

Lập trình nhiều kênh ADC trên STM32F4 sử dụng DMA

Trong bài viết trước về ADC, chúng ta đã sử dụng chức năng ADC trên vi điều khiển STM32F407VG và sử dụng ngắt trên 1 kênh. Trong bài viết này, mình sẽ hướng dẫn các bạn đọc giá trị ADC trên nhiều kênh khác nhau, kết hợp với DMA để lưu kết quả thu được vào bộ nhớ.

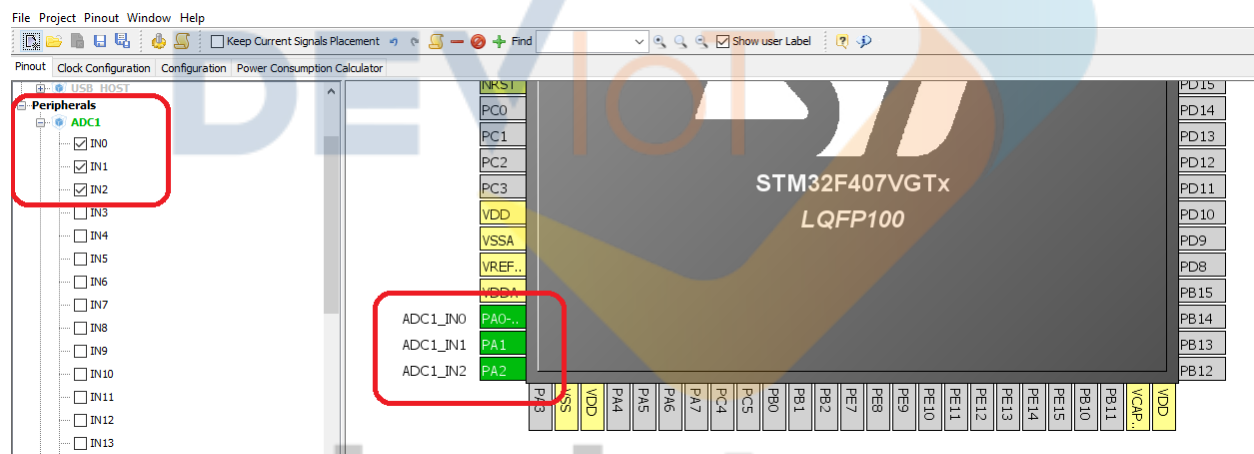
Mình sẽ sử dụng 3 Channel 0, 1, 2 của bộ ADC1 trên kit STM32F407VG để đọc giá trị từ 3 điện trở khác nhau. Các kênh này được kết nối lần lượt với các chân PA0, PA1, PA2 của KIT.

I, Cấu hình với CubeMX

Khởi động phần mềm CubeMX, chọn vi điều khiển STM32F407VG.

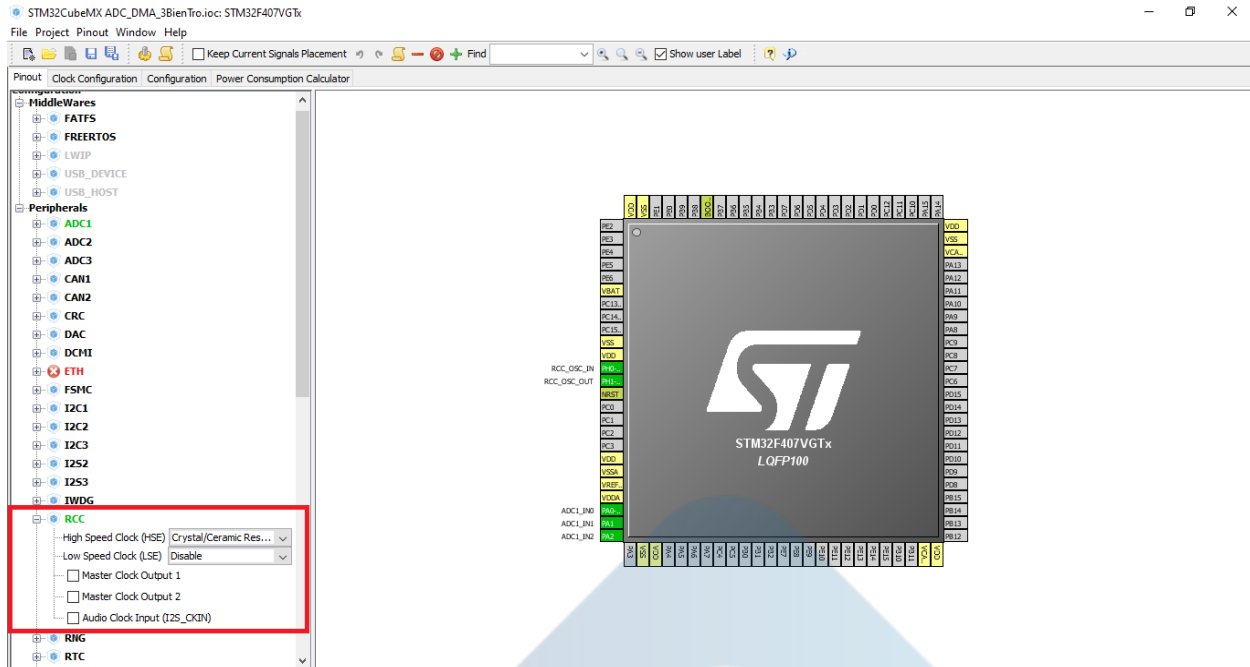
Tại phần **“Peripherals”**, trong mục ADC1, chúng ta click chọn **“IN0”**, **“IN1”**, **“IN2”** để kích hoạt các channel 0, 1, 2 của bộ ADC1.

Hoặc các bạn có thể tìm đến các chân PA0, PA1, PA2 và cấu hình cho từng chân.



Cấu hình thạch anh và clock cho chip

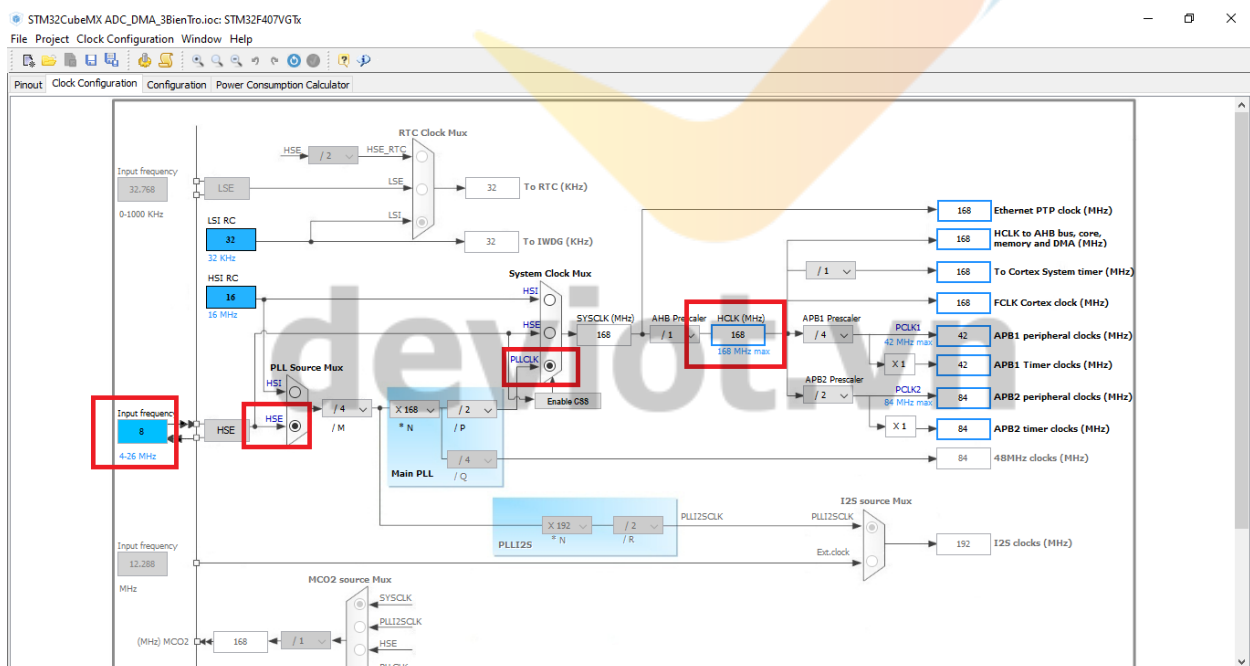
Tại mục RCC, trong phần **“High Speed Clock(HSE)”**, chúng ta chọn **“Crystal/Ceramic Resonator”** để cấu hình cho chip hoạt động với thạch anh ngoại.



Trong Tab “**Clock Configuration**”, các bạn thiết lập các thông số cho Clock của Chip.

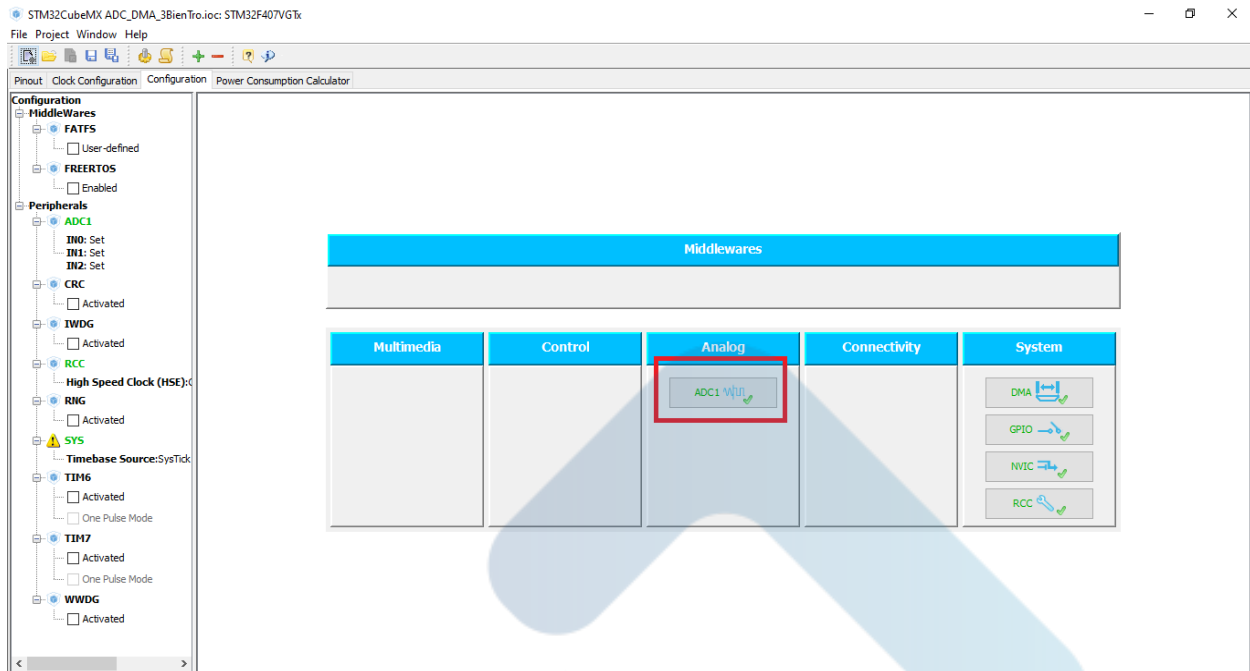
Chọn HSE, xung clock sẽ đi qua bộ nhân tần PLLCLK, giúp Chip hoạt động với tần số cao nhất.

Điền “8” ở mục “**Input frequency**” (thông số của thạch anh ngoại gắn trên KIT của mình) và điền “168” ở mục “**HCLK**” (168MHz là tần số tối đa của chip).



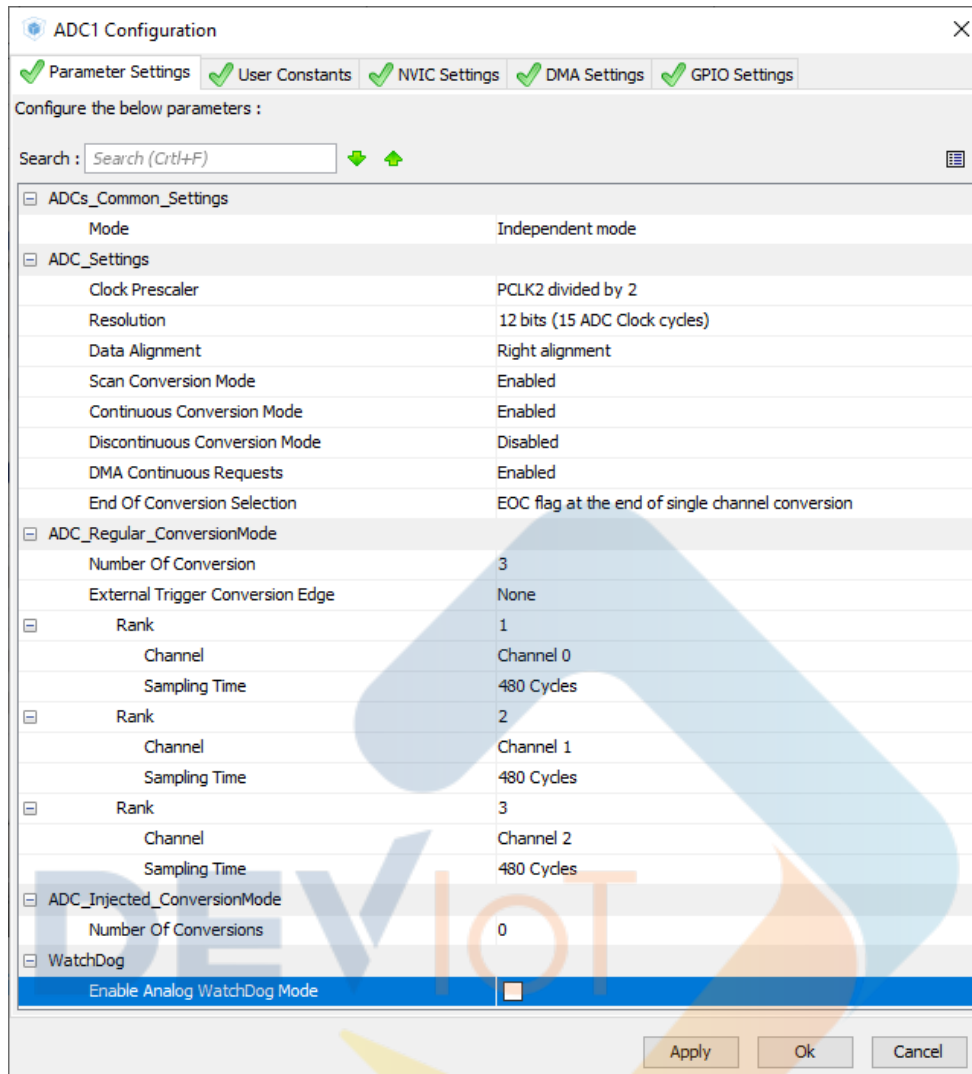
Cấu hình cho bộ ADC:

Chuyển qua Tab “**Configuration**”, chúng ta chọn mục ADC1:



Khi cửa sổ “**ADC1 Configuration**” hiện lên, các bạn thiết lập các thông số như hình dưới đây:

deviot.vn

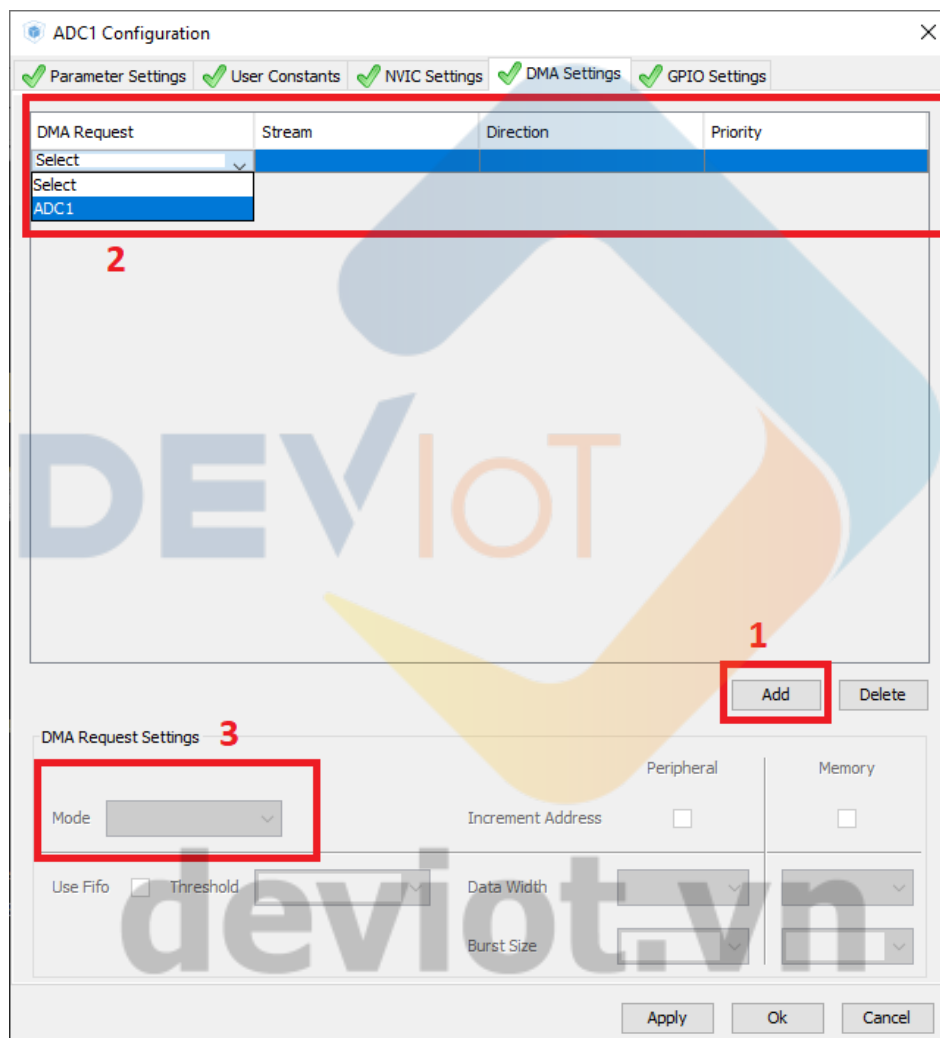


Mình sẽ giải thích qua một chút về các thông số đáng chú ý:

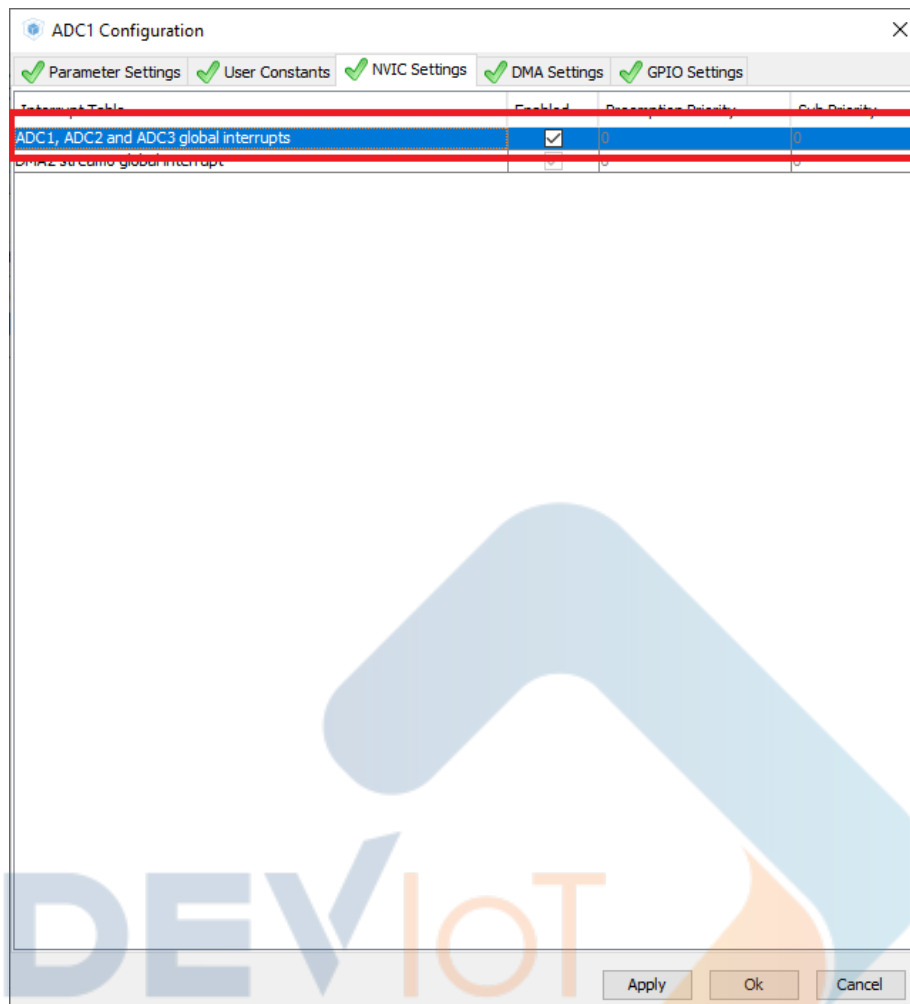
- **Scan Conversion Mode:** khi sử dụng 1 bộ ADC nhưng cần lấy mẫu trên nhiều kênh, chúng ta cần Enable cho chế độ này để quét liên tục các kênh
- **Continuous Conversion Mode:** cho phép lấy mẫu liên tục
- **DMA Continuous Requests:** ở bài này chúng ta sử dụng DMA nên sẽ phải Enable chức năng này.
- **Number Of Conversion:** số chuyển đổi được thực hiện, ở đây mình sẽ điền "3" vì sử dụng 3 kênh ADC.
- **Rank:** thiết lập mức ưu tiên cho các kênh ADC, kênh có rank nhỏ hơn thì độ ưu tiên cao hơn.

Tại Tab “**DMA Settings**”, chúng ta sẽ chọn “**Add**”, sau đó chọn “**ADC1**” ở mục “DMA Request”. Tiếp theo, tại mục Mode, các bạn chọn “Circular”. Ở mục Mode này có 2 lựa chọn là “**Normal**” và “**Circular**”:

- **Normal**: tại tất cả các lần lấy mẫu, DMA đều lưu trữ kết quả vào cùng 1 vị trí trong bộ nhớ, bị ghi đè lên nhau.
- **Circular**: các lần lấy mẫu sẽ được lưu vào các vị trí kế tiếp nhau trong bộ nhớ, không bị ghi đè lên nhau.



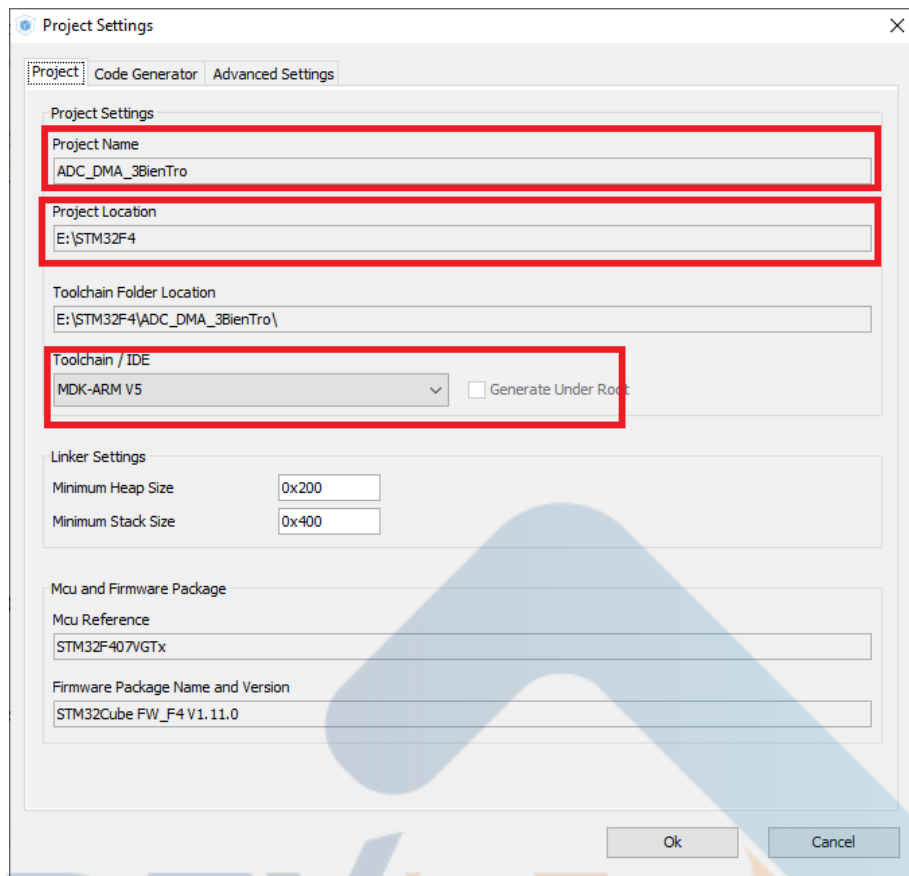
Chuyển qua Tab “**NVIC Settings**”, chúng ta sẽ Enable cho phép ngắt đối với các bộ ADC.



Setting project và sinh code

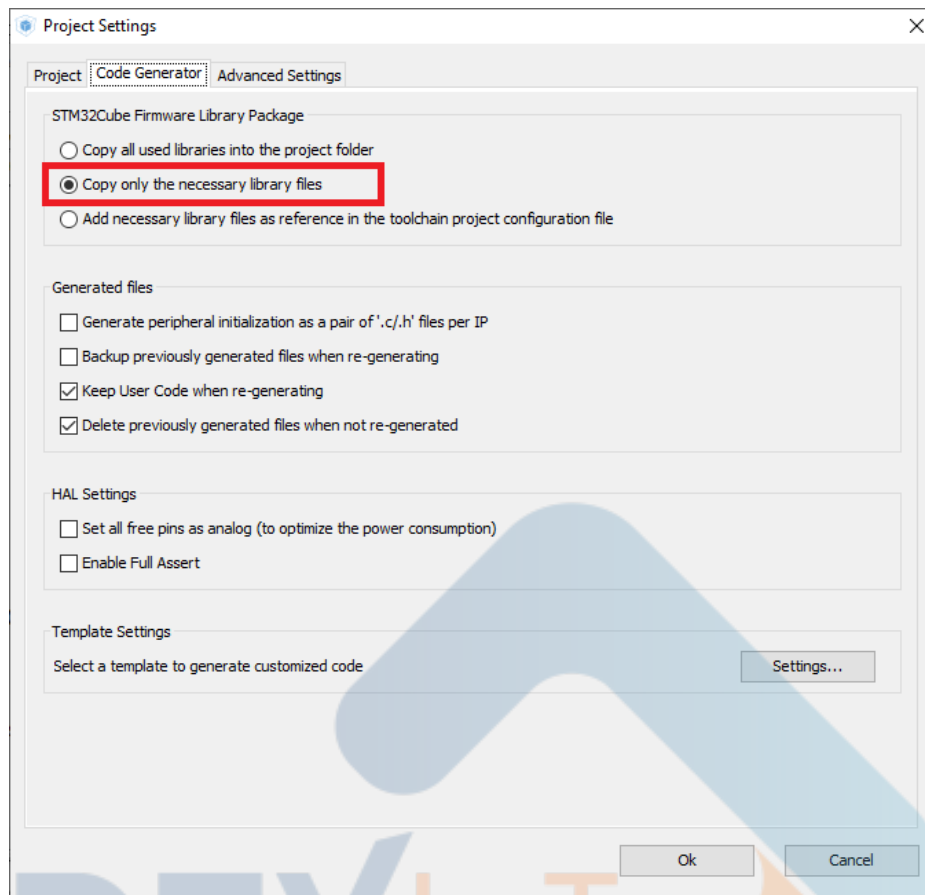
Các bạn thiết lập các thông tin cơ bản cho Project

deviot.vn



Chọn mục **“Copy only the necessary library files”** trong Tab **“Code Generator”** để Project sinh ra chỉ có những thư viện cần thiết, giảm dung lượng Project và tiết kiệm thời gian Build code nhé!

deviot.vn



II, Lập trình với KeilC

Sau khi CubeMX sinh code xong, chúng ta hãy chọn **“Open Project”** để mở chương trình và lập trình với KeilC.

Trong file “main.c”, chúng ta khai báo 1 mảng để lưu giá trị các giá trị nhận về sau khi quét

ADC : `volatile uint16_t res_value[3];`

- `res_value[0]` : lưu giá trị nhận về của kênh 0
- `res_value[1]` : lưu giá trị nhận về của kênh 1
- `res_value[2]` : lưu giá trị nhận về của kênh 2

Ở đây mình sử dụng **“volatile”** tránh những lỗi sai khó phát hiện do tính năng Optimization của Compiler vì giá trị của biến **“adc_value”** sẽ bị thay đổi giá trị do tác động của phần cứng.

Trong hàm **“main”**, mình sẽ viết câu lệnh sau :


```
HAL_ADC_Start_DMA(&hadcl, (uint32_t *) res_value, 3);
```

Câu lệnh này cho phép kích hoạt bộ ADC1 và sử dụng DMA, lưu giá trị nhận về vào mảng “res_value”, “3” ở đây là độ dài của dữ liệu chuyển đổi ADC nhận về từ ngoại vi đến bộ nhớ.

Lưu ý ép kiểu (uint32_t *) cho biến “res_value” để cùng định dạng dữ liệu với biến cục bộ được khai báo khi khởi tạo hàm HAL_ADC_Start_DMA()

```
volatile uint16_t res_value[3];

int main(void)
{
    /* MCU Configuration-----*/

    /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
    HAL_Init();

    /* Configure the system clock */
    SystemClock_Config();

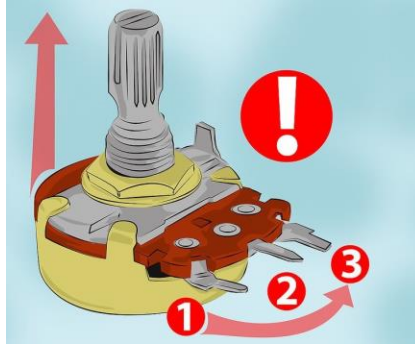
    /* Initialize all configured peripherals */
    MX_GPIO_Init();
    MX_DMA_Init();
    MX_ADC1_Init();

    HAL_ADC_Start_DMA(&hadcl, (uint32_t *) res_value, 3);

    while (1)
    {
    }
    /* USER CODE END 3 */
}
```

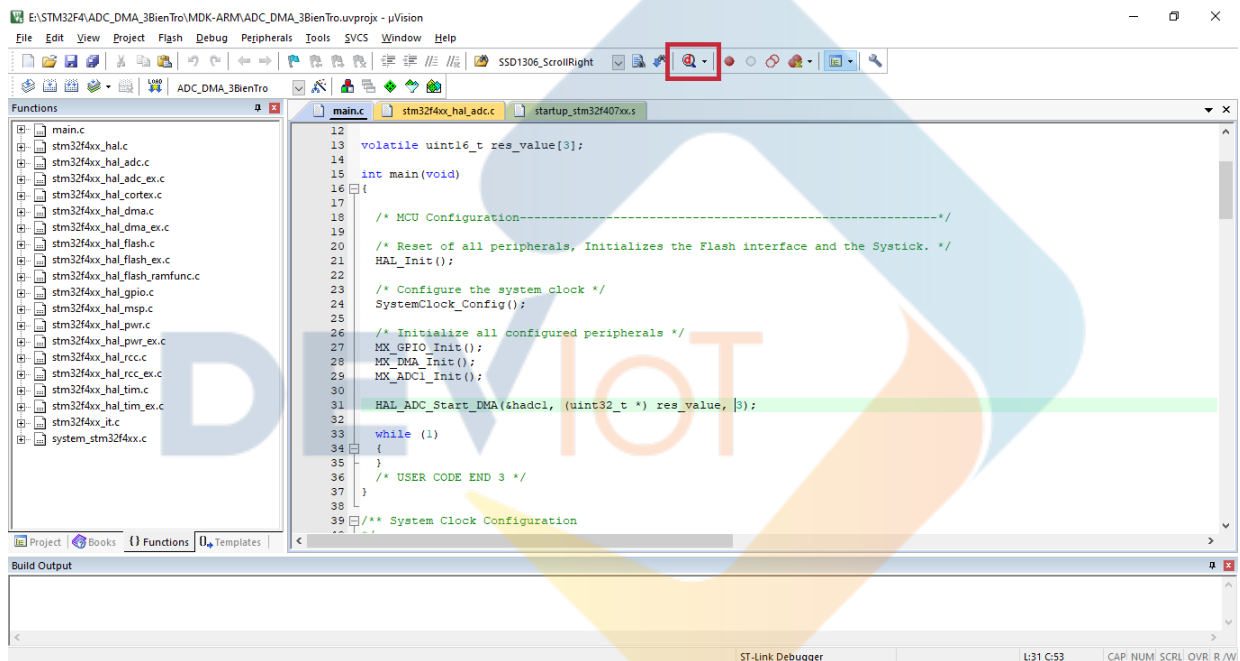
Chúng ta build chương trình (F7) và load code xuống kit (F8), sau đó nhấn nút reset kit

Kết nối các chân 2 của biến trở với các chân PA0, PA1, PA2, các chân số 1 với GND và các chân số 3 với 3V.



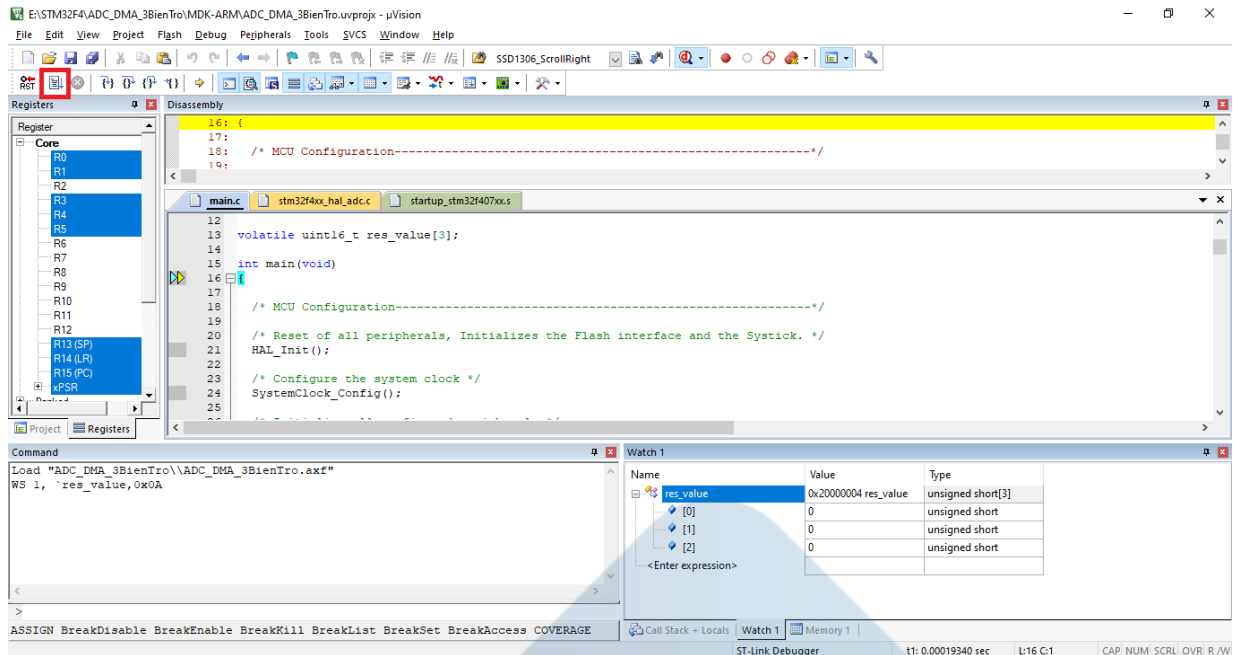
Tiếp theo, chúng ta bật chế độ Debug để quan sát các giá trị nhận về.

Trên thanh công cụ, chúng ta click vào Icon Debug.



Khi cửa sổ Debug hiện ra, các bạn add mảng “res_value” vào cửa sổ Watch1.

Chọn Icon Run để tiến hành chạy Debug.




Điều chỉnh biến trở.

Tại cửa sổ Watch1, chúng ta sẽ quan sát sự thay đổi giá trị của các biến. Muốn xem giá trị các biến ở hệ thập phân, các bạn click chuột phải vào “res_value” tại đây và bỏ chọn “Hexadecimal Display”

Name	Value	Type
res_value	0x20000004 res_value	unsigned short[3]
[0]	4095	unsigned short
[1]	3160	unsigned short
[2]	2618	unsigned short
<Enter expression>		

Trên đây là 1 project nhỏ để các bạn cùng làm quen với quét nhiều kênh ADC trên kit STM32F407VG, sử dụng DMA.

DEVIOT - CÙNG NHAU HỌC LẬP TRÌNH IOT

 Website: deviot.vn

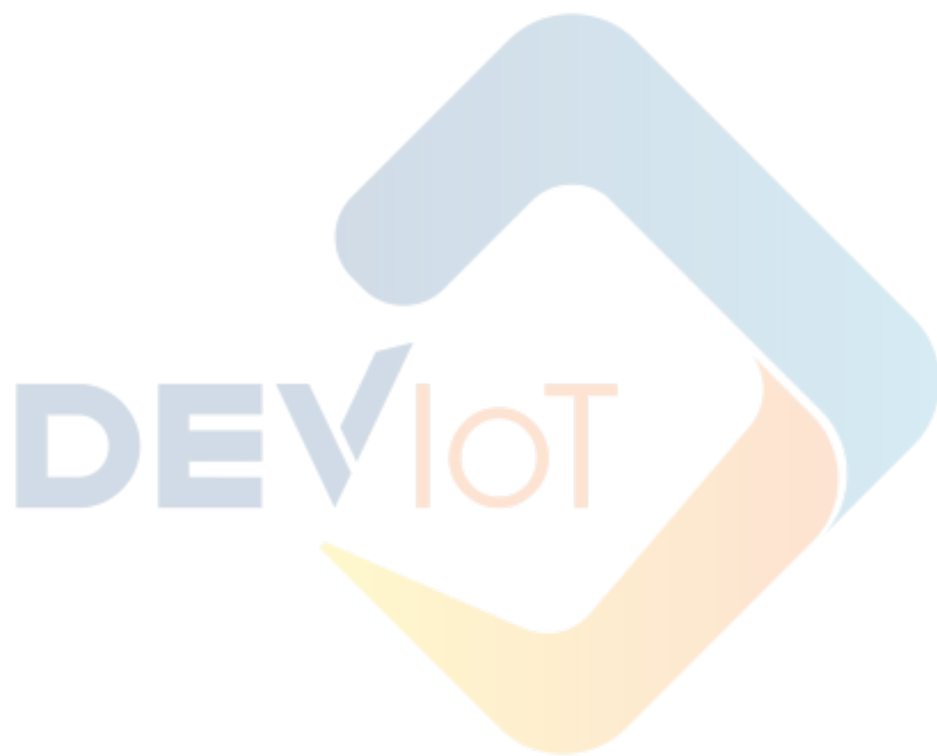
📌 FanPage: Deviot - Thời sự kỹ thuật & IoT

📌 Group: Deviot - Cùng nhau học lập trình IoT

📌 Hotline: 0969.666.522

📌 Address: Số 101C, Xã Đàn 2

📌 Đào tạo thật, học thật, làm thật



deviot.vn