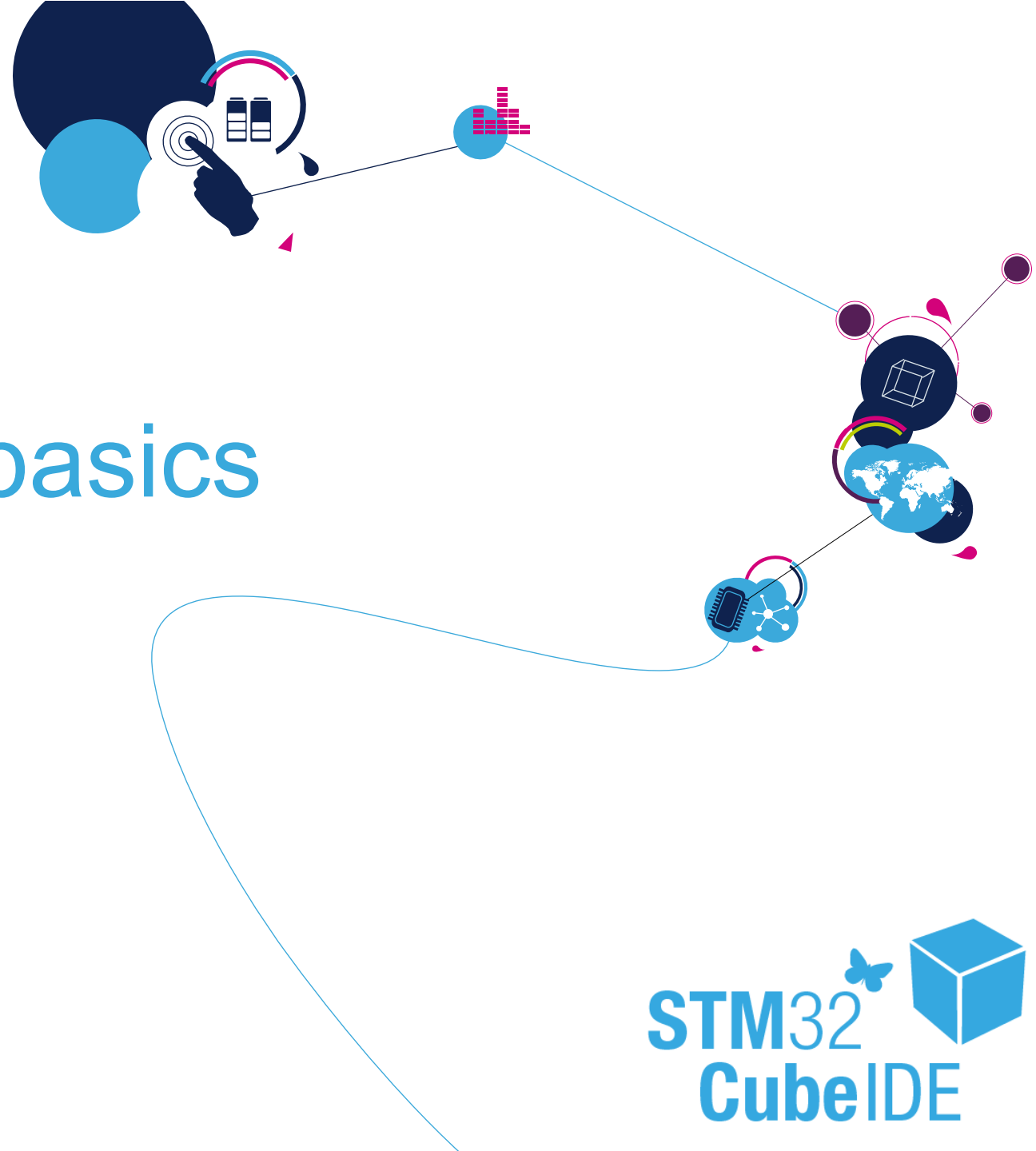
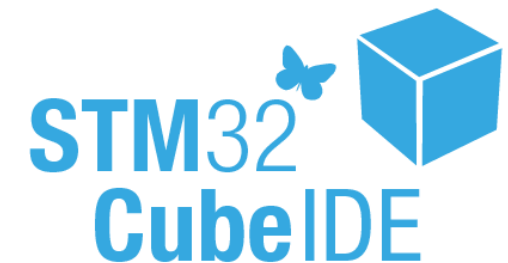


STM32CubeIDE basics

GPIO lab: LED toggling using HAL libraries



Lab: LED toggling using HAL library

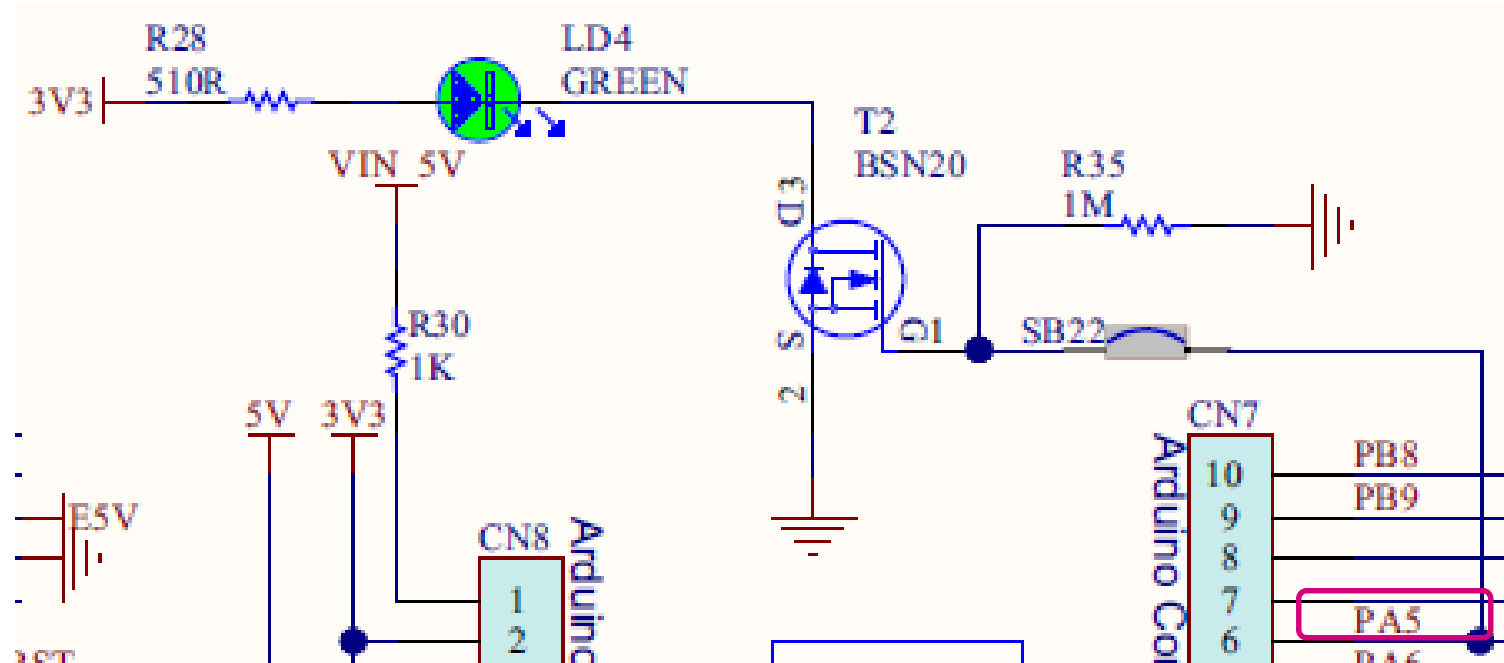
Objective:

- In this project we are going to toggle GREEN LED connected to PA5 pin on NUCLEO-G071RB board using Hardware Abstraction Layer (HAL) library

Pin Configuration

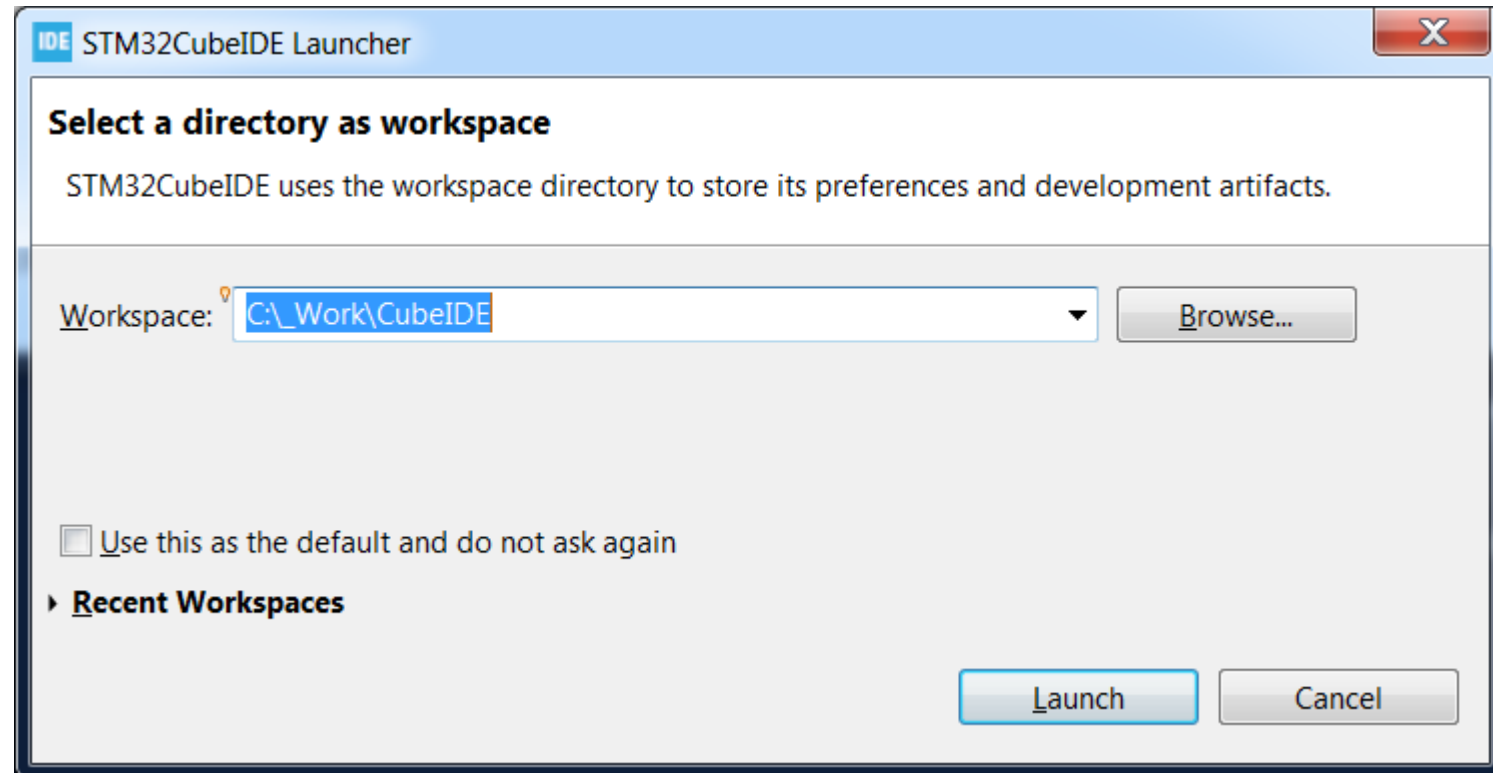
Green LED

- In this example we are going to use one of the LEDs present on the STM32G0 Nucleo board (connected to PA5 as seen in the schematic below)

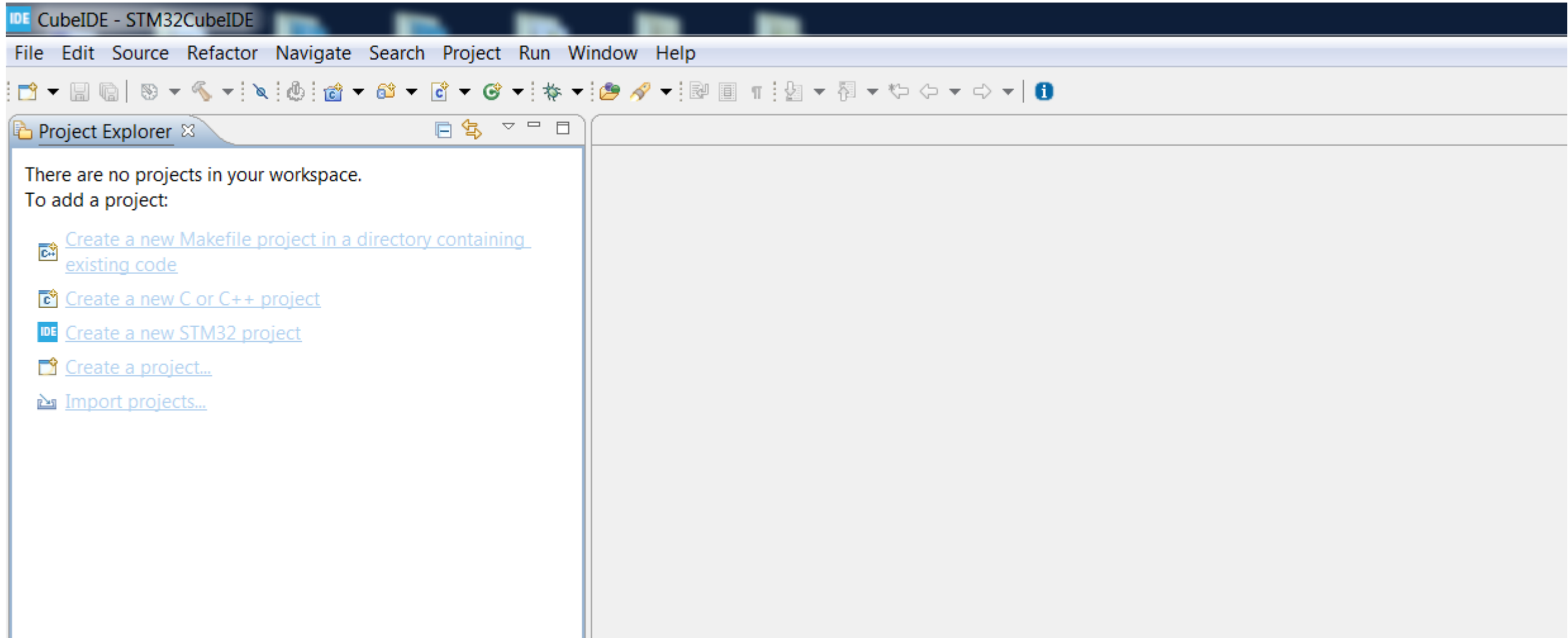


Start a new workspace

- Run STM32CubeIDE
- Select a folder to store a workspace

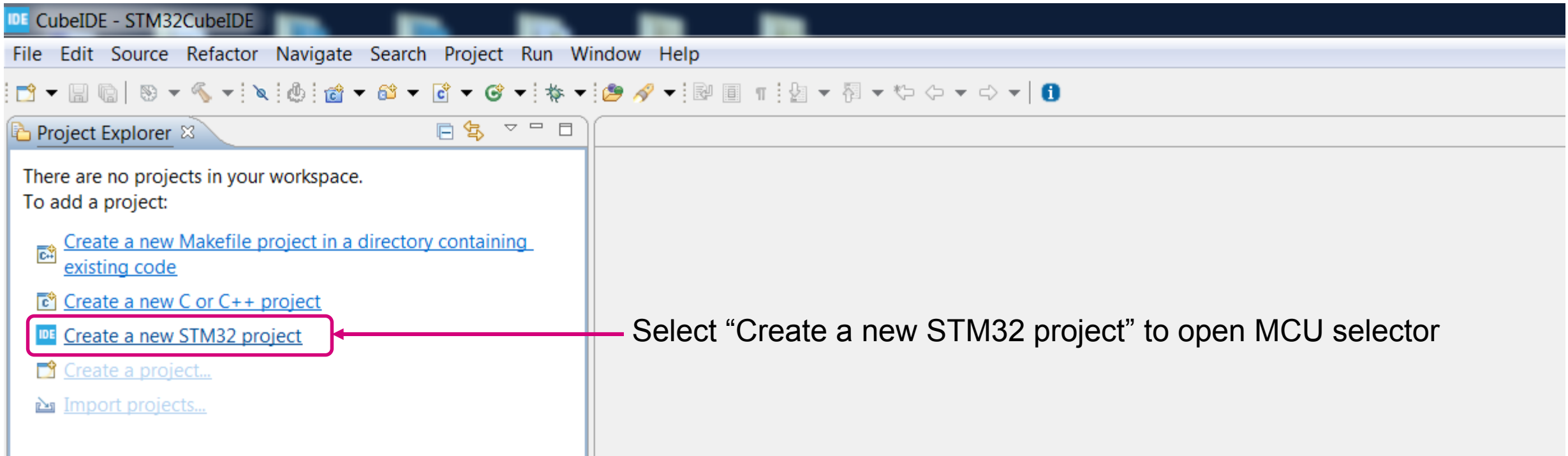


View on an empty workspace



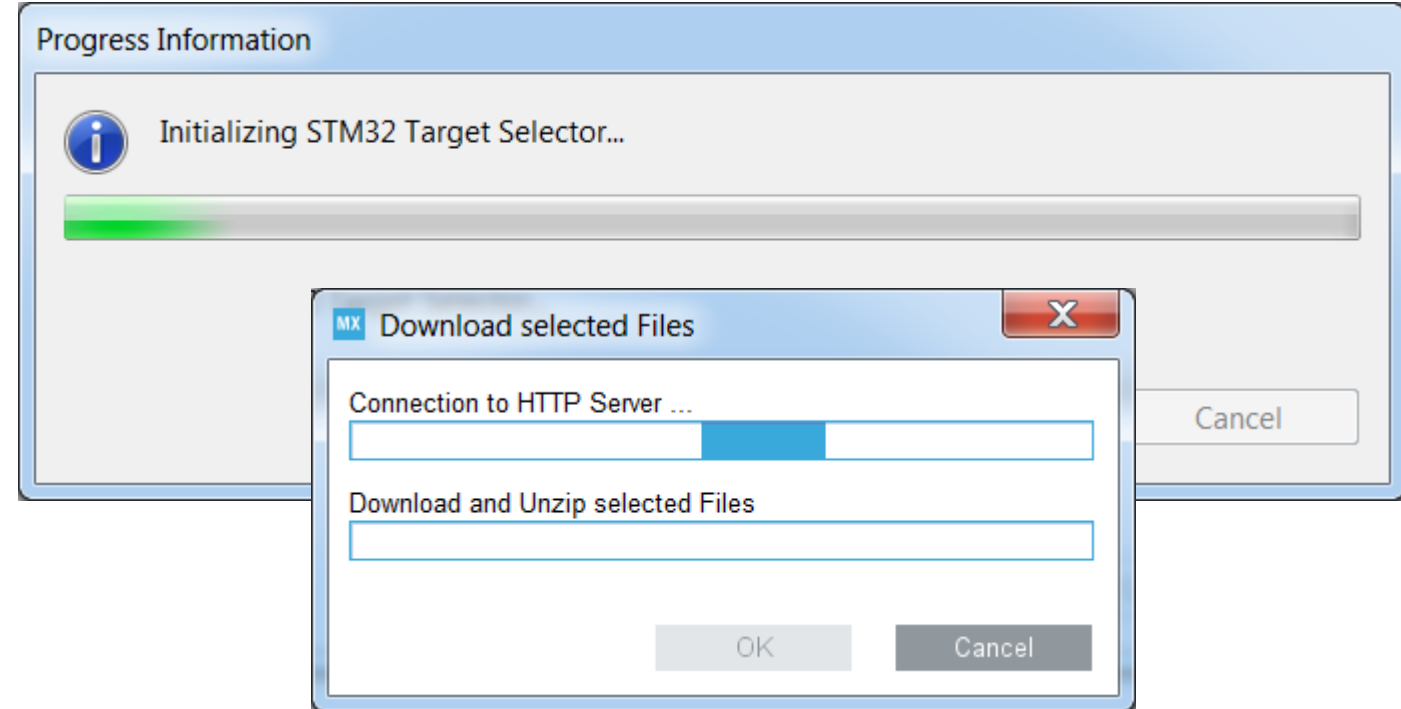
Create a new project

- Click on “Create a new STM32 project”

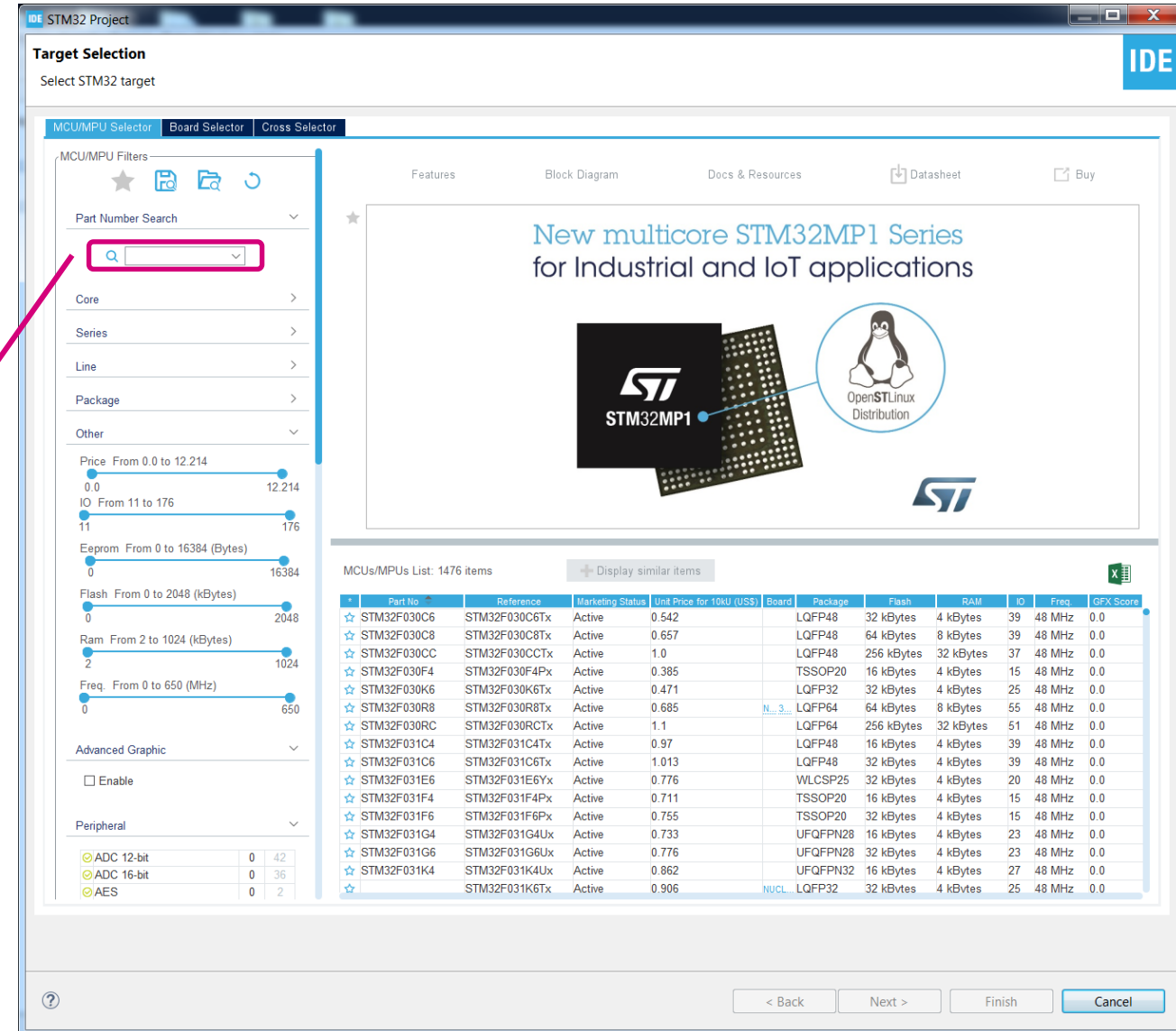
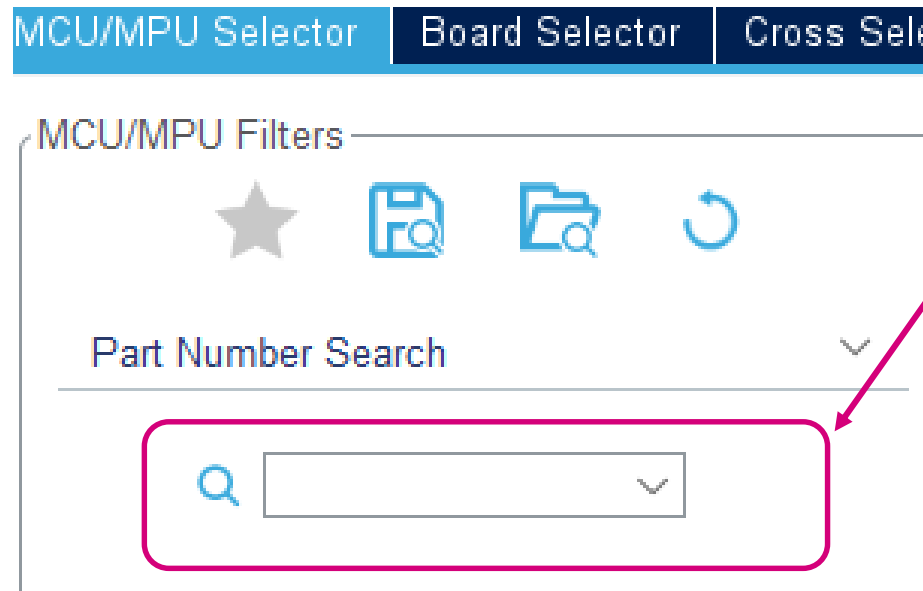


Checking for libraries update

- Application will look for updates on the server
- It may take a while ...

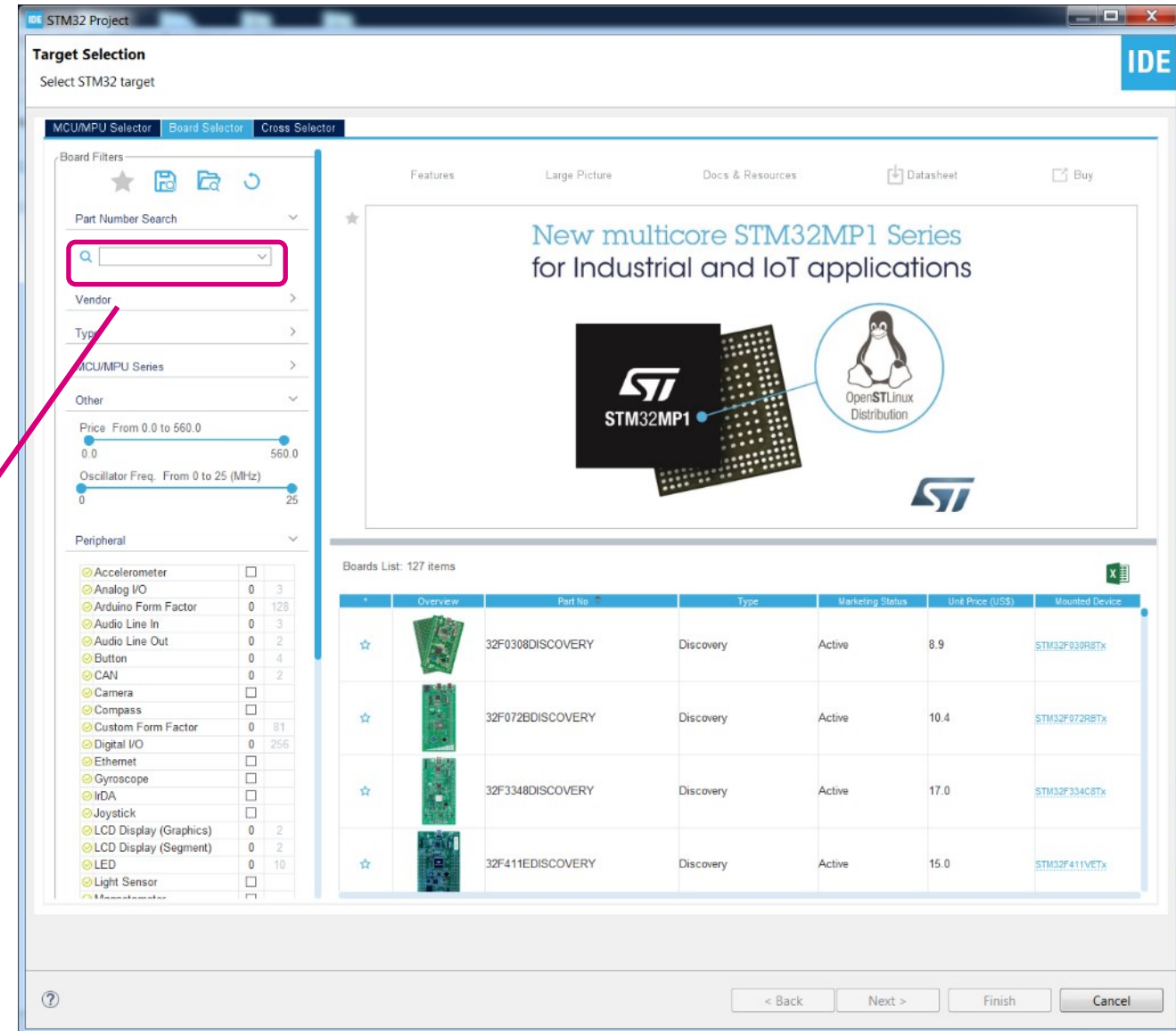
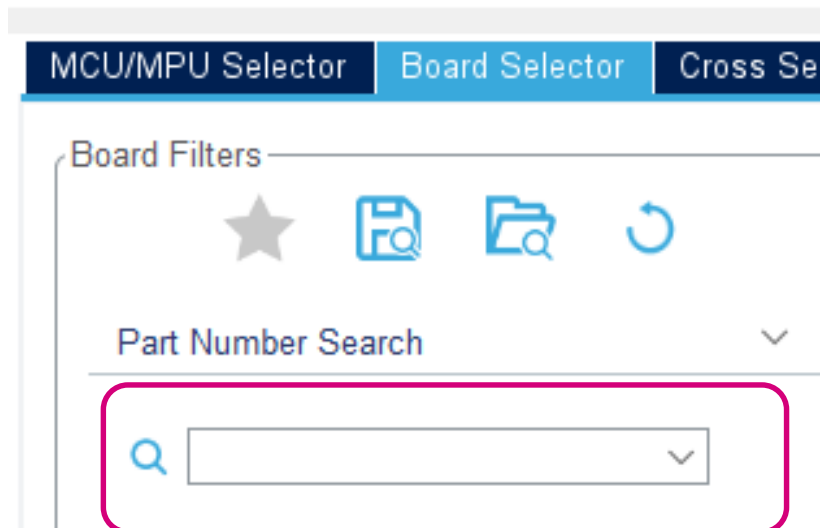


- As a next step we should select target MCU



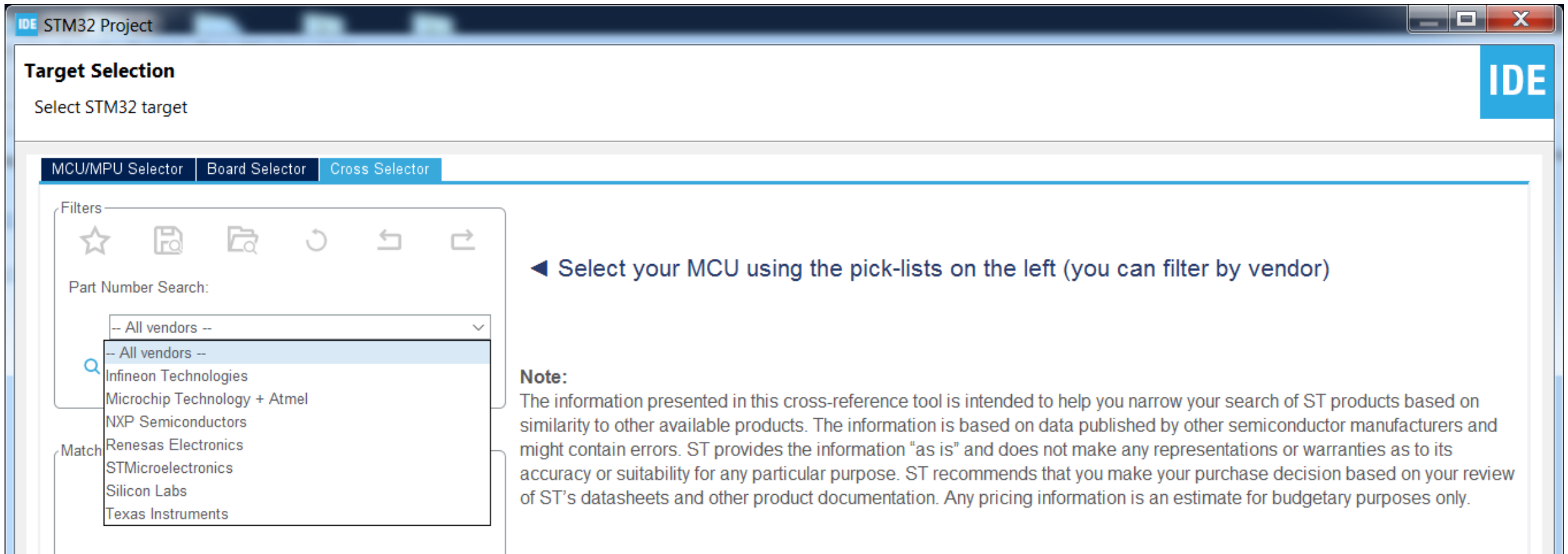
ST MCU board selector

- We can select an ST board instead



Cross reference selector

- It is possible to use a cross reference to select an STM32 device



Target Selection
Select STM32 target

MCU/MPU Selector | Board Selector | **Cross Selector**

Filters

Part Number Search:

-- All vendors --

-- All vendors --

Infineon Technologies

Microchip Technology + Atmel

NXP Semiconductors

Renesas Electronics

STMicroelectronics

Silicon Labs

Texas Instruments

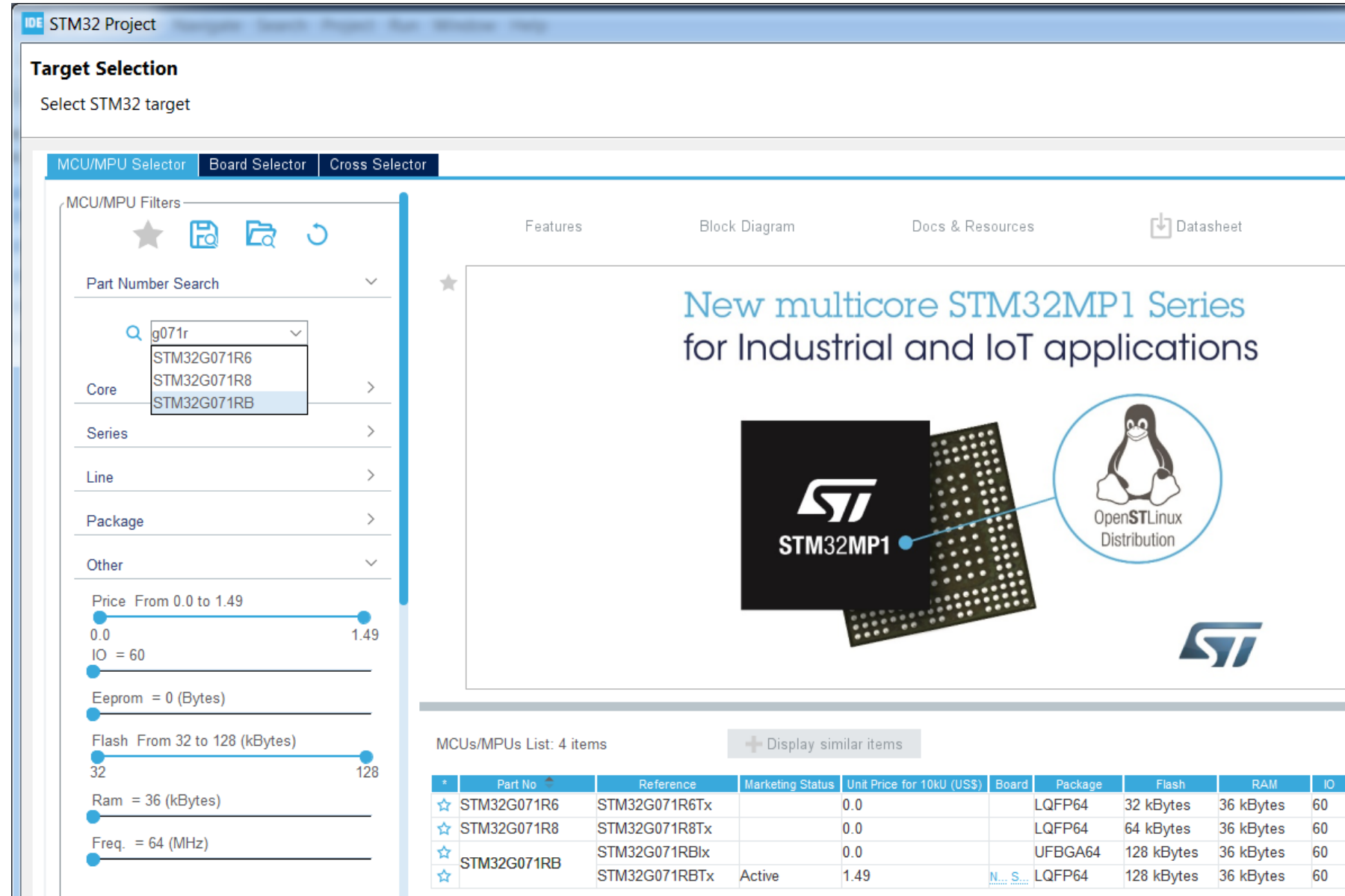
Match

◀ Select your MCU using the pick-lists on the left (you can filter by vendor)

Note:
The information presented in this cross-reference tool is intended to help you narrow your search of ST products based on similarity to other available products. The information is based on data published by other semiconductor manufacturers and might contain errors. ST provides the information "as is" and does not make any representations or warranties as to its accuracy or suitability for any particular purpose. ST recommends that you make your purchase decision based on your review of ST's datasheets and other product documentation. Any pricing information is an estimate for budgetary purposes only.

Select target MCU: STM32G071RBTx

- We will use STM32G071RBTx MCU



The screenshot shows the STM32 Project IDE interface. The 'Target Selection' window is open, displaying the 'MCU/MPU Selector' tab. The 'Part Number Search' field contains 'g071r', and the 'Core' dropdown menu is open, showing 'STM32G071R6', 'STM32G071R8', and 'STM32G071RB'. The 'MCU/MPU Filters' section on the left shows various filters like Price, IO, Eeprom, Flash, Ram, and Freq. The main area displays a promotional banner for the 'New multicore STM32MP1 Series for Industrial and IoT applications'. Below the banner, a table lists the available MCUs/MPUs.

MCU/MPU Filters

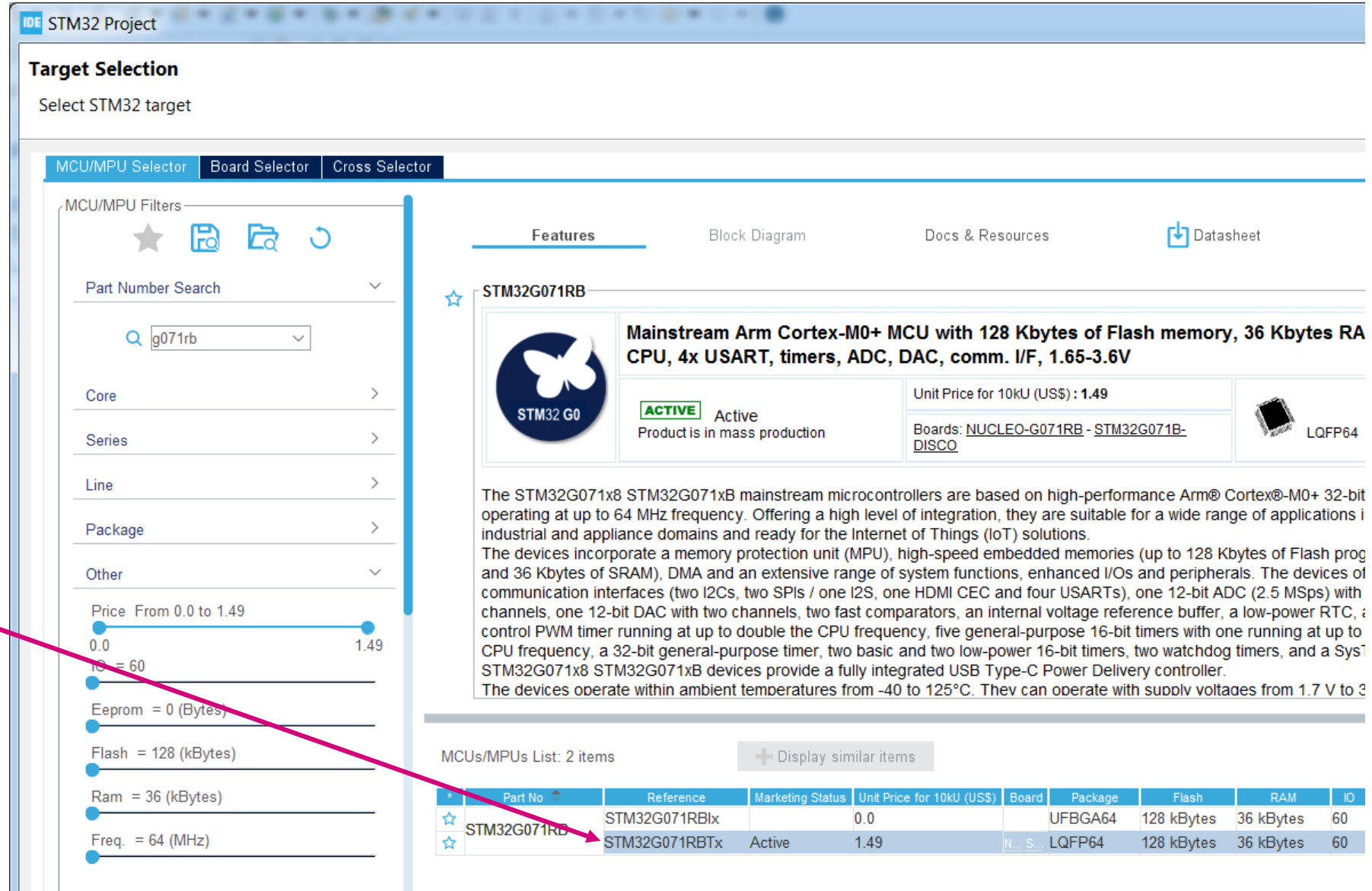
- Part Number Search: g071r
- Core: STM32G071R6, STM32G071R8, STM32G071RB
- Series: >
- Line: >
- Package: >
- Other: >
- Price: From 0.0 to 1.49
- IO: = 60
- Eeprom: = 0 (Bytes)
- Flash: From 32 to 128 (kBytes)
- Ram: = 36 (kBytes)
- Freq.: = 64 (MHz)

MCUs/MPUs List: 4 items

| | Part No | Reference | Marketing Status | Unit Price for 10kU (US\$) | Board | Package | Flash | RAM | IO |
|---|---------------|---------------|------------------|----------------------------|-----------|---------|------------|-----------|----|
| ☆ | STM32G071R6 | STM32G071R6Tx | | 0.0 | | LQFP64 | 32 kBytes | 36 kBytes | 60 |
| ☆ | STM32G071R8 | STM32G071R8Tx | | 0.0 | | LQFP64 | 64 kBytes | 36 kBytes | 60 |
| ☆ | STM32G071RB | STM32G071RBIx | | 0.0 | | UFPGA64 | 128 kBytes | 36 kBytes | 60 |
| ☆ | STM32G071RBTx | STM32G071RBTx | Active | 1.49 | N... S... | LQFP64 | 128 kBytes | 36 kBytes | 60 |

Select target MCU: STM32G071RBTx

- It is possible to view on main MCU features, download its documentation
- To start a new project we need to double click on the part number



Target Selection
Select STM32 target

MCU/MCU Selector | Board Selector | Cross Selector

MCU/MCU Filters

Part Number Search: g071rb

Core: >

Series: >

Line: >

Package: >

Other: >

Price: From 0.0 to 1.49

IO = 60

Eeprom = 0 (Bytes)

Flash = 128 (kBytes)

Ram = 36 (kBytes)

Freq. = 64 (MHz)

STM32G071RB

Mainstream Arm Cortex-M0+ MCU with 128 Kbytes of Flash memory, 36 Kbytes RAM, CPU, 4x USART, timers, ADC, DAC, comm. I/F, 1.65-3.6V

ACTIVE Active
Product is in mass production

Unit Price for 10kU (US\$): 1.49

Boards: [NUCLEO-G071RB - STM32G071B-DISCO](#)

LQFP64

The STM32G071x8 STM32G071xB mainstream microcontrollers are based on high-performance Arm® Cortex®-M0+ 32-bit operating at up to 64 MHz frequency. Offering a high level of integration, they are suitable for a wide range of applications in industrial and appliance domains and ready for the Internet of Things (IoT) solutions.

The devices incorporate a memory protection unit (MPU), high-speed embedded memories (up to 128 Kbytes of Flash program and 36 Kbytes of SRAM), DMA and an extensive range of system functions, enhanced I/Os and peripherals. The devices of communication interfaces (two I2Cs, two SPIs / one I2S, one HDMI CEC and four USARTs), one 12-bit ADC (2.5 MSps) with channels, one 12-bit DAC with two channels, two fast comparators, an internal voltage reference buffer, a low-power RTC, a control PWM timer running at up to double the CPU frequency, five general-purpose 16-bit timers with one running at up to CPU frequency, a 32-bit general-purpose timer, two basic and two low-power 16-bit timers, two watchdog timers, and a System

STM32G071x8 STM32G071xB devices provide a fully integrated USB Type-C Power Delivery controller.

The devices operate within ambient temperatures from -40 to 125°C. They can operate with supply voltages from 1.7 V to 3.6 V.

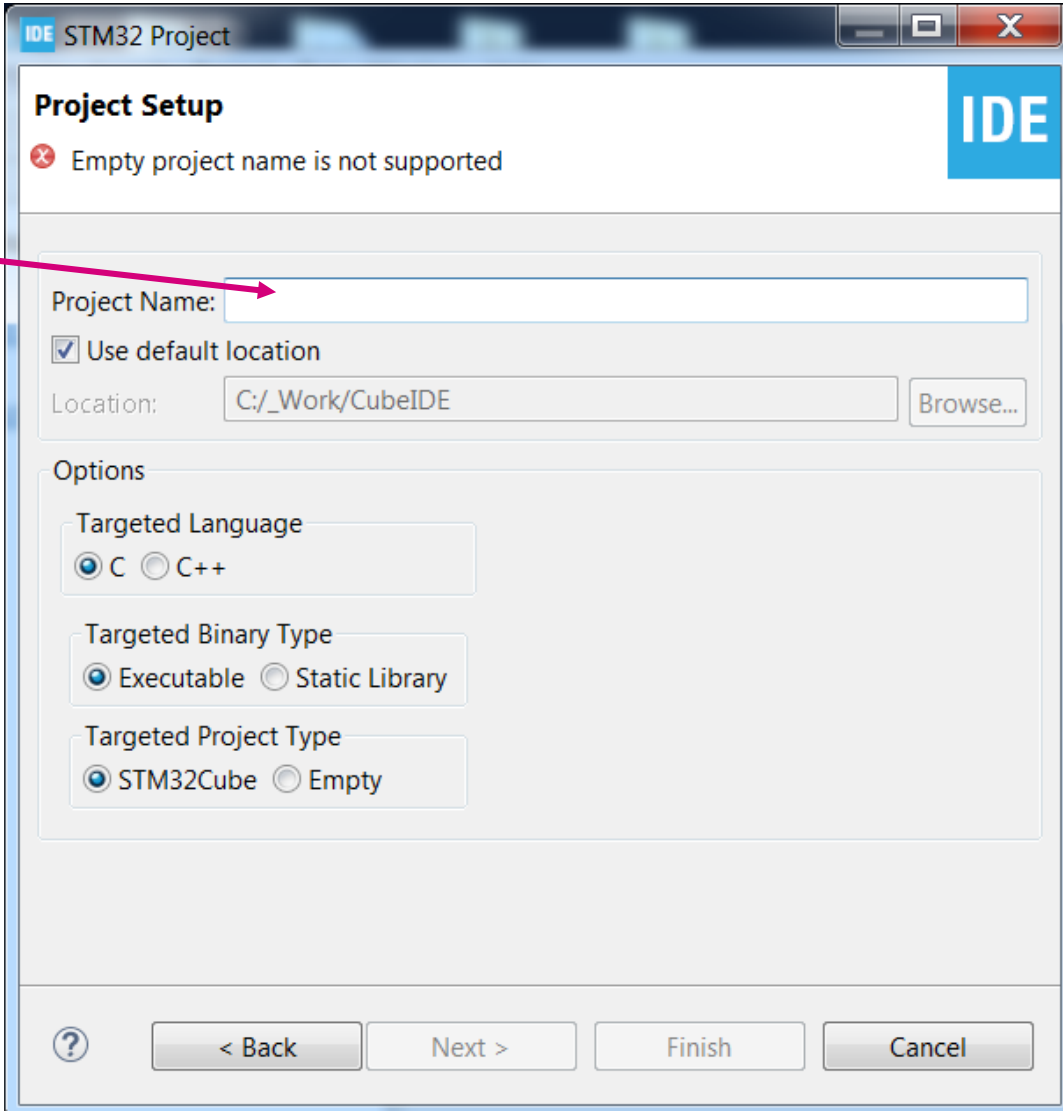
MCUs/MPUs List: 2 items

| | Part No | Reference | Marketing Status | Unit Price for 10kU (US\$) | Board | Package | Flash | RAM | IO |
|---|---------------|---------------|------------------|----------------------------|-----------|---------|------------|-----------|----|
| ☆ | STM32G071RBx | STM32G071RBx | 0.0 | | | UFPGA64 | 128 kBytes | 36 kBytes | 60 |
| ☆ | STM32G071RBTx | STM32G071RBTx | Active | 1.49 | N... S... | LQFP64 | 128 kBytes | 36 kBytes | 60 |

Display similar items

Enter project name

- Specify project name, optionally its location (if different from workspace one)
- Additionally we can specify target language (C or C++), binary type (executable or static library) and project type (generated by STM32CubeMX or an empty one)



STM32 Project

Project Setup

Empty project name is not supported

Project Name:

☒ Use default location

Location:

Options

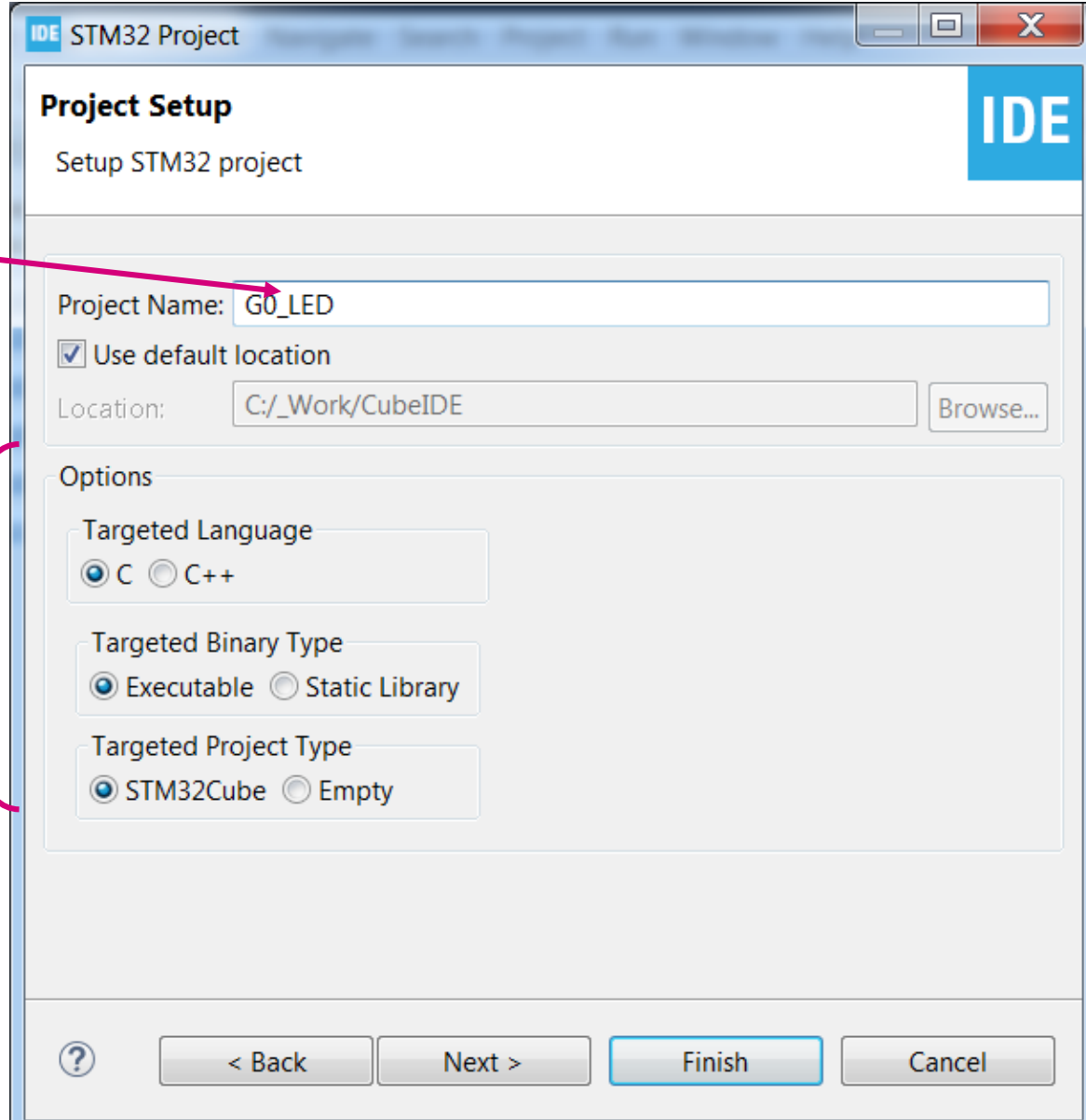
Targeted Language
☒ C ☐ C++

Targeted Binary Type
☒ Executable ☐ Static Library

Targeted Project Type
☒ STM32Cube ☐ Empty

Enter project name

- Specify project name, optionally its location (if different from workspace one)
- Additionally we can specify target language (C or C++), binary type (executable or static library) and project type (generated by STM32CubeMX or an empty one)



IDE STM32 Project

Project Setup

Setup STM32 project

Project Name: GO_LED

☒ Use default location

Location: C:/_Work/CubeIDE Browse...

Options

Targeted Language

☒ C ☐ C++

Targeted Binary Type

☒ Executable ☐ Static Library

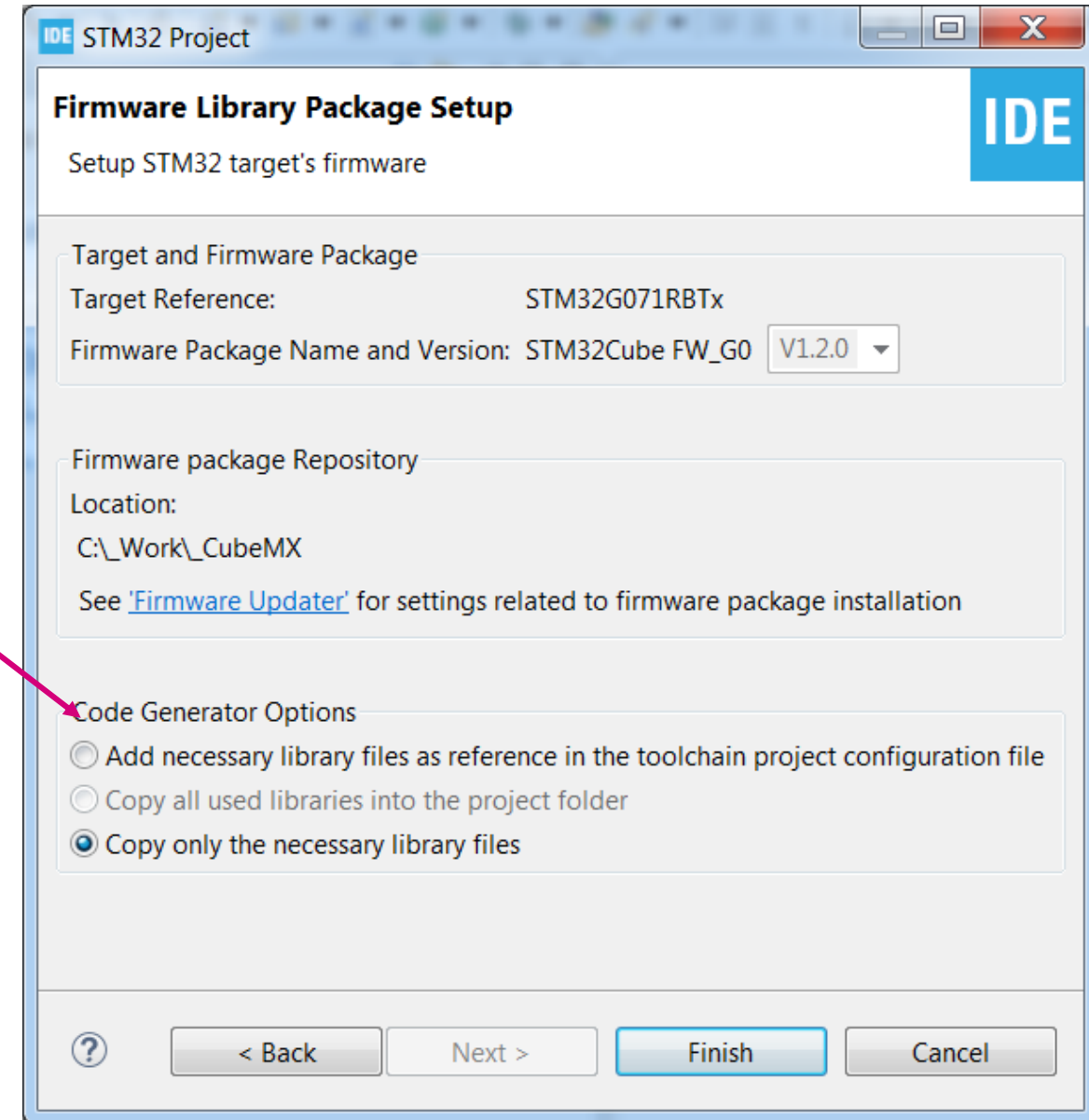
Targeted Project Type

☒ STM32Cube ☐ Empty

? < Back Next > Finish Cancel

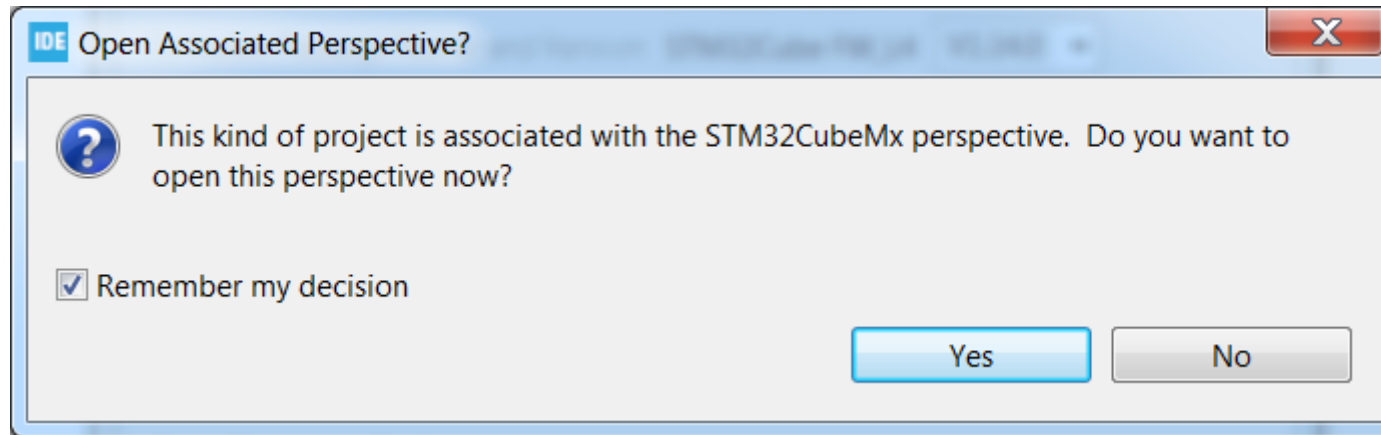
Select code generation options

- It is possible to specify code generator options (which library files would be added to the project)

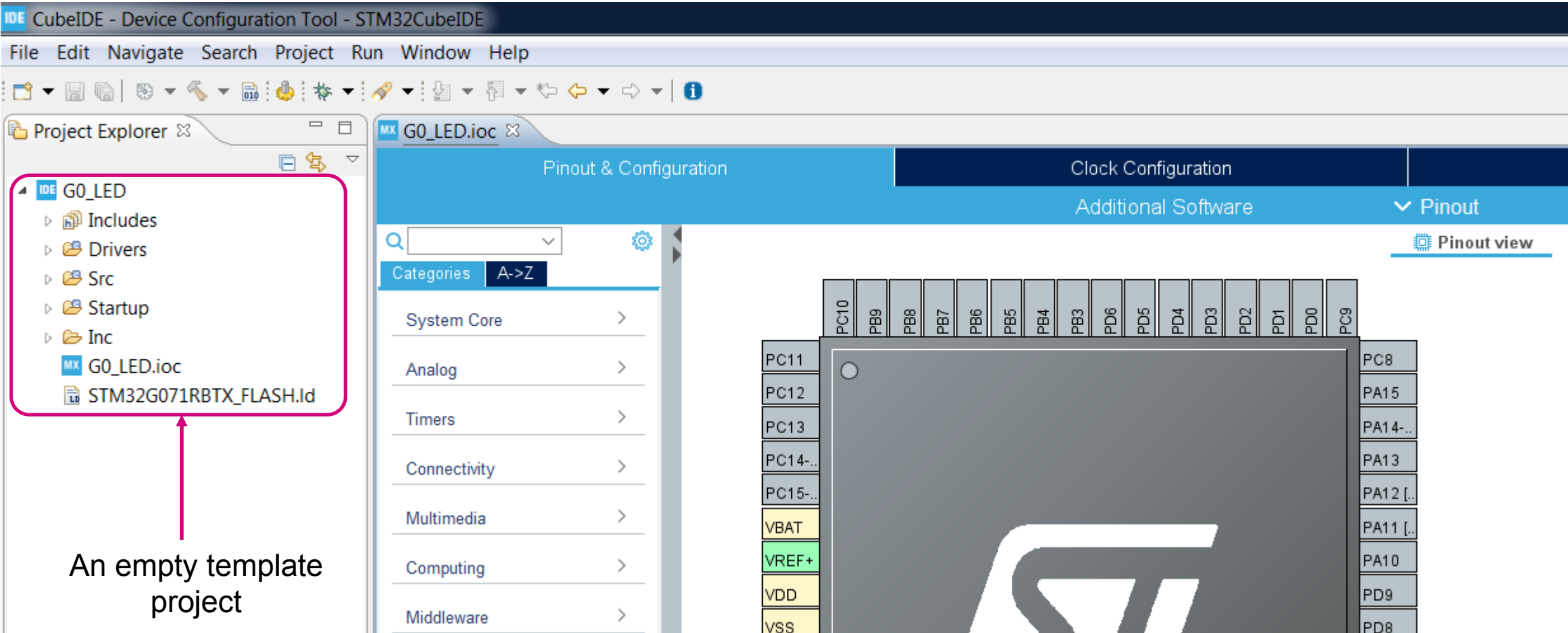


Confirm change of perspective

- At the end of project generation there is an information window about change of perspective to STM32CubeMX one



New project in STM32CubeMX perspective



The screenshot shows the STM32CubeMX IDE interface. The title bar reads "IDE CubeIDE - Device Configuration Tool - STM32CubeIDE". The menu bar includes File, Edit, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and configuration.

The **Project Explorer** on the left shows a project named **GO_LED** with the following structure:

- Includes
- Drivers
- Src
- Startup
- Inc
- GO_LED.ioc** (selected)
- STM32G071RBTX_FLASH.Id

A pink box highlights the **GO_LED** project, and a pink arrow points to it with the text: "An empty template project".

The main workspace displays the **Pinout & Configuration** tab for the **GO_LED.ioc** file. It features a search bar, a "Categories" dropdown set to "A->Z", and a list of categories with expandable arrows:

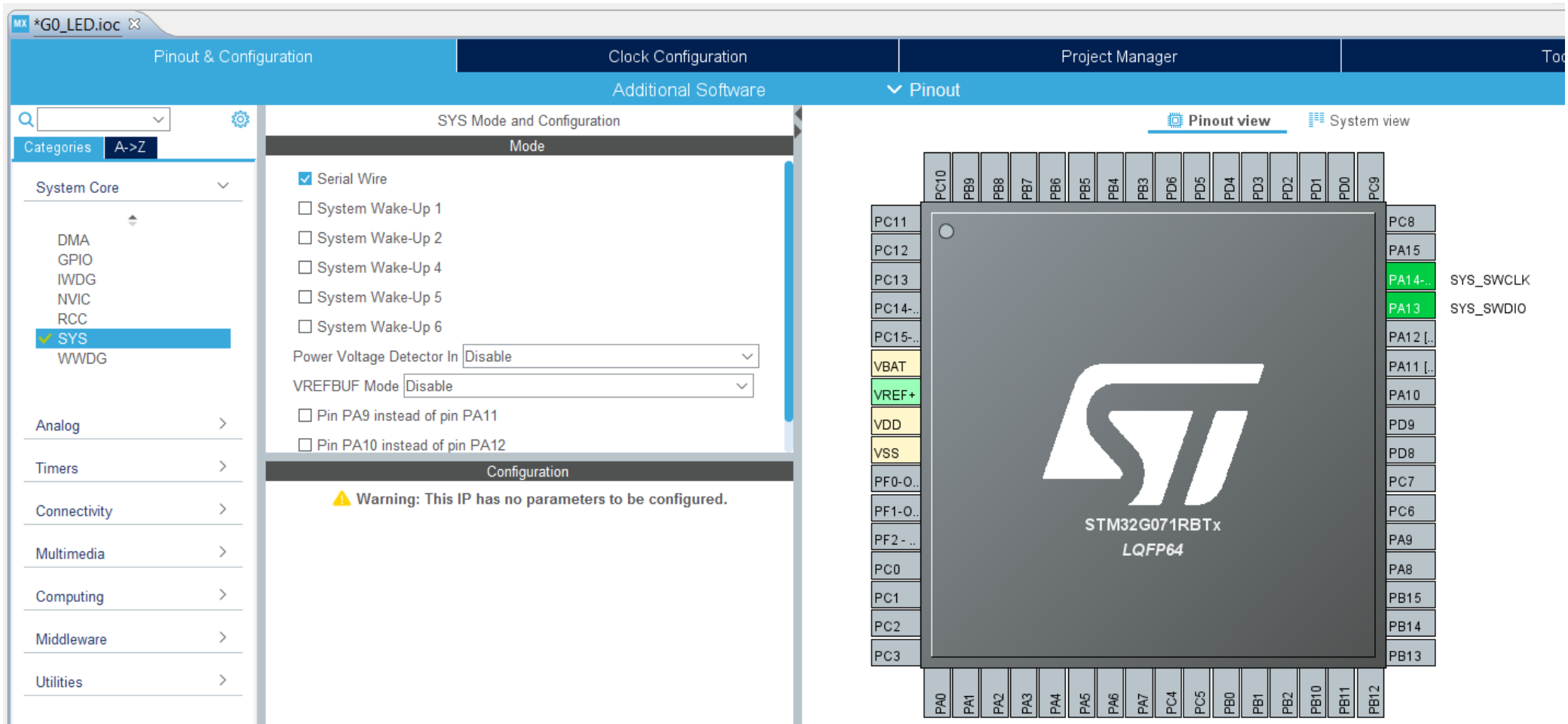
- System Core
- Analog
- Timers
- Connectivity
- Multimedia
- Computing
- Middleware

On the right, the **Pinout view** shows a pinout diagram for the STM32G071RBTX. The pins are arranged in a grid:

| | | | | | | | | | | | | | | | |
|------|------|------|---------|---------|------|-------|-----|-----|-----|------|---------|------|----------|----------|------|
| PC10 | PB9 | PB8 | PB7 | PB6 | PB5 | PB4 | PB3 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 | PC9 |
| PC11 | PC12 | PC13 | PC14-.. | PC15-.. | VBAT | VREF+ | VDD | VSS | PC8 | PA15 | PA14-.. | PA13 | PA12 [.. | PA11 [.. | PA10 |
| | | | | | | | | | | | | | | | PD9 |
| | | | | | | | | | | | | | | | PD8 |

Enabling Serial Wire debug interface

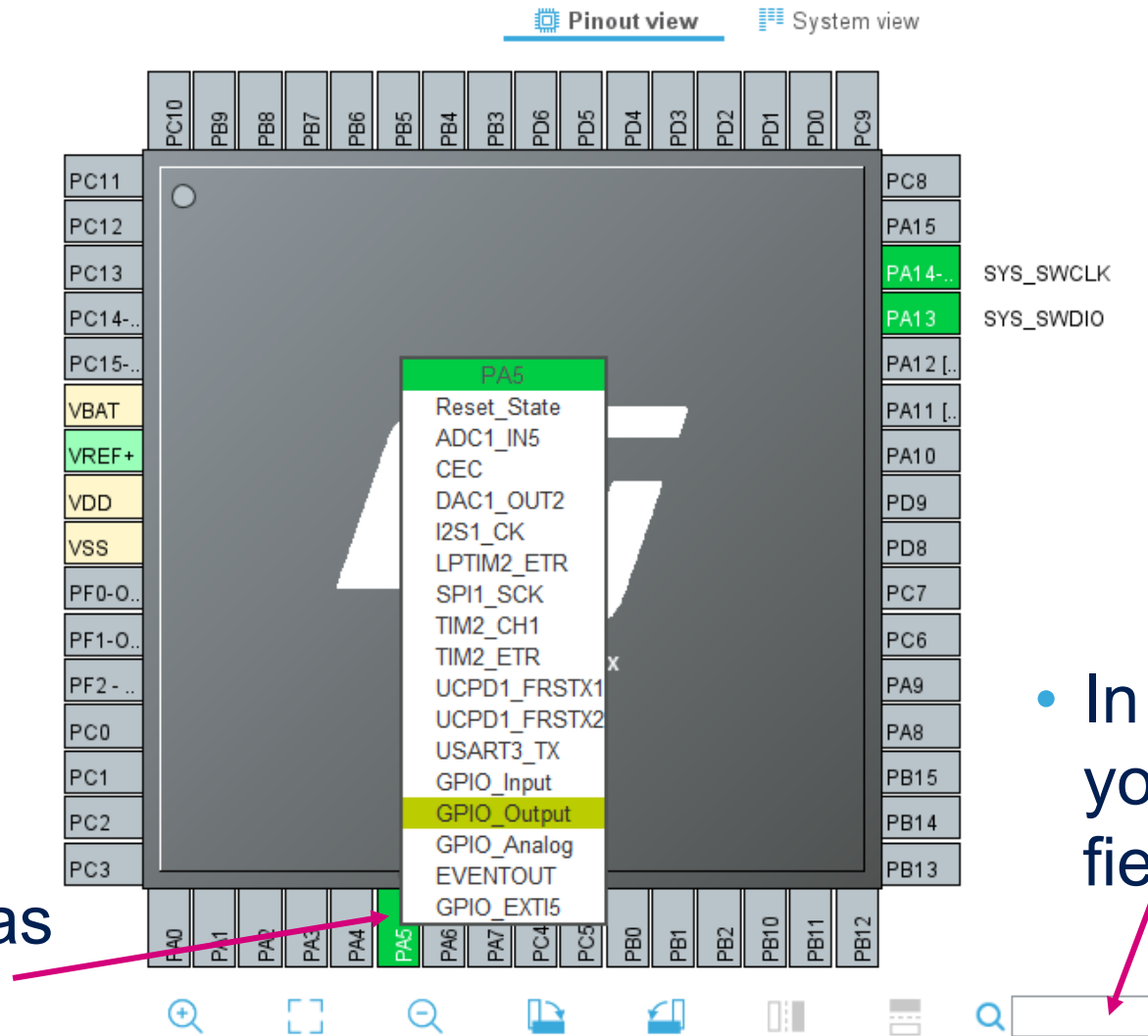
- Select “Serial Wire” from System Core -> SYS peripheral group
- As a result PA13 and PA14 will be assigned to SWD interface



The screenshot shows the STM32CubeIDE Pinout & Configuration window for the project *G0_LED.ioc. The 'Pinout & Configuration' tab is active, and the 'SYS Mode and Configuration' section is expanded. Under the 'Mode' section, 'Serial Wire' is checked, and 'System Wake-Up 1' through 'System Wake-Up 6' are unchecked. The 'Power Voltage Detector In' is set to 'Disable', and the 'VREFBUF Mode' is set to 'Disable'. Under the 'Configuration' section, 'Pin PA9 instead of pin PA11' and 'Pin PA10 instead of pin PA12' are unchecked. A warning message states: 'Warning: This IP has no parameters to be configured.'

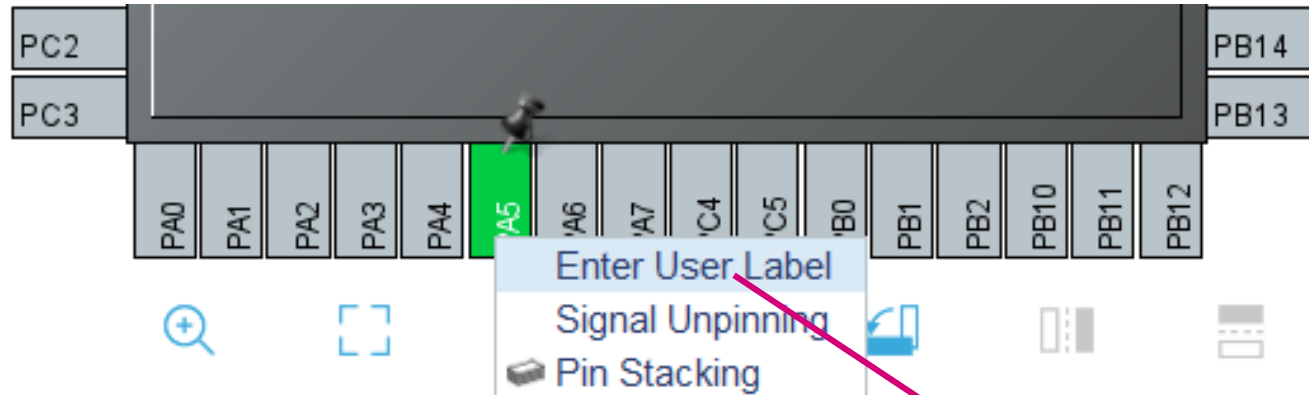
The 'Pinout view' is displayed on the right, showing the pin configuration for the STM32G071RBTx LQFP64 package. The pins are arranged in a grid around the chip. The pins PA13 and PA14 are highlighted in green, indicating they are assigned to the SWD interface. The labels for these pins are 'PA14-...' and 'PA13', with the corresponding functions 'SYS_SWCLK' and 'SYS_SWDIO' listed to the right.

Configuring PA5 as Output



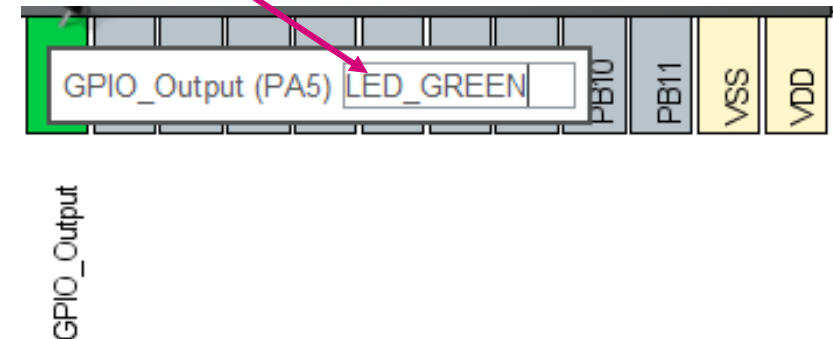
Assign label to PA5

- Using select Enter User Label and insert LED_GREEN label

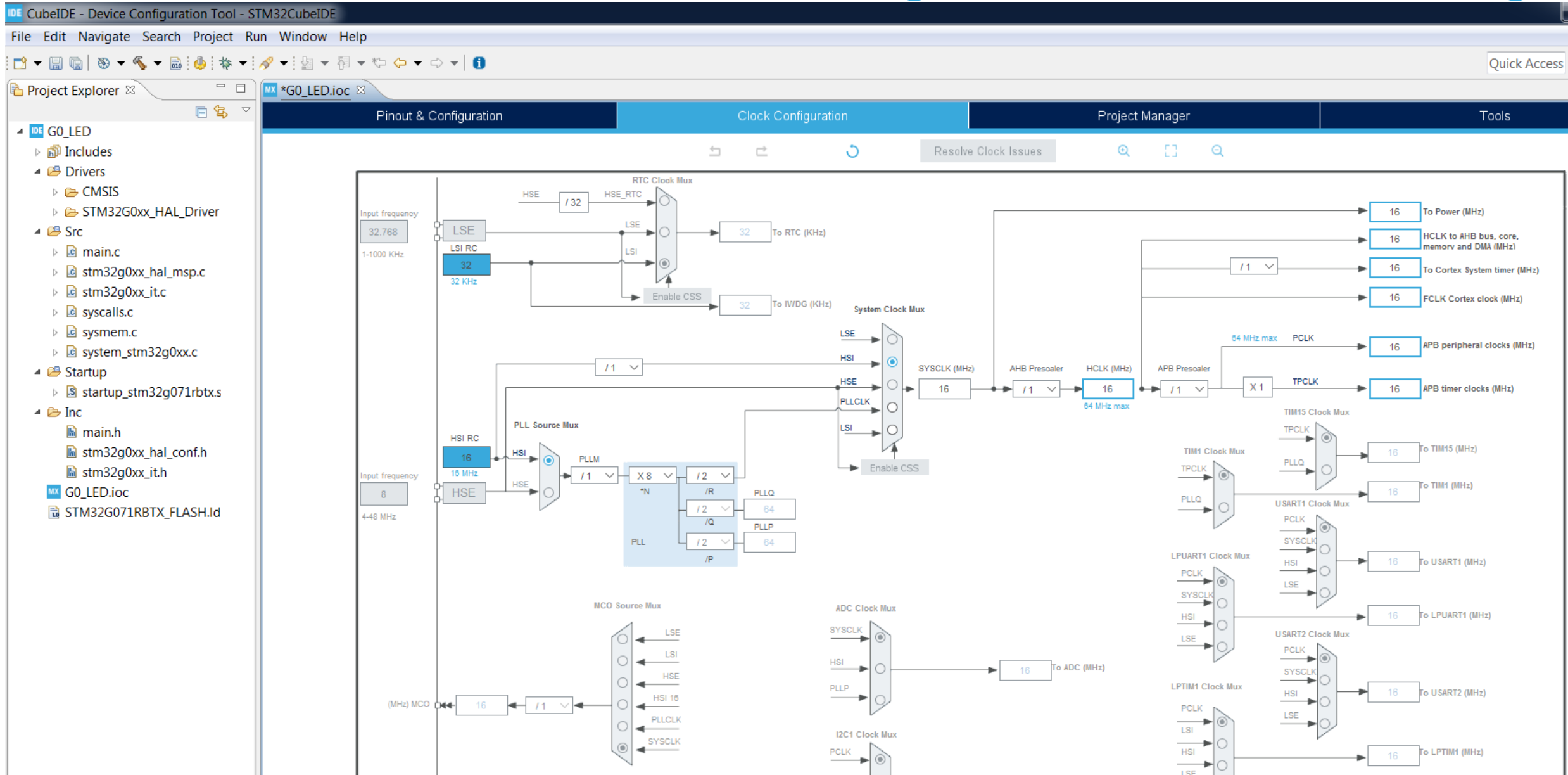


Hint:

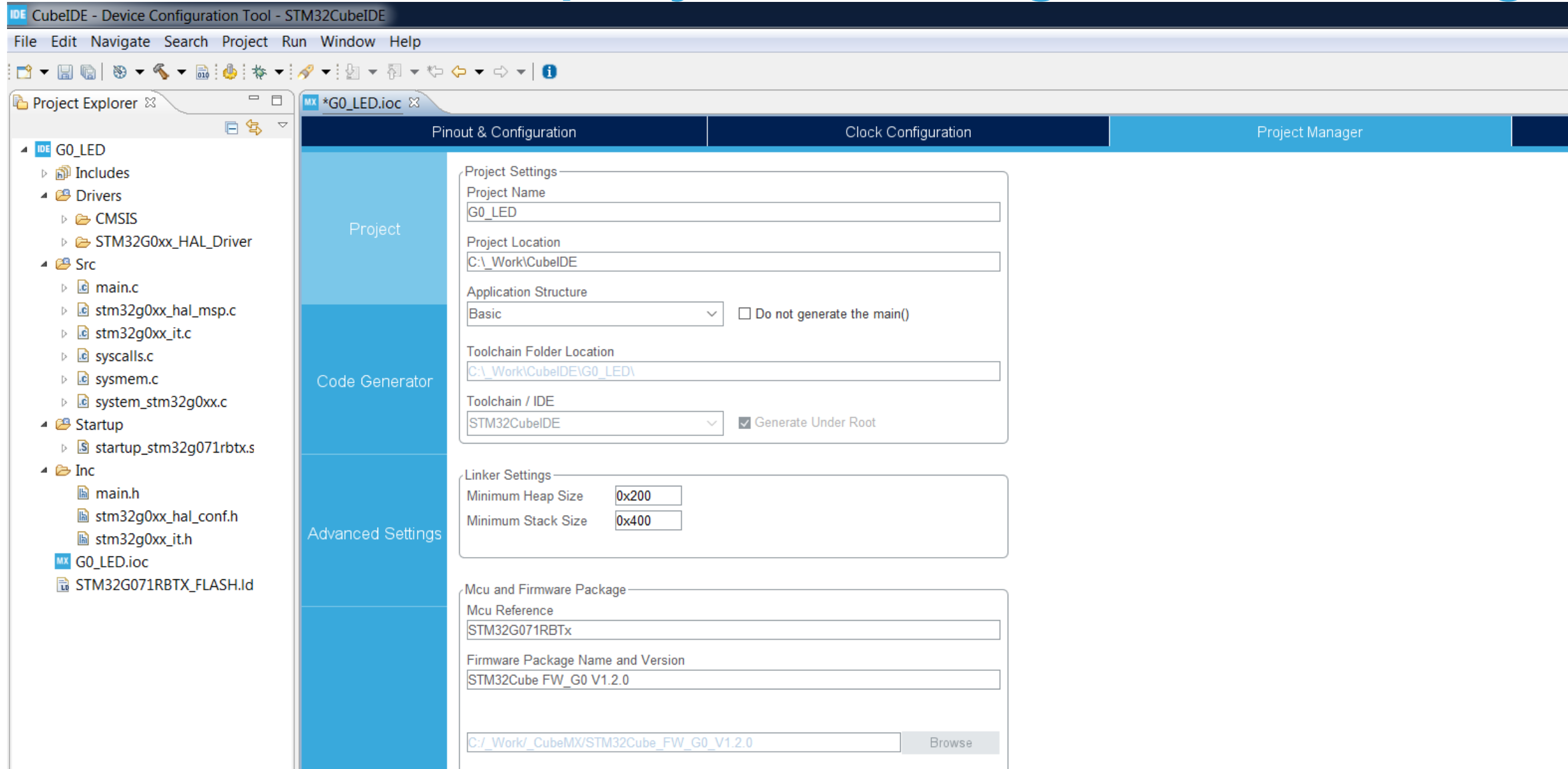
Labels are defined in **main.h** file within generated project (private defines section)



Default clock configuration – no change



Basic project settings – no change

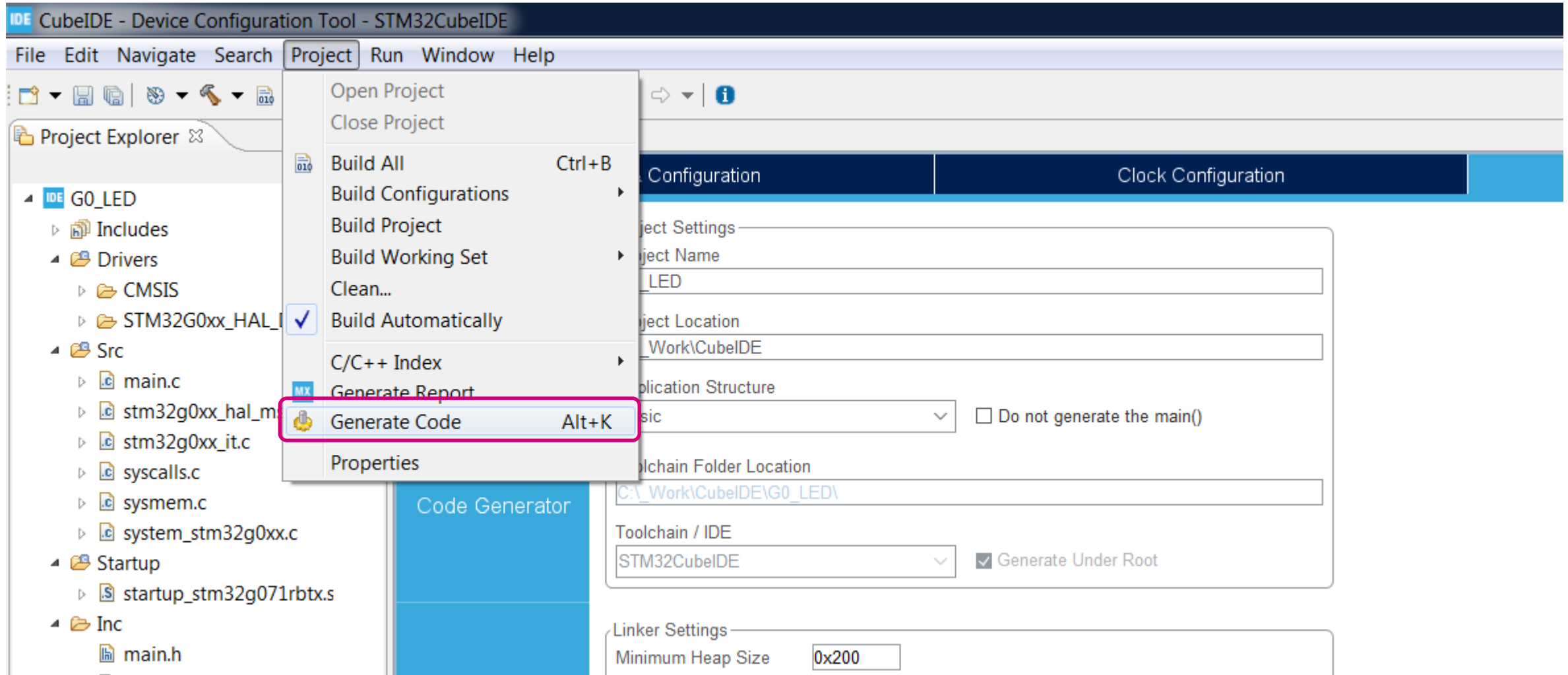


The screenshot displays the STM32CubeIDE interface with the 'G0_LED' project selected. The 'Project Explorer' on the left shows the project structure, including 'Includes', 'Drivers', 'Src', 'Startup', and 'Inc' folders. The main workspace is divided into three tabs: 'Pinout & Configuration', 'Clock Configuration', and 'Project Manager'. The 'Project Manager' tab is active, showing the 'Project Settings' section. The settings are as follows:

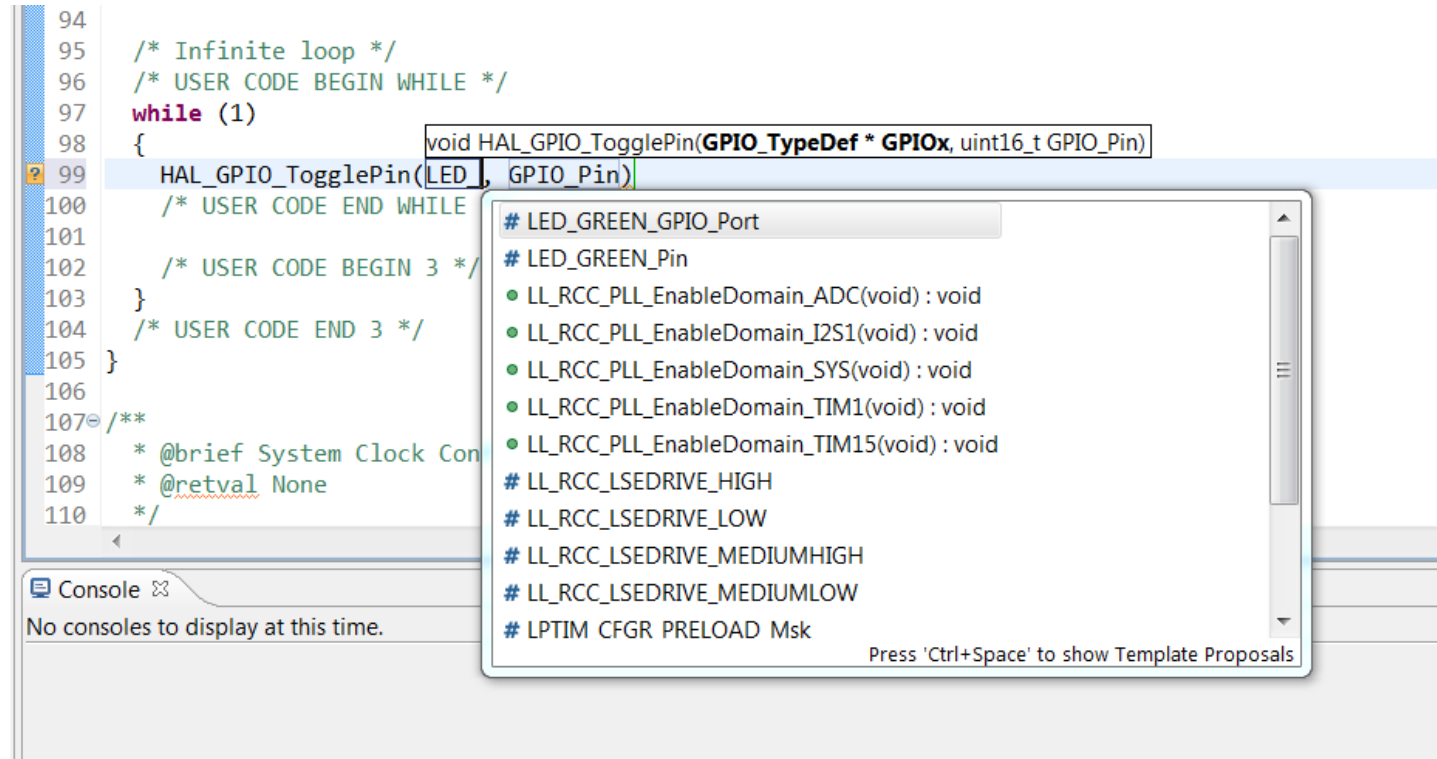
- Project Settings**
 - Project Name: G0_LED
 - Project Location: C:_Work\CubeIDE
 - Application Structure: Basic (dropdown menu) ☐ Do not generate the main()
 - Toolchain Folder Location: C:_Work\CubeIDE\G0_LED\
 - Toolchain / IDE: STM32CubeIDE (dropdown menu) ☒ Generate Under Root
- Linker Settings**
 - Minimum Heap Size: 0x200
 - Minimum Stack Size: 0x400
- Mcu and Firmware Package**
 - Mcu Reference: STM32G071RBTx
 - Firmware Package Name and Version: STM32Cube FW_G0 V1.2.0
 - Path: C:/_Work/_CubeMX/STM32Cube_FW_G0_V1.2.0 (with a 'Browse' button)

Code generation

- It is necessary to add to an empty template project our configuration done



- We can use Ctrl+Space help to find proper function or argument
- Please use “USER CODE” sections only



```

94
95  /* Infinite loop */
96  /* USER CODE BEGIN WHILE */
97  while (1)
98  {
99      HAL_GPIO_TogglePin(LED_1, GPIO_Pin)
100  /* USER CODE END WHILE */
101
102  /* USER CODE BEGIN 3 */
103  }
104  /* USER CODE END 3 */
105 }
106
107 /**
108  * @brief System Clock Configuration
109  * @retval None
110  */

```

void HAL_GPIO_TogglePin(GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)

- # LED_GREEN_GPIO_Port
- # LED_GREEN_Pin
- LL_RCC_PLL_EnableDomain_ADC(void) : void
- LL_RCC_PLL_EnableDomain_I2S1(void) : void
- LL_RCC_PLL_EnableDomain_SYS(void) : void
- LL_RCC_PLL_EnableDomain_TIM1(void) : void
- LL_RCC_PLL_EnableDomain_TIM15(void) : void
- # LL_RCC_LSEDRIVE_HIGH
- # LL_RCC_LSEDRIVE_LOW
- # LL_RCC_LSEDRIVE_MEDIUMHIGH
- # LL_RCC_LSEDRIVE_MEDIUMLOW
- # LPTIM_CFGR_PRELOAD_Msk

Press 'Ctrl+Space' to show Template Proposals

Console No consoles to display at this time.

Final code (main.c file)

- We need to toggle LED_GREEN pin with 500ms delay in between

```

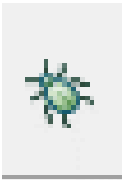
89  /* Initialize all configured peripherals */
90  MX_GPIO_Init();
91  /* USER CODE BEGIN 2 */
92
93  /* USER CODE END 2 */
94
95  /* Infinite loop */
96  /* USER CODE BEGIN WHILE */
97  while (1)
98  {
99      HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin);
100     HAL_Delay(500);
101     /* USER CODE END WHILE */
102
103     /* USER CODE BEGIN 3 */
104 }
105 /* USER CODE END 3 */
106 }

```

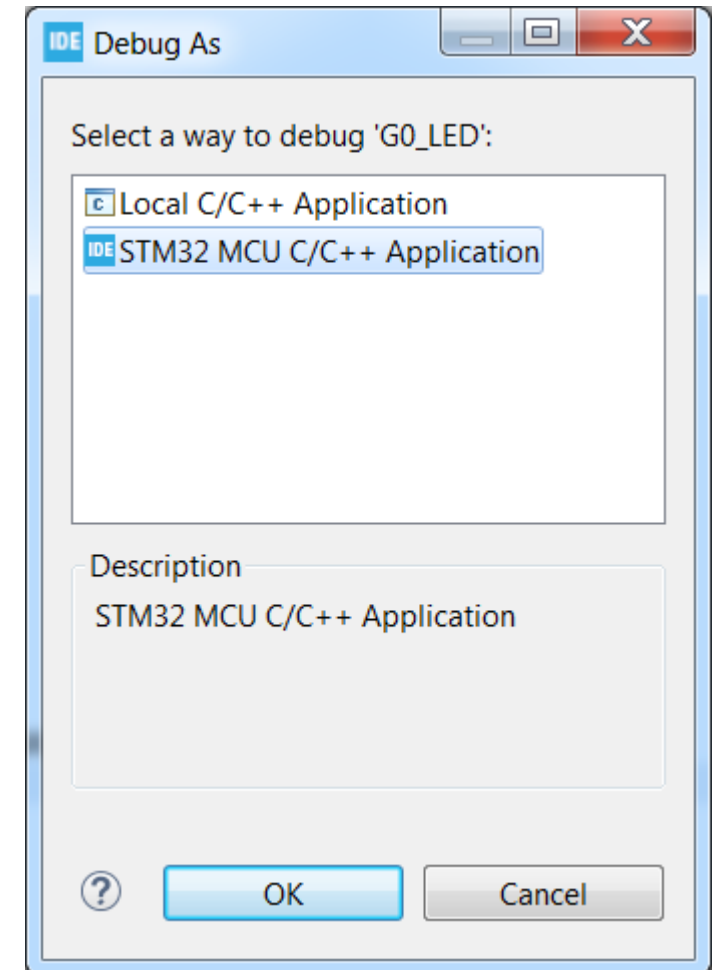
Compile and run debug session



- We can compile the code with “hammer” or build with automatic start of debug session using “bug” button

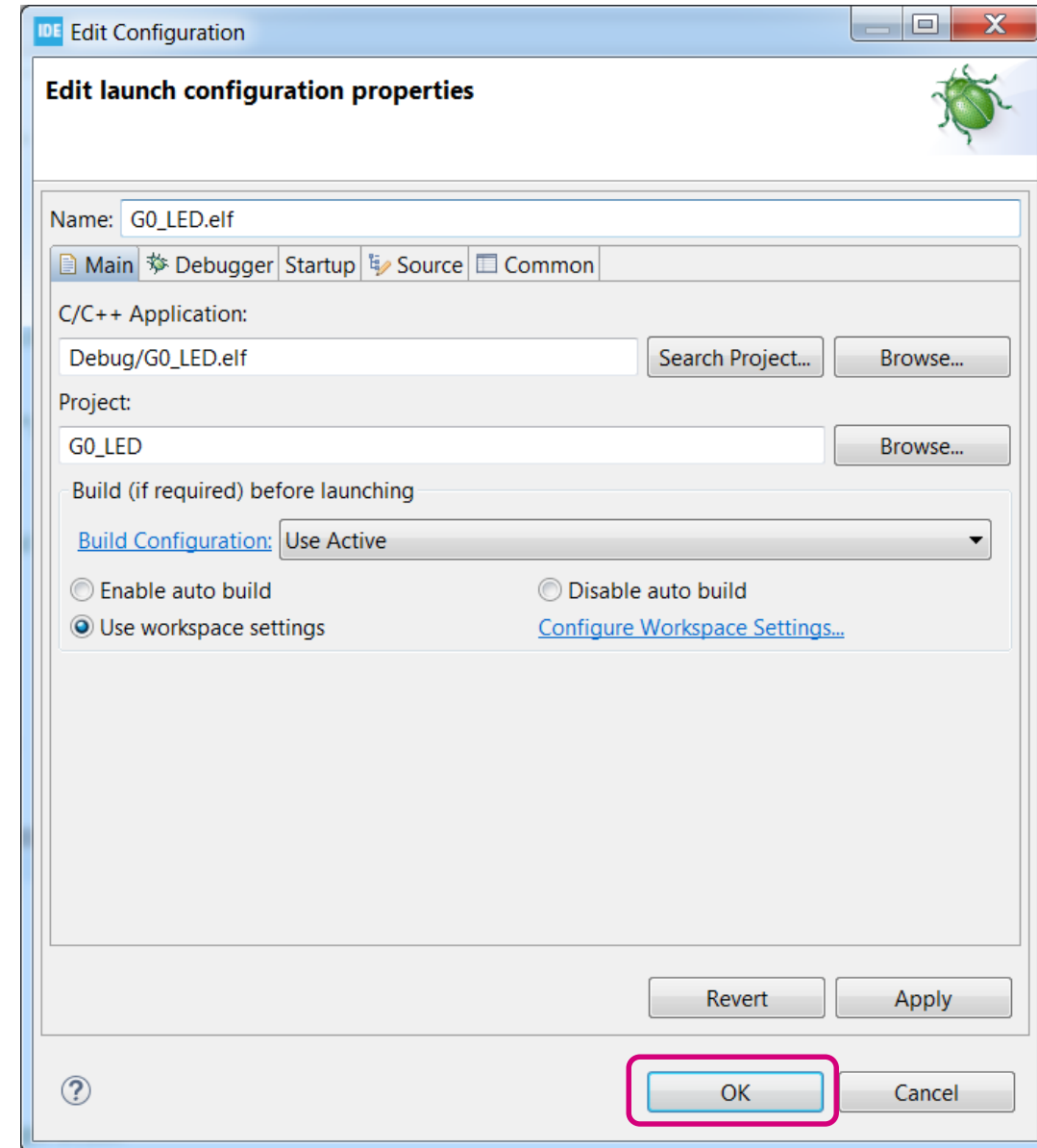


- Starting debug session we will be asked how we would like to debug the code



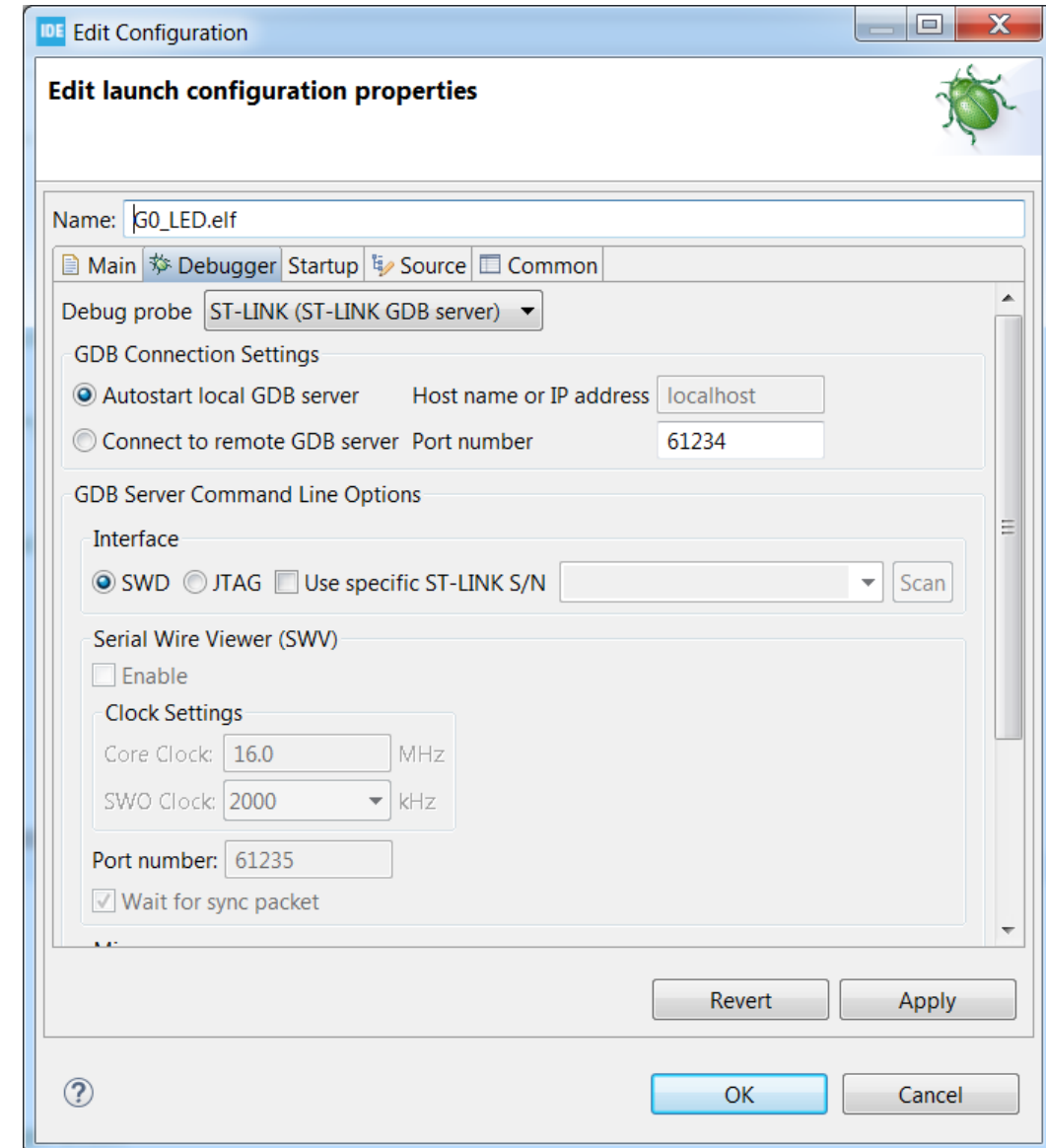
- When running debug (after successful compilation) there is no need to configure anything for debug
- Just connect target board to STLink and press ok

Debug settings 1/3



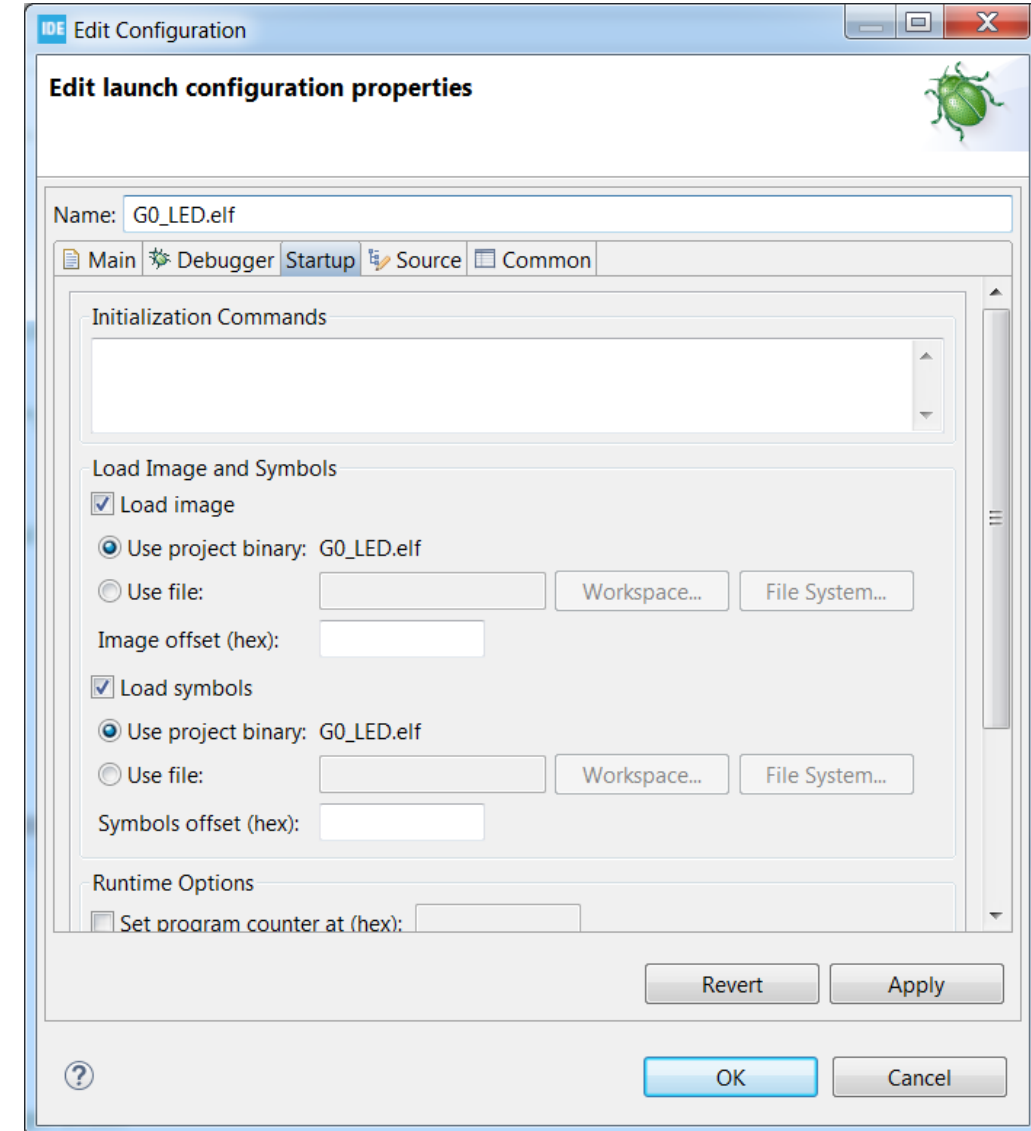
Debug settings 2/3

- It is possible to select debug probe and its interface



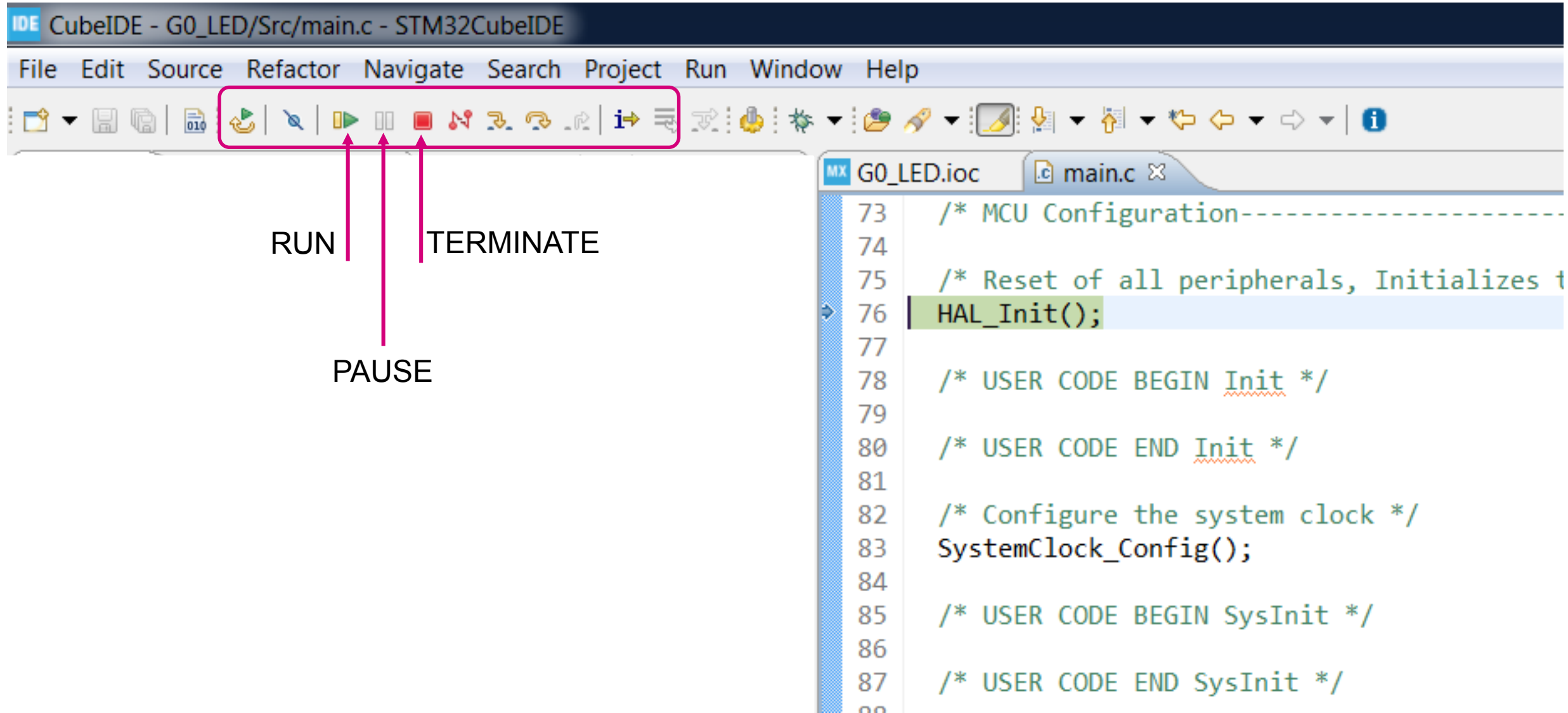
Debug settings 3/3

- Some more advanced commands are possible as well



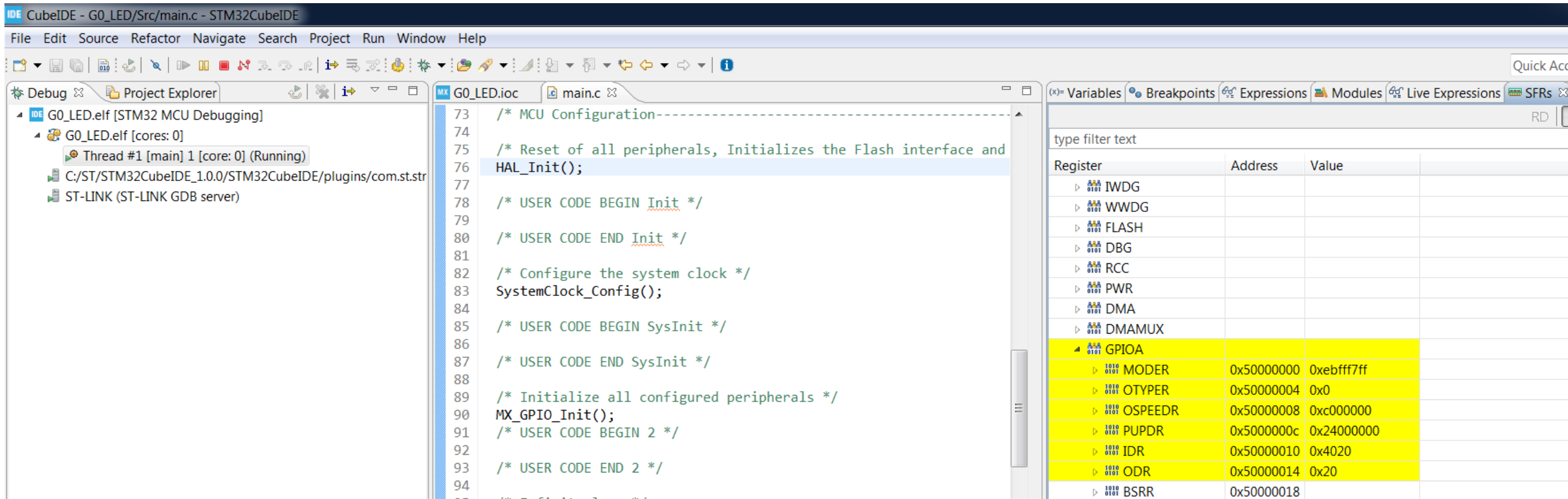
Debug perspective (toolbar)

- Once we enter into debug session, a new debug toolbar pop up



Debug perspective – registers view

- Within debug perspective it is possible to monitor code flow, variables and registers content (on application pause, breakpoint stop or live)



The screenshot displays the STM32CubeIDE interface in the debug perspective. The left pane shows the 'Project / debug explorer' with the 'G0_LED.elf' project and a running thread. The middle pane shows the 'Source code' for 'main.c', with the 'HAL_Init()' function highlighted. The right pane shows the 'Registers view' with a table of registers and their values.

| Register | Address | Value |
|----------|------------|------------|
| IWDG | | |
| WWDG | | |
| FLASH | | |
| DBG | | |
| RCC | | |
| PWR | | |
| DMA | | |
| DMAMUX | | |
| GPIOA | | |
| MODER | 0x50000000 | 0xebff7ff |
| OTYPER | 0x50000004 | 0x0 |
| OSPEEDR | 0x50000008 | 0xc000000 |
| PUPDR | 0x5000000c | 0x24000000 |
| IDR | 0x50000010 | 0x4020 |
| ODR | 0x50000014 | 0x20 |
| BSRR | 0x50000018 | |

Project / debug explorer

Source code

Variables, breakpoints, registers view

... Let's check it

- After all code processing we can build the project, start debug session and run the application
- As an effect Green LED should toggle each 1second

