Getting started with the STM32Cube function pack for IoT node with BLE connectivity and time-of-flight sensors

Introduction

The FP-SNS-FLIGHT1 is an STM32Cube function pack, which lets your IoT node connect to a smartphone via BLE and uses a suitable Android™ or iOS™ application like the STBLESensor app to view real-time object distance data read by the Time-of-Flight sensor.

The package also enables advanced functions, such as presence detection inside a fixed range distance.

This package, together with the suggested combination of the STM32 and ST devices, can be used to develop wearable applications or smart thing applications in general.

The software runs on the STM32 microcontroller and includes all the necessary drivers to recognize the devices on the STM32 Nucleo development board.

The software is available also on GitHub, where the users can signal bugs and propose new ideas through [Issues] and [Pull Requests] tabs.

1 Acronyms and abbreviations

Table 1. Acronyms and abbreviations

Acronym	Description
BLE	Bluetooth low energy
ToF	Time-of-Flight

2 FP-SNS-FLIGHT1 software description

2.1 Overview

The key features of the FP-SNS-FLIGHT1 package are:

- Complete firmware to develop an IoT node with BLE connectivity, and Time-of-Flight sensors
- Compatible with STBLESensor application for Android/iOS to perform distance data reading and firmware update (FOTA)
- Multitarget ranging sensor application based on the VL53L3CX Time-of-Flight (ToF) sensor
- Sample implementation available for X-NUCLEO-53L3A2 (or VL53L3CX-SATEL) and X-NUCLEO-BNRG2A1 connected to a NUCLEO-F401RE, NUCLEO-L476RG or NUCLEO-U575ZI-Q
- Compatible with STM32CubeMX, can be downloaded from and installed directly into STM32CubeMX
- Easy portability across different MCU families, thanks to STM32Cube
- Free, user-friendly license terms

This software creates the following Bluetooth services:

- the first service exposes all the hardware features and contains the following characteristics:
  - LED on/off
  - object distances (up to four objects) or presence
- the second is the Console service that includes two characteristics:
  - stdin/stdout with bidirectional communication between client and server
  - stderr for a mono-directional channel from the STM32 Nucleo board to an Android/iOS device
- the last service is used for configuration purpose

This software gathers the LED status and distance for the VL53L3CX device for STM32 Nucleo expansion boards with X-NUCLEO-53L3A2 or VL53L3CX-SATEL.

This package is compatible with the STBLESensor Android/iOS (version 5.2.0/5.2.0 or higher) application, available at the respective Google Play/iTunes stores, which can be used to display information sent via the BLE.

The STBLESensor application allows Over-The-Air firmware updates.

2.2 Architecture

This software is based on the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller. The package extends STM32Cube by providing a board support package (BSP) for the BlueNRG-2 network processor (embedded in the BlueNRG-M2SP module), Time-of-Flight ranging expansion board and middleware components for communication with other Bluetooth devices.

BlueNRG is a very low-power Bluetooth low energy (BLE) single-mode network processor.

The drivers abstract low-level hardware details and allow the middleware components and applications to access at the Time-of-Flight sensor in a hardware-independent manner.

The package includes a sample application that the developer can use to start experimenting with the code.

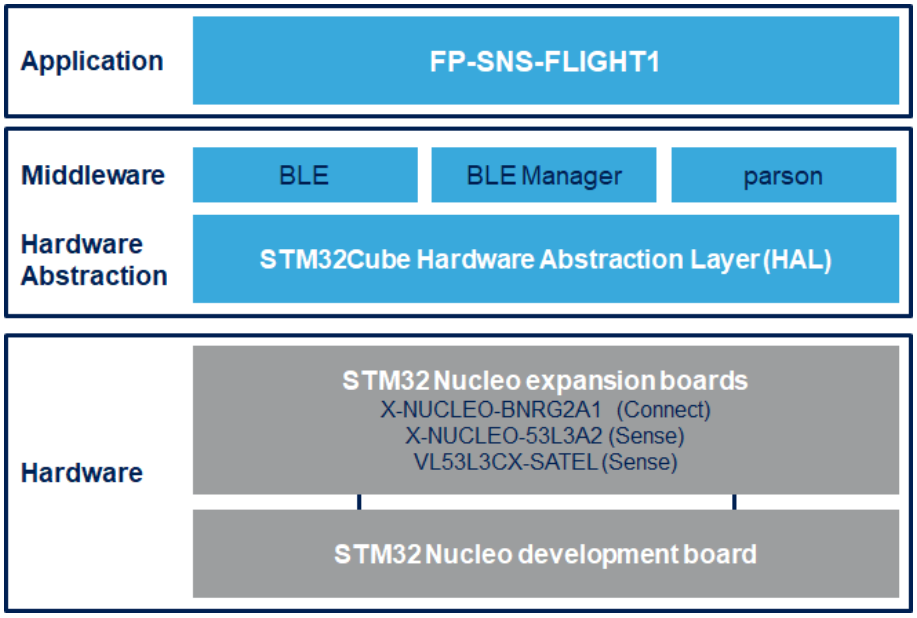
The sample application transmits via BLE the values read from the Time-of-Flight sensors to a Bluetooth low energy enabled device such as an Android™ or iOS™ based smartphone.

You can use the STBLESensor Android/iOS application, from the respective Google Play™ and iTunes™ stores, to display the values read from the Time-of-Flight sensors.

The software layers used by the application software to access and use the sensor expansion boards are:

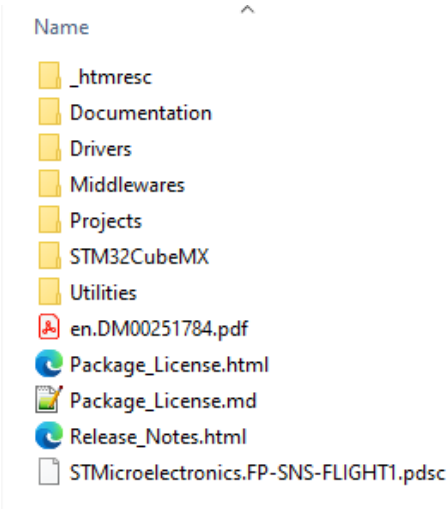
- **STM32Cube HAL driver layer**: provides a simple, generic, multi-instance set of APIs (application programming interfaces) to interact with the upper layers (application, libraries, and stacks). It is composed of generic and extension APIs and is directly built around a generic architecture and allows the layers built on top of it, such as the middleware layer, to implement their functions without depending on the specific hardware configuration for a given microcontroller unit (MCU). This structure improves library code reusability and guarantees an easy portability across other devices.
- **Board Support Package (BSP) layer**: provides support for the peripherals (excluding the MCU) on the STM32 Nucleo board through the board support package (BSP). The BSP is a limited set of APIs, which provides a programming interface for certain board-specific peripherals like the LED, the user button, etc.

Figure 1. **FP-SNS-FLIGHT1 software architecture**



2.3 Folder structure

Figure 2. **FP-SNS-FLIGHT1 package folder structure**



The following folders are included in the software package:

- **Documentation**: contains a compiled HTML file generated from the source code, detailing the software components and APIs.
- **Drivers**: contains the HAL drivers, the board specific drivers for each supported board or hardware platform, including the on-board components and the CMSIS vendor-independent hardware abstraction layer for the Cortex-M processor series.
- **Middlewares**: contains libraries and protocols for BlueNRG-2 Bluetooth low energy module, the Meta Data Manager, BLE Manager and json.
- **Projects**: contains a sample application used to transmit the time-of-flight sensor data via the Bluetooth low energy protocol provided for the NUCLEO-F401RE/NUCLEO-L476RG/NUCLEO-U575ZI-Q through the IAR Embedded Workbench for ARM, RealView Microcontroller Development Kit (MDK-ARM) and STM32CubeIDE development environments.
- **Utilities**: contains the boot loader binary ready to be flashed for the NUCLEO-F401RE/NUCLEO-L476RG development boards.

2.4 Flash management

For NUCLEO-F401RE, NUCLEO-L476RG and NUCLEO-U575ZI-Q the FP-SNS-FLIGHT1 uses the Flash memory to:

1. Save firmware information (node name, firmware ID, etc.) in the dedicate flash address.
2. Allow the Firmware-Over-The-Air update.

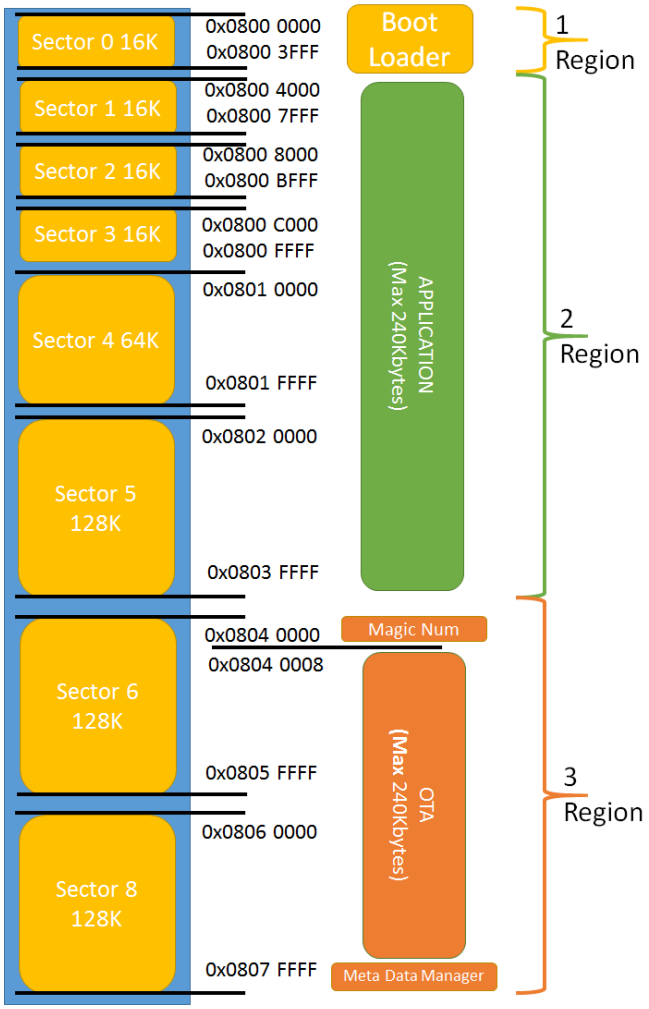
For NUCLEO-F401RE and NUCLEO-L476RG, to enable these features, the whole Flash is divided into the following distinct regions:

1. Contains a custom boot loader.
2. Contains the FP-SNS-FLIGHT1 firmware.
3. Used to store FOTA before the update.

The same Flash management applies to both the STM32F401RE and the STM32L476RG boards, even if they have different cache sizes (respectively 512 and 1024 Kbytes), and two different configurations. For further info on Flash configuration please refer to:

- RM0368 Reference manual STM32F401xB/C and STM32F401xD/E advanced ARM®-based 32-bit MCUs.
- RM0351 Reference manual STM32L4x6 advanced ARM®-based 32-bit MCUs.

Figure 3. FP-SNS-FLIGHT1 Flash structure



For NUCLEO-U575ZI-Q the dual bank flash features to allow Firmware-Over-the-Air update of a running program is used.

For further info on Flash configuration please refer to:

- RM0456 STM32U5 Series Arm®-based 32-bit MCUs.

2.5 The Boot process

2.5.1 NUCLEO-F401RE and NUCLEO-L476RG Boot Process

The FP-SNS-FLIGHT1 cannot be flashed at the beginning of the Flash memory (address 0x08000000); therefore it is compiled to run from the beginning of the second Flash region (0x08004000).

To enable this behavior, the vector table offset has been set in Src/system_stm32f4xx.c (for STM32F401) and Src/system_stm32l4xx.c (for STM32L476) thus: #define VECT_TAB_OFFSET 0x4000.

The linker script has also been changed.

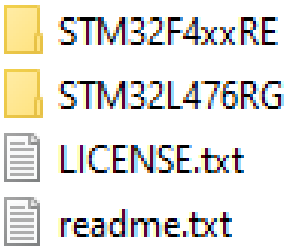
For example, the linker script for SP-SNS-FLIGHT1 running on STM32F401RE and compiled using IAR Embedded Workbench for ARM is:

```
define symbol __ICFEDIT_intvec_start__ = 0x08004000;
      /*-Memory Regions-*/
      define symbol __ICFEDIT_region_ROM_start__ = 0x08004000;
      define symbol __ICFEDIT_region_ROM_end__ = 0x0803FFFF;
      define symbol __ICFEDIT_region_RAM_start__ = 0x20000000;
      define symbol __ICFEDIT_region_RAM_end__ = 0x20017FFF;
      /*-Sizes-*/
      define symbol __ICFEDIT_size_cstack__ = 0x8000;
      define symbol __ICFEDIT_size_heap__ = 0x800;
```

Using the above linker script, the maximum usable code size is fixed at 240 KB.

You must flash the appropriate bootloader binary for STM32F401RE or STM32L476RG , found in the Utilities\BootLoader folder, to the first Flash region (address 0x08000000).

Figure 4. Bootloader utility content



On any board reset, the board starts executing the boot loader.

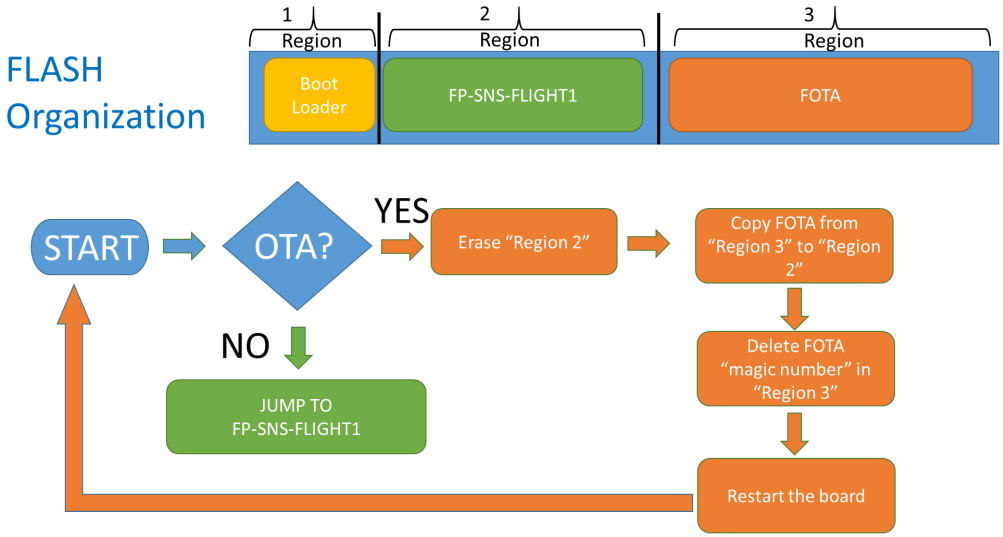
The boot loader checks whether a FOTA is available: the check is based on the presence of a "magic number" at the beginning of the third Flash region.

If there is a FOTA available, the bootloader:

1. erases the second Flash region (containing the FP-SNS-FLIGHT1 firmware)
2. replaces its content with the FOTA
3. erases the “magic number” used to check FOTA presence
4. restarts the board

If there is no FOTA available, the boot loader jumps directly to the FP-SNS-FLIGHT1 firmware.

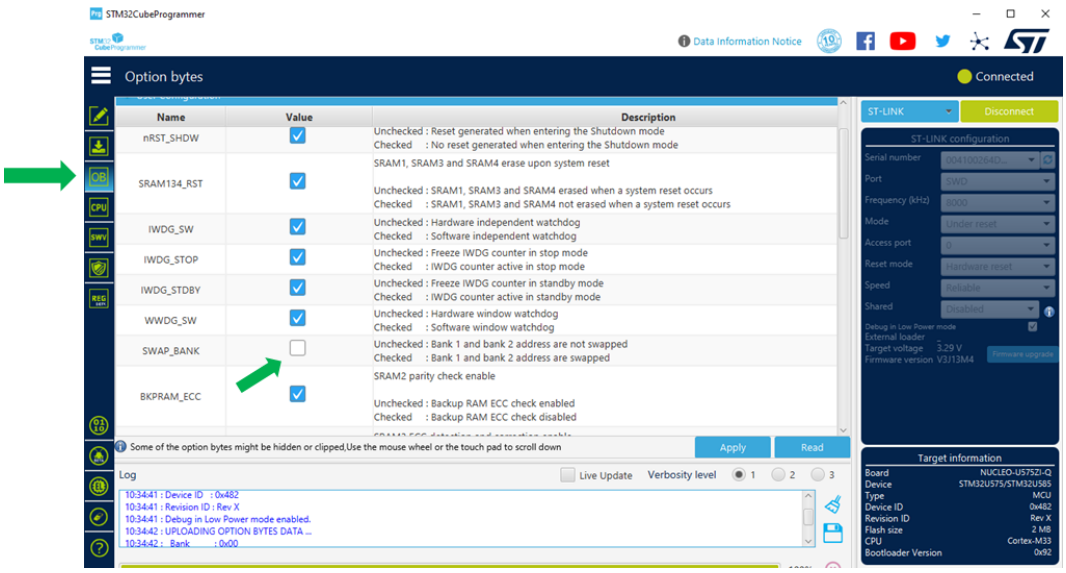
Figure 5. FP-SNS-FLIGHT1 Flash structure



2.5.2 NUCLEO-U575ZI-Q boot process

The FLIGHT1 firmware uses the dual bank flash features to allow firmware over-the-air update of a running program without using the bootloader. For the first installation, after the full flash erases (suggest procedure), use the STM32CubeProgrammer to set STM32 MCU user byte settings to use the bank 1 for flash the firmware and starts the application, as shown in the figure below:

Figure 6. STM32CubeProgrammer option bytes



After the boot (using the reset button on the board) the FLIGHT1 firmware receives the new firmware from the STBLESensor application, saves it on one flash bank (either bank1 or bank2) and performs a reboot executing the new code saved on the other flash bank.

A program related to a specific region can run in that region only.

The FLIGHT1 application, however, can swap among different flash banks and each program can run in any flash memory bank. As the application does not erase the previous version of the code after the update, it allows the rollback to the previous program in case of a hardware fault.

Note: The FOTA update procedure are not available for CUSTOM application

2.6 The installation process for NUCLEO-F401RE or NUCLEO-L476RG board

The package Binary directory contains an image (in .bin format) for each platform (NUCLEO-F401RE, NUCLEO-L476RG, including:

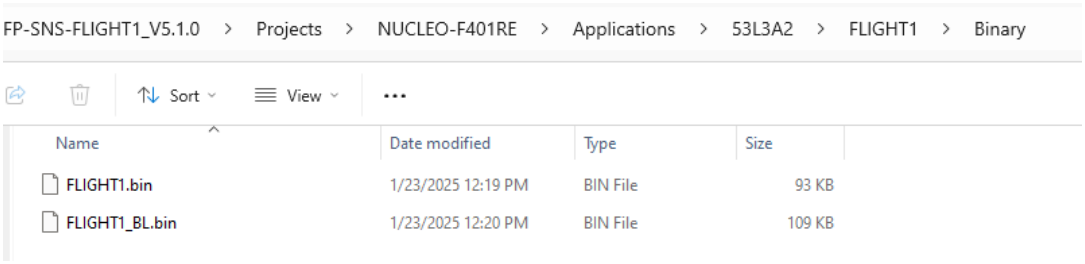
- precompiled FLIGHT1 firmware to be flashed with ST-LINK to the correct memory address (0x08004000) of a supported STM32 Nucleo development board

Note: This precompiled binary is compatible with the FOTA update procedure.

- precompiled FLIGHT1 plus BootLoader firmware to be directly flashed to a supported STM32 Nucleo development board with the ST-LINK or via drag and drop (STM32 Nucleo boards only)

Note: This precompiled binary is not compatible with the FOTA update procedure.

Figure 7. Binary folder content



To flash modified FLIGHT1 firmware, simply flash the compiled FP-SNS-FLIGHT1 firmware to the correct address (0x08004000).

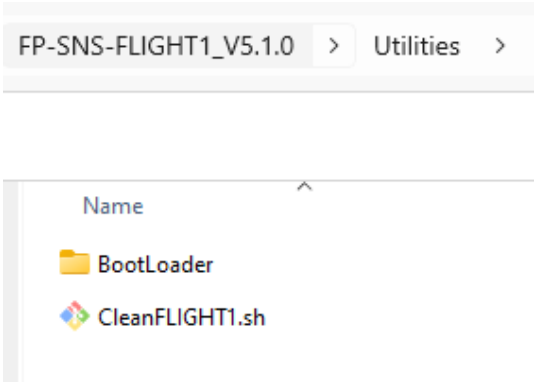
In the folder Utilities there is a scripts *.sh that makes the following operations:

- Full Flash Erase.
- Load the BootLoader on the rigth flash region.
- Load the Program (after the compilation) on the rigth flash region (This could be used for a FOTA).
- Dump back one single binary that contain BootLoader+Program that could be flashed at the flash beginning (address 0x08000000) (This COULD BE NOT used for FOTA).
- Reset the board.

Before to execute the *.sh script, it is necessary to edit it to set the installation path for STM32CubeProgrammer.

BootLoaderPath/<BootLoader file name> and BinaryPath as input are required when execute *.sh script

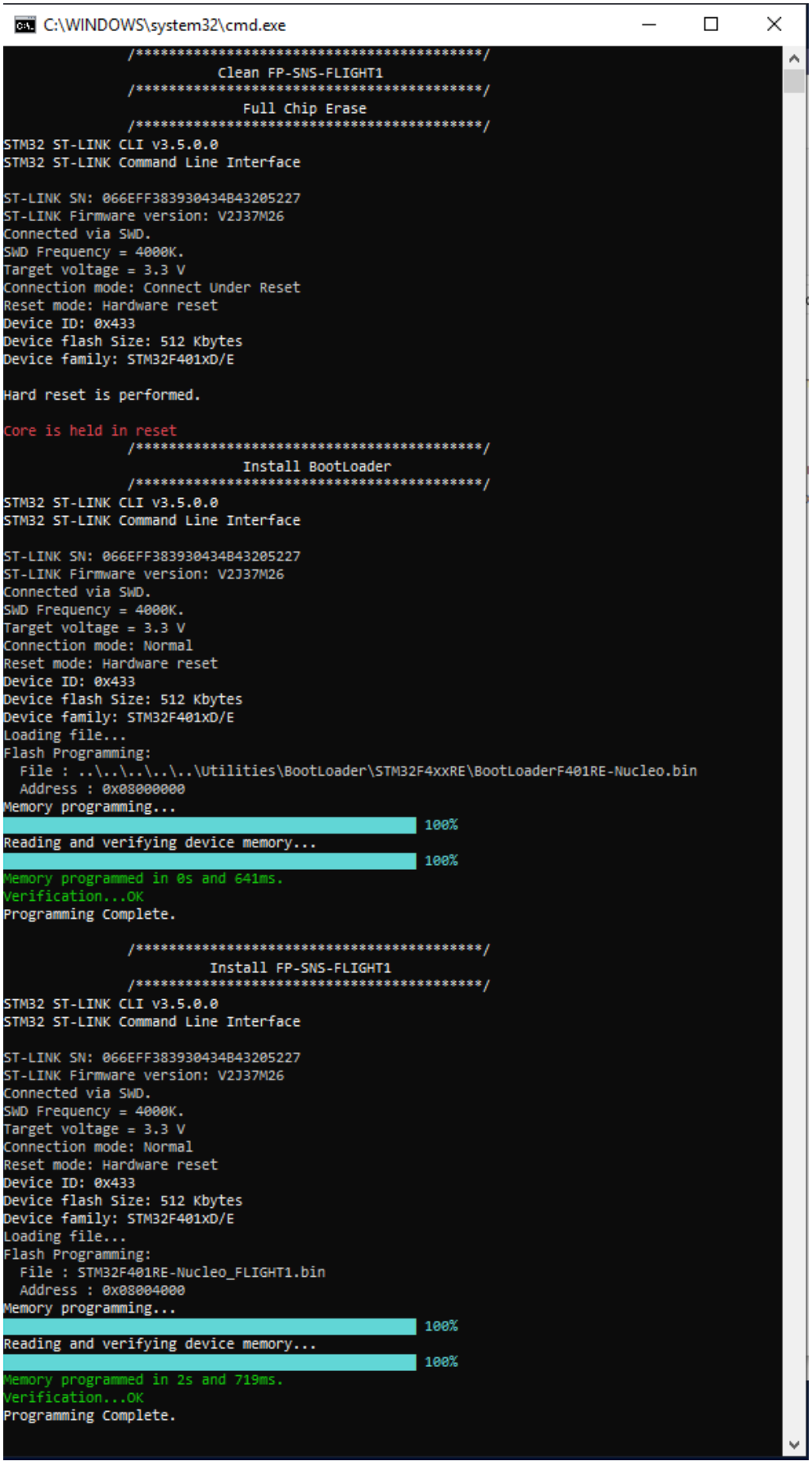
Figure 8. Utilities folder content



This script:

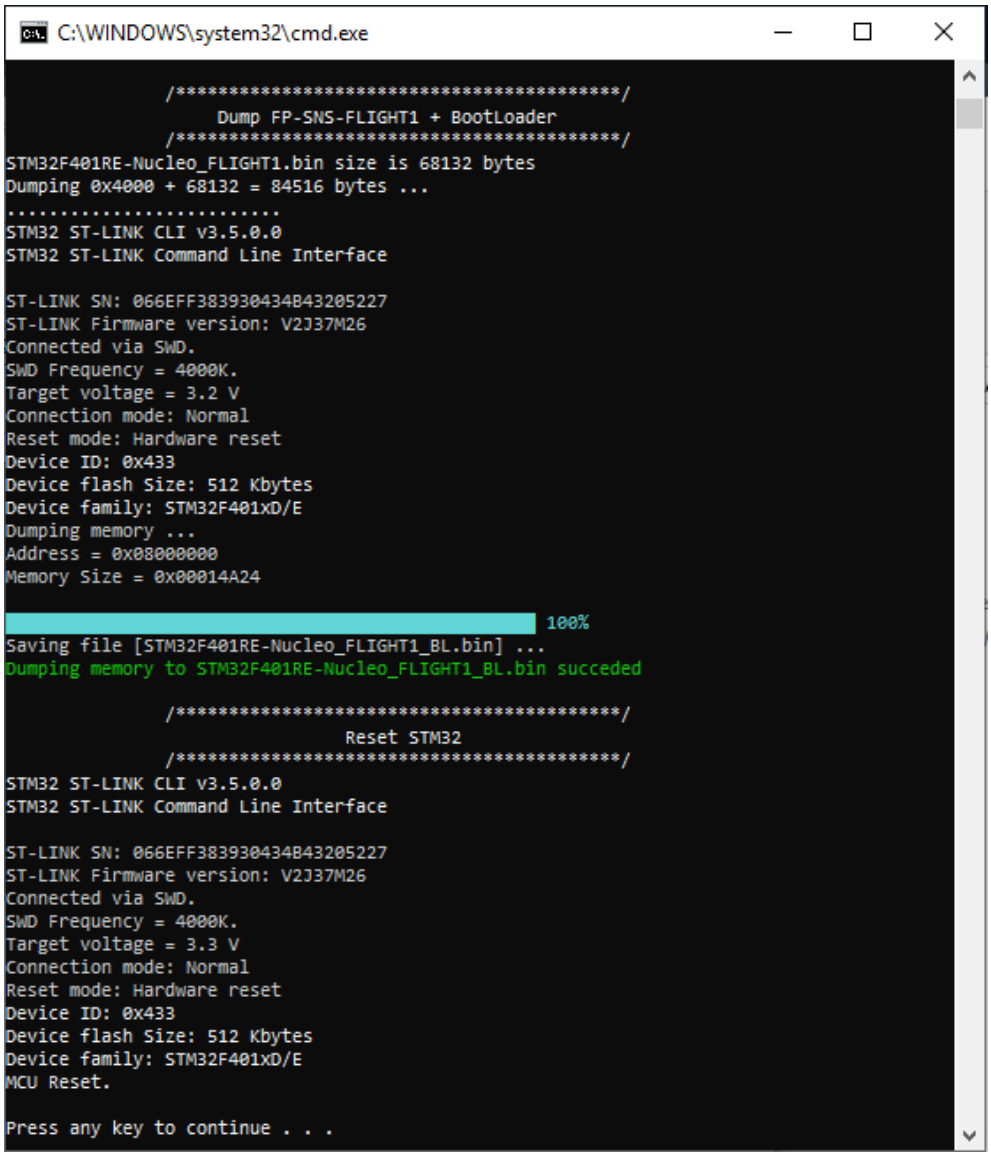
- performs a full Flash erase to start from a clean system
- flashes the BootLoader to the correct position 0x08000000
- flashes the firmware to the correct position 0x08004000

Figure 9. **BootLoader and FP-SNS-FLIGHT1 installation**



The script also dumps an image containing the BootLoader and the firmware. This image file can be directly flashed to the beginning of the Flash memory like in the same way as the image provided in the Binary folder.

Figure 10. FP-SNS-FLIGHT1 dump process



2.7 Firmware-over-the-air (FOTA) update

The FP-SNS-FLIGHT1 firmware can be updated in FOTA through the Bluetooth low energy protocol, communicating with an Android/iOS device, via the STBLESensor application (version 5.2.0 and above) available on their respective stores.

To update the firmware, the STBLESensor application sends the update size (bytes) and its associated CRC (cyclic redundancy check) value to the FP-SNS-FLIGHT1. Once the update has been received, the FP-SNS-FLIGHT1 uses the hardware CRC calculation unit included in the processor to check update integrity.

For NUCLEO-F401RE and NUCLEO-L476RG, if the CRC computed matched the CRC expected, the FP-SNS-FLIGHT1 writes the “magic number”at the beginning of the third Flash region, just before the saved FOTA, to signal the boot loader a Firmware update has been received and checked, and is ready to update the FP-SNS-FLIGHT1 (see Section 2.11: Firmware over-the-air (FOTA) update with STBLESensor).

2.8 APIs

Detailed technical information regarding the APIs available to the user can be found in a compiled HTML file located inside the “Documentation” folder of the software package, where all the functions and parameters are fully described.

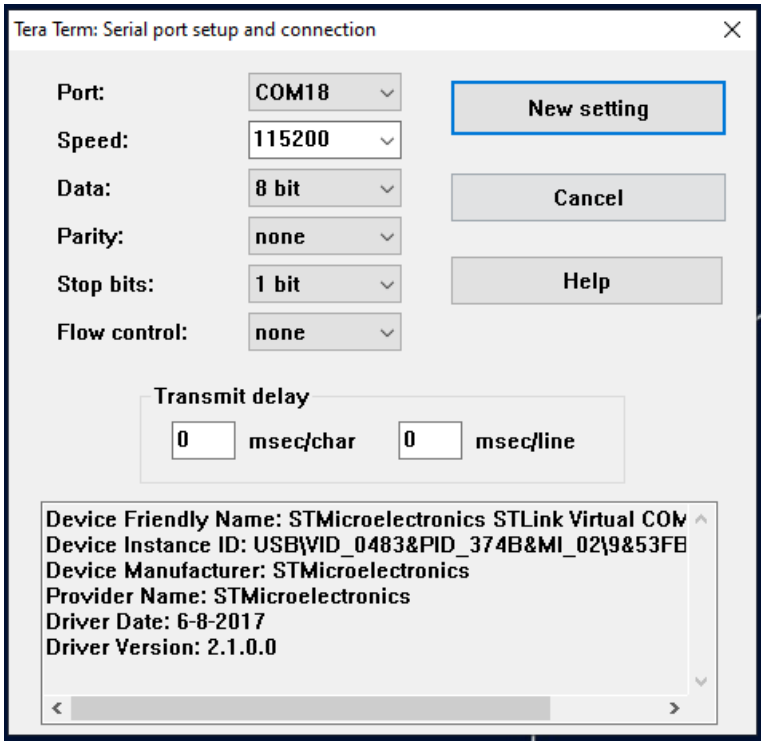
2.9 Sample application description

The available sample application uses the X-NUCLEO-53L3A2 (or VL53L3CX-SATEL) and X-NUCLEO-BNRG2A1 expansion boards with the NUCLEO-F401RE or NUCLEO-L476RG or NUCLEO-U575ZI-Q development boards.

Ready to be built projects are available for multiple IDEs.

You can set up a terminal window for the appropriate UART communication port to control the initialization phase.

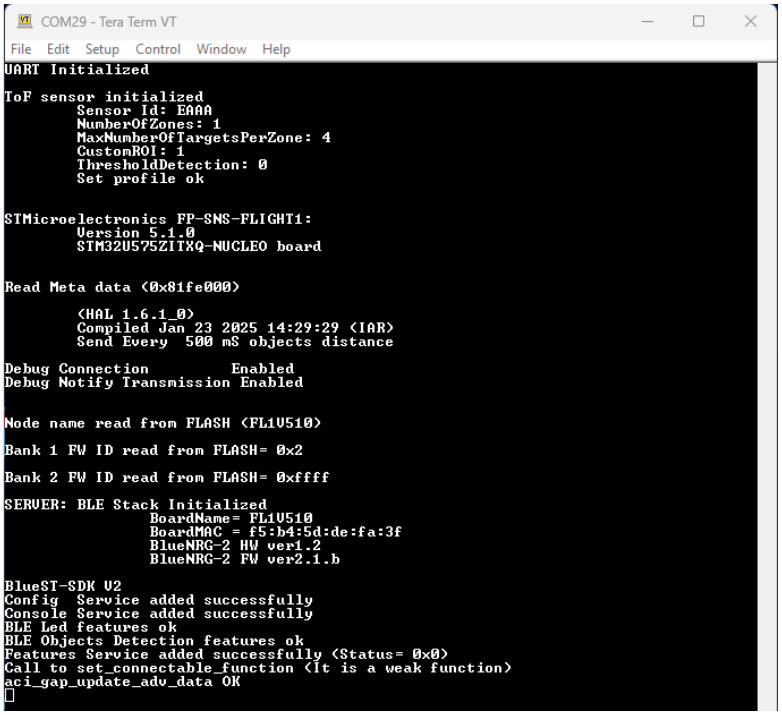
Figure 11. Terminal setting



When pressing the reset button, the application:

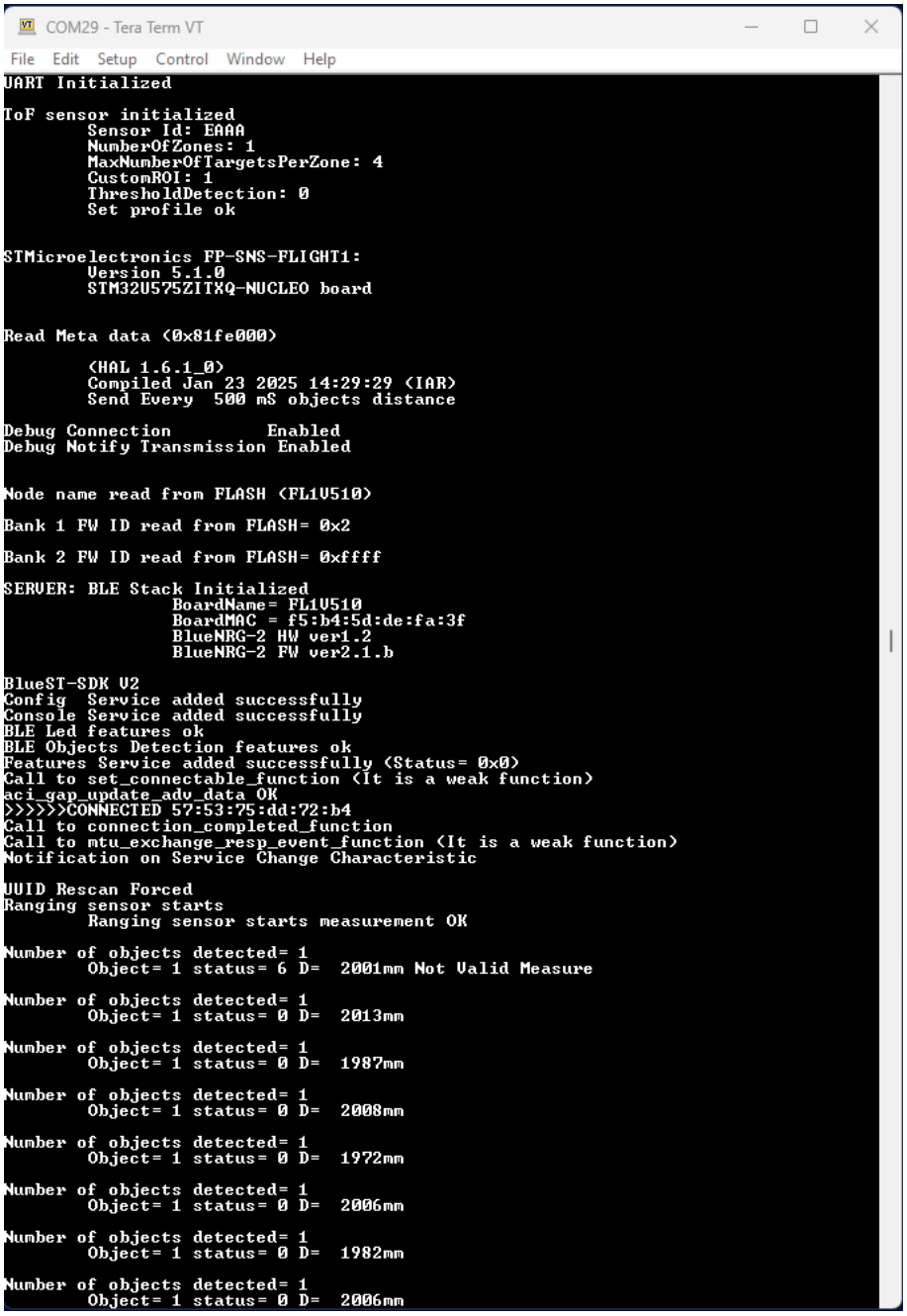
1. Starts initializing the UART interface.
2. Initializes the X-NUCLEO-53L3A2 expansion board or VL53L3CX-SATEL board.
3. Initializing the BLE stack.
4. Shows the board name and the BLE MAC address.
5. Initializes the BLE service.
6. Initializes the BLE console service adding the stdin/stdout and stderr characteristics.
7. Initializes the BLE configuration service.
8. Tests the Boot Loade Compliance.

Figure 12. **UART console output**



When an Android/iOS device is connected to the NUCLEO-F401RE or NUCLEO-L476RG or NUCLEO-U575ZI-Q board, it is possible to control data transmitted by the board (see the following figure).

Figure 13. **UART console output when a device connects to the board**



2.10 Android and iOS sample client application

The FP-SNS-FLIGHT1 software for STM32Cube is compatible with the STBLESensor Android/iOS applications (version 5.2.0/5.2.0 or higher), available at the respective Google Play/iOS stores.

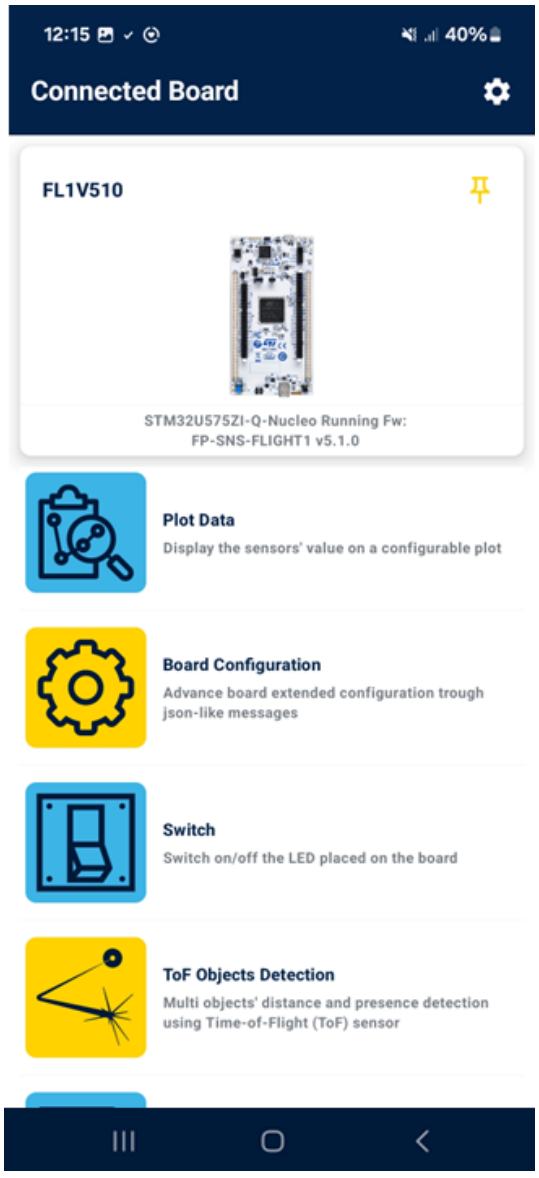
The application allows firmware over-the-air update.

The Android version is used here to show how the application works.

After connection, STBLESensor starts with the page shown below, where the ToF distance plot is displayed when pressing the [**Play**] button.

Following connection, STBLESensor starts with the main page shown below, where there are a list of the available features.

Figure 14. ST BLE Sensor Connected Board



With the "Plot Data" selection the ToF Distance plot is displayed.

Figure 15. **STBLESensor (Android version) ToF distance plot**

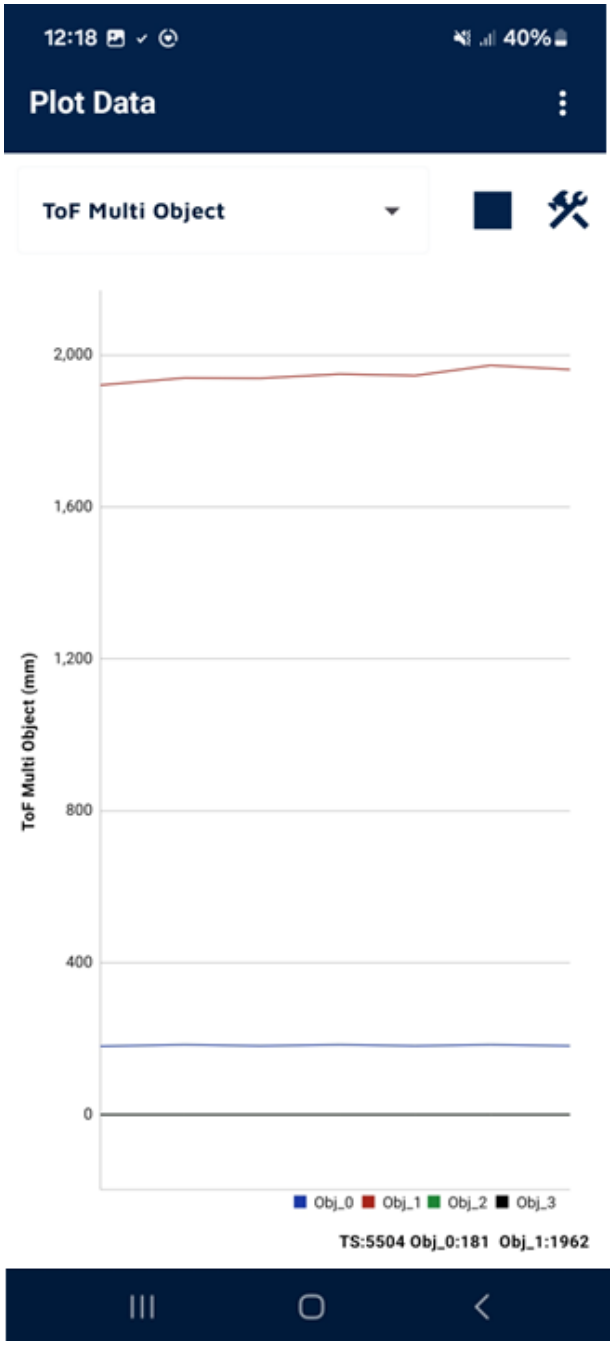
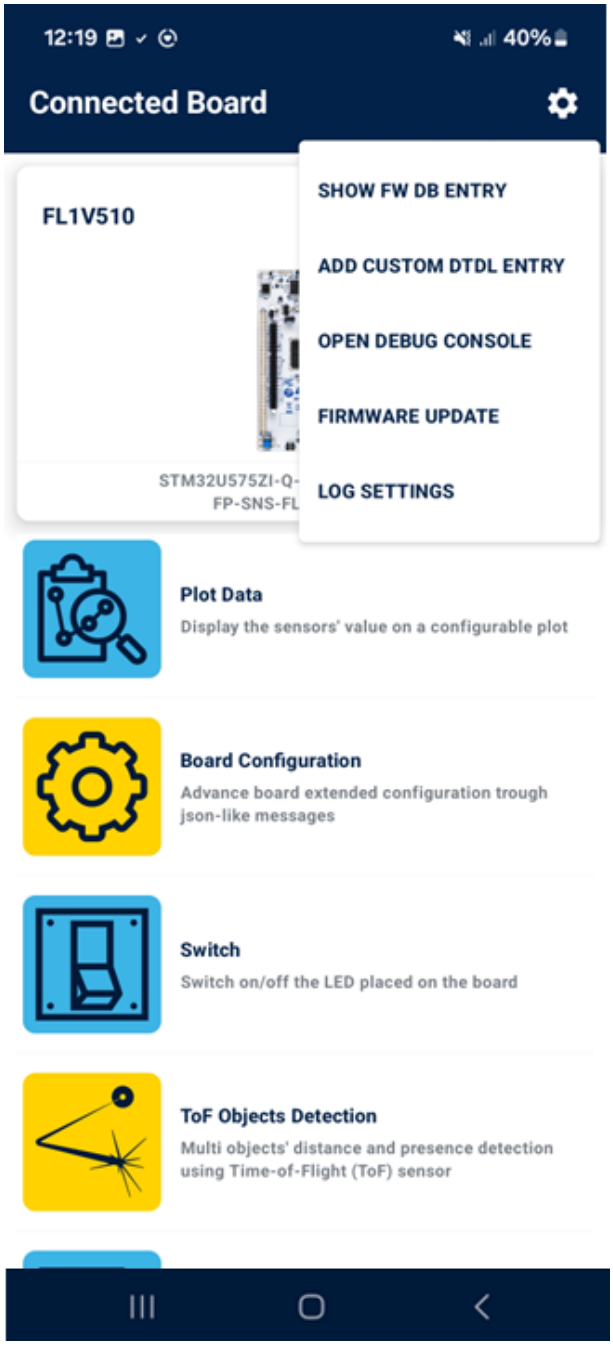


Figure 16. **STBLESensor (Android version) menu selection**

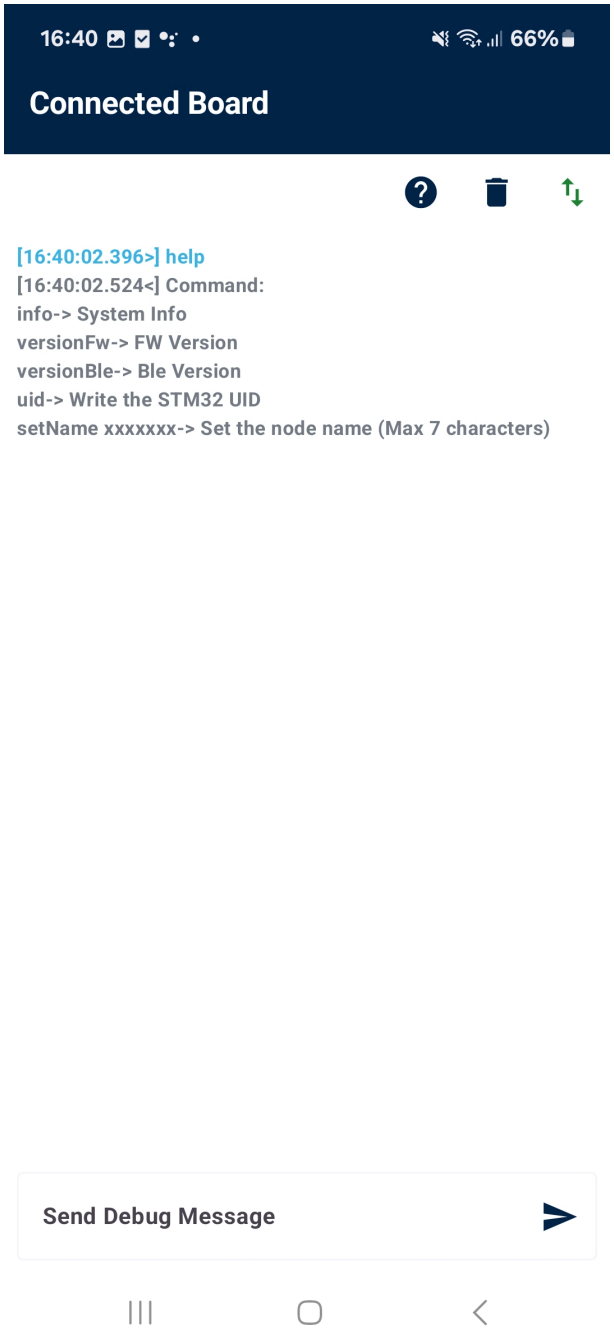


If the debug console is enabled, stdin is displayed and any message written triggers a reply with the same message if the command line is not recognized, as shown below.

Figure 17. **STBLESensor (Android version) Debug console (stdin/stdout/stderr)**

The [**help**] command shows a list of the recognized commands.

Figure 18. **STBLESensor (Android version) Debug console - help command**



[**ToF Objects Detection**] shows the distances of the detected objects (up to four objects) or the person presence.

Figure 19. **STBLESensor (Android version) ToF Objects Detection - distance**

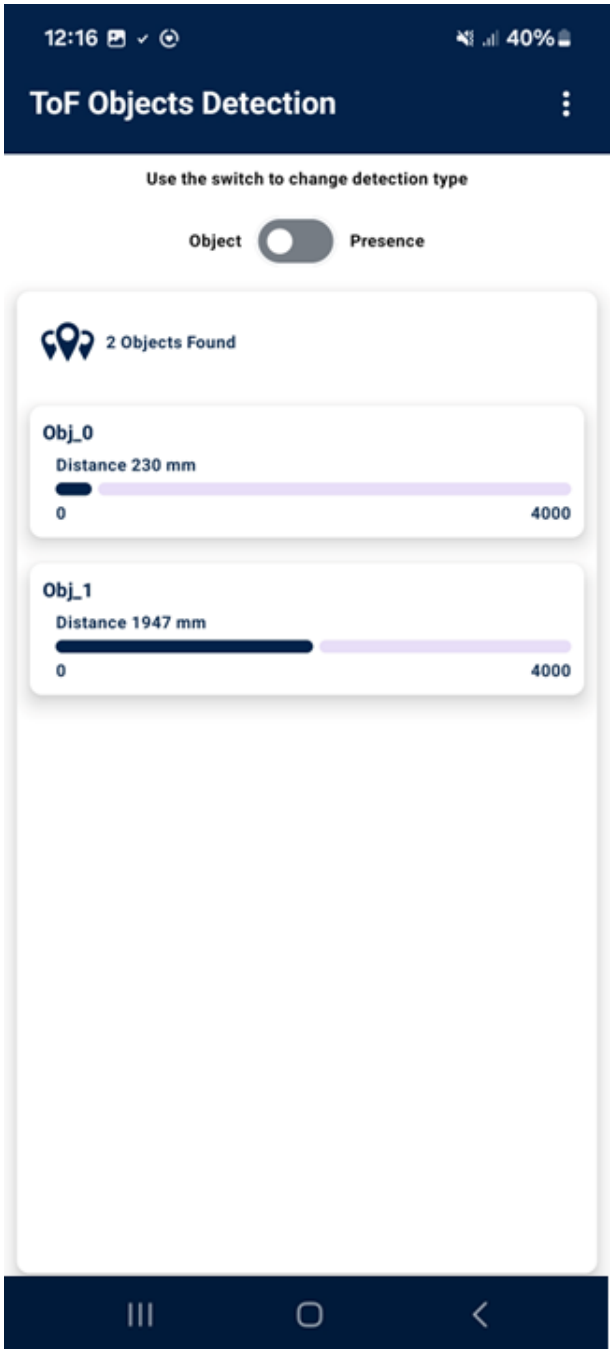


Figure 20. **STBLESensor (Android version) ToF Objects Detection - presence**

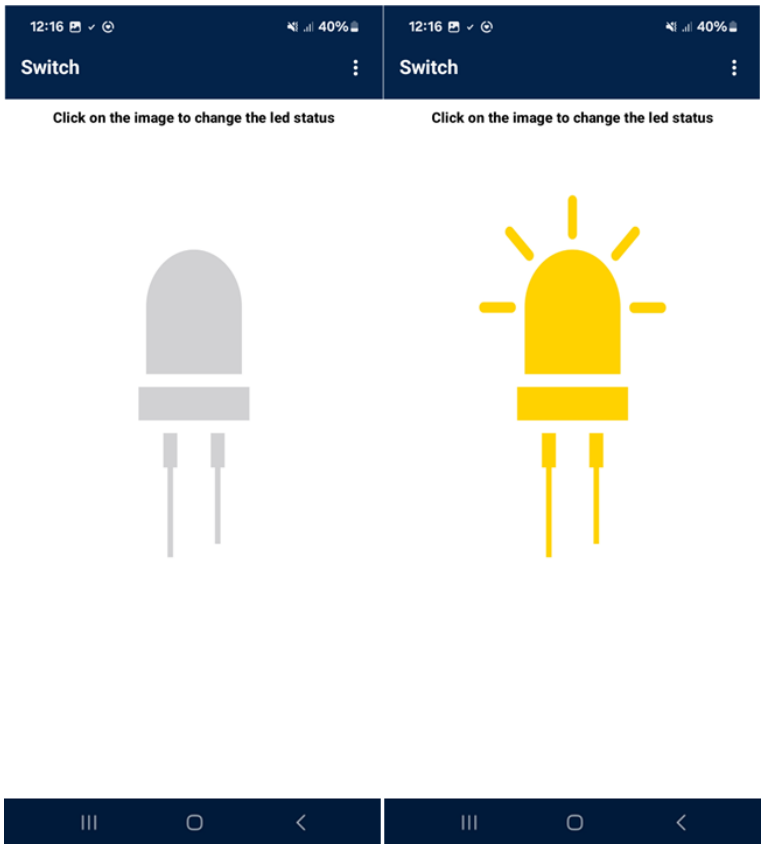
The presence is identified inside a fixed range distance that can be modified by the following defines:

- `#define PRESENCE_MIN_DISTANCE_RANGE 300`
- `#define PRESENCE_MAX_DISTANCE_RANGE 800`

in Applications\FLIGHT1\Inc\FLIGHT1_config.h.

The status of the STM32Nucleo on-board LED can be switched in the following app page.

Figure 21. **STBLESensor (Android version) LED switch page**



With the board configuration page a few firmware details can be shown. The image below shows the available commands:

Figure 22. **STBLESensor (Android version) Board Configuration**

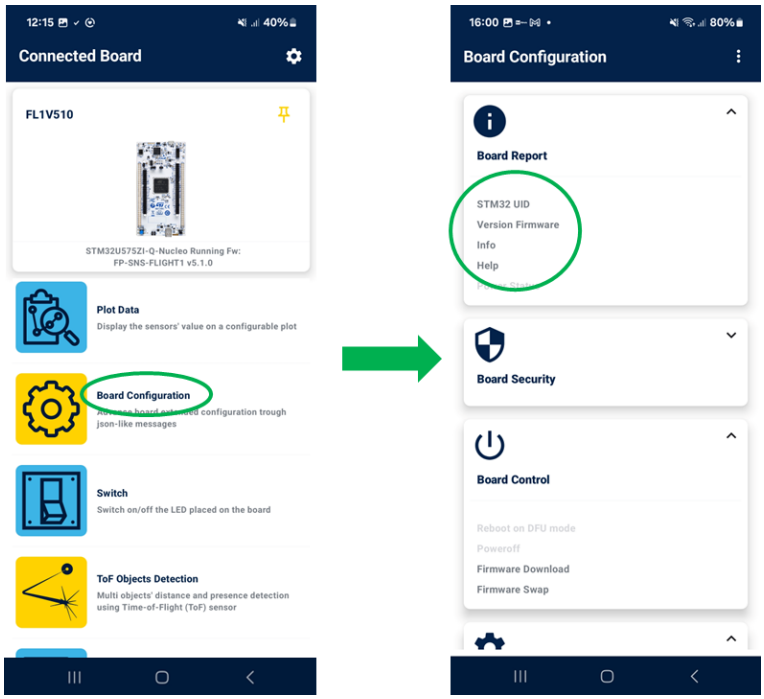
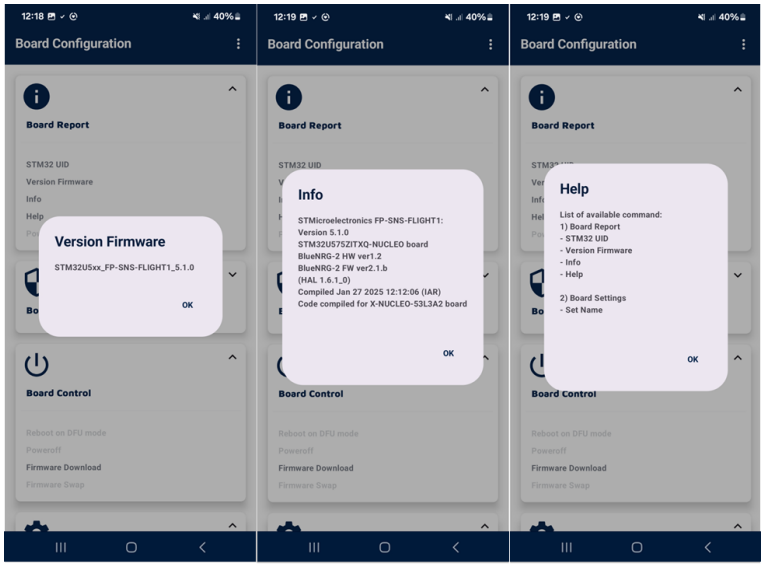
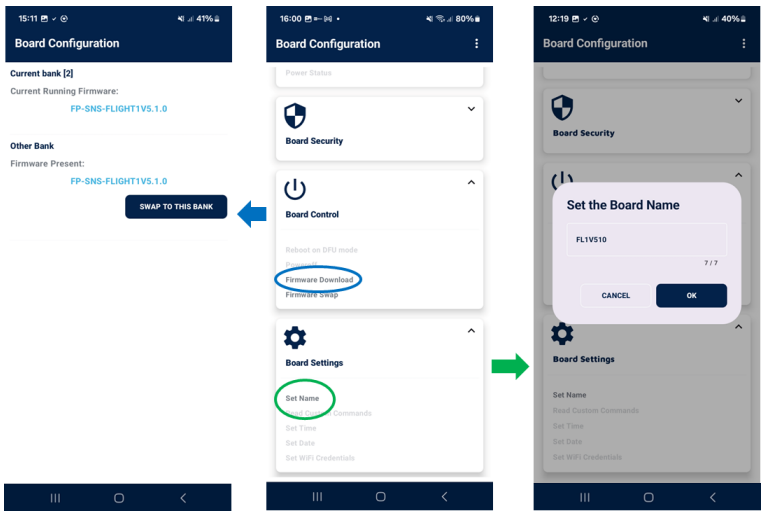


Figure 23. STBLESensor (Android version) Board Configuration - Commad Version Firmware, Info, Help



A new IoT node name can selected and firmware download or swap memory bank can be do, too.

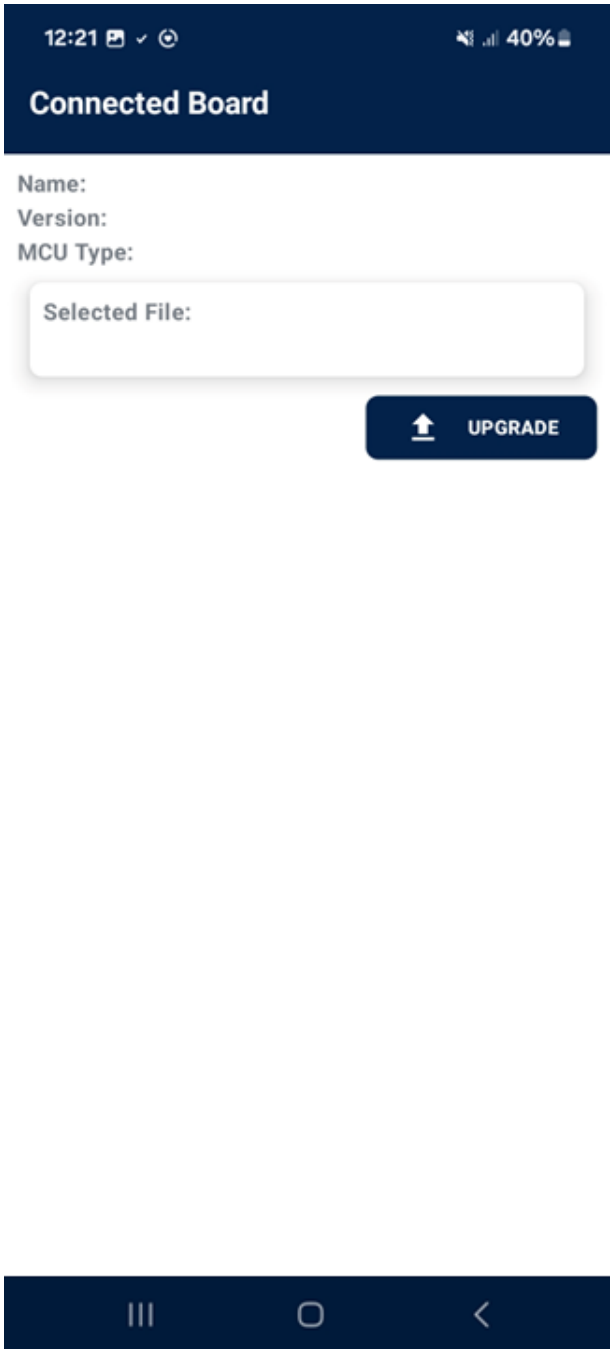
Figure 24. STBLESensor (Android version) Board Configuration - Commad Firmware Download, Swap Bank



2.11 Firmware over-the-air (FOTA) update with STBLESensor

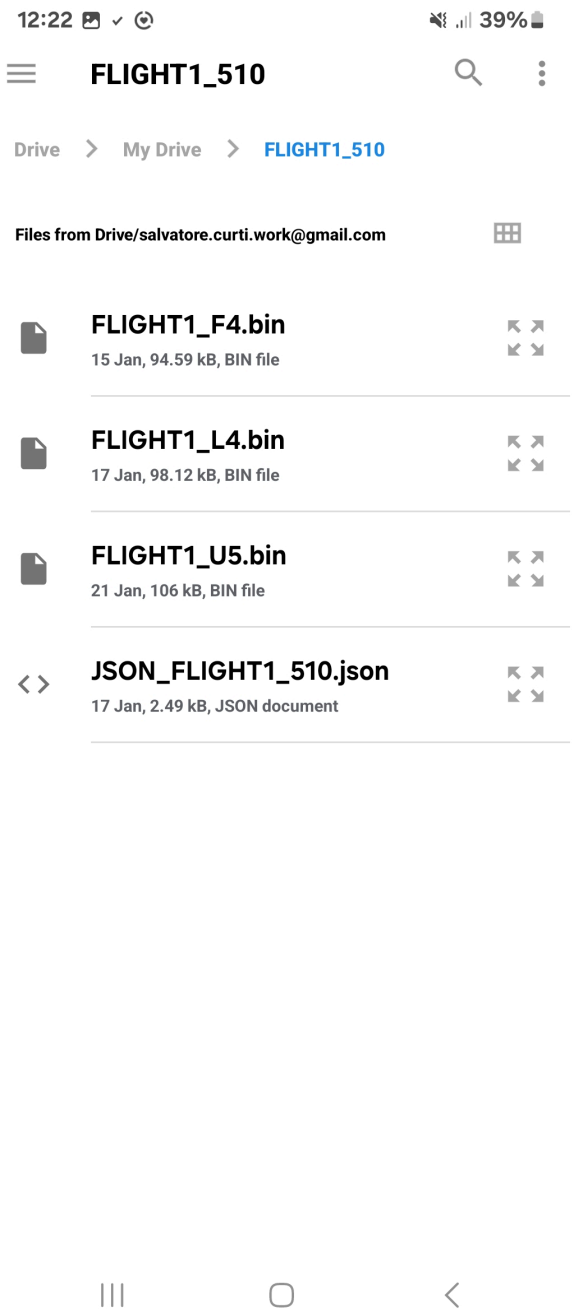
If the Firmware upgrade option menu is selected in the STBLESensor main application page, the following page appears.

Figure 25. STBLESensor (Android version) firmware upgrade page



The STBLESensor application shows which version of the FP-SNS-FLIGHT1 software is running and the board type. To apply an update, select the appropriate update file and press the red button.

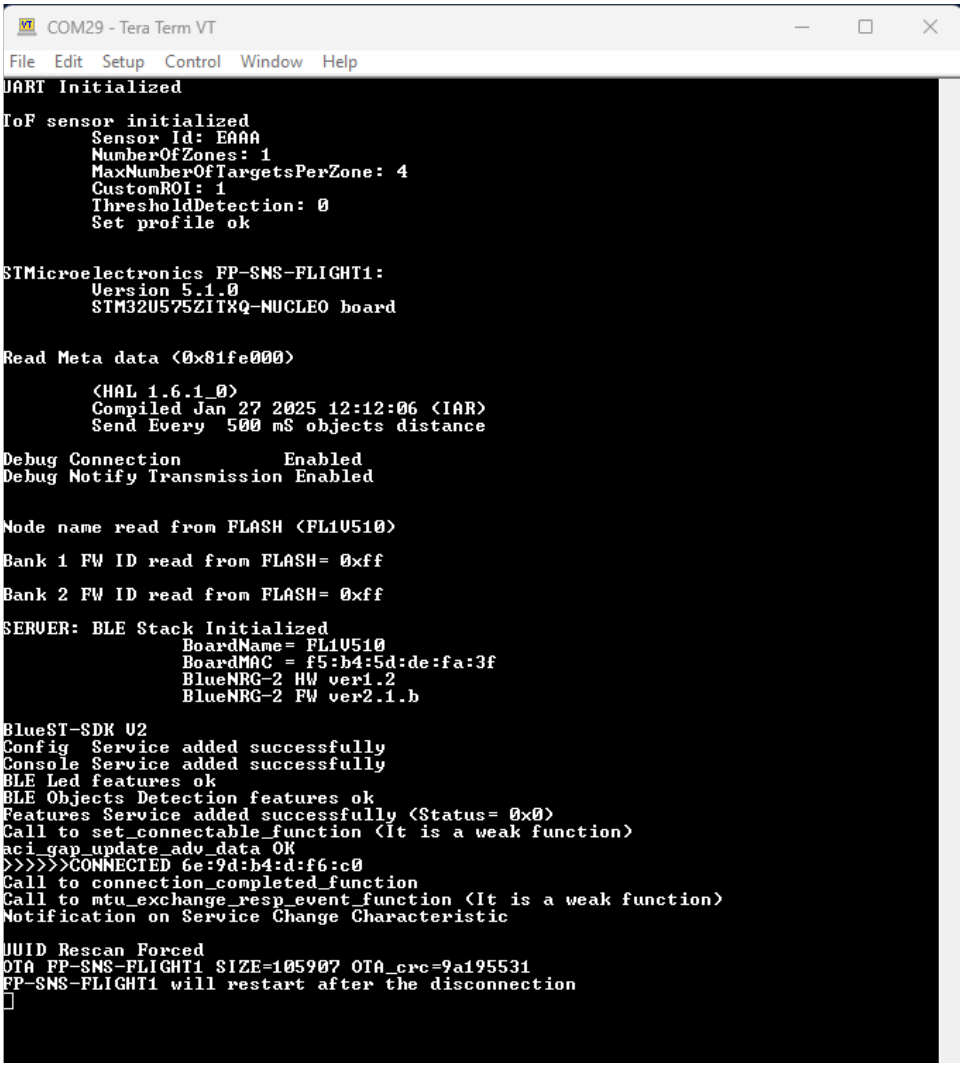
Figure 26. **STBLESensor (Android version) firmware update file selection**



STBLESensor sends an update of a certain byte size and corresponding CRC value to FP-SNS-FLIGHT1.

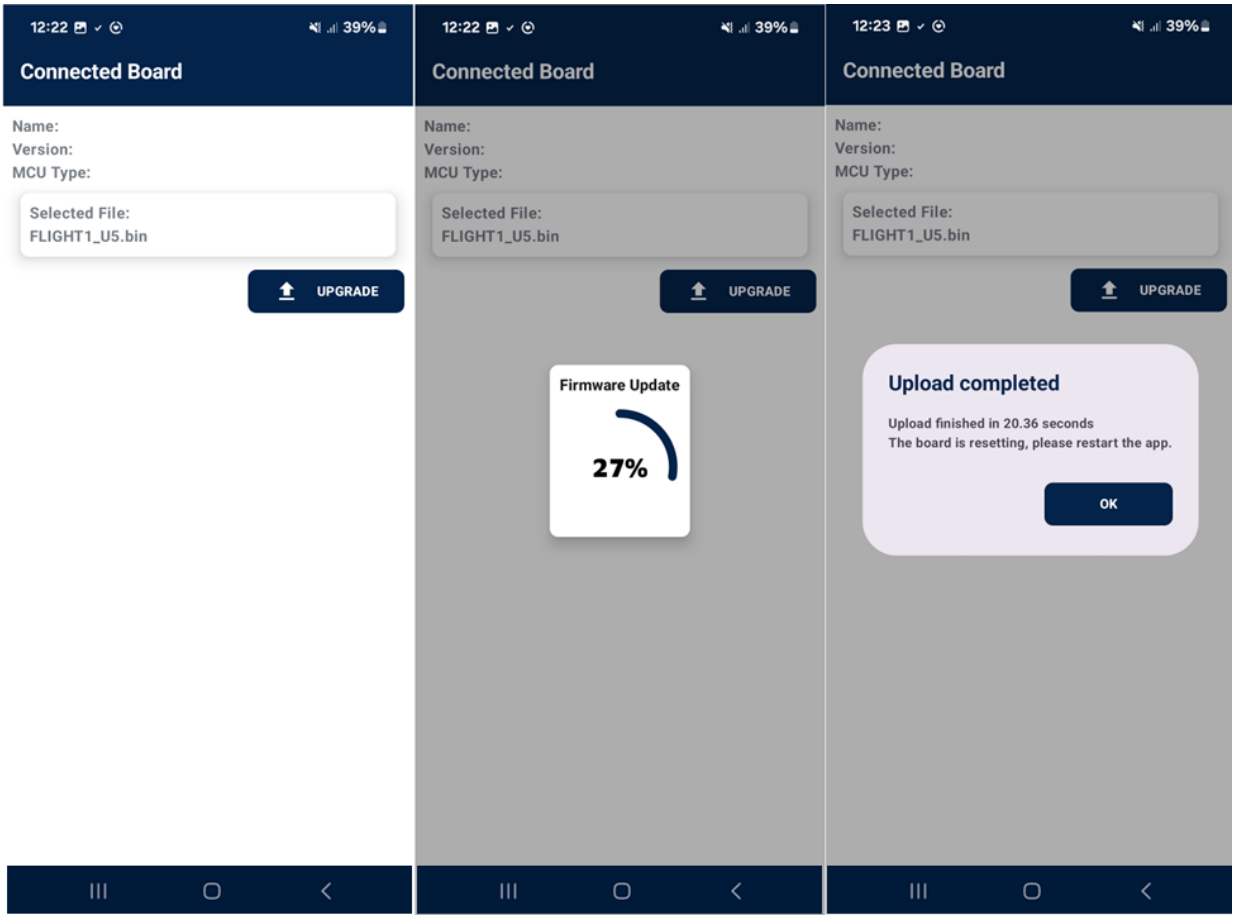
The figure below shows the terminal window with the debug information returned during FOTA for the STM32 Nucleo platform when FP-SNS-FLIGHT1 is controlled via UART.

Figure 27. **Terminal window information during FOTA**



During the FOTA procedure, the STBLESensor application shows the remaining packets to be sent and the total update time when the procedure ends.

Figure 28. **STBLESensor (Android version) application page during FOTA and on completion**



3 System setup guide

3.1 Hardware description

This section describes the hardware components needed for sensor-based application development.

3.1.1 STM32 Nucleo

STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

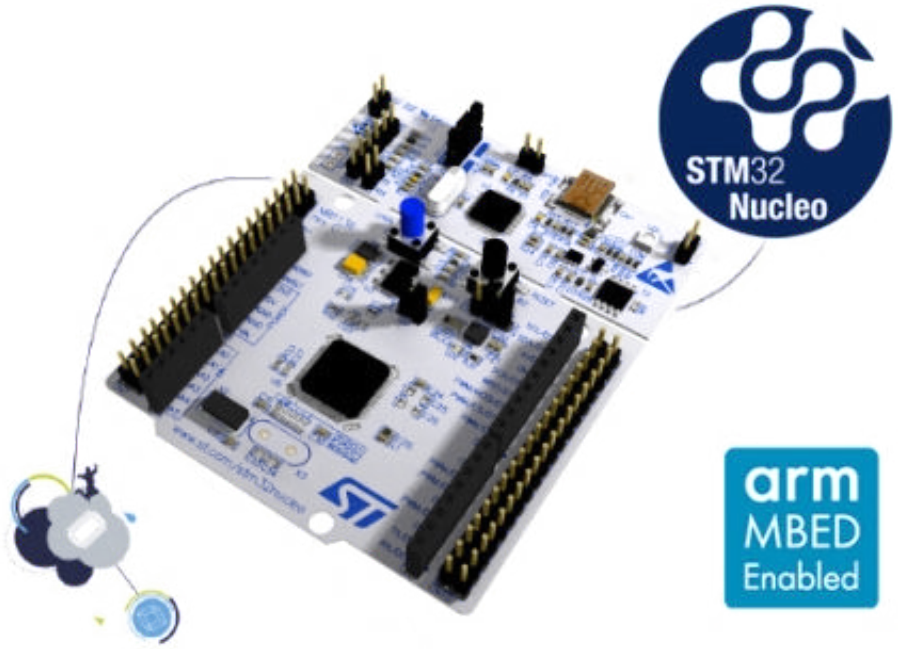
The Arduino connectivity support and ST morpho connectors make it easy to expand the functionality of the STM32 Nucleo open development platform with a wide range of specialized expansion boards to choose from.

The STM32 Nucleo board does not require separate probes as it integrates the ST-LINK/V2-1 debugger/ programmer.

The STM32 Nucleo board comes with the comprehensive STM32 software HAL library together with various packaged software examples for different IDEs (IAR EWARM, Keil MDK-ARM, STM32CubeIDE, mbed and GCC/ LLVM).

All STM32 Nucleo users have free access to the mbed online resources (compiler, C/C++ SDK and developer community) at www.mbed.org to easily build complete applications.

Figure 29. STM32 Nucleo board



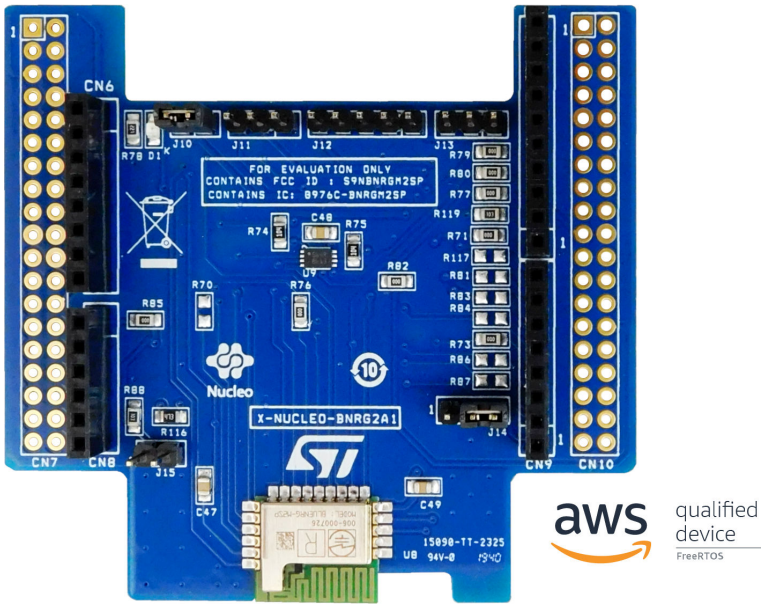
3.1.2 X-NUCLEO-BNRG2A1 expansion board

The X-NUCLEO-BNRG2A1 expansion board provides Bluetooth® Low Energy connectivity for developer applications and can be plugged onto an STM32 Nucleo development board (for example, NUCLEO-L476RG with an ultra-low power STM32 microcontroller) through its Arduino UNO R3 connectors.

The expansion board features the Bluetooth® v5.2 compliant and FCC certified BlueNRG-M2SP application processor module based on the ST BlueNRG-2 System-on-Chip. This SoC manages the complete Bluetooth® Low Energy stack and protocols on its Cortex-M0 core and programmable flash memory, which can accommodate custom applications developed using the SDK. The BlueNRG-M2SP module supports master and slave modes, increased transfer rates with data length extension (DLE), and AES-128 security encryption.

The X-NUCLEO-BNRG2A1 interfaces with the STM32 Nucleo microcontroller via SPI connections and GPIO pins, some of which can be configured through the hardware.

Figure 30. X-NUCLEO-BNRG2A1 BLE expansion board



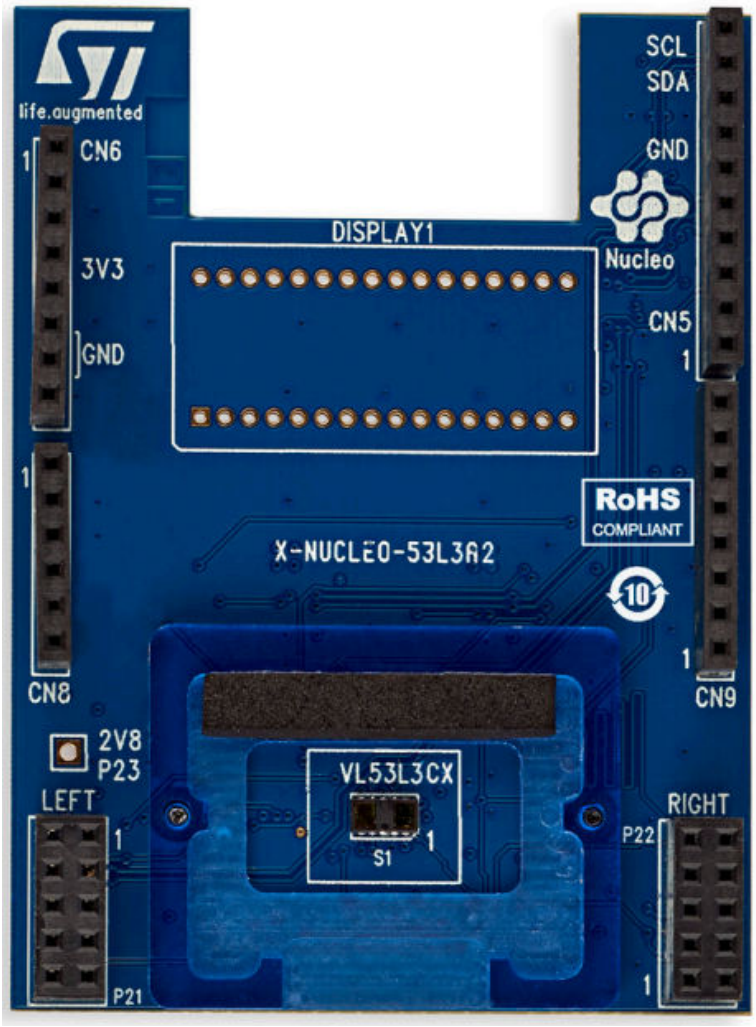
3.1.3 X-NUCLEO-53L3A2 expansion board

The X-NUCLEO-53L3A2 expansion board is designed around the VL53L3CX ranging sensor and is based on the ST patented FlightSense technology.

The expansion board is delivered with two VL53L3CX breakout boards.

To allow the user to validate the VL53L3CX in an environment as close as possible to its final application, the X-NUCLEO-53L3A2 expansion board package includes a holder in which three different height spacers of 0.25 mm, 0.5 mm, and 1 mm can be fitted with the cover glass above the spacer. The height spacers are used to simulate different air gap distances between the VL53L3CX sensor and the cover glass.

Figure 31. X-NUCLEO-53L3A2 expansion board

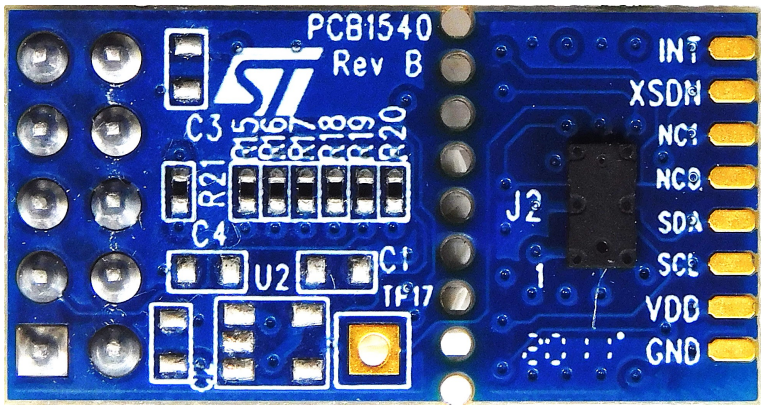


3.1.4 SATEL breakout boards

The SATEL breakout boards can be used for easy integration into customer devices. Thanks to the voltage regulator, the breakout boards can be used in any application in the supply range 2.8 V to 5 V.

The PCB section supporting the module is perforated so that developers can break off the mini-PCB for use in a2V8 or 3V3 supply application using flying leads. This makes it easier to integrate the SATEL breakout boards into development and evaluation devices due to their small form factor.

Figure 32. Example of a VL53L3CX-SATEL breakout board



3.2 Software description

The following software components are required to create a suitable development environment for applications based on the STM32 Nucleo equipped with time-of-flight sensor and BlueNRG expansion boards:

- FP-SNS-FLIGHT1: a Bluetooth low energy and Time-of-Flight sensors software for STM32Cube. The FP-SNS-FLIGHT1 firmware and related documentation is available on www.st.com.
- Development tool-chain and Compiler: The STM32Cube expansion software supports the following environments:
  - IAR Embedded Workbench for ARM® (EWARM) toolchain + ST-LINK
  - RealView Microcontroller Development Kit (MDK-ARM) toolchain + ST-LINK
  - STM32CubeIDE + ST-LINK

3.3 Hardware setup

The following hardware components are needed:

1. One STM32 Nucleo development platform (order code: NUCLEO-U575ZI-Q or NUCLEO-F401RE or NUCLEOL476RG).
2. One VL53L3CX ToF ranging sensor expansion board (order code: X-NUCLEO-53L3A2 or VL53L3CX-SATEL).
3. One BlueNRG Bluetooth low energy expansion board (order code: X-NUCLEO-BNRG2A1).
4. One USB type A to Mini-B USB cable to connect the STM32 Nucleo to the PC for NUCLEO-F401RE or NUCLEO-L476RG.
5. One USB type A to Micro-B USB cable to connect the STM32 Nucleo to the PC for NUCLEO-U575ZI-Q.

3.4 System setup guide

3.4.1 STM32 Nucleo and sensor expansion board setup

The STM32 Nucleo development board integrates the ST-LINK/V2-1 debugger/programmer. You can download the relevant version of the ST-LINK/V2-1 USB driver by searching STSW-LINK008 or STSW-LINK009 on www.st.com (based on your version of Microsoft Windows).

Connect the expansion boards through the Arduino UNO R3 extension connector as shown in the following pictures.

Figure 33. STM32 Nucleo development board plus X-NUCLEO-BNRG2A1 expansion board

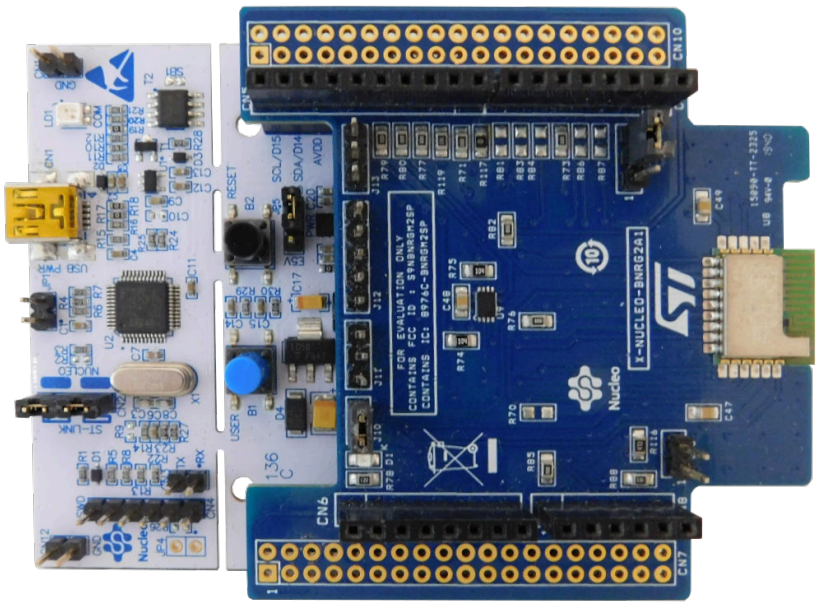
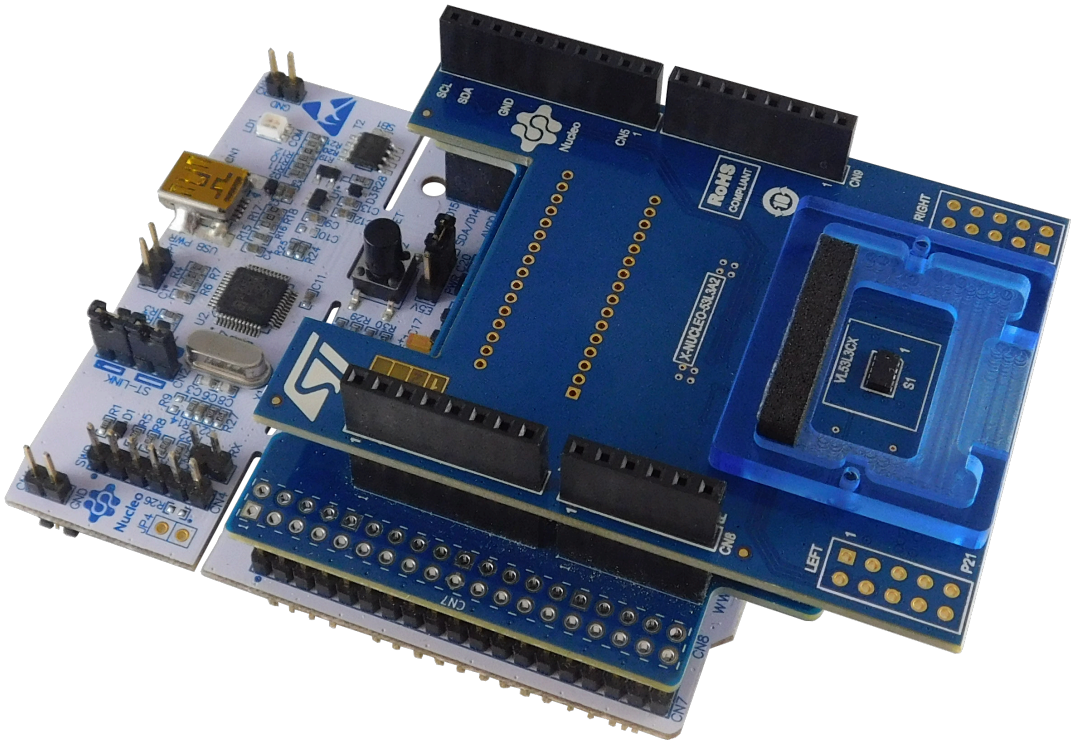


Figure 34. STM32 Nucleo development board plus X-NUCLEO-BNNRG2A1 and X-NUCLEO-53L3A2 expansion boards



Note: You must connect the boards in the sequence shown above to optimize the performance of the BlueNRG-2 module on the X-NUCLEO-BNRG2A1 expansion board and to reduce interference from its antenna.

Revision history

Table 2. **Document revision history**

Date	Version	Changes
25-Feb-2016	1	Initial release.
05-Dec-2016	2	Updated Section "Introduction", Section 2.1: "Overview ", Figure 4: "Initialization phase", Figure 5: "UART console output when the BLE services are started", Figure 6: "UART console output when a device first connects with the board", Figure 15: "BlueMS (android version) initial page after BLE connection", Figure 1: "FP-SNS-FLIGHT1 software architecture" and Figure 18: "BlueMS (android version) Serial console (stdout/stderr)"; Added Section 2.6: "Flash organization", Section 2.7: "The boot process", Section 2.8: "Firmware-Over-The-Air (FOTA) update" and Section 2.9: "Firmware Over The Air (FOTA) update with BlueMS"
20-Feb-2017	3	Throughout document: - minor text and formatting changes. - added IKS01A2 expansion board compatibility information Added Section 2.6: " The installation process" Updated Section 2.9: "Sample application description", Section 2.10: " Android and iOS sample client application" and Section 2.11: "Firmware-over-the-air (FOTA) update with BlueMS"
31-Mar-2017	4	Updated title, Introduction, Section 2.1: "Overview ", Section 2.2: "Architecture", Section 2.3: "Folder structure", Section 2.6: "The installation process", Section 2.9: "Sample application description", Section 2.10: "Android and iOS sample client application", Section 2.11: "Firmware over-the-air (FOTA) update with BlueMS". Added X-NUCLEO-53L0A1 expansion board compatibility information. Added Section 3.1.8: "X-NUCLEO-53L0A1 expansion board".
03-Sep-2018	5	Updated Introduction, Section 2.1 Overview, Figure 2. FP-SNS-FLIGHT1 software architecture, Section 2.6 The installation process, Section 2.9 Sample application description, Figure 16. BlueMS (android version) main page (after BLE connection), Figure 21. BlueMS (android version) Serial console (stdout/stderr), Section 3.3.1 Hardware setup and Section 3.3.3.1 STM32 Nucleo and sensor expansion board setup. Added Section 3.1.4 X-NUCLEO-NFC04A1 expansion board, Section 3.3.3.2 STEVAL-BCNKT01V1 setup and Section 3.3.3.3 Important additional hardware information.
01-Oct-2020	6	Updated Introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 2.3 Folder structure, Section 2.6 The installation process, Section 2.7 Firmware-over-the-air (FOTA) update, Section 2.9 Sample application description, Section 2.10 Android and iOS sample client application, Section 2.11 Firmware over-the-air (FOTA) update with STBLESensor, Section 3.2 Software description, Section 3.3 Hardware setup, Section 3.4.1 STM32 Nucleo and sensor expansion board setup and Section 3.4.2 Important additional hardware information. Added Section 3.1.3 X-NUCLEO-IDB05A2 expansion board and Section 3.1.4 X-NUCLEO-53L1A2 expansion board.
02-Nov-2020	7	Updated title.

Date	Version	Changes
15-Oct-2021	8	Updated *Section 2.1 Overview, Section 2.2 Architecture, Section 2.6 The installation process, Section 2.7 Firmwareover-the-air (FOTA) update, Section 2.9 Sample application description, Section 2.10 Android and iOS sample client application, Section 2.11 Firmware over-the-air (FOTA) update with STBLESensor, Section 3.3 Hardware setup and Section 3.4.1 STM32 Nucleo and sensor expansion board setup*. Removed Section 3.1.4 X-NUCLEO-53L1A2 expansion board and Section 3.4.3 Important additional software information. Added *Section 3.1.4 X-NUCLEO-53L3A2 expansion board*.
03-May-2023	9	Updated introduction, *Section 1 Acronyms and abbreviations, Section 2.1 Overview , Section 2.2 Architecture, Section 2.3 Folder structure, Section 2.5 The boot process, Section 2.6 The installation process, Section 2.7 Firmwareover-the-air (FOTA) update, Section 2.9 Sample application description, Section 2.10 Android and iOS sample client application, Section 2.7 Firmware-over-the-air (FOTA) update, Section 3.2 Software description, Section 3.3 Hardware setup, Section 3.4.1 STM32 Nucleo and sensor expansion board setup* Removed Section 3.1.2 X-NUCLEO-NFC04A1 expansion board, Section 3.1.2 X-NUCLEO-NFC04A1 expansion board and Section 3.4.2 Important additional hardware information. Added *Section 3.1.2 X-NUCLEO-BNRG2A1 expansion board.*
06-Jun-2023	10	Updated Overview. Added compatibility to STM32CubeMX.
05-Mar-2025	11	Updated Section 2: FP-SNS-FLIGHT1 software description and Section 2: FP-SNS-FLIGHT1 software description.

Contents

List of tables

List of figures