FRA222 Microcontroller Interface

04 –POLLING, INTERRUPT AND DMA

Blocking & Non-Blocking Method

Blocking – Program will wait until event done.

- HAL_Delay(1000);
- Polling

Non-Blocking – Program will never wait for event to happen

- Interrupt
- DMA

Polling

Polling is the process where the computer or controlling device **waits for an external device** to **check for its readiness or state**

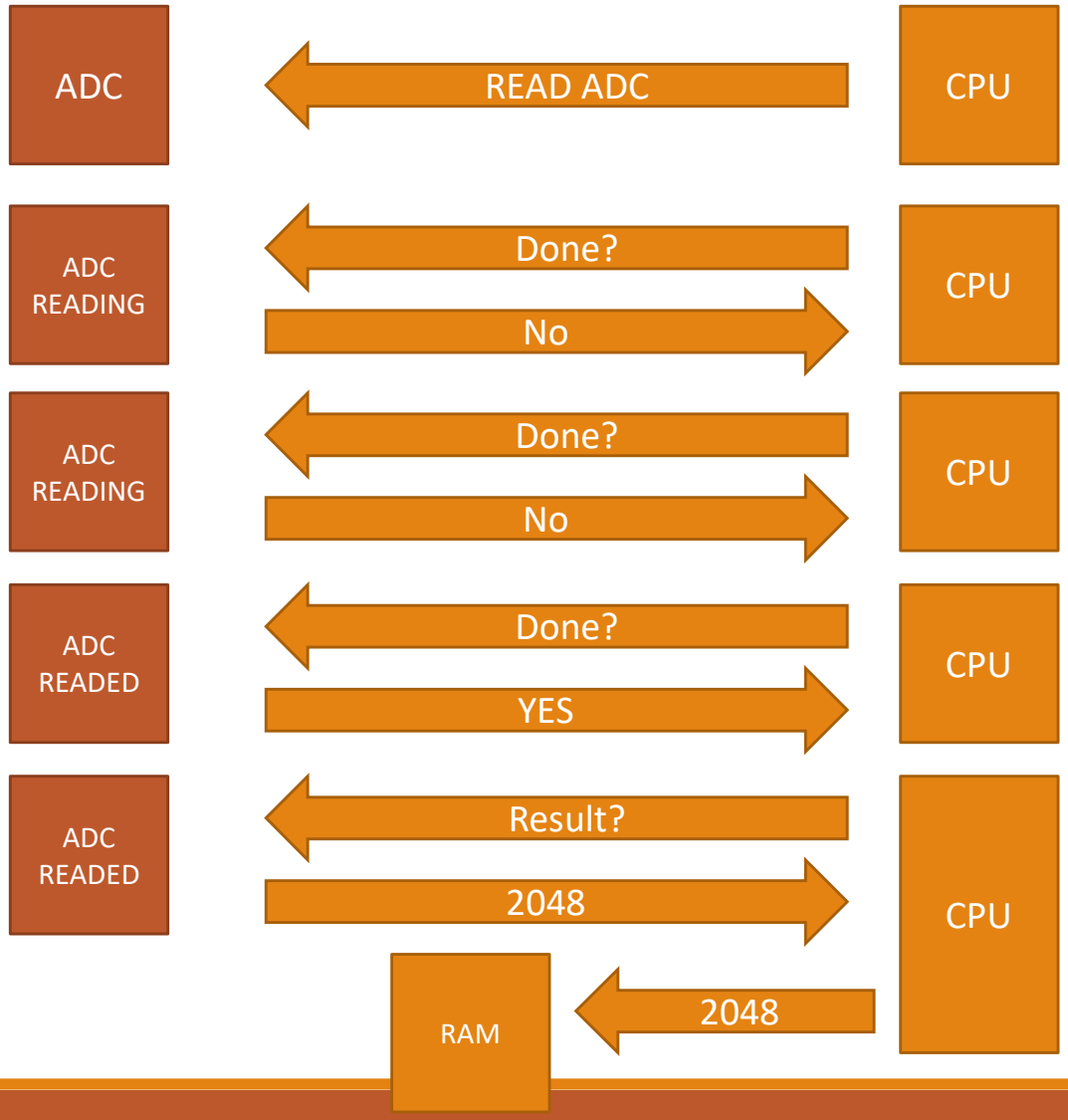
Normal Polling Method

Pro

- Easy to implement
- Predicable output

Con

- Blocking process



interrupt

an interrupt is a response by the processor to an event that needs attention from the software. An interrupt condition alerts the processor and serves as a request for the processor to interrupt the currently executing code when permitted, so that the event can be processed in a timely manner. If the request is accepted, the processor responds by suspending its current activities, saving its state, and executing a function called an interrupt handler (or an interrupt service routine, ISR) to deal with the event. This interruption is temporary, and, unless the interrupt indicates a fatal error, the processor resumes normal activities after the interrupt handler finishes

an interrupt is a response by the processor to an event that needs attention from the software.

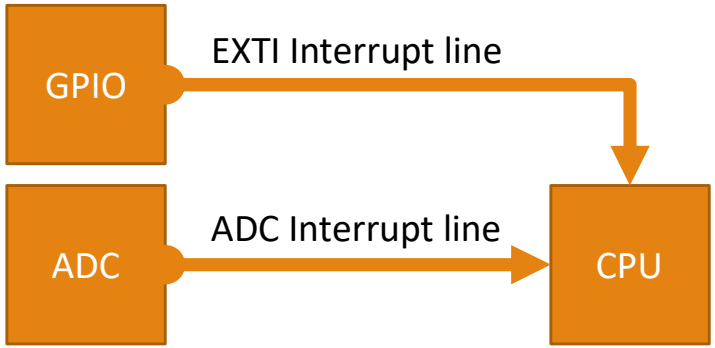
An **interrupt condition** alerts the processor and serves as a **request for the processor to interrupt the currently executing code** when permitted

event can be processed in a timely manner.

something was done without wasting any time

If the request is accepted, the processor responds by suspending its current activities, saving its state, and executing a function called an interrupt handler (or an interrupt service routine, ISR) to deal with the event.

This interruption is temporary, and, unless the interrupt indicates a fatal error, the processor resumes normal activities after the interrupt handler finishes



Main Code
while true
    task A
    task B
    task C
    task D
    task E
    task F
end while

interrupt condition : GPIO PA4 Falling
read time
store time in variable
toggle LED

Use A lot of Interrupt? – DON'T DO THAT

the processor responds by suspending its current activities, **saving its state**, and executing a function called an interrupt handler

ใช้ CPU Resource มากกว่า ปกติ

interrupt condition มีจำกัด
และเป็น **Hardware Implement**

ข้อแนะนำ

1.ใช้ **Interrupt** เมื่อ ต้องทำงานที่เฉพาะเจาะจงเวลามาก ๆ หรือ ต้องทำ ณ ตอนที่เกิดเหตุการณ์นั้นเลย เช่น **emergency stop**

2.งานที่ทำตอน **interrupt** ควรจะ สั้น และ เร็ว

3.ทั้งนี้ควรพิจราณา งานที่**controller** ทำ ประกอบการเลือกใช้ **interrupt**

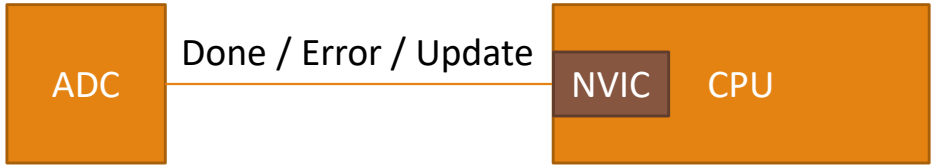
interrupt condition มีจำกัด และเป็น Hardware Implement

Table 37. Vector table for STM32F411xC/E (continued)

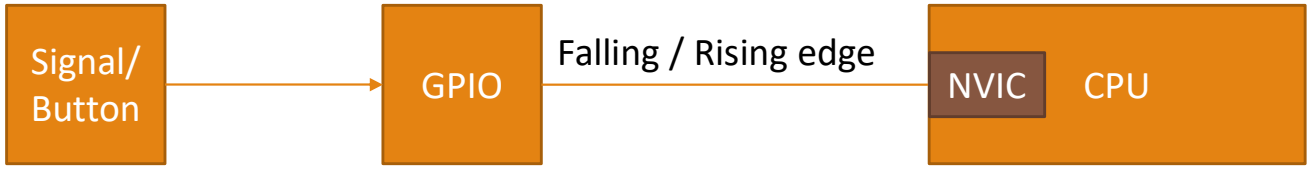
Position	Priority	Type of priority	Acronym	Description	Address
	-2	fixed	NMI	Non maskable interrupt, Clock Security System	0x0000 0008
	-1	fixed	HardFault	All class of fault	0x0000 000C
	0	settable	MemManage	Memory management	0x0000 0010
	1	settable	BusFault	Pre-fetch fault, memory access fault	0x0000 0014
	2	settable	UsageFault	Undefined instruction or illegal state	0x0000 0018
	-	-	-	Reserved	0x0000 001C - 0x0000 002B
	3	settable	SVCall	System Service call via SWI instruction	0x0000 002C
	4	settable	Debug Monitor	Debug Monitor	0x0000 0030
		-	-	Reserved	0x0000 0034
	5	settable	PendSV	Pendable request for system service	0x0000 0038
	6	settable	Systick	System tick timer	0x0000 003C
0	7	settable	WWDG	Window Watchdog interrupt	0x0000 0040
1	8	settable	EXTI16 / PVD	EXTI Line 16 interrupt / PVD through EXTI line detection interrupt	0x0000 0044
				EXTI Line 21 interrupt /	
15	22	settable	DMA1_Stream4	DMA1 Stream4 global interrupt
16	23	settable	DMA1_Stream5	DMA1 Stream5 global interrupt
17	24	settable	DMA1_Stream6	DMA1 Stream6 global interrupt
18	25	settable	ADC	ADC1 global interrupts
23	30	settable	EXTI9_5	EXTI Line[9:5] interrupts
24	31	settable	TIM1_BRK_TIM9	TIM1 Break interrupt and TIM9 global interrupt
25	32	settable	TIM1_UP_TIM10	TIM1 Update interrupt and TIM10 global interrupt
26	33	settable	TIM1_TRG_COM_TIM11	TIM1 Trigger and Commutation interrupts and TIM11 global interrupt
27	34	settable	TIM1_CC	TIM1 Capture Compare interrupt
28	35	settable	TIM2	TIM2 global interrupt
29	36	settable	TIM3	TIM3 global interrupt
30	37	settable	TIM4	TIM4 global interrupt
31	38	settable	I2C1_EV	I^2C1 event interrupt
32	39	settable	I2C1_ER	I^2C1 error interrupt
33	40	settable	I2C2_EV	I^2C2 event interrupt
34	41	settable	I2C2_ER	I^2C2 error interrupt
35	42	settable	SPI1	SPI1 global interrupt
36	43	settable	SPI2	SPI2 global interrupt
37	44	settable	USART1	USART1 global interrupt

Interrupt in STM32

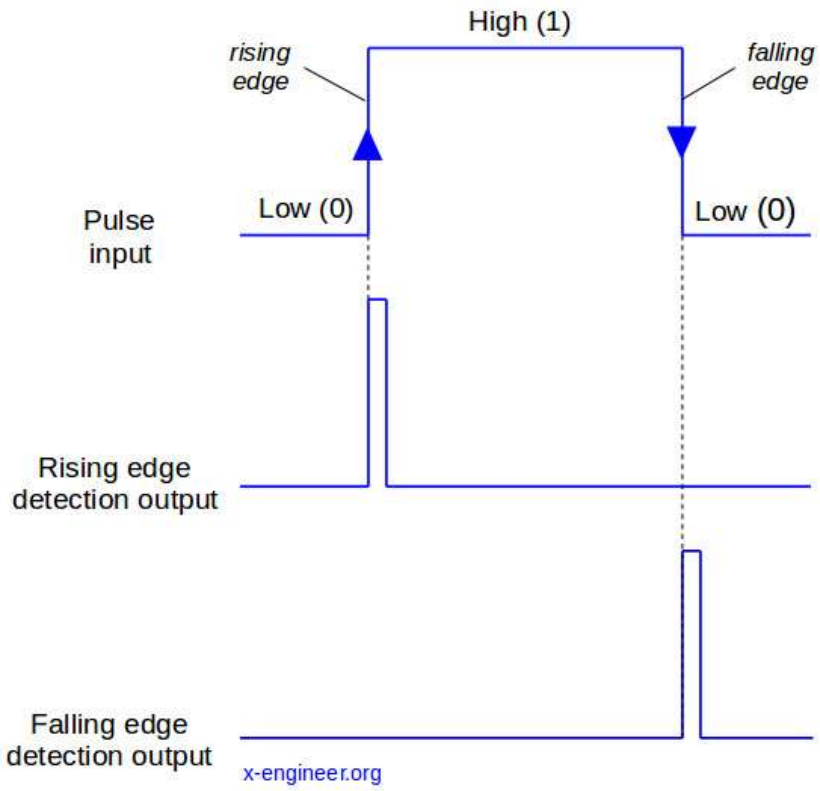
Peripheral interrupt



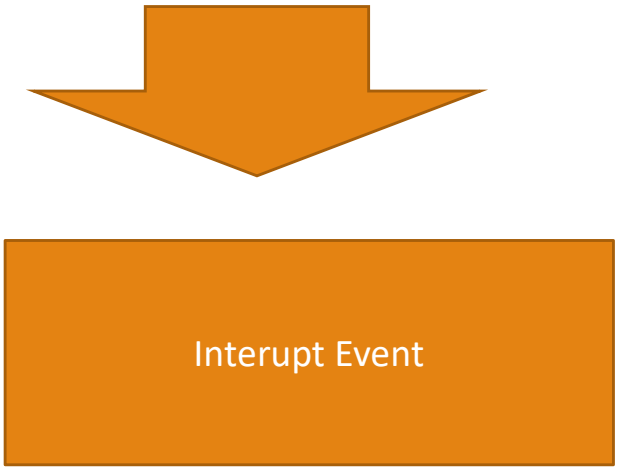
External interrupt



External Interrupt – Interrupt via GPIO



Falling edge
Rising edge
Both



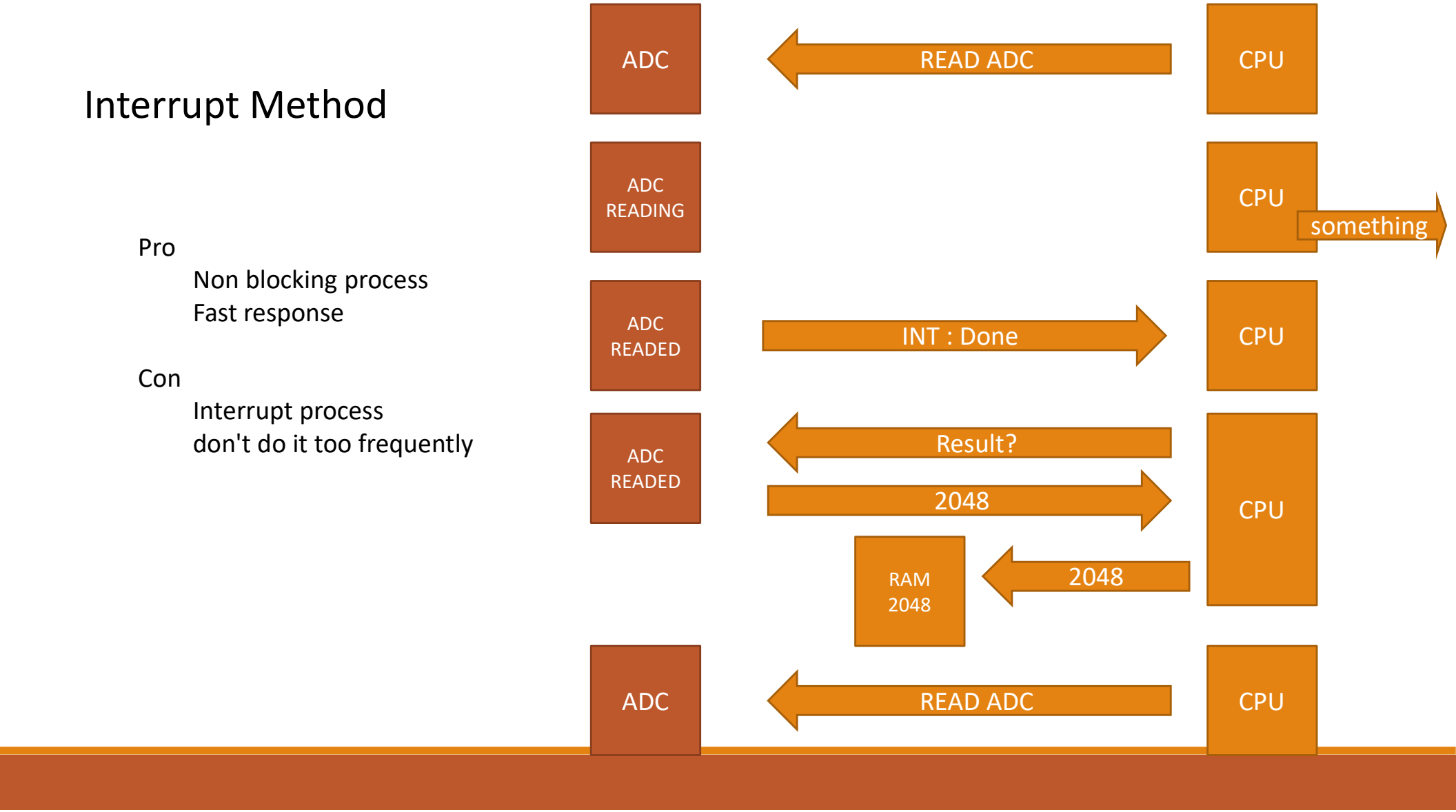
Interrupt Method

Pro

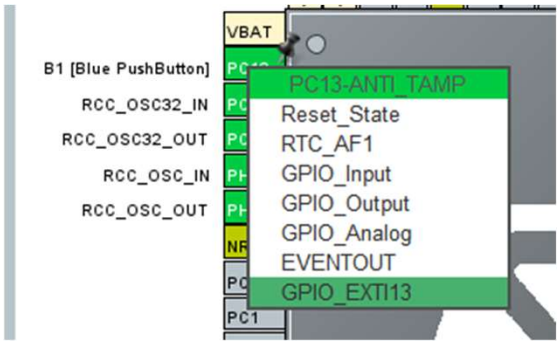
- Non blocking process
- Fast response

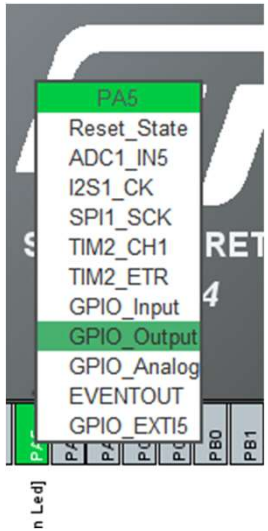
Con

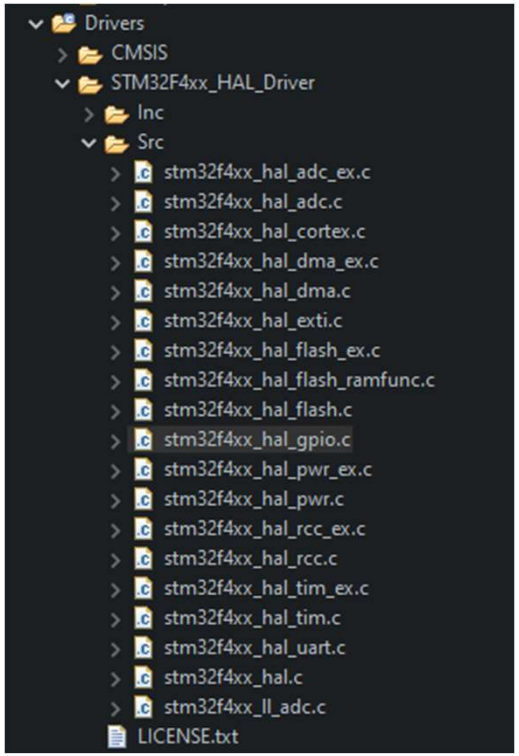
- Interrupt process
- don't do it too frequently

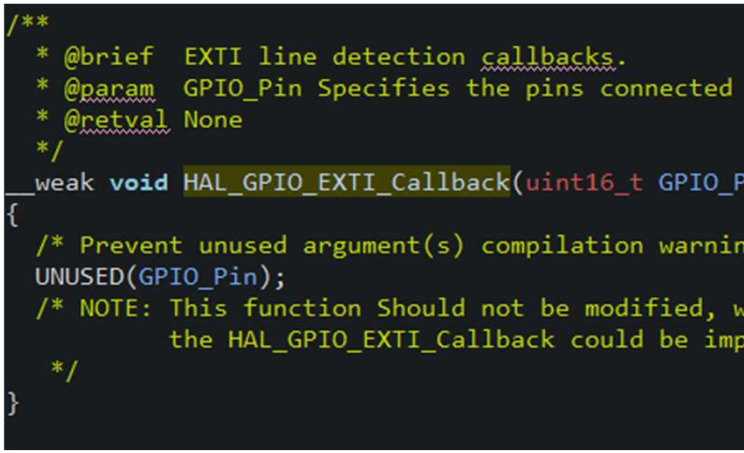


Example : External interrupt to toggle pin









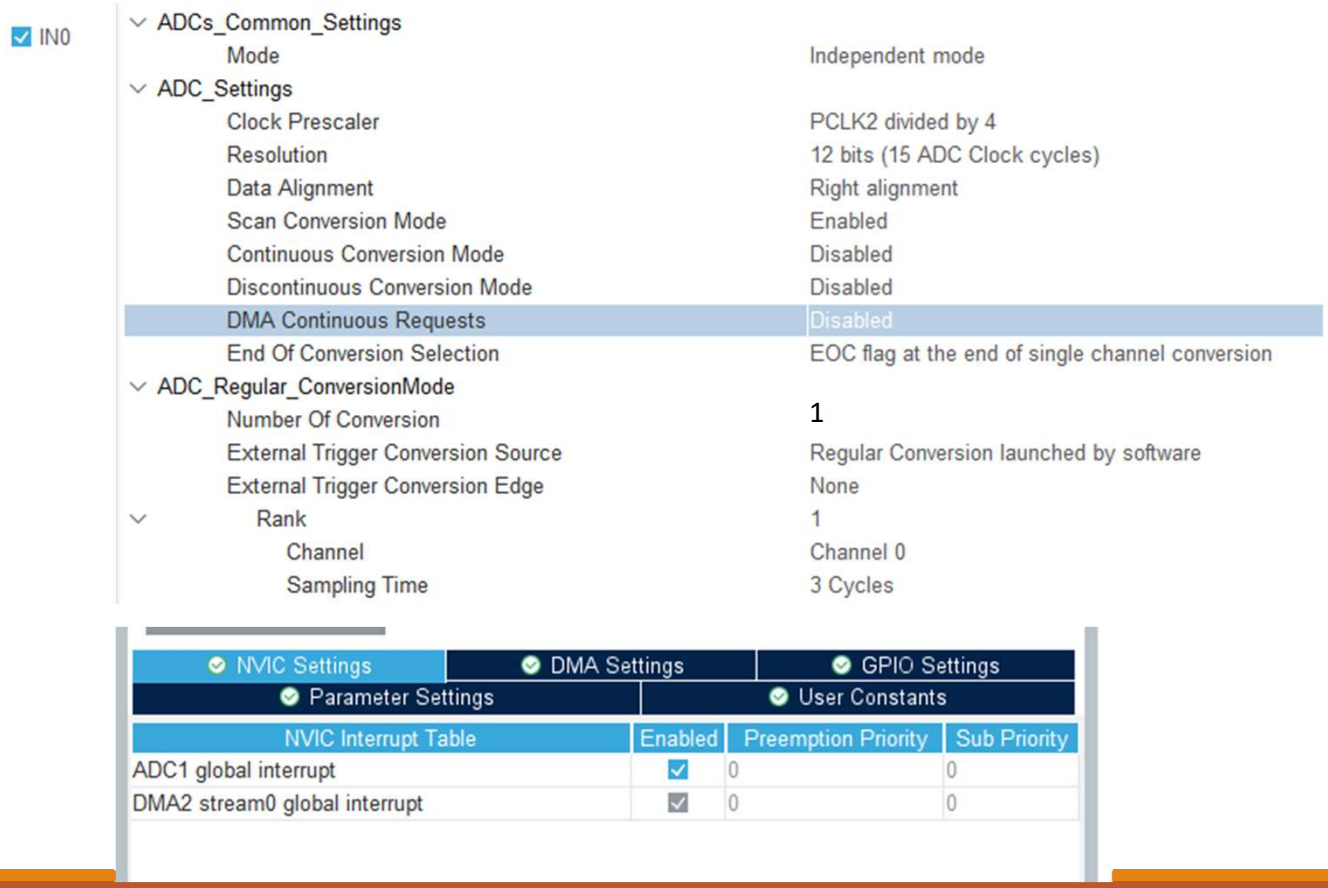
```
/**
  * @brief  EXTI line detection callbacks.
  * @param  GPIO_Pin Specifies the pins connected
  * @retval None
  */
__weak void HAL_GPIO_EXTI_Callback(uint16_t GPIO_P
{
  /* Prevent unused argument(s) compilation warnin
  UNUSED(GPIO_Pin);
  /* NOTE: This function Should not be modified, w
           the HAL_GPIO_EXTI_Callback could be imp
   */
}
```

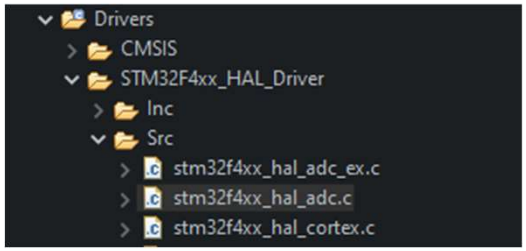
Pin N...	Signal on ...	GPIO out...	GPIO mode	G
PA5	n/a	Low	Output Push Pull	N
PC13-AN...	n/a	n/a	External Interrupt Mode with Falling edge tri...	N

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
      //Simulate Task
      HAL_Delay(10000);

  }
  /* USER CODE END 3 */
```

```
308 /* USER CODE BEGIN 4 */
309
310 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
311 {
312     if(GPIO_Pin == GPIO_PIN_13)
313     {
314         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
315     }
316 }
317
318 /* USER CODE END 4 */
```

Example : External interrupt to call interrupt ADC






```
1574 /**
1575   * @brief  Regular conversion complete callback in non blocking mode
1576   * @param  hadc pointer to a ADC_HandleTypeDef structure that contains
1577   *         the configuration information for the specified ADC.
1578   * @retval None
1579   */
1580 __weak void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
1581 {
1582   /* Prevent unused argument(s) compilation warning */
1583   UNUSED(hadc);
1584   /* NOTE : This function Should not be modified, when the callback is needed,
1585             the HAL_ADC_ConvCpltCallback could be implemented in the user file
1586    */
1587 }
1588
```

```
307
308 /* USER CODE BEGIN 4 */
309
310 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
311 {
312     if(GPIO_Pin == GPIO_PIN_13)
313     {
314         HAL_ADC_Start_IT(&hadc1);
315     }
316 }
317
318 void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
319 {
320     adcRawData = HAL_ADC_GetValue(&hadc1);
321
322     /* DO NOT DO THIS */
323     //HAL_ADC_Start_IT(&hadc1);
324 }
325
326 /* USER CODE END 4 */
327
```

DMA – Direct Memory Access

provide high-speed data transfer between

- peripherals and memory
- memory and memory

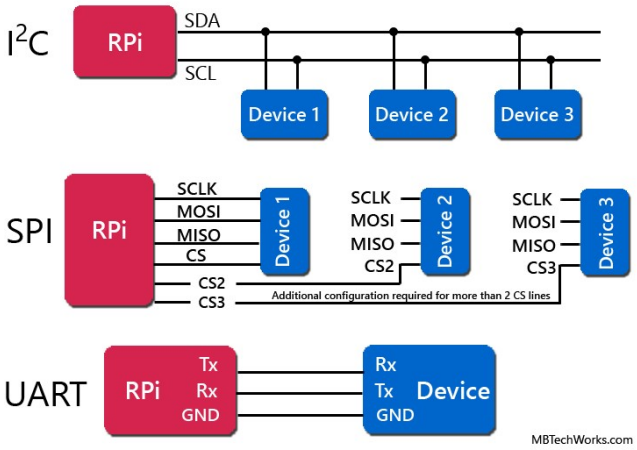
Data can be quickly moved by DMA **without** any CPU action

- keeps CPU resources free for other operations

USE DMA

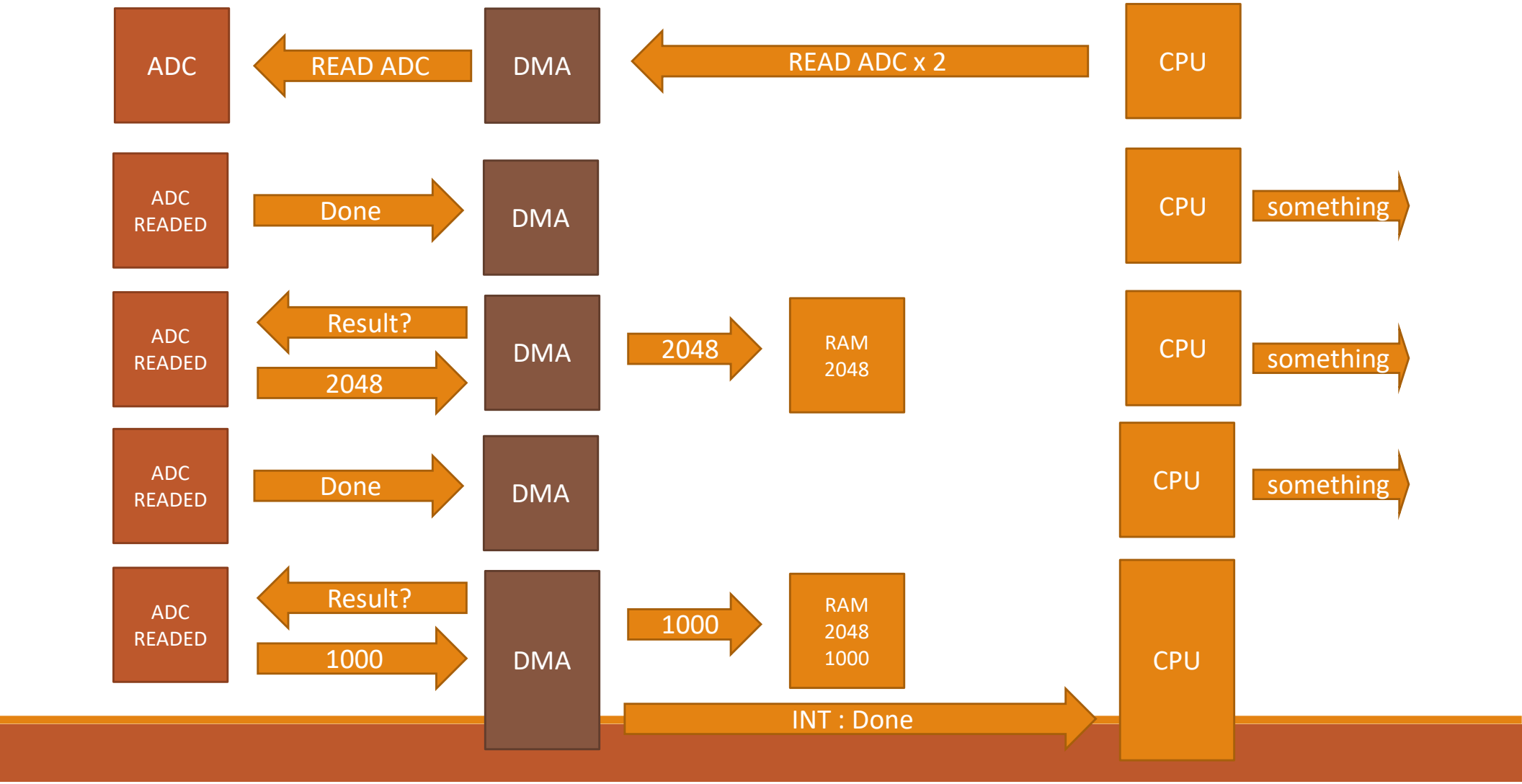
Move Large amount of data

Continues streaming data



https://youtu.be/vnEwzN14BsU

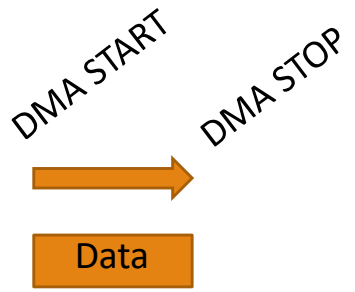
DMA Method


ADC
READ ADC
DMA
READ ADC x 2
CPU
ADC READED
Done
DMA
CPU
something
ADC READED
Result?
2048
DMA
2048
RAM 2048
CPU
something
ADC READED
Done
DMA
CPU
something
ADC READED
Result?
1000
DMA
1000
RAM 2048 1000
CPU
INT : Done

Buffer
Normal , Circular

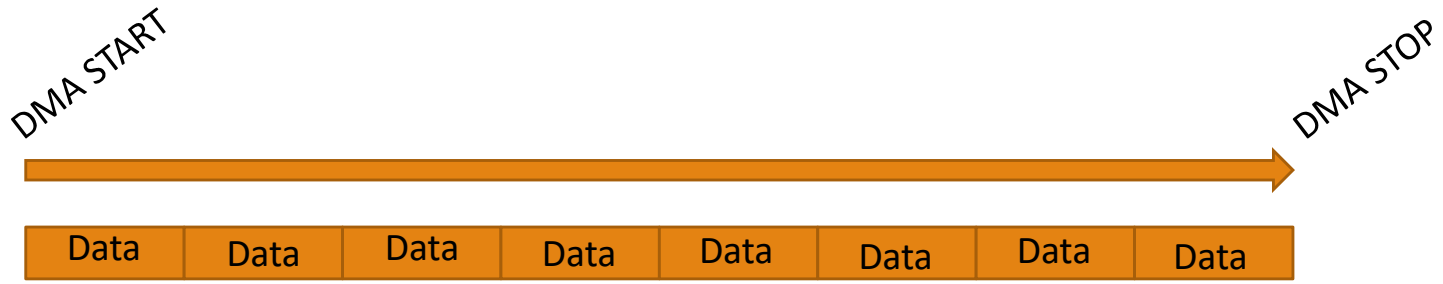
Buffer - a region of a physical memory storage used to temporarily store data while it is being moved from one place to another

Normal buffer can hold specific amount of data



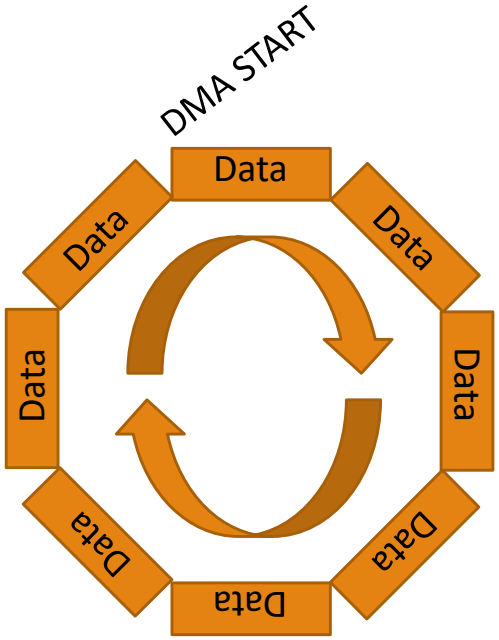
Buffer size 1

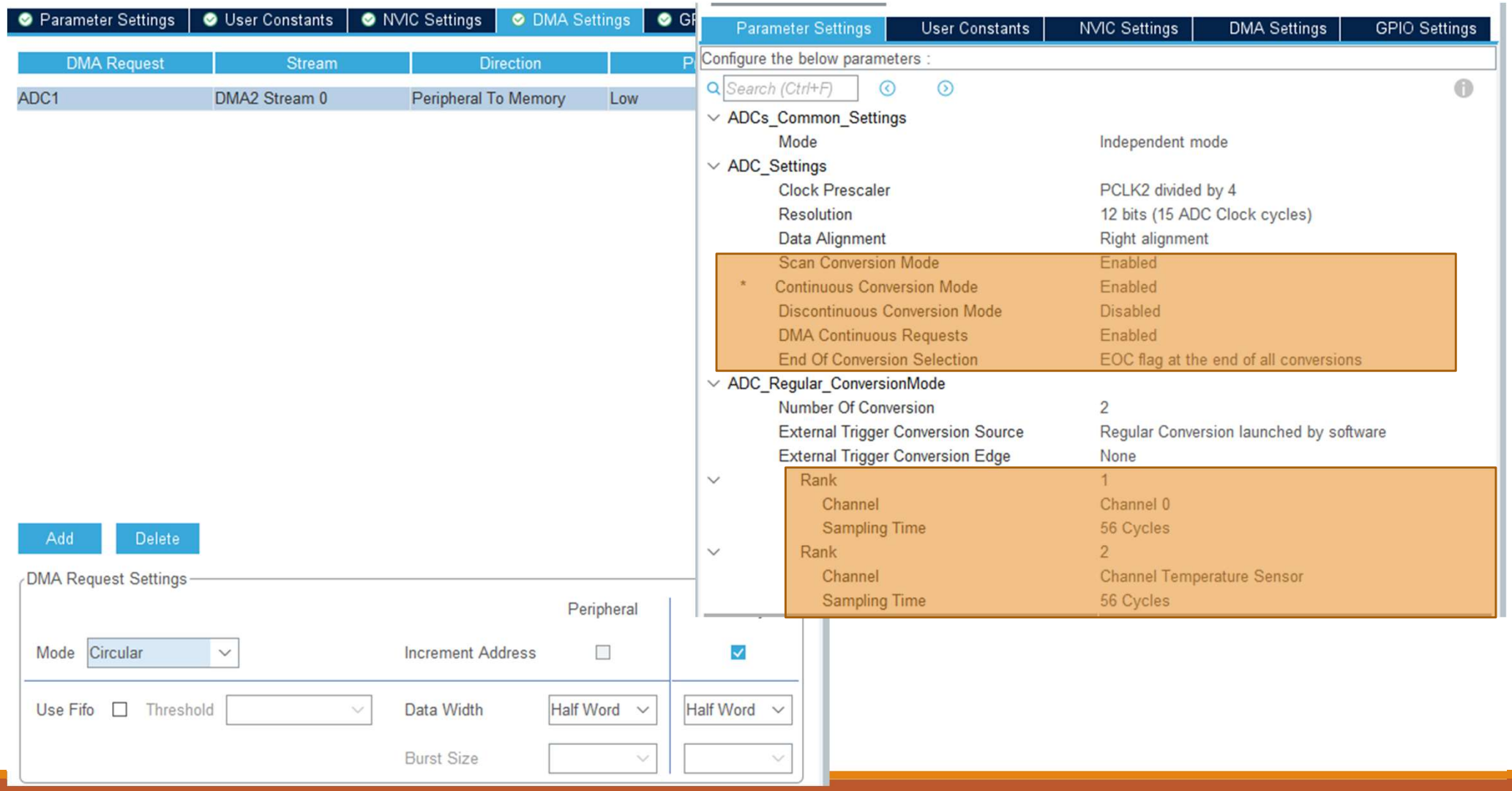
Buffer
Normal



Buffer size 8

Buffer
Circular




Parameter Settings
User Constants
NVIC Settings
DMA Settings
GP
DMA Request
Stream
Direction
P
ADC1
DMA2 Stream 0
Peripheral To Memory
Low
Add
Delete
DMA Request Settings
Peripheral
Mode
Circular
Increment Address
Use Fifo
Threshold
Data Width
Half Word
Half Word
Burst Size
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Configure the below parameters :
Search (Ctrl+F)
ADCs_Common_Settings
Mode
Independent mode
ADC_Settings
Clock Prescaler
PCLK2 divided by 4
Resolution
12 bits (15 ADC Clock cycles)
Data Alignment
Right alignment
Scan Conversion Mode
Enabled
* Continuous Conversion Mode
Enabled
Discontinuous Conversion Mode
Disabled
DMA Continuous Requests
Enabled
End Of Conversion Selection
EOC flag at the end of all conversions
ADC_Regular_ConversionMode
Number Of Conversion
2
External Trigger Conversion Source
Regular Conversion launched by software
External Trigger Conversion Edge
None
Rank
1
Channel
Channel 0
Sampling Time
56 Cycles
Rank
2
Channel
Channel Temperature Sensor
Sampling Time
56 Cycles

```c
typedef union
{
    struct
    {
        uint16_t ADC_IN0;
        uint16_t TempSensor;
    }subData;
    uint16_t buffer[2];
}DMA_ADC_BufferType;

DMA_ADC_BufferType buffer[10];
```

```c
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin == GPIO_PIN_13)
    {
        HAL_ADC_Start_DMA(&hadc1, (uint32_t*)buffer, 200);
        //Buffer size 100 * 2
    }
}

void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
{
    // this function sill call after ADC - DMA Finish.
}
```

LAB 2 : just a very simple voltage meter++

สร้าง voltage meter โดยใช้ ADC โดย **ใช้ voltage divider เพื่อลดแรงดัน ไฟฟ้า จาก 5 V ➔ 2.5V**

- Vin มีค่าระหว่าง 0 mV – 5000mV จำลองVoltage sourceโดยใช้
  - VR ต่อเข้ากับ ไฟ 5V Vin และ GND
  - ต่อVin เข้า 5V
  - ต่อVin เข้า 3.3V

- โปรแกรมจะต้อง อ่านค่า ADC และ คำนวนแรงดัน Vin ออกมาในหน่วย mV
- ความถี่ในการคำนวนค่าที่ได้จาก ADC อยู่ที่ 1 Hz
- ในการอ่าน ADC จะต้องอ่านโดยใช้ DMA ในการอ่าน และใช้อย่างน้อย 10 ค่า และนำมาหาค่าเฉลี่ยเพื่อใช้ในการคำนวน
- แสดงค่าให้เห็นผ่านตัวแปร อะไรก็ได้ ใน live expression โดยแสดงในหน่วย mV
- แสดงอุณหภูมิของ MCU ในหน่วย K ด้วย

- คำถาม
  - ความละเอียดของมิเตอร์ของน้องๆ ถ้าคำนวนจาก ADC Resolution มีความละเอียดที่สามารถอ่านผ่าน Vin ได้เท่าไร
  - ความละเอียดจริงๆ ที่สามารถอ่านได้จากตัวแปรปัจจุบัน ได้ตรงกับที่คิดหรือไม่ เพราะเหตุใด
  - เมื่อเราใช้VR ในการจำลอง Voltage source เราสามารถอ่านค่าได้ถึง 5000 mV หรือไม่ เพราะเหตุใด