



life.augmented



STM32H5 I/Os output state retention in STANDBY

I/Os output state retention in STANDBY

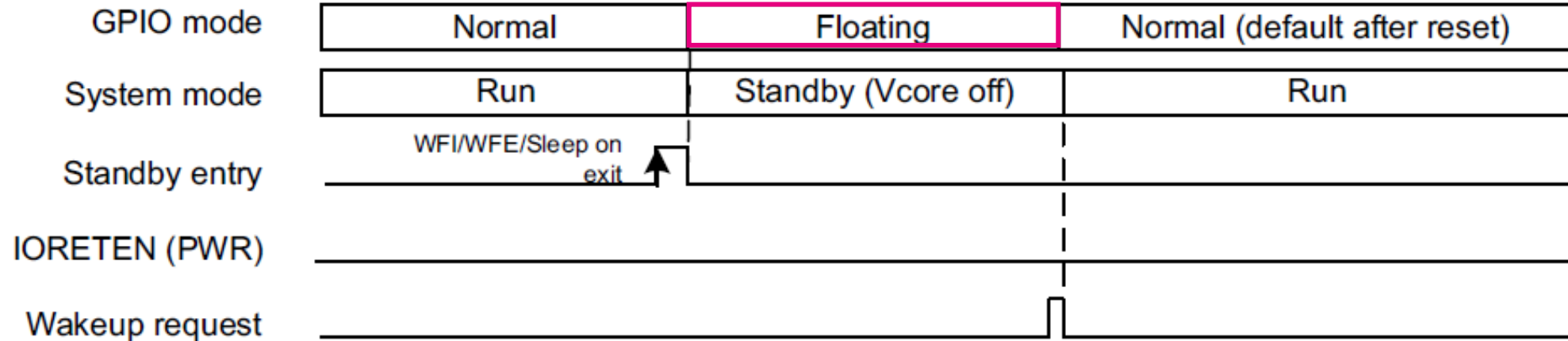
- In the STANDBY mode the I/Os are by default in floating (High-Z) state (not being driven to any defined logic level)
- This might be **an issue for control signals for external devices** such as chip select/slave select or low-power mode control signal, as we need to **ensure the proper voltage level is maintained** even if the MCU is in a deep low-power mode such as STANDBY
- If the I/Os output state retention is enabled, **the I/O state is automatically sampled at STANDBY entry and the corresponding voltage level is applied through a pull-up or a pull-down resistor**

I/Os output state retention after wake-up

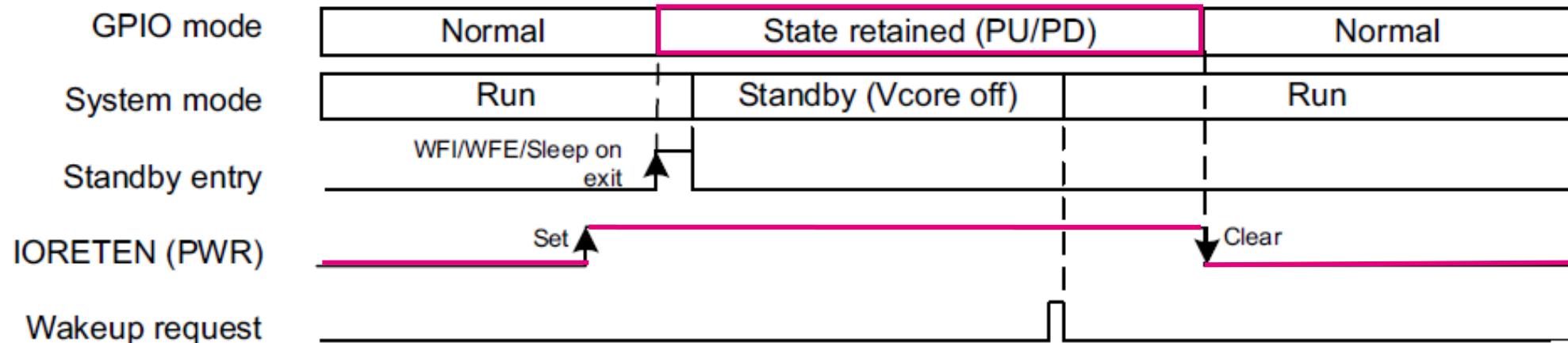
- The pull-up/pull-down resistors remain connected and maintain the voltage level until the I/O state retention is disabled by software
(thus, they are still active even after wake-up from STANDBY in the RUN mode)
- They should be disabled after reset (= wake-up from STANDBY),
as soon as peripherals and GPIOs are configured and ready to drive the signals

I/Os output state retention in STANDBY

IO state retention disabled



IO state retention enabled



Hands-on exercise: I/Os output state retention in STANDBY

Objective

- The purpose of this LAB is to:
 - Get familiar with the **NUCLEO-H563ZI** development board
 - Observe the difference in behavior w/ and w/o **I/Os state retention** enabled
 - Use HAL peripheral drivers
(part of STM32Cube software ecosystem for STM32 MCUs)
- Steps:
 - Configure MCU pinout and peripherals in STM32CubeMX
 - Extend the generated code with HAL functions to observe the I/Os state retention
 - Verify the correct functionality

Let's get started

Open STM32CubeIDE

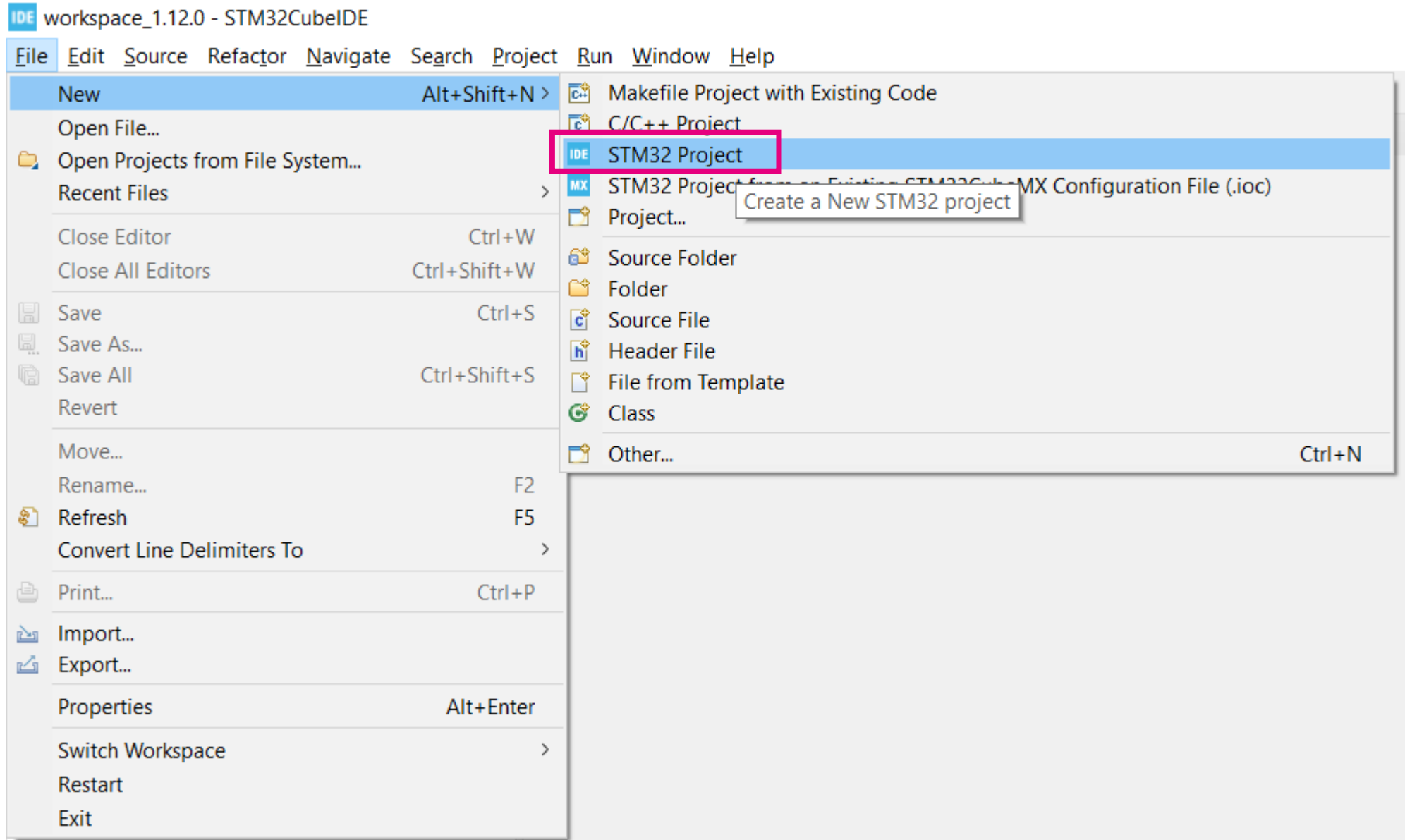
IDE

STM32
CubeIDE



File -> New -> STM32 Project

Start a new STM32 project



Select STM32H563ZITx in MCU/MPU Selector

STM32 Project

Target Selection
Select STM32 target or STM32Cube example

MCU/MPU Selector Board Selector Example Selector Cross Selector

MCU/MPU Filters

Commercial Part Number **1** H563ZI

PRODUCT INFO

- Segment >
- Series >
- Line >
- Marketing Status >
- Price >
- Package >
- Core >
- Coprocessor >

MEMORY

Flash = 2048 (kBytes)

2048

Features

Block Diagram Docs & Resources CAD Resources Datasheet Buy

STM32H5 Series

STM32H563ZIT6 High-performance, Arm Cortex-M33 with TrustZone, MCU with 2MByte Flash, 640KB RAM, 250 MHz CPU

COMING SOON
Stay tuned !

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

Board: [NUCLEO-H563ZI](#)

LQFP 144 20x20x1.4 mm

High-performance, Arm Cortex-M33 with TrustZone, MCU with 2MByte Flash, 640KB RAM, 250 MHz CPU

Features

MCUs/MPUs List: 2 items

+ Display similar items

Export

	Part	Reference	Ma...	Unit Price...	Board	Package	Flash	RAM	Commercial Part No
☆	STM32H563ZIT6	STM32H563ZIT6	Com...	NA		LQFP 144 ...	2048 kBytes	640 kBytes	STM32H563ZIT6Q
☆	STM32H563ZITx	STM32H563ZITx	Com...	NA	NUCLEO-H5...	LQFP 144 ...	2048 kBytes	640 kBytes	STM32H563ZIT6

2

3

< Back Next > Finish Cancel

Choose project name and click on „Finish“

IDE STM32 Project

Setup STM32 project

Project

Project Name: STM32H5_STANDBY_IORetention

☒ Use default location

Location: C:/Users/jaroslav becka/STM32CubeIDE/workspace_1.12. Browse...

Options

Targeted Language

☒ C ☐ C++

Targeted Device Usage

☐ Enable TrustZone

Targeted Binary Type

☒ Executable ☐ Static Library

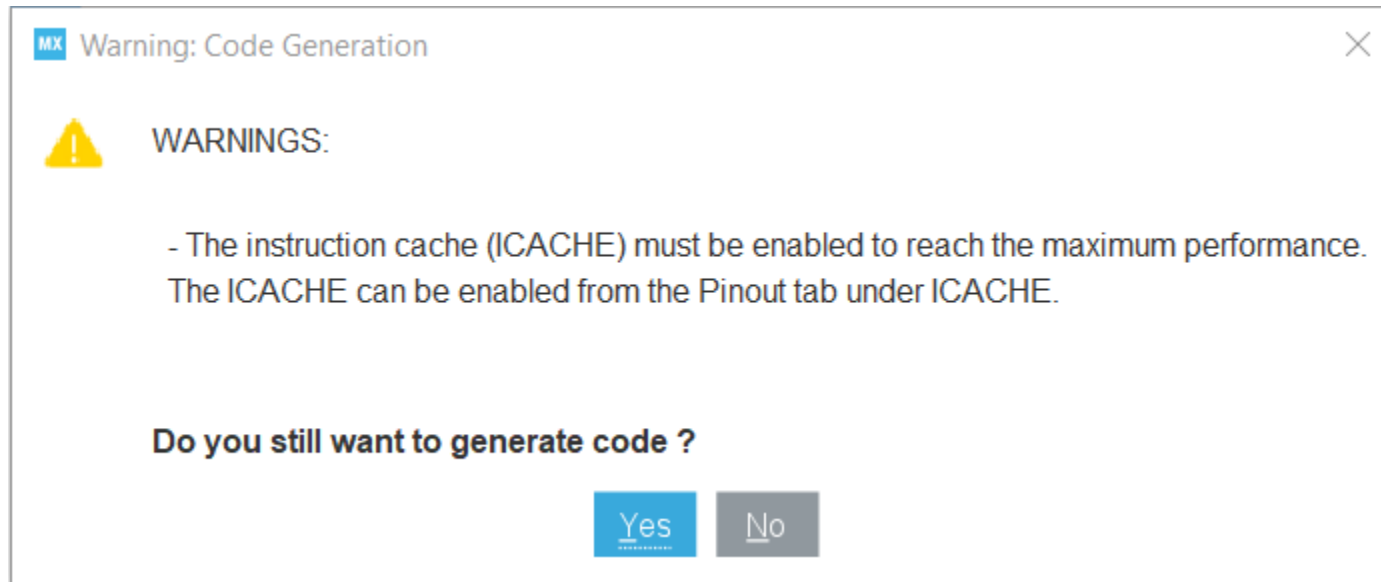
Targeted Project Type

☒ STM32Cube ☐ Empty

? < Back Next > Finish Cancel

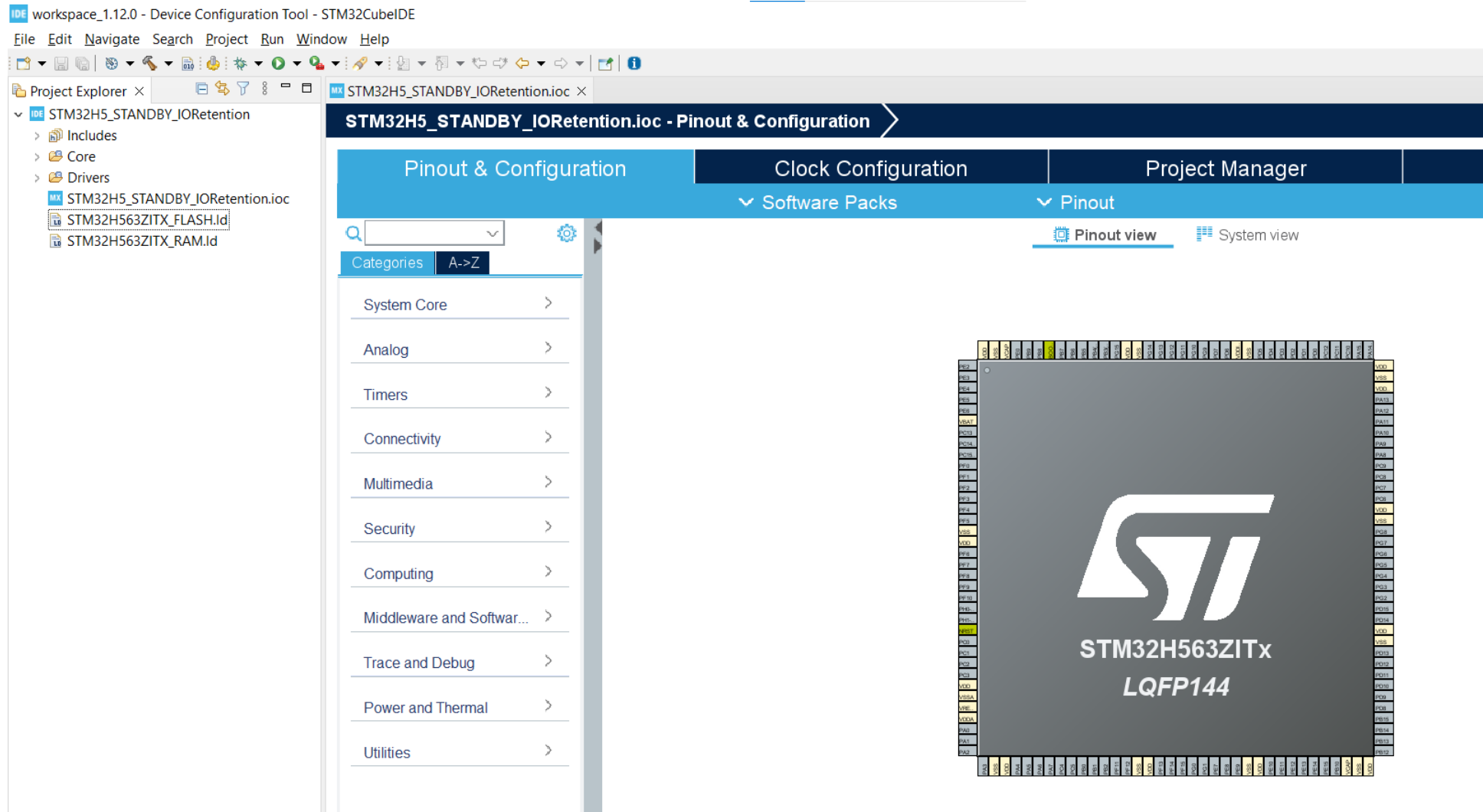
Ignore the warning

- The warning just tells us that it's highly recommended to enable the ICACHE, in order to reach the best performance
- ICACHE introduced on STM32 MCUs not so long ago
- We will do it, although it's not essential for this hands-on exercise (we'll do it just to get rid of this warning)
- **Click Yes**



STM32 project successfully generated

- We have an empty project with basic project structure generated ✓



**Now, we'll configure the MCU using
STM32CubeMX
(built-in version inside the STM32CubeIDE)**

ICACHE configuration in Pinout & Configuration

Pinout & Configuration

Clock Configuration

Project Manager

Tools

Software Packs

Pinout

Pinout view

System view

Categories

A->Z

Switch to A->Z tab

Memory address remap

Mode 1-way (direct mapped cache)

We activate ICACHE only to get rid of the warning!
ICACHE is not essential for the feature demonstrated
in this hands-on exercise!

Configuration

Reset Configuration

Parameter Settings

NVIC Settings

Warning: This peripheral has no parameters to be configured

ICACHE



STM32H563ZITx
LQFP144

USART3 configuration in Pinout & Configuration

The screenshot shows the 'Pinout & Configuration' tab in STM32CubeMX. On the left, a list of peripherals is shown, with 'USART3' highlighted by a red box and a red circle with the number '1'. In the center, the 'USART3 Mode and Configuration' window is open. The 'Mode' dropdown is set to 'Asynchronous' and is highlighted by a red box and a red circle with the number '2'. Below the mode, 'Hardware Flow Control (RS232)' and 'Slave Select(NSS) Management' are both set to 'Disable'. At the bottom, the 'Configuration' section shows 'Parameter Settings' selected, with 'Baud Rate' set to '115200 Bits/s', 'Word Length' to '8 Bits (including Parity)', 'Parity' to 'None', and 'Stop Bits' to '1'.

Pinout & Configuration

Clock Configuration

Software Packs

USART3 Mode and Configuration

Categories A->Z

Mode

Mode Asynchronous

Hardware Flow Control (RS232) Disable

Slave Select(NSS) Management Disable

115200 8N1 (default)

Configuration

Reset Configuration

NVIC Settings DMA Settings GPIO Settings

Parameter Settings User Constants

Configure the below parameters :

Search (Ctrl+F)

Basic Parameters

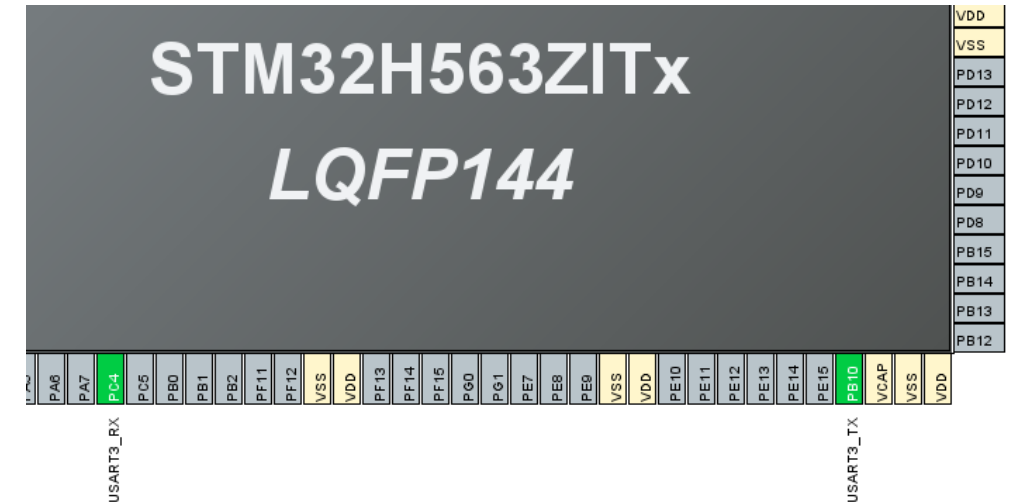
Baud Rate 115200 Bits/s

Word Length 8 Bits (including Parity)

Parity None

Stop Bits 1

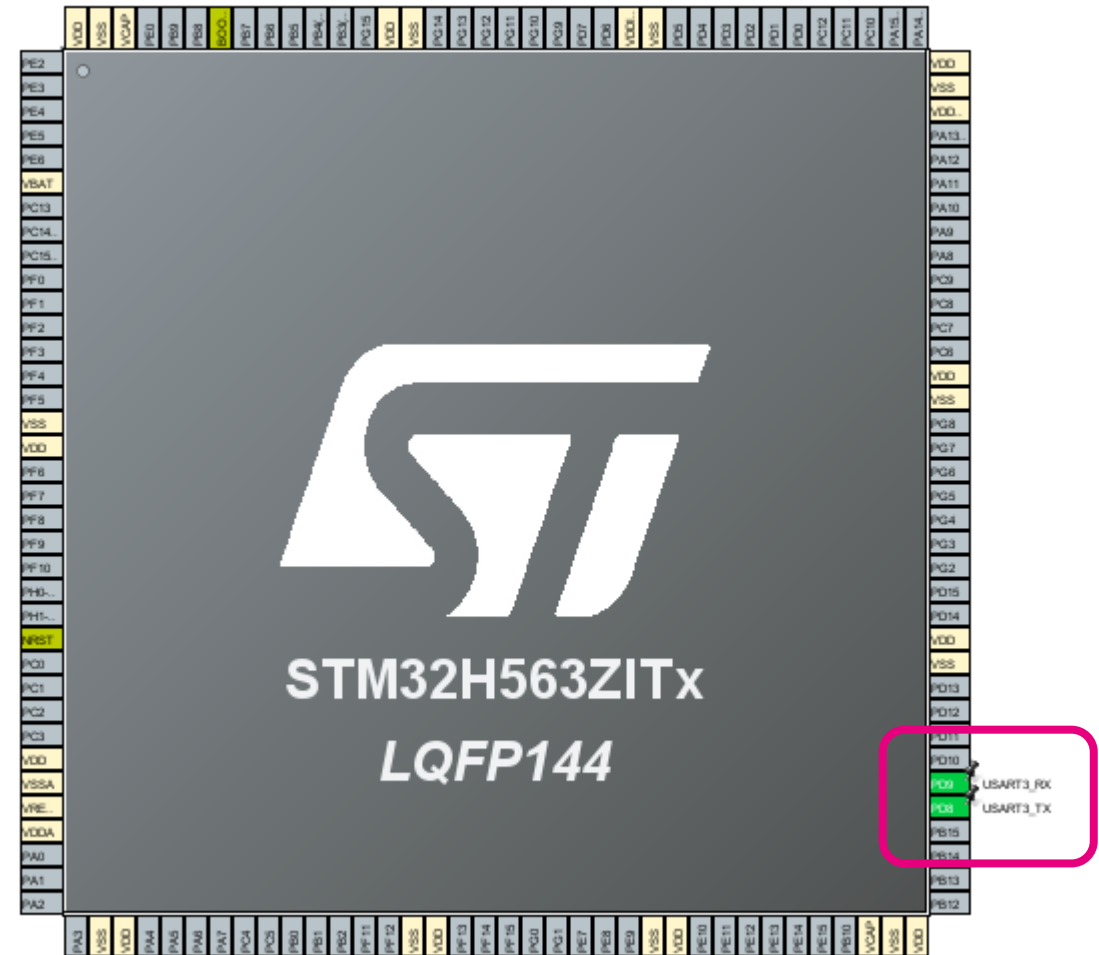
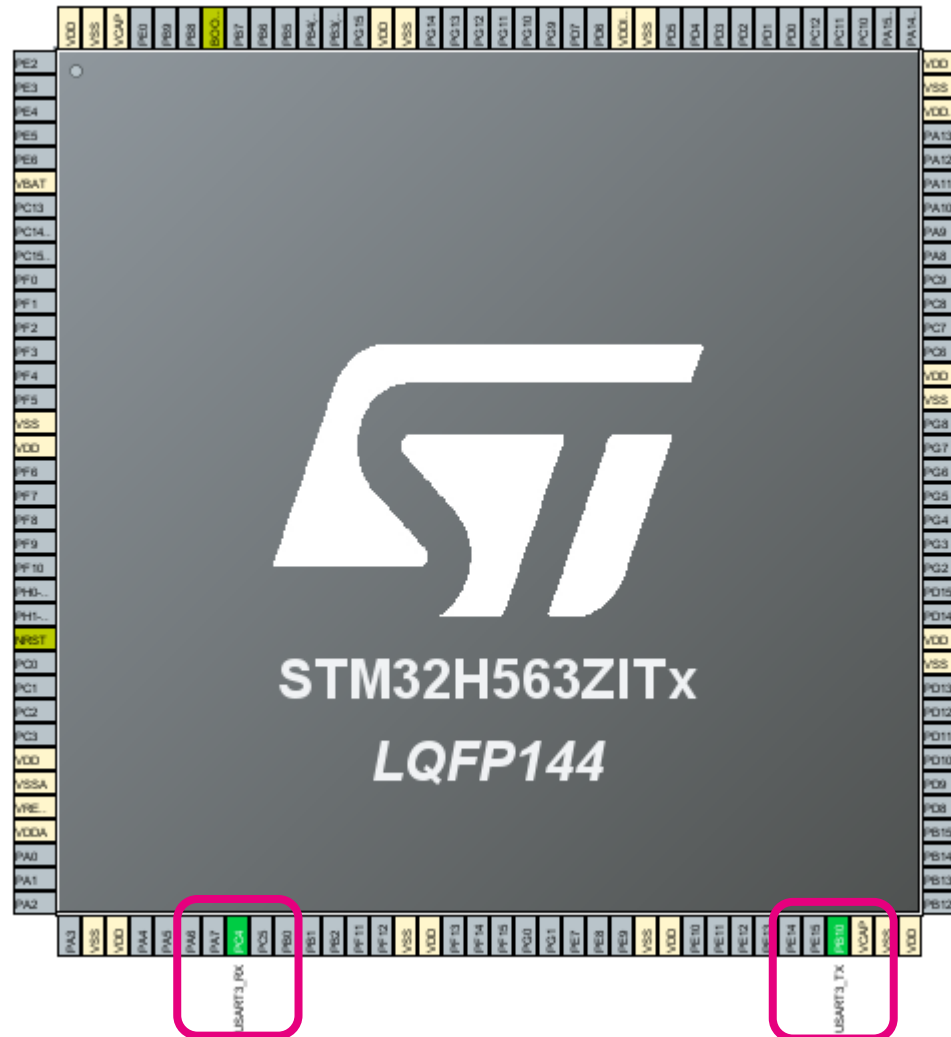
Advanced Parameters



USART3 signals automatically assigned to pins

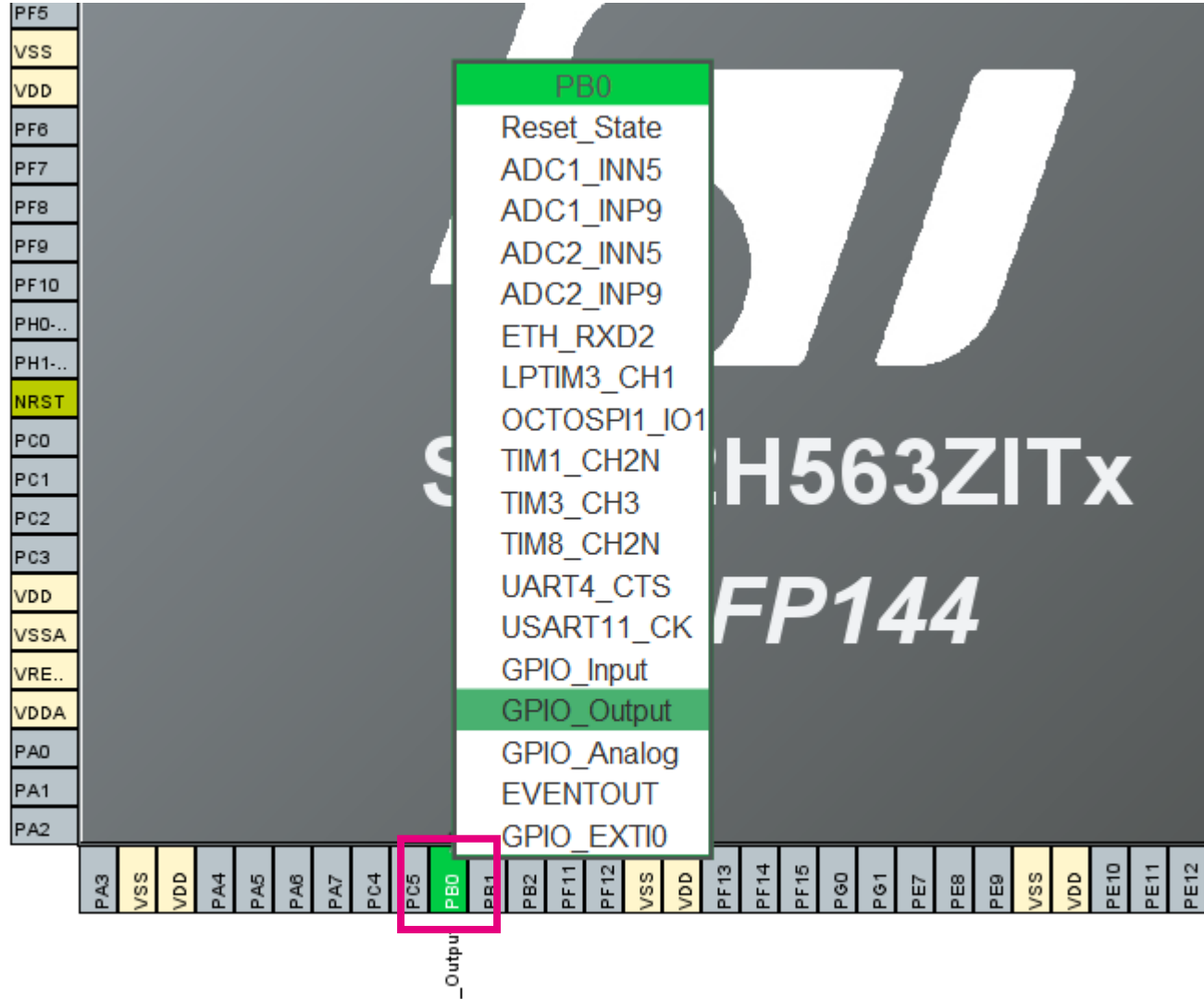
STM32CubeMX USART3 pins remap

Remap USART3 signals to PD8/PD9 using CTRL + left-click and drag&drop



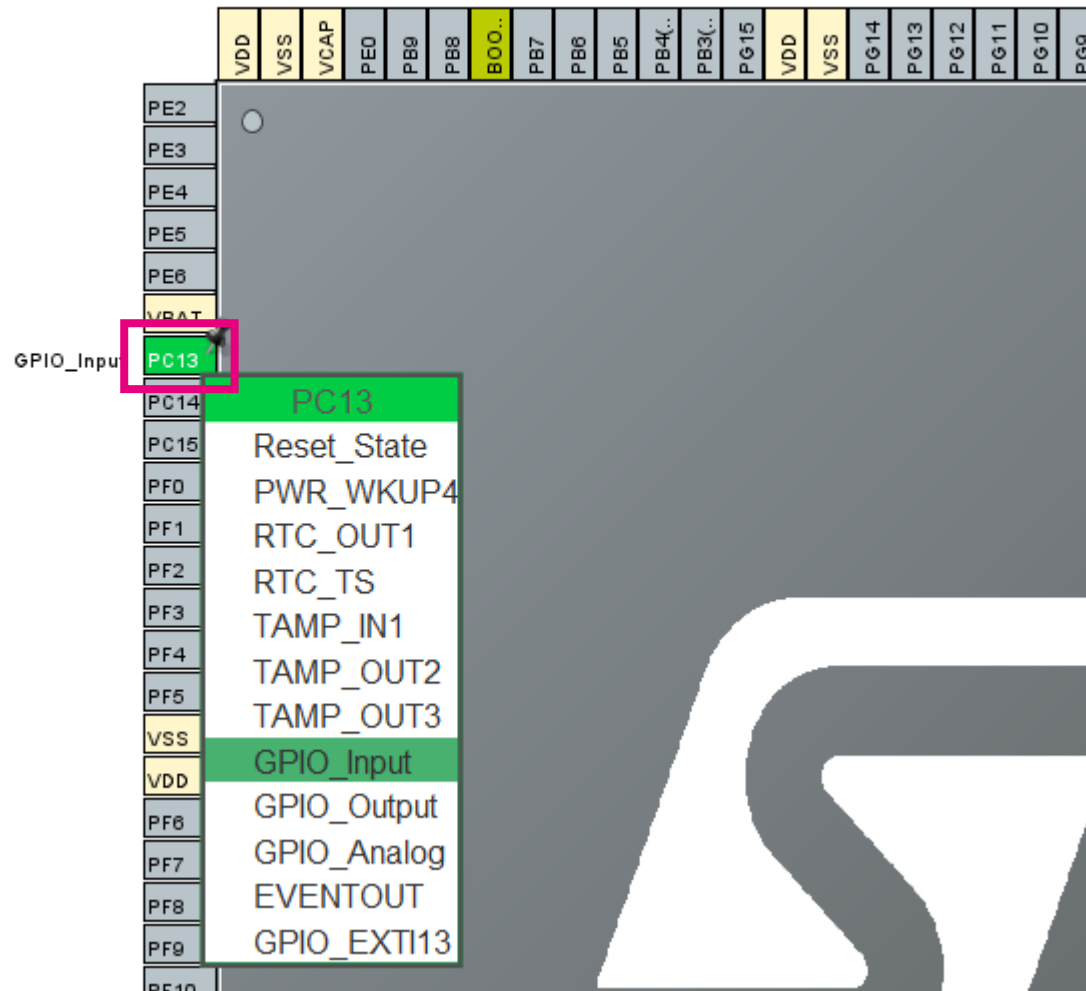
Configure PB0 (pin 46) as GPIO_Output

- PB0 is connected to a green LED (LD1)



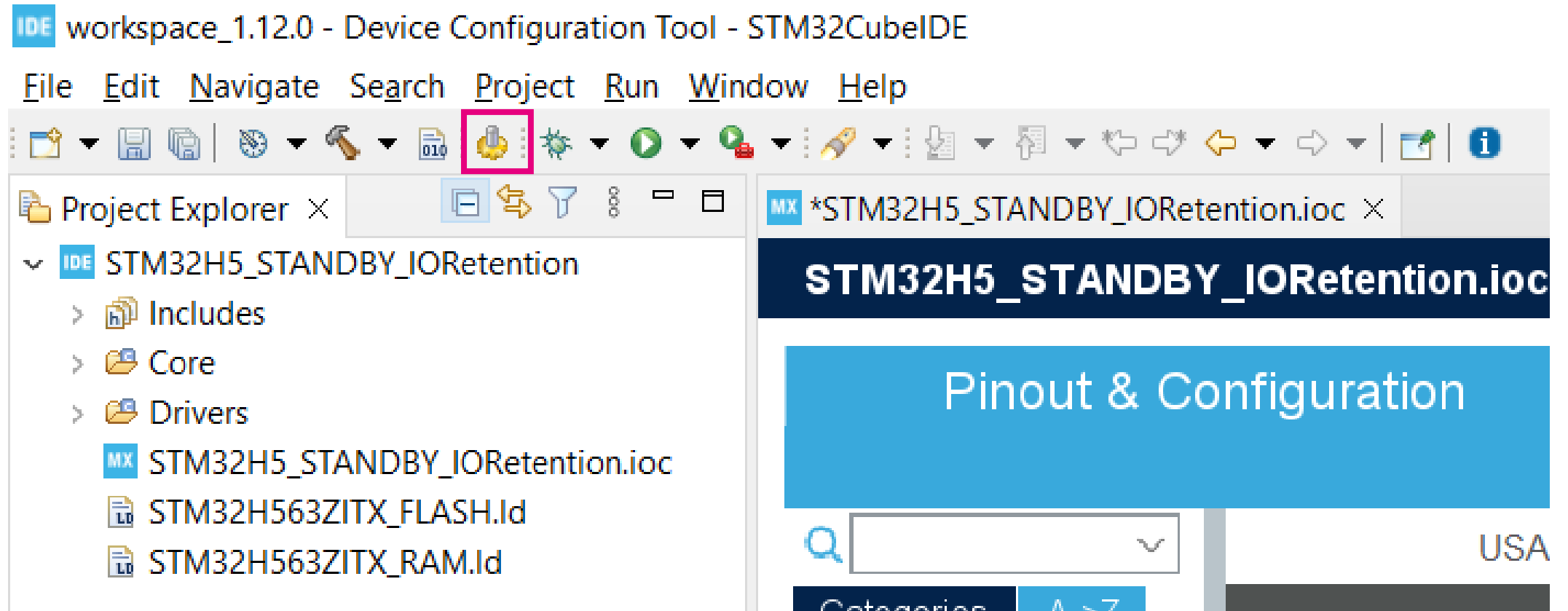
Configure PC13 (pin 7) as GPIO_Input

- PC13 is connected to a user button (blue button)

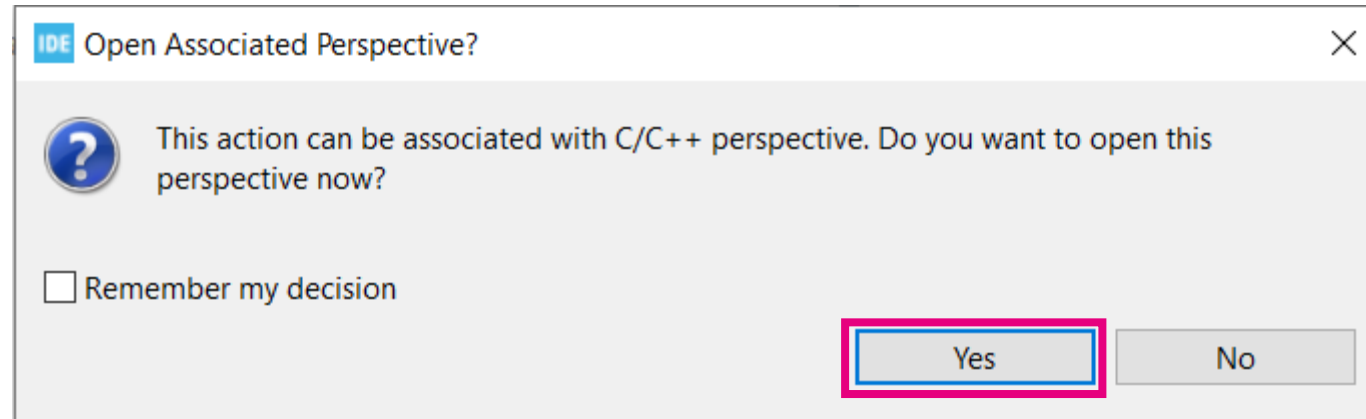


Generate code

- Click on the symbol below or press CTRL+S (save) to generate code

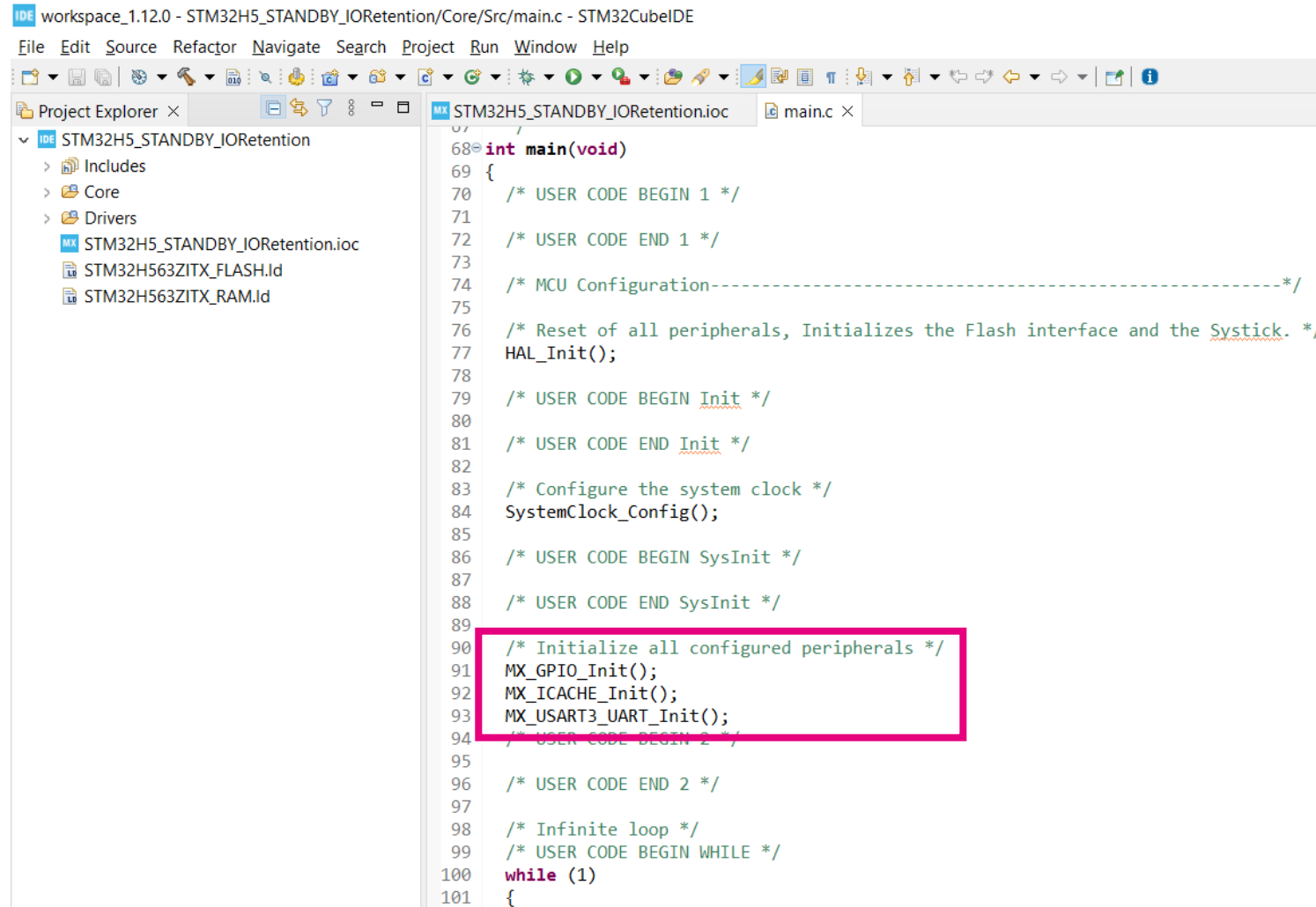


Open Associated Perspective



Code generated

- (Initialization) code has been generated according to our configuration in STM32CubeMX ✓



```
IDE workspace_1.12.0 - STM32H5_STANDBY_IORetention/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer x STM32H5_STANDBY_IORetention.ioc main.c x
STM32H5_STANDBY_IORetention
├── Includes
├── Core
├── Drivers
│   ├── STM32H5_STANDBY_IORetention.ioc
│   ├── STM32H563ZITX_FLASH.Id
│   └── STM32H563ZITX_RAM.Id
└── STM32H5_STANDBY_IORetention.ioc

68 int main(void)
69 {
70     /* USER CODE BEGIN 1 */
71
72     /* USER CODE END 1 */
73
74     /* MCU Configuration-----*/
75
76     /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
77     HAL_Init();
78
79     /* USER CODE BEGIN Init */
80
81     /* USER CODE END Init */
82
83     /* Configure the system clock */
84     SystemClock_Config();
85
86     /* USER CODE BEGIN SysInit */
87
88     /* USER CODE END SysInit */
89
90     /* Initialize all configured peripherals */
91     MX_GPIO_Init();
92     MX_ICACHE_Init();
93     MX_USART3_UART_Init();
94     /* USER CODE BEGIN 2 */
95
96     /* USER CODE END 2 */
97
98     /* Infinite loop */
99     /* USER CODE BEGIN WHILE */
100    while (1)
101    {
```

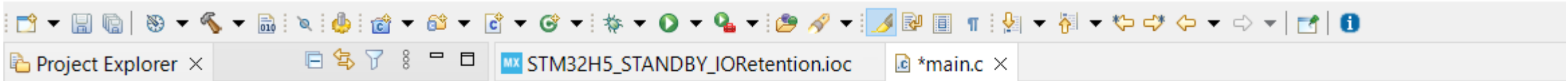
**Now, we'll fulfill the generated code with
our application code**

Open *main.c*, go to line ~24

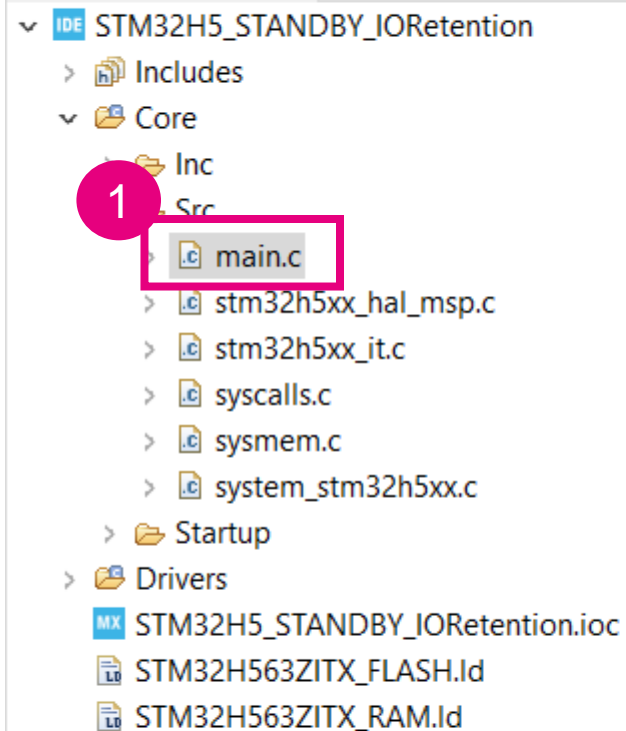
Extend the code

workspace_1.12.0 - STM32H5_STANDBY_IORetention/Core/Src/main.c - STM32CubeIDE

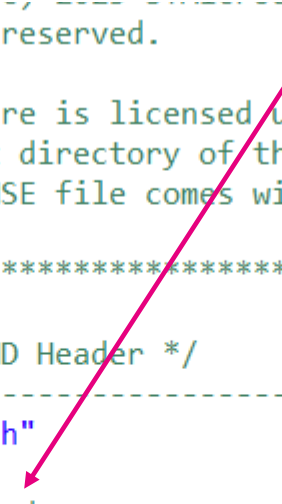
File Edit Source Refactor Navigate Search Project Run Window Help



Open *STM32H5_IO_State_Retention_code.txt*
and copy-paste the **USER CODE Includes** section



```
10  * All rights reserved.
11  *
12  * This software is licensed un
13  * in the root directory of thi
14  * If no LICENSE file comes wit
15  *
16  *****
17  */
18  /* USER CODE END Header */
19  /* Includes -----
20  #include "main.h"
21
22  /* Private includes -----
23  /* USER CODE BEGIN Includes */
24  #include <stdio.h>
25  /* USER CODE END Includes */
26
```

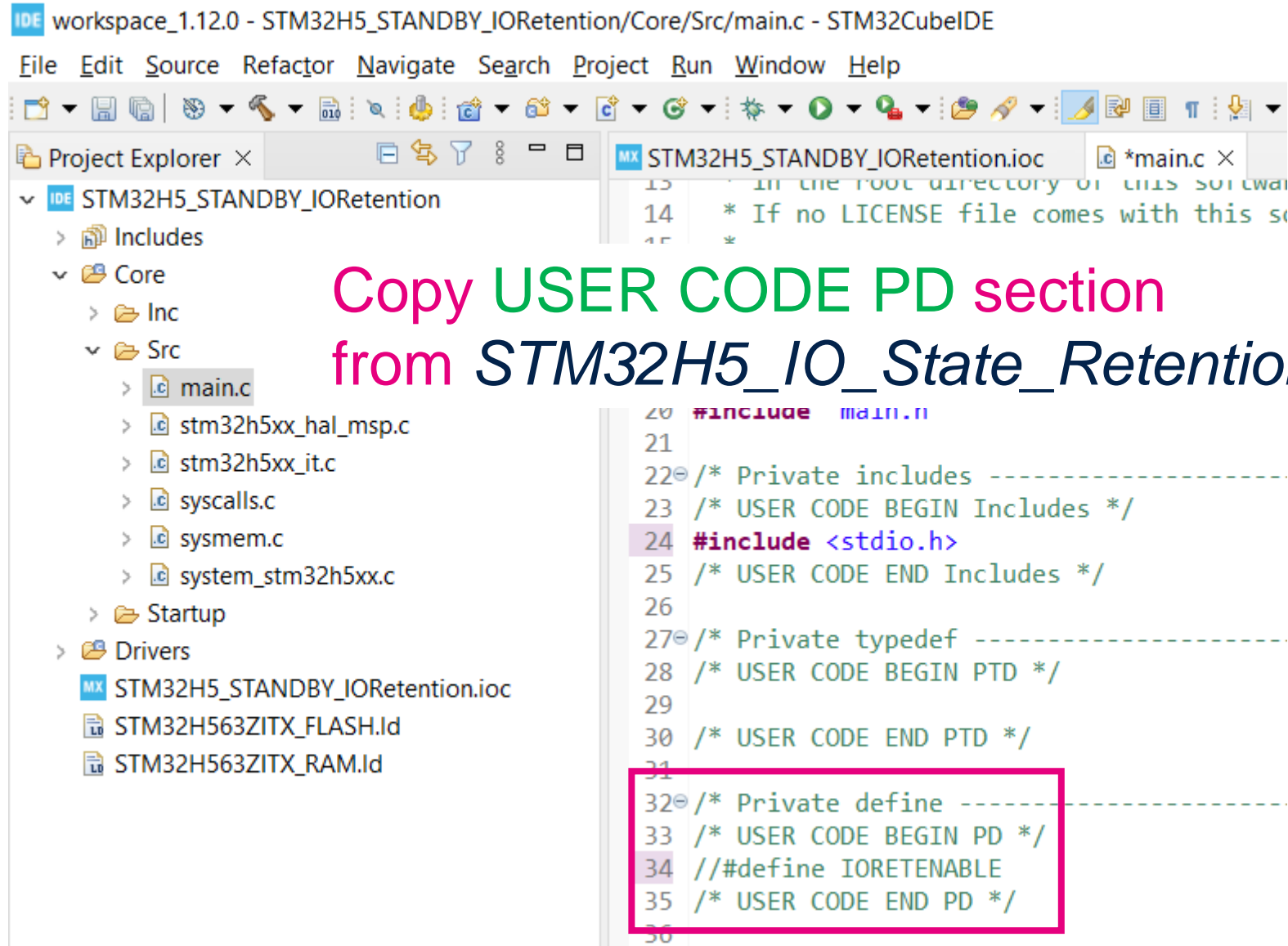


```
STM32H5_IO_State_Retention_code.txt - Notepad
File Edit Format View Help
/* USER CODE BEGIN Includes */
#include <stdio.h>
/* USER CODE END Includes */

/* USER CODE BEGIN PD */
#define IORETENABLE
/* USER CODE END PD */

/* USER CODE BEGIN PFP */
int __io_putchar(int ch)
{
    HAL_UART_Transmit(&huart3, (uint8_t *)&ch, 1, 0xFFFF);
    return ch;
}
/* USER CODE END PFP */

/* USER CODE BEGIN 2 */
printf("\n\r ***** IO RETENTION EXAMPLE *****\n\r");
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
printf(" LD1 turned on \r\n");
if(READ_BIT(PWR->IORETR, PWR IORETR IORETEN) == PWR IORETR IORETEN)
```

Copy USER CODE PD section
from STM32H5_IO_State_Retention_code.txt

workspace_1.12.0 - STM32H5_STANDBY_IORetention/Core/Src/main.c - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer ×

- IDE STM32H5_STANDBY_IORetention
 - > Includes
 - > Core
 - > Inc
 - > Src
 - main.c
 - stm32h5xx_hal_msp.c
 - stm32h5xx_it.c
 - syscalls.c
 - systemem.c
 - system_stm32h5xx.c
 - > Startup
 - > Drivers
 - STM32H5_STANDBY_IORetention.ioc
 - STM32H563ZITX_FLASH.Id
 - STM32H563ZITX_RAM.Id

STM32H5_STANDBY_IORetention.ioc

```
45
46 /* USER CODE BEGIN PV */

52 static void MX_GPIO_Init(void);
53 static void MX_ICACHE_Init(void);
54 static void MX_USART3_UART_Init(void);
55
56 /* USER CODE BEGIN PFP */
57 int __io_putchar(int ch)
58 {
59     HAL_UART_Transmit(&huart3, (uint8_t *)&ch, 1, 0xFFFF);
60     return ch;
61 }
62 /* USER CODE END PFP */

63
64 /* Private user code -----
65 /* USER CODE BEGIN 0 */
```

Copy **USER CODE PFP** section
from *STM32H5_IO_State_Retention_code.txt*

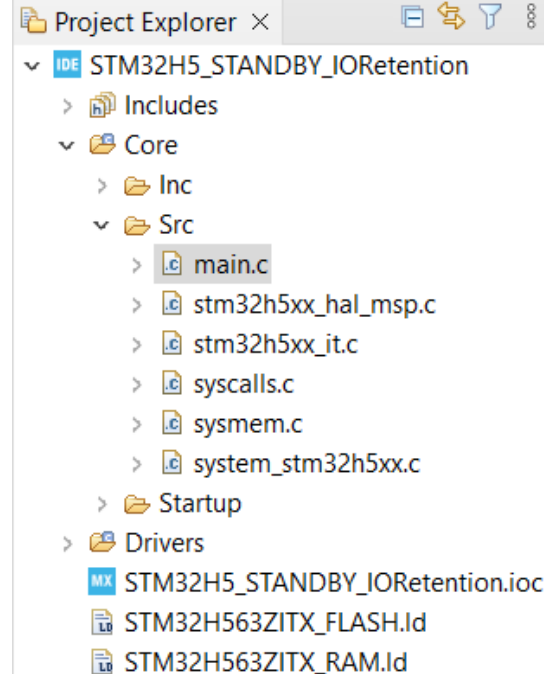
main.c, line ~100

workspace 1.12.0 - STM32H5_STANDBY_IORetention/Core/Src/main.c - STM32CubeIDE

File Refactor Navigate Search Project Run Window Help

Extend the code

Copy USER CODE 2 section from STM32H5_IO_State_Retention_code.txt



```
99 /* USER CODE BEGIN 2 */
100 printf("\n\r ***** IO RETENTION EXAMPLE *****\n\r");
101 HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
102 printf( "LD1 turned on          \r\n");
103 if(READ_BIT(PWR->IORETR, PWR_IORETR_IORETEN)== PWR_IORETR_IORETEN)
104 {
105     printf( "IO retention bit set          \r\n");
106     HAL_Delay(1000);
107     HAL_PWREx_DisableStandbyIORetention();
108     printf( "Disabling IO retention          \r\n");
109     HAL_Delay(10);
110 }
111 printf( " *****\r\n");
112 printf( " ***** Push user button to enter STANDBY *****\r\n");
113 printf( " *****\r\n");
114 while(HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13)== GPIO_PIN_RESET)
115 {
116     HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_0);
117     HAL_Delay(100);
118 }
119 printf( "MCU entering STANDBY: use reset button to wake-up \r\n");
120 HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
121 #if defined(IORETENABLE)
122 printf("Enabling IO retention \r\n");
123 HAL_PWREx_EnableStandbyIORetention();
124 #endif
125
126 printf( " \r\n");
127 HAL_Delay(1000);
128 HAL_PWR_EnterSTANDBYMode();
129 /* USER CODE END 2 */
```

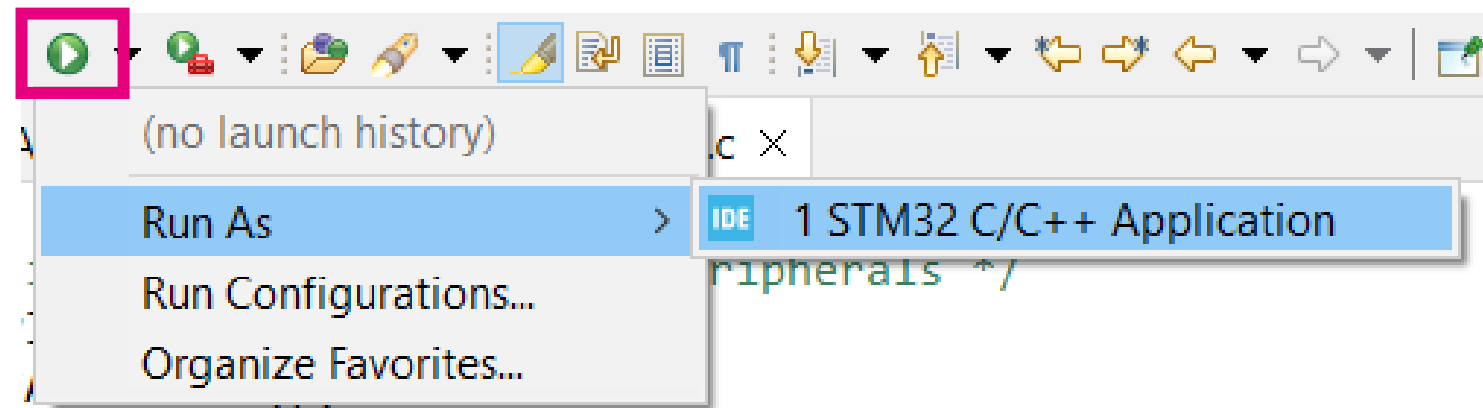
**At first we'll observe the behavior
without IO state retention enabled**

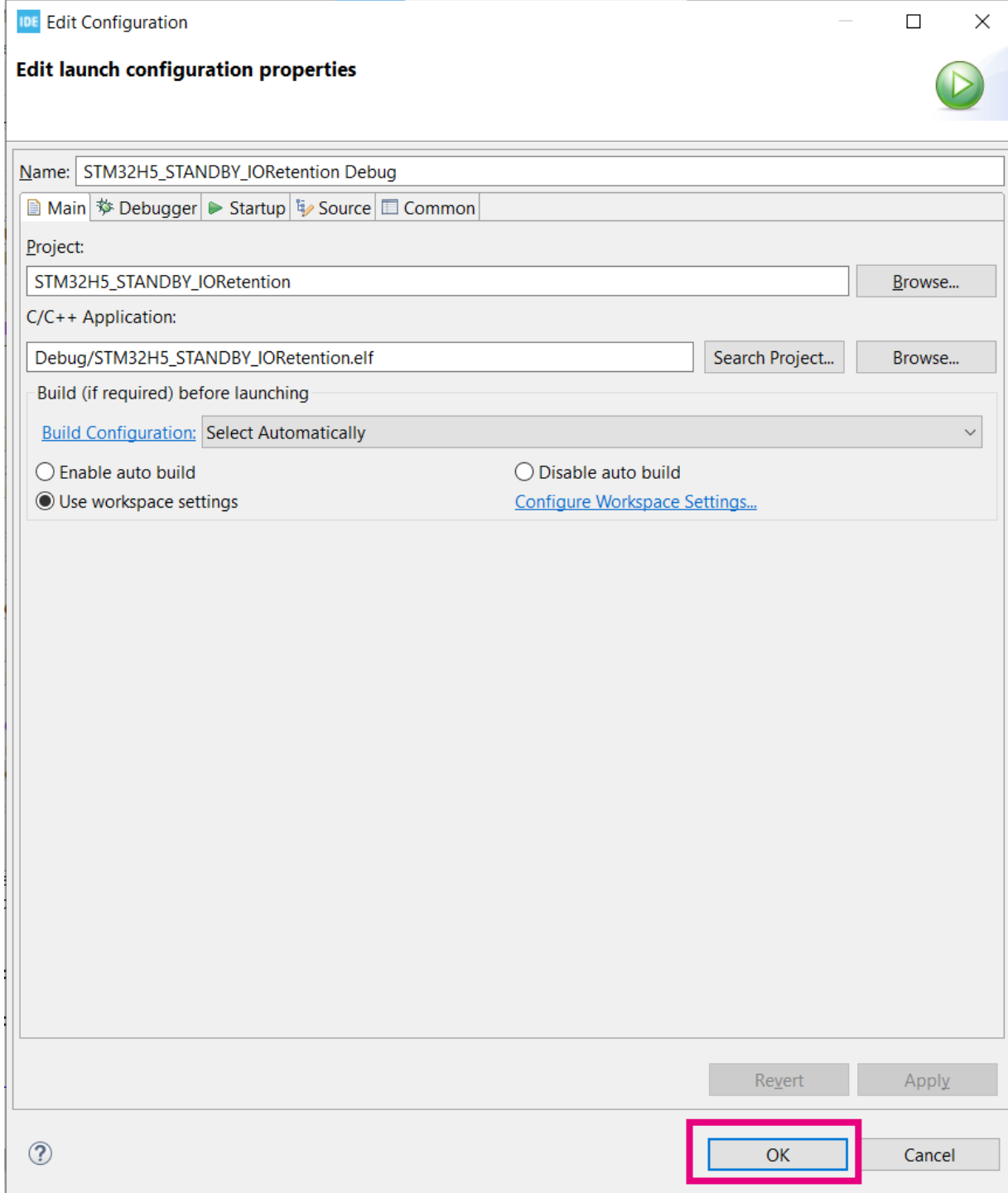
Build and flash the project

1) Build



2) Run As -> 1 STM32 C/C++ Application





Edit Configuration

Just click OK

Configure serial port connection

- Open Console -> Command Shell Console

The screenshot shows an IDE interface with the following components:

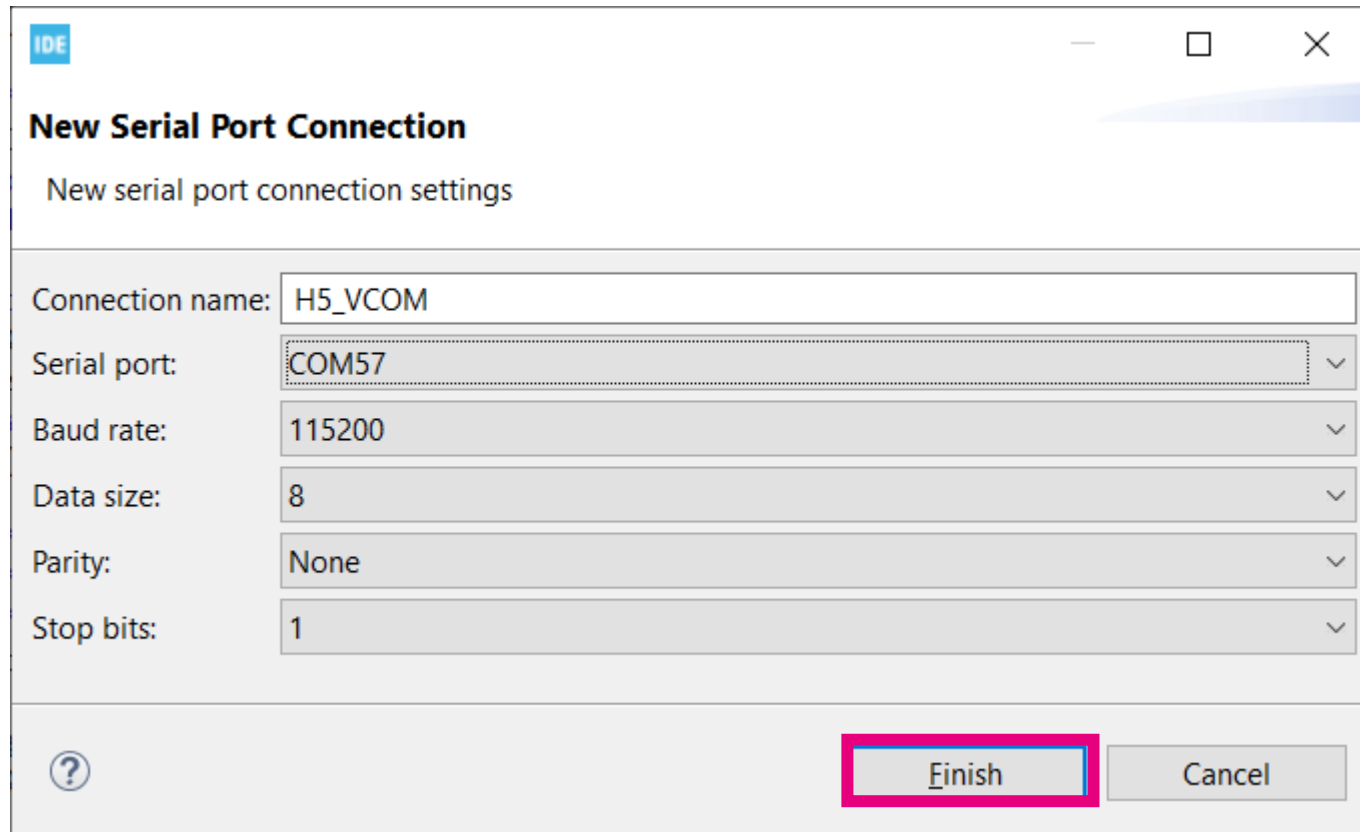
- Console Window:** Displays the output of a debug session. The text includes: `<terminated> STM32H5_STANDBY_IORetention Debug [STM32 C/C++ Application] ST-LINK (ST-LINK GDB server) (Terminated May 11, 2023, Verifying ...` and `Download verified successfully`. Below this, it says `Shutting down... Exit.`
- Select Remote Connection Dialog:** A dialog box with the title "Select Remote Connection". It contains three fields:
 - Connection Type: Serial Port
 - Connection name: NUCLEO-H563ZI
 - Encoding: windows-1252At the bottom are "OK" and "Cancel" buttons.
- New Console View Menu:** A menu with four options:
 - 1 New Console View
 - 2 C/C++ Build Console
 - 3 Command Shell Console
 - 4 Device Configuration Tool Console

Numbered callouts (1-4) highlight the following steps:

- 1: Click on the "Console" tab in the IDE.
- 2: Click on the "New Console View" menu.
- 3: Click on the "Command Shell Console" option.
- 4: Click on the "New..." button in the "Select Remote Connection" dialog.

Configure serial port connection

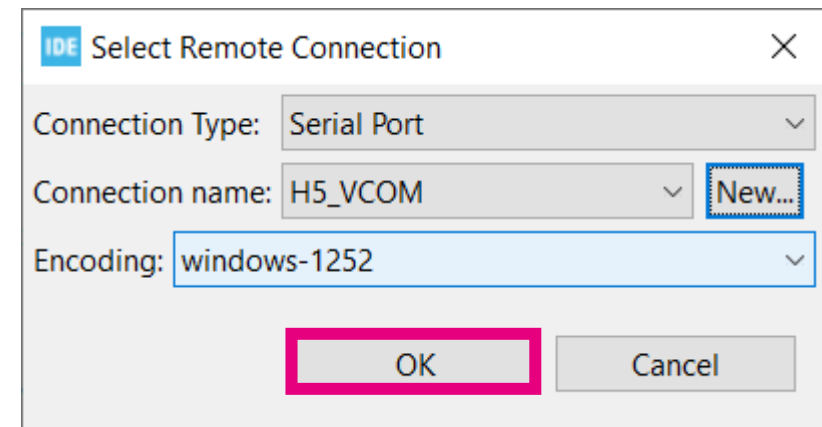
- Configure the connection as shown below, click on „Finish“ and then „OK“



The 'New Serial Port Connection' dialog box is shown. It has a title bar with 'IDE' and standard window controls. The subtitle is 'New serial port connection settings'. The fields are as follows:

Field	Value
Connection name:	H5_VCOM
Serial port:	COM57
Baud rate:	115200
Data size:	8
Parity:	None
Stop bits:	1

At the bottom, there is a question mark icon, a 'Finish' button (highlighted with a red rectangle), and a 'Cancel' button.



The 'Select Remote Connection' dialog box is shown. It has a title bar with 'IDE' and a close button. The fields are as follows:

Field	Value
Connection Type:	Serial Port
Connection name:	H5_VCOM
Encoding:	windows-1252

At the bottom, there is a 'New...' button next to the connection name field, and 'OK' and 'Cancel' buttons. The 'OK' button is highlighted with a red rectangle.

Verify if the serial port is working properly

Press the black reset button,
you should see traces in the terminal



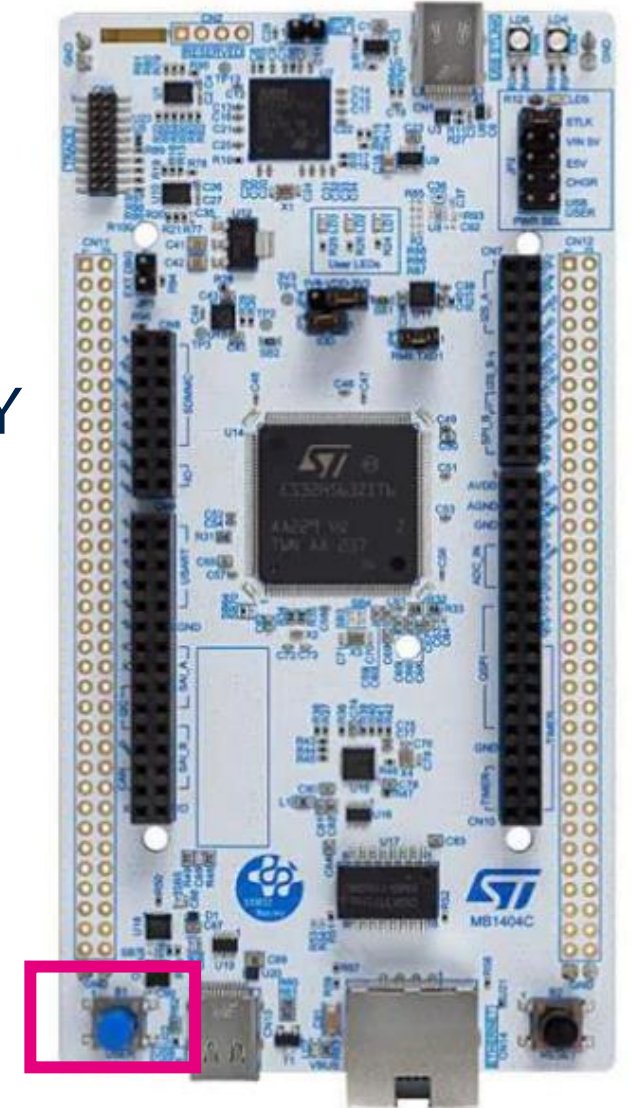
```
Problems Tasks Console × Properties
H5_VCOM (CONNECTED)

***** IO RETENTION EXAMPLE *****
LD1 turned on
*****
***** Push user button to enter STANDBY *****
*****
```

Observe the behavior w/o IO state retention in STANDBY enabled

- Observe the behavior:
After reset the MCU boots and waits for a user button press
- While waiting green LED is toggling
(indicating we're in the main loop)
- Once user button (blue) pressed, the MCU enters STANDBY
LD6 is OFF (pin is in High-Z, thus not driven),
**even though we set the HIGH state
shortly before entering the STANDBY mode**
(see next slide and the section USER CODE 2)
- Observe traces in terminal

User button



Observe the behavior w/o IO state retention in STANDBY enabled

The green LED (LD1) should be ON

```
printf( "MCU entering STANDBY: use reset button to wake-up \r\n");
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
#ifdef IORETENABLE
printf("Enabling IO retention \r\n");
HAL_PWREx_EnableStandbyIORetention();
#endif

printf( " \r\n");
HAL_Delay(1000);
HAL_PWR_EnterSTANDBYMode();
/* USER CODE END 2 */
```

When the MCU enters the STANDBY mode, all I/Os are floating (High-Z), thus not driven, so the LED goes out

**Let's enable the IO state retention in
STANDBY**

Uncomment #define IORETENABLE

workspace_1.12.0 - STM32H5_STANDBY_IORetention/Core/Src/main.c - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer ×

- IDE STM32H5_STANDBY_IORetention
 - > Binaries
 - > Includes
 - > Core
 - > Inc
 - > Src
 - main.c
 - stm32h5xx_hal_msp.c
 - stm32h5xx_it.c
 - syscalls.c
 - systemem.c
 - system_stm32h5xx.c
 - > Startup
 - > Drivers
 - > Debug
 - MX STM32H5_STANDBY_IORetention.ioc
 - STM32H5_STANDBY_IORetention Debug.laur
 - STM32H563ZITX_FLASH.Id
 - STM32H563ZITX_RAM.Id

STM32H5_STANDBY_IORetention.ioc *main.c

```
11
12  * This software is licensed under
13  * in the root directory of this sc
14  * If no LICENSE file comes with th
15  *
16  ****

20  #include "main.h"
21
22  /* Private includes -----
23  /* USER CODE BEGIN Includes */
24  #include <stdio.h>
25  /* USER CODE END Includes */
26
27  /* Private typedef -----
28  /* USER CODE BEGIN PTD */
29
30  /* USER CODE END PTD */
31
32  /* Private define -----
33  /* USER CODE BEGIN PD */
34  #define IORETENABLE
35  /* USER CODE END PD */
36
```

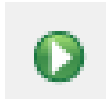
Uncomment #define IORETENABLE

Build and flash the project

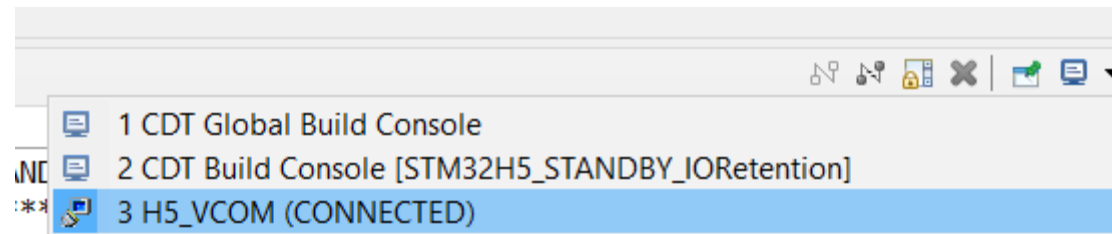
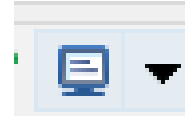
1) Build



2) Run



3) Display serial port connection console



4) Press the reset button



Observe the behavior w/ IO state retention in STANDBY enabled

The green LED (LD1) should be ON




```
printf( "MCU entering STANDBY: use reset button to wake-up \r\n");
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
#ifdef IORETENABLE
printf("Enabling IO retention \r\n");
HAL_PWREx_EnableStandbyIORetention();
#endif

printf( " \r\n");
HAL_Delay(1000);
HAL_PWR_EnterSTANDBYMode();
/* USER CODE END 2 */
```

IO state retention in STANDBY is enabled

Entering the STANDBY mode

Observe the behavior w/ IO state retention in STANDBY enabled

- Observe the behavior:
After reset the MCU boots and waits for a user button press
- While waiting green LED is toggling
(indicating we're in the main loop)
- Once user button is pressed, 
the MCU enters the STANDBY mode, LD6 is ON
(the logical level on the pin is maintained),
we set the HIGH state shortly before entering the STANDBY
- Observe traces in terminal
- Press reset button to repeat 
- The logical level is maintained even under reset
(press and hold the reset button to test that) 



User button

Reset button



- What have we learned?
 - How to use the I/Os state retention in STANDBY
 - How to use the STM32Cube software ecosystem (STM32CubeIDE)
 - We familiarized ourselves with the NUCLEO-H563ZI development board