life.augmented

ST67W611
Wi-Fi 6 + Bluetooth® LE + Thread

Santa Clara Wireless Team

Agenda

- # X-Cube-ST67W61 Package
- # Device Setup and Flashing
- # BLE Commissioning App
- # Generating BLE Commissioning app using NUCLEO-U575ZI-Q as host processor (Hands-on)
- # Porting to a new host mcu using STM32CubeMX
- # Generating the BLE Commissioning app using NUCLEO-G0B1RE processor (Hands-on)
- # Documentation

Prerequisites

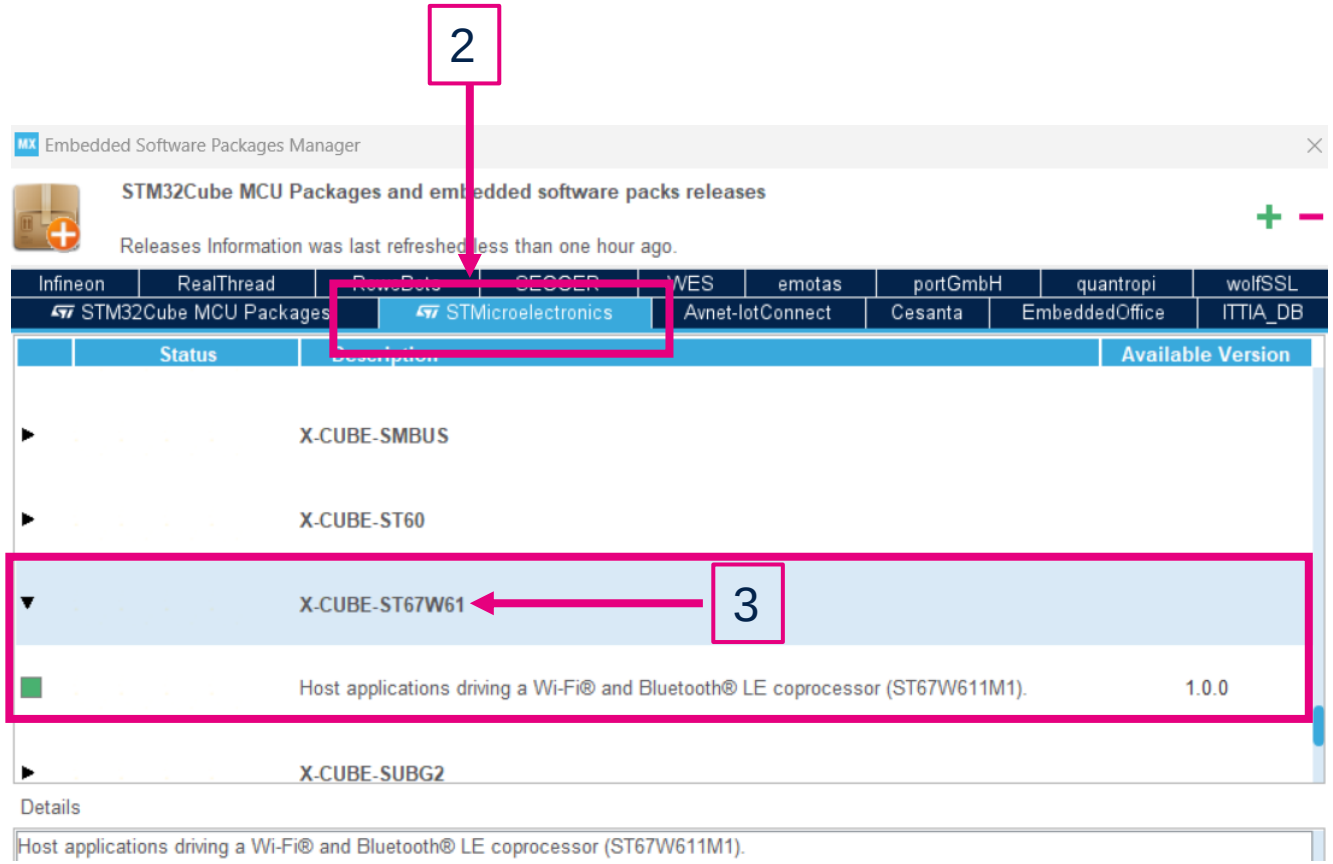
PLEASE DOWNLOAD ALL THE TOOLS LISTED BELOW

- X_CUBE_ST67W61_V1.0.0 (To be installed under C drive)
- STM32CubeIDE v1.16.1+
- STM32CubeProgrammer v2.18.0+ To be installed in default directory (C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer)
- STM32CubeMX_v6.15.0
- HyperTerminal – TeraTerm v.5.4.0
- ST BLE Toolbox (To be downloaded on mobile)

Installing X-CUBE-ST67W61

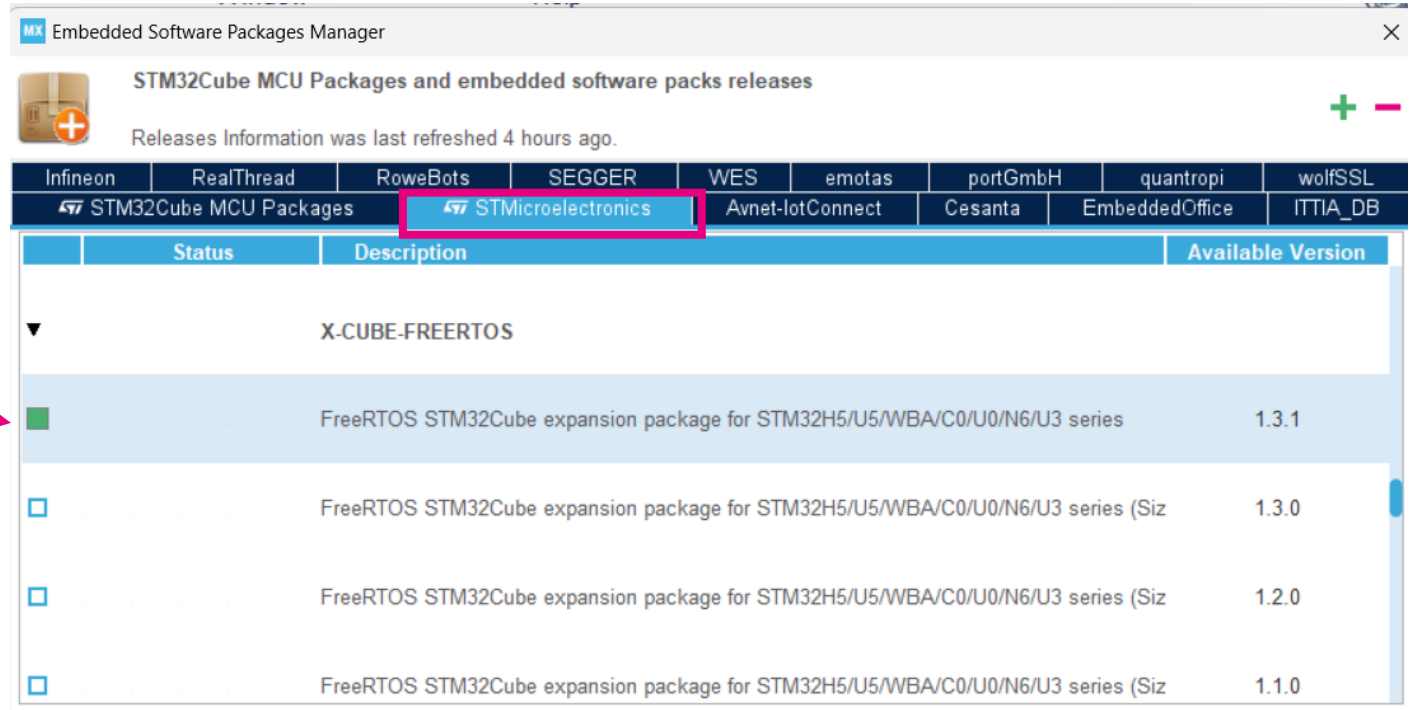
Open STM32CubeMX with admin rights

1. Go to “Install/Remove” software packages
2. Under STMicroelectronics tab
3. Install the X-CUBE-ST67W61

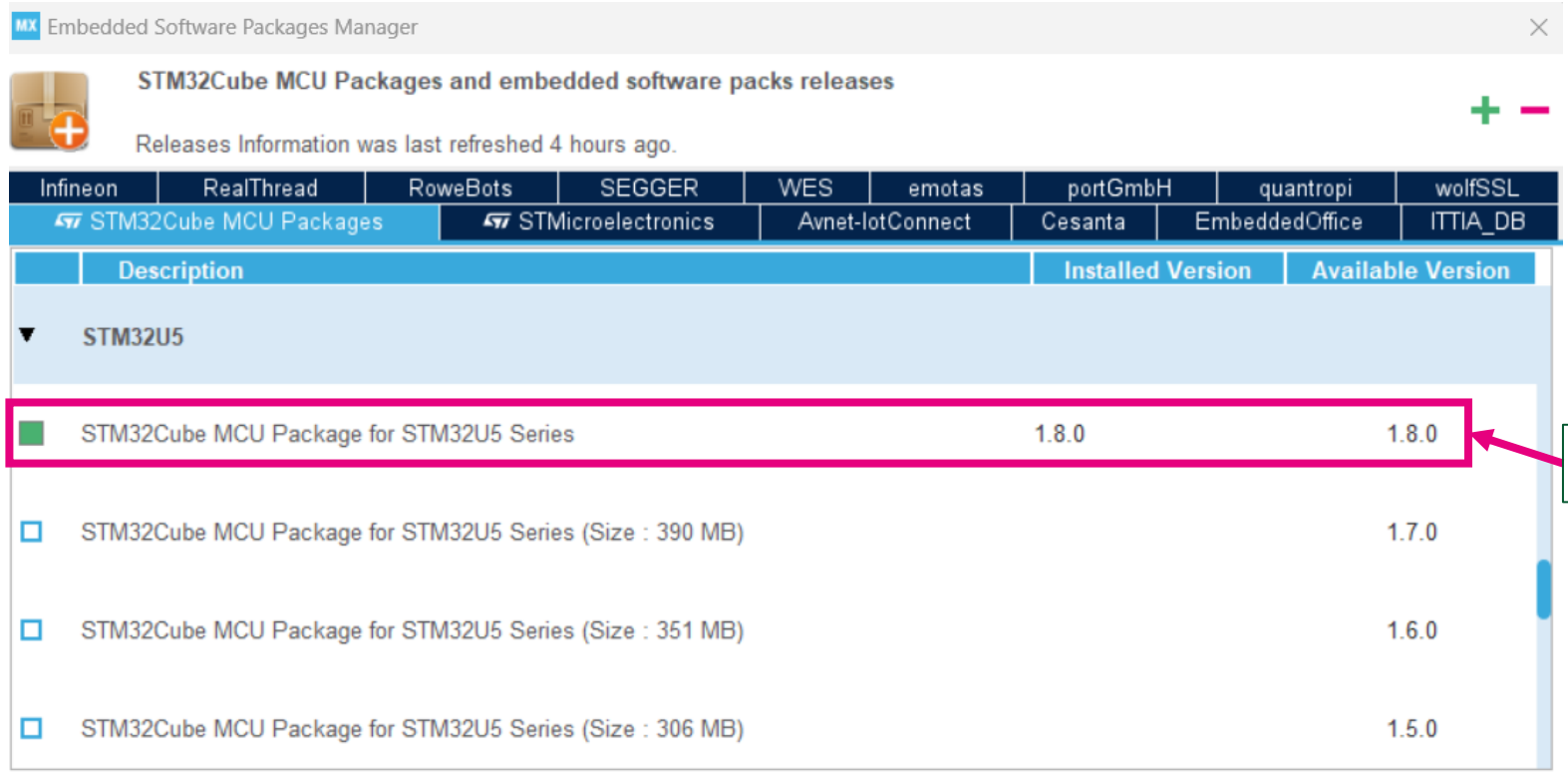


X-CUBE-FREERTOS

Download latest X-CUBE-FREERTOS



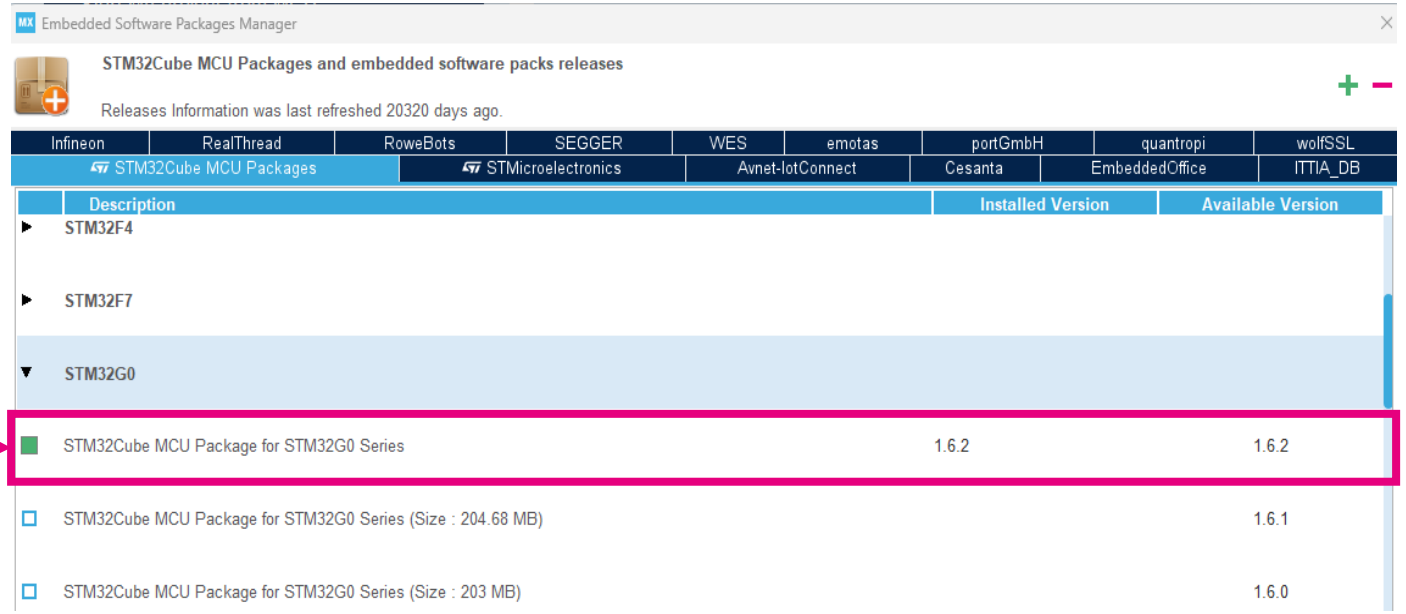
STM32U5



Download latest STM32U5

STM32G0

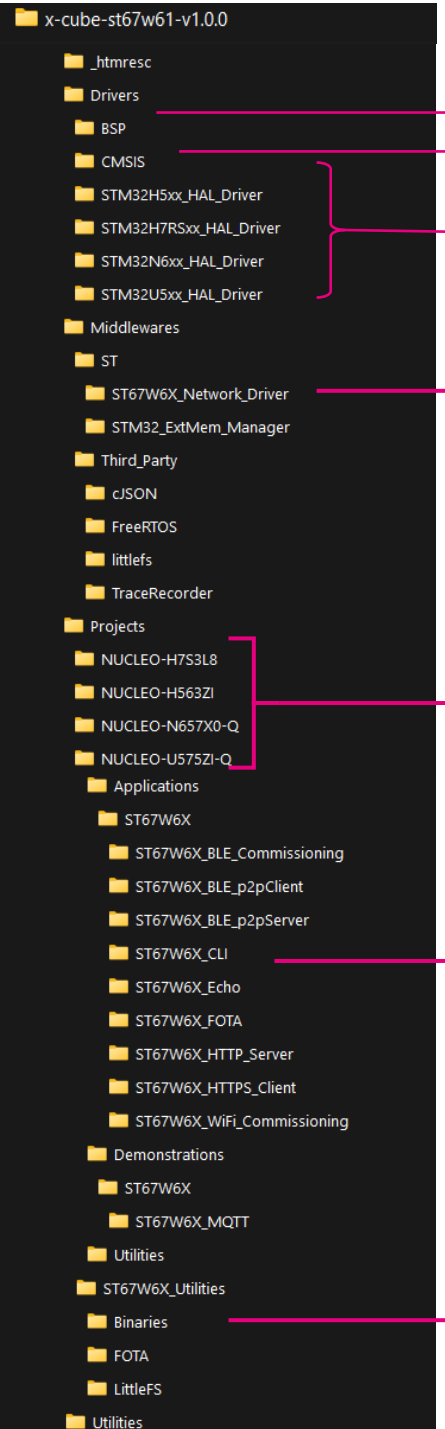
Download latest STM32G0



life.augmented

X-Cube-ST67W61

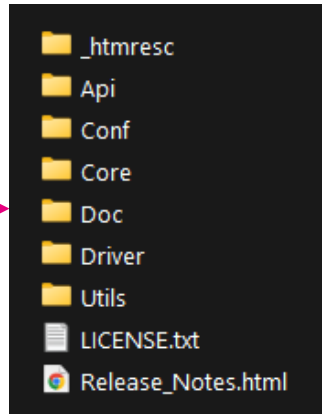
X-CUBE-ST67W61 package overview



Board support package

CMSIS libraries

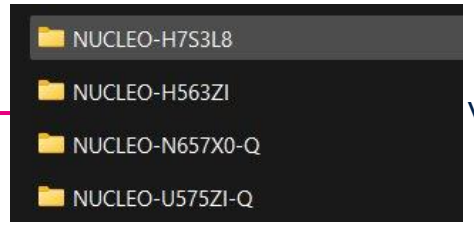
Device HAL drivers



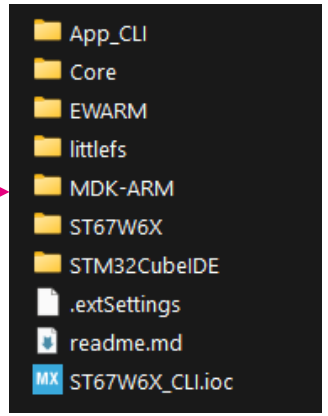
APIs include files
Templates for middleware and driver config files
Integration between generic API and driver

W61 driver
AT commands / responses
SPI driver

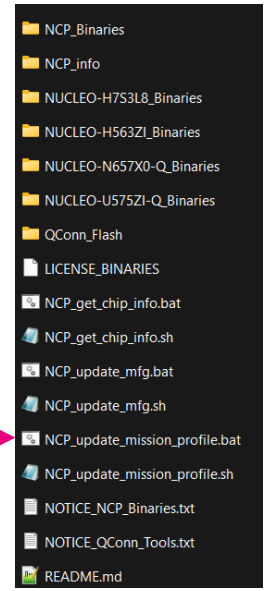
UART Output logs
Memory usage statistics
CLI Console



Variety of **Nucleo** hosts



STM32CubeIDE and **EWARM** Project Files



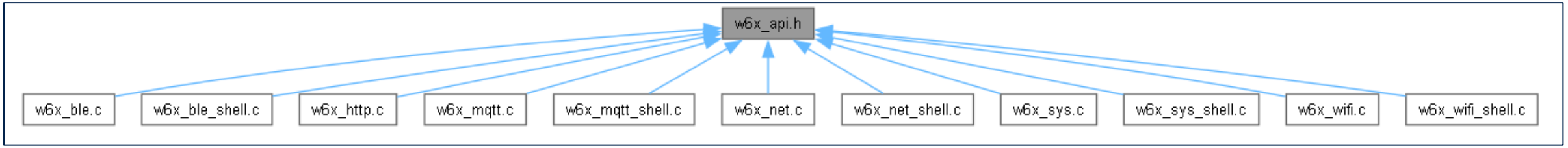
NCP and Host binaries

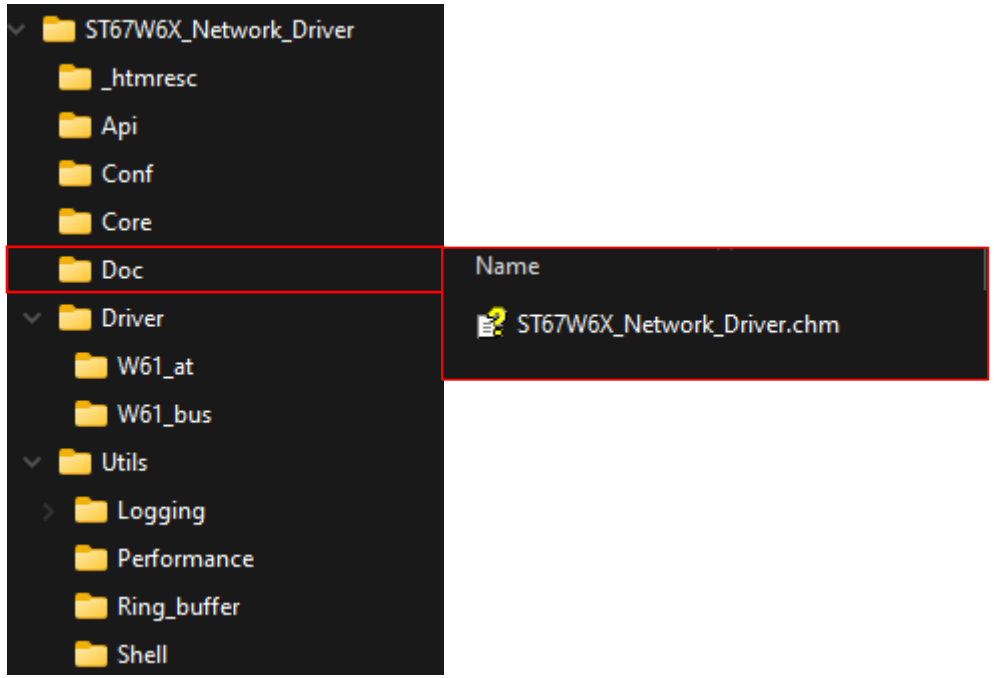
X-CUBE-ST67W61 middleware

	Scope / description
W61_bus	This is the **SPI driver for communication** between the host and the ST67W61.
W61_at	It does the **formatting of the AT commands** to be sent, and it implements a receive tasks **to decode the received messages** from the W61 and dispatch them to the correspondent module.
W6x service API	The Core directory implements an adaptation layer between the W61 AT driver and the API proposed to the application. The APIs are "Zephyr-like" for the Wi-Fi® and socket API for the network operations.
Utils	Console handling, debug, trace, performance measurements.

All the above submodules make use of FreeRTOS™

API Documentation

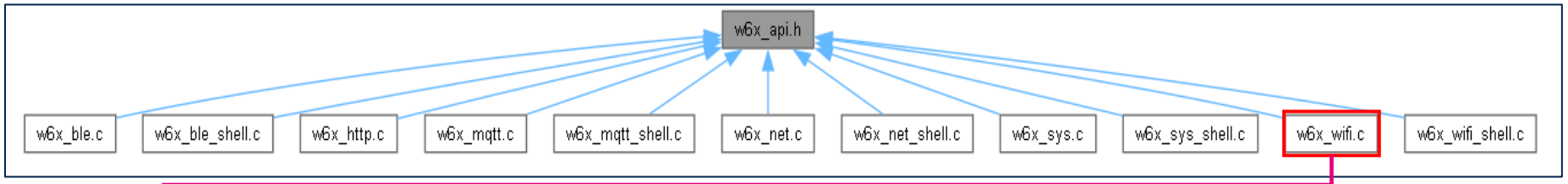


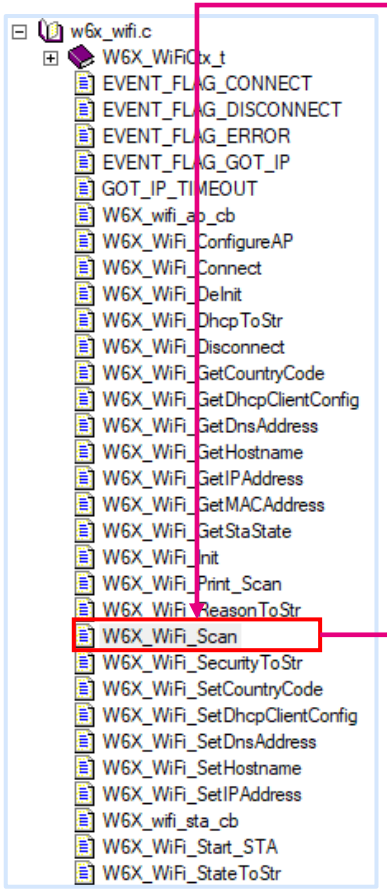


Found in \Middlewares\ST\ST67W6X_Network_Driver**\Doc**

- API Definitions
- Overview of API calls
- Correlated AT Commands

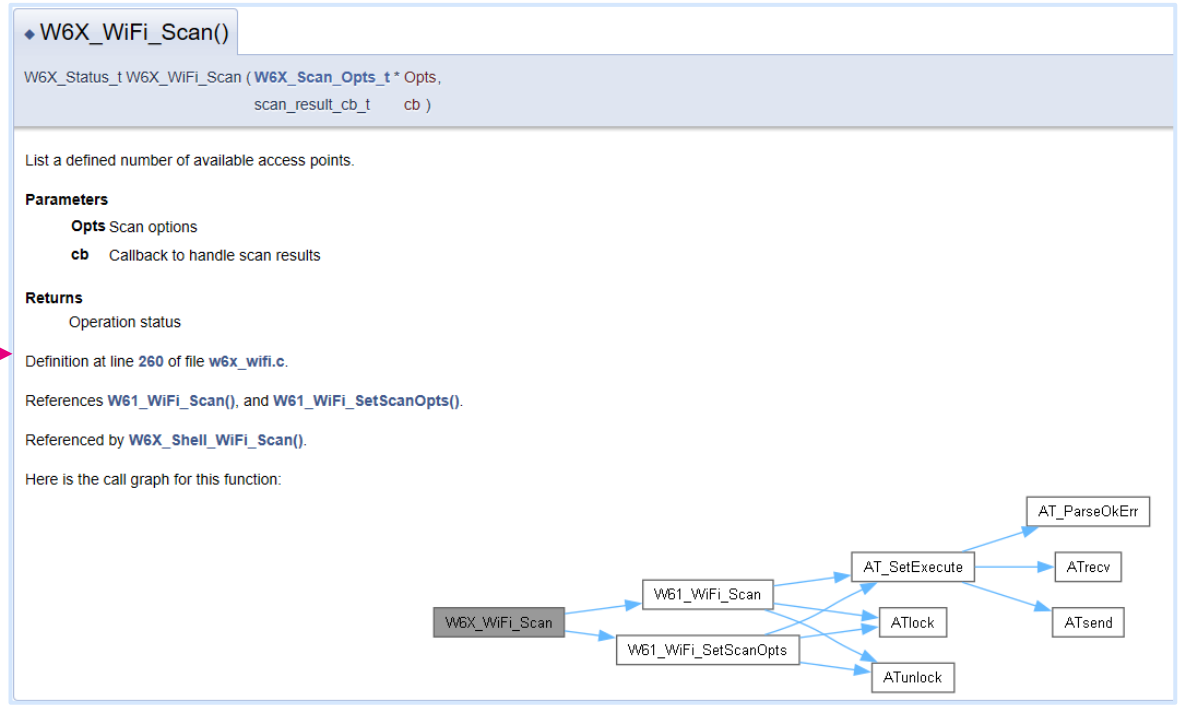
API Documentation cont…



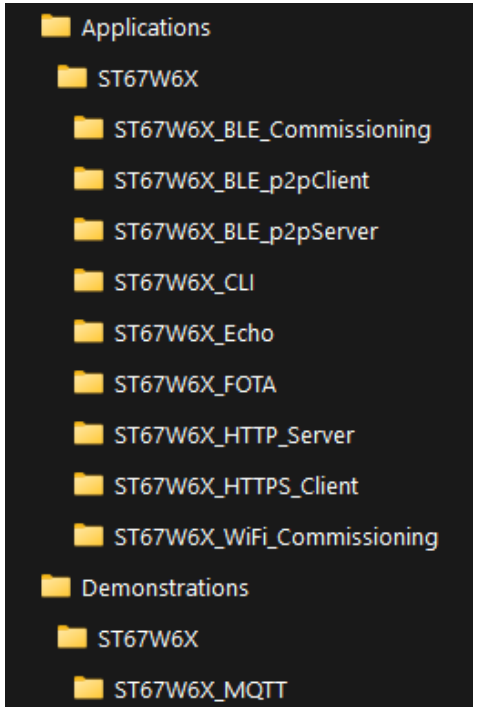


Selecting **w6x_wifi.c** will provide the different API categories

Selecting the API, **W6X_WiFi_Scan(),** will provide information and the call graph



X-CUBE-ST67W61 applications

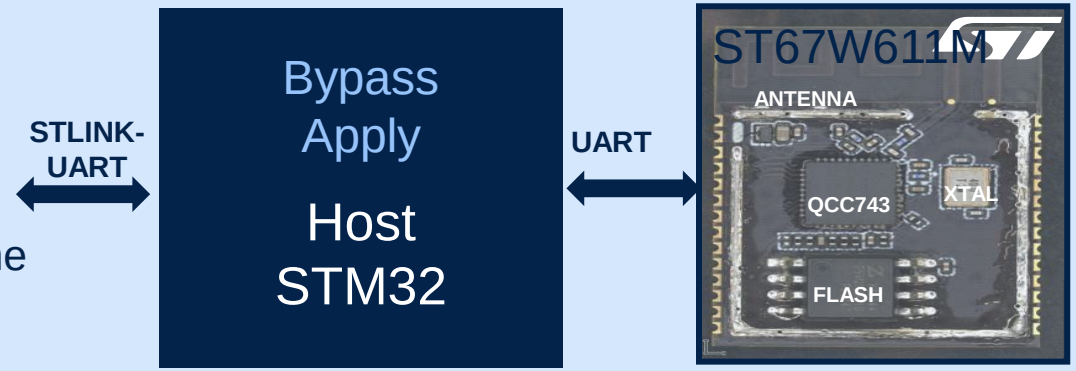


	Scope / description
Bluetooth® LE commissioning	Uses Bluetooth® LE to provide Wi-Fi® SSID and password after scanning available AP
Bluetooth® LE p2p client	Point-to-Point communication demo using Bluetooth® LE (GATT Client) Scan and connect to a p2pServer Write messages & receive notifications from the p2pServer
Bluetooth® LE p2p server	Point-to-Point communication demo using Bluetooth® LE (GATT server) Advertises and wait for a connection from either: - p2pClient app - *ST BLE Toolbox* smartphone application
Echo	TCP Echo feature over Wi-Fi - Good starting point for generic user development
CLI	ST67 evaluation via CLI and for WTS tests the Wi-Fi solution. - Wi-Fi API: scan, connect, disconnect, etc - NET socket API and iperf - Bluetooth® LE API
Echo FOTA	FOTA (firmware update over-the-air) feature demo over Wi-Fi - Using HTTP
HTTP server	HTTP server over Network API The server, running on the host board, has been validated with 1 client, using HTTP requests to send and receive data to and from the server.
HTTPS client	HTTPS Client over Network API demo - Basic GET requests to retrieve worldwide weather reports
Wi-Fi® commissioning	Wi-Fi® commissioning project using an HTTP server, hosted by a device configured as a soft access point (SoftAP) and station (STA).
MQTT Demo	Starting point for MQTT user development - Sends IKS sensor data to an MQTT broker

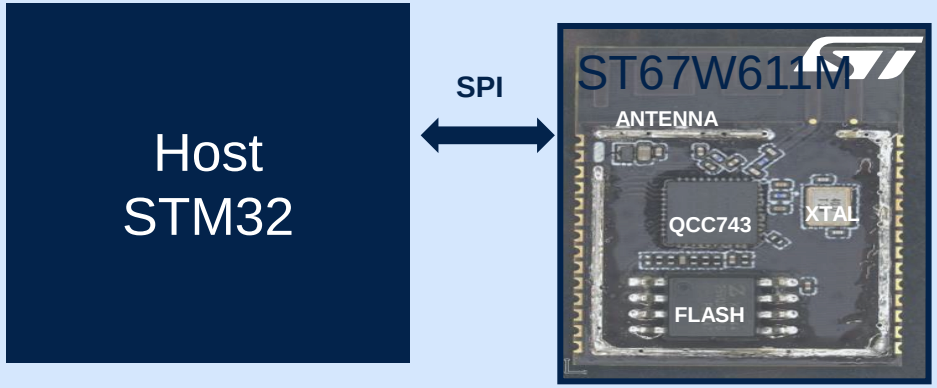
ST67W611M Modes of Operation

Manufacturing and Mission Modes

- **Manufacturing mode**
  - Host-less Application
  - Dedicated to RF performances monitoring and calibration.
  - QC provides graphical tools to control the device through the **UART** interface

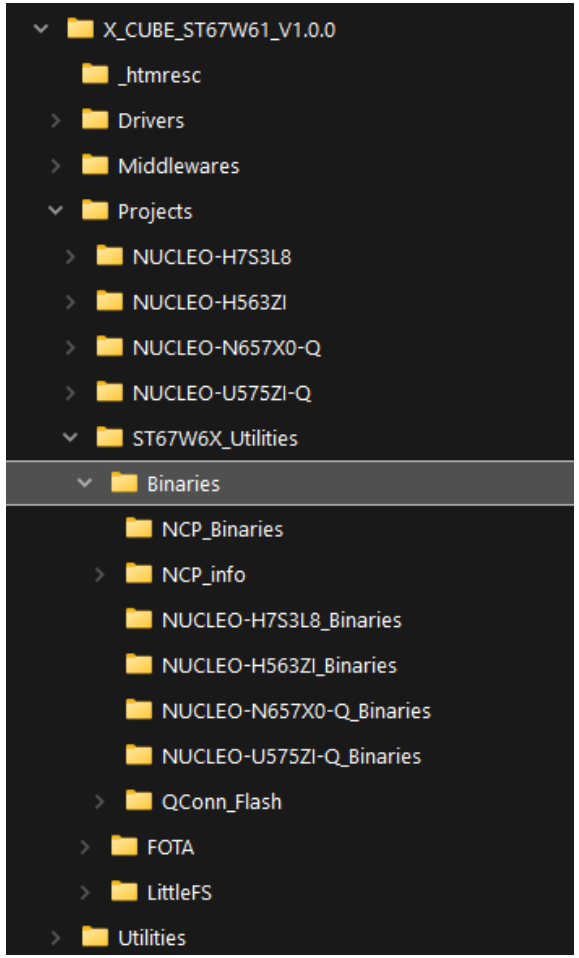


- **Mission mode (Functional)**
  - Implementing the AT command for the different supported protocol (WIFI / BLE / MQTT /..) through the **SPI** interface.
  - The ROM boot loader loads and executes the program from SPI Flash to boot the system.



ST67W611M1 BOOT and bootstrap modes - stm32mcu

X-CUBE-ST67W1 NCP tools



Scripts for flashing the latest mission or manufacturing firmware:

..\x-cube-st67w61-v1.0.0\Projects\ST67W6X_Utilities\Binaries

Tool	Description
NCP_get_chip_info.bat	Script to retrieve ST67 MAC address and locking information even if no firmware is loaded onto the ST67 device.
NCP_update_mfg.bat	Script to flash the ST67 module with the manufacturing application, which is used for RF evaluation.
NCP_update_mission_profile.bat	Script to flash the ST67 module with the mission profile application, which is used for normal operations and development purposes.

ST67W611M Security Overview

Two modes, each with a header containing a digital signature and versioning info

A. Mission mode:

- Bootloader binary
  - Authenticated by the ROM code during initial boot process
  - Versioning info to prevent rollback
- Mission profile binary
  - Wi-Fi® & Bluetooth® LE binary used for operational tasks
  - Authenticated by bootloader to ensure integrity & authenticity before execution

B. Manufacturing mode:

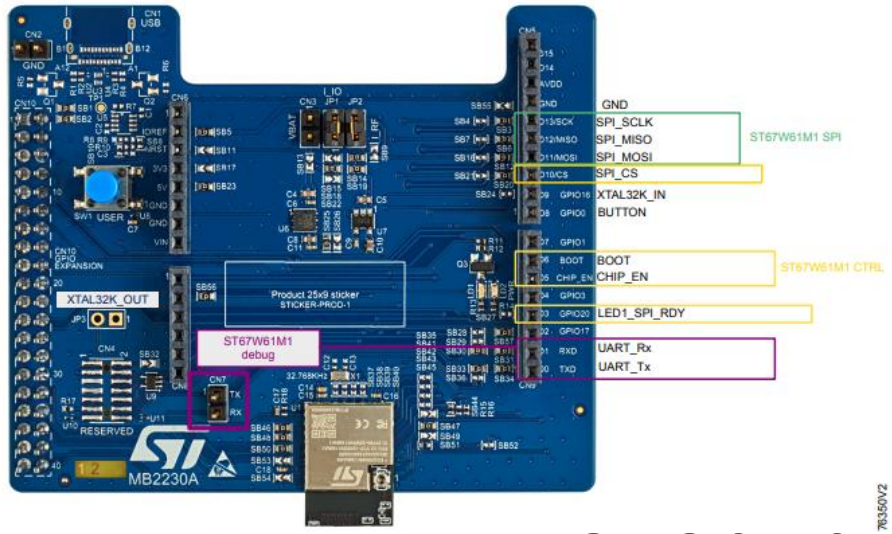
- Bootloader binary
  - Authenticated by the ROM code during initial boot process
  - Versioning info to prevent rollback
- MFG firmware binary
  - Used for test and production configuration
  - Authenticated by bootloader to ensure integrity & authenticity before execution

life.augmented

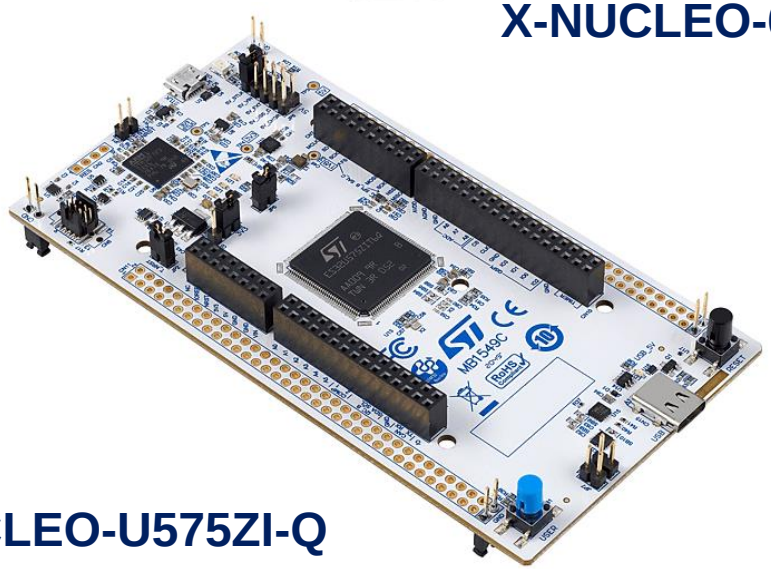
Flashing ST67 & Device Setup

STM32U575 and ST67W611M configuration

- 1x NUCLEO-U575ZI-Q board
- 1x X-NUCLEO-67W61M1 board
- Boards connected through Arduino® connectors
  - Key signals: SPI (SCK, MISO,MOSI,CS), SPI_RDY, CHIP_EN, BOOT, UART_TX/RX
- Power: MicroUSB connection to NUCLEO-U575ZI-Q

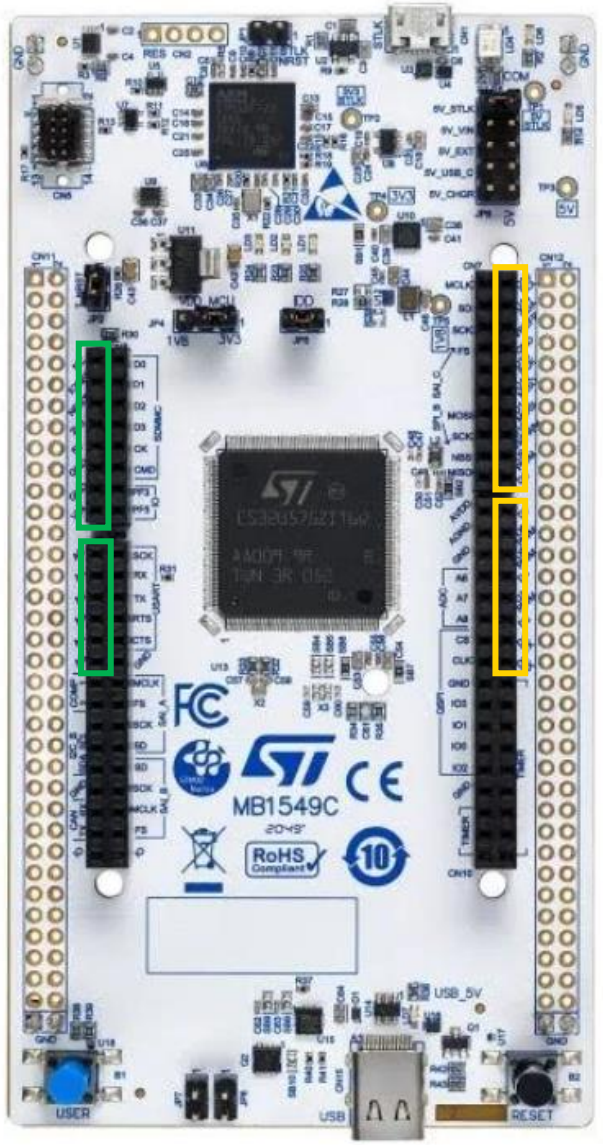


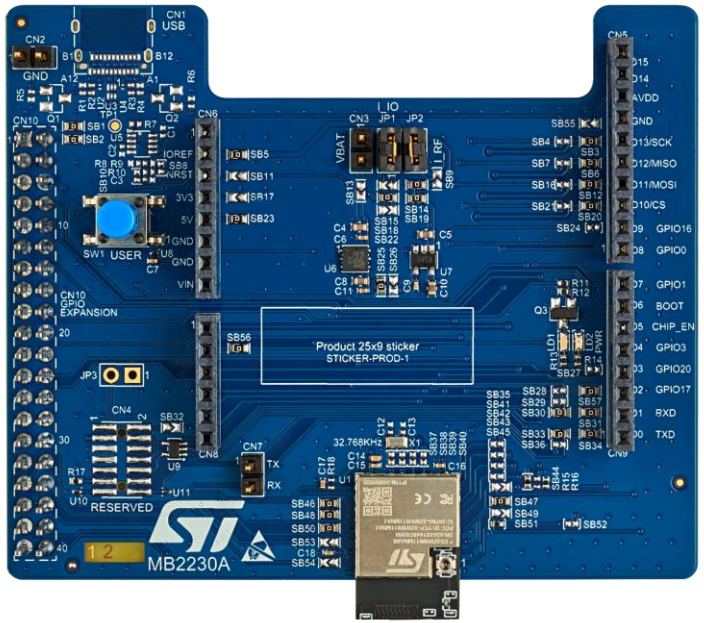
X-NUCLEO-67W61M1



NUCLEO-U575ZI-Q

Nucleo-U575ZI and X-Nucleo-67W61M1 Configuration





When stacking the Nucleo and X-Nucleo boards the Arduino connectors must be aligned.

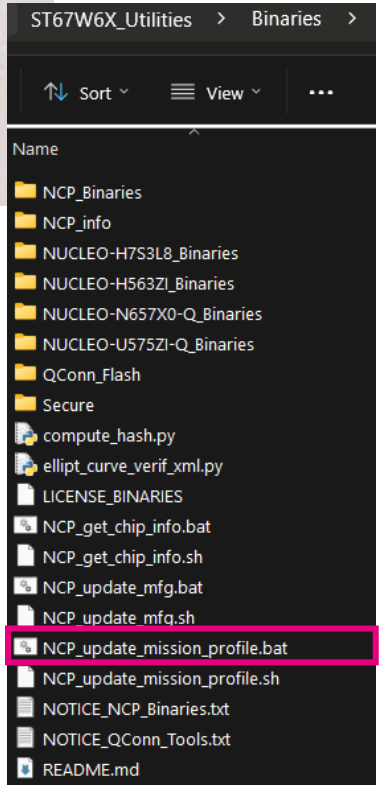
The MicroUSB port is used when programming the STM32U575 or ST67W611M1.

The MicroUSB port is also used to read the UART messages from the STM32U575 in a serial terminal.

Flashing ST67W611M1
Mission Mode (Functional Mode) Setup

Prerequisites:

- Installed STM32CubeProgrammer
  - Installed in default directory (C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer
- Installed X-CUBE-ST67W61
  - Installed on C: drive (C:\x-cube-st67w61-v1.0.0)**



Using batch scripts located:
C:\ x-cube-st67w61-v1.0.0\Projects\ST67W6X_Utilities\Binaries

1. Nucleo-U575ZI-Q and X-Nucleo-67W61M1 stacked together

2. USB cable connected to microUSB port on Nucleo-U575ZI-Q and Windows PC.

3. Double-click the batch files to execute
   1. NCP_update_mission_profile.bat will flash the ST67W61M with the spi_wifi binary and the host processor with CLI application.

***Note: Path length limits and paths with spaces in them for the x-cube-st67w61-v1.0.0 may cause issues. Saving on C: drive will prevent these issues.*

Flashing ST67W611M1

Mission Mode (Functional Mode) Process

```
"#########################################################"
"# You are about to load a signed binary to the NCP."
"# This will lock the ST67W61M if not yet locked."
"#########################################################"
Are you sure to proceed? (Y/N)
Press Y to continue or N to abort:
```

1. When executing NCP_update_mission_profile.bat the script will prompt the user with a warning message about locking the NCP. This is expected, and all devices have previously been locked with older firmware at manufacturing. Please press “Y” to continue.

```
Opening and parsing file: Bootloader.bin


Memory Programming ...
  File          : Bootloader.bin
  Size          : 8.96 KB
  Address       : 0x08000000


Erasing memory corresponding to segment 0:
Erasing internal memory sectors [0 1]
Download in Progress:
                                                  100%

File download complete
Time elapsed during download operation: 00:00:00.168
```

```
[11:54:50.657] - Load efuse 0
[11:54:50.657] - Load efuse 1
[11:54:50.657] - Load efuse remainder
[11:54:50.657] - Finished
[11:54:50.657] - All time cost(ms): 16587.53369140625
[11:54:50.767] - close interface
[11:54:50.767] - [All Success]
```

```
Opening and parsing file: ST67W6X_CLI.bin


Memory Programming ...
  File          : ST67W6X_CLI.bin
  Size          : 186.91 KB
  Address       : 0x08000000


Erasing memory corresponding to segment 0:
Erasing internal memory sectors [0 23]
Download in Progress:
                                                  100%

File download complete
Time elapsed during download operation: 00:00:02.186
```

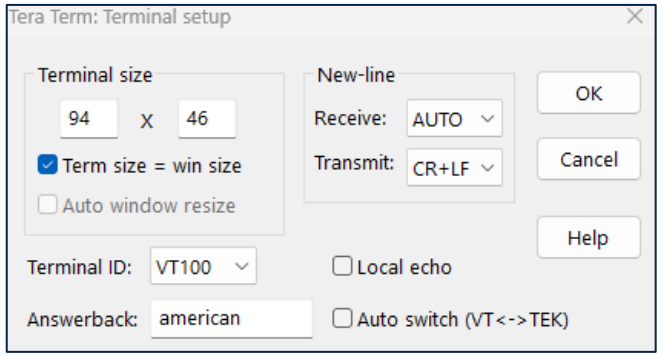
2. The NCP_update_mission_profile.bat will have 3 sequential flashing stages. The script will flash the STM32U5 with Bootloader.bin, then the ST67 with the signed mission mode binary, and the STM32U5 with ST67W6X_CLI.bin. Please verify success of all 3 flashing stages.

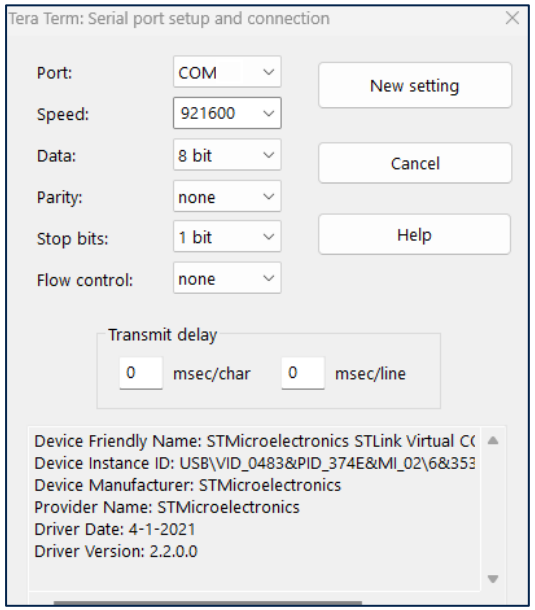
Flashing ST67W611M1

Mission Mode (Functional Mode) Verification

Flashing success can be verified by connecting to the Nucleo-U575ZI-Q with a serial terminal.

Configure the serial terminal with the following parameters and connect to the corresponding COM port for the Nucleo-U575ZI-Q.





With the serial terminal program configured, press the reset button the Nucleo-U575ZI-Q and verify application and SDK version.

```
    >#### Welcome to ST67W6X CLI Application #####
# build: 22:39:24 May 28 2025
--------------- Host info ---------------
Host FW Version:          1.0.0
-------------- ST67W6X info -------------
ST67W6X MW Version:       1.0.0
AT Version:               1.0.0.1
SDK Version:              2.0.75
MAC Version:              1.6.38
Build Date:               May 20 2025 23:01:38
Module ID:                C6AFDBD111400004
BOM ID:                   1
Manufacturing Year:       2024
Manufacturing Week:       47
Battery Voltage:          3.334 V
Trim Wi-Fi hp:            6,6,6,6,6,6,5,5,6,6,6,7,7,7
Trim Wi-Fi lp:            7,7,8,8,8,9,9,9,10,10,10,11,11,11
Trim BLE:                 5,4,5,6,7
Trim XTAL:                37
MAC Address:              40:82:7b:00:33:26
Anti-rollback Bootloader: 0
Anti-rollback App:        0
-----------------------------------------
mount success
Wi-Fi init is done
Net init is done
MQTT init is done
Starting FOTA task
ready
Application started from bank 1
```

***Note: "Manufacturing Year" date may be incorrect and populated with the year "2000". This will have no effect on the functionality.*

life.augmented

How to use X-CUBE-ST67W61 within STM32CubeMX

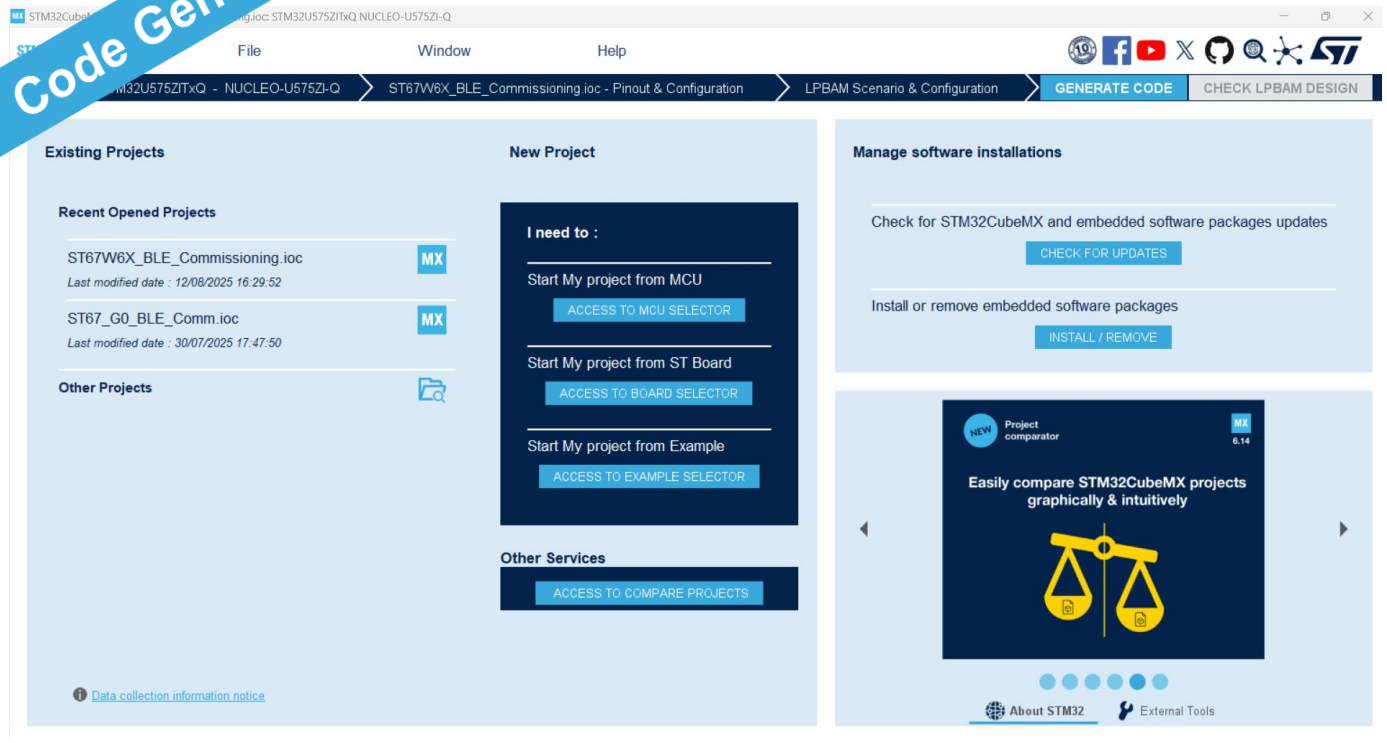
Software Development Tools

“STM32CubeIDE”
development environment

“STM32CubeMX”
initialization code generator

“X-CUBE-ST67W61”
application code and drivers for Wi-Fi

STM32CubeMX
an All-in-1 Development Tool



Very powerful configuration and code generation

Support all STM32 MCU and MPU, with integrated powerful Finder

Pinouts, clock tree, peripherals and middleware configuration.

Expandable to support wireless connectivity, sensors and much more!

Three possible cases to generate the code from the Cube MX:

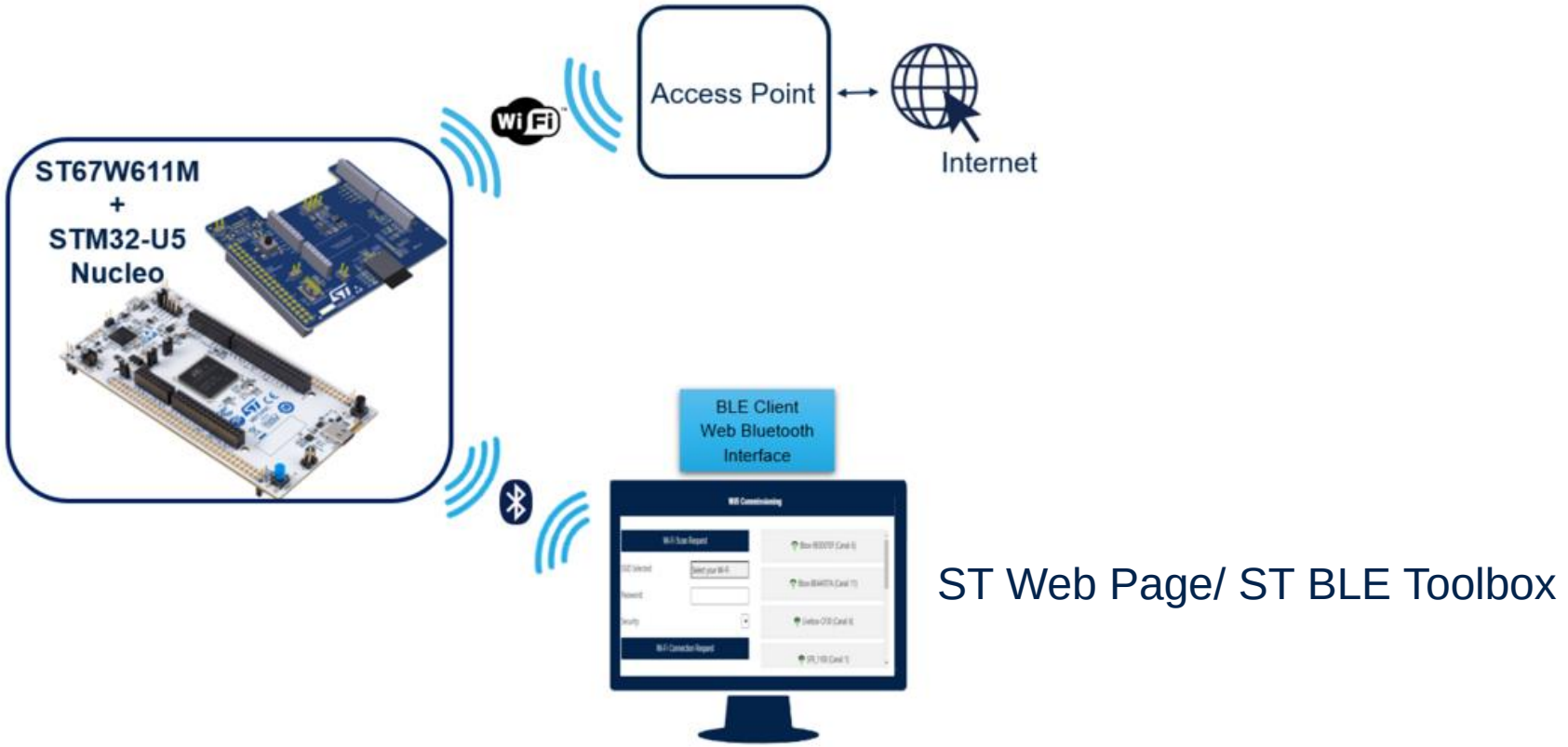
- **Case 1**: How to generate a project starting from an existing .ioc
- **Case 2**: How to export a project to a pinout compatible MCU
- **Case 3**: How to generate a project starting from scratch

In this section: We would be having **hands-on** to generate **BLE Commissioning** code using the following:

- Case 1 with NUCLEO-U575ZI-Q as host processor
- Case 3 with NUCLEO-G0B1RE as host processor

BLE Commissioning App

- This application aims to demonstrate **how to provision Wi-Fi credentials via Bluetooth® LE** to establish a Wi-Fi connection to an access point.



ST Web Page/ ST BLE Toolbox

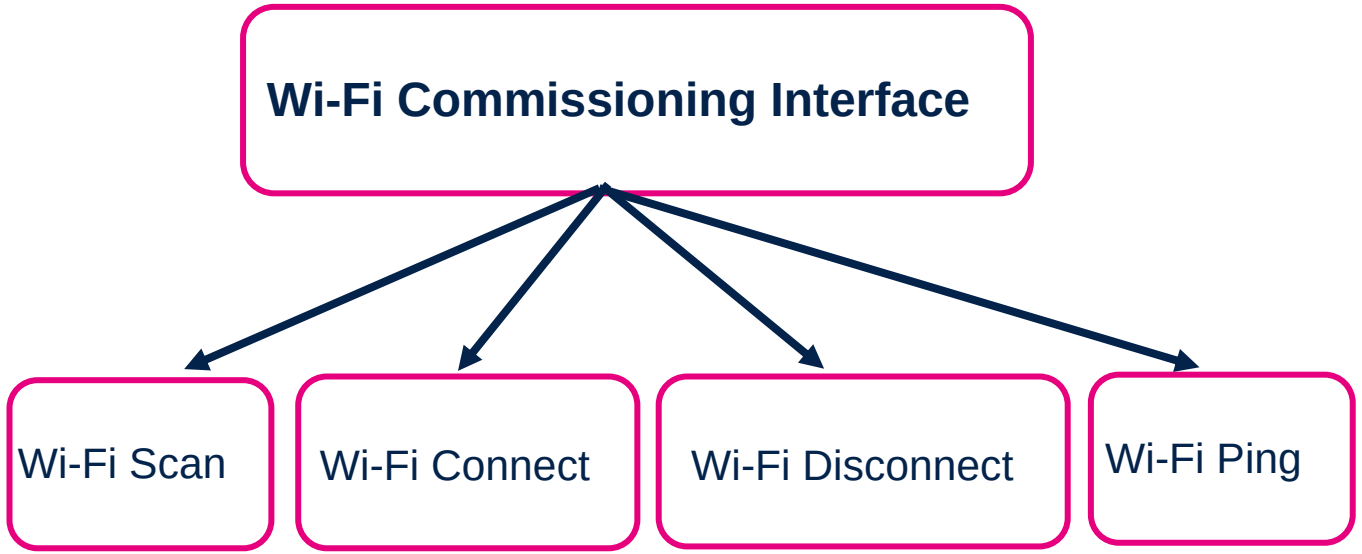
BLE Commissioning App cont..

The application acts as a peripheral device embedding the commissioning profile and its four characteristics.

At startup, the application starts to advertise.

ST Web Bluetooth® Interface must be used to scan and connect to the commissioning application: (Web-Interface available from the browser) / ST BLE Toolbox (Mobile Application)

Once connected, a Wi-Fi commissioning interface is available on web Bluetooth® page/ ST BLE Toolbox



Commissioning profile overview

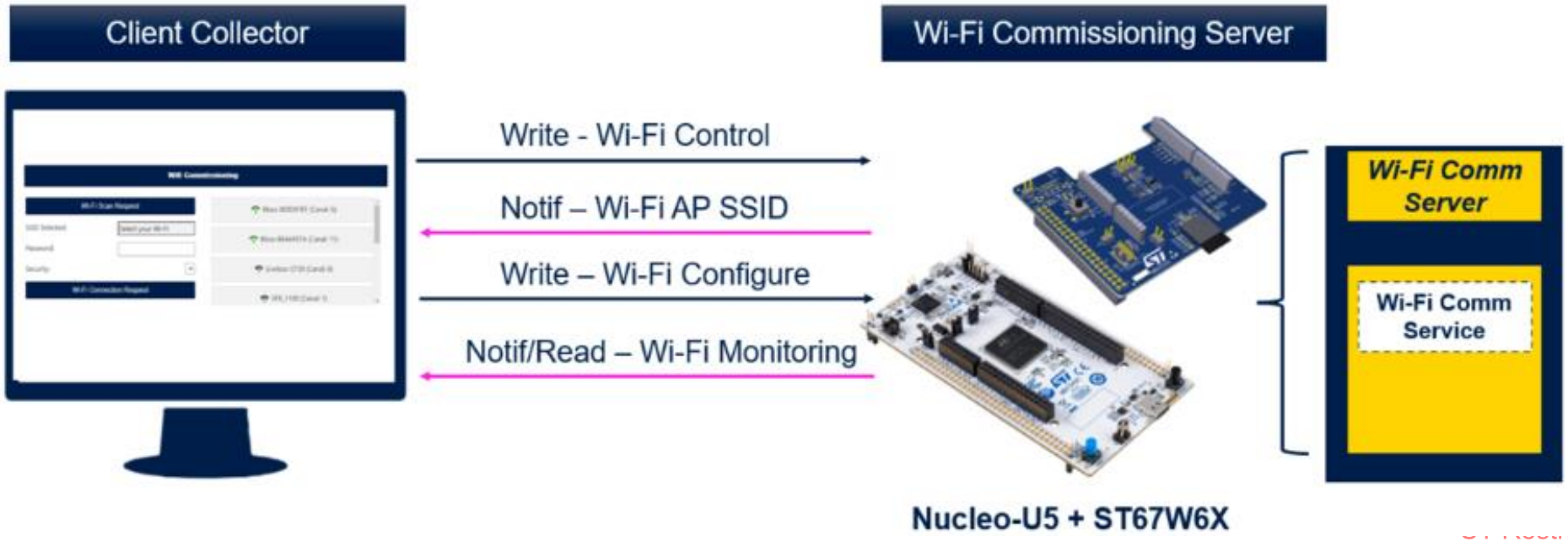
The Wi-Fi commissioning application demonstrates point to point communication using proprietary service and characteristics. It acts as a Peripheral device with one GATT service and four characteristics.

Wi-Fi commissioning server:

- Contains the Wi-Fi commissioning service, which exposes four characteristics: **Wi-Fi Control (write), Wi-Fi Configure (write), Wi-Fi Access Point List (notification), Monitoring (Notification)** to control and monitor a Wi-Fi connection.
- Is the GATT server

Client Collector: (STBLE Toolbox/ ST Web Bluetooth page)

- Accesses the information exposed by the Wi-Fi commissioning application, **configures and controls the Wi-Fi connection** with the write characteristics, **monitors the Wi-Fi connection status** by receiving notifications from it
- Is the GATT client



The table below describes the structure of the commissioning service:

Bluetooth® LE commissioning service specification

Service	Characteristic	Mode
Wi-Fi commissioning		
	Wi-Fi Control	Write with Response/Read
	Wi-Fi Configure	Write with Response/Read
	Wi-Fi AP List	Notify
	Monitoring	Notify

Wi-Fi commissioning - Wi-Fi control

Byte Index	0
Name	Action
Value	0x01: Wi-Fi Start Scan 0x03: Wi-Fi Connect 0x04: Wi-Fi Disconnect 0x05: Wi-Fi Ping

Wi-Fi control characteristic:
Used to drive the Wi-Fi by launching scans, connecting or disconnecting from a network

Wi-Fi commissioning - Wi-Fi configure

Byte Index	0	1
Name	Type	Data[ ]
Value	0x01: AP-SSID 0x02: PWD	"ASCII" "ASCII"

Wi-Fi configure characteristic:
Used to setup the Wi-Fi parameters before establishing a connection

Wi-Fi Commissioning - Wi-Fi AP List

Byte Index	0	1	2 to 3	4 to 7	8 to 40
Name	SSID Length	Channel	Signal level	Security Flag	SSID
Value	0x00 ... 0x20	0x00 ... 0xFF	0x00 ... 0xFFFF	Security_Flag[ ]	Data[ ]

Wi-Fi access point list characteristic:
Used to notify information about scanned access points

Wi-Fi commissioning - Wi-Fi monitoring

Byte Index	0	1 up to 239
Name	Type	Data[ ]
Value	0x03: Connecting 0x04: Connection established 0x05: Ping response 0x06: Error	X SSID Refer to **table below** 0x01: Connection Timeout

Monitoring characteristic:
Used to notify information about Wi-Fi network.

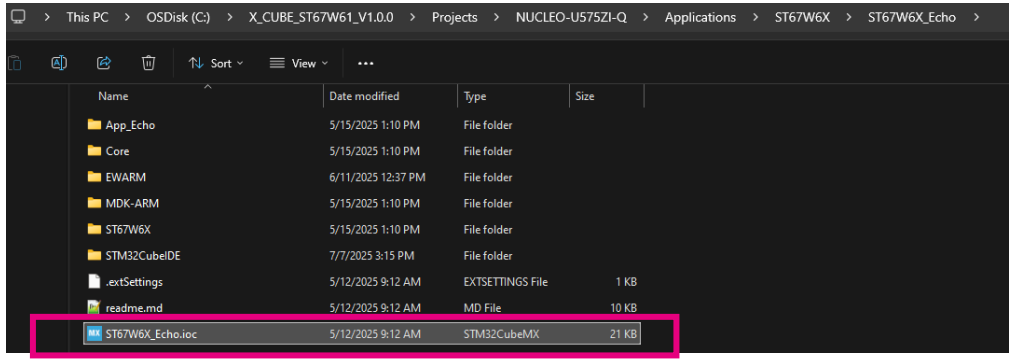
life.augmented

Generating BLE Commissioning app using NUCLEO-U575ZI-Q as host processor (Hands-on)

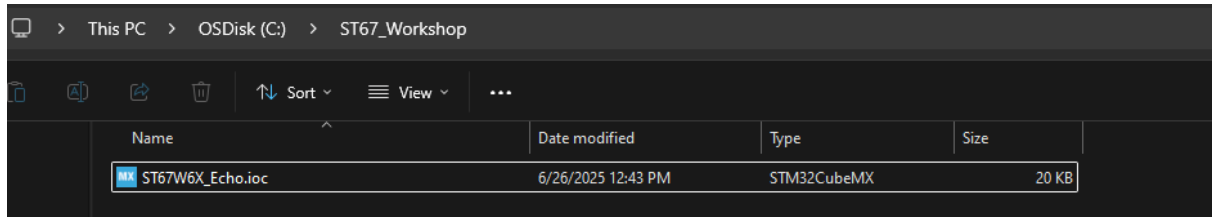
Case 1: Generating BLE Commissioning app using NUCLEO-U575ZI-Q as host processor

We would start with the ST67W6X_Echo.ioc file as a starting point and change the configuration to generate the BLE Commissioning application from the STM32CubeMX.

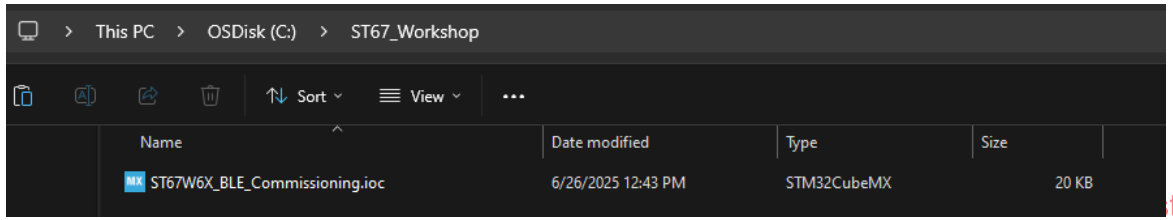
1. Browse through C:\X_CUBE_ST67W61_V1.0.0\Projects\NUCLEO-U575ZI Q\Applications\ST67W6X\ST67W6X_Echo
2. Copy ST67W6X_Echo.ioc
3. Create a new directory C:\ST67_Workshop
4. Paste and rename to ST67W6X_BLE_Commissioning.ioc



Copy and Paste



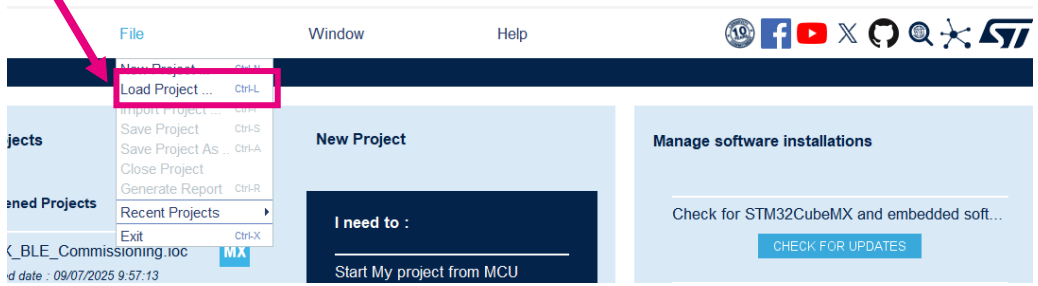
Rename



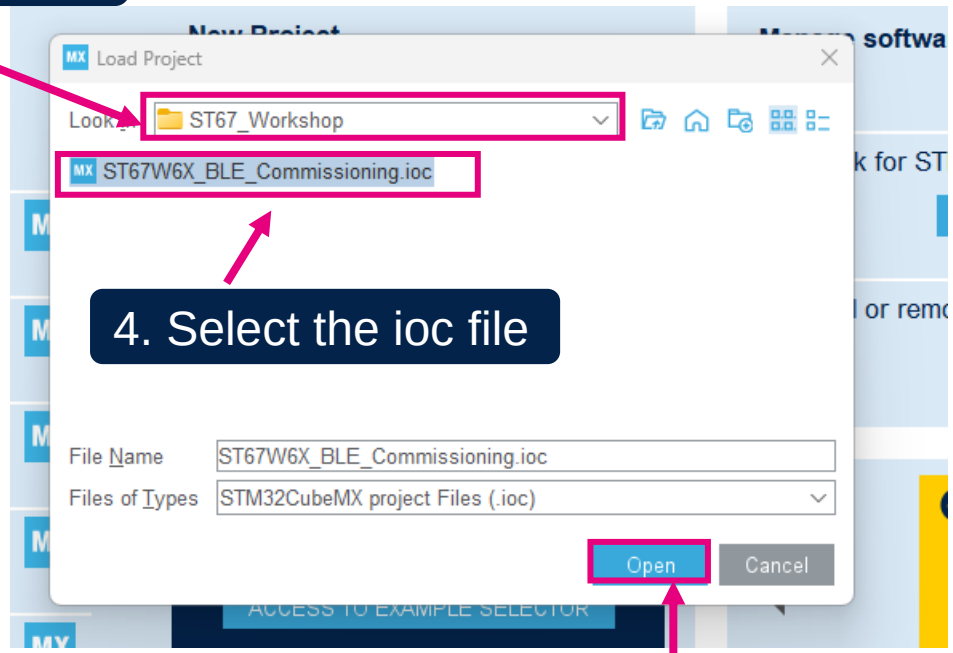
Loading ioc file in STM32CubeMX

1. Run STM32CubeMX with **admin rights**

2. Load Project

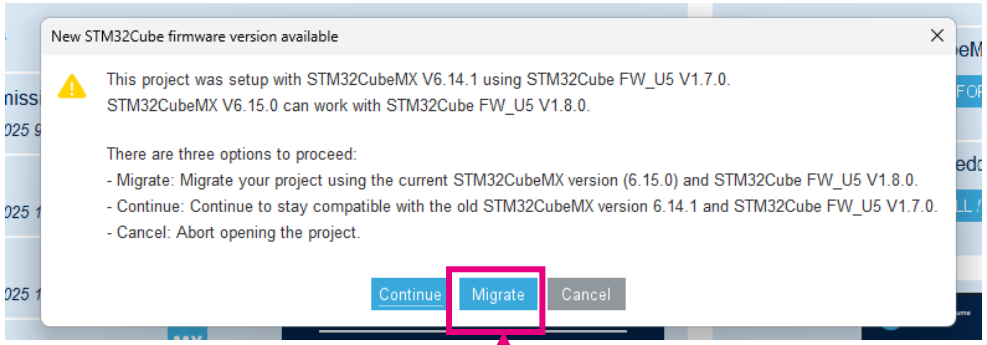


3. Go the ST67_Workshop

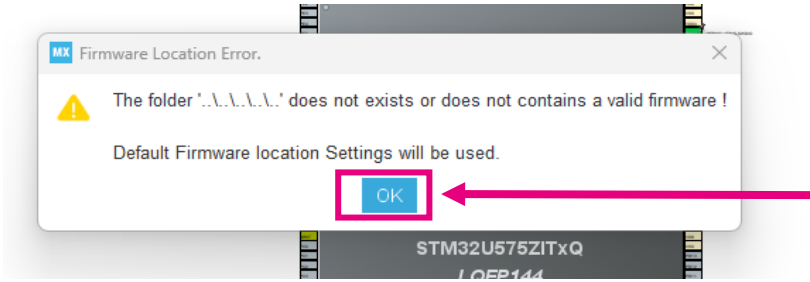


4. Select the ioc file

5. Click open

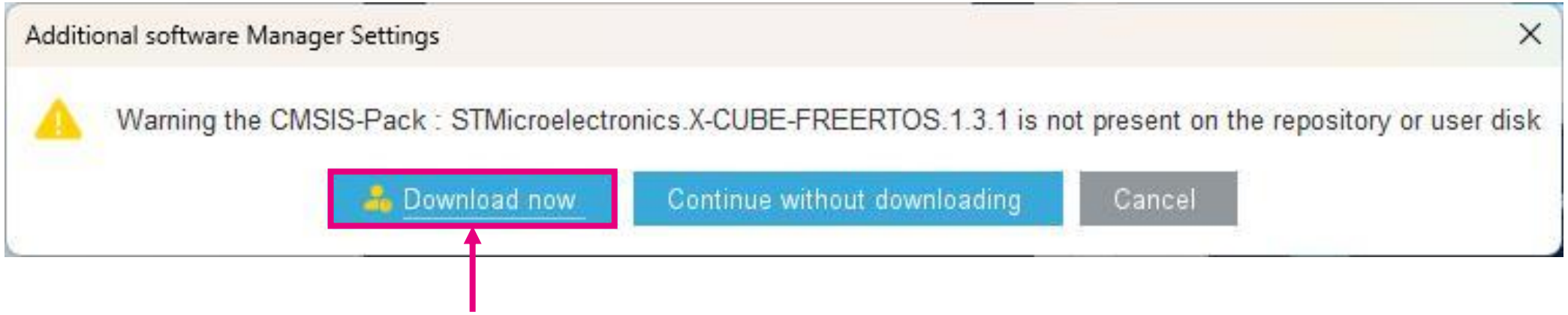


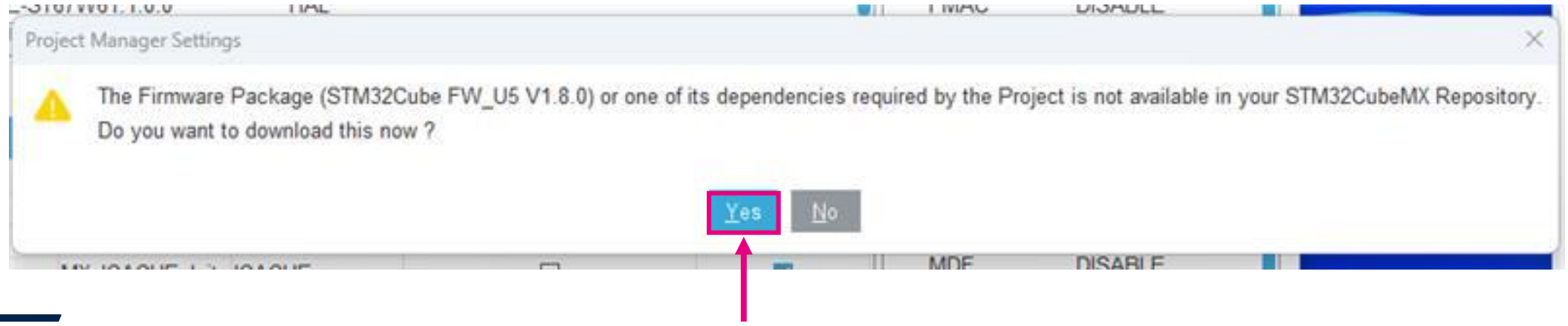
6. Select Migrate



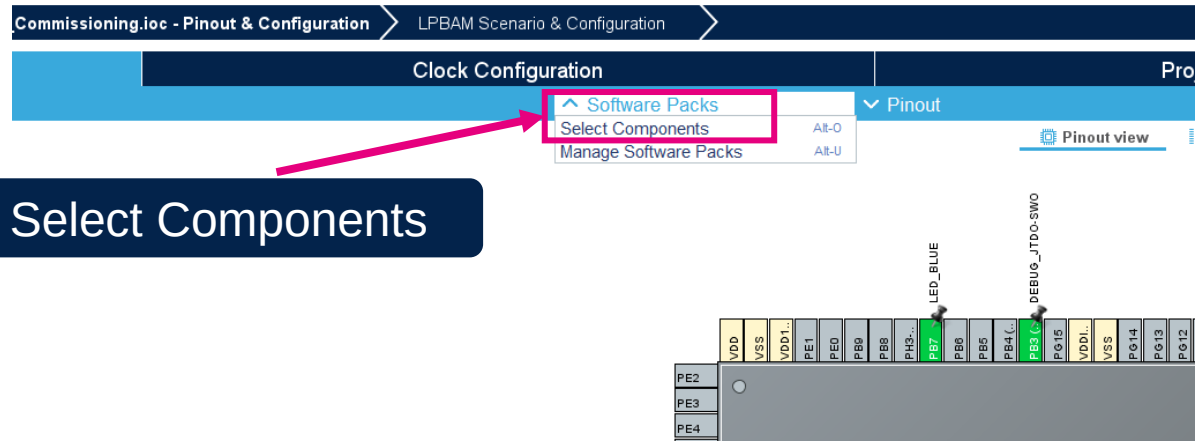
7. Click Ok

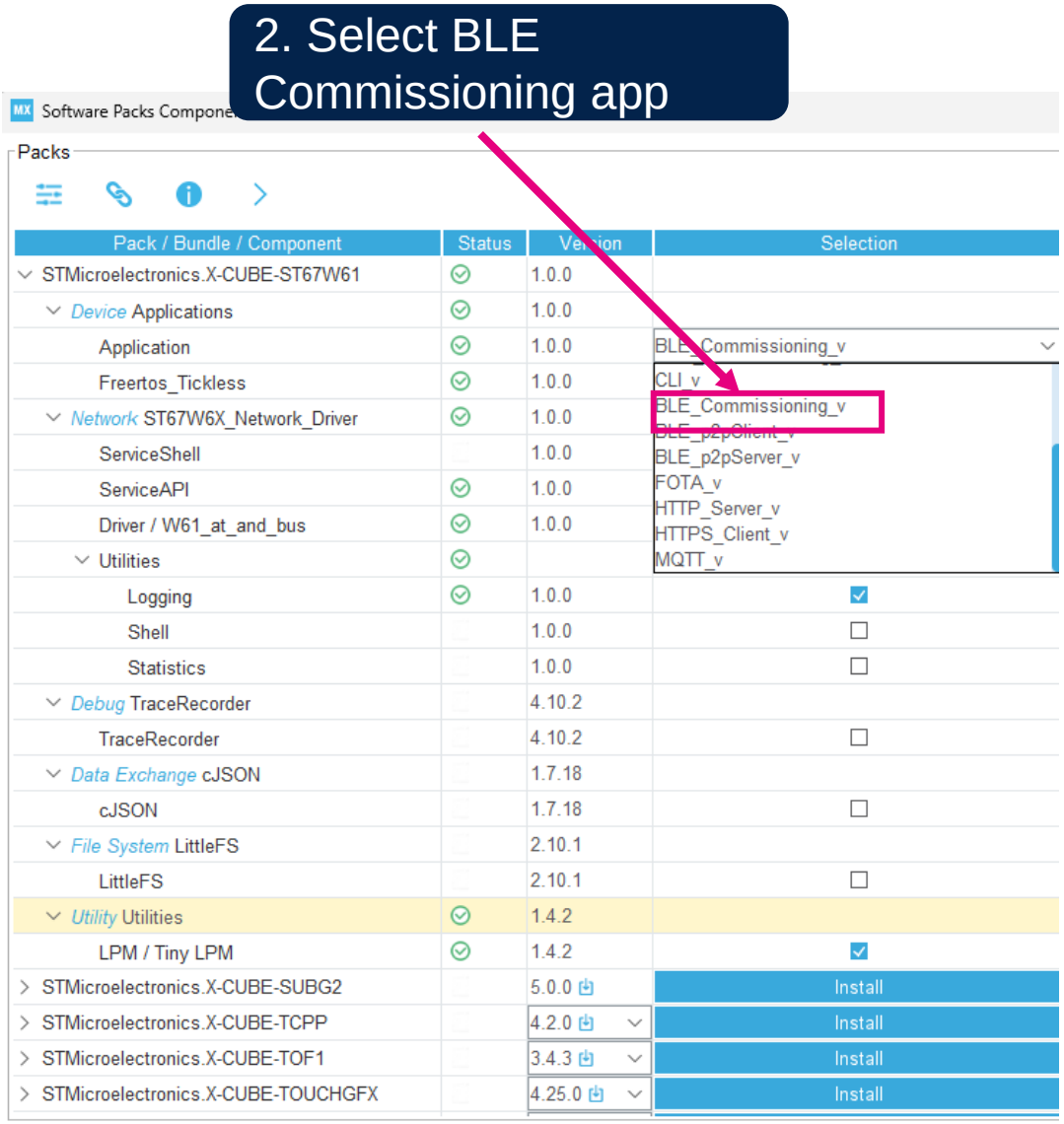
If you get the below warnings





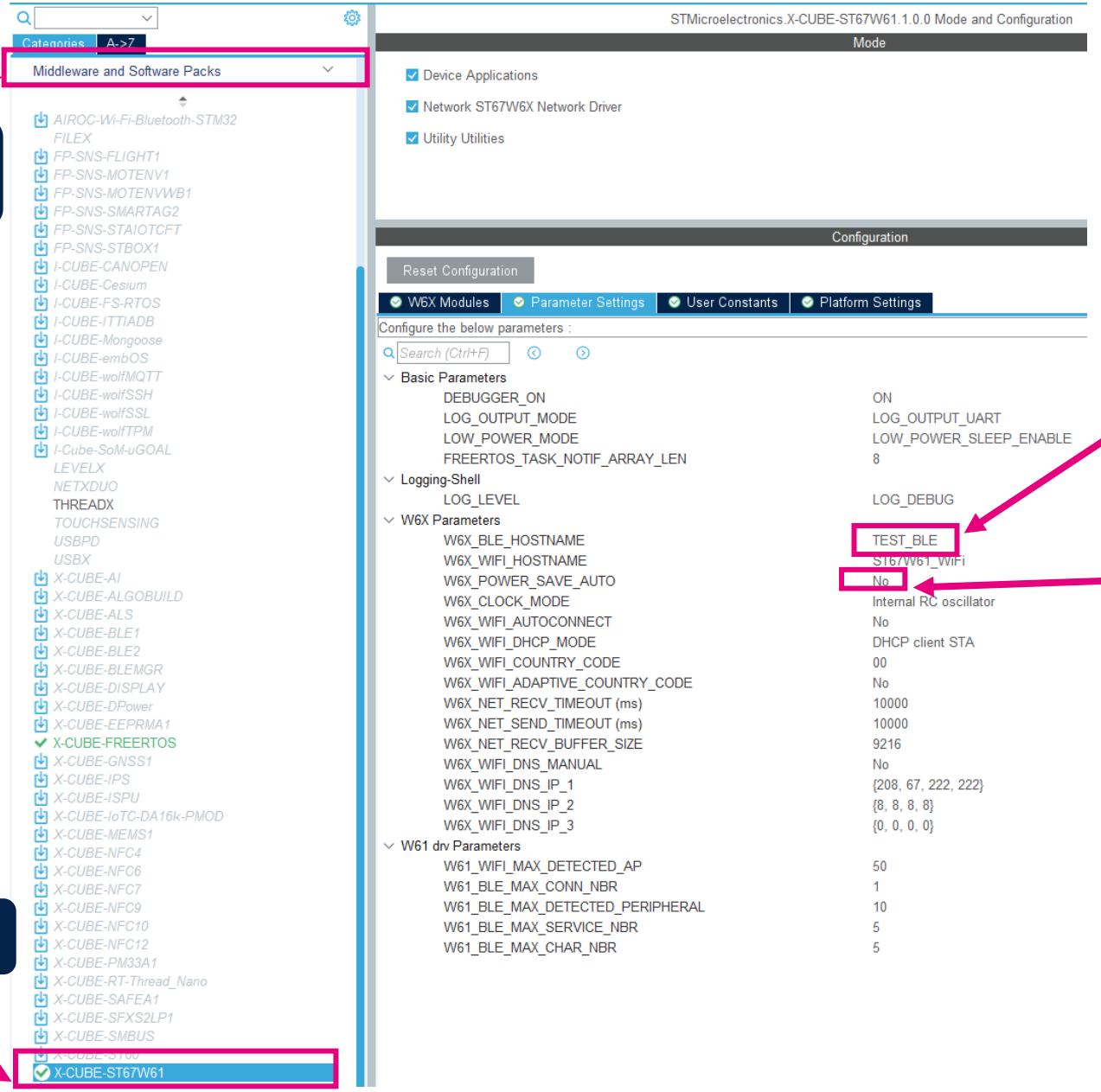
Configuration





Configuration

3. Under Middleware and Software Packs



4. X-CUBE-ST67W61

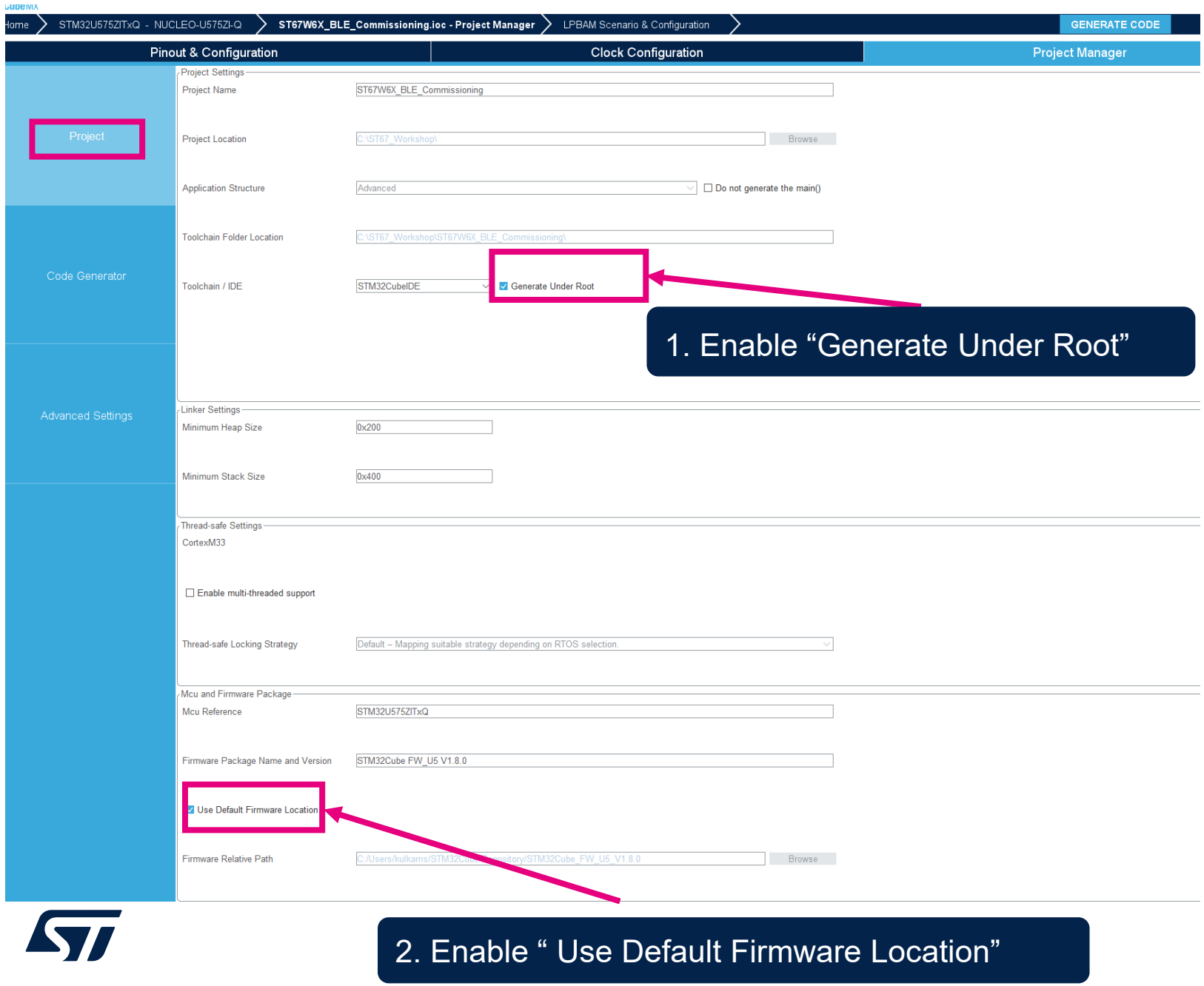
5.Rename W6X_BLE_HOSTNAME to something unique

6. Disable W6X_POWER_SAVE_AUTO

Bluetooth Low Energy connection with NCP in Power save mode requires accurate external low-frequency clock.
To setup and use external clock, SW and HW settings must be modified.

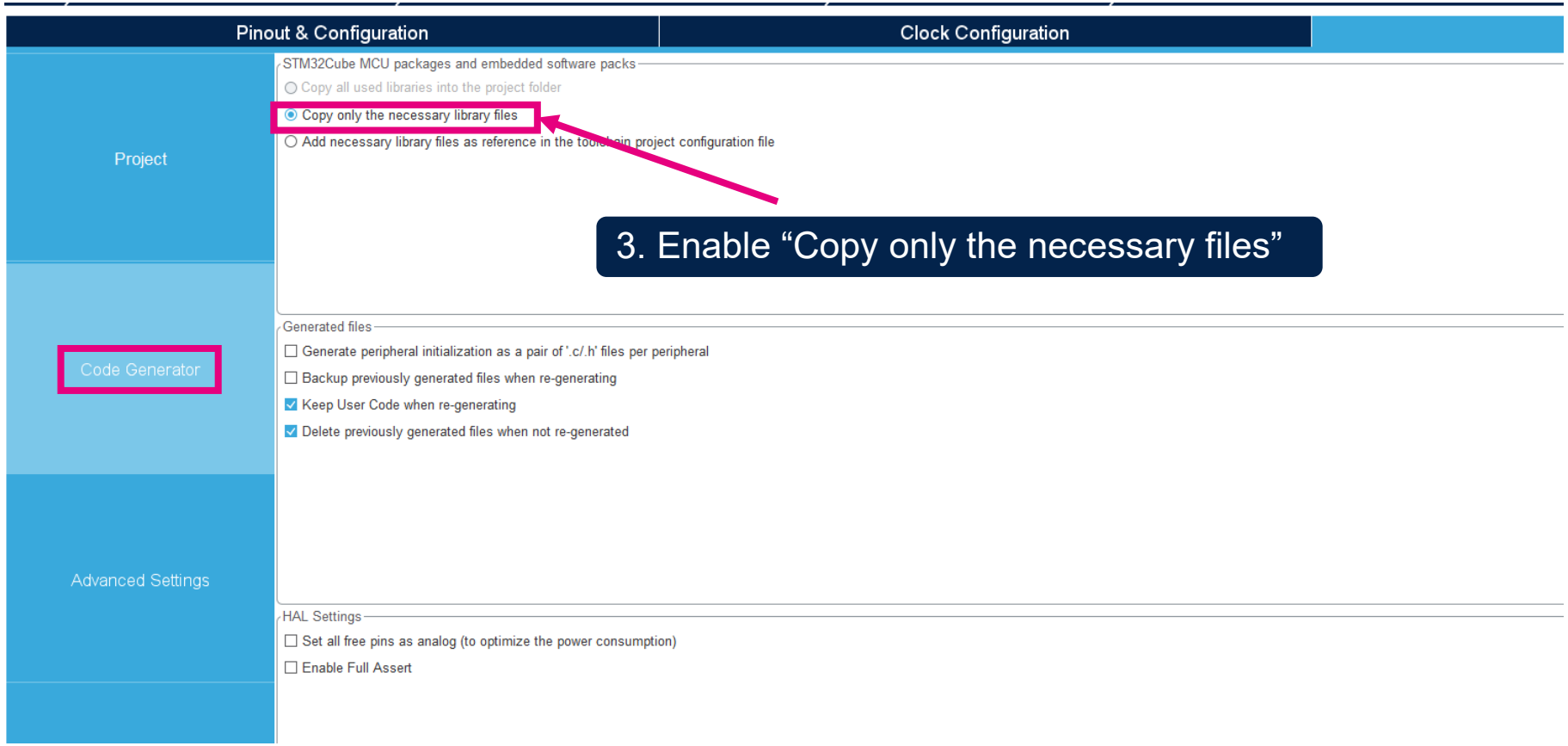
Refer to [Wiki Bluetooth LE with low-power setup](https://wiki.st.com/stm32mcu/wiki/Connectivity:How_to_measure_ST67W611M_current_consumption#Bluetooth-C2-AE_LE_with_low-power_setup) page to be informed about required changes.

Project Settings

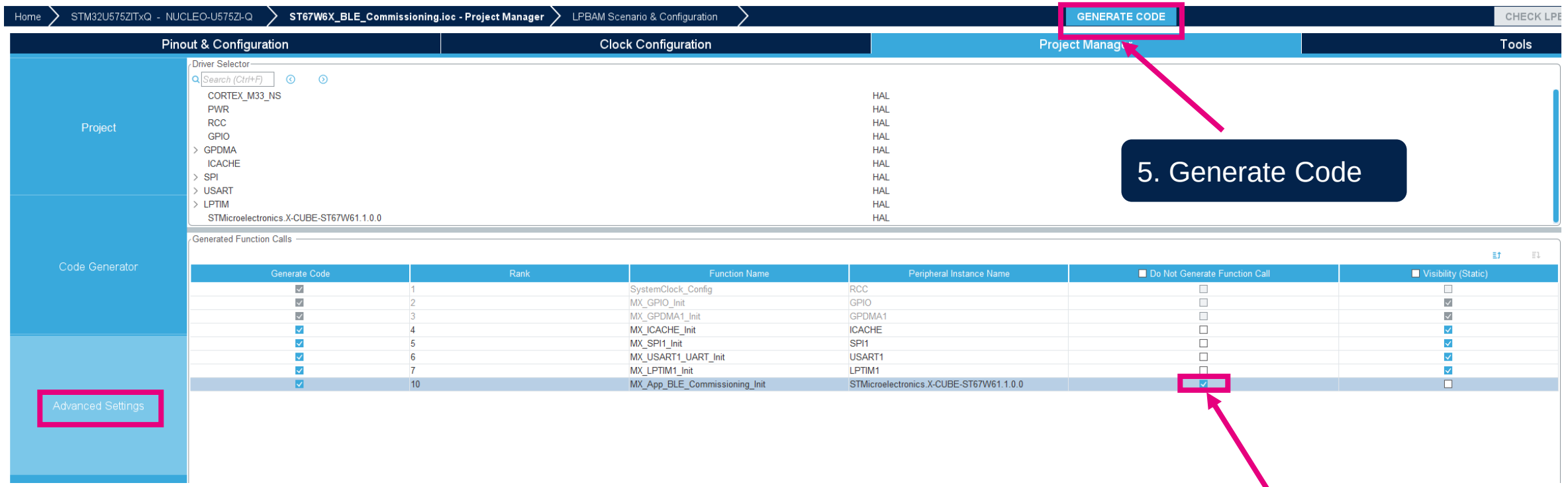


Here, we are enabling the Flat directory structure: (i.e. Driver and MW directories copied into your project directory)

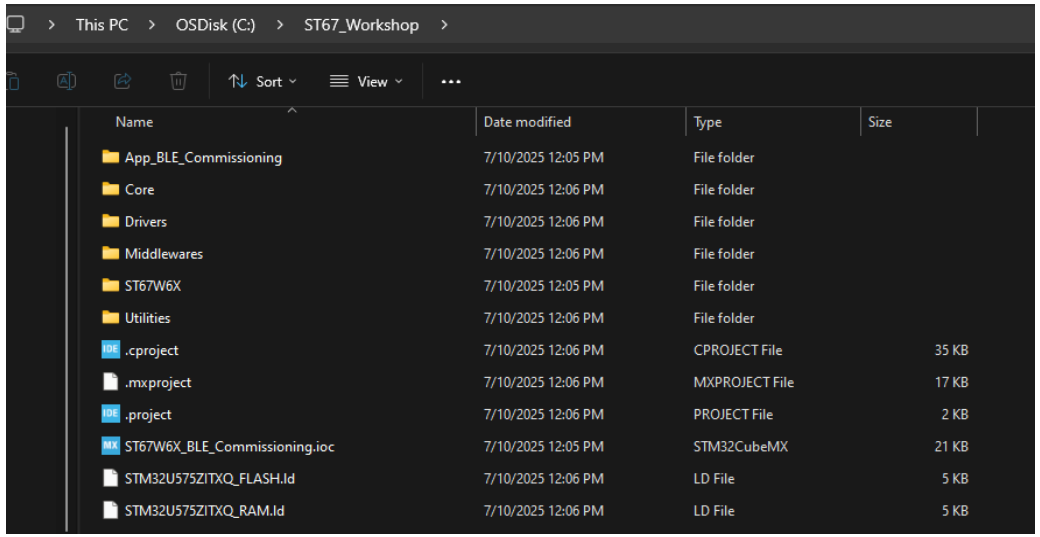
Project Settings cont..



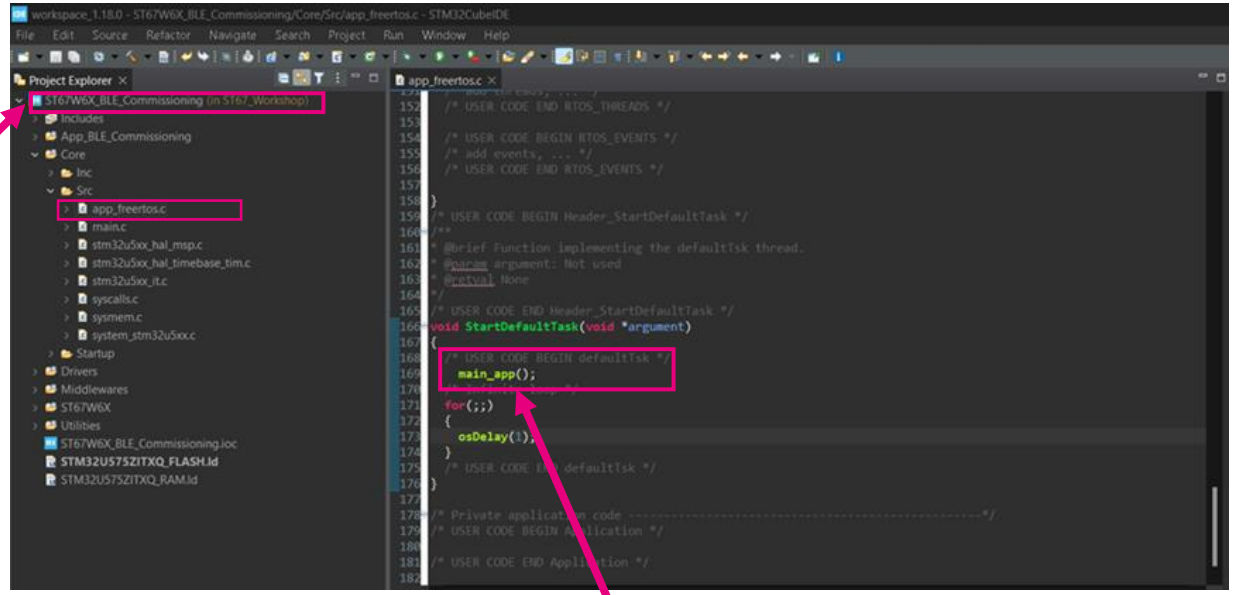
Generate Code



STM32CubeIDE



After, generating the code, you would view similar folder structure under ST67_Workshop



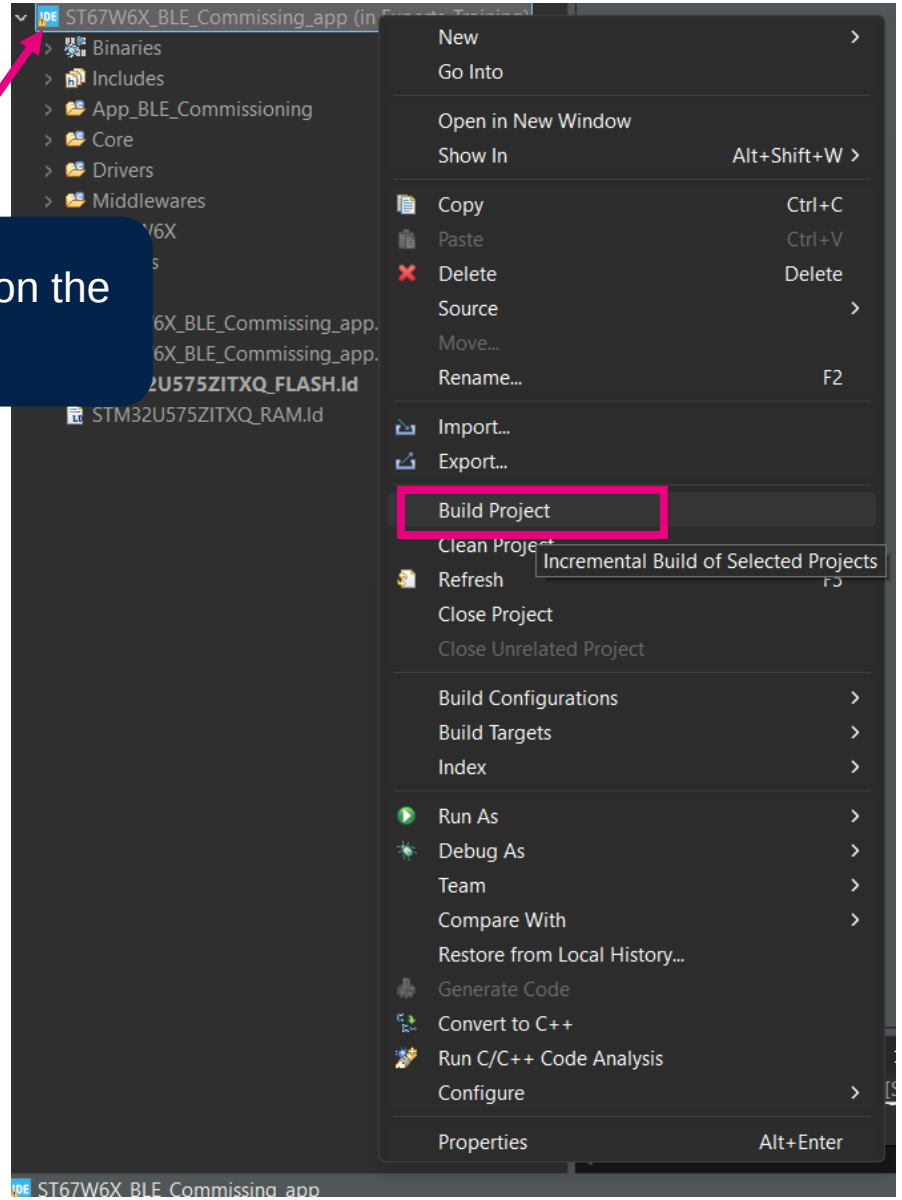
1. Load the project on the STM32CubeIDE

```
/* Includes ------------------------------------------------------------------*/
#include "app_freertos.h"
#include "main_app.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
```

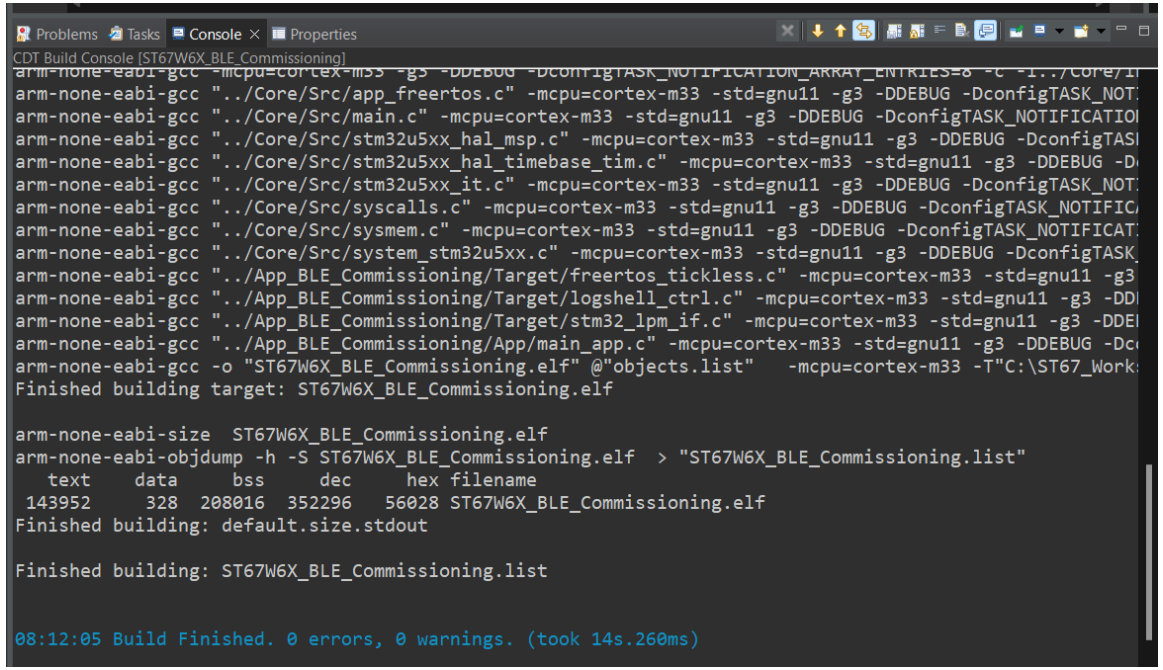
2. Include main_app.h in app_freertos.c file
#include “main_app.h”

3. Call main_app() function under StartDefaultTask() in app_freertos.c file

Time to build

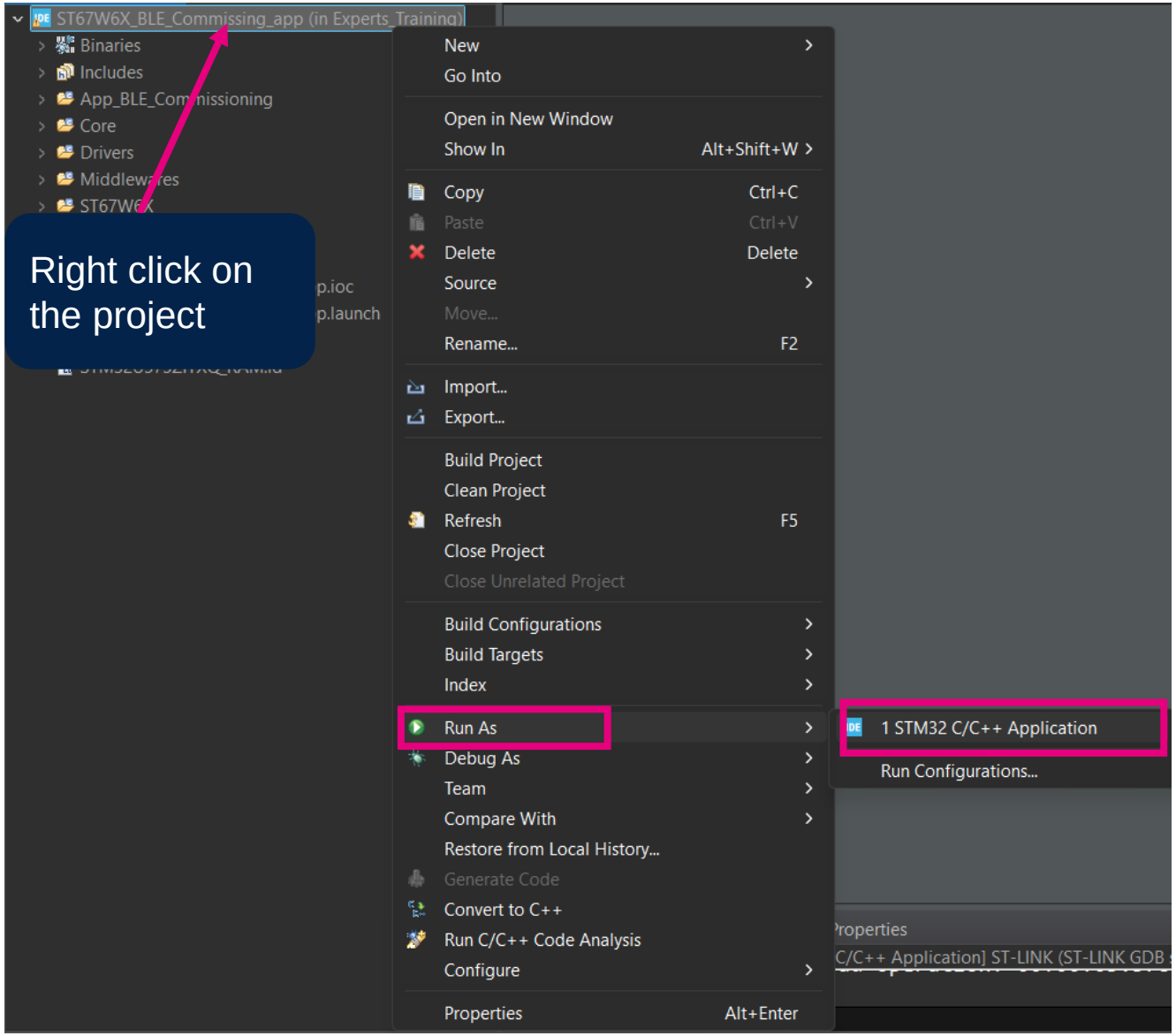


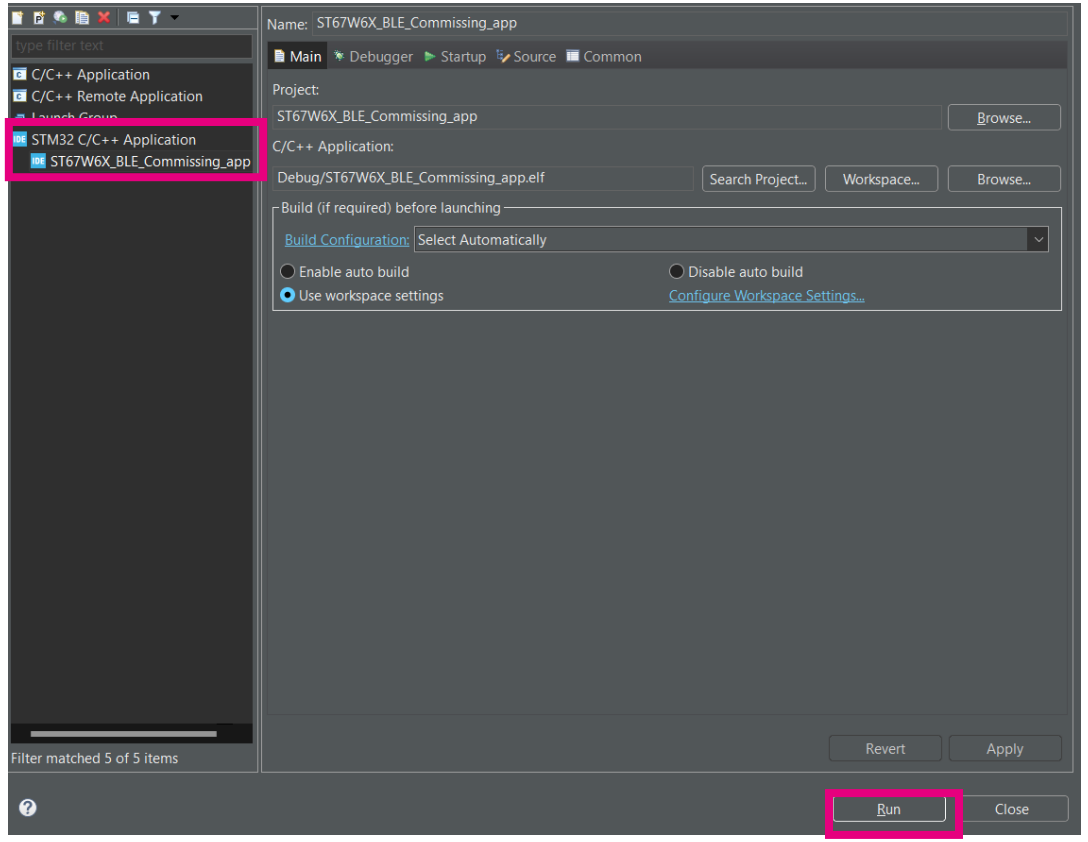
Right click on the project



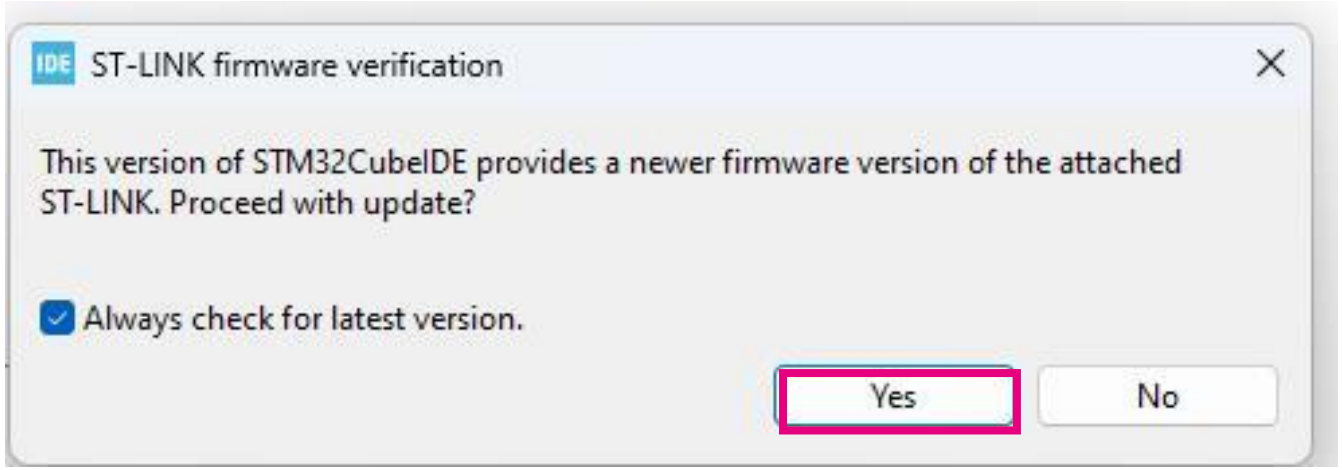
Console after building the application

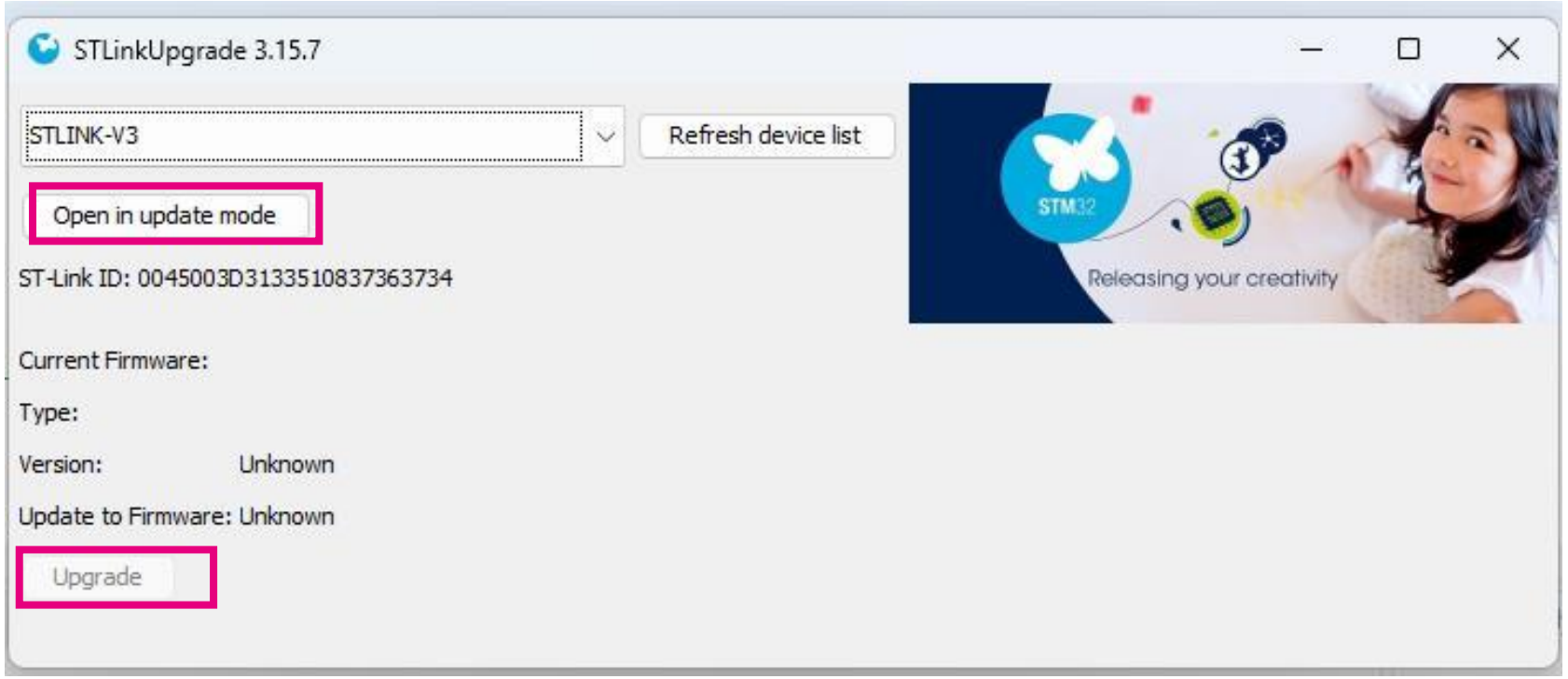
Flash the project





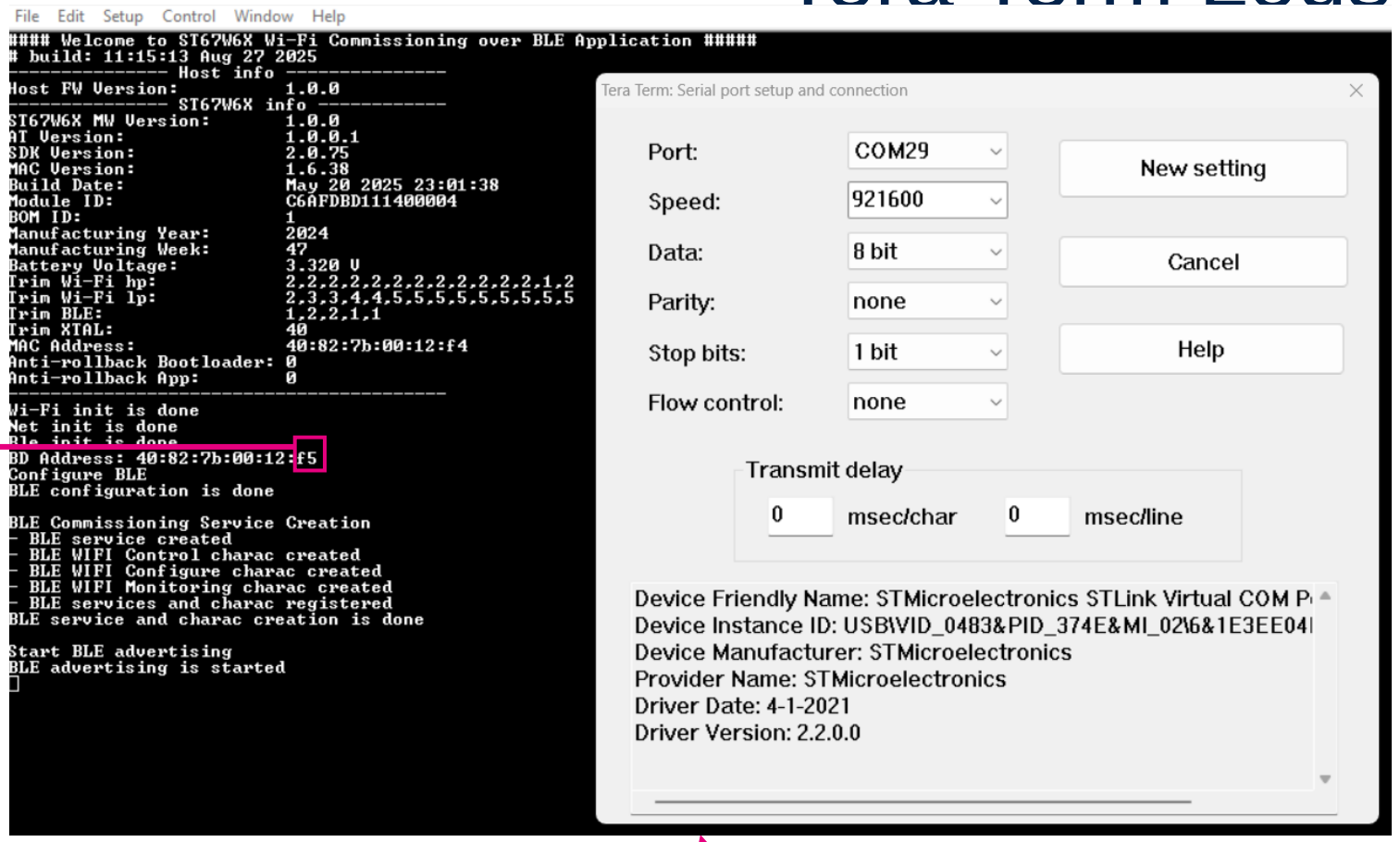
ST-LINK Update Warning





Tera Term Logs

Last 1 byte of BD_Address would match last byte advertising on the ST_BLE_Toolbox



Tera Term logs after flashing the BLE Commissioning application

ST BLEToolbox/ ST BLE Webpage

To interface with the Wi-Fi commissioning embedded application, you can use STBLEToolbox Andoid/iOS application / ST Web Bluetooth® Interface

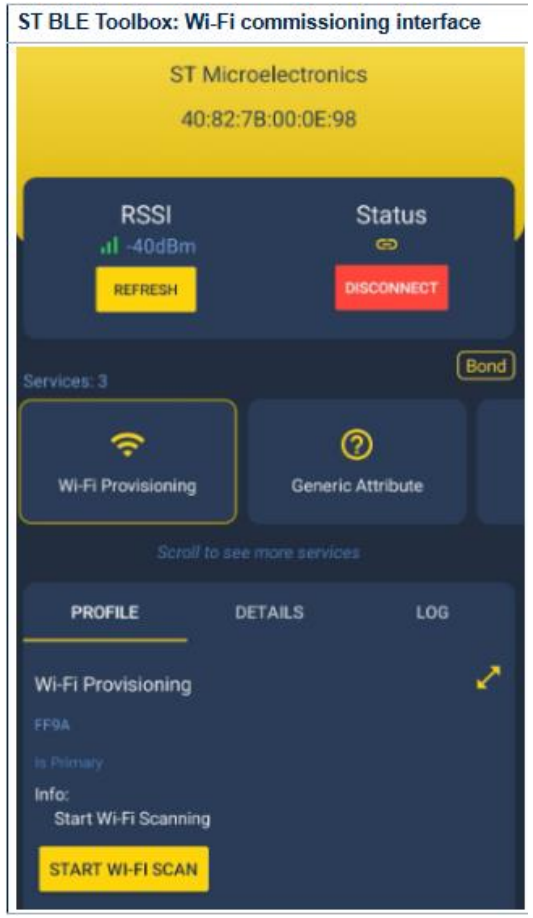
Download **STBLEToolBox**

The versions supporting Wi-Fi commissioning service are **v1.4.3 for Android and 1.2.8 for iOS**

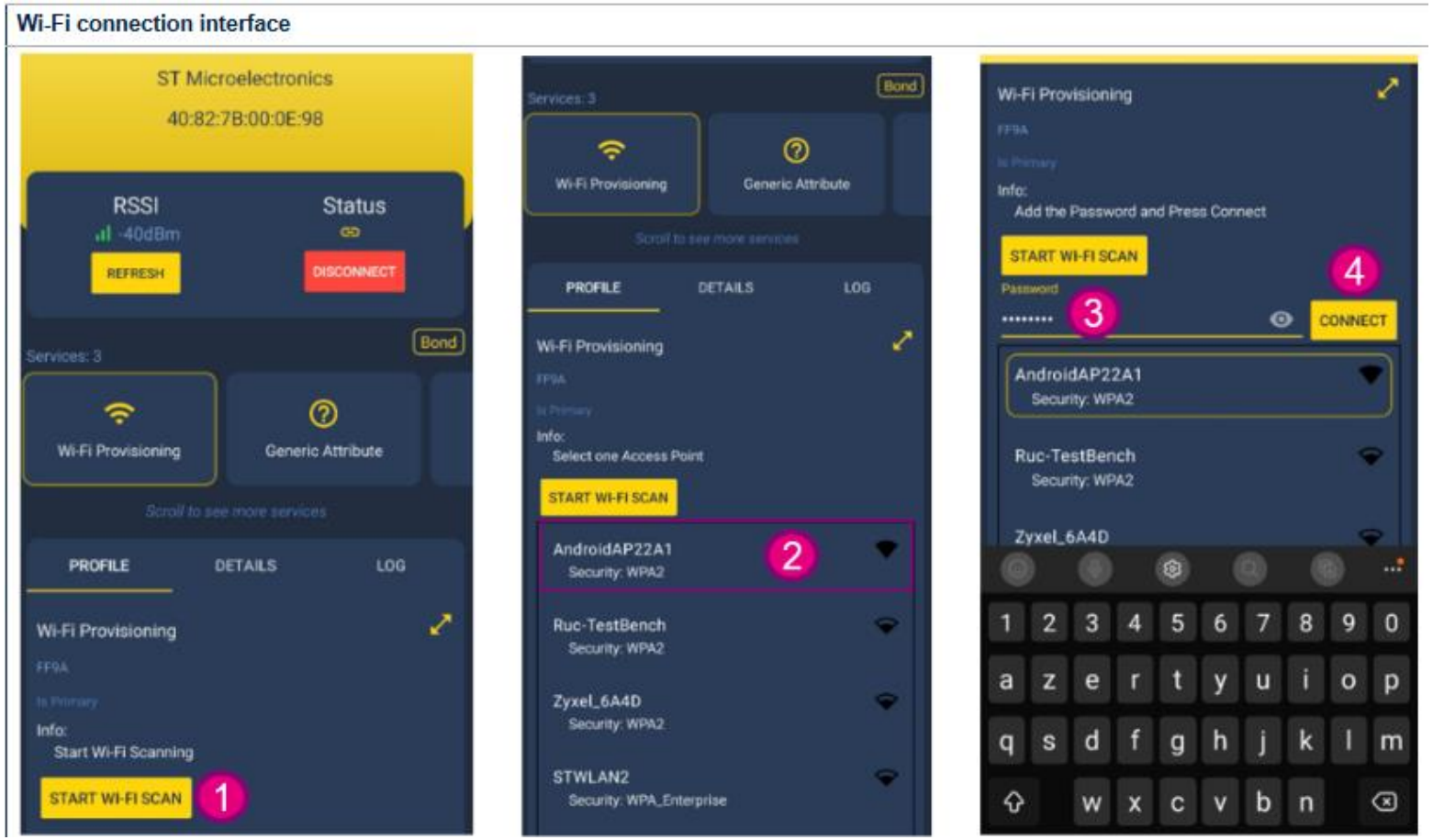
Steps to launch and connect to Wi-Fi AP:

1. Launch the smartphone application and enable the scan button, if it is not already activated.
2. You can filter scanned devices by clicking on the SHOW FILTERS button and select ST BLE Wi-Fi to show only devices with Wi-Fi commissioning service.
3. Select your device, named ST_WiFi_XX where XX represents the last BD address digit, and connect to it**.**

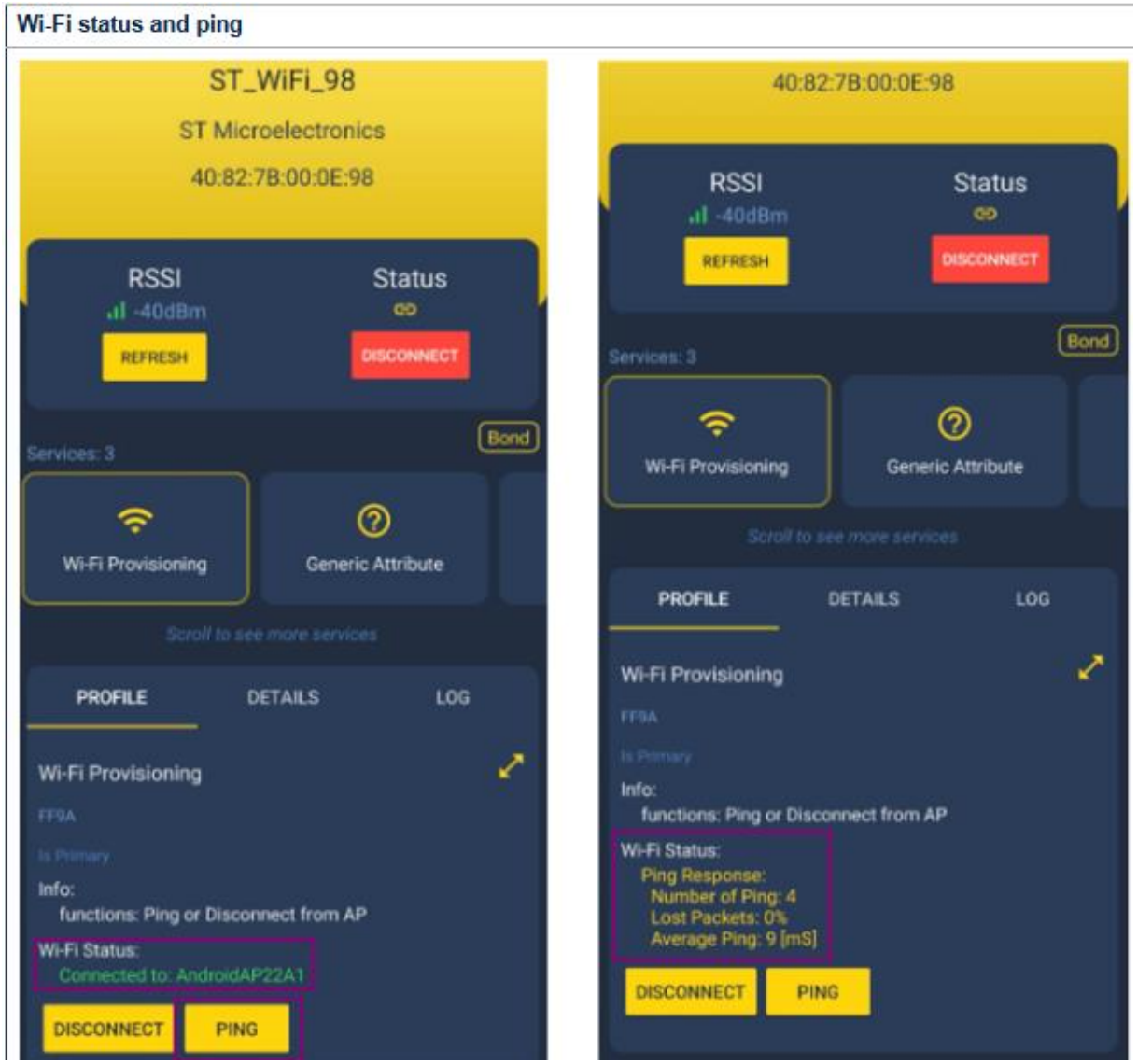
4. Once connected to the Bluetooth® LE device, click on Wi-Fi commissioning button to open the commissioning dedicated interface.



From this interface,
Launch a Wi-Fi scan (1),
To detect and list all available networks(2).
Once the available networks are displayed, select the one you want to connect to (2)
Type a password if needed (3)
Click on CONNECT button (4).

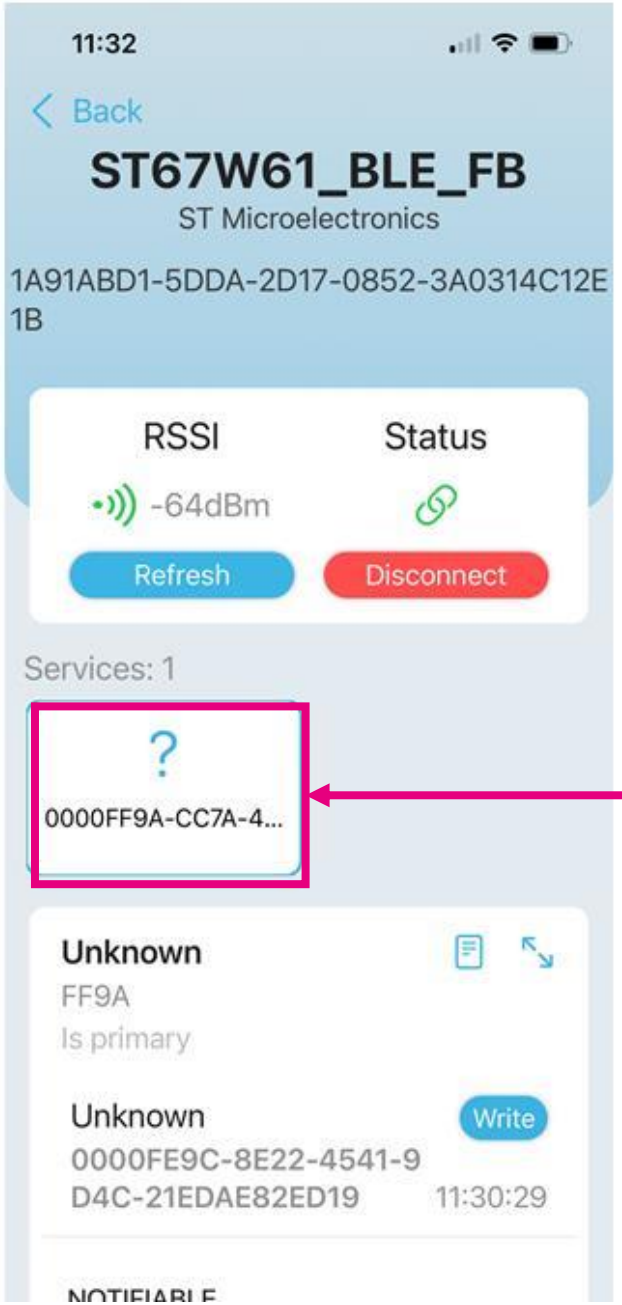


Once the connection request is done, the connection status is displayed in the interface.
Clicking on the Ping button start a ping test with the Wi-Fi access point and returns results in the interface.



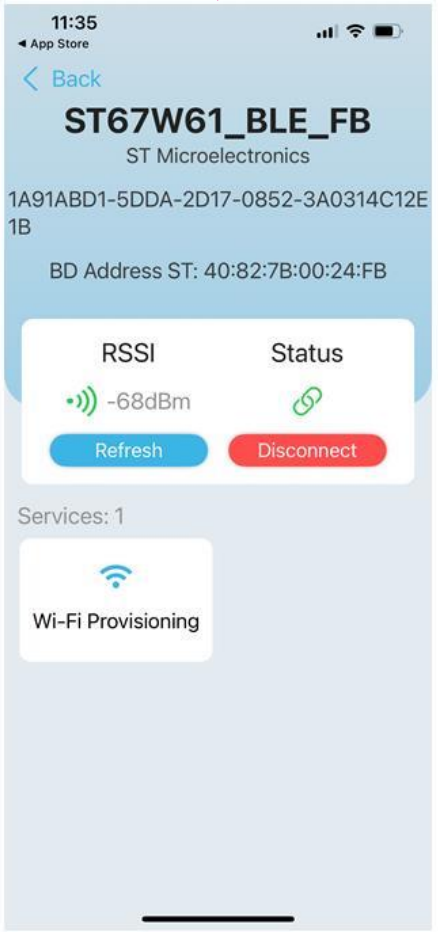

```
BLE service and charac creation is done

Start BLE advertising
BLE advertising is started
BLE connected
 -> BLE CONNECTED: Conn_Handle: 0
BLE MTU exchange
BLE notification enabled
 -> BLE NOTIFICATION ENABLED [Service: 0, Charac: 2].
BLE write
 -> BLE WRITE.
 -> Conn_Handle: 0, Service: 0, Charac: 0, length 1, Data:
0x01
WIFI Control Charac
WIFI Scan Enable
WiFi Scan done.
MAC : [ac:8f:a9:df:eb:c4] | Channel:  6 |          WPA2 | RSSI:  -35 | SSID:  Yggdrasil
MAC : [bc:9a:8e:c0:70:94] | Channel:  6 |          WPA2 | RSSI:  -40 | SSID:  ATTAAAA
MAC : [b4:63:6f:aa:22:a4] | Channel: 11 |          WPA2 | RSSI:  -42 | SSID:  DesiWifi
MAC : [ac:db:48:72:1e:a4] | Channel:  1 |          WPA2 | RSSI:  -49 | SSID:  canyouseewhereIP
MAC : [bc:9a:8e:fa:34:94] | Channel: 11 |          WPA2 | RSSI:  -59 | SSID:  HUST_WIRELESS5G
MAC : [00:e6:3a:44:8b:52] | Channel: 11 |          WPA2 | RSSI:  -61 | SSID:  Vista99_WiFi
MAC : [00:e6:3a:44:8b:51] | Channel: 11 |      WPA-EAP | RSSI:  -61 | SSID:  Passpoint
MAC : [00:e6:3a:44:8b:54] | Channel: 11 |          WPA2 | RSSI:  -61 | SSID:  Vista99_Staff
MAC : [00:e6:3a:44:8b:50] | Channel: 11 |          WPA2 | RSSI:  -62 | SSID:  Vista99_IoT
MAC : [44:a5:6e:2f:76:c8] | Channel: 11 |          WPA2 | RSSI:  -62 | SSID:  NETGEAR53
MAC : [00:e6:3a:44:c6:92] | Channel: 11 |          WPA2 | RSSI:  -62 | SSID:  Vista99_WiFi
MAC : [00:e6:3a:44:c6:90] | Channel: 11 |          WPA2 | RSSI:  -63 | SSID:  Vista99_IoT
MAC : [00:e6:3a:44:c6:91] | Channel: 11 |      WPA-EAP | RSSI:  -63 | SSID:  Passpoint
MAC : [00:e6:3a:44:c6:94] | Channel: 11 |          WPA2 | RSSI:  -63 | SSID:  Vista99_Staff
MAC : [4a:bd:ce:d9:7b:7c] | Channel: 11 |          WPA2 | RSSI:  -65 | SSID:  vista3341
MAC : [00:e6:3a:58:79:80] | Channel:  1 |          WPA2 | RSSI:  -69 | SSID:  Vista99_IoT
MAC : [00:e6:3a:58:79:81] | Channel:  1 |      WPA-EAP | RSSI:  -69 | SSID:  Passpoint
MAC : [00:e6:3a:58:79:84] | Channel:  1 |          WPA2 | RSSI:  -69 | SSID:  Vista99_Staff
MAC : [bc:9a:8e:09:cc:f4] | Channel:  6 |          WPA2 | RSSI:  -70 | SSID:  ATTSheldon
MAC : [00:e6:3a:4b:2d:c0] | Channel:  6 |          WPA2 | RSSI:  -71 | SSID:  Vista99_IoT
MAC : [00:e6:3a:4b:2d:c4] | Channel:  6 |          WPA2 | RSSI:  -71 | SSID:  Vista99_Staff
MAC : [00:e6:3a:4b:2d:c1] | Channel:  6 |      WPA-EAP | RSSI:  -71 | SSID:  Passpoint
MAC : [bc:9a:8e:1a:6b:74] | Channel: 11 |          WPA2 | RSSI:  -71 | SSID:  Lair3
MAC : [00:e6:3a:4b:2d:c2] | Channel:  6 |          WPA2 | RSSI:  -72 | SSID:  Vista99_WiFi
MAC : [b4:63:6f:bf:0a:d4] | Channel: 11 |          WPA2 | RSSI:  -72 | SSID:  X.C455
MAC : [a8:b1:3b:d7:33:4c] | Channel:  2 |          WPA2 | RSSI:  -73 | SSID:  DIRECT-4A-HP DeskJet
2700 series
MAC : [c8:a6:08:67:68:44] | Channel:  6 |          WPA2 | RSSI:  -73 | SSID:  Vista99_Staff
MAC : [14:59:c0:ad:4b:e2] | Channel:  6 |          WPA2 | RSSI:  -73 | SSID:  NETGEAR69
MAC : [c8:a6:08:67:68:40] | Channel:  6 |          WPA2 | RSSI:  -73 | SSID:  Vista99_IoT
MAC : [ac:8f:a9:9c:31:94] | Channel:  2 |          WPA2 | RSSI:  -74 | SSID:  ally&latte
MAC : [c8:a6:08:67:68:41] | Channel:  6 |      WPA-EAP | RSSI:  -74 | SSID:  Passpoint
MAC : [bc:9a:8e:e6:83:04] | Channel:  6 |          WPA2 | RSSI:  -74 | SSID:  notyourwifi
MAC : [80:ab:4d:b2:8e:74] | Channel:  6 |          WPA2 | RSSI:  -75 | SSID:  TuWo
MAC : [a8:70:5d:93:a5:11] | Channel:  1 |          WPA2 | RSSI:  -76 | SSID:  access5015
MAC : [d4:6c:6d:ba:16:ff] | Channel:  1 |          WPA2 | RSSI:  -76 | SSID:  home4025
MAC : [8a:9e:68:e4:96:cf] | Channel:  1 |          WPA2 | RSSI:  -76 | SSID:  Jahee Home
MAC : [80:ce:62:b0:f9:e6] | Channel:  6 |          WPA2 | RSSI:  -76 | SSID:  DIRECT-E5-HP DeskJet
3700 series
MAC : [3c:b7:4b:2a:a5:1b] | Channel: 11 |          WPA2 | RSSI:  -77 | SSID:  XFWHG
MAC : [d8:1f:12:b9:9e:ad] | Channel:  6 |          OPEN | RSSI:  -78 | SSID:  SmartLife-9EAD
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
BLE Send Notification
```

ST BLE Toolbox



Cannot see the Wi-Fi Provisioning button

Solution: Uninstall the app and re-install the app back



life.augmented

Porting to a new host mcu using STM32CubeMX

Porting over new host mcu(1/3)

1. **Select the Host MCU**

   Choose the target STM32 part number from the STM32CubeMX

2. **Check schematics of Host MCU and ST67**

   Review pin mappings for SPI, GPIO, USART

3. **Configure USART for Debugging**

   Used for debugging logs via STLink virtual port

4. **Configure the SPI Interface for Host MCU and ST67 communication**

   Set SPI pins, SPI mode, SPI Priority

5. **Configure DMA Channels for SPI**

   Fast and reliable SPI data transfer, reducing CPU load during transmission

Porting over new host mcu(2/3)

6. **Configure GPIO Pins** (CHIP_EN, SPI_RDY, SPI_CS, BOOT, User_Button)

   Set GPIO Mode, GPIO output level, GPIO Maximum Output speed

7. **Configure SYS** (Time base Source)
8. **Configure Clock**
9. **Enable FreeRTOS Middleware**

   Setup default task, heap size, stack size and required priorities

10. **Enable X-CUBE-ST67W61**

    Setup Application, W61 Drivers, Utilities

11. **Configure NVIC Interrupt Settings**

    SPI, DMA, USART, EXTI

Porting over new host mcu(3/3)

12. Project Creation and Code Generation

Generate project files (STM32CubeIDE/IAR)

Open in IDE for application development

13. Application Entry: Call main_app()

Inside StartDefaultTask, invoke main_app()

This initializes ST67W6X Driver, ST67W6X Wi-Fi module, ST67W6X Network module, Application specific functions

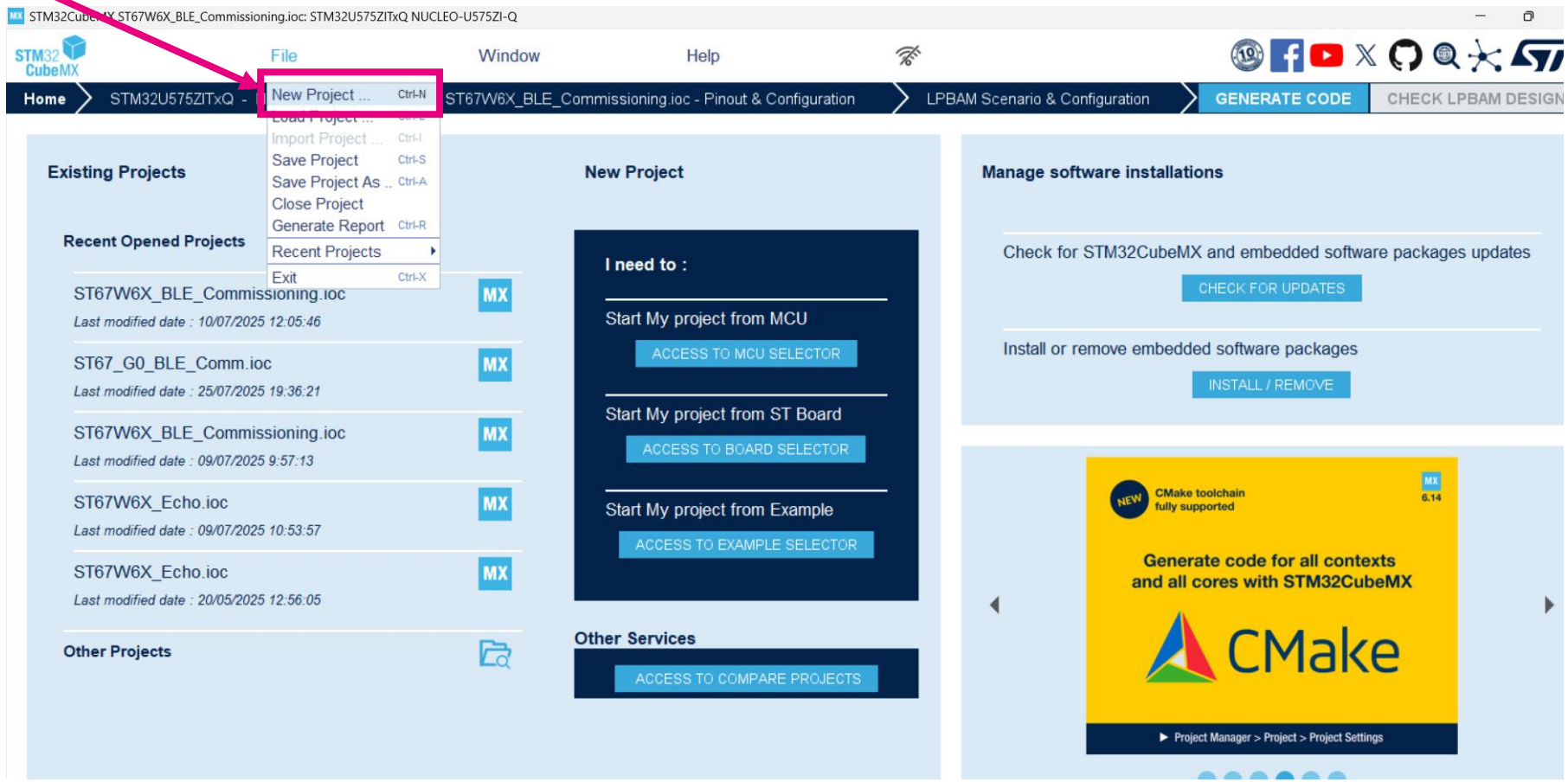
14. Build and Flash the application

life.augmented

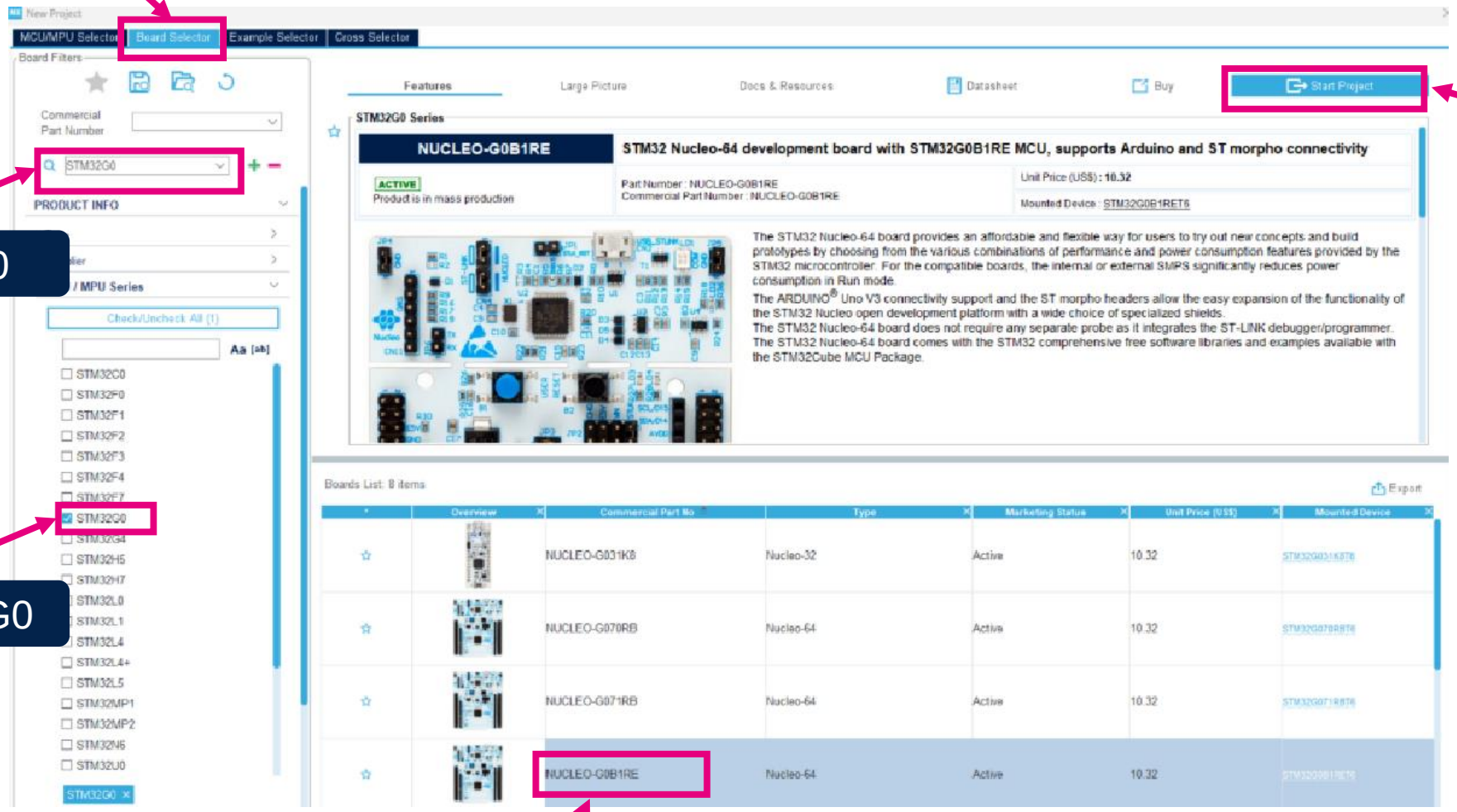
Generating BLE Commissioning app using NUCLEO-G0B1RE as host processor (Hands-On)

Case 3: Generating BLE Commissioning app using NUCLEO-G0B1RE as host processor

Open STM32CubeMX and create the new project.



Select the board selector

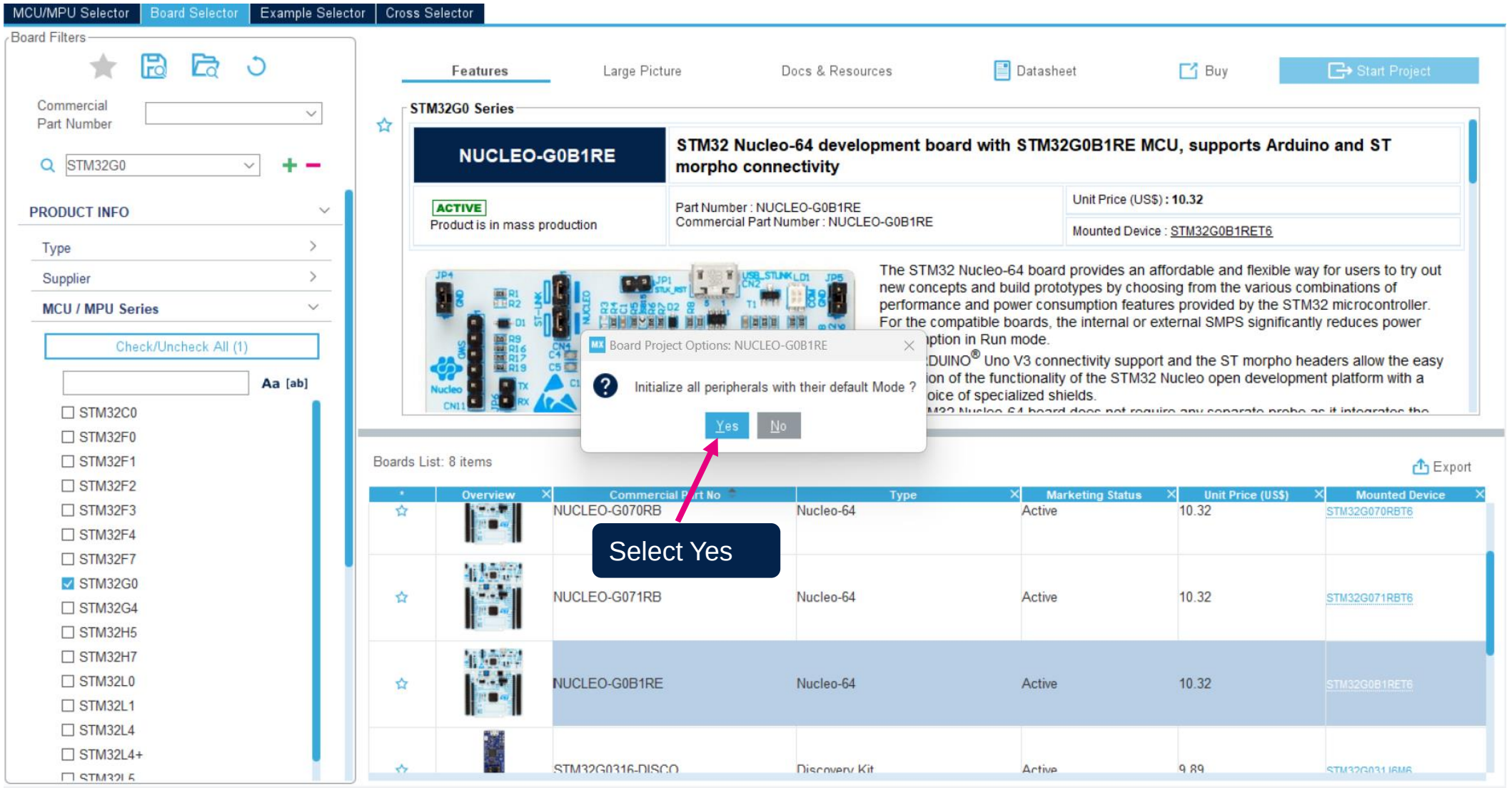


Start Project

Type STM32G0

Select STM32G0

Select NUCLEO-G0B1RE

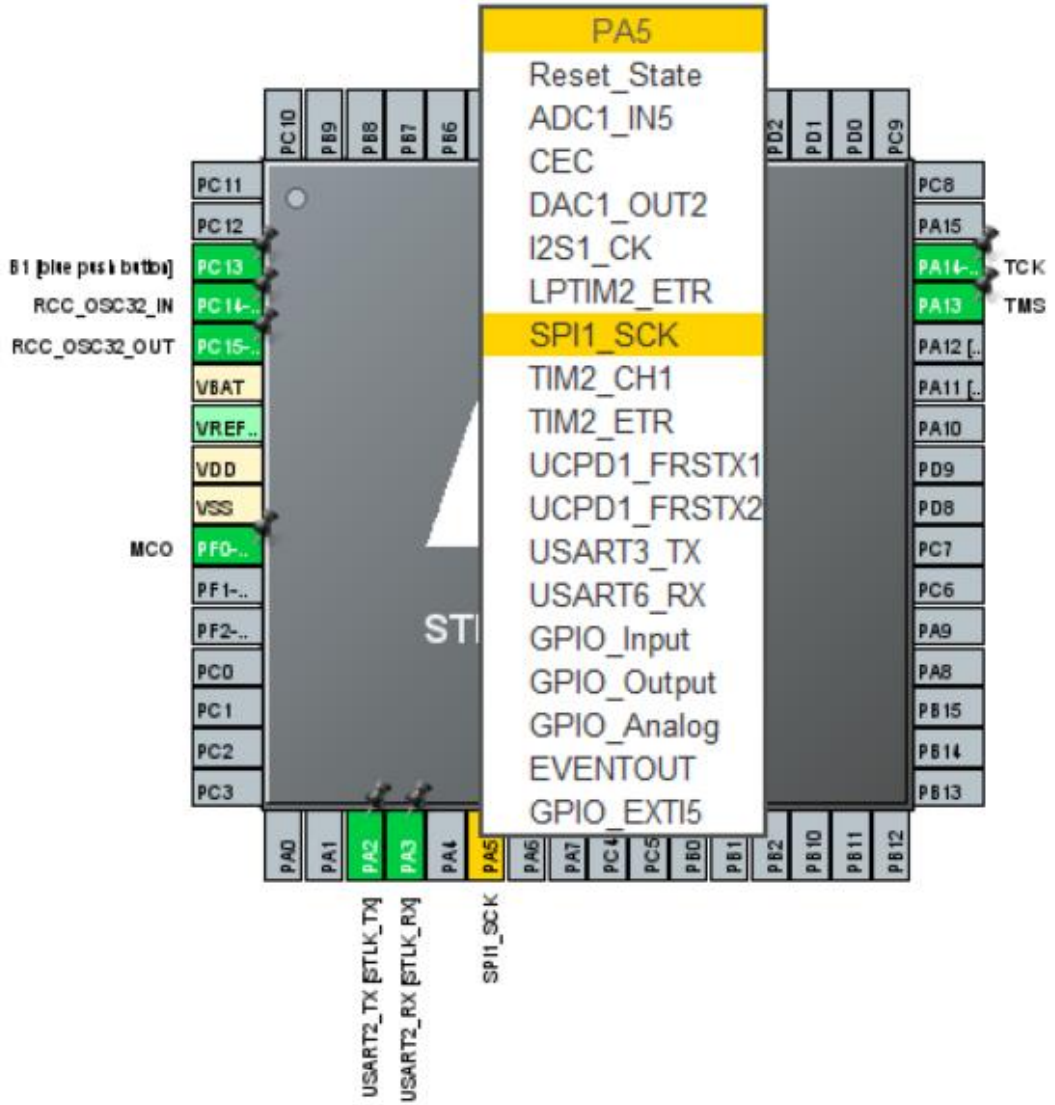


Pinout

When generating an application from scratch and if X-Nucleo-67W61M1 is used, the pinout must be set in accordance with the ST67W61M1 Arduino interfaces.

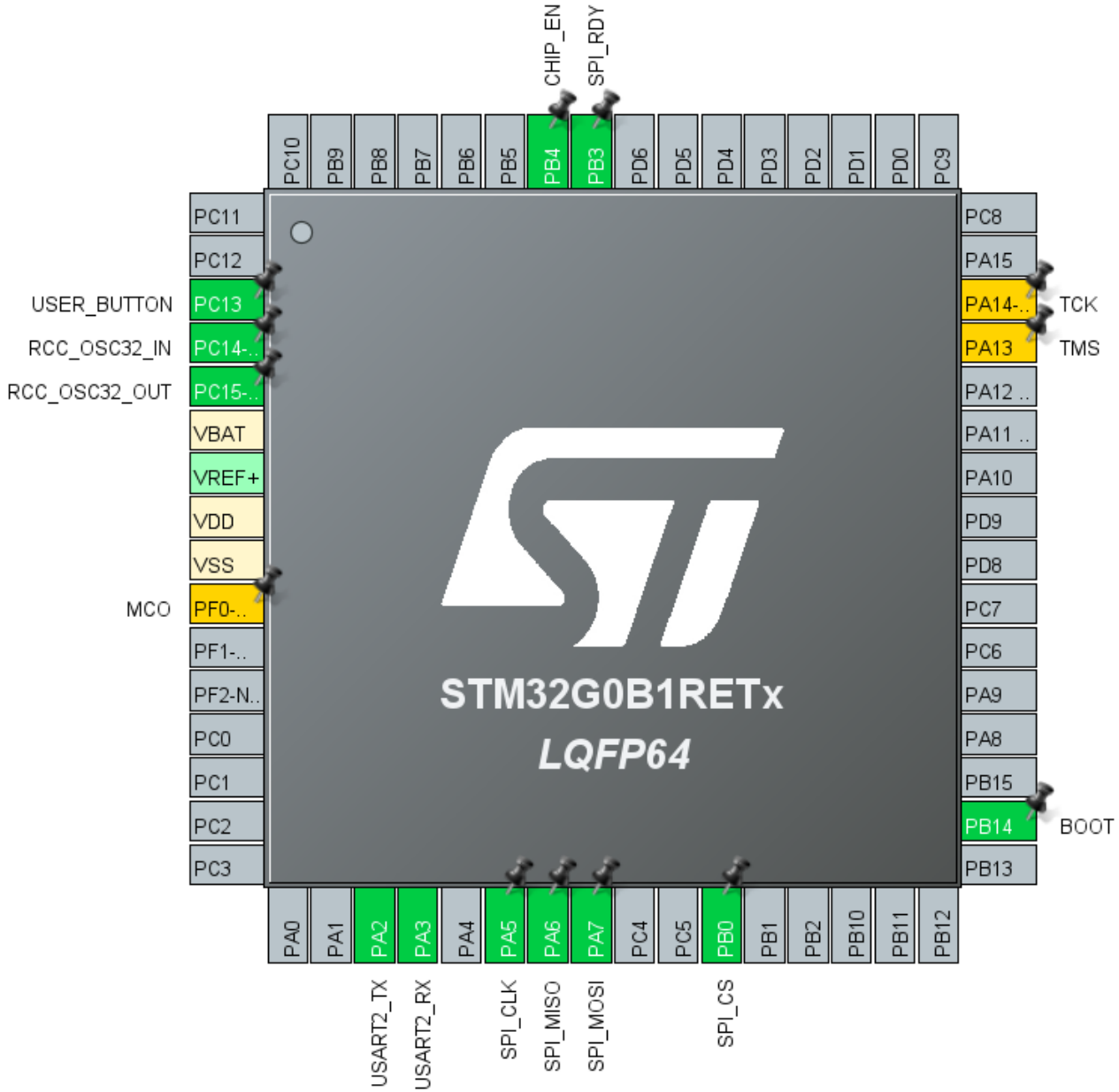
Pin function	Arduino Connector	STM32G0 Pin	GPIO mode	GPIO pull mode	GPIO speed
SPI_CLK	CN5.D13	PA5	AF PP	No Pull	Very high
SPI_MISO	CN5.D12	PA6	AF PP	No Pull	Very high
SPI_MOSI	CN5.D11	PA7	AF PP	No Pull	Very high
SPI_CS	CN5.D10	PB0	Output PP	No Pull	high
USER_BUTTON	-	PC13	EXTI Falling	Pull-up	N/A
CHIP_EN	CN9.D5	PB4	Output PP	No Pull	High
BOOT	CN9.D6	PB14	Output PP	No Pull	Low
SPI_RDY	CN9.D3	PB3	EXTI Falling/Rising	No Pull	N/A

Assigning functionality



For assigning, click on the pin and select functionality.
(e.g) PA5->SPI_SCK

Pinout View



After, assigning all the pins mentioned from the pin out table, this would be overall pinout view

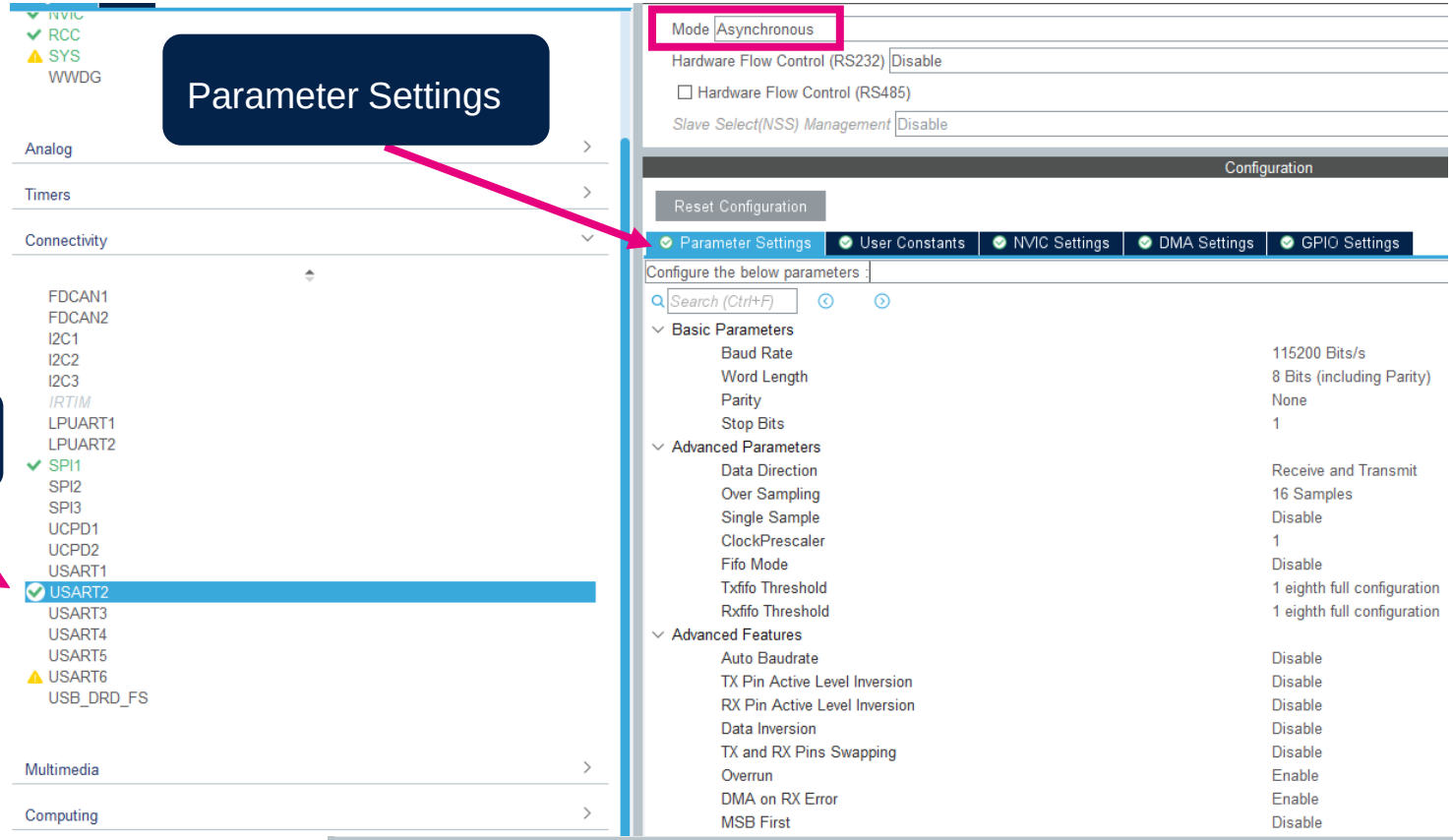
UART GPIO(USART 2)

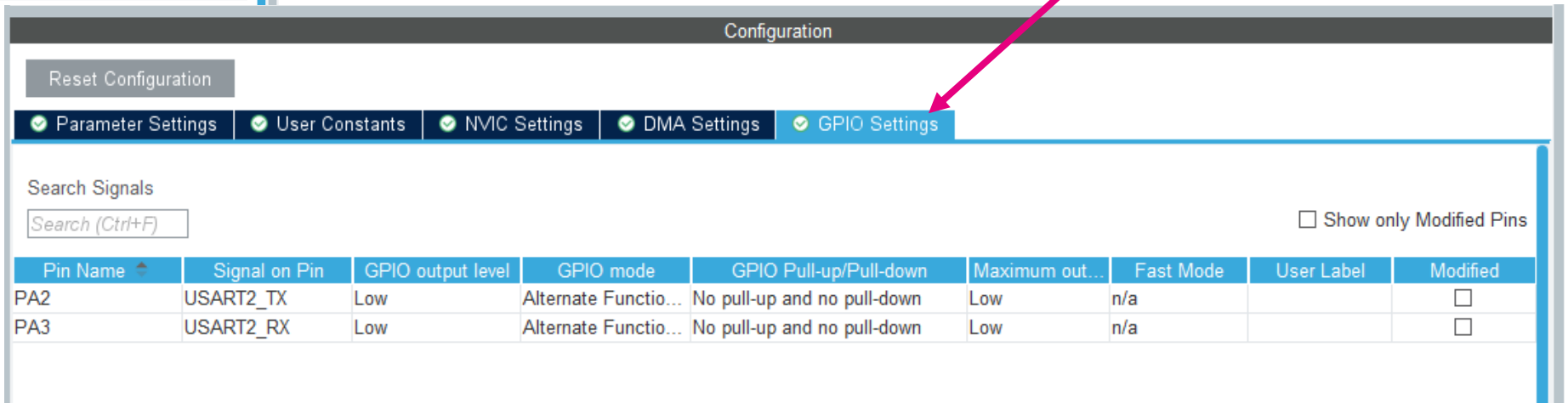
PA2: USART2_TX
PA3: USART2_RX

Parameter Settings

Select USART2

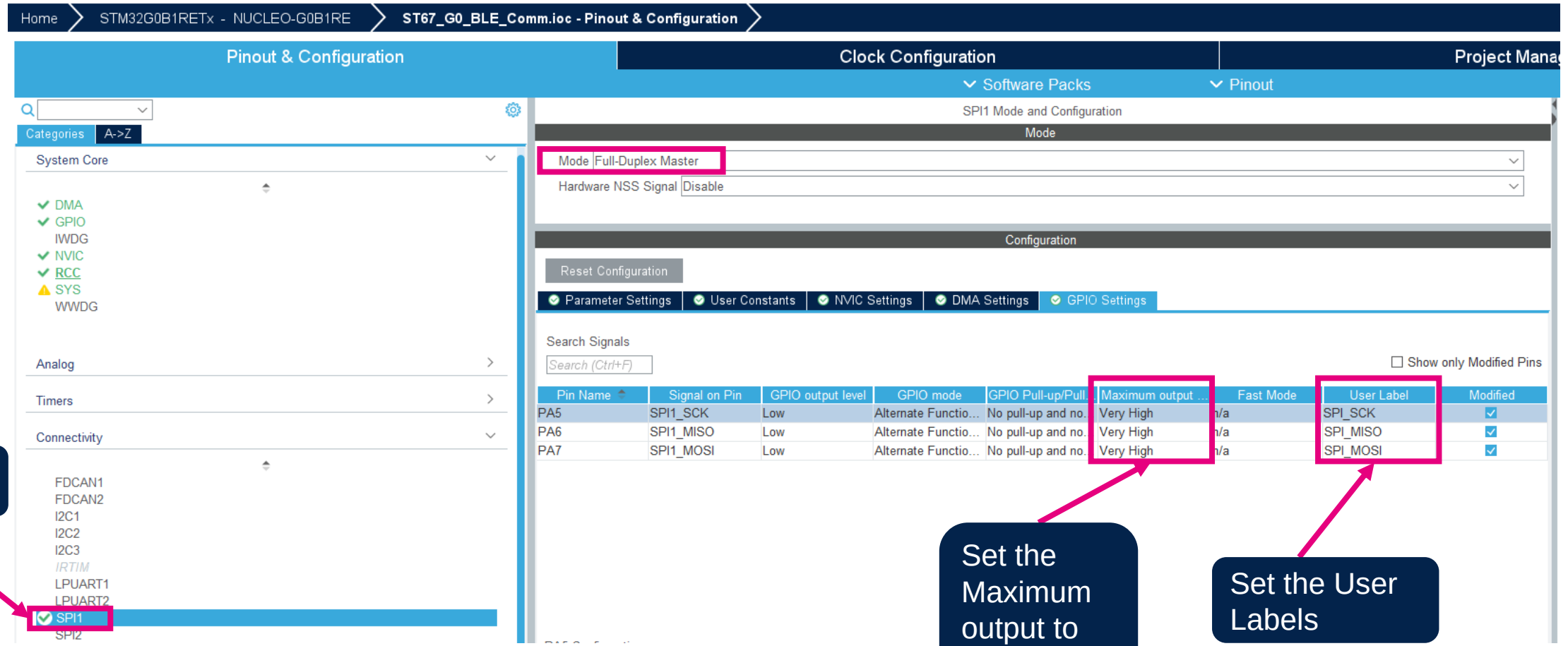
GPIO Settings





SPI GPIO(SPI1)

PA5: SPI_SCK
PA6: SPI_MISO
PA7: SPI_MOSI

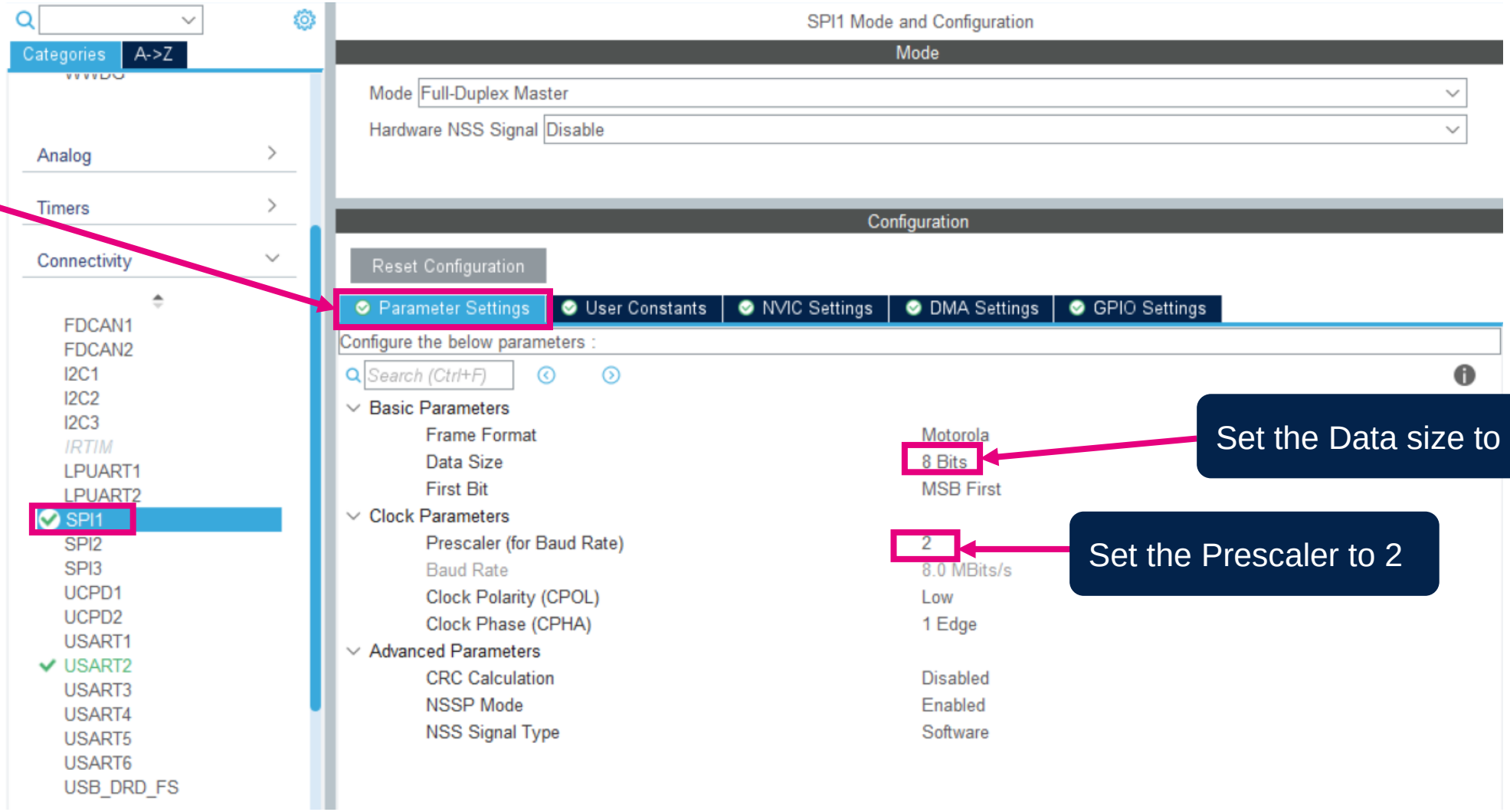


Select SPI1

Set the Maximum output to **Very High**

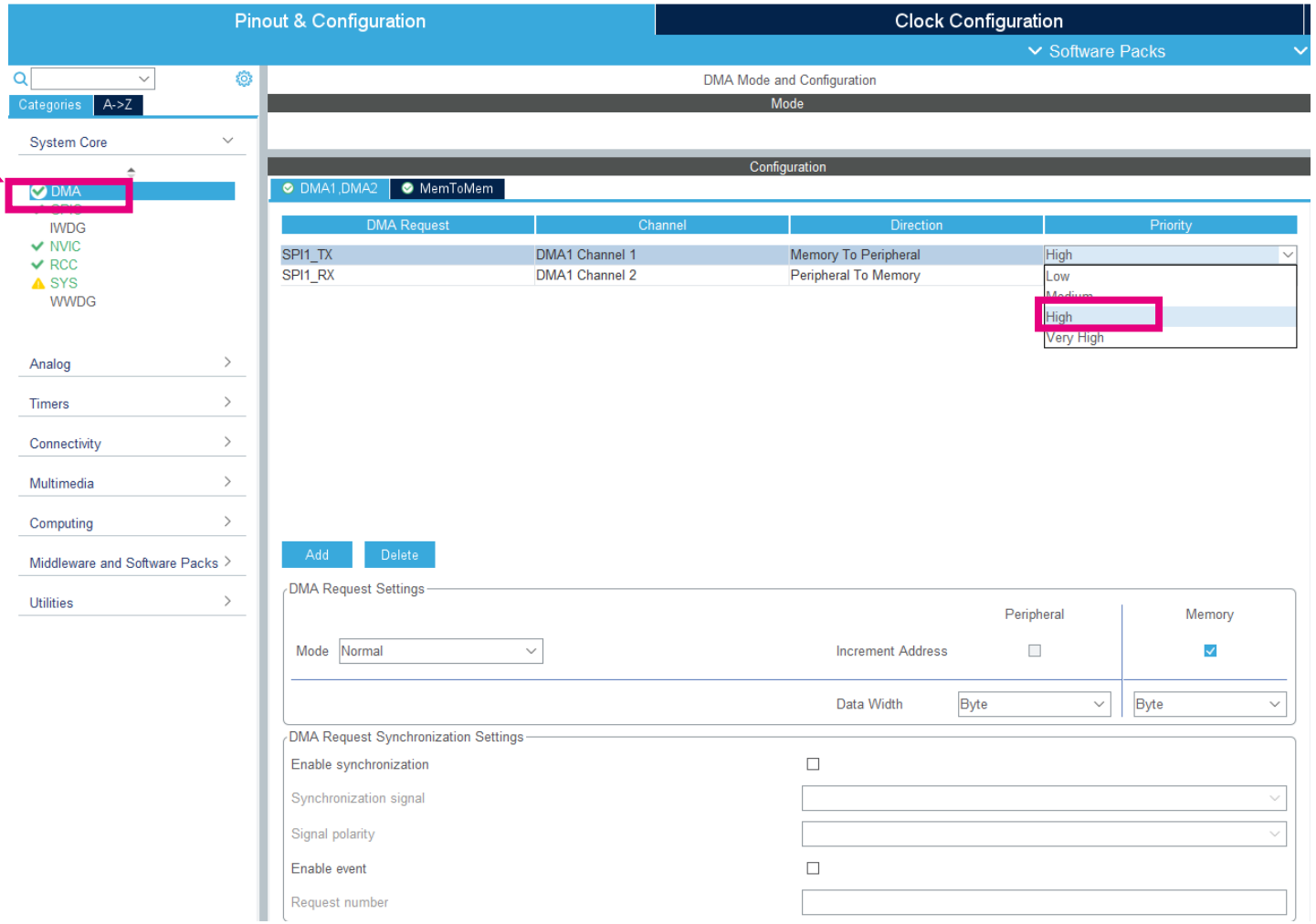
Set the User Labels

SPI GPIO(SPI1)



DMA Channels

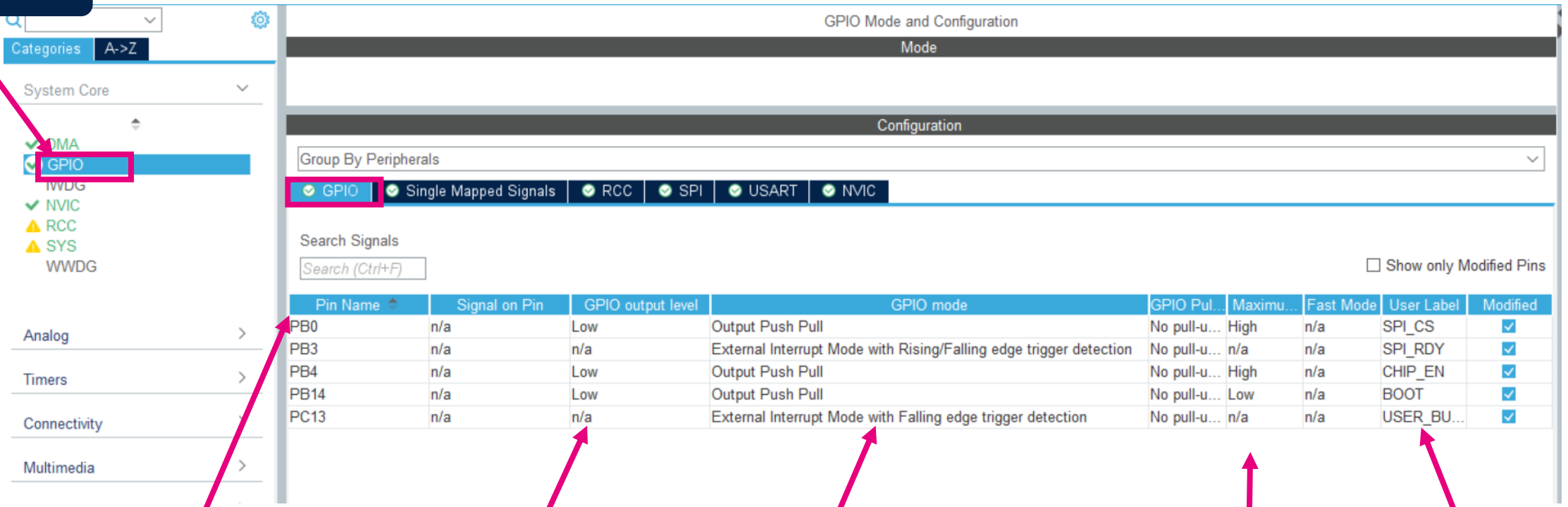
Select DMA



SPI1_TX -> DMA1 CHANNEL 1->High
SPI1_RX -> DMA1 CHANNEL 2-> High

GPIO

Select GPIO



GPIO pin names

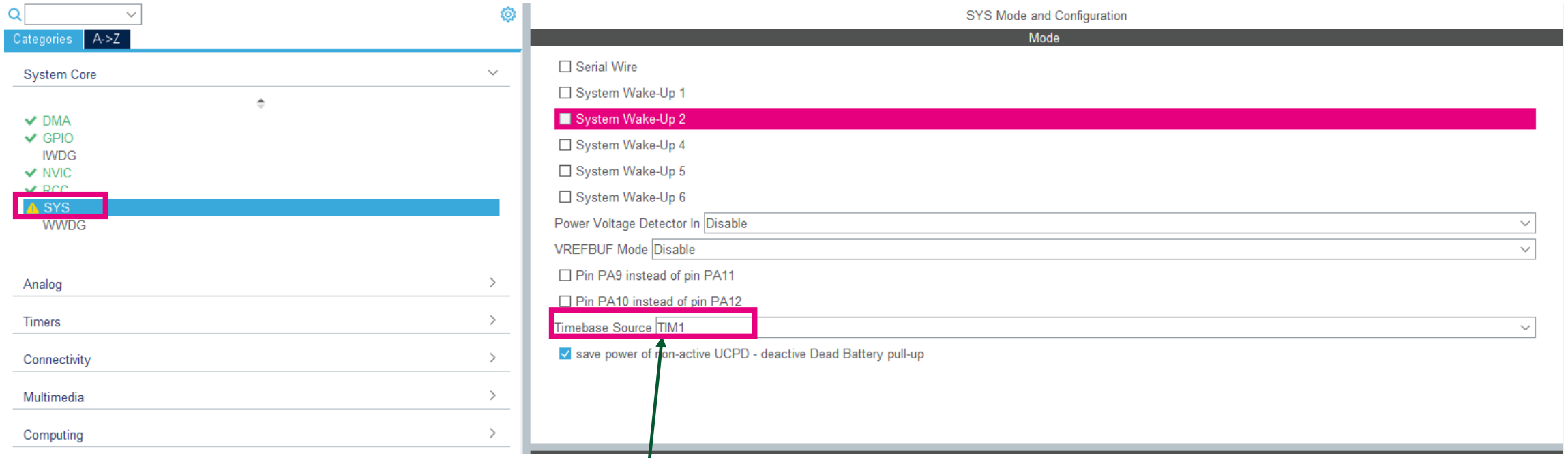
GPIO output level

GPIO mode

GPIO maximum speed

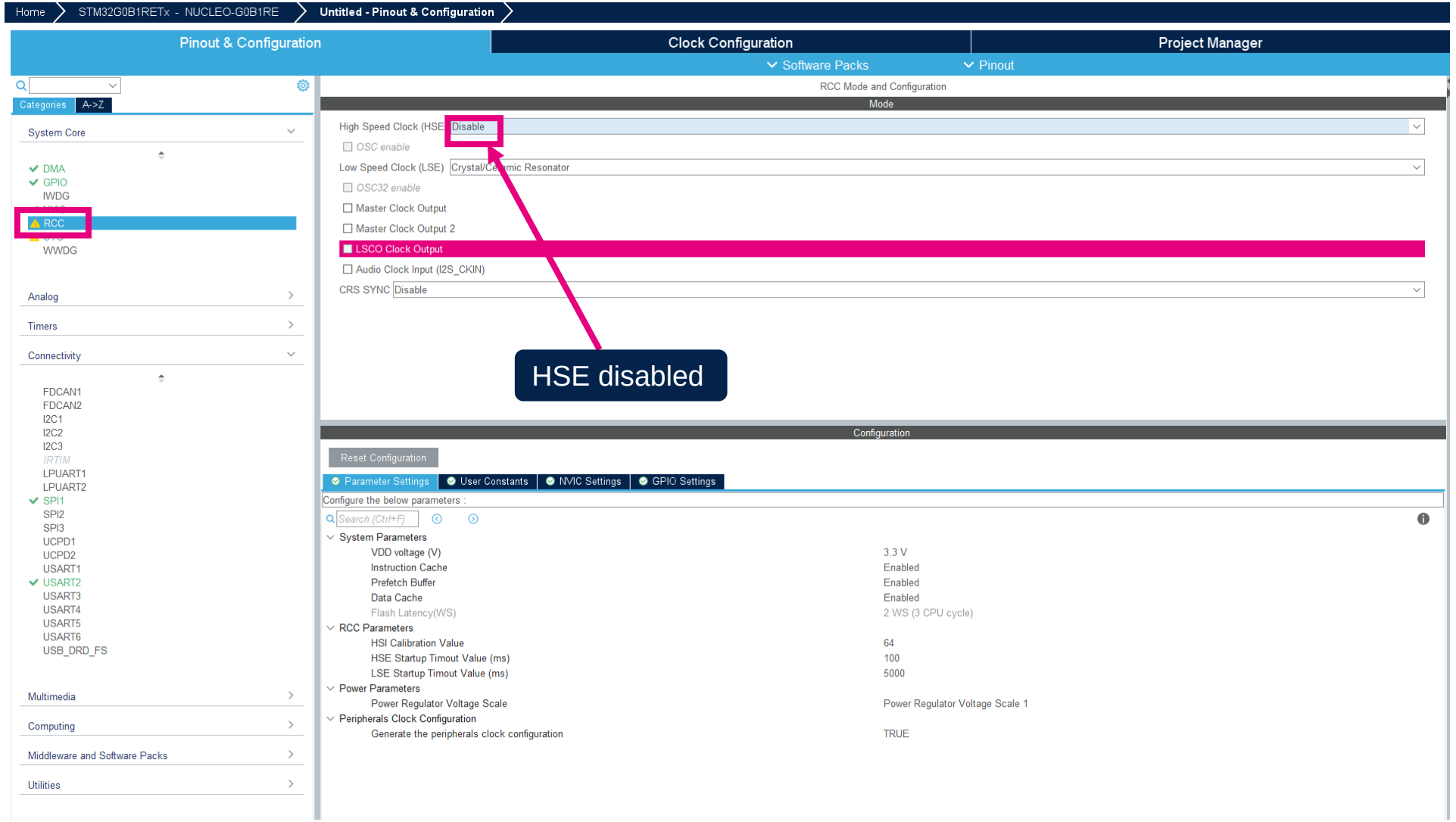
GPIO user label

Timer



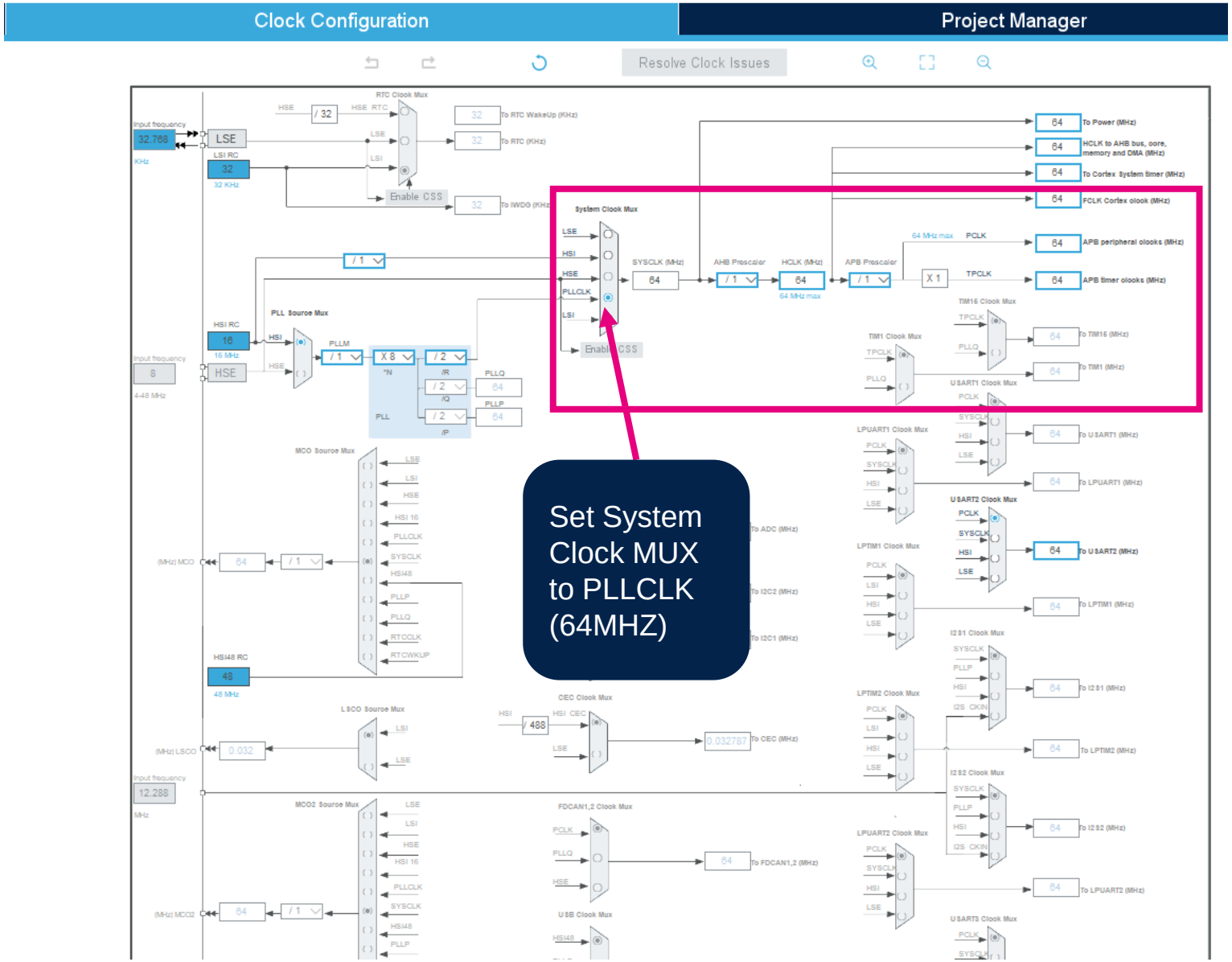
Sys mode configuration: TIM1 in place of Systick: to avoid RTOS conflicts, enable low power operation.

RCC



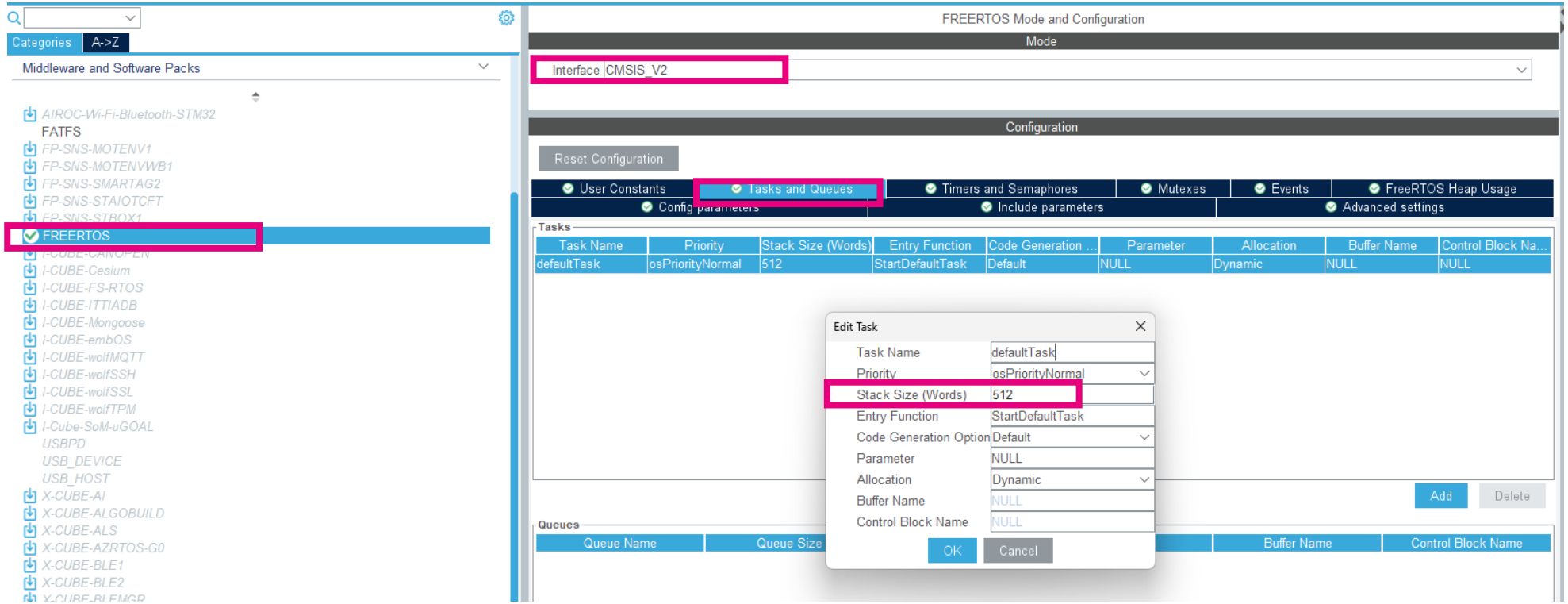
Clock Configuration

- In the "Clock Configuration" panel, system clock is set to the maximum frequency of 64 MHz, which increases the SPI clock speed to 32 MHz



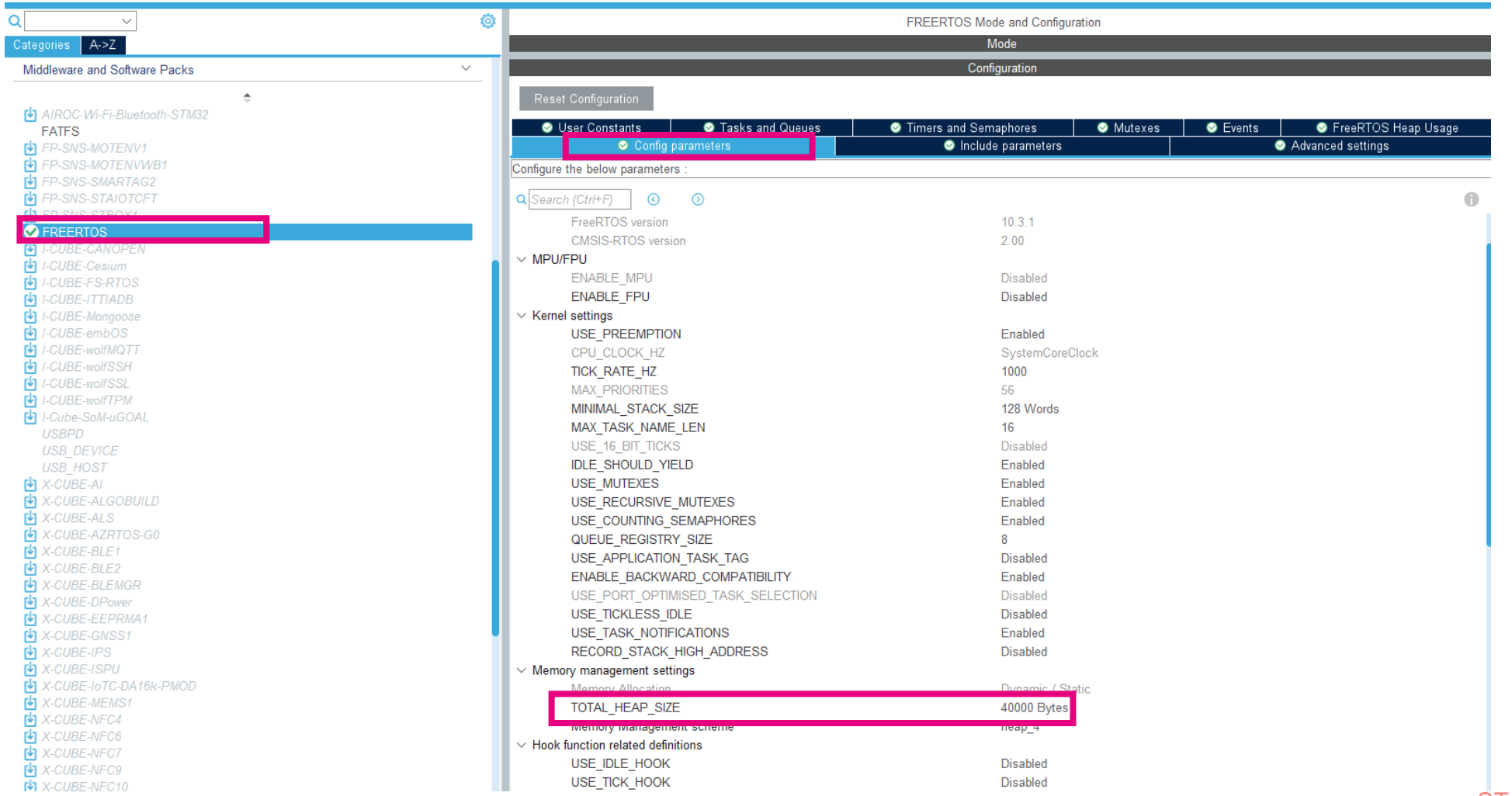
Middleware & Software Packs
FreeRTOS

- FreeRTOS component can be either be selected from Middleware (FREERTOS) or from the Software Pack (X-CUBE-FREERTOS) depending on the chosen STM32.
- By default, the stack size of the default task defined by STM32CubeMx tool is equal to 128 words. The application default task required **512 words**.

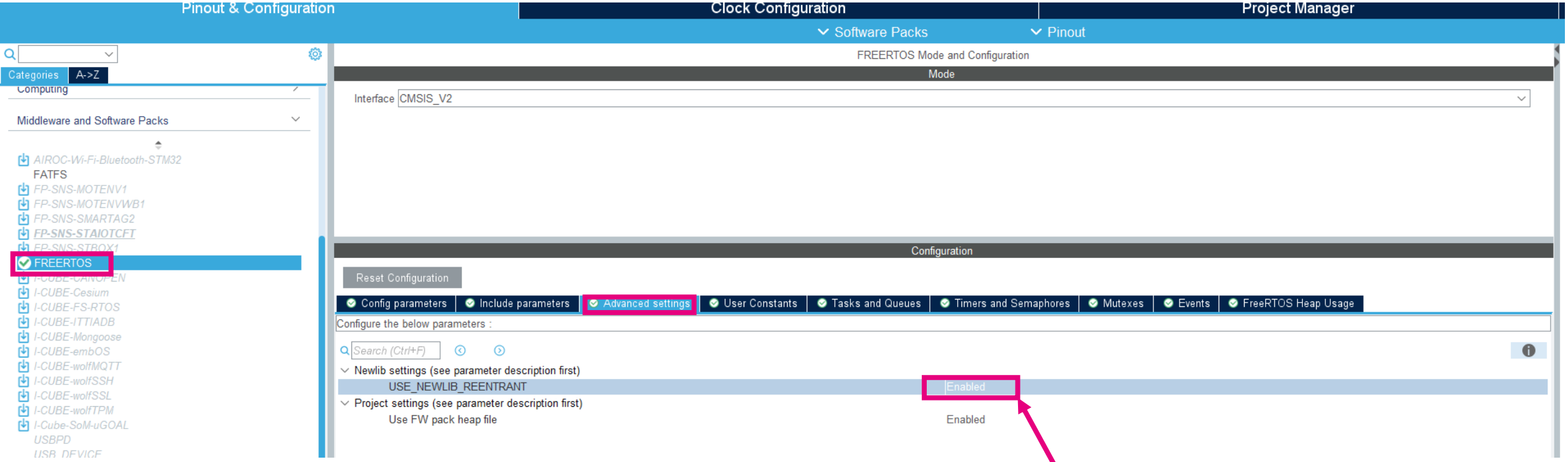


Middleware & Software Packs FreeRTOS

- Adjust heap :The size heap value should be set at least to 40Kbytes. To optimize the memory, this value could be adjusted during integration and validation tests of the application

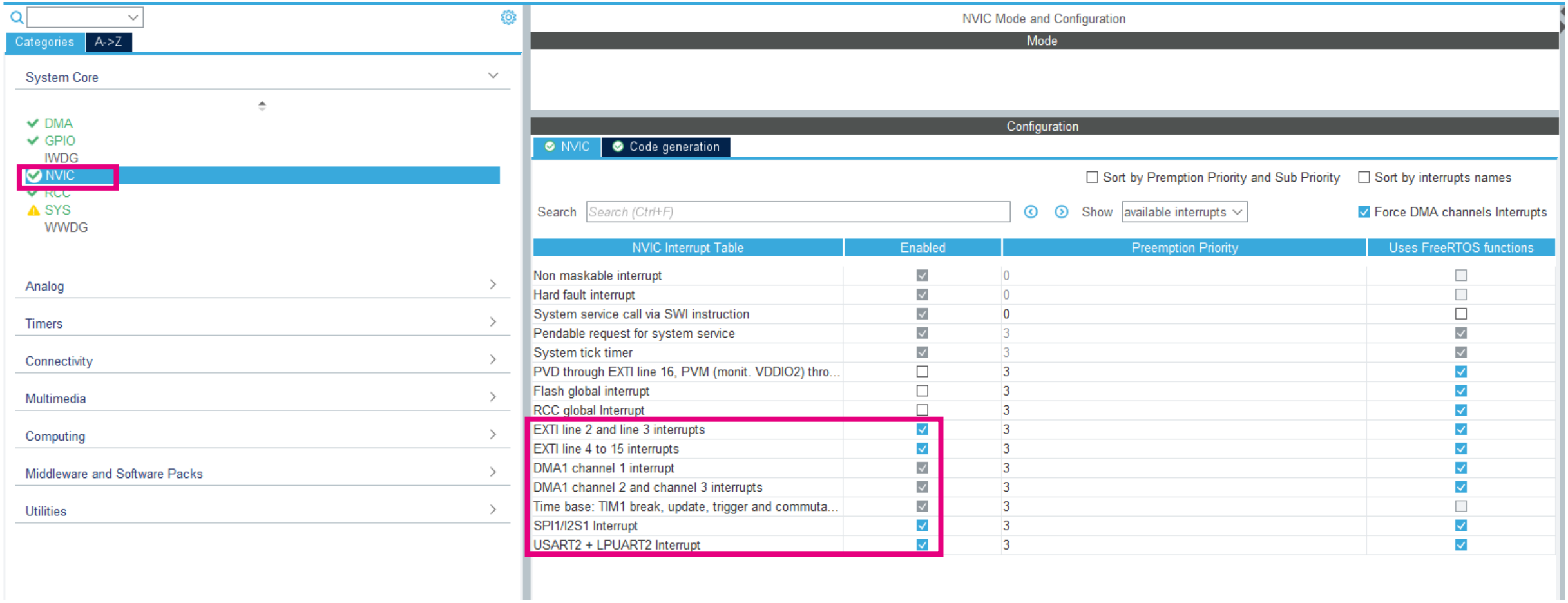


Middleware & Software Packs
FreeRTOS

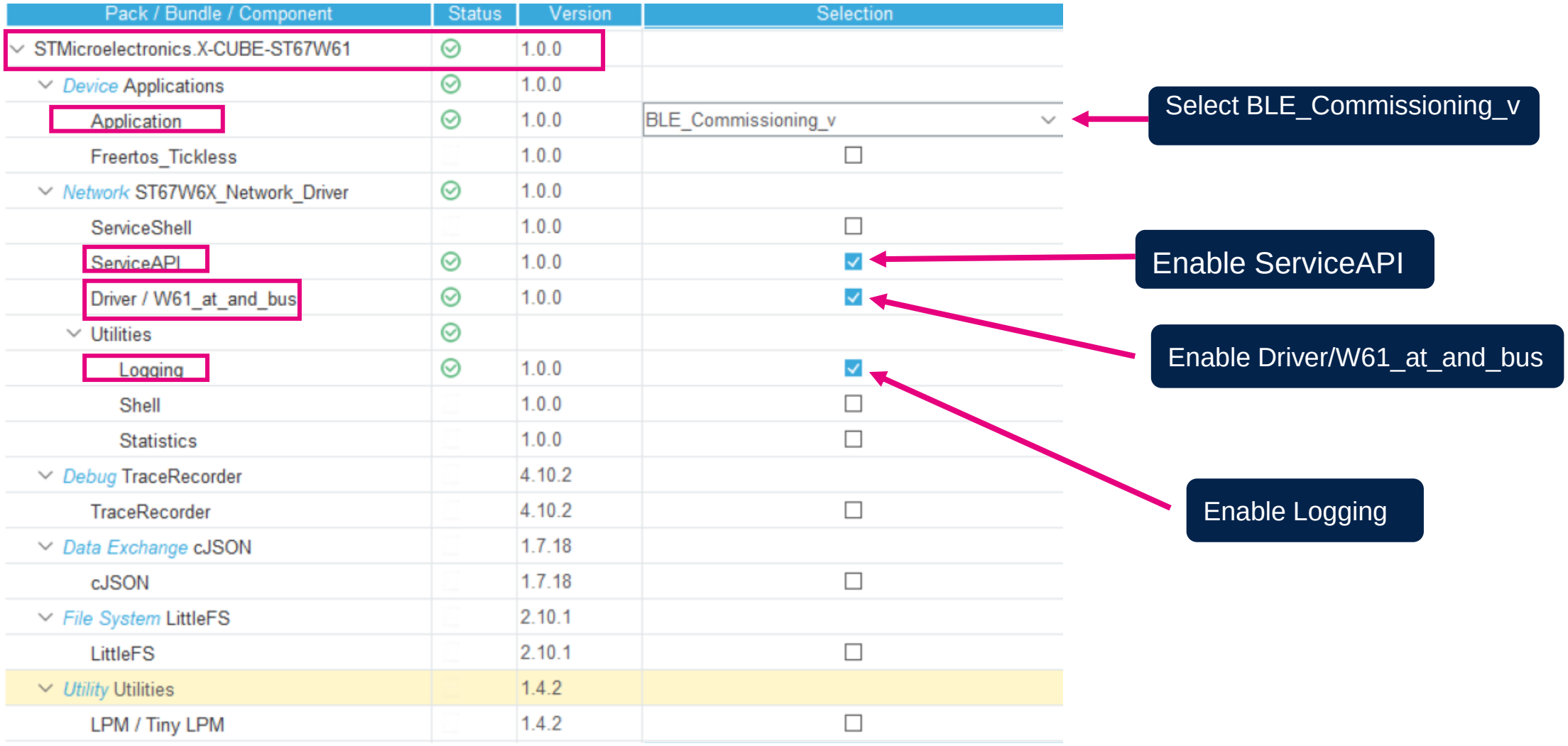


Enable
USE_NEWLIB_REENTRANT

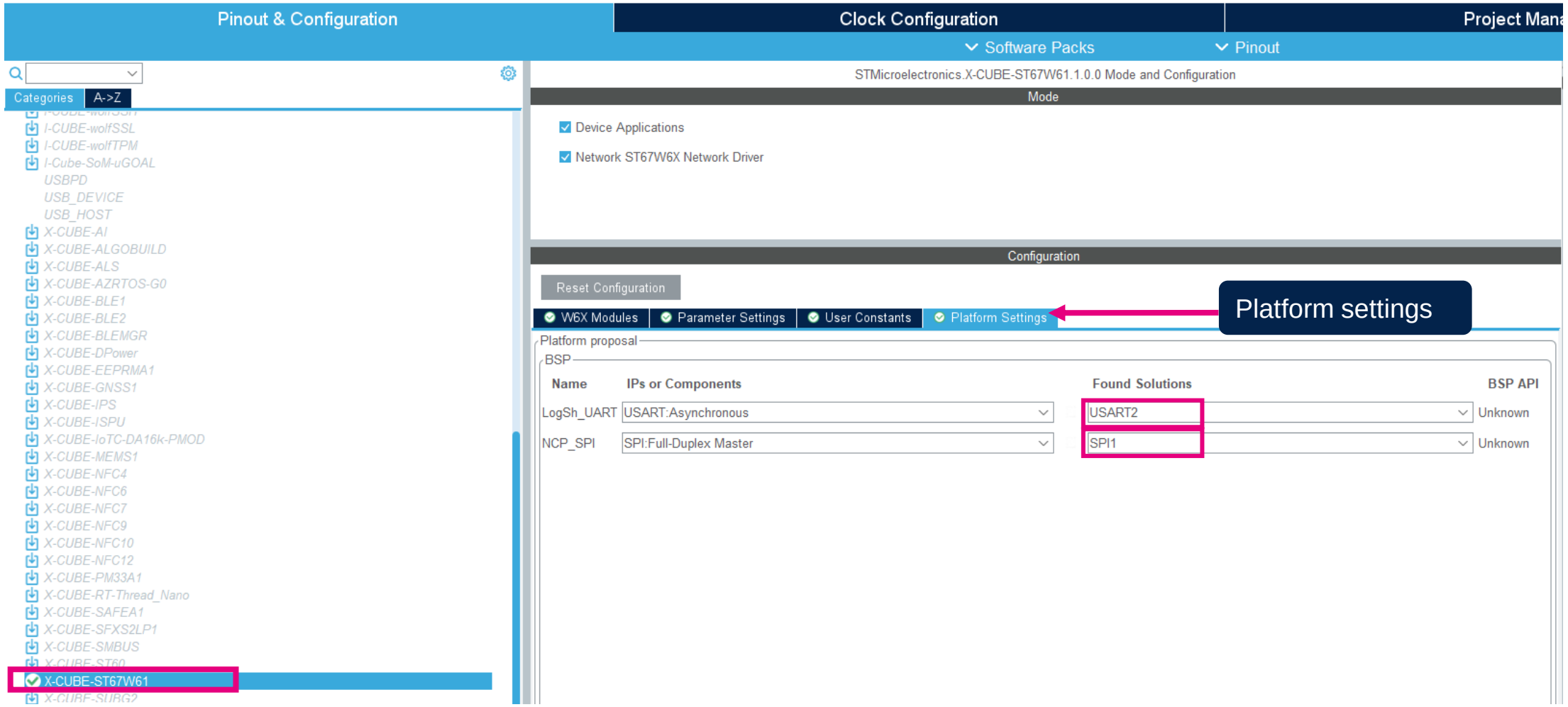
NVIC



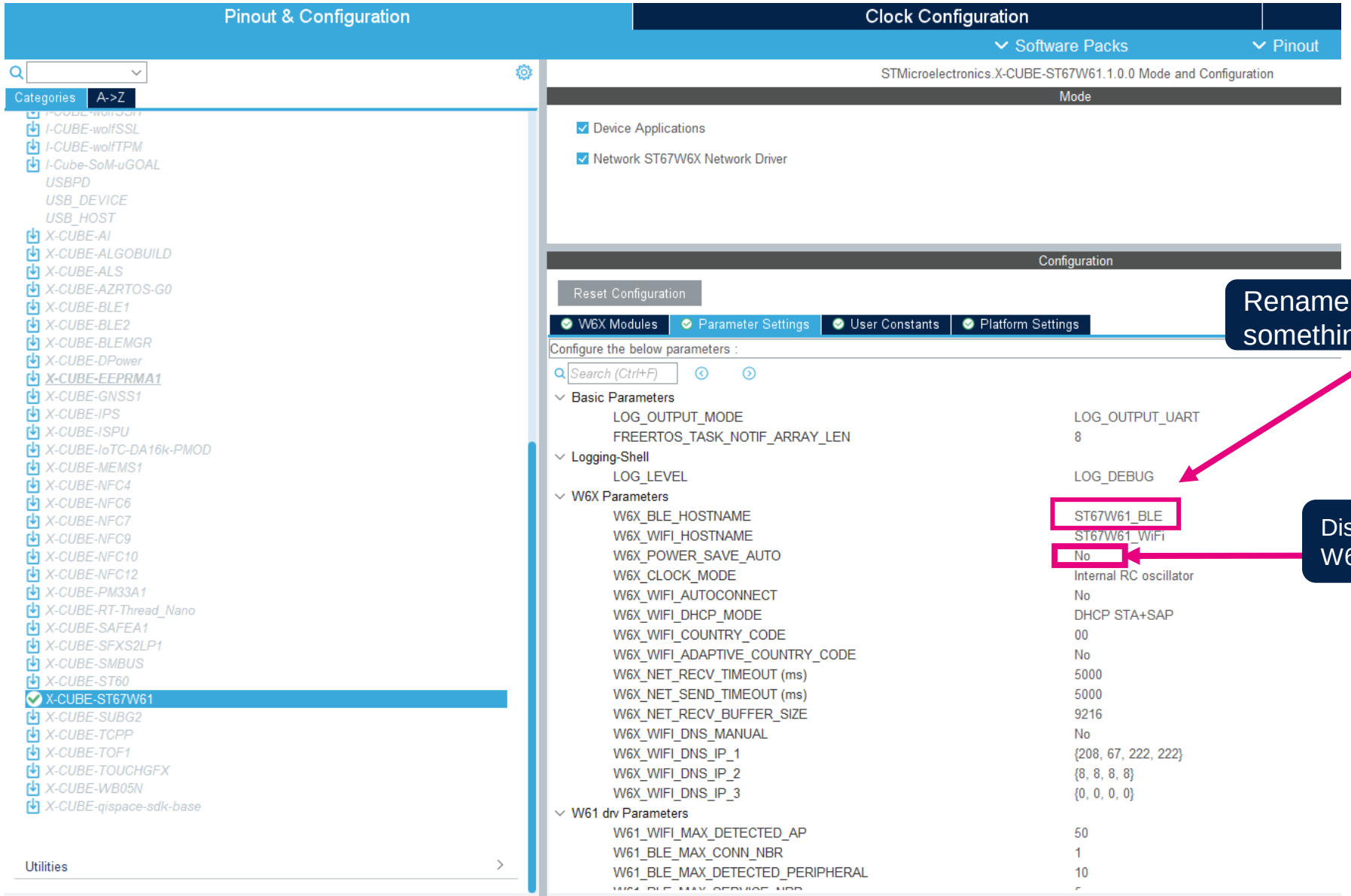
Middleware & Software Packs
X-CUBE-ST67W61 Software package



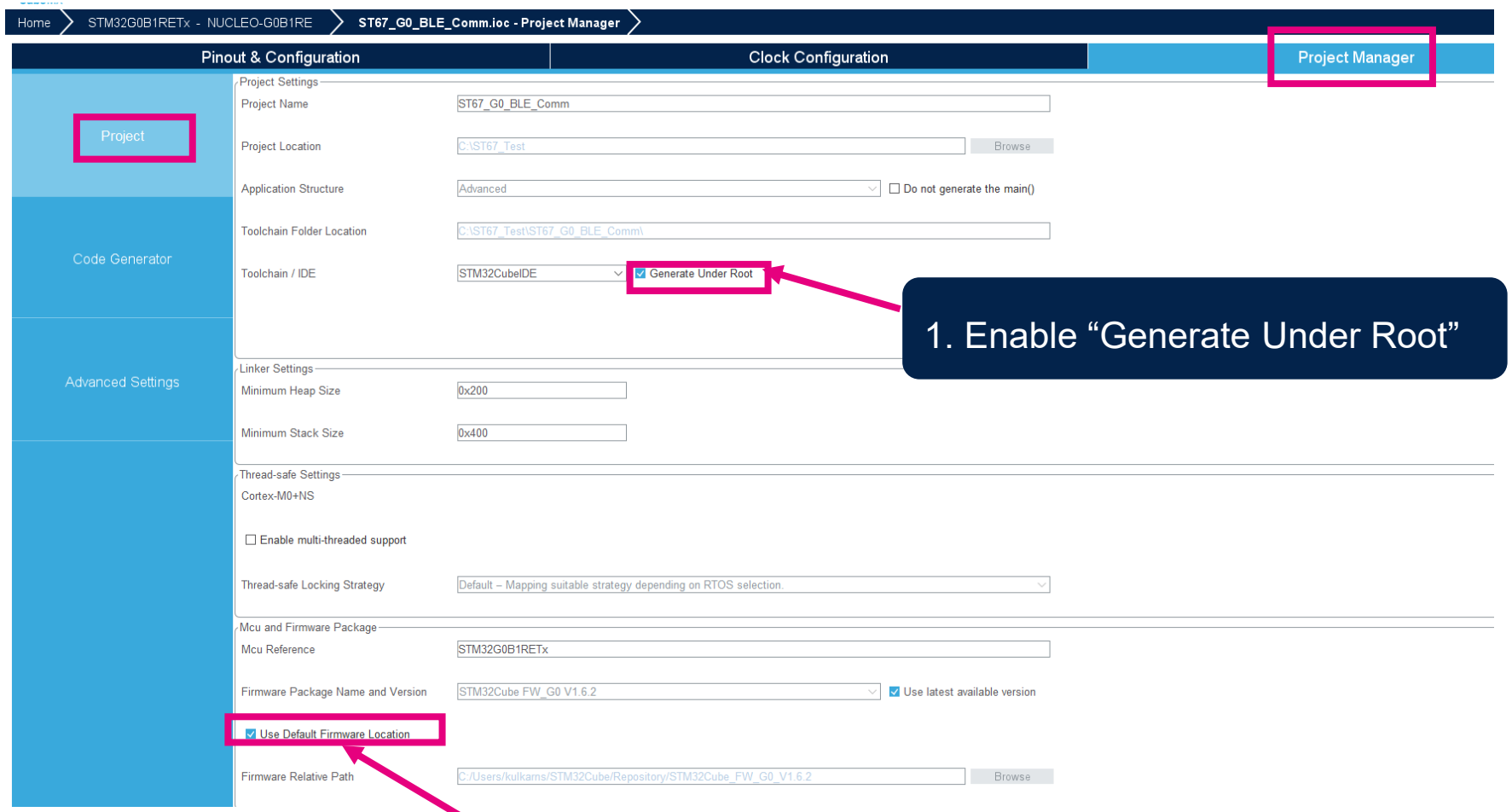
Middleware & Software Packs
X-CUBE-ST67W61 Software package



Middleware & Software Packs
X-CUBE-ST67W61 Software package



Project Creation

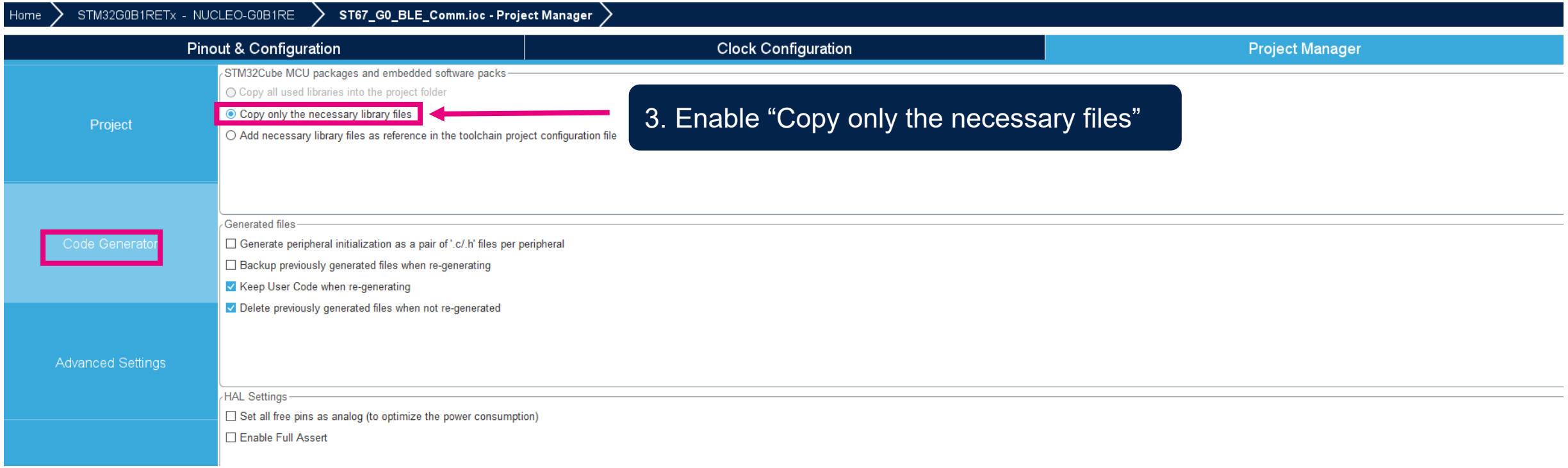


1. Enable “Generate Under Root”

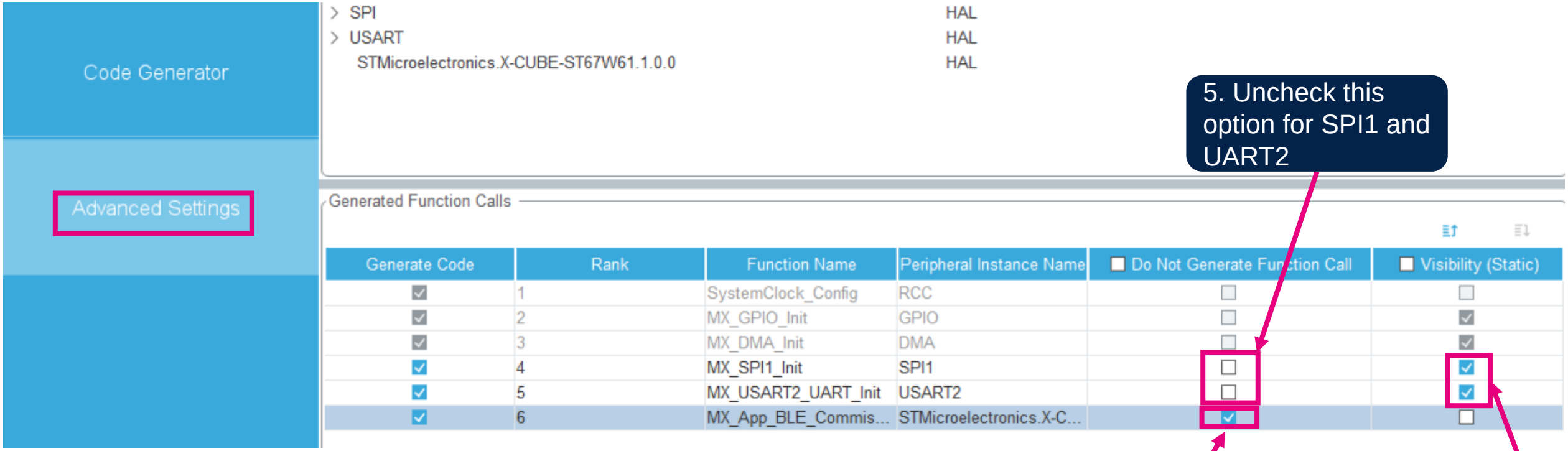
2. Enable “ Use Default Firmware Location”

Here, we are enabling the Flat directory structure: (i.e. Driver and MW directories copied into your project directory)

Code Generator

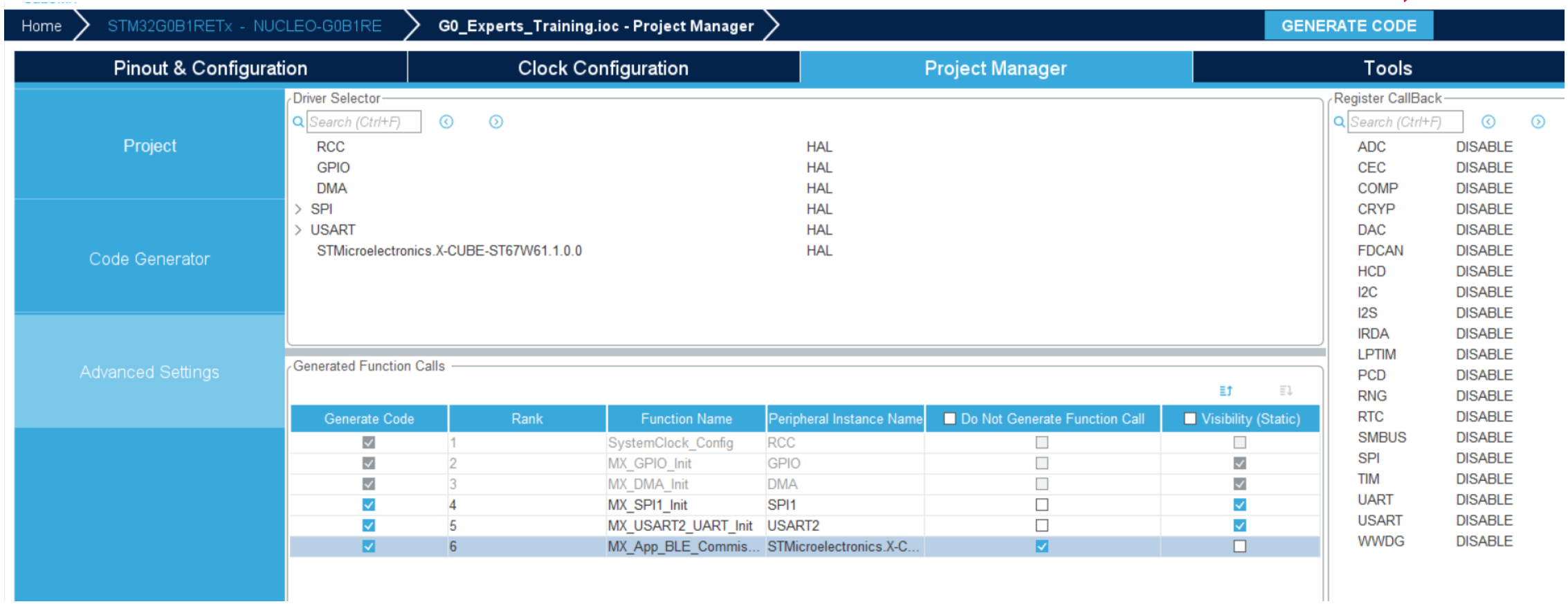


Advanced settings

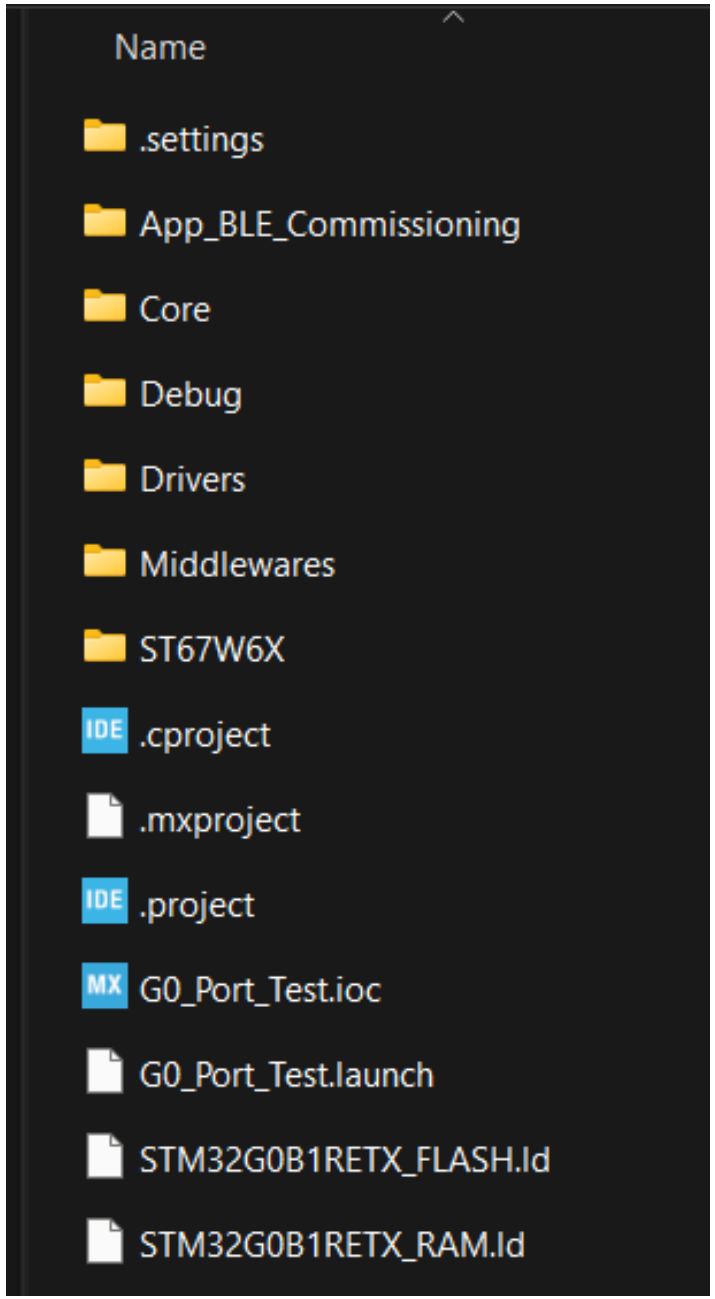


Generate Code

5. Generate Code



Folder Structure

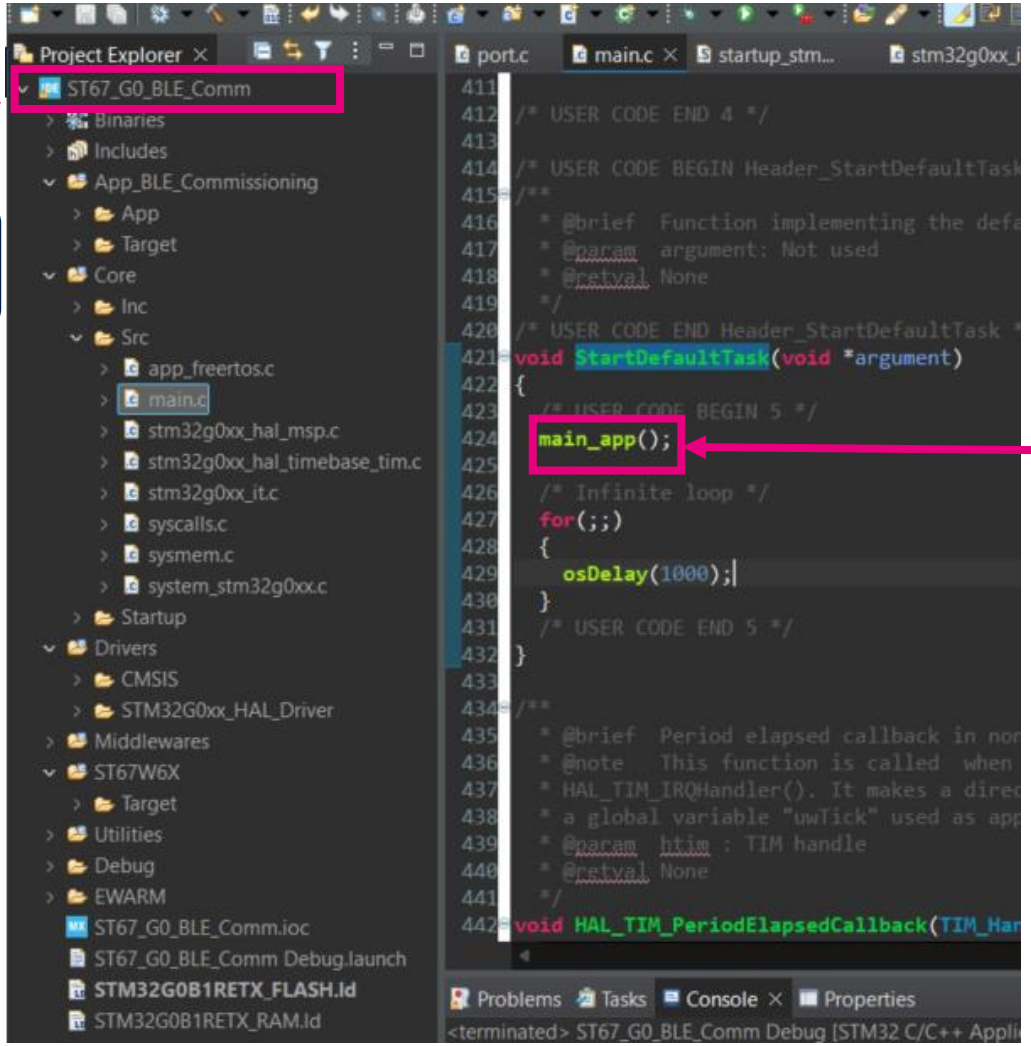


Folder structure view after generating the code from CubeMX

- App code
- Core
- Drivers
- Middlewares
- ST67W6X

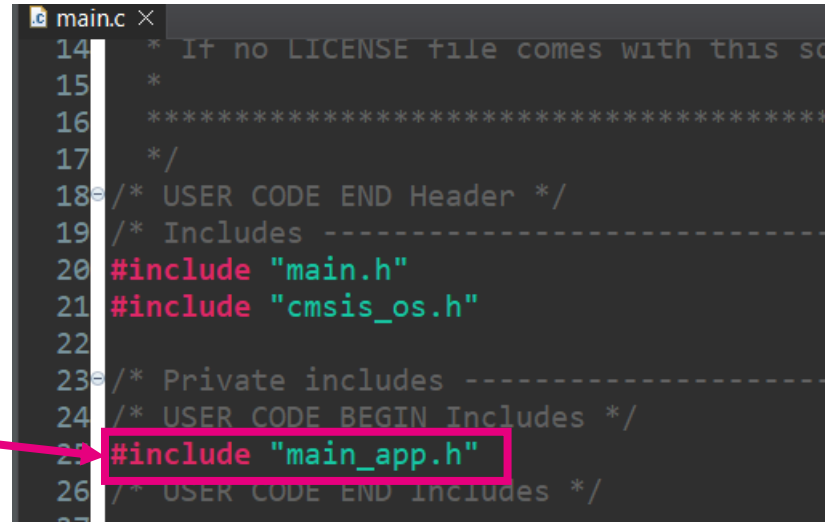
STM32CubeIDE

Load the project on the STM32CubeIDE



1. Call main_app() function under StartDefaultTask() in main.c file

2. Call #include “main_app.h”

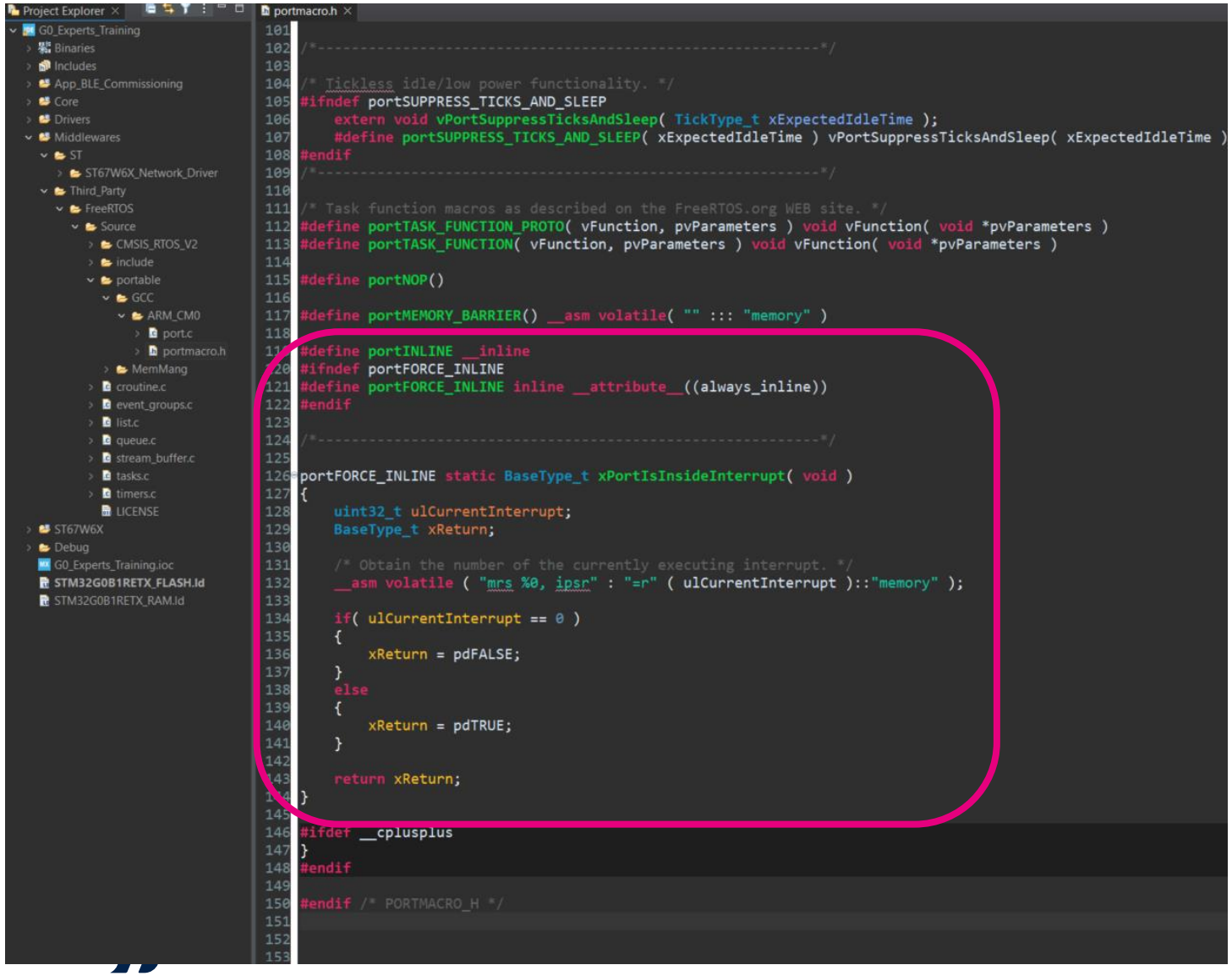


xPortIsInsideInterrupt()

- If the used device embeds an Arm® Cortex®M0+ (e.g. STM32G0) and its FreeRTOS version (v10.3.1) which is older than of V10.6, definition below **must be added manually after project has been generated**.
- The below function definition must be added in the **portmacro.h** file (Project/Middlewares/Third_Party/FreeRTOS/Source/portable/GCC/ARM_CM0)

```
#define portINLINE __inline
#ifndef portFORCE_INLINE
#define portFORCE_INLINE inline __attribute__((always_inline))
#endif

/*-----------------------------------------------------------*/

portFORCE_INLINE static BaseType_t xPortIsInsideInterrupt( void )
{
   uint32_t ulCurrentInterrupt;
   BaseType_t xReturn;

   /* Obtain the number of the currently executing interrupt. */
   __asm volatile ( "mrs %0, ipsr" : "=r" ( ulCurrentInterrupt )::"memory" );

   if( ulCurrentInterrupt == 0 )
   {
      xReturn = pdFALSE;
   }
   else
   {
      xReturn = pdTRUE;
   }

   return xReturn;
}
```

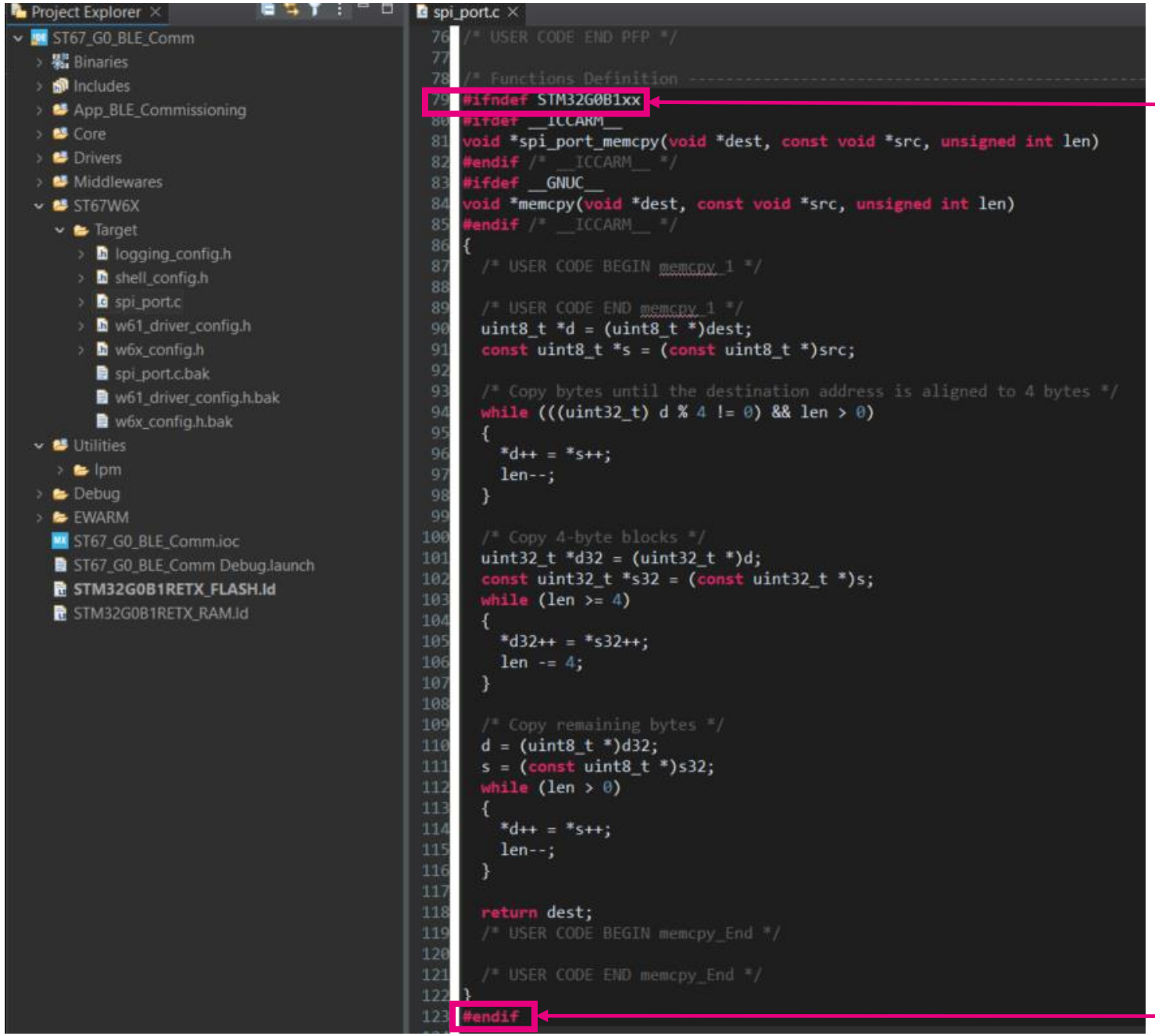
STM32CubeIDE



Call xPortIsInsideInterrupt() function under portmacro.h file
IMP: As we are using G0 (CortexM0) -> older version of FreeRTOS, so we need to add this function definition manually.

```
#define portINLINE __inline
#ifndef portFORCE_INLINE
#define portFORCE_INLINE inline
__attribute__((always_inline))
#endif

/*-----------------------------------------------------------*/

portFORCE_INLINE static BaseType_t xPortIsInsideInterrupt(
void )
{
   uint32_t ulCurrentInterrupt;
   BaseType_t xReturn;

   /* Obtain the number of the currently executing interrupt. */
   __asm volatile ( "mrs %0, ipsr" : "=r" ( ulCurrentInterrupt
)::"memory" );

   if( ulCurrentInterrupt == 0 )
   {
      xReturn = pdFALSE;
   }
   else
   {
      xReturn = pdTRUE;
   }

   return xReturn;
}
```

STM32CubeIDE



Add **#ifndef STM32G0B1xx**

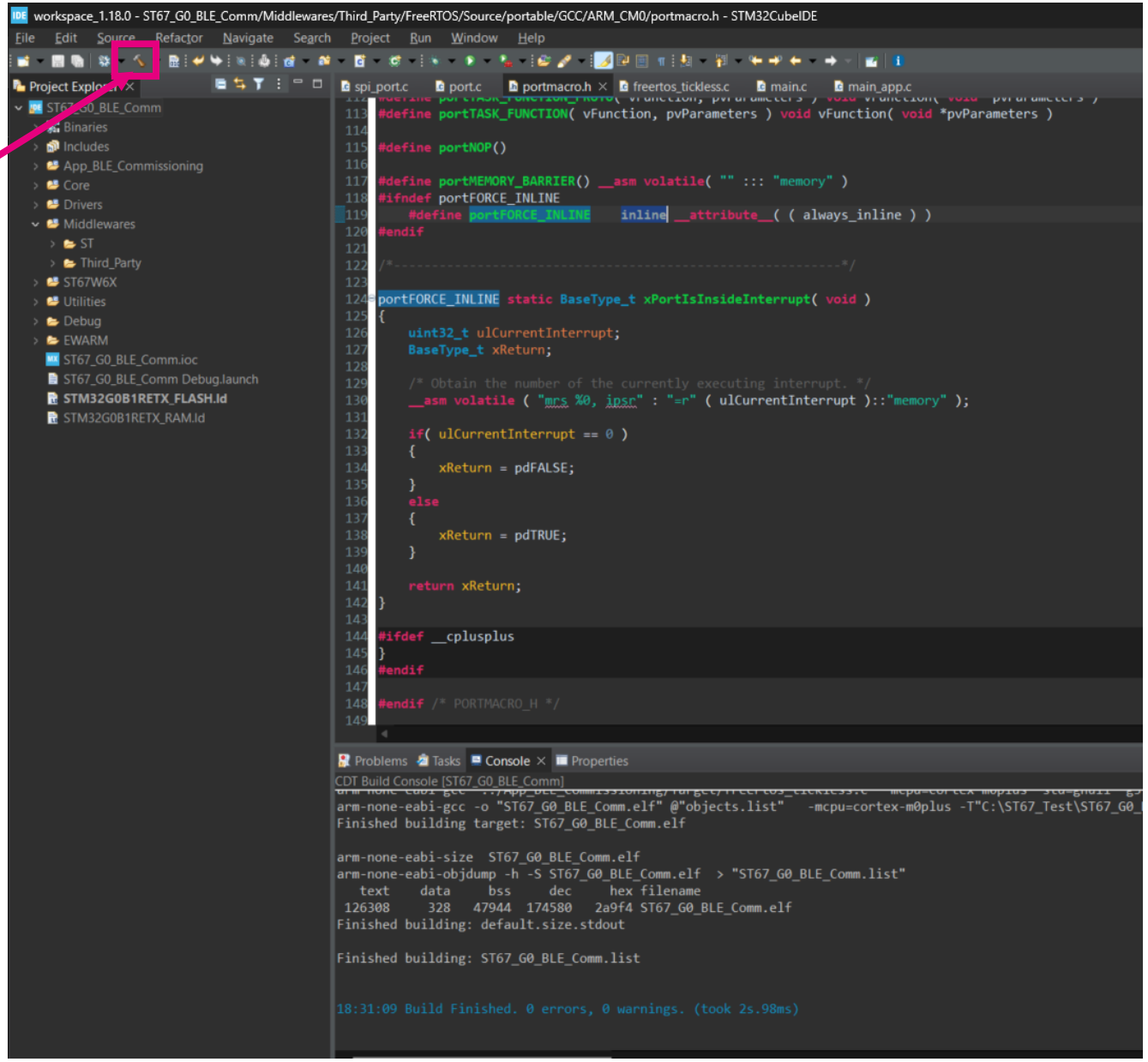
Here we are disabling the custom memcpy() definition, as we would be using the memcpy() definition in string.h

Project/ST67W6X/spi_port.c

Add **#endif**

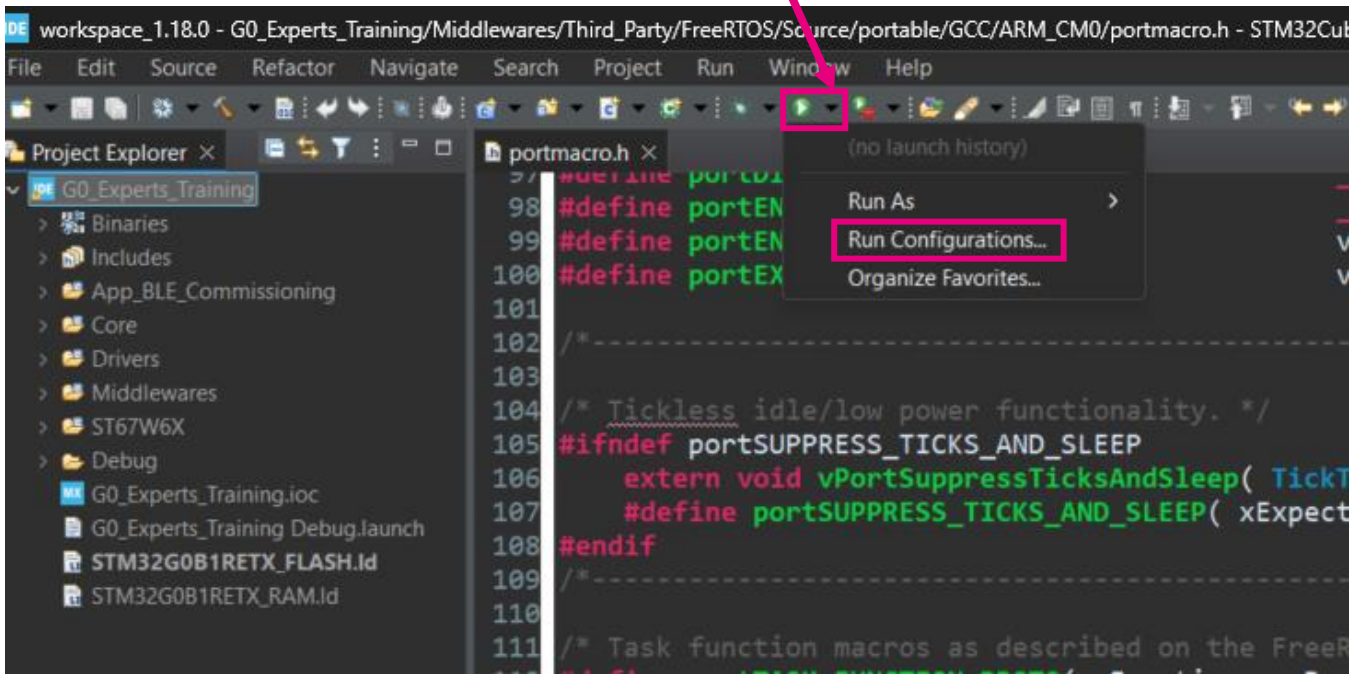
Build!

Build the project

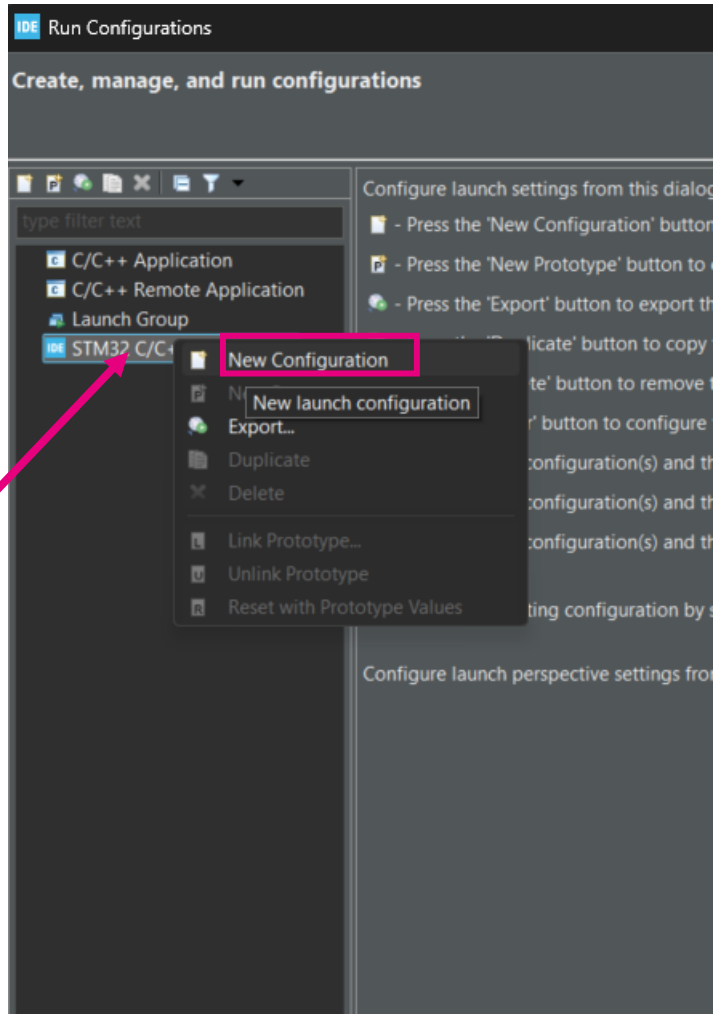


Flash!

Flash the project

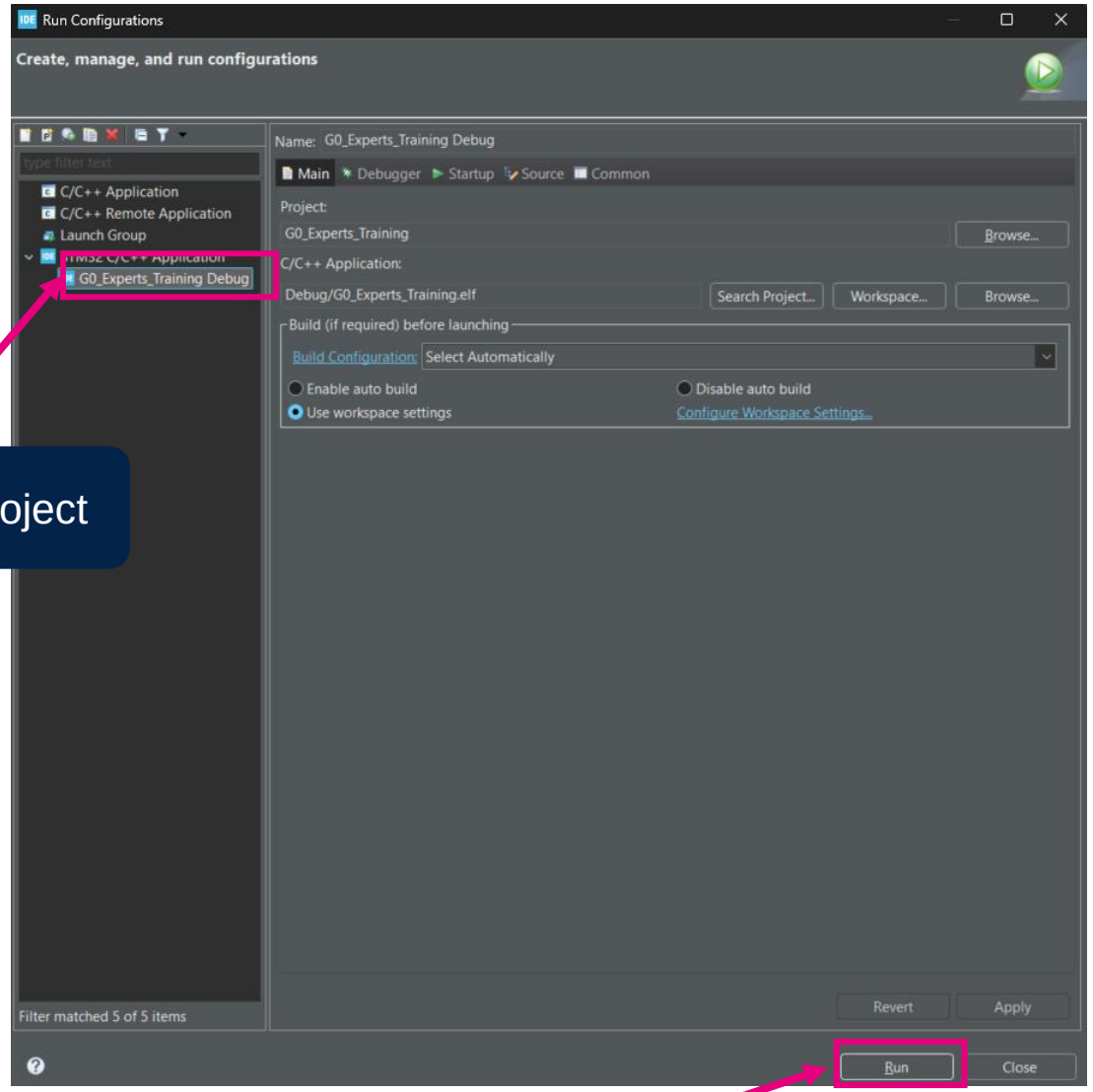


Right-Click on STM32 C/C++ application



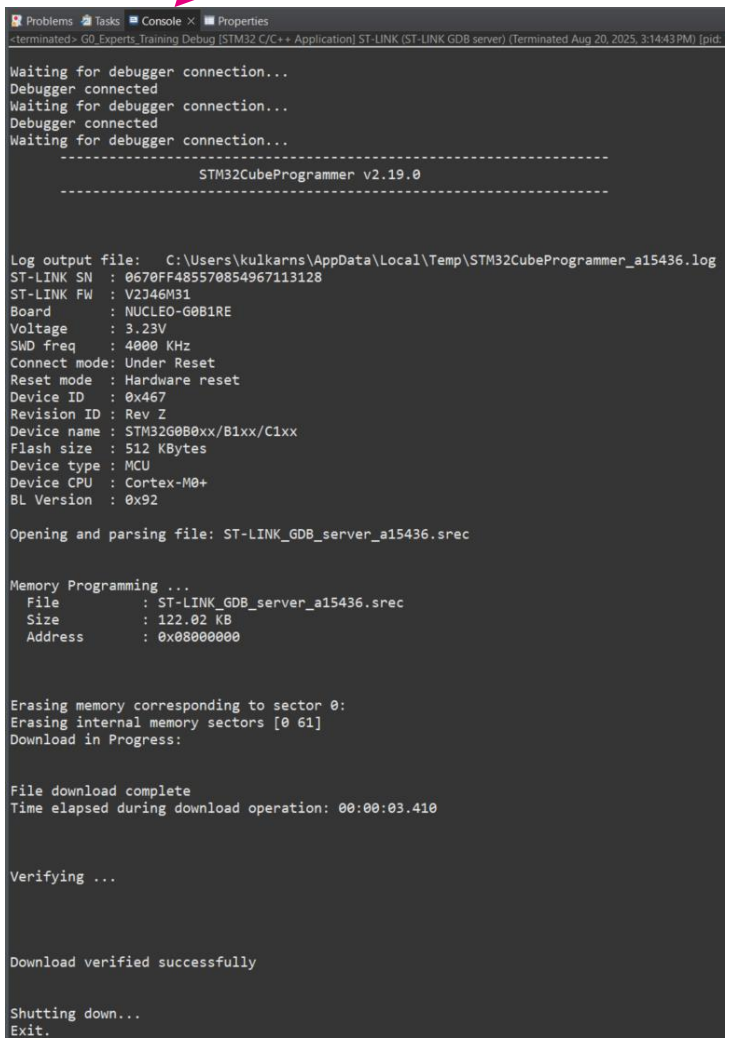
Flash!

Select the Project



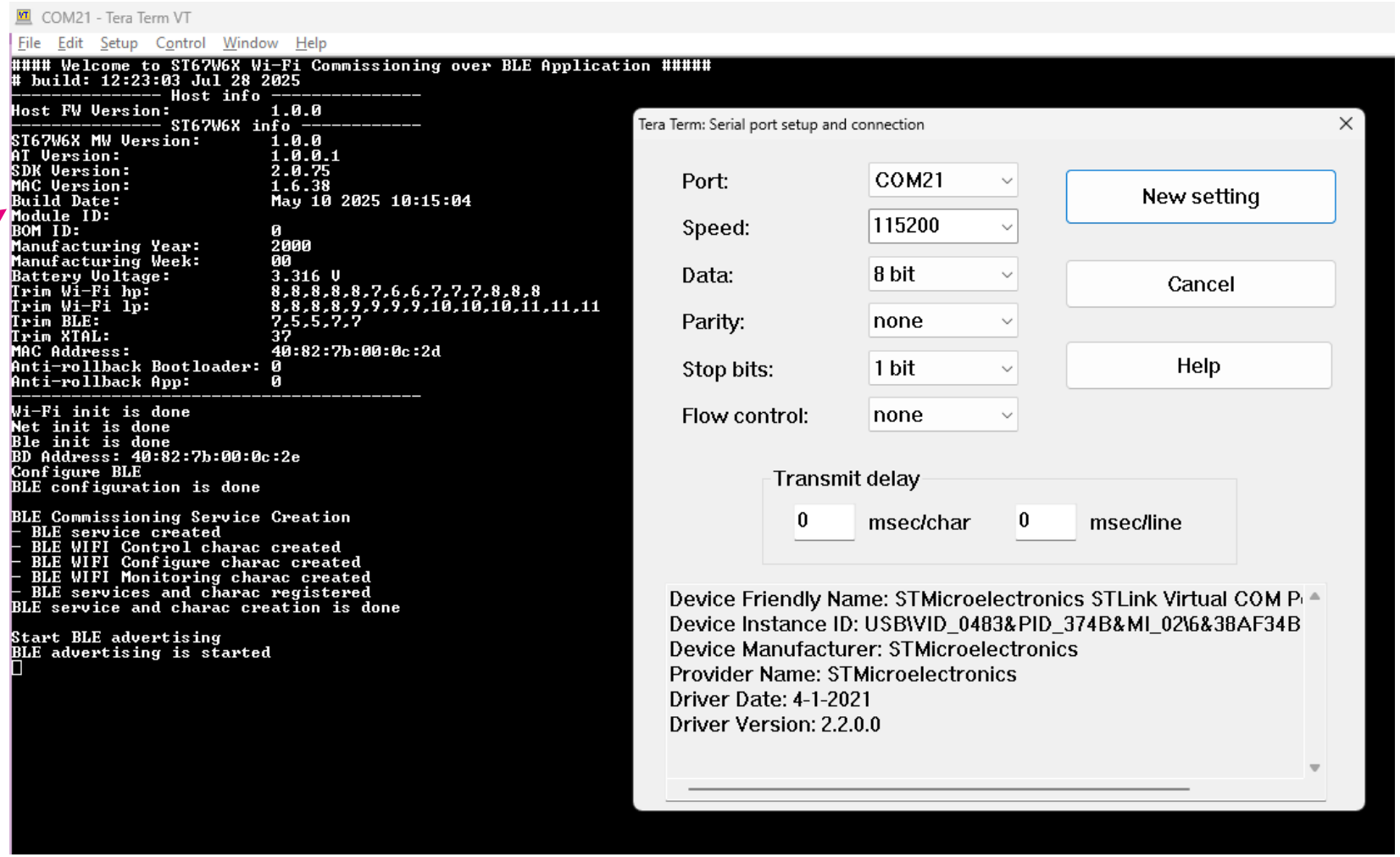
Run

Console logs after flashing the project



Tera Term Logs

Tera Term Logs after flashing



life.augmented

Documentation

Wifi wiki

Access:

The wifi wiki is opened to all

Wiki address of the main page

https://wiki.st.com/stm32mcu/wiki/Connectivity:Introduction_to_Wi-Fi

We will be happy to get your feedback!

Wiki Wi-Fi®: pages breakdown

About Wi-Fi

Details about ST67W611M1

X-CUBE-ST67W61

Host boards supported

Tools & References

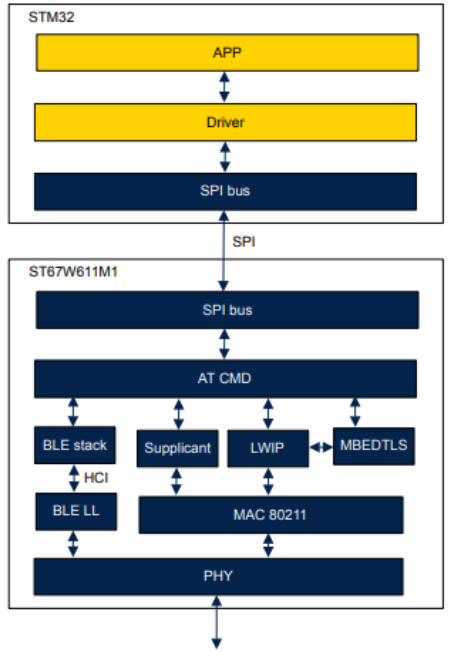
Other documentation

- Getting started with X-CUBE-ST67W61 here

2 X-CUBE-ST67W61 architecture overview

The X-CUBE-ST67W61 is a set of software components implementing host applications driving a Wi-Fi® and Bluetooth® LE coprocessor. The coprocessor is controlled via AT command over SPI interface.
The figure below illustrates the architecture of the solution:

Figure 1. **X-CUBE-ST67W61 architecture view**



- MW API documentation is available:
  - X-CUBE-ST67W61_V1.0.0\Middlewares\ST\ST67W6X_Network_Driver\Doc\ST67W6X_Network_Driver.chm

ST Customer Support

- ST Support Home
  - Tickets can be submitted under the part number X-CUBE-ST67W61 or ST67W611M1

Our technology starts with You

Find out more at www.st.com