Getting started with the STM32Cube function pack for IoT node with GNSS and cellular connectivity for Asset Tracking applications based on TomTom

Introduction

FP-ATR-TOMTOM1 is an STM32Cube function pack which lets you connect your IoT node to a cellular network and send positioning coordinates, provided by a global navigation satellite system (GNSS) receiver, to TomTom online services. These coordinates are used to perform Reverse Geocoding which is the translation into a street address.

This software, together with the suggested combination of STM32 and ST devices, can be used, for example, to develop asset tracking applications.

The software is provided for STM32F401RE and STM32L496AG MCUs and it is easily portable across different MCU families thanks to STM32Cube.

It provides sample implementation for STM32 Nucleo platforms equipped with the X-NUCLEO-GNSS1A1 expansion board, featuring a GNSS receiver based on Teseo-LIV3F module, and the 2G/3G or the LTE cellular expansion board featuring a Quectel UG96 or BG96 module.

The cellular expansion board, provided within the P-L496G-CELL01 and P-L496G-CELL02 packages, can be either connected directly to the STMod+ connector of a 32L496GDISCOVERY board, or to an STM32 Nucleo development board, through the X-NUCLEO-STMODA1 expansion board.

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
BSP	Board support package
C2C	Cellular to cloud
GNSS	Global navigation satellite system
GPS	Global positioning system
HAL	Hardware abstraction layer
I2C	Inter-integrated circuit
RTC	Real-time clock
SIM	Subscriber identity module
TLS	Transport layer security
UART	Universal asynchronous receiver-transmitter

2 What is STM32Cube?

STM32Cube™ represents the STMicroelectronics initiative to make developers' lives easier by reducing development effort, time and cost. STM32Cube covers the STM32 portfolio.

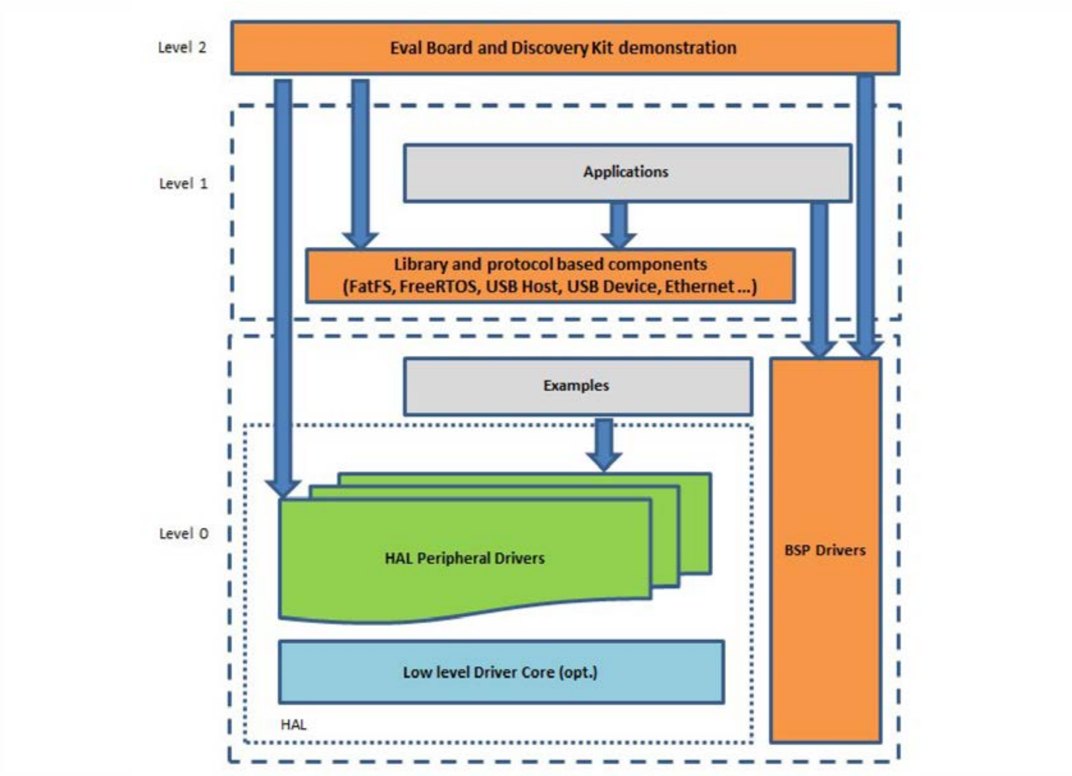
STM32Cube version 1.x includes:

- STM32CubeMX, a graphical software configuration tool that allows the generation of C initialization code using graphical wizards.
- A comprehensive embedded software platform specific to each series (such as the STM32Cube for the STM32 series), which includes:
  – the STM32Cube HAL embedded abstraction-layer software, ensuring maximized portability across the STM32 portfolio
  – a consistent set of middleware components such as RTOS, USB, TCP/IP and graphics
  – all embedded software utilities with a full set of examples

2.1 STM32Cube architecture

The STM32Cube firmware solution is built around three independent levels that can easily interact with one another, as described in the diagram below.

Figure 1. Firmware architecture



Level 0: This level is divided into three sub-layers:

- Board Support Package (BSP): this layer offers a set of APIs relative to the hardware components in the hardware boards (Audio codec, IO expander, Touchscreen, SRAM driver, LCD drivers. etc…); it is based on modular architecture allowing it to be easily ported on any hardware by just implementing the low level routines. It is composed of two parts:

   – Component: is the driver relative to the external device on the board and not related to the STM32, the component driver provides specific APIs to the external components of the BSP driver, and can be ported on any other board.
   – BSP driver: links the component driver to a specific board and provides a set of easy to use APIs. The API naming convention is BSP_FUNCT_Action(): e.g., BSP_LED_Init(), BSP_LED_On().
- Hardware Abstraction Layer (HAL): this layer provides the low level drivers and the hardware interfacing methods to interact with the upper layers (application, libraries and stacks). It provides generic, multi-instance and function-oriented APIs to help offload user application development time by providing ready to use processes. For example, for the communication peripherals (I²C, UART, etc.) it provides APIs for peripheral initialization and configuration, data transfer management based on polling, interrupt or DMA processes, and communication error management. The HAL Drivers APIs are split in two categories: generic APIs providing common, generic functions to all the STM32 series and extension APIs which provide special, customized functions for a specific family or a specific part number.
- Basic peripheral usage examples: this layer houses the examples built around the STM32 peripherals using the HAL and BSP resources only.

Level 1: This level is divided into two sub-layers:
- Middleware components: set of libraries covering USB Host and Device Libraries, STemWin, FreeRTOS, FatFS, LwIP, and PolarSSL. Horizontal interaction among the components in this layer is performed directly by calling the feature APIs, while vertical interaction with low-level drivers is managed by specific callbacks and static macros implemented in the library system call interface. For example, FatFs implements the disk I/O driver to access a microSD drive or USB Mass Storage Class.
- Examples based on the middleware components: each middleware component comes with one or more examples (or applications) showing how to use it. Integration examples that use several middleware components are provided as well.

Level 2: This level is a single layer with a global, real-time and graphical demonstration based on the middleware service layer, the low level abstraction layer and basic peripheral usage applications for board-based functions.

3 FP-ATR-TOMTOM1 software expansion for STM32Cube

3.1 Overview

The FP-ATR-TOMTOM1 software package expands STM32Cube functionality.

The key features of the package are:

- Complete firmware to connect an IoT node with GNSS module to a 2G/3G or LTE cellular network
- Support for Reverse Geocoding through TomTom online services
- Middleware libraries with support for FreeRTOS, mbedTLS, GNSS, NMEA and JSON parsing functionalities
- Sample implementation available for X-NUCLEO-GNSS1A1 and 2G/3G or LTE cellular expansion boards, when connected to a NUCLEO-F401RE or a 32L496GDISCOVERY board
- Easy portability across different MCU families, thanks to STM32Cube
- Free, user-friendly license terms

This software enables gathering global positioning coordinates with the Teseo-LIV3F GNSS and to transmit via cellular network connection to TomTom online services, to translate the coordinates to a street address, and to a link to the corresponding map tile.

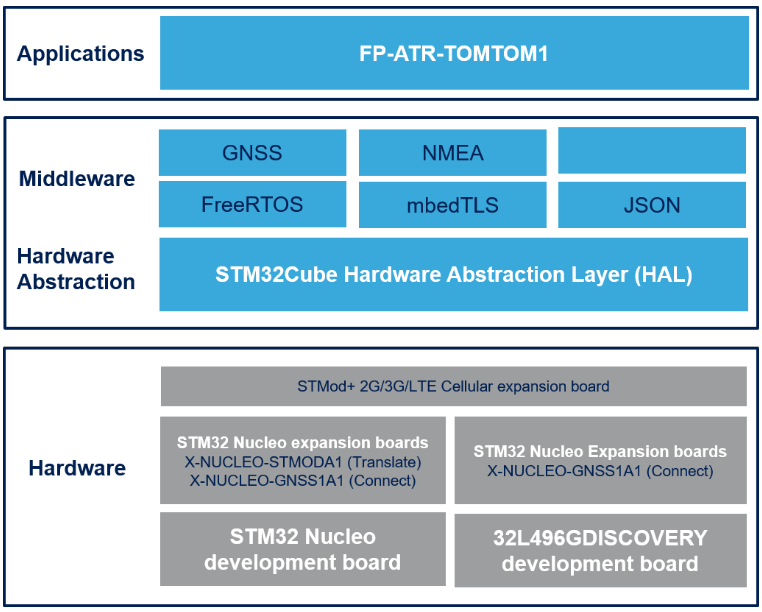
3.2 Architecture

The application software accesses the X-NUCLEO-GNSS1A1 and cellular expansion boards via:

The **STM32Cube HAL** driver layer, which provides a simple, generic, multi-instance set of application programming interfaces (APIs) to interact with the upper application, library and stack layers. It has generic and extension APIs and is directly built around a generic architecture and allows successive layers like the middleware layer to implement functions without requiring specific hardware configurations for a given microcontroller unit (MCU). This structure improves library code reusability and facilitates portability to other devices.

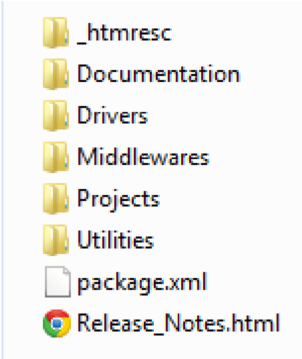
The **board support package** (BSP) layer supports all the peripherals on the STM32 Nucleo except the MCU. This limited set of APIs provides a programming interface for certain board-specific peripherals like the LED, the user button, etc. This interface also helps in identifying the specific board version.

Figure 2. FP-ATR-TOMTOM1 software architecture



3.3 Folder structure

Figure 3. **FP-ATR-TOMTOM1 package folder structure**



The following folders are included in the software package:

- **Documentation**: contains a compiled HTML file generated from the source code which details the software components and APIs.
- **Drivers**: contains the HAL drivers and 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 ARM Cortex-M processor series.
- **Middlewares**: contains libraries and protocols related to the serial communication of sensor data with a connected PC application.
- **Projects**: contains a sample application used to perform the Reverse Geocoding. This application is provided for the P-L496G-CELL01 and NUCLEO-F401RE platforms with three development environments (IAR Embedded Workbench for ARM, RealView Microcontroller Development Kit (MDK-ARM), and System Workbench for STM32 (SW4STM32).
- **Utilities**: contains some complementary project files.

3.4 APIs

Detailed technical information with full user API function and parameter description are in a compiled HTML file in the "Documentation" folder.

3.5 Sample application description

A sample application using the X-NUCLEO-GNSS1A1 and 2G/3G or LTE cellular expansion boards with either P-L496G-CELL01/P-L496G-CELL02 or NUCLEO-F401RE is provided in the "Projects" directory. Ready to be built projects are available for multiple IDEs.

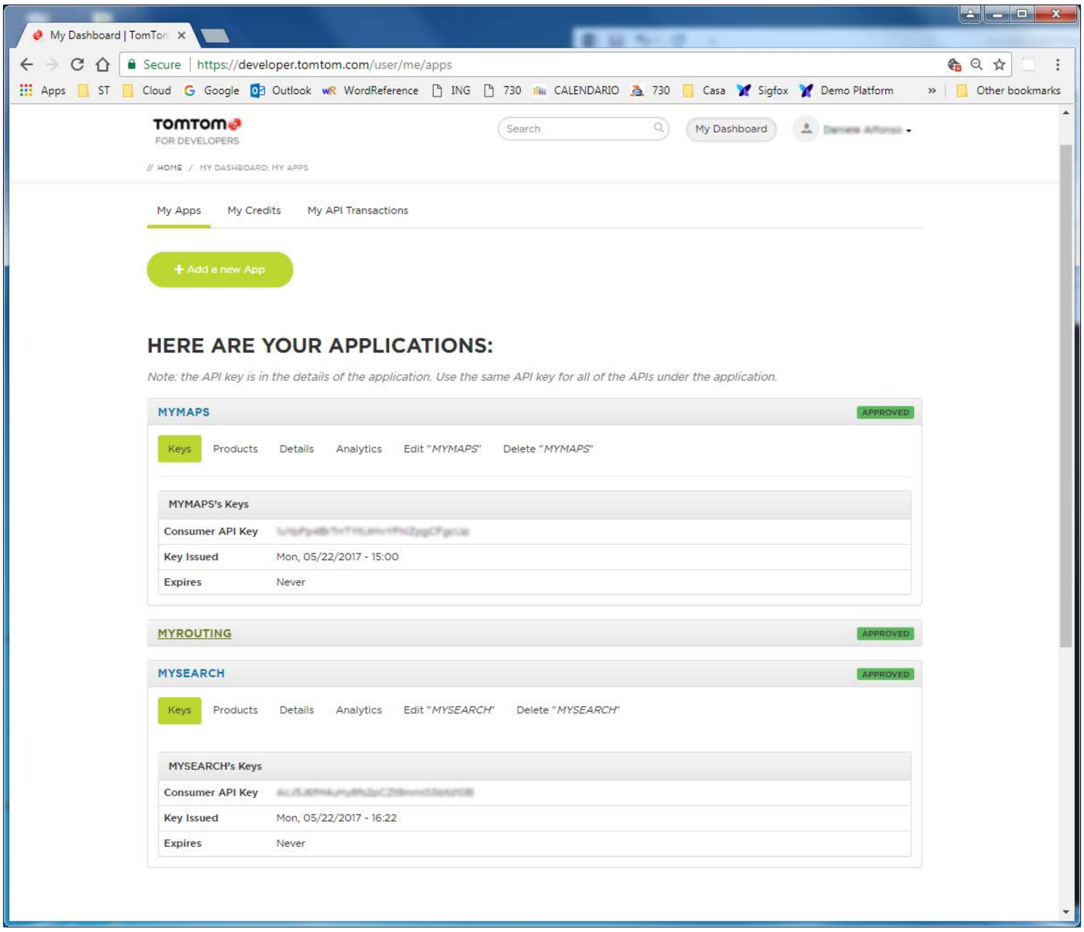
The user interface is provided via serial port and has to be configured with baud rate 115200, 8N1 parameters. It is also necessary to select "Auto" or "LF" as a new line parameter in receiver mode, and "CR+LF" in transmitter mode, and to enable the local echo.

Step 1. Activate the embedded SIM integrated in the STM32-C2C cellular expansion board, as explained in 1..

As an alternative, you can use your own SIM card through the external SIM connector of the cellular expansion board.

Step 2. Register at the TomTom developer portal (https://developer.tomtom.com) to get the search API key and the map API key to be entered in the file Projects/Common/TomTom/inc/connection.h.

Figure 4. **TomTom developer portal page showing API keys**

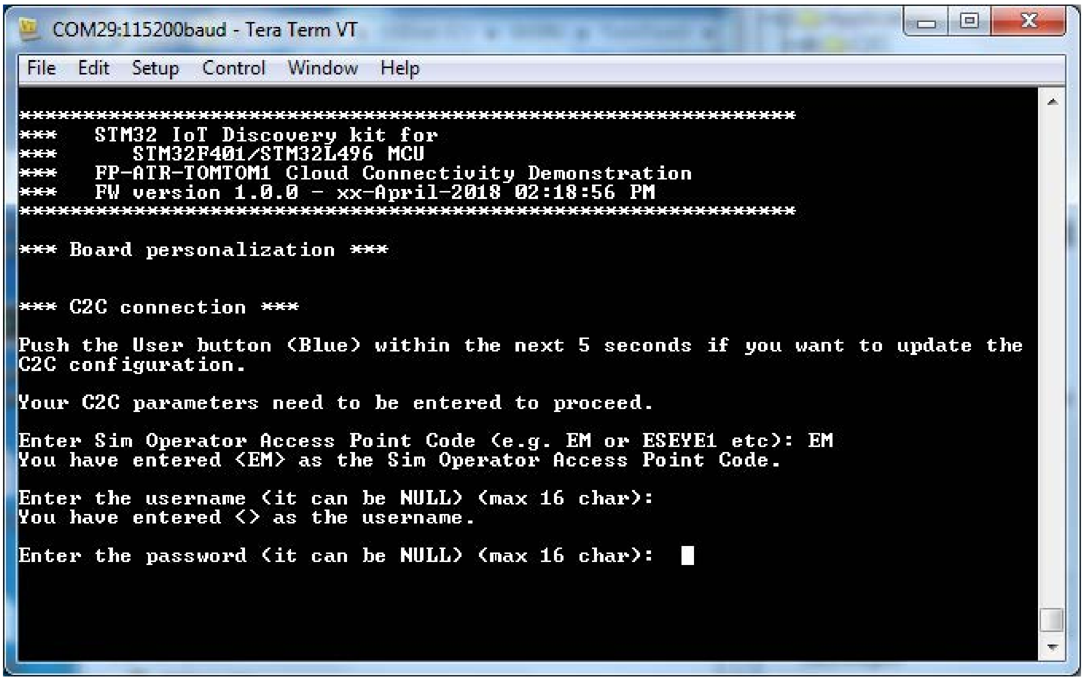


Step 3. Compile and flash the application.

The application initializes the cellular modem

Step 4. Provide the C2C (cellular-to-cloud) connection parameters: cellular SIM operator (use “EM” for the embedded SIM), username and password (blank for the embedded SIM).

Figure 5. **FP-ATR-TOMTOM1 - C2C parameters**



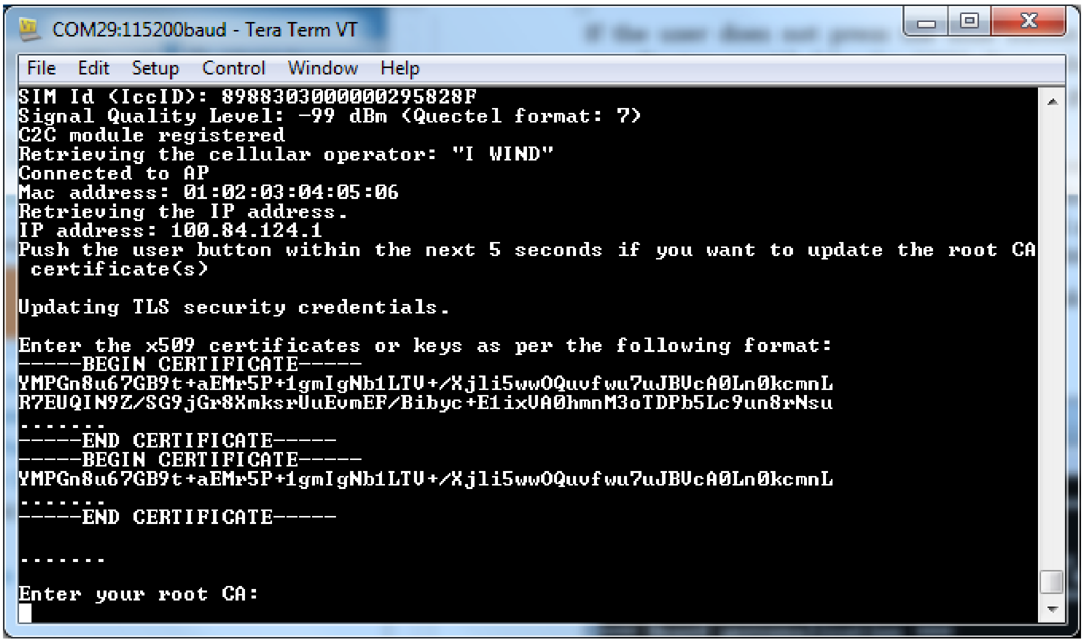
The parameters are then saved in the flash memory and recovered at each successive execution, so you do not have to enter them again.

The application connects to the cellular network.

The modem tries to use an external SIM if present, otherwise it uses the embedded SIM.

Step 5. After the cellular connection is established, enter the TLS credentials, i.e. the security certificates.

Figure 6. **FP-ATR-TOMTOM1 - TLS certificates**



The application needs the concatenation of two certificates provided in the Projects/Common/TomTom folder: the “Comodo” certificate, required for the time server and used to retrieve the current date and time, and the “TomTom” certificate to connect to the TomTom online services website.

Step 6. Copy the two certificates in a single text file and then copy and paste the whole text in the serial terminal.

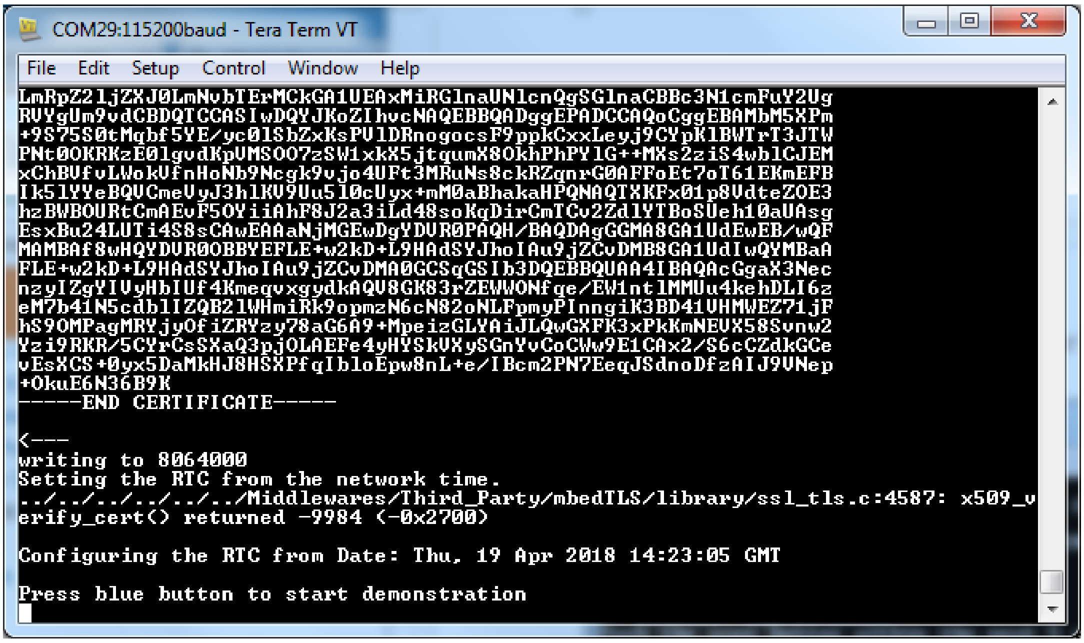
Step 7. In Tera Term, apply a transmit delay of 10 msec/char and 10 msec/line in Setup/Serial Port to avoid data loss.

The certificates are then saved in the flash memory for further use.

The application connects to the time server to update the RTC. At the first boot, the certificate verification produces an error message with code “0x2700”, which is simply ignored by the application.

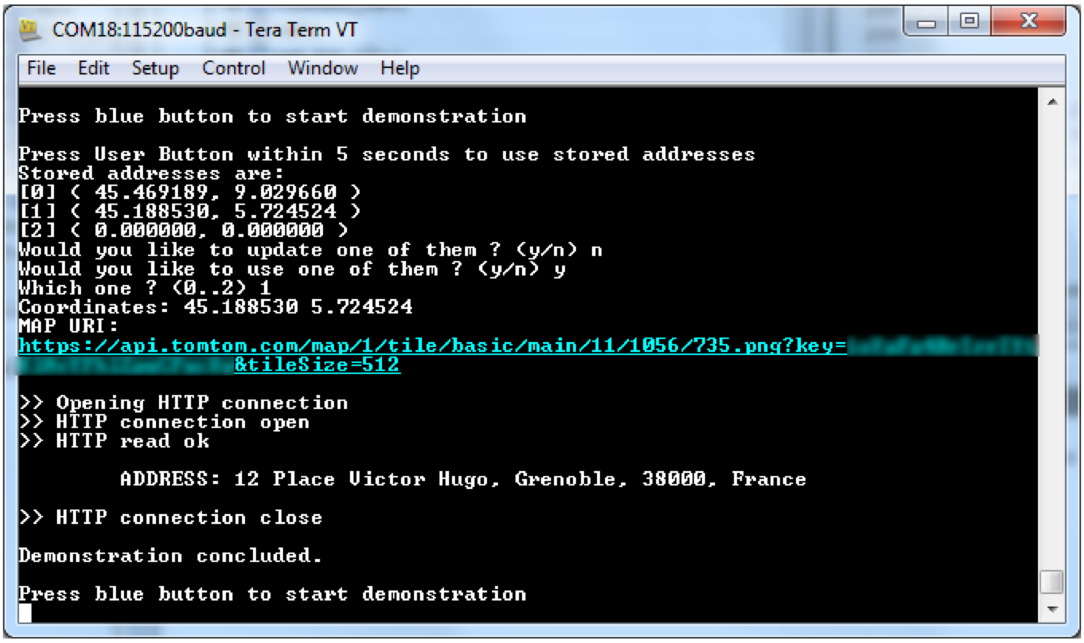
Step 8. Press the blue button to start the demonstration.

Figure 7. **FP-ATR-TOMTOM1 - user interaction request**



Step 9. Push the user button within 5 seconds to use stored addresses, a set of pre-stored GPS coordinates saved in the flash memory, which can be used to test the application in place of the coordinates provided by the GNSS receiver (useful to test the application where the GNSS cannot work, such as indoor).

Figure 8. **FP-ATR-TOMTOM1 - stored addresses to be used**

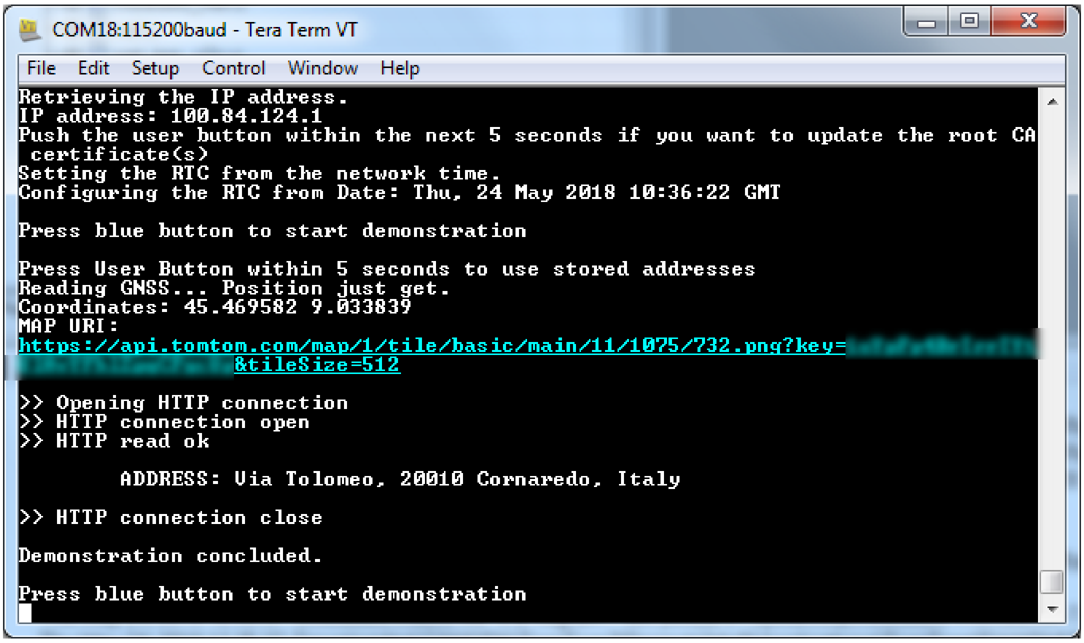


After that, the application connects to the TomTom online services, sends the GPS coordinates and receives in response the corresponding street address, which is printed on the serial console. Furthermore, the applications print on the serial console a web link to the TomTom map tile containing the GPS coordinates (see the blue link in the figure above).

Step 10. Press the button to select the coordinates among three different slots and to update the coordinates with new values.

If the you do not press the blue button within 5 seconds, the application uses the coordinates provided by the GNSS receiver.

Figure 9. **FP-ATR-TOMTOM1 - GNSS coordinates identified**



Note: Remember to connect the GNSS antenna and place it outside the window to receive a good signal. If the signal is present, the GNSS receiver provides the GPS coordinates.

After that, the application connects to the TomTom online services, sends the GPS coordinates and receives the corresponding street address, printed on the serial console. The application also prints a web link to the TomTom map tile containing the coordinates (see the blue link in the figure above).

Step 11. Open the link with any web browser to display the map tile.

Step 12. Press the blue button again for a new run, after the application stops.

4 System setup guide

4.1 Hardware description

4.1.1 STM32 Nucleo platform

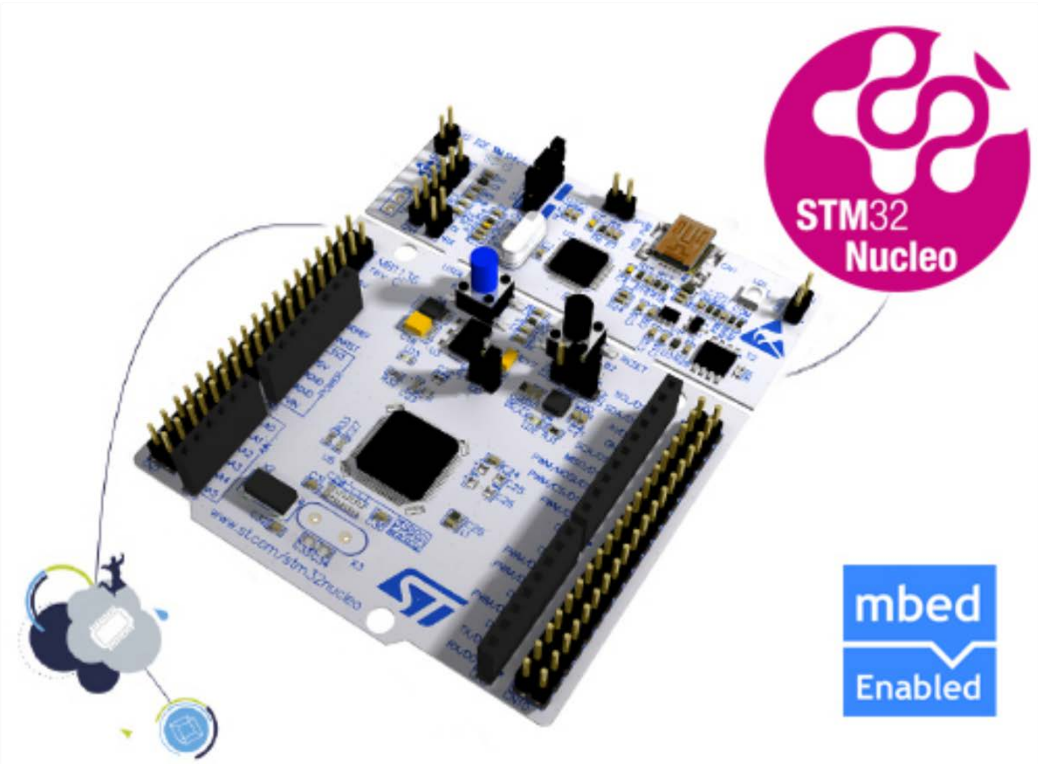
STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

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.

Figure 10. STM32 Nucleo board



Information regarding the STM32 Nucleo board is available at www.st.com/stm32nucleo

4.1.2 P-L496G-CELL01 discovery pack

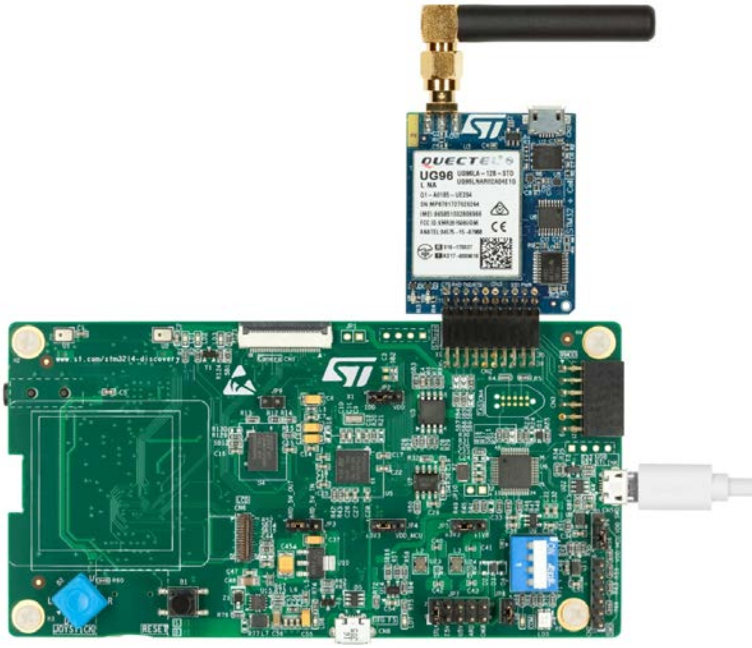
The P-L496G-CELL01 STM32 discovery pack for 2G/3G cellular to cloud (STM32-C2C/2G-3G) is a turnkey development platform for cellular and cloud technology-based solutions.

The pack is composed of an STM32L496AGI6-based low-power discovery mother board with preloaded firmware (32L496GDISCOVERY), and an STMod+ cellular expansion board with antenna.

It features STM32L496AGI6 microcontroller with 1 Mbyte of Flash memory and 320 Kbytes of RAM in a UFBGA169 package.

Board expansion features a Quectel UG96 worldwide cellular modem penta-band 2G/3G module with 7.2 Mbps downlink and 5.76 Mbps uplink.

Figure 11. P-L496G-CELL01 2G/3G cellular to cloud pack



4.1.3 P-L496G-CELL02 discovery pack

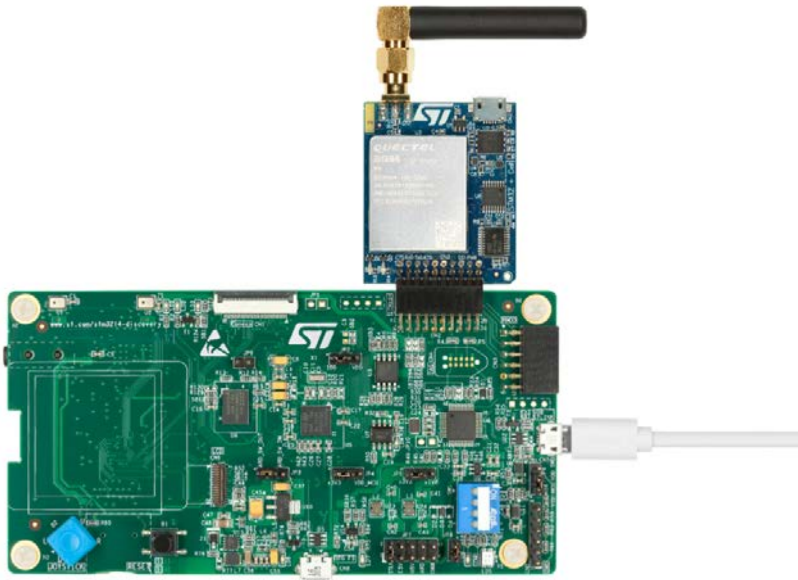
The P-L496G-CELL02 STM32 discovery pack for LTE IoT cellular to cloud (STM32-C2C/LTE IoT) is a turnkey development platform for cellular and cloud technology-based solutions.

The pack is composed of an STM32L496AGI6-based low-power discovery mother board with preloaded firmware (32L496GDISCOVERY), and an STMod+ cellular expansion board with antenna.

It features STM32L496AGI6 microcontroller with 1 Mbyte of Flash memory and 320 Kbytes of RAM in a UFBGA169 package.

Board expansion features a Quectel BG96 worldwide cellular modem LTE Cat M1/Cat NB1/EGPRS module 300 kbps downlink, 375 kbps uplink.

Figure 12. P-L496G-CELL02 LTE cellular to cloud pack



4.1.4 X-NUCLEO-GNSS1A1 expansion board

The X-NUCLEO-GNSS1A1 expansion board is based on the Teseo-LIV3F tiny GNSS module.

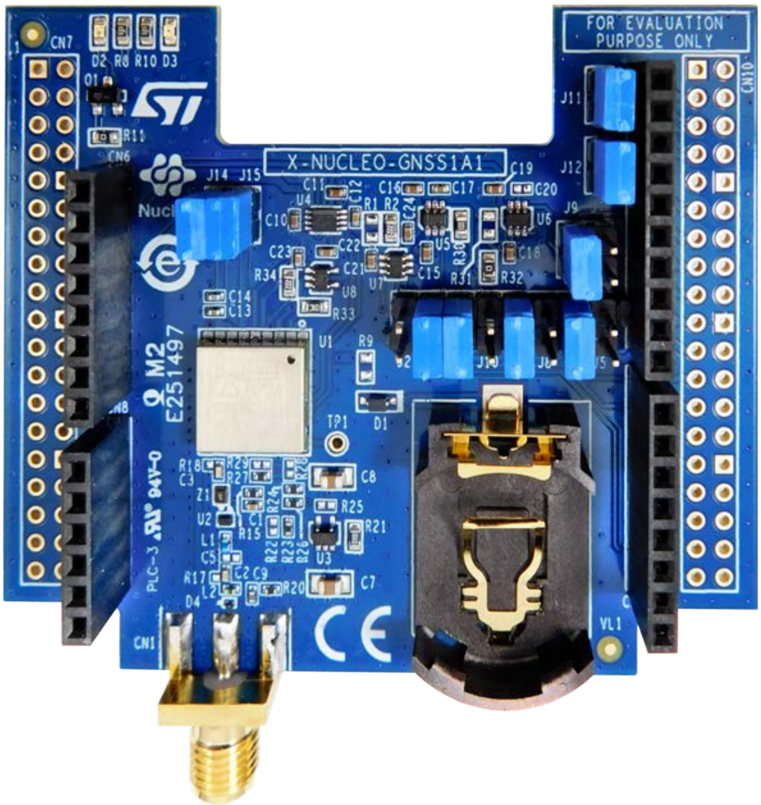
It represents an affordable, easy-to-use, global navigation satellite system (GNSS) module, embedding a TeseoIII single die standalone positioning receiver IC, usable in different configurations in your STM32 Nucleo project.

The Teseo-LIV3F is a compact (9.7x10.1 mm) module that provides superior accuracy thanks to the on-board 26 MHz temperature compensated crystal oscillator (TCXO) and a reduced time-to-first fix (TTFF) with its dedicated 32 KHz real-time clock (RTC) oscillator.

The Teseo-LIV3F module runs the GNSS firmware (X-CUBE-GNSS1) to perform all GNSS operations including acquisition, tracking, navigation and data output without external memory support.

The X-NUCLEO-GNSS1A1 expansion board is compatible with the Arduino™ UNO R3 connector and the ST morpho connector, so it can be plugged to the STM32 Nucleo development board and stacked with additional STM32 Nucleo expansion boards.

Figure 13. X-NUCLEO-GNSS1A1 expansion board



4.1.5 X-NUCLEO-STMODA1 expansion board

The X-NUCLEO-STMODA1 provides an easy way to expand your STM32 Nucleo board with the STMod+ connector, which allows interaction with the new set of STM32 Nucleo development boards using this connector. It provides an easy way to evaluate the STMod+ board solution together with other STM32 Nucleo boards.

The STMod+ is a 2x10-pin connector providing a set of interfaces such as SPI, UART, I²C and other functions such as RESET, INTERRUPT, ADC, PWM and general purpose I/Os. The X-NUCLEO-STMODA1 has a female STMod+ connector with 2 mm pitch.

The X-NUCLEO-STMODA1 expansion board is equipped with a set of jumpers for the added flexibility of allowing you to also use the board with the STM32 B-L475E-IOT01A discovery kit node board.

Figure 14. **X-NUCLEO-STMODA1 expansion board**

4.2 Hardware setup

The following hardware components are needed:

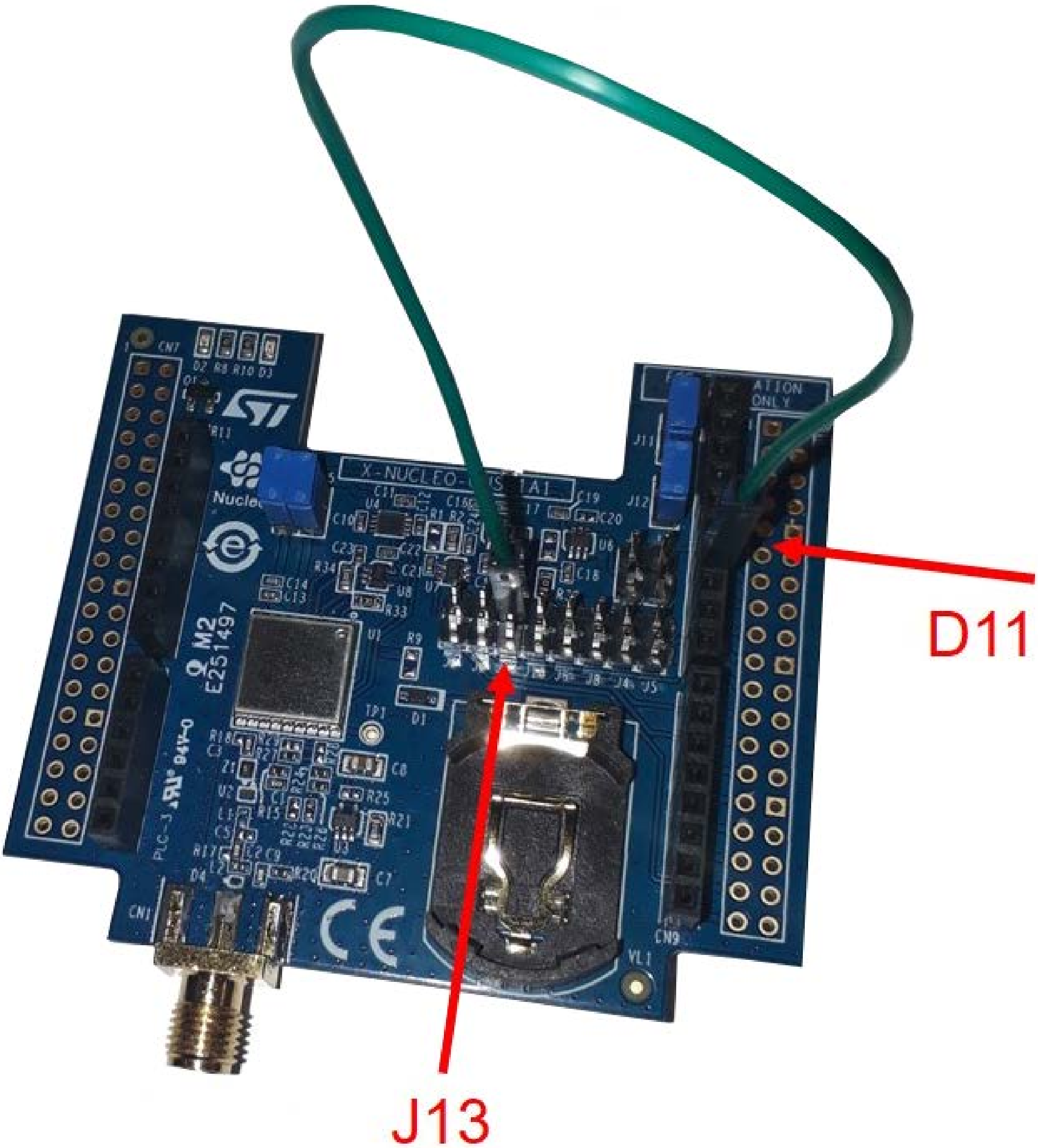
1. One STM32 discovery pack for cellular to cloud (STM32-C2C) development platform (order code: P-L496G-CELL01 for 2G/3G or P-L496G-CELL02 for LTE) or STM32 Nucleo development board (order code: NUCLEO-F401RE) with an STMod+ adapter expansion board (order code: X-NUCLEO-STMODA1) and an STM32-C2C 2G/3G or LTE cellular expansion boards (provided in the P-L496G-CELL01 or P-L496G-CELL02 packs).
2. One Teseo-LIV3F expansion board (order code: X-NUCLEO-GNSS1A1)
3. One USB type A to Mini-B (for NUCLEO-F401RE) or to Micro-B (for P-L496G-CELL01 and P-L496G-CELL02) USB cable to connect the STM32 Nucleo board to the PC.

The following jumpers settings need to be made on the boards:

1. On the X-NUCLEO-GNSS1A1 expansion board, the following jumpers must be open: J2, J3, J4, J5, J6, J7, J8, J9, J10 and J13. The following jumpers must be closed: J11, J12, J14 and J15.

   Moreover, a male-female wire must be connected to the upper pin of jumper J13 and to pin D11 of the Arduino connector, as shown below.

Figure 15. **X-NUCLEO-GNSS1A1 expansion board: wire connected between J13 and D11**



2. On the X-NUCLEO-STMODA1 expansion board, the following jumpers must be closed: JP6, JP7, JP8, JP9, JP10, JP11, JP12, and JP13. JP1 and JP2 must be in position 2-3. JP3, JP4 and JP5 must be in position 1-2.

4.3 Software setup

The following software components are required for the setup of a suitable development environment to create applications for the STM32 Nucleo board with the sensor expansion board:

- FP-ATR-TOMTOM1 firmware, available on www.st.com.
- Development tool-chain and Compiler. The STM32Cube expansion software supports the three following environments to select from:
  – IAR Embedded Workbench for ARM® (EWARM) toolchain + ST-LINK
  – RealView Microcontroller Development Kit (MDK-ARM) toolchain + ST-LINK
  – System Workbench for STM32 (SW4STM32) + ST-LINK

4.4 System setup

The STM32 Nucleo board integrates the ST-LINK/V2-1 debugger/programmer.

The developer can download the ST-LINK/V2-1 USB driver by looking at the STSW-LINK009 software on www.st.com.

The X-NUCLEO-GNSS1A1 epansion board can be easily connected to the STM32 Nucleo through the Arduino UNO R3 extension connector. The board interfaces with the external STM32 microcontroller on STM32 Nucleo using inter-integrated circuit (I²C) transport layer.

The Quectel UG96 and BG96 modem expansion board can be directly connected to the 32L496GDISCOVERY board through the STMod+ port.

When using a NUCLEO-F401RE development board, an STMod+ adapter expansion board (X-NUCLEO-STMODA1) is needed. The board interfaces with the external STM32 microcontroller on the STM32 Nucleo using universal asynchronous receiver-transmitter (UART) connection.

A References

Freely available on www.st.com:

1. UM2347 *Getting started with X-CUBE-CLD-GEN IoT cloud generic software expansion for STM32Cube*
2. UM2322 *STM32 Discovery pack for 2G/3G cellular to cloud*
3. UM2327 *Getting started with the X-NUCLEO-GNSS1A1 expansion board based on Teseo-LIV3F tiny GNSS module for STM32 Nucleo*
4. UM2400 *Getting started with the X-NUCLEO-STMODA1 expansion board for STM32 Nucleo*

Revision history

Table 2. Document revision history

Date	Version	Changes
05-Sep-2018	1	Initial release.
11-Oct-2018	2	Updated Introduction, Section 3.1 Overview, Figure 2. FP-ATR-TOMTOM1 software architecture and Section 4.2 Hardware setup. Added Section 4.1.3 P-L496G-CELL02 discovery pack.
04-Feb-2019	3	Added Figure 15. X-NUCLEO-GNSS1A1 expansion board: wire connected between J13 and D11.

Contents

List of tables

List of figures