

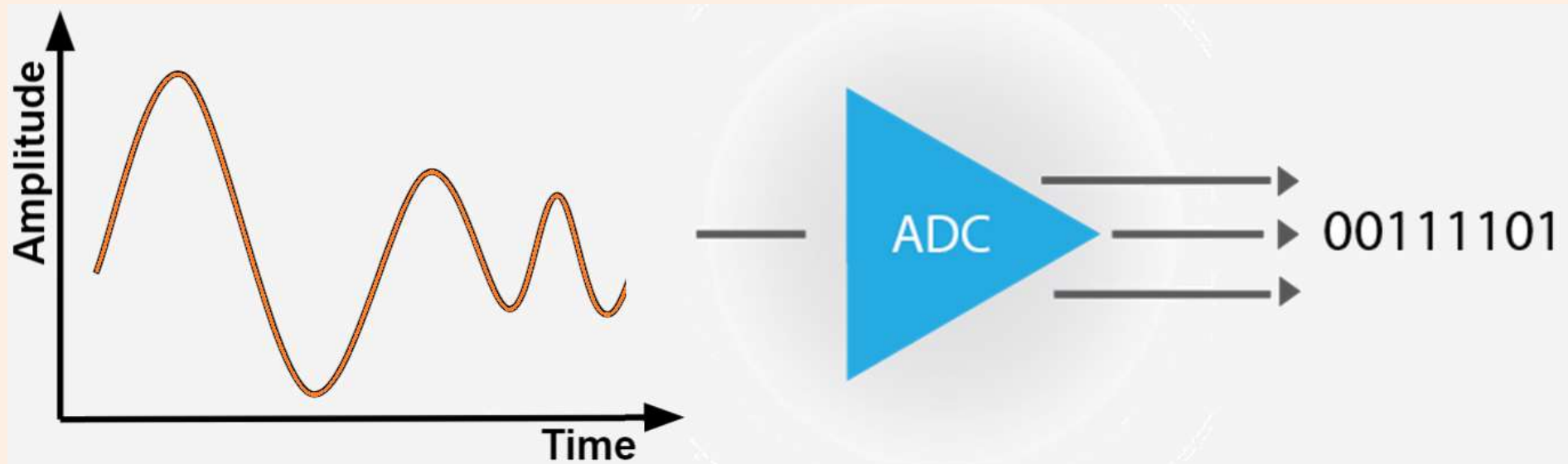
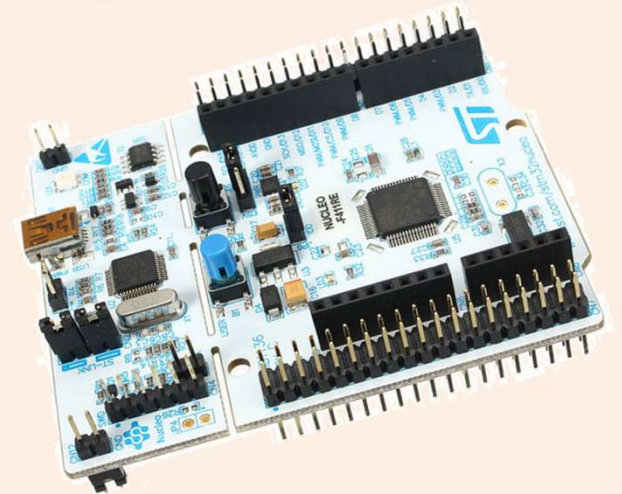
# Competências Transferíveis

## Microcontroladores e Interação com Sensores e Atuadores

2021-2022

Rui Escadas Martins

### ADC – Analog to Digital Converters

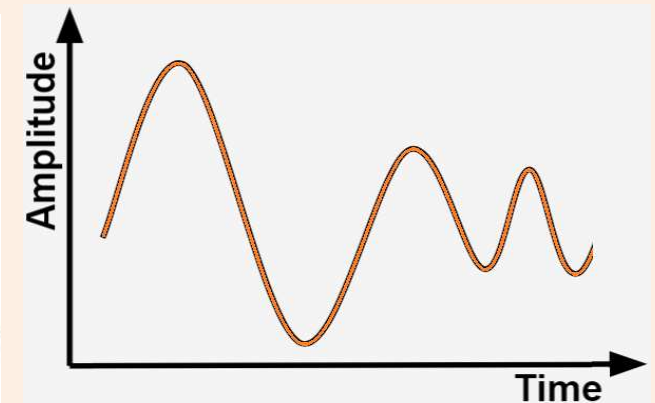
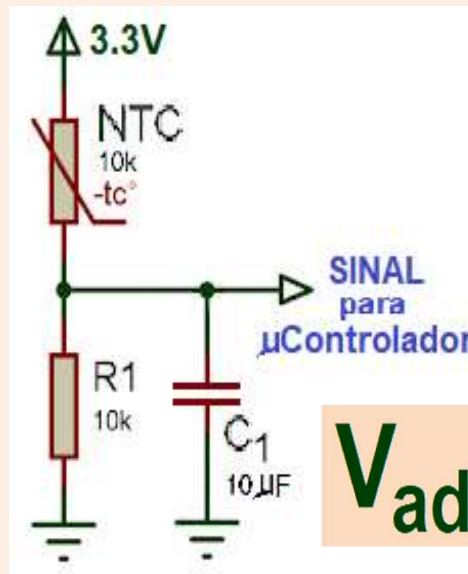
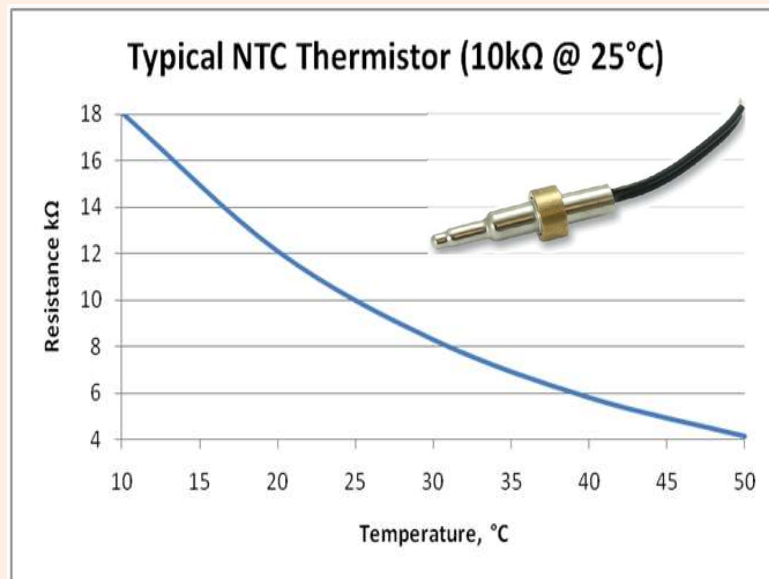


# Analog Digital converters

O que são:

São componentes que convertem uma grandeza analógica, geralmente uma tensão, na sua representação binária.

Tipicamente, a tensão é gerada através do condicionamento (amplificação, filtragem, etc) do sinal gerado por um sensor.

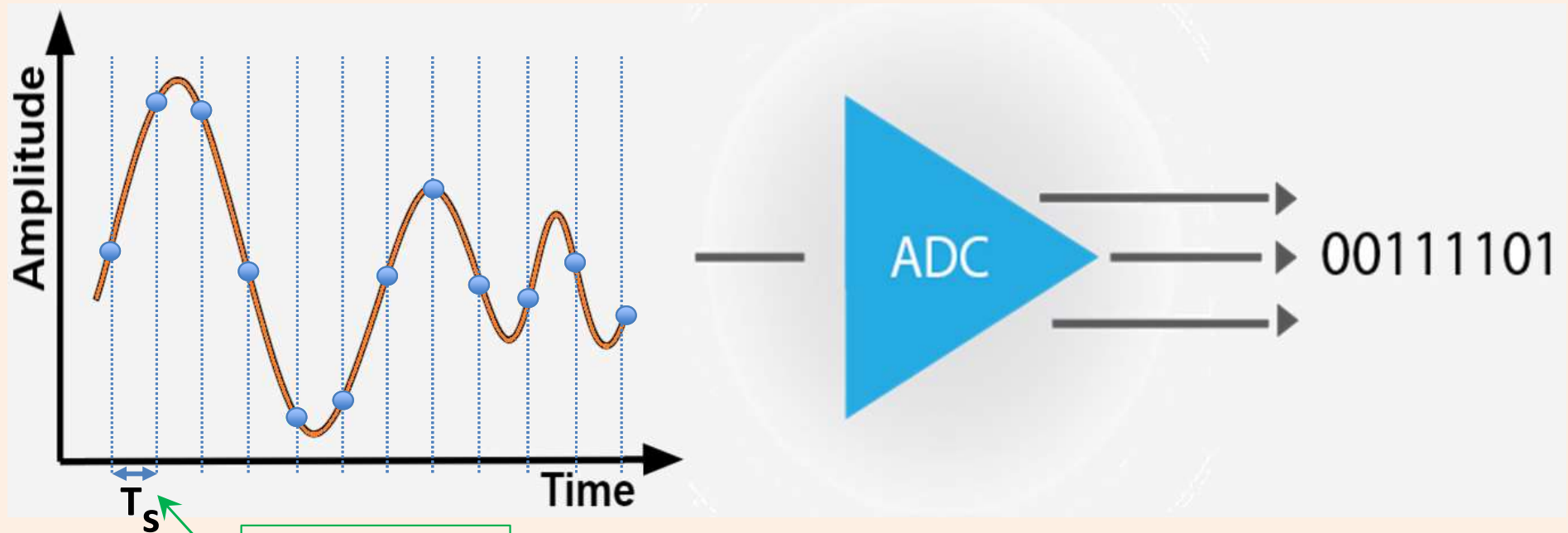


$$V_{adc} = R_1 / (R_1 + R_{NTC})$$

# Analog Digital converters

O que são:

São componentes que convertem uma grandeza analógica, geralmente uma tensão, na sua representação binária.



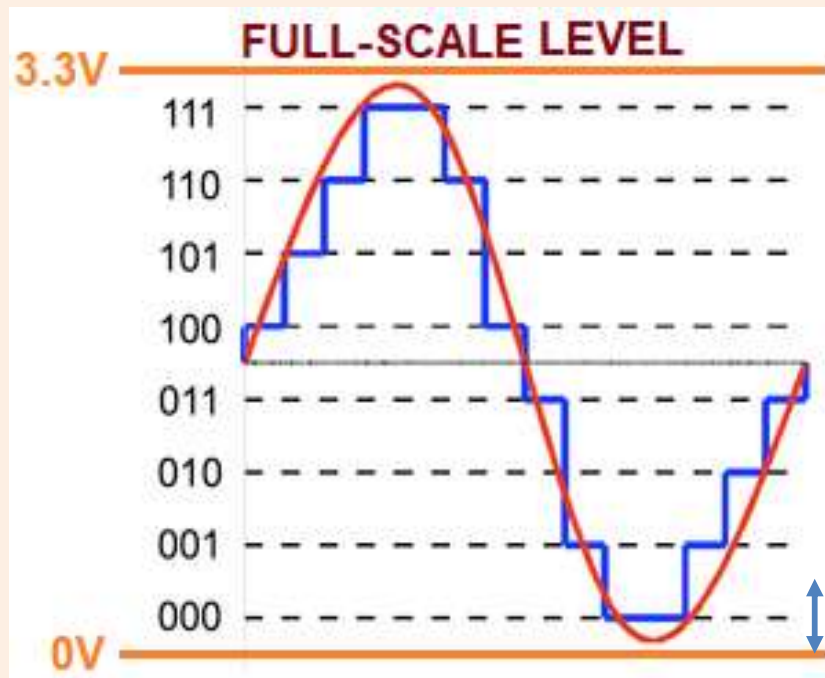
Período de amostragem

Normalmente especifica-se a **frequência de amostragem**:  $F_s = 1/T_s$  (amostras/s)

# Analog Digital converters

O que são:

São componentes que convertem uma grandeza analógica, geralmente uma tensão, na sua representação binária.



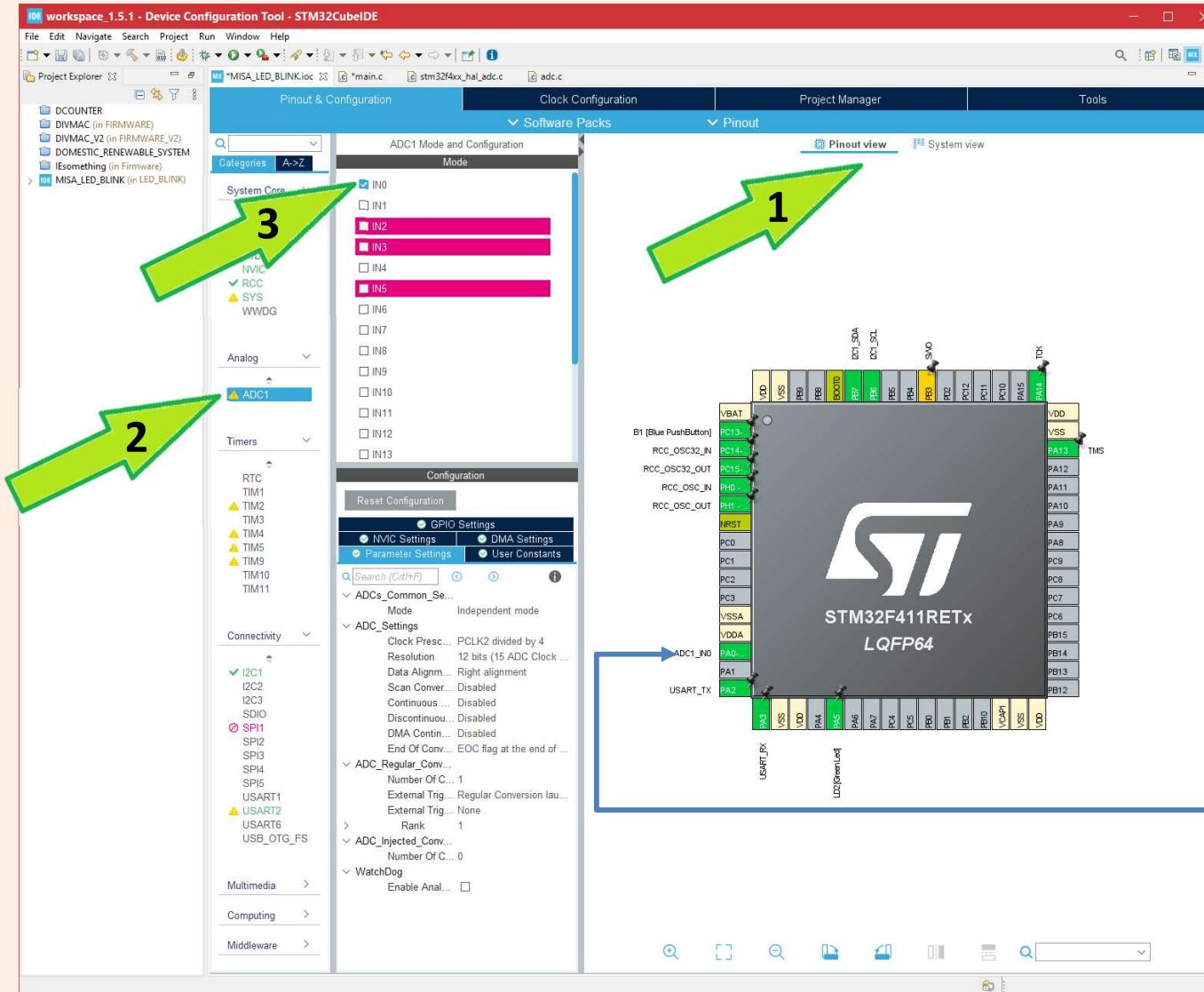
Todos os valores (sinal contínuo) que estão compreendidos neste intervalo são codificados com o mesmo Código Binário!

A **Resolução** de uma ADC é o valor mínimo do intervalo de tensão que causa a mudança da saída digital.

Normalmente especifica-se em **Número de bits** – por exemplo, uma ADC de 3-bits (como na figura) tem  $2^3 = 8$  níveis possíveis. Para se conseguir uma representação mais “fiel” na prática são usadas resoluções muito superiores. A ADC usada tem 12-bits (ou seja  $2^{12} = 4096$  níveis).

# Analog Digital converters

Procedimento: em **Pinout View** (Seta1) seleccionar **ADC1** (Seta2), depois **IN0** (Seta3),



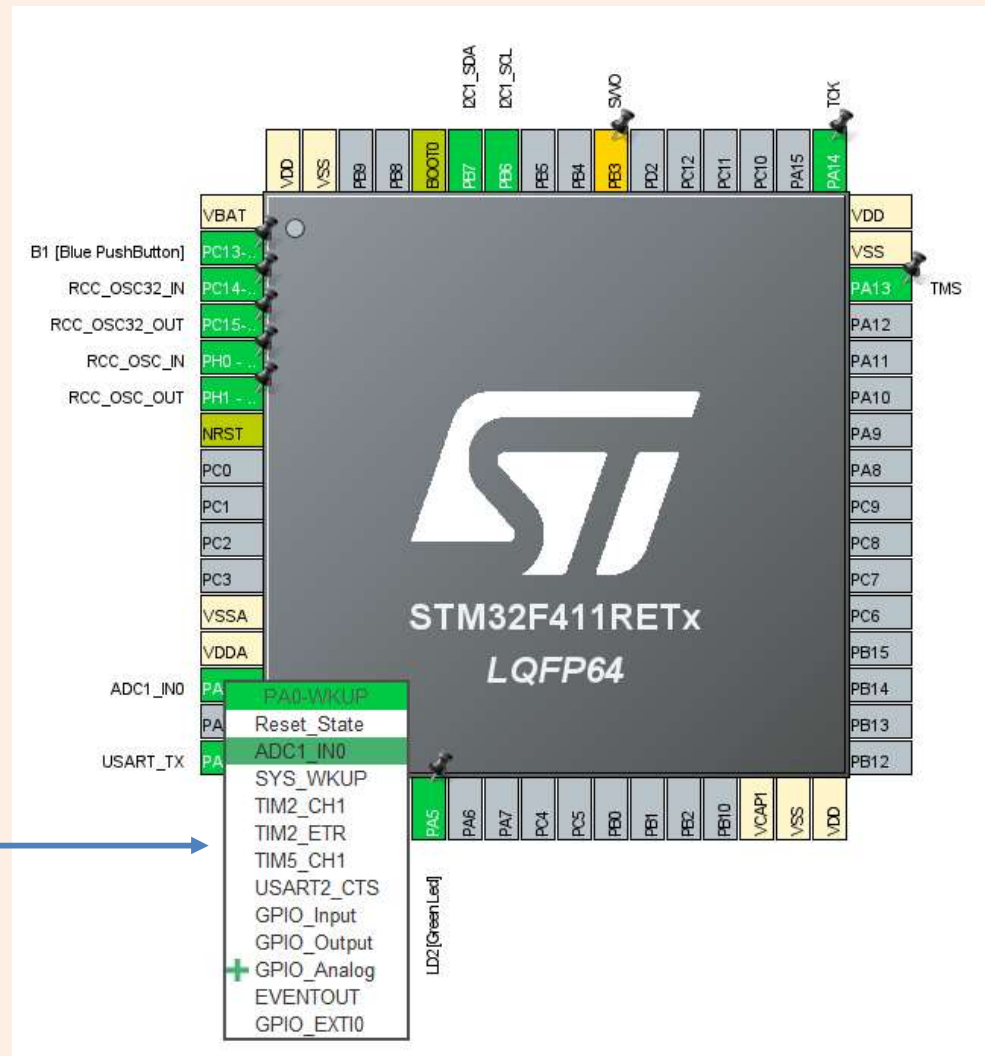
Reparar que este  
pino fica verde  
(significa que está  
a ser usado).

# Analog Digital converters

Procedimento: seleccionar o pino **ADC1\_IN0** com o botão esquerdo

Pode-se ver o tipo de funcionalidades possíveis para um dado pino. Neste caso está como ADC1\_IN0 + GPIO\_Analog

NOTA: Caso se queira fazer regressar o pino a um estado “Não utilizado” seleccionar: **Reset\_State**



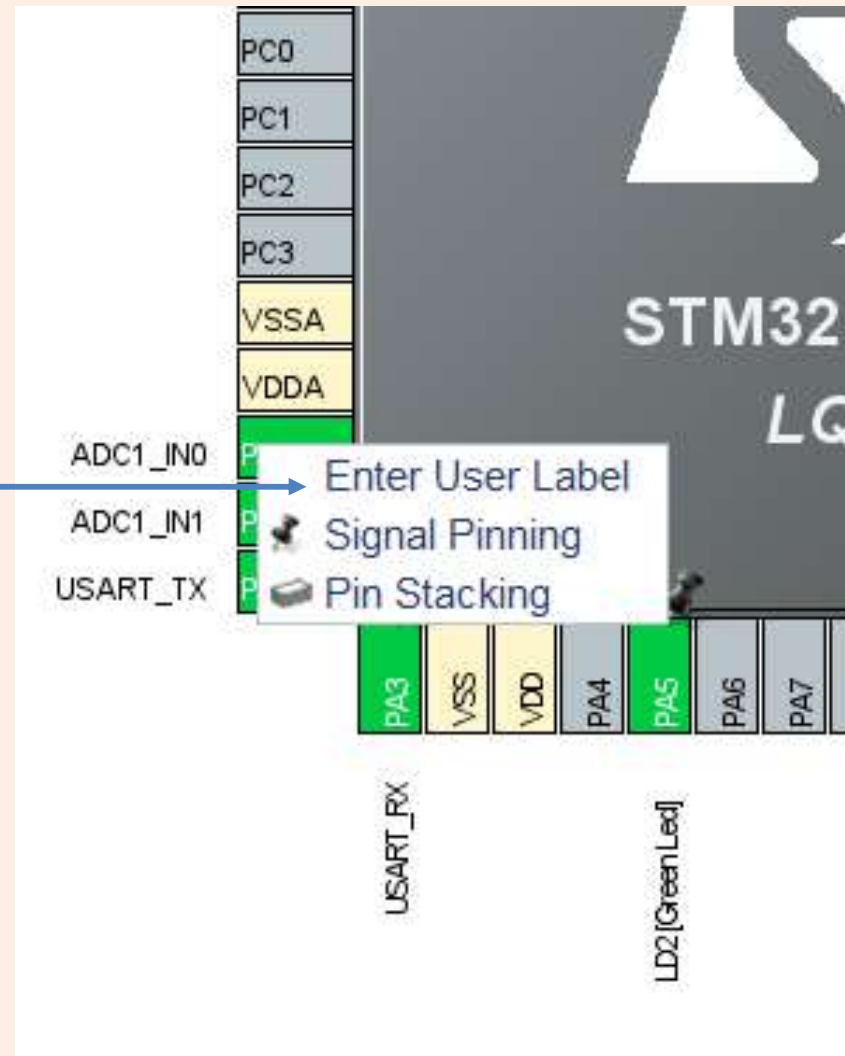


# Analog Digital converters

Procedimento: seleccionar o pino **ADC1\_IN0** com o botão direito

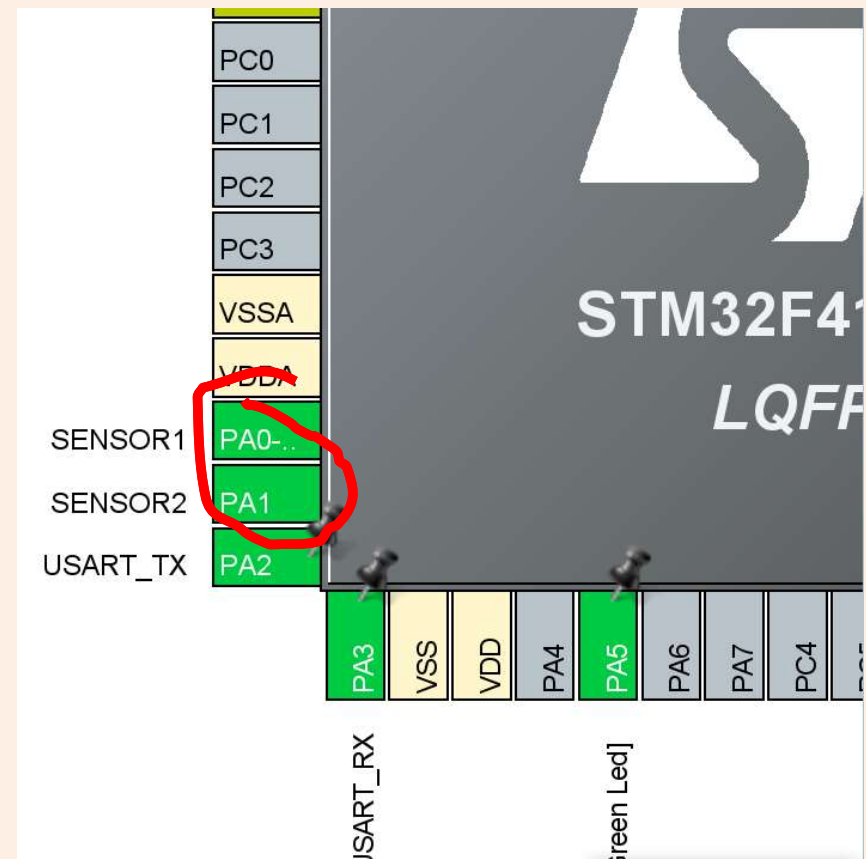
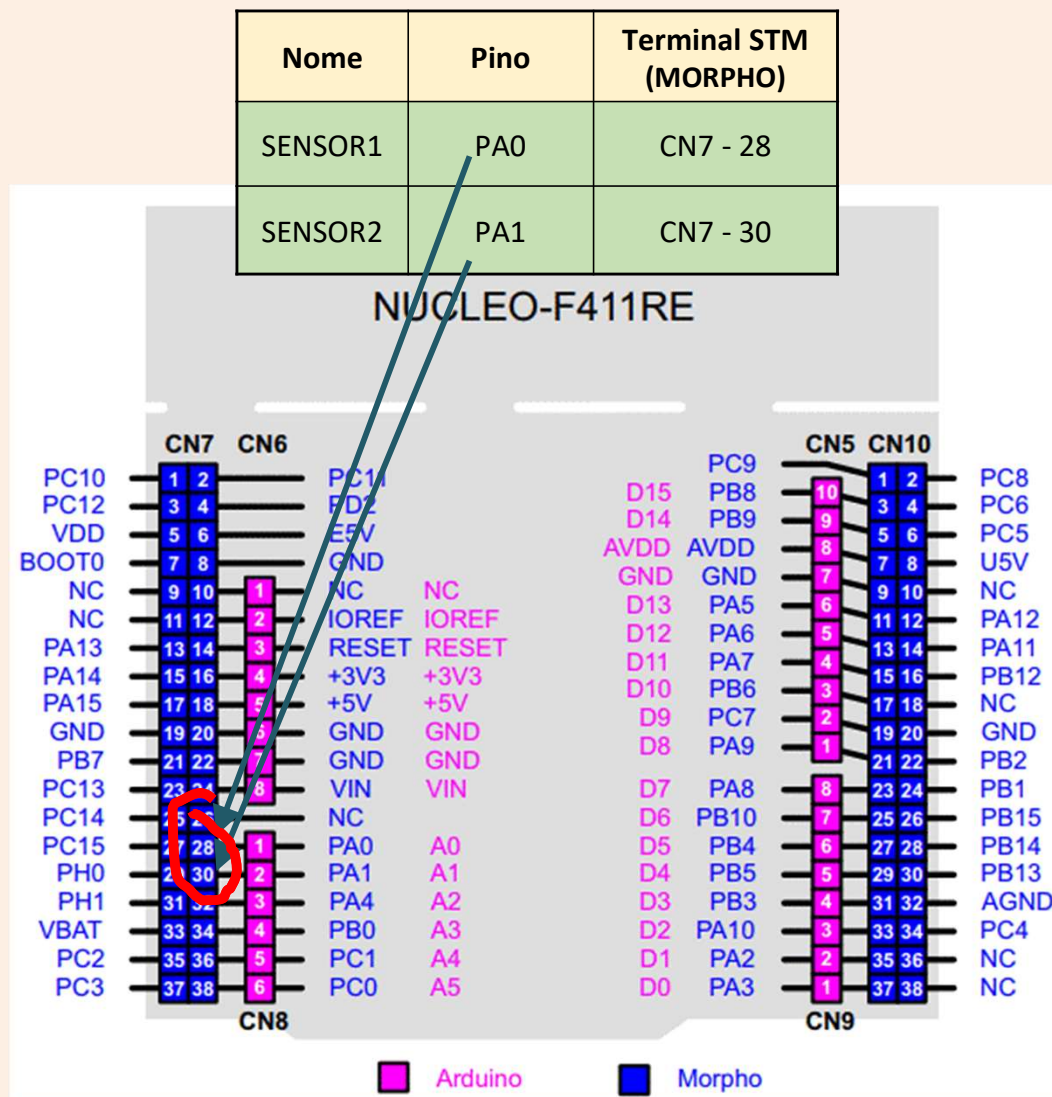
Pode-se alterar o nome do pino para ser mais fácil entender a sua função...

Este nome vai ter correspondência no Código Gerado, para também facilitar a sua identificação



# Analog Digital converters

Procedimento: identificar os pinos e relacioná-los com os terminais físicos.



O procedimento é idêntico usando o ambiente de desenvolvimento baseado em Arduino!

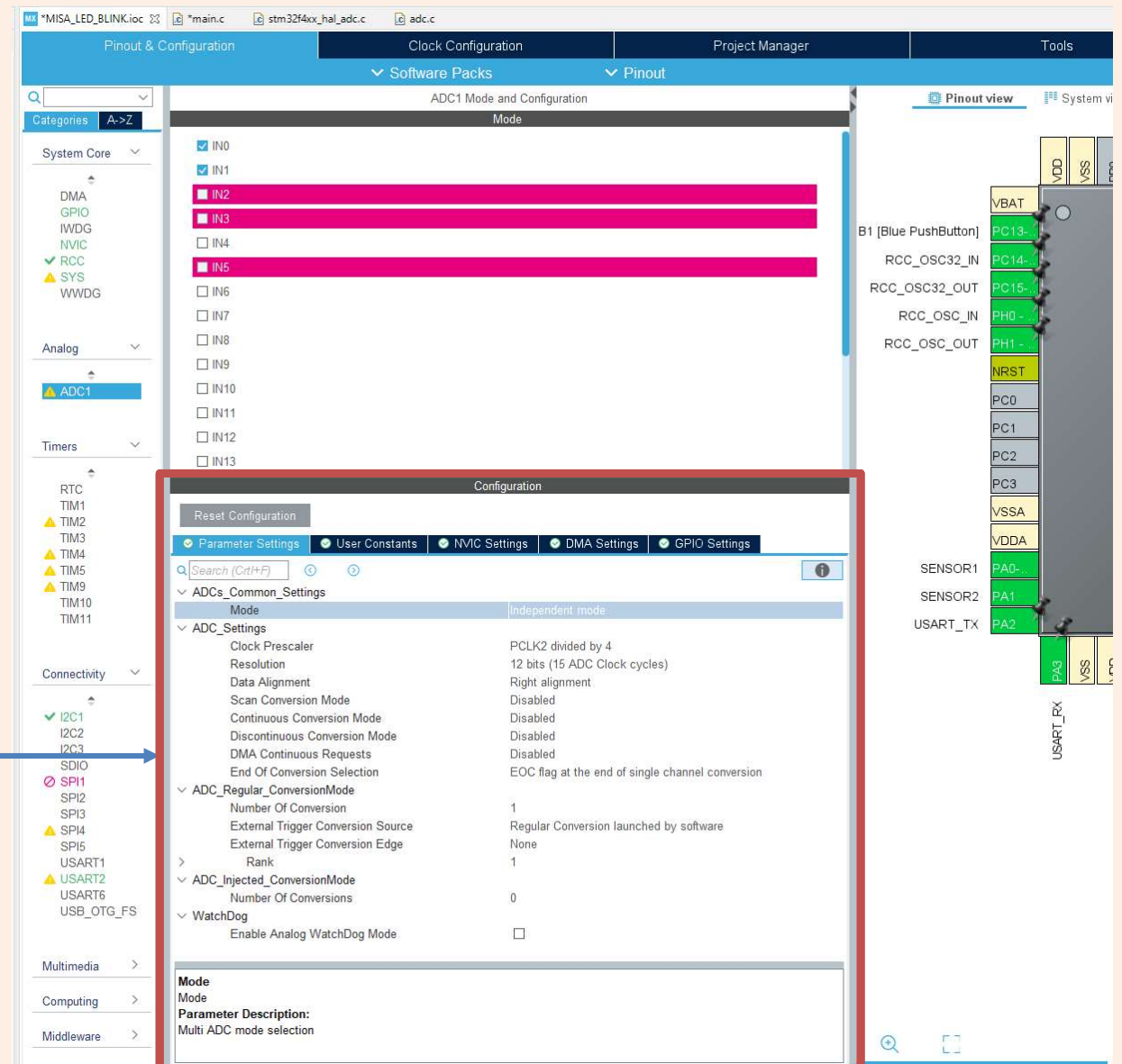


# Analog Digital converters

Nota extra:

Na zona das configurações dos periféricos é possível aceder a um conjunto de parâmetros que se podem configurar.

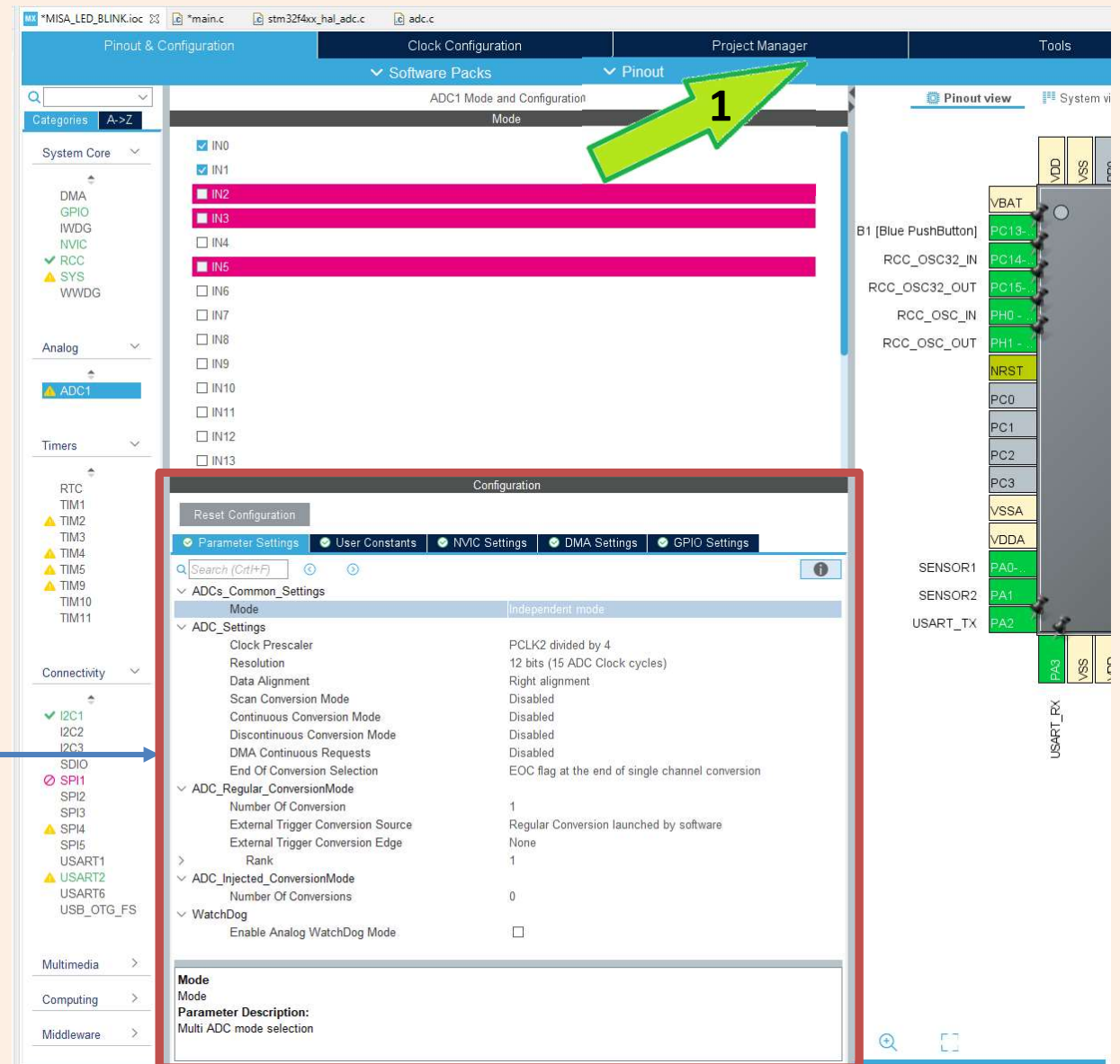
Para este curso recomendo que se não alterem as configurações standard.



# Analog Digital converters

Nota extra:

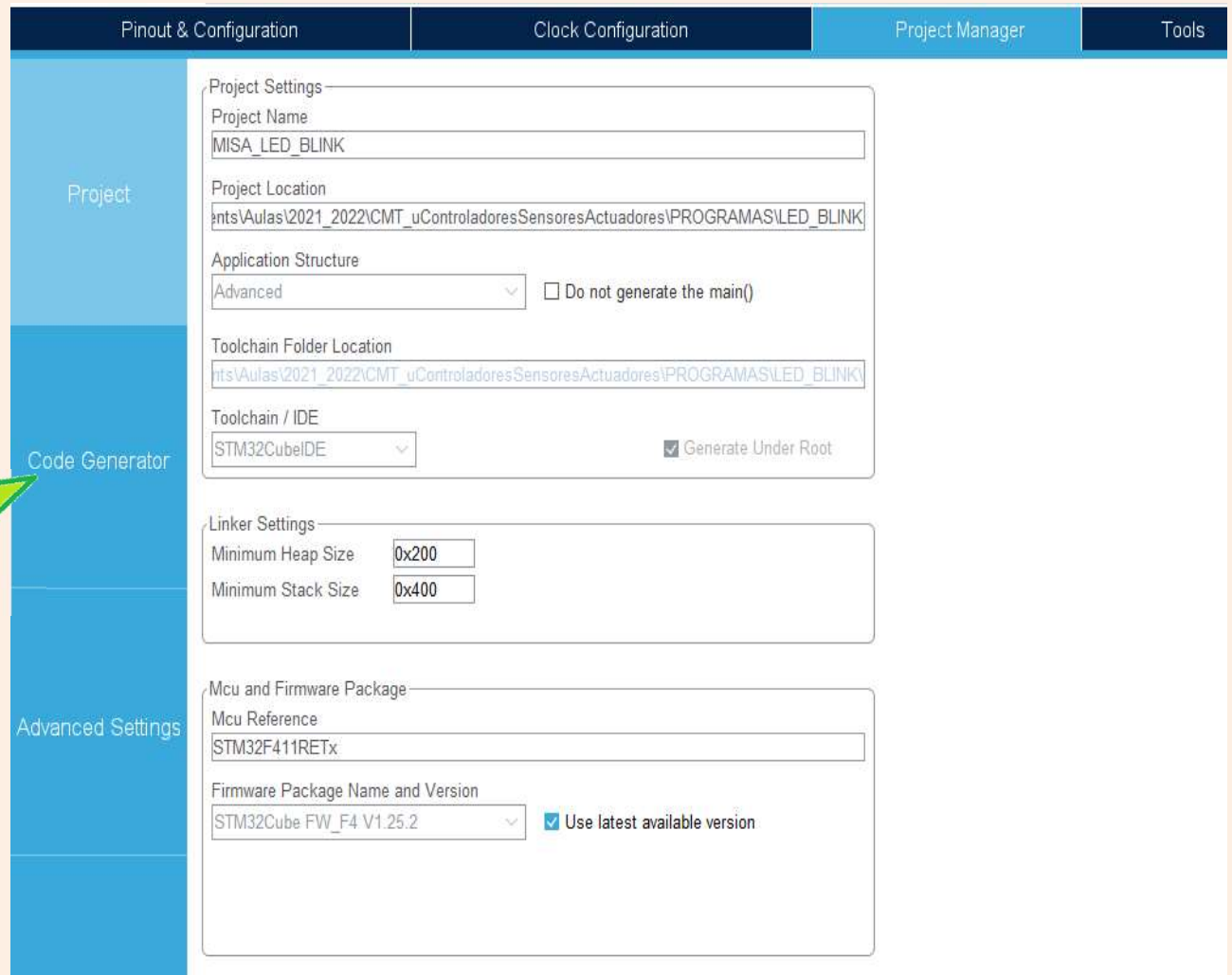
Selecionando o **Project Manager** é possível aceder a um menu sobre o projecto.



# Analog Digital converters

Nota extra:

Selecionando **Code Generator**  
pode-se fazer algumas  
configurações interessantes



The screenshot shows the STM32CubeIDE Project Manager window. The left sidebar has four tabs: 'Project', 'Code Generator', 'Advanced Settings', and 'Tools'. The 'Code Generator' tab is selected and highlighted in blue. A green arrow with the number '1' points to this tab. The main area displays the 'Code Generator' settings, which are organized into three sections: 'Project Settings', 'Linker Settings', and 'Mcu and Firmware Package'. The 'Project Settings' section includes fields for 'Project Name' (MISA\_LED\_BLINK), 'Project Location' (a long file path), 'Application Structure' (Advanced), and 'Toolchain Folder Location' (another long file path). There is a checkbox for 'Do not generate the main()' which is unchecked, and a checkbox for 'Generate Under Root' which is checked. The 'Linker Settings' section includes 'Minimum Heap Size' (0x200) and 'Minimum Stack Size' (0x400). The 'Mcu and Firmware Package' section includes 'Mcu Reference' (STM32F411RETx) and 'Firmware Package Name and Version' (STM32Cube FW\_F4 V1.25.2), with a checkbox for 'Use latest available version' which is checked.

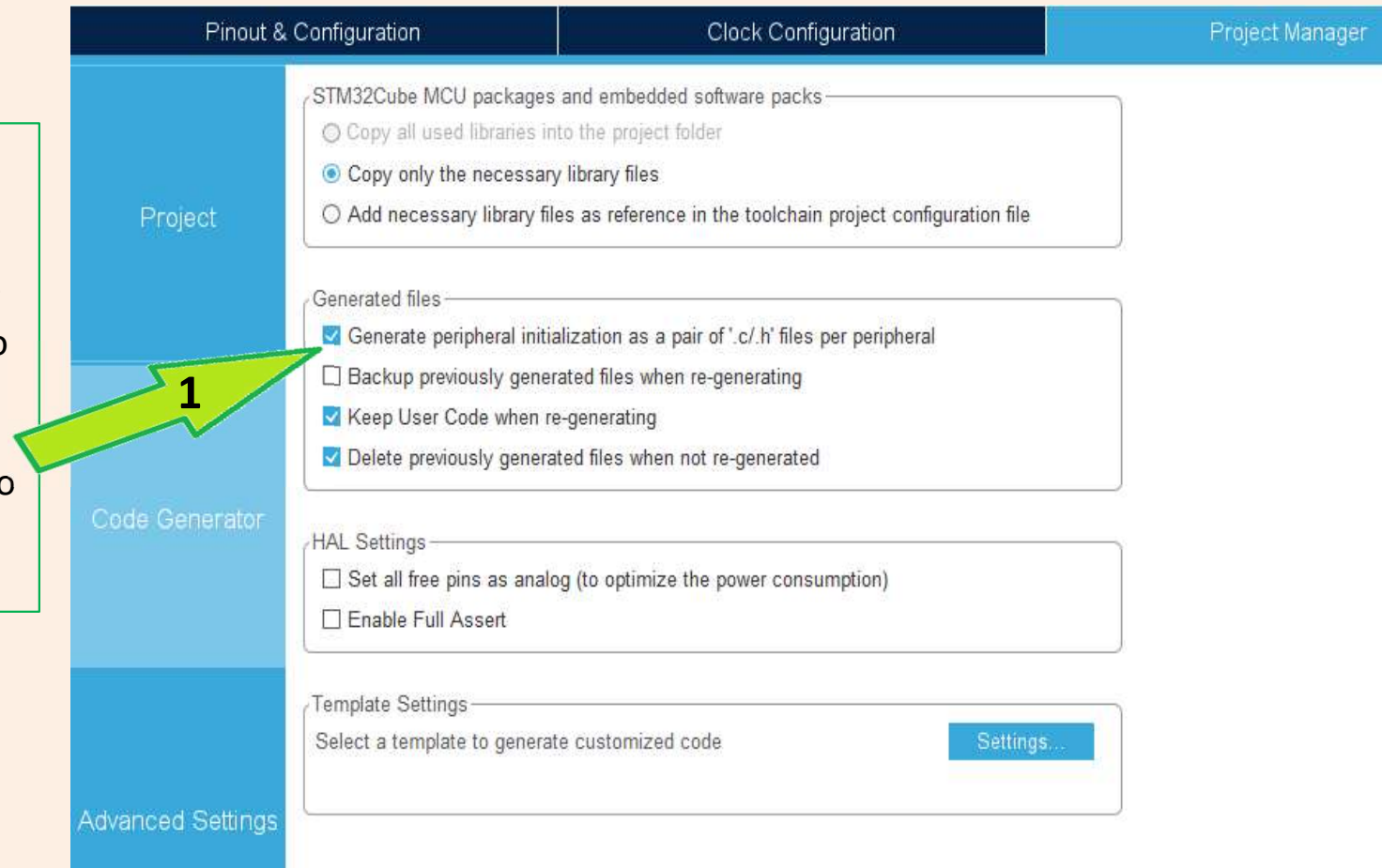
Section	Field	Value	Checkbox
Project Settings	Project Name	MISA_LED_BLINK	
	Project Location	nts\Aulas\2021_2022\ICMT_uControladoresSensoresAtuadores\PROGRAMAS\LED_BLINK\	
	Application Structure	Advanced	<input type="checkbox"/> Do not generate the main()
	Toolchain Folder Location	nts\Aulas\2021_2022\ICMT_uControladoresSensoresAtuadores\PROGRAMAS\LED_BLINK\	
Linker Settings	Minimum Heap Size	0x200	
	Minimum Stack Size	0x400	
Mcu and Firmware Package	Toolchain / IDE	STM32CubeIDE	<input checked="" type="checkbox"/> Generate Under Root
	Mcu Reference	STM32F411RETx	
	Firmware Package Name and Version	STM32Cube FW_F4 V1.25.2	<input checked="" type="checkbox"/> Use latest available version

# Analog Digital converters

Nota extra:

Em vez de ficar tudo concentrado em “main.c” seleccionando esta opção, gera-se um par de “.c/.h” por cada periférico, o que torna o código mais leve.

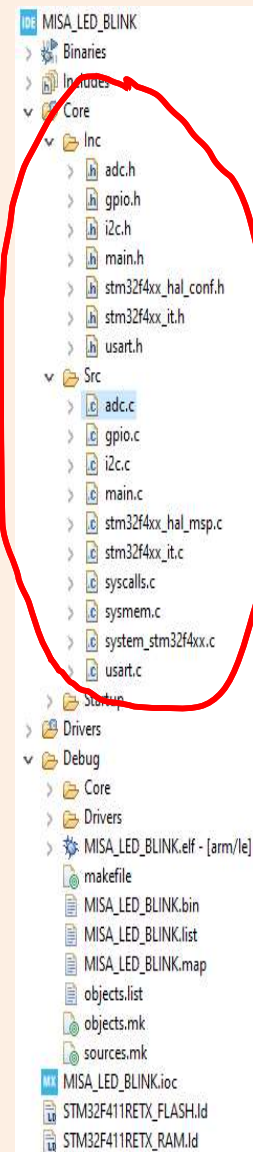
As outras opções também são fáceis de entender, mas em caso de dúvida não mexer!



Por último: “Salvar” as configurações e gerar Código automaticamente!

# Analog Digital converters

Reparar que foram adicionados, entre outros, os ficheiros: **adc.c** e **adc.h**



```

5  @uriet : main program body
6  *****
7  * @attention
8  *
9  * <h2><center>&copy; Copyright (c) 2022 STMicroelectronics.
10 * All rights reserved.</center></h2>
11 *
12 * This software component is licensed by ST under BSD 3-Clause license,
13 * the "License"; You may not use this file except in compliance with the
14 * License. You may obtain a copy of the License at:
15 * opensource.org/licenses/BSD-3-Clause
16 *
17 *****
18 */
19 @/* USER CODE END Header */
20 /* Includes ----- */
21 #include "main.h"
22 #include "adc.h"
23 #include "i2c.h"
24 #include "usart.h"
25 #include "gpio.h"
26
27 @/* Private includes ----- */
28 /* USER CODE BEGIN Includes */
29
30 /* USER CODE END Includes */
31
32 @/* Private typedef ----- */
33 /* USER CODE BEGIN PTD */
34
35 /* USER CODE END PTD */
36
37 @/* Private define ----- */
38 /* USER CODE BEGIN PD */
39 /* USER CODE END PD */
40
41 @/* Private macro ----- */

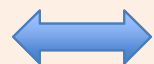
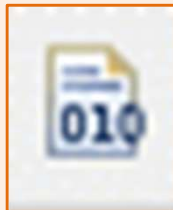
```

# Analog Digital converters

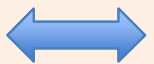
---

## Fazer como já apresentado:

- Escrever o Código.
- Compilar/criar executável: **Project > Build-All**
- e para enviar o ficheiro executável para a placa fazer: **Run > Debug!**
- Em alternativa, também podem seleccionar os icons correspondents:



**Build-All**



**Debug!**