FRA222 Microcontroller Interface

03 –PERIPHERALS & ADC

Source: https://www.st.com/resource/en/datasheet/stm32f411re.pdf
https://www.st.com/resource/en/reference_manual/dm00119316-stm32f411xce-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

Peripherals

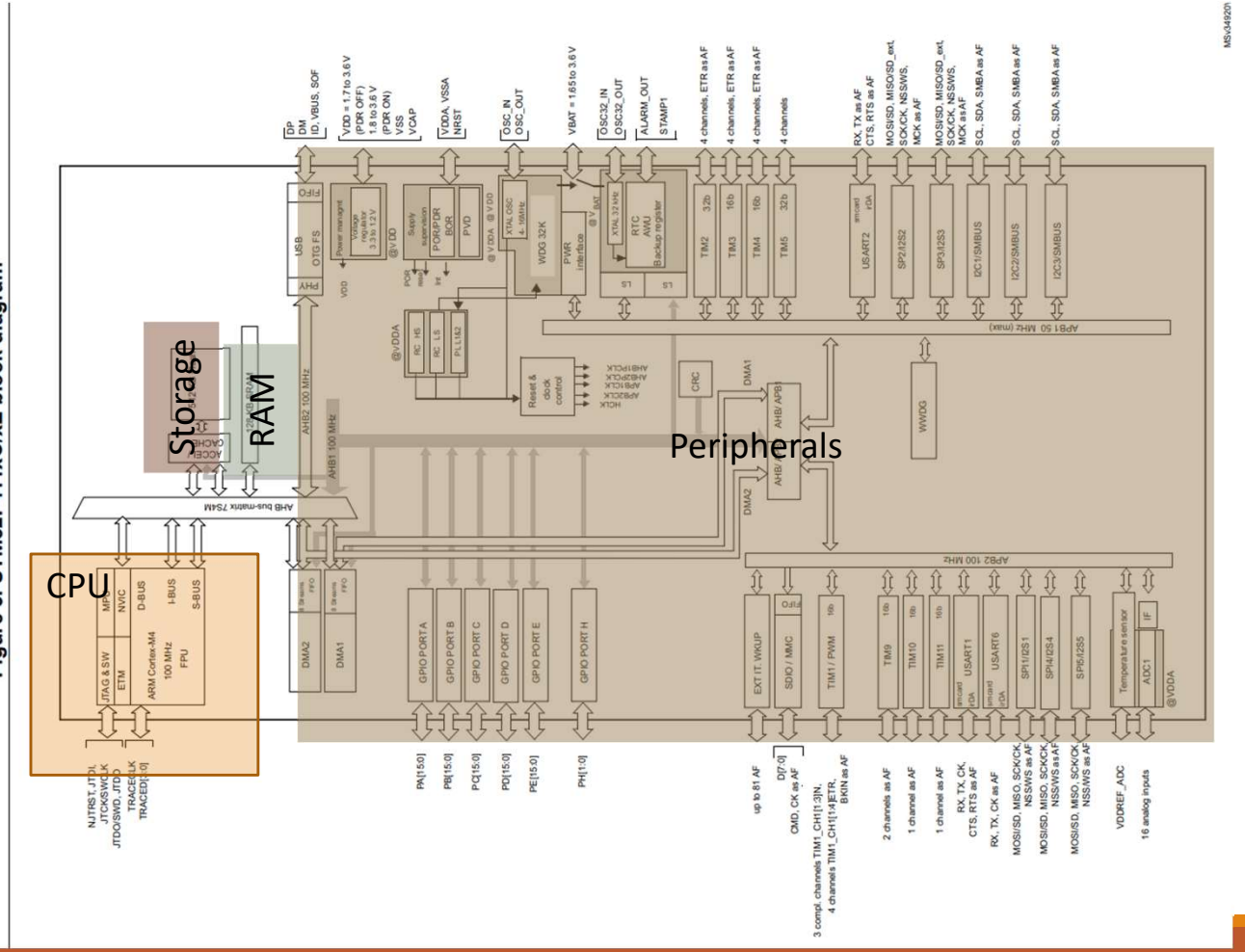
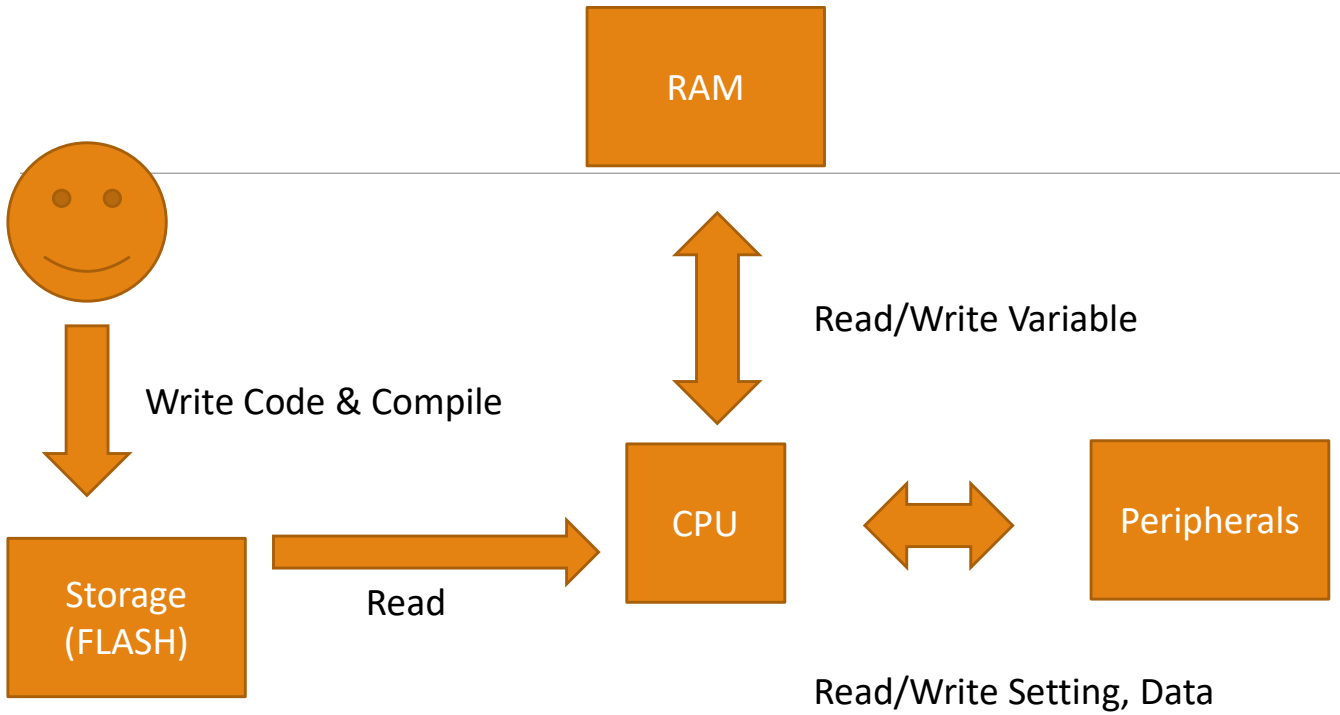


Figure 3. STM32F411xC/xE block diagram

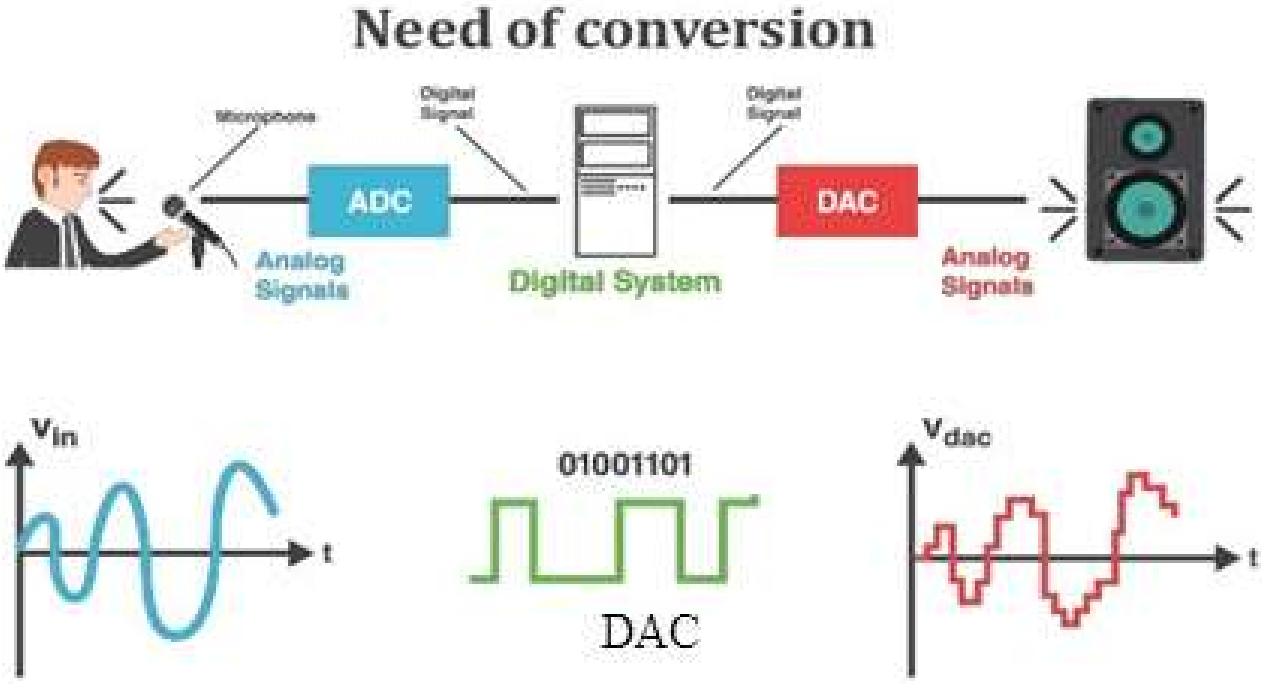


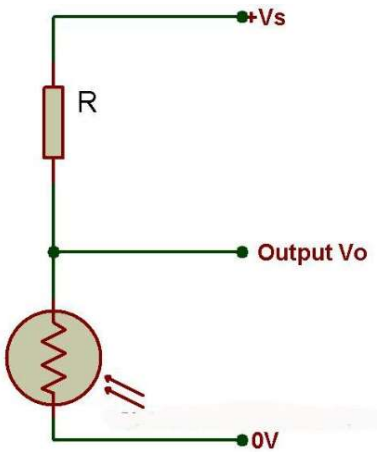
Peripherals can parallel processing with CPU

CPU	Peripheral A	Peripheral B	Peripheral C
Hey A, Measure this	Ok ,1 min		
Hey B output this		Ok, done	
Hey C ,Have any input?	[A done his task]		No
Hey A, Are you done?	Yes, data is 1023		[C have new input]
[Do something with data]			
Hey C ,Have any input?			Yes
Hey C ,what is it?			It’s “Hello World”

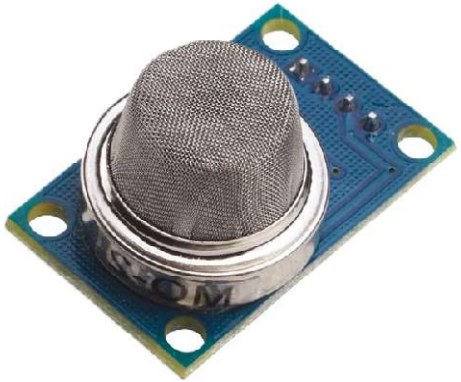
Peripherals - ADC

Microcontroller is Digital
BUT
World is Analog





Sensor แบบ Voltage divider

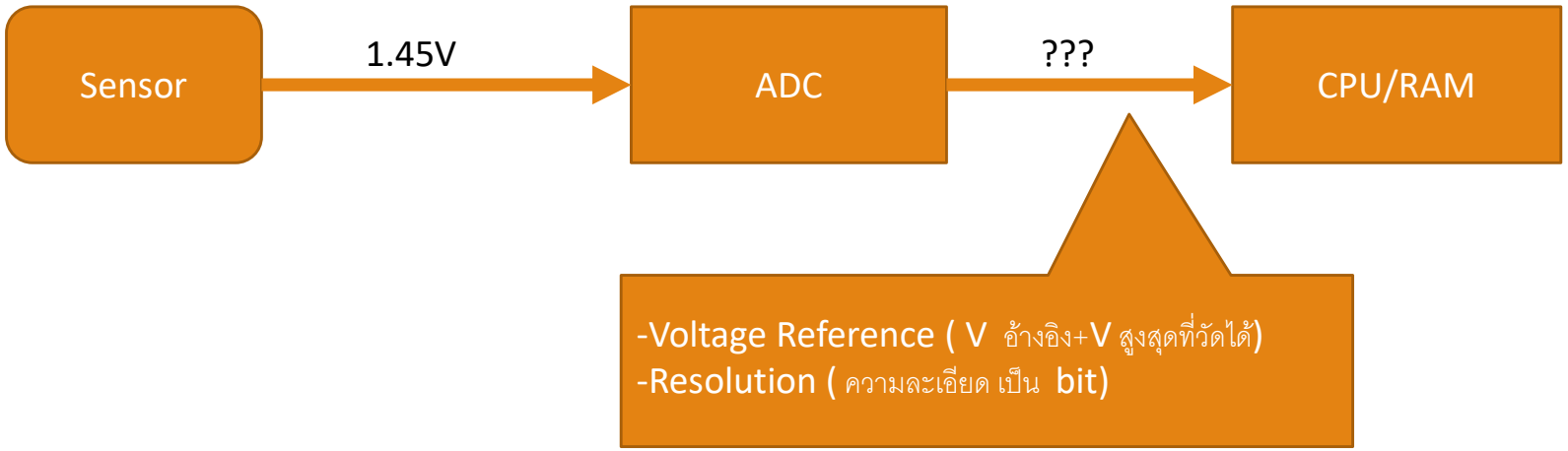


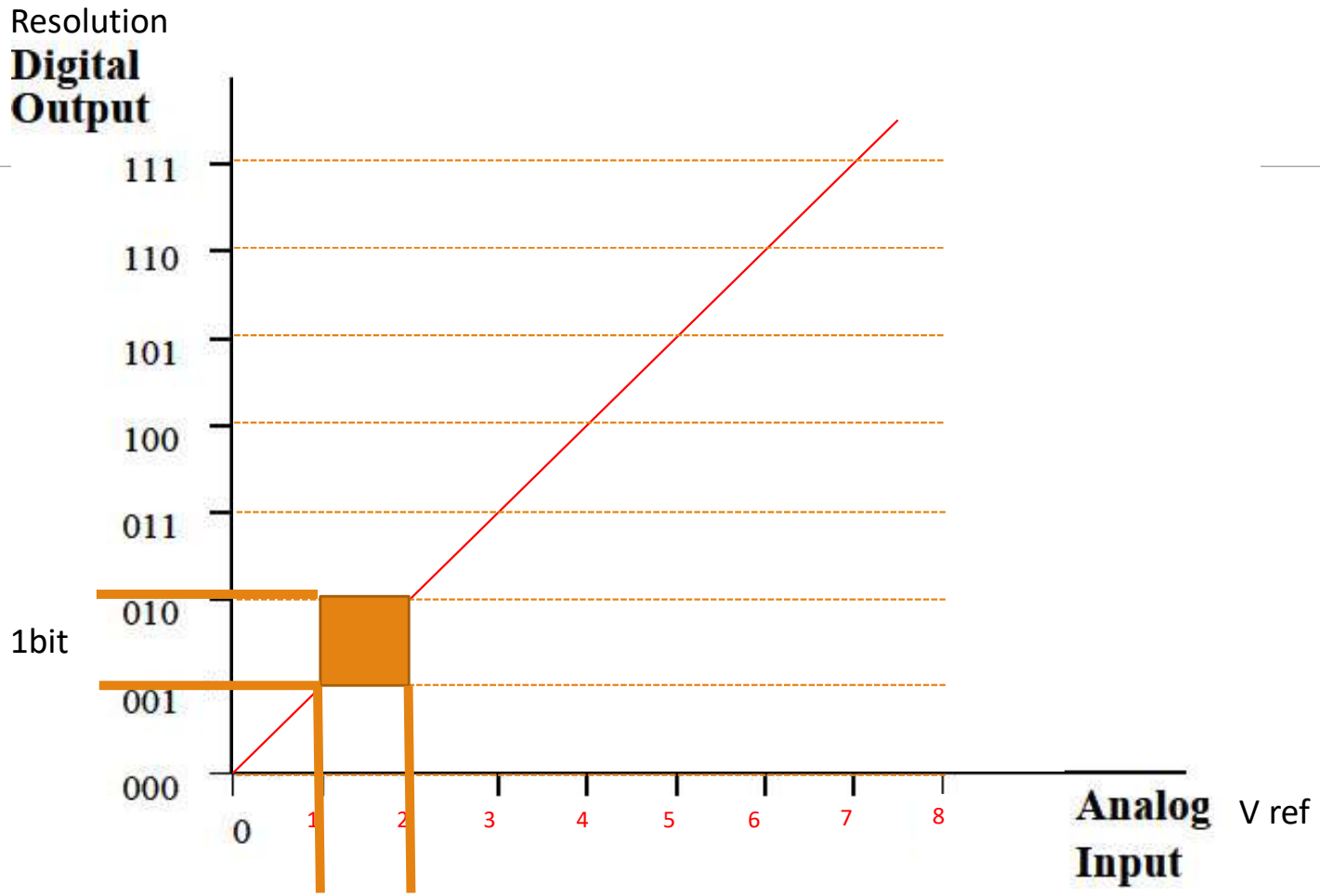
MQ-2 FC-22 Smoke Gas Sensor

ความเข้มข้นของปริมาณก๊าซแสดงในรูปของแรงดัน 0-5V

ความเข้มข้นยิ่งสูง แรงดันที่อ่านได้ยิ่งสูง

ADC


Sensor
1.45V
ADC
???
CPU/RAM
-Voltage Reference (V อ้างอิง+V สูงสุดที่วัดได้)
-Resolution (ความละเอียด เป็น bit)



1LSB = step ของ V / 1bits

Resolution & V Ref

Resolution - จำนวน **bit** ของ **ADC** ยิ่งจำนวน **bit** มาก ความละเอียดยิ่งสูง

V Ref - แรงดันสูงสุดที่สามารถ **convert** ได้

LSB / Step – ความละเอียด ของ 1 ขั้น ของ **ADC**

$$LSB = \frac{V_{ref}}{2^{Res}}$$

1 V , 2 Bit	3.3V ,12bit
$\frac{1}{2^2} = \frac{1}{4} = 0.25\ V/LSB$	$\frac{3.3}{2^{12}} = \frac{3.3}{4096} \approx 0.8mV/LSB$

2 bit ADC

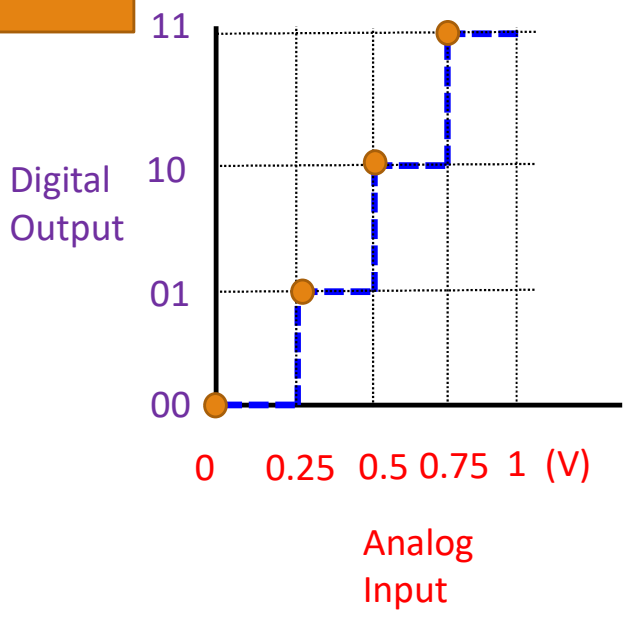
1 V , 2 Bit

$$\frac{1}{2^2} = \frac{1}{4} = 0.25\ V/LSB$$

Vref: 1V

truncate

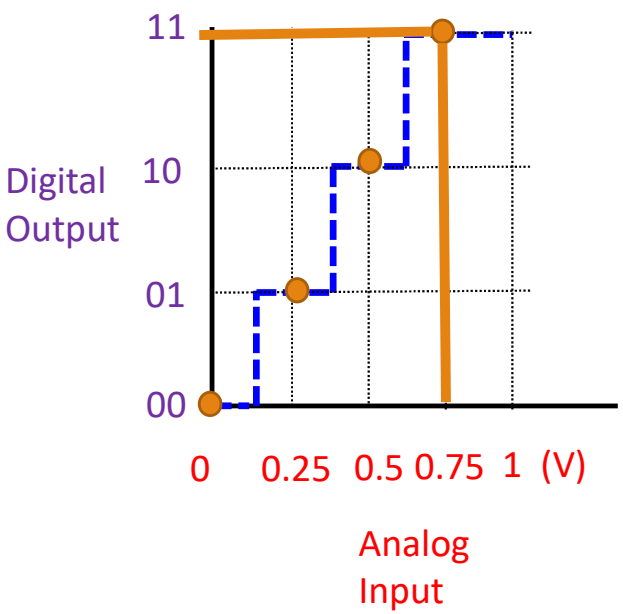
ปัดเศษทิ้ง



Vref: 1V

round-off

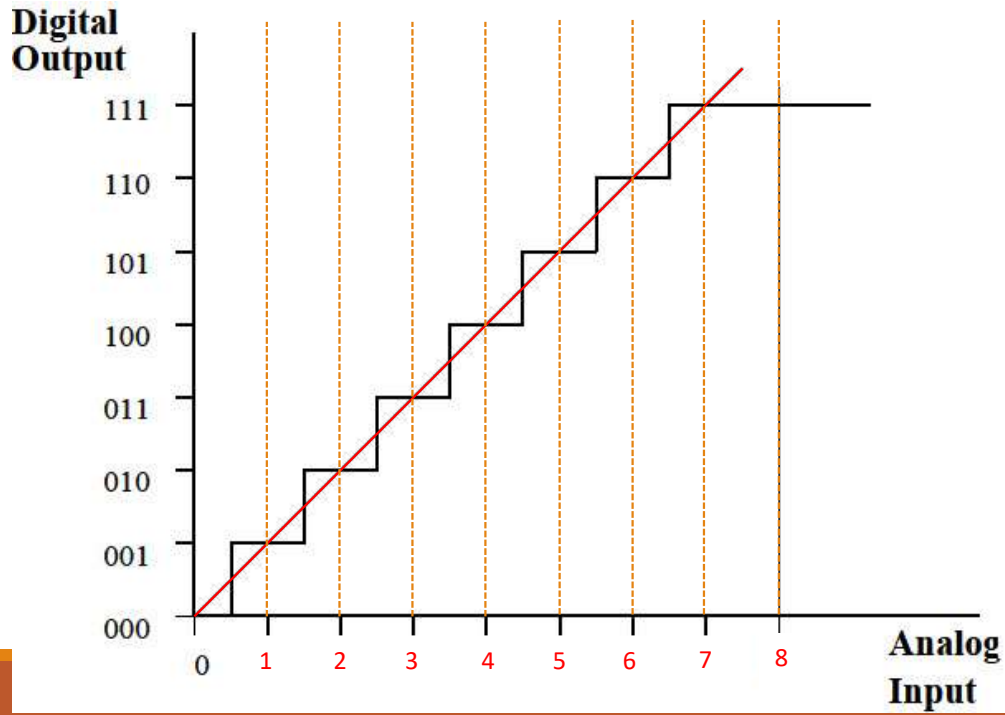
ปัดเศษขึ้น/ลง



3 bit ADC with Vref 8V

กรณี 3-bit ADC: ค่าแรงดันจะถูกแบ่งเป็น $2^3 = 8$ ระดับ

กรณี Vref = 8V แสดงว่าแต่ละ step ของค่า Digital แสดงถึงระดับแรงดัน $8/2^3 = 1V$

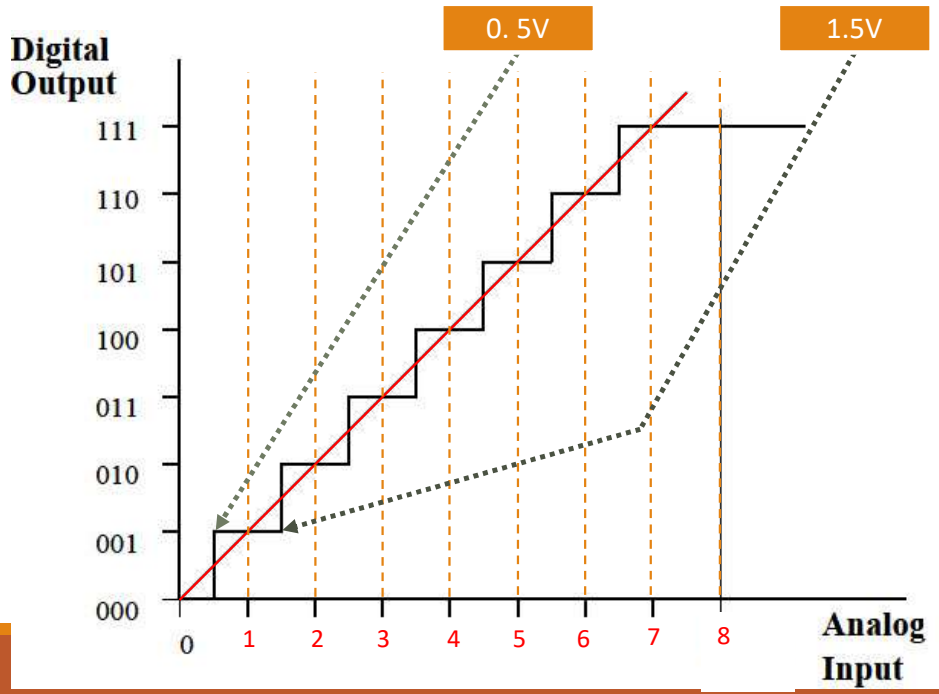


Quantisation error

Resolution ของ ADC ส่งผลถึง quantisation error ด้วย เช่น กรณี 3-bit ADC จะมี step width = 1V และ worst case quantisation error ที่ 0.5 V

เช่นค่า Digital เท่ากับ 001 สามารถเกิดขึ้นเมื่อแรงดัน Analog มีค่าตั้งแต่

$$1 - 0.5 < \text{Analog} < 1 + 0.5$$



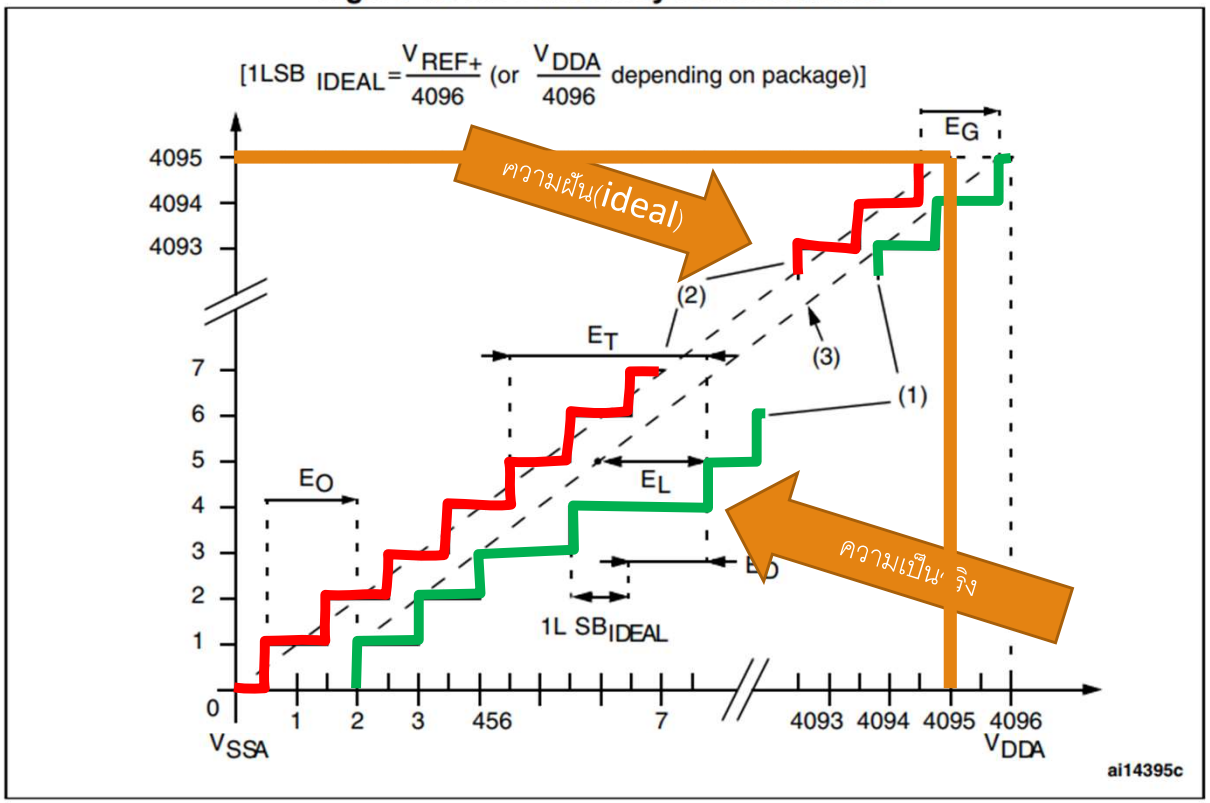
ADC – Hardware Error

Table 68. ADC accuracy at f_{ADC} = 36 MHz(1)

Symbol	Parameter	Test conditions	Typ	Max(2)	Unit
ET	Total unadjusted error	f_{ADC} =36 MHz, V_{DDA} = 2.4 to 3.6 V, V_{REF} = 1.7 to 3.6 V V_{DDA} –V_{REF} < 1.2 V	±4	±7	LSB
EO	Offset error		±2	±3	
EG	Gain error		±3	±6	
ED	Differential linearity error		±2	±3	
EL	Integral linearity error		±3	±6	

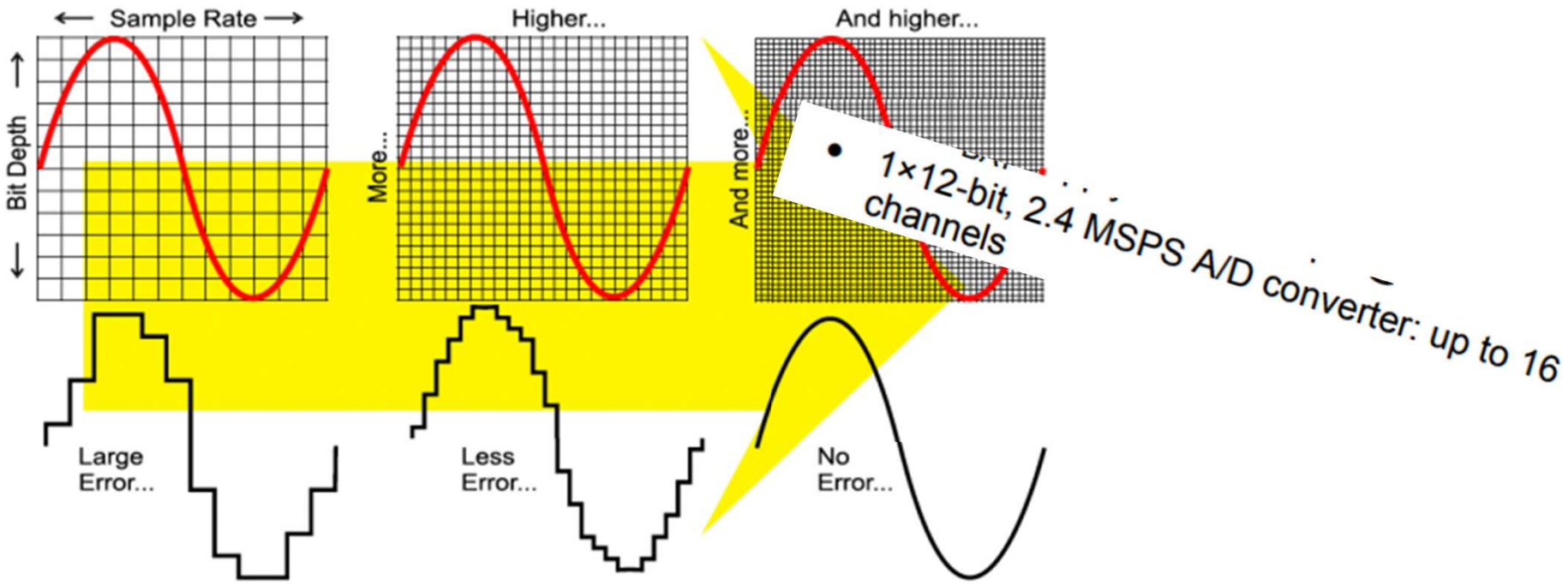
1. Better performance could be achieved in restricted V_{DD}, frequency and temperature ranges.
2. Guaranteed by characterization results.

Figure 40. ADC accuracy characteristics



1. See also *Table 67*.
2. Example of an actual transfer curve.
3. Ideal transfer curve.
4. End point correlation line.
5. E_T = Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.
   EO = Offset Error: deviation between the first actual transition and the first ideal one.
   EG = Gain Error: deviation between the last ideal transition and the last actual one.
   ED = Differential Linearity Error: maximum deviation between actual steps and the ideal one.
   EL = Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

ADC – Sampling Rate



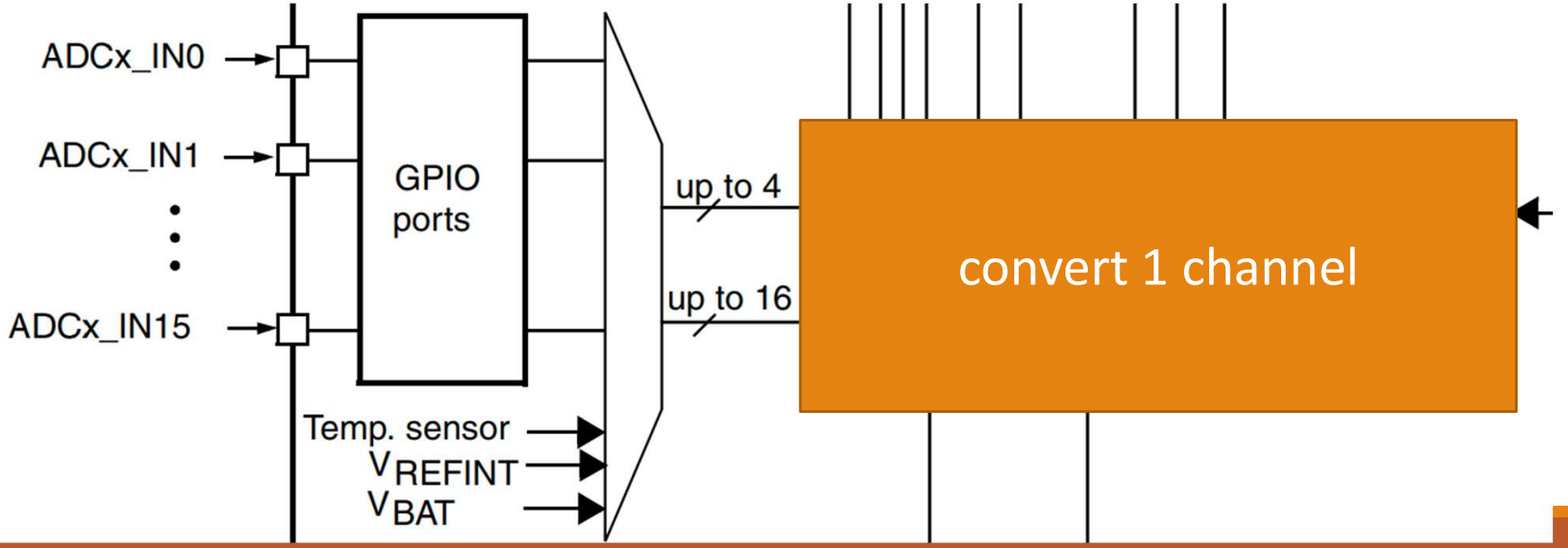
*ยิ่ง sampling rate สูง ก็จะได้จำนวน data เพิ่มขึ้น ต่อวินาที

*ขึ้นกับ software ที่เขียน

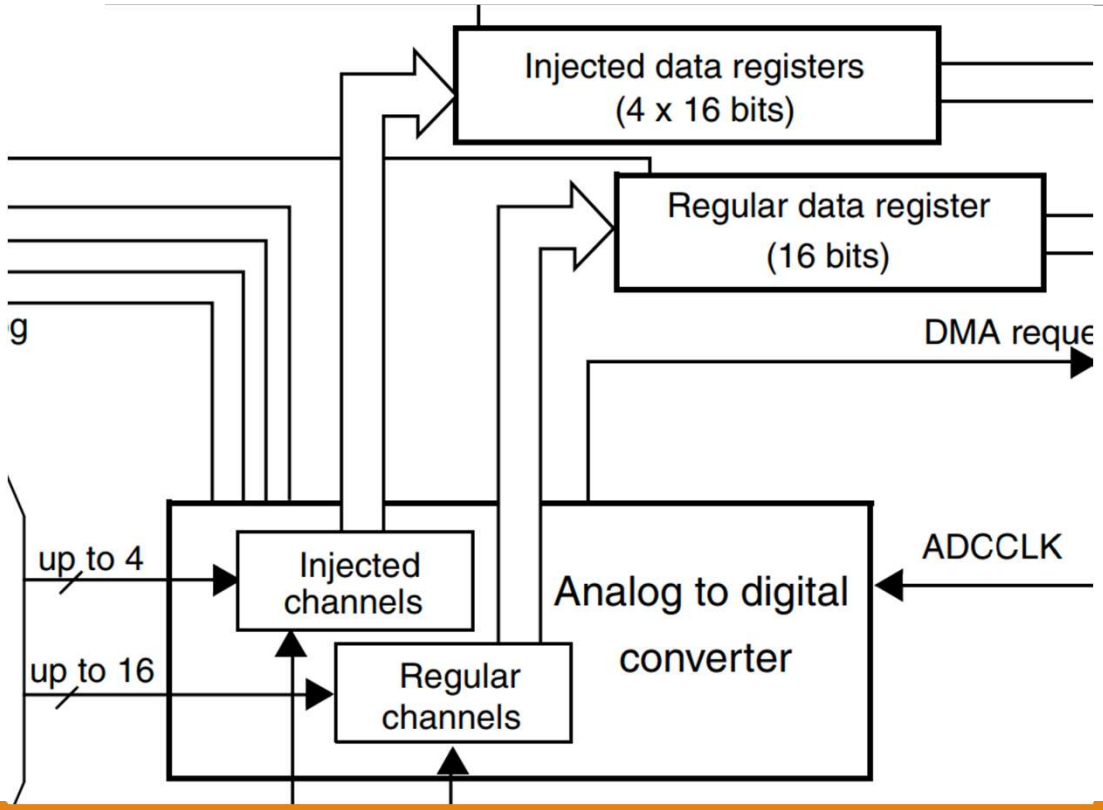
use 1 channel = 2.4MSPS / Channel
use 16 channel = 150kSPS/Channel

ADC – channel

- 1×12-bit, 2.4 MSPS A/D converter: up to 16 channels



ADC – channel – STM32



Injected – High priority Channel ใช้เวลาต้องการให้อ่านค่า ณ เวลานั้นๆ เลย

Regular – Normal use

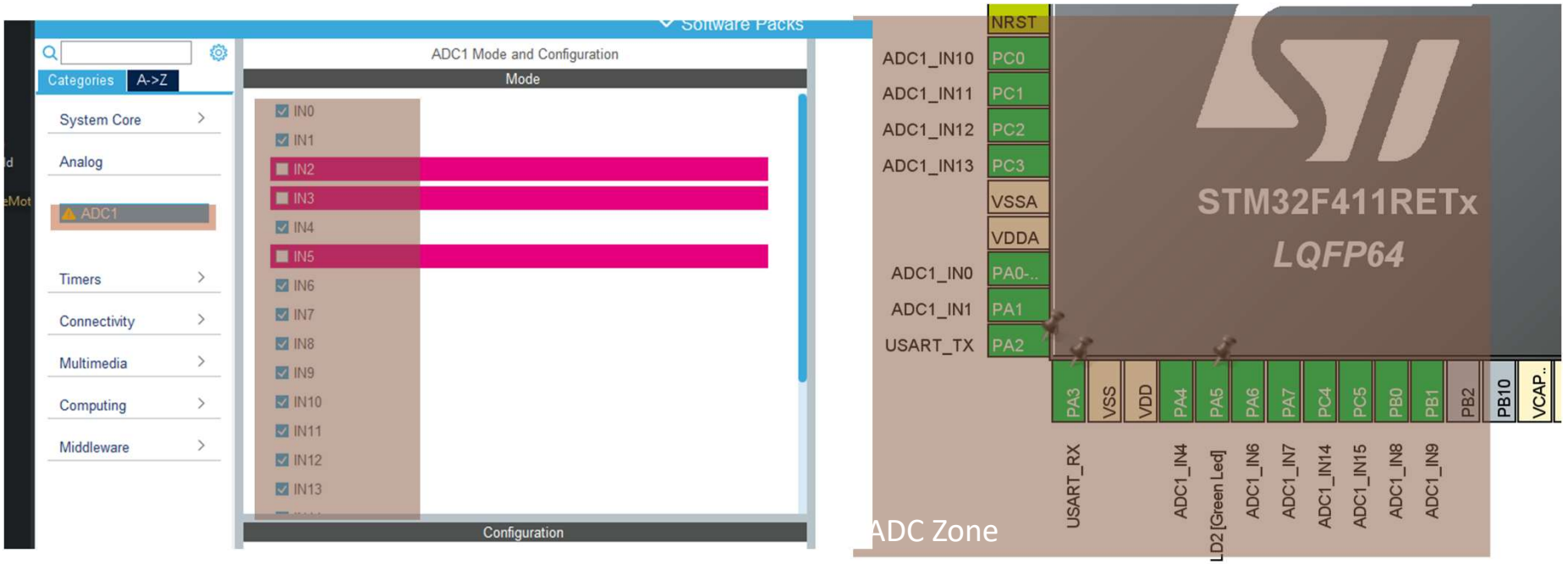
ค่าที่อ่านได้จากทั้งสองแบบข้างต้นจะเก็บไว้ต่างที่กัน

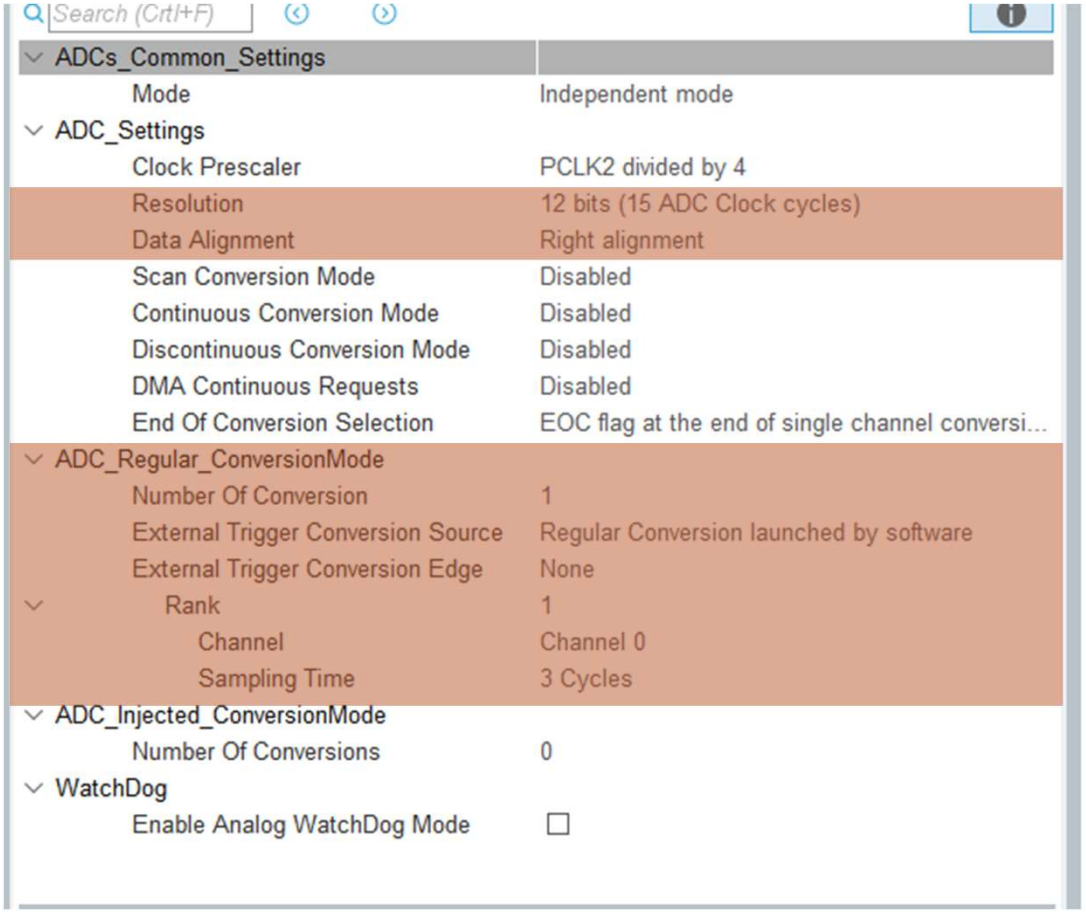
ADC - Clock

-เป็นหน่วยเวลา ของ ADC มักจะใช้ แยกกับ CPU และ ช้ากว่าพอสมควร

-ยิ่ง clock สูง ยิ่งทำงานเร็วขึ้น แต่แลกกับ Noise และ error ที่เพิ่มขึ้น

ADC - SETTING UP





ปรับ resolution /Data alignment

Data alignment

รูปแบบ output ข้อมูล12บิตว่าให้ชิดฝั่งไหนในตัวแปร16บิต

เช่น ถ้าข้อมูลคือ 0b 1111 1111 1111

left คือข้อมูลในรูป 0b 1111 1111 1111 0000

Right จะได้ 0b 0000 1111 1111 1111

ใช้ใน DMA โหมด /Scan mode เท่านั้น

DMA โหมด???? – Incomming , Next week

But For now.

CPU สามารถ เรียกใช้peripheral ต่างๆ กันได้ โดยทั่วไป 3 แบบ

1 Polling - CPU สั่งงาน peripheral **แล้ว cpu วนถามว่า เสร็จยัง ๆ ไปเรื่อย ๆ จนกว่า จะเสร็จ**แล้วจึงอ่านค่า

- ข้อดี - ข้อมูลได้ทันทีที่ peripheral ประมวลผลเสร็จ ทำให้ระบบไม่Lag
- ข้อเสีย – CPU แทบจะไม่สามารถทำอย่างอื่นได้ระหว่างรอpolling ทำให้เปลืองCPU time โดยเปล่าประโยชน์

2 Intterrupt (IT) – CPU สั่งงานแล้วไปทำอย่างอื่นก่อน พอ peripheral เสร็จ จะทักไปหยุดงานที่cpu ทำอยู่เพื่อให้อ่านค่าจาก peripheral ก่อน

- ข้อดี – CPU สามารถทำอย่างอื่นรอได้ ระหว่าง peripheral ทำงาน , ข้อมูลที่ได้ไม่ lag(ขึ้นกะการเขียนcode)
- ข้อเสีย – ถ้าเกิด Intterrupt บ่อยเกินไป หรือ ยาวเกินไป จะกลายเป็นว่า cpu ไม่ได้ทำงานของตัวเอง

DMA โหมด???? – Incomming , Next week

3 DMA – CPU สั่งงาน peripheral พิเศษ ที่เรียกว่า DMA เพื่อให้ DMA คุยกะ peripheral ที่ต้องการอีกที โดย DMA จะเป็นคนสั่ง และรวบรวมข้อมูล และค่อยส่งให้CPU อีกทีพอได้ข้อมูลจำนวนหนึ่ง

- ข้อดี – CPU สามารถทำอย่างอื่นรอได้ ระหว่าง peripheral ทำงาน , สามารถจัดการข้อมูลจำนวนมากๆได้
- ข้อเสีย – ถ้าใช้กับข้อมูลน้อยๆ ที่ไม่ต่อเนื่อง จะช้ากว่า IT

ADC – Common Command

HAL_ADC_ConfigChannel(...);

ตั้งค่าช่องที่จะใช้แปลง

HAL_ADC_Start(…); , HAL_ADC_Start_IT(…); HAL_ADC_Start_DMA(…);

เริ่มการแปลงในแบบต่างๆ

HAL_ADC_PollForConversion(…);

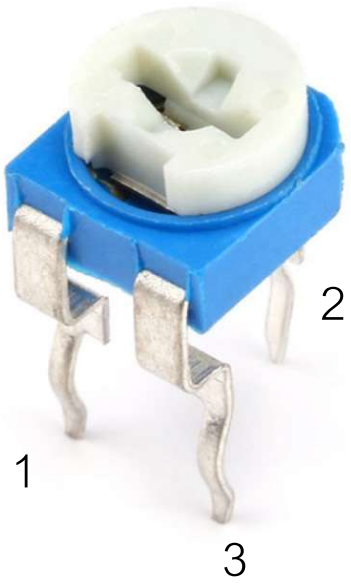
รอการทำงานในแบบ polling

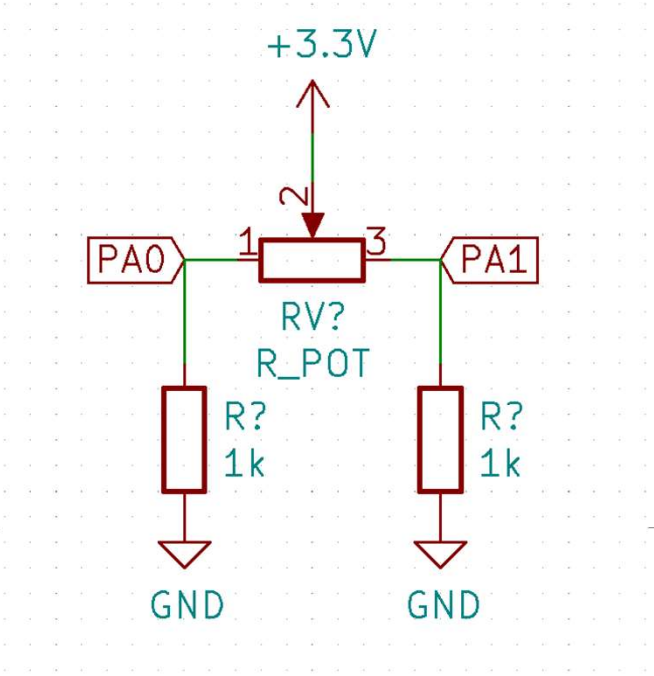
HAL_ADC_GetValue(…);

อ่านค่า ADC

HAL_ADC_Stop(…);

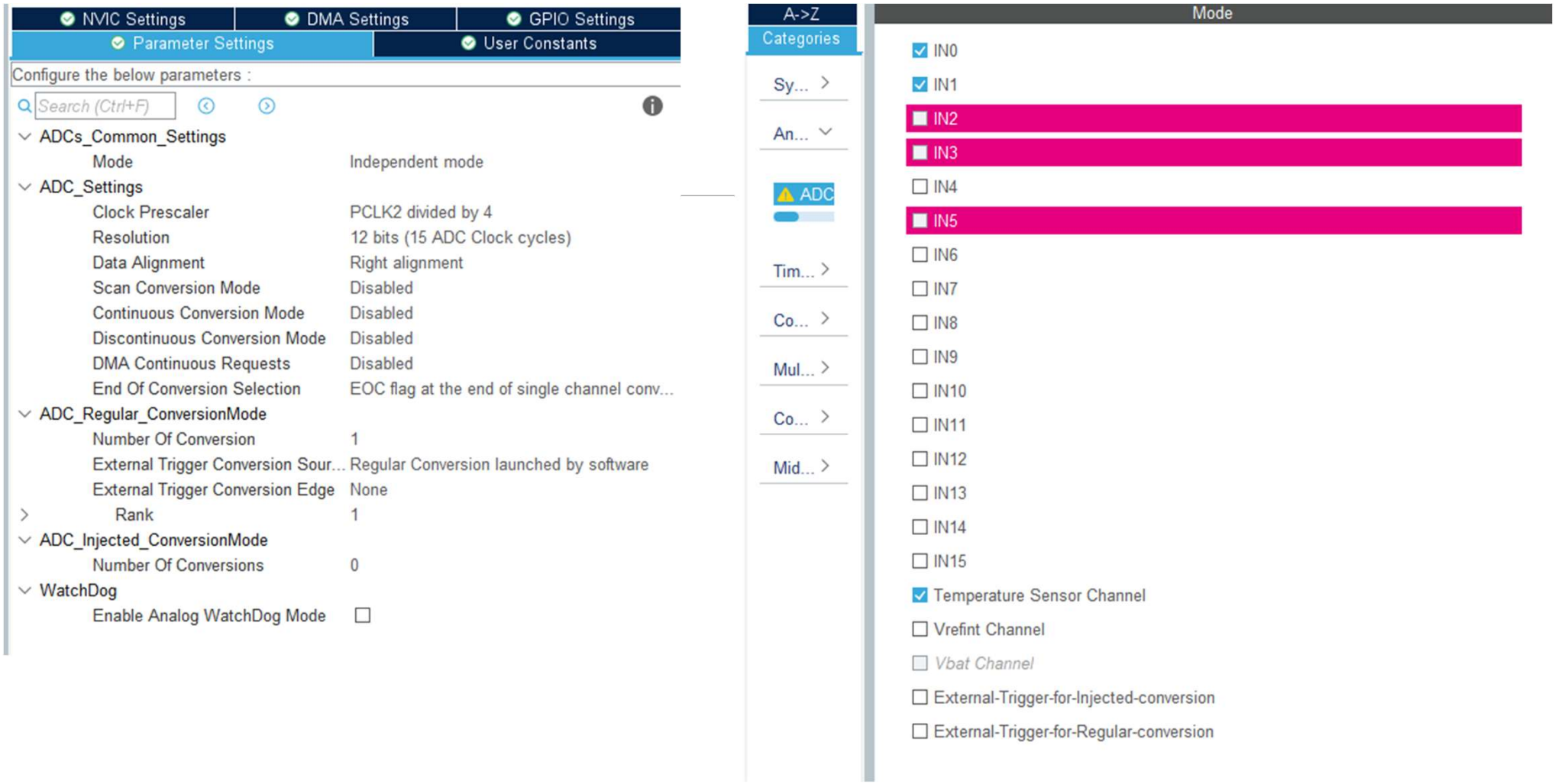
หยุดการทำงาน ADC


2
1
3


+3.3V
2
PA0
1
3
PA1
RV?
R_POT
R?
1k
R?
1k
GND
GND

Example 3

-สร้างโปรแกรมอ่านค่าanalog จาก 3 แหล่ง PA0,PA1,Temp Sensor


NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
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
Disabled
Continuous Conversion Mode
Disabled
Discontinuous Conversion Mode
Disabled
DMA Continuous Requests
Disabled
End Of Conversion Selection
EOC flag at the end of single channel conv...
ADC_Regular_ConversionMode
Number Of Conversion
1
External Trigger Conversion Sour...
Regular Conversion launched by software
External Trigger Conversion Edge
None
Rank
1
ADC_Injected_ConversionMode
Number Of Conversions
0
WatchDog
Enable Analog WatchDog Mode
A->Z
Categories
Sy...
An...
ADC
Tim...
Co...
Mul...
Co...
Mid...
Mode
IN0
IN1
IN2
IN3
IN4
IN5
IN6
IN7
IN8
IN9
IN10
IN11
IN12
IN13
IN14
IN15
Temperature Sensor Channel
Vrefint Channel
Vbat Channel
External-Trigger-for-Injected-conversion
External-Trigger-for-Regular-conversion

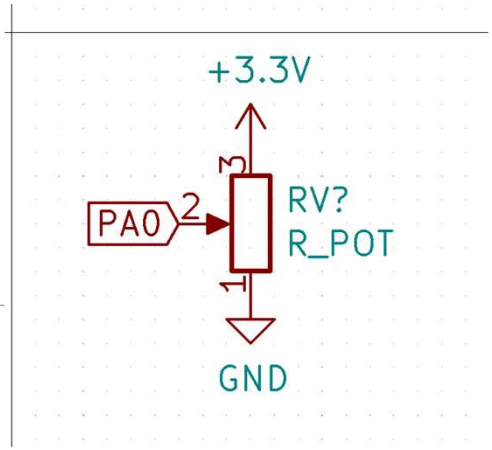
```c
/* USER CODE BEGIN PV */
typedef struct
{
    ADC_ChannelConfTypeDef Config;
    uint16_t data;
}ADCStructure;

ADCStructure ADCChannel[3];
/* USER CODE END PV */
```

```c
/* USER CODE BEGIN 2 */
ADCConfigInit();
/* USER CODE END 2 */

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

  /* USER CODE BEGIN 3 */

    //function call ADC
    ADCPollingMethodUpdate();
}
/* USER CODE END 3 */
```

```c
/* USER CODE BEGIN 4 */

void ADCConfigInit()
{
    //Config ADC Chennel PA0
    ADCChannel[0].Config.Channel = ADC_CHANNEL_0;
    ADCChannel[0].Config.Rank = 1;
    ADCChannel[0].Config.SamplingTime = ADC_SAMPLETIME_3CYCLES;
    //Config ADC Chennel PA1
    ADCChannel[1].Config.Channel = ADC_CHANNEL_1;
    ADCChannel[1].Config.Rank = 1;
    ADCChannel[1].Config.SamplingTime = ADC_SAMPLETIME_3CYCLES;
    //Config ADC Chennel Temp
    ADCChannel[2].Config.Channel = ADC_CHANNEL_TEMPSENSOR;
    ADCChannel[2].Config.Rank = 1;
    ADCChannel[2].Config.SamplingTime = ADC_SAMPLETIME_3CYCLES;

}
```

```c
void ADCPollingMethodUpdate()
{
    for (int i = 0; i < 3; i++) {
        //setting channal of ADC
        HAL_ADC_ConfigChannel(&hadc1, &ADCChannel[i].Config);

        //start Sampling
        HAL_ADC_Start(&hadc1);

        //Wait for sampling and conversion
        if(HAL_ADC_PollForConversion(&hadc1, 10)==HAL_OK)
        {
            //Read data
            ADCChannel[i].data = HAL_ADC_GetValue(&hadc1);
        }
        //stop adc
        HAL_ADC_Stop(&hadc1);
    }


}



/* USER CODE END 4 */
```

exercise ADC Converter



1.สร้างตัวแปรชื่อ ADCOutputConverted และ ADCMode

2.ใช้ปุ่มไหนก็ได้ สับADCMode ระหว่าง 2 ค่า คือ

2.1 ADCMode = 0 ,ADCOutputConverted = ค่าที่อ่านได้จาก PA0 ในหน่วย mV

2.2 ADCMode = 1 , ADCOutputConverted = อุณหภูมิปัจจุบัน

3.แสดงทั้งสองตัวแปรผ่าน debugmode Live Expresstion

Hint 8. Calculate the temperature using the following formula:

Temperature (in °C) = {(V_{SENSE} – V_{25}) / Avg_Slope} + 25

Where:

– V_{25} = V_{SENSE} value for 25° C

– Avg_Slope = average slope of the temperature vs. V_{SENSE} curve (given in mV/°C or µV/°C)

Refer to the datasheet's electrical characteristics section for the actual values of V_{25} and Avg_Slope.