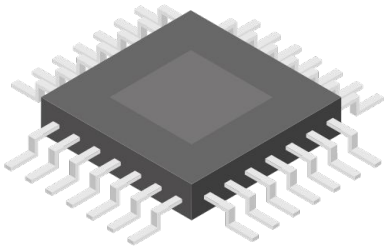
PADO

Labs

Microcontroladores



Prof.º: Pablo Jean Rozário

pablo.jean@padotec.com.br

/in/pablojeanrozario

https://github.com/Pablo-Jean

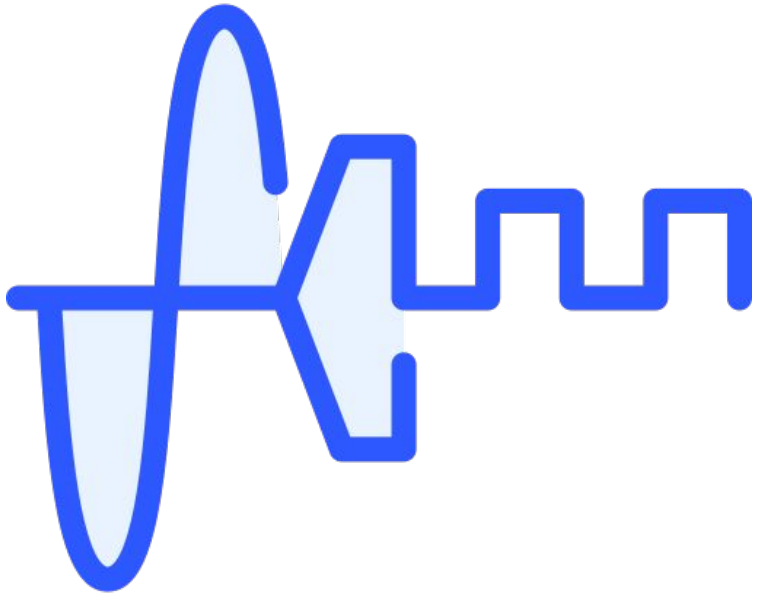
Conversor Analógico-Digital

Índice da Aula #4

- Introdução
- Parâmetros do ADC
  - Resolução, Sample and Hold, Sample Rate.
  - Arquiteturas
- ADC no STM32
- Localização dos canais
- Modos de Leitura
- Experimentação
- Interrupções e DMA
- Lista de Exercícios #4

PADO Labs

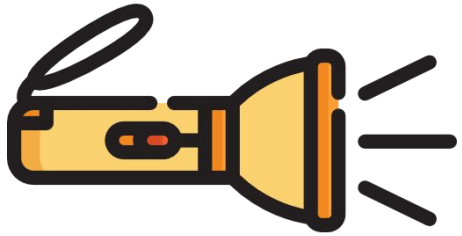
Analog-Digital Converter
ADC

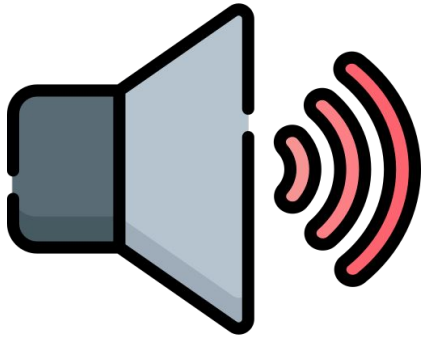


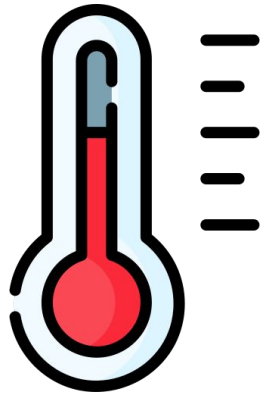
ADC - Introdução

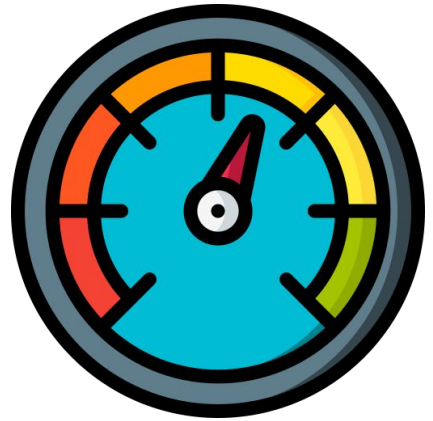
O periférico *Analog-Digital Converter*, ou apenas ADC, é um dispositivo que tem a finalidade de converter um sinal analógico para digital. Permitindo que o microcontrolador possa compreender e realizar tarefas com estes valores.

Todas as informações do mundo são analógicas.





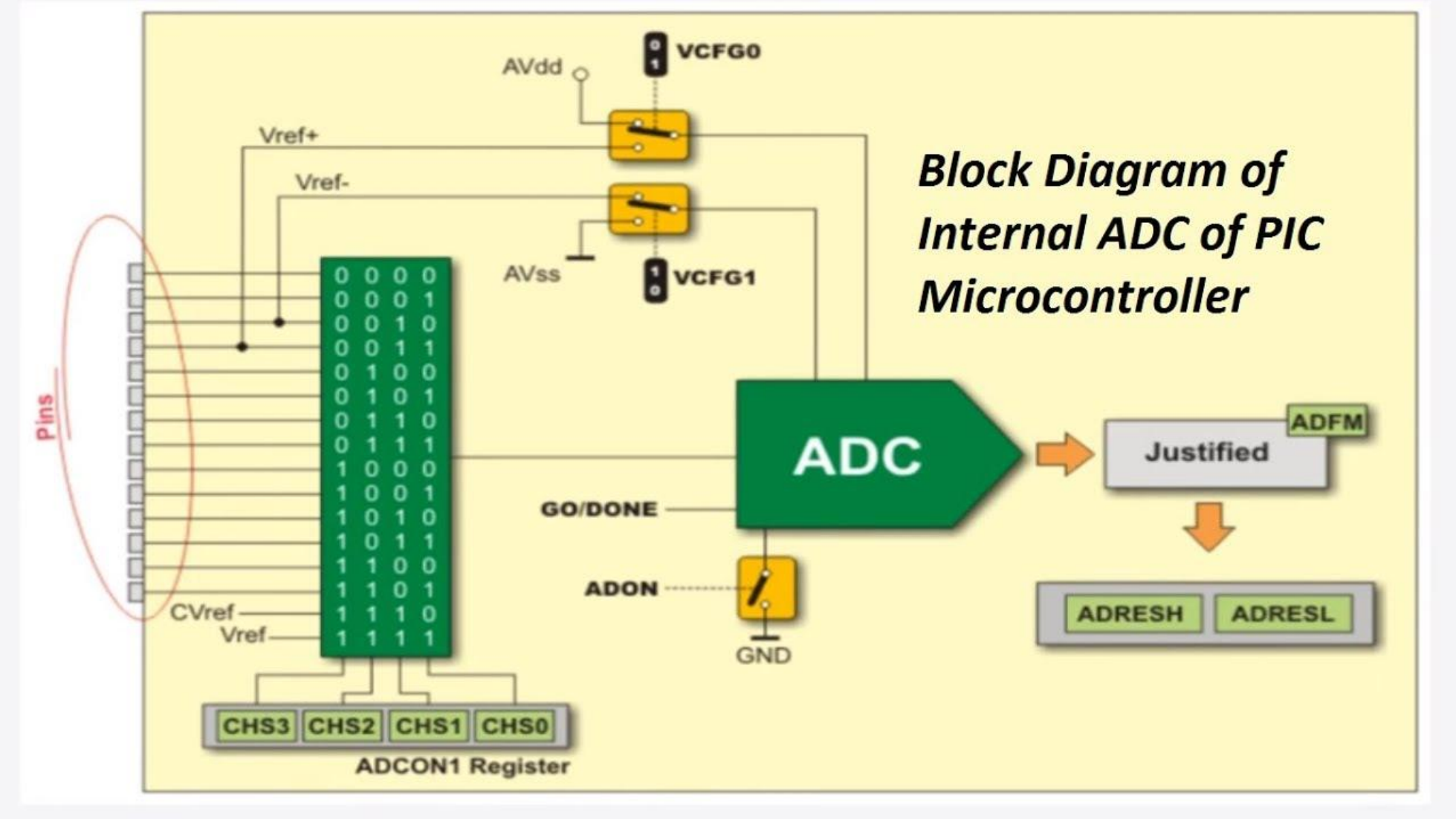




ADC

PADO Labs

ADC - Estrutura



ADC - Quantização

Refere-se ao processo de **transformar** a amostra de um sinal **analógico** em uma representação de **números finitos e inteiros**.

Inicia em **0**, representa o valor da **referência negativa** (GND em geral)

Vai até o valor da **potência de 2** que corresponde a quantidade de *bits* do conversor, representa a **referência positiva**.

ADC - Quantização

Por exemplo, um conversor de 10*bits* representa, com referência em 0V e 3.3V:

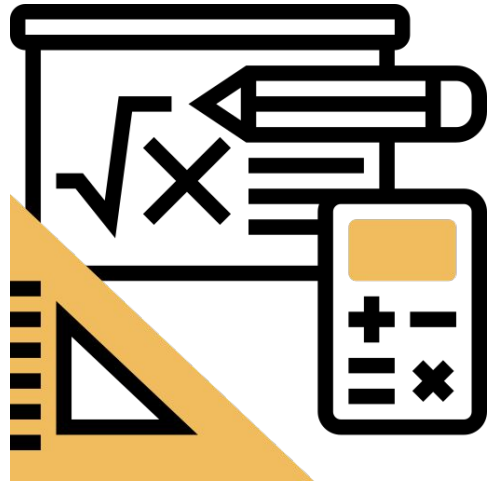
0 (0x0) = 0V
1023 (0x3FF) = 3.3V

E valores de tensão entre as referências são proporcionais e lineares.

ADC - Quantização Equação

$$Q = (2^N - 1) * \frac{V_o - V_{r-}}{V_{r+} - V_{r-}}$$

- Q valor quantizado do sinal amostrado;
- N número de bits do conversor;
- V_o tensão de entrada do ADC;
- V_{r-} tensão de referência negativa;
- V_{r+} tensão de referência positiva.

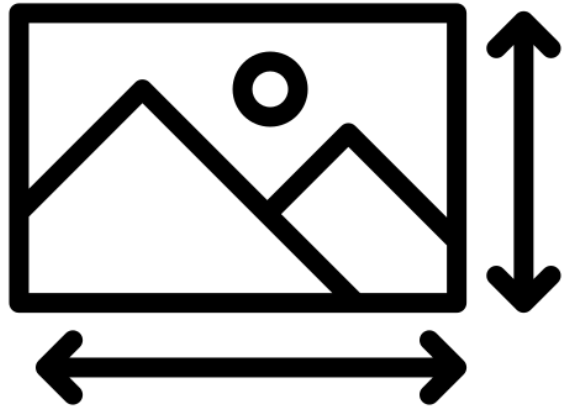


ADC - Parâmetros

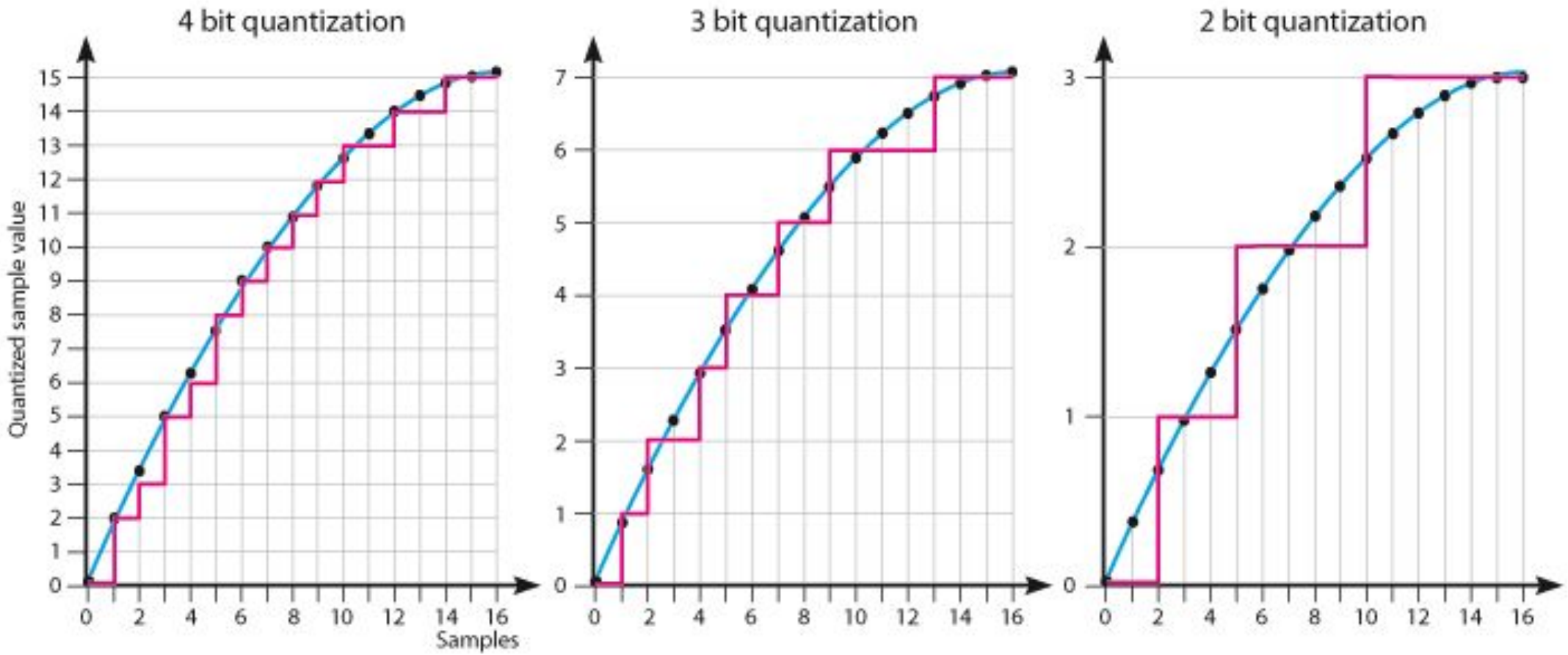
-> Resolução

Trata-se da quantidade de *bits* que o conversor utiliza para quantizar o valor analógico. É o *N* que vimos na quantização.

Naturalmente, quanto **maior melhor**.
Mas maiores resoluções, em geral, implicam em tempo de conversão **maior**.



ADC - Parâmetros



ADC - Equação

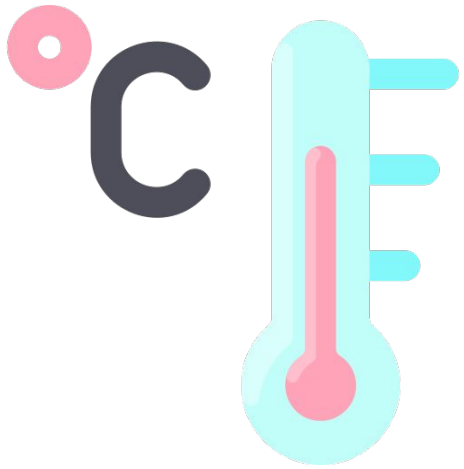
$$V_r = (V_{r+} - V_{r-}) * \frac{ADC_{result}}{2^N - 1} + V_{r-}$$

Se a referência negativa for o GND e a positiva for o VDD, podemos simplificar a equação:

$$V_r = V_{DD} * \frac{ADC_{result}}{2^N - 1}$$

ADC - Impacto da Resolução

Sensor de temperatura que varia de **0ºC à 50ºC**, condicionado a trabalhar entre **0V** e a tensão de alimentação **VDD**.



Resolução (bits)	Passo (ºC/bit)
6	0,781
8	0,195
10	0,049
12	0,012
16	763×10^{-6}
20	$47,7 \times 10^{-6}$
24	$2,98 \times 10^{-6}$

ADC - Parâmetros

Sample and Hold Time

Tempo necessário para o ADC efetuar uma conversão, em **ciclos de clock**.

Alguns microcontroladores permitem configurar este tempo, pois quanto maior esse tempo, as leituras tendem a ser mais estáveis.

Este tempo impacta diretamente no **Sample Rate**.

ADC - Parâmetros

Sample Rate

Taxa de amostragem de um ADC, ou seja, quantas **leituras** o mesmo pode efetuar por **segundo**.

Pode ser calculado pela equação:

$$S_{ADC} = \frac{f_{ADC}}{K_{ADC}}$$

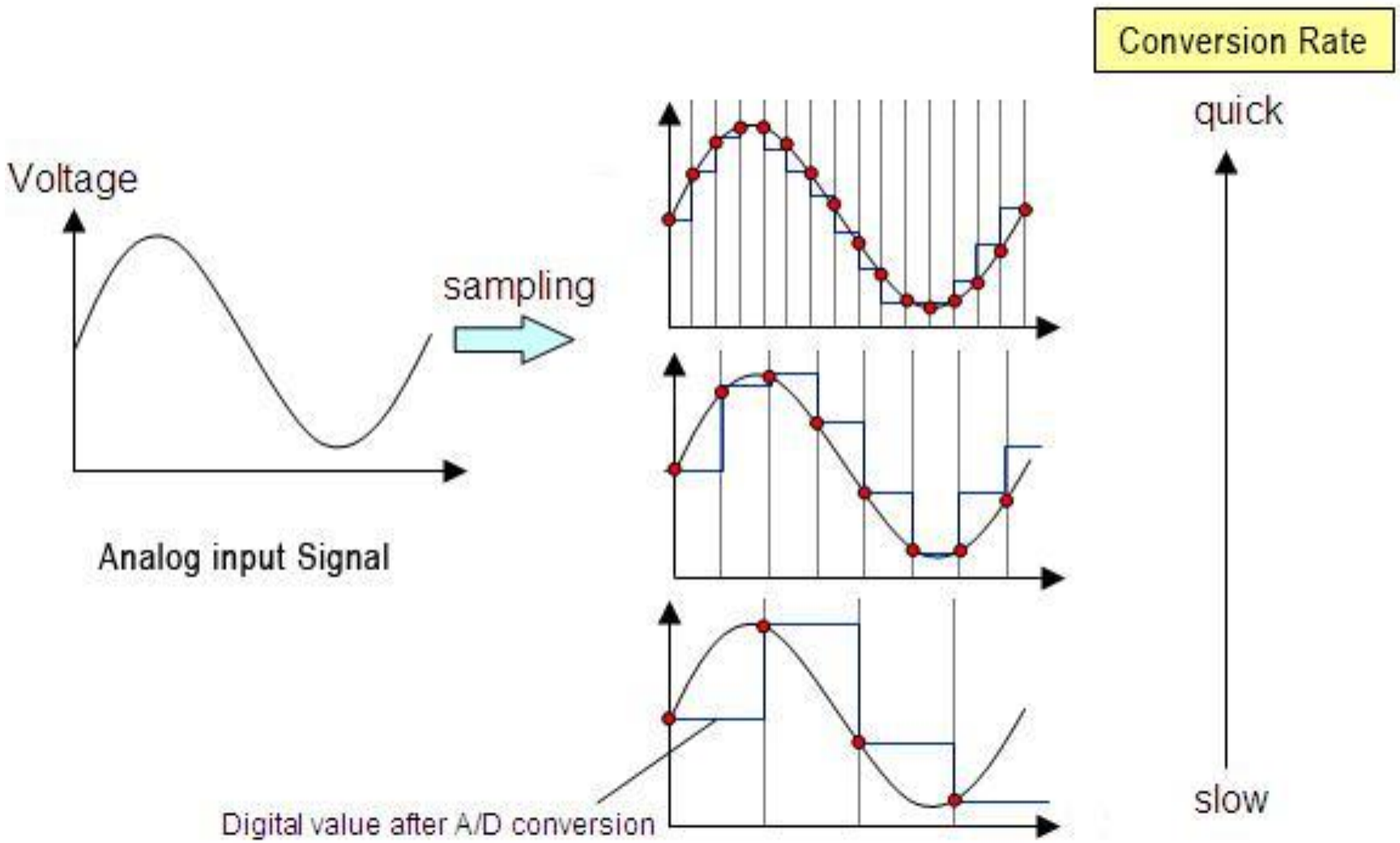
- S_{ADC} é o *sample rate* do ADC em Hz;
- f_{ADC} é a frequência do *clock* do periférico em Hz;
- K_{ADC} refere-se a quantidade de ciclos necessárias para uma leitura.

ADC - Sample Rate

Tomamos um microcontrolador com ADC com frequência de **1MHz** e tempo de conversão de **13 ciclos**.
Obteríamos um *sample rate* de **77kHz**.

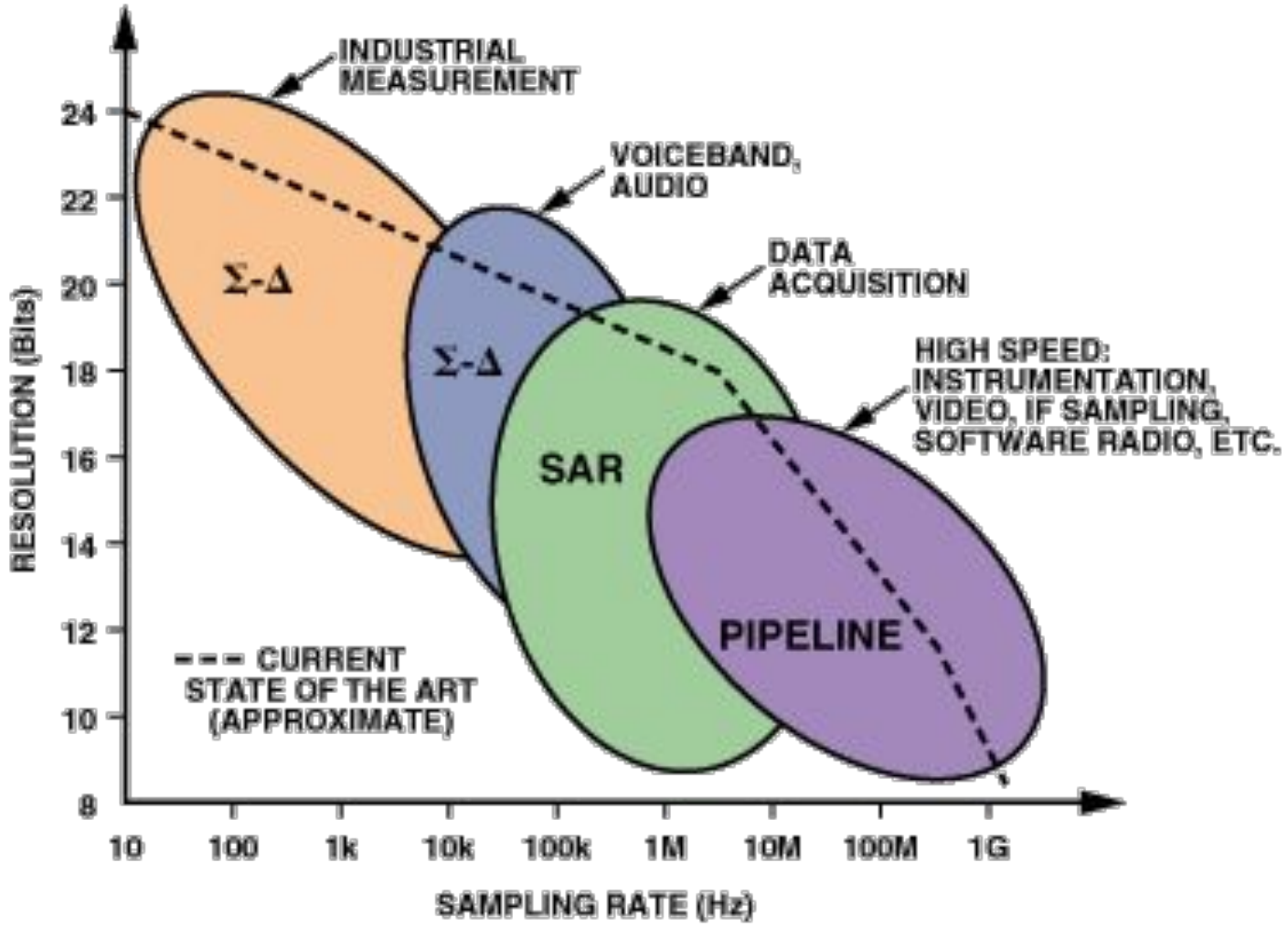
Com esta taxa poderíamos medir um sinal de frequência de até 38.5kHz.

ADC - Impacto do Sample Rate

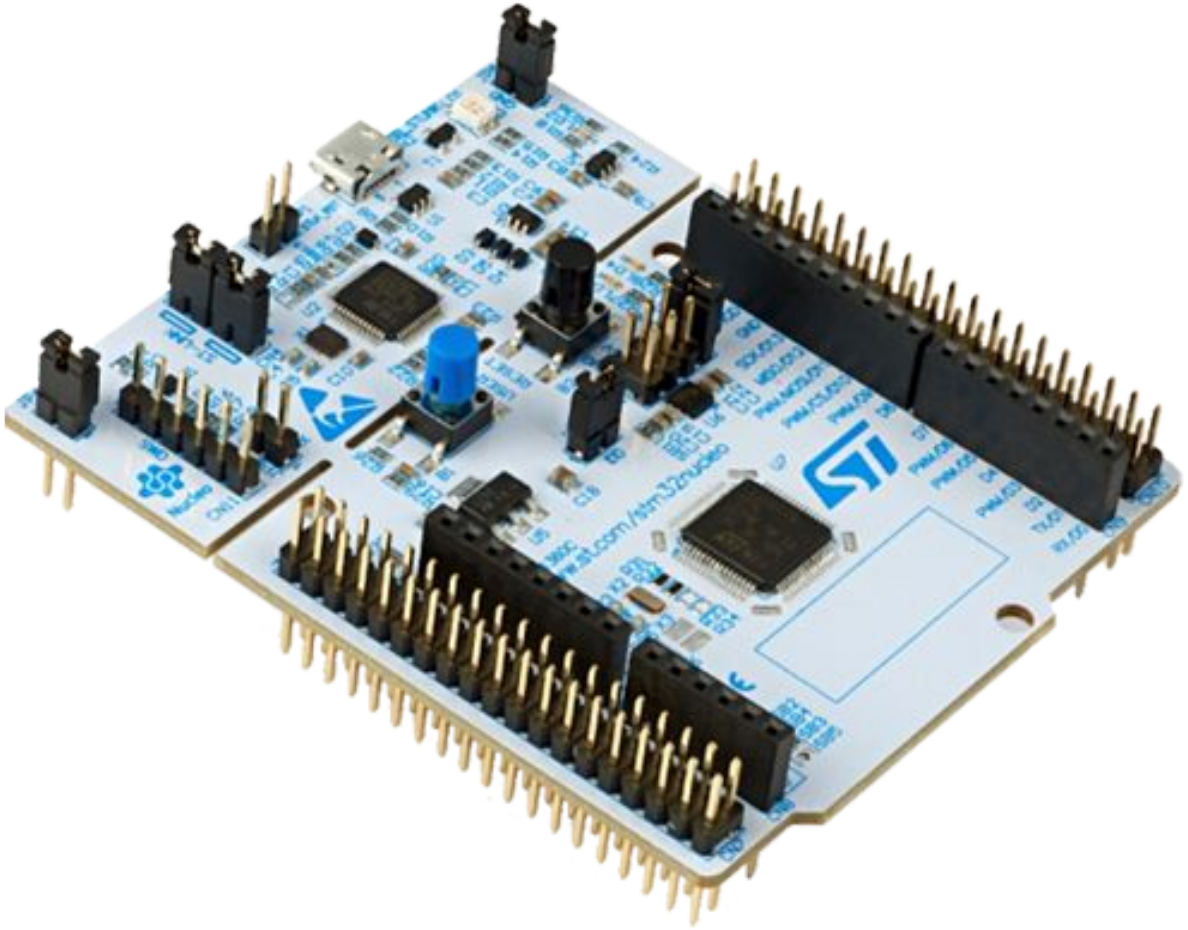


ADC - Arquiteturas

Existem diversas arquiteturas de ADC, sendo algumas delas: Sigma-Delta, Pipeline, SAR (*Successive Approximation Register*), *Ramp Compare*, Wilkinson, entre outras.

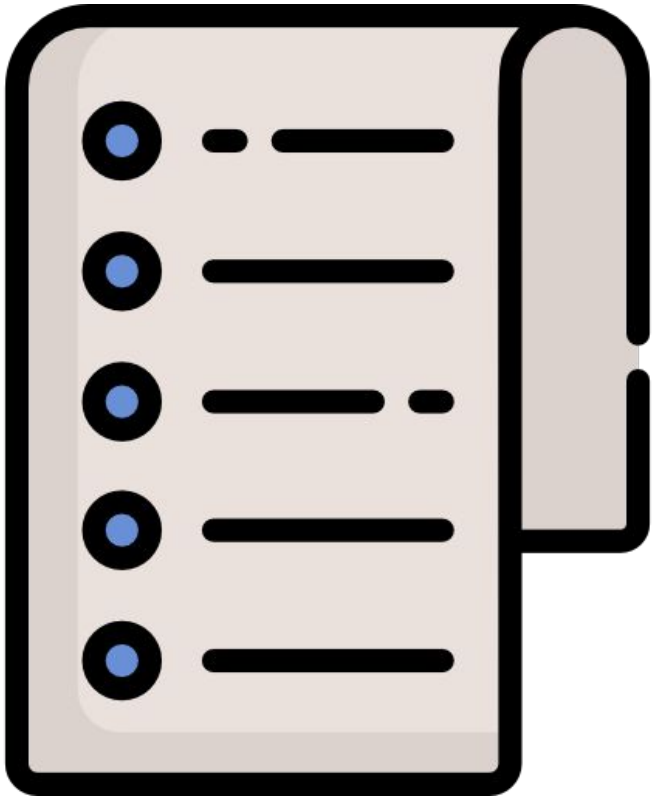


ADC no STM32G0

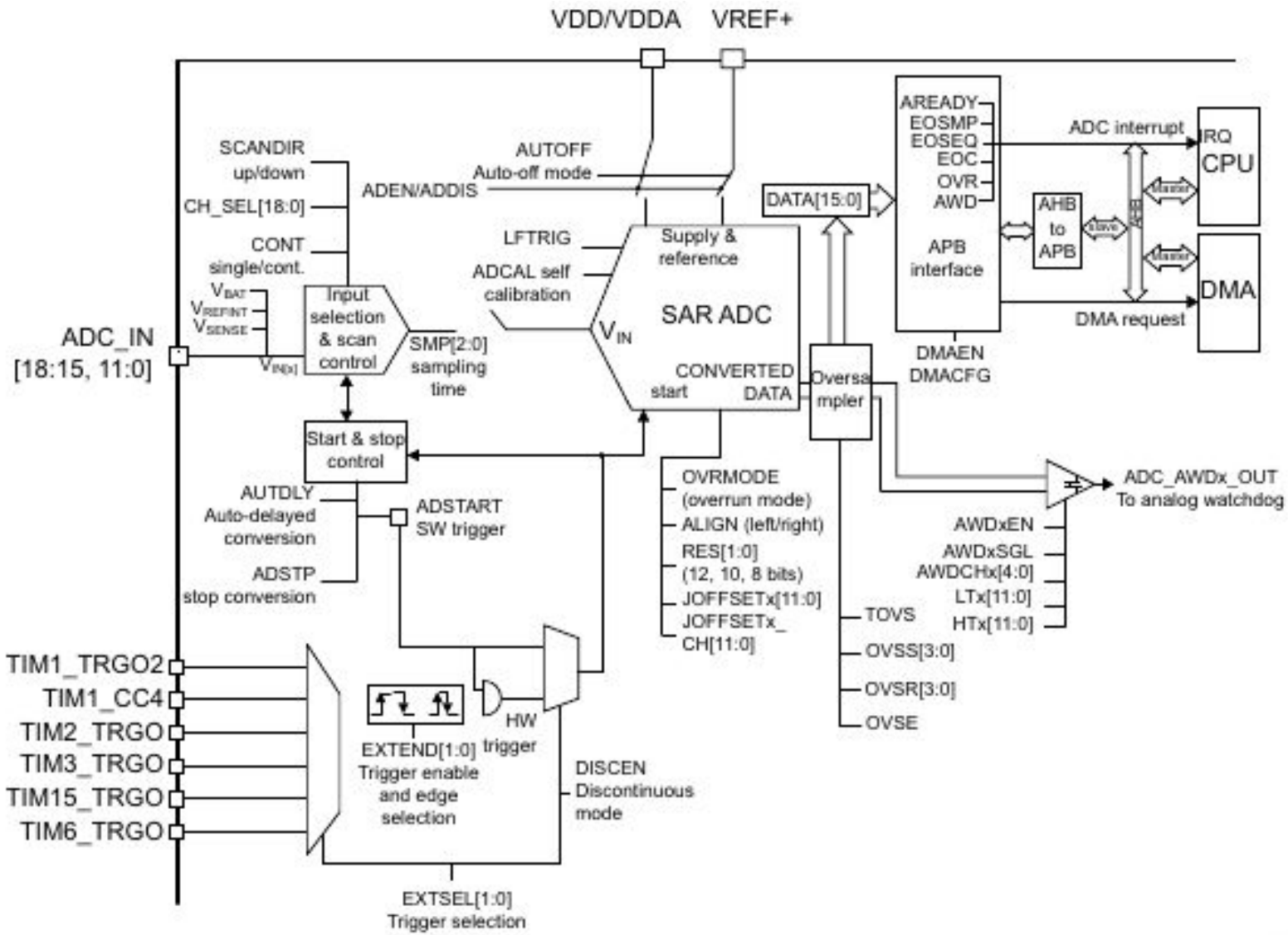


STM32G0 - Características do ADC

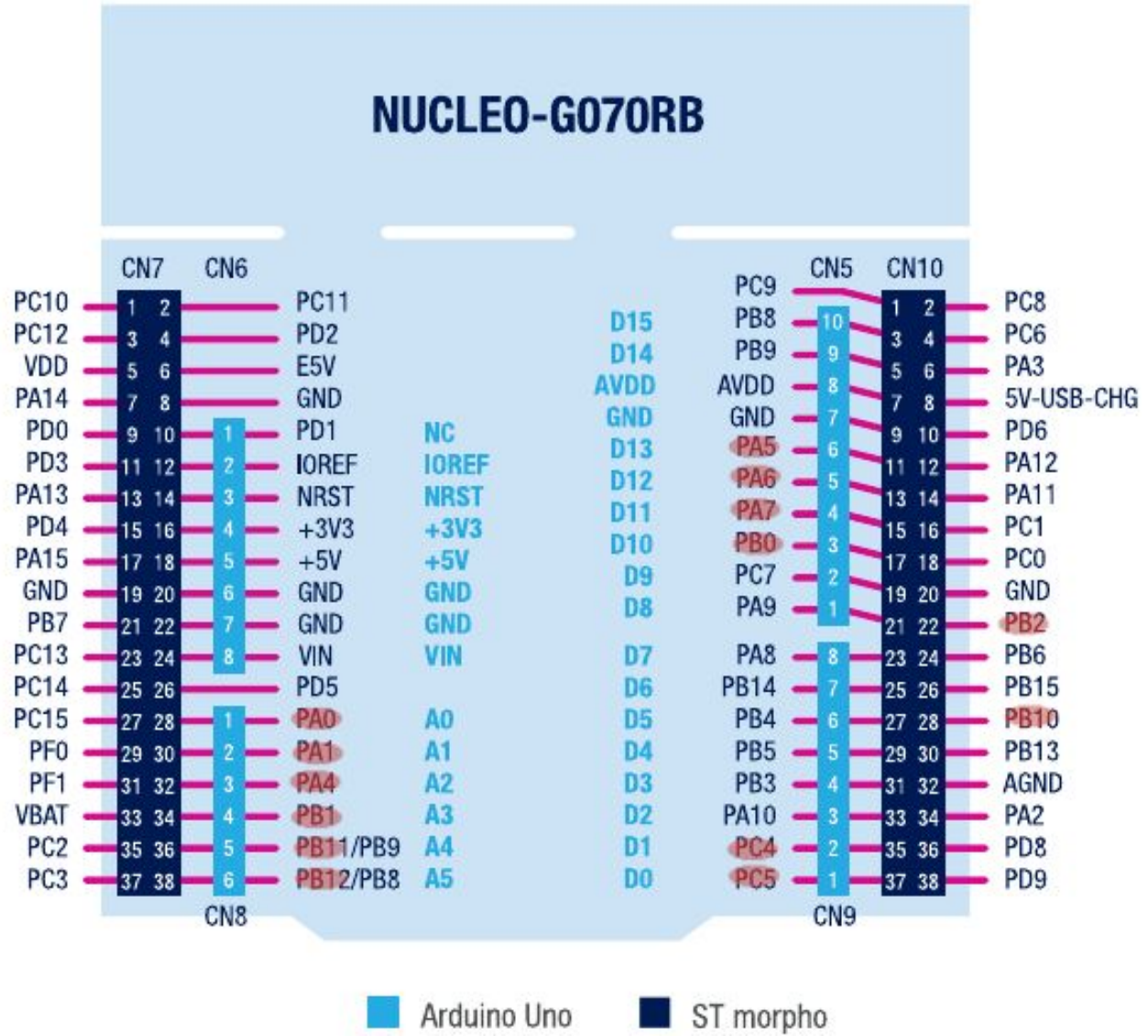
- Resolução de 12bits;
- Tempo de conv. de 400ns;
- Suporte ao DMA;
- Preparado para Low Power
- 16 canais externos
- Sensor de temperatura interno
- Tensão Vbat
- Vários modos de operação
- etc



STM32G0 - Diagrama



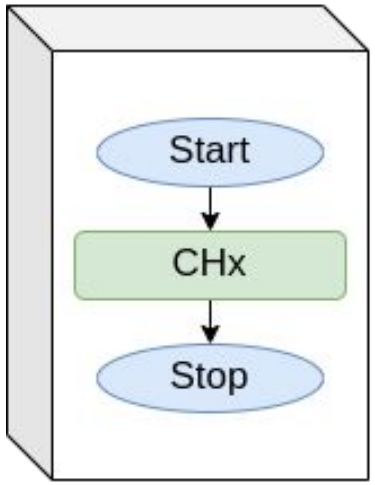
STM32G0 - Terminais



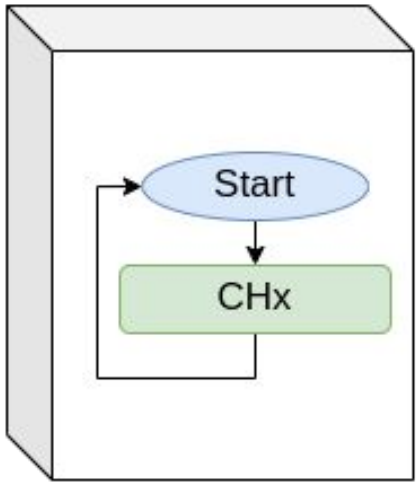
STM32G0 - Arduino Uno Canais

Terminal	Canal
A0	IN0
A1	IN1
A2	IN4
A3	IN9
A4	IN15
A5	IN16

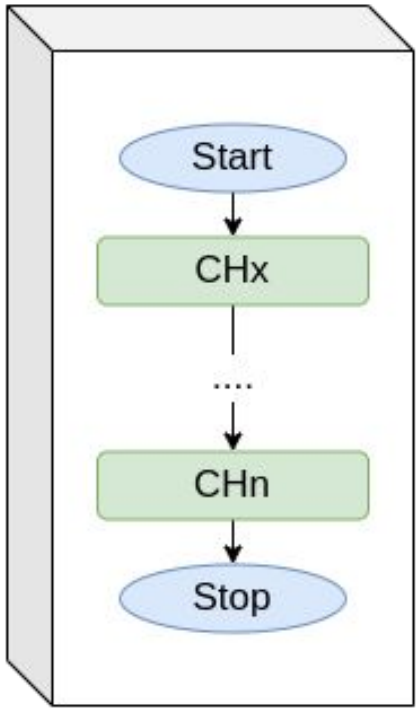
STM32G0 - Modos de Operação



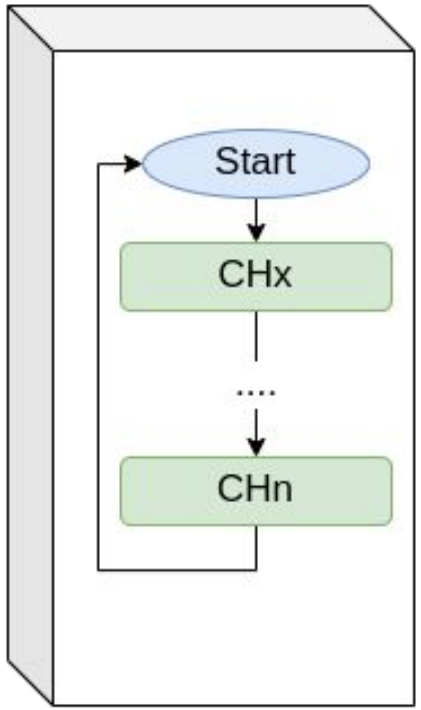
Single



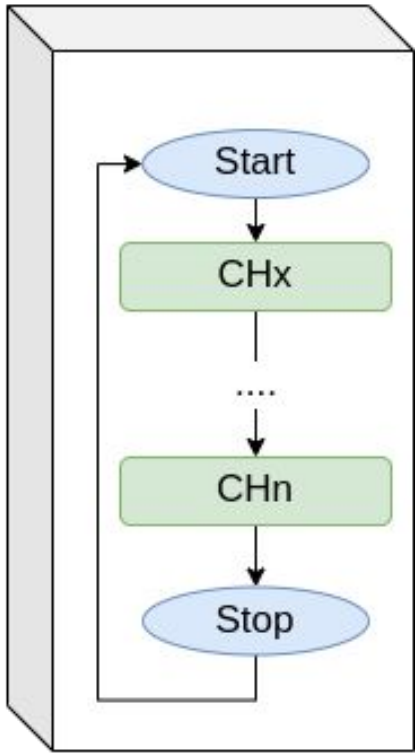
Single
Continous



Scan
Multi-
Channels



Scan
Continuous



Discontinuous
Mode

STM32G0 - Funções Para Polling

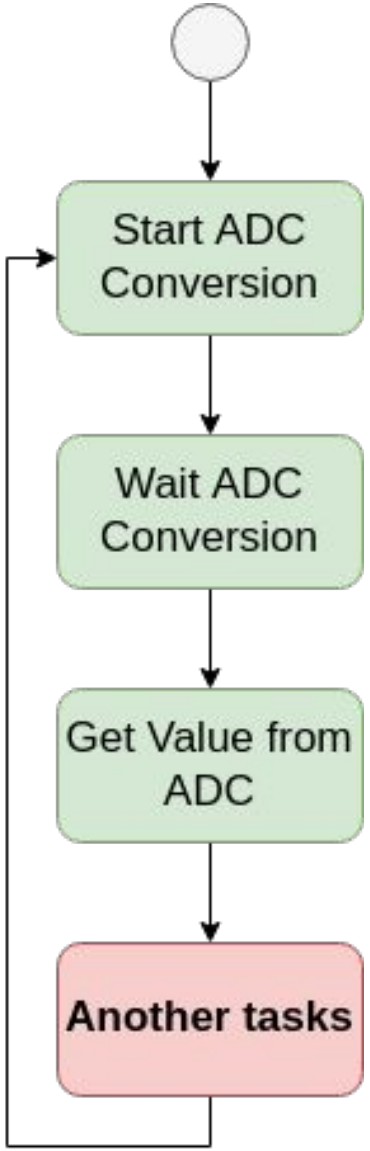
```
1   // (1) Inicia a conversao do ADC
2   HAL_ADC_Start(ADC_HandleTypeDef *hadc);
3
4   // (2) Aguarda a conclusao da conversao
5   HAL_ADC_PollForConversion(ADC_HandleTypeDef *hadc, uint32_t Timeout);
6
7   // (3) Obtem o valor lido pelo ADC
8   uint32_t value = HAL_ADC_GetValue(ADC_HandleTypeDef *hadc);
9
10  // (4) Configura o canal que sera habilitado para a conversao
11  HAL_ADC_ConfigChannel(ADC_HandleTypeDef *hadc, ADC_ChannelConfTypeDef *
    pConfig);
12
13  // (5) Para o processo de conversao e o periferico
14  HAL_ADC_Stop(ADC_HandleTypeDef *hadc);
```

STM32G0 - Experimentação

Programa que fará a leitura de um potenciômetro.
Verão como o ADC se comporta ao
ler a entrada analógica.

A seguir, explicarei sobre o uso de
interrupção e do periférico de DMA.

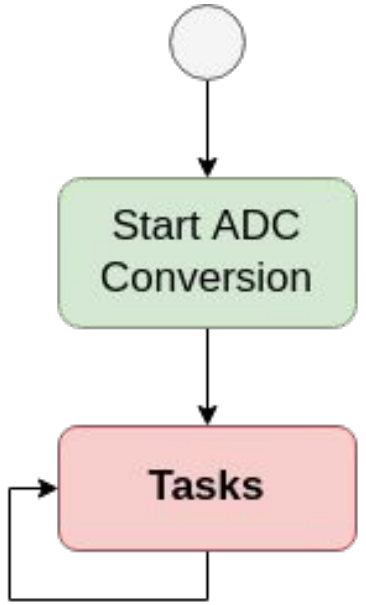
STM32G0 - Interrupção

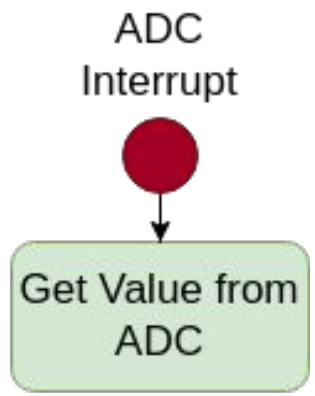


O uso de interrupção permite que o microcontrolador faça outras operações enquanto a conversão é realizada.

<< Polling

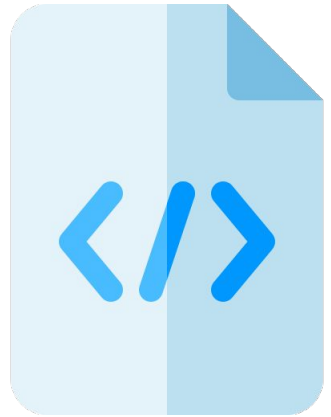
Interrupção>>





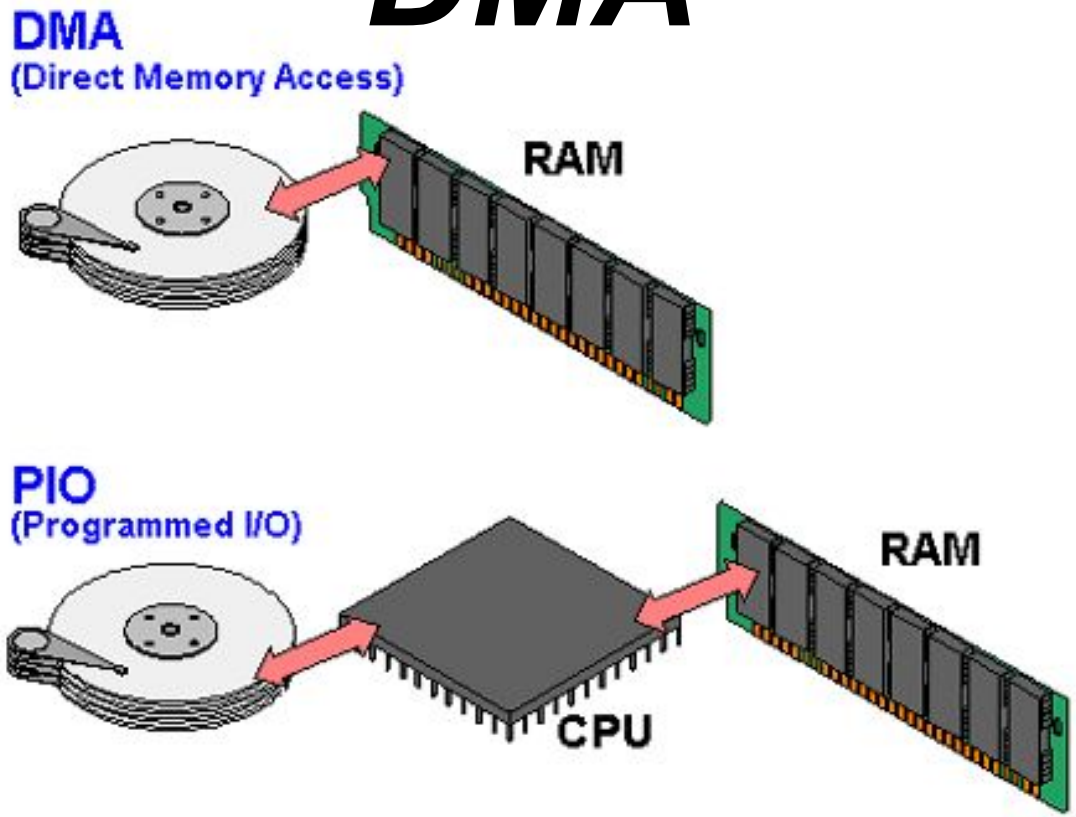
STM32G0 - Funções

```c
uint32_t value;

void main(){
    // (1) Inicia a conversao do ADC em modo de interrupcao
    HAL_ADC_Start_IT(ADC_HandleTypeDef *hadc);

}

// (2) Callback de interrupcao do ADC, chamado quando
// uma conversao ou sequencia foi finalizada por completo
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef *hadc){

    // (3) Obtem o valor lido pelo ADC
    value = HAL_ADC_GetValue(hadc);
}
```



PADO Labs

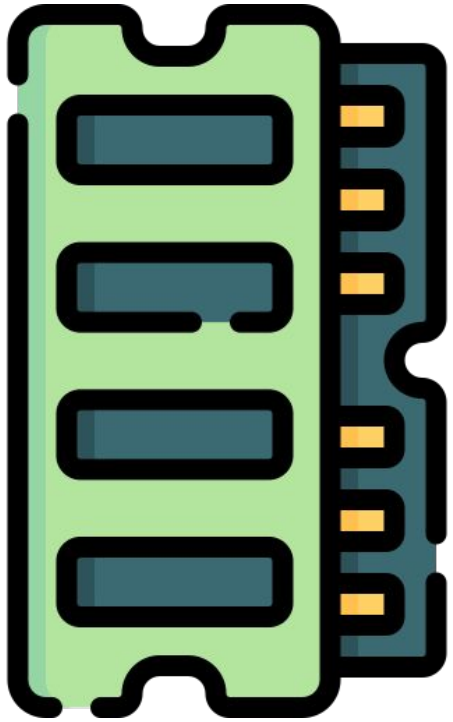
Direct Memory Access
DMA



STM32G0 - DMA

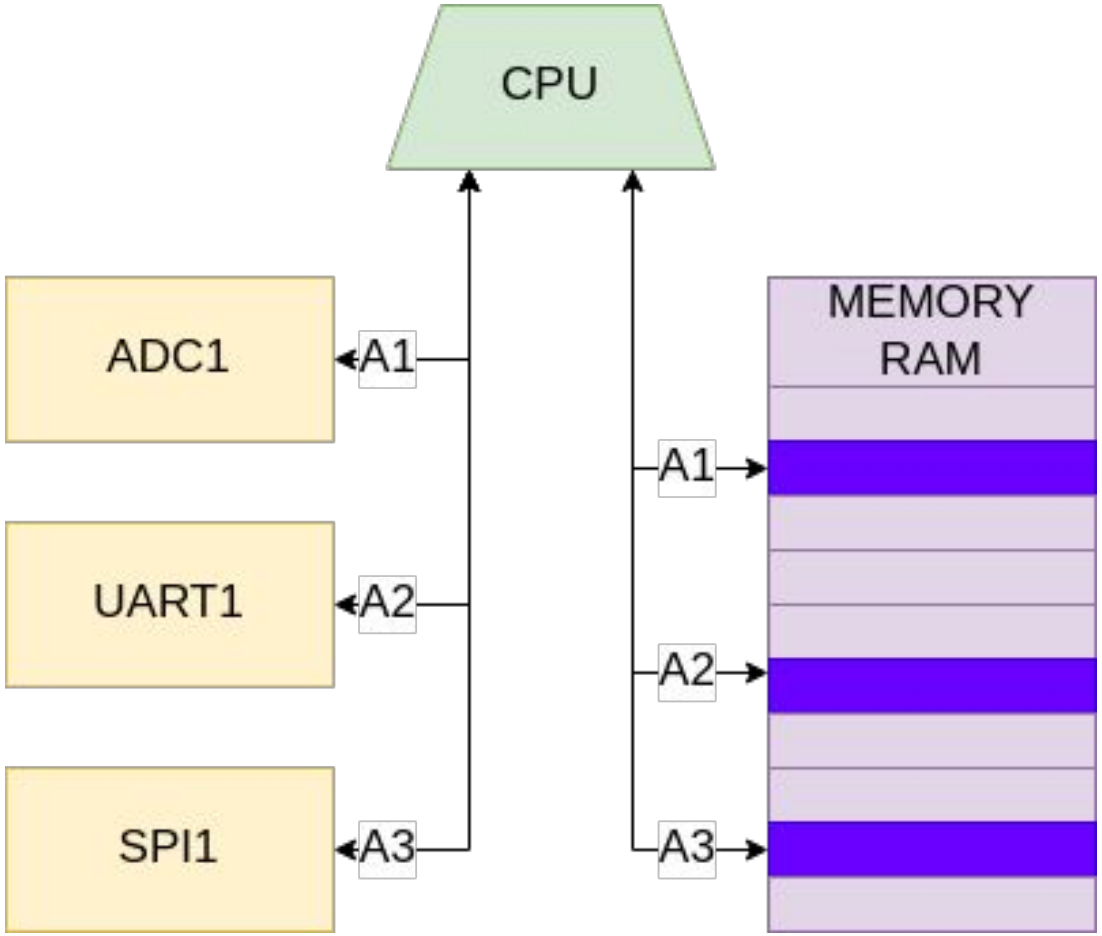
O DMA, ou *Direct Memory Access*, é uma unidade lógica que trabalha em conjunto com a CPU. Ela realiza as operações de transferência de memória, diminuindo a carga da CPU.
Ela pode transferir dados de:

Periférico <-> Memória
Memória <-> Memória



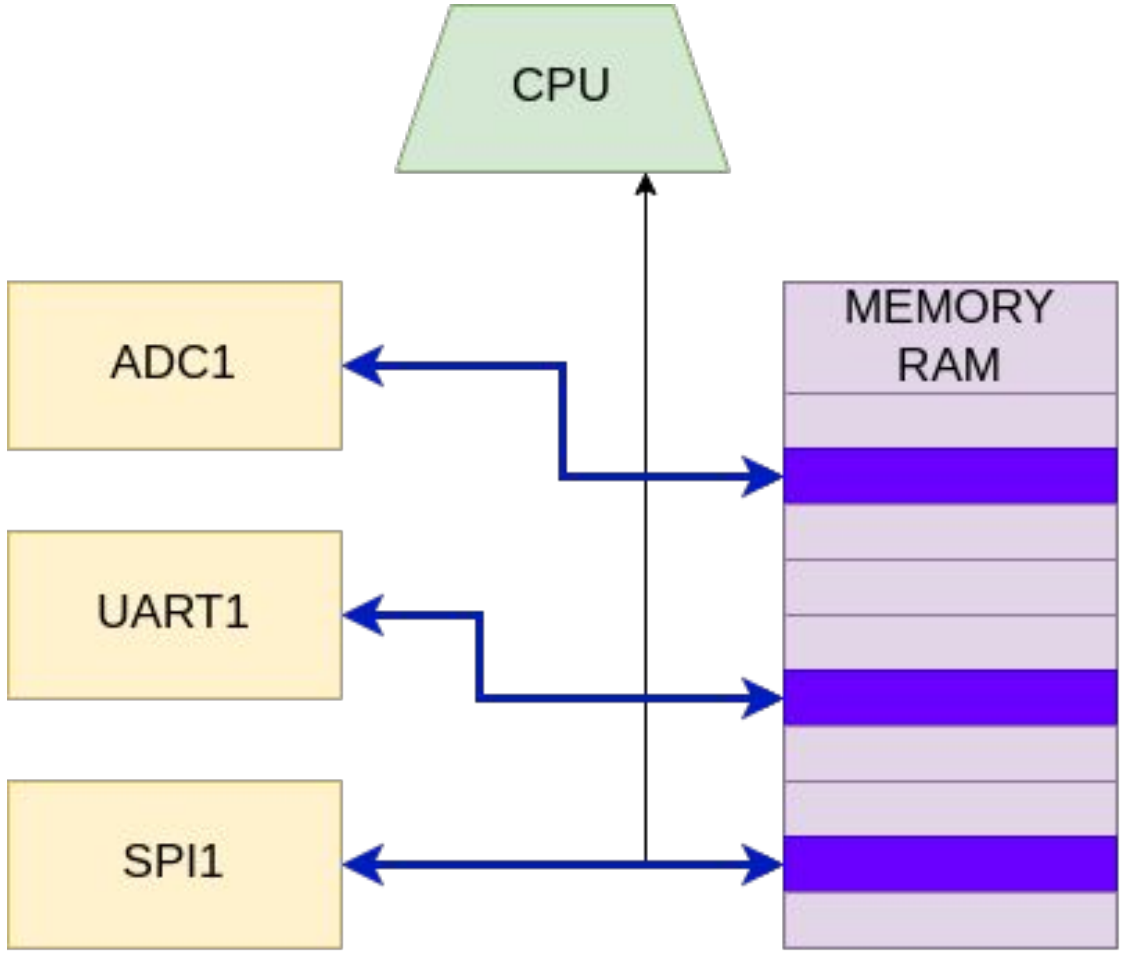
STM32G0 - DMA

Sem DMA temos o seguinte diagrama:



STM32G0 - DMA

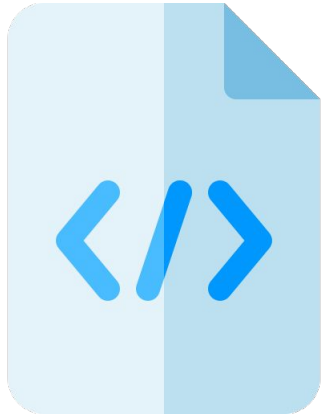
Com o uso de DMA, a memória é conectada diretamente ao periférico.



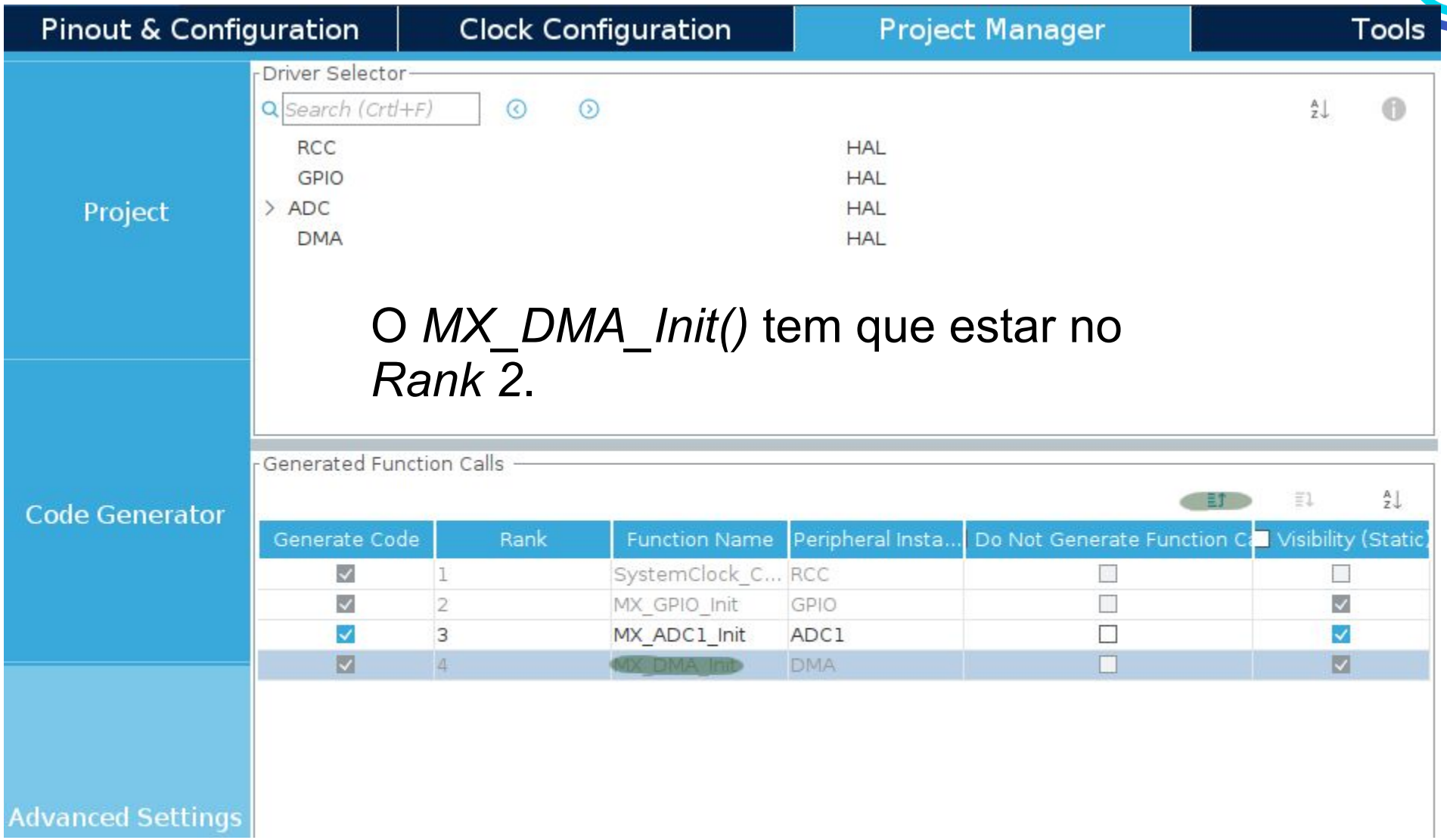
STM32G0 - Funções

```
1    uint32_t value[NUMBER_OF_SAMPLES];
2
3    // (1) Inicia a conversao do ADC em DMA.
4    HAL_ADC_Start_DMA(&hadc1, (uint32_t*)value, NUMBER_OF_SAMPLES);
```

Podemos também utilizar os Callbacks do ADC para indicar que uma (ou mais) conversões foram concluídas.



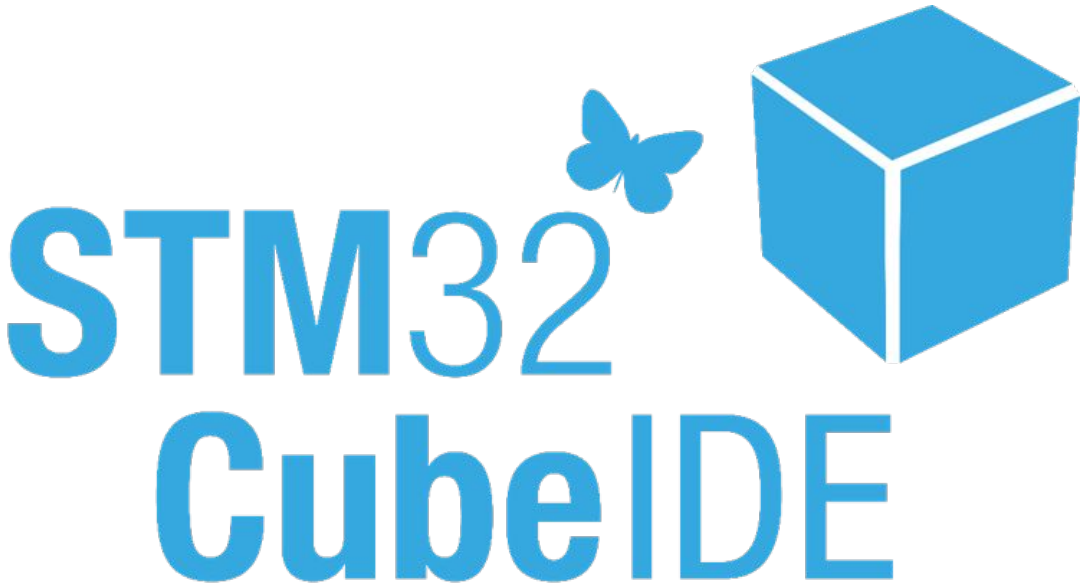
STM32G0 - *bug* ADC com DMA



O *MX_DMA_Init()* tem que estar no *Rank 2*.

STM32G0 - ADC com DMA

Ao utilizar o ADC com DMA, indicamos qual endereço de memória irá receber as conversões, e a quantidade destes dados.
Irei mostrar como configuramos pela CubeIDE utilizando o exemplo anterior.



Dúvidas ??


They said I had to flash my microcontroller
So I did, but it's still not working...
ifunny.co

Referências

ATMEL. A**VR127: Understanding ADC Parameters**. 2016. http://ww1.microchip.com/downloads/en/appnotes/atmel-8456-8-and-32-bit-avr-microcontrollers-avr127-understanding-adc-parameters_application-note.pdf . Acesso em 18 de Janeiro de 2022.

MAGDY, Khaled. **STM32 ADC Tutorial – Complete Guide With Examples**. 2020. https://deepbluembedded.com/stm32-adc-tutorial-complete-guide-with-examples/ . Acesso em 21 de Janeiro de 2022.

__. **STM32 DMA Tutorial – Using Direct Memory Access (DMA) In STM32**. 2021. https://deepbluembedded.com/stm32-dma-tutorial-using-direct-memory-access-dma-in-stm32/ . Acesso em 21 de Janeiro de 2022.

STMICROELETRONICS. **AN3116 : STM32's ADC modes and their applications**. 1. ed. [S.l.], 2010. Acesso em 19 de Janeiro de 2022.

__. **RM0444 - Reference Manual**. 5. ed. [S.l.], 2020. STM32G0x1 advanced Arm ®-based 32-bit MCUs.

__. **UM2319: Description of STM32G0 HAL and low-layer drivers**. 2. ed. [S.l.], 2020.

__. **UM2324 - User Manual.** 4. ed. [S.l.], 2021. STM32 Nucleo-64 boards (MB1360).

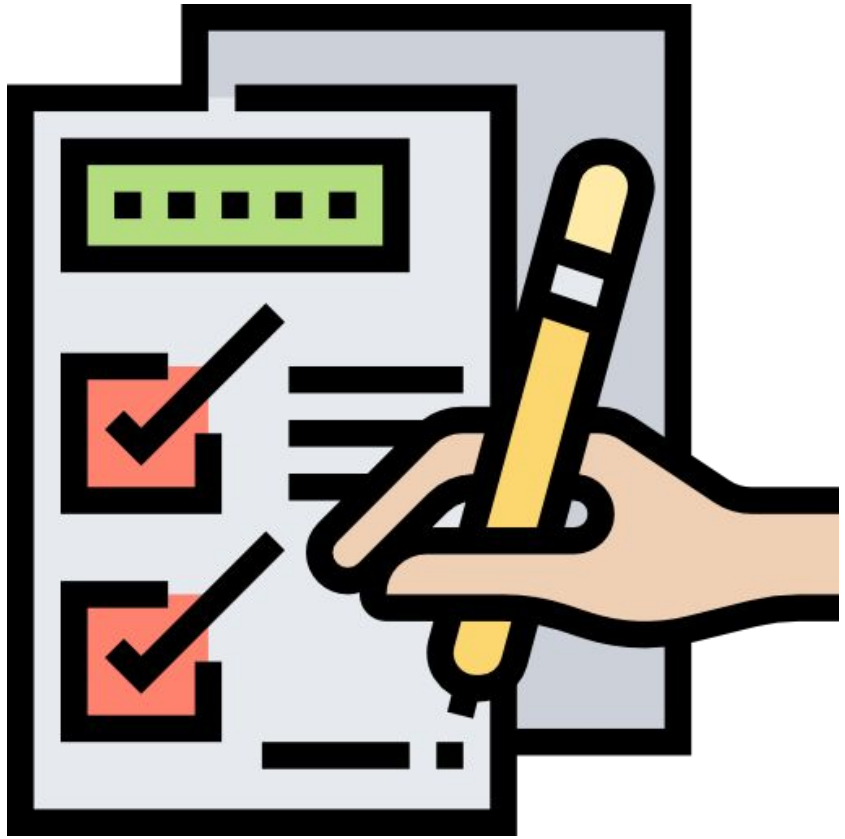
THORNTON, Scott. B**uilt-in analog-to-digital converters**. 2018. https://www.microcontrollertips.com/built-in-analog-to-digital-converters/ . Acesso em 17 de Janeiro de 2022.

WIKIPEDIA. **Analog-to-digital converter**. 2022. https://en.wikipedia.org/wiki/Analog-to-digital_converter . Acesso em 18 de Janeiro de 2022.

Mão na Massa

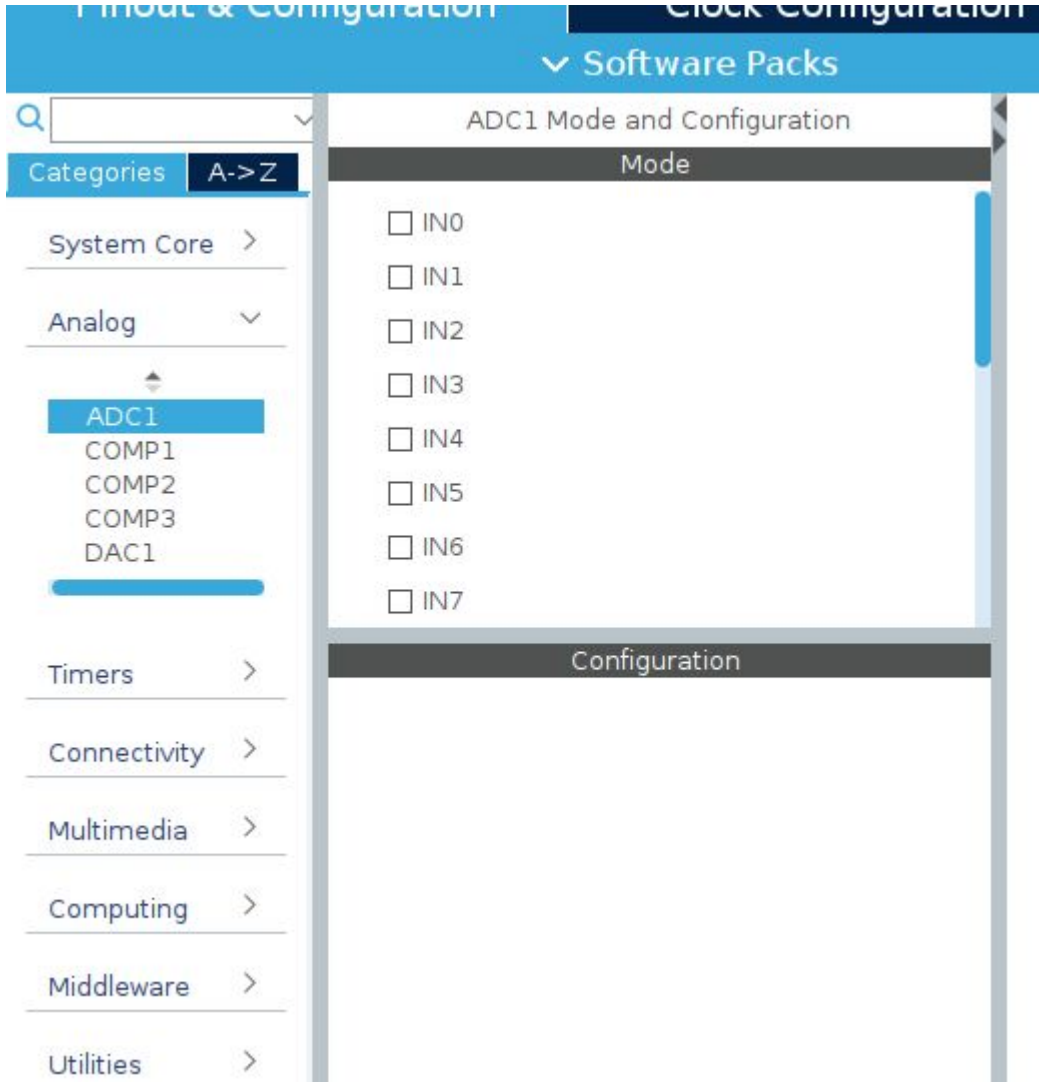
PADO Labs

Lista de Exercícios #3



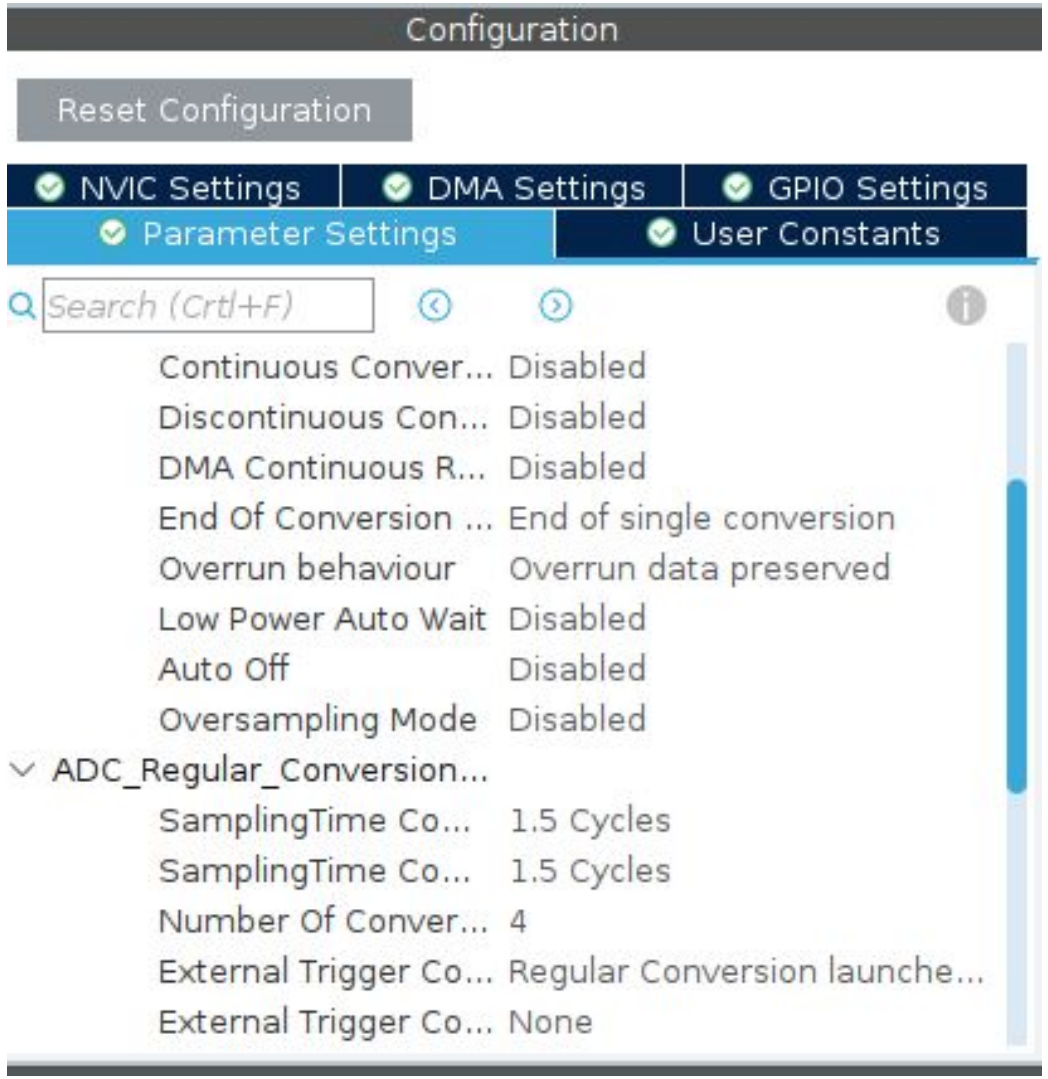
Estruturas em C e Interrupções

ADC em modo polling



Selecionamos os canais desejados na aba do periférico ADC

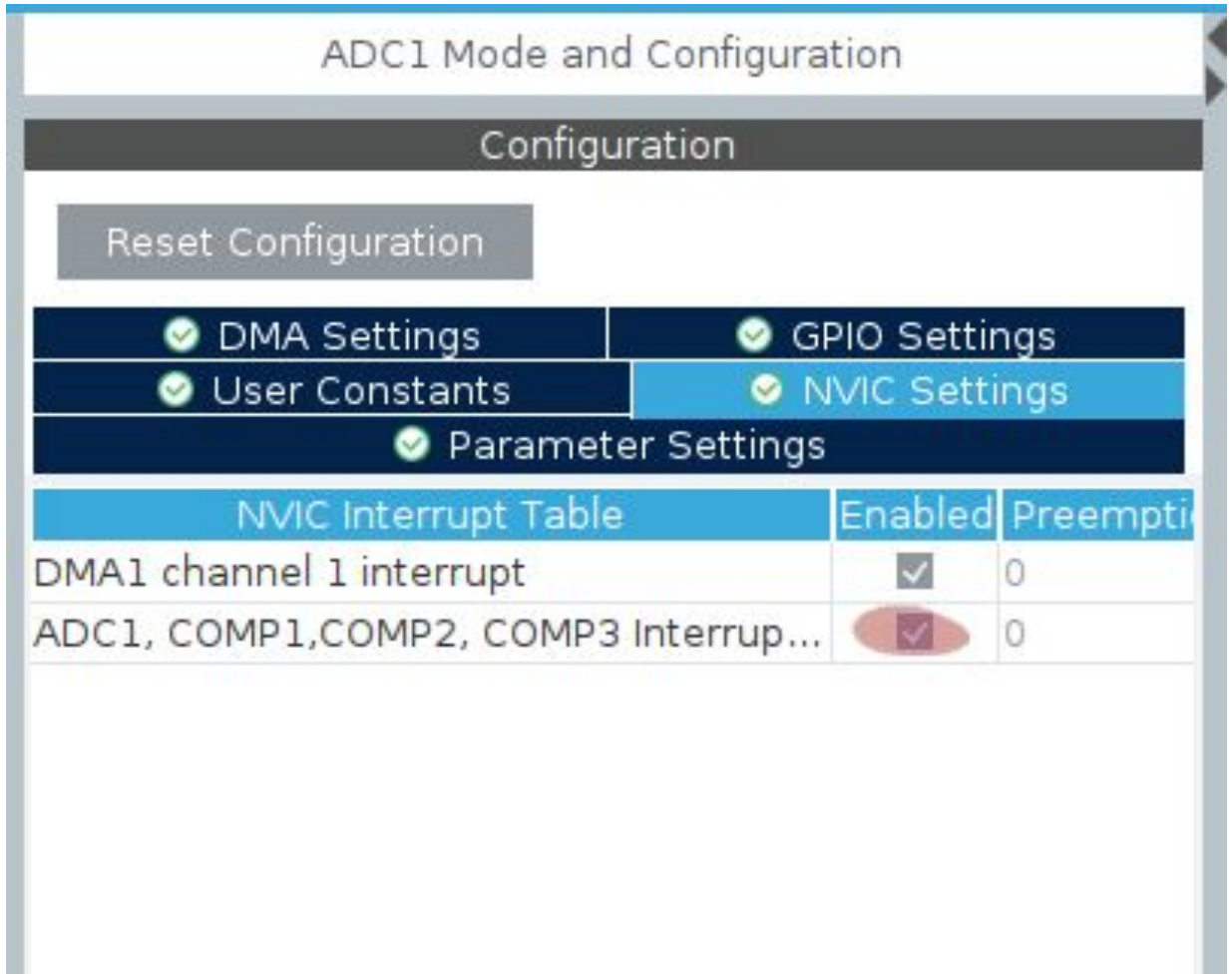
ADC em modo polling



Configuramos então o periférico de ADC conforme necessário.

Então geramos o código ao salvar.

ADC em Interrupção



Vamos até a aba do NVIC Settings.

E habilitamos a interrupção para o ADC1.

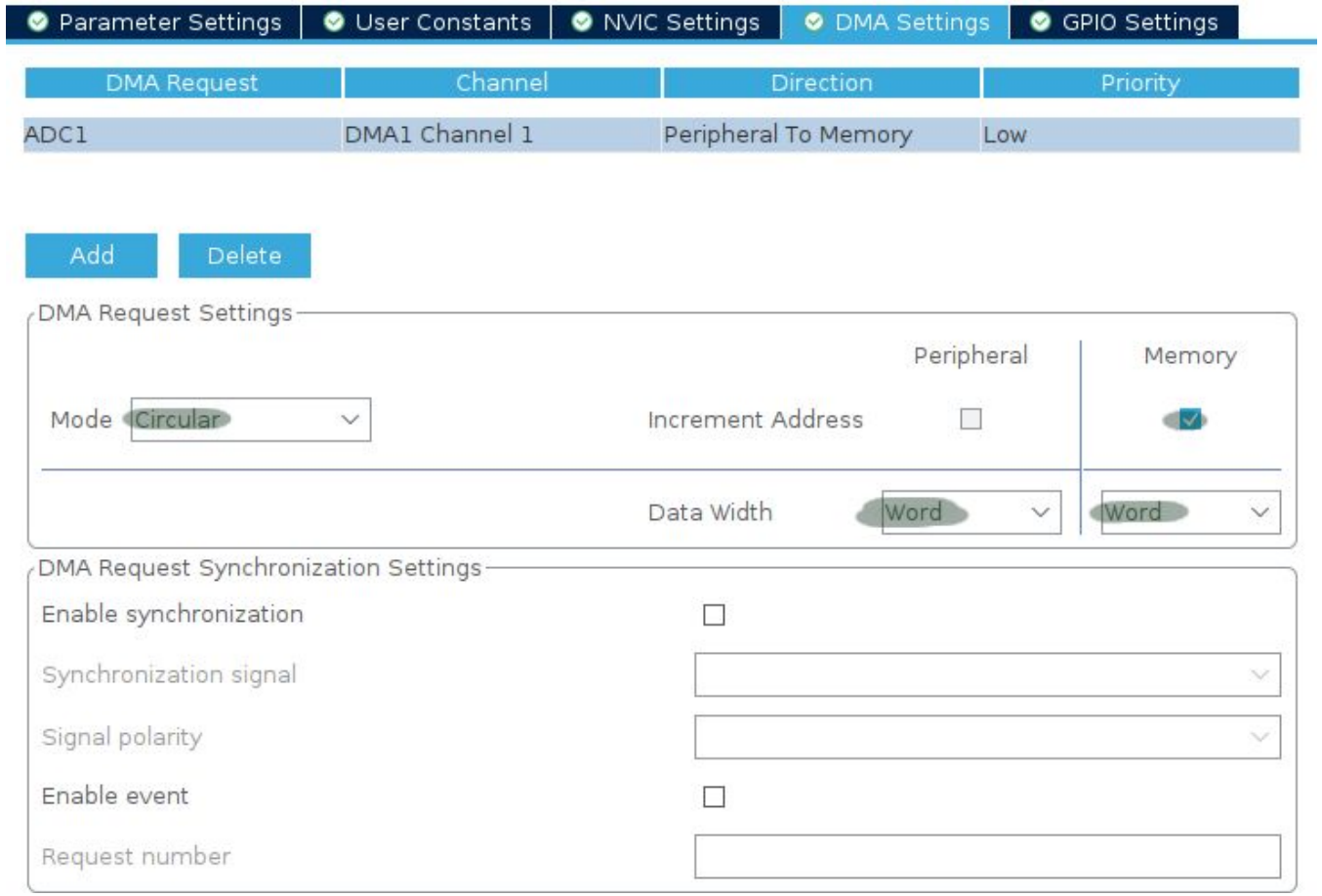
Basta gerar o código e utiliar as funções já descritas.

ADC em DMA

Vamos até a aba do DMA Settings.

Clicamos em Add e selecionamos o canal *ADC1*.

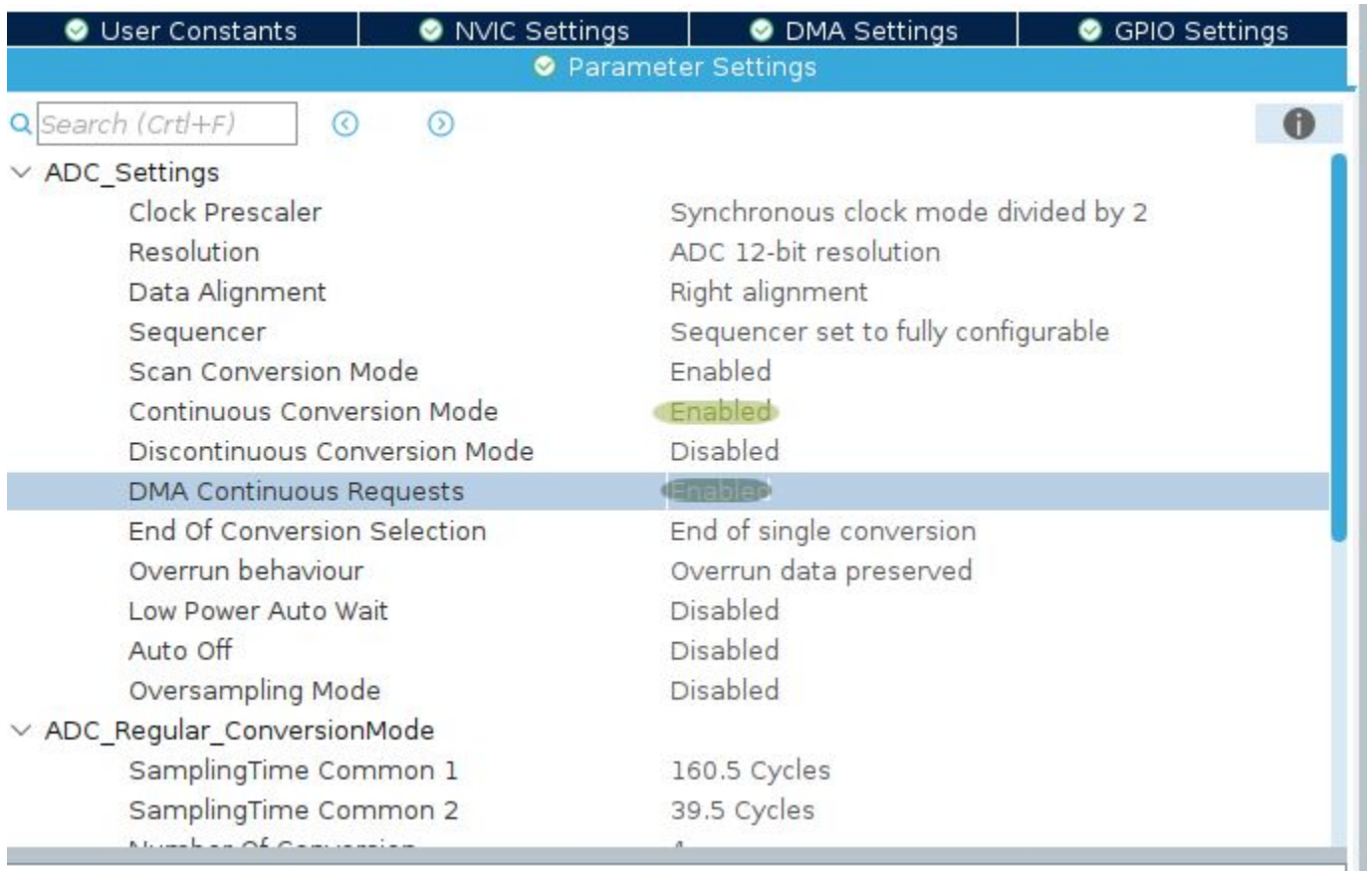
ADC em DMA



Clicamos sobre o canal para configurá-lo.

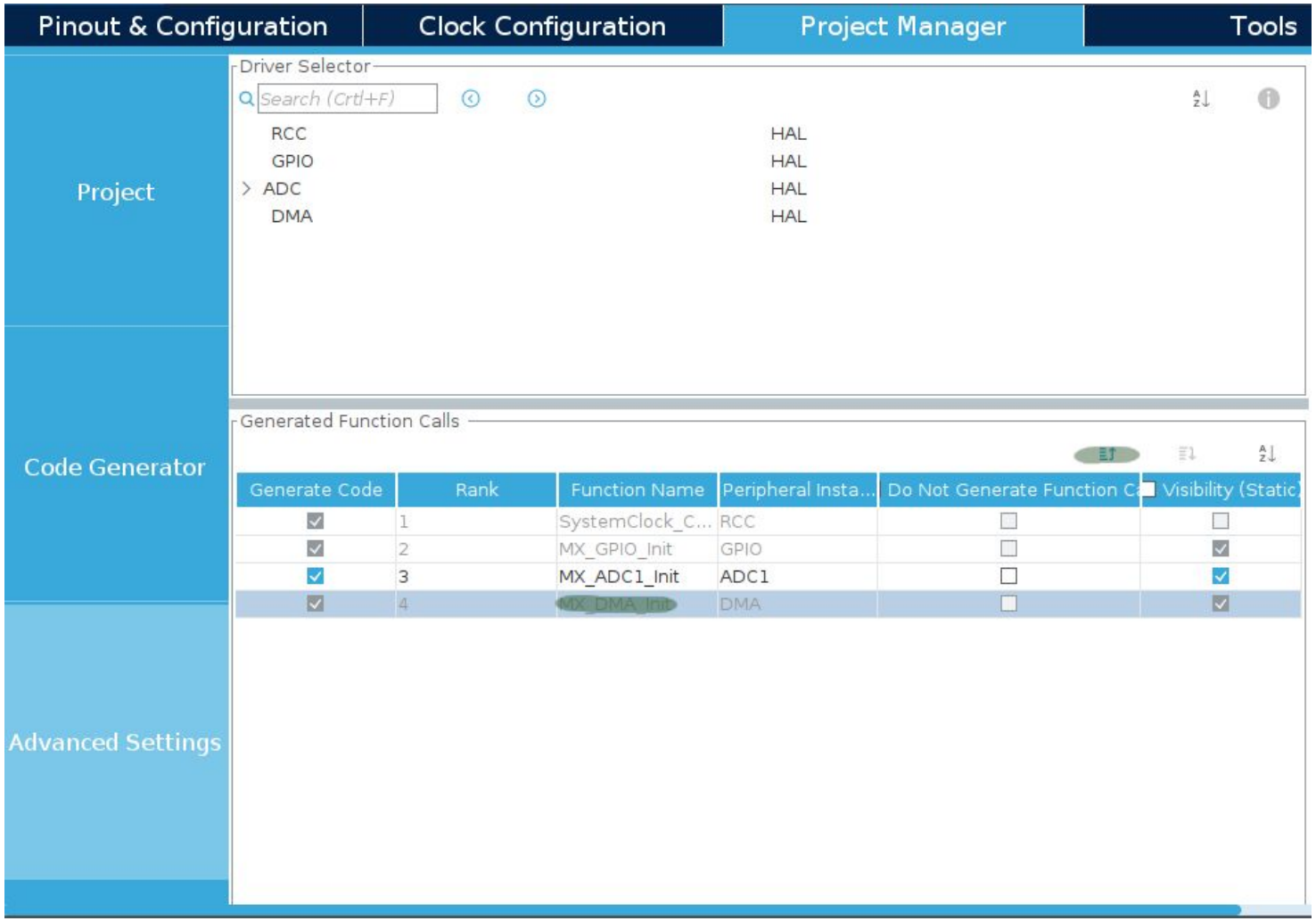
Colocamos as configurações como descrito ao lado.

ADC em DMA



Na aba Parameter Settings habilitamos o Continuous Conversion Mode e o DMA Continuous Requests.

ADC em DMA



Corrigimos o *bug* da STM32CubeIDE, colocando o **MX_DMA_Init()** no *Rank 2*.

PADO

Labs