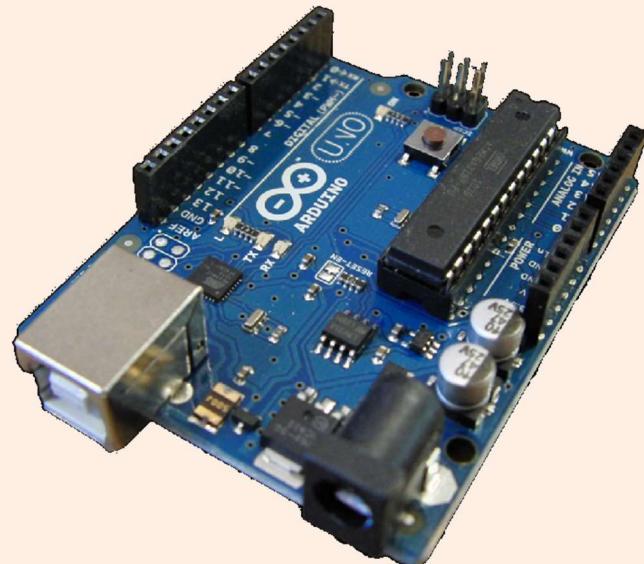


# Competências Transferíveis

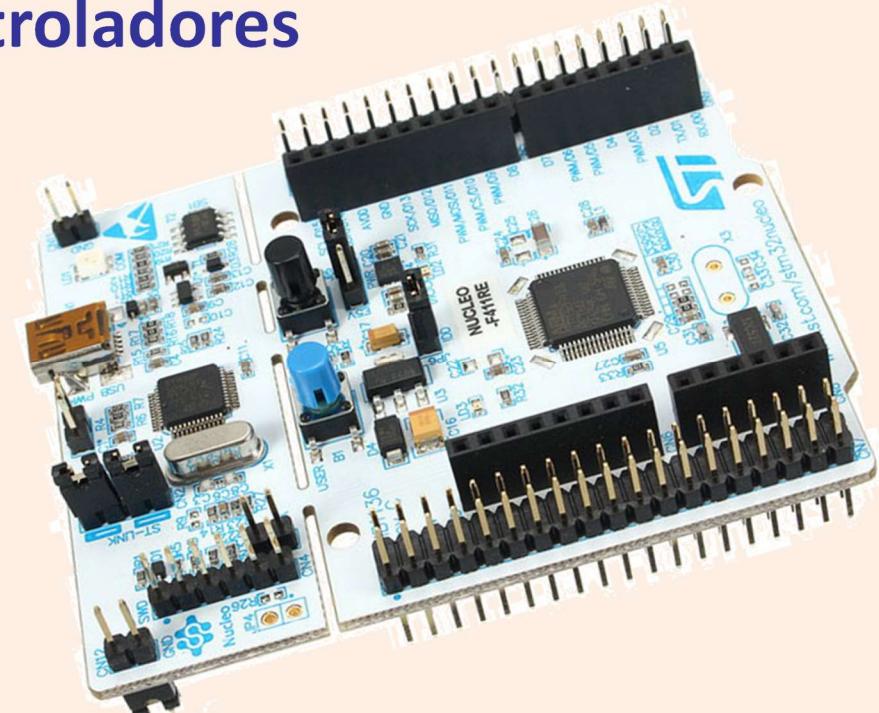
## Microcontroladores e Interação com Sensores e Atuadores

2021-2022

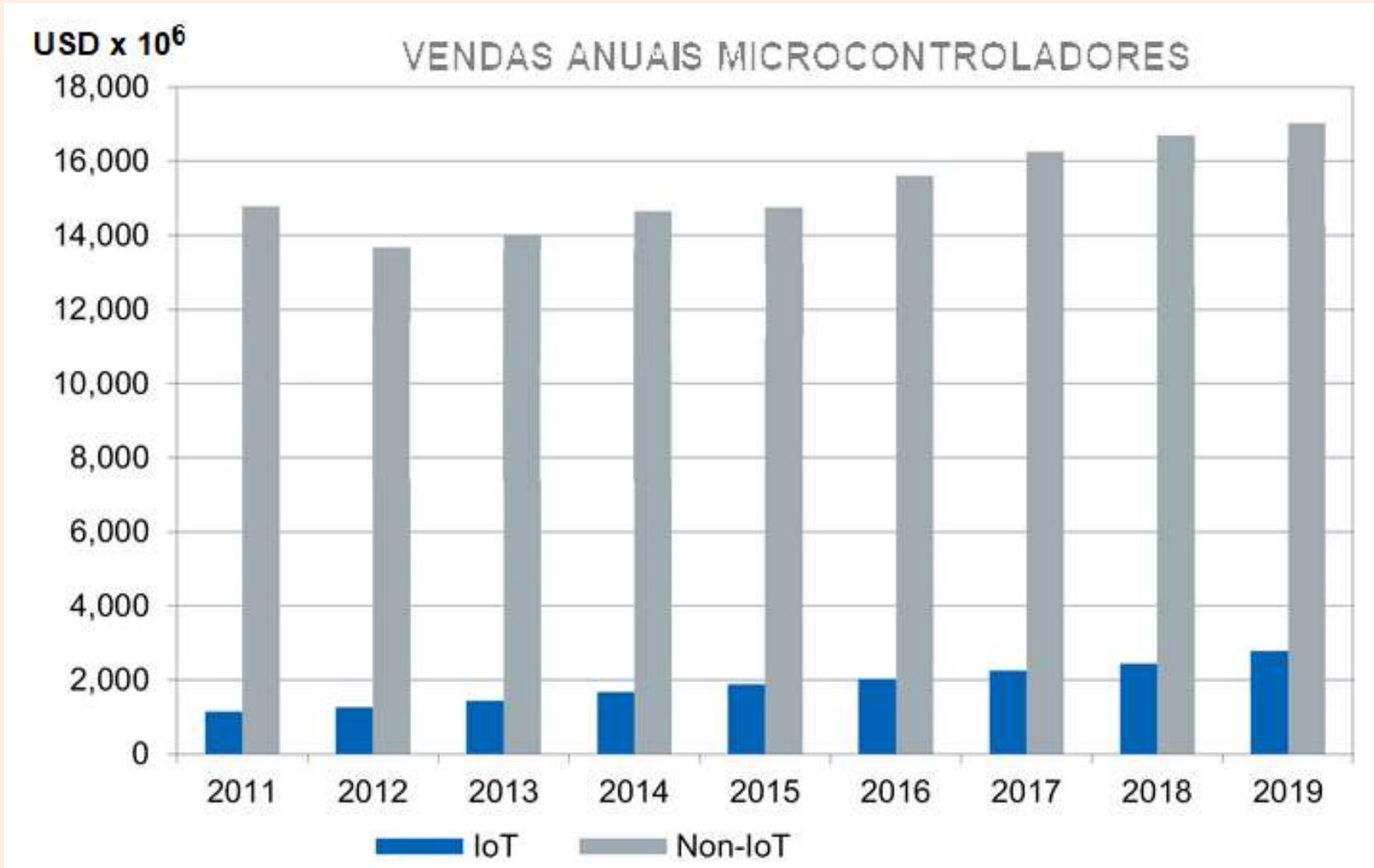
Rui Escadas Martins



### Introdução aos Microcontroladores



# Introdução aos Microcontroladores



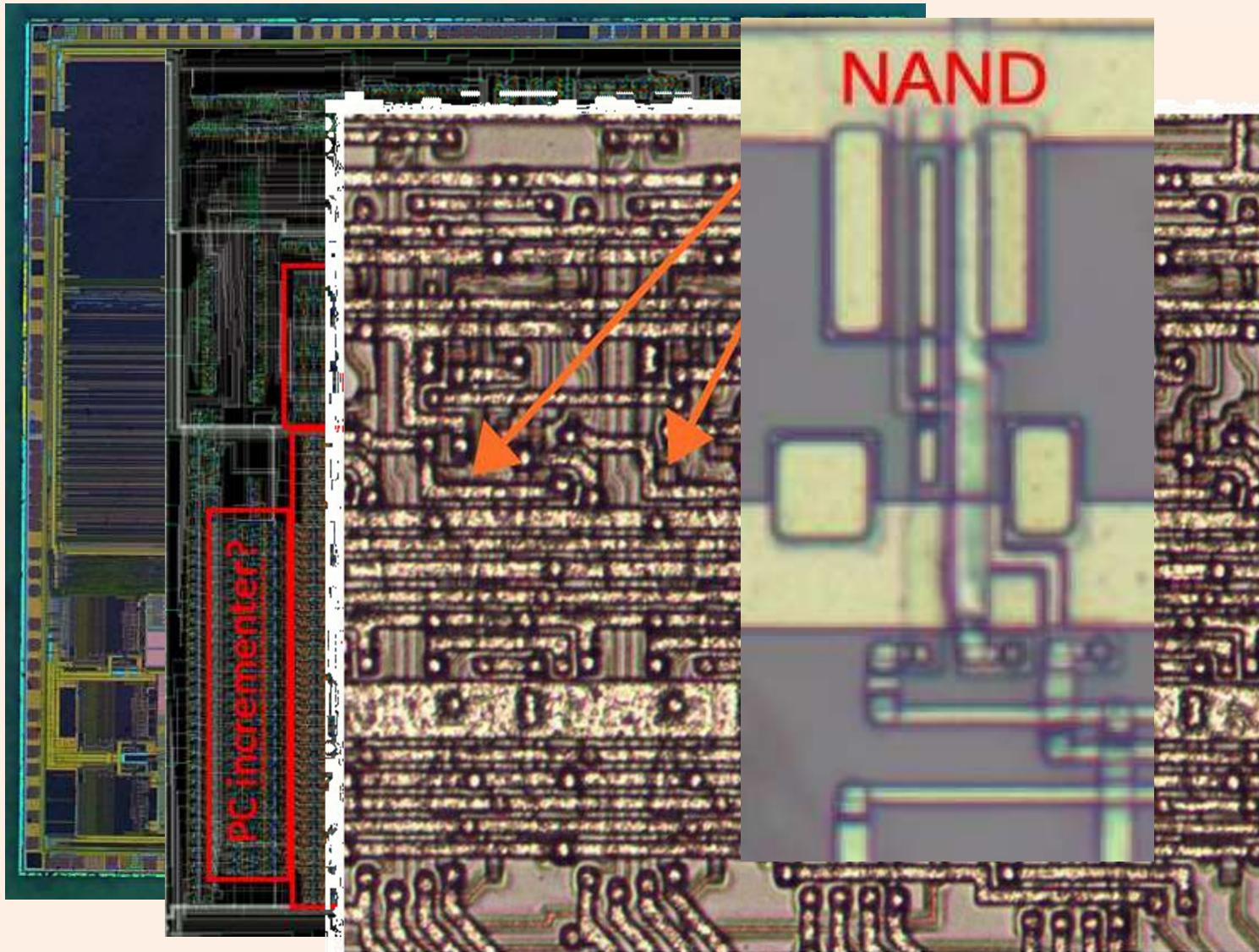
# Introdução aos Microcontroladores

---

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

- Mais memória -> Maiores programas
- Mais memória nem sempre implica maior desempenho.

# Como escolher um microcontrolador?

---

Características mais importantes:

## Tamanho memória RAM

- 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 digital para processamento por um computador digital.

# 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:

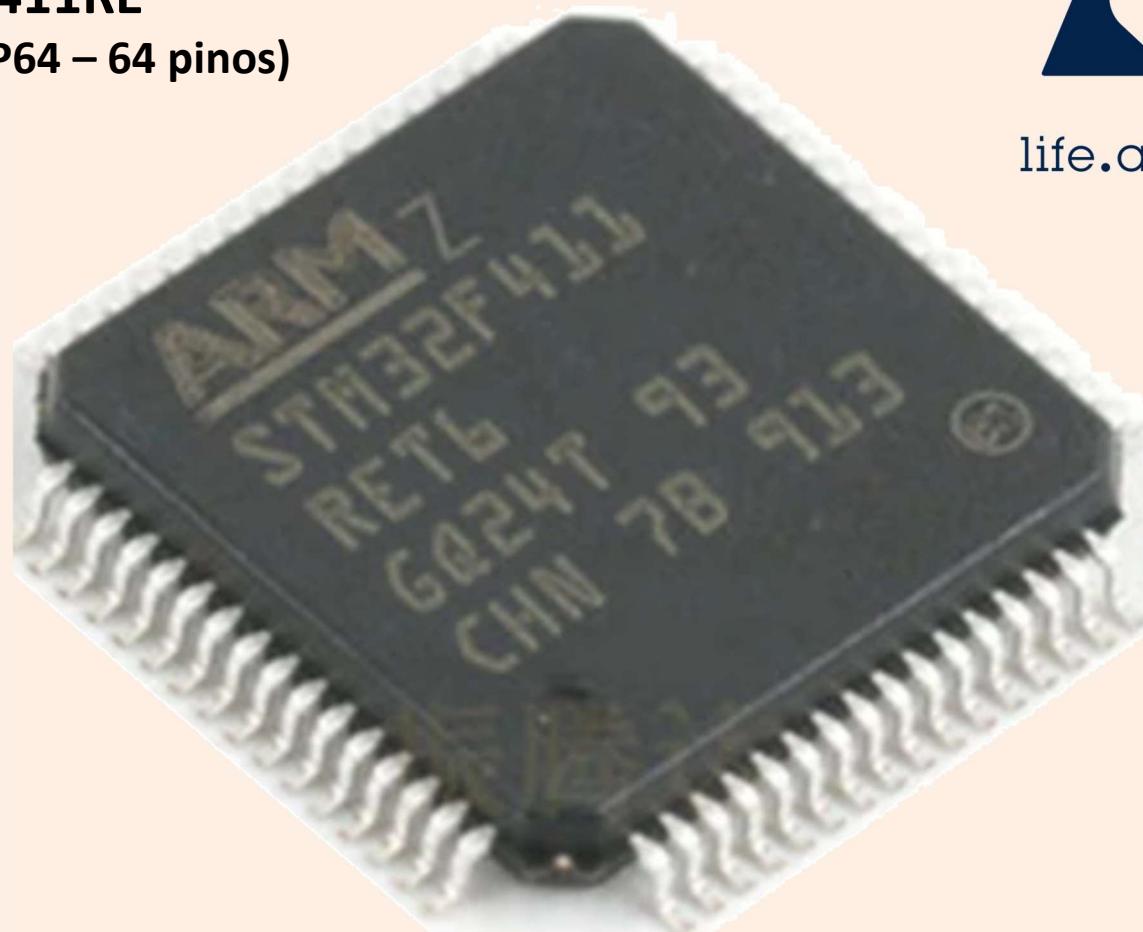
## Funções especiais

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

# Introdução aos Microcontroladores

Chip:

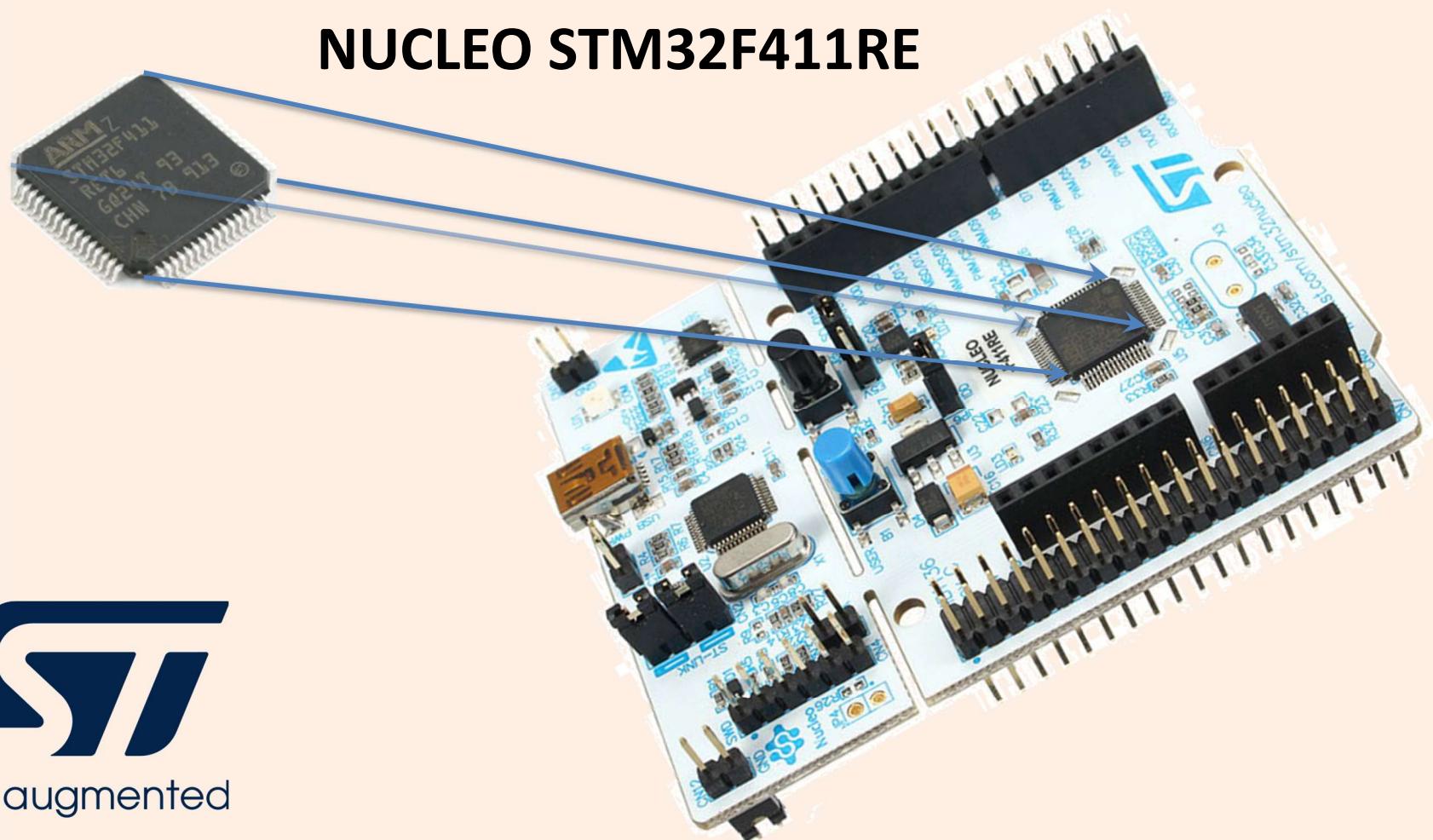
**STM32F411RE**  
**(LQFP64 – 64 pinos)**



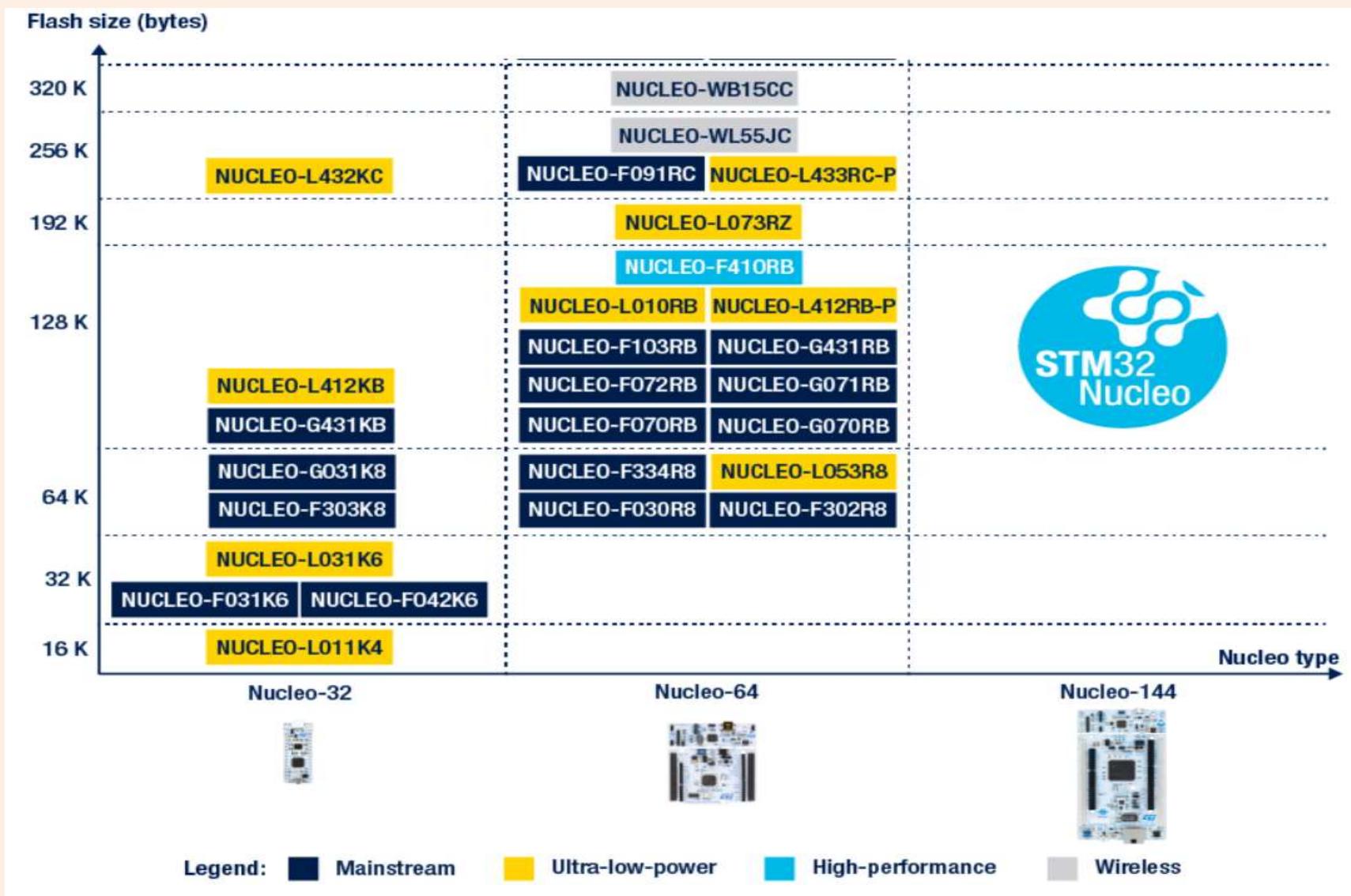
# Introdução aos Microcontroladores

Placa desenvolvimento:

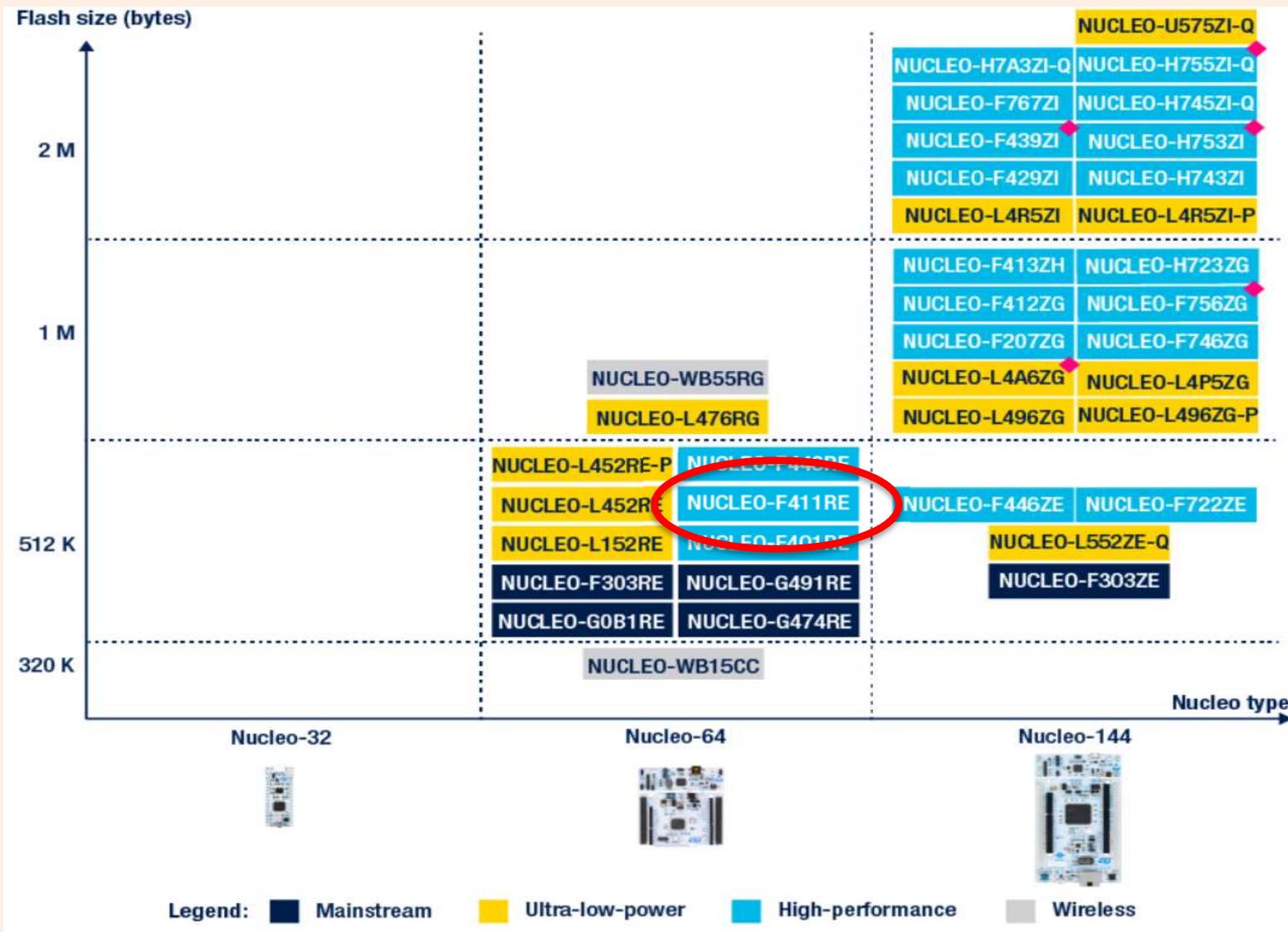
**NUCLEO STM32F411RE**



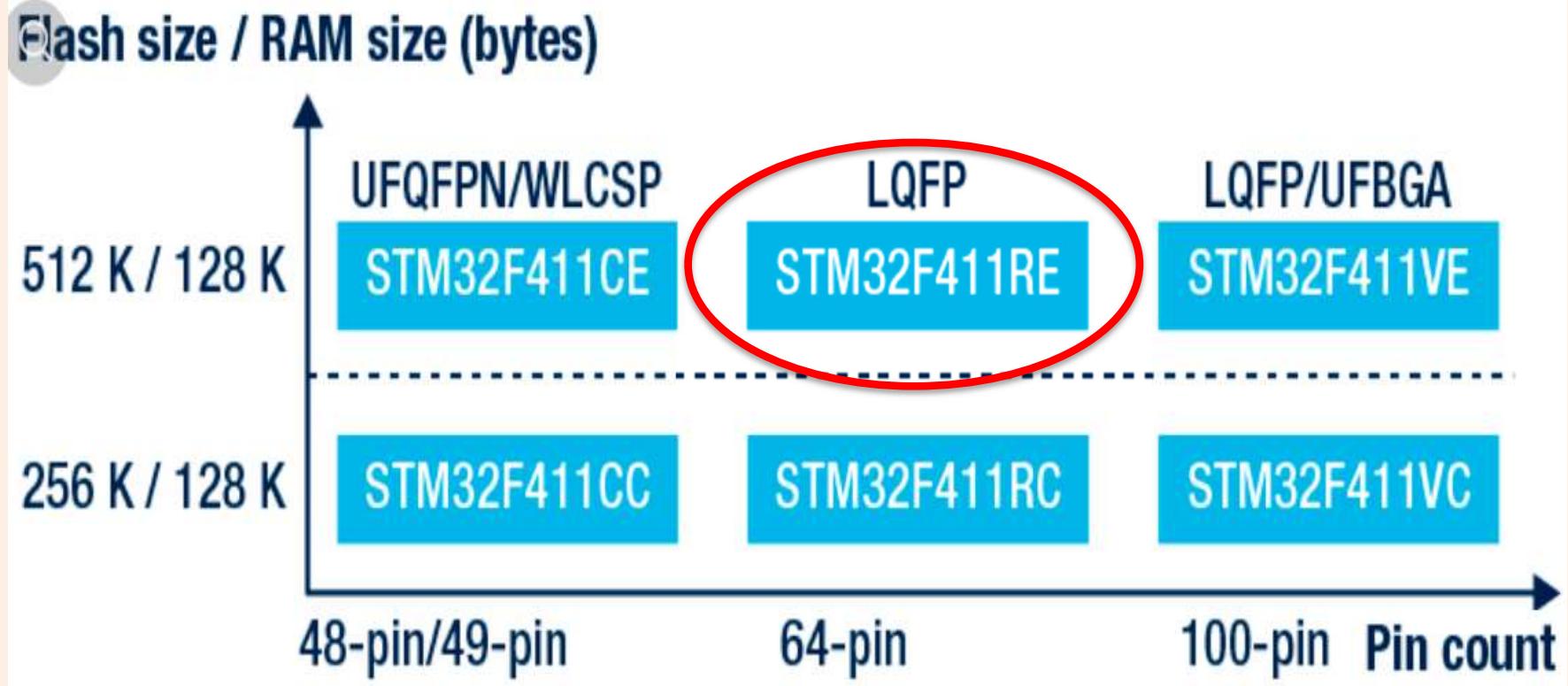
# Introdução aos Microcontroladores



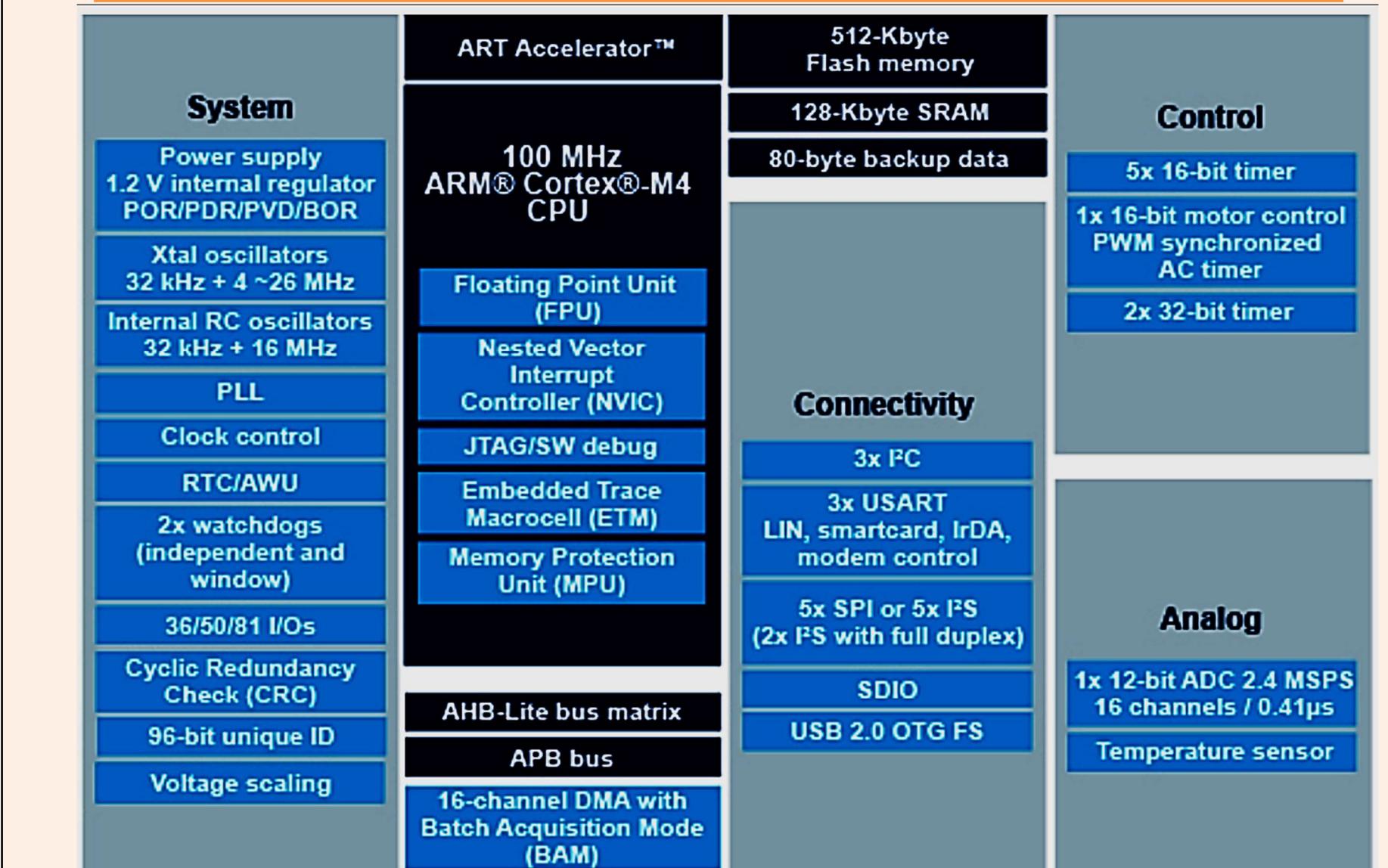
# Introdução aos Microcontroladores



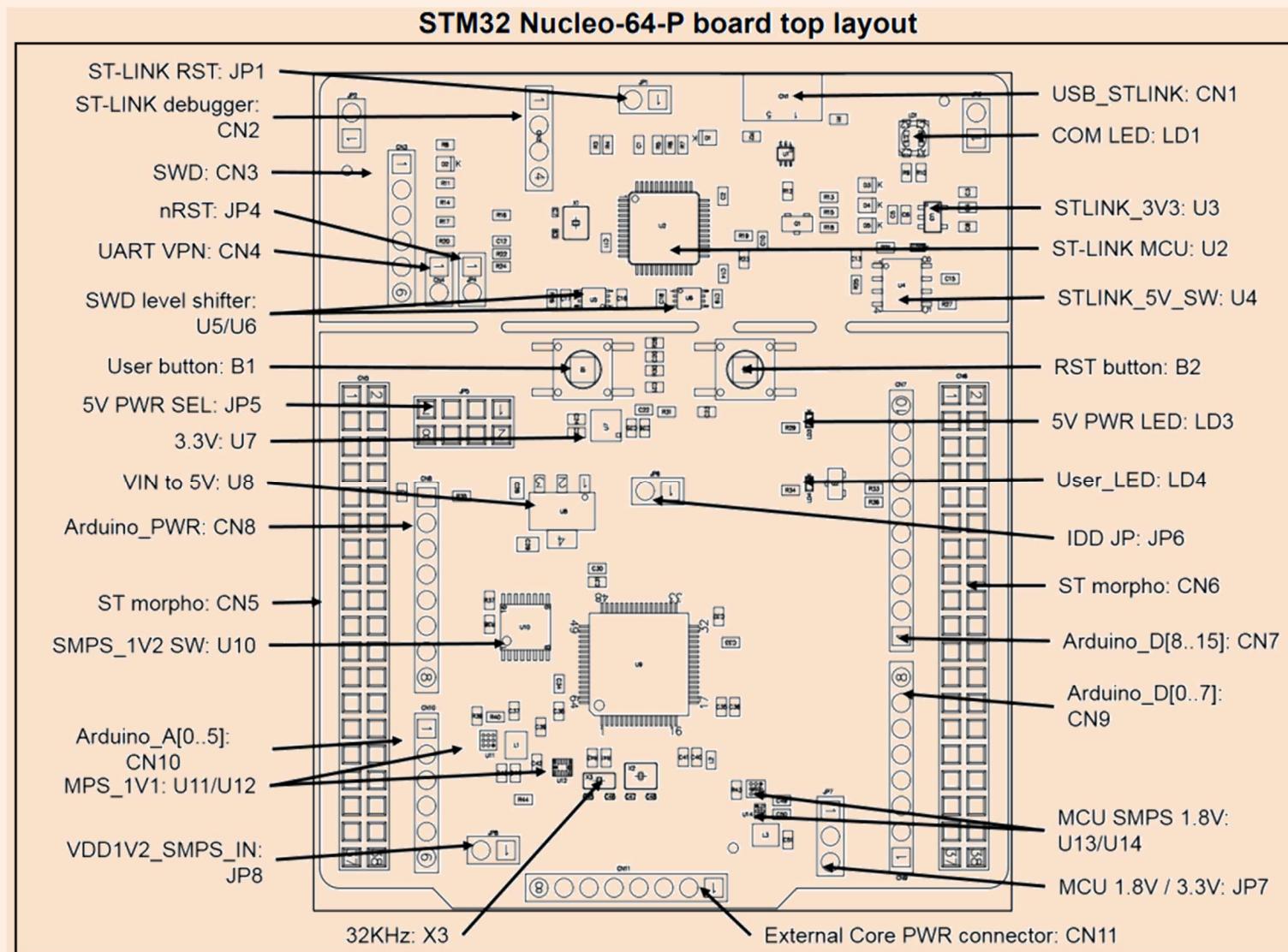
# Introdução aos Microcontroladores



# Introdução aos Microcontroladores



# Introdução aos Microcontroladores



# Get Started

---

Procedimento:

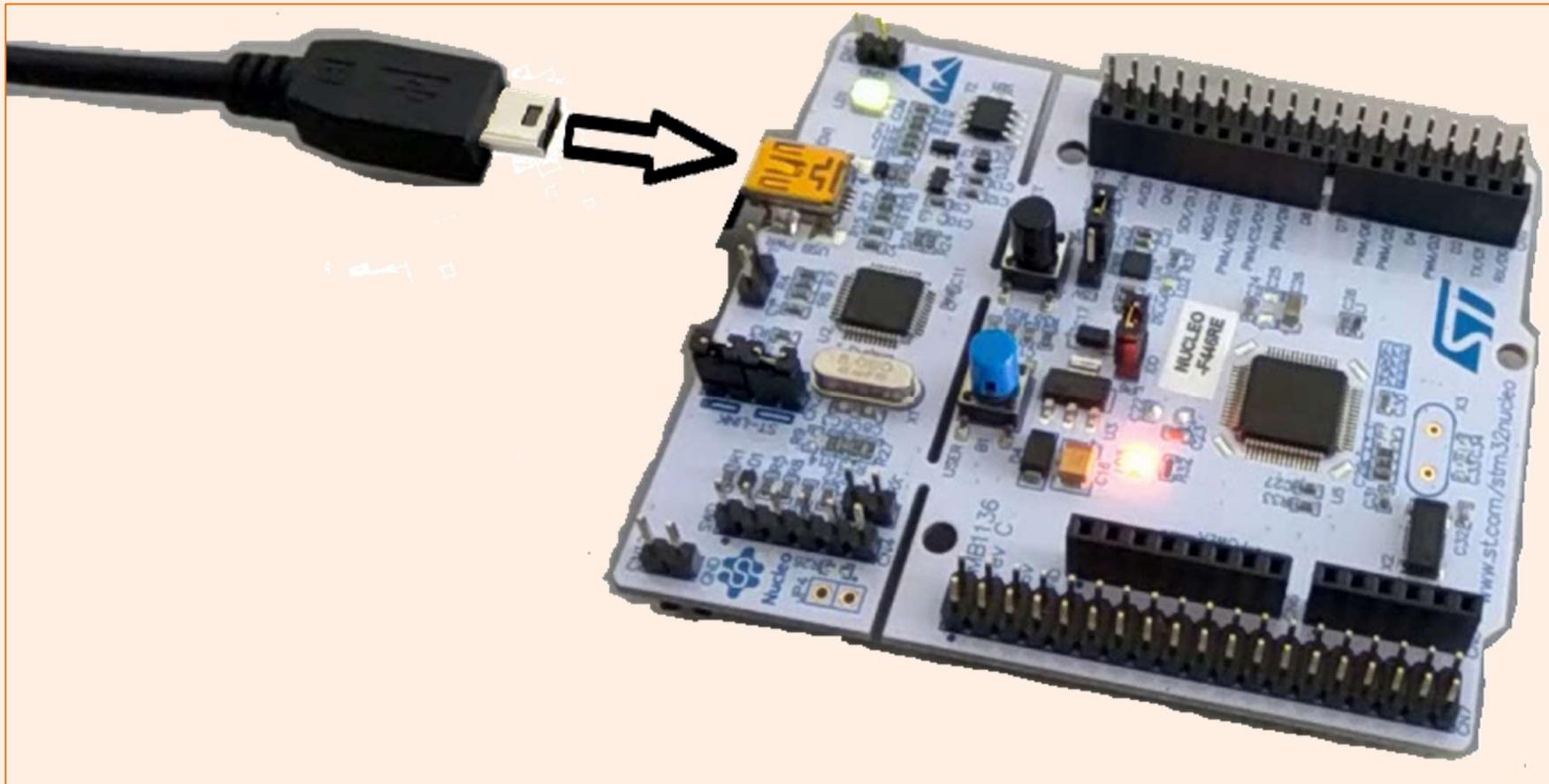
**É só ligar a placa a um PC através de um cabo USB – Mini-USB**



# Get Started

Procedimento:

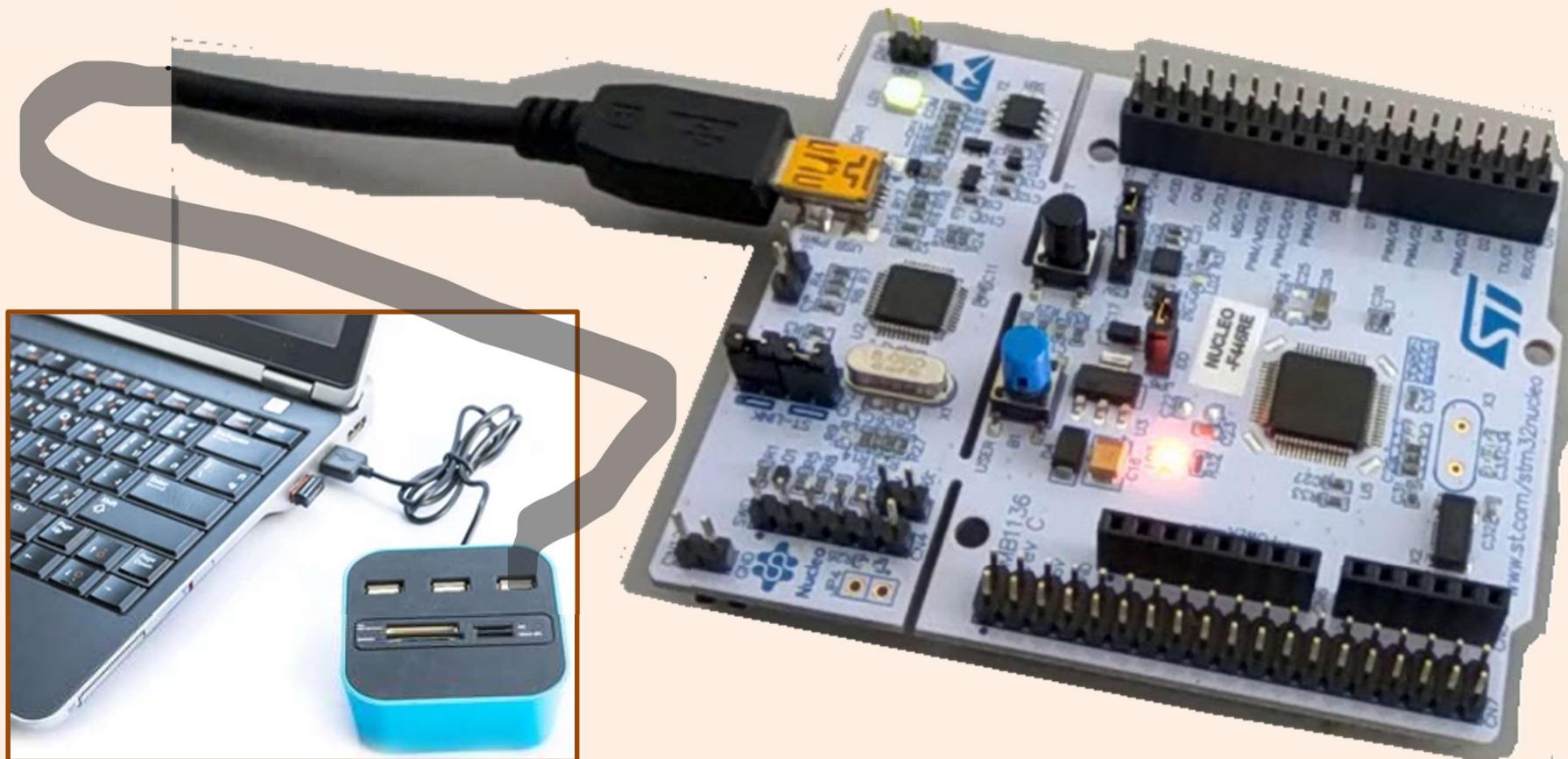
**É só ligar a placa a um PC através de um cabo USB – Mini-USB**



# Get Started

Procedimento:

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



# Get Started

---

Procedimento:

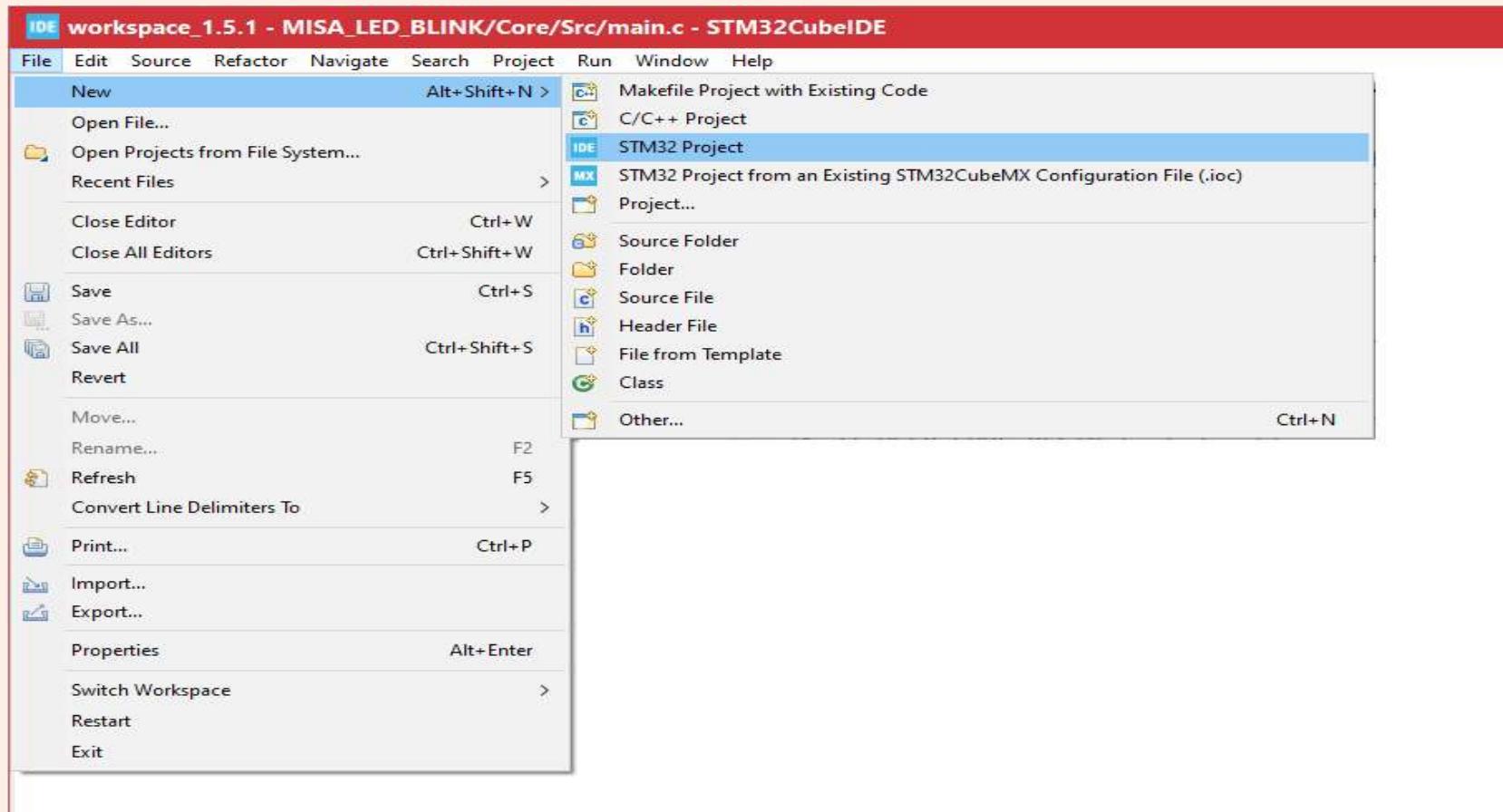
**Arrancar o IDE**



# Get Started

Procedimento:

## Criar um Projeto tipo “STM32”



# Get Started

Procedimento:

## Seleccionar a placa: NUCLEO-F411RE

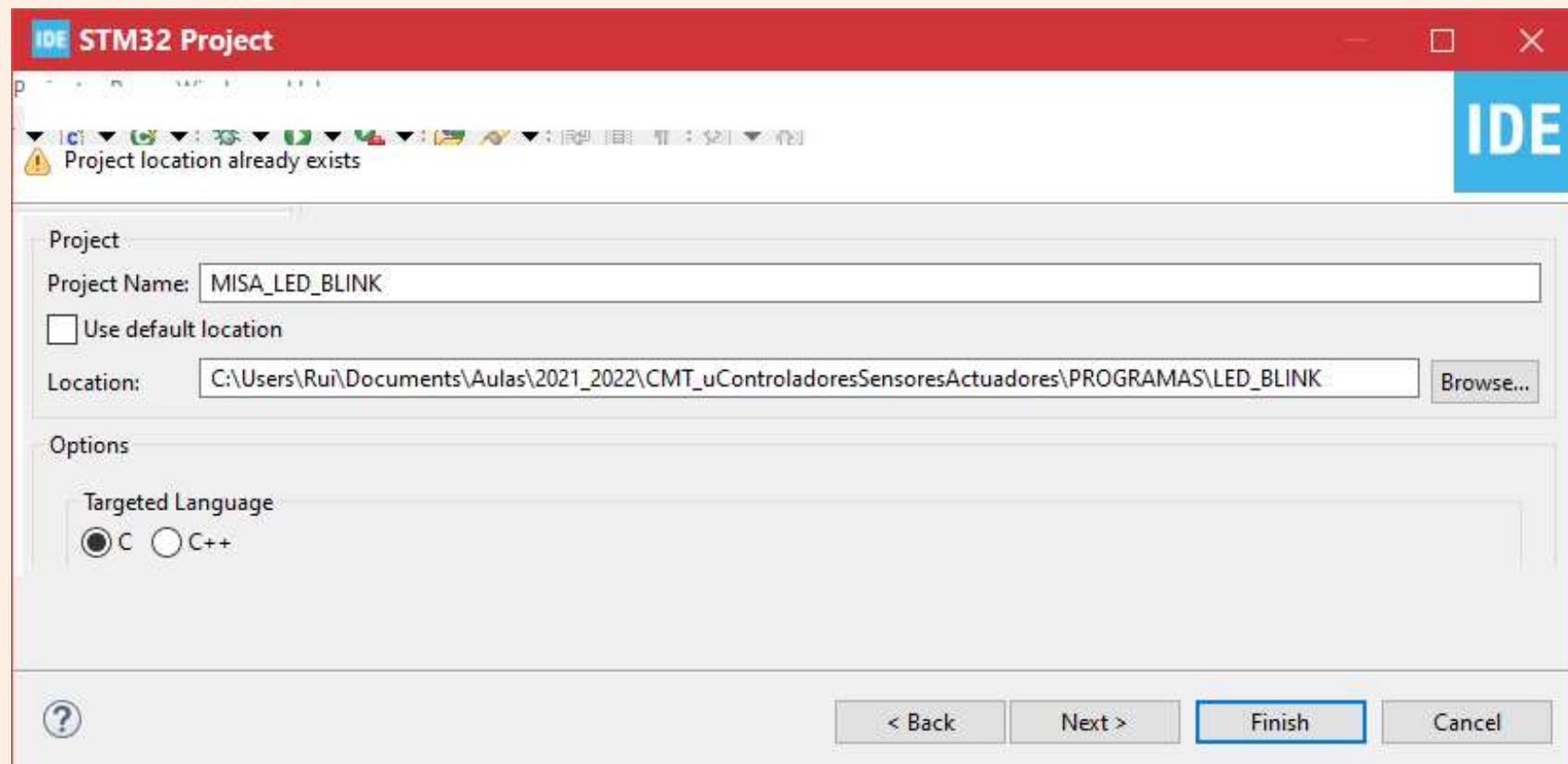
The screenshot shows the STMicroelectronics Board Selector interface. At the top, there are four tabs: MCU/MPU Selector, Board Selector (which is selected), Example Selector, and Cross Selector. Below the tabs is a 'Board Filters' section with dropdown menus for Commercial Part Number, Vendor, Type, MCU/MPU Series, Other, and Peripheral. To the right of the filters is a 'Features' button, a 'Large Picture' link, a 'Docs & Resources' link, a 'Datasheet' link, and a 'Buy' link. A star icon indicates the 'STM32F4 Series'. The main area displays a list titled 'STM32F4 Series' with 'NUCLEO-F411RE' highlighted. Below this is a sub-section titled 'STMicroelectronics NUCLEO-F411RE Board Support and Examples'. The 'Boards List: 158 items' table has columns for Overview, Commercial Part ..., Type, Marketing Status, Unit Price (US\$), and Mounted Device. The NUCLEO-F411RE row is selected and highlighted in blue. The table includes links for each board: STM32F334R8Tx, STM32F401RETx, STM32F410RBTx, STM32F411RETx, STM32F412ZGTx, and STM32F413ZHtx. At the bottom of the interface are 'Next >', 'Finish', and 'Cancel' buttons.

	Overview	Commercial Part ...	Type	Marketing Status	Unit Price (US\$)	Mounted Device
★		NUCLEO-F334R8	Nucleo-64	Active	10.32	<a href="#">STM32F334R8Tx</a>
★		NUCLEO-F401RE	Nucleo-64	Active	13.0	<a href="#">STM32F401RETx</a>
★		NUCLEO-F410RB	Nucleo-64	Active	13.0	<a href="#">STM32F410RBTx</a>
★		NUCLEO-F411RE	Nucleo-64	Active	13.0	<a href="#">STM32F411RETx</a>
★		NUCLEO-F412ZG	Nucleo-144	Active	19.0	<a href="#">STM32F412ZGTx</a>
★		NUCLEO-F413ZH	Nucleo-144	Active	19.0	<a href="#">STM32F413ZHtx</a>

# Get Started

Procedimento:

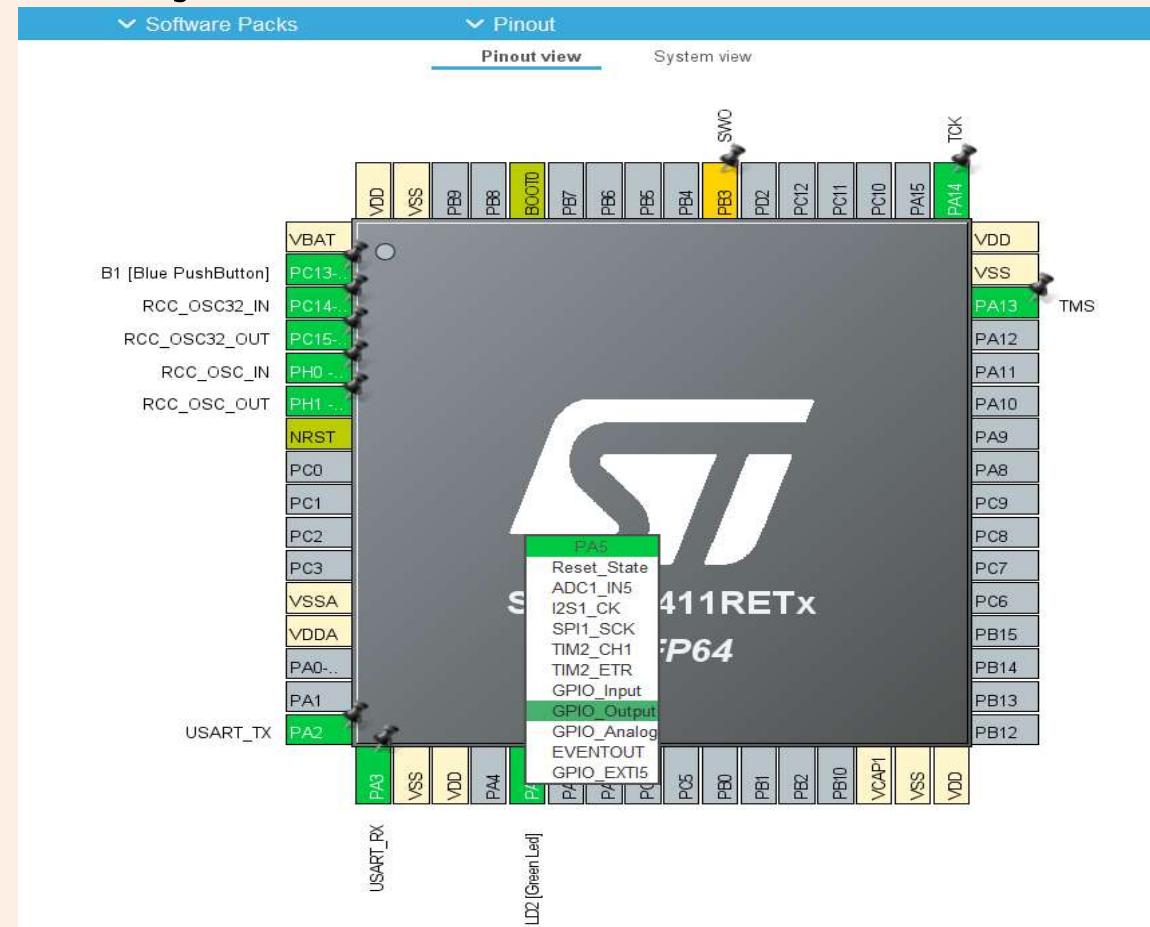
## Selecionar nome e local destino



# Get Started

Procedimento:

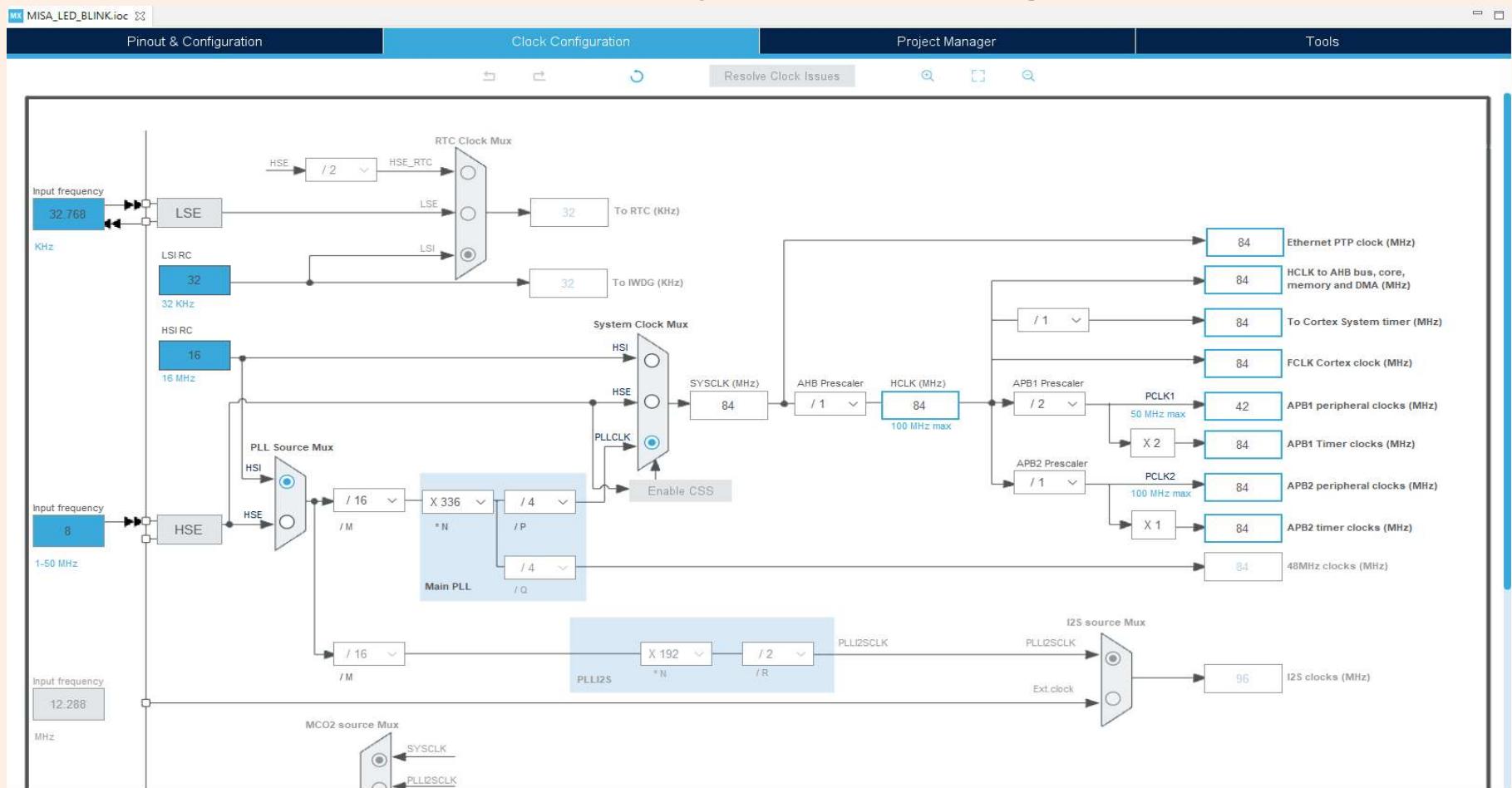
## Selecionar pinos e funções



# Get Started

Procedimento:

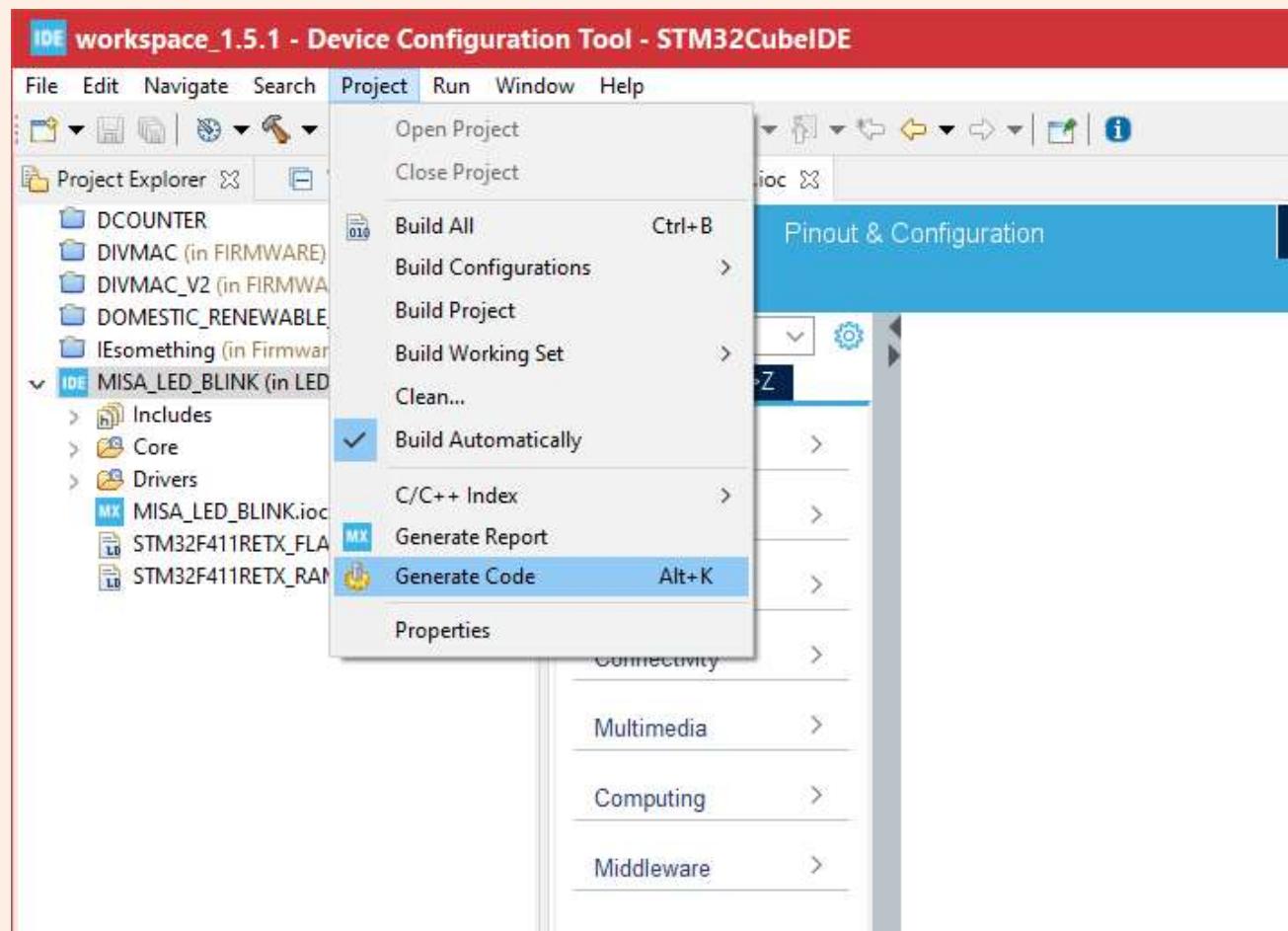
## Selecionar oscilador e frequência de relógio



# Get Started

Procedimento:

## Gerar Código (automaticamente)



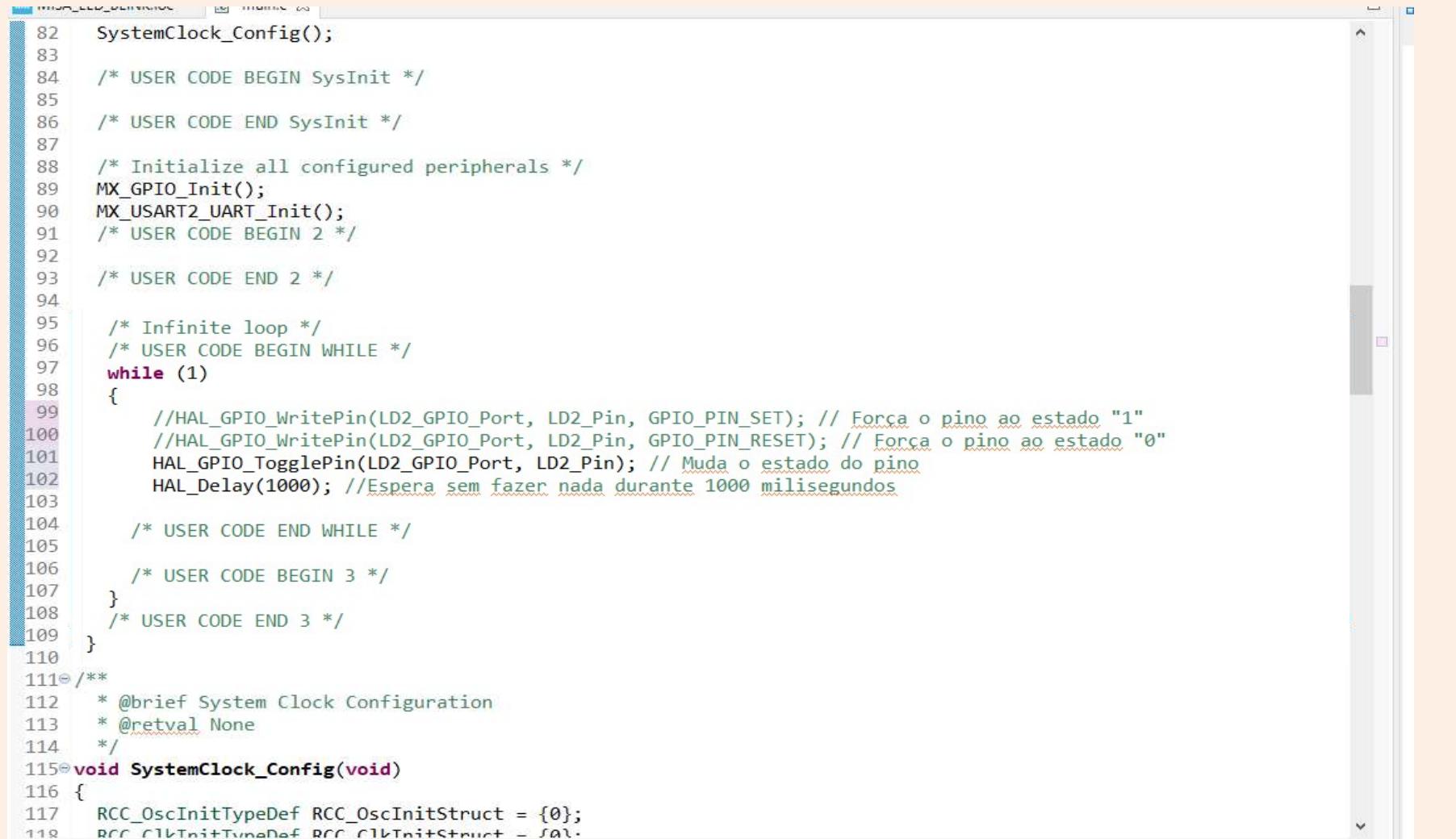
# STM32CubeIDE

The screenshot shows the STM32CubeIDE interface with the following details:

- Title Bar:** IDE workspace\_1.5.1 - MISA\_LED\_BLINK/Core/Src/main.c - STM32CubeIDE
- File Menu:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbars:** Standard toolbar with icons for file operations, search, and project navigation.
- Project Explorer:** Shows C/C++ Projects with items like DCOUNTER, DIVMAC, DIVMAC\_V2, DOMESTIC\_RENEWABLE\_SYSTEM, Esomething, and MISA\_LED\_BLINK.
- Editor:** Displays the main.c file content. The code includes a BSD 3-Clause license header, includes for main.h and system headers, and sections for private includes, type definitions, and defines.
- Outline View:** Shows the structure of the main.c file, including function prototypes for main(), SystemClock\_Config(), MX\_GPIO\_Init(), MX\_USART2\_UART\_Init(), and Error\_Handler(). It also lists assert\_failed() as a missing symbol.
- Build Targets:** Shows build targets for main.h and various USART-related functions.
- Bottom Status Bar:** Shows tabs for Problems, Tasks, Console, Properties, and Build Analyzer. The Build Analyzer tab indicates "No search results available. Start a search from the [search dialog...](#)".
- Bottom Navigation Bar:** Includes Writable, Smart Insert, and a timestamp of 1:1:0.

# STM32CubeIDE

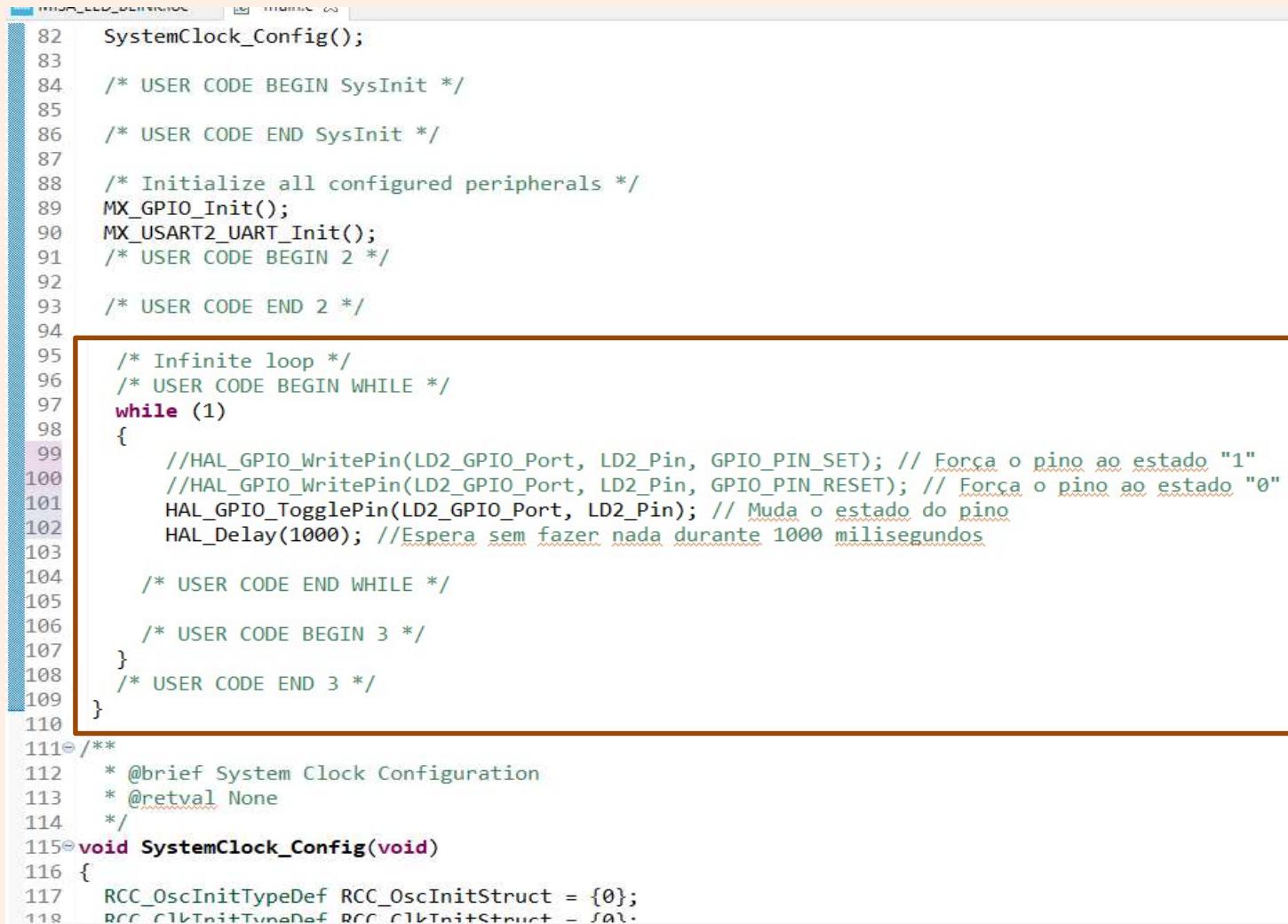
E agora é só escrever o Código...



```
82 SystemClock_Config();
83 /* USER CODE BEGIN SysInit */
84
85 /* USER CODE END SysInit */
86
87 /* Initialize all configured peripherals */
88 MX_GPIO_Init();
89 MX_USART2_UART_Init();
90 /* USER CODE BEGIN 2 */
91
92 /* USER CODE END 2 */
93
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 void SystemClock_Config(void)
115 {
116     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
117     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
```

# STM32CubeIDE

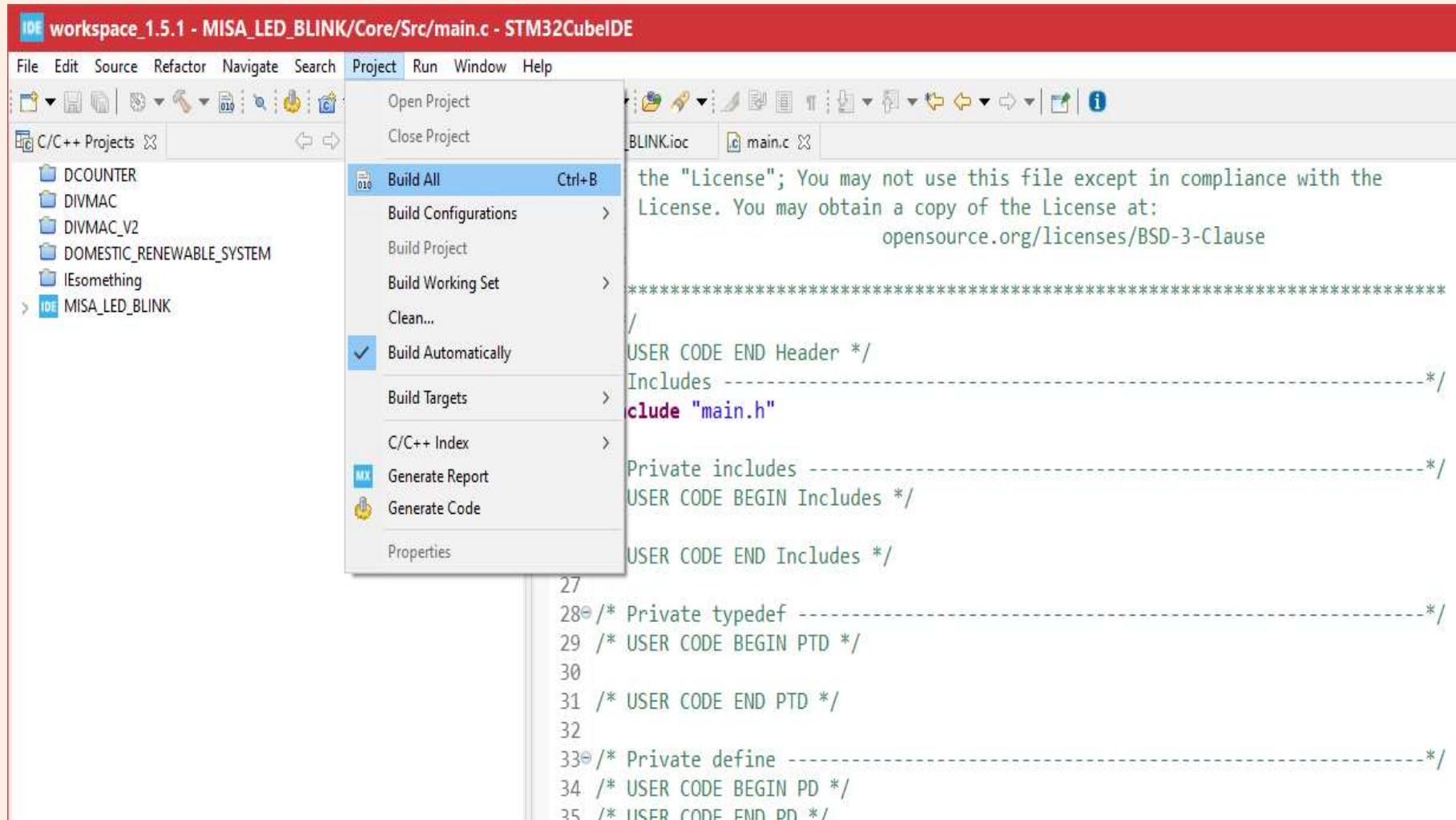
E agora é só escrever o Código...



```
82 SystemClock_Config();
83 /* USER CODE BEGIN SysInit */
84
85 /* USER CODE END SysInit */
86
87 /* Initialize all configured peripherals */
88 MX_GPIO_Init();
89 MX_USART2_UART_Init();
90 /* USER CODE BEGIN 2 */
91
92 /* USER CODE END 2 */
93
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 /**
112 * @brief System Clock Configuration
113 * @retval None
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



# STM32CubeIDE

Para enviar o executável para o microcontrolador fazer “Debug”.

