



ST67W611 Wi-Fi 6 + Bluetooth® LE + Thread

Santa Clara Wireless Team

Agenda

X-Cube- ST67W61 Package

How to use X-CUBE-ST67W61
within STM32CubeMX
(Walk-through)

Device Setup and Flashing

Walk through of the MQTT
application

Introduction to MLC

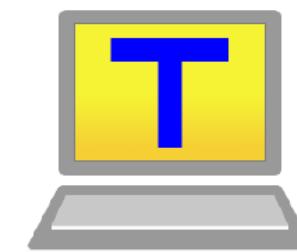
Demo: MLC Workflow in MEMS Studio

MQTT Hands-On
Integrating MLC into ST67 MQTT Demo

Documentation

Prerequisites

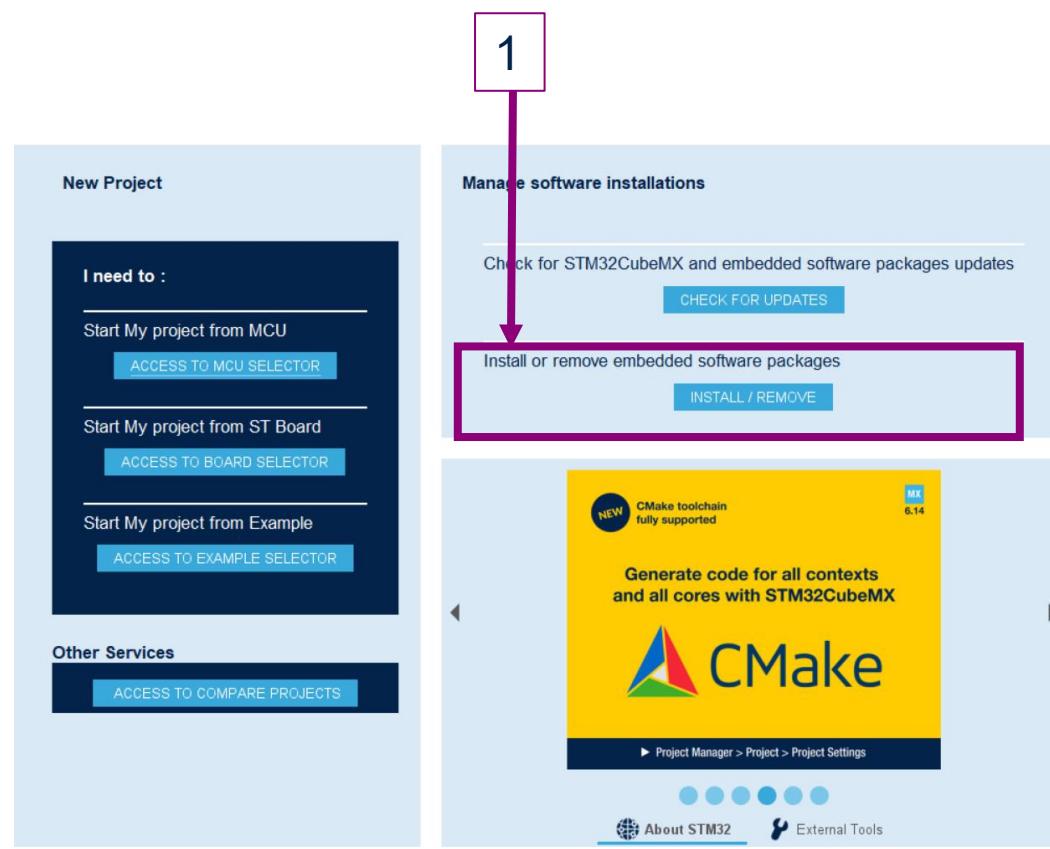
- [X_CUBE_ST67W61_V1.0.0](#)
- [STM32CubeIDE v1.16.1+](#)
- [STM32CubeProgrammer v2.18.0+](#)
- [STM32CubeMX v6.14.1+](#)
- [HyperTerminal – TeraTerm v.5.4.0](#)
- [Python v.3.12.7+](#)
- [STBLE Toolbox](#)
- IoT MQTT Panel App



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



This screenshot shows the 'STM32Cube MCU Packages and embedded software packs releases' page. At the top, there's a header with tabs for various companies. The 'STMicroelectronics' tab is highlighted with a red box and a red arrow pointing to it from step 2. Below the tabs, a list of packages is shown. The 'X-CUBE-ST67W61' package is highlighted with a red box and a red arrow pointing to it from step 3. The package details are visible at the bottom.

Company	Package	Description	Available Version
Infineon	X-CUBE-SMBUS		
RealThread	X-CUBE-ST60		
RowBoT	X-CUBE-ST67W61	Host applications driving a Wi-Fi® and Bluetooth® LE coprocessor (ST67W611M1).	1.0.0
CECER	X-CUBE-SUBG2		
WES			
emotas			
portGmbH			
quantropi			
wolfSSL			
Avnet-IoTConnect			
Cesanta			
EmbeddedOffice			
ITTIA_DB			

STM32U5 and X-CUBE-FREERTOS

MX Embedded Software Packages Manager

STM3Cube MCU Packages and embedded software packs releases

Releases Information was last refreshed 4 hours ago.

Infineon	RealThread	RoweBots	SEGGER	WES	emotas	portGmbH	quantropi	wolfSSL
STM3Cube MCU Packages	STMicroelectronics	Avnet-IoTConnect	Cesanta	EmbeddedOffice	ITIJA_DB			

Description Installed Version Available Version

▼ STM32U5

<input checked="" type="checkbox"/> STM3Cube MCU Package for STM32U5 Series	1.8.0	1.8.0
<input type="checkbox"/> STM3Cube MCU Package for STM32U5 Series (Size : 390 MB)		1.7.0
<input type="checkbox"/> STM3Cube MCU Package for STM32U5 Series (Size : 351 MB)		1.6.0
<input type="checkbox"/> STM3Cube MCU Package for STM32U5 Series (Size : 306 MB)	1.5.0	

Download latest STM32U5

MX Embedded Software Packages Manager

STM3Cube MCU Packages and embedded software packs releases

Releases Information was last refreshed 4 hours ago.

Infineon	RealThread	RoweBots	SEGGER	WES	emotas	portGmbH	quantropi	wolfSSL
STM3Cube MCU Packages	STMicroelectronics	Avnet-IoTConnect	Cesanta	EmbeddedOffice	ITIJA_DB			

Status Description Available Version

▼ STM3Cube Expansion Packages X-CUBE-FREERTOS

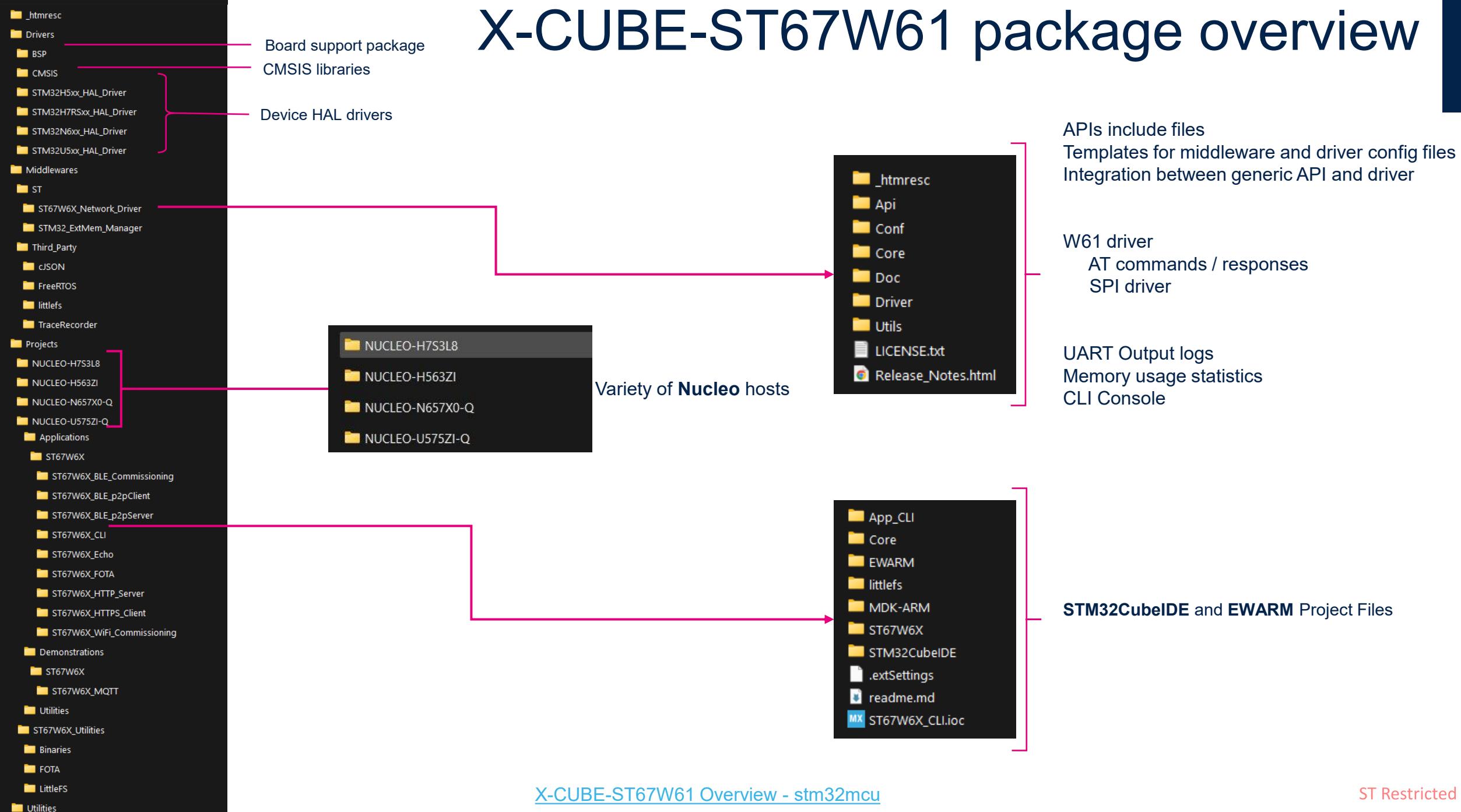
<input checked="" type="checkbox"/> STM32Cube X-CUBE-FREERTOS	FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series	1.3.1
<input type="checkbox"/> STM32Cube X-CUBE-FREERTOS	FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series (Size : 10.5 MB)	1.3.0
<input type="checkbox"/> STM32Cube X-CUBE-FREERTOS	FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series (Size : 10.5 MB)	1.2.0
<input type="checkbox"/> STM32Cube X-CUBE-FREERTOS	FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series (Size : 10.5 MB)	1.1.0

Download latest X-CUBE-FREERTOS



X-Cube-ST67W61

X-CUBE-ST67W61 package overview

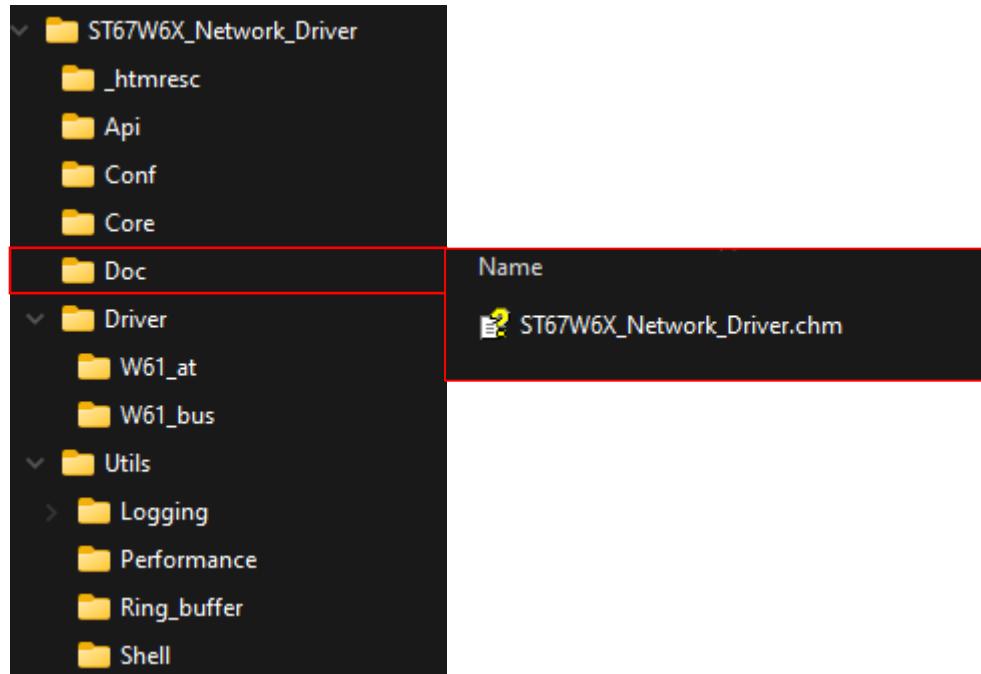
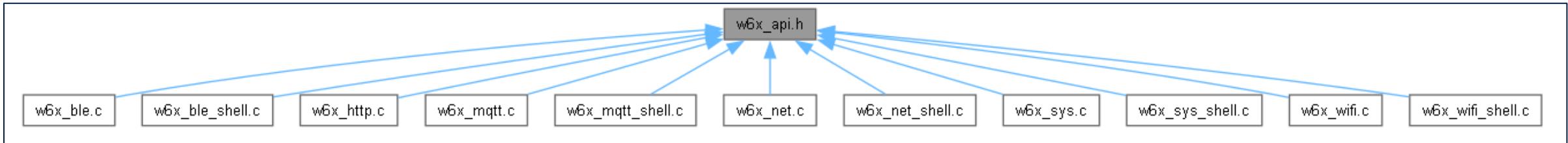


X-CUBE-ST67W1 middleware

	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™

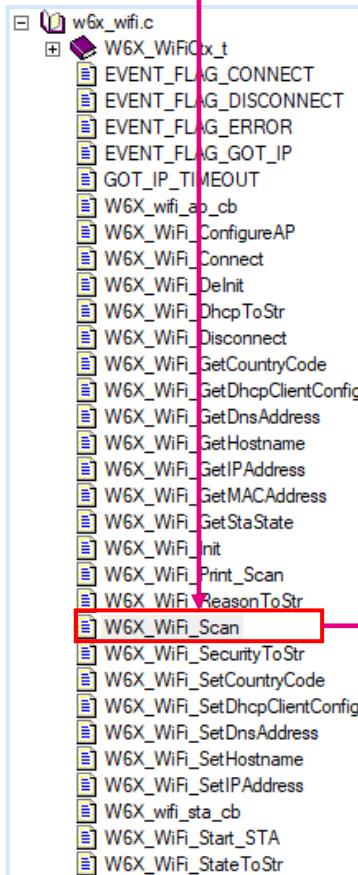
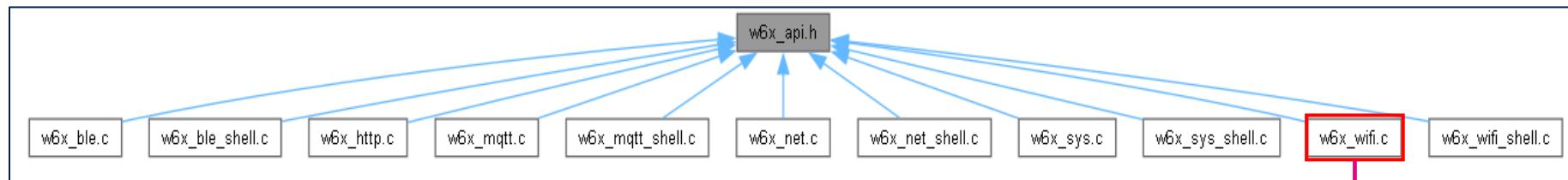
API Documentation



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

◆ **W6X_WiFi_Scan()**

```
W6X_Status_t W6X_WiFi_Scan (W6X_Scan_Opts_t * Opt,
                             scan_result_cb_t cb)
```

List a defined number of available access points.

Parameters

- Opts Scan options
- cb Callback to handle scan results

Returns

- Operation status

Definition at line 260 of file [w6x_wifi.c](#).

References [W61_WiFi_Scan\(\)](#), and [W61_WiFi_SetScanOpts\(\)](#).

Referenced by [W6X_Shell_WiFi_Scan\(\)](#).

Here is the call graph for this function:

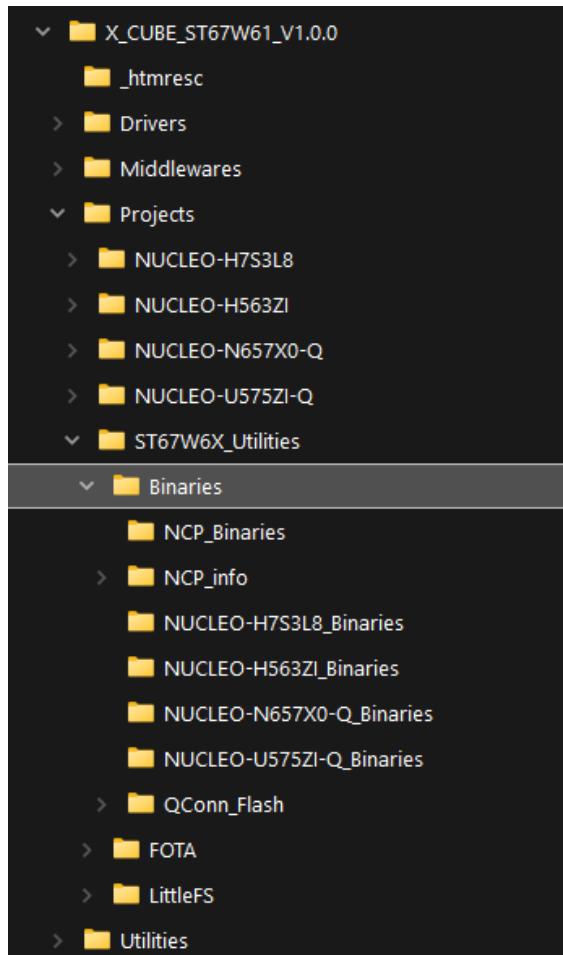
```
graph TD; W6X_WiFi_Scan[W6X_WiFi_Scan] --> W61_WiFi_Scan[W61_WiFi_Scan]; W6X_WiFi_Scan --> W61_WiFi_SetScanOpts[W61_WiFi_SetScanOpts]; W61_WiFi_Scan --> AT_SetExecute[AT_SetExecute]; W61_WiFi_SetScanOpts --> ATLock[ATLock]; W61_WiFi_SetScanOpts --> ATUnlock[ATUnlock]; AT_SetExecute --> AT_ParseOkErr[AT_ParseOkErr]; AT_SetExecute --> ATrecv[ATrecv]; AT_SetExecute --> ATLock
```

X-CUBE-ST67W61 applications

Applications
ST67W6X
ST67W6X_BLE_Commissioning
ST67W6X_BLE_p2pClient
ST67W6X_BLE_p2pServer
ST67W6X_CLI
ST67W6X_Echo
ST67W6X_FOTA
ST67W6X_HTTP_Server
ST67W6X_HTTPS_Client
ST67W6X_WiFi_Commissioning
Demonstrations
ST67W6X
ST67W6X_MQTT

	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

X-CUBE-ST67W1 NCP tools



Scripts for flashing the latest mission or manufacturing firmware:

..\\x-cube-st67w61-v1.0.0\\Projects\\ST67W6X_Utilsities\\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.
NCP_update_mission_profile.bat	Script to flash the ST67 module with the mission profile application.

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

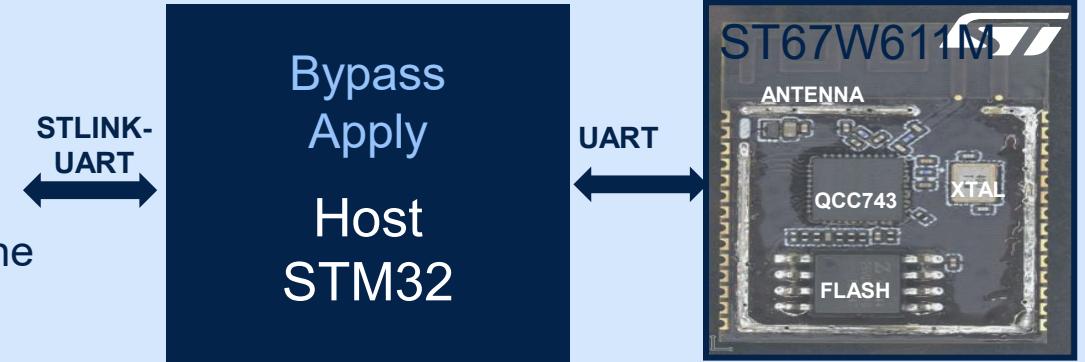


Flashing ST67 & Device Setup

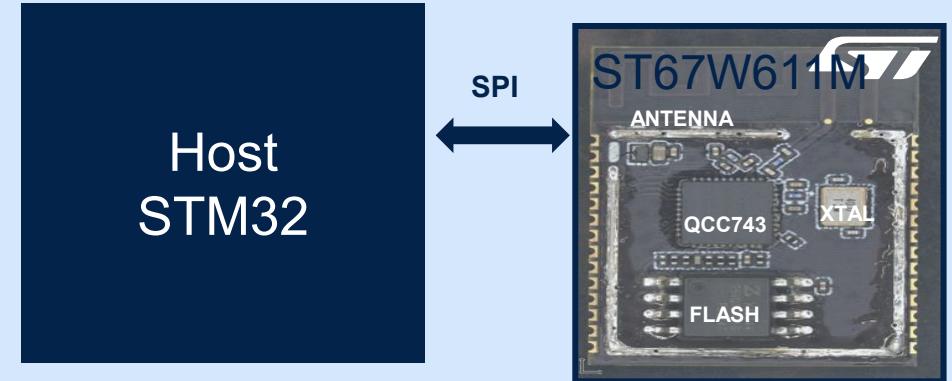
ST67W611M Modes of Operation

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

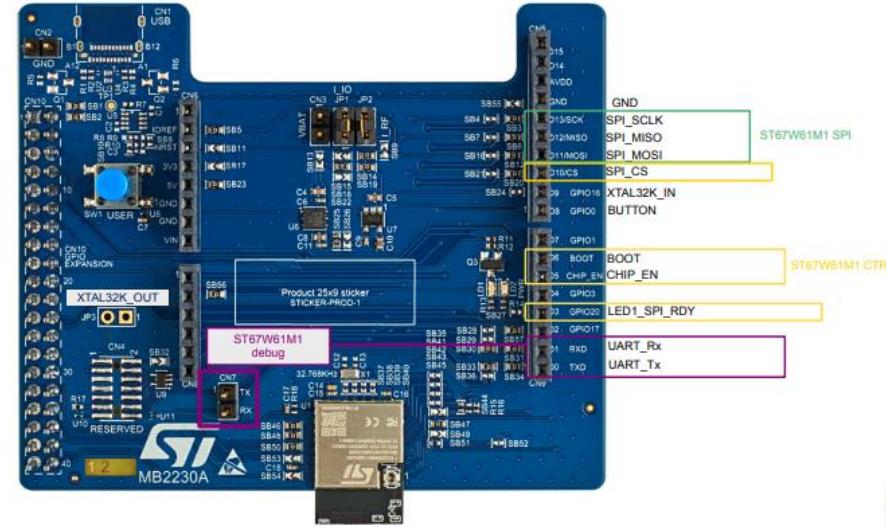


- **Mission mode (Functional)**
 - Implementing the AT command for the different supported protocol (WIFI / BLE / MQTT /..) through the **SPI** interface.
 - The ROM boot loader loads and executes the program from SPI Flash to boot the system.

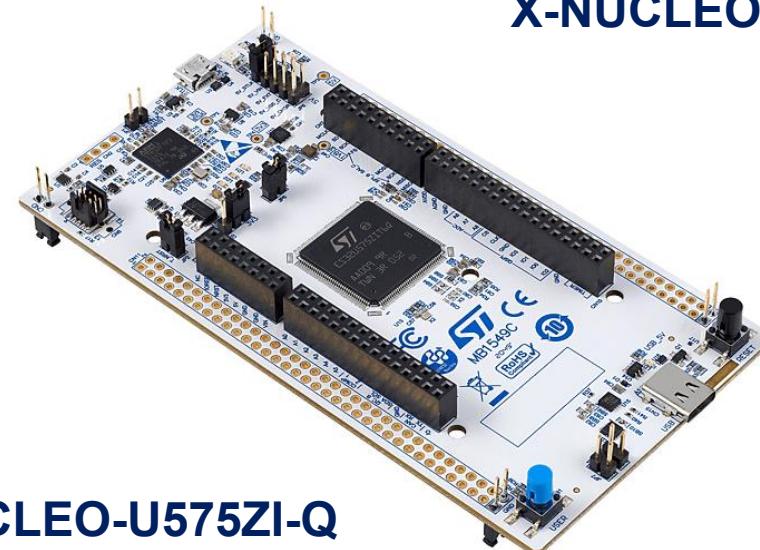


STM32U575 and ST67W611M configuration

- 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

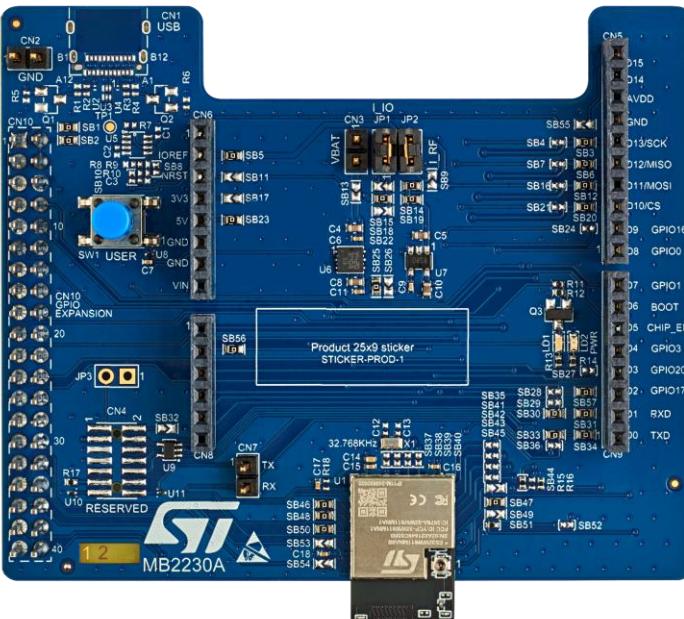


X-NUCLEO-67W61M1



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.

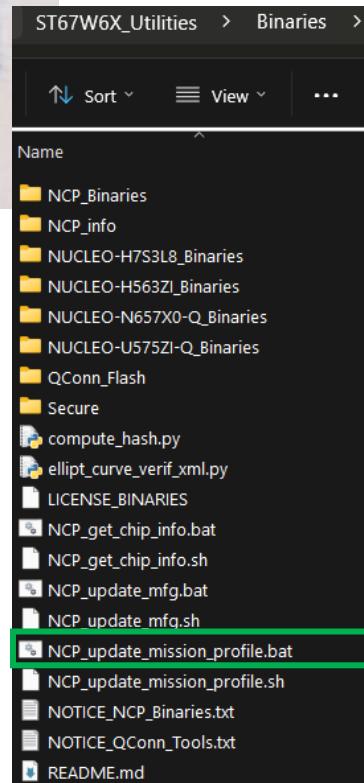
Flashing ST67W611M1

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_Utilsities\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.

```
"#####
## 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: |
```

Flashing ST67W611M1 Mission Mode (Functional Mode) Process

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
```

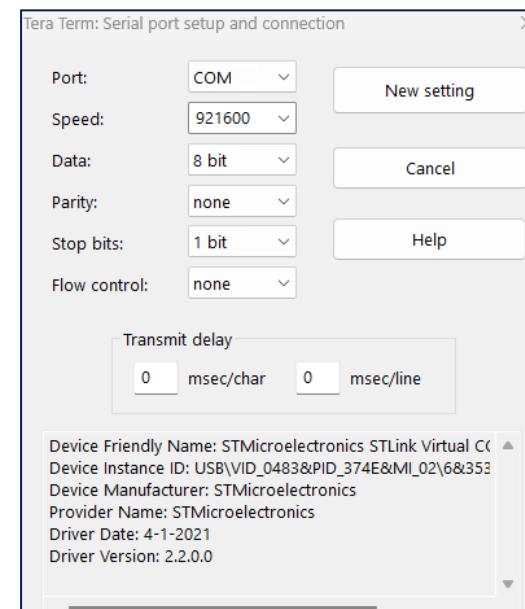
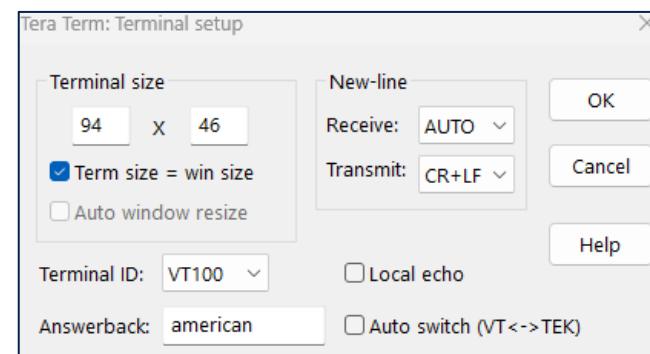
2. The NCP_update_mission_profile.bat will have 3 sequential flashing stages. The script will flash the STM32U5 with Bootloader.bin, then the ST67 with the signed mission mode binary, and the STM32U5 with ST67W6X_CLI.bin. Please verify success of all 3 flashing stages.

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: 6,6,6,6,6,5,5,6,6,7,7,7
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
```



How to use X-CUBE-ST67W61 within STM32CubeMX

Software Development Tools



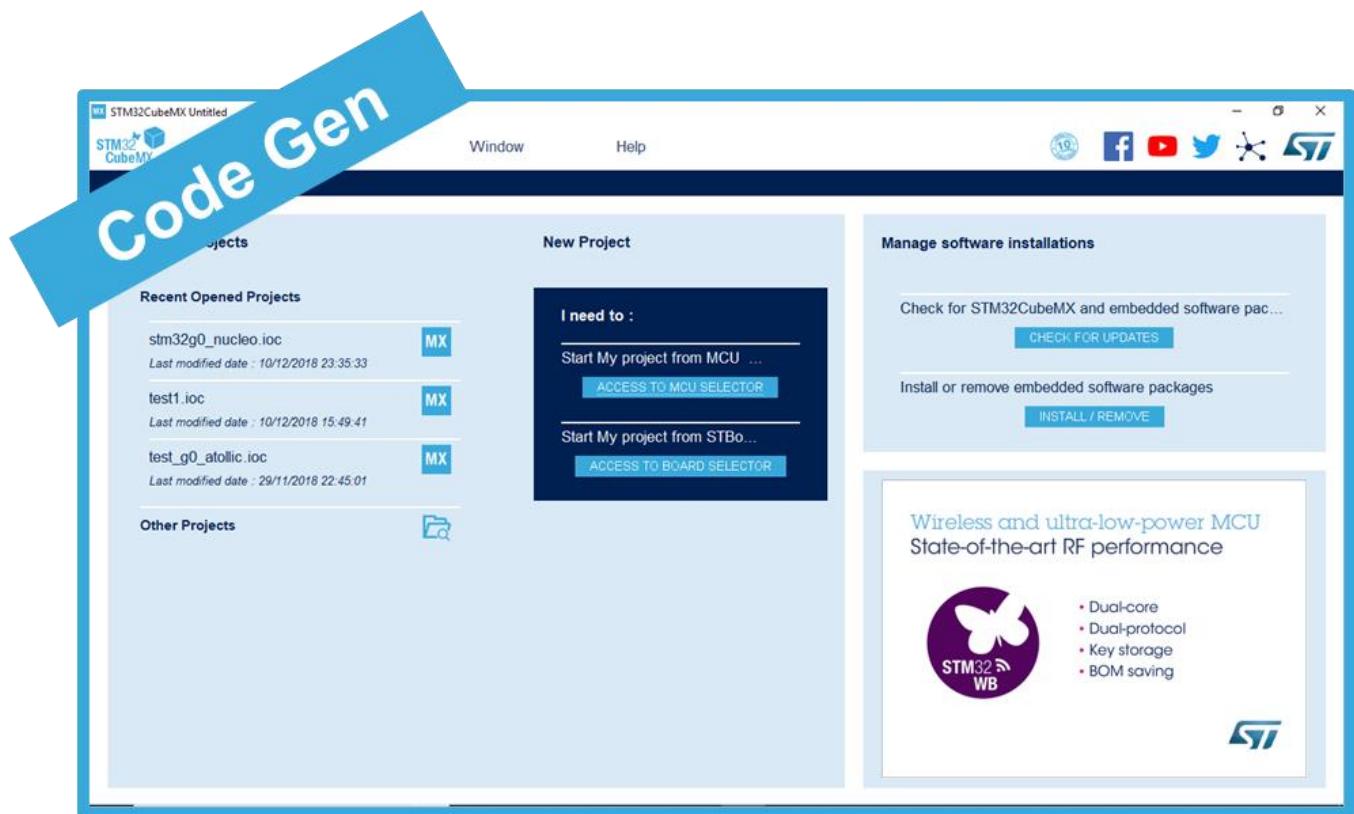
"STM32CubeIDE"
development environment

↳ **"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

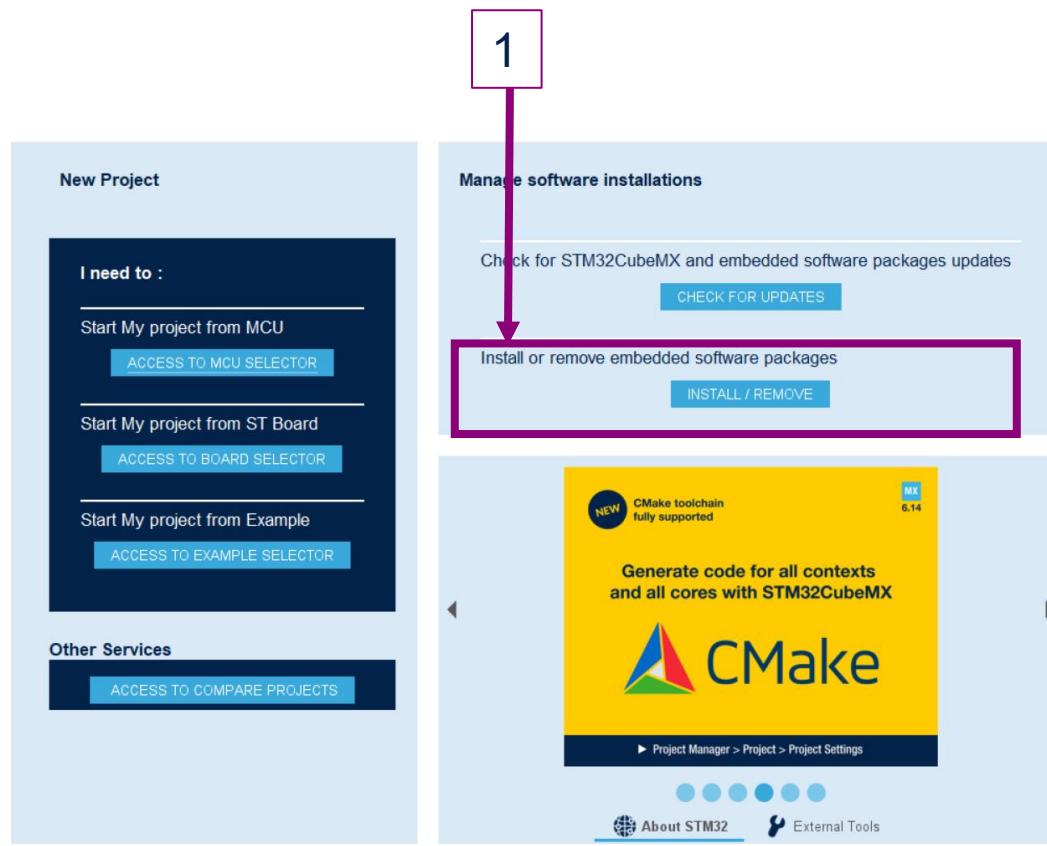
Pinouts, clock tree, peripherals and middleware configuration.

Expandable to support wireless connectivity, sensors and much more!

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



The screenshot shows the 'STM32Cube MCU Packages and embedded software packs releases' page. At the top, there's a header with tabs for various companies. The 'STMicroelectronics' tab is highlighted with a red box and a red arrow pointing to it from step 2. Below the header, there's a list of packages. The 'X-CUBE-ST67W61' package is highlighted with a red box and a red arrow pointing to it from step 3. The package details show it's for host applications driving a Wi-Fi® and Bluetooth® LE coprocessor (ST67W611M1) and is version 1.0.0.

Company	Status	Description	Available Version
Infineon		X-CUBE-SMBUS	
RealThread		X-CUBE-ST60	
RowBoT			
CECER			
WES			
emotas			
portGmbH			
quantropi			
wolfSSL			
STM32Cube MCU Packages			
STMicroelectronics		X-CUBE-ST67W61	1.0.0
Avnet-IoTConnect			
Cesanta			
EmbeddedOffice			
ITTIA_DB			

STM32U5 and X-CUBE-FREERTOS

MX Embedded Software Packages Manager

STM3Cube MCU Packages and embedded software packs releases

Releases Information was last refreshed 4 hours ago.

Infineon	RealThread	RoweBots	SEGGER	WES	emotas	portGmbH	quantropi	wolfSSL
STM3Cube MCU Packages	STMicroelectronics	Avnet-IoTConnect	Cesanta	EmbeddedOffice	ITIJA_DB			

Description Installed Version Available Version

▼ STM32U5

STM32Cube MCU Package for STM32U5 Series	1.8.0	1.8.0
STM32Cube MCU Package for STM32U5 Series (Size : 390 MB)		1.7.0
STM32Cube MCU Package for STM32U5 Series (Size : 351 MB)		1.6.0
STM32Cube MCU Package for STM32U5 Series (Size : 306 MB)	1.5.0	

Download latest STM32U5

MX Embedded Software Packages Manager

STM3Cube MCU Packages and embedded software packs releases

Releases Information was last refreshed 4 hours ago.

Infineon	RealThread	RoweBots	SEGGER	WES	emotas	portGmbH	quantropi	wolfSSL
STM3Cube MCU Packages	STMicroelectronics	Avnet-IoTConnect	Cesanta	EmbeddedOffice	ITIJA_DB			

Status Description Available Version

▼ X-CUBE-FREERTOS

FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series	1.3.1
FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series (Size : 10.1 MB)	1.3.0
FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series (Size : 10.1 MB)	1.2.0
FreeRTOS STM32Cube expansion package for STM32H5/U5/WBA/C0/U0/N6/U3 series (Size : 10.1 MB)	1.1.0

Download latest X-CUBE-FREERTOS

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 walking through 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.



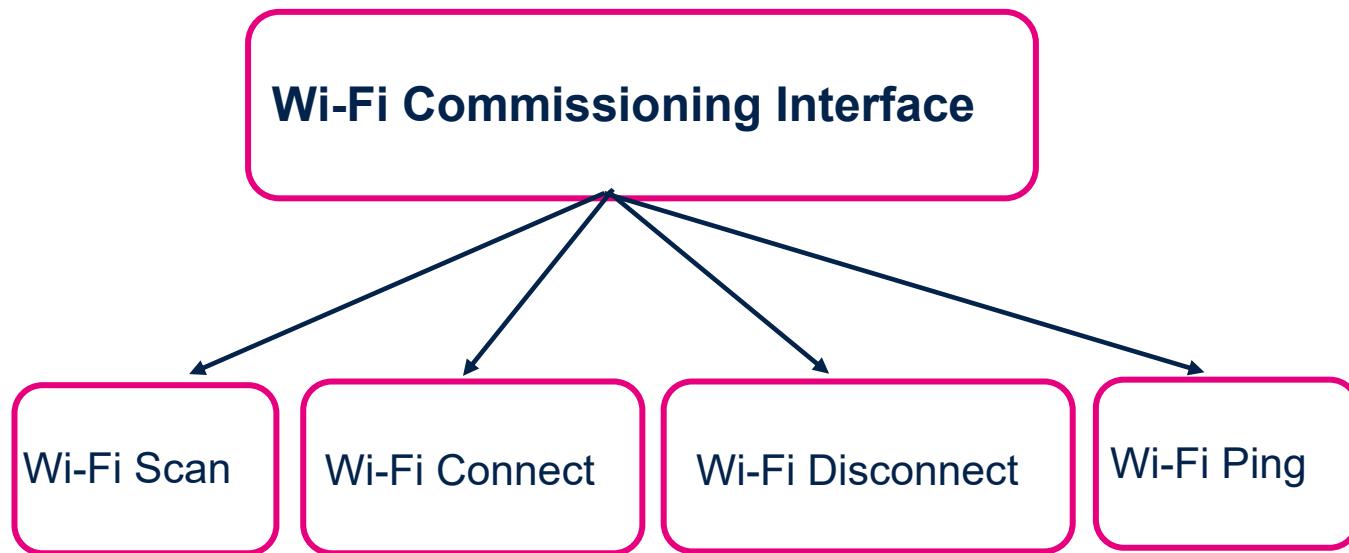
BLE Commissioning App cont..

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

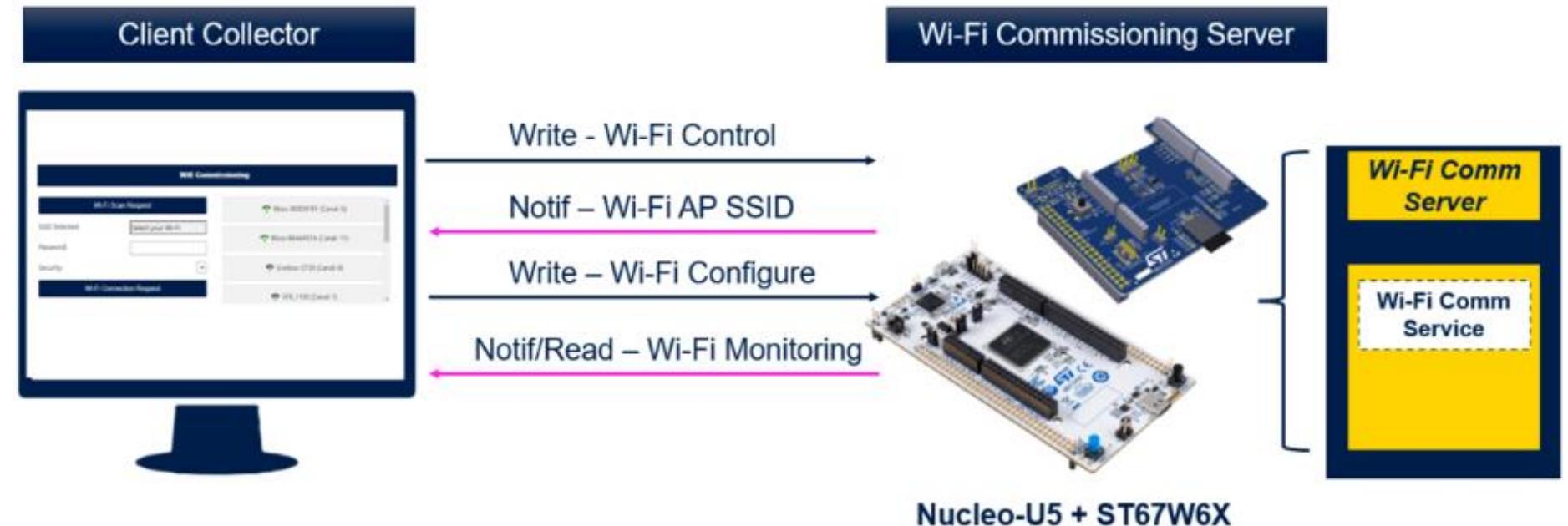
The Wi-Fi commissioning application demonstrates point to point communication using proprietary service and characteristics. It acts as a Peripheral device with one GATT service and four characteristics.

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.

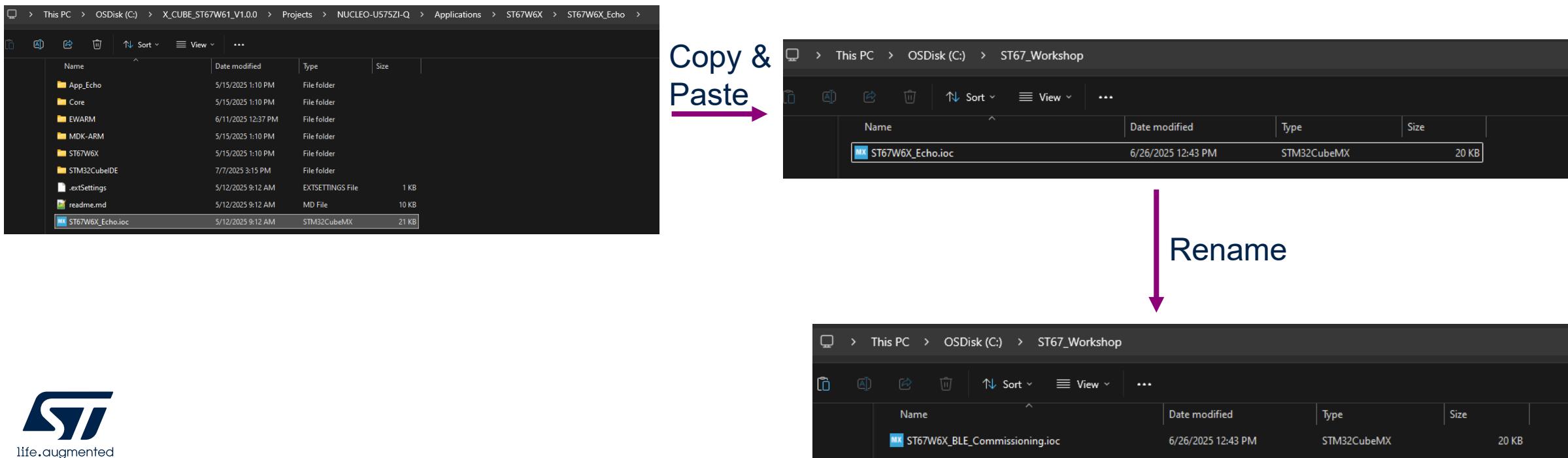


Generating BLE Commissioning
app using NUCLEO-U575ZI-Q as
host processor (Hands-on)

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.

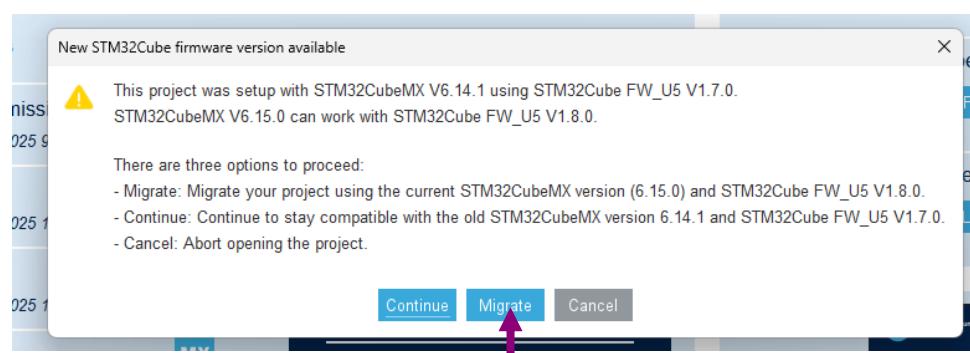
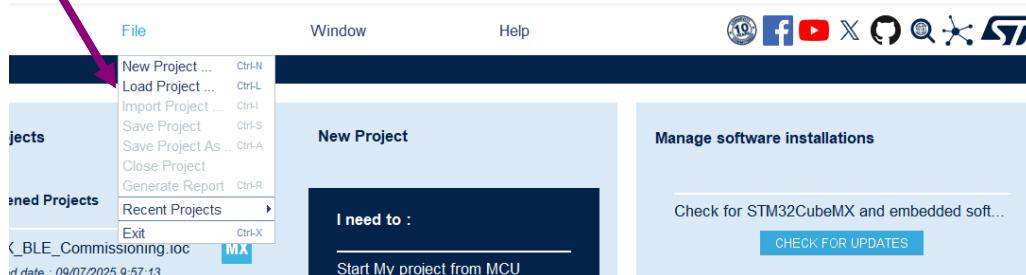
1. Browse through C:\X_CUBE_ST67W61_V1.0.0\Projects\NUCLEO-U575ZI_Q\Applications\ST67W6X\ST67W6X_Echo
2. Copy ST67W6X_Echo.ioc
3. Create a new directory C:\ST67_Workshop
4. Paste and rename to ST67W6X_BLE_Commissioning.ioc



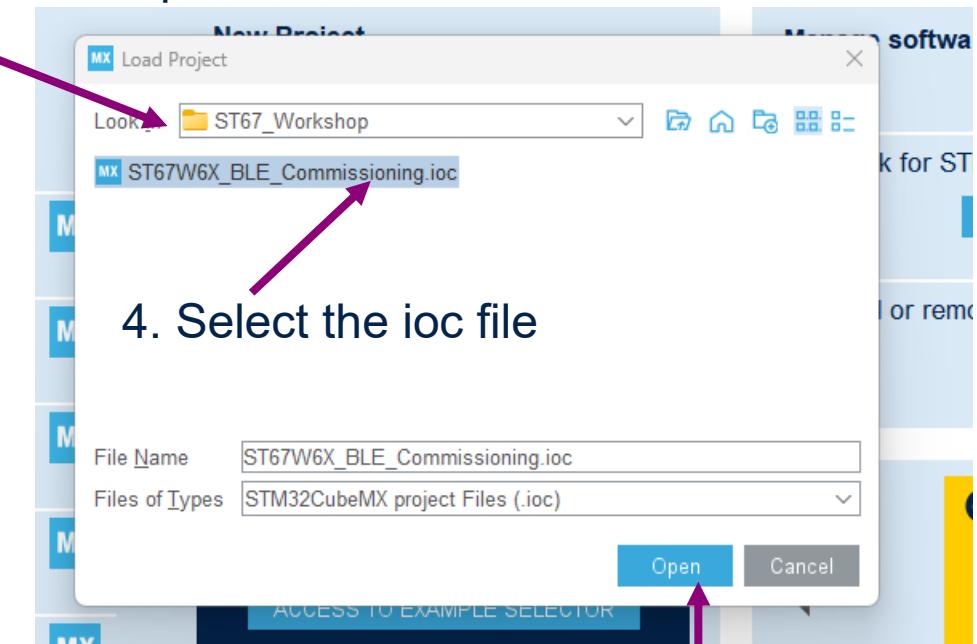
Loading ioc file in STM32CubeMX

1. Run STM32CubeMX with admin rights

2. Load Project



3. Go the ST67_Workshop



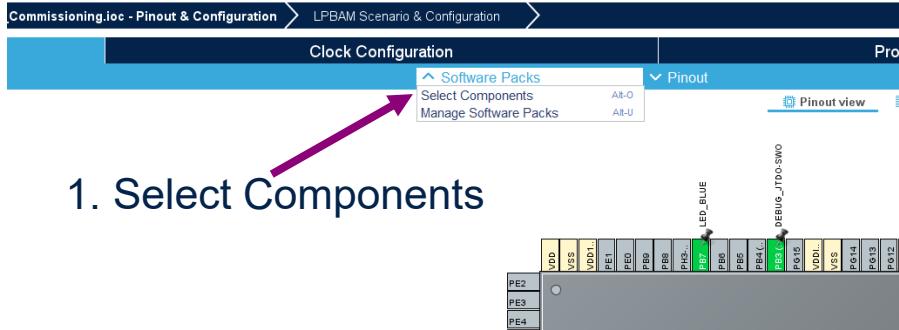
4. Select the ioc file



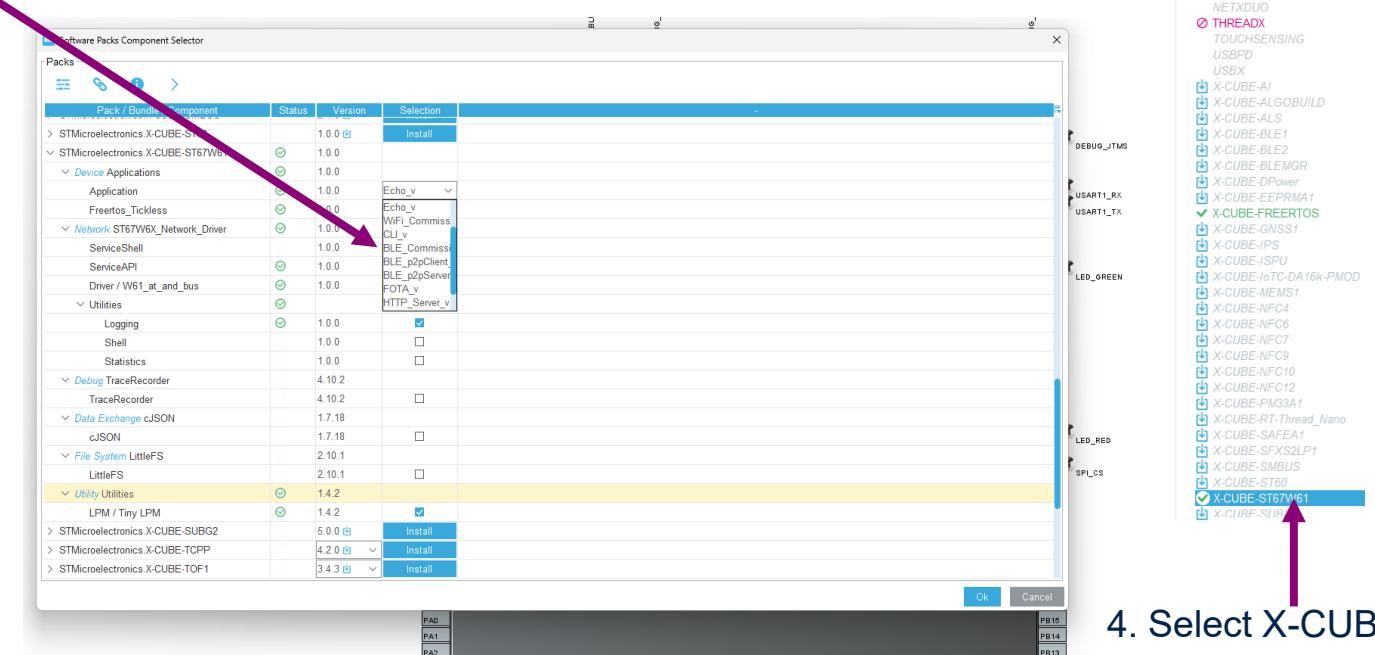
6. Migrate (If you get the warning)

Configuration

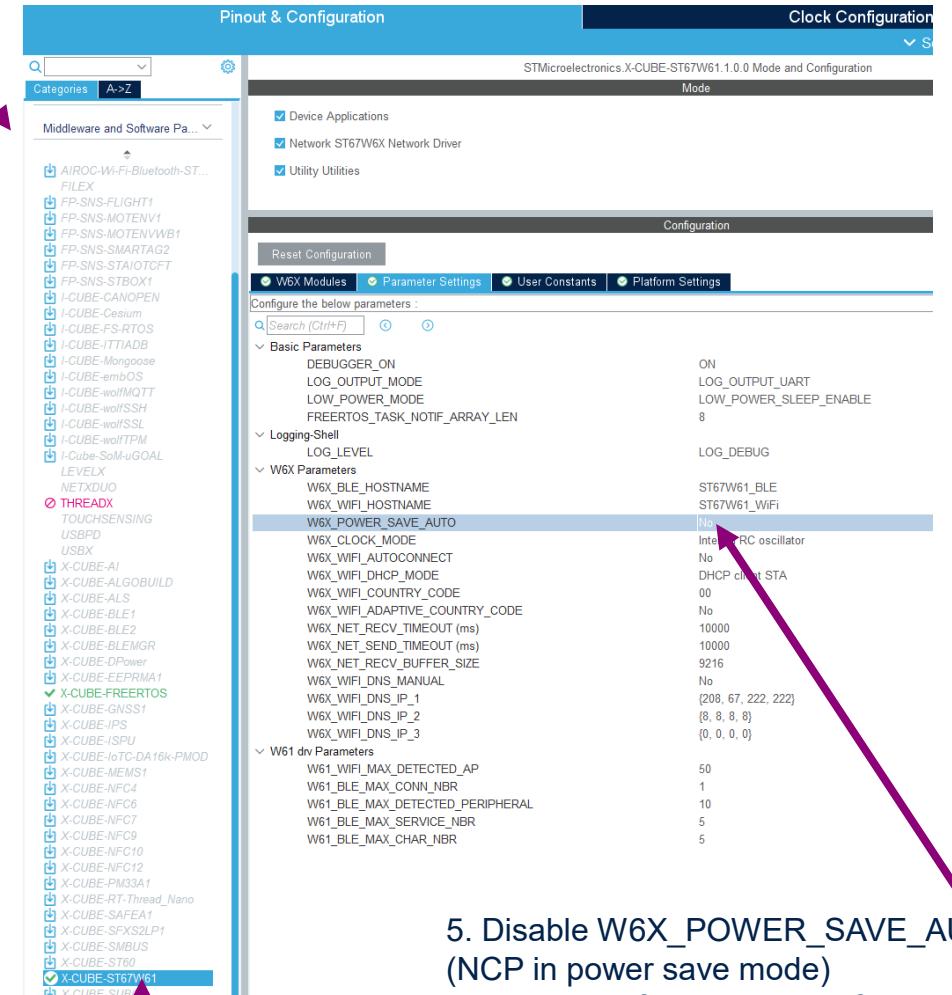
3. Under Middleware and Software Packs



1. Select Components



Commissioning app



4. Select X-CUBE-ST67W61

5. Disable W6X_POWER_SAVE_AUTO
(NCP in power save mode)
Note: This define must be 0 if BLE is enabled

Project Settings

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)

Project Settings cont..

The screenshot shows the STM32CubeMX software's Project Settings interface. The left sidebar has tabs for 'Project', 'Code Generator' (which is selected and highlighted with a red box), and 'Advanced Settings'. The main area has tabs for 'Pinout & Configuration' and 'Clock Configuration' (which is selected and highlighted with a blue bar). In the 'Code Generator' section, there are three radio button options under 'STM32Cube MCU packages and embedded software packs':

- Copy all used libraries into the project folder
- Copy only the necessary library files
- Add necessary library files as reference in the toolchain project configuration file

A purple arrow points from the text '3. Enable “Copy only the necessary files”' to the second radio button. Other sections visible include 'Generated files' (with checkboxes for peripheral initialization, backup, user code, and delete previous) and 'HAL Settings' (with checkboxes for analog pins and full assert).

3. Enable “Copy only the necessary files”

Generate Code

The screenshot shows the STM32CubeMX software interface with the following details:

- Project Path:** Home > STM32U575ZITxQ - NUCLEO-U575ZI-Q > ST67W6X_BLE_Commissioning.ioc - Project Manager > LPBAM Scenario & Configuration
- Toolbar:** GENERATE CODE (highlighted with a red box and arrow)
- Code Generator Tab:** Contains a table of generated function calls. One row is highlighted with a red box and arrow, showing the 'Generate Code' column checked.
- Advanced Settings:** A section on the left containing a button labeled "Advanced Settings".

Generate Code	Rank	Function Name	Peripheral Instance Name	<input type="checkbox"/> Do Not Generate Function Call	<input type="checkbox"/> Visibility (Static)
<input checked="" type="checkbox"/>	1	SystemClock_Config	RCC	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	2	MX_GPIO_Init	GPIO	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	3	MX_GPDMA1_Init	GPDMA1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	4	MX_ICACHE_Init	ICACHE	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	5	MX_SPI1_Init	SPI1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	6	MX_USART1_UART_Init	USART1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	7	MX_LPTIM1_Init	LPTIM1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	10	MX_App_BLE_Commissioning_Init	STMicroelectronics.X-CUBE-ST67W61.1.0.0	<input checked="" type="checkbox"/>	<input type="checkbox"/>

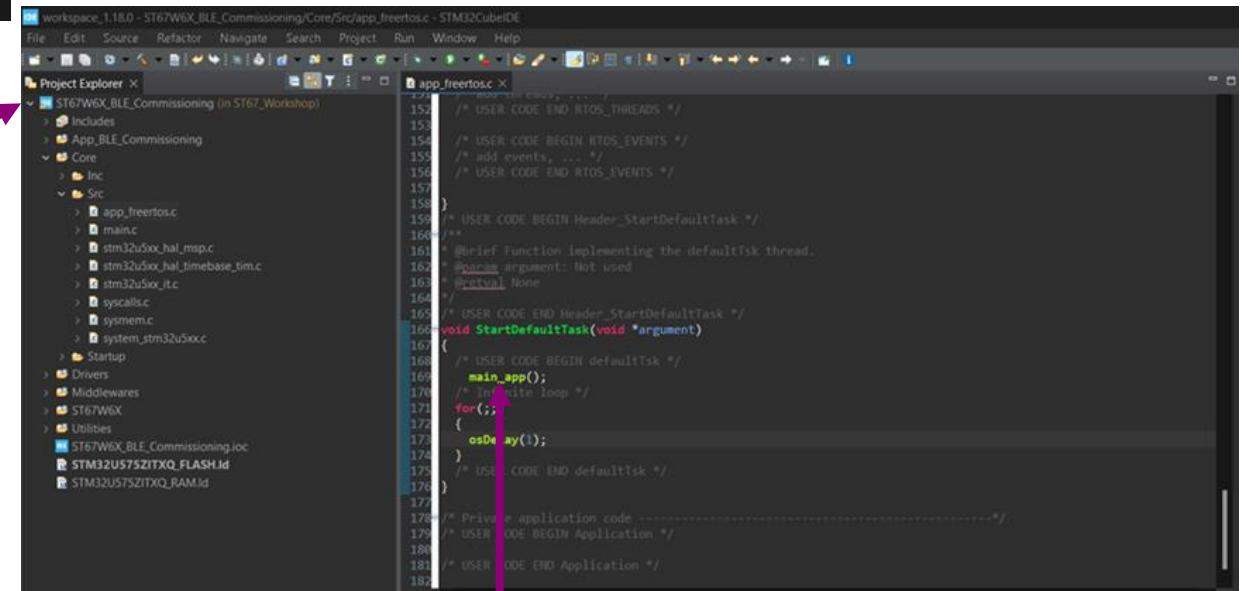
5. Generate Code

4. Enable this option
(here: we are not generating
this function call)

STM32CubeIDE

Name	Date modified	Type	Size
App_BLE_Commissioning	7/10/2025 12:05 PM	File folder	
Core	7/10/2025 12:06 PM	File folder	
Drivers	7/10/2025 12:06 PM	File folder	
Middlewares	7/10/2025 12:06 PM	File folder	
ST67W6X	7/10/2025 12:05 PM	File folder	
Utilities	7/10/2025 12:06 PM	File folder	
.cproject	7/10/2025 12:06 PM	CPROJECT File	35 KB
.mxproject	7/10/2025 12:06 PM	MXPROJECT File	17 KB
.project	7/10/2025 12:06 PM	PROJECT File	2 KB
STM32U575ZITXQ_FLASH.id	7/10/2025 12:05 PM	STM32CubeMX	21 KB
STM32U575ZITXQ_RAM.id	7/10/2025 12:06 PM	LD File	5 KB
STM32U575ZITXQ.id	7/10/2025 12:06 PM	LD File	5 KB

Load the project on the STM32CubeIDE



The screenshot shows the STM32CubeIDE interface. The Project Explorer on the left lists the project structure under 'ST67W6X.BLE_Commissioning (in ST67_Workshop)'. The code editor on the right displays the 'app_freertos.c' file. A pink arrow points from the text 'Load the project on the STM32CubeIDE' to the Project Explorer. Another pink arrow points from the text 'After, generating the code, you would view similar folder structure under ST67_Workshop' to the code editor area.

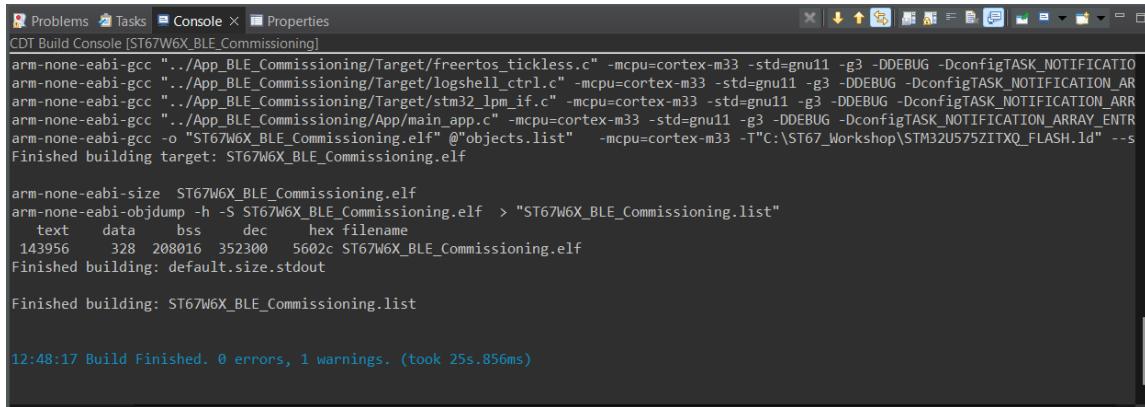
```
/* USER CODE BEGIN Header_StartDefaultTask */
void StartDefaultTask(void *argument)
{
    /* USER CODE BEGIN defaultTask */
    main_app();
    /* infinite loop */
    for(;;)
    {
        osDelay(1);
    }
    /* USER CODE END defaultTask */
}

/* Private application code */
/* USER CODE BEGIN Application */
/* USER CODE END Application */

```

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

Time to build and flash



```
Problems Tasks Console Properties
CDT Build Console [ST67W6X_BLE_Commissioning]
arm-none-eabi-gcc ".../App_BLE_Commissioning/Target/freertos_tickless.c" -mcpu=cortex-m33 -std=gnu11 -g3 -DDEBUG -DconfigTASK_NOTIFICATION_ARRAY_ENTRIES
arm-none-eabi-gcc ".../App_BLE_Commissioning/Target/logshell_ctrl.c" -mcpu=cortex-m33 -std=gnu11 -g3 -DDEBUG -DconfigTASK_NOTIFICATION_ARRAY_ENTRIES
arm-none-eabi-gcc ".../App_BLE_Commissioning/Target/stm32_lpm_if.c" -mcpu=cortex-m33 -std=gnu11 -g3 -DDEBUG -DconfigTASK_NOTIFICATION_ARRAY_ENTRIES
arm-none-eabi-gcc ".../App_BLE_Commissioning/App/main_app.c" -mcpu=cortex-m33 -std=gnu11 -g3 -DDEBUG -DconfigTASK_NOTIFICATION_ARRAY_ENTRIES
arm-none-eabi-gcc -o "ST67W6X_BLE_Commissioning.elf" "@objects.list" -mcpu=cortex-m33 -T"C:\ST67_Worksop\STM32U575ZITXQ_FLASH.ld" --s
Finished building target: ST67W6X_BLE_Commissioning.elf

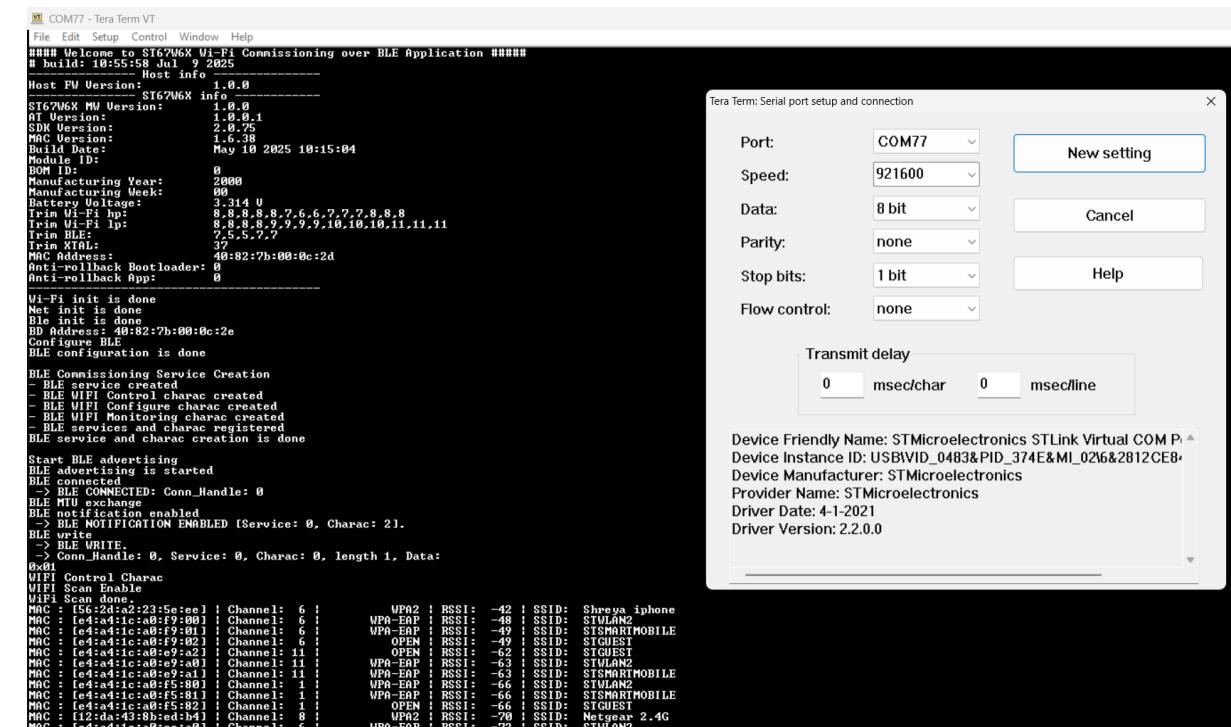
arm-none-eabi-size ST67W6X_BLE_Commissioning.elf
arm-none-eabi-objdump -h -S ST67W6X_BLE_Commissioning.elf > "ST67W6X_BLE_Commissioning.list"
    text      data      bss      dec      hex filename
143956      328   208016   352300   5602c ST67W6X_BLE_Commissioning.elf
Finished building: default.size.stdout

Finished building: ST67W6X_BLE_Commissioning.list

12:48:17 Build Finished. 0 errors, 1 warnings. (took 25s.856ms)
```



Console after building the application



COM77 - Tera Term VT

File Edit Setup Control Window Help

Welcome to ST67W6X Wi-Fi Commissioning over BLE Application

build: 10:55:58 Jul 9 2025

Host FW Version: 1.0.0

ST67W6X FW Version: 1.0.0

AT Version: 1.0.0.1

SDK Version: 2.0.75

MMC Version: 1.6.35

Build Date: May 10 2025 10:15:04

Module ID:

BOM ID: 0

Manufacturing Year: 2000

Manufacturing Week: 00

Battery Voltage: 3.314 U

Trim Wi-Fi hp: 8.8-8.8-7.6.6-7.7-7.8.8.8

Trim Wi-Fi lp: 8.8-8.8-9.9-9.9-10.10.10.11.11.11

Trim BT: 3.5-5.7.7

Trim XTLS: 39

MAC Address: 40:82:7b:00:0c:2d

Anti-rollback Bootloader: 0

Anti-rollback App: 0

Wi-Fi init is done

Net init is done

BT init is done

BD Address: 40:82:7b:00:0c:2e

Configure BLE

BLE configuration is done

BLE Commissioning Service Creation

- BLE service created
- BLE WIFI Control charac created
- BLE WIFI Configure charac created
- BLE WIFI Network config charac created
- BLE services and charac registered
- BLE service and charac creation is done

Start BLE advertising

BLE advertising is started

BLE connected

BLE CONNECTED: Conn_Handle: 0

BLE write

BLE notification enabled

-> BLE NOTIFICATION ENABLED (Service: 0. Charac: 21).

BLE write

-> Conn_Handle: 0, Service: 0, Charac: 0, length 1, Data: 0x01

WIFI Control Charac

WIFI Scan Profile

WIFI scan done

MAC : (56:2d:a2:23:b6:e1) Channel: 6 | MPA-EAP | RSSI: -42 | SSID: Shreya iphone

MAC : (e4:ad:1c:ca:f9:b0) Channel: 6 | MPA-EAP | RSSI: -41 | SSID: STWLAN2

MAC : (e4:ad:1c:ca:f9:b2) Channel: 6 | MPA-EAP | RSSI: -40 | SSID: STMARMOBILE

MAC : (e4:ad:1c:ca:f9:b3) Channel: 6 | OPEN | RSSI: -49 | SSID: STGUEST

MAC : (e4:ad:1c:ca:f9:a2) Channel: 11 | OPEN | RSSI: -62 | SSID: STWLAN2

MAC : (e4:ad:1c:ca:f9:a0) Channel: 11 | MPA-EAP | RSSI: -63 | SSID: STWLAN2

MAC : (e4:ad:1c:ca:f5:80) Channel: 11 | MPA-EAP | RSSI: -61 | SSID: STMARMOBILE

MAC : (e4:ad:1c:ca:f5:81) Channel: 1 | MPA-EAP | RSSI: -66 | SSID: STWLAN2

MAC : (e4:ad:1c:ca:f5:82) Channel: 1 | OPEN | RSSI: -61 | SSID: STGUEST

MAC : (12:da:43:8b:ed:b4) Channel: 8 | MPA-EAP | RSSI: -70 | SSID: Netgear 2.4G

MAC : (e4:a4:1c:ec:a0) Channel: 6 | MPA-EAP | RSSI: -72 | SSID: STWLAN2



Tera Term logs after flashing the BLE Commissioning application



Porting to a new host processor using STM32CubeMX

Porting over new host processor(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

5. Configure DMA Channels for SPI

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

Porting over new host processor(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 processor(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

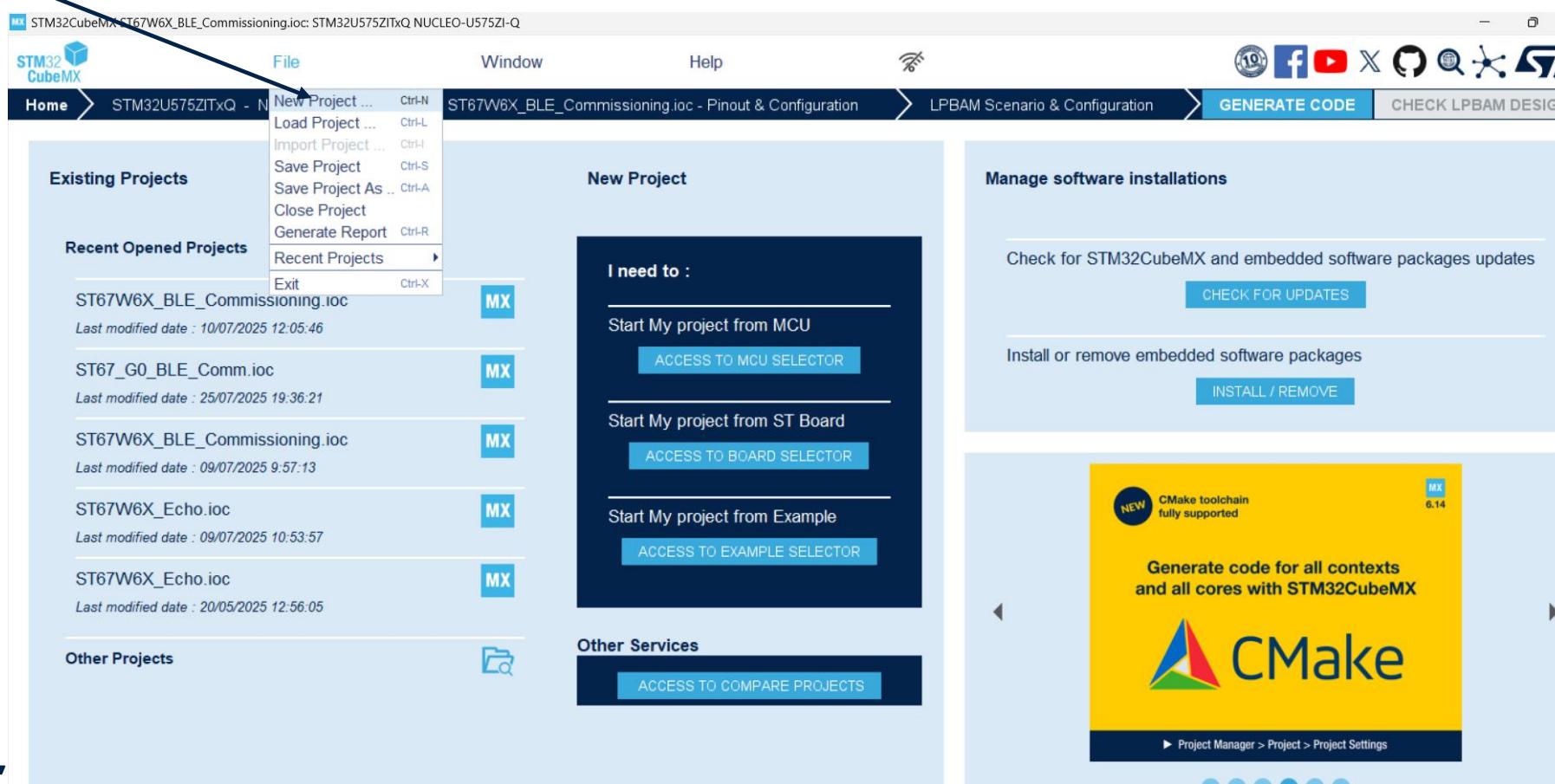
14. Build and Flash the application



Generating BLE Commissioning
app using NUCLEO-G0B1RE as
host processor (Walk-through)

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

Open STM32CubeMX and create the new project.



Select the board selector

The screenshot shows the STBoardSelector interface. At the top, there are tabs for 'New Project', 'MCU/MPU Selector', 'Board Selector' (which is selected), 'Example Selector', and 'Cross Selector'. Below the tabs, there are 'Board Filters' with a search bar containing 'STM32G0'. A sidebar on the left lists various STM32 series, with 'STM32G0' checked. The main area displays the 'STM32G0 Series' section, featuring the 'NUCLEO-G0B1RE' board. This board is described as an STM32 Nucleo-64 development board with STM32G0B1RE MCU, supporting Arduino and ST morpho connectivity. It includes two images of the board, its part number (NUCLEO-G0B1RE), commercial part number (INUCLEO-G0B1RE), unit price (\$10.32), and mounted device (STM32G0B1RET6). Below this, a 'Boards List' table shows eight items, with the 'NUCLEO-G0B1RE' row highlighted in blue.

Thumbnail	Commercial Part No.	Type	Marketing Status	Unit Price (\$)	Mounted Device
	NUCLEO-G0B1RE	Nucleo-64	Active	10.32	STM32G0B1RET6
	NUCLEO-G070RB	Nucleo-64	Active	10.32	STM32G070RB76
	NUCLEO-G071RB	Nucleo-64	Active	10.32	STM32G071RB76
	NUCLEO-G031KB	Nucleo-32	Active	10.32	STM32G031KB76
	NUCLEO-G070BT	Nucleo-64	Active	10.32	STM32G070BT76
	NUCLEO-G071BT	Nucleo-64	Active	10.32	STM32G071BT76
	NUCLEO-G031KT	Nucleo-32	Active	10.32	STM32G031KT76

Type STM32G0

Start Project

Select STM32G0

Select NUCLEO-G0B1RE

MCU/MPU Selector | Board Selector | Example Selector | Cross Selector

Board Filters

- Commercial
-
-
-

PRODUCT INFO

- Type >
- Supplier >
- MCU / MPU Series >

Aa [ab]

- STM32C0
- STM32F0
- STM32F1
- STM32F2
- STM32F3
- STM32F4
- STM32F7
- STM32G0
- STM32G4
- STM32H5
- STM32H7
- STM32L0
- STM32L1
- STM32L4
- STM32L4+
- STM32F5

Features

Large Picture

Docs & Resources

Datasheet

Buy

Start Project

STM32G0 Series

NUCLEO-G0B1RE

STM32 Nucleo-64 development board with STM32G0B1RE MCU, supports Arduino and ST morpho connectivity

ACTIVE
Product is in mass production

Part Number : NUCLEO-G0B1RE
Commercial Part Number : NUCLEO-G0B1RE
Unit Price (US\$) : 10.32
Mounted Device : STM32G0B1RET6

The STM32 Nucleo-64 board provides an affordable and flexible way for users to try out new concepts and build prototypes by choosing from the various combinations of performance and power consumption features provided by the STM32 microcontroller. For the compatible boards, the internal or external SMPS significantly reduces power consumption in Run mode.

DUINO® Uno V3 connectivity support and the ST morpho headers allow the easy expansion of the functionality of the STM32 Nucleo open development platform with a wide choice of specialized shields.

STM32 Nucleo-64 board does not require any separate probe as it integrates the

Board Project Options: NUCLEO-G0B1RE

Initialize all peripherals with their default Mode ?

Yes No

Bands List: 8 items

Export

	Overview	Commercial Part No	Type	Marketing Status	Unit Price (US\$)	Mounted Device
★		NUCLEO-G070RB	Nucleo-64	Active	10.32	STM32G070RBT6
★		NUCLEO-G071RB	Nucleo-64	Active	10.32	STM32G071RBT6
★		NUCLEO-G0B1RE	Nucleo-64	Active	10.32	STM32G0B1RET6
		STM32G0316-DISCO	Discovery Kit	Active	9.89	STM32G0316RM6
		NUCLEO-F030R8	Nucleo-F030R8	Active	10.32	STM32F030R8T6
		NUCLEO-F030K6	Nucleo-F030K6	Active	10.32	STM32F030K6T6
		NUCLEO-F030R8	Nucleo-F030R8	Active	10.32	STM32F030R8T6
		NUCLEO-F030K6	Nucleo-F030K6	Active	10.32	STM32F030K6T6

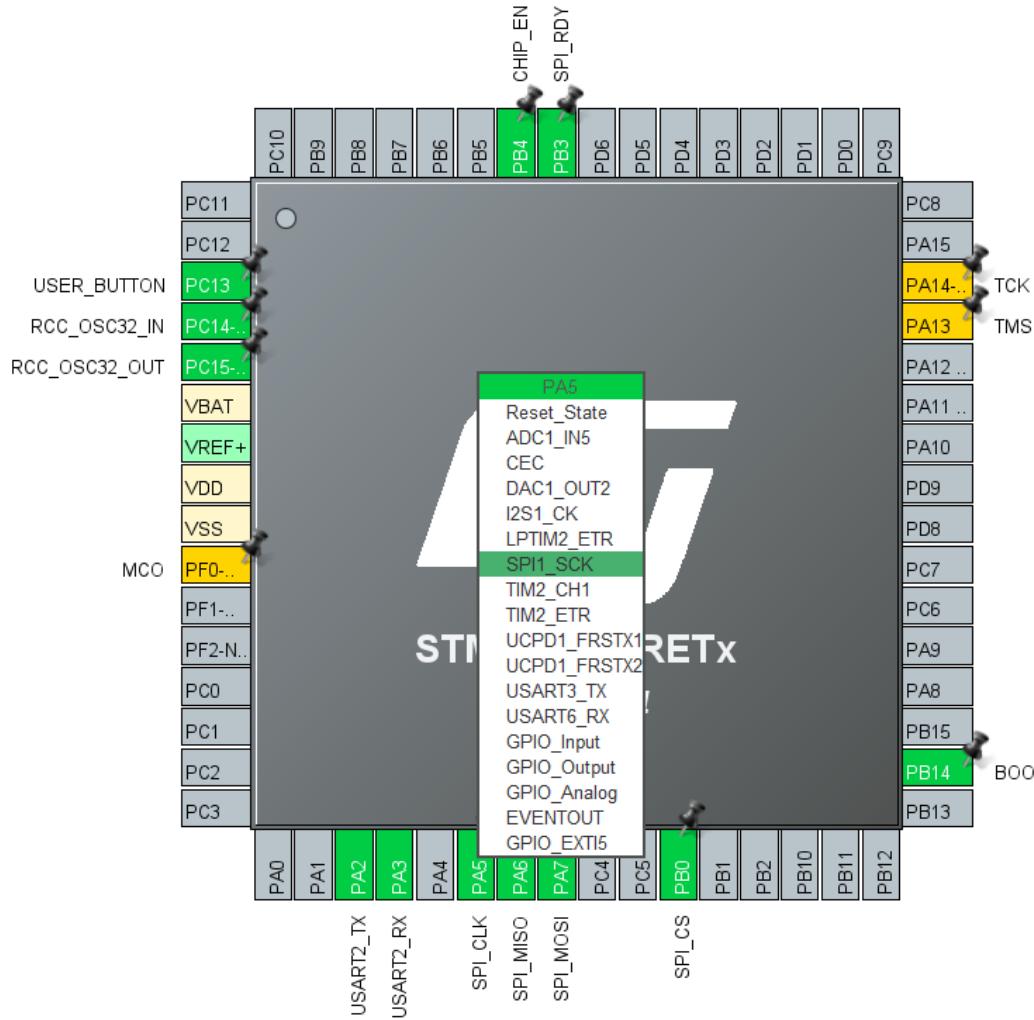
Select Yes

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.

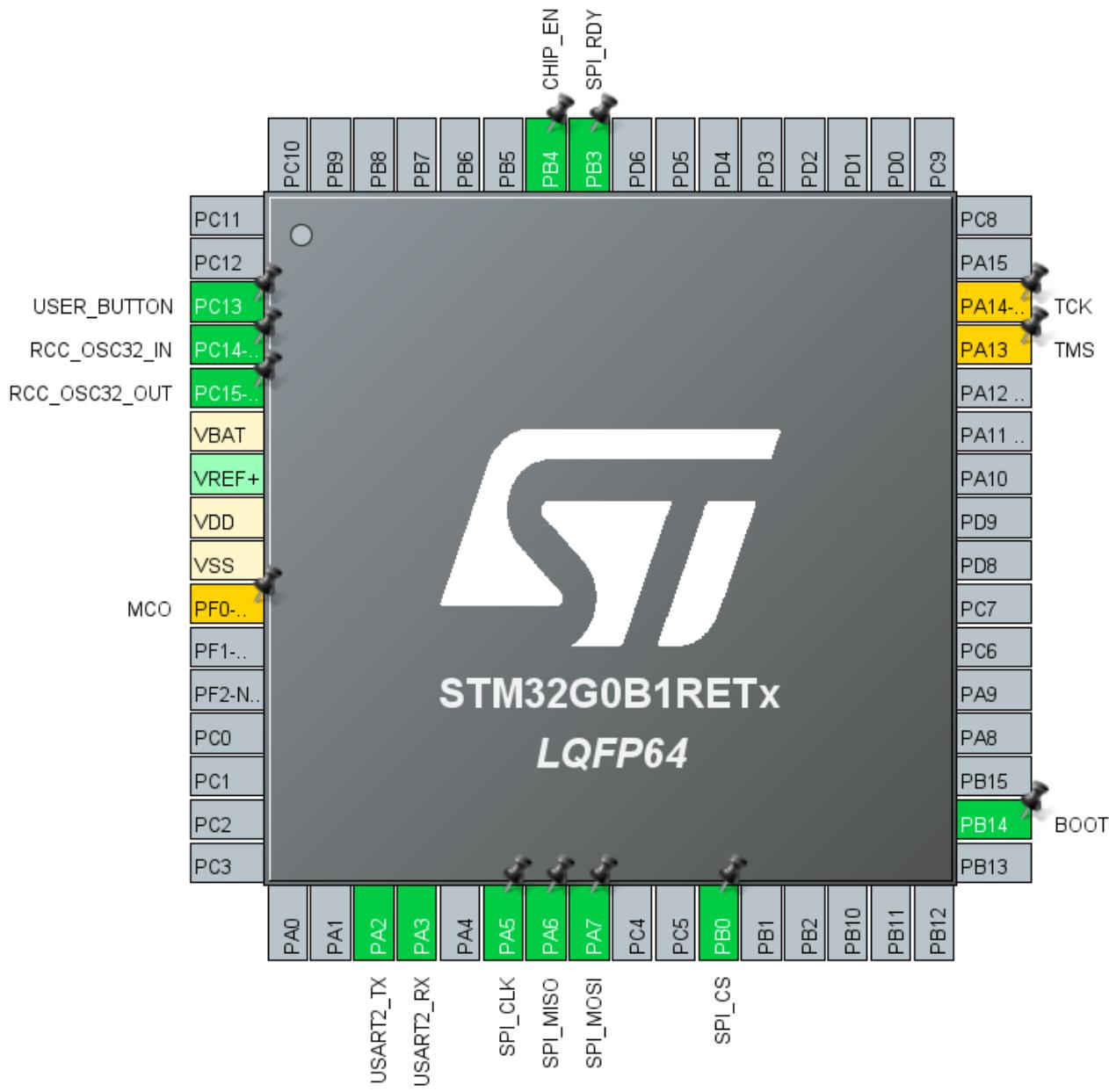
Pin function	Arduino Connector	STM32G0 Pin	GPIO mode	GPIO pull mode	GPIO speed
SPI_CLK	CN5.D13	PA5	AF PP	No Pull	Very high
SPI_MISO	CN5.D12	PA6	AF PP	No Pull	Very high
SPI_MOSI	CN5.D11	PA7	AF PP	No Pull	Very high
SPI_CS	CN5.D10	PB0	Output PP	No Pull	high
USER_BUTTON	-	PC13	EXTI Falling	Pull-up	N/A
CHIP_EN	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)

Parameter
Settings

The screenshot shows the STM32CubeMX software's Parameter Settings window. On the left, a tree view lists various peripherals: Analog, Timers, Connectivity, FDCAN1, FDCAN2, I2C1, I2C2, I2C3, IRTIM, LPUART1, LPUART2, SPI1, SPI2, SPI3, UCPD1, UCPD2, USART1, USART2, USART3, USART4, USART5, USART6, USB_DRD_FS, Multimedia, and Computing. A blue arrow points from the 'Parameter Settings' text to the USART2 node in the tree. USART2 is highlighted with a blue selection bar. Another blue arrow points from the USART2 node to the configuration panel on the right.

Mode: Asynchronous
Hardware Flow Control (RS232): Disable
Hardware Flow Control (RS485):
Slave Select(NSS) Management: Disable

Configuration

Reset Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

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

- Data Direction: Receive and Transmit
- Over Sampling: 16 Samples
- Single Sample: Disable
- ClockPrescaler: 1
- Fifo Mode: Disable
- Tx fifo Threshold: 1 eighth full configuration
- Rx fifo Threshold: 1 eighth full configuration

Advanced Features

- Auto Baudrate: Disable
- TX Pin Active Level Inversion: Disable
- RX Pin Active Level Inversion: Disable
- Data Inversion: Disable
- TX and RX Pins Swapping: Disable
- Overrun: Enable
- DMA on RX Error: Enable
- MSB First: Disable

PA2: USART2_TX
PA3: USART2_RX

Select
USART2

GPIO
Settings

The screenshot shows the STM32CubeMX software's GPIO Settings window. At the top, there are tabs: Reset Configuration, Parameter Settings, User Constants, NVIC Settings, DMA Settings, and GPIO Settings. The GPIO Settings tab is highlighted with a blue selection bar. A blue arrow points from the 'GPIO Settings' text in the 'Parameter Settings' window to this tab. Below the tabs, there is a search bar labeled 'Search Signals' and a checkbox 'Show only Modified Pins'. The main area displays a table of pin configurations:

Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum out...	Fast Mode	User Label	Modified
PA2	USART2_TX	Low	Alternate Function	No pull-up and no pull-down	Low	n/a		<input type="checkbox"/>
PA3	USART2_RX	Low	Alternate Function	No pull-up and no pull-down	Low	n/a		<input type="checkbox"/>

SPI GPIO(SPI1)

PA5: SPI_SCK
PA6: SPI_MISO
PA7: SPI_MOSI

Select SPI1

The screenshot shows the Pinout & Configuration interface for the STM32G0B1RETx NUCLEO-G0B1RE board. The left sidebar lists various peripherals under System Core, Analog, Timers, and Connectivity. The Connectivity section includes FDCAN1, FDCAN2, I2C1, I2C2, I2C3, IRTIM, LPUART1, LPUART2, SPI1 (which is selected and highlighted in blue), and SPI2.

The main area displays the Clock Configuration and Project Manager tabs. Under Clock Configuration, the SPI1 Mode and Configuration section is shown with Mode set to Full-Duplex Master and Hardware NSS Signal set to Disable. The Configuration section includes tabs for Parameter Settings, User Constants, NVIC Settings, DMA Settings, and GPIO Settings. The Parameter Settings tab is active, showing configuration for SPI1_SCK, SPI1_MISO, and SPI1_MOSI pins.

A callout arrow points from the text "Select SPI1" to the SPI1 entry in the Connectivity sidebar. Another callout arrow points from the text "Parameter settings" to the Parameter Settings tab in the SPI1 Mode and Configuration section.

Parameter Settings Tab Content:

Parameter	Value
Frame Format	Motorola
Data Size	8 Bits
First Bit	MSB First
Prescaler (for Baud Rate)	2
Baud Rate	32.0 MBits/s
Clock Polarity (CPOL)	Low
Clock Phase (CPHA)	1 Edge
CRC Calculation	Disabled
NSSP Mode	Enabled
NSS Signal Type	Software

DMA Channels

Select DMA

The screenshot shows the Pinout & Configuration interface for a DMA request. On the left, under 'System Core' categories, 'DMA' is selected. The main area displays two rows of configuration for DMA requests:

DMA Request	Channel	Direction	Priority
SPI1_TX	DMA1 Channel 1	Memory To Peripheral	High
SPI1_RX	DMA1 Channel 2	Peripheral To Memory	Very High

Below the table, there are sections for 'DMA Request Settings' and 'DMA Request Synchronization Settings'. In 'DMA Request Settings', the mode is set to 'Normal', increment address is checked for memory, and data width is set to 'Byte'. In 'DMA Request Synchronization Settings', enable synchronization is checked, and other fields like synchronization signal and polarity are present.

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

GPIO

Select GPIO

The screenshot shows the STM32CubeMX software interface for selecting and configuring GPIO pins.

Left Panel (Categories):

- Search bar:
- Categories: Categories A-Z
- System Core: DMA, GPIO (selected), IWDG, NVIC, RCC, SYS, WWDG
- Analog
- Timers
- Connectivity
- Multimedia
- Computing
- Middleware and Software Packs
- Utilities

Right Panel (GPIO Mode and Configuration):

Mode: GPIO Mode and Configuration

Configuration: Group By Peripherals (checked), GPIO (checked), Single Mapped Signals (checked), RCC (checked), SPI (checked), USART (checked), NVIC (checked)

Search Signals: Search (Ctrl+F)

Show only Modified Pins:

Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up...	Maximum out...	Fast Mode	User Label	Modified
PB0	n/a	Low	Output Push Pull	No pull-up a...	High	n/a	SPI_CS	<input checked="" type="checkbox"/>
PB3	n/a	n/a	External Interrupt Mode with Ri...	No pull-up a...	n/a	n/a	SPI_RDY	<input checked="" type="checkbox"/>
PB4	n/a	Low	Output Push Pull	No pull-up a...	High	n/a	CHIP_EN	<input checked="" type="checkbox"/>
PB14	n/a	Low	Output Push Pull	No pull-up a...	Low	n/a	BOOT	<input checked="" type="checkbox"/>
PC13	n/a	n/a	External Interrupt Mode with F...	Pull-up	n/a	n/a	USER_BUTTON	<input checked="" type="checkbox"/>

Annotations:

- GPIO pin names:** Points to the "Pin Name" column in the table.
- GPIO mode:** Points to the "GPIO mode" column in the table.
- GPIO output level:** Points to the "GPIO output level" column in the table.
- GPIO Maximum Output speed:** Points to the "Maximum out..." column in the table.
- GPIO User label:** Points to the "User Label" column in the table.

Timer

SYS Mode and Configuration

Mode

- Serial Wire
- System Wake-Up 1
- System Wake-Up 2
- System Wake-Up 4
- System Wake-Up 5
- System Wake-Up 6

Power Voltage Detector In

VREFBUF Mode

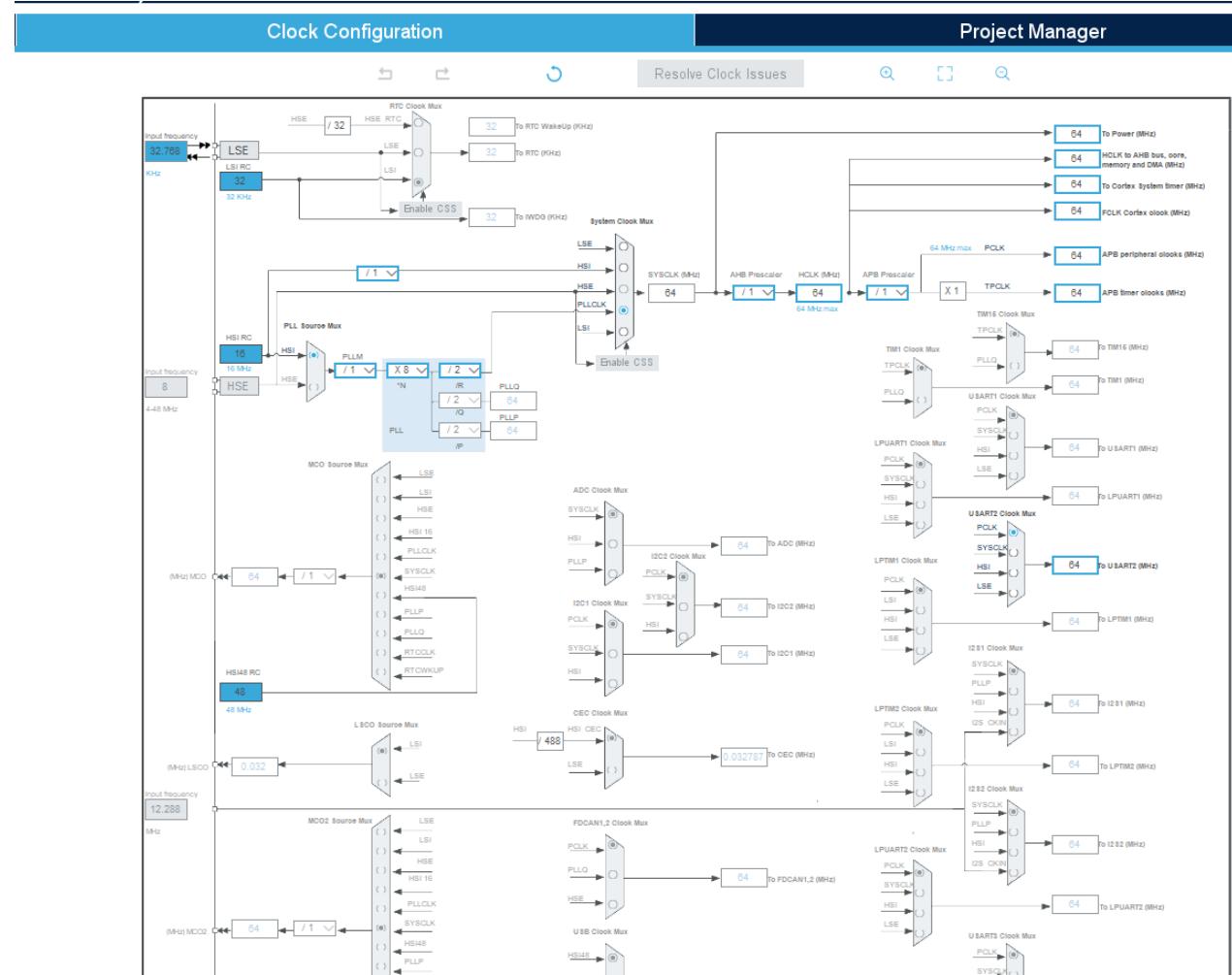
Timebase Source

save power of non-active UCPD - deactivate Dead Battery pull-up

Sys mode configuration: TIM1 in place of systick

Clock Configuration

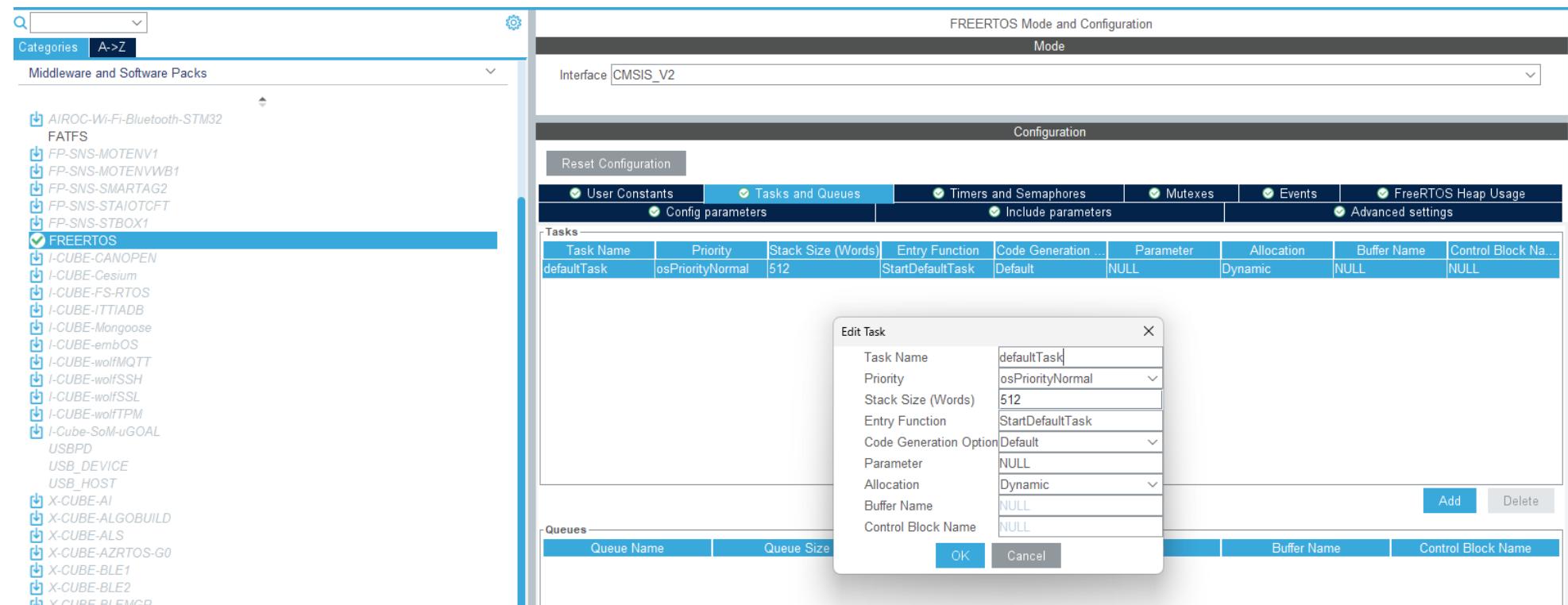
- In the "Clock Configuration" panel, system clock is set to the maximum frequency of 64 MHz, which increases the SPI clock speed to 32 MHz.



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

The screenshot shows the 'FreeRTOS Mode and Configuration' section of the STMicroelectronics software. The left sidebar lists various middleware and software packs, with 'FREERTOS' selected. The main area displays configuration parameters for FreeRTOS version (10.3.1), CMSIS-RTOS version (2.00), and various kernel settings like USE_PREEMPTION, CPU_CLOCK_HZ, and TOTAL_HEAP_SIZE. A red arrow points from the text 'heap_4' in the list to the 'TOTAL_HEAP_SIZE' field, which is currently set to '40000 Bytes'.

Parameter	Value
FreeRTOS version	10.3.1
CMSIS-RTOS version	2.00
MPU/FPU	ENABLE_MPU: Disabled ENABLE_FPU: Disabled
Kernel settings	USE_PREEMPTION: Enabled CPU_CLOCK_HZ: SystemCoreClock TICK_RATE_HZ: 1000 MAX_PRIORITIES: 56 MINIMAL_STACK_SIZE: 128 Words MAX_TASK_NAME_LEN: 16 USE_16_BIT_TICKS: Disabled IDLE_SHOULD_YIELD: Enabled USE_MUTEXES: Enabled USE_RECURSIVE_MUTEXES: Enabled USE_COUNTING_SEMAPHORES: Enabled QUEUE_REGISTRY_SIZE: 8 USE_APPLICATION_TASK_TAG: Disabled ENABLE_BACKWARD_COMPATIBILITY: Enabled USE_PORT_OPTIMISED_TASK_SELECTION: Disabled USE_TICKLESS_IDLE: Disabled USE_TASK_NOTIFICATIONS: Enabled RECORD_STACK_HIGH_ADDRESS: Disabled
Memory management settings	Memory Allocation: Dynamic / Static TOTAL_HEAP_SIZE: 40000 Bytes Memory Management scheme: heap_4
Hook function related definitions	USE_IDLE_HOOK: Disabled USE_TICK_HOOK: Disabled

NVIC

Search: Categories: A-Z

System Core

- DMA
- GPIO
- IWDG
- NVIC**
- RCC
- SYS
- WWDG

Analog >

Timers >

Connectivity >

Multimedia >

Computing >

Middleware and Software Packs >

Utilities >

NVIC Mode and Configuration

Mode

Configuration

NVIC Code generation

Sort by Preemption Priority and Sub Priority Sort by interrupts names

Search: Search (Ctrl+F) Show available interrupts Force DMA channels Interrupts

NVIC Interrupt Table	Enabled	Preemption Priority	Uses FreeRTOS functions
Non maskable interrupt	<input checked="" type="checkbox"/>	0	<input type="checkbox"/>
Hard fault interrupt	<input checked="" type="checkbox"/>	0	<input type="checkbox"/>
System service call via SWI instruction	<input checked="" type="checkbox"/>	0	<input type="checkbox"/>
Pendable request for system service	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
System tick timer	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
PVD through EXTI line 16, PVM (monit. VDDIO2) thro...	<input type="checkbox"/>	3	<input checked="" type="checkbox"/>
Flash global interrupt	<input type="checkbox"/>	3	<input checked="" type="checkbox"/>
RCC global Interrupt	<input type="checkbox"/>	3	<input checked="" type="checkbox"/>
EXTI line 2 and line 3 interrupts	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
EXTI line 4 to 15 interrupts	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
DMA1 channel 1 interrupt	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
DMA1 channel 2 and channel 3 interrupts	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
Time base: TIM1 break, update, trigger and commuta...	<input checked="" type="checkbox"/>	3	<input type="checkbox"/>
SPI1/I2S1 Interrupt	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>
USART2 + LPUART2 Interrupt	<input checked="" type="checkbox"/>	3	<input checked="" type="checkbox"/>

Middleware & Software Packs

X-CUBE-ST67W61 Software package

The screenshot shows the STMicroelectronics X-CUBE-ST67W61 software package interface. The main window has three tabs: Pinout & Configuration, Clock Configuration, and Project Manager. The Pinout & Configuration tab is active, displaying a search bar and a list of categories. The Project Manager tab shows the 'Software Packs' mode. A modal dialog titled 'Software Packs Component Selector' is open, listing various software components. In the 'Device Applications' section, the 'Application' component is selected, and the dropdown menu shows 'BLE_Commissioning_v' highlighted with a blue arrow and the text 'Select BLE_Commissioning_v'.

Pinout & Configuration

Clock Configuration

Project Manager

Software Packs

Mode

STMicroelectronics.X-CUBE-ST67W61.1.0.0 Mode and Configuration

Device Applications

Network ST67W6X Network Driver

Software Packs Component Selector

Packs

Pack / Bundle / Component	Status	Version	Selection
STMicroelectronics.X-CUBE-ST60	1.0.0		Install
STMicroelectronics.X-CUBE-ST67W61	1.0.0		
Device Applications	1.0.0		
Application	1.0.0	BLE_Commissioning_v	
Freertos_Tickless	1.0.0		
Network ST67W6X_Network_Driver	1.0.0		
ServiceShell	1.0.0		
ServiceAPI	1.0.0		<input checked="" type="checkbox"/>
Driver / W61_at_and_bus	1.0.0		<input checked="" type="checkbox"/>
Utilities	1.0.0		
Logging	1.0.0		<input checked="" type="checkbox"/>
Shell	1.0.0		<input type="checkbox"/>
Statistics	1.0.0		<input type="checkbox"/>
Debug TraceRecorder	4.10.2		
TraceRecorder	4.10.2		<input type="checkbox"/>
Data Exchange cJSON	1.7.18		
cJSON	1.7.18		<input type="checkbox"/>
File System LittleFS	2.10.1		<input type="checkbox"/>
LittleFS	2.10.1		<input type="checkbox"/>
Utility Utilities	1.4.2		

Ok Cancel

W61_WIFI_MAX_DETECT_AP
W61_BLE_MAX_CONN_NBR
W61_DLE_MAX_DETECT_AP_TIMEOUT

50
1
40

Utilities

Select BLE_Commissioning_v

Middleware & Software Packs

X-CUBE-ST67W61 Software package

Pinout & Configuration Clock Configuration Project Man...

Categories A>Z

- FOODLE-wiFiSSH
- I-CUBE-wolfSSL
- I-CUBE-wolfTPM
- I-Cube-SoM-uGOAL
- USBPD
- USB_DEVICE
- USB_HOST
- X-CUBE-AI**
- X-CUBE-ALGOBUILD
- X-CUBE-ALS
- X-CUBE-AZRTOS-G0
- X-CUBE-BLE1
- X-CUBE-BLE2
- X-CUBE-BLEMGR
- X-CUBE-DPower
- X-CUBE-EEPRMA1
- X-CUBE-GNSS1
- X-CUBE-IPS
- X-CUBE-ISPU
- X-CUBE-IoTC-DA16k-PMOD
- X-CUBE-MEMS1
- X-CUBE-NFC4
- X-CUBE-NFC6
- X-CUBE-NFC7
- X-CUBE-NFC9
- X-CUBE-NFC10
- X-CUBE-NFC12
- X-CUBE-PM33A1
- X-CUBE-RT-Thread_Nano
- X-CUBE-SAFEA1
- X-CUBE-SFXS2LP1
- X-CUBE-SMBUS
- X-CUBE-ST60
- X-CUBE-ST67W61**
- X-CURF-SIURG2

STMicroelectronics X-CUBE-ST67W61.1.0.0 Mode and Configuration
Mode

Device Applications
Network ST67W6X Network Driver

Configuration

Reset Configuration

W6X Modules Parameter Settings User Constants Platform Settings

Platform proposal

BSP

Name	IPs or Components	Found Solutions	BSP API
LogSh_UART	USART:Asynchronous	USART2	Unknown
NCP_SPI	SPI:Full-Duplex Master	SPI1	Unknown

Platform Settings

Middleware & Software Packs

X-CUBE-ST67W61 Software package

The screenshot shows the X-CUBE-ST67W61 software configuration interface. On the left, the 'Pinout & Configuration' tab is active, displaying a search bar, categories (A-Z), and a list of software packs. The 'X-CUBE-ST67W61' pack is selected. On the right, the 'Clock Configuration' tab is active, showing 'Software Packs' and 'Pinout' dropdowns, and a title 'STMicroelectronics X-CUBE-ST67W61.1.0.0 Mode and Configuration'. Under 'Mode', 'Device Applications' and 'Network ST67W6X Network Driver' are checked. The 'Configuration' tab is also visible. A callout points to the 'W6X_POWER_SAVE_AUTO' parameter in the 'W6X Parameters' section, which is set to 'No'.

Disable W6X_POWER_SAVE_AUTO

Parameter	Value
W6X_BLE_HOSTNAME	ST67W61_BLE
W6X_WIFI_HOSTNAME	ST67W61_WiFi
W6X_POWER_SAVE_AUTO	No
W6X_CLOCK_MODE	Internal RC oscillator
W6X_WIFI_AUTOCONNECT	No
W6X_WIFI_DHCP_MODE	DHCP STA+SAP
W6X_WIFI_COUNTRY_CODE	00
W6X_WIFI_ADAPTIVE_COUNTRY_CODE	No
W6X_NET_RECV_TIMEOUT (ms)	5000
W6X_NET_SEND_TIMEOUT (ms)	5000
W6X_NET_RECV_BUFFER_SIZE	9216
W6X_WIFI_DNS_MANUAL	No
W6X_WIFI_DNS_IP_1	{208, 67, 222, 222}
W6X_WIFI_DNS_IP_2	{8, 8, 8, 8}
W6X_WIFI_DNS_IP_3	{0, 0, 0, 0}
W61_WIFI_MAX_DETECTED_AP	50
W61_BLE_MAX_CONN_NBR	1
W61_BLE_MAX_DETECTED_PERIPHERAL	10
W61_BT_MAX_SERVICE_NBR	5

Project Creation

The screenshot shows the STM32CubeMX Project Manager interface for a project named "ST67_G0_BLE_Comm". The interface is divided into several sections: Project, Code Generator, and Advanced Settings. The "Project Manager" tab is highlighted with a purple border. A purple arrow points from the text "1. Enable ‘Generate Under Root’" to the "Generate Under Root" checkbox in the "Toolchain / IDE" section. Another purple arrow points from the text "2. Enable ‘Use Default Firmware Location’" to the "Use Default Firmware Location" checkbox in the "Firmware Relative Path" section.

Project Name: ST67_G0_BLE_Comm

Project Location: C:\ST67_Test

Application Structure: Advanced

Toolchain Folder Location: C:\ST67_Test\ST67_G0_BLE_Comm

Toolchain / IDE: STM32CubeIDE

Generate Under Root:

Minimum Heap Size: 0x200

Minimum Stack Size: 0x400

Cortex-M0+NS

Enable multi-threaded support:

Thread-safe Locking Strategy: Default – Mapping suitable strategy depending on RTOS selection.

Mcu Reference: STM32G0B1RETx

Firmware Package Name and Version: STM32Cube FW_G0 V1.6.2

Use latest available version:

Use Default Firmware Location:

Firmware Relative Path: C:/Users/kulkarni/STM32Cube/Repository/STM32Cube_FW_G0_V1.6.2

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

Project Creation

Home > STM32G0B1RETx - NUCLEO-G0B1RE > ST67_G0_BLE_Comm.ioc - Project Manager >

	Pinout & Configuration	Clock Configuration	Project Manager
Project	<p>STM32Cube MCU packages and embedded software packs</p> <p><input type="radio"/> Copy all used libraries into the project folder</p> <p><input checked="" type="radio"/> Copy only the necessary library files</p> <p><input type="radio"/> Add necessary library files as reference in the toolchain project configuration file</p>		<p>3. Enable “Copy only the necessary files”</p>
Code Generator	<p>Generated files</p> <p><input type="checkbox"/> Generate peripheral initialization as a pair of 'c/h' files per peripheral</p> <p><input type="checkbox"/> Backup previously generated files when re-generating</p> <p><input checked="" type="checkbox"/> Keep User Code when re-generating</p> <p><input checked="" type="checkbox"/> Delete previously generated files when not re-generated</p>		
Advanced Settings	<p>HAL Settings</p> <p><input type="checkbox"/> Set all free pins as analog (to optimize the power consumption)</p> <p><input type="checkbox"/> Enable Full Assert</p>		

Generate Code

5. Generate Code

The screenshot shows the STMicroelectronics Project Manager interface for a STM32G0B1RETx - NUCLEO-G0B1RE project. The 'Generated Function Calls' table lists various peripheral initializations:

Rank	Function Name	Peripheral Instance Name	Do Not Generate Function Call	Visibility (Static)
1	SystemClock_Config	RCC	<input type="checkbox"/>	<input type="checkbox"/>
2	MX_GPIO_Init	GPIO	<input type="checkbox"/>	<input checked="" type="checkbox"/>
3	MX_DMA_Init	DMA	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	MX_SPI1_Init	SPI1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
5	MX_USART2_UART_Init	USART2	<input type="checkbox"/>	<input checked="" type="checkbox"/>
6	MX_App_BLE_Commissioning_Init	STMMicroelectronics X-CUBE-ST67W61.1.0.0	<input checked="" type="checkbox"/>	<input type="checkbox"/>

At the top right, there is a 'GENERATE CODE' button highlighted with a purple box and an arrow.

4. Enable this option
(here: we are not generating
this function call)

Folder Structure

Name	Date modified	Type	Size
📁 .settings	7/24/2025 1:37 PM	File folder	
📁 App_BLE_Commissioning	7/24/2025 7:22 PM	File folder	
📁 Core	7/24/2025 1:35 PM	File folder	
📁 Debug	7/29/2025 6:31 PM	File folder	
📁 Drivers	7/24/2025 1:34 PM	File folder	
📁 EWARM	7/24/2025 6:37 PM	File folder	
📁 Middlewares	7/24/2025 1:34 PM	File folder	
📁 ST67W6X	7/24/2025 1:34 PM	File folder	
📁 Utilities	7/24/2025 11:28 PM	File folder	
IDE .cproject	7/25/2025 7:36 PM	CPROJECT File	34 KB
MX .mxproject	7/25/2025 7:36 PM	MXPROJECT File	25 KB
IDE .project	7/24/2025 1:35 PM	PROJECT File	2 KB
ST67_G0_BLE_Comm_Debug.launch	7/29/2025 9:18 PM	LAUNCH File	10 KB
MX ST67_G0_BLE_Comm.ioc	7/25/2025 7:36 PM	STM32CubeMX	16 KB
ST32G0B1RETX_FLASH.Id	7/25/2025 7:36 PM	LD File	6 KB
ST32G0B1RETX_RAM.Id	7/24/2025 1:35 PM	LD File	6 KB

Folder structure view after generating the code from CubeMX

- App code
- Core
- Drivers
- Middlewares
- ST67W6X
- Utilities

STM32CubeIDE

Load the project on the STM32CubeIDE



The screenshot shows the STM32CubeIDE interface. On the left is the Project Explorer, displaying a hierarchical tree of files and folders for a project named "ST67_G0_BLE_Comm". The main window on the right shows the code editor with the file "main.c" open. The code contains the following snippet:

```
411  /* USER CODE END 4 */
412
413  /* USER CODE BEGIN Header_StartDefaultTask */
414  /**
415   * @brief  Function implementing the default task
416   * @param  argument: Not used
417   * @retval None
418   */
419
420  /* USER CODE END Header_StartDefaultTask */
421  void StartDefaultTask(void *argument)
422  {
423      /* USER CODE BEGIN 5 */
424      main_app();
425      /* Infinite loop */
426      for(;;)
427      {
428          osDelay(1000);
429      }
430      /* USER CODE END 5 */
431  }
432
433
434  /**
435   * @brief  Period elapsed callback in non-isolated mode
436   * @note   This function is called when a TIM
437   *         HAL_TIM_IRQHandler() of a direct
438   *         a global variable "uwTick" used as application
439   *         @param  htim : TIM handle
440   *         @retval None
441   */
442  void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef htim)
```

A red arrow points from the text "Load the project on the STM32CubeIDE" to the Project Explorer icon in the IDE interface.

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

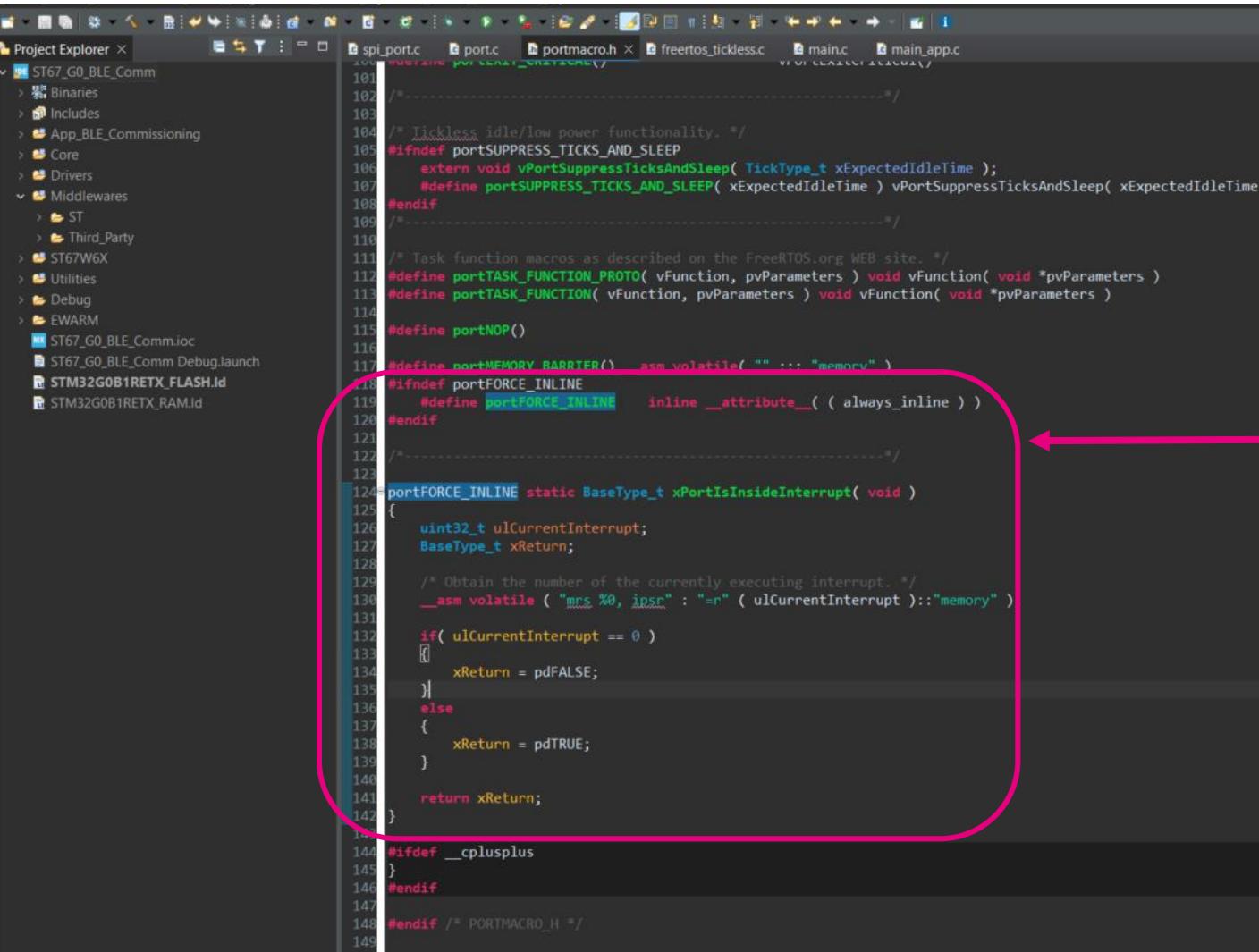
xPortIsInsideInterrupt()

- If the used device embeds an Arm® Cortex®M0+ (e.g. STM32G0) and its FreeRTOS version 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/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



The screenshot shows the STM32CubeIDE interface with the Project Explorer on the left and the code editor on the right. The code editor displays the portmacro.h file. A red box highlights the xPortIsInsideInterrupt function, which is circled in red. A pink arrow points from the text "Call xPortIsInsideInterrupt() function under portmacro.h file" to this highlighted section.

```
101 //*
102 */
103 /* Tickless idle/low power functionality. */
104 #ifndef portSUPPRESS_TICKS_AND_SLEEP
105     extern void vPortSuppressTicksAndSleep( TickType_t xExpectedIdleTime );
106     #define portSUPPRESS_TICKS_AND_SLEEP( xExpectedIdleTime ) vPortSuppressTicksAndSleep( xExpectedIdleTime )
107 #endif
108 */
109 /* Task function macros as described on the FreeRTOS.org WEB site. */
110 #define portTASK_FUNCTION_PROTO( vFunction, pvParameters ) void vFunction( void *pvParameters )
111 #define portTASK_FUNCTION( vFunction, pvParameters ) void vFunction( void *pvParameters )
112
113 #define portNOP()
114
115 #define portMEMORY_BARRIER() __asm volatile( "" :: "memory" )
116
117 #ifndef portFORCE_INLINE
118     #define portFORCE_INLINE inline __attribute__( ( always_inline ) )
119 #endif
120
121 */
122
123 portFORCE_INLINE static BaseType_t xPortIsInsideInterrupt( void )
124{
125    uint32_t ulCurrentInterrupt;
126    BaseType_t xReturn;
127
128    /* Obtain the number of the currently executing interrupt. */
129    __asm volatile ( "mrs %0, ipsr" : "=r" ( ulCurrentInterrupt )::"memory" )
130
131    if( ulCurrentInterrupt == 0 )
132    {
133        xReturn = pdFALSE;
134    }
135    else
136    {
137        xReturn = pdTRUE;
138    }
139
140    return xReturn;
141}
142
143 #ifdef __cplusplus
144#endif
145
146#endif /* PORTMACRO_H */
```

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`

The screenshot shows the STM32CubeIDE interface. On the left, the Project Explorer displays the project structure for 'ST67_G0_BLE_Comm'. It includes subfolders like Binaries, Includes, App_BLE_Commissioning, Core, Drivers, Middlewares, and ST67W6X. Within ST67W6X, there's a Target folder containing logging_config.h, shell_config.h, spi_port.c, w61_driver_config.h, w6x_config.h, spi_port.bak, w61_driver_config.h.bak, and w6x_config.h.bak. There are also Utilities, Ipm, Debug, and EWARM sections, along with launch files for ST67_G0_BLE_Comm.ioc, STM32G0B1RETX_FLASH.id, and STM32G0B1RETX_RAM.id.

The code editor on the right shows the file 'spi_port.c'. The code implements a custom memcpy function for the STM32G0B1 microcontroller. It includes conditional compilation blocks for different compilers: _ICCARM_ and _GNUC_. The code handles memory alignment and copying 4-byte blocks before copying remaining bytes. A pink arrow points to the first '#ifndef STM32G0B1xx' directive at line 79.

```
76 /* USER CODE END PFP */
77
78 /* Functions Definition -
79 #ifndef STM32G0B1xx ←
80 #ifdef __ICCARM__
81 void *spi_port_memcpy(void *dest, const void *src, unsigned int len)
82 #endif /* __ICCARM__ */
83 #ifdef __GNUC__
84 void *memcpy(void *dest, const void *src, unsigned int len)
85 #endif /* __ICCARM__ */
86 {
87     /* USER CODE BEGIN memcpy_1 */
88
89     /* USER CODE END memcpy_1 */
90     uint8_t *d = (uint8_t *)dest;
91     const uint8_t *s = (const uint8_t *)src;
92
93     /* Copy bytes until the destination address is aligned to 4 bytes */
94     while (((uint32_t) d % 4 != 0) && len > 0)
95     {
96         *d++ = *s++;
97         len--;
98     }
99
100    /* Copy 4-byte blocks */
101    uint32_t *d32 = (uint32_t *)d;
102    const uint32_t *s32 = (const uint32_t *)s;
103    while (len >= 4)
104    {
105        *d32++ = *s32++;
106        len -= 4;
107    }
108
109    /* Copy remaining bytes */
110    d = (uint8_t *)d32;
111    s = (const uint8_t *)s32;
112    while (len > 0)
113    {
114        *d++ = *s++;
115        len--;
116    }
117
118    return dest;
119    /* USER CODE BEGIN memcpy_End */
120
121    /* USER CODE END memcpy_End */
122}
123#endif ←
```

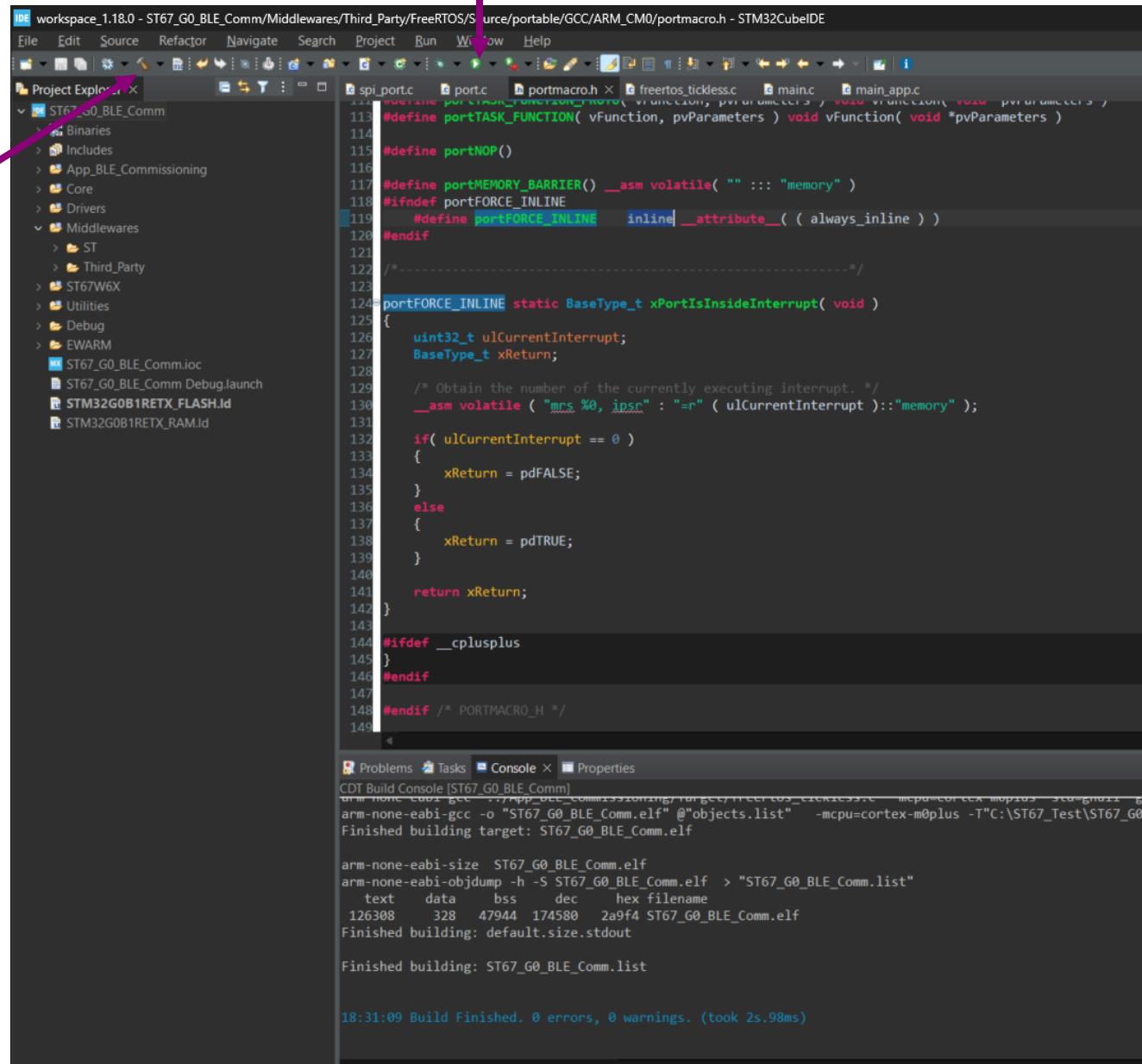
Here we are disabling the custom memcpy() definition, as we would be using the memcpy() definition in string.h

Project/ST67W6X/spi_port.c

Add `#endif`

Build and flash

1. Build the project



The screenshot shows the STM32CubeIDE interface. The left pane displays the Project Explorer with the 'ST67_G0_BLE_Comm' project selected. The right pane shows a code editor with the file 'portmacro.h' open, containing definitions for porting functions like portTASK_FUNCTION and portFORCE_INLINE. At the bottom, the Console tab shows the build log:

```
CDT Build Console [ST67_G0_BLE_Comm]
arm-none-eabi-gcc -o "ST67_G0_BLE_Comm.elf" @"objects.list" -mcpu=cortex-m0plus -T"C:\ST67_Test\ST67_G0_BLE_Comm\linker\linker.ld"
Finished building target: ST67_G0_BLE_Comm.elf

arm-none-eabi-size ST67_G0_BLE_Comm.elf
arm-none-eabi-objdump -h -S ST67_G0_BLE_Comm.elf > "ST67_G0_BLE_Comm.list"
    text      data      bss      dec   filename
126308       328     47944   174580  2a9f4 ST67_G0_BLE_Comm.elf
Finished building: default.size.stdout

Finished building: ST67_G0_BLE_Comm.list

18:31:09 Build Finished. 0 errors, 0 warnings. (took 2s.98ms)
```

Tera Term Logs

Tera Term Logs after flashing

The screenshot shows the Tera Term interface with two main windows. On the left is a terminal window titled 'COM21 - Tera Term VT' displaying a log of commissioning steps for an ST67W6X device. On the right is a configuration dialog titled 'Tera Term: Serial port setup and connection'. The configuration settings are as follows:

Setting	Value
Port	COM21
Speed	115200
Data	8 bit
Parity	none
Stop bits	1 bit
Flow control	none

Below the configuration window, a status bar displays device information:

- Device Friendly Name: STMicroelectronics STLink Virtual COM P
- Device Instance ID: USBVID_0483&PID_374B&MI_02&38AF34B
- Device Manufacturer: STMicroelectronics
- Provider Name: STMicroelectronics
- Driver Date: 4-1-2021
- Driver Version: 2.2.0.0

ST BLEToolbox/ ST BLE Webpage

To interface with the Wi-Fi commissioning embedded application, you can use
[STBLEToolbox Android/iOS application](#) / [ST Web Bluetooth® Interface](#)

Download [STBLEToolBox](#)

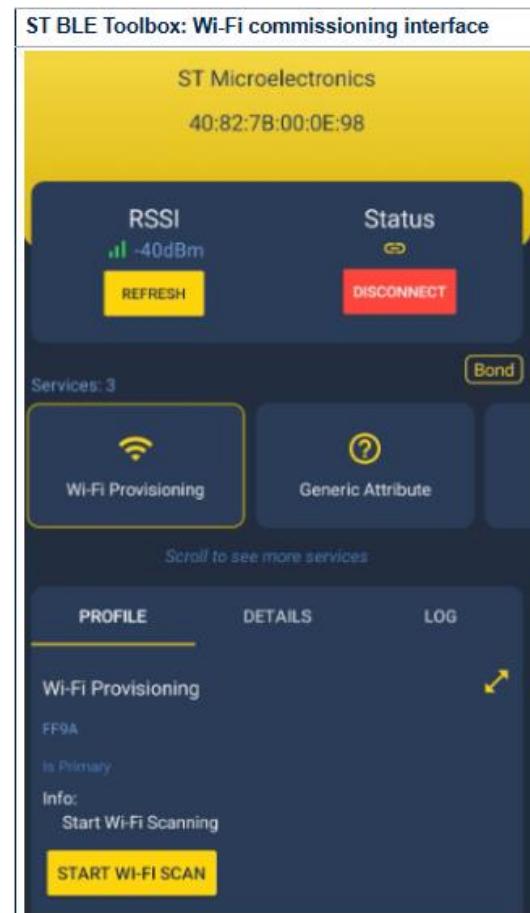
The versions supporting Wi-Fi commissioning service
are **v1.4.3 for Android and 1.2.8 for iOS**

Steps to launch and connect to Wi-Fi AP:

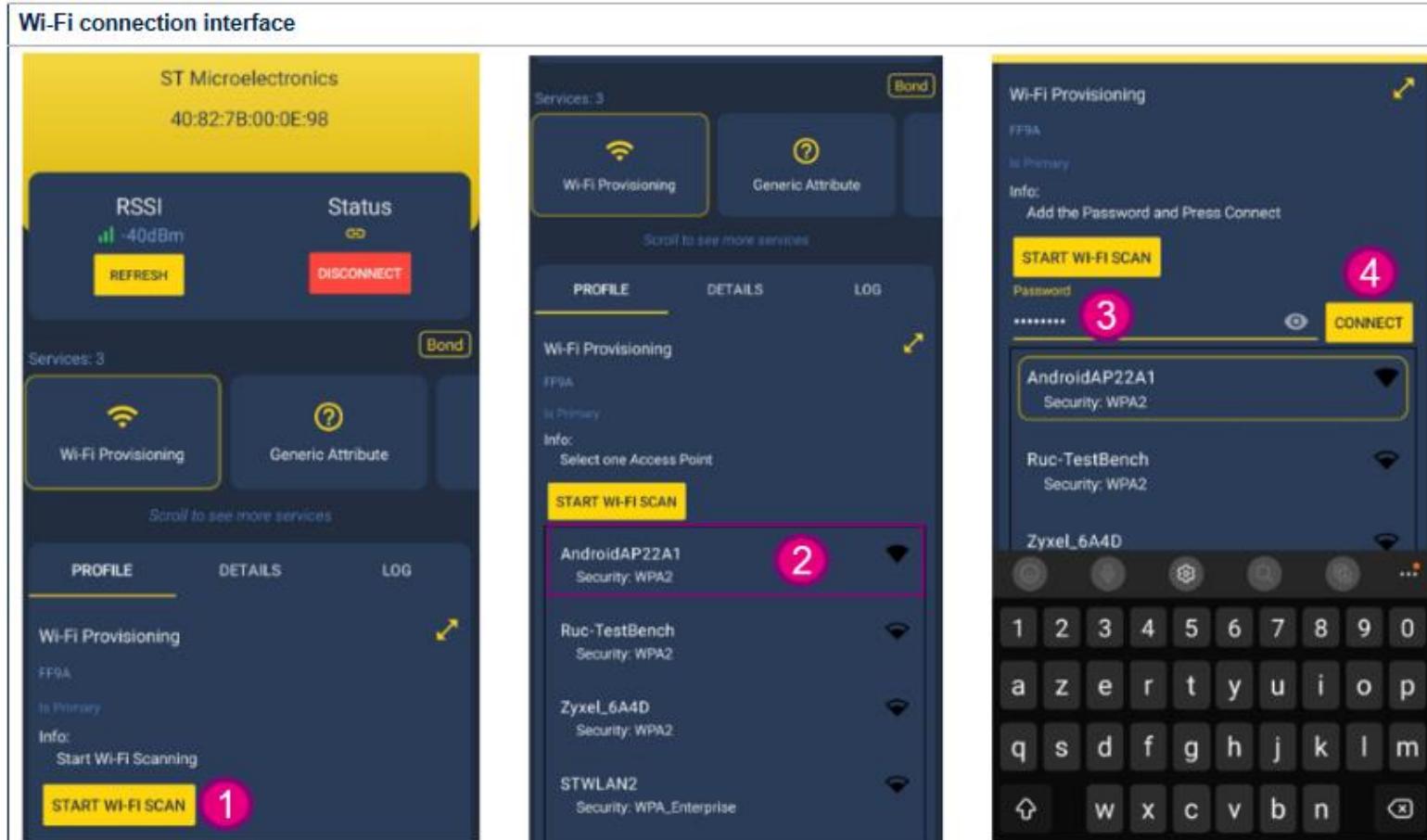
1. Launch the smartphone application and enable the scan button, if it is not already activated.
2. You can filter scanned devices by clicking on the SHOW FILTERS button and select ST BLE Wi-Fi to show only devices with Wi-Fi commissioning service.
3. Select your device, named ST_WiFi_XX where XX represents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



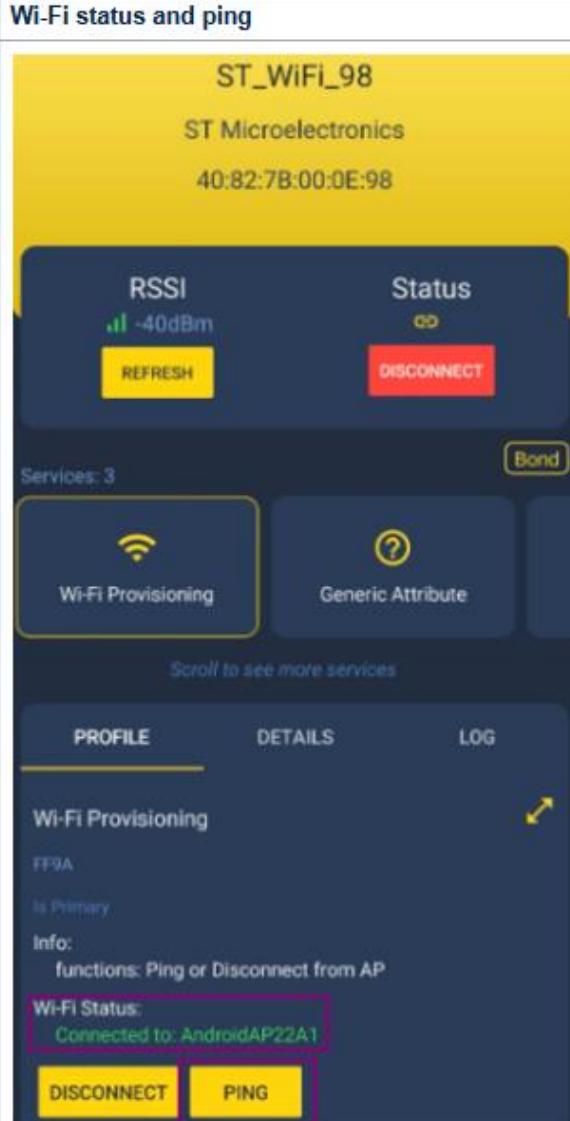
4. Once connected to the Bluetooth® LE device, click on Wi-Fi commissioning button to open the commissioning dedicated interface.



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.



ST_WIFI_98
ST Microelectronics
40:82:7B:00:0E:98

RSSI -40dBm **Status**  **DISCONNECT**

Services: 3

Wi-Fi Provisioning **Generic Attribute**

PROFILE DETAILS LOG

Wi-Fi Provisioning

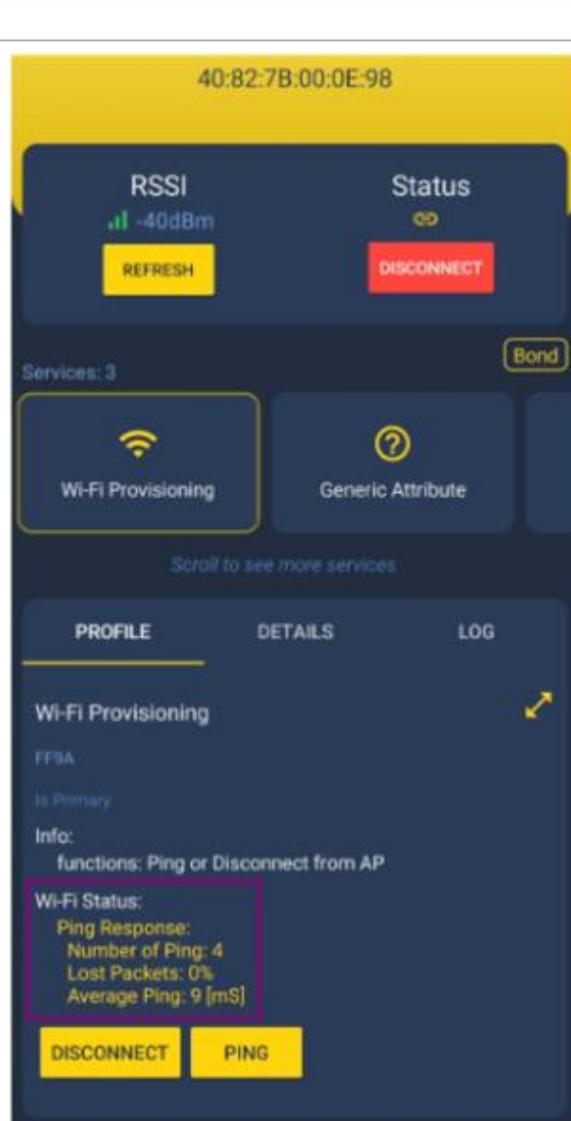
FF9A

Is Primary

Info:
functions: Ping or Disconnect from AP

Wi-Fi Status:
Connected to: AndroidAP22A1

DISCONNECT **PING**



40:82:7B:00:0E:98

RSSI -40dBm **Status**  **DISCONNECT**

Services: 3

Wi-Fi Provisioning **Generic Attribute**

PROFILE DETAILS LOG

Wi-Fi Provisioning

FF9A

Is Primary

Info:
functions: Ping or Disconnect from AP

Wi-Fi Status:
Ping Response:
Number of Ping: 4
Lost Packets: 0%
Average Ping: 9 [mS]

DISCONNECT **PING**

```

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:0:0:94] | Channel: 6 | WPA2 | RSSI: -40 | SSID: ATTAAA
MAC : [b4:63:f:aa:22:4] | Channel: 11 | WPA2 | RSSI: -42 | SSID: DesiWifi
MAC : [bc:9a:8e:42:7c:e4] | Channel: 1 | WPA2 | RSSI: -49 | SSID: canyouseewhereIP
MAC : [bc:9a:8e:a3:34:94] | Channel: 11 | WPA2 | RSSI: -59 | SSID: HUST_WIRELESS5G
MAC : [00:e6:3a:44:b5:82] | 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:50] | Channel: 11 | WPA2 | RSSI: -61 | SSID: Vista99_Staff
MAC : [00:e6:3a:44:8b:50] | Channel: 11 | WPA2 | RSSI: -62 | SSID: Vista99_loT
MAC : [00:e6:3a:44:c6:02] | Channel: 11 | WPA2 | RSSI: -62 | SSID: NETGEAR69
MAC : [00:e6:3a:44:c6:02] | Channel: 11 | WPA2 | RSSI: -62 | SSID: Vista99_WiFi
MAC : [00:e6:3a:44:66:90] | Channel: 11 | WPA2 | RSSI: -63 | SSID: Vista99_14T
MAC : [00:e6:3a:44:66:91] | Channel: 11 | WPA-EAP | RSSI: -63 | SSID: Passpoint
MAC : [00:e6:3a:44:66:94] | Channel: 11 | WPA2 | RSSI: -63 | SSID: Vista99_Staff
MAC : [4a:bd:c9:49:7b:26] | Channel: 11 | WPA2 | RSSI: -65 | SSID: vista3241
MAC : [00:e6:3a:58:29:80] | Channel: 1 | WPA2 | RSSI: -69 | SSID: Vista99_IoT
MAC : [00:e6:3a:58:29:81] | Channel: 1 | WPA-EAP | RSSI: -69 | SSID: Passpoint
MAC : [00:e6:3a:58:29:84] | 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:01] | 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:f:bf:0a:44] | Channel: 11 | WPA2 | RSSI: -72 | SSID: X_C455
MAC : [b4: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 : [14:59:0:0:4b: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:66:83:04] | Channel: 6 | WPA2 | RSSI: -74 | SSID: notyourwififi
MAC : [b9:ab:4d:02:8e:74] | Channel: 6 | WPA2 | RSSI: -75 | SSID: TuWo
MAC : [a8:7b:5d:93:a5:11] | Channel: 1 | WPA2 | RSSI: -76 | SSID: access5015
MAC : [d4:bc:d:d:ba:16:ff] | Channel: 1 | WPA2 | RSSI: -76 | SSID: home4025
MAC : [a8:9e:8:4:96:f1] | Channel: 1 | WPA2 | RSSI: -76 | SSID: Jahee_Home
MAC : [80:ce:62:b0:f9:66] | Channel: 6 | WPA2 | RSSI: -76 | SSID: DIRECT-E5-HP DeskJet
3700 series
MAC : [3c:b7:4:2:a:5:1b] | Channel: 11 | WPA2 | RSSI: -77 | SSID: XFWHG
MAC : [d8:1f:12:99:9e:ad] | Channel: 6 | OPEN | RSSI: -78 | SSID: SmartLife-9EAD

```



Walk through of the MQTT application

[ST67W611M Wi-Fi® – ST67W6X_MQTT Demonstration - stm32mcu](#)

ST67W6X MQTT Overview

- **Description**

The MQTT demonstration available in the X-CUBE-ST67W61 expansion package implements an MQTT client which connects to an MQTT Broker in order to publish environmental sensor data and device local information and subscribe on MQTT topics to receive command.

- In this example, the subscribed topic can be used to receive additional 'control' actions from a remote client publishing to the same topic.

1. Change the LED state on the host board

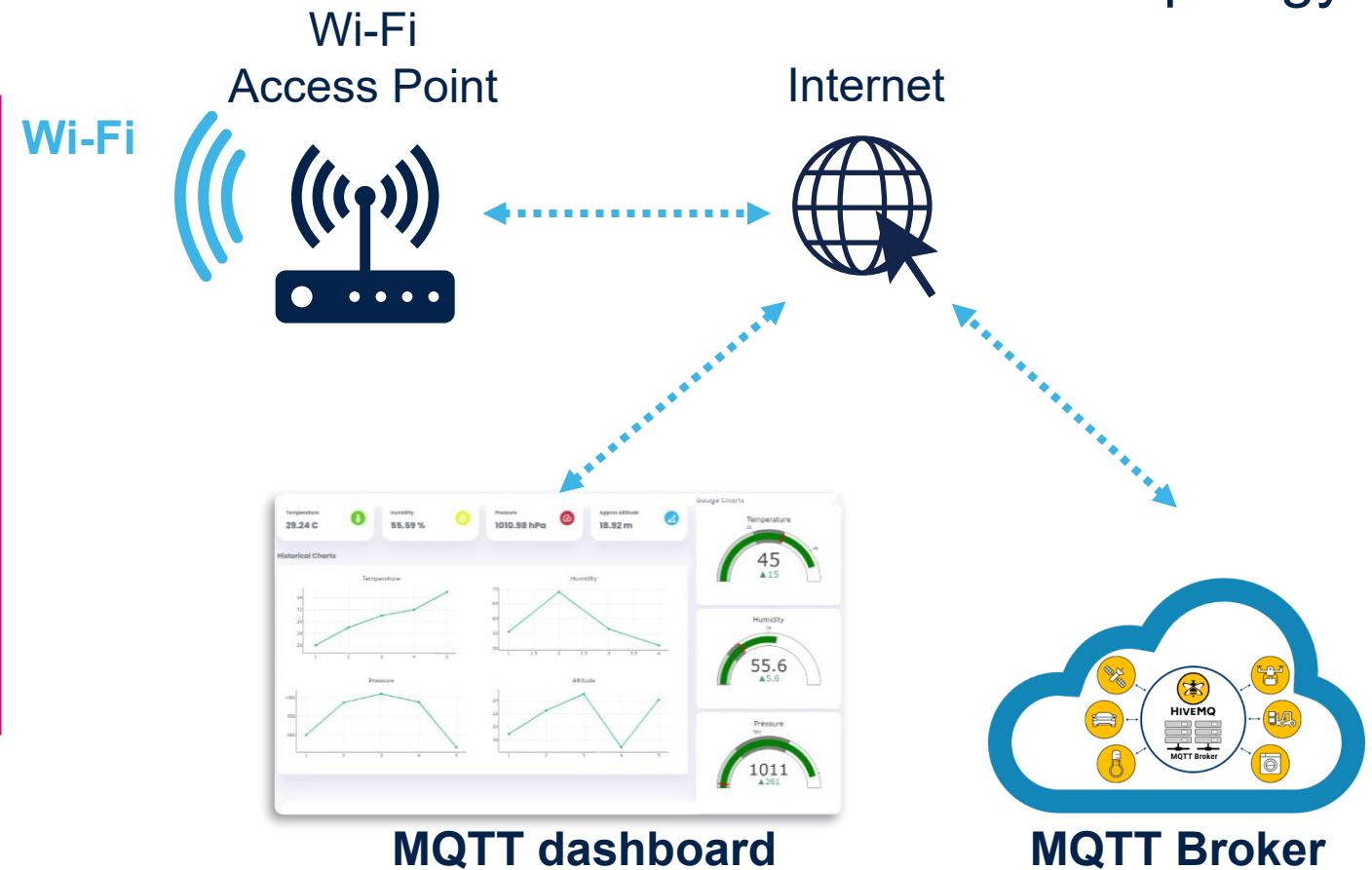
- To monitor the data sent and received, to/from the topics, a MQTT client can be installed on a PC, smartphone or tablet. Subscribing to the configured topics, allows the client to be notified every time the device publishes data to this topic.

- **Requirements**

- A X-NUCLEO-67W61M1 Wi-Fi Expansion board.
- A NUCLEO-U575ZI-Q Host board.
- A Wi-Fi Access Point with internet access and no firewall blocking the MQTT requests.
- [Optional] A X-NUCLEO-IKS4A1 Environmental sensors expansion board.

ST67W6X MQTT Topology

NUCLEO-U575ZI-Q
X-NUCLEO-67W61M1
X-NUCLEO-IKS4A1



ST67W6X MQTT

Usage

- Initialization of Wi-Fi, Net and MQTT modules in the Network Coprocessor.
- Wi-Fi connection establishment with the local Access Point.
- Synchronization of local time in using the Network Coprocessor SNTP client.
- Initialization of MEMS expansion board to get environmental sensor values.
- Connection of MQTT Client to a remote MQTT broker.
- MQTT topic subscription to receive messages from cloud.
- Periodic publication of message including:
 - Environmental sensor values**
 - Local time
 - RSSI of Access Point

MQTT Initialization process

Description	Function	File
Initialize the system (HAL, Clocks, SPI, I2C, UART, RTC, GPIO)	main()	Core/Src/main.c
Create the default FreeRTOS task containing the main application execution	MX_FREERTOS_Init()	Core/Src/app_feertos.c
Start all configurated tasks	osKernelStart()	cmsis_os2.c
Run the main_app() enter application function	StartDefaultTask()	Core/Src/app_freertos.c
Initialize the logging utility to forward all traces on UART peripheral	W6X_Util_Init()	ST67W6X_Network_Driver/Core/w6x_sys.c
Initialize the application callbacks to receive events from ST67W6X_Network_Driver	W6X_RegisterAppCb()	ST67W6X_Network_Driver/Core/w6x_sys.c
Initialize the ST67W6X_Network_Driver Power up the Network Coprocessor Display global identification	W6X_Init()	ST67W6X_Network_Driver/Core/w6x_sys.c
Initialize the ST67W6X_Network_Driver Wi-Fi module Set the STA mode Enable the DHCP Client Configure the Country Code Set the Hostname	W6X_WiFi_Init()	ST67W6X_Network_Driver/Core/w6x_wifi.c
Initialize the ST67W6X_Network_DriverNetwork module	W6X_Net_Init()	ST67W6X_Network_Driver/Core/w6x_net.c
Initialize the ST67W6X_Network_Driver MQTT module Register pre-allocated buffer to receive subscription messages	W6X_MQTT_Init()	ST67W6X_Network_Driver/Core/w6x_mqtt.c

MQTT Application Process

The different steps of the application **Process** are described below:

Description	Function	File
Connect the Wi-Fi station device to an available Access Point	W6X_WiFi_Connect()	ST67W6X_Network_Driver/Core/w6x_wifi.c
Enable the time synchronization through a remote SNTP server	W6X_Net_SetSNTPConfiguration()	ST67W6X_Network_Driver/Core/w6x_net.c
Retrieve the SNTP time to update the RTC time configuration	W6X_Net_GetTime()	ST67W6X_Network_Driver/Core/w6x_net.c
Initialize the sensor expansion board	Sys_Sensors_Init()	X-CUBE-MEMS1/App/sys_sensors.c
Configure the device to a MQTT broker	W6X_MQTT_Configure()	ST67W6X_Network_Driver/Core/w6x_mqtt.c
Connect the device to a MQTT broker	W6X_MQTT_Connect()	ST67W6X_Network_Driver/Core/w6x_mqtt.c
Start a topic subscription. When a new message is received, a callback catches and parses the frame.	W6X_MQTT_Subscribe()	ST67W6X_Network_Driver/Core/w6x_mqtt.c
Publish a new message on the selected topic. The RTC time is always set by calling HAL_RTC_GetTime() and HAL_RTC_GetDate(). If a sensor board is available, the environmental measures are added in the frame by calling Sys_Sensors_Read()	W6X_MQTT_Publish()	ST67W6X_Network_Driver/Core/w6x_mqtt.c

Two topics subscription are started:

- /devices/<MQClientID>/control
- /sensors/<MQClientID>

- Local Access Point credentials: *App_MQTT\app\app_config.h*

```
#define WIFI_SSID           "ST_DEMO"  
#define WIFI_PASSWORD        "stdemo01"
```

- MQTT parameters: *App_MQTT\app\app_config.h*

```
#define MQTT_HOST_NAME "broker.emqx.io"
```

```
#define MQTT_HOST_PORT 1883
```

```
#define MQTT_SECURITY_LEVEL 0
```

```
#define MQTT_CLIENT_ID "mySTM32_772"
```

```
#define MQTT_USERNAME ""
```

```
#define MQTT_USER_PASSWORD ""
```

Notes :

It uses a public broker

Requires at least one device publisher and one subscriber
TLS (secure broker connection)

ST67W6X MQTT

IoT MQTT panel app



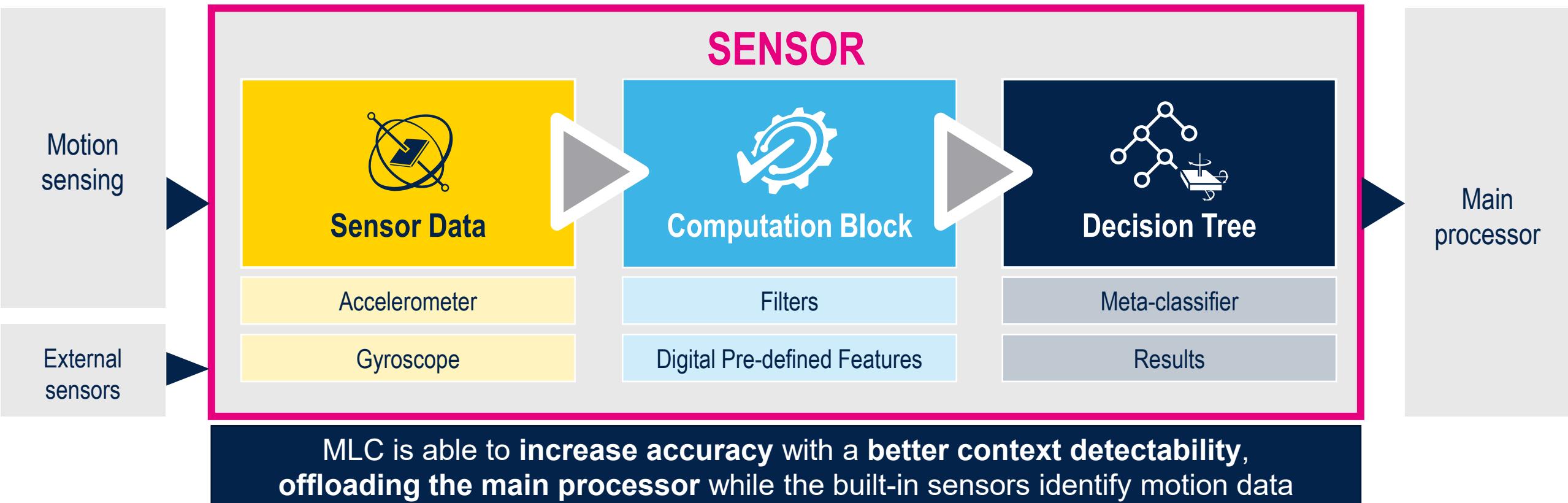
- Smartphone client : IoT MQTT Panel
 - Android play store: [IoT MQTT Panel - Apps on Google Play](#)
 - Android APK-pure: <https://apkpure.com/iot-mqtt-panel/snr.lab.iotmqtpanel.prod>
 - Apple store: [IoT MQTT Panel on the App Store](#)
- PC / Web client (no firewall)
 - HiveMQ: [MQTT Client by HiveMQ](#)
 - Mosquitto: [Eclipse Mosquitto](#)



MLC - Machine Learning Core

MLC - Machine Learning Core

MLC is an in-sensor classification engine based on a decision tree logic

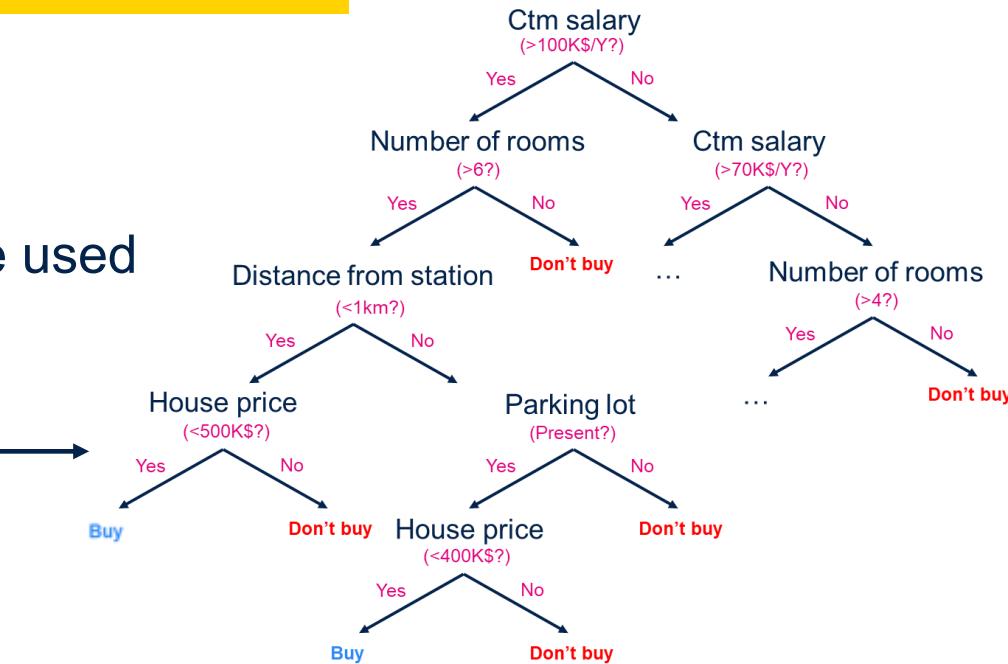


ML model training

What the ML model learn from the data?



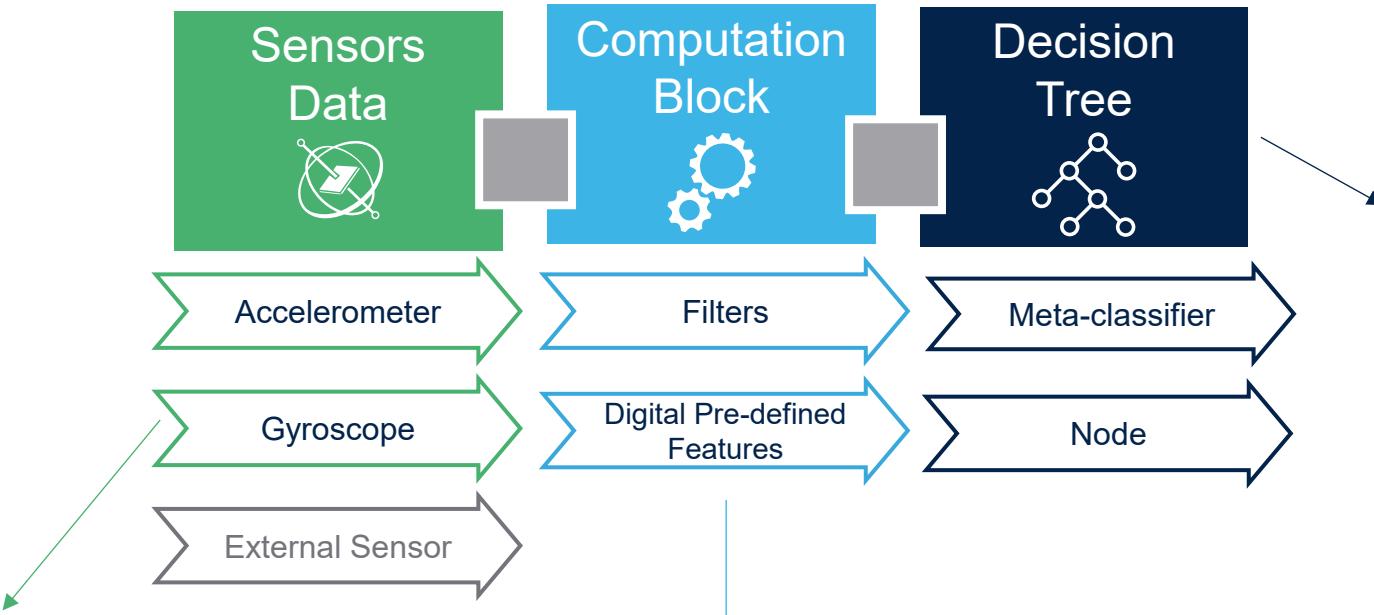
- The **features** that will be used
- In which **decision nodes** the **features** will be used
- The **threshold values** for **decision nodes**
- The results for the **leaf nodes**



Training the same Machine Learning Model with another dataset (i.e. houses in Milan), the resulting decision tree will be different in order to “best fit” the new data given for training



IMU Machine Learning Core - Overview



Sensors Data:

- **Accelerometer** → $[a_x \ a_y \ a_z], [a_v], [a^2_v]$
- **Gyroscope** → $[g_x \ g_y \ g_z], [g_v], [g^2_v]$
- **External sensor** → $[m_x \ m_y \ m_z], [m_v], [m^2_v]$ (i.e. mag)
- **Magnitude** → $V = \sqrt{X^2 + Y^2 + Z^2}$

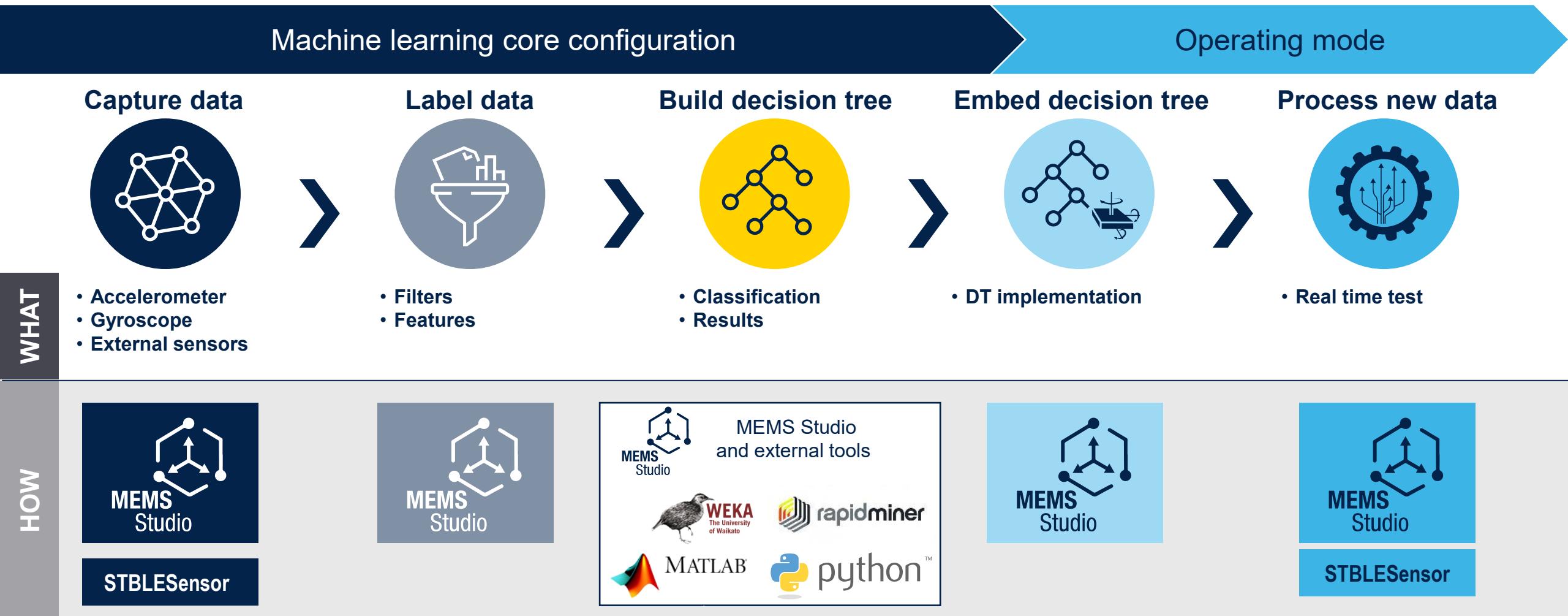
Computation Block

- Sensors data can be filtered with a 2nd order IIR **Filter**.
- **Features** are statistical parameters calculated from: Input Data or Filtered Input Data
- Examples of features are: Mean, Variance, Energy, Peak to Peak, ...

Decision Tree

- is a predictive model built from **training data**. Outputs of the computation blocks are the inputs of the decision tree.
- Each **Node** is characterized by one «if-then-else» condition.
 - Mean on Acc_X < 0.5 g
 - Variance on Gyro_Z < 200 dps
- Decision tree can either generate a result at every sample or filter the results with a **Meta-classifier**, to have a more robust output.

ST toolbox for machine learning core (MLC) in the sensor



Data collection: Key points

Class definition

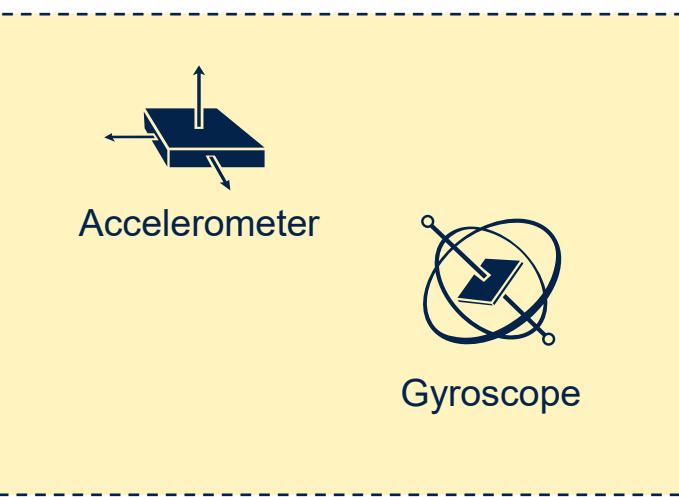
Class 1 >> Motor stop...
Class 2 >> Motor running OK
Class 3 >> Motor running NG

Motor vibration example

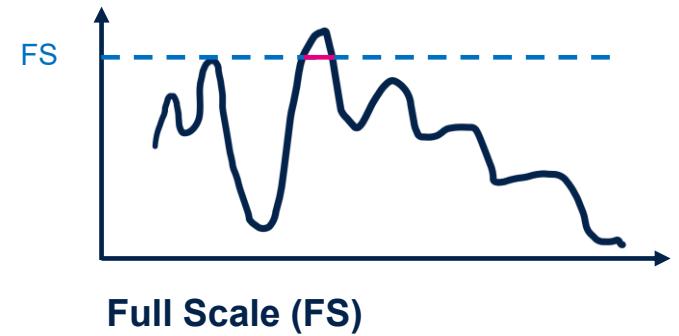
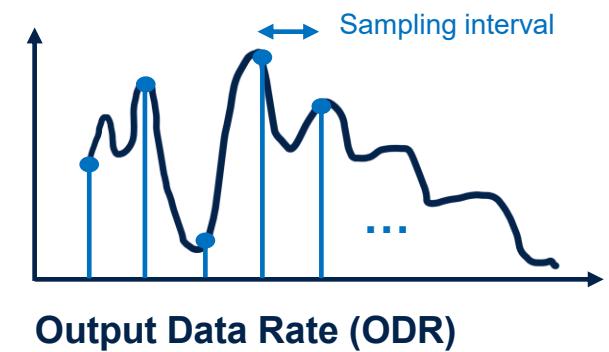
Class 1 >> Still
Class 2 >> Walking
Class 3 >> Running
Class 4 >> Riding bicycle
Class 5 >> Car

Human Activity Recognition example

Sensor selection



Sensor configuration





Raw data



Acc_X, Acc_Y, Acc_Z
Acc_V, Acc_V2



Gyro_X, Gyro_Y, Gyro_Z
Gyro_V, Gyro_V2



ExtSens_X, ExtSens_Y,
ExtSens_Z
ExtSens_V, Ext_Sens_V2



Filters

Filtered data
(High-pass, Band-pass,
IIR1 and IIR2 filters)



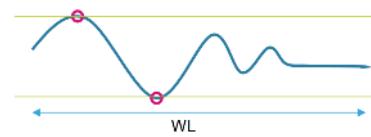
Features calculation inside MEMS

Features

Mean

$$\frac{1}{WL} \sum_{k=0}^{WL-1} I_k$$

Peak-to-Peak



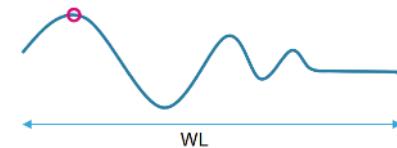
Variance

$$\left(\frac{\sum_{k=0}^{WL-1} I_k^2}{WL} \right) - \left(\frac{\sum_{k=0}^{WL-1} I_k}{WL} \right)^2$$

Energy

$$\sum_{k=0}^{WL-1} I_k^2$$

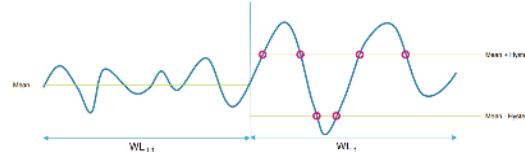
Maximum



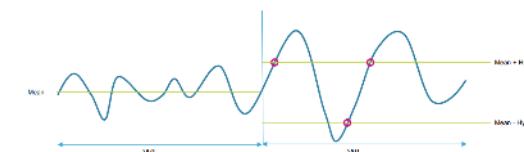
Minimum



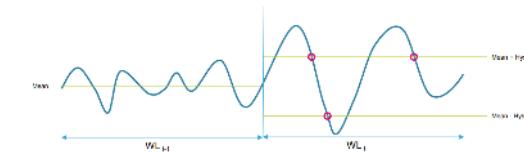
Zero crossing



Positive Zero crossing



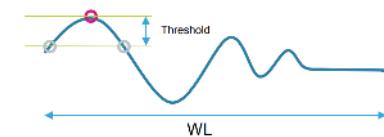
Negative Zero crossing



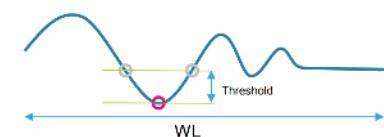
Peak detector



Positive Peak detector



Negative Peak detector

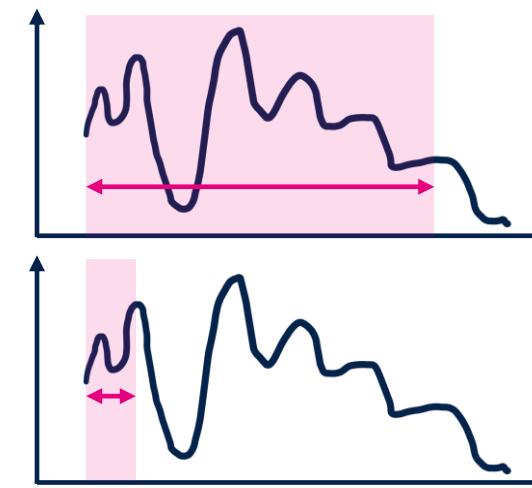
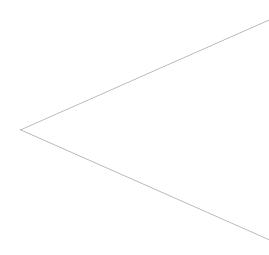


Note: the window for features computation (WL) is configurable (from 1 to 255 samples). It is not a moving window

Window size selection

Window length is strongly dependent on the application

The window should capture all of the information for the decision tree to separate different classes



A few tips for selecting the right size:



- Selection by the user **after data analysis and visualization**
- If the **motion is repetitive** at some frequency, ideal window size should be long enough to capture the entire motion pattern
- Short window length could make the **response quick**
- **Consider the slowest varying signal**



Decision Tree Output

- Decision tree results are accessible through dedicated registers in IMU
- An interrupt is generated when the decision tree result changes
Interrupt status bits are available in dedicated registers
- Interrupt can be routed to INT1/2 pad through dedicated register

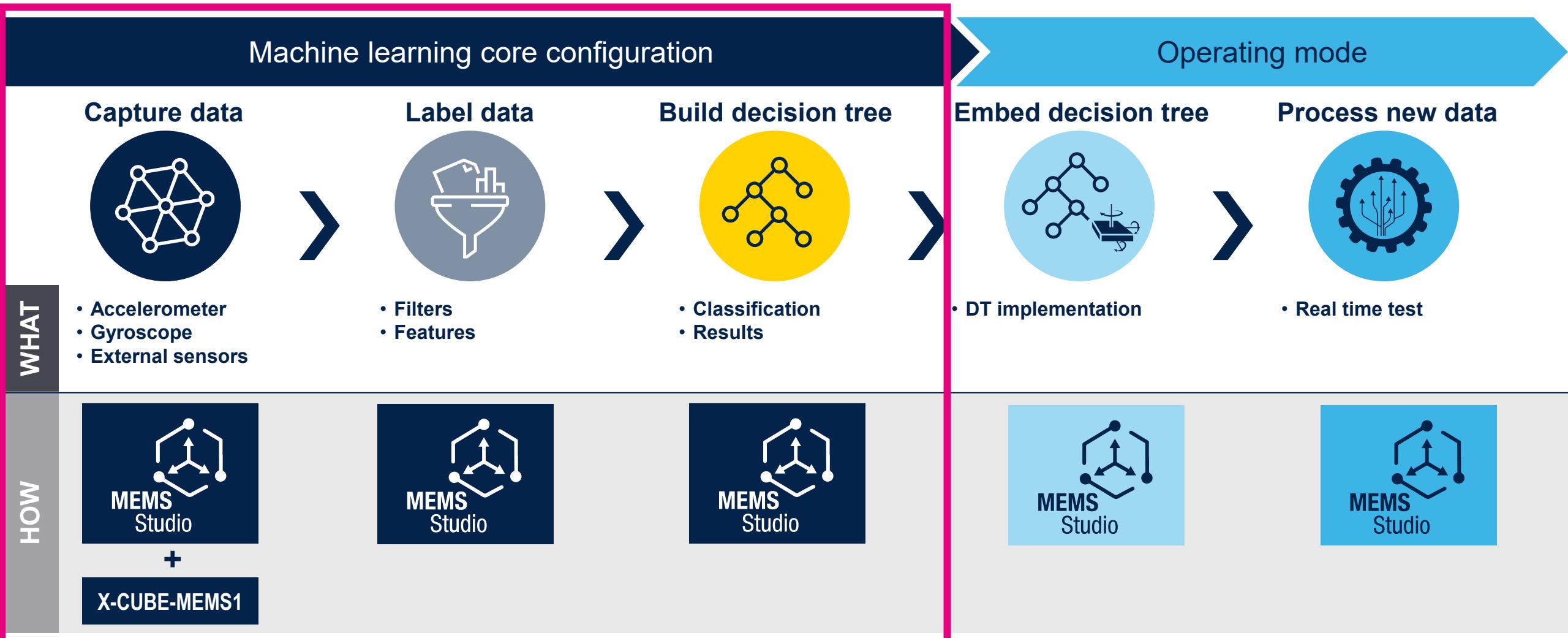
Register	Content
MLC0_SRC (70h)	Result of decision tree 1
MLC1_SRC (71h)	Result of decision tree 2
MLC2_SRC (72h)	Result of decision tree 3
MLC3_SRC (73h)	Result of decision tree 4
MLC4_SRC (74h)	Result of decision tree 5
MLC5_SRC (75h)	Result of decision tree 6
MLC6_SRC (76h)	Result of decision tree 7
MLC7_SRC (77h)	Result of decision tree 8

Register	Content
MLC_STATUS_MAINPAGE (38h)	Contains interrupt status bits for changes in the decision tree result
MLC_STATUS (15h)	Contains interrupt status bits for changes in the decision tree result
MLC_INT1 (0Dh)	Allows routing of interrupt status bits for decision trees to INT1 pin
MLC_INT2 (11h)	Allows routing of interrupt status bits for decision trees to INT2 pin



MLC Configuration Demo

MLC Configuration



NOTE: This section is intended to be a live demo, showing how the flow to generate mlc config file works using MEM Studio

Step 1 – Capture Data

Prepare Board for Data Capture

- Use Datalog Extended Firmware from X-CUBE-MEMS1 Package
 - **Note:** Firmware is Board Specific – use the right combo for the board stack you have
 - X-CUBE-MEMS1\Projects\NUCLEO-U575ZI-Q\Examples\IKS4A1\DataLogExtended\Binary
- Use STM32CubeProgrammer to flash the DatalogExtended binary to the board
- Disconnect from board
- Reset the board

Step 1 – Capture Data

Configure Sensors for Data Capture

- Open MEMS Studio
- Connect to the board
- Select LIS2DUXS12 as accelerometer and click select
- In Quick Setup tab, select accelerometer FS=2g and ODR 12.5Hz (these match Motion Intensity example on github)
- Click Save to File tab
- Set log file location, e.g. C:\ST67_Workshop\MLC_Demo\
- In Data box, right click then unselect all. Then check Timestamp and Accelerometer.
- Ensure Time base is set to timestamp

Step 1 – Capture Data

Run Data Capture

- Open Line charts tab to show accel data
- Click Play button (top left) to start data capture
- Hold board in hand for 20s – this is motion intensity 0
- Gently shake the board along all 3 axes for 20s – this is motion intensity 1
- Vigorously shake the board along all 3 axes for 20s – this is motion intensity 2
- Stop the capture by pressing stop button in top left corner of MEMS Studio

Step 2 – Label Data

- Open Data Analysis tab
- Load the accel data capture file you created with the 3 types of motion
- Use time domain graph
- Click on graph to select regions and label Motion Intensity 0, 1 and 2: MI0, MI1, MI2
- Once all 3 MI's are labeled, click Save Labeled Data to generate 3 individual motion capture files, save them to same folder as original data capture file, e.g: C:\ST67_Workshop\MLC_Demo
- Use file explorer to show that 3 files have been created with label in file name

Step 3 – Build Decision Tree

Load Data Patterns

- Open Advanced Features tab and select MLC
- Make sure Data Patterns tab is selected
- Set workspace to C:\ST67_Workshop\MLC_Demo
- Select the LIS2DUXS12 sensor
- Using Browse button or file explorer drag & drop to load MI0 file. Assign class label MI0 to it. Click “Load”.
- Repeat above step for MI1 and MI2
- You should have 3 pattern files loaded with classes MI0 through MI2

Step 3 – Build Decision Tree

Configure AFS

- Open AFS tab
- Set Machine Learning Core Settings: ODR 12.5 to match captured data, Window length=25 (2s of data)
- Configure Inputs (to match captured data)
 - Accelerometer_only
 - Full Scale 2g
 - ODR: 12.5
- Leave AFS settings at default and click Run to start automatic feature selection
- Once AFS completes, click Apply to Settings to transfer filters/features to ARFF panel

Step 3 – Build Decision Tree

ARFF Generation

- Open ARFF Generation tab
- Note settings from AFS have been set, review them
- Click Generate ARFF File to generate the ARFF file used in the next step (DT generation)

Step 3 – Build Decision Tree

Decision Tree Generation

- Open Decision Tree Generation tab
- Set number of decision trees to 1
- Click Generate Decision Tree
- Once DT is generated review results and show generated decision tree (show Dectree button)

Step 3 – Build Decision Tree

Generation of Configuration Files

- Open Config generation tab
- For output values for the classes select 0, 1 and 2 to match MI0,1 and 2
- Leave counters at 0
- Click Generate Config file
- Show json file
- Click Export to show how to generate corresponding header file

Step 3 – Build Decision Tree

Test Generated Tree

- Click Load Config into Sensor
- Click Decision Tree Output Viewer
- Press Play button to start data capture/testing
- Show Motion Intensities 0, 1 and 2



Hands-On: Embed MLC into MQTT App

Enabling MLC in Your STM32 Project

- In this hands-on part of the workshop, we'll use a generated MLC example available on ST's github
- Github: <https://github.com/STMicroelectronics/st-mems-machine-learning-core>
- We'll use the LIS2DUXS12 sensor



A screenshot of a GitHub repository page. The URL in the address bar is `st-mems-machine-learning-core / examples / motion_intensity / lis2duxs12 /`. The repository is owned by `lorenzobracco` and is the `first version`. The repository contains the following files:

Name	Last commit message
...	
README.md	first version
lis2duxs12_motion_intensity.h	first version
lis2duxs12_motion_intensity.json	first version

Enabling MLC in Your STM32 Project

- The README file provides useful information for integration
- Overview, Sensor configuration, MLC outputs, interrupts

1 - Introduction

Simple example of motion intensity detection implemented using the variance feature on the accelerometer norm. The classes recognized in this example are eight different levels of intensity, from 0 to 7.

For information on how to integrate this algorithm in the target platform, please follow the instructions available in the README file of the [examples](#) folder.

For information on how to create similar algorithms, please follow the instructions provided in the [tutorials](#) folder.

2 - Sensor configuration and orientation

The accelerometer is configured with $\pm 2 \text{ g}$ full scale and 12.5 Hz output data rate.

Any sensor orientation is allowed for this algorithm.

3 - Machine Learning Core configuration

Just one feature (variance), has been applied to the norm of the accelerometer data. The MLC runs at 12.5 Hz, computing features on windows of 39 samples (corresponding to almost 3 seconds). One decision tree with 7 nodes has been configured to detect the different classes. A meta-classifier has not been used.

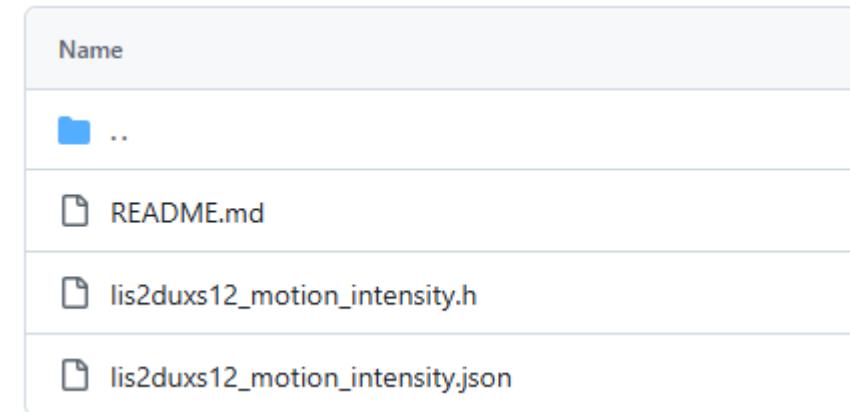
- MLC0_SRC (70h) register values
 - 0 = Intensity_0
 - 1 = Intensity_1
 - 2 = Intensity_2
 - 3 = Intensity_3
 - 4 = Intensity_4
 - 5 = Intensity_5
 - 6 = Intensity_6
 - 7 = Intensity_7

4 - Interrupts

The configuration generates an interrupt (pulsed and active high) on the INT1 pin every time the register MLC0_SRC (70h) is updated with a new value. The duration of the interrupt pulse is 77 ms in this configuration.

Enabling MLC in Your STM32 Project

- Folder contains 3 files:
 - README.md – useful info
 - JSON config file – useful for testing using ST Tools e.g MEMS Studio
 - C header file – useful for integration into STM32 projects



- We'll use the header file to enable the MLC in our MQTT Project
- We'll send the detected motion intensity to the MQTT broker and display it in the mobile app

Enabling MLC in Your STM32 Project

- Header file contains an array with a sequence of commands to execute to configure the sensor and MLC
- Each array entry contains an opcode and relevant info, such as register addresses and data

```
static const struct mems_conf_op lis2duxs12_motion_intensity_conf_0[] = {  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x13, .data = 0x10 },  
    { .type = MEMS_CONF_OP_TYPE_DELAY, .data = 5 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x14, .data = 0x00 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x3F, .data = 0x80 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x04, .data = 0x00 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x05, .data = 0x00 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x17, .data = 0x40 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x02, .data = 0x01 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x08, .data = 0xB8 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x09, .data = 0xE6 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x09, .data = 0x00 },  
    { .type = MEMS_CONF_OP_TYPE_WRITE, .address = 0x09, .data = 0xF0 },
```

Enabling MLC in Your STM32 Project

- We've added the necessary code to configure the MLC, read the outputs and publish the motion intensity to the MQTT broker
- Modified files:
 - Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT\X-CUBE-MEMS1\App\sys_sensors.h
 - Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT\X-CUBE-MEMS1\App\sys_sensors.c
 - Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT\App_MQTT\App\app_config.h
 - Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT\App_MQTT\App\main_app.c

Enabling MLC in Your STM32 Project

- Modified files: sys_sensors.h

```
26 /* Includes ----- */  
27 /* Config ----- */  
28 #define SENSOR_ENABLED 1  
29 #define MLC_CONFIGURATION lis2duxs12_motion_intensity_conf_0  
30
```

```
42 typedef struct  
43 {  
44     float pressure;          /*!< in mbar */  
45     float temperature;       /*!< in degC */  
46     float humidity;         /*!< in % */  
47     Sys_Sensors_Axes_t acceleration;  
48     Sys_Sensors_Axes_t magnetic_field;  
49     Sys_Sensors_Axes_t angular_velocity;  
50 #ifdef USE_MLC  
51     uint8_t motion_intensity; /* motion intensity from mlc, values from 0->7 */  
52 #endif  
53 } Sys_Sensors_Data_t;  
54
```

Enabling MLC in Your STM32 Project

- Modified files: sys_sensors.c

```
#include "logging.h"

#include "iks4a1_motion_sensors_ex.h"
#include "lis2dusxs12_motion_intensity.h" /* MLC Config file */
```

```
/* Private function prototypes -----*/
static int32_t MLC_Init(void);
```

```
int32_t Sys_Sensors_Read(Sys_Sensors_Data_t *data)
{
...
#ifndef USE_MLC
    /* Get mlc outputs */
    (void)IKS4A1_MOTION_SENSOR_Write_Register(IKS4A1_LIS2DUXS12_0, LIS2DUXS12_FUNC_CFG_ACCESS, 0x80);
    (void)IKS4A1_MOTION_SENSOR_Read_Register(IKS4A1_LIS2DUXS12_0, LIS2DUXS12_MLC1_SRC, &data->motion_intensity);
    (void)IKS4A1_MOTION_SENSOR_Write_Register(IKS4A1_LIS2DUXS12_0, LIS2DUXS12_FUNC_CFG_ACCESS, 0x00);
#endif
...
}
```

Enabling MLC in Your STM32 Project

- Modified files: sys_sensors.c

```
static int32_t Sys_Sensors_IKS4A1_Init(void)
{
...
#ifndef USE_MLC
    /* Initialize the MLC */
    ret = MLC_Init();
#endif
...
}
```

Enabling MLC in Your STM32 Project

- Modified files: sys_sensors.c

```
int32_t MLC_Init(void)
{
...
length = MEMS_CONF_ARRAY_LEN(MLC_CONFIGURATION);
i = 0;
while ((i < length) & (ret == BSP_ERROR_NONE))
{
    if (MLC_CONFIGURATION[i].type == MEMS_CONF_OP_TYPE_DELAY)
    {
        HAL_Delay(MLC_CONFIGURATION[i].data);
    }
    else if (MLC_CONFIGURATION[i].type == MEMS_CONF_OP_TYPE_WRITE)
    {
        ret = IKS4A1_MOTION_SENSOR_Write_Register(IKS4A1_LIS2DUXS12_0, MLC_CONFIGURATION[i].address,
MLC_CONFIGURATION[i].data);
    }
    else
    {
        /* Unsupported opcode */
        ret = BSP_ERROR_FEATURE_NOT_SUPPORTED;
    }
    i++;
}
...
```

Enabling MLC in Your STM32 Project

- Modified files: app_config.h

```
/* Local Access Point (e.g. gateway, hotspot, etc) connection parameters */
#define WIFI_SSID                "<wifi-ssid>"

#define WIFI_PASSWORD             "<wifi-password>"

/** Host name of remote MQTT Broker
 * Multiple options are possible:
 * - broker.hivemq.com
 * - broker.emqx.io
 * - test.mosquitto.org
 */
#define MQTT_HOST_NAME           "broker.hivemq.com"

/** Port of remote MQTT Broker */
#define MQTT_HOST_PORT            1883

/** Security level. only non-secure is currently available */
#define MQTT_SECURITY_LEVEL       0

/** MQTT Client ID to be identified on MQTT Broker */
#define MQTT_CLIENT_ID            "<your-client-id>"
```

Enabling MLC in Your STM32 Project

- Injecting Client ID into IoTMQTTPanel config file:

Open a command prompt

cd to
Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT\resources>

Run the command:

```
IoTMQTTPanel_Helper.py IoTMQTTPanel-250618_134546.json --clientid <Your ClientID> -- broker <MQTT broker> -o  
IoTMQTTPanel-MLC-MotionIntensity.json
```

Copy the output file IoTMQTTPanel-MLC-MotionIntensity.json to your mobile device

Load the configuration file into the IoTMQTTPanel app

Enabling MLC in Your STM32 Project

- Modified files: main_app.c

```
static void Subscription_process_task(void *arg)
{
...
/* Process the field 'motionintensity'. Value type: Int. Format: 1 (range 0-7) */
json = cJSON_GetObjectItemCaseSensitive(child, "motionintensity");
if (json != NULL)
{
    if (cJSON_IsNumber(json) == true)
    {
        LogInfo(" %s: %d\n", json->string, json->valueint);
    }
    else
    {
        LogError("JSON parsing error of %s value.\n", json->string);
    }
}
...
}
```

Enabling MLC in Your STM32 Project

- Modified files: main_app.c

```
#ifdef USE_MLC
    len += snprintf((char *)&mqtt_pubmsg[len], MQTT_MSG_BUFFER_SIZE - len,
                    ", \"temperature\": %.2f, \"humidity\": %.2f, \"pressure\": %.2f, \"motionintensity\":%d",
                    (double)sensors_data.temperature, (double)sensors_data.humidity,
                    (double)sensors_data.pressure, (int)sensors_data.motion_intensity);
#else
    len += snprintf((char *)&mqtt_pubmsg[len], MQTT_MSG_BUFFER_SIZE - len,
                    ", \"temperature\": %.2f, \"humidity\": %.2f, \"pressure\": %.2f",
                    (double)sensors_data.temperature, (double)sensors_data.humidity,
                    (double)sensors_data.pressure);
#endif
```

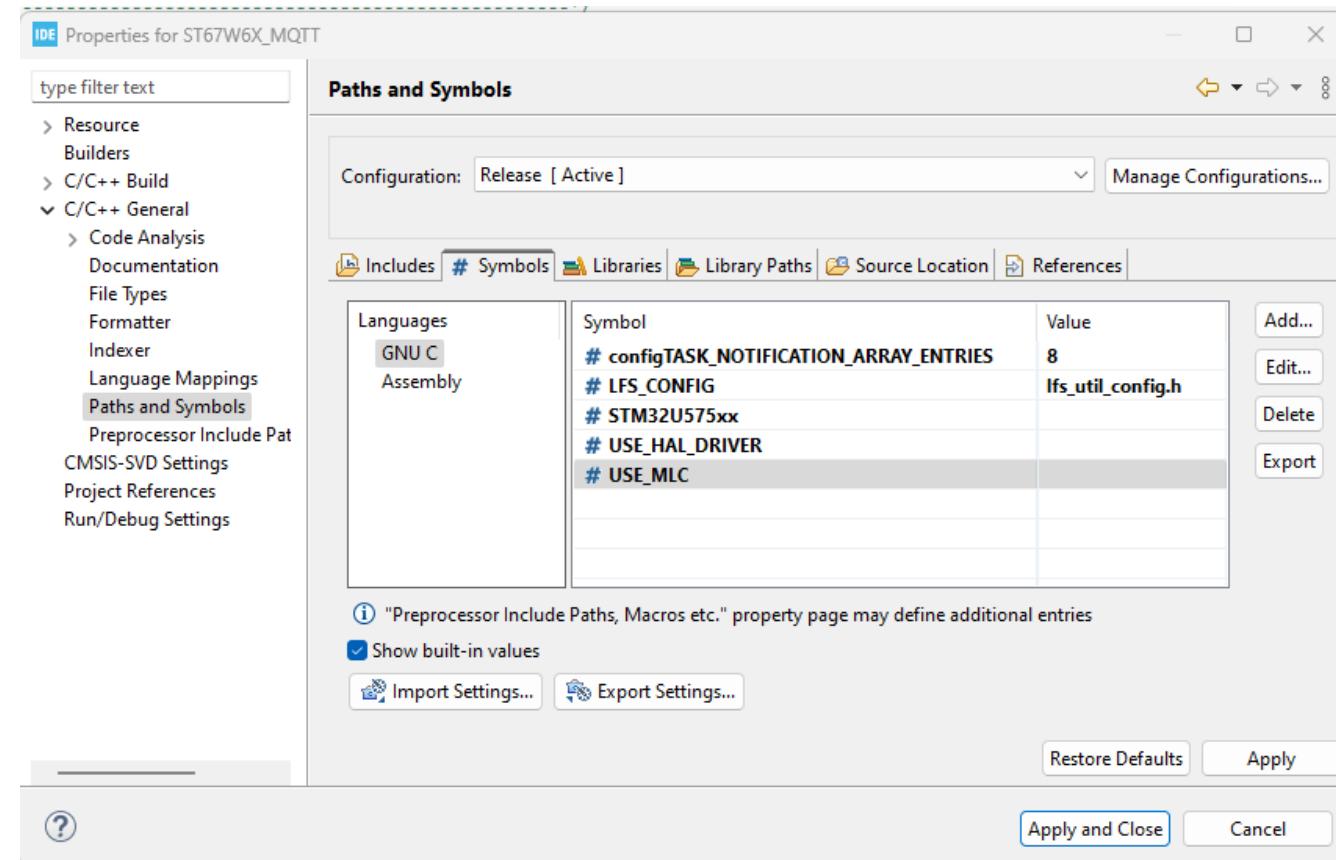
Enabling MLC in Your STM32 Project

- Modified files: main_app.c

```
static void Subscription_process_task(void *arg)
{
...
/* Process the field 'motionintensity'. Value type: Int. Format: 1 (range 0-7) */
json = cJSON_GetObjectItemCaseSensitive(child, "motionintensity");
if (json != NULL)
{
    if (cJSON_IsNumber(json) == true)
    {
        LogInfo(" %s: %d\n", json->string, json->valueint);
    }
    else
    {
        LogError("JSON parsing error of %s value.\n", json->string);
    }
}
...
}
```

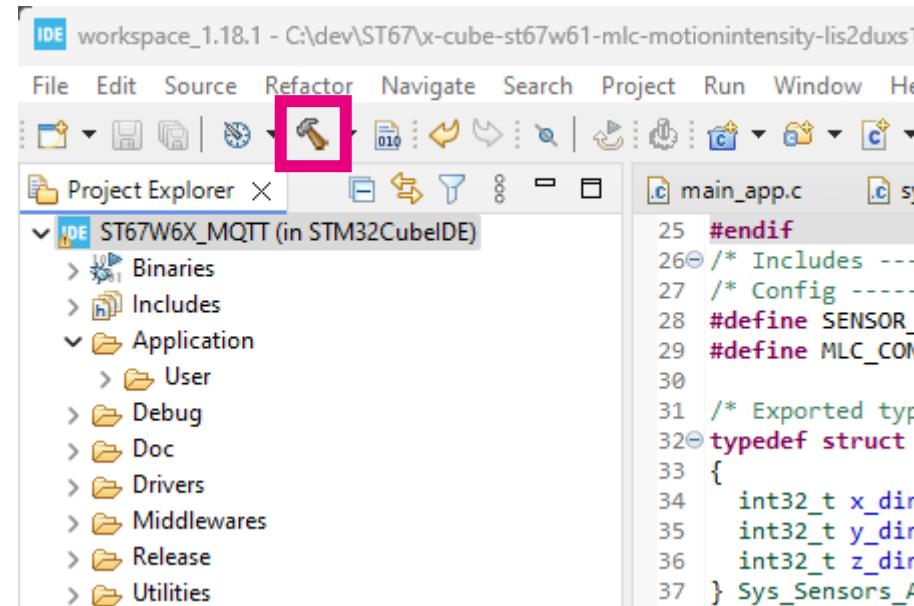
Enabling MLC in Your STM32 Project

- To enable the MLC code, add USE_MLC to project properties



Enabling MLC in Your STM32 Project

- Build the project



- Make sure project builds successfully

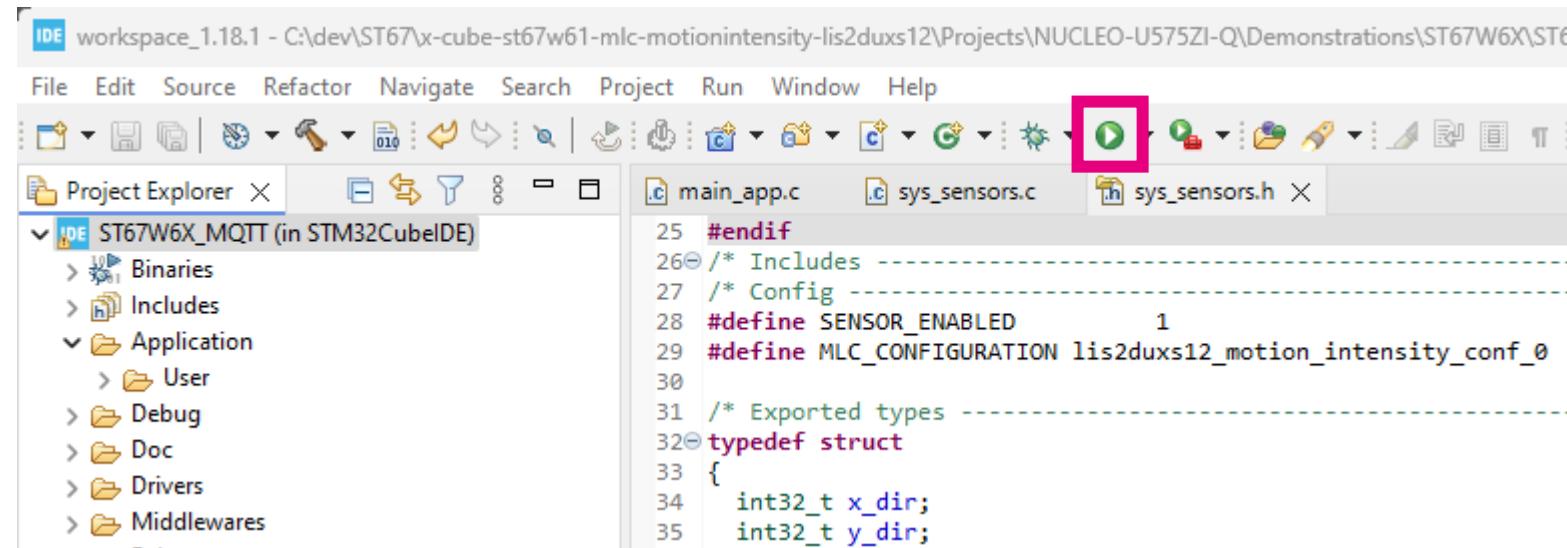
The screenshot shows the CDT Build Console for the project "ST67W6X_MQTT". The console output is as follows:

```
16:48:11 **** Incremental Build of configuration Release for project ST67W6X_MQTT ****
make -j8 all
arm-none-eabi-size ST67W6X_MQTT.elf
    text   data   bss   dec   hex filename
 153188    1616  208944  363748  58ce4 ST67W6X_MQTT.elf
Finished building: default.size.stdout

16:48:12 Build Finished. 0 errors, 0 warnings. (took 1s.51ms)
```

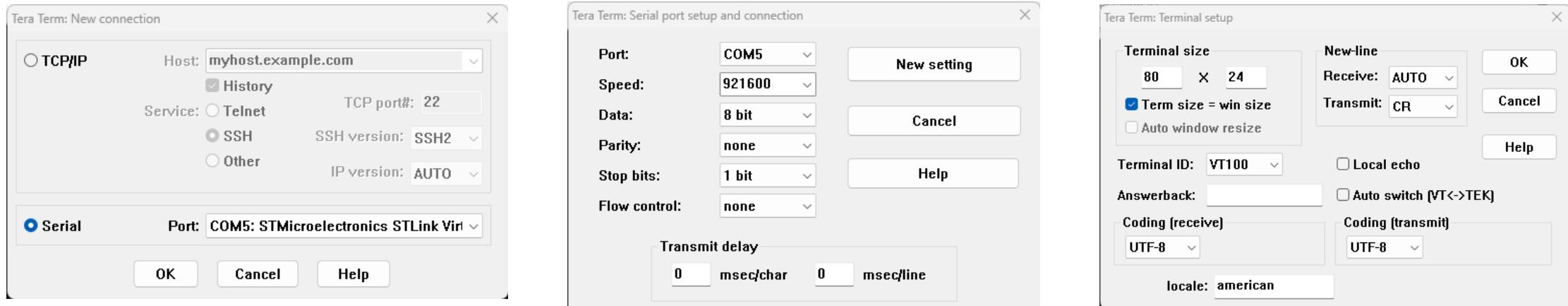
Enabling MLC in Your STM32 Project

- Flash the board



Enabling MLC in Your STM32 Project

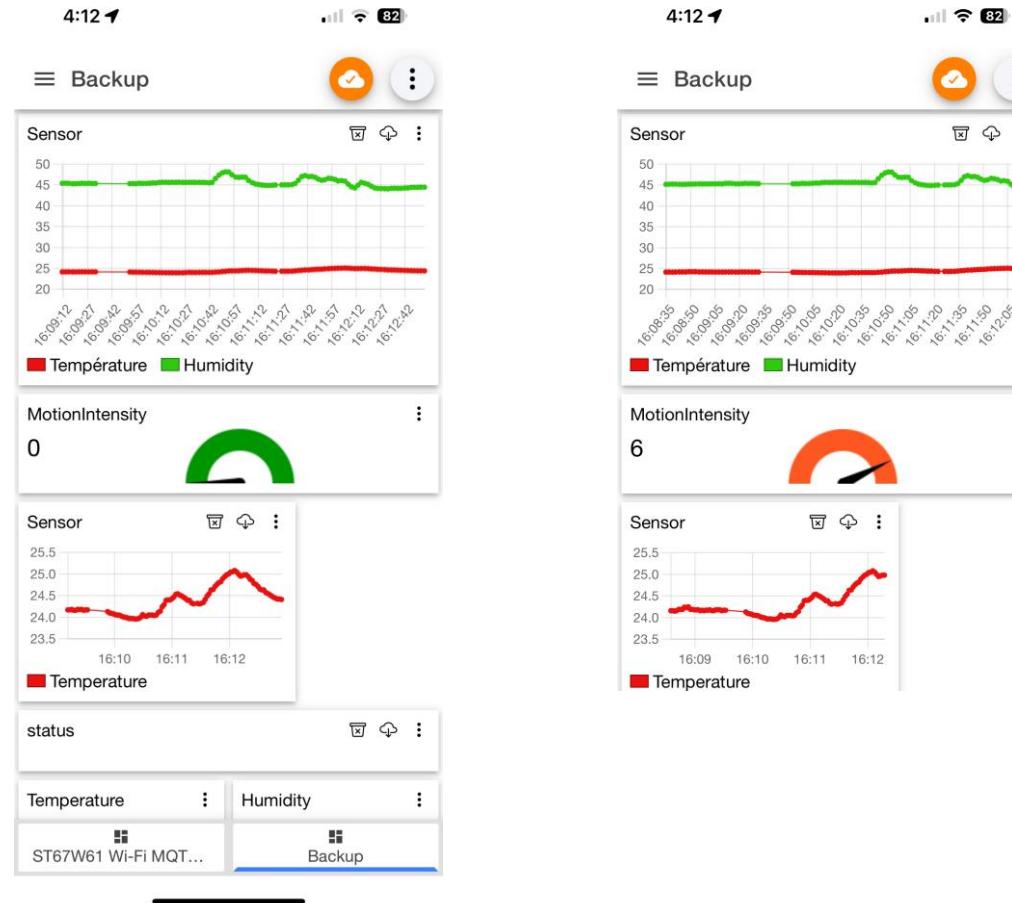
- Connect to board using Tera Term



```
Connecting to Local Access Point
NCP is treating the connection request
DHCP client start, this may take few seconds
Connected to following Access Point :
[00:1e:e5:7a:83:f5] Channel: 6 : RSSI: 0 : SSID: BeamsOpenNet
App connected
SNTP Time: Mon Jul 14 00:58:10 2025
WARNING: Accelerometer ODR changed during MLC Config: before 100.00Hz, after 12.50Hz
MEMS Sensors init successful
MQTT Configure successful
MQTT Connect successful
Subscribing to topic /devices/EGT001/control.
Subscribing to topic /sensors/EGT001.
MQTT Publish OK
MQTT Subscription Received on topic "/sensors/EGT001"
  time: 25-07-14 00:58:10
  rssi: -44
  mac: 40:82:7b:00:08:8b
  temperature: 25.50
  pressure: 1005.17
  humidity: 45.79
  motionintensity: 0
```

Enabling MLC in Your STM32 Project

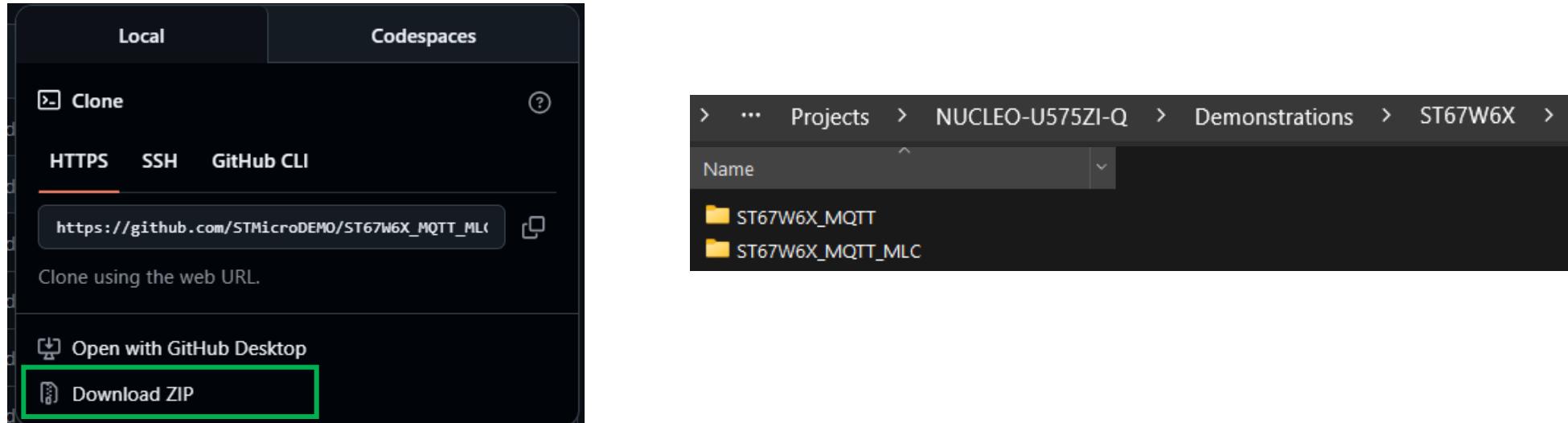
- Open the IoTMQTTPanel app and load the configuration file you generated
- Move/shake the board to see motion intensity value change in the Backup panel





Hands-On: Integrating Embed MLC into MQTTS

- In this hands-on we will use a modified MQTT project that includes the MLC portion as well as additional resources located on GitHub
- [STMicroDEMO/ST67W6X_MQTT_MLC](#)
- Download the project from GitHub and save the file uncompressed within the X-Cube package:
 - C:\X_CUBE_ST67W61_V1.0.0\Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X



**Note: GitHub may save the project as "ST67W6X_MQTT_MLC-main"*

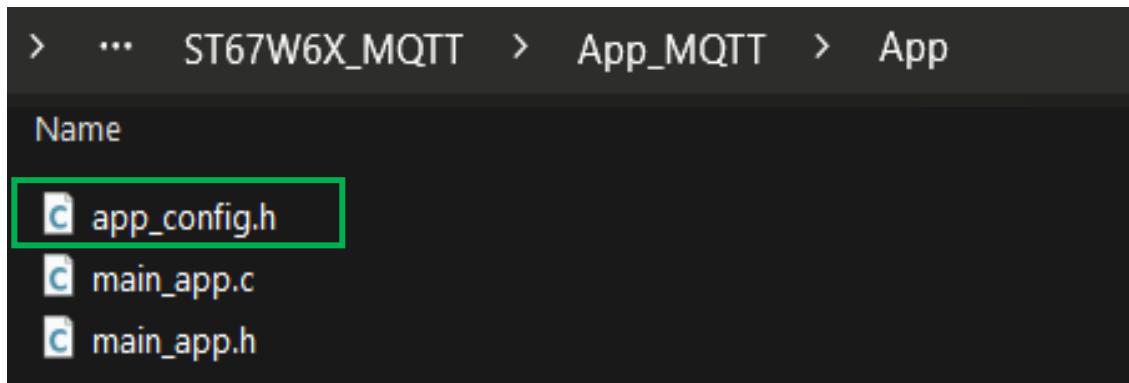
- Higher level security schemes can be enabled to utilize TLS to prevent third party interception of data when publishing data to the cloud.
- Security Schemes:
 - 0 - MQTT over TCP with no authentication and no encryption
 - 1 - MQTT over TLS with username authentication
 - 2 - MQTT over TLS with server certificate
 - 3 - MQTT over TLS with client certificate
 - 4 - MQTT over TLS with both certificates
- The different configurations must be followed when configuring the broker and client.

```
127 /* MQTT Broker connection */
128 static W6X_MQTT_Connect_t mqtt_connect =
129 {
130     .HostName = MQTT_HOST_NAME,      /*!< Host name of remote MQTT Broker */
131     .HostPort = MQTT_HOST_PORT,    /*!< Port of remote MQTT Broker */
132     .Scheme = MQTT_SECURITY_LEVEL, /*!< Security Level. only non-secure is currently available */
133     .MQClientID = MQTT_CLIENT_ID, /*!< MQTT Client ID to be identified on MQTT Broker */
134     .MQUserName = MQTT_USERNAME,   /*!< MQTT Username to be identified on MQTT Broker. Not used in non-secure */
135     .MQUserPwd = MQTT_USER_PASSWORD, /*!< MQTT Password to be identified on MQTT Broker. Not used in non-secure */
136     .Certificate = MQTT_Certificate, /*!< Client Certificate. Required when the scheme is greater or equal to 2 */
137     .PrivateKey = MQTT_Key,        /*!< Client Private key. Required when the scheme is greater or equal to 2 */
138     .CACertificate = MQTT_CA_CERTIFICATE, /*!< CA certificate. Required when the scheme is greater or equal to 2 */
139     .SNI_enabled = 0
140 };
```

mqtt_connect structure that must be configured for each MQTT use case.

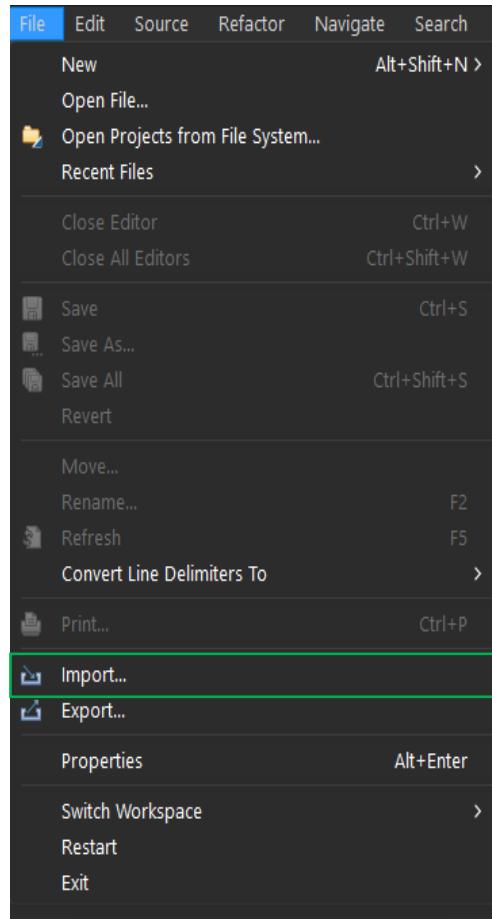
- When enabling security schemes 2 or higher, certificates and keys must be provided to verify security credentials.
- The NCP utilizes littlefs to read certificates and keys from the host MCU that are then stored internally.
- To load certificates and keys onto the host MCU, desired certificates and keys must be placed within the Ifs folder in the MQTT demonstration folder:
 - *Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT\littlefs\Ifs*
- The desired certificates and keys are loaded onto the host MCU as a littlefs.bin using linker input specified in the project settings.
- This binary can be built from the user specified certificates and keys using the build.bat script once the certificates and keys are placed in the Ifs folder.

- In this hands-on portion of the work-shop TLS will be utilized within the MQTT demo using security scheme 1.
- Security scheme 1 - MQTT over TLS with username and password
- MQTT Local Client configuration – Setup on ST67 within CubeIDE in app_config.h



ST67W6X MQTT

Import Project into STM32CubeIDE



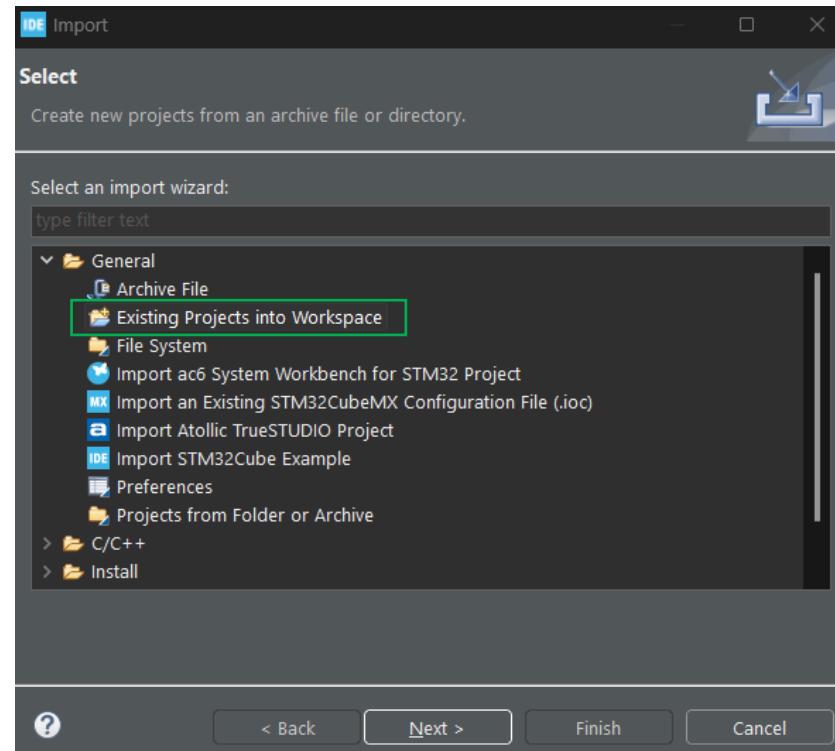
Import the modified MQTT application into CubeIDE

1. Select “File”
2. Select “Import...”

ST67W6X MQTT

Import Project into STM32CubeIDE

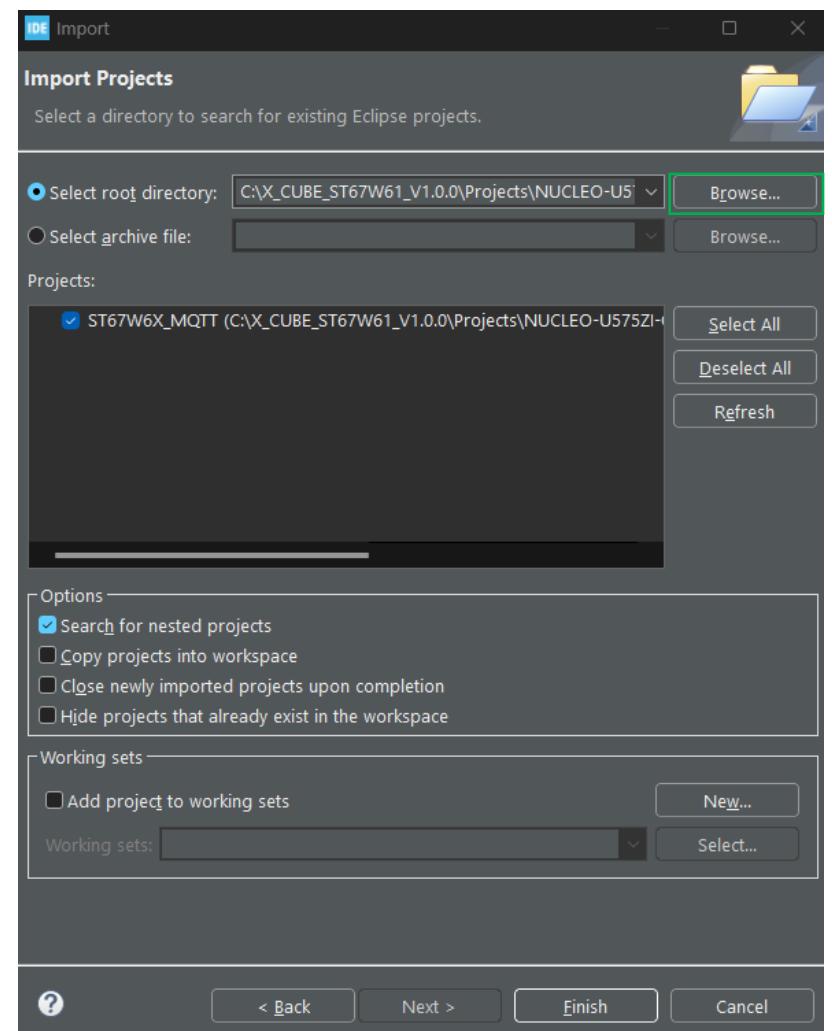
3. Select “Existing Projects into Workspace”
4. Select “Next”



ST67W6X MQTT

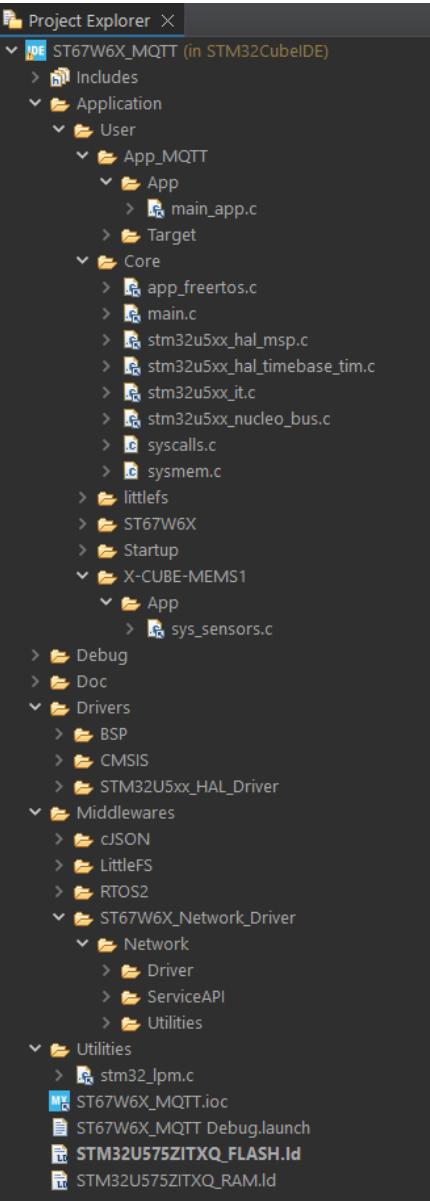
Import Project into STM32CubeIDE

5. Select “Browse...”
6. Navigate to “C:\ x-cube-st67w61-v1.0.0\Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT_MLC”
7. Select “Finish”



ST67W6X MQTT

Project into STM32CubeIDE

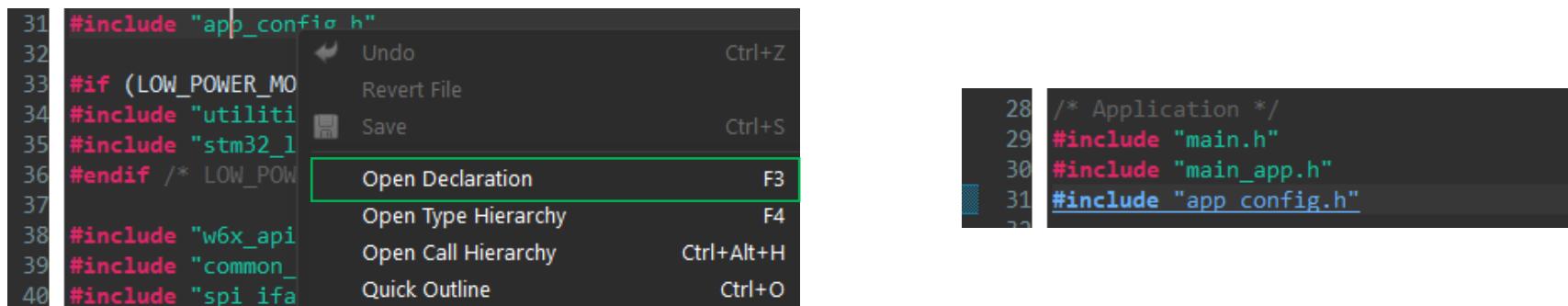


Project Structure:

- Application – Contains the files for the application functionality (i.e. main.c, main_app.c, sys_sensors.c, etc.)
- Drivers – Contains the HAL driver, CMSIS, and BSP libraries for STM32U575 and IKS4A1.
- Middlewares – Contains the network driver libraries for the ST67 modules.
- Utilities – Contains the stm32_lpm.c file.

Open app_config.h:

- Open the file main_app.c found in the Application library under User/App_MQTT/App.
- With main_app.c open, to open app_config.h, “right-click” and select “Open Declaration” on app_config.h or hold CTRL and “left-click” on app_config.h.**



**Note: You must first try building the application to open the file by selecting “Open Declaration”.

- Modified app_config.h to enable security scheme 1

```
83  /** Port of remote MQTT Broker */
84  #define MQTT_HOST_PORT          8883
85
86  /** Security level. only non-secure is currently available */
87  #define MQTT_SECURITY_LEVEL      1
88
89  /** MQTT Client ID to be identified on MQTT Broker */
90  #define MQTT_CLIENT_ID           "hands-on"
91
92  /** MQTT Username to be identified on MQTT Broker. Not used in non-secure */
93  #define MQTT_USERNAME             "st67workshop"
94
95  /** MQTT Password to be identified on MQTT Broker. Not used in non-secure */
96  #define MQTT_USER_PASSWORD        "12345678"
97
98  /** MQTT Client Certificate. Required when the scheme is greater or equal to 3 */
99  #define MQTT_CERTIFICATE         "client_1.crt"
100
101 /** MQTTClient Private key. Required when the scheme is greater or equal to 3 */
102 #define MQTT_KEY                  "client_1.key"
103
104 /** MQTT Client CA certificate. Required when the scheme is greater or equal to 2 */
105 #define MQTT_CA_CERTIFICATE       "ca_1.crt"
106
107 /** MQTT Server Name Indication (SNI) enabled */
108 #define MQTT_SNI_ENABLED          1
```

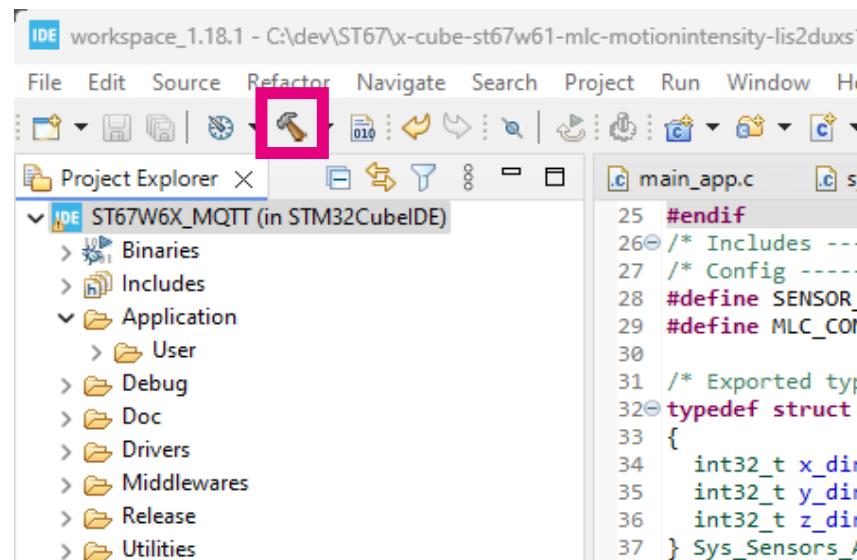
- Modifying the port to 8883 corresponding to TLS rather than TCP
- Security scheme set to 1 corresponding to use username and password fields
- Specified to unique client ID (not necessary for when enabling security)
- Specified username, please avoid username with spaces
- Specified password

- Modified app_config.h to provide access point credentials

```
61 /* Local Access Point (e.g. gateway, hotspot, etc) connection parameters */
62 #define WIFI_SSID           "SSID"
63
64 #define WIFI_PASSWORD        "PSWD"
```

Note: SSID **cannot** have special characters, and both SSID and password should **not** have spaces

- Once modified with these security changes and with the changes made to enable to MLC portion, save the project and build the application.



- Make sure project builds successfully

```
CDT Build Console [ST67W6X_MQTT]
16:48:11 **** Incremental Build of configuration Release for project ST67W6X_MQTT ****
make -j8 all
arm-none-eabi-size ST67W6X_MQTT.elf
    text     data      bss      dec      hex filename
 153188      1616   208944   363748   58ce4 ST67W6X_MQTT.elf
Finished building: default.size.stdout

16:48:12 Build Finished. 0 errors, 0 warnings. (took 1s.51ms)
```

ST67W6X MQTT

Nucleo-U575ZI-Q Serial Terminal Output

```
#### Welcome to ST67W6X MQTT Application #####
# build: 14:31:36 Jul 17 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 10 2025 10:15:04
Module ID:
BOM ID: 0
Manufacturing Year: 2000
Manufacturing Week: 00
Battery Voltage: 3.340 V
Trim Wi-Fi hp: 6,6,6,6,6,5,5,5,5,5,4,5
Trim Wi-Fi lp: 5,5,6,6,7,7,7,7,8,8,8,8
Trim BLE: 4,4,4,3,4
Trim XTAL: 38
MAC Address: 40:82:7b:00:0e:c1
Anti-rollback Bootloader: 0
Anti-rollback App: 0

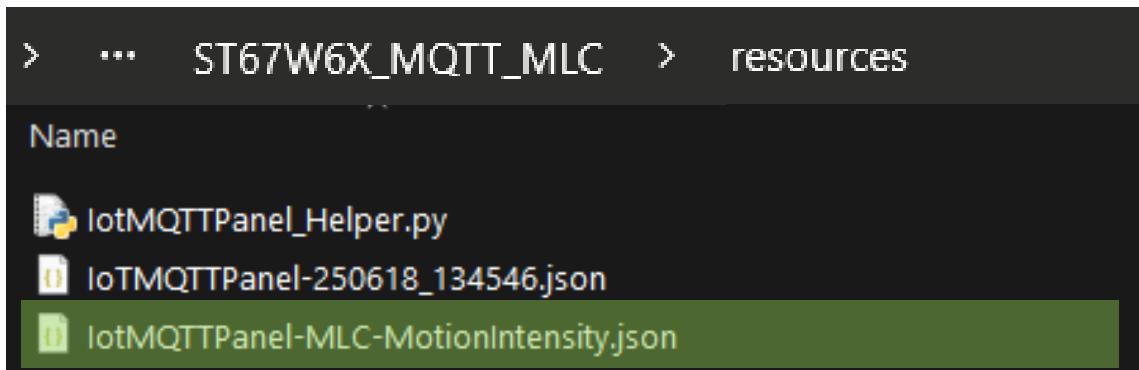
mount success
Wi-Fi init is done
Net init is done
MQTT init is done
```

```
MQTT Publish OK
MQTT Subscription Received on topic "/sensors/hands-on"
  time: 25-07-23 18:37:22
  rssi: -47
  mac: 40:82:7b:00:0e:c1
  temperature: 23.55
  pressure: 1005.77
  humidity: 63.05
  motionintensity: 0
MQTT Publish OK
MQTT Subscription Received on topic "/sensors/hands-on"
  time: 25-07-23 18:37:24
  rssi: -48
  mac: 40:82:7b:00:0e:c1
  temperature: 23.57
  pressure: 1005.78
  humidity: 63.16
  motionintensity: 0
MQTT Publish OK
MQTT Subscription Received on topic "/sensors/hands-on"
  time: 25-07-23 18:37:26
  rssi: -48
  mac: 40:82:7b:00:0e:c1
  temperature: 23.56
  pressure: 1005.78
  humidity: 63.06
  motionintensity: 0
```

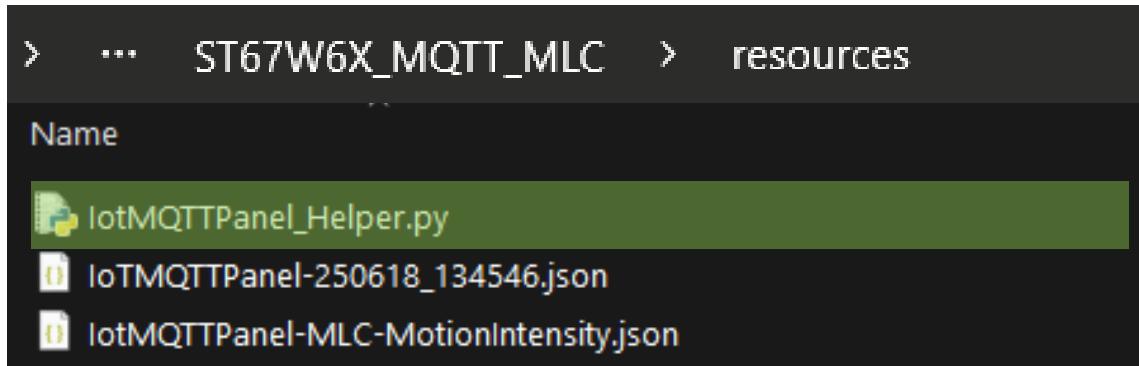
Application running and displaying messages on serial terminal and expected output:

- See FW versions for both host MCU and ST67W6X
- Module initializations (i.e. Wi-Fi, Net, MQTT)
- Scan and connection to AP
- MEMS Init
- MQTT Configure
- MQTT Publish and Subscription received

- The IoT MQTT Panel application must be modified to use a username and password as well as match user specifications.
- The IoTMQTPanel-MLC-MotionIntensity.json is used to configure the IoT MQTT Panel application.



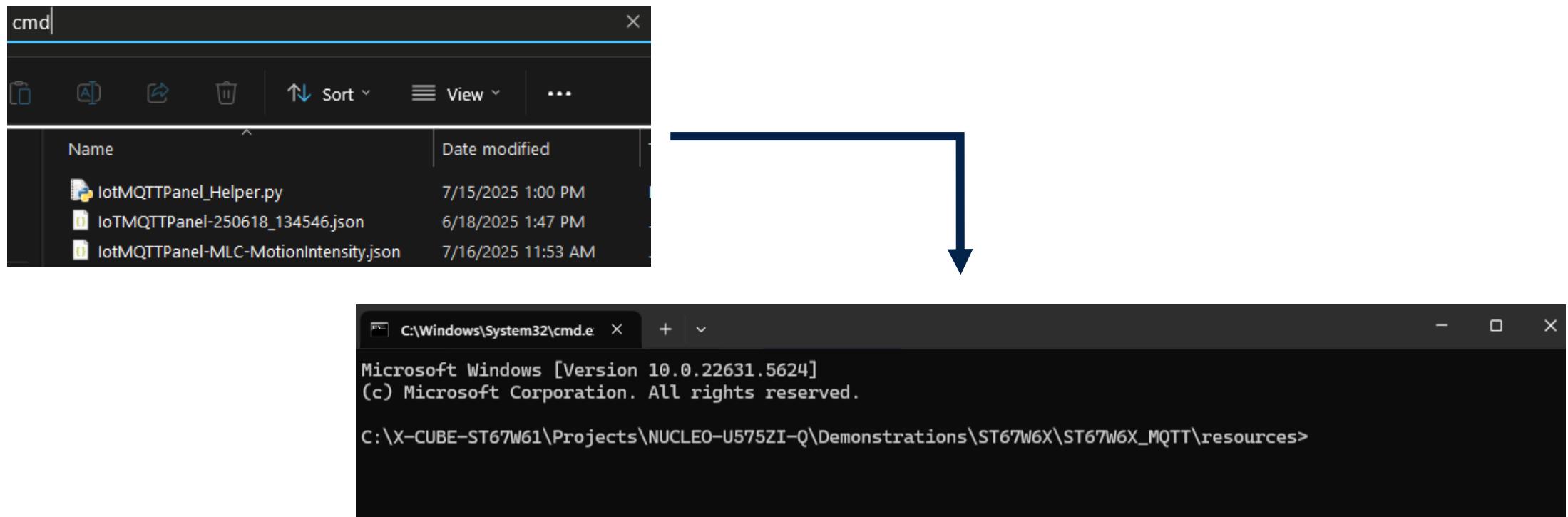
- The .json file can be modified quickly to match the user's specifications using the `IoTMQTPanel_Helper.py` script.
 - The Python script is located within the project directory at:
 - `Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT_MLC\resources`



ST67W6X MQTT

IoT MQTT Panel Client Setup - Security Scheme 1

- A command window can be opened at the directory containing the python script by typing in “cmd” in the address bar of the file explorer window.



- To see the available parameters that can be edited within the IoTMQTTPanel_Helper.py provide the option ‘-h’ when calling the script

```
C:\X_CUBE_ST67W61_V1.0.0\Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT_MLC\resources>python IoTMQTTPanel
_Helper.py -h
usage: IoTMQTTPanel_Helper [-h] [-o OUT] [-b BROKER] [-c CLIENTID] [-u USERNAME] [-p PASSWORD] [--version] JSONin

Pretty print JSON converter and ClientID injection

positional arguments:
  JSONin           Input JSON File

options:
  -h, --help        show this help message and exit
  -o OUT, --out OUT    Output JSON to file
  -b BROKER, --broker BROKER
                      MQTT broker
  -c CLIENTID, --clientid CLIENTID
                      MQTT Client ID to inject
  -u USERNAME, --username USERNAME
                      MQTT Username credentials
  -p PASSWORD, --password PASSWORD
                      MQTT Password credentials
  --version         show program's version number and exit
```

- When calling the script the JSONin file **must** be specified
- For this hands-on we will be using the `IotMQTTPanel-MLC-MotionIntensity.json`
- We also recommend specifying the “out” file which will save any changes specified in the python script to another .json file

ST67W6X MQTT

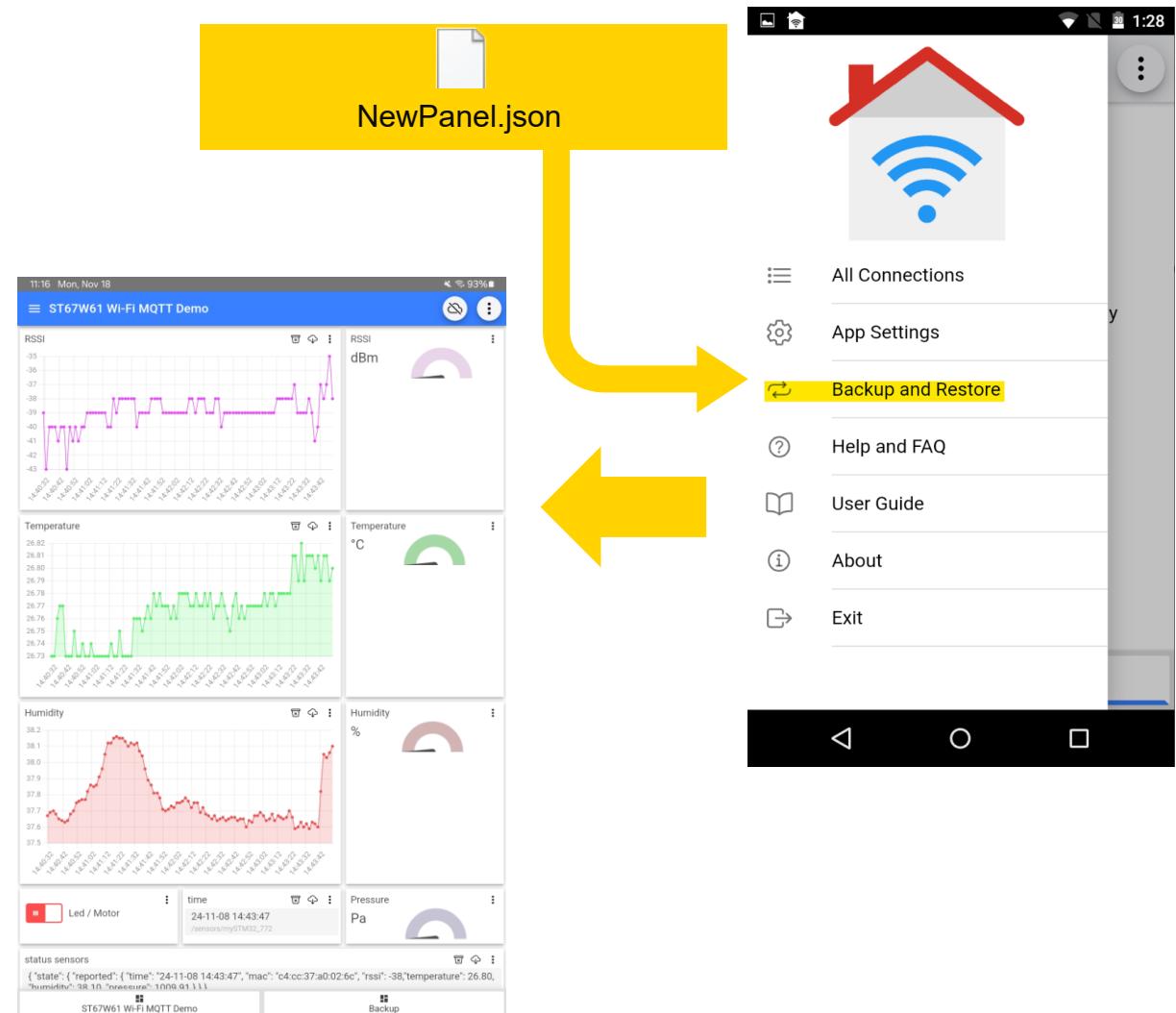
IoT MQTT Panel Client Setup - Security Scheme 1

- Configure the .json file with the configurations specified in app_config.h

```
C:\X_CUBE_ST67W61_V1.0.0\Projects\NUCLEO-U575ZI-Q\Demonstrations\ST67W6X\ST67W6X_MQTT_MLC\resources>python IoTMQTTPanel.py IoTMQTTPanel_Helper.py IoTMQTTPanel-MLC-MotionIntensity.json -o NewPanel.json -c hands-on -u st67workshop -p 12345678
```

```
IoTMQTTPanel_Helper
JSON in: IoTMQTTPanel-MLC-MotionIntensity.json
Inserted ClientID "hands-on" 16 times
Inserted Username "st67workshop" 1 times
Inserted Password "12345678" 1 times
Saving output to: NewPanel.json
Done
```

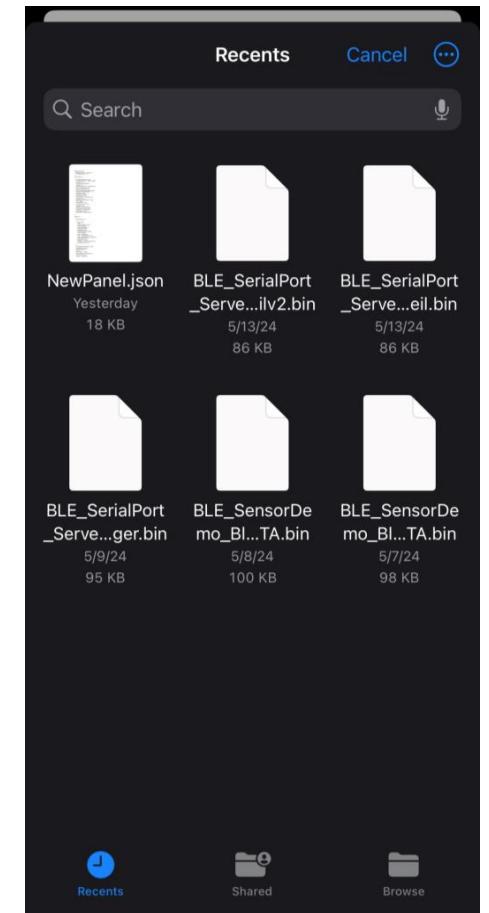
- Setup MQTT Client on IoT MQTT Panel Application
 - 1. Load modified .json file onto smart device
 - 2. Restore connection based on configurations set in .json file
 - 3. Verify connection and data in app with application running on STM32U5+ST67



ST67W6X MQTT

IoT MQTT Panel Client Setup - Security Scheme 1

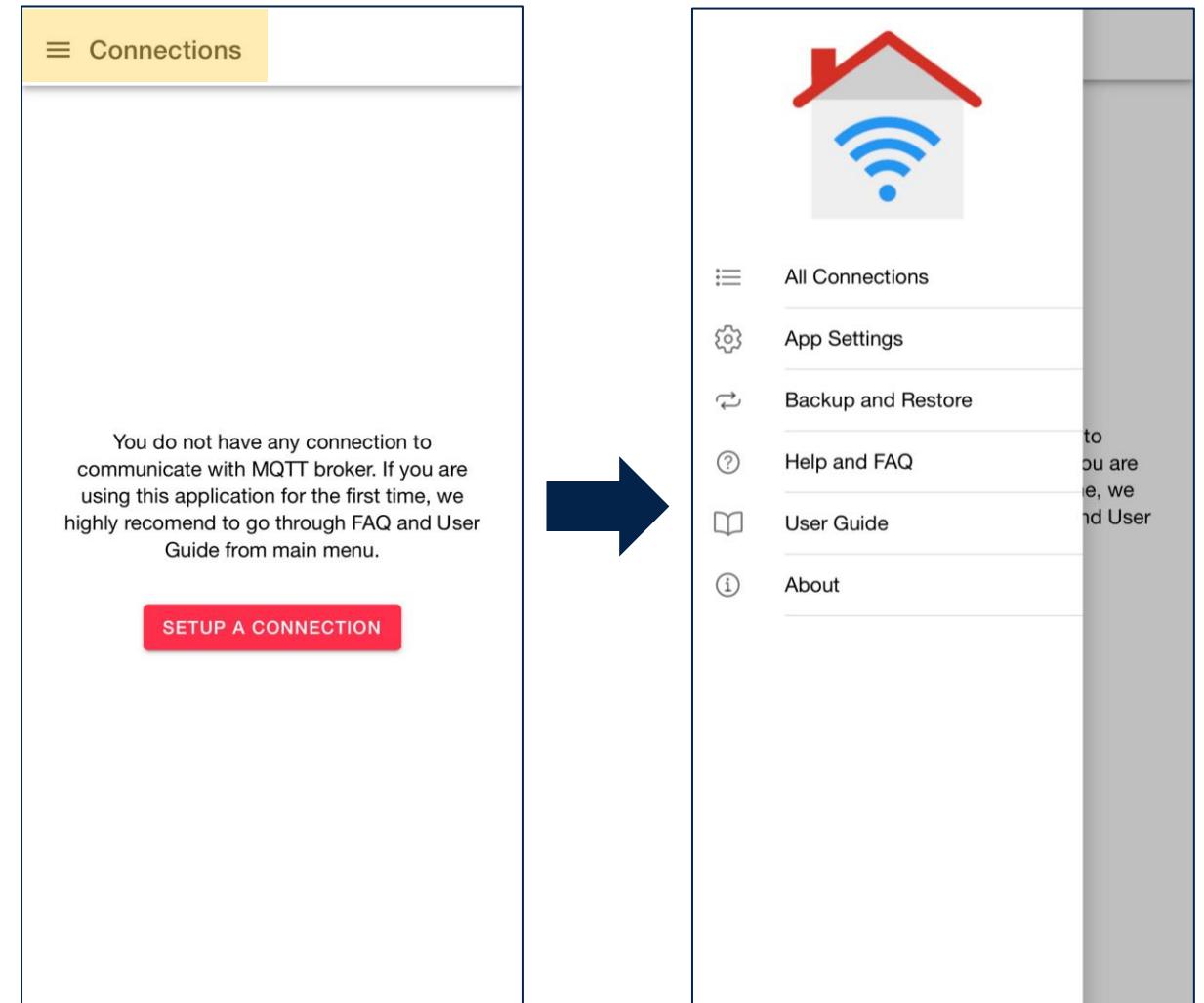
- Modified .json file must be provided to a mobile device with IoT MQTT Panel installed
 - Recommended method is to use a personal email to send the file to your mobile device.



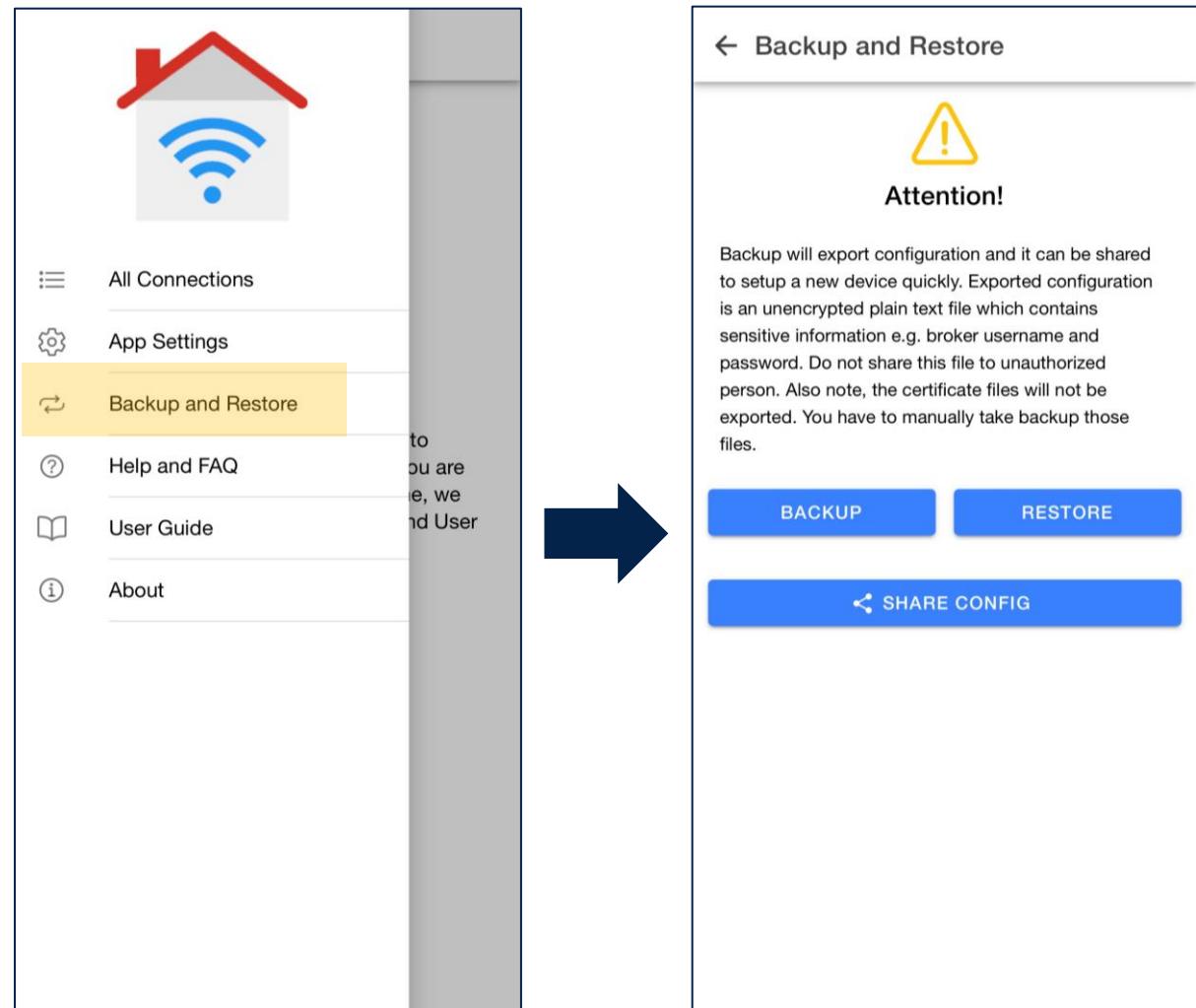
ST67W6X MQTT

IoT MQTT Panel Client Setup – Connection Setup

- With IoT MQTT Panel Application open select “Connections” to bring up MQTT Panel application menu.



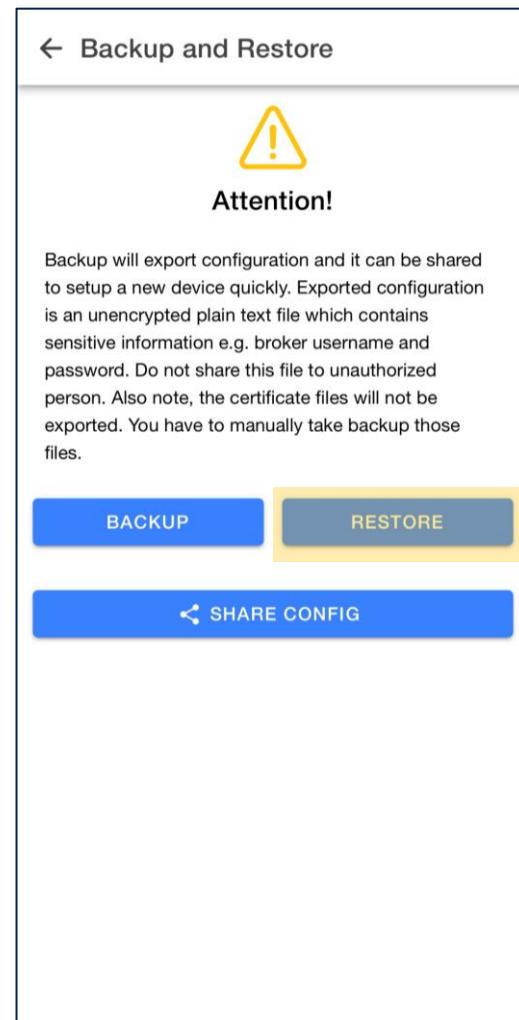
- Select “Backup and Restore” to “Backup” an existing connection or “Restore” a preconfigured connection.



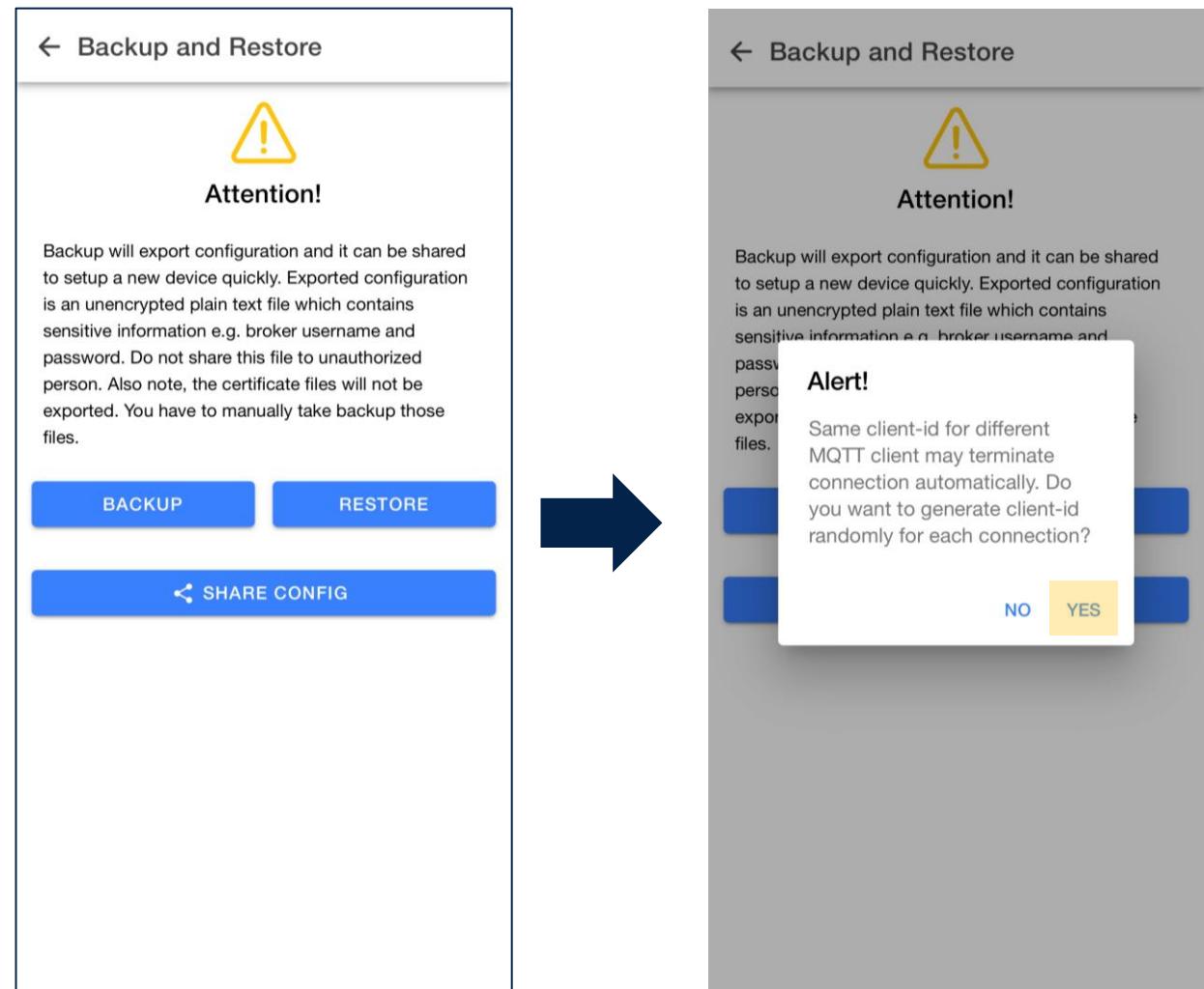
ST67W6X MQTT

IoT MQTT Panel Client Setup – Connection Setup

- Select “Restore” to bringup file storage and select preconfigured .json file stored within smart device’s memory



- With preconfigured .json file selected, the IoT MQTT Panel application will “Alert!” the user regarding same client-id.
- Select “Yes”.
 - Client-ID referred to here is correlated to the connection. Client-ID configured previously in the .json and app_config.h is correlated to the topic data is published under.



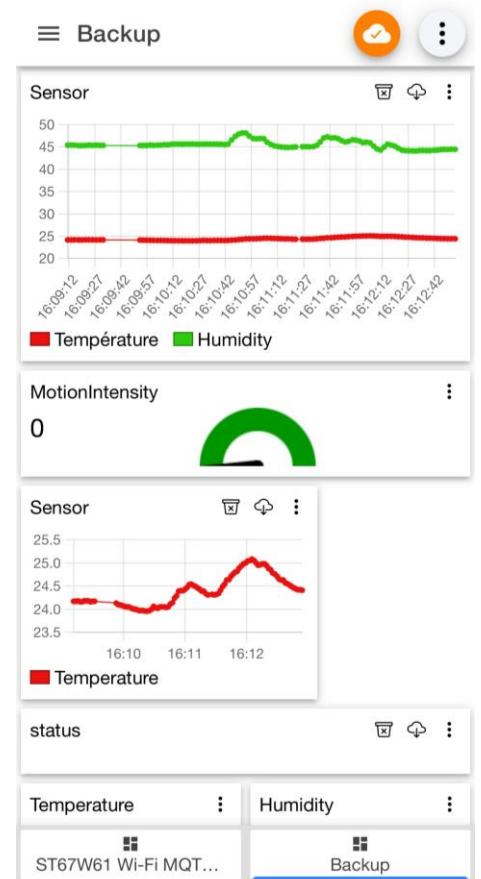
- The main dashboard:

- Display the graphical values of RSSI, Temperature, Humidity and Pressure from device published message
- The Green LED statues on the NUCLEO-U575 Host board can be managed through Button panel



- The backup dashboard:

- Display the graphical values of Humidity, Temperature and Motion Intensity





Documentation



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](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

- 1 Wi-Fi® overview
 - 1.1 What is Wi-Fi®
 - 1.2 Wi-Fi® network architecture
 - 1.3 Wi-Fi® features details

Details about ST67W611M1

- 2 ST67W611M1
 - 2.1 ST67W611M1 overview
 - 2.2 X-CUBE-ST67W61
 - 2.3 ST67W611M1: Hardware and board description
 - 2.4 ST67W611M1 Security overview
 - 2.5 ST67W611M1-based customer end-Product
 - 2.6 ST67W611M1 evaluation

X-CUBE-ST67W61

- 3 X-CUBE-ST67W61 software application notes and user manuals
 - 3.1 Getting started with ST67W611M1

Host boards supported

- 4 ST67W611M1 associated hosts boards

Tools & References

- 5 Specific tools
- 6 Terms and definitions
- 7 References

Other documentation

2

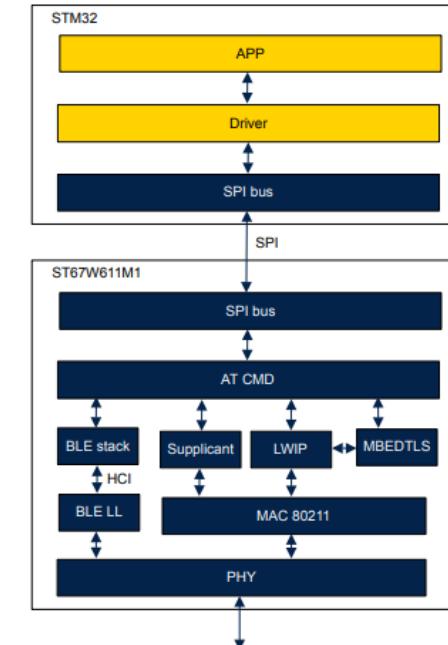
X-CUBE-ST67W61 architecture overview

- Getting started with X-CUBE-ST67W61 [here](#)

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 ST67W61M1

Our technology starts with You



Find out more at www.st.com

© STMicroelectronics - All rights reserved.

ST logo is a trademark or a registered trademark of STMicroelectronics International NV or its affiliates in the EU and/or other countries.

For additional information about ST trademarks, please refer to www.st.com/trademarks.

All other product or service names are the property of their respective owners.

Backup Slides