

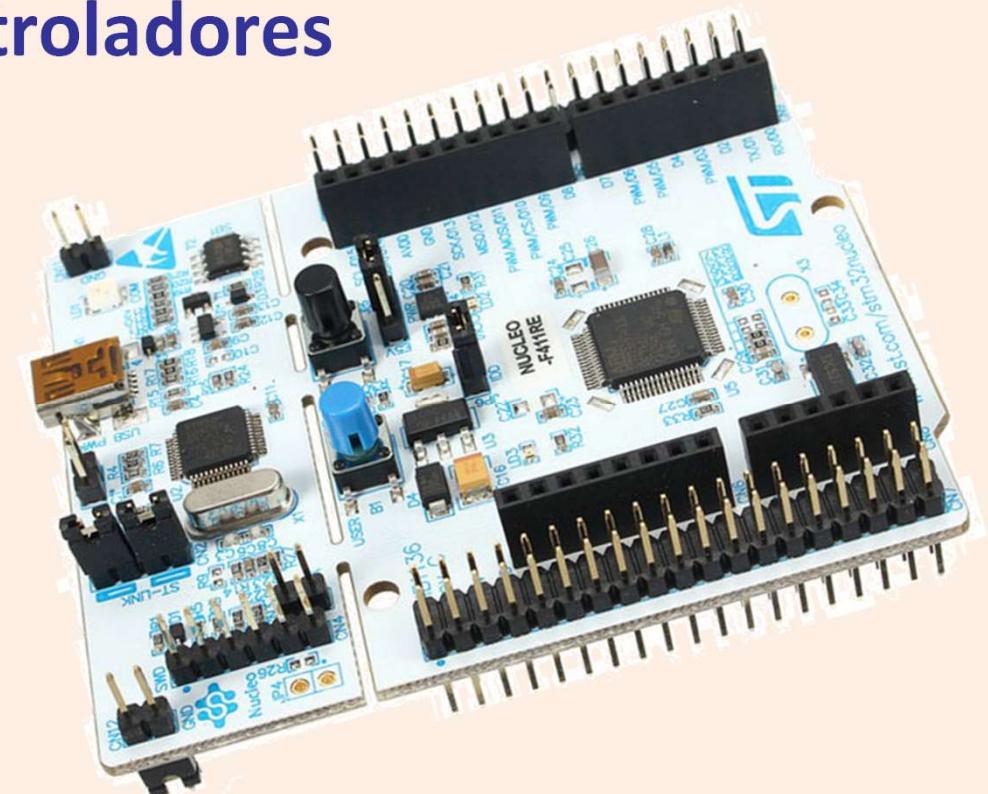
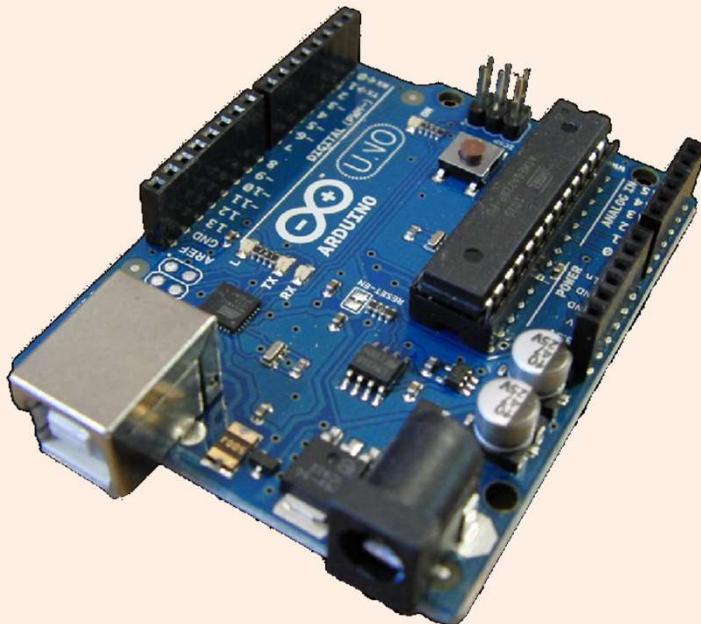
# Competências Transferíveis

## Microcontroladores e Interação com Sensores e Atuadores

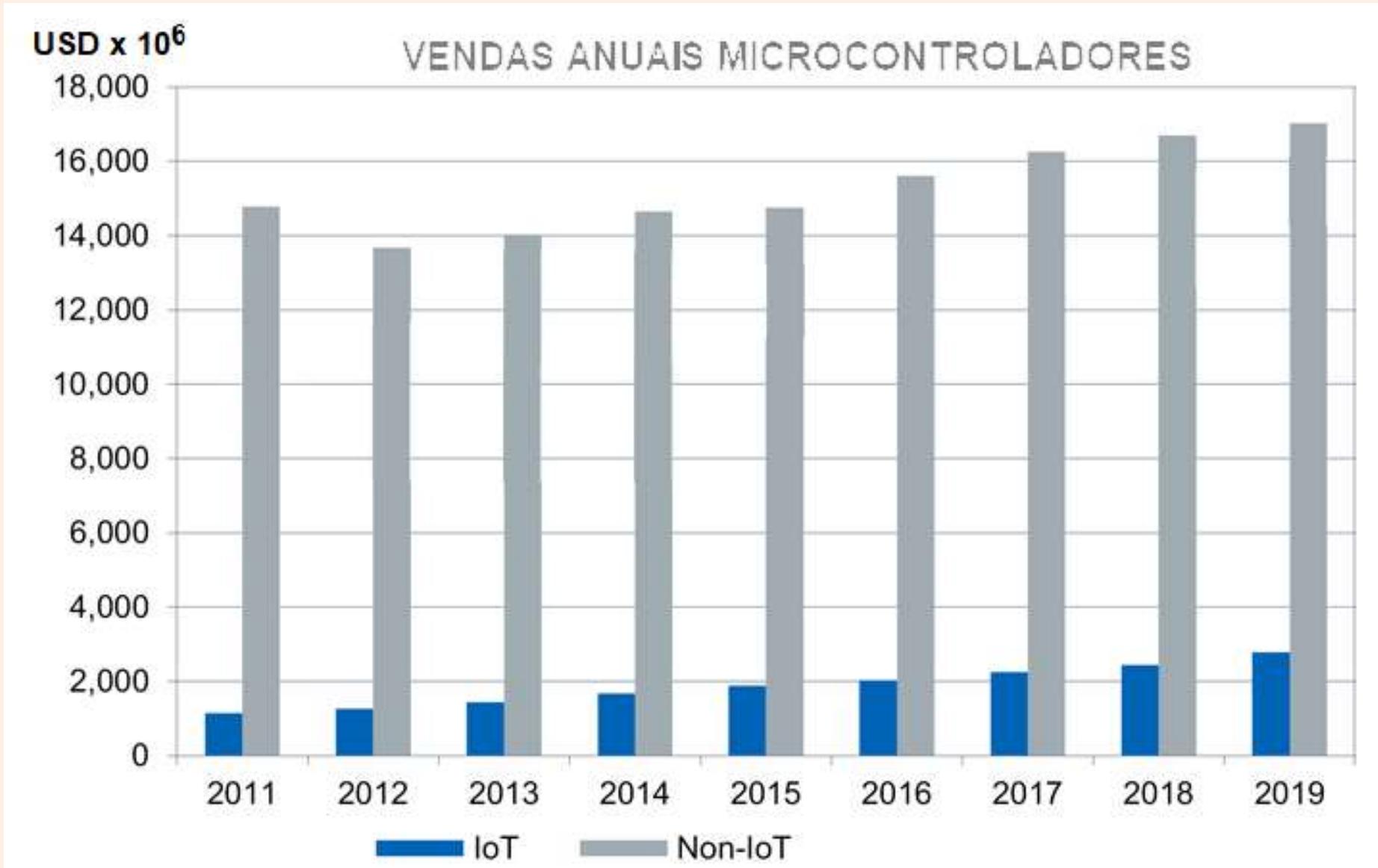
V3\_3

Rui Escadas Martins

### Introdução aos Microcontroladores



# Introdução aos Microcontroladores



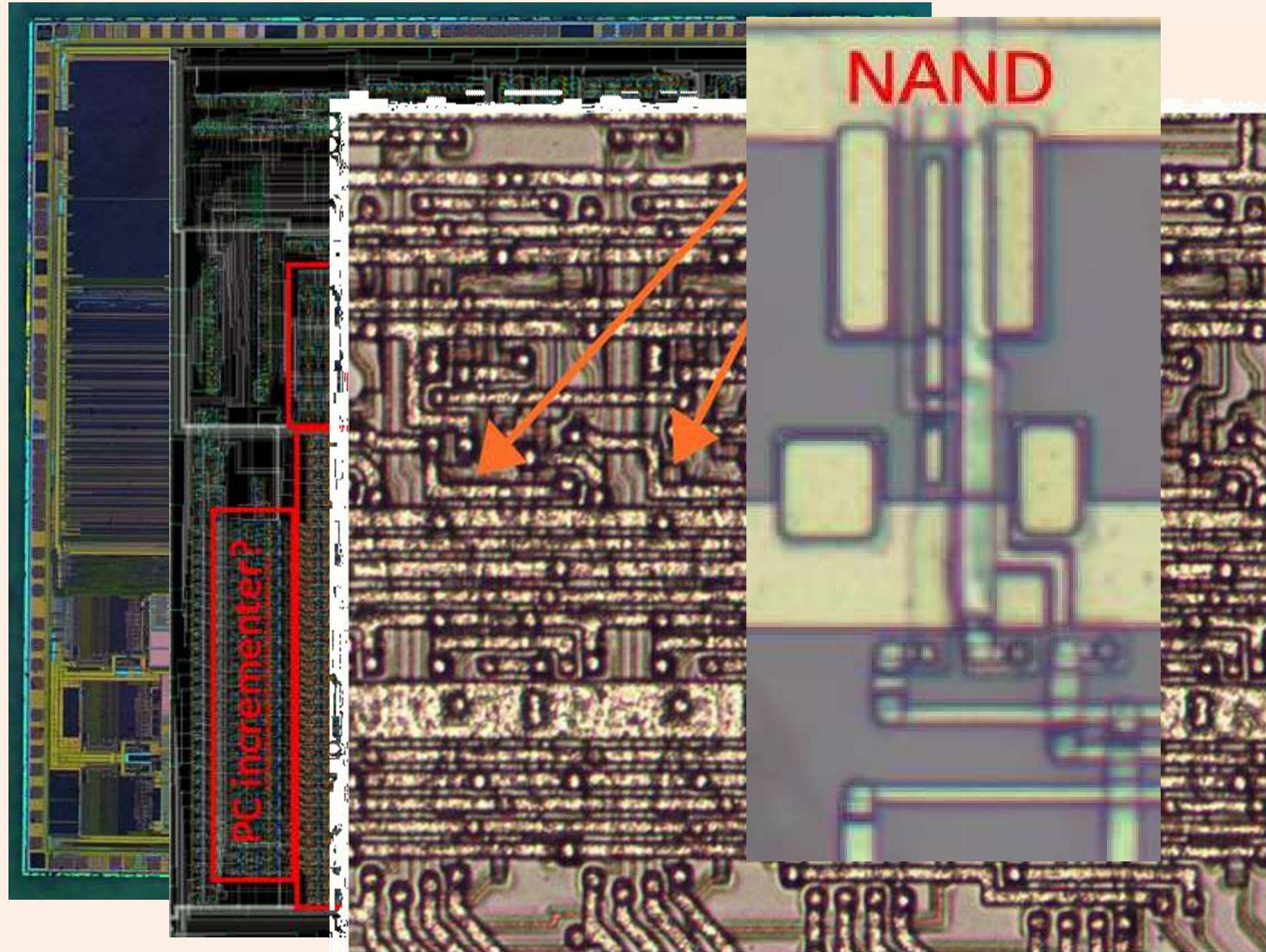
# Introdução aos Microcontroladores

---

Um microcontrolador (MCU) é um pequeno computador implementado num único Circuito Integrado tipicamente baseado num CPU (Central Processing Unit) relativamente simples, integrando um razoável número de periféricos, como:

- Memória RAM;
- Memória Flash;
- I/Os programáveis;
- Timers e contadores;
- A/D – Conversores Analógico/Digitais;
- D/A – Conversores Digital/Analógicos;
- Etc...

# Introdução aos Microcontroladores



# Como escolher um microcontrolador?

---

Características mais importantes:

- Arquitectura: nº de bits, ARM, RISC V, MIPS, ...
- Potência consumida + Clock freq./Desempenho
- Nº de I/O pins
- Tamanho memória programa
- Tamanho memória RAM
- Possibilidades Analógicas:
  - Nº de ADCs + resolução + tempo de conversão
  - Nº de DACs + resolução + tempo de conversão
  - Nº de comparadores analógicos, opamps, etc...
- Nº de Timers + resolução
- Funções especiais

# Como escolher um microcontrolador?

---

Características mais importantes:

Arquitectura: nº de bits, ARM, RISC V, MIPS, ...

- 8-bits é excelente para pequenas aplicações;
- 16-bit ok para aplicações mais complicadas;
- 32-bit para aplicações exigentes;
- ARM é a dominante para 32 bits.

# Como escolher um microcontrolador?

---

Características mais importantes:

Potência consumida + Clock freq./Desempenho

- Mais clock -> Melhor desempenho;
- Mais clock -> Maior consumo;
- Maior consumo -> Menor duração da bateria.

# Como escolher um microcontrolador?

---

Características mais importantes:

## Nº de I/O pins

- Mais pinos -> Mais possibilidades de ligar periféricos;
- Mais pinos -> packages maiores -> maior custo;
- Mais pinos nem sempre implica maior desempenho.

# Como escolher um microcontrolador?

---

Características mais importantes:

Tamanho memória programa (memória não volátil - actualmente quase sempre do tipo FLASH)

- Mais memória -> Permite maiores programas (o que por sua vez permite algoritmos mais complexos ou que usem mais dados fixos);
- Mais memória nem sempre implica maior desempenho.

# Como escolher um microcontrolador?

---

Características mais importantes:

## Tamanho memória RAM (volátil)

- Mais memória RAM-> Melhor desempenho em programs mais complexos;
- Mais memória permite mais fácil processamento de grande quantidade de dados.

# Como escolher um microcontrolador?

---

Características mais importantes:

## Possibilidades Analógicas

- Muito importante no interface e interacção com o mundo;
- A maioria das grandezas a medir são analógicas e precisam de ser convertidas em representação digital para processamento por um computador (é semelhante para a actuação);
- Nº ADCs, sua resolução e canais de entrada;
- Nº de DACs e sua resolução.

# Como escolher um microcontrolador?

---

Características mais importantes:

Nº de Timers + resolução

- Muito importante para sistemas que são “disciplinados” por tempo;
- Muito importante para contagem muito precisa de tempo e eventos.

# Como escolher um microcontrolador?

---

Características mais importantes:

## Funções especiais

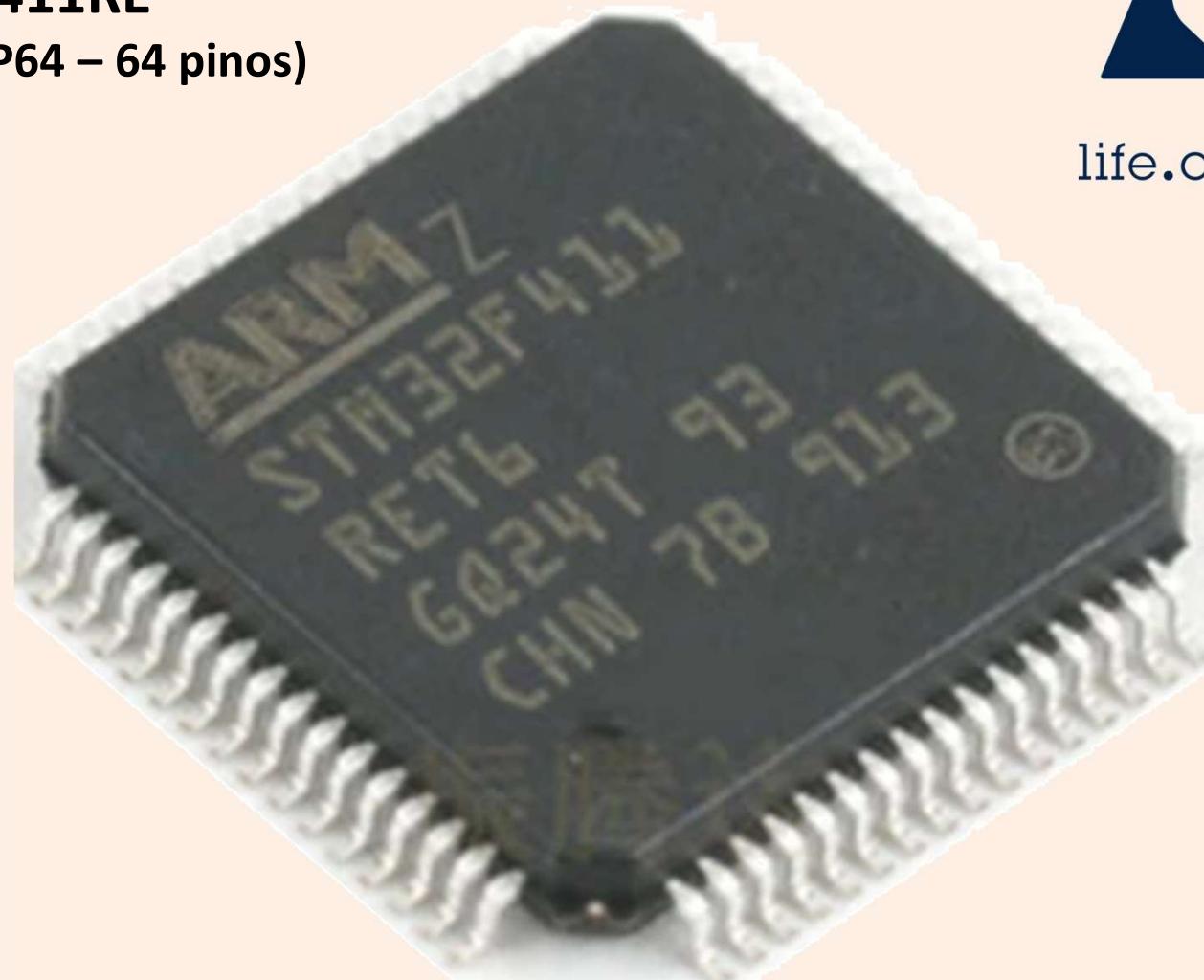
- Há muitas funções especiais.
- Por exemplo: comunicações:
  - **USB**
  - **U(S)ART**
  - **CAN**
  - **LIN**
  - **BLE**
  - **WiFi**
  - **Etc...**

# Introdução aos Microcontroladores

---

Chip:

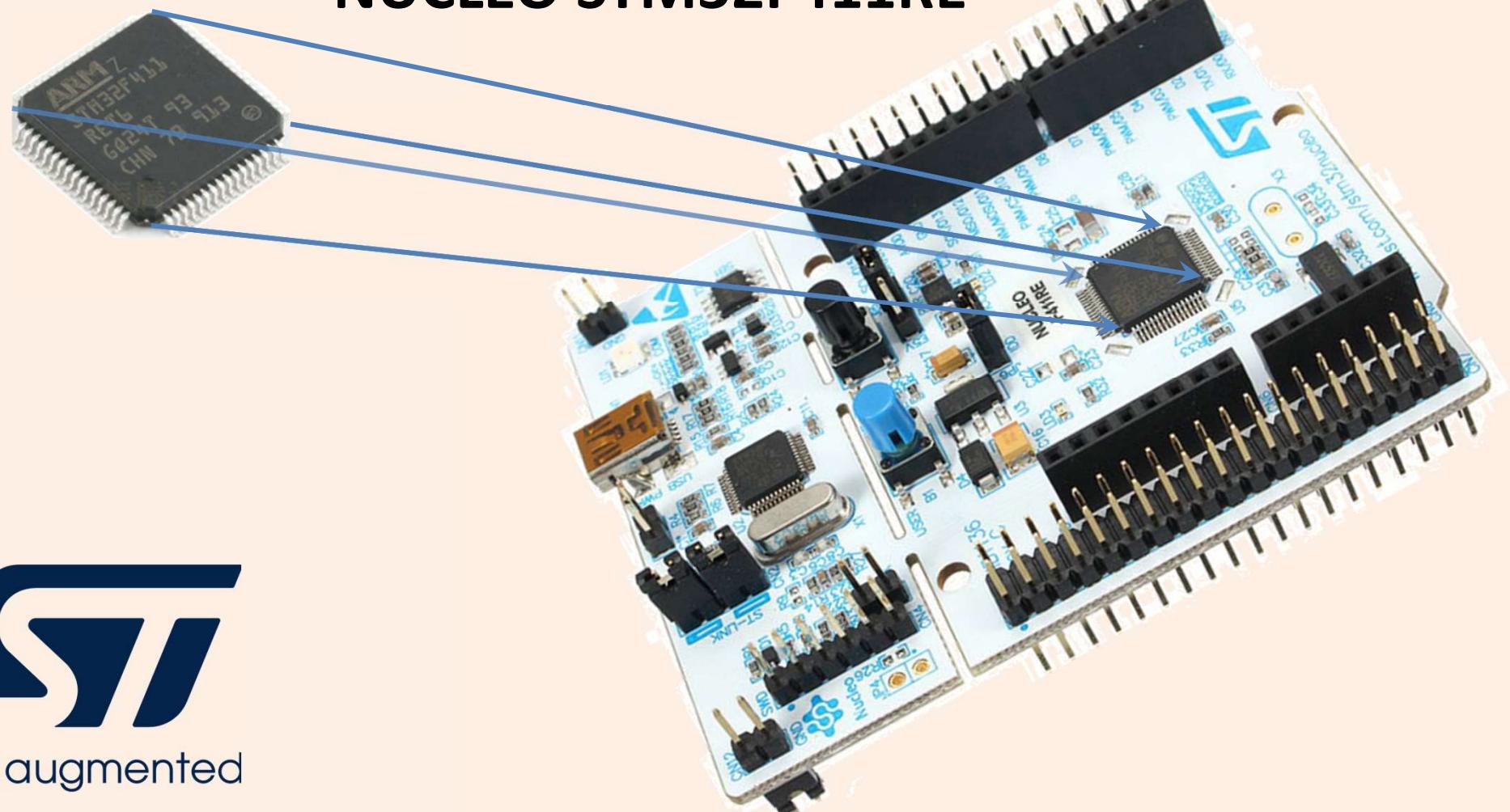
**STM32F411RE**  
(LQFP64 – 64 pinos)



# Introdução aos Microcontroladores

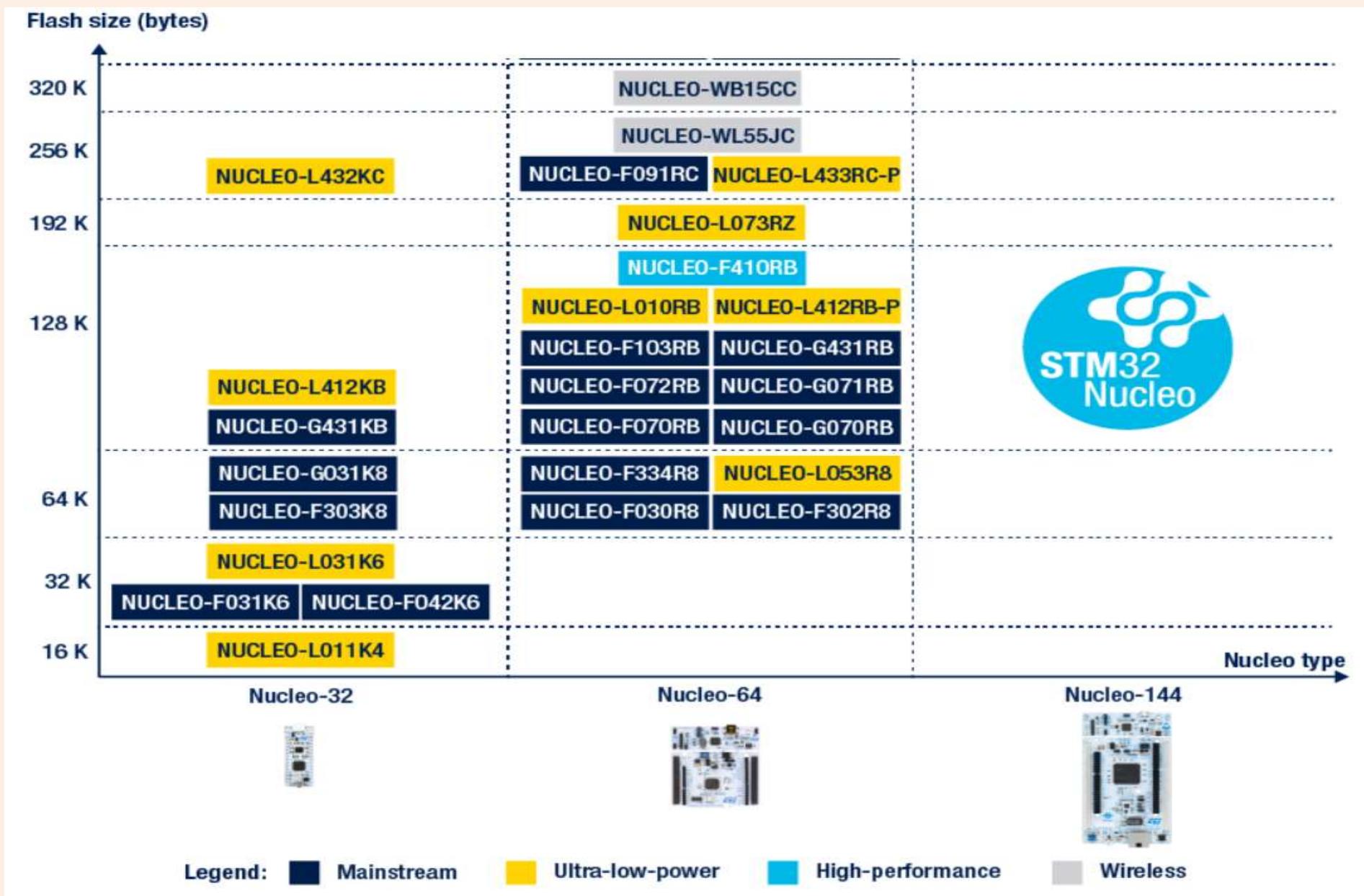
Placa desenvolvimento:

**NUCLEO STM32F411RE**

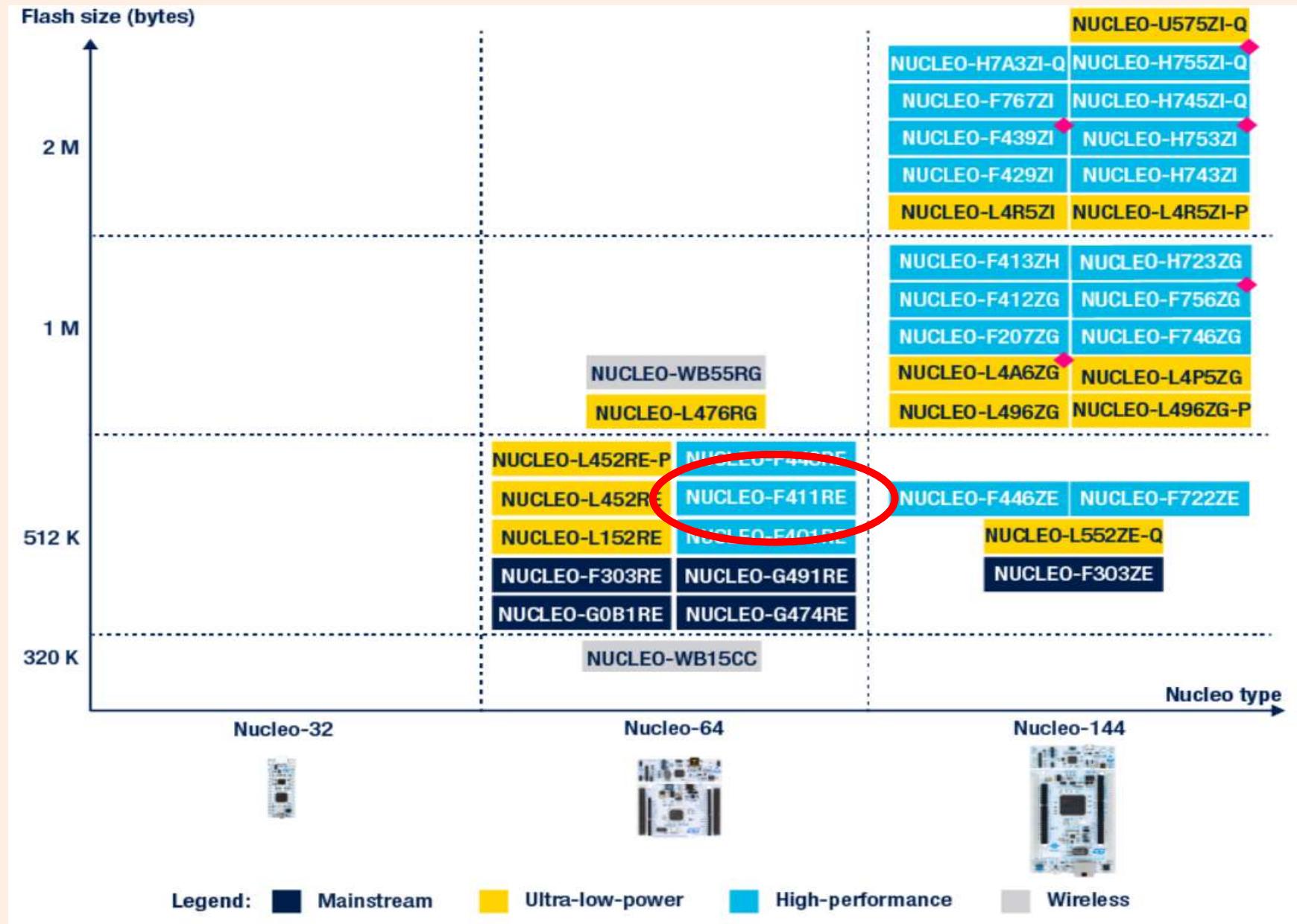


**ST**  
life.augmented

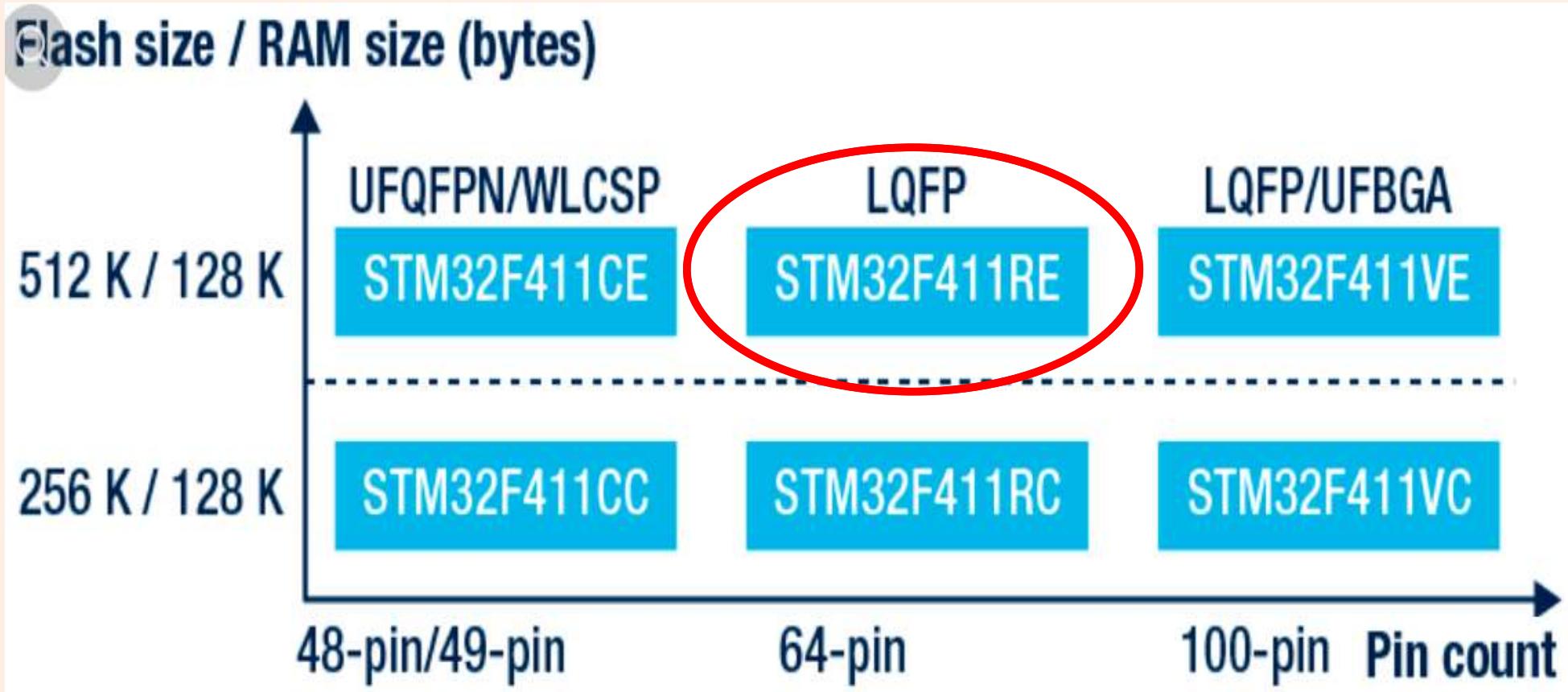
# Introdução aos Microcontroladores



# Introdução aos Microcontroladores



# Introdução aos Microcontroladores

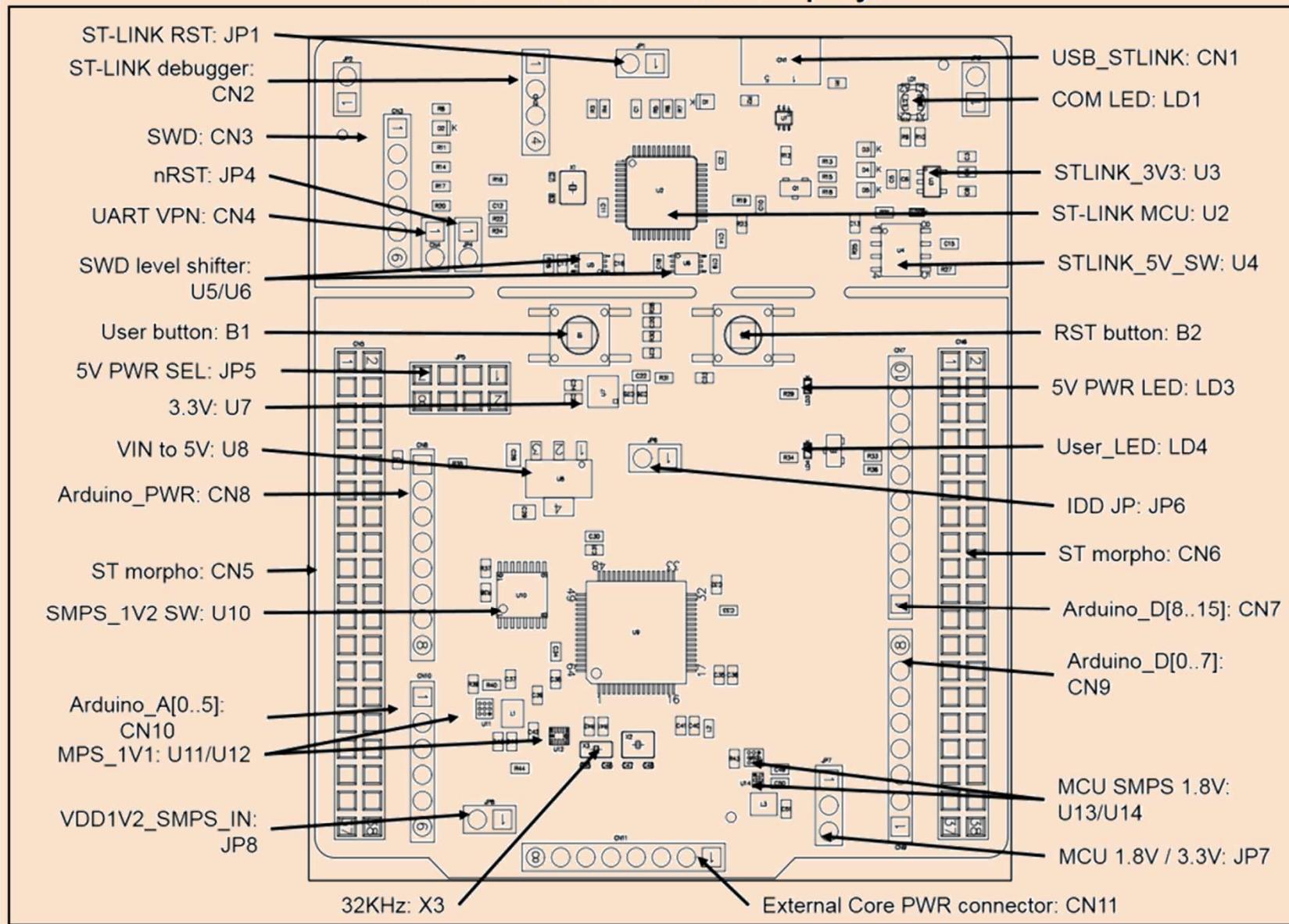


# Introdução aos Microcontroladores

System	ART Accelerator™	512-Kbyte Flash memory	Control
Power supply 1.2 V internal regulator POR/PDR/PVD/BOR	100 MHz ARM® Cortex®-M4 CPU	128-Kbyte SRAM	5x 16-bit timer
Xtal oscillators 32 kHz + 4 ~26 MHz	Floating Point Unit (FPU)	80-byte backup data	1x 16-bit motor control PWM synchronized AC timer
Internal RC oscillators 32 kHz + 16 MHz	Nested Vector Interrupt Controller (NVIC)		2x 32-bit timer
PLL	JTAG/SW debug		
Clock control	Embedded Trace Macrocell (ETM)		
RTC/AWU	Memory Protection Unit (MPU)		
2x watchdogs (independent and window)	AHB-Lite bus matrix	3x I²C	
36/50/81 I/Os	APB bus	3x USART LIN, smartcard, IrDA, modem control	
Cyclic Redundancy Check (CRC)	16-channel DMA with Batch Acquisition Mode (BAM)	5x SPI or 5x I²S (2x I²S with full duplex)	
96-bit unique ID		SDIO	
Voltage scaling		USB 2.0 OTG FS	
Connectivity			Analog
			1x 12-bit ADC 2.4 MSPS 16 channels / 0.41µs
			Temperature sensor

# Introdução aos Microcontroladores

STM32 Nucleo-64-P board top layout



# Get Started

---

Procedimento:

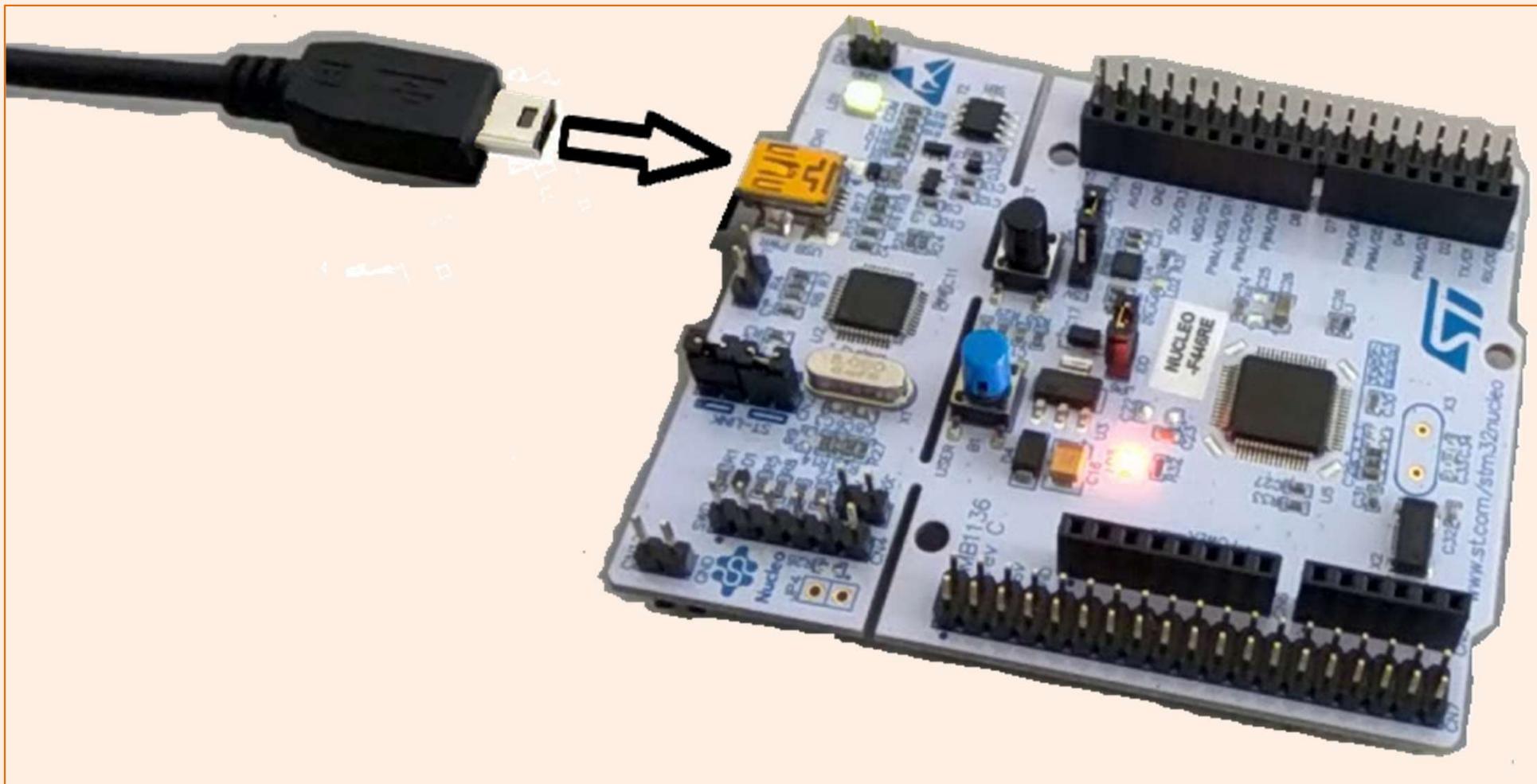
**É só ligar a placa a um PC através de um cabo USB – Mini-USB**



# Get Started

## Procedimento:

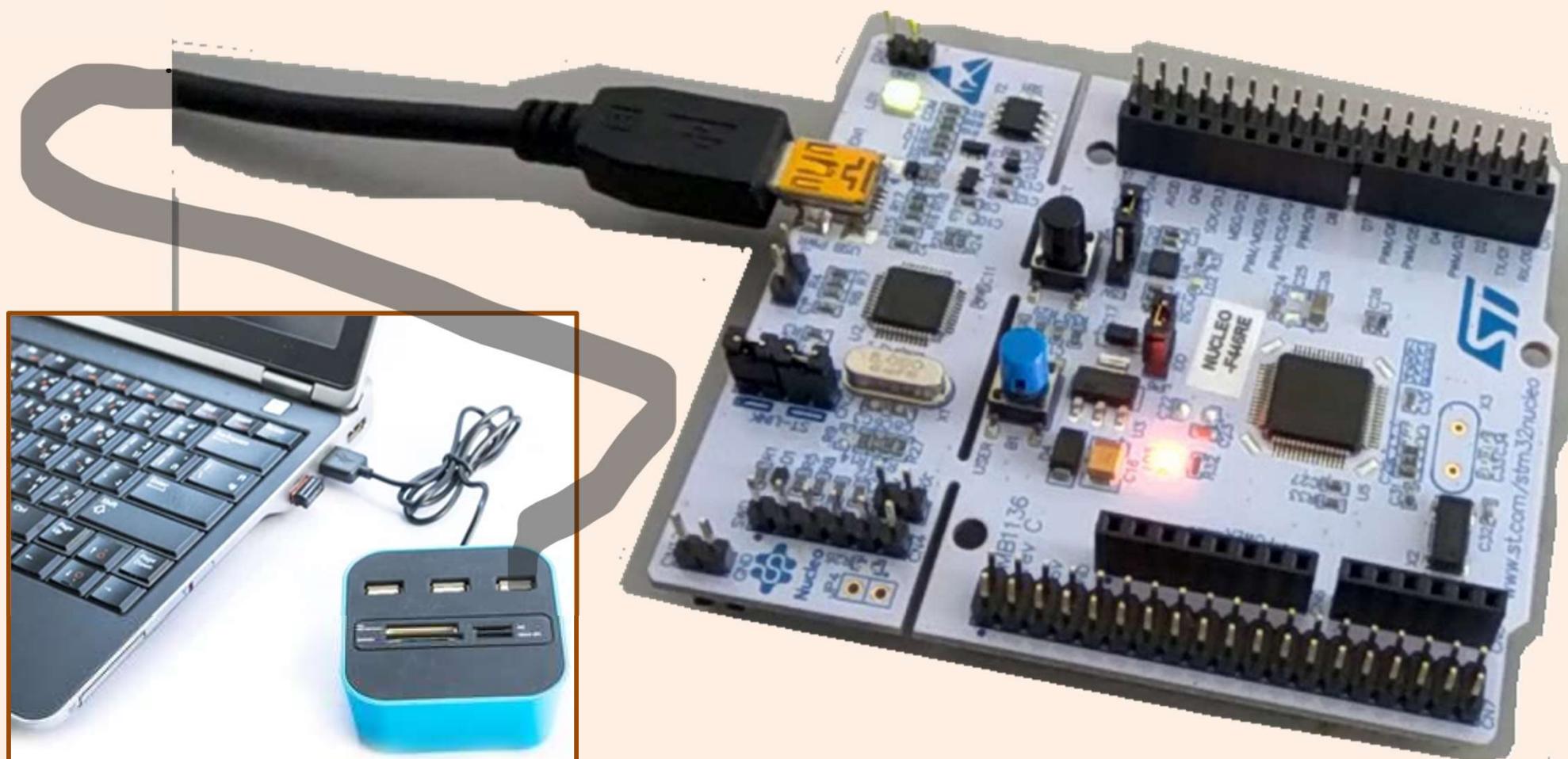
**É só ligar a placa a um PC através de um cabo USB – Mini-USB**



# Get Started

Procedimento:

**É só ligar a placa a um PC (se tiverem um Hub ficam mais protegidos contra acidentes)**



# Get Started

---

Procedimento:

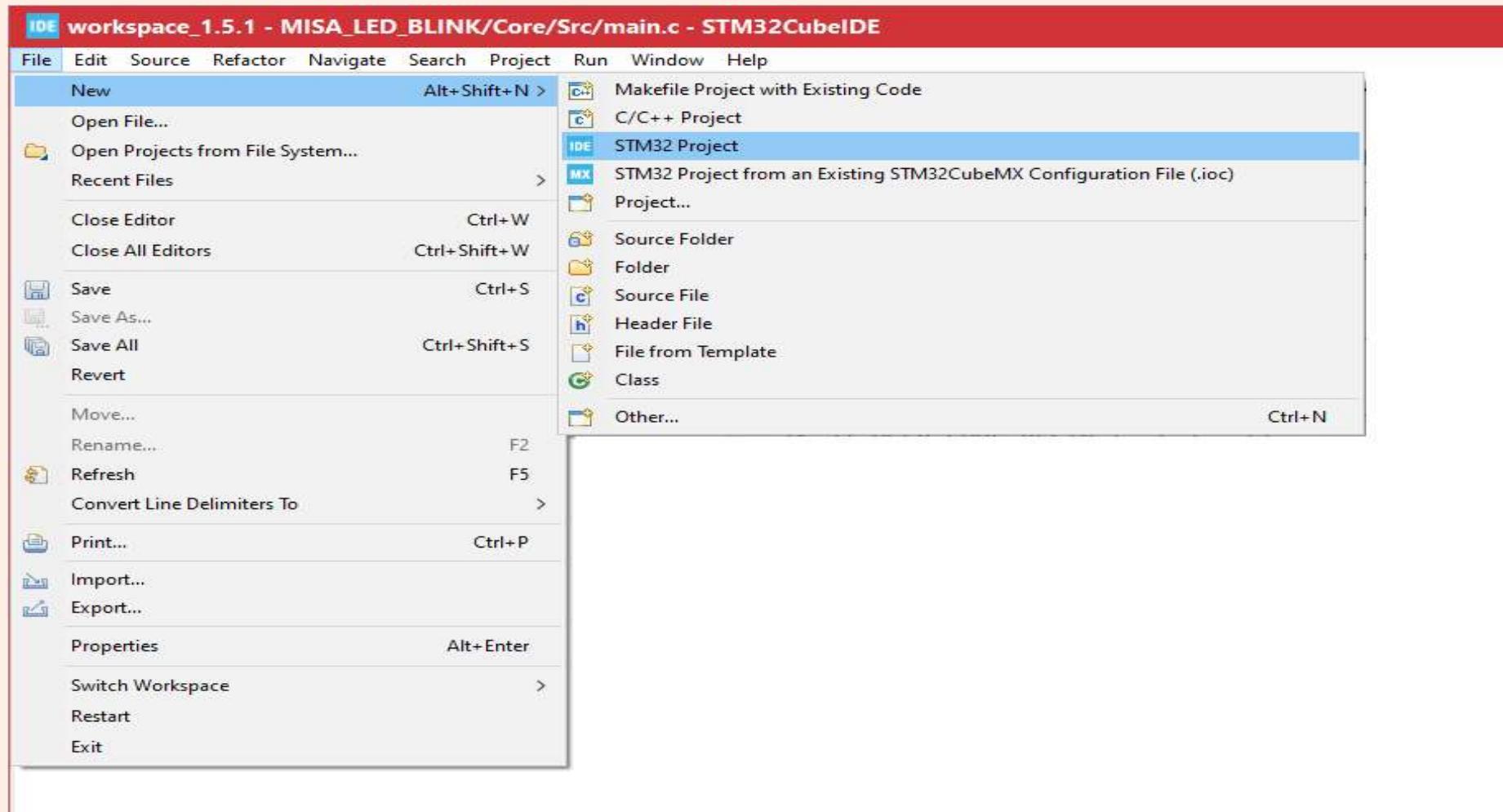
## Arrancar o IDE



# Get Started

Procedimento:

## Criar um Projeto tipo “STM32”



# Get Started

Procedimento:

## Seleccionar a placa: NUCLEO-F411RE

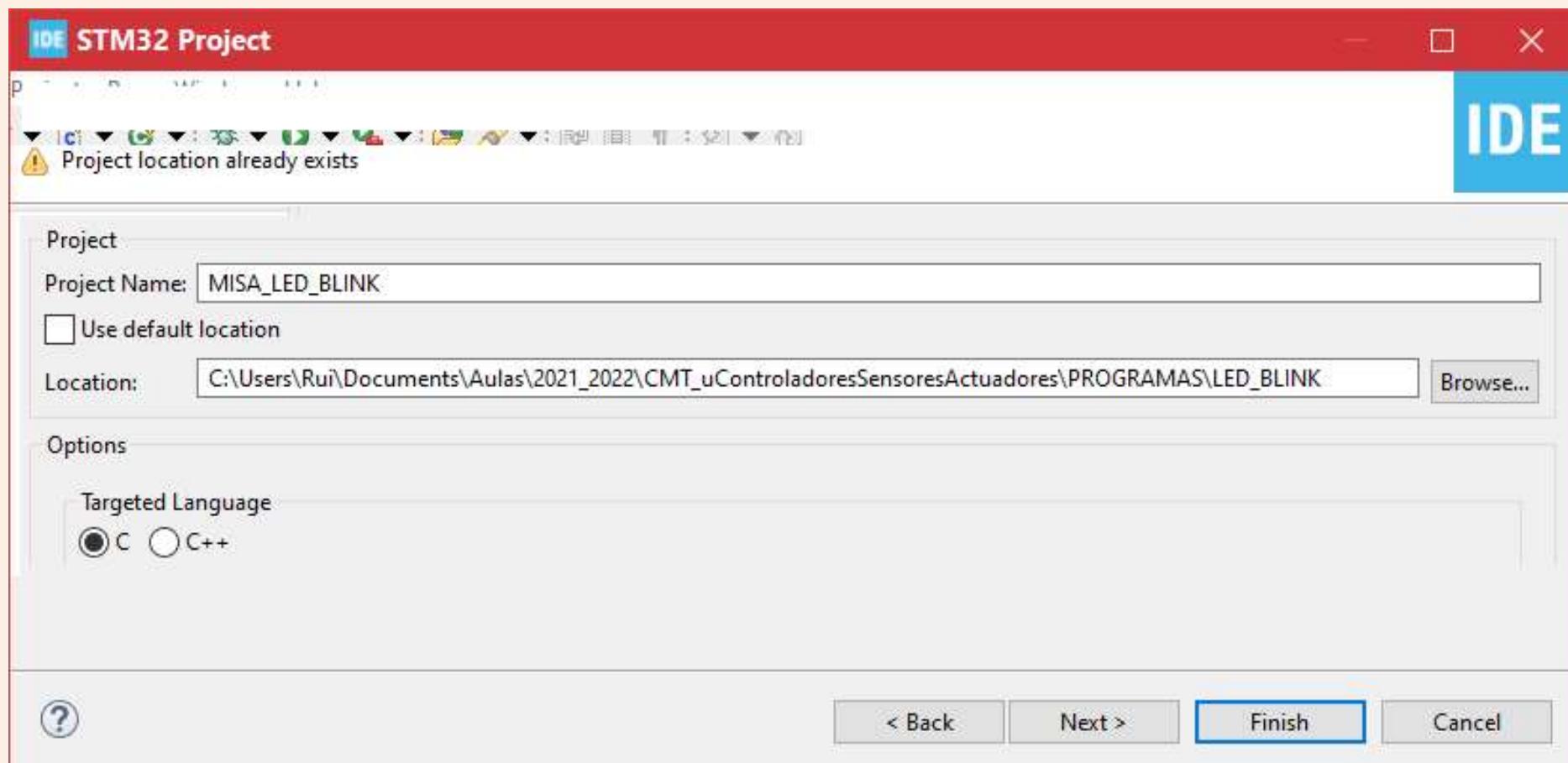
The screenshot shows the STMicroelectronics Board Selector interface. At the top, there are four tabs: MCU/MPU Selector, Board Selector (which is selected), Example Selector, and Cross Selector. Below the tabs is a 'Board Filters' section with dropdown menus for Commercial Part Number, Vendor, Type, MCU/MPU Series, Other, and Peripheral. To the right of the filters is a navigation bar with 'Features' (selected), Large Picture, Docs & Resources, Datasheet (with a download icon), and Buy. A 'STM32F4 Series' section highlights the 'NUCLEO-F411RE'. Below this is a title 'STMicroelectronics NUCLEO-F411RE Board Support and Examples'. A table titled 'Boards List: 158 items' displays seven rows of boards, with the 'NUCLEO-F411RE' row being highlighted. The columns in the table are: Overview (with a thumbnail image), Commercial Part ..., Type, Marketing Status, Unit Price (US\$), and Mounted Device. The 'NUCLEO-F411RE' row shows the following details: Overview thumbnail, Commercial Part ... (NUCLEO-F411RE), Type (Nucleo-64), Marketing Status (Active), Unit Price (US\$) (13.0), and Mounted Device (STM32F411RETx). At the bottom of the interface are buttons for Next >, Finish, and Cancel.

*	Overview	Commercial Part ...	Type	Marketing Status	Unit Price (US\$)	Mounted Device
★		NUCLEO-F334R8	Nucleo-64	Active	10.32	STM32F334R8Tx
★		NUCLEO-F401RE	Nucleo-64	Active	13.0	STM32F401RETx
★		NUCLEO-F410RB	Nucleo-64	Active	13.0	STM32F410RBTx
★		NUCLEO-F411RE	Nucleo-64	Active	13.0	STM32F411RETx
★		NUCLEO-F412ZG	Nucleo-144	Active	19.0	STM32F412ZGTx
★		NUCLEO-F413ZH	Nucleo-144	Active	19.0	STM32F413ZHTx

# Get Started

Procedimento:

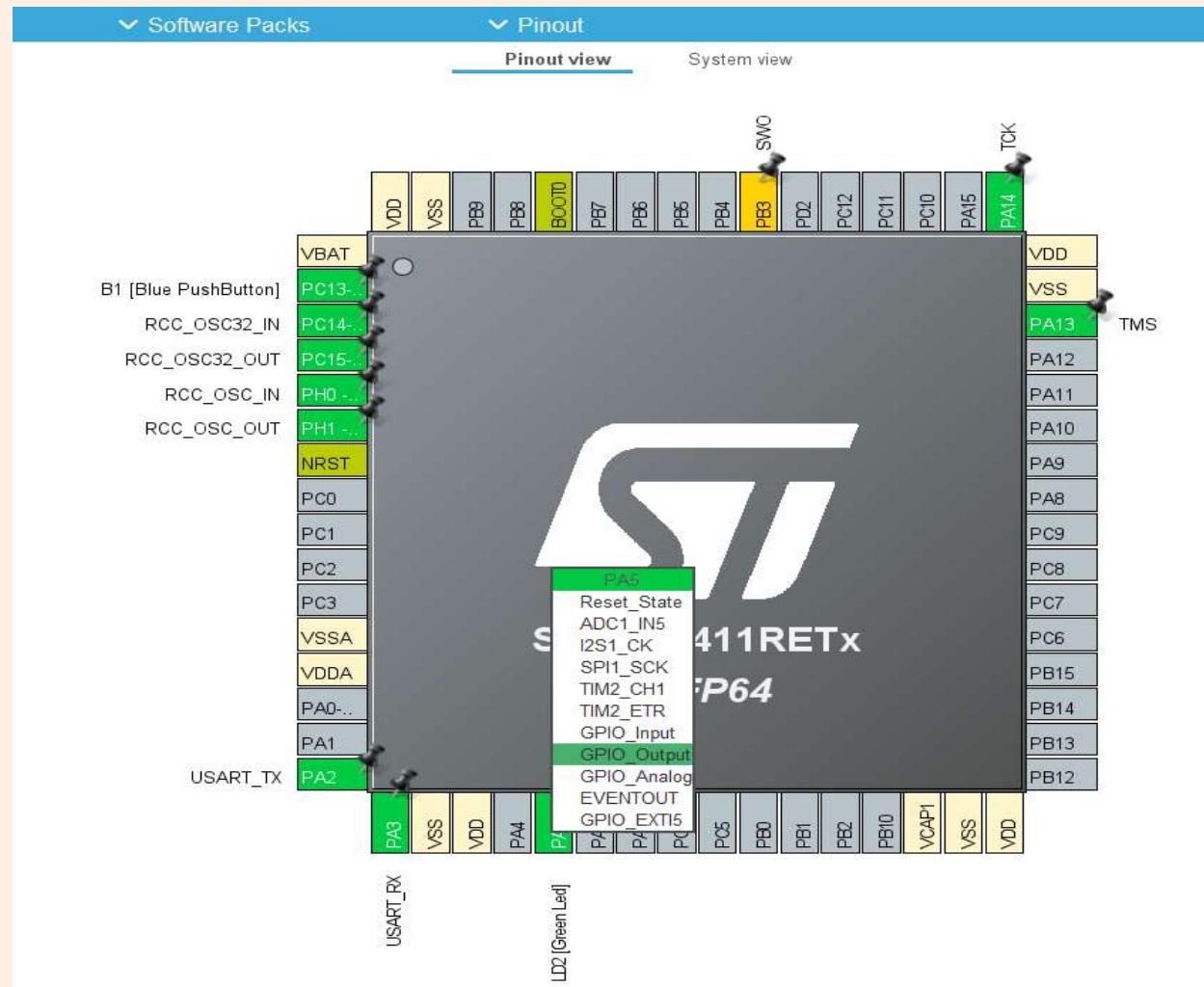
## Selecionar nome e local destino



# Get Started

Procedimento:

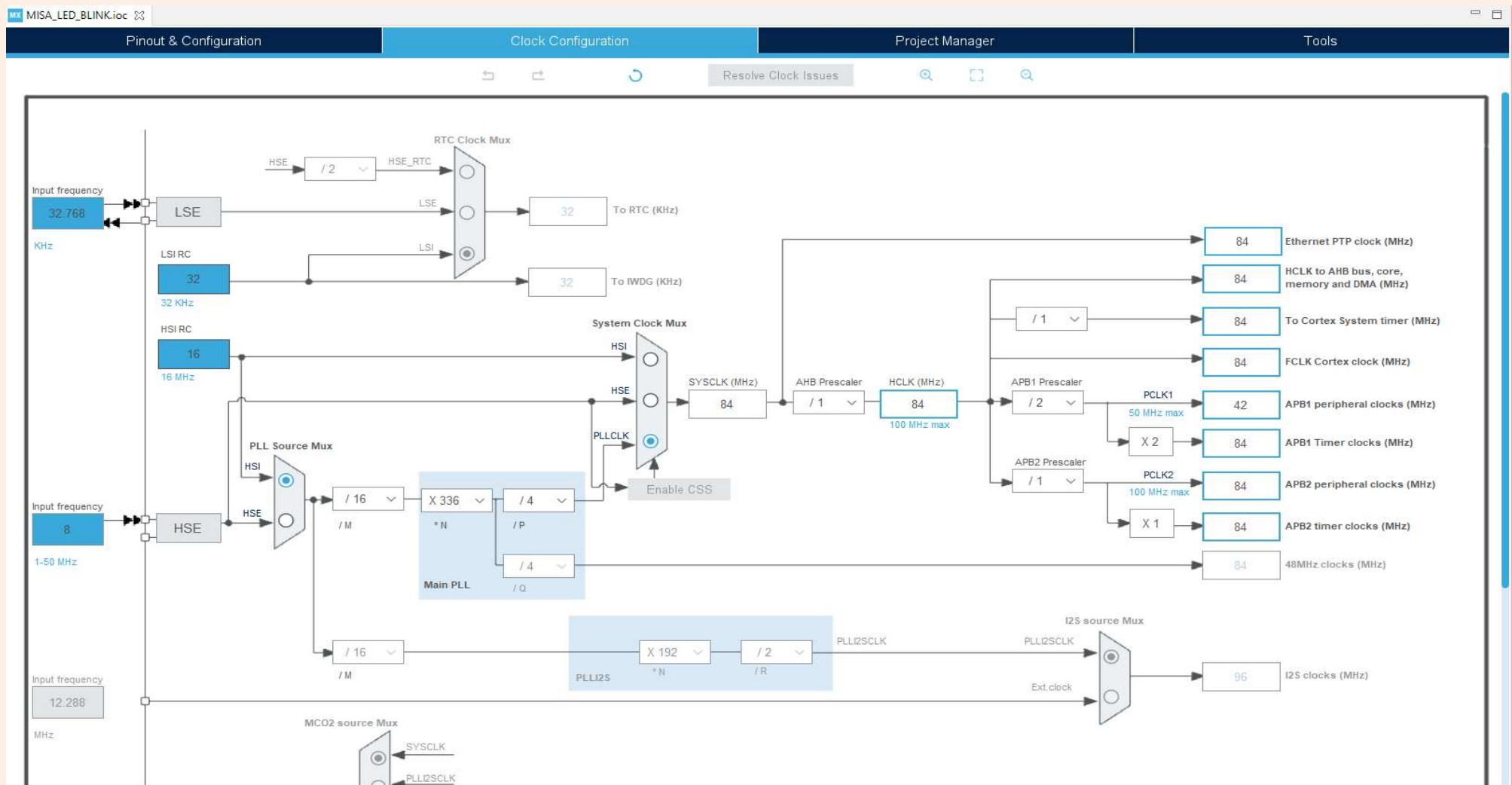
## Selecionar pinos e funções



# Get Started

Procedimento:

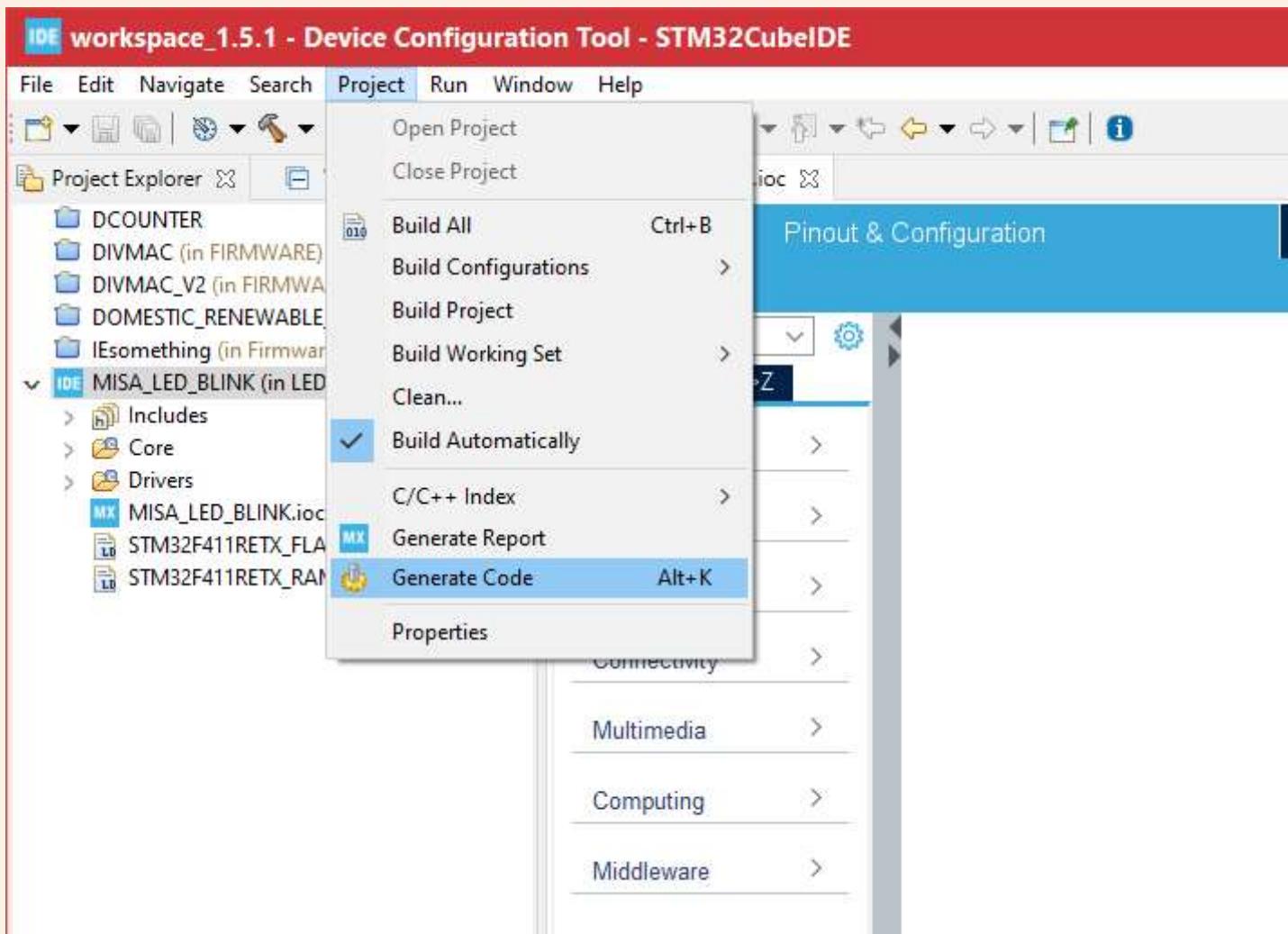
## Selecionar oscilador e frequência de relógio



# Get Started

Procedimento:

## Gerar Código (automaticamente)



# STM32CubeIDE

The screenshot shows the STM32CubeIDE interface with the following details:

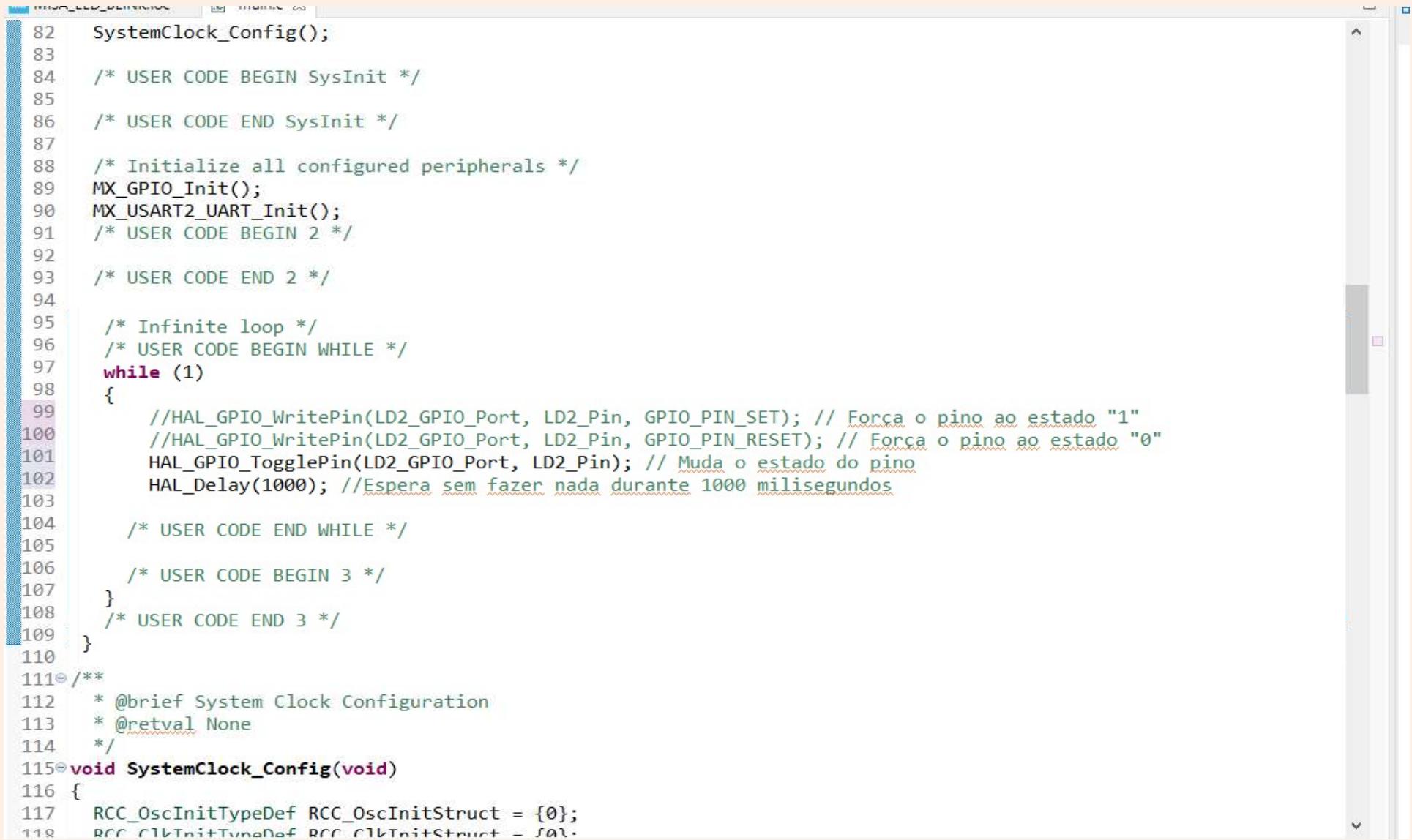
- Title Bar:** workspace\_1.5.1 - MISA\_LED\_BLINK/Core/Src/main.c - STM32CubeIDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbars:** Standard toolbar with icons for file operations, search, and project management.
- Project Explorer:** Shows C/C++ Projects with the following items:
  - DCOUNTER
  - DIVMAC
  - DIVMAC\_V2
  - DOMESTIC\_RENEWABLE\_SYSTEM
  - IEsomething
  - MISA\_LED\_BLINK (selected)
- Code Editor:** The main.c file is open, displaying the following code snippet:

```
1 /* USER CODE BEGIN Header */
2 /**
3  * @file      : main.c
4  * @brief     : Main program body
5  * @attention
6  *
7  * <h2><center>&copy; Copyright (c) 2022 STMicroelectronics.
8  * All rights reserved.</center></h2>
9  *
10 * This software component is licensed by ST under BSD 3-Clause license,
11 * the "License"; You may not use this file except in compliance with the
12 * License. You may obtain a copy of the License at:
13 *          opensource.org/licenses/BSD-3-Clause
14 *
15 ****
16 */
17 /* USER CODE END Header */
18 /* Includes -----*/
19 #include "main.h"
20
21 /* Private includes -----*/
22 /* USER CODE BEGIN Includes */
23
24 /* USER CODE END Includes */
25
26 /* Private typedef -----*/
27 /* USER CODE BEGIN PTD */
28
29 /* USER CODE END PTD */
30
31 /* USER CODE END PTD */
32
33 /* Private define -----*/
34 /* USER CODE BEGIN PD */
35 /* USER CODE END PD */
36
37 /* Private macro -----*/
38
```

- Outline View:** Shows the structure of main.h, including function prototypes for SystemClock\_Config, MX\_GPIO\_Init, MX\_USART2\_UART\_Init, main, SystemClock\_Config, MX\_USART2\_UART\_Init, MX\_GPIO\_Init, Error\_Handler, and assert\_failed.
- Build Targets:** Shows the build targets for the project.
- Bottom Status Bar:** Displays tabs for Problems, Tasks, Console, Properties, Build Analyzer, Static Stack Analyzer, Search, Call Hierarchy, and other status indicators like Writable and Smart Insert.

# STM32CubeIDE

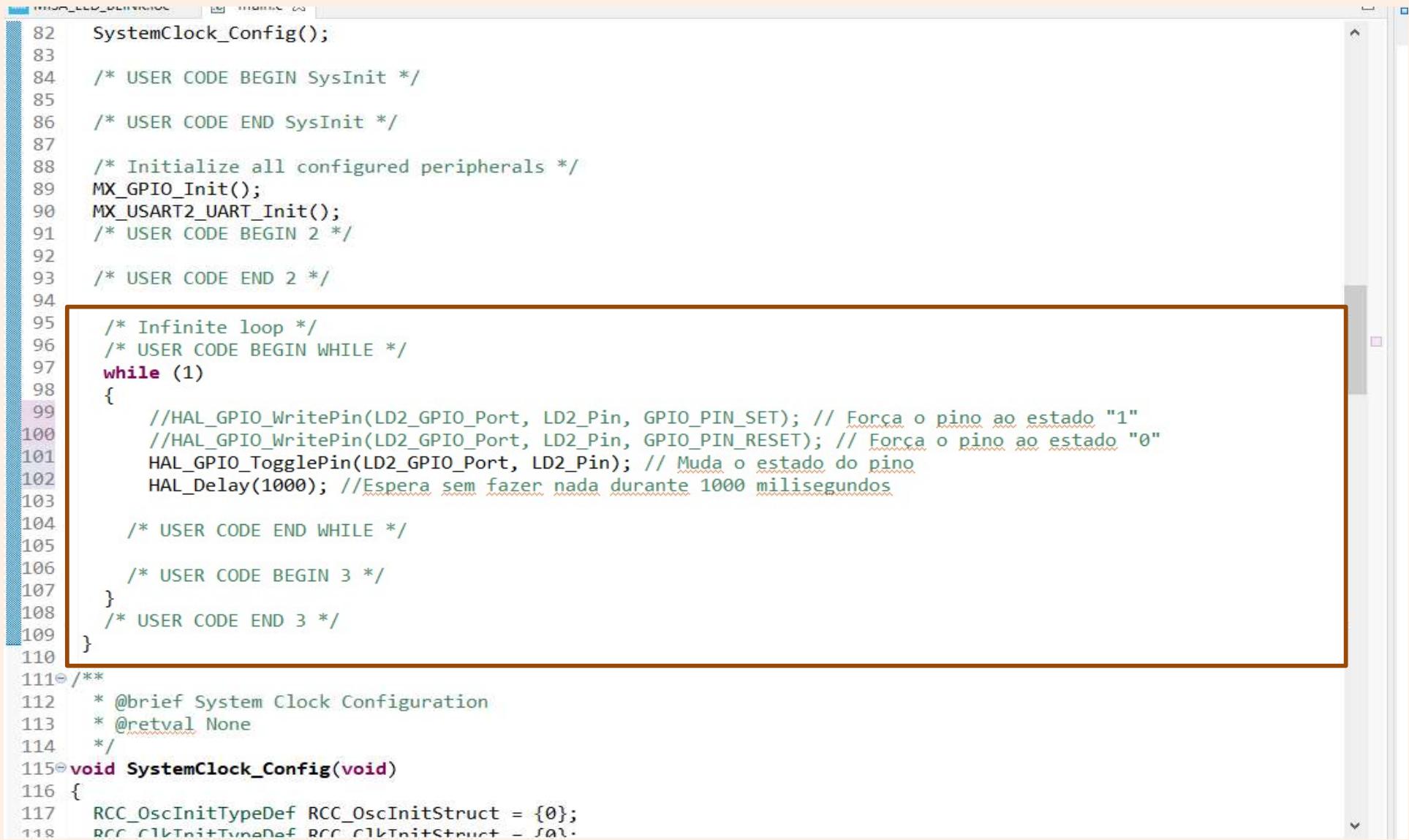
E agora é só escrever o Código...



```
82 SystemClock_Config();
83 
84 /* USER CODE BEGIN SysInit */
85 
86 /* USER CODE END SysInit */
87 
88 /* Initialize all configured peripherals */
89 MX_GPIO_Init();
90 MX_USART2_UART_Init();
91 /* USER CODE BEGIN 2 */
92 
93 /* USER CODE END 2 */
94 
95 /* Infinite loop */
96 /* USER CODE BEGIN WHILE */
97 while (1)
98 {
99     //HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_SET); // Força o pino ao estado "1"
100    //HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_RESET); // Força o pino ao estado "0"
101    HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin); // Muda o estado do pino
102    HAL_Delay(1000); //Espera sem fazer nada durante 1000 milisegundos
103 
104 /* USER CODE END WHILE */
105 
106 /* USER CODE BEGIN 3 */
107 }
108 /* USER CODE END 3 */
109 }
110 /**
111 * @brief System Clock Configuration
112 * @retval None
113 */
114 
115 void SystemClock_Config(void)
116 {
117     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
118     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
```

# STM32CubeIDE

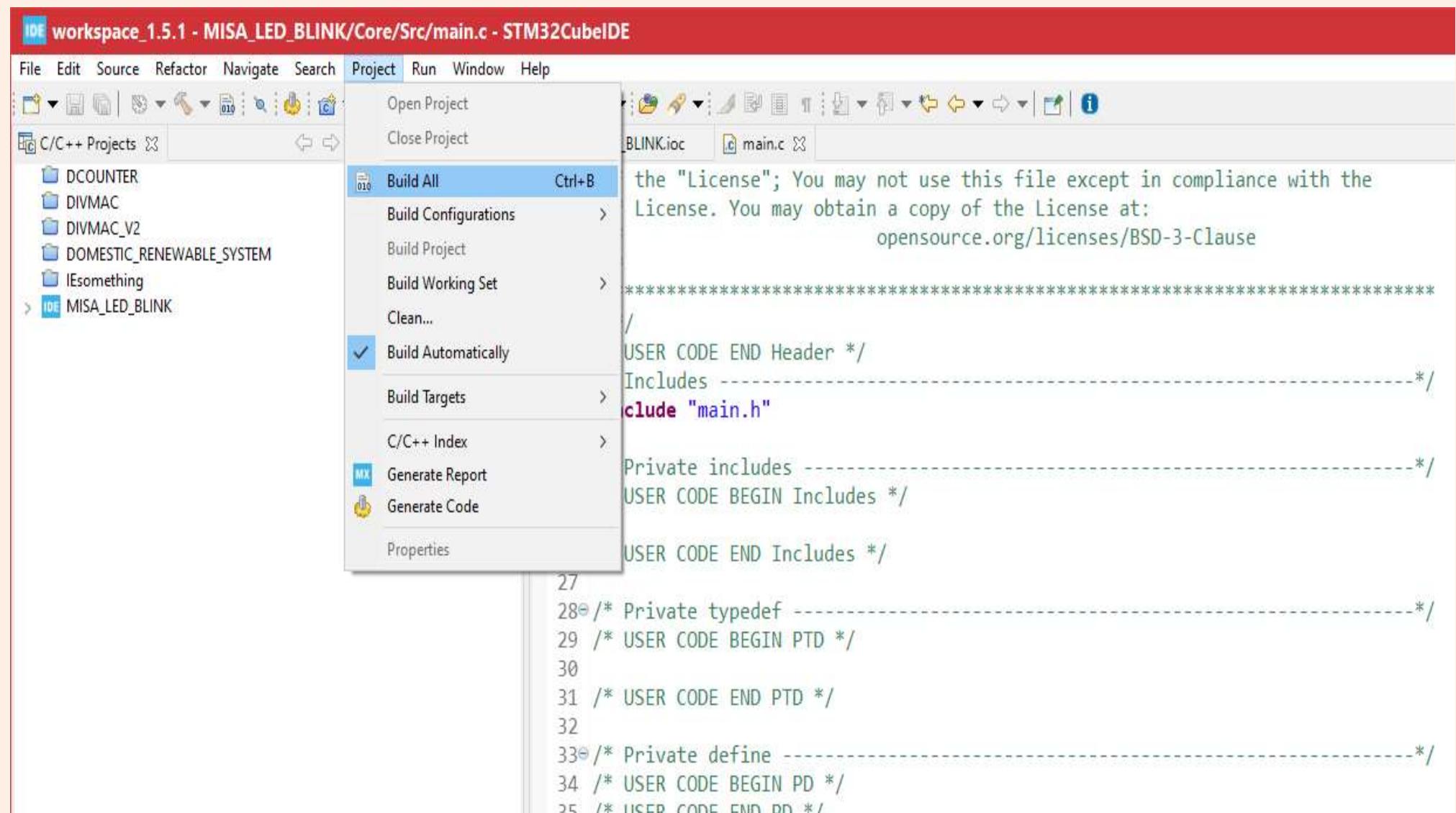
E agora é só escrever o Código...



```
82 SystemClock_Config();
83 
84 /* USER CODE BEGIN SysInit */
85 
86 /* USER CODE END SysInit */
87 
88 /* Initialize all configured peripherals */
89 MX_GPIO_Init();
90 MX_USART2_UART_Init();
91 /* USER CODE BEGIN 2 */
92 
93 /* USER CODE END 2 */
94 
95 /* Infinite loop */
96 /* USER CODE BEGIN WHILE */
97 while (1)
98 {
99     //HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_SET); // Força o pino ao estado "1"
100    //HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_RESET); // Força o pino ao estado "0"
101    HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin); // Muda o estado do pino
102    HAL_Delay(1000); //Espera sem fazer nada durante 1000 milisegundos
103 
104    /* USER CODE END WHILE */
105 
106    /* USER CODE BEGIN 3 */
107 }
108 /* USER CODE END 3 */
109 }
110 /**
111 * @brief System Clock Configuration
112 * @retval None
113 */
114 
115 void SystemClock_Config(void)
116 {
117     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
118     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
```

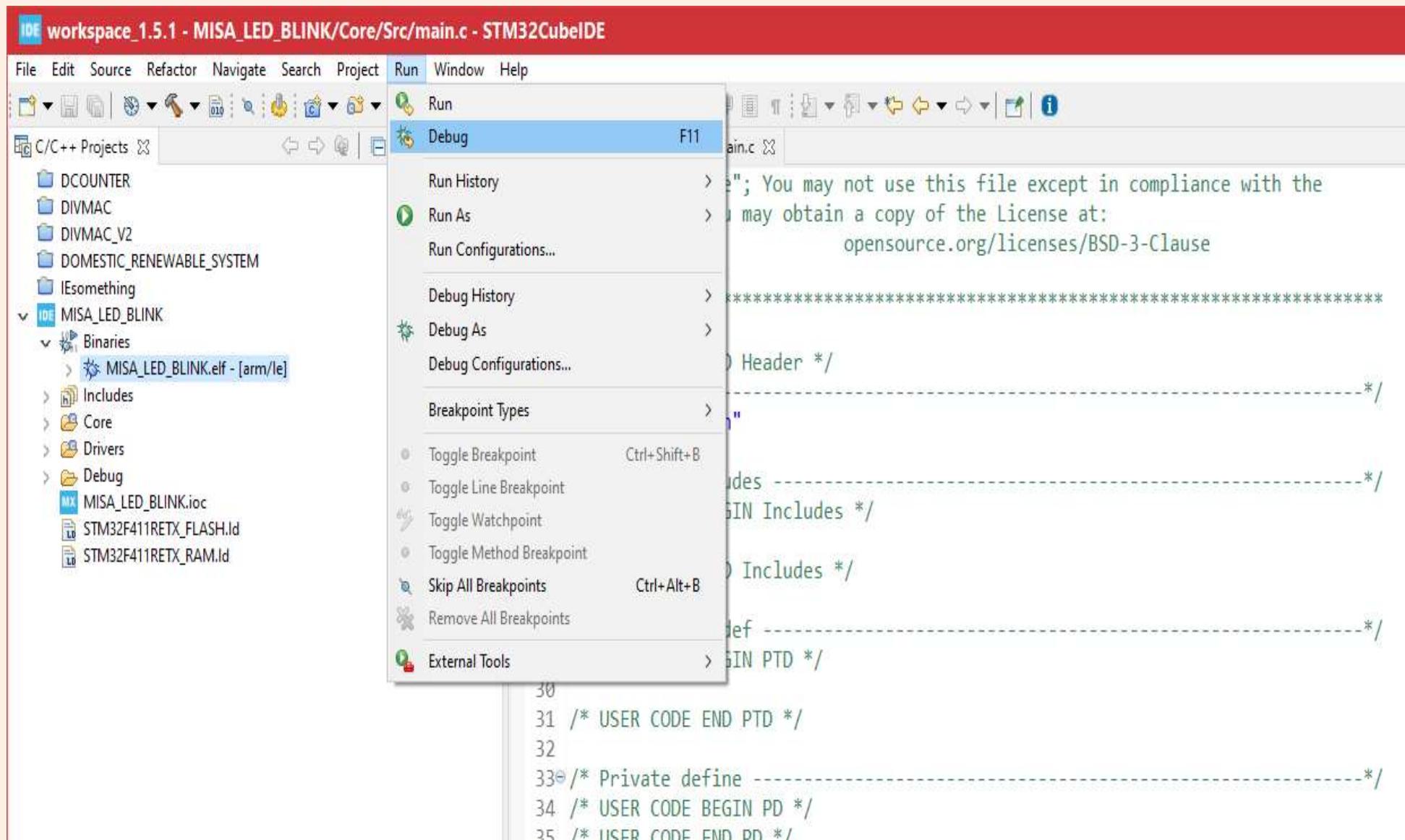
# STM32CubeIDE

Compilar o Código e Criar executável (selecionar **Build All**)



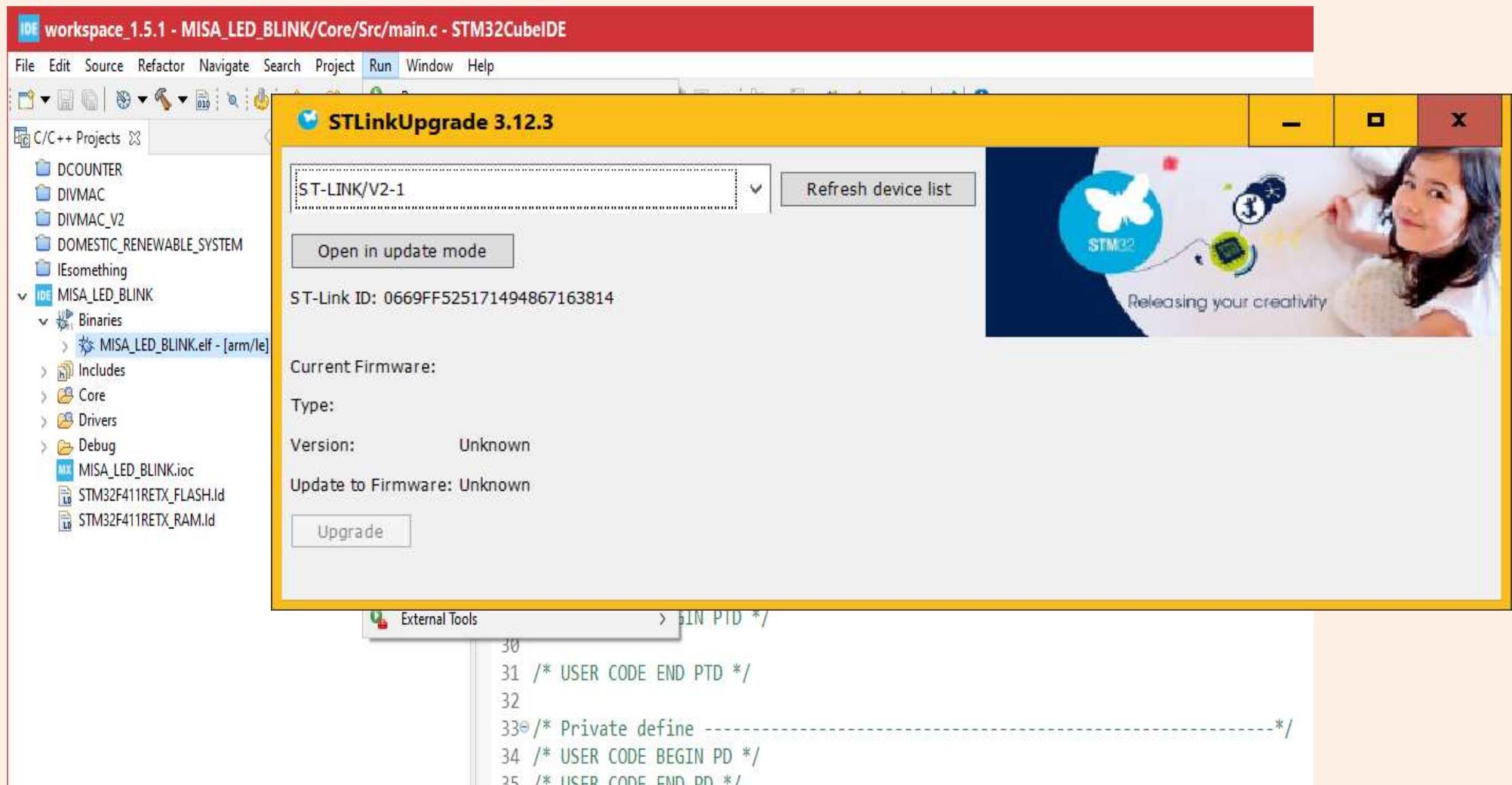
# STM32CubeIDE

Para enviar o executável para o microcontrolador fazer **Debug**.



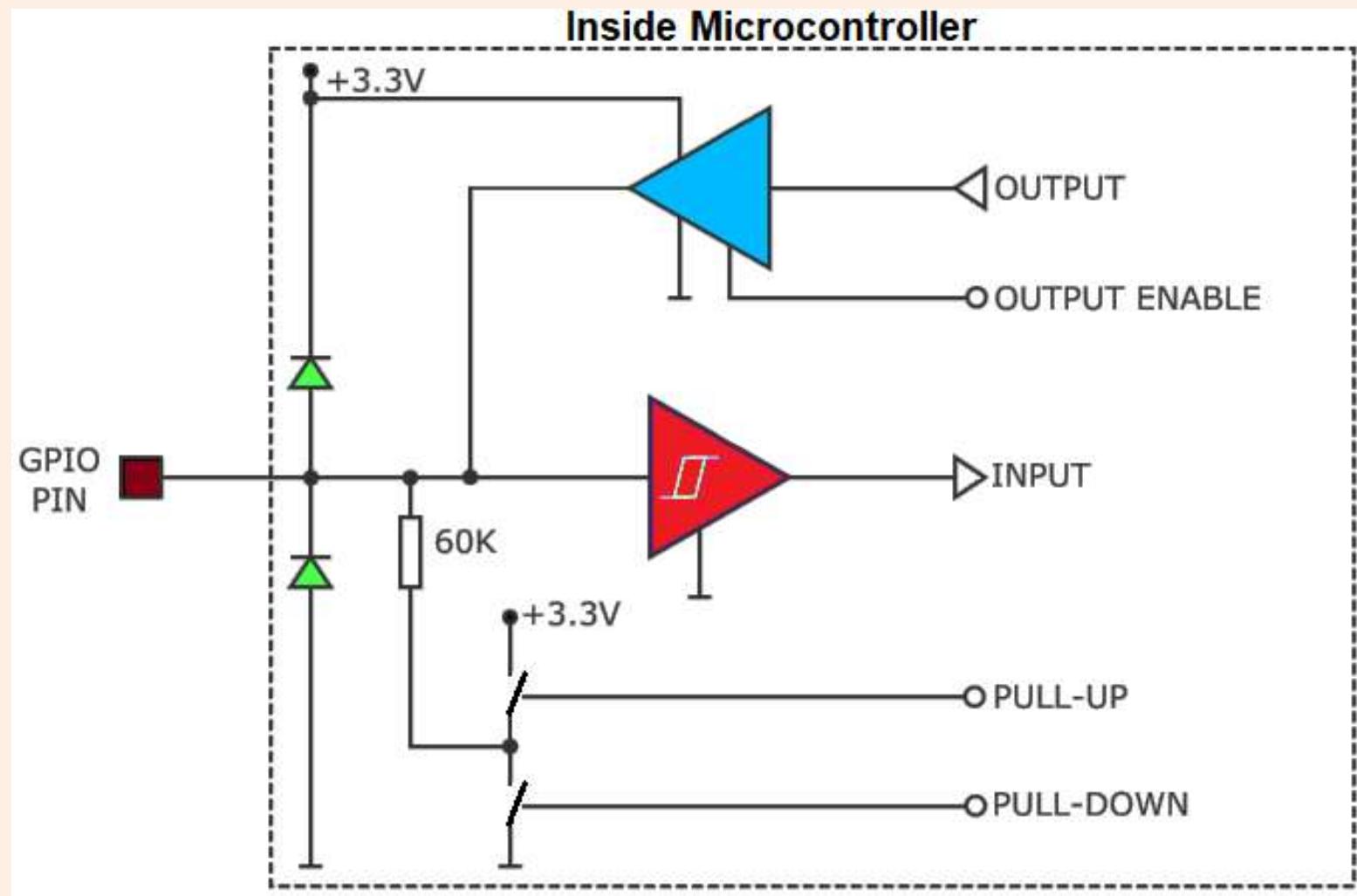
# STM32CubeIDE

Pode acontecer que o firmware da placa de desenvolvimento já seja antigo, pelo que possa necessitar de ser actualizado. Se aparecer a janela da figura, carreguem em “Open in update mode” e depois no botão “Update”



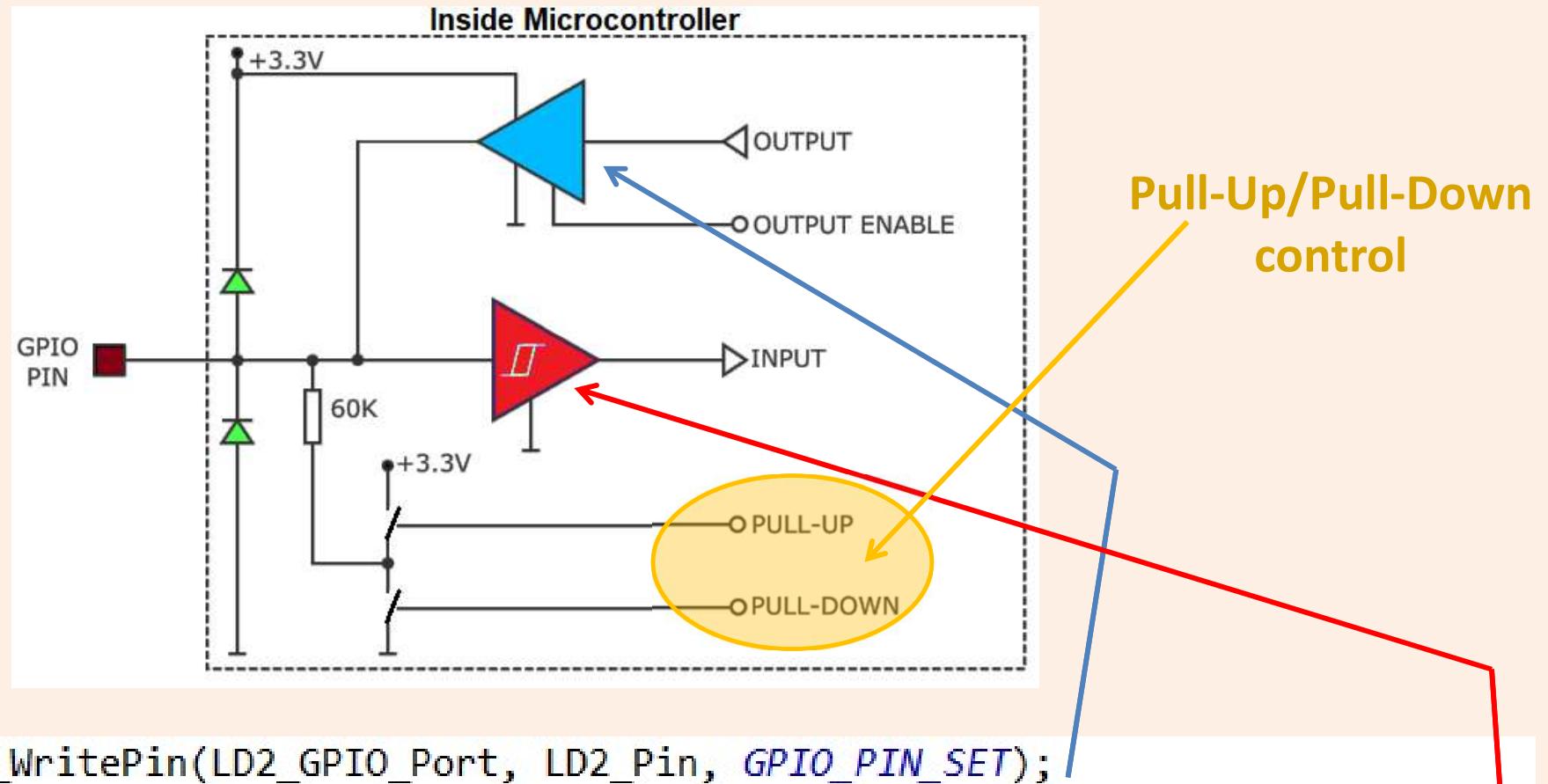
# STM32 General Purpose I/O Pin (GPIO)

GPIO pin – General Purpose Input/Output pin

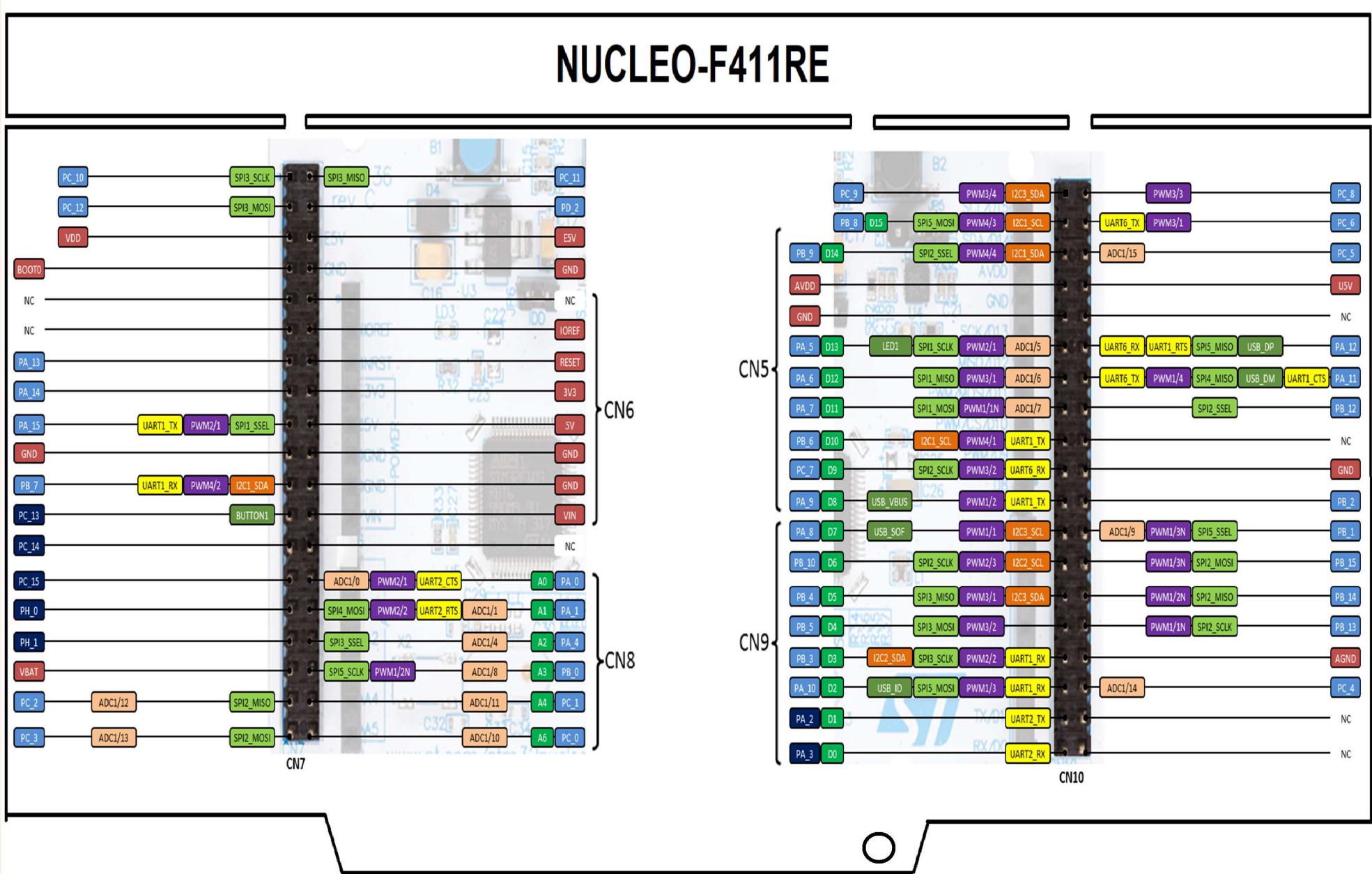


# STM32 General Purpose I/O Pin (GPIO)

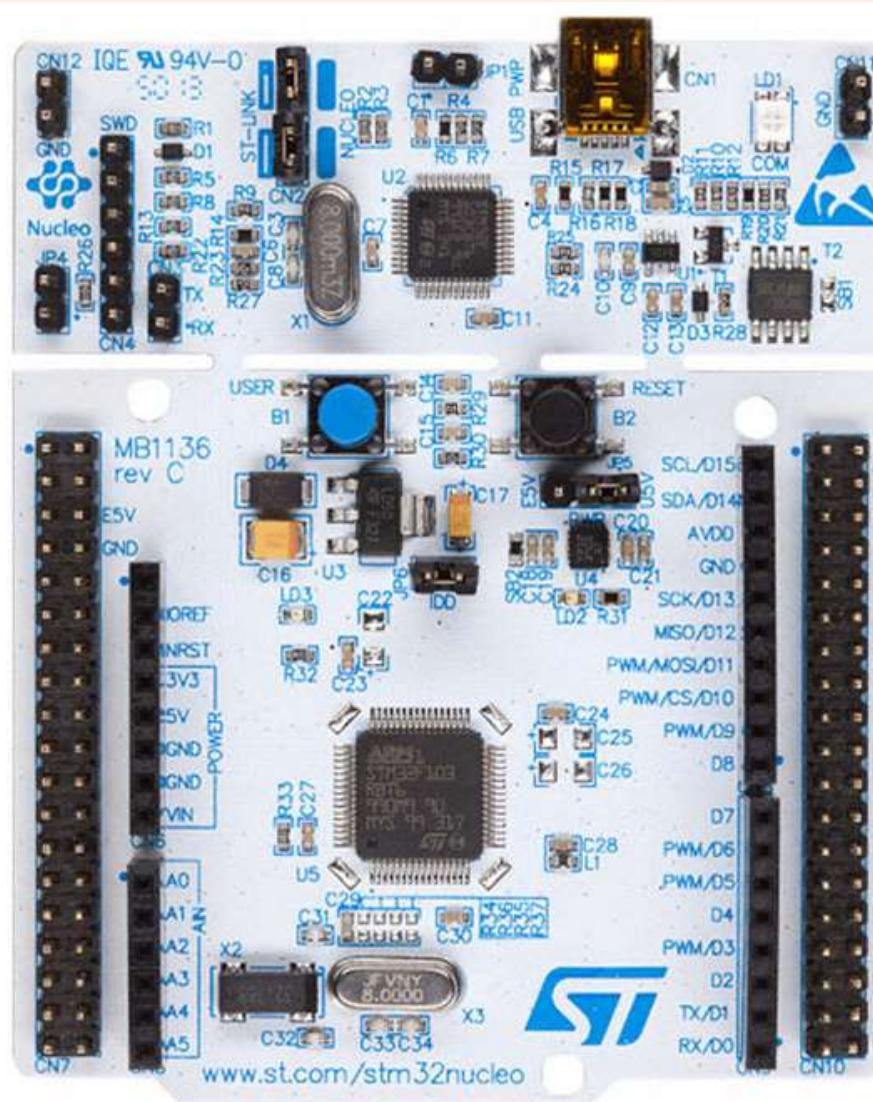
GPIO pin – General Purpose Input/Output pin



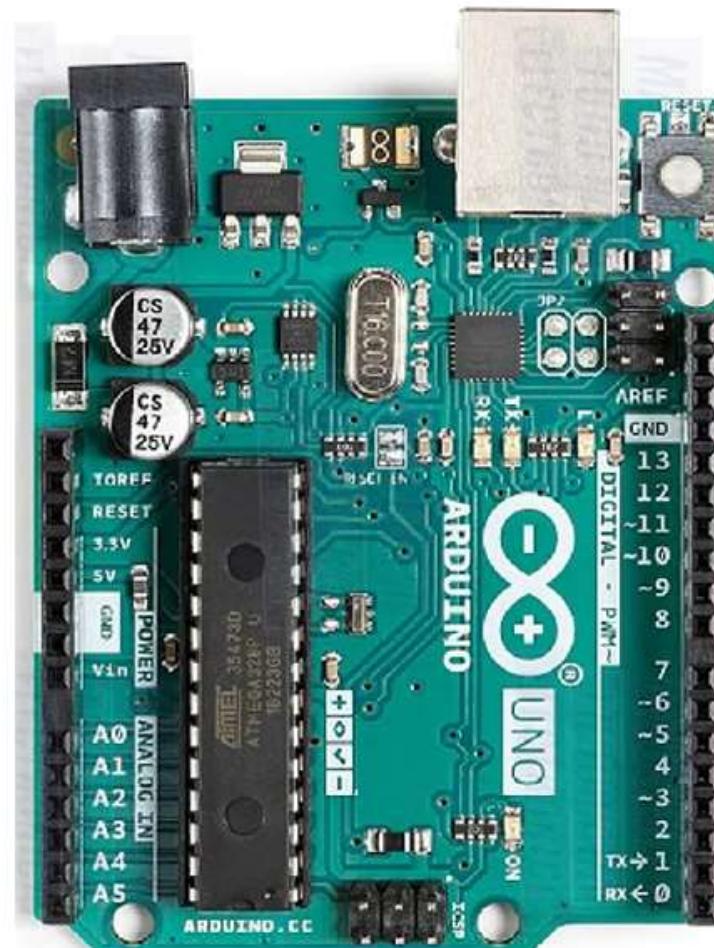
# STM NUCLEO64 – mapa de pinos



# Placas “Arduino UNO” e “STM NUCLEO64”



STM NUCLEO64



ARDUINO UNO

# Placas “Arduino UNO” e “STM NUCLEO64”

