Getting started with the STMicroelectronics GNSS software package for STM32CubeMX

Introduction

This document provides the guidelines to configure and use the X-CUBE-GNSS1 software package V6.0.0 for STM32CubeMX (minimum required version V6.6.0). The document contains a description of the provided sample applications, a description of the steps required to configure a generic project using the GNSS (Global Navigation Satellite System) driver and middleware, as well as a description of the steps to configure and use the sample applications provided in the package.

Information and documentation related to the Teseo-LIV3F and Teseo-VIC3DA GNSS modules, the X-NUCLEO-GNSS1A1 and the X-NUCLEO-GNSS2A1 expansion boards and the ST expansion software for GNSS are available on www.st.com.

NOTE: Unless it is explicitly written, all settings and configuration steps referred in this document to the Teseo-LIV3F device and the X-NUCLEO-GNSS1A1 expansion board apply also to the Teseo-VIC3DA device and the X-NUCLEO-GNSS2A1 expansion board.

Contents

List of figures

1 Acronyms and abbreviations

Table 1: list of acronyms

Acronym	Description
GNSS	Global Navigation Satellite System
HAL	Hardware Abstraction Layer
I2C	Inter Integrated Circuit
NVIC	Nested Vectored Interrupt Controller
RTOS	Real Time Operating System

2 What is STM32Cube?

STM32Cube is a combination of a full set of PC software tools and embedded software blocks running on STM32 microcontrollers and microprocessors:

- STM32CubeMX configuration tool for any STM32 device; it generates initialization C code for Cortex-M cores and the Linux device tree source for Cortex-A cores
- STM32CubeIDE integrated development environment based on open-source solutions like Eclipse or the GNU C/C++ toolchain, including compilation reporting features and advanced debug features
- STM32CubeProgrammer programming tool that provides an easy-to-use and efficient environment for reading, writing and verifying devices and external memories via a wide variety of available communication media (JTAG, SWD, UART, USB DFU, I2C, SPI, CAN, etc.)
- STM32CubeMonitor family of tools (STM32CubeMonRF, STM32CubeMonUCPD, STM32CubeMonPwr) to help developers customize their applications in real-time
- STM32Cube MCU and MPU packages specific to each STM32 series with drivers (HAL, low-layer, etc.), middleware, and lots of example code used in a wide variety of real-world use cases
- STM32Cube expansion packages for application-oriented solutions

3 License

The software provided in this package is licensed under Software License Agreement SLA0055.

4 Sample Application Description

4.1 GetPos

In this application, real-time GNSS data received by the Teseo-LIV3F device can be displayed through a serial connection and a serial terminal on a PC.
The GetPos application is built on top of an RTOS (either FreeRTOS or AZRTOS ThreadX) support introducing a task (consumer), to parse the messages (enqueued in a shared queue) coming from the Teseo-LIV3F device, and a task (listener) to parse commands coming from the serial terminal.
The sample application also shows three advanced features supported by the Teseo-LIV3F device:

- **Geofencing**: allows the Teseo-LIV3F receiver to raise an NMEA message when the resolved GNSS position is close to enter or exit from a specific circle.
- **Odometer**: provides information on the traveled distance using only the resolved GNSS position.
- **Data Logging**: allows the Teseo-LIV3F receiver to save the resolved GNSS position on the local Flash memory to be retrieved on demand from the host.

The Teseo-LIV3F device sends the received GNSS data via UART (or I2C) to the STM32 microcontroller on the STM32 Nucleo board according to the NMEA 0183 Version 4.0 protocol.

4.2 SimOSGetPos

This application is the simplest of the package. Its main function is to show how real time GNSS data ($GPGGA) received by the Teseo-LIV3F device can be displayed, through a serial connection and a serial terminal, on a PC.
This application works in an infinite loop and retrieves the information provided by the Teseo-LIV3F module. By enabling the Geofence feature, the user can set up a circle and find out if we are in, out or at the border.
This application is tailored to STM32 Nucleo L0 family.
The Teseo-LIV3F device sends via a UART (or I2C) interface the received GNSS data to the STM32 microcontroller, hosted on the Nucleo board, according to the NMEA 0183 Version 4.0 protocol.
This SimOSGetPos sample application is able to:

- Establish a serial connection between the STM32 Nucleo and the X-NUCLEO-GNSS1A1 boards and the PC.
- Parse periodic $GPGGA sentences.
- Enable the Geofence feature.

4.3 VirtualCOMPort

This sample application is used in conjunction with the ST TESEO-SUITE PC software tool (available at www.st.com, together with a Quick Training Guide on how to use it) for managing, configuring and evaluating the Teseo GNSS device.
Also, the VirtualCOMPort application is built on top of an RTOS (either FreeRTOS or AZRTOS ThreadX) support.
The application can:

- Establish a serial connection between the STM32 Nucleo and the X-NUCLEO-GNSS1 boards and the PC (I2C or UART).
- Allow the user to communicate with the Teseo-LIV3F device via the TESEO-SUITE.
- Display on a PC data received by the Teseo-LIV3F device through a serial connection and a serial terminal on a PC.
-

4.4 FWUpdater

FWUpdater is the application to be loaded in order to use the tool (FWUPG.jar) for updating the GNSS Teseo-LIV3F firmware on the X-NUCLEO-GNSS1A1 expansion boards.
The FWUPG tool is available in *Utilities\PC_Software\FirmwareUpdaterTool*.
Plug the X-NUCLEO-GNSS1A1 expansion board on top of a STM32 Nucleo board.
Connect the STM32 Nucleo board to your PC.
Flash the generated binary (also included in Binary folder) on the Nucleo drive.
Run the java tool FWUPG.jar.
From the java GUI, after selecting the right serial port, click **Open** to start a connection with your STM32 Nucleo and X-NUCLEO-GNSS1A1 boards.
If the FW version on the Teseo-LIV3F module is not the latest one, click the **Update FW >>>** button to start the firmware upgrading process.

NOTE: This application can be used only for updating the Teseo-LIV3F device on the X-NUCLEO-GNSS1A1 expansion board. It cannot not be used for the Teseo-VIC2DA on the X-NUCLEO_GNSS2A1.

5 Hardware setup

The following hardware components are needed:

1. One STM32 Nucleo development platform
2. One GNSS expansion board (order code: X-NUCLEO-GNSS1A1 or X-NUCLEO-GNSS2A1)
3. One GPS/GLONASS/Beidou antenna to be connected to the X-NUCLEO-GNSS1A1/X-NUCLEO-GNSS2A1 (bundled with the GNSS expansion board)
4. One USB type A to Mini-B USB cable to connect the STM32 Nucleo to the PC

5.1 X-NUCLEO-GNSS1A1

For the X-NUCLEO-GNSS1A1 proper operations on most Nucleo 64-pins boards, the following jumper settings must be used:

- J2 open
- J3 closed
- J4 closed
- J5 open
- J6 closed
- J7 open
- J8 open
- J9 closed
- J10 open
- J11 closed
- J12 closed
- J13 closed
- J14 closed
- J15 closed

The X-NUCLEO-GNSS1A1 expansion board is easily connected to the STM32 Nucleo development through the Arduino UNO R3 extension connector and can interface with the external STM32 microcontroller on the STM32 Nucleo board either via UART or Inter-Integrated Circuit (I^2C) channels.

If a Nucleo 144 pins is used, to correctly use the UART channel, the following changes in the jumper configuration are required:

- J2 closed

- J3 open
- J4 open
- J5 closed

5.2 X-NUCLEO-GNSS2A1

For the X-NUCLEO-GNSS2A1 proper operations on most Nucleo 64-pins boards, the following jumper settings must be used:

- J11 closed
- J12 closed
- J14 closed
- J15 closed
- J23 1-2
- J24 2-3
- J25 1-2
- J26 2-3
- J27 2-3
- J28 2-3
- J29 1-2
- J30 2-3

The X-NUCLEO-GNSS2A1 expansion board is easily connected to the STM32 Nucleo development through the Arduino UNO R3 extension connector and can interface with the external STM32 microcontroller on the STM32 Nucleo board either via UART or Inter-Integrated Circuit (I^2C) channels.

If a Nucleo 144 pins is used, to correctly use the UART channel, the following change in the jumper configuration is required:

- J28 1-2

Figure 1 STM32 Nucleo 64 pins and X-NUCLEO-GNSS1A1

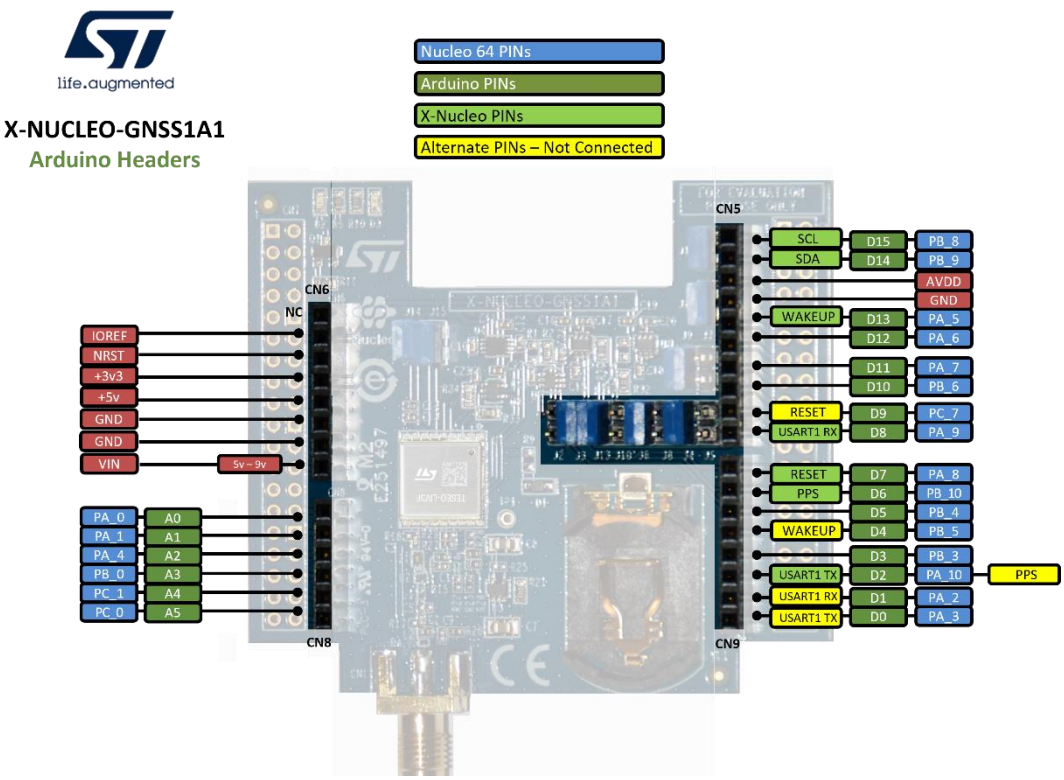

Figure 2 X-NUCLEO-GNSS1A1 pinout

6 Starting a new project with STM32CubeMX

After launching the STM32CubeMX, click on the Start My project from ST Board buttons (anyway the user can choose another option).

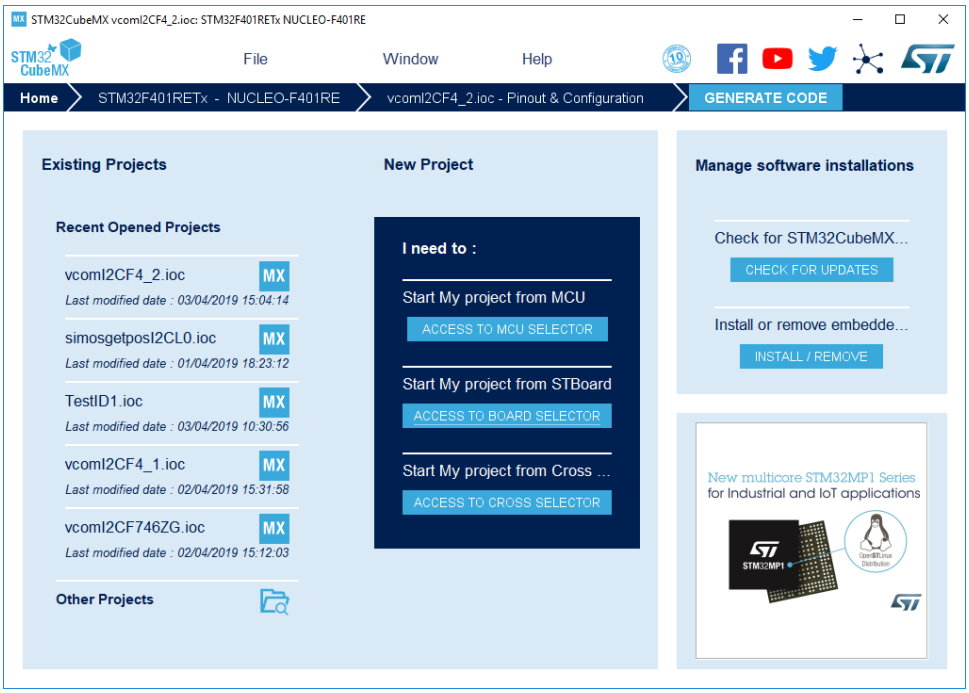

***Figure 3** STM32CubeMX main page*

The MCU/Board selector window will pop up. From this window, the STM32 MCU or platform can be selected. The user can choose between an STM32 Nucleo 64 pins board and an STM32 Nucleo 144 pins board.

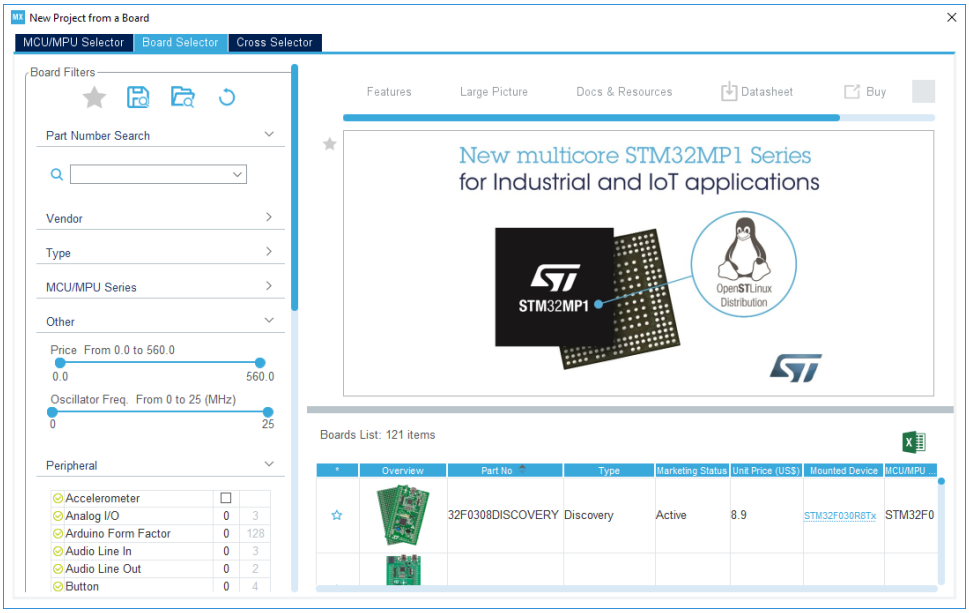

***Figure 4** STM32CubeMX MCU/Board selector*

After selecting the MCU or the Board, the selected STM32 pinout will appear. From this view the user can set up the project, by adding one or more Additional Software and peripherals and configuring the clock.

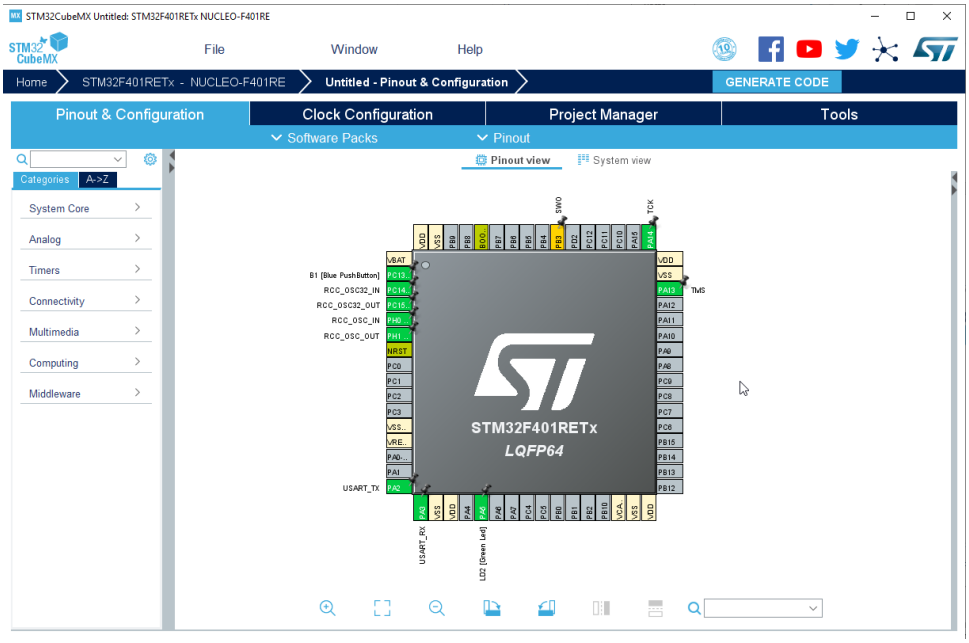

Figure 5 *STM32CubeMX Configuration window*

The GNSS additional software can be added to the project by clicking on the Software Packs tab and then on the Select Components menu item. From the Software Packs Component Selector window, the user can either choose to generate, for the selected MCU/Board, one of the enclosed sample applications or a new project. In this latter case, the user must just implement the main application logic without bothering with the pinout and peripherals configuration code, which will be automatically generated by STM32CubeMX.
From the Software Packs Component Selector window, select one of the four applications available and click on status to solve dependencies.
Notice that each sample application can be configured for using alternatively either the X-NUCLEO-GNSS1A1 or a custom board with the Teseo-LIV3F device.
In the first case the *Board Extension GNSS / GNSS1A1* component must be selected. In the second case, instead, the *Board Support STM32Cube_Custom_BSP_Drivers / Custom / GNSS* component must be selected.

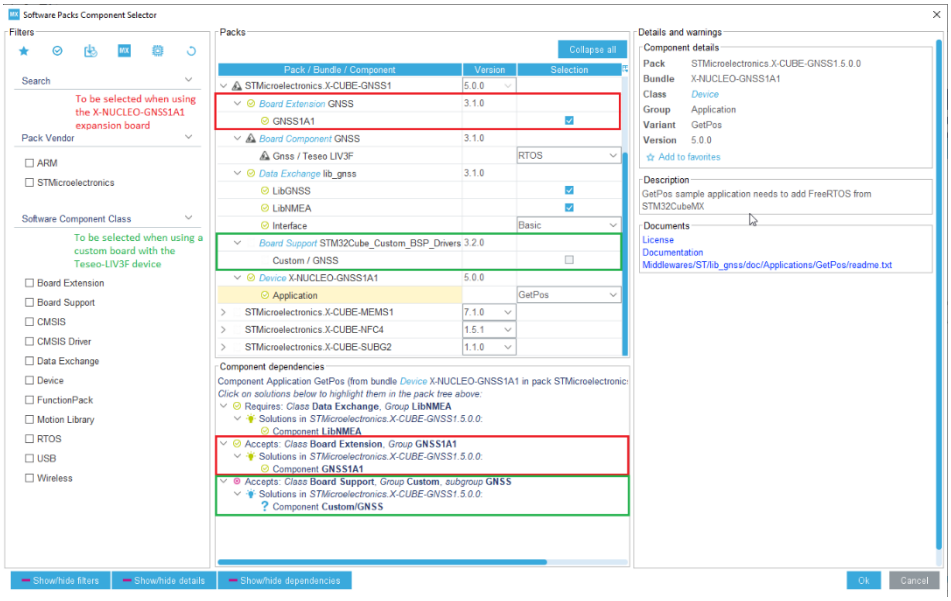

Figure 6 *STM32CubeMX component selection for X-NUCLEO-GNSS1A1 and custom board*

Clicking on the *Status* icon, the user can see the condition to be satisfied for each Pack/Bundle. Notice that the FWUpdater application works only using the UART channel.

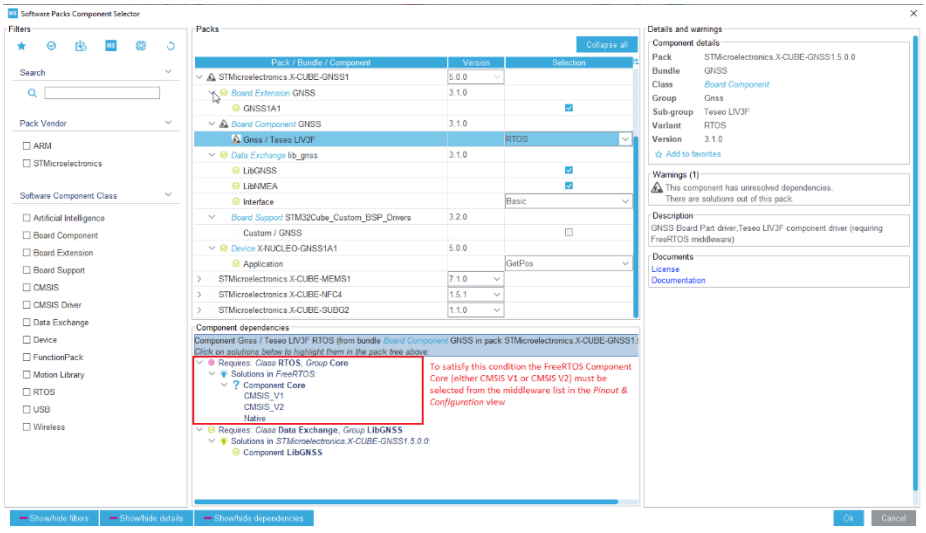

Figure 7 *STM32CubeMX Software Packs Component Selector window*

The figure above shows the Software Packs Component Selector window.
In case a component requires an RTOS middleware (either FreeRTOS or AZRTOS ThreadX), this must be selected from the Middleware list in the Pinout & Configuration view (see picture below).

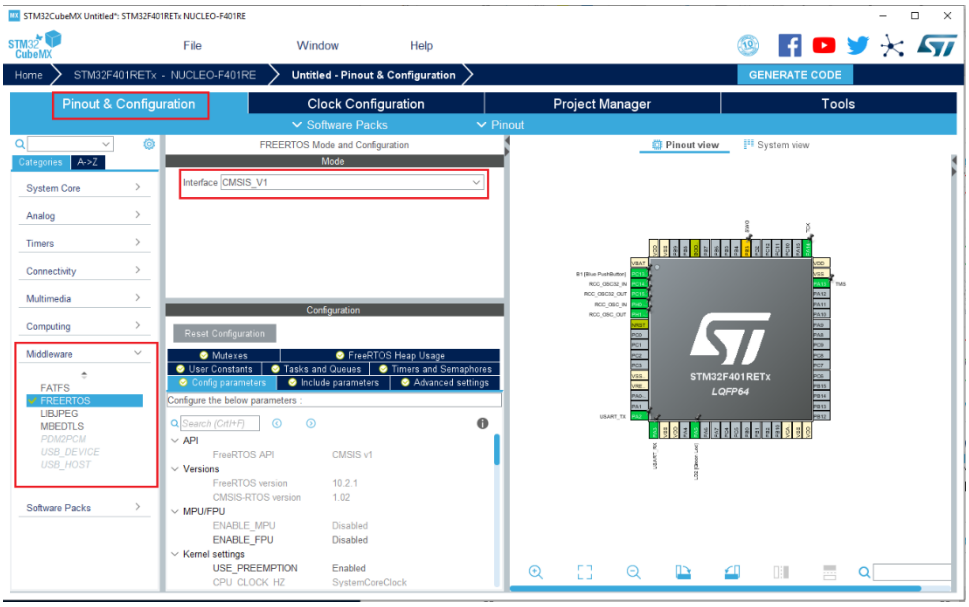

***Figure 8** Enabling the FreeRTOS middleware*

7 Configuration steps

7.1 Use of the expansion sw with I2C

This section outlines how to configure CubeMX when the use of the sample applications is required. With such setup, all the components of the expansion software package, including applications, will be properly configured.

In the Pinout & Configuration tab:

- Click on Pinout → Clear Pinouts to clean the current state and avoid possible conflicts (only for Nucleo 144)
- Set the state of PA5 to GPIO_Output
- Set the state of PA8 to GPIO_Output (Nucleo 64 only)
- Set the state of PF13 to GPIO_Output (Nucleo 144 only)
- From Middleware list, enable either the FreeRTOS or the AZRTOS ThreadX (for GetPos and VirtualCOMPort applications only)
- From the pinout scheme, enable I2C1_SDA on PB9
- From the pinout scheme, enable I2C1_SCL on PB8
- From Connectivity list, enable the I2C1
- If not already enabled, from Connectivity list, enable the USART2 on Asynchronous mode (most of Nucleo 64)
- If not already enabled, from Connectivity list, enable the USART3 on Asynchronous mode (most of Nucleo 144) *

In summary, check the settings as reported in the following table:

Nucleo 64 pins	Mode	Nucleo 144 pins	Mode
PA5	GPIO_Output	PA5	GPIO_Output
PA8	GPIO_Output	PF13	GPIO_Output
PB9	I2C1_SDA	PB9	I2C1_SDA

PB8	I2C1_SCL	PB8	I2C1_SCL
PA2	USART2_TX	PD9	USART3_TX *
PA3	USART2_RX	PD8	USART3_RX *

Table 1 *Pinout settings (I2C)*

In the Pinout & Configuration tab check the settings as reported in the picture below:

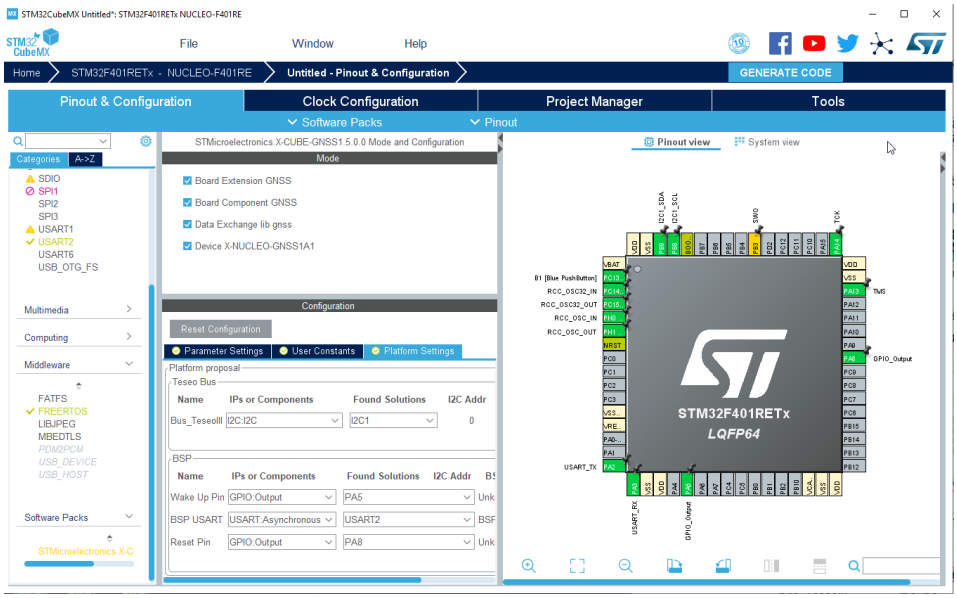

Figure 9 *STM32CubeMX Pinout & Configuration tab (I2C)*

The Platform settings are summarized in the following table:

Name	IPS or Component	Found solutions	BSP Api
Bus_TeseoIII	I2C:I2C	I2C1	BSP_BUS_DRIVER
BSP USART	UART:Asynchronous	USART2 (USART3 *)	BSP_Common_Driver
BSP BUTTON (only VirtualCOMPort)	GPIO:EXTI	PC13	HAL_EXTI_DRIVER
Reset pin	GPIO:Output	PA8 (PF13)	
Wake up Pin	GPIO:Output	PA5	

Table 2 *Platform settings (I2C)*

In the Parameter Settings the user can enable/disable four more features as shown in the picture below (Odometer, Geofence and Datalog features are valid for GetPos application only):

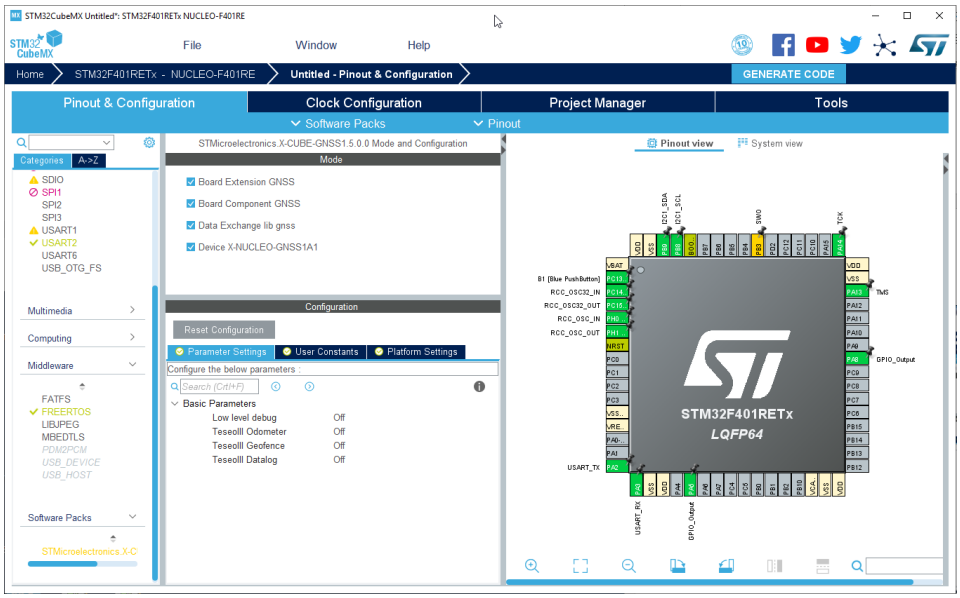

***Figure 10** STM32CubeMX Parameter Settings*

It the FreeRTOS component is used, check the following parameter settings:

- MINIMAL_STACK_SIZE ≥ 128 (in case of CMSIS_V1 interface) or MINIMAL_STACK_SIZE ≥ 256 (in case of CMSIS_V2 interface)
- TOTAL_HEAP_SIZE ≥ 15360 Bytes.

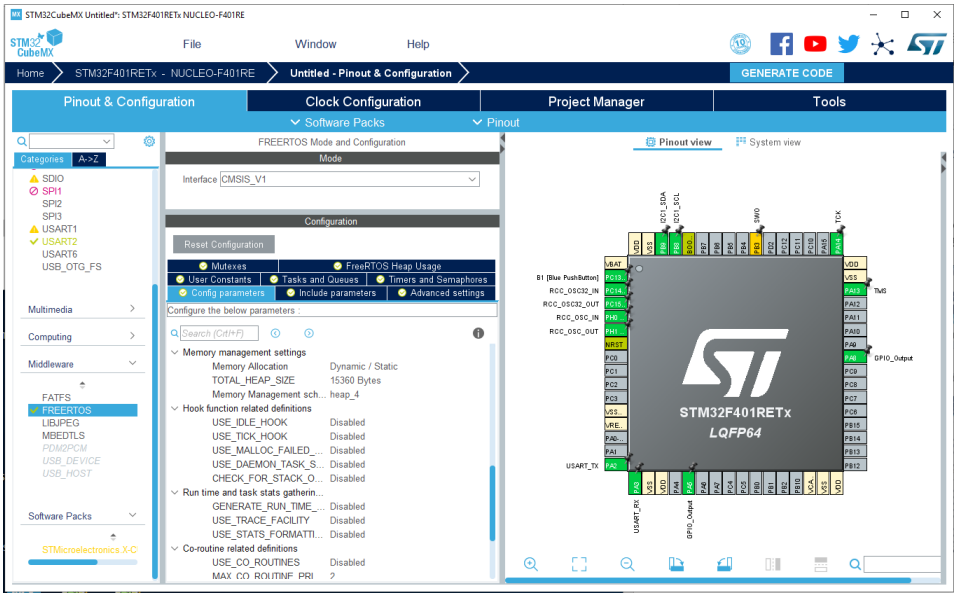

***Figure 11** STM32CubeMX FreeRTOS settings*

Moving to System View and choosing NVIC under System Core list, check the Interrupt setting reported in the picture below (EXTI line [15 10] for Button interrupt events enabled only for VirtualCOMPort application):

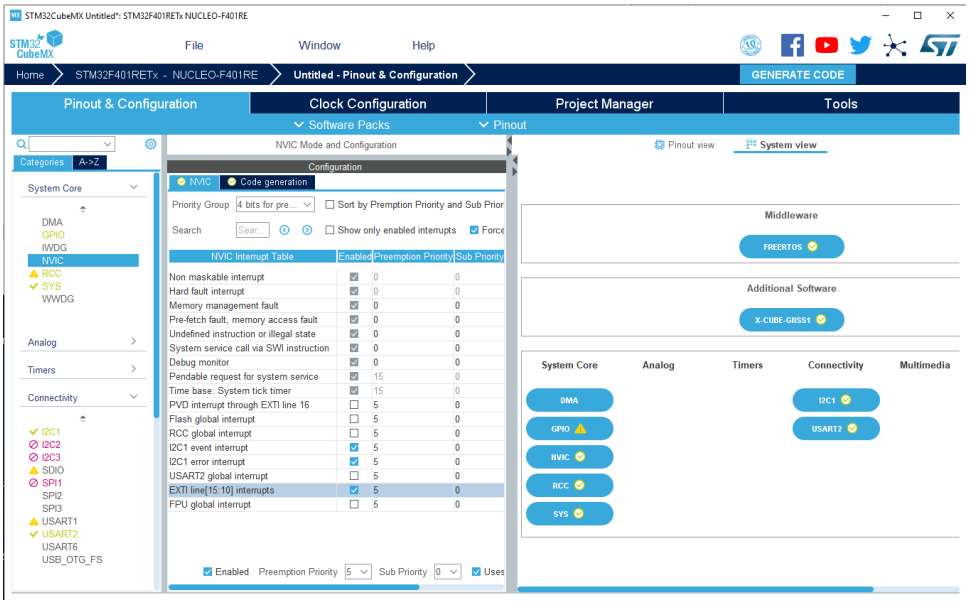

Figure 12 *STM32CubeMX NVIC settings*

In the same NVIC Interrupt Table, showed in the picture above, be sure that the Preemption Priority values of the I2C1 and of the EXTI line (and of the all interrupt handlers calling RTOS functions) are lower than the highest syscall interrupt priority.
Before generating the code, go to the System Core list, select the SYS and change the SYS Timebase Source from SysTick to TIM1.
Then go to the Project Manager tab / Advanced Settings and in the Register CallBack section enable the I2C register callback.

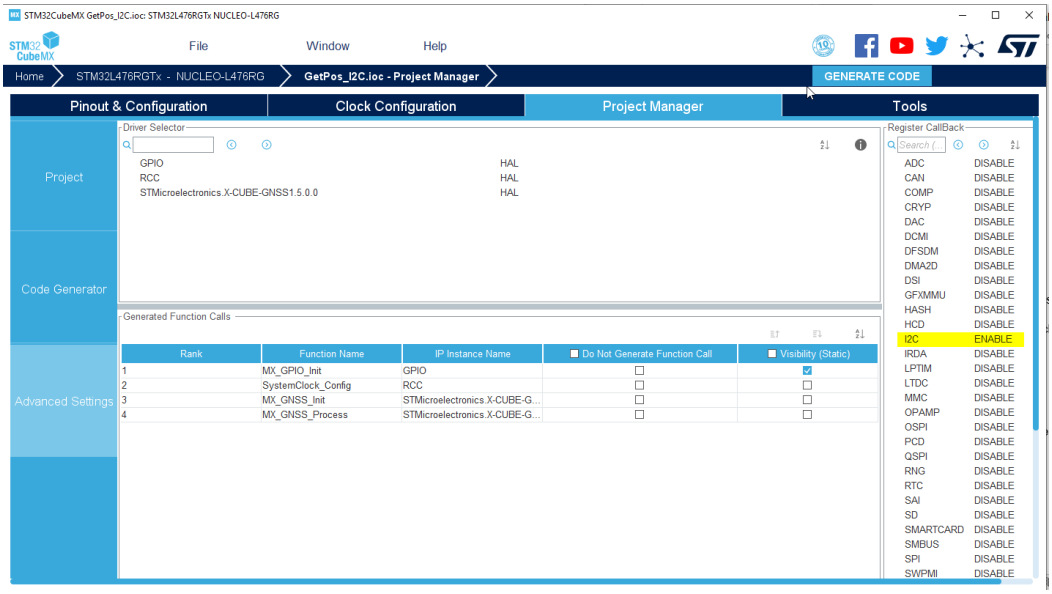

Figure 13 *Enabling the I2C register callback*

Now you can set your project preferences and click the GENERATE CODE button to proceed with the code generation.

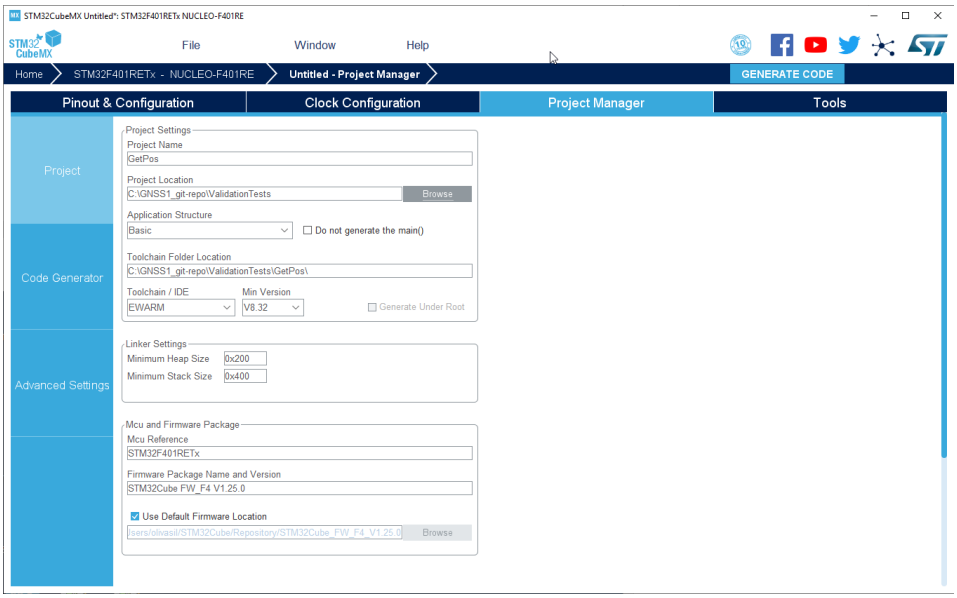

Figure 14 *STM32CubeMX code generation step*

7.2 Use of the expansion sw with UART

This section outlines how to configure CubeMX when the use of the sample applications is required. With such setup, all the components of the expansion software package, including applications, will be properly configured. If a Nucleo 144 pins is used, to set the USART connection correctly, in the X-NUCLEO-GNSS1A1 board you need to close the jumpers J2 and J5 and open the jumpers J4, J3.
In the Pinout & Configuration tab:

- Click on Pinout → Clear Pinouts to clean the current state and avoid possible conflicts (only for Nucleo 144)
- Set the state of PA5 to GPIO_Output
- Set the state of PA8 to GPIO_Output (Nucleo 64 only)
- Set the state of PF13 to GPIO_Output (Nucleo 144 only)
- From Middleware list, enable FREERTOS (for GetPos and VirtualCOMPort applications only)
- From Connectivity list, enable the USART1 on Asynchronous mode (most of Nucleo 64)
- From Connectivity list, enable the USART6 on Asynchronous mode (most of Nucleo 144)
- If not already enabled, from Connectivity list, enable the USART2 on Asynchronous mode (most of Nucleo 64)
- If not already enabled, from Connectivity list, enable the USART3 on Asynchronous mode (most of Nucleo 144) *

In summary, check the settings as reported in the following table:

Nucleo 64 pins	Mode	Nucleo 144 pins	Mode
PA5	GPIO_Output	PA5	GPIO_Output
PA8	GPIO_Output	PF13	GPIO_Output
PA9	USART1_TX	PG14	USART6_TX
PA10	USART1_RX	PG9	USART6_RX
PA2	USART2_TX	PD9	USART3_TX *
PA3	USART2_RX	PD8	USART3_RX *

Table 3 *Pinout settings (UART)*

In the Pinout & Configuration tab check the settings as reported in the picture below:

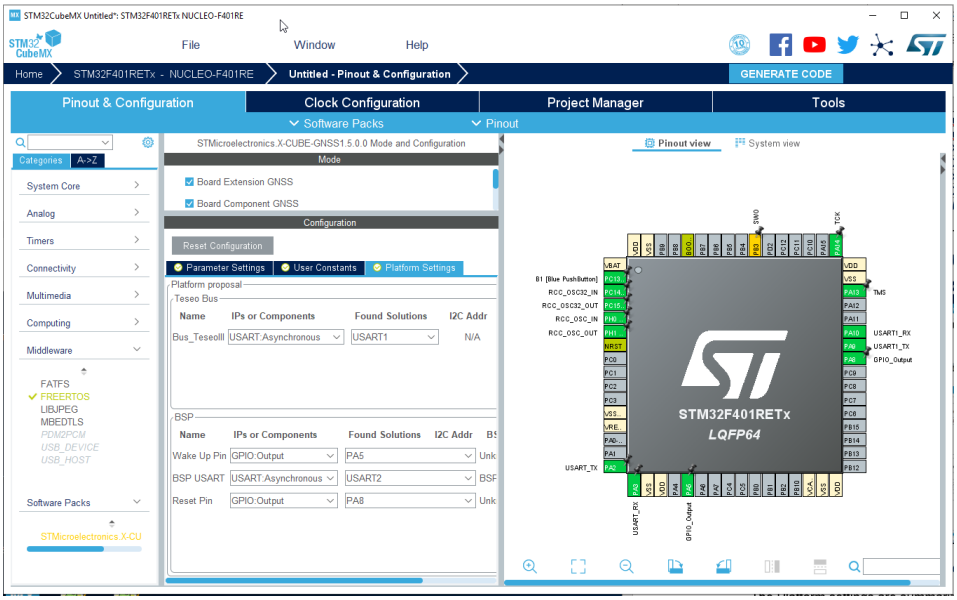

Figure 15 *STM32CubeMX Pinout & Configuration tab (UART)*

The Platform settings are summarized in the following table:

Name	IPS or Component	Found solutions	BSP Api
Bus_TeseoIII	UART:Asynchronous	USART1 (USART6)	BSP_BUS_DRIVER
BSP USART	UART:Asynchronous	USART2 (USART3 *)	BSP_Common_Driver
BSP BUTTON (only VirtualCOMPort)	GPIO:EXTI	PC13	HAL_EXTI_DRIVER
Reset pin	GPIO:Output	PA8 (PF13)	
Wake up Pin	GPIO:Output	PA5	

Table 4 *Platform settings (UART)*

Notice that the Baud Rate for USART1 (USART6), connecting the Teseo-LIV3F and the STM32 MCU, should be set to 9600 as reported in picture below for all applications, apart from the FWUpdater which requires 115200.

NOTE: if the Teseo-VIC3DA device and the X-NUCLEO-GNSS2A1 expansion board are used, the Baud Rate for USART1 (USART6) must be always set to 115200.

NOTE: for the GetPos application, it is suggested to set, from the software pack Parameter Settings tab, the GNSS device parameter according to the targeting device.

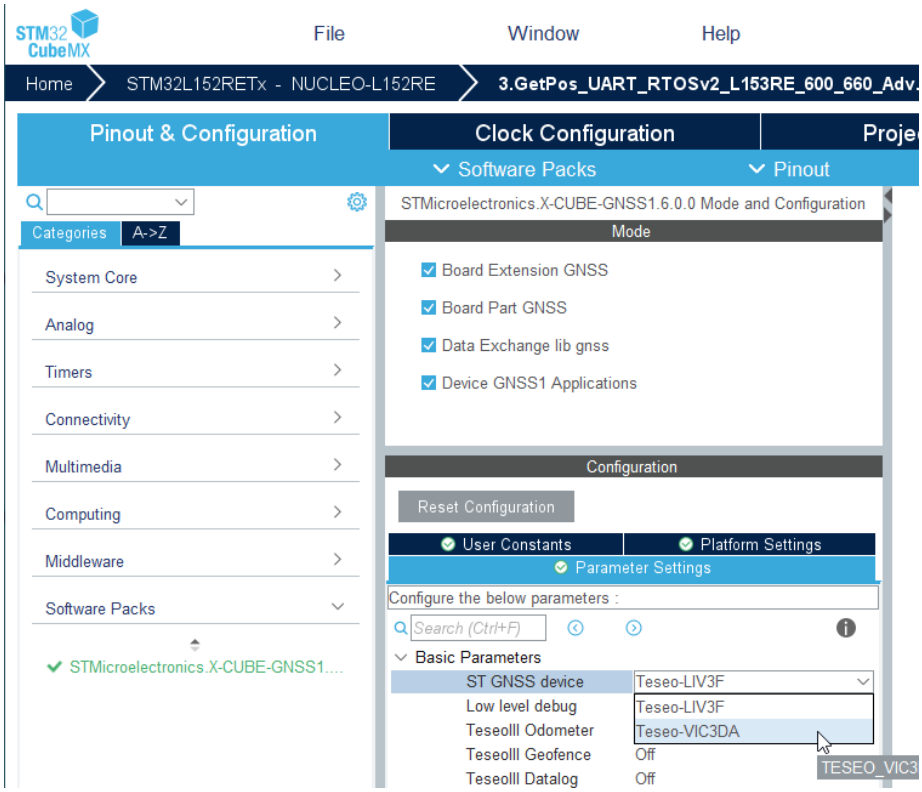

***Figure 16** X-CUBE-GNSS1 Parameter Settings view*

From Connectivity list, choose USART1 (USART6) then update the Baud Rate in the Parameter Settings tab:

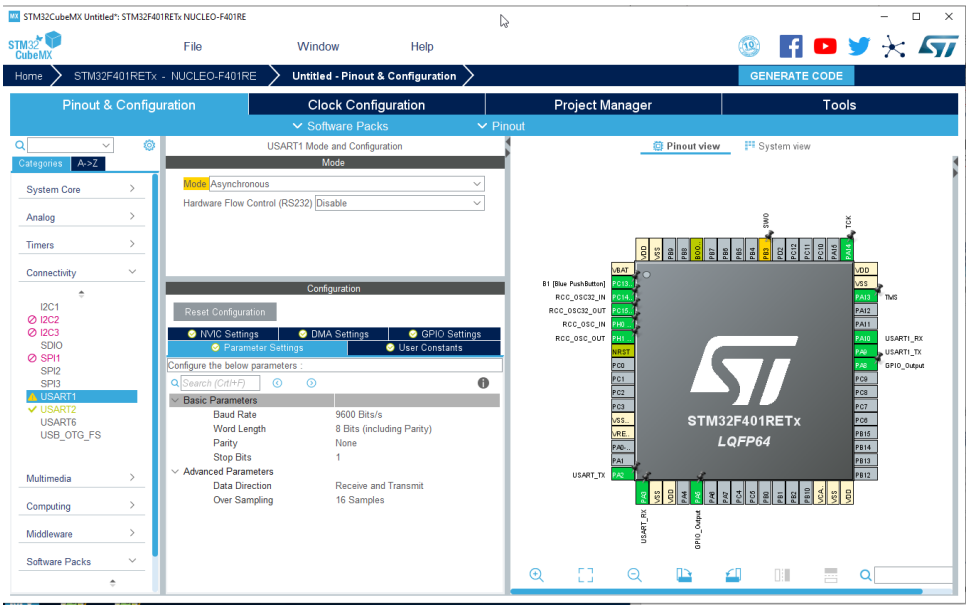

***Figure 17** STM32CubeMX Parameter Settings (UART)*

For other pin, peripheral and middleware settings and the code generation, follow the steps indicated in the previous section (I2C case).

* NOTE: For some Nucleo-144 pins it may happens that another USART must be used for the BSP Common Driver. For instance, for the NUCLEO-L496ZG, NUCLEO-L496ZG-P, NUCLEO-L4R5ZI, NUCLEO-L4R5ZI-P, NUCLEO-L4A6ZG and NUCLEO-U575ZI-Q the LPUART1 must be used with PG7 for the LPUART1_TX and PG8 for the LPUART1_RX. Information on the peripheral to be used is reported in the STM32 Nucleo board user manuals available on st.com.

8 Known limitations and workaround

- For the GetPos application to properly work 15KB RAM is the minimum requirement.
- The following Preprocessor macros are required:
  - **USE_HAL_I2C_REGISTER_CALLBACKS=1**
    (GetPos, VirtualCOMPort and SimOSGetPos applications, I2C channel option)
  - **USE_HAL_UART_REGISTER_CALLBACKS=1**
    (GetPos, VirtualCOMPort and SimOSGetPos applications, UART channel option)

  They can be set through the STM32CubeMX UI, as described in **Figure 13**.
  Otherwise, after generating the code, edit the file stm32YYxx_hal_conf.h (in the application Inc folder) setting the I2C/UARTregister callback macro to 1.
- For STM32CubeIDE projects, the floating point for the printf function must be enabled.
  From the menu bar select*: Project* > *Properties* > *C/C++ Build* > *Settings* > *Tool Settings* and then check *Use float with printf from newlib-nano*.
- The FWUpdater application works only using the UART channel.
- For MDK-ARM projects, the Use MicroLib option in the project settings must be manually set.
- On dual-core STM32 series this expansion software can be used on both cores but exclusively.

9 References

[1] UM2334 – User Manual - *Getting started with the X-CUBE-GNSS1 Global Navigation Satellite System software expansion for STM32Cube.*
[2] UM2327 – User Manual - *Getting started with the X-NUCLEO-GNSS1A1 expansion board based on Teseo-LIV3F tiny GNSS module for STM32 Nucleo.*

10 Revision History

	Version	Changes
02-Apr-2019	1	Initial release.
16-Sep-2019	2	Modify sections: 7, 8.
28-Jan-2020	3	Update section 6.
18-May-2020	4	Update screenshots.
13-Dec-2021	5	Add references to Azure RTOS ThreadX. Add picture with X-NUCLEO-GNSS1A1 pinout.
08-Jun-2022	6	Add references to Teseo-VIC3DA device and X-NUCLEO-GNSS2A1 expansion board.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders.
ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.
Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.
No license, express or implied, to any intellectual property right is granted by ST herein.
Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.
ST and the ST logo are trademarks of ST. 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.
Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2020 STMicroelectronics – All rights reserved