life.augmented

STM32H5 I/Os output state retention in STANDBY

I/Os output state retention in STANDBY

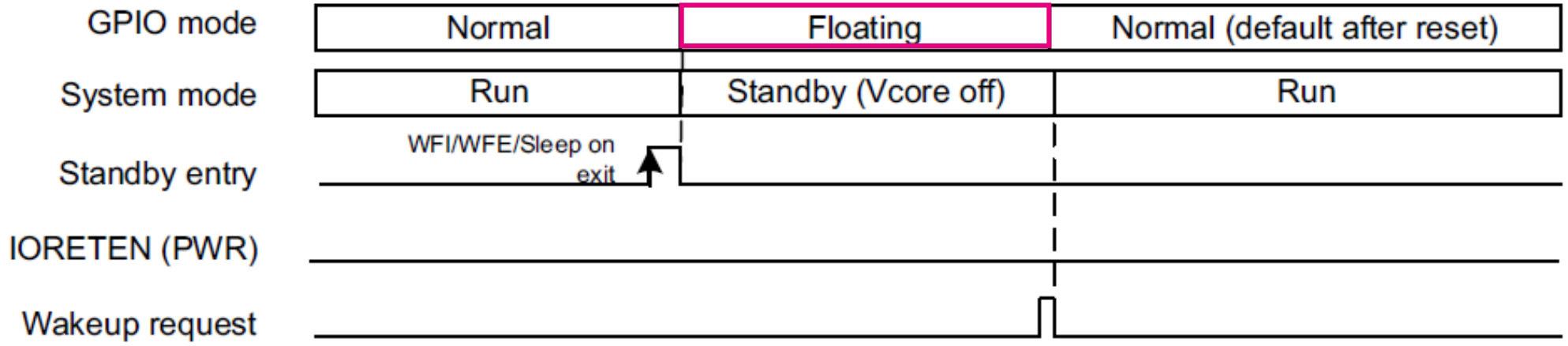
- In the STANDBY mode the I/Os are by default in floating (High-Z) state (not being driven to any defined logic level)
- This might be an issue for control signals for external devices such as chip select/slave select or low-power mode control signal, as we need to ensure the proper voltage level is 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

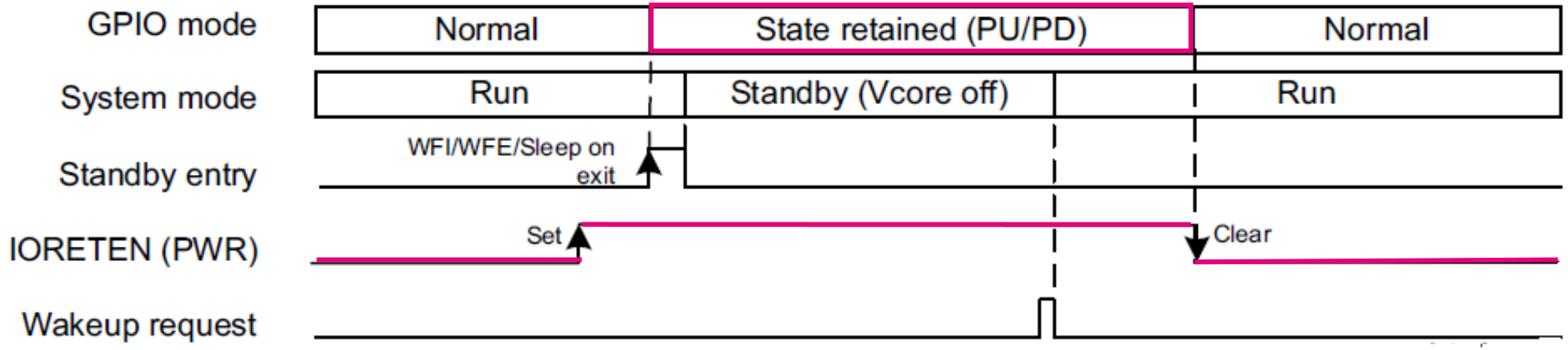
- The pull-up/pull-down resistors remain connected and maintain the voltage level until the I/O state retention is disabled by software
  (thus, they are still active even after wake-up from STANDBY in the RUN mode)
- They should be disabled after reset (= wake-up from STANDBY),
  as soon as peripherals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

life.augmented

Objective

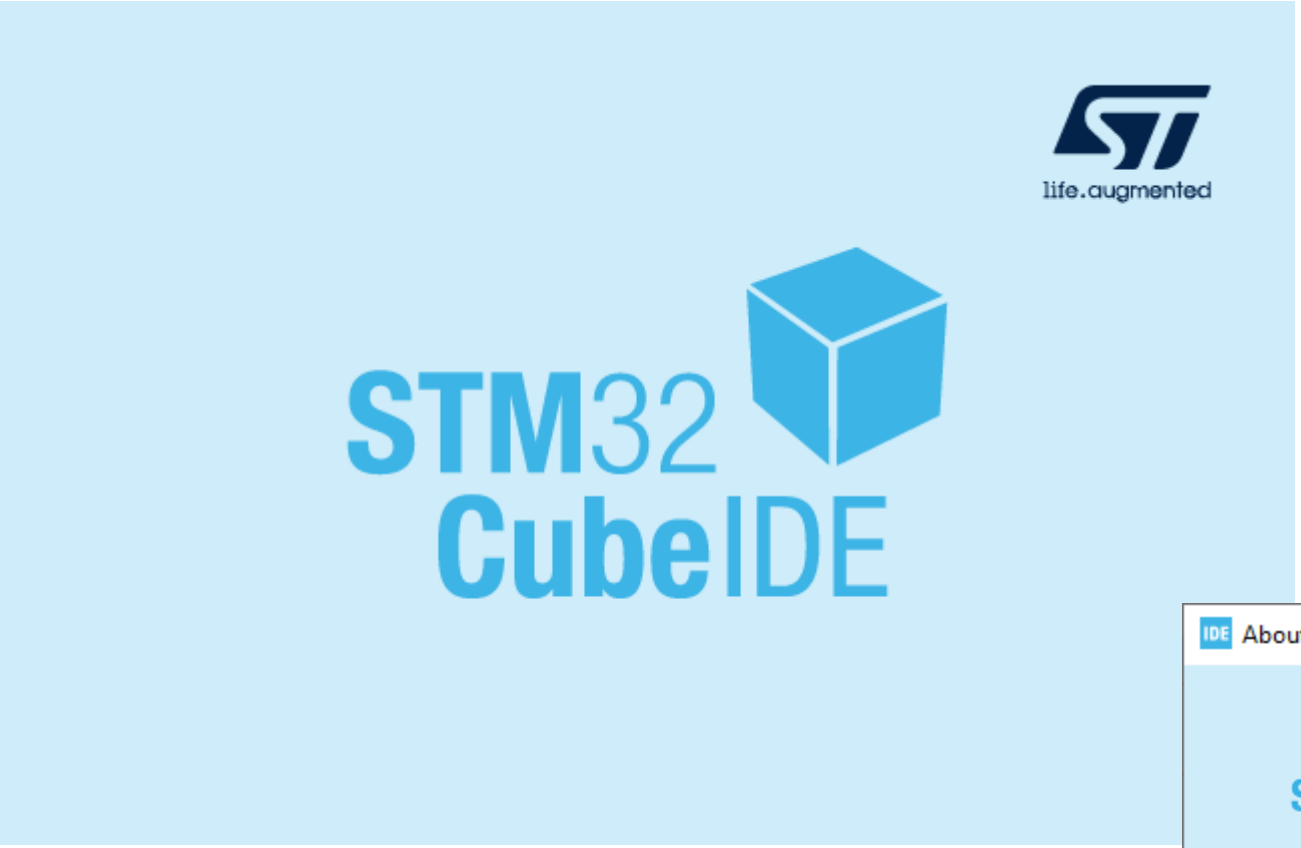
- The purpose of this LAB is to:
  - Get familiar with the **NUCLEO-H563ZI** development board
  - Observe the difference in behavior w/ and w/o I/Os state retention enabled
  - Use HAL peripheral drivers
    (part of STM32Cube software ecosystem for STM32 MCUs)
- Steps:
  - Configure MCU pinout and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

life.augmented

Let‘s get started

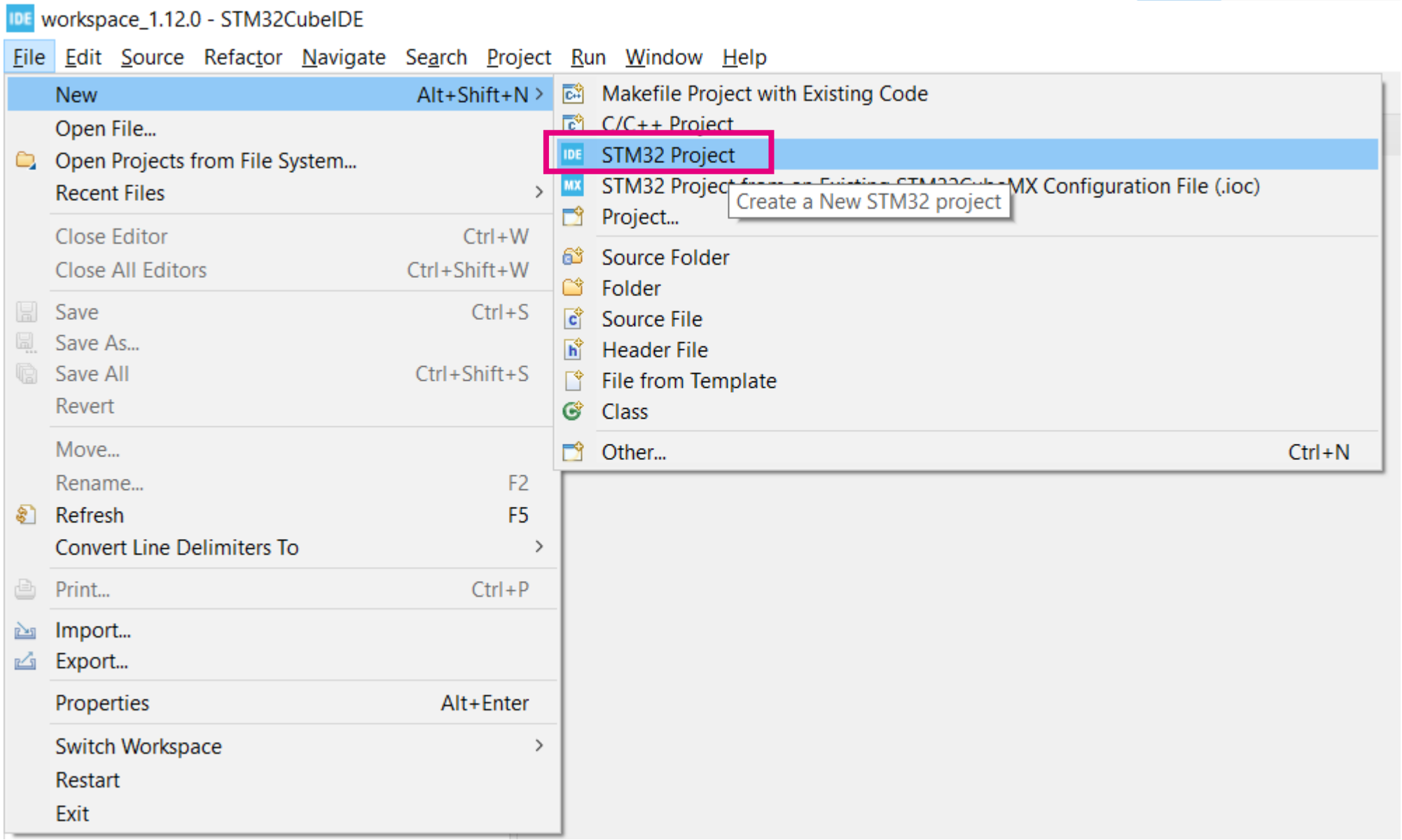
life.augmented

Open STM32CubeIDE



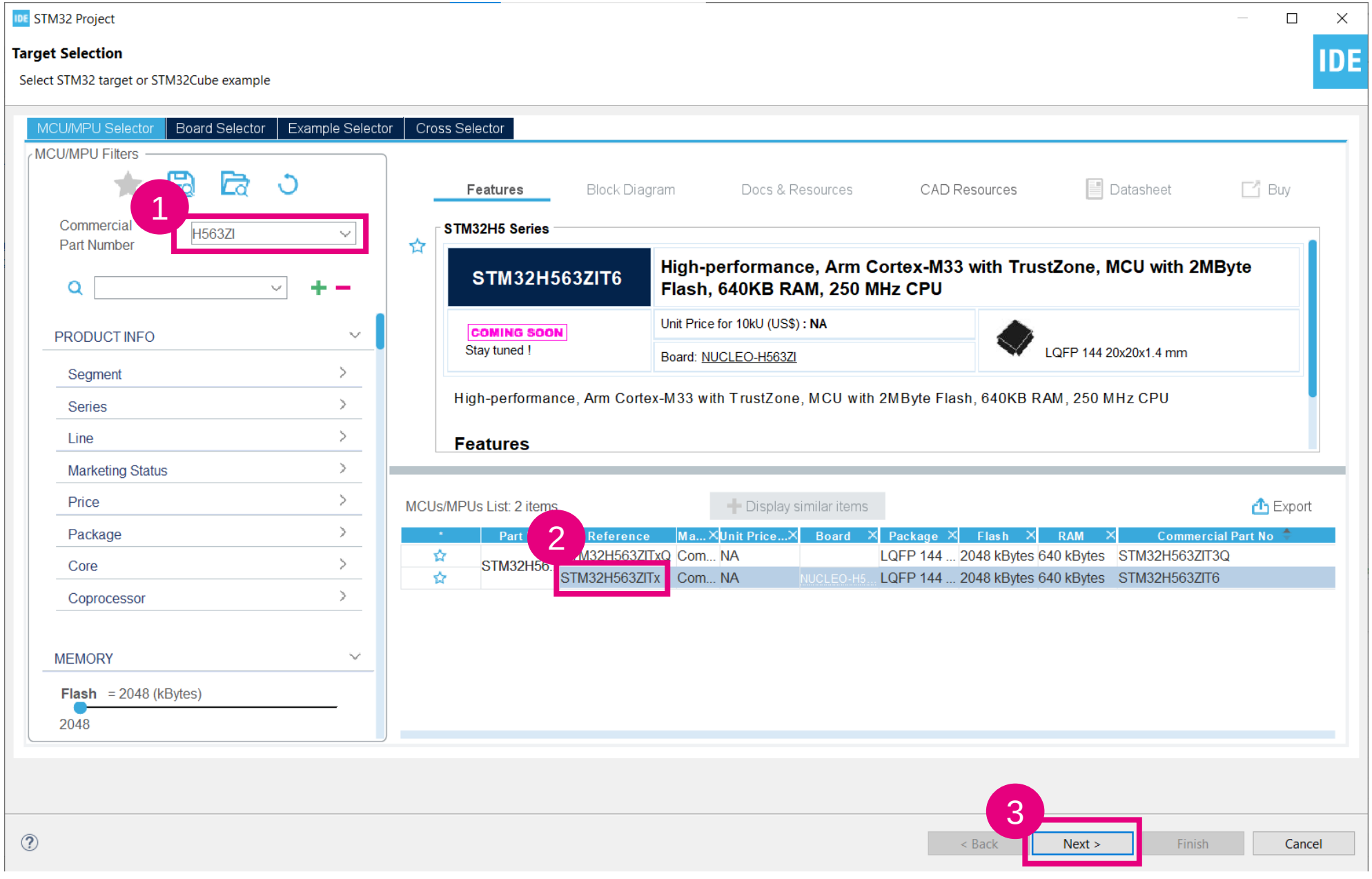
Start a new STM32 project

File -> New -> STM32 Project



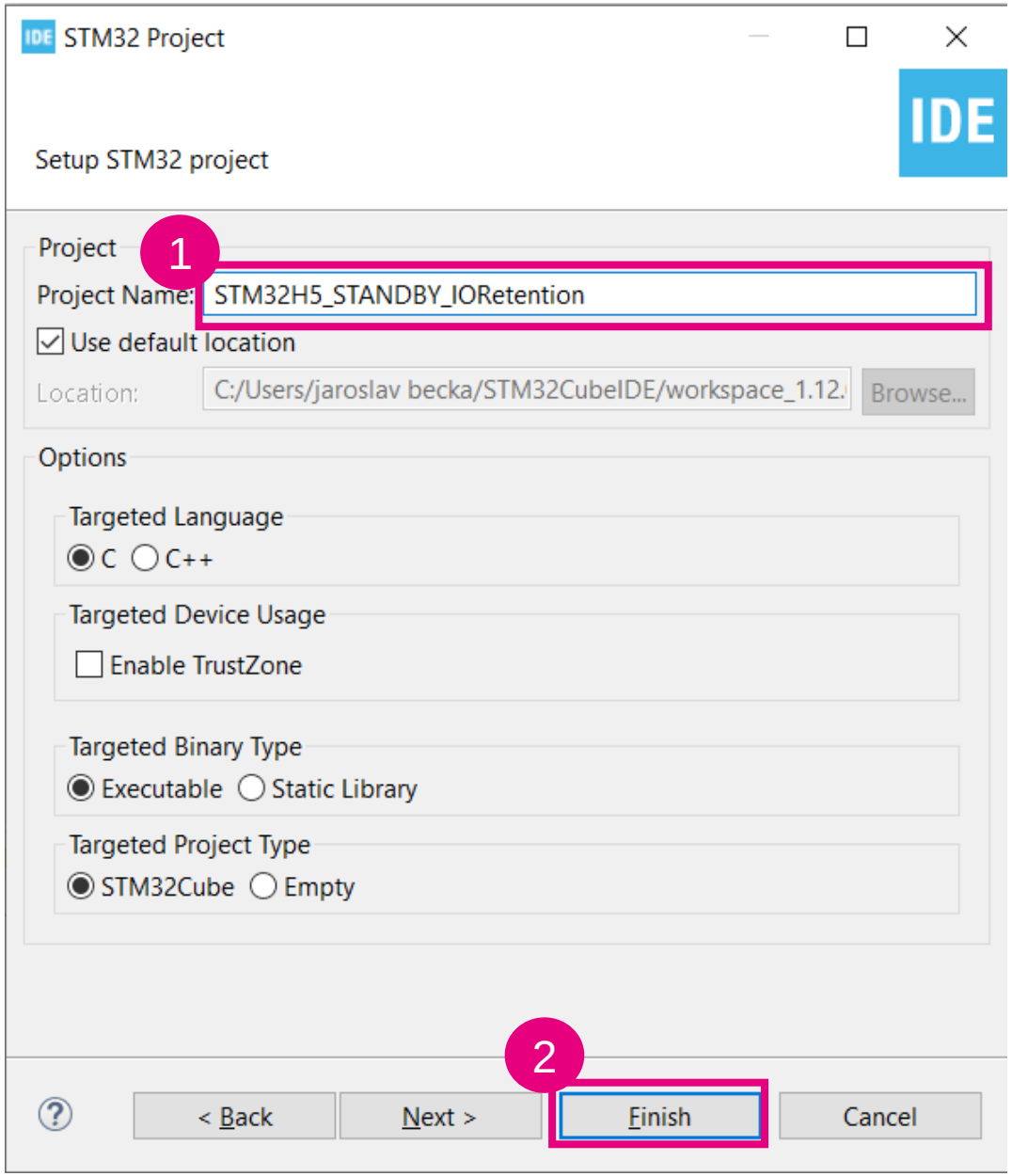
life.augmented

Select STM32H563ZITx in MCU/MPU Selector



life.augmented

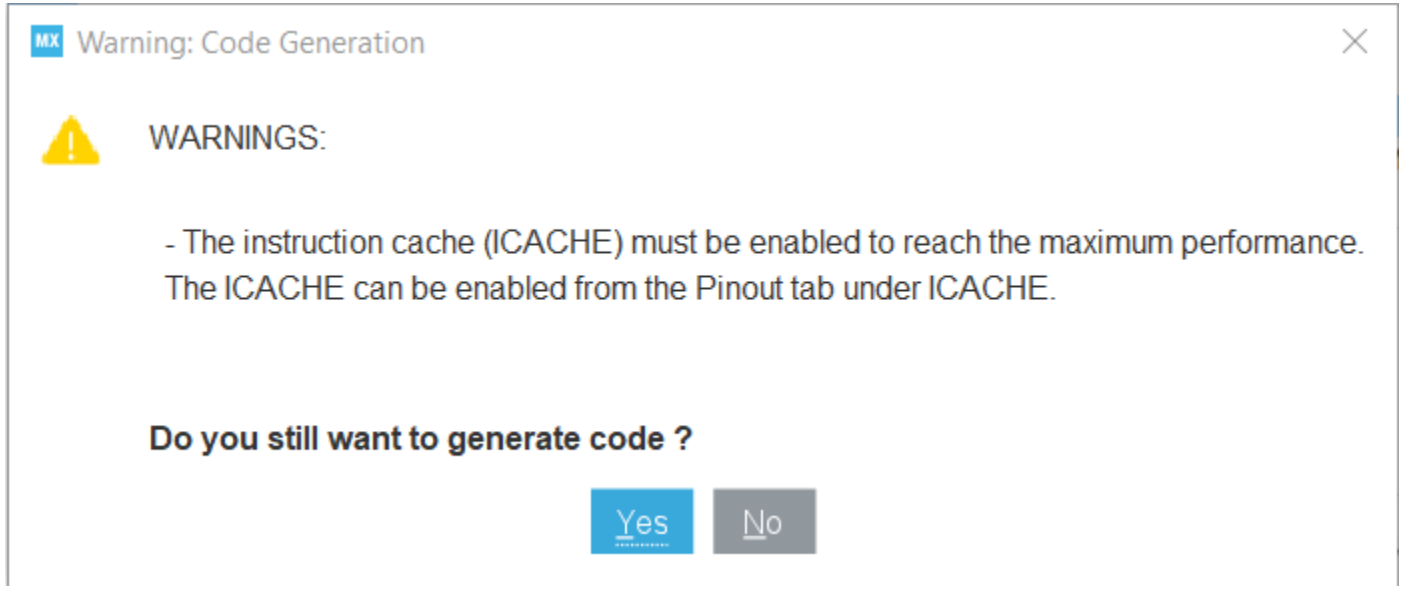
Choose project name and click on „Finish“



life.augmented

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 ✓

workspace_1.12.0 - Device Configuration Tool - STM32CubeIDE

File Edit Navigate Search Project Run Window Help

Project Explorer

STM32H5_STANDBY_IORetention
- Includes
- Core
- Drivers
- STM32H5_STANDBY_IORetention.ioc
- STM32H563ZITX_FLASH.ld
- STM32H563ZITX_RAM.ld

STM32H5_STANDBY_IORetention.ioc

STM32H5_STANDBY_IORetention.ioc - Pinout & Configuration

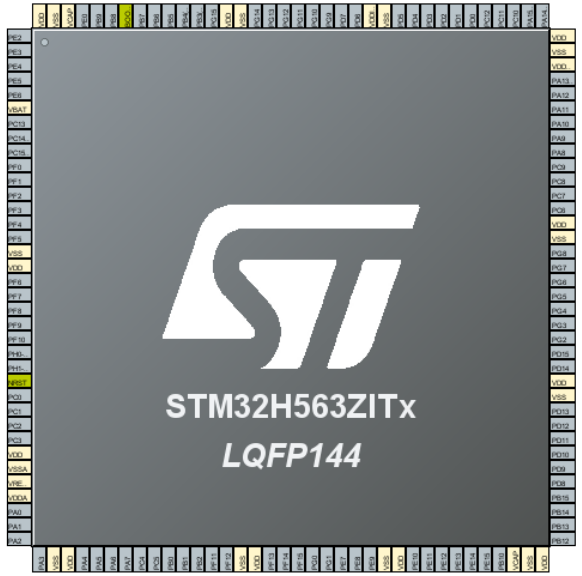
Pinout & Configuration | Clock Configuration | Project Manager

Software Packs | Pinout

Pinout view | System view

Categories | A->Z

- System Core
- Analog
- Timers
- Connectivity
- Multimedia
- Security
- Computing
- Middleware and Softwar...
- Trace and Debug
- Power and Thermal
- Utilities



life.augmented

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

life.augmented

ICACHE configuration in Pinout & Configuration

Pinout & Configuration | Clock Configuration | Project Manager | Tools

Software Packs | Pinout

1 Switch to A->Z tab

Categories | A->Z

DTS
ETH
FDCAN1
FDCAN2
FILEX
FMAC
FMC
GPDMA1
GPDMA2
GPIO
GTZC
GUI_INTERFACE
HASH
HDMI_CEC
I-CUBE-embOS
I-CUBE-wolfSSL
I-Cube-SoM-uGOAL
I2C1
I2C2
I2C3
I2C4
I2S1
I2S2
I2S3
I3C1
2 ICACHE
IWDG

and Configuration

ode

3 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!

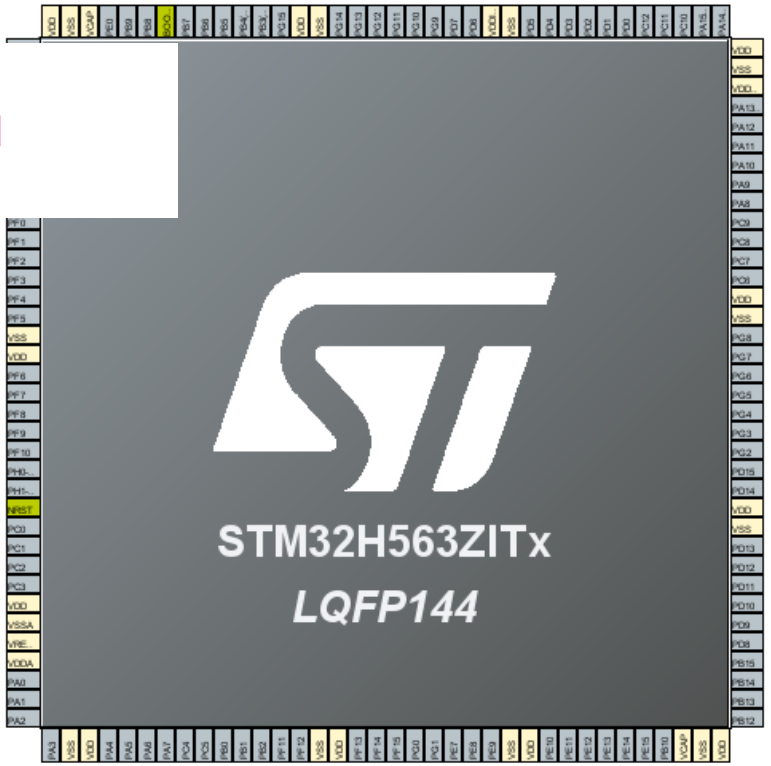
Configuration

Reset Configuration

Parameter Settings | NVIC Settings

Warning: This peripheral has no parameters to be configured

Pinout view | System view



USART3 configuration in Pinout & Configuration

Pinout & Configuration | Clock Configuration

Software Packs

Categories | A->Z

TIM7
TIM8
TIM12
TIM13
TIM14
TIM15
TIM16
TIM17
TRACER_EMB
UART4
UART5
UART7
UART8
UART9
UART12
UCPD1
USART1
USART2
USART3
USART6
USART10
USART11
USB
USBPD
USBX
VREFBUF
WWDG
X-CUBE-AI
X-CUBE-AI GOBU

1

2

USART3 Mode and Configuration

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

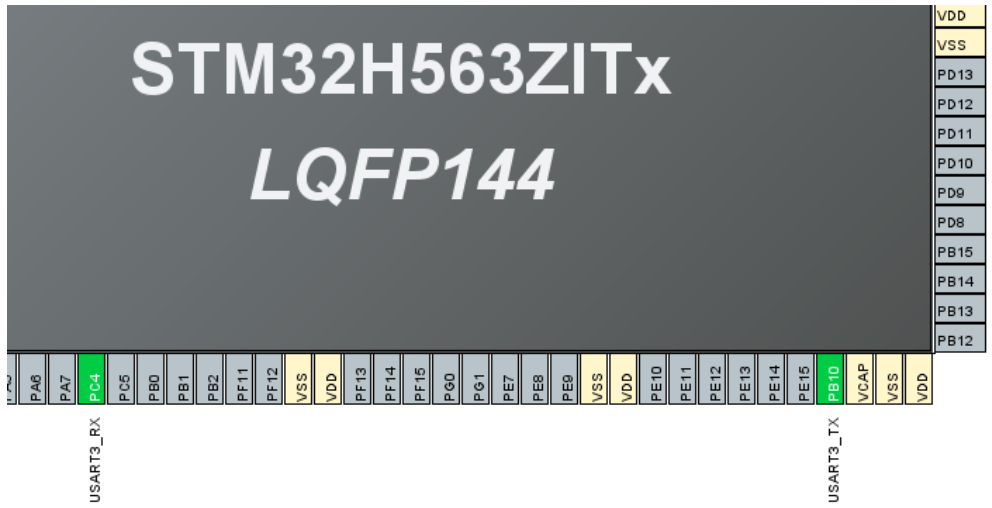
Configure the below parameters :

Search (Ctrl+F)

Basic Parameters

Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

Advanced Parameters

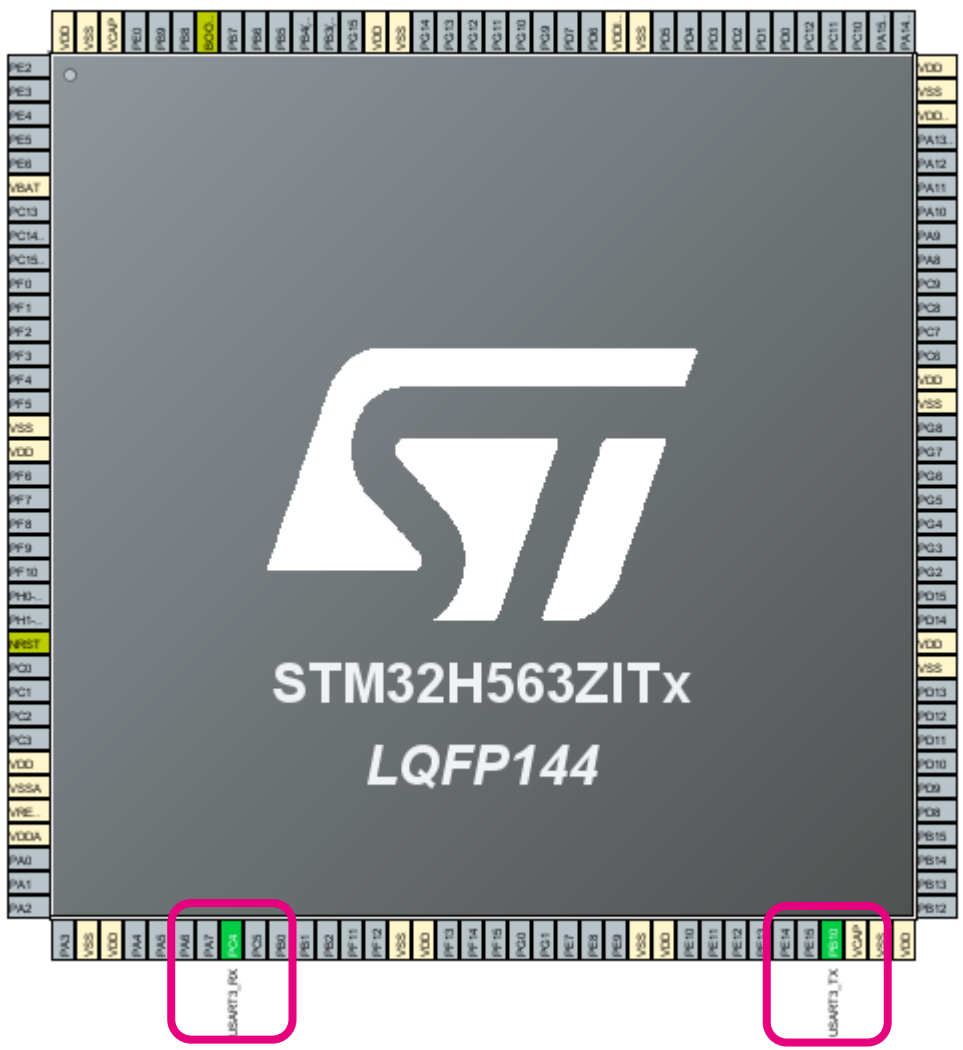


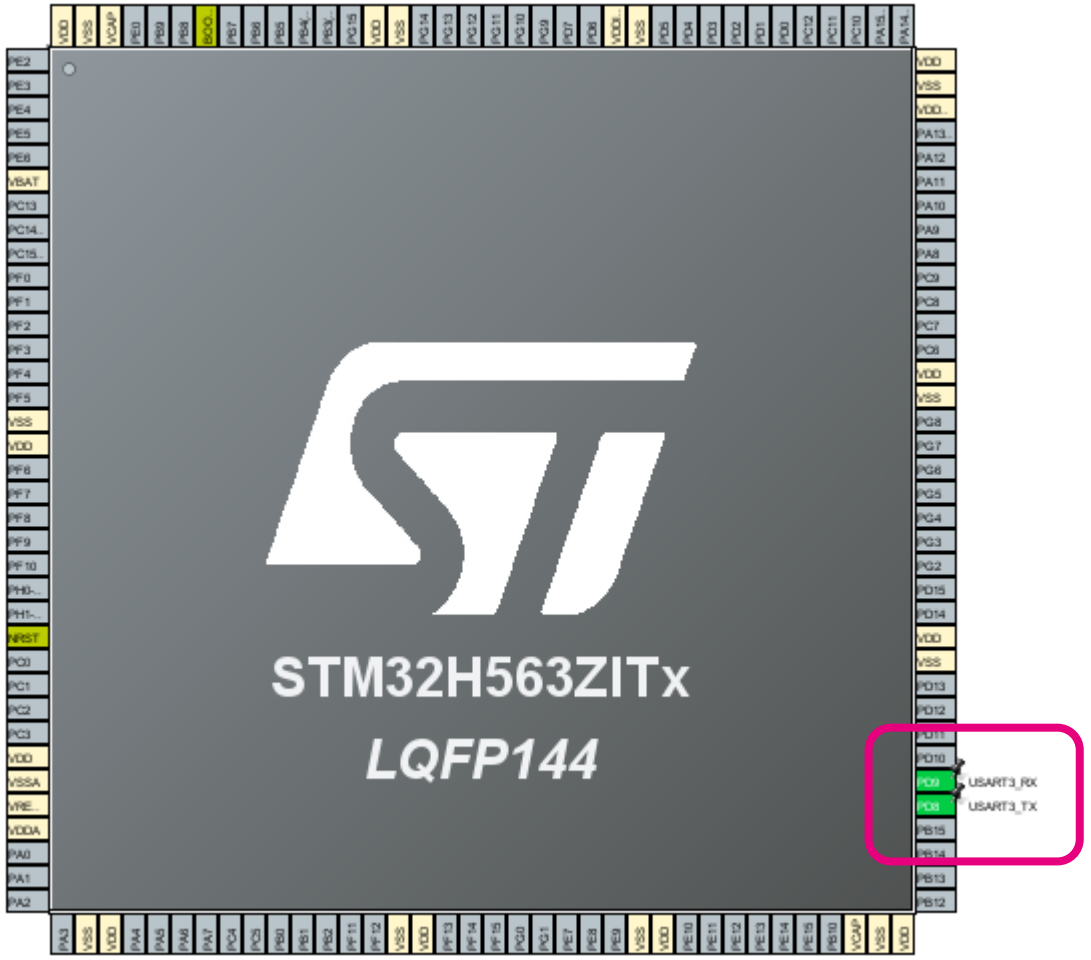
USART3 signals automatically assigned to pins

life.augmented

STM32CubeMX USART3 pins remap

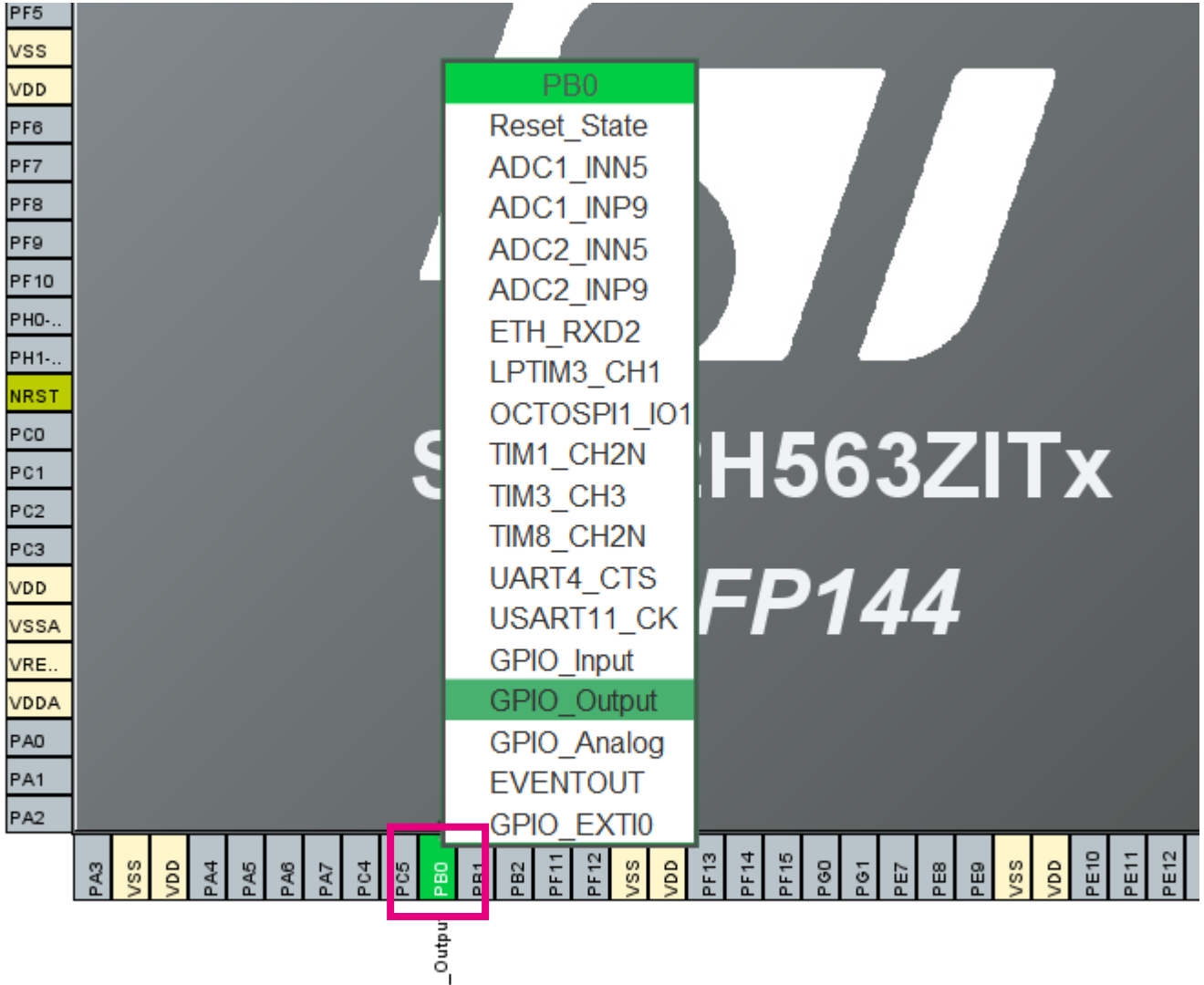
Remap USART3 signals to PD8/PD9 using CTRL + left-click and drag&drop





Configure PB0 (pin 46) as GPIO_Output

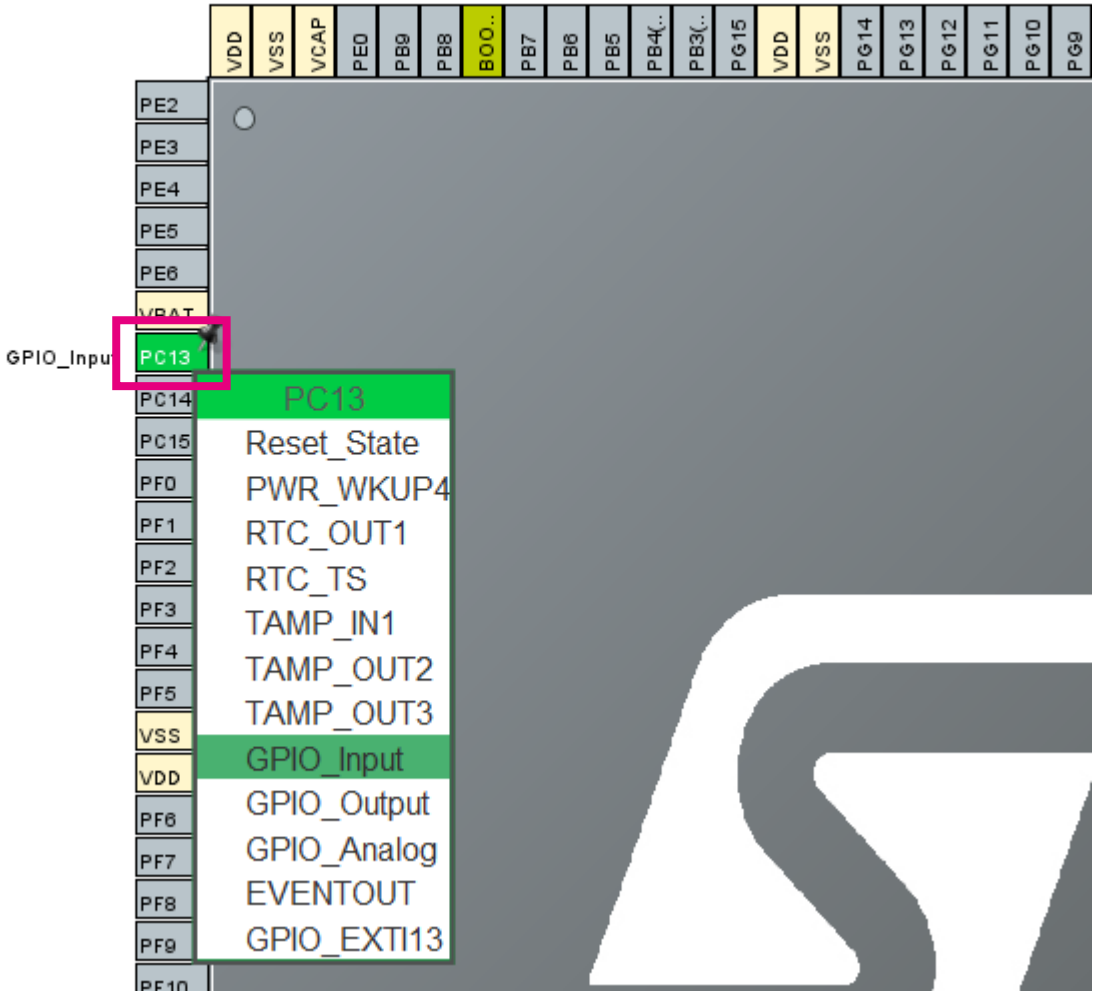
- PB0 is connected to a green LED (LD1)



life.augmented

Configure PC13 (pin 7) as GPIO_Input

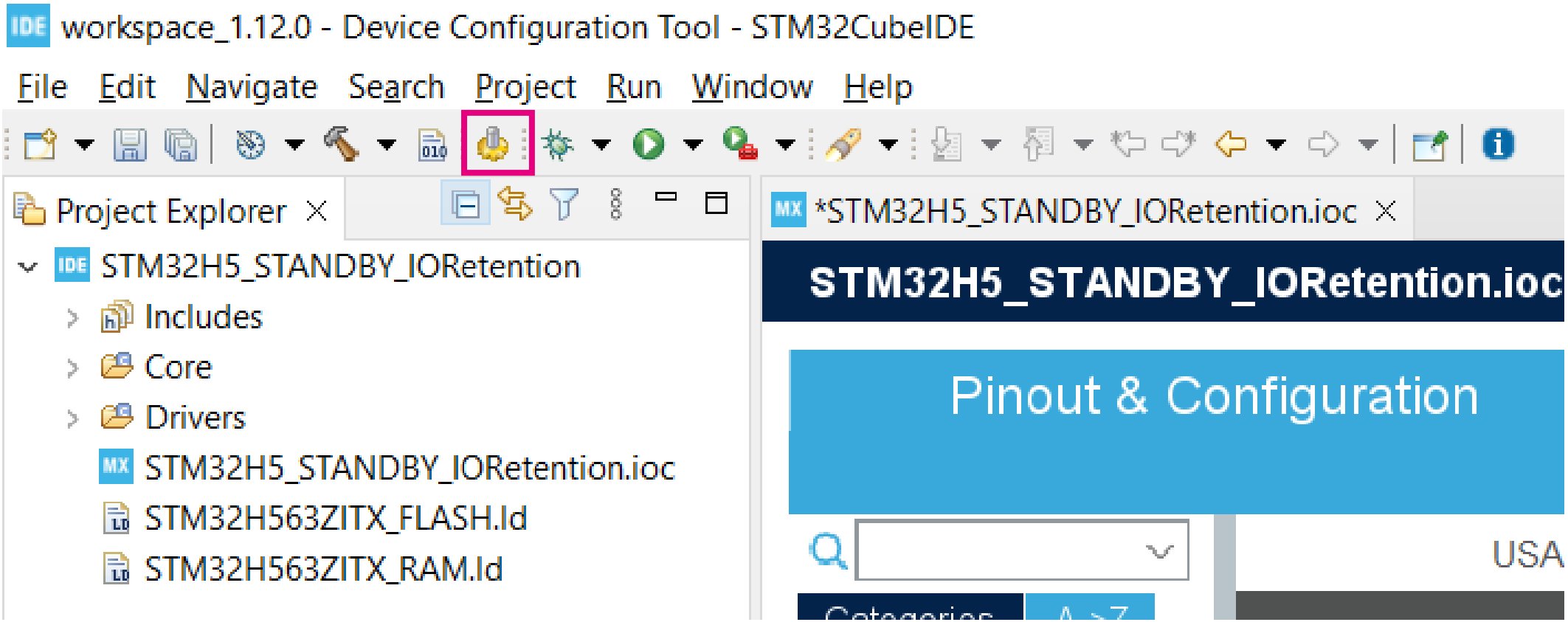
- PC13 is connected to a user button (blue button)



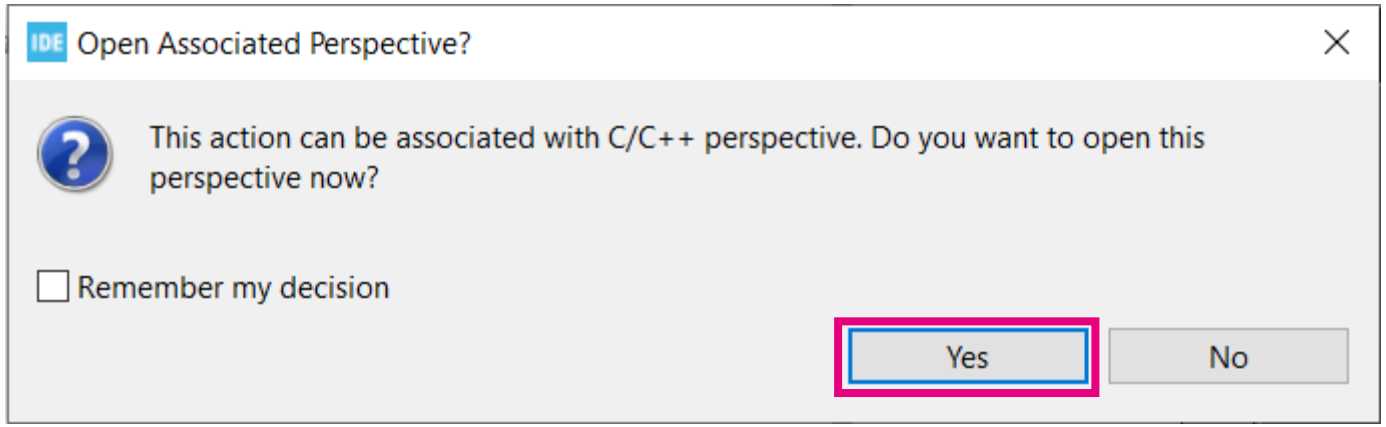
life.augmented

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 ✓

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

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer

- STM32H5_STANDBY_IORetention
  - Includes
  - Core
  - Drivers
  - STM32H5_STANDBY_IORetention.ioc
  - STM32H563ZITX_FLASH.ld
  - STM32H563ZITX_RAM.ld

STM32H5_STANDBY_IORetention.ioc | main.c

```c
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_ICACHE_Init();
  MX_USART3_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
```

life.augmented

Now, we‘ll fulfill the generated code with our application code

Extend the code

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

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

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer

- STM32H5_STANDBY_IORetention
  - Includes
  - Core
    - Inc
    - Src
      - main.c
      - stm32h5xx_hal_msp.c
      - stm32h5xx_it.c
      - syscalls.c
      - sysmem.c
      - system_stm32h5xx.c
    - Startup
  - Drivers
  - STM32H5_STANDBY_IORetention.ioc
  - STM32H563ZITX_FLASH.ld
  - STM32H563ZITX_RAM.ld

1

STM32H5_STANDBY_IORetention.ioc | *main.c

```
1  /* USER CODE BEGIN Header */
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"

2
/* Private includes ----------
23 /* USER CODE BEGIN Includes */
24 #include <stdio.h>
25 /* USER CODE END Includes */
26
```

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

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)
```

main.c, line ~34

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

Project Explorer

- STM32H5_STANDBY_IORetention
  - Includes
  - Core
    - Inc
    - Src
      - main.c
      - stm32h5xx_hal_msp.c
      - stm32h5xx_it.c
      - syscalls.c
      - sysmem.c
      - system_stm32h5xx.c
    - Startup
  - Drivers
  - STM32H5_STANDBY_IORetention.ioc
  - STM32H563ZITX_FLASH.ld
  - STM32H563ZITX_RAM.ld

STM32H5_STANDBY_IORetention.ioc | *main.c

```
 * If no LICENSE file comes with this s
```

Copy USER CODE PD section

from *STM32H5_IO_State_Retention_code.txt*

```
#include "main.h"

/* Private includes ----------------------
/* USER CODE BEGIN Includes */
#include <stdio.h>
/* USER CODE END Includes */

/* Private typedef -----------------------
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------
/* USER CODE BEGIN PD */
//#define IORETENABLE
/* USER CODE END PD */
```

life.augmented

main.c, line ~56

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

Project Explorer

- STM32H5_STANDBY_IORetention
  - Includes
  - Core
    - Inc
    - Src
      - main.c
      - stm32h5xx_hal_msp.c
      - stm32h5xx_it.c
      - syscalls.c
      - sysmem.c
      - system_stm32h5xx.c
    - Startup
  - Drivers
  - STM32H5_STANDBY_IORetention.ioc
  - STM32H563ZITX_FLASH.ld
  - STM32H563ZITX_RAM.ld

STM32H5_STANDBY_IORetention.ioc | *main.c

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

Copy USER CODE PFP section

from *STM32H5_IO_State_Retention_code.txt*

```
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 */
```

life.augmented

main.c, line ~100

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

ce Refactor Navigate Search Project Run Window Help

Extend the code

Project Explorer

- STM32H5_STANDBY_IORetention
  - Includes
  - Core
    - Inc
    - Src
      - main.c
      - stm32h5xx_hal_msp.c
      - stm32h5xx_it.c
      - syscalls.c
      - sysmem.c
      - system_stm32h5xx.c
    - Startup
  - Drivers
  - STM32H5_STANDBY_IORetention.ioc
  - STM32H563ZITX_FLASH.ld
  - STM32H563ZITX_RAM.ld

Copy USER CODE 2 section from *STM32H5_IO_State_Retention_code.txt*

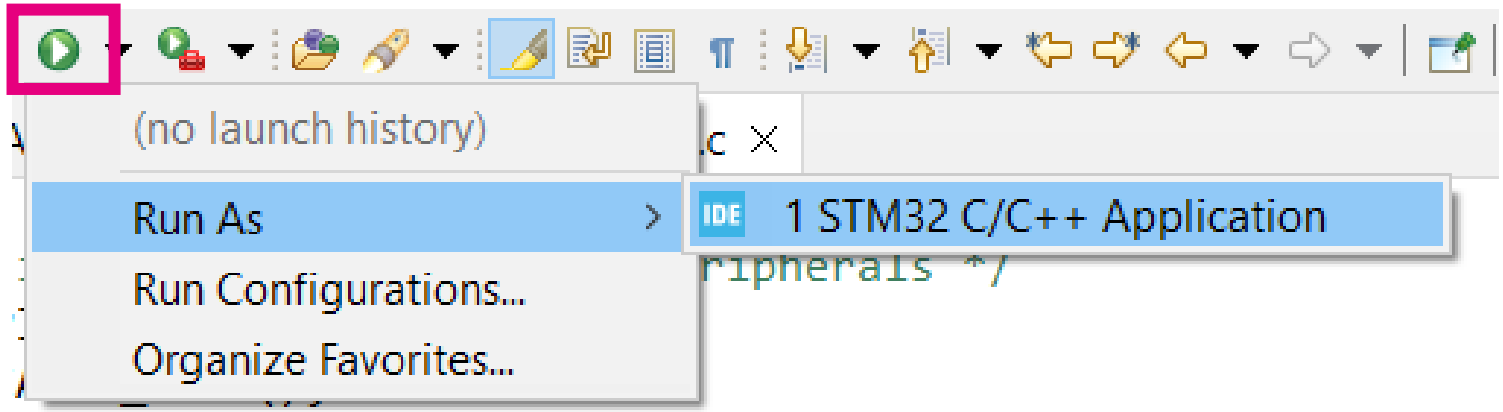
```c
/* USER CODE BEGIN 2 */
printf("\n\r ***************** IO RETENTION EXAMPLE ****************\n\r");
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
printf( "LD1 turned on              \r\n");
if(READ_BIT(PWR->IORETR, PWR_IORETR_IORETEN)== PWR_IORETR_IORETEN)
{
  printf( "IO retention bit set             \r\n");
        HAL_Delay(1000);
        HAL_PWREx_DisableStandbyIORetention();
        printf( "Disabling IO retention            \r\n");
        HAL_Delay(10);
}
printf( " ********************************************************\r\n");
printf( " *********** Push user button to enter STANDBY ***********\r\n");
printf( " ********************************************************\r\n");
while(HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13)== GPIO_PIN_RESET)
{
        HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_0);
        HAL_Delay(100);
}
printf( "MCU entering STANDBY: use reset button to wake-up  \r\n");
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);
#if defined(IORETENABLE)
printf("Enabling IO retention  \r\n");
HAL_PWREx_EnableStandbyIORetention();
#endif

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

life.augmented

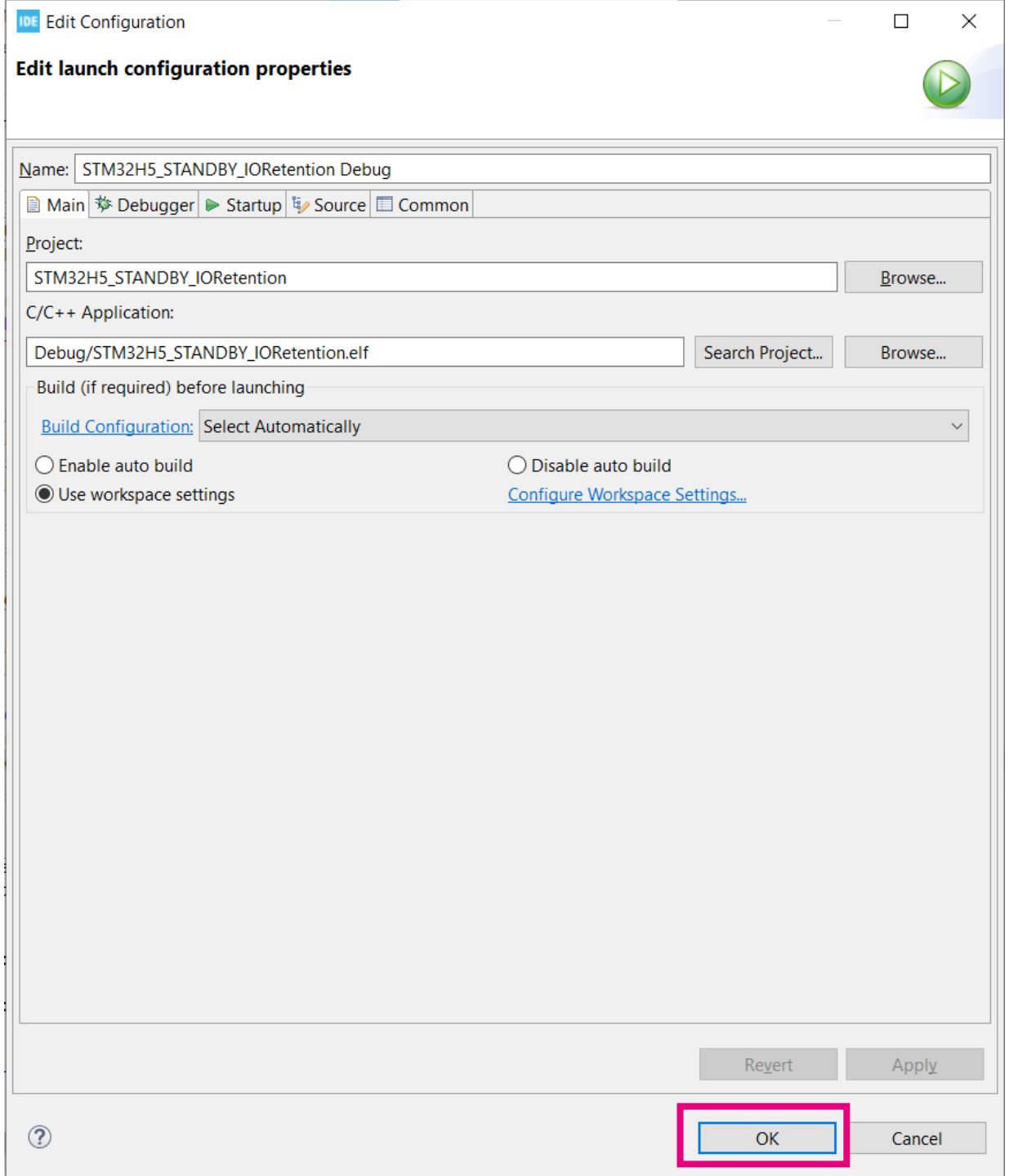
At first we‘ll observe the behavior without IO state retention enabled

Build and flash the project

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



Edit Configuration

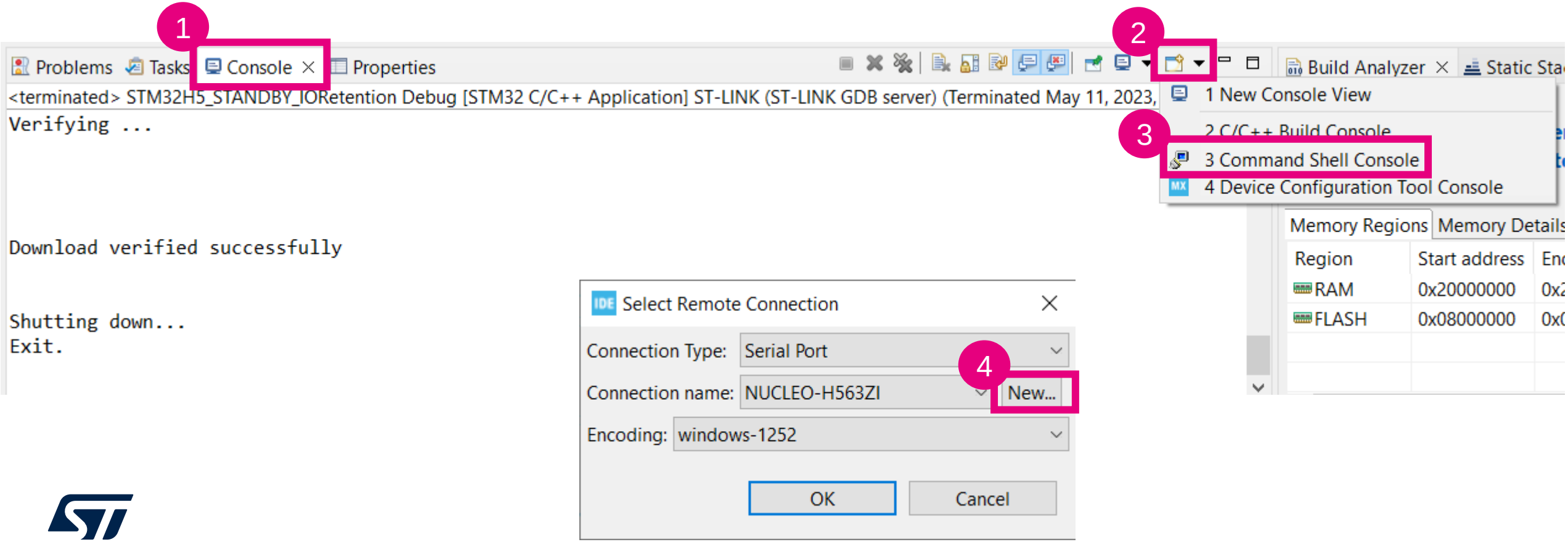


Just click OK

life.augmented

Configure serial port connection

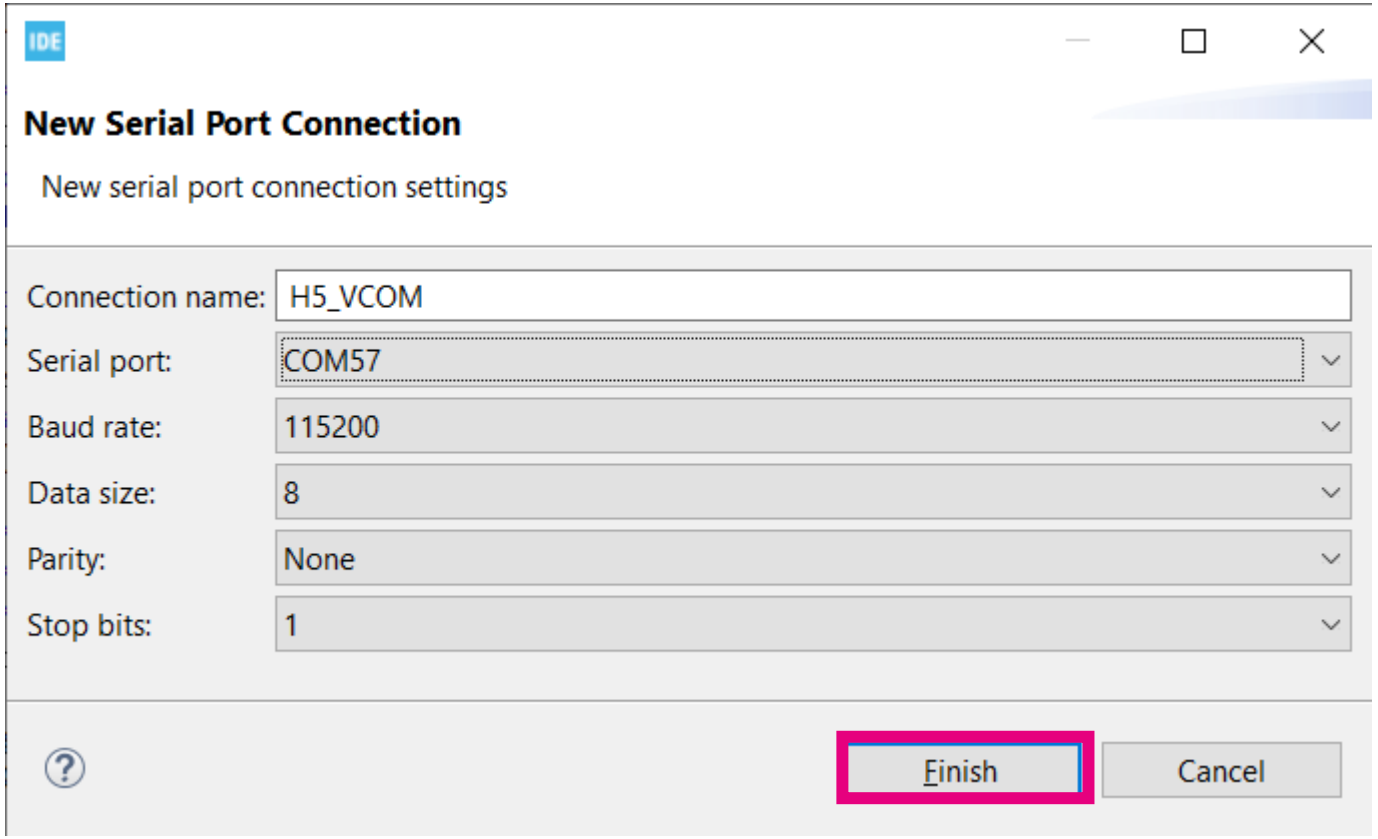
- Open Console -> Command Shell Console

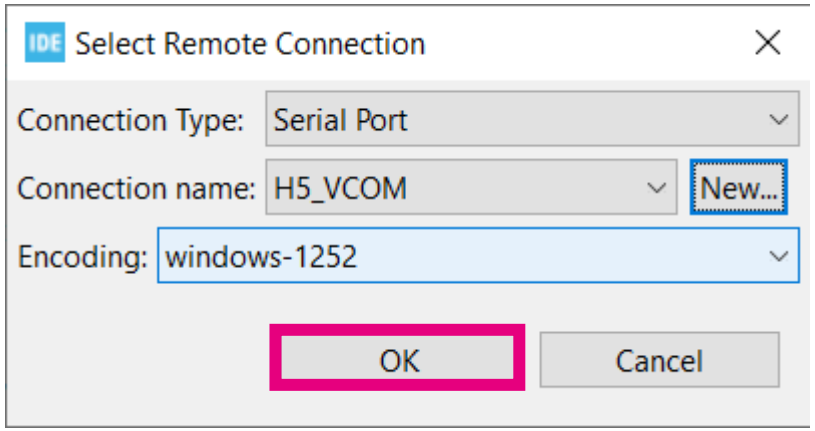


life.augmented

Configure serial port connection

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





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)

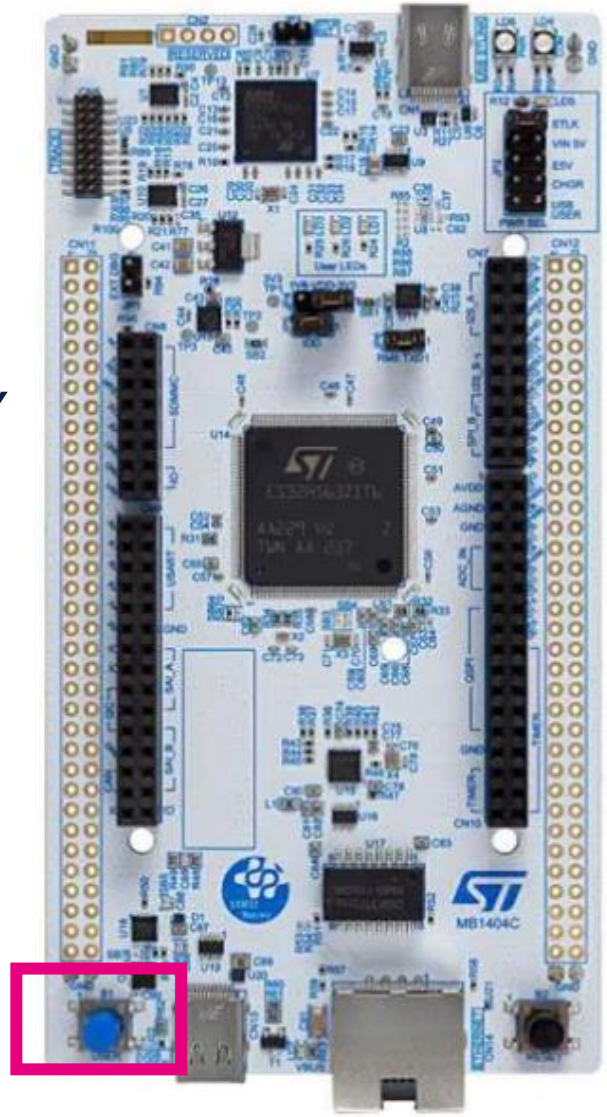
```
****************** IO RETENTION EXAMPLE ******************
LD1 turned on
*************************************************************
*********** Push user button to enter STANDBY ***********
*************************************************************
```

life.augmented

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



life.augmented

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);
#if defined(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

life.augmented

Let‘s enable the IO state retention in STANDBY

main.c, line ~34

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

- STM32H5_STANDBY_IORetention
  - Binaries
  - Includes
  - Core
    - Inc
    - Src
      - main.c
      - stm32h5xx_hal_msp.c
      - stm32h5xx_it.c
      - syscalls.c
      - sysmem.c
      - system_stm32h5xx.c
    - Startup
  - Drivers
  - Debug
  - STM32H5_STANDBY_IORetention.ioc
  - STM32H5_STANDBY_IORetention Debug.laur
  - STM32H563ZITX_FLASH.ld
  - STM32H563ZITX_RAM.ld

STM32H5_STANDBY_IORetention.ioc | *main.c

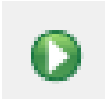
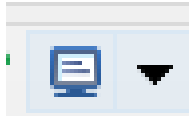
```
12   * This software is licensed under
13   * in the root directory of this sc
14   * If no LICENSE file comes with th
15   *
16   ******************************
```

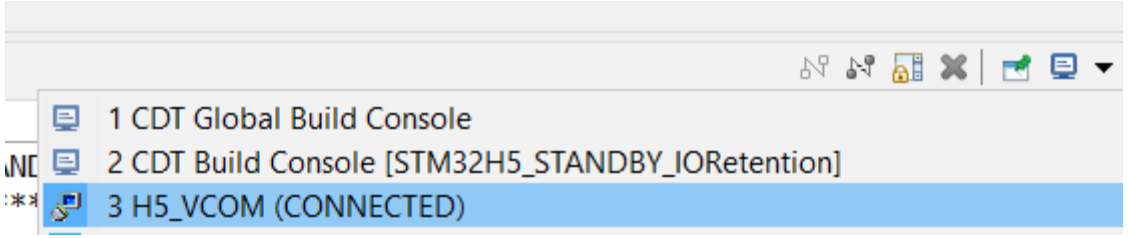
Uncomment #define IORETENABLE

```
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
```

life.augmented

Build and flash the project

1) Build 
2) Run 
3) Display serial port connection console 



4) Press the reset button

life.augmented

Observe the behavior w/ IO state retention in STANDBY enabled

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

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

The green LED (LD1) should be ON

IO state retention in STANDBY is enabled

Entering the STANDBY mode

life.augmented

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)

life.augmented

Summary

- 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