CHUYÊN ĐỀ THIẾT KẾ HỆ THỐNG NHÚNG 1

BÀI THỰC HÀNH 1: LẬP TRÌNH NGOẠI VI TRÊN MÔ HÌNH XE CE-BOSCH

I. Mục tiêu

- Giúp sinh viên cài đặt một số công cụ hỗ trợ lập trình và làm quen với mô hình xe CE-BOSCH sử dụng trong quá trình thực hành
- Giúp sinh viên biết cách tạo các dự án mới và lập trình cho vi điều khiển STM32F103
- Giúp sinh viên làm quen với việc lập trình điều khiển GPIO, UART, TIMER, … trên vi điều khiển STM32F103

II. Chuẩn bị trước

- Tìm hiểu các thông tin cơ bản về KIT STM32F103C8T6 Bluepill
- Tìm hiểu các thông tin cơ bản trong datasheet của vi điều khiển STM32F103C8T6 [1]
- Tải schematic của KIT STM32F103C8T6, KIT ESP32 Node MCU, mạch Wifi, Actuator và Sensor trong mô hình xe CE-BOSCH.

III. Nội dung thực hành

1. Cài đặt phần mềm STM32CubeIDE

Sinh viên tải và cài đặt phần mềm STM32CubeIDE. Đây là một công cụ để hỗ trợ sinh viên soạn thảo chương trình, biên dịch chương trình thành các file thực thi để có thể nạp xuống cho các dòng vi điều khiển của STMicroelectronics.

2. Tạo dự án mới trên STM32CubeIDE

Sinh viên làm quen với việc tạo một dự án mới trên công cụ STM32CubeIDE để lập trình cho vi điều khiển STM32F103C8T6. Cấu hình xung clock cho bộ xử lý. Cấu hình các driver và cài đặt các chế độ hoạt động cho các ngoại vi.

3. Lập trình điều khiển GPIO

Sinh viên thực hiện lập trình điều khiển các đèn LED và các nút nhấn thực hiện các chức năng theo yêu cầu của Giảng viên hướng dẫn.

4. Lập trình UART

Sinh viên thực hiện lập trình UART và thực hiện giao tiếp UART giữa các mạch và và in dữ liệu ra màn hình LCD đã được cung cấp.

5. Lập trình TIMER

Sinh viên làm quen với việc cài đặt và lập trình Timer để sử dụng định thời và tìm hiểu các tính năng khác của Timer.

IV. Hướng dẫn thực hành

1. Cài đặt phần mềm STM32CubeIDE

Sinh viên truy cập vào đường link sau để tải phần mềm STM32CubeIDE:

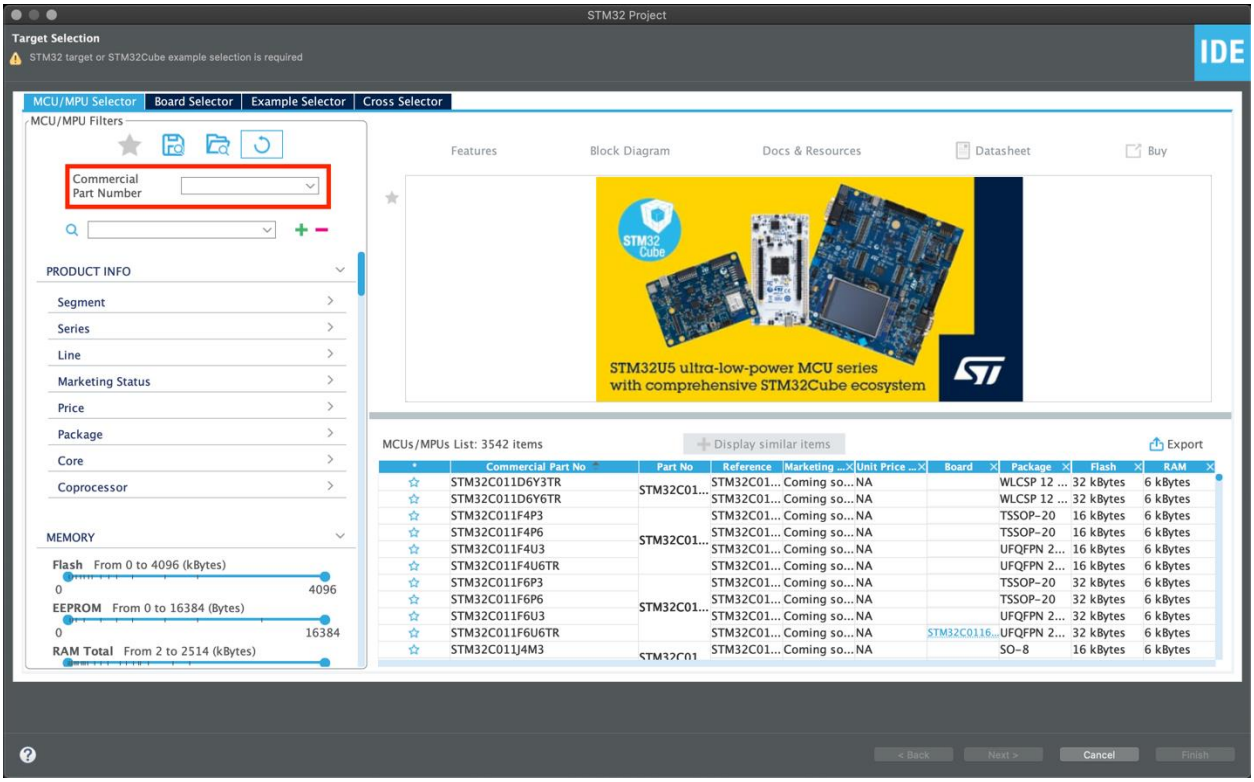
https://www.st.com/en/development-tools/stm32cubeide.html

Để tải được phần mềm, sinh viên cần đăng ký tài khoản và đăng nhập vào tài khoản đã đăng ký. Sau khi tải xong phần mềm, sinh viên tiến hành cài đặt như các phần mềm thông thường.

2. Tạo dự án mới trên STM32CubeIDE

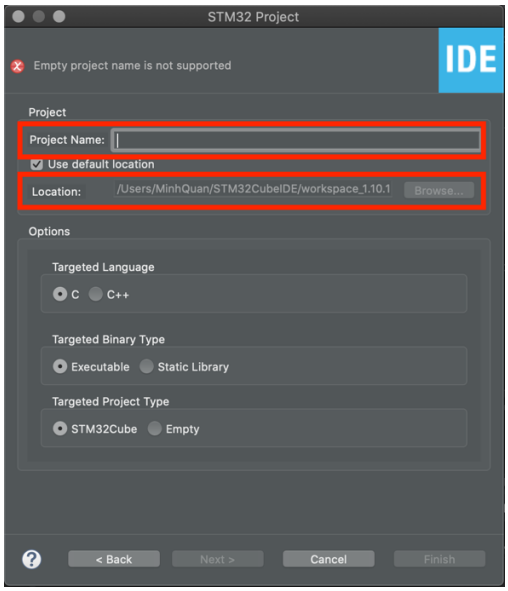
Tạo dự án mới (New->STM32 Project) trên công cụ STM32CubeIDE để lập trình cho vi điều khiển STM32F103C8T6. Tại ô tìm kiếm Commercial part number nhập vào mã của vi điều khiển là: STM32F103C8T6. Sau đó chọn đúng mã vi điều khiển ở phần cửa sổ bên phải như **Hình 1**. Sau đó bấm nút next để tiếp tục thực hiện các cài đặt tuỳ chỉnh cho dự án.

Tại cửa sổ STM32 Project đặt tên cho dự án (project) mình vừa tạo và lựa chọn vị trí folder để lưu lại dự án. Sau đó, sinh viên có thể nhấn nút next để thực hiện các điều chỉnh khác hoặc chọn nút finish để hoàn thành phần đặt tên cho dự án như **Hình 2**.

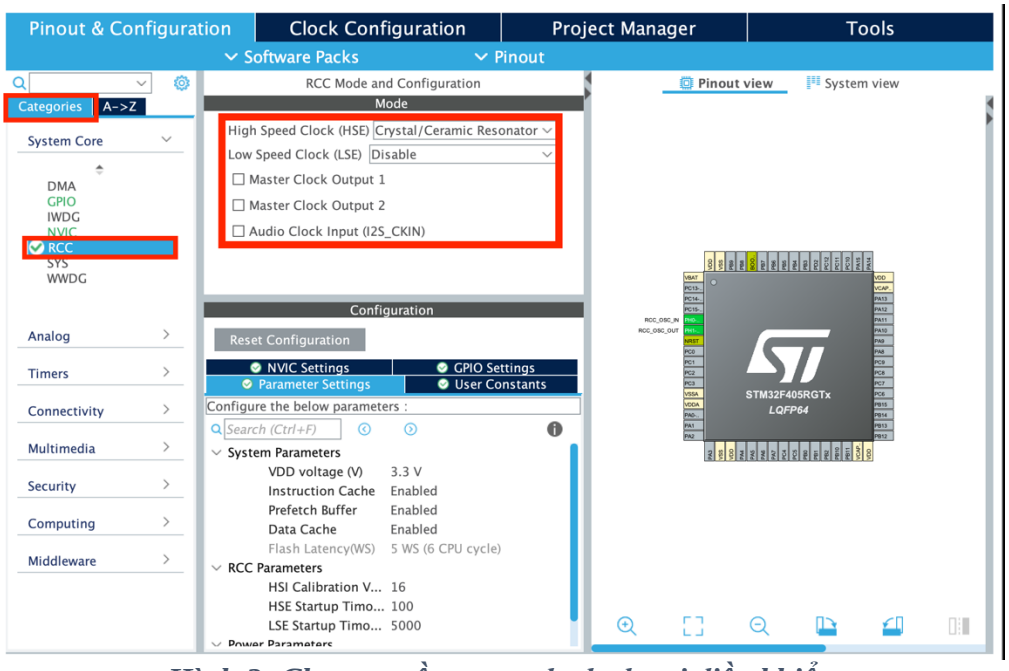


Hình 1: Màn hình chọn mã vi điều khiển

Sau khi tạo xong dự án mới, sinh viên thực hiện cấu hình chọn nguồn xung clock cho vi điều khiển hoạt động. Tại thẻ Categories->System Core, sinh viên chọn RCC để cấu hình nguồn xung clock chính cho vi điều khiển. Với High Speed Clock, sinh viên có thể chọn Crystal/Ceramic Resonator để có thể sử dụng thạch anh ngoại 8Mhz đã được gắn sẵn trên Kit như là nguồn xung clock chính cho vi điều khiển. Nếu sinh viên muốn sử dụng nguồn xung clock cho các tác vụ thời gian thực thì sinh viên chọn nguồn xung tại dòng Low Speed Clock là Crystal/Ceramic Resonator để sử dụng thạch anh 32.768kHz là nguồn xung clock cho các tác vụ thời gian thực như **Hình 3**.

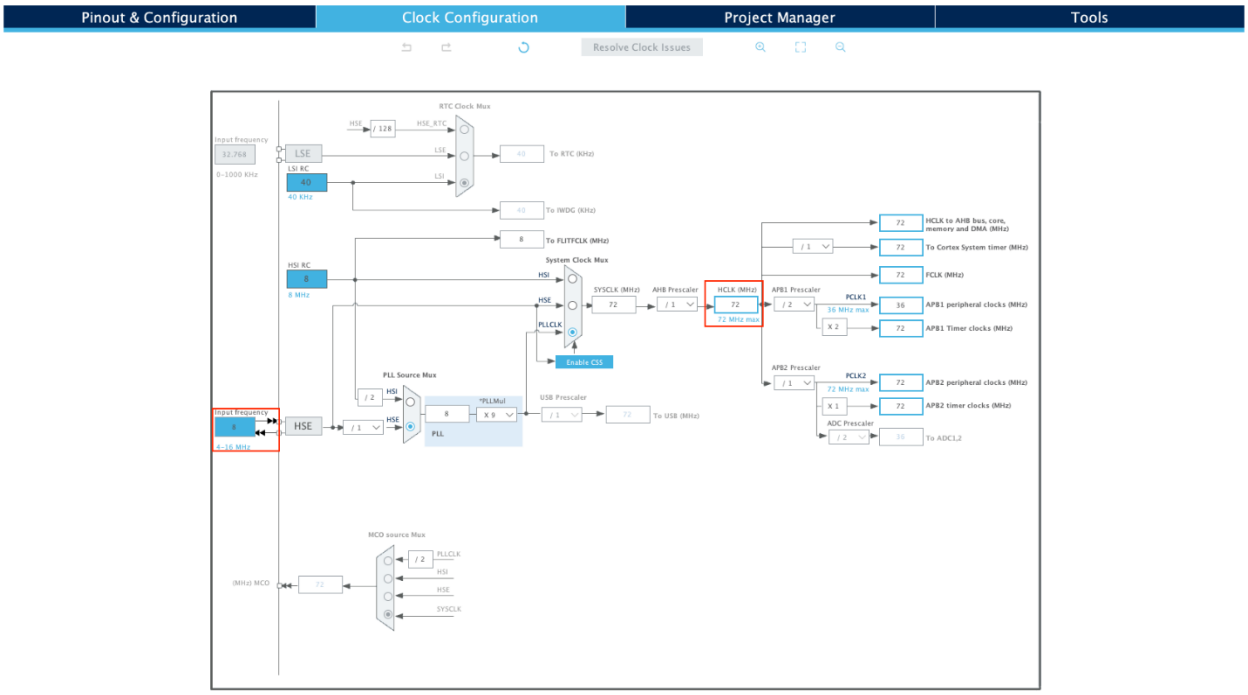


Hình 2: Đặt tên cho dự án



Hình 3: Chọn nguồn xung clock cho vi điều khiển

Sau khi đã chọn xong nguồn xung clock, sinh viên chọn thẻ Clock Configuration để thực hiện cấu hình đúng tần số mong muốn để vi điều khiển hoạt động. Theo như sơ đồ nguyên lý của nhà sản xuất, Kit được hàn sẵn thạch anh 8MHz là nguồn xung clock chính cho vi điều khiển hoạt động nên chúng ta sẽ nhập 8 vào vị trí ô input frequency. Vi điều khiển STM32F103C8T6 có thể hoạt động với tần số tối đa 72MHz nên chúng ta sẽ nhập vào ô HCLK là 72 để vi điều khiển hoạt động ở tần số cao nhất. Sau đó, chúng ta đợi cho phần mềm tính toán và điều chỉnh các hệ số của bộ nhân tần số và chúng ta có được kết quả như **Hình 4**.

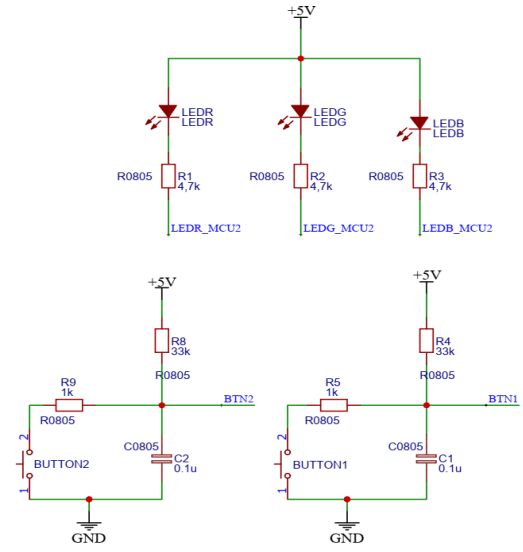


Hình 4: Cấu hình xung Clock cho vi điều khiển

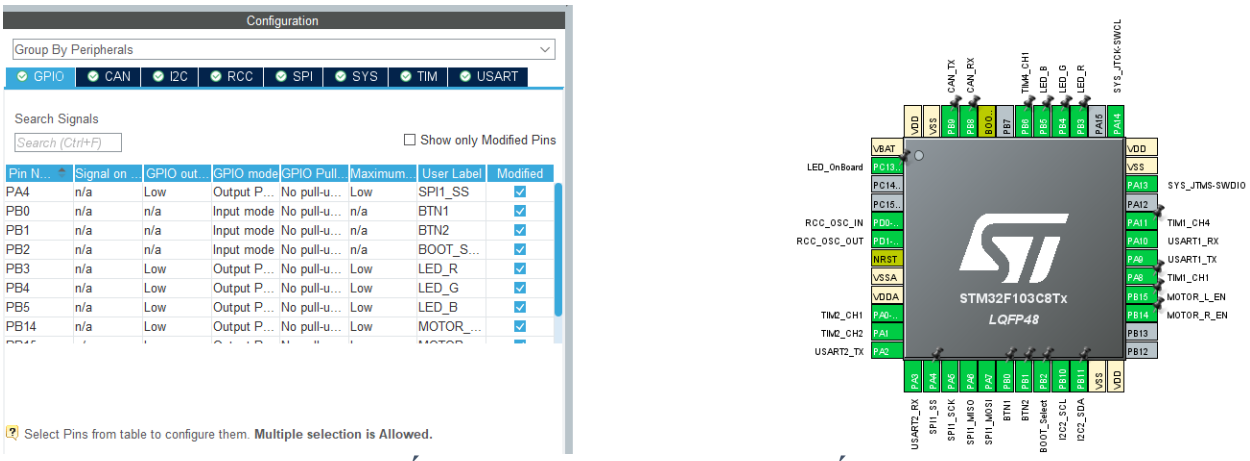
3. Lập trình điều khiển GPIO

Sinh viên xem sơ đồ nguyên lý của mạch Actuator, chúng ta có 3 loại LEDs với 3 màu khác nhau và 2 nút nhấn để phục vụ sinh viên trong quá trình debug sau này như trong **Hình 5**.

Sau đó, sinh viên cài đặt cấu hình trên STM32CubeIDE cho các chân PB5, PB4, PB3 lần lượt là các chân GPIO_Output, các chân PA12, PA15 là các chân GPIO_Input. Sau đó, sinh viên có thể cấu hình các thuộc tính, các tần số và đặt tên gợi nhớ cho các chân trên tại thẻ Pinout & Configuration như trong **Hình 6**.



Hình 5: Sơ đồ nguyên lý các nút nhấn và LEDs



Hình 6: Cấu hình và đặt tên có các nút nhấn và LEDs

Để điều khiển các chân LED, sinh viên có thể gọi 2 hàm của thư viện HAL gồm:

- HAL_GPIO_Toggle: Đảo giá trị hiện tại của chân output.
- HAL_GPIO_WritePin: Ghi một giá trị ra chân output.

Ví dụ về hàm điều khiển LED như trong **Hình 7**.

```
HAL_GPIO_TogglePin(GPIOB, LED0_Pin);
HAL_GPIO_WritePin(LED0_GPIO_Port, LED0_Pin, GPIO_PIN_RESET);
HAL_GPIO_WritePin(LED0_GPIO_Port, LED0_Pin, GPIO_PIN_SET);
```

Hình 7: Ví dụ hàm điều khiển các chân LED

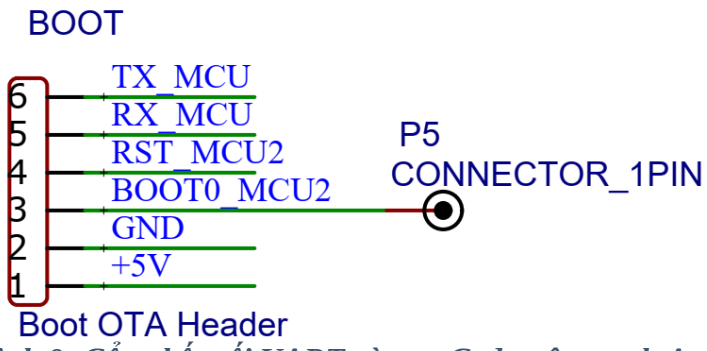
Để đọc được trạng thái của các nút nhấn, sinh viên có thể sử dụng hàm HAL_GPIO_ReadPin như ví dụ trong **Hình 8**.

```
HAL_GPIO_ReadPin(KEY0_GPIO_Port, KEY0_Pin);
```

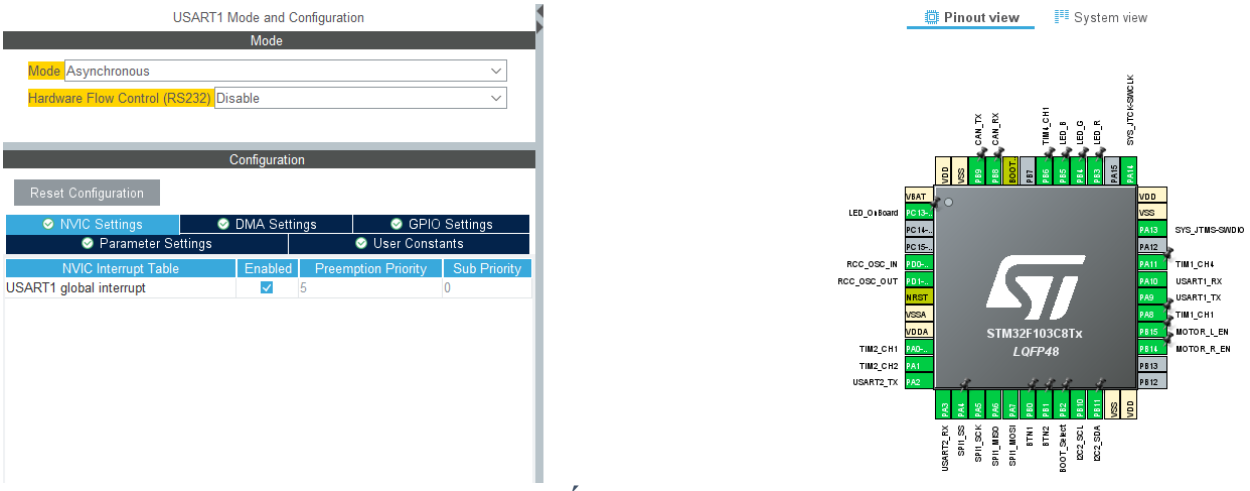
Hình 8: Ví dụ hàm đọc trạng thái nút nhấn

4. Lập trình UART

Theo sơ đồ nguyên lý của mạch Actuator, chúng ta sẽ có 2 port UART trong đó: port UART1 như trong **Hình 9** dùng để nạp Code và giao tiếp với mạch Wifi, port UART2 được mở để có thể thực hiện kết nối và giao tiếp với máy tính. Trong phần thực hành này, sinh viên sẽ thực hiện viết chương trình và cấu hình cho cổng UART1 để giao tiếp với mạch Wifi và theo dõi dữ liệu được xuất ra trên màn hình LED.



Hình 9: Cổng kết nối UART và nạp Code trên mạch Actuator



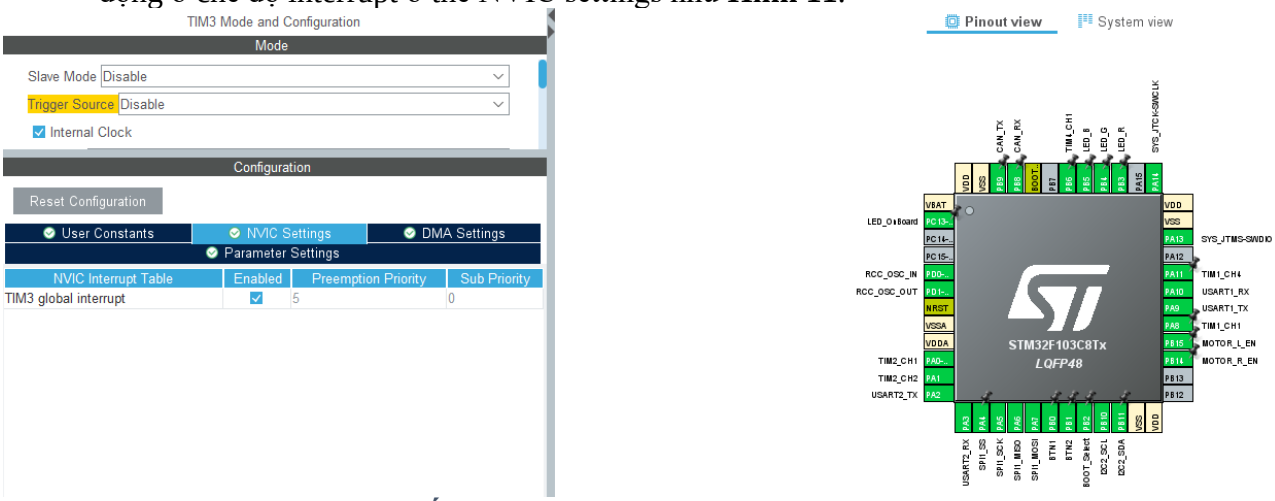
Hình 10 Cấu hình cho UART1

5. Lập trình TIMER

Trên vi điều khiển STM32F103C8T6 có cung cấp rất nhiều TIMER cho người dùng với nhiều chức năng rất đa dạng. Tuy nhiên, trong nội dung bài học này, sinh viên chỉ tìm hiểu cơ bản về timer interrupt với chức năng ngắt theo thời gian để có thể thay thế cho việc sử dụng HAL_Delay để điều chỉnh thời gian chu kỳ cho một tác vụ nào đó. Đối với chức năng cơ bản, sinh viên có thể sử dụng TIM3 là một timer 16 bit cơ bản để lập trình.

Sinh viên thực hiện cấu hình TIMER trên STM32CubeIDE theo các bước sau:

- Tại thẻ Timers, sinh viên chọn TIM3 sau đó chọn Clock source là Internal Clock. Sau đó sinh viên sẽ cấu hình các giá trị cần thiết cho timer cũng như bật cờ cho phép timer hoạt động ở chế độ interrupt ở thẻ NVIC settings như **Hình 11**.



Hình 11 Cấu hình cho phép interrupt cho TIMER 3

Để bắt đầu cho TIMER hoạt động theo chế độ interrupt sinh viên sử dụng hàm sau:

```
HAL_TIM_Base_Start_IT(&htim3);
```

Để xử lý các tác vụ trong interrupt sinh viên cần hiện thực hàm callback sau:

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  //Check which version of the timer triggered this callback and toggle LED
  if (htim == &htim3 )
  {
    //To do:
  }
}
```

6. Bài tập ôn tập

Sau khi hoàn thành các phần trên, sinh viên tổng hợp lại những kiến thức bên trên để hoàn thành chương trình điều khiển GPIO theo yêu cầu sau.

- Sinh viên viết chương trình để tạo ra ít nhất 3 hiệu ứng chớp/tắt trên các LEDs sử dụng TIMER Interrupt để điều khiển chu kỳ chớp tắt trong khoảng thời gian từ 100ms đến 2s.
- Sinh viên đọc trạng thái nút nhấn BTN1 trên mạch Actuator: nếu nút nhấn BTN1 được nhấn với thời lượng dưới 500ms thì sinh viên thực hiện giảm thời gian chu kỳ hiện tại của các hiệu ứng LED ở trên xuống 100ms (Nếu thời gian chu kỳ giảm về 0 thì sinh viên thực hiện gán lại thời gian chu kỳ là 2s); nếu nút nhấn BTN1 được nhấn giữ lâu hơn 500ms thì sinh viên thực hiện giảm thời gian chu kỳ của hiệu ứng LED đi 100ms sau mỗi khoảng thời gian 200ms nút nhấn được giữ.
- Sinh viên đọc trạng thái nút nhấn BTN2 trên mạch Actuator: nếu nút nhấn BTN2 được nhấn dưới 500ms thì sinh viên thực hiện thay đổi lần lượt các hiệu ứng LED ở trên; nếu nút nhấn BTN2 được nhấn giữ lâu hơn 500ms thì sinh viên thực hiện tăng thời gian chu kỳ của hiệu ứng LED thêm 100ms sau mỗi khoảng thời gian 200ms nút nhấn được giữ.

V. Báo cáo thực hành

Sinh viên viết báo cáo theo mẫu đã được cung cấp để trình bày lại các chương trình và các kết quả đã làm được trong bài thực hành. Giải thích các đoạn chương trình mà sinh viên đã hiện thực để giải quyết các yêu cầu của bài thực hành.

VI. Tài liệu tham khảo