Họ và tên	Phạm Quốc Tiến	Nguyễn Đức Trung	Bùi Thái Toàn
MSSV	22521472	22521564	22521485
Họ và tên	Trịnh Thành Trung	Trần Quốc Trinh	Nhật Tin
MSSV	22521570	22521542	22521479
Họ và tên	Nguyễn Hữu Bảo Trọng		
MSSV	22521545		

LAB 3: GIAO TIẾP VIRTUAL COM PORT THÔNG QUA CỔNG MICRO USB

1 CHUẨN BỊ

1.1 Phần cứng

- KIT STM32F4 Discovery for STM32F429 MCU.

1.2 Phần mềm

- STM32CubeIDE: sử dụng để lập trình, build, nạp và debug code.

1.3 Kiến thức

- Biết cách tạo project và cấu hình project sử dụng STM32CubeIDE
- Hiểu về cách thức giao tiếp qua serial

Hình 1.1. KIT STM32F4 Discovery

2 HƯỚNG DẪN

2.1 Cấu hình USB OTG HS

STM32F429ZIT6 hỗ trợ 2 loại USB OTG (On-the-go) bao gồm:

- USB OTG FS (Full speed): có thể hoạt động như một thiết bị/host/ngoại vi OTG ở mức full speed (12Mbps)

- USB OTG HS (High speed): có thể hoạt động như một thiết bị/host/ngoại vi OTG ở mức full speed (12Mbps), hoặc high speed (480Mbps). Lưu ý, HS core trong hầu hết các dòng MCU của ST sẽ yêu cầu external ULPI PHY để chip có thể thực sự chạy ở mức high speed. Trong trường hợp sử dụng PHY on-chip (internal PHY) thì nó chỉ có thể chạy ở mức full speed.

Trong giao tiếp giữa KIT STM32F429ZIT6 và máy tính, ta sẽ chọn máy tính là host và KIT là device. Để sử dụng cổng micro-USB giao tiếp, theo tài liệu hướng dẫn sử dụng, ta cần cài đặt KIT chạy USB OTG HS ở chế độ full speed (sử dụng internal PHY).

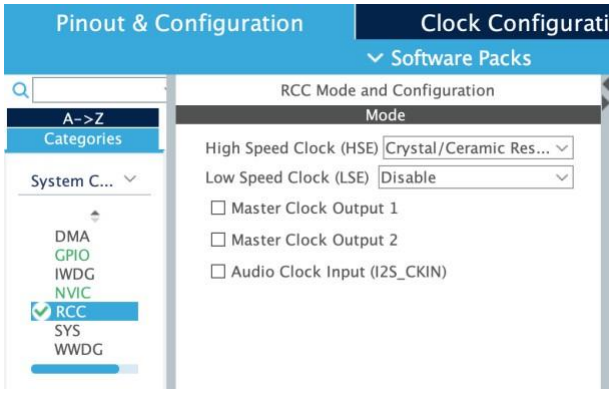
PIN OUT của USB OTG được trình trong Hình 2.1:

USB	FS Mode	HS in FS Mode	Description
Data +	PA12	PB15	USB Data+ line
Data –	PA11	PB14	USB Data- line
ID	PA10	PB12	USB ID pin
VBUS	PA9	PB13	USB activate

Hình 2.1. Pin out các chân khi sử dụng USB OTG

Thực hiện tạo project (phần Code Generation Options chọn **Copy only the necessary library files**) và cấu hình như sau:

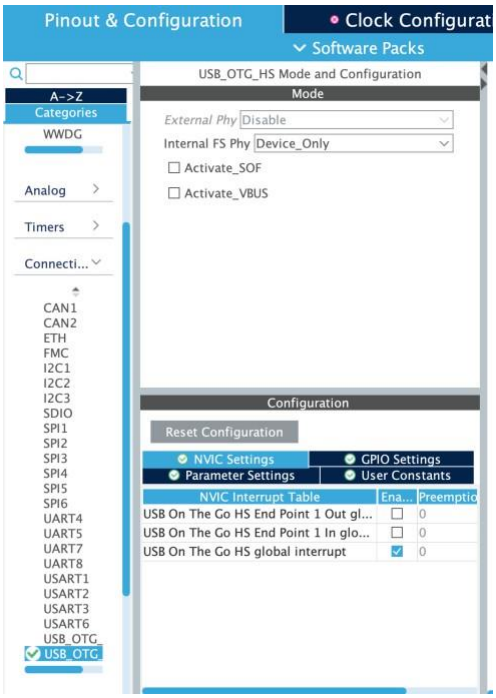
- Cài đặt HSE



Hình 2.2. Cài đặt HSE trong mục RCC

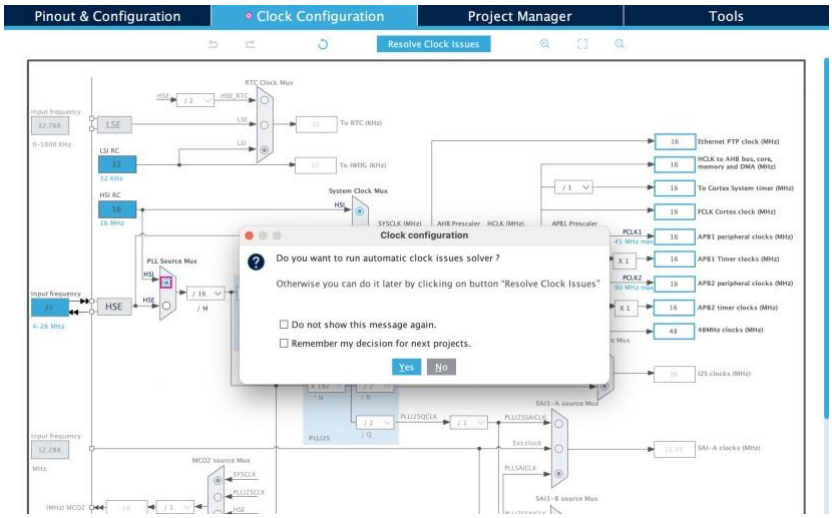
- Trong mục Connectivity, chọn **USB OTG HS**:

- Cài đặt **Internal FS PHY**: **Device_Only** (cài đặt KIT sử dụng internal PHY và hoạt động như một device)
- Cài đặt **NVIC**: bật **USB On The Go HS global interupt** (cho phép MCU sử dụng các ngắt, thường sử dụng cho các thao tác truyền hoặc nhận dữ liệu)



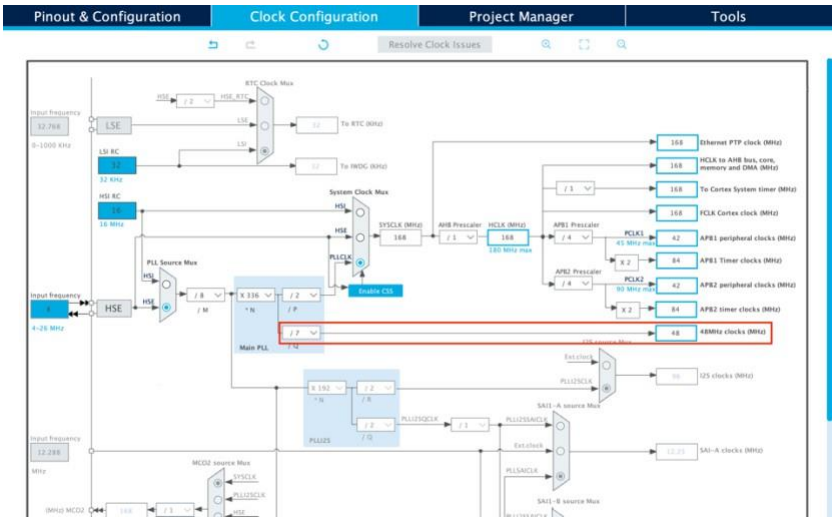
Hình 2.3. Cài đặt USB OTG HS

- Cài đặt Clock:
  - Hệ thống sẽ báo là clock đang bị lỗi và có muốn hệ thống tự chỉnh clock hay không, ở bước này, ta chọn **No**.



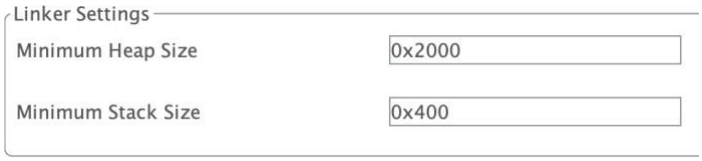
Hình 2.4. Chọn No để tự cấu hình clock

- Chỉnh clock như hình bên dưới, trong đó, ta sẽ thấy có một đường Clock mới do KIT có sử dụng USB OTG. Để đạt được tốc độ tối đa trên đường clock này là **48MHz**, ta buộc phải giảm tốc độ SYSCLOCK xuống còn **168MHz** để có thể đưa vào các bộ chia tương ứng.



Hình 2.5. Cấu hình clock

- Tiếp theo, trong tab **Project Manager**, ta cần chỉnh lại **Minimum Heap Size** là 0x2000, điều này sẽ tăng vùng nhớ tối theo dành cho Heap section lên là 8KB (0x2000h = 8192d).



Hình 2.6. Đặt lại Minimum Heap Size

Sau khi thực hiện cấu hình hoàn tất, bấm **File → Save** để lưu và sinh code.

2.2 Lập trình cho USB OTG HS

Việc giao tiếp qua cổng micro-USB sẽ được máy tính thực hiện như đang giao tiếp với một Virtual Com Port - VCP, việc giao tiếp này tương tự như giao tiếp qua UART. Để kiểm tra xem liệu KIT có thể gửi và nhận dữ liệu hay không, ta sẽ viết một chương trình echo dữ liệu, tức là KIT nhận được dữ liệu gì thì sẽ gửi lại dữ liệu tương tự cho máy tính.

Trước tiên, ta cần thêm thư viện **usbd_cdc_if.h** vào file main.c:

```
/* USER CODE END Header */
/* Includes ------------------------------------------*/
#include "main.h"
#include "usb_device.h"
#include "usbd_cdc_if.h"
```

Thư viện này sẽ chứa các hàm dùng để gửi và nhận dữ liệu qua USB CDC.

Tiếp theo, trong project, tìm file **USB_DEVICE/App/usbd_cdc_if.h** để thực hiện chỉnh lại hàm nhận dữ liệu **CDC_Receive_HS** như bên dưới:

```
tatic int8_t CDC_Receive_HS(uint8_t* Buf, uint32_t *Len)
{
  /* USER CODE BEGIN 11 */
  USBD_CDC_SetRxBuffer(&hUsbDeviceHS, &Buf[0]);
  USBD_CDC_ReceivePacket(&hUsbDeviceHS);
  CDC_Transmit_HS(Buf,*Len);
```

```
  return (USBD_OK);
  /* USER CODE END 11 */
}
```

Dễ thấy, sau khi chỉnh sửa, hàm này sẽ thực hiện **CDC_Transmit_HS** tức là gửi chính xác lại những dữ liệu mà nó nhận được.

Tiếp theo, để thử nghiệm chương trình, ta cần cài đặt:

- Driver Virtual Com Port cho STM32F429 (chỉ dùng cho hệ điều hành Windows, hệ điều hành OSX có thể nhận mà không cần cài thêm driver)
- Phần mềm gửi serial: sinh viên có thể sử dụng bất kỳ phần mềm nào để gửi hoặc nhận serial, ví dụ như putty.

Sau khi đã cài đặt các phần mềm hỗ trợ cần thiết, cắm dây mini-USB vào đầu cổng ST-Link để thực hiện nạp code và debug như bình thường. Sau khi chương trình được nạp và chạy, dùng 1 dây micro-USB cắm vào cổng tương ứng trên KIT, đầu còn lại cắm vào máy tính. Trong máy tính Windows, ta có thể kiểm tra bằng cách mở **Device Manager** → **Port** để kiểm tra xem máy tính đã nhận đủ 2 cổng COM có dạng:

- *ST-Link Virtual Com Port*
- *Virtual Com Port*

Sau khi thấy máy tính đã nhận Virtual Com Port (không có chữ ST-Link), ta mở putty, chọn giao tiếp Serial. Các thông số cài đặt như Baudrate, Stop bit, Parity có thể giữ nguyên vì KIT sẽ nhận được hết.

Thực hiện thử nghiệm bằng cách gửi một thông điệp bất kỳ, KIT sẽ phản hồi lại dữ liệu tương tự.

3 BÀI TẬP

Bài tập 1: Viết một chương trình nhúng thực hiện những việc sau:

- Thực hiện cài đặt USB OTG HS để giao tiếp với máy tính
- Đầu tiên, màn hình LCD hiển thị dòng chữ "Good evening, it's too dark here"

- Nếu KIT nhận được dữ liệu "Hello STM" từ máy tính thì thực hiện in dòng chữ "Do you want to turn on light" và chờ liên tục các phản hồi sau:
  - Nếu máy tính gửi dữ liệu "Turn on LED 3" thì thực hiện bật LED 3
  - Nếu máy tính gửi dữ liệu "Turn on LED 4" thì thực hiện bật LED 4
  - Nếu máy tính gửi dữ liệu "Turn off LED 3" thì thực hiện tắt LED 3
  - Nếu máy tính gửi dữ liệu "Turn off LED 4" thì thực hiện tắt LED 4
  - Nếu máy tính gửi dữ liệu "Good night" thì tắt cả 2 đèn

Khai báo trong file main.c

```
/* USER CODE BEGIN PV */
char data[15];
```

Khai báo trong file usbd_cdc_if.c

```
/* Private variables ---------
extern char data[15];
/* USER CODE END PV */
```

Mục đích của việc khai báo extern là để có thể mở rộng phạm vi biến data từ file main.c qua file usbd_cdc_if.c

Tiếp theo trong file usbd_cdc_if.c nhóm sẽ sử dụng hàm sprintf để convert data nhận được sang dạng chuỗi char mỗi khi kit nhận tín hiệu serial

```
static int8_t CDC_Receive_HS(uint8_t* Buf, uint32_t *Len)
{
  /* USER CODE BEGIN 11 */
  USBD_CDC_SetRxBuffer(&hUsbDeviceHS, &Buf[0]);
  USBD_CDC_ReceivePacket(&hUsbDeviceHS);
  if (*Len > 0) {
     snprintf(data, sizeof(data), "%.*s", (int)*Len, Buf);
  }
  return (USBD_OK);
  /* USER CODE END 11 */
}
```

Điều này giúp nhóm có thể đưa data sang file main.c để có thể xử lý dễ dàng hơn

Trong hàm main của file main.c nhóm sẽ setup màn hình LCD đồng thời hiện dòng chữ "Good evening, it's too dark here" lên LCD

```
92    BSP_LCD_Init();
93    BSP_LCD_LayerDefaultInit(1, SDRAM_DEVICE_ADDR);
94    BSP_LCD_SelectLayer(1);
95    BSP_LCD_DisplayOn();
96    BSP_LCD_Clear(LCD_COLOR_BLUE);
97    BSP_LCD_SetBackColor(LCD_COLOR_BLUE);
98    BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
99    char text[15];
100   sprintf(text, "Good evening, ");
101   BSP_LCD_DisplayStringAtLine(1, (uint8_t*)text);
102   sprintf(text, "it's too dark");
103   BSP_LCD_DisplayStringAtLine(2, (uint8_t*)text);
104   sprintf(text, "here");
105     BSP_LCD_DisplayStringAtLine(3, (uint8_t*)text);
```

Trong phần while(1) nhóm sẽ thực hiện xử lý dữ liệu từ file usbd_cdc_if.c bằng hàm strcmp và thực hiện theo yêu cầu của đề bài

```
110   while (1)
111   {
112      if(strcmp(data, "Hello STM") == 0)
113      {
114          BSP_LCD_Clear(LCD_COLOR_BLUE);
115          sprintf(text, "Do you want to");
116          BSP_LCD_DisplayStringAtLine(1, (uint8_t*)text);
117          sprintf(text, "turn on light");
118            BSP_LCD_DisplayStringAtLine(2, (uint8_t*)text);
119          while(1)
120          {
121              if (strcmp(data, "Turn on LED 3") == 0) {
122                  HAL_GPIO_WritePin(GPIOG, LED_3_Pin, 1);
123              } else if (strcmp(data, "Turn on LED 4") == 0) {
124                  HAL_GPIO_WritePin(GPIOG, LED_4_Pin, 1);
125              } else if (strcmp(data, "Turn off LED 3") == 0) {
126                  HAL_GPIO_WritePin(GPIOG, LED_3_Pin, 0);
127              } else if (strcmp(data, "Turn off LED 4") == 0) {
128                  HAL_GPIO_WritePin(GPIOG, LED_4_Pin, 0);
129              } else if (strcmp(data, "Good night") == 0) {
130                  HAL_GPIO_WritePin(GPIOG, LED_3_Pin, 0);
131                  HAL_GPIO_WritePin(GPIOG, LED_4_Pin, 0);
132              }
133          }
134      }
```