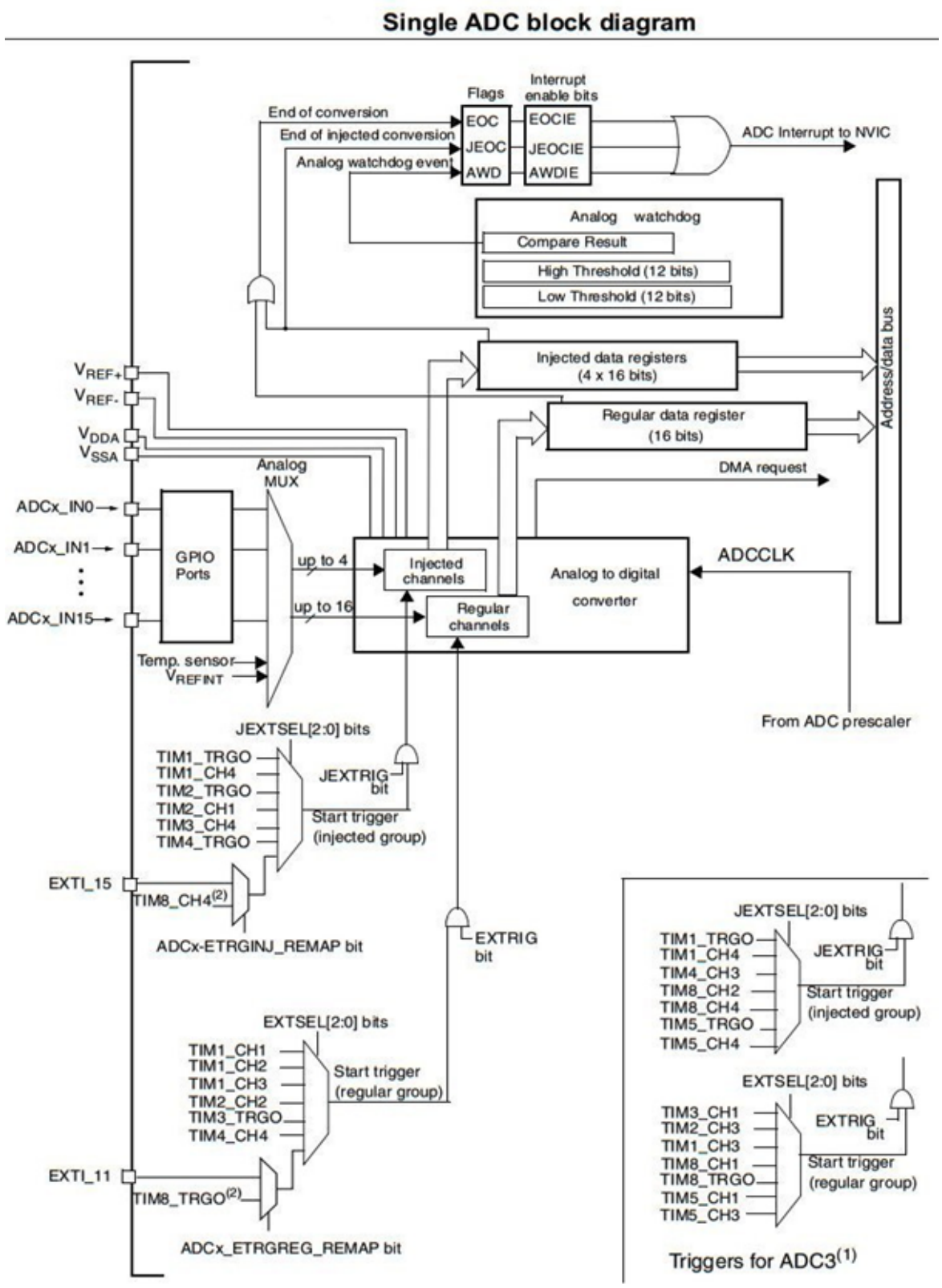
## I. Lý thuyết

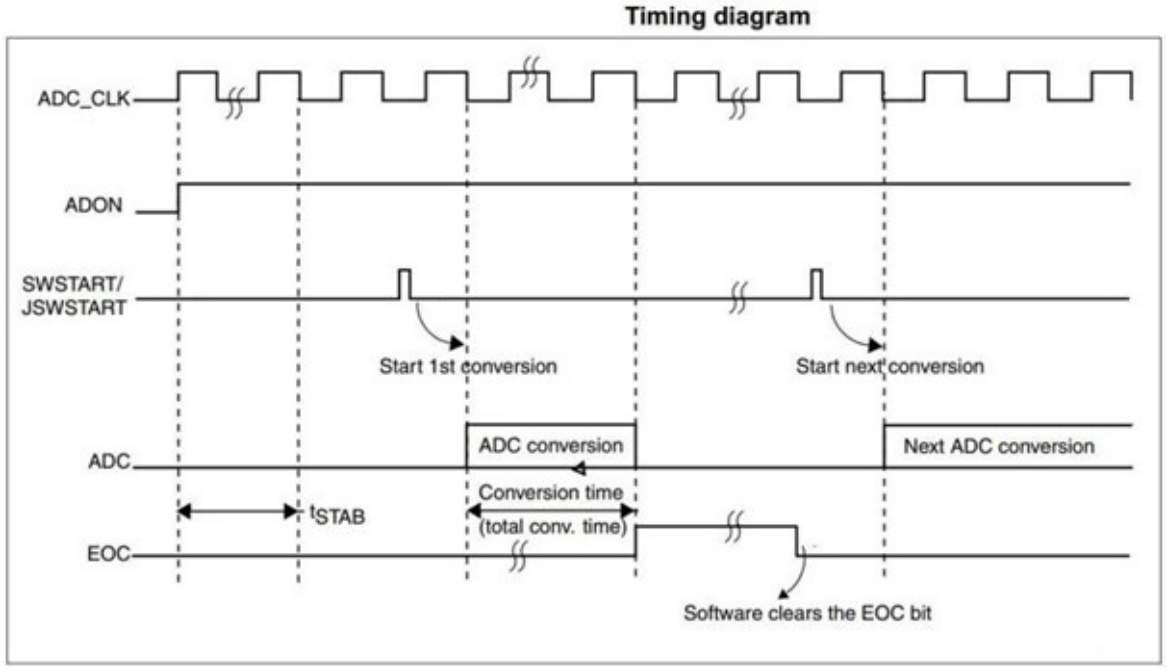
**ADC (Analog-to-Digital Converter)** là 1 mạch điện tử lấy điện áp tương tự làm đầu vào và chuyển đổi nó thành dữ liệu số (1 giá trị đại diện cho mức điện áp trong mã nhị phân). Ảnh dưới mô tả sơ đồ khối của 1 ADC.




**Các khái niệm cần biết:**

**Độ phân giải (resolution):** dùng để chỉ số bit cần thiết để chứa hết các mức giá trị số (digital) sau quá trình chuyển đổi ở ngõ ra. Bộ chuyển đổi ADC của STM32F103C8T6 có độ phân giải mặc định là 12 bit, tức là có thể chuyển đổi ra 212= 4096 giá trị ở ngõ ra số.

**Thời gian lấy mẫu (sampling time):** là khái niệm được dùng để chỉ thời gian giữa 2 lần số hóa của bộ chuyển đổi, thời gian lấy mẫu càng lâu độ chính xác càng cao. Nhìn vào đồ thị dưới ta sẽ thấy ADC cần 1 khoảng thời gian ổn định tSTAB trước khi bắt đầu chuyển đổi. Sau khi chuyển đổi xong cờ EOC sẽ được set, và kết quả được lưu vào thanh ghi. Trước khi bắt đầu quá trình chuyển đổi tiếp theo thì cờ EOC clear.




Để hiểu quá trình số hóa trong STM32 diễn ra như thế nào ta theo dõi ví dụ sau. Giả sử ta cần đo điện áp tối thiểu là 0V và tối đa là 3.3V, trong STM32 sẽ chia 0 → 3.3V thành 4096 khoảng giá trị (từ 0 → 4095, do 212 = 4096), giá trị đo được từ chân IO tương ứng với 0V sẽ là 0, tương ứng với 1.65V là 2047 và tương ứng 3.3V sẽ là 4095.

Các mode hoạt động của ADC:

**Single conversion mode**: Trong chế độ này, ADC sẽ chỉ thực hiện 1 chuyển đổi cho tới khi người dùng cho phép chuyển đổi tiếp.

**Continuous Conversion Mode**: ở chế độ này, ADC sẽ ngay lập tức thực hiển 1 chuyển đổi khác khi chuyển đổi trước vừa kết thúc.

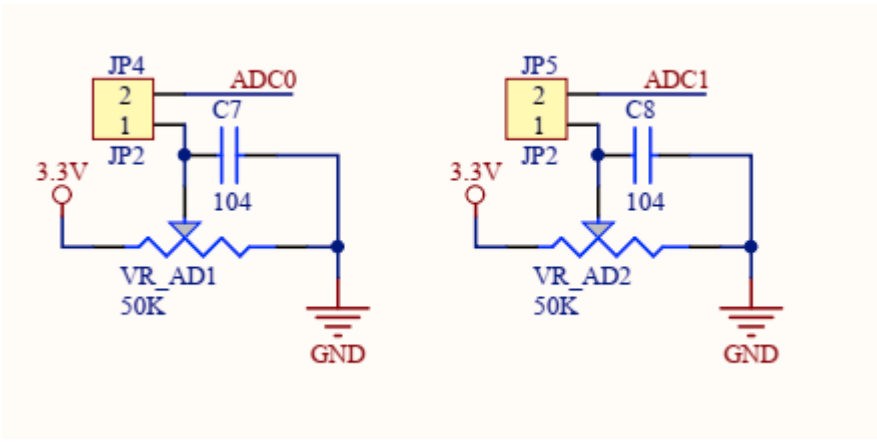
**Scan Mode**: Chế độ này được sử dụng để quét 1 nhóm các kênh. 1 chuyển đổi duy nhất được thực hiện cho mỗi kênh. Sau khi 1 kênh chuyển đổi xong, kênh tiếp theo sẽ tự động chuyển đổi.

**Discontinuous Mode**: Chế độ này được sử dụng để chuyển đổi n lần (n <= 8). Giá trị của n được xác định tại bit DISCNUM[2:0] trong thanh ghi ADC_CR1.

**II. Lập trình ADC trên STM32:**

Ở ví dụ này ta sẽ Dùng bộ ADC để đọc giá trị của biến trở thông qua điện áp. Sử dụng 2 chế độ continuos mode và scan mode.

Đấu nối mạch theo sơ đồ: (với ADC 0 là PA0, ADC1 là PA1)



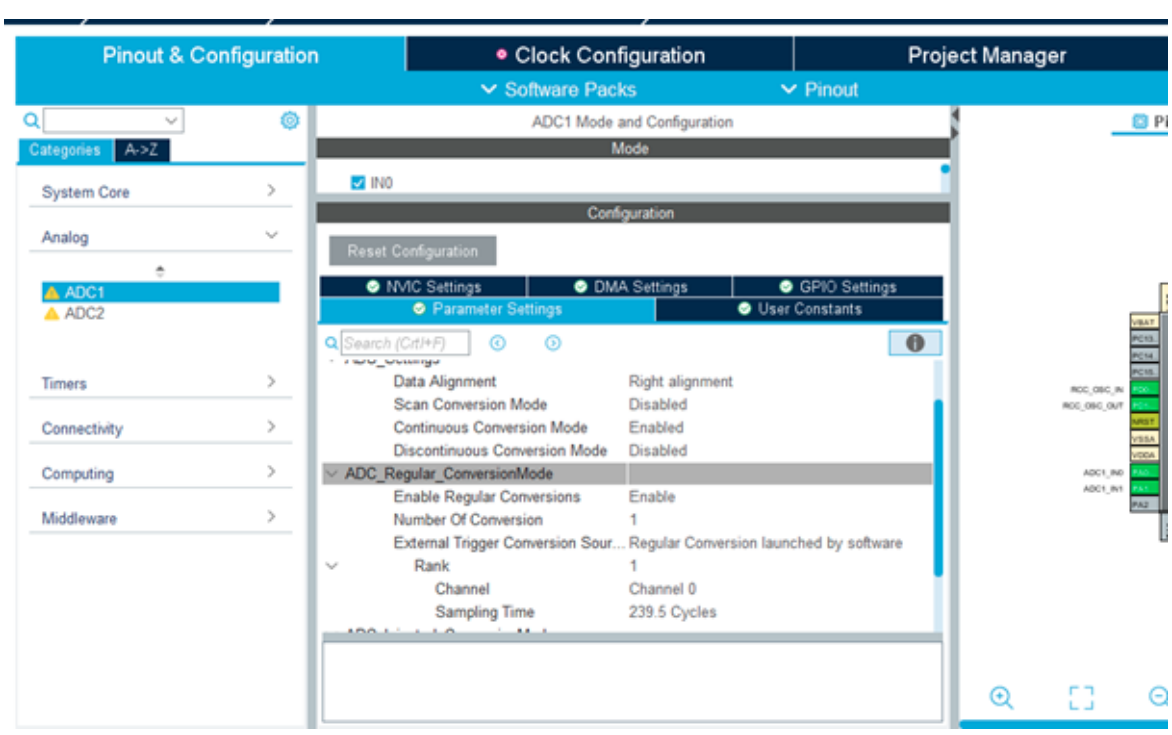

**CONTINOUS MODE:**

**1. Khởi tạo project với CubeMx**

Bước 1: Bước đầu tiên ta sẽ làm tương tự như bài 1 (cấu hình thạch anh, debug, …)

Bước 2: Cấu hình ADC: (đối với continous mode).

Ở đây ta chỉ sử dụng ADC1 chennel 0 (PA0).




Enable continous mode (biến đổi liên tục).
Chọn Sampling Time = 239.5 (để tối ưu độ chính xác).
Kích hoạt ngắt trong NVIC settings. Chọn mức ưu tiên = 2.

Bước 3: Đặt tên project và GENERATE CODE.

## 2. Giải thích 1 số hàm quan trọng

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
```

Sau khi ADC chuyển đổi xong giá trị thì chương trình sẽ thực thi các lệnh trong câu lệnh này.
Tham số:
hadc: trỏ tới adc được kích hoạt ngắt (&hadc1 hoặc &hadc2).

```
HAL_StatusTypeDef HAL_ADC_Start_IT(ADC_HandleTypeDef* hadc)
```

Kích hoạt ADC ở chế độ ngắt.
Tham số:
hadc: trỏ tới adc được start ở chế độ ngắt

```
HAL_StatusTypeDef HAL_ADC_Start(ADC_HandleTypeDef* hadc)
```

Kích hoạt ADC ở chế độ normal
Tham số:
hadc: trỏ adc được start ở chế đọ normal.

```
HAL_StatusTypeDef HAL_ADC_Stop(ADC_HandleTypeDef* hadc)
```

Stop ADC ở chế độ normal
Tham số:
hadc: trỏ tới adc được stop.

```
uint32_t HAL_ADC_GetValue(ADC_HandleTypeDef* hadc)
```

Đọc giá trị ADC, giá trị trả về chính là kết quả sau khi biến đổi xong.
Tham số:
hadc: trỏ tới adc được lấy giá trị.

```
HAL_StatusTypeDef HAL_ADC_PollForConversion(ADC_HandleTypeDef* hadc,
uint32_t Timeout)
```

Chuyển đổi ADC.
Tham số:
hadc: trỏ tới adc chờ chuyển đổi.
Timeout: thời gian chuyển đổi tối đa.

## 3. Lập trình trên Keil C

B1: khai báo giá trị ADC trả về:

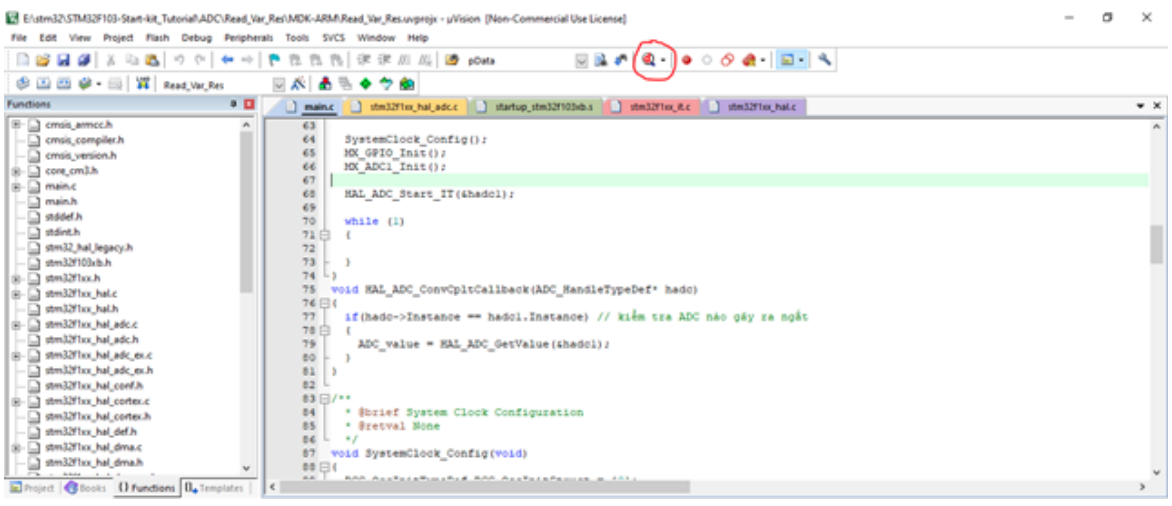
B2: kích hoạt ADC ta sẽ thực hành trên cả 2 chế độ: Normal, ngắt.
Normal: Đọc giá trị ADC với chu kỳ 100ms.

```
int main(void)
{
 HAL_Init();
 SystemClock_Config();
 MX_GPIO_Init();
 MX_ADC1_Init();
 HAL_ADC_Start(&hadc1);
 while (1)
 {
   ADC_value = HAL_ADC_GetValue(&hadc1);
   HAL_Delay(100);
 }
}
```

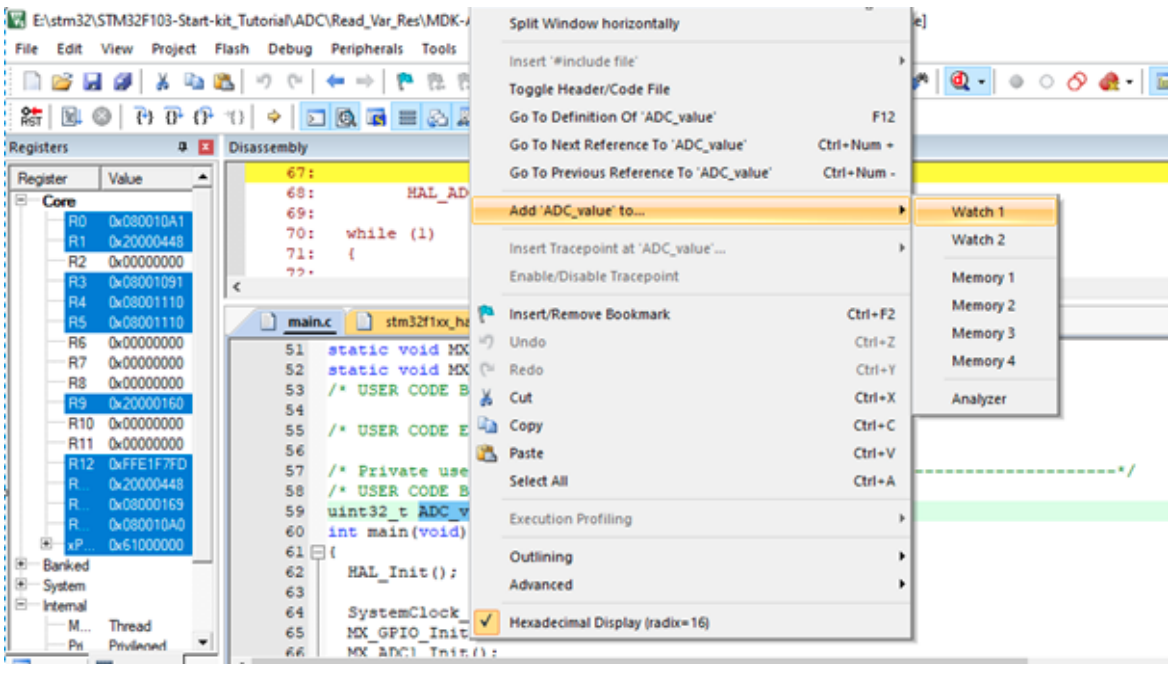
Ngắt: Sau khi chuyển đổi xong, chương trình sẽ thực hiện các câu lệnh trong hàm **HAL_ADC_ConvCpltcallback().**

```
uint32_t ADC_value;
int main(void)
{
 HAL_Init();
 SystemClock_Config();
 MX_GPIO_Init();
 MX_ADC1_Init();
 HAL_ADC_Start_IT(&hadc1);
 while (1)
 {
 }
}
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
{
 if(hadc->Instance == hadc1.Instance) // kiểm tra ADC nào gây ra ngắt
 {
   ADC_value = HAL_ADC_GetValue(&hadc1);
 }
}
```

B3: bật chế độ Debug để theo dõi giá trị ADC trả về:
Click chọn vào biểu tượng Debug session.



Add ADC_value vào watch 1 để theo dõi:




Sau đó sẽ click chọn Run để theo dõi Debug.

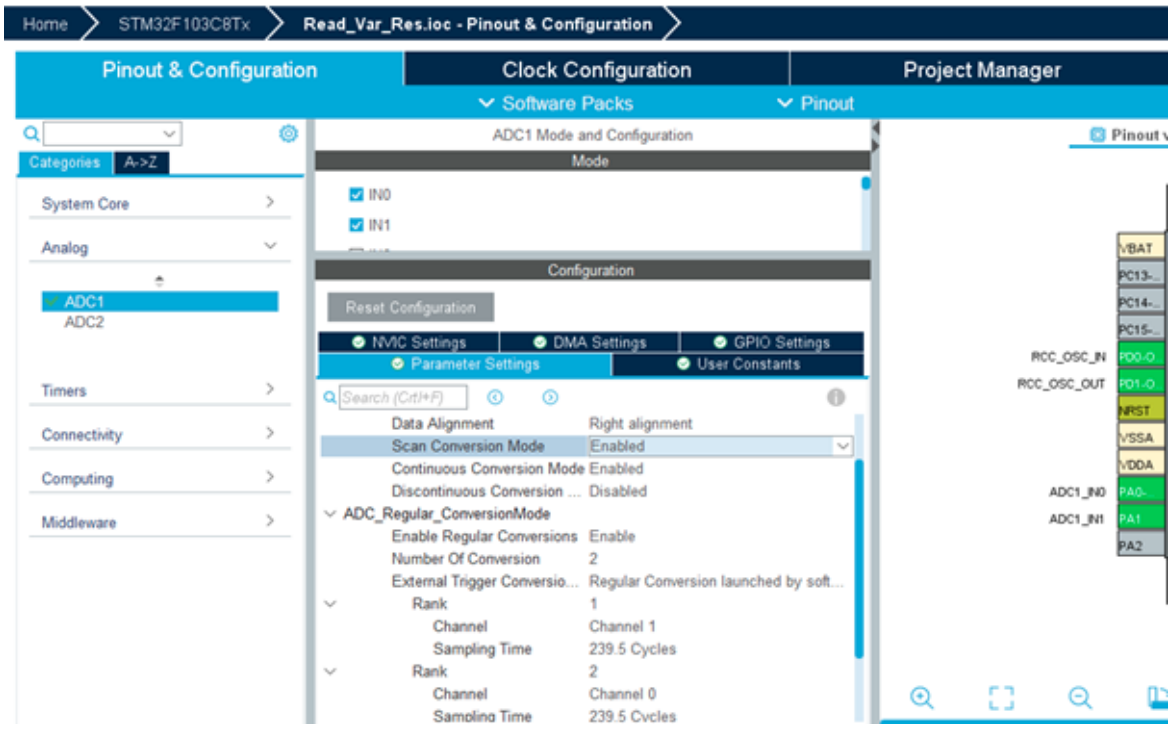
**SCAN MODE:**

## 1. Khởi tạo project với CubeMx

B1: Cấu hình lại ADC

Ta dùng 2 kênh ADC nên là chọn number of conversion = 2

Enable scan mode và continuous mode, đồng thời chọn thời gian chuyển đổi.




B2: Gen code.

## 2. Lập trình trên Keil C

B1: Tách phần init 2 kênh ADC trong hàm MX_ADC1_Init() thành 2 hàm riêng biệt.

```
// khởi tạo ADC1 channel 0
void ADC_Select_CH0 (void)
{
 ADC_ChannelConfTypeDef sConfig = {0};
 sConfig.Channel = ADC_CHANNEL_0;
 sConfig.Rank = ADC_REGULAR_RANK_1;
 sConfig.SamplingTime = ADC_SAMPLETIME_239CYCLES_5;
 if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
 {
   Error_Handler();
 }
}
// khởi tạo ADC1 channel 1
void ADC_Select_CH1 (void)
{
 ADC_ChannelConfTypeDef sConfig = {0};
 sConfig.Channel = ADC_CHANNEL_1;
 sConfig.Rank = ADC_REGULAR_RANK_1;
 sConfig.SamplingTime = ADC_SAMPLETIME_239CYCLES_5;
 if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
 {
   Error_Handler();
 }
}
```

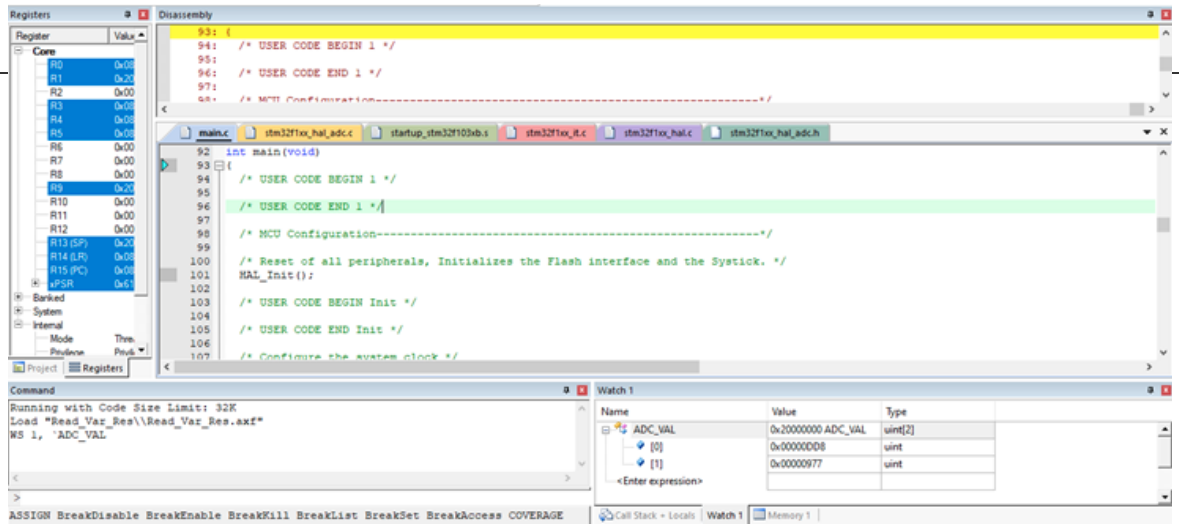
B2: Thực hiện việc đọc giá trị lần lượt ở 2 kênh ADC.

```c
uint32_t ADC_VAL[2];
int main(void)
{
 HAL_Init();
 SystemClock_Config();

 MX_GPIO_Init();
 MX_ADC1_Init();

 while (1)
 {
   ADC_Select_CH0();
   HAL_ADC_Start(&hadc1);
   HAL_ADC_PollForConversion(&hadc1, 1000);
   ADC_VAL[0] = HAL_ADC_GetValue(&hadc1);
   HAL_ADC_Stop(&hadc1);
   HAL_Delay(10);
   ADC_Select_CH1();
   HAL_ADC_Start(&hadc1);
   HAL_ADC_PollForConversion(&hadc1, 1000);
   ADC_VAL[1] = HAL_ADC_GetValue(&hadc1);
   HAL_ADC_Stop(&hadc1);
   HAL_Delay(10);
 }
}
```

B3: nạp chương trình và chạy ở chế độ debug để theo dõi giá trị ADC đọc về




**1 số lưu ý:**

Ở đây giá trị ta đọc được chỉ là giá trị của điện áp trả về. Muốn biết được giá trị điện trở ta phải xem mạch nguyên lý.

Cụ thể ở đây : Từ giá trị đo được ta sẽ tính điện trở theo công thức:

$$R = \frac{ADC\_Value}{2^{12} - 1} * 50\ (kOhm)$$