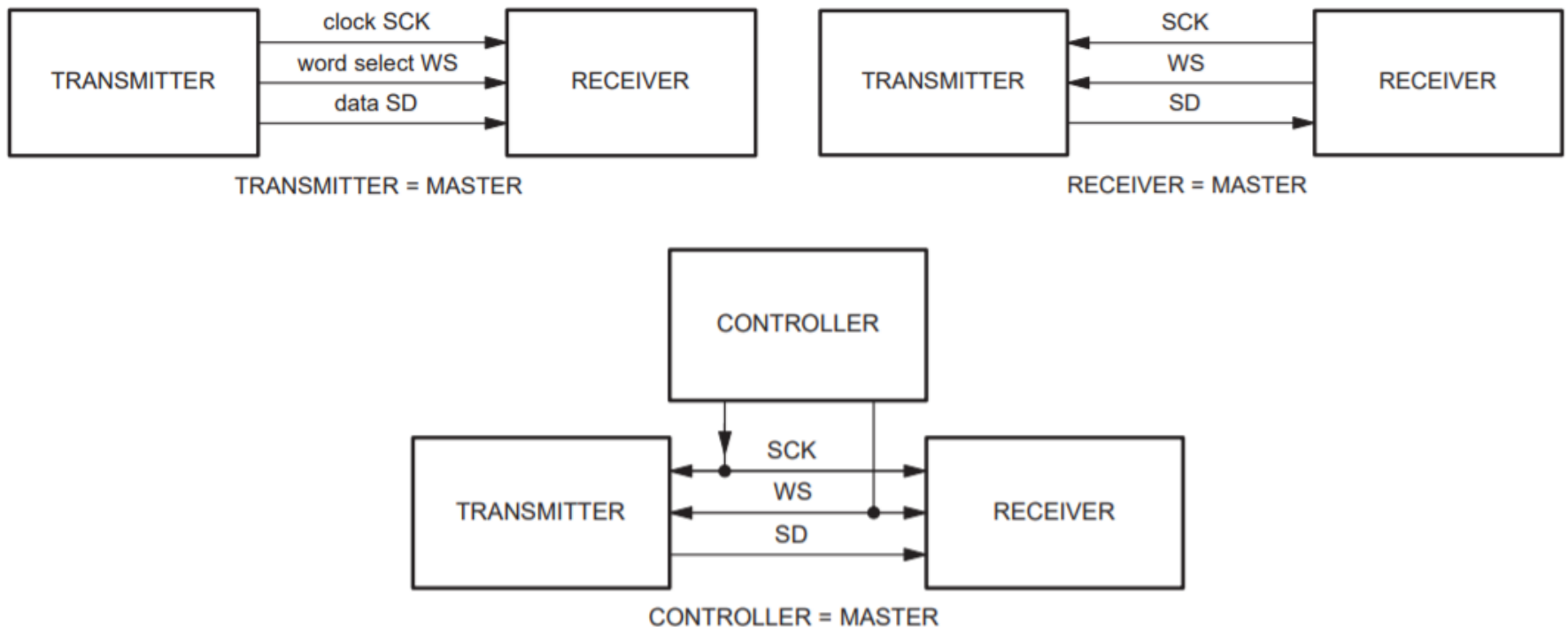
## **Bài tập: Ghép nối STM32F429 với OLED SH1106**

- ❑ Tự làm: đọc và hiển thị mã thẻ RFID lên màn hình SH1106.
- ❑ Tự làm: ghép nối thêm timer, hiển thị giờ, phút, giây lên màn hình SH1106.



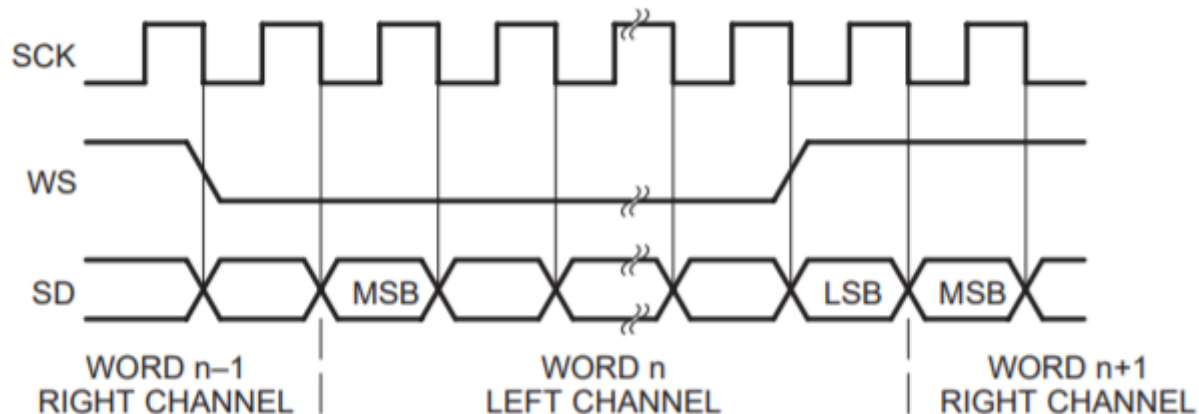
## 4.3. I<sup>2</sup>S

- ❑ Inter-IC Sound
- ❑ Phát triển bởi Philips Semiconductor (NXP) năm 1986
- ❑ Truyền dữ liệu âm thanh PCM 2 kênh



# I<sup>2</sup>S protocol

- ❑ Giao thức rất đơn giản
- ❑ Truyền MSB trước → transmitter và receiver có thể hỗ trợ số bit khác nhau.

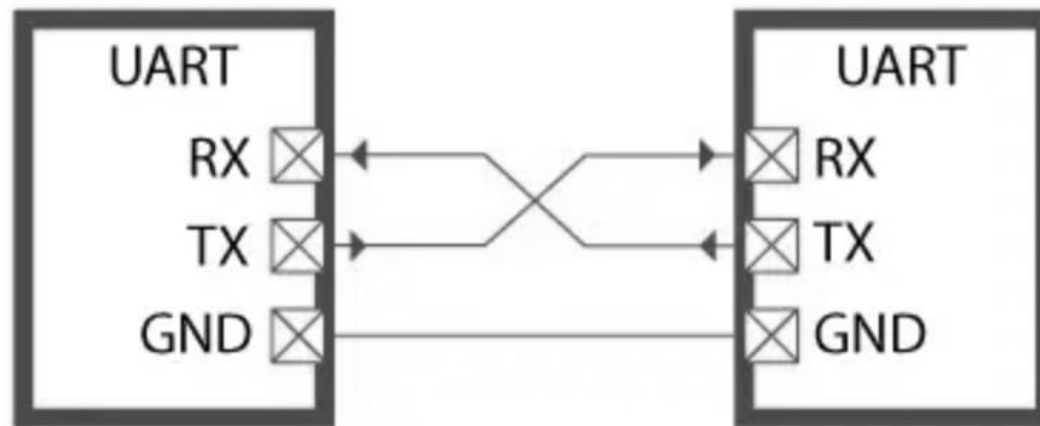


- WS = 0; channel 1 (left);
- WS = 1; channel 2 (right)

## 4.4 UART

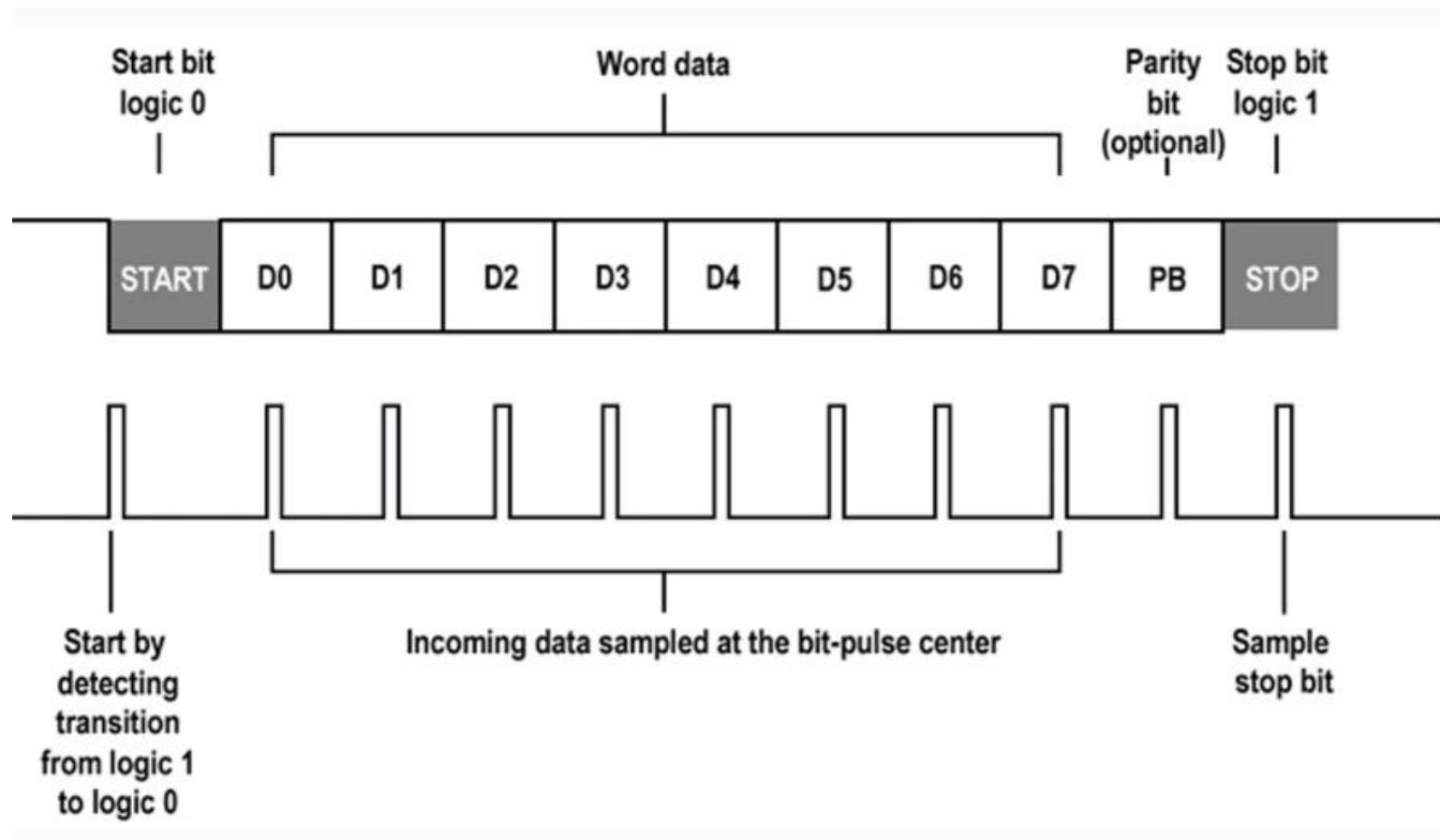
---

- ❑ Là chuẩn truyền nối tiếp không đồng bộ (Universal Asynchronous Receiver and Transmitter).
- ❑ Thường được gọi nhanh là serial port trên máy tính/hệ nhúng.
- ❑ Đóng khung dữ liệu và quy ước tốc độ truyền (thay cho tín hiệu clock).



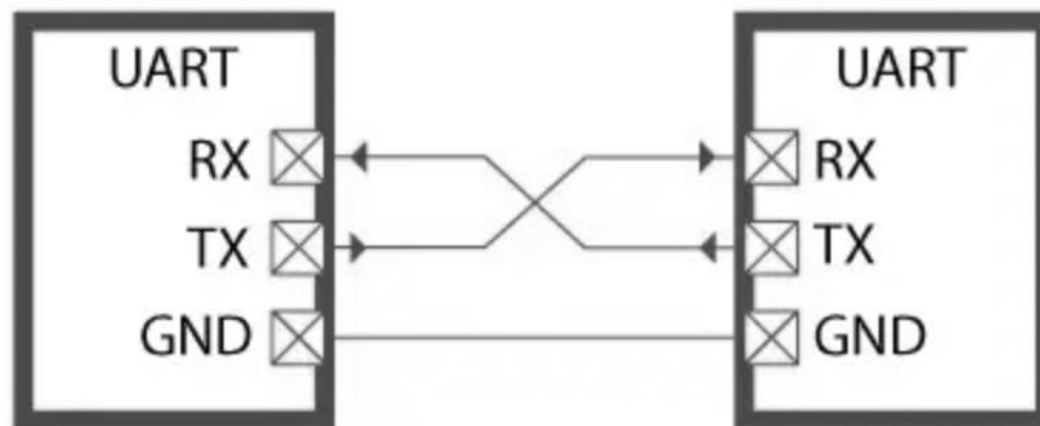
# UART data frame

- ❑ 1 start bit, 5-8 data bits, [parity], 1-2 stop bits



# Chuẩn UART và ghép nối UART trên STM32F429

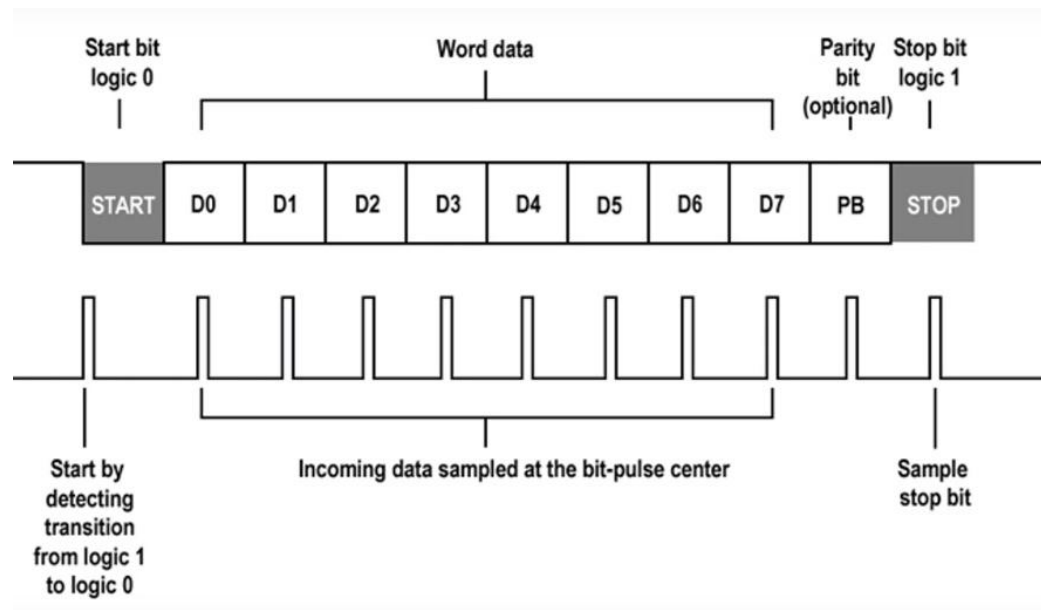
- ❑ UART: chuẩn truyền nối tiếp không đồng bộ (Universal Asynchronous Receiver and Transmitter).
- ❑ Thường được gọi nhanh là serial port trên máy tính/hệ nhúng.
- ❑ Có thể cấu hình half duplex hoặc full duplex
- ❑ Sơ đồ kết nối.



# Chuẩn UART và ghép nối UART trên STM32F429

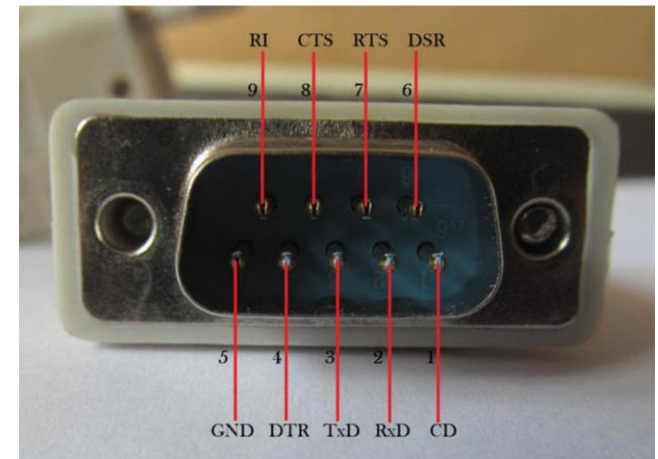
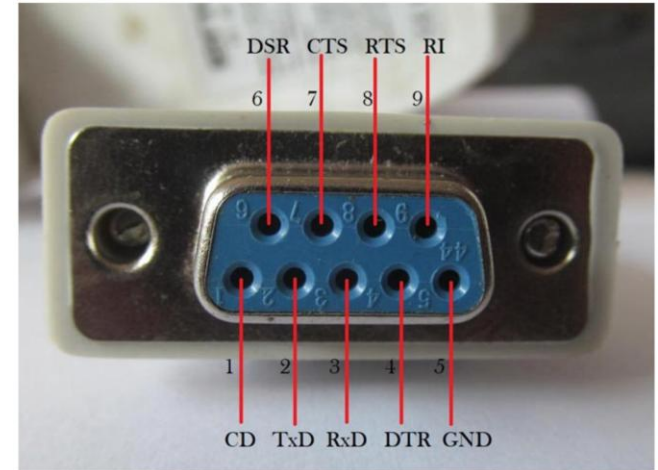
- ❑ Chỉ có 1 bit dây nối mỗi chiều, làm thế nào để truyền các byte dữ liệu?
  - | Frame dữ liệu được truyền/nhận thành luồng bit với chu kỳ bit biết trước: các mức tốc độ của UART được chuẩn hóa.
  - | Có cơ chế đồng bộ (synchronization) để đánh dấu đầu/cuối frame dữ liệu.

## ❑ UART data frame



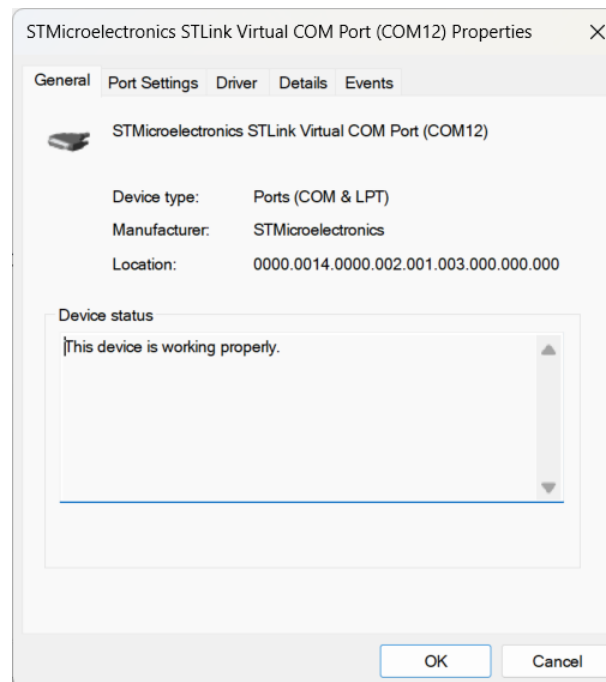
# Chuẩn RS232C

- ❑ Quy chuẩn tín hiệu truyền nối tiếp đồng bộ giữa các thiết bị, sử dụng UART
- ❑ Mức điện áp:
  - | 0: +3 đến +15V
  - | 1: -3 đến -15V
- ❑ Tốc độ truyền (baudrate): 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
- ❑ Ngoài Rx, Tx còn có thêm 6 tín hiệu điều khiển khác.



# UART trên STM32F429

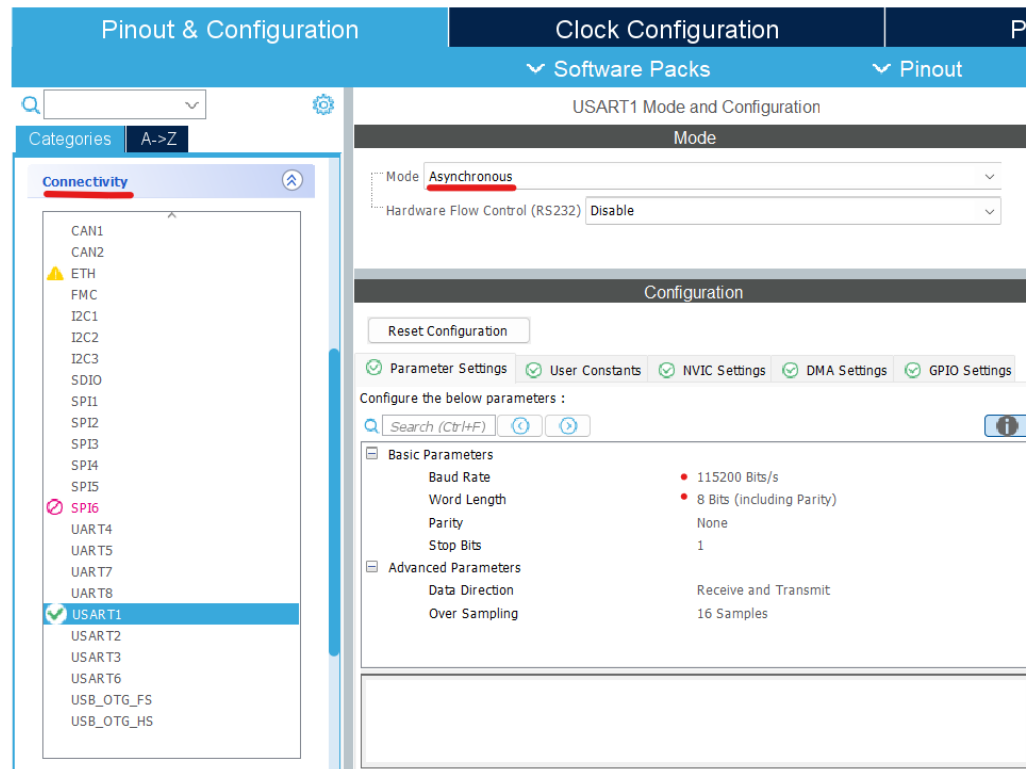
- ❑ STM32F429 hỗ trợ tối đa 8 cổng UART/USART, cấu hình trong tab Connectivity.
- ❑ Điện áp giao tiếp: 3.3V
- ❑ Trên board STM32F429-DISC: UART1 được nối sẵn với mạch debug, ánh xạ vào cổng COM ảo trên PC.





# Bài tập: lập trình ghép nối UART

- ❑ Lập trình gửi đoạn text “Hello” từ STM32F429 lên PC
  - | Cấu hình USART1 trong Connectivity
  - | Mode: Asynchronous, Baud rate 115200 bps, Word length 8 bits, Parity none.



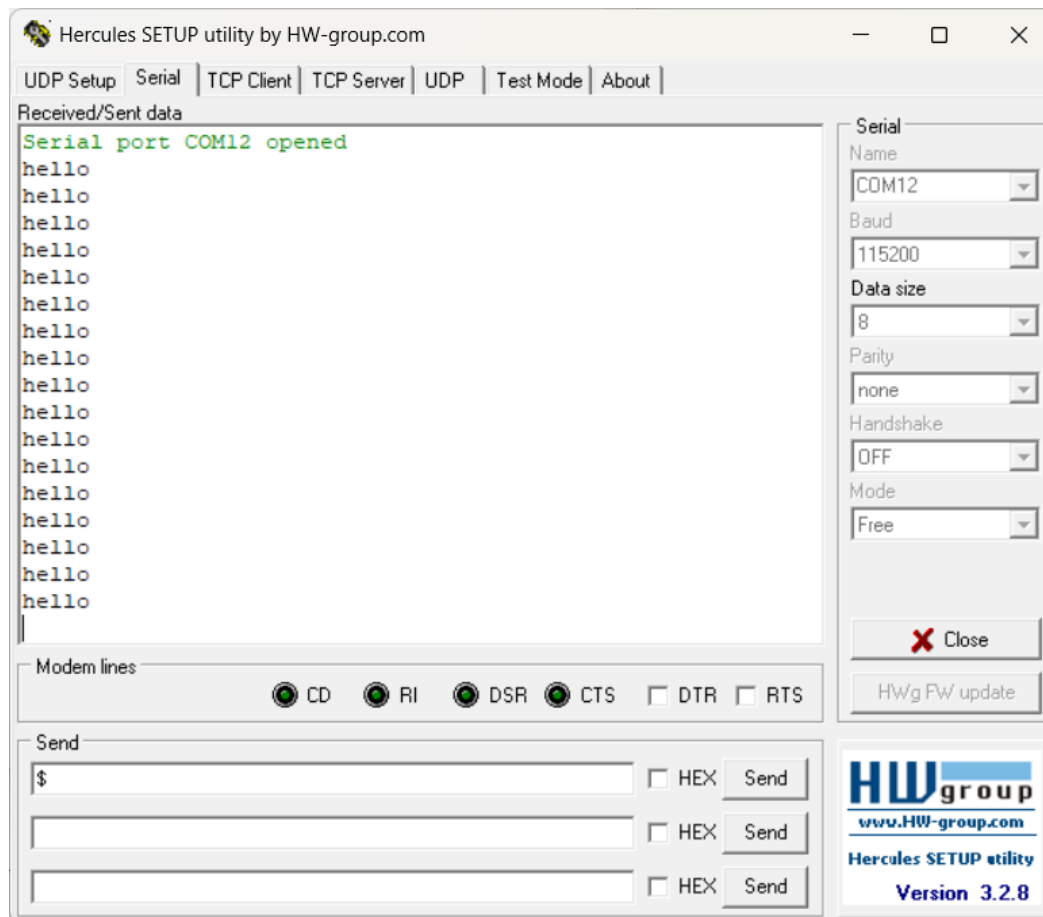
## Bài tập: lập trình ghép nối UART

- ❑ Lập trình gửi đoạn text “Hello” từ STM32F429 lên PC
  - | Cấu hình USART1 trong Connectivity
  - | Mode: Asynchronous, Baud rate 115200 bps, Word length 8 bits, Parity none.
- ❑ Gọi hàm gửi dữ liệu trong main loop

```
main.c x stm32f4xx_it.c startup_stm32f429zitx.s InterruptEx.ioc system_stm32f4xx.c
100  /* Infinite loop */
101  /* USER CODE BEGIN WHILE */
102  while (1)
103  {
104      /* USER CODE END WHILE */
105      char buf[10] = "hello\r\n";
106      HAL_UART_Transmit(&huart1,
107                      (const uint8_t*)buf,
108                      strlen(buf) ,
109                      2);
110      HAL_Delay(500);
111      /* USER CODE BEGIN 3 */
112  }
113  /* USER CODE END 3 */
```

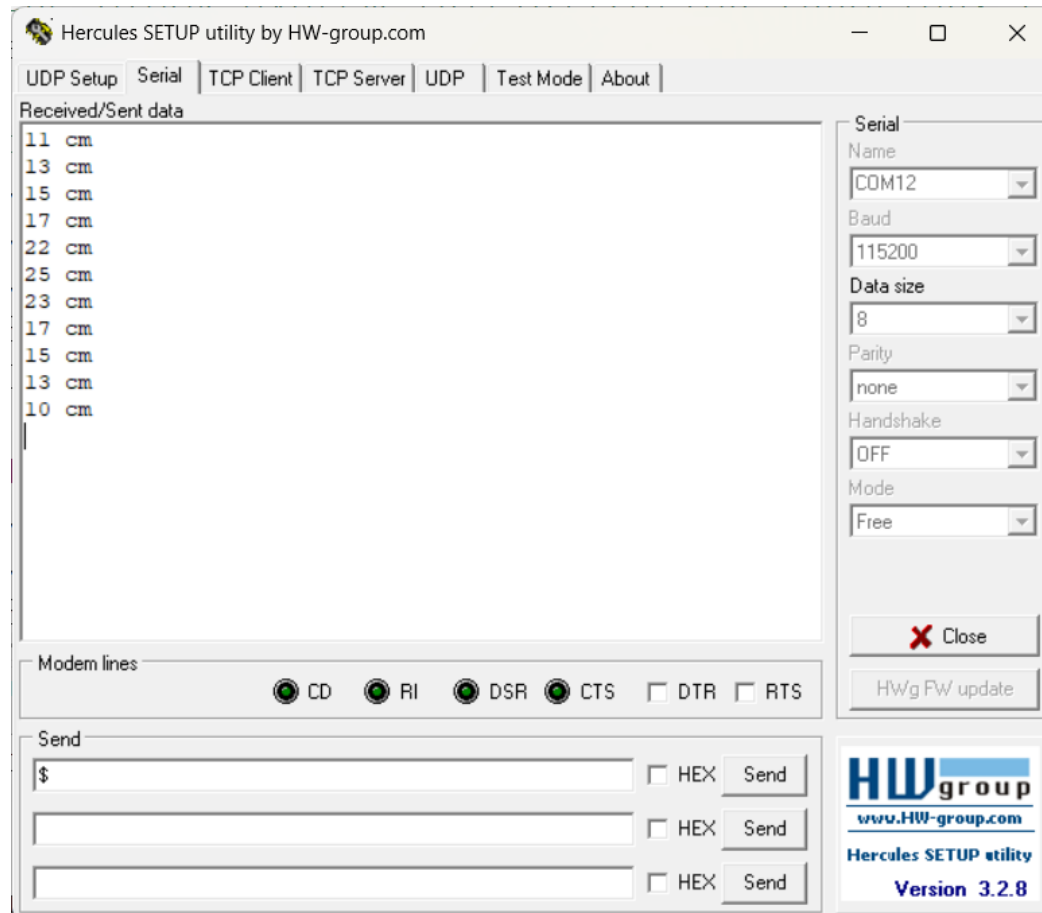
# Bài tập: lập trình ghép nối UART

- ❑ Chạy và quan sát kết quả nhận được trên Hercules
  - | Chú ý cấu hình Hercules tương thích cấu hình UART trên STM32F4



# Bài tập

- ❑ Kết hợp các bài tập trên, gửi liên tục dữ liệu khoảng cách nhận được từ cảm biến siêu âm về máy tính qua cổng UART.



## Bài tập

### ❑ Nối 2 board STM32F429I bằng cổng USART1

- | Board 1 – TX (PA9)                       $\longleftrightarrow$       Board 2 – RX (PA10)
- | Board 1 – RX (PA10)                    $\longleftrightarrow$       Board 2 – TX (PA9)
- | Board 1 – GND                            $\longleftrightarrow$       Board 2 – GND

### ❑ Viết chương trình cho 1 board làm transmitter, 1 board làm receiver

- | Truyền một mảng 100 số nguyên từ board 1 sang board 2.
- | Khi board 1 truyền đủ 100 số thì bật LED3 trên board 1.
- | Khi board 2 nhận đủ 100 số thì bật LED3 trên board 2.

PA9	I/O	FT	-	TIM1_CH2, I2C3_SMBA, USART1_TX, DCMI_D0, EVENTOUT
PA10	I/O	FT	-	TIM1_CH3, USART1_RX, OTG_FS_ID, DCMI_D1, EVENTOUT

# Bài tập

The screenshot displays the STM32CubeMX Pinout & Configuration window. The interface is divided into several sections:

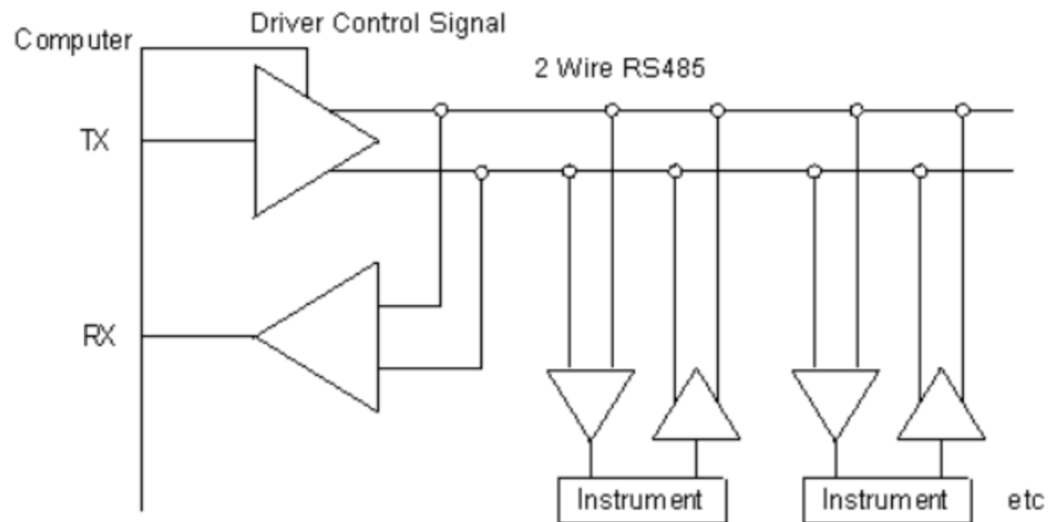
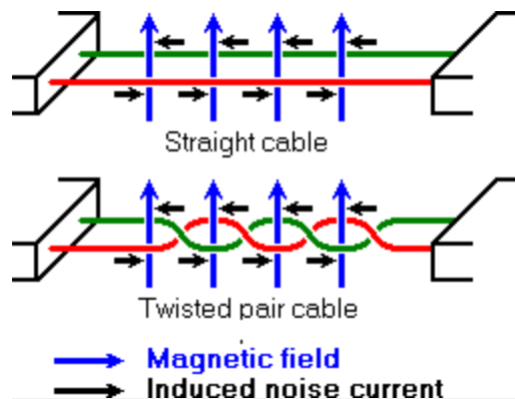
- Pinout & Configuration** (Active tab):
  - Categories**: A list of peripherals including SDIO, SPI1, SPI2, SPI3, SPI4, SPI5, SPI6, UART4, UART5, UART7, UART8, **USART1** (selected), USART2, USART3, USART6, USB\_OTG\_FS, and USB\_OTG\_HS.
  - USART1 Mode and Configuration**:
    - Mode**: Mode is set to **Asynchronous**. Hardware Flow Control (RS232) is set to **Disable**.
    - Configuration**: Includes a **Reset Configuration** button and tabs for **GPIO Settings**, **NVIC Settings**, **DMA Settings**, **Parameter Settings** (active), and **User Constants**.
    - Parameter Settings**:
      - Basic Parameters**: Baud Rate is 115200 Bits/s, Word Length is 8 Bits (including Parity), and Parity is None.
- Clock Configuration**
- Project Manager**
- Tools**

The **Pinout view** on the right shows the pin configuration for the selected USART1 peripheral. The pins are listed in a column on the right, with their corresponding functions indicated:

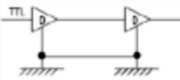
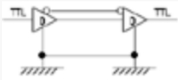
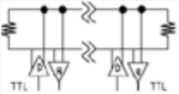
- VDD
- VSS
- VCAP..
- PA13
- PA12
- PA11
- PA10** (highlighted in green) is assigned to **USART1\_RX**.
- PA9** (highlighted in green) is assigned to **USART1\_TX**.
- PA8
- PC9
- PC8
- PC7
- PC6
- VDD

# RS485

- ❑ Mở rộng từ RS232C
- ❑ Sử dụng cáp xoắn truyền tín hiệu vi sai
- ❑ Mở rộng khoảng cách hoạt động: lên tới 1200m
- ❑ Tăng tốc độ truyền tối đa: lên tới 10 Mbps



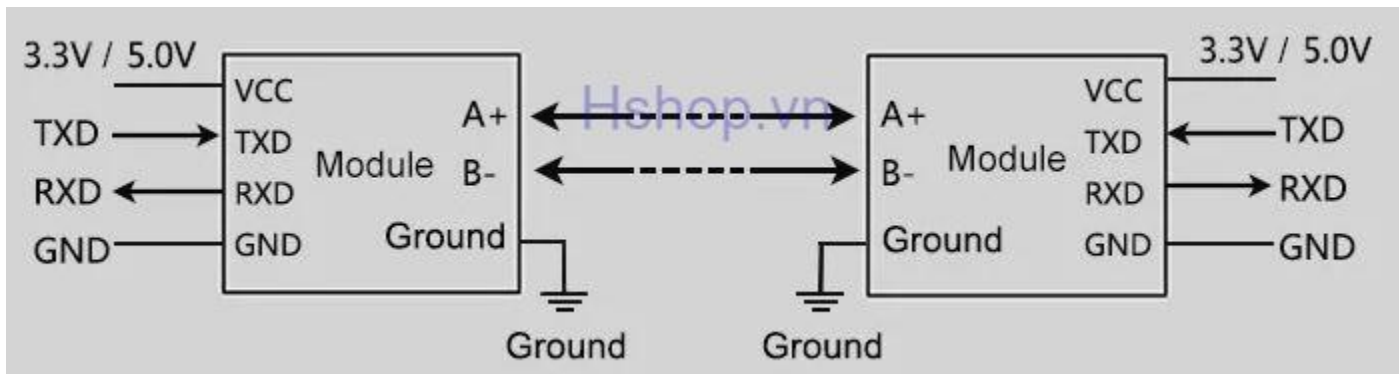
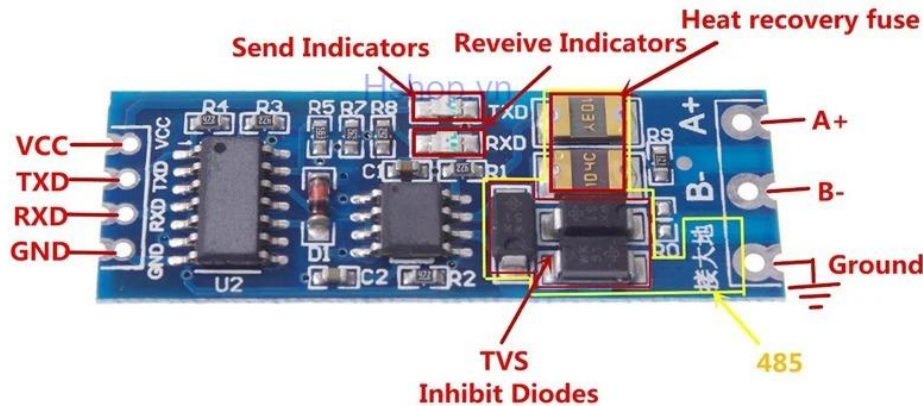
# So sánh RS-232C

Parameter	RS-232C	RS-422A	RS-485
Transmission mode	Simplex	Multi-point simplex	Multi-point multiplex
Max. connected devices	1 driver 1 receiver	1 driver 10 receivers	32 drivers 32 receivers
Max. transmission rate	20Kbps	10Mbps	10Mbps
Max. cable length	15m	1200m	1200m
Operation mode	Single-ended (unbalanced type)	Differential (balanced type)	Differential (balanced type)
Connection image			
Features	Short distance Full-duplex 1:1 connection	Long distance Full-duplex, half-duplex 1:N connection	Long distance Full-duplex, half-duplex N:N connection



# VD: Xây dựng mạng RS485

## Module UART - RS485








Mô hình kết nối 2 node qua RS485

➔ cần xây dựng giao thức truyền/nhận dữ liệu

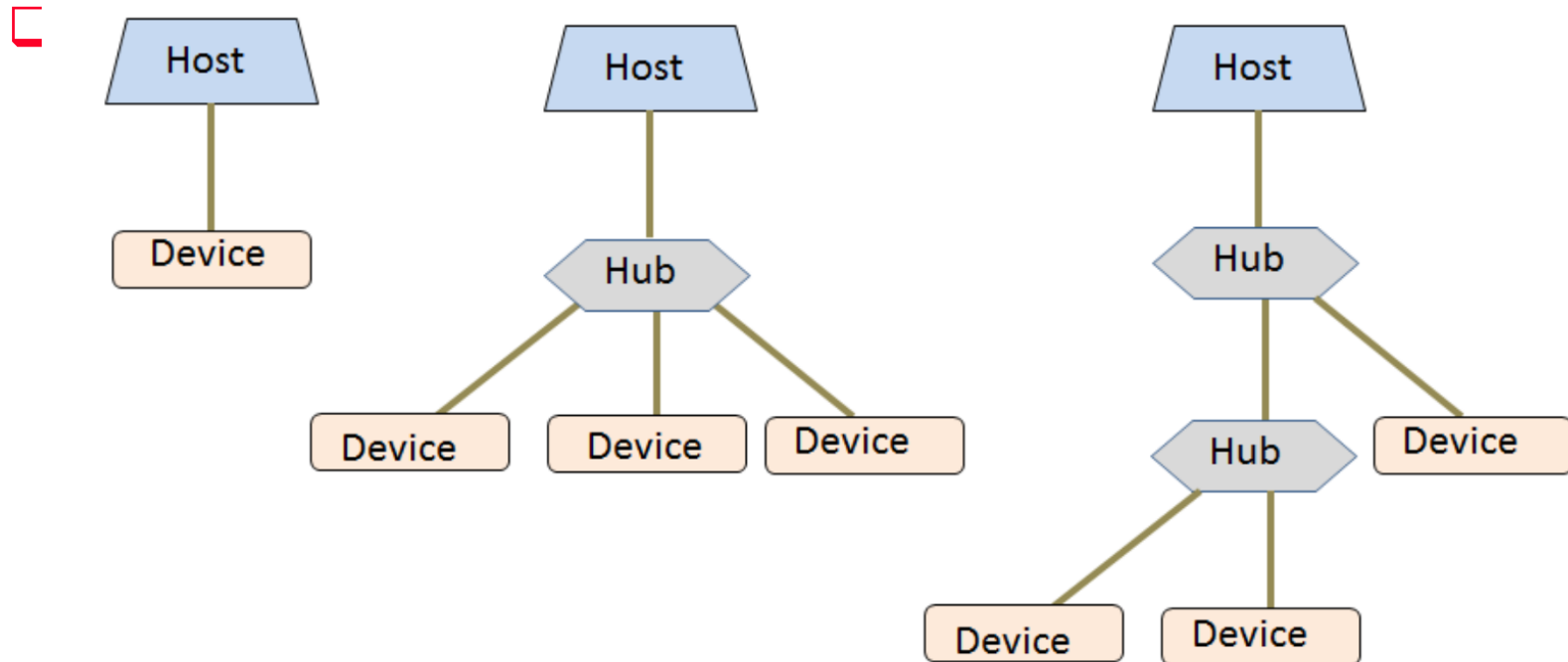
## 4.5. USB

- ❑ Universal Serial Bus, từ 1994 (Compaq, DEC, IBM, Intel, Microsoft, NEC and Nortel)
- ❑ Chuẩn ghép nối phổ biến nhất hiện nay (cho PC)
- ❑ Truyền thông tin nối tiếp đồng bộ tốc độ cao

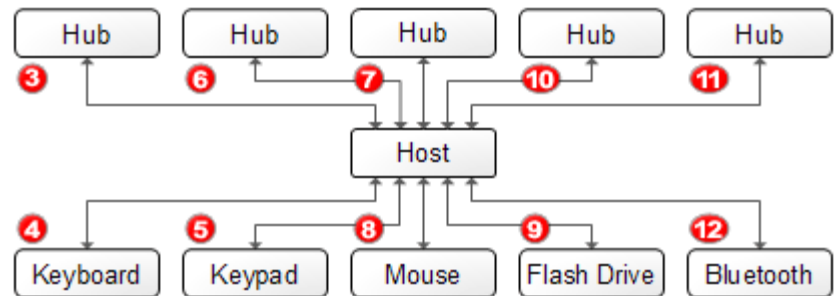
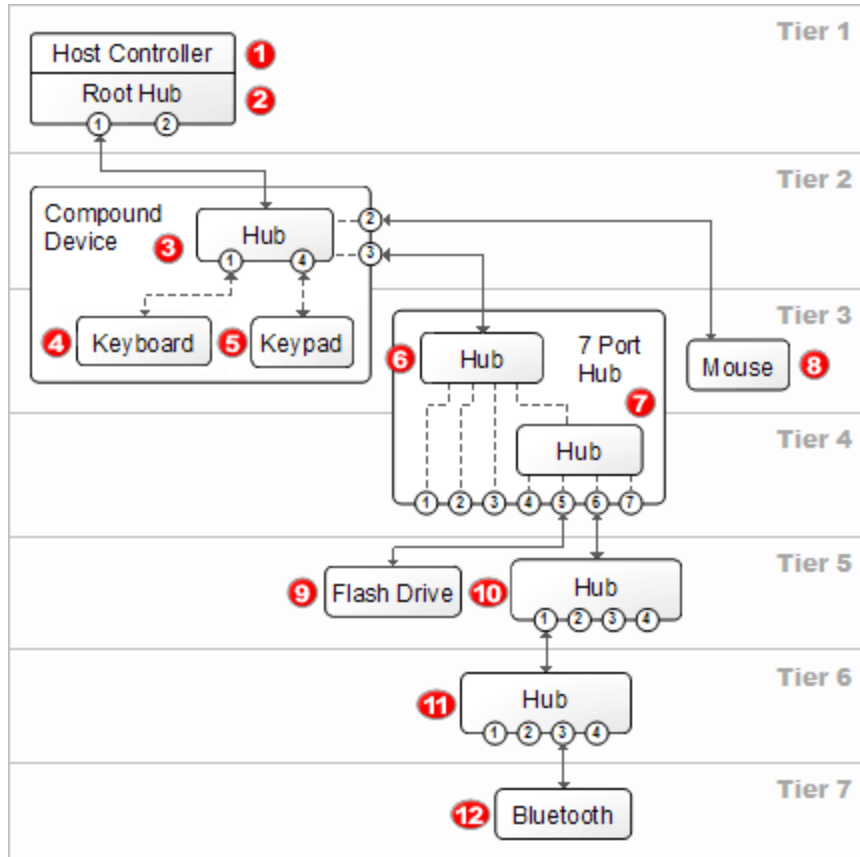
Connectors		USB 1.0 1996	USB 2.0 2001	USB 2.0 Revised	USB 3.0 2011	USB 3.1 2014	USB 3.2 2017	USB4 2019
Data rate		1.5 Mbit/s (Low Speed)	1.5 Mbit/s (Low Speed)		5 Gbit/s (SuperSpeed)	10 Gbit/s (SuperSpeed+)	20 Gbit/s (SuperSpeed+)	40 Gbit/s (SuperSpeed+ and Thunderbolt 3)
		12 Mbit/s (Full Speed)	12 Mbit/s (Full Speed)  480 Mbit/s (High Speed)					
Standard	A	<div>Type A</div> 				<div>Type A</div>  <div>Type-A SuperSpeed</div>		Deprecated
	B	<div>Type B</div> 				<div>Type B</div>  <div>Type B SuperSpeed</div>		Deprecated
	C	N/A					<div>Type C (enlarged)</div> 	

## Các thành phần

- ❑ USB host: là host/master trên bus USB, điều khiển mọi hoạt động trên bus.
- ❑ USB hub: mở rộng cho phép 1 host nối nhiều thiết bị



# Usb bus topology



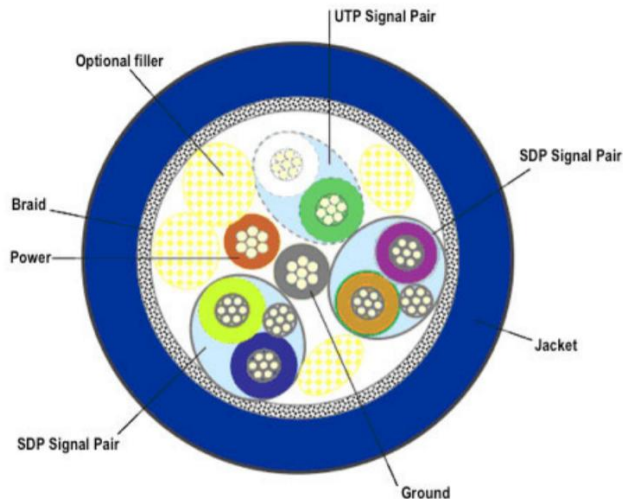
# Tín hiệu USB

USB 1.0/2.0

Pin	Name	Cable color	Description
1	VCC	Red	+5 VDC
2	D-	White	Data -
3	D+	Green	Data +
4	GND	Black	Ground

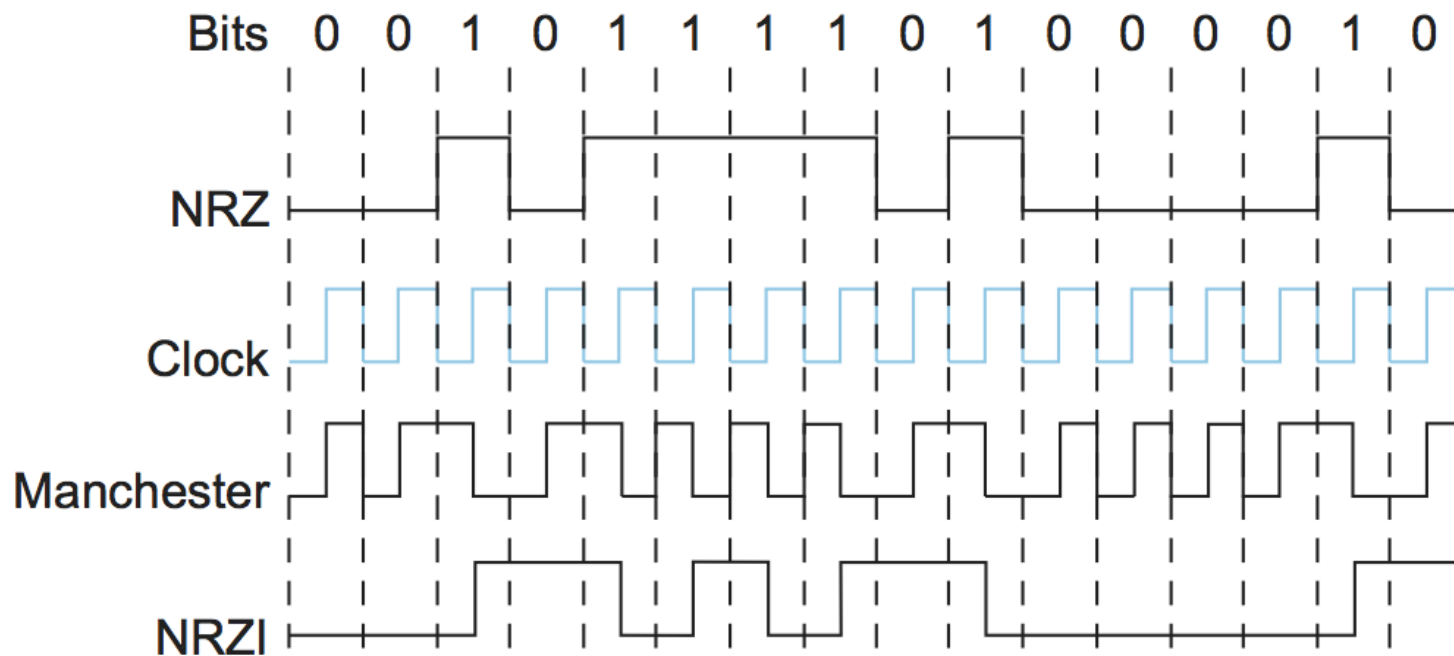
USB 3.x

Pin	Pin Name	Description
1	VBus	+5V Power
2	USB D-	USB 2.0 data
3	USB D+	
4	GND	Ground for power return
5	StdA_SSRX-	SuperSpeed receiver
6	StdA_SSRX+	SuperSpeed receiver
7	GND_DRAIN	Ground for signal return
8	StdA_SSTX-	SuperSpeed transmitter
9	StdA_SSTX+	SuperSpeed transmitter



# USB data communication

- ❑ Mã hóa NRZI (Non-return to zero inverted)



## Hoạt động trên bus

---

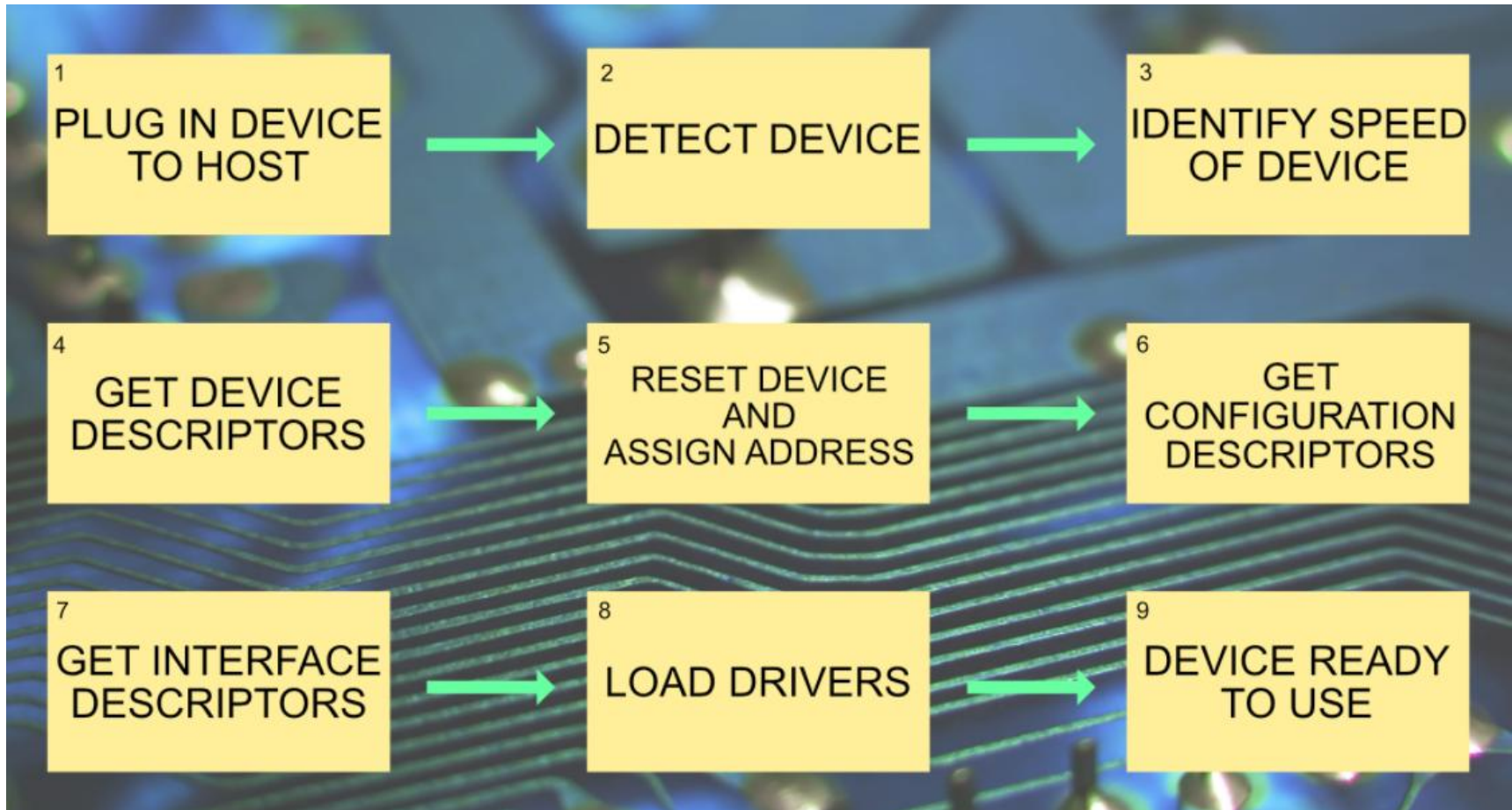
### ❑ Khởi tạo thiết bị (USB Enumeration)

- | Là toàn bộ quá trình từ khi USB host phát hiện có 1 device được nối vào bus, xác định loại thiết bị, đến khi khởi tạo xong cấu hình hoạt động và nạp driver.
- | Thực hiện 1 lần khi thiết bị nối vào bus.
- | Mỗi thiết bị có 1 địa chỉ (từ 1-127).
- | Địa chỉ mặc định là 0.

### ❑ Trao đổi dữ liệu:

- | USB host liên tục query trạng thái tất cả các thiết bị trên bus.
- | Thiết bị nào cần trao đổi dữ liệu với host thì trả lời.

# Khởi tạo thiết bị

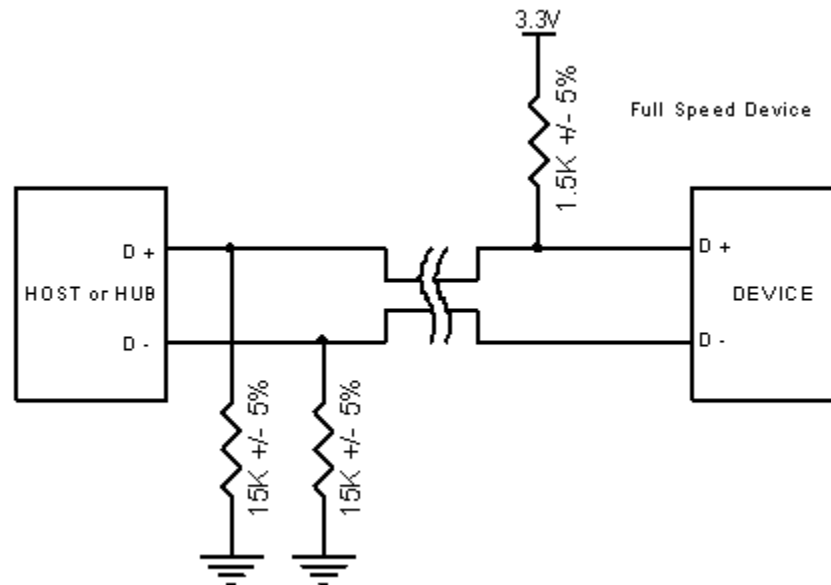


<https://www.totalphase.com>



## USB enumeration

- Khi thiết bị được cắm vào host/hub: host nhận được sự kiện có thiết bị mới.
- Host gửi tín hiệu reset thiết bị mới → thiết bị mới được đặt ở địa chỉ mặc định = 00H. Thiết bị được phép lấy dòng max = 150 mA từ bus.



## USB enumeration

- Host gửi “Get\_Device\_Descriptor” request đến địa chỉ 00H. → Device trả lại 8 byte đầu của Device Descriptor (để lấy max packet size cho Endpoint 0).
- Host gửi “Set\_Address” request để gán địa chỉ (duy nhất) cho thiết bị.
- Thiết bị nhận địa chỉ và lưu lại.
- Host gửi “Get\_Device\_Descriptor” lần nữa để lấy toàn bộ Device Descriptor đầy đủ.
- Host decode Device Descriptor để lấy thông tin về thiết bị: loại thiết bị, Product ID/Vendor ID, tên nhà sản xuất...

# USB enumeration

---

Offset (decimal)	Field	Size (bytes)	Description
0	bLength	1	Descriptor size in bytes
1	bDescriptorType	1	The constant DEVICE (01h)
2	bcdUSB	2	USB specification release number (BCD)
4	bDeviceClass	1	Class code
5	bDeviceSubclass	1	Subclass code
6	bDeviceProtocol	1	Protocol Code
7	bMaxPacketSize0	1	Maximum packet size for Endpoint 0
8	idVendor	2	Vendor ID
10	idProduct	2	Product ID
12	bcdDevice	2	Device release number (BCD)
14	iManufacturer	1	Index of string descriptor for the manufacturer
15	iProduct	1	Index of string descriptor for the product
16	iSerialNumber	1	Index of string descriptor containing the serial number
17	bNumConfigurations	1	Number of possible configurations

## USB enumeration

---

- Host gửi “Get\_Configuration\_Descriptor” và “Get\_Interface\_Descriptor” để lấy thêm các thông tin về thiết bị.

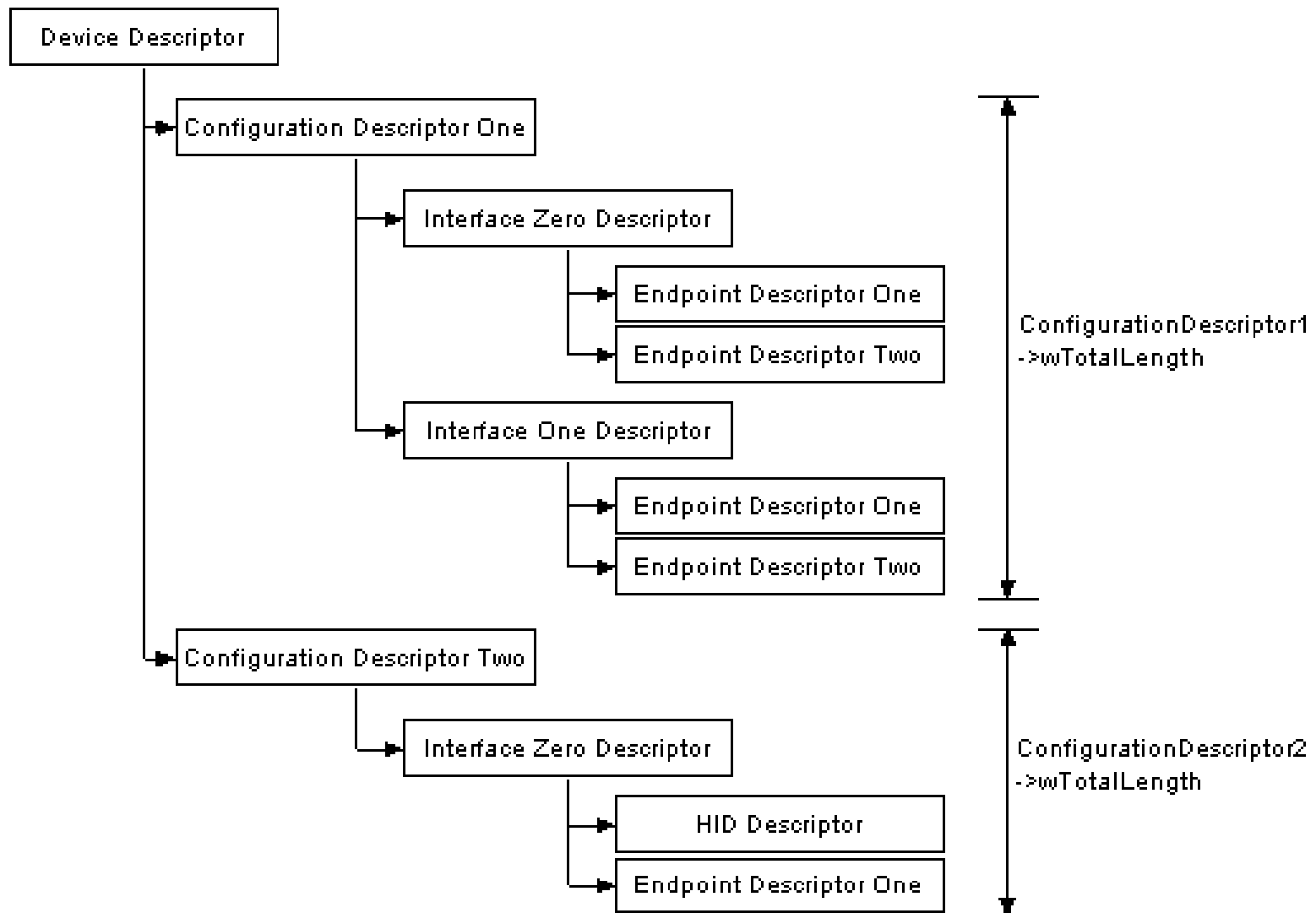
Offset (decimal)	Field	Size (bytes)	Description
0	bLength	1	Descriptor size in bytes
1	bDescriptorType	1	The constant Configuration (02h)
2	wTotalLength	2	The number of bytes in the configuration descriptor and all of its subordinate descriptors
4	bNumInterfaces	1	Number of interfaces in the configuration
5	bConfigurationValue	1	Identifier for Set_Configuration and Get_Configuration requests
6	iConfiguration	1	Index of string descriptor for the configuration
7	bmAttributes	1	Self/bus power and remote wakeup settings
8	bMaxPower	1	Bus power required, expressed as (maximum milliamperes/2)

---

---

Offset (decimal)	Field	Size (bytes)	Description
0	bLength	1	Descriptor size in bytes
1	bDescriptorType	1	The constant Interface (04h)
2	bInterfaceNumber	1	Number identifying this interface
3	bAlternateSetting	1	Value used to select an alternate setting
4	bNumEndpoints	1	Number of endpoints supported, not counting Endpoint 0
5	bInterfaceClass	1	Class code
6	bInterfaceSubclass	1	Subclass code
7	bInterfaceProtocol	1	Protocol code
8	iInterface	1	Index of string descriptor for the interface

# USB descriptor



# USB Enumeration

---

- ❑ Cuối cùng host nạp device driver tương ứng với thiết bị.
- ❑ Hệ điều hành xác định driver dựa vào cặp giá trị Product ID/Vendor ID để load file INF tương ứng.
- ❑ Host gửi “Set\_Configuration” request để thiết lập cấu hình hoạt động cho thiết bị.
- ❑ Thiết bị chuyển sang trạng thái Configured và bắt đầu trao đổi dữ liệu (theo điều khiển của Host).

## Trao đổi dữ liệu trên bus USB

- ❑ Host liên tục query trạng thái của các thiết bị có trên bus.
- ❑ Request từ host được gửi kèm địa chỉ của thiết bị tương ứng. Thiết bị nào có địa chỉ trùng với địa chỉ này sẽ được trả lời host.
- ❑ Thiết bị không có dữ liệu → trả lời không có
- ❑ Thiết bị có dữ liệu cần trao đổi dữ liệu → trả lời ACK để host biết và khởi tạo phiên truyền dữ liệu.



## Bài tập

---

- ❑ Sử dụng phần mềm HHD USB Monitor quan sát quá trình trao đổi dữ liệu với thiết bị USB.

## Bài tập

---

- ❑ Lập trình cấu hình cổng USB-OTG trên STM32F429 thành thiết bị HID với profile của USB mouse, và mỗi lần nút PA0 được bấm thì có sự kiện mouse click được gửi tới PC.

- ❑ Tạo project mới, cấu hình RCC và HCLK = 168 MHz (để có clock 48 MHz cho USB).



# Bài tập: USB HID

## ❑ Cấu hình hardware trong ioc

- | Chân PA0: GPIO External interrupt Rising/Falling edges, bật cho phép ngắt của GPIO\_EXTI0 trong NVIC.
- | USB\_OTG\_HS: Internal FS Phy **Device\_Only**

The screenshot displays the STM32CubeMX Pinout & Configuration window. The 'Pinout & Configuration' tab is active, showing the 'USB\_OTG\_HS Mode and Configuration' settings. The 'Mode' section has 'External Phy' set to 'Disable' and 'Internal FS Phy' set to 'Device\_Only' (highlighted with a red underline). The 'Configuration' section shows 'Activate\_SOF' as unchecked and 'Activate\_VBUS' as checked. The 'Configuration' section also includes a 'Reset Configuration' button and a list of settings: 'NVIC Settings', 'GPIO Settings', 'Parameter Settings', and 'User Constants'. The 'Configure the below parameters :' section shows 'Speed' set to 'Device Full Speed 12M...' and 'Enable internal I' set to 'Disabled'. The 'Pinout view' on the right shows the pin configuration for the USB\_OTG\_HS module, with pins PB15, PB14, and PB13 highlighted in green and labeled as USB\_OTG\_HS\_DP, USB\_OTG\_HS\_DM, and USB\_OTG\_HS\_VBUS respectively. The 'System' view on the right shows the pin configuration for the system, with pins PE12, PE13, PE14, PE15, PB10, PB11, VCAP, and VDD highlighted in yellow.

Pinout & Configuration | Clock Configuration | Project Manager

Software Packs | Pinout

Search

Categories | A-Z

- SPI5
- SPI6
- UART4
- UART5
- UART7
- UART8
- ✓ USART1
- USART2
- USART3
- USART6
- USB\_OTG\_FS
- ✓ USB\_OTG\_HS

Multimedia >

Security >

Computing >

USB\_OTG\_HS Mode and Configuration

Mode

External Phy: Disable

Internal FS Phy: Device\_Only

☐ Activate\_SOF

☒ Activate\_VBUS

Configuration

Reset Configuration

✓ NVIC Settings | ✓ GPIO Settings

✓ Parameter Settings | ✓ User Constants

Configure the below parameters :

Search (Ctrl+F)

Speed: Device Full Speed 12M...

Enable internal I: Disabled

Pinout view | System

PD13

PD12

PD11

PD10

PD9

PD8

PB15: USB\_OTG\_HS\_DP

PB14: USB\_OTG\_HS\_DM

PB13: USB\_OTG\_HS\_VBUS

PB12

PE12

PE13

PE14

PE15

PB10

PB11

VCAP

VDD

# Bài tập: USB HID

## ❑ Cấu hình Middleware and Software component

- | Chọn USB\_DEVICE
- | Đặt chế độ Human Interface Device (HID)

The screenshot displays the STM32CubeMX Pinout & Configuration window. The 'Pinout & Configuration' tab is active, showing the 'USB\_DEVICE Mode and Configuration' section. The 'Mode' section has 'Class For HS IP' set to 'Human Interface Devic...' and 'Class For FS IP' set to 'Disable'. The 'Configuration' section includes a 'Reset Configuration' button and 'User Constants' checked. The 'Parameter Settings' and 'Device Descriptor' tabs are selected. The 'Device Descriptor' section shows 'VID (Vendor ID..1155)', 'LANGID\_STRI.. English(United States)', and 'MANUFACTU... STMicroelectronics'. The 'Device Descriptor HS' section is also visible. On the right, the 'Pinout' section shows a list of pins: PD13, PD12, PD11, PD10, PD9, PD8, PB15, PB14, PB13, and PB12. The 'PB13' pin is highlighted in green. The bottom of the window shows a search bar and navigation icons.

## Bài tập: USB HID

- ❑ Khai báo kiểu mouseHID trong main.h, đóng gói 4 byte dữ liệu ứng với USB Mouse report
  - | button: trạng thái nút chuột được bấm. 1 là nút trái, 2 là nút phải.
  - | mouse\_x, mouse\_y: di chuyển của chuột theo 2 phương x và y.
  - | wheel: giá trị cuộn.

```
37- /* Exported types -----  
38  /* USER CODE BEGIN ET */  
39- typedef struct  
40  {  
41      uint8_t button;  
42      int8_t mouse_x;  
43      int8_t mouse_y;  
44      int8_t wheel;  
45  } mouseHID;  
46  /* USER CODE END ET */
```

## Bài tập: USB HID

---

### ❑ Khai báo biến trong main.c

```
58- /* Private user code -----  
59  /* USER CODE BEGIN 0 */  
60  #include "usbd_hid.h"  
61  
62  extern USB_D_HandleTypeDef hUsbDeviceHS;  
63  
64  mouseHID mousehid = {0,0,0,0};  
65  /* USER CODE END 0 */
```

# Bài tập: USB HID

---

## ❑ Thêm code điều khiển mouse vào main loop

```
94  /* Initialize all configured peripherals */
95  MX_GPIO_Init();
96  MX_USART1_UART_Init();
97  MX_USB_DEVICE_Init();
98  /* USER CODE BEGIN 2 */
99
100 /* USER CODE END 2 */
101
102 /* Infinite loop */
103 /* USER CODE BEGIN WHILE */
104 while (1)
105 {
106     int button_pressed = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0);
107     if (button_pressed)
108     {
109         mousehid.mouse_x = 1;
110         mousehid.mouse_y = 1;
111         USBD_HID_SendReport(&hUsbDeviceHS, (uint8_t *)&mousehid, sizeof (mousehid));
112     }
113
114     HAL_Delay(100);
115     /* USER CODE END WHILE */
116
117     /* USER CODE BEGIN 3 */
118 }
119 /* USER CODE END 3 */
```



## Bài tập: USB HID

---

- ❑ Nạp và chạy code trên mạch.
- ❑ Cắm dây micro USB vào cổng USB USER (cạnh dưới màn hình LCD) và nối với máy tính.
- ❑ Quan sát: khi bấm nút User (PA0) thì con trỏ chuột di chuyển.

# Bài tập: USB HID

---

## ❑ Giải thích hoạt động của mạch.



## Bài tập: USB HID

---

- ❑ Tự làm: xử lý ngắt khi bấm nút PA0 để gửi sự kiện mouse up/mouse down tương ứng khi nút PA0 được bấm/nhả.