Chương 5: Ghép nối với thế giới thực

5.1 Giới thiệu

5.2 ADC/DAC

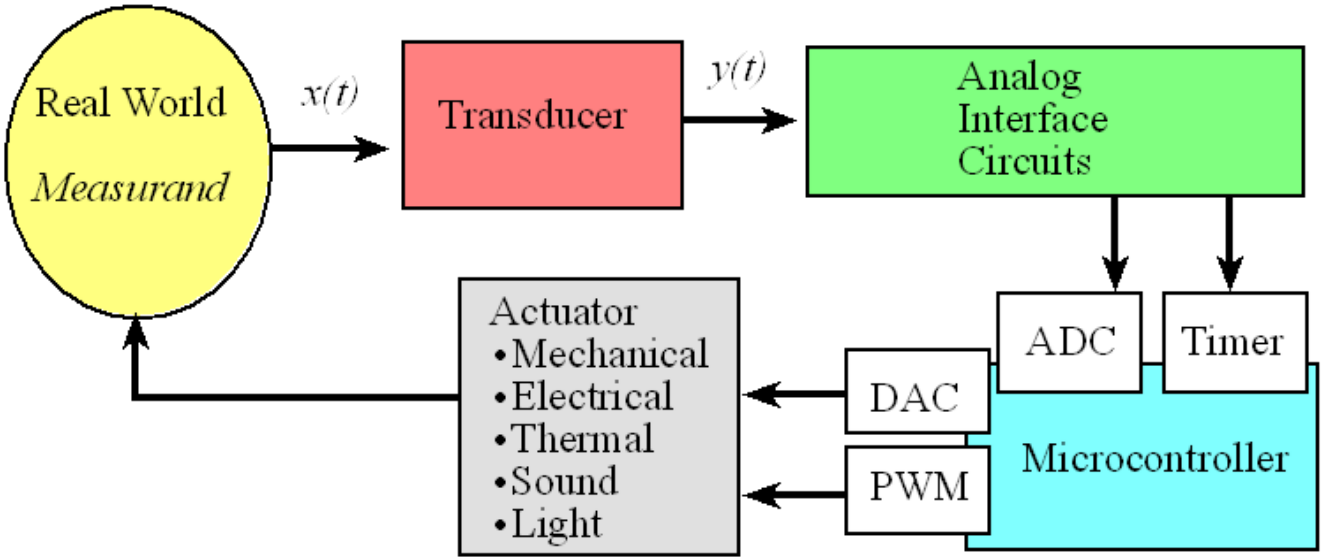
5.3 Ghép nối với ADC

5.4 Ghép nối thiết bị công suất cao

5.5 Vào ra hiệu quả với ngắt và với DMA

Giới thiệu

- Hệ nhúng là một thành phần quan trọng của hệ thống đo lường, điều khiển số
  - Đầu vào: các thông tin về đối tượng (nhiệt độ, độ ẩm, áp suất, độ pH, độ mặn,…) tùy thuộc bài toán.
  - Đầu ra: cơ cấu chấp hành tác động đến đối tượng.



©Jonathan Valvano and Ramesh Yerraballi

Giới thiệu

- Ví dụ: máy ấp trứng tự động
- Các tính năng
  - Đo nhiệt độ, độ ẩm
  - Bật/tắt đèn sưởi, quạt, phun sương
  - Tự động duy trì nhiệt độ, độ ẩm
  - Tự đảo trứng
- Nguyên tắc làm việc?

Cảm biến

- Là thiết bị phát hiện sự thay đổi của đại lượng vật lý, rồi gửi thay đổi đó về máy tính dưới dạng tín hiệu đọc được
- Thiết bị biến đổi đại lượng vật lý cần đo thành tín hiệu điện, cho phép máy tính đo lường đại lượng đó

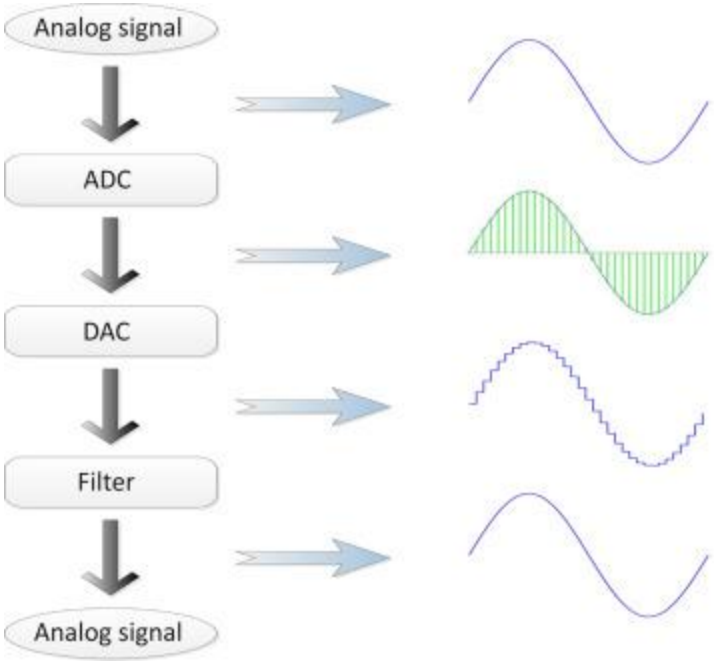
Cơ cấu chấp hành

- Là thiết bị nhận lệnh điều khiển và năng lượng đầu vào để biến thành chuyển động phù hợp của một hệ thống điện-cơ.
- Mở rộng: là các thiết bị nhận điều khiển từ hệ trung tâm để tác động đến môi trường vật lý.
- Thường hoạt động với công suất lớn.
- Ví dụ:
  - Động cơ
  - Van
  - …

ADC/DAC

- ADC: Analog to Digital Converter
  - Là thiết bị chuyển đổi đổi tín hiệu tương tự (điện áp) thành chuỗi giá trị số tương ứng theo thời gian (tín hiệu số).
- DAC:
  - Thiết bị chuyển đổi đổi tín hiệu số thành tín hiệu tương tự (điện áp).

$$U(t) \leftrightarrow X(\mathrm{k})$$



ADC/DAC

- Công thức chuyển đổi

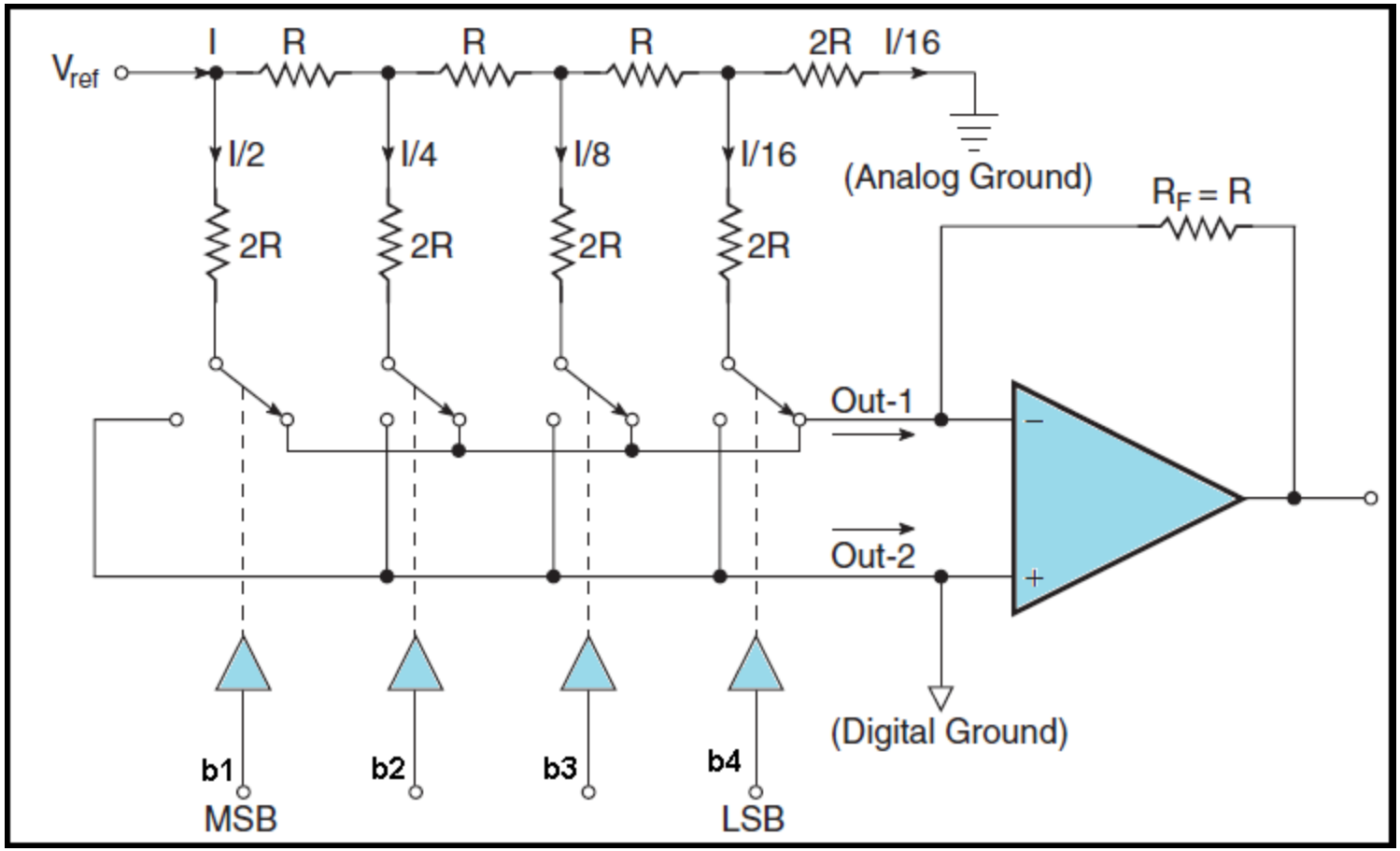
$$U = U_{ref} * \frac{x_{n\_1}x_{n\_2}\ldots x_1x_0}{2^n}$$

U_{ref}: điện áp tham chiếu, là hằng số của mạch ADC/DAC

- ADC: có U_{in}, cần tìm X tương ứng.
- DAC: có X, cần tạo ra điện áp U_{out} tương ứng.

ADC/DAC

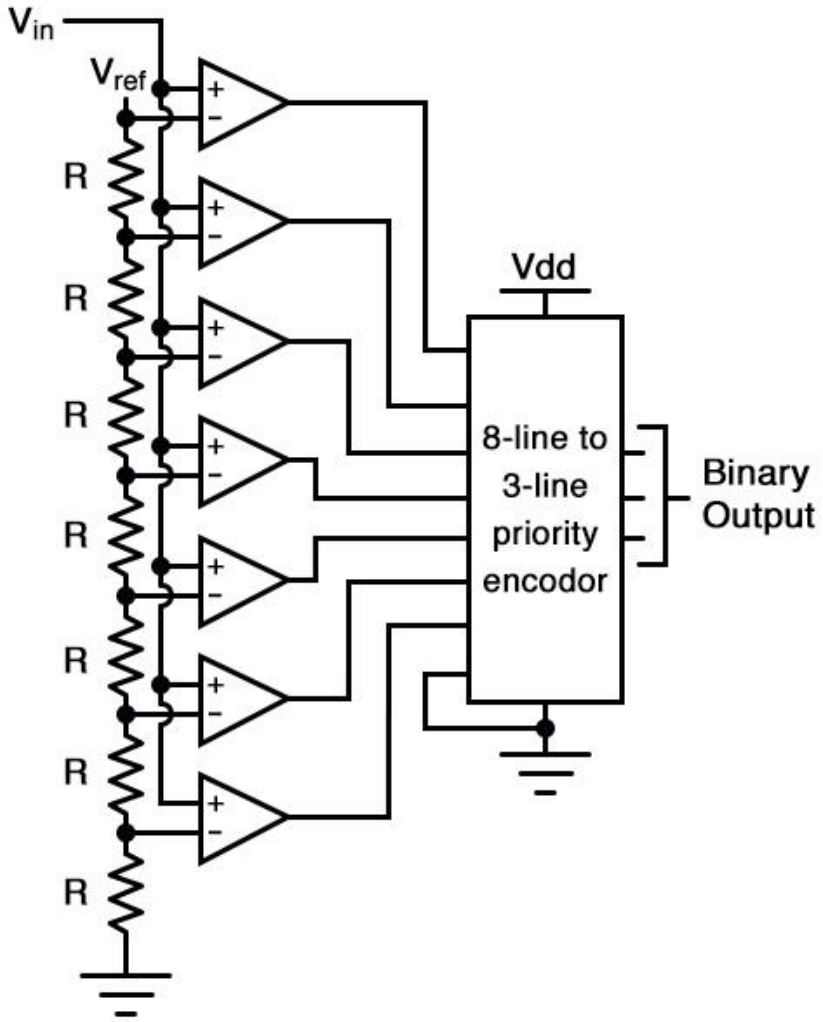
- R-2R ladder DAC



Find Vout value?

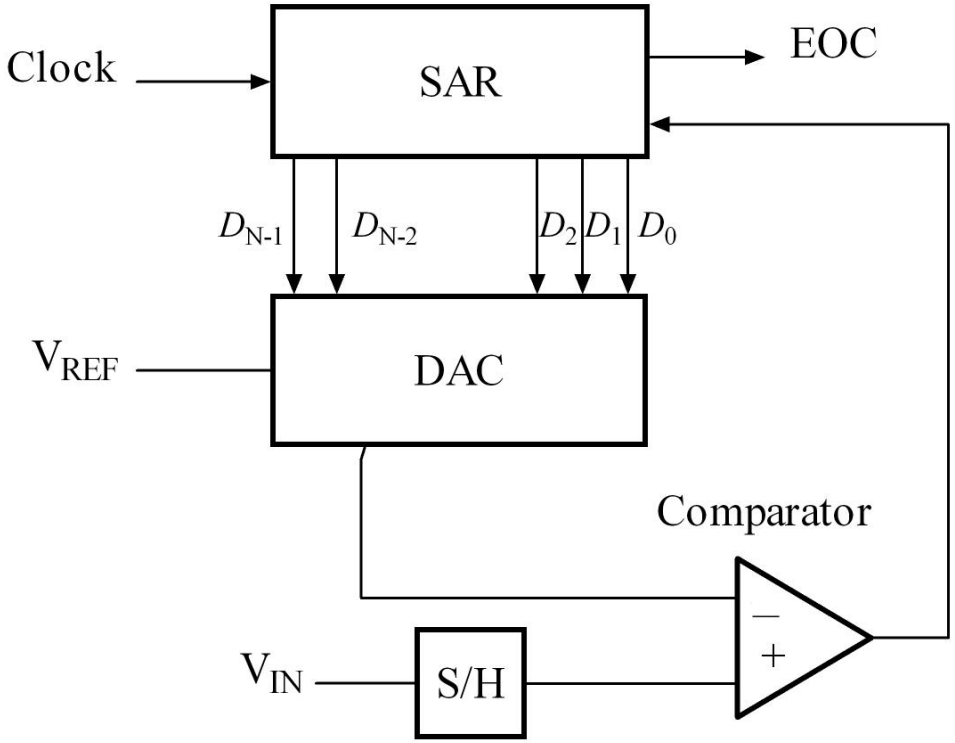
ADC/DAC

- Flash ADC

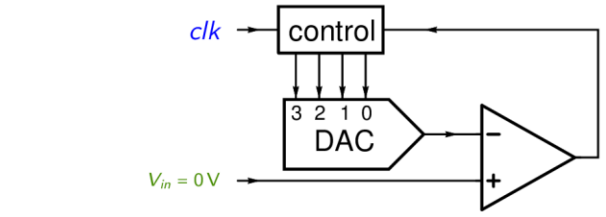


ADC/DAC

- ADC xấp xỉ liên tiếp



Successive Approximation – example of a 4-bit ADC

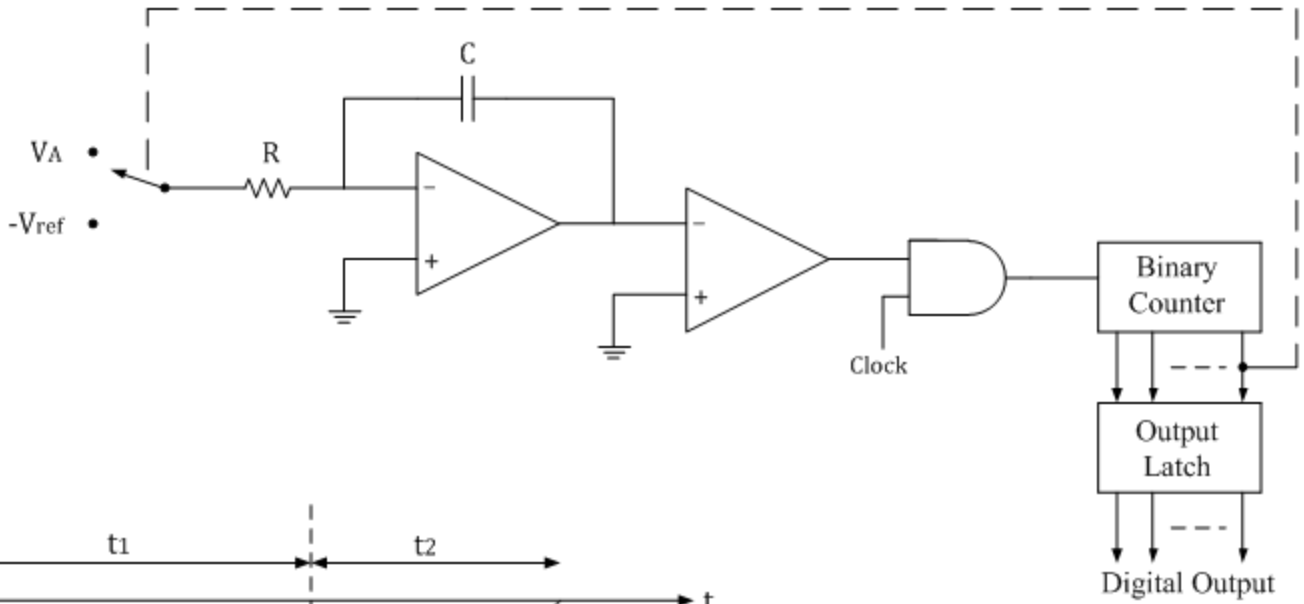


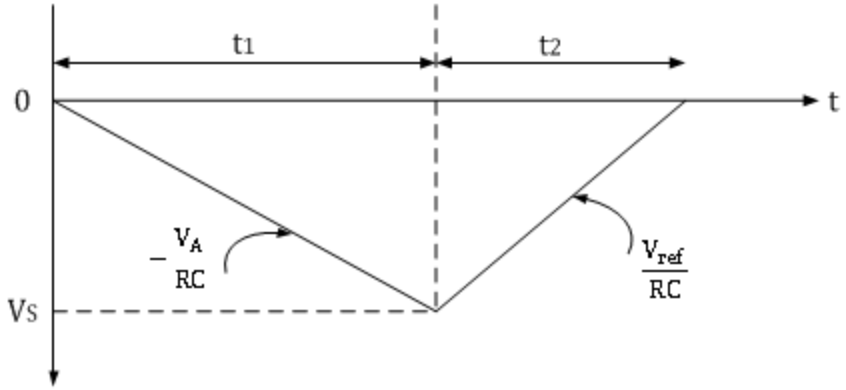
Resolution:

5 V × 1/2	2.5000 V
5 V × 1/4	1.2500 V
5 V × 1/8	0.6250 V
5 V × 1/16	0.3125 V
...	
5 V × 1/1024	0.0049 V

ADC/DAC

- ADC tích phân 2 sườn dốc

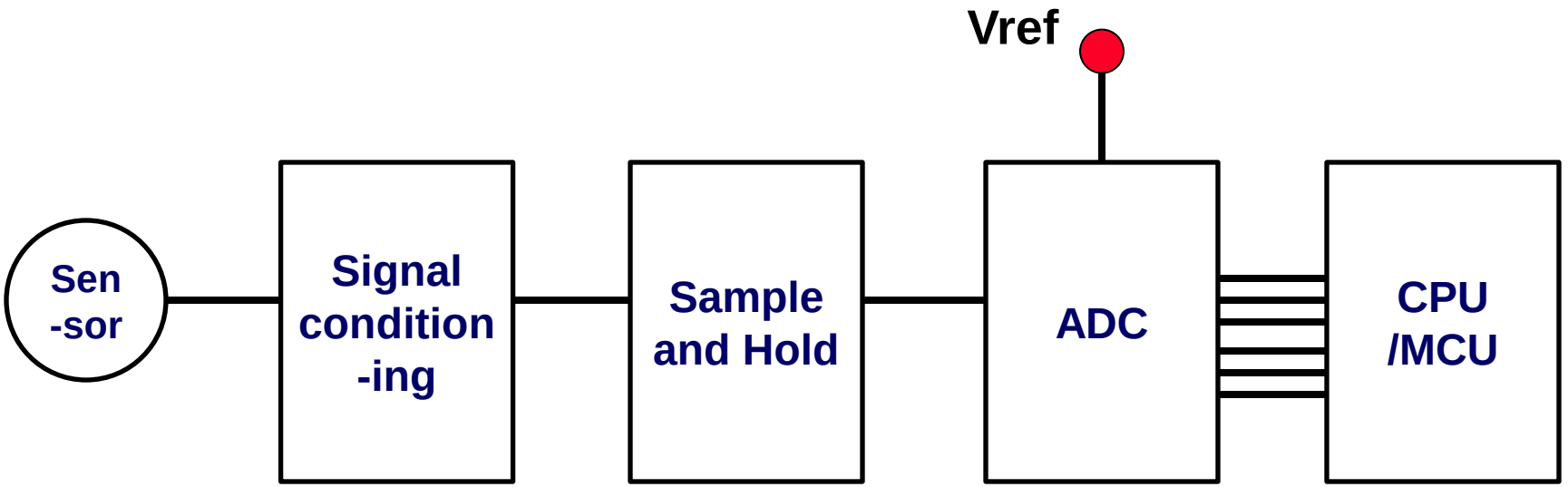




$V_A = -V_{ref} \times t1/t2$

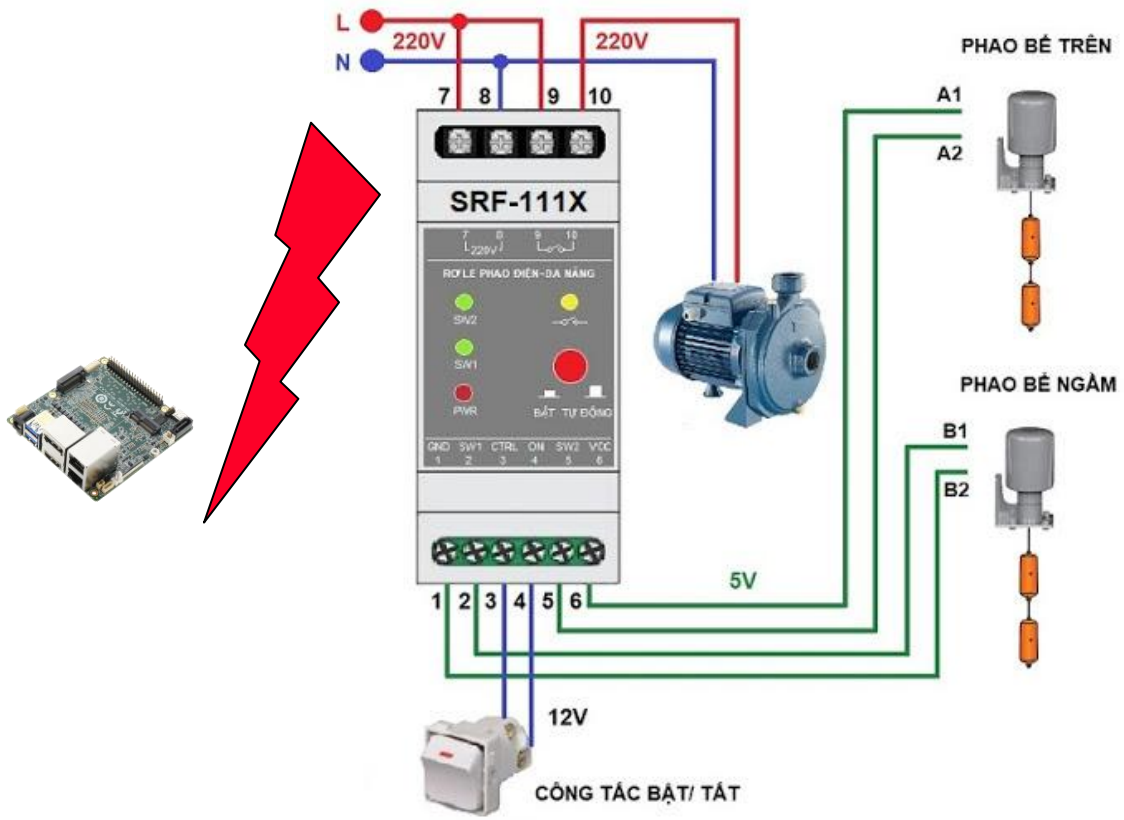
Ghép nối ADC với hệ vi xử lý

- Sơ đồ khối ghép nối ADC



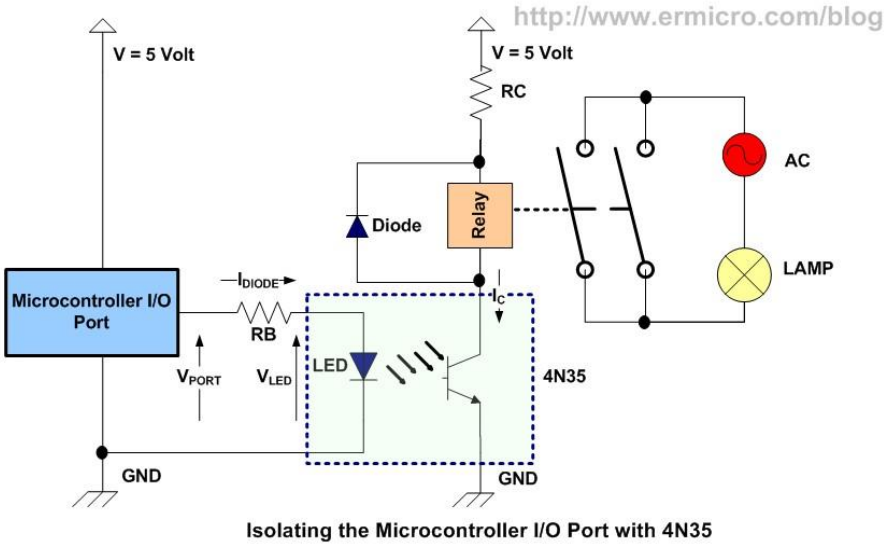
Ghép nối thiết bị công suất cao

- Nhiễu nguồn từ tải có thể làm MCU treo
- Khi có sự cố điện áp cao từ tải có thể "xông" sang mạch điều khiển → nguy hiểm

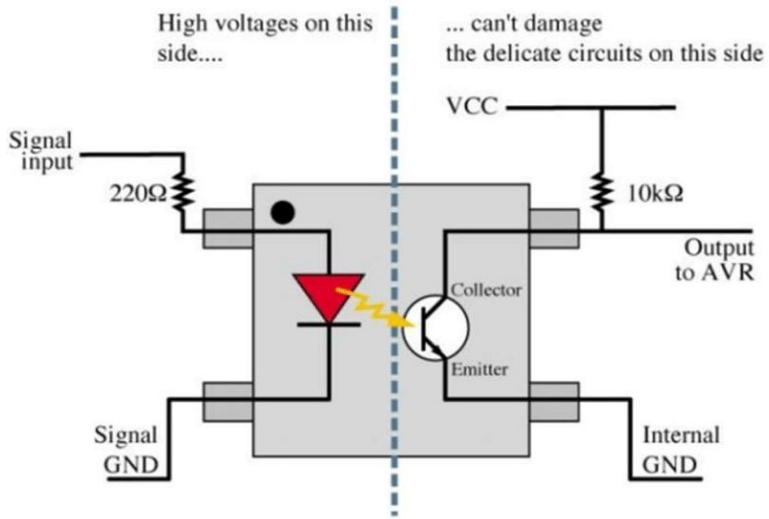


Ghép nối thiết bị công suất cao

- Cách ly: relay + optocoupler
  - Relay: đóng cắt mạch điện áp cao từ mạch điều khiển điện áp thấp
  - Optocoupler: cách ly quang chống nhiễu cho mạch điều khiển. Có thể dùng cả input và output



Isolating the Microcontroller I/O Port with 4N35



Lập trình ghép nối ADC

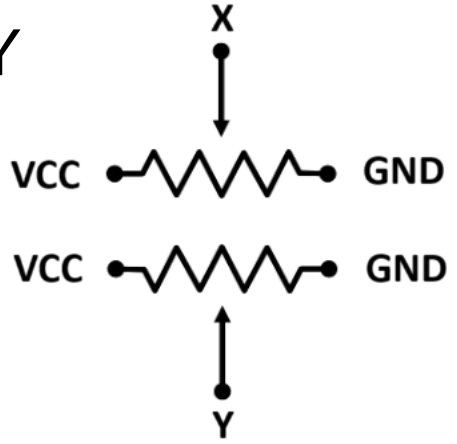
HAL_ADC_Init()
HAL_ADC_DeInit()
HAL_ADC_RegisterCallback()
HAL_ADC_UnRegisterCallback()
HAL_ADC_MspInit()
HAL_ADC_MspDeInit()

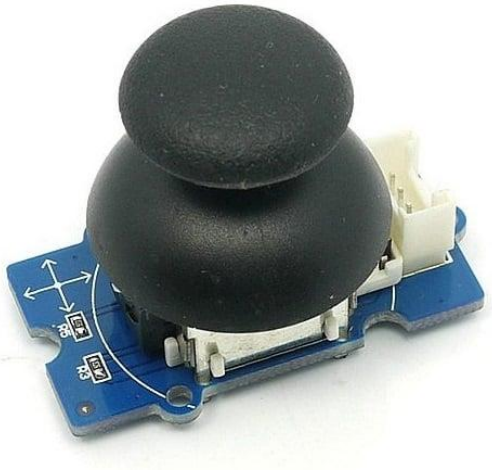
HAL_ADC_ErrorCallback()
HAL_ADC_ConfigChannel()
HAL_ADC_AnalogWDGConfig()
HAL_ADC_GetState()
HAL_ADC_GetError()

HAL_ADC_Start()
HAL_ADC_Stop()
HAL_ADC_PollForConversion()
HAL_ADC_PollForEvent()
HAL_ADC_Start_IT()
HAL_ADC_Stop_IT()
HAL_ADC_IRQHandler()
HAL_ADC_Start_DMA()
HAL_ADC_Stop_DMA()
HAL_ADC_GetValue()
HAL_ADC_ConvCpltCallback()
HAL_ADC_ConvHalfCpltCallback()
HAL_ADC_LevelOutOfWindowCallback()

Bài tập: Ghép nối Groove joystick

- 2 chân X, Y nối với đầu ra biến trở
- Vị trí joystick tỷ lệ với điện áp tại X và Y
  - Dùng 2 chân ghép nối ADC





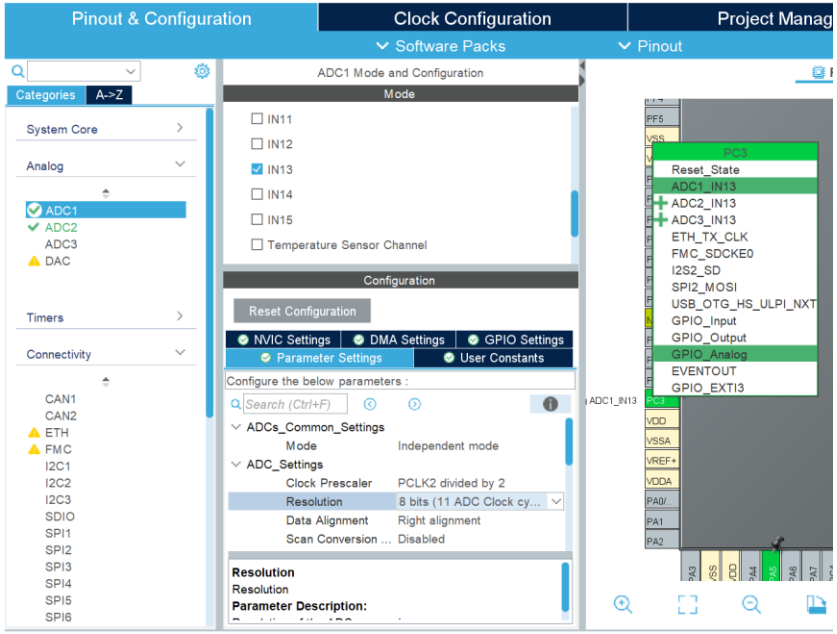
Bài tập: Ghép nối Groove joystick

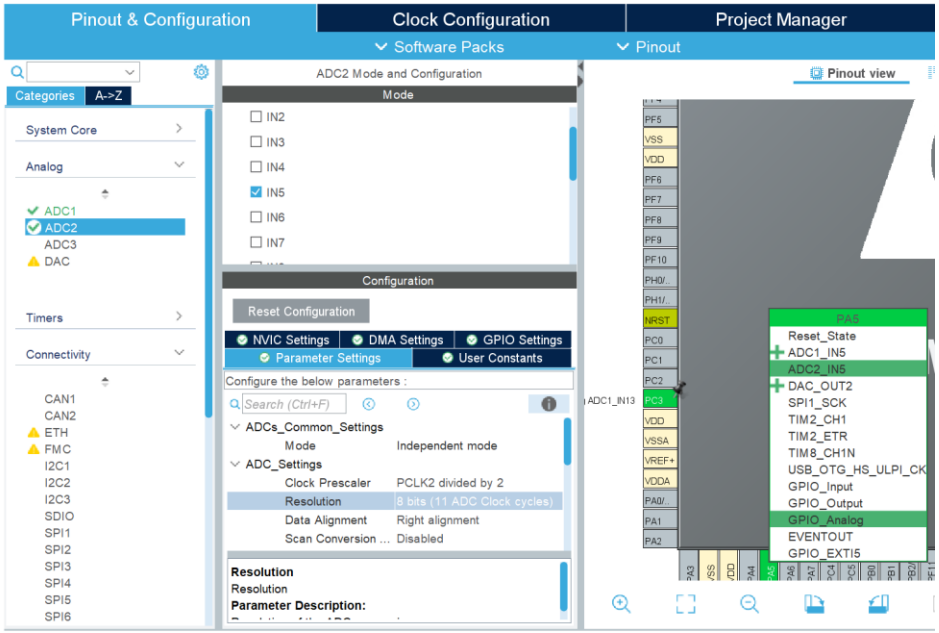
- Lắp mạch theo sơ đồ

Joystick	STM32
VCC	3V
GND	GND
X	PC3 (ADC1 IN13)
Y	PA5 (ADC2 IN5)

Bài tập: Ghép nối Groove joystick

- Tạo project và cấu hình IOC
  - System clock 180 MHz
  - USART1 để kết nối PC
  - PC3: ADC IN13
  - PA5: ADC IN5





Bài tập: Ghép nối Groove joystick

- Lập trình đọc giá trị từ joystick
- Hoàn thiện đoạn code sau trong hàm main() để gửi dữ liệu về PC.

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    HAL_ADC_Start(&hadc1);     //start ADC1 channel 13 (PC3)
    HAL_ADC_Start(&hadc2);     //start ADC2 channel  5 (PA5)
    HAL_ADC_PollForConversion(&hadc1, 10);
    HAL_ADC_PollForConversion(&hadc2, 10);
    int JoystickX = HAL_ADC_GetValue(&hadc1); //wait for conversion
    int JoystickY = HAL_ADC_GetValue(&hadc2); //wait for conversion
    //to-do: send this to PC or use these values to control another device
    char s[20];
    sprintf(s, "%3d-%3d\n", JoystickX, JoystickY);
    HAL_UART_Transmit(&huart1, s, strlen(s), 10);
    HAL_Delay(50);
  /* USER CODE END WHILE */
```

Bài tập: Ghép nối Groove joystick

- Kết nối thêm màn hình OLED1106 vào CN2
- Lập trình để dùng joystick điều khiển một quả bóng hình tròn chạy trên màn hình theo 2 trục của joystick.

Vào ra với ngắt và với DMA

- Ở bài tập trên: vào ra bằng polling,
  - Các hàm HAL_ADC_Start(), HAL_ADC_PollForConversion() đều là blocking.
  - Tốn tài nguyên của CPU, không hiệu quả khi CPU nhanh.
- Vào ra hiệu quả hơn: sử dụng ngắt hoặc DMA.

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    HAL_ADC_Start(&hadc1);     //start ADC1 channel 13 (PC3)
    HAL_ADC_Start(&hadc2);     //start ADC2 channel  5 (PA5)
    HAL_ADC_PollForConversion(&hadc1, 10);
    HAL_ADC_PollForConversion(&hadc2, 10);
    int JoystickX = HAL_ADC_GetValue(&hadc1); //wait for conversion
    int JoystickY = HAL_ADC_GetValue(&hadc2); //wait for conversion
    //to-do: send this to PC or use these values to control another device
    char s[20];
    sprintf(s, "%3d-%3d\n", JoystickX, JoystickY);
    HAL_UART_Transmit(&huart1, s, strlen(s), 10);
    HAL_Delay(50);
  /* USER CODE END WHILE */
```

Vào ra bằng ngắt

- Cần cho phép ngắt của ADC kích hoạt sau mỗi lần chuyển đổi → có thể implement hàm callback tự động gọi.
- Khởi tạo ADC conversion bằng hàm HAL_ADC_Start_IT().
- Không cần chờ ADC thực hiện conversion.
- Khi ADC thực hiện conversion xong, ngắt và hàm callback sẽ được gọi. Thường chỉ cần implement logic trong hàm callback.
  - Cần tạo buffer chứa dữ liệu, và logic xử lý dữ liệu.

Vào ra bằng ngắt

- VD: sử dụng ADC1 ở chế độ interrupt
  - Bật interrupt tab NVIC của ADC1 settings.
  - Khai báo biến adc1_buf và adc1_count trong main.c
  - Gọi hàm HAL_ADC_Start_IT() trước main loop.

```
33 /* Private define ----------------
34 /* USER CODE BEGIN PD */
35 #define ADC_BUF_LEN 4096
36 /* USER CODE END PD */
37
38 /* Private macro -----------------
39 /* USER CODE BEGIN PM */
40
41 /* USER CODE END PM */
42
43 /* Private variables -------------
44 ADC_HandleTypeDef hadc1;
45 ADC_HandleTypeDef hadc2;
46
47 UART_HandleTypeDef huart1;
48
49 /* USER CODE BEGIN PV */
50 uint8_t adc1_buf[ADC_BUF_LEN];
51 int adc1_count;
52 /* USER CODE END PV */
```

```
96   /* Initialize all configured peripherals */
97   MX_GPIO_Init();
98   MX_ADC1_Init();
99   MX_ADC2_Init();
100  MX_USART1_UART_Init();
101  /* USER CODE BEGIN 2 */
102  adc1_count = 0;
103  HAL_ADC_Start_IT(&hadc1);
104  /* USER CODE END 2 */
```

Vào ra bằng ngắt

- VD: sử dụng ADC1 ở chế độ interrupt
  - Bật interrupt tab NVIC của ADC1 settings.
  - Khai báo biến adc1_buf và adc1_count trong main.c
  - Gọi hàm HAL_ADC_Start_IT() trước main loop.
  - Khai báo hàm callback **`HAL_ADC_ConvCpltCallback()`**
  - Xử lý dữ liệu trong main loop (tính trung bình 1000 ADC reading rồi gửi về PC).

```
/* USER CODE BEGIN 4 */
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc) {
  if (hadc == &hadc1)
    adc1_buf[adc1_count] = HAL_ADC_GetValue(&hadc1);

  adc1_count++;
  if (adc1_count < 1000)
      HAL_ADC_Start_IT(&hadc1);
}
/* USER CODE END 4 */
```

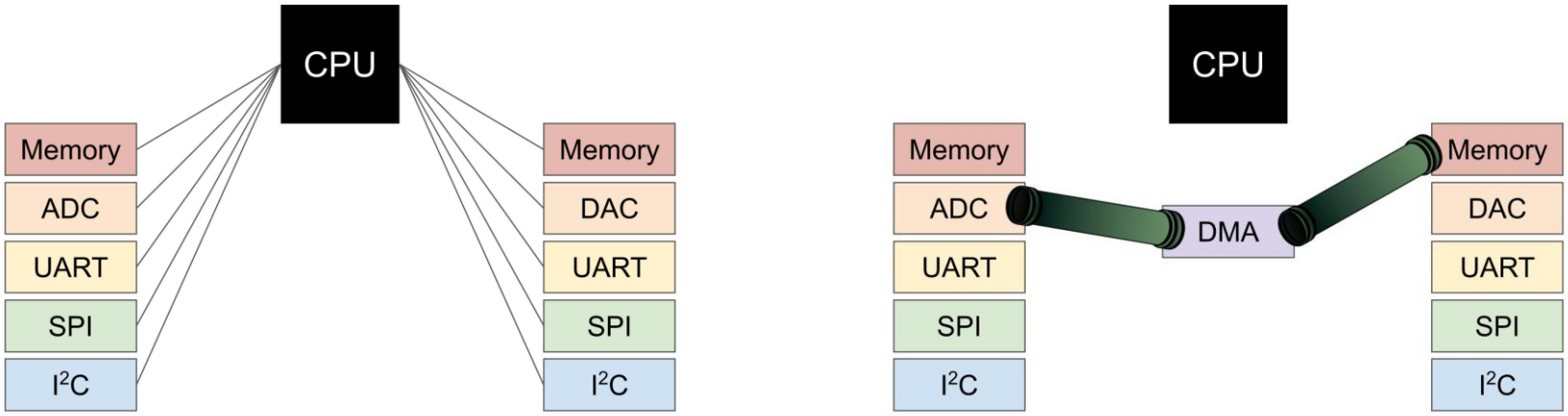
```
/* USER CODE BEGIN 2 */
//HAL_ADC_Start_DMA(&hadc1, adc1_buf, ADC_BUF_LEN);
adc1_count = 0;
HAL_ADC_Start_IT(&hadc1);
/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    if (adc1_count >= 1000)
    {
        char s[20];
        int avg = 0;
        while (adc1_count > 0)
        {
            adc1_count--;
            avg += adc1_buf[adc1_count];
        }
        avg /= 1000;
        sprintf(s, "%3d\n", avg);
        HAL_UART_Transmit(&huart1, s, 4, 10);
        HAL_ADC_Start_IT(&hadc1);
    }
    HAL_Delay(10);
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
}
```

Vào ra bằng ngắt

- Ưu điểm: không mất thời gian chờ, không cần gọi hàm blocking.
- Nhược điểm:
  - Mỗi lần đọc giá trị tương ứng 1 lần gọi ngắt.
  - CPU cần gọi hàm HAL_ADC_GetValue() cho mỗi giá trị.
  - ➔ chi phí xử lý ngắt cao khi lượng dữ liệu vào ra lớn
- Cần có cơ chế vào ra tốt hơn khi truyền nhận dữ liệu khối lượng lớn.
- ➔ Vào ra bằng DMA

Vào ra bằng DMA

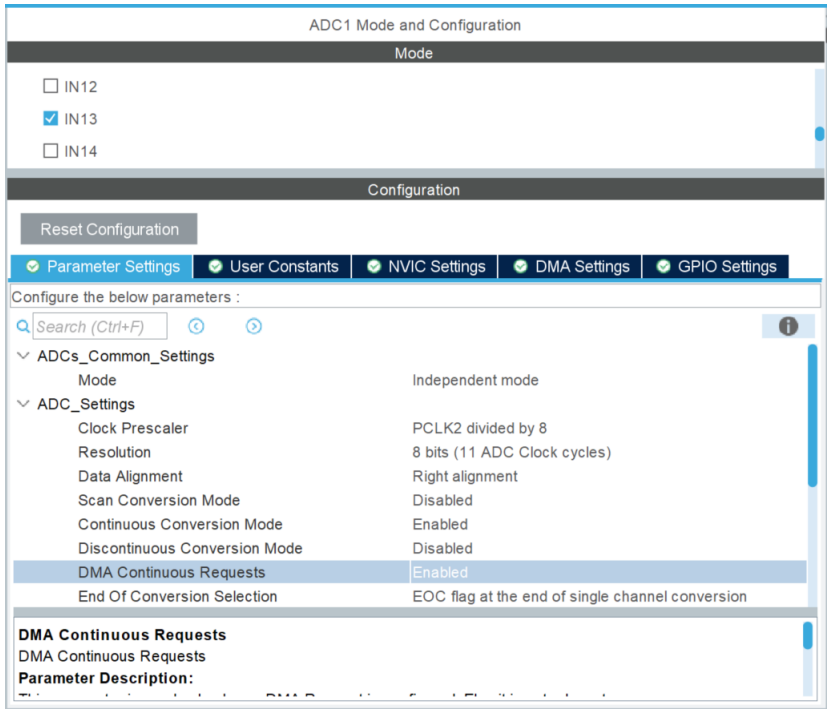
- Dữ liệu được truyền trực tiếp giữa ngoại vi và bộ nhớ, không qua CPU.
- Điều khiển bởi DMA Controller (DMAC).
  - STM32F429 có 2 DMAC
- Có ngắt khi truyền/nhận xong block dữ liệu.
- Hỗ trợ double buffer (ping pong).

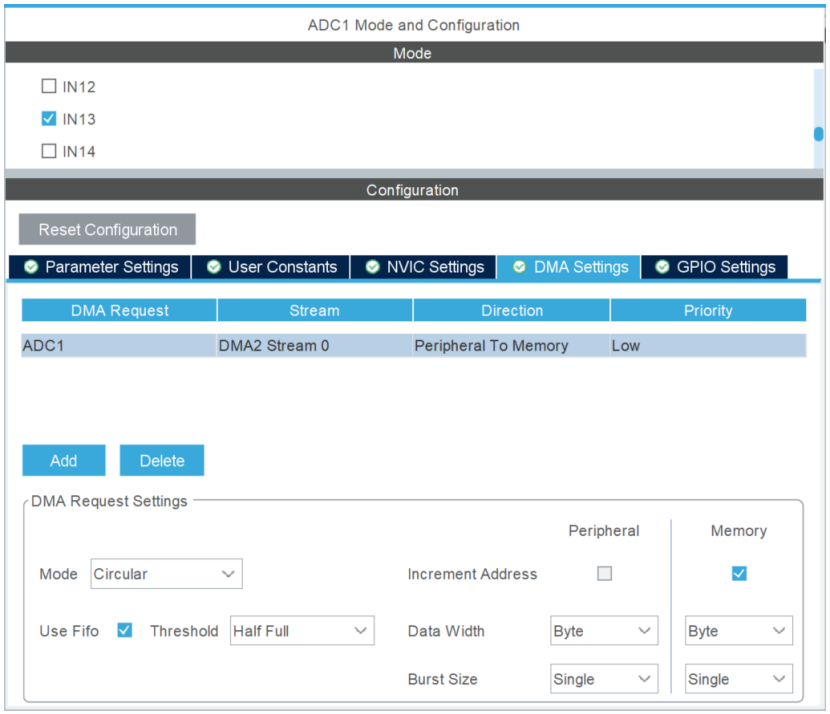


ShawnHymel, Maker.io

Vào ra bằng DMA

- VD: lập trình vào ra bằng DMA cho ADC1
  - Cấu hình Continuos Conversion Mode, DMA Continuous Requests
  - Cấu hình DMA Settings
    - Mode Circular: ring buffer, quay vòng.
    - Fifo, Threshold Half Full: gọi ngắt/call back khi buffer đầy một nửa dung lượng (+ đầy hoàn toàn).





Vào ra bằng DMA

- Gọi hàm HAL_ADC_Start_DMA()
  - Gọi HAL_ADC_Stop_DMA() để ngừng
- Implement 2 hàm callback.
  - Đặt breakpoint ở 2 hàm và chạy debug.

```
/* USER CODE BEGIN 2 */
HAL_ADC_Start_DMA(&hadc1, adc1_buf, ADC_BUF_LEN);
/* USER CODE END 2 */


/* USER CODE BEGIN 4 */
// Called when first half of buffer is filled
void HAL_ADC_ConvHalfCpltCallback(ADC_HandleTypeDef* hadc) {
  if (hadc == &hadc1)
    HAL_GPIO_WritePin(GPIOG, GPIO_PIN_13, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOG, GPIO_PIN_14, GPIO_PIN_SET);
}

// Called when buffer is completely filled
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc) {
  if (hadc == &hadc1)
    HAL_GPIO_WritePin(GPIOG, GPIO_PIN_13, GPIO_PIN_RESET);
  else
    HAL_GPIO_WritePin(GPIOG, GPIO_PIN_14, GPIO_PIN_RESET);
}
/* USER CODE END 4 */
```

Bài tập

- Với ứng dụng ghép nối joystick, chế độ nào là phù hợp:
  - Polling
  - Ngắt
  - DMA?
- Giải thích lý do.

Bài tập

- Lập trình ghép nối với joystick sử dụng ADC ở chế độ vào ra bằng ngắt.
- Đọc giá trị từ joystick và gửi về PC qua USART1 với tần số 20 reading/second.