# How to use STMicroelectronics' X-CUBE-TOF1 Time-of-Flight sensor software packages for STM32CubeMX

## Introduction

The X-CUBE-TOF1 expansion software package for STM32Cube runs on the STM32. It includes drivers that recognize the sensors, and it performs simple ranging on single or multiple devices.

The expansion is built on STM32Cube software technology to ease portability across different STM32 microcontrollers.

The software comes with a sample implementation of the drivers, running on different Time-of-Flight sensor evaluation boards, connected to a featured STM32 Nucleo development board.

This user manual gives an overview of the use of the STM32CubeMX with the following Time-of-Flight ranging sensors: VL53L4CX, VL53L4CD, VL53L7CX, and VL53L8CX. For further information on the Time-of-Flight sensors supported by the X-CUBE-TOF1, please refer to the software page on www.st.com.

The evaluation boards supported by the X-CUBE-TOF1 expansion software package include:

- X-NUCLEO expansion board
- SATEL breakout board

The X-CUBE-TOF1 software provides some sample applications for the Time-of-Flight sensors listed above.

Visit the STM32Cube ecosystem web page on www.st.com for further information.

# 1 Acronyms and abbreviations

| Acronym | Definition |
|---|---|
| API | application programming interface |
| BSP | board support package |
| HAL | hardware abstraction layer |
| $I^2C$ | inter-integrated circuit |
| IDE | integrated development environment |
| MCU | microcontroller unit |
| NVIC | nested vector interrupt control |
| PCB | printed circuit board |
| SDK | software development kit |
| ToF | Time-of-Flight sensor |
| USB | universal serial bus |

# 2 X-CUBE-TOF1 software expansion for STM32Cube

## 2.1 Overview

The X-CUBE-TOF1 software package expands the STM32Cube functionality. The key features are:

- Complete software to build applications using the evaluation boards listed in the "Introduction".
- Several application examples to show the innovative technology for the accurate distance ranging capability.
- Sample application to transmit real-time sensor data to a PC.
- Precompiled binaries available on all evaluation boards (listed in the "Introduction") connected to a NUCLEO-F401RE or NUCLEO-L476RG development board.
- Package compatible with STM32CubeMX, can be downloaded from, and installed directly into, STM32CubeMX.
- Easy portability across different MCU families, thanks to the STM32Cube.
- Free, user-friendly license terms.

## 2.2 Architecture

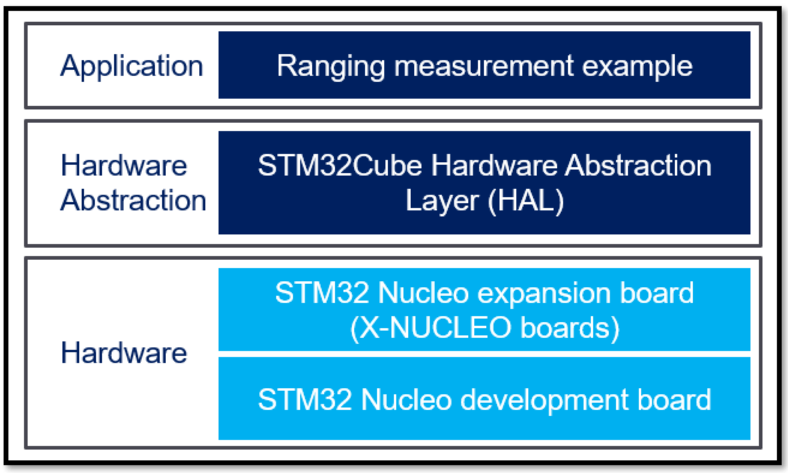
This software is a fully compliant expansion of the STM32Cube. It enables the development of applications using Time-of-Flight sensors.

The software is based on the hardware abstraction layer for the STM32 microcontroller, STM32CubeHAL. The package extends the STM32Cube by providing a board support package (BSP) for the sensor expansion board, and a sample application for serial communication with a PC.

The software layers used by the application software to access the sensor expansion board are:

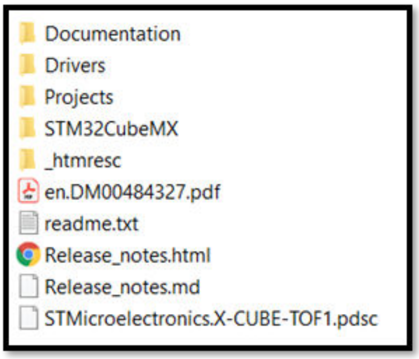
- The STM32Cube HAL driver layer. It provides a simple, generic, and multi-instance set of APIs (application programming interfaces) to interact with the upper layers (application, libraries, and stacks). It includes generic and extension APIs and is based on a generic architecture. This allows the layers that are built on it (such as the middleware layer) to implement their functionalities without dependence on the specific hardware configuration of a given microcontroller unit (MCU). This structure improves library code reusability and guarantees high portability across other devices.
- The BSP layer. It provides supporting software for the peripherals on the STM32 Nucleo board, except for the MCU. It has a set of APIs to provide a programming interface for certain board-specific peripherals (for example, the LED and the user button). The BSP layer allows identification of the specific board version. For the sensor expansion board, it provides the programming interface for various Time-of-Flight sensors, and support for initializing and reading sensor data.

**Figure 1. X-CUBE-TOF1 software architecture**



## 2.3 Folder structure

Figure 2. **X-CUBE-TOF1 package folder structure**




The following folders are included in the software package:

- The Documentation folder contains the current user manual and detailed documentation regarding the software components and APIs. In the .chm file, you can find a list of pin resources used for the XNUCLEO. It is updated when a new board is released.
- The Drivers folder contains:
  – HAL drivers
  – Board-specific drivers for each supported board or hardware platform. This includes drivers for the on-board components and the CMSIS layer (which is a vendor-independent hardware abstraction layer for the Cortex®-M processor series).
- The Projects folder contains several examples and applications for NUCLEO-L476RG and NUCLEO-F401RE platforms. They show the use of sensor APIs provided with three development environments:
  – IAR Embedded Workbench for Arm
  – MDK-ARM® microcontroller development kit
  – STM32CubeIDE
- The STM32CubeMX folder contains all the templates used by the CubeMX ToF pack.

## 2.4 APIs

Detailed technical information about the APIs available to the user can be found in the compiled HTML file "X-CUBE-TOF1.chm". This file is in the Documentation folder of the software package. All the functions and parameters are fully described.

# 3 System setup guidelines

## 3.1 Hardware description

In this section, the VL53L7CX hardware is used as an example. For each sensor listed in the introduction, the same hardware is available.

### 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.

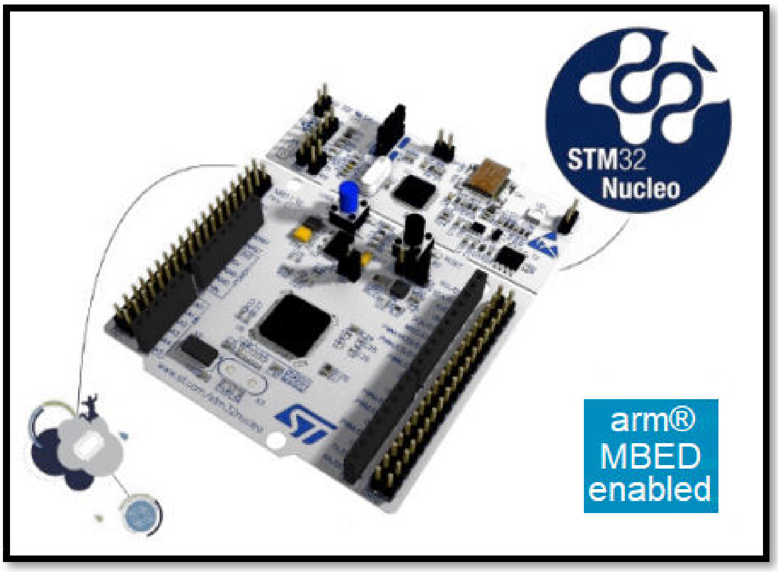
The Arduino® connectivity support and ST morpho connectors make it easy to expand the functionality of the STM32 Nucleo open development platform. They have a wide range of specialized expansion boards to choose from. The STM32 Nucleo board does not require separate probes as it integrates the ST-LINK/V2-1 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 Arm

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 3. STM32 Nucleo board**




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

### 3.1.2 X-NUCLEO expansion boards

The X-NUCLEO expansion boards are for use with any Nucleo 64 development board. Figure 4. Example of an X-NUCLEO-53L7A1 expansion board with a cover glass is an example of the X-NUCLEO-53L7A1 expansion board, with a cover glass and spacers. They provide a complete evaluation kit allowing anyone to learn, evaluate, and develop their applications using the Time-of-Flight ranging sensor.

The X-NUCLEO expansion boards are delivered with a cover glass holder. It contains three different spacers of 0.25, 0.5, and 1 mm height. They can be fitted below the cover glass to simulate various air gaps.

The X-NUCLEO expansion boards are compatible with the STM32 Nucleo board family, and with the Arduino® UNO R3 connector layout.

Several STMicroelectronics' X-NUCLEO expansion boards can be superposed through the Arduino connectors. It allows the development of applications with Bluetooth or Wi-Fi interfaces.

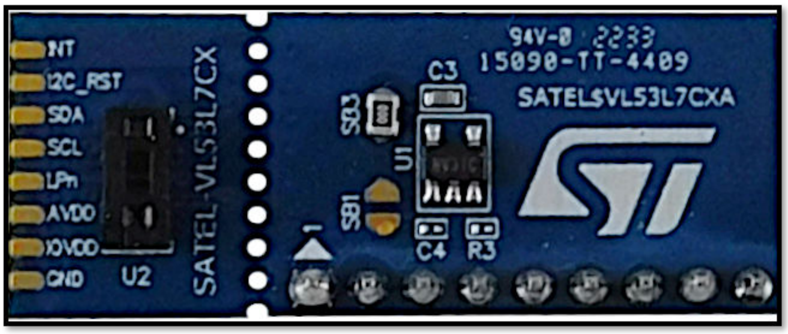
**Figure 4. Example of an X-NUCLEO-53L7A1 expansion board with a cover glass**

### 3.1.3 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 a 2V8 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 5. Example of a SATEL-VL53L7CX breakout board**




## 3.2 Software description

The following software components are required to establish a suitable development environment for creating applications for the STM32 Nucleo equipped with the sensor expansion board:

- X-CUBE-TOF1: an STM32Cube expansion for sensor application development. The X-CUBE-TOF1 firmware and associated documentation is available on www.st.com.
- Development tool-chain and compiler: The STM32Cube expansion software supports the following three environments:
  - IAR Embedded Workbench for Arm(EWARM) toolchain + STLINK
  - RealView microcontroller development kit (MDK-ARM-STR) toolchain + STLINK
  - STM32CubeIDE for STM32 + STLINK

## 3.3 Hardware setup

The following hardware components are required:

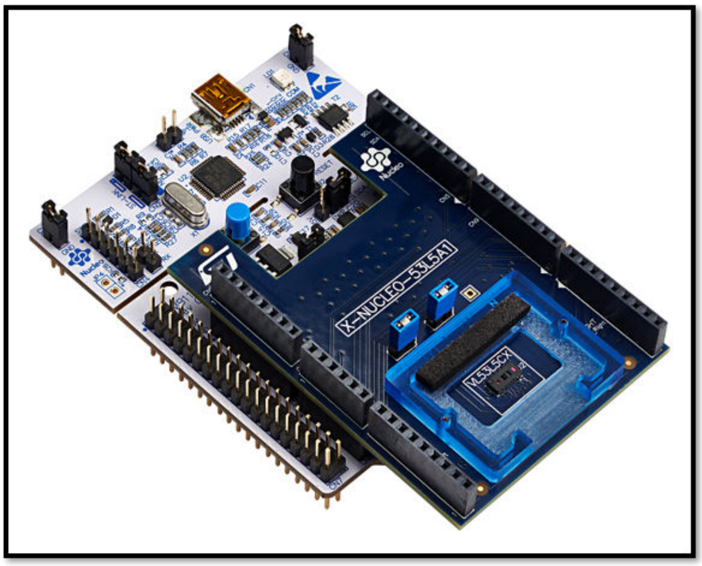
1. One STM32 Nucleo development platform (suggested order code: NUCLEO-F401RE or NUCLEO-L476RG)
2. An X-NUCLEO expansion board or a SATEL breakout board
3. One USB type A to mini-B USB cable to connect the STM32 Nucleo to a PC

### 3.3.1 Setup using an STM32 Nucleo and an X-NUCLEO expansion board

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

The X-NUCLEO expansion boards can be easily connected to the STM32 Nucleo board through the Arduino UNO R3 extension connector. They can also interface with the external STM32 microcontroller (on the STM32 Nucleo) via the inter-integrated circuit (I²C) transport layer.

**Figure 6. Sensor expansion board plugged to STM32 Nucleo board**



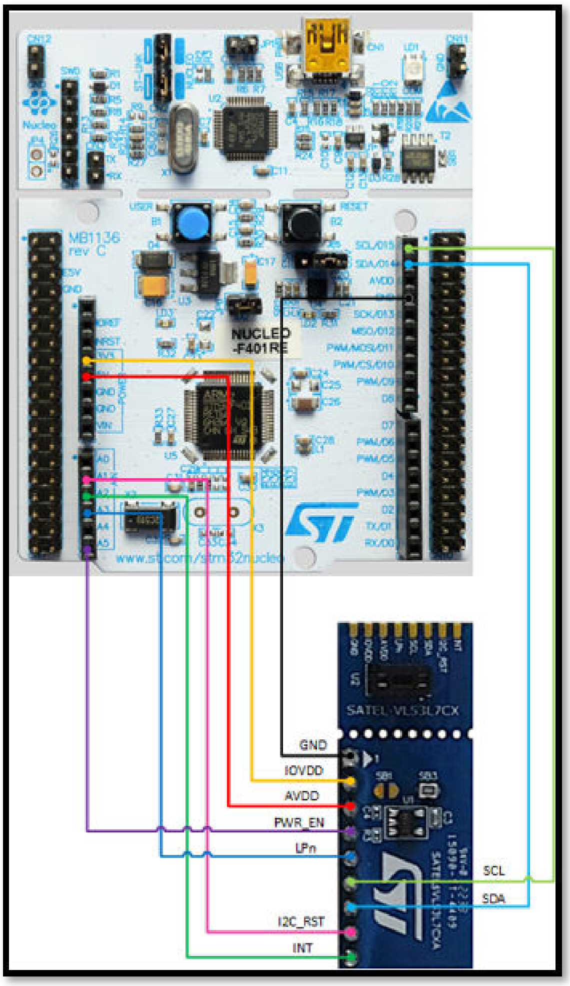
### 3.3.2 Setup using an STM32 Nucleo board and a SATEL breakout board

The SATEL breakout boards can be connected to the STM32 Nucleo board through flying wires. They can interface with the external STM32 microcontroller on the STM32 Nucleo board via the I²C transport layer.

Figure 7. Example of the connection between an STM32 Nucleo board and a SATEL shows the connection between the SATEL-53L7CX and the STM32 Nucleo board.

Refer to the Time-of-Flight application notes on st.com, which explain how to connect each product SATEL to the Nucleo board.

**Figure 7. Example of the connection between an STM32 Nucleo board and a SATEL**



# 4 Sample application examples

The X-CUBE-TOF1 package comes with several examples for each sensor. For each example, you can either:

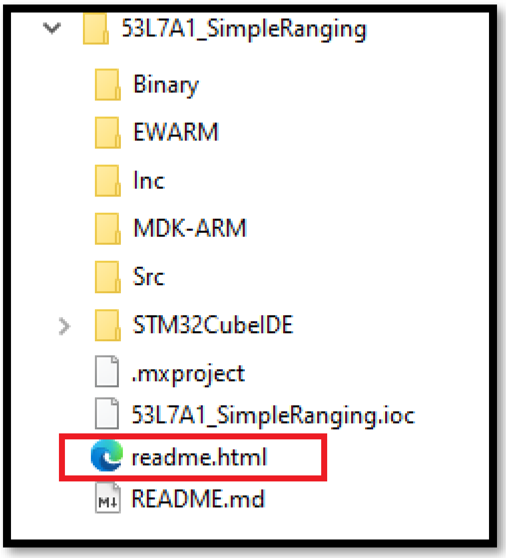
- use the precompiled file delivered in the binary folder, or
- configure the project to build the binary file used to configure the system.

This section describes two ways of running a sampling application using the evaluation boards and a STM32F401RE Nucleo board.

The same method can be used for each sensor listed in the introduction.

Note that information concerning the examples is in the readme.html file, which is delivered with the examples.

Figure 8. **SimpleRanging folder architecture**

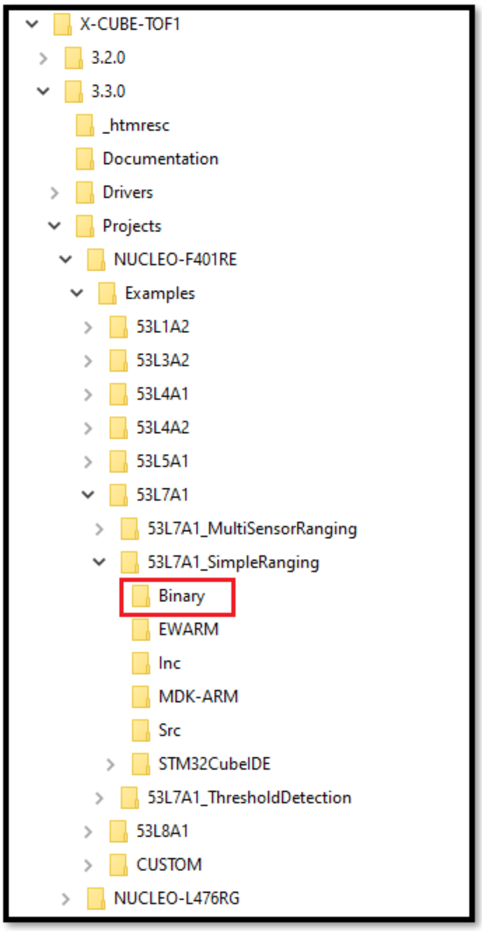


## 4.1 SimpleRanging singular form using an STM32 Nucleo board and an X-Nucleo expansion board

### 4.1.1 Loading a prebuilt binary file

Figure 9. NUCLEO-F401RE folder architecture with an X-NUCLEO expansion board shows the folder architecture for the X-CUBE-TOF1 package examples. It is based on version 3.3.0 of the X-CUBE-TOF1, but can be used for all versions.

Figure 9. NUCLEO-F401RE folder architecture with an X-NUCLEO expansion board

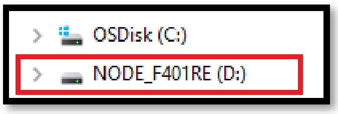



The binary file is placed in the "Binary" folder (see Figure 9. NUCLEO-F401RE folder architecture with an X-NUCLEO expansion board).

Follow the steps below to run a SimpleRanging on a VL53L7CX sensor using the precompiled file:

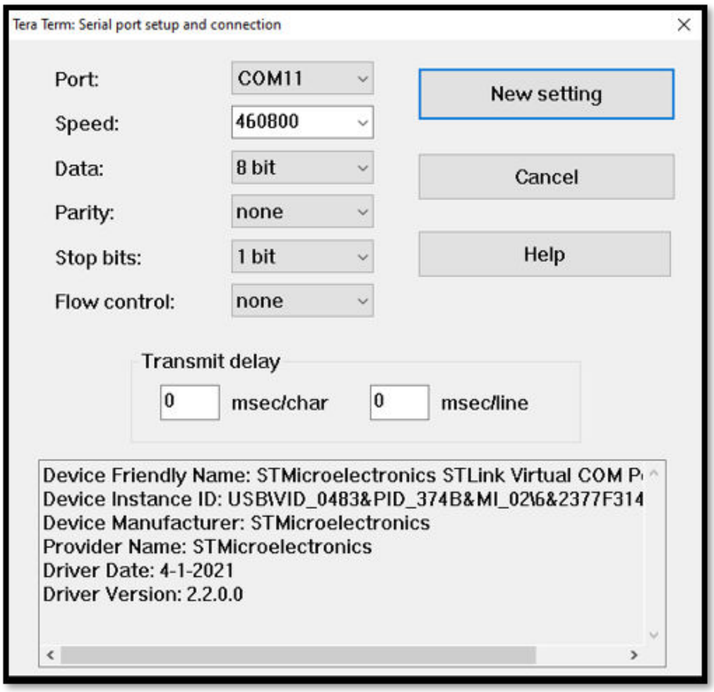
1. Set up the hardware (see Section 3.3.1 Setup using an STM32 Nucleo and an X-NUCLEO expansion board).
2. Flash the NUCLEO-F401RE board with the prebuilt binary by dragging and dropping the binary file onto the NODE drive. You can call the disk name “NOD” or “NODE” depending on the Nucleo board variant.

Figure 10. NODE drive



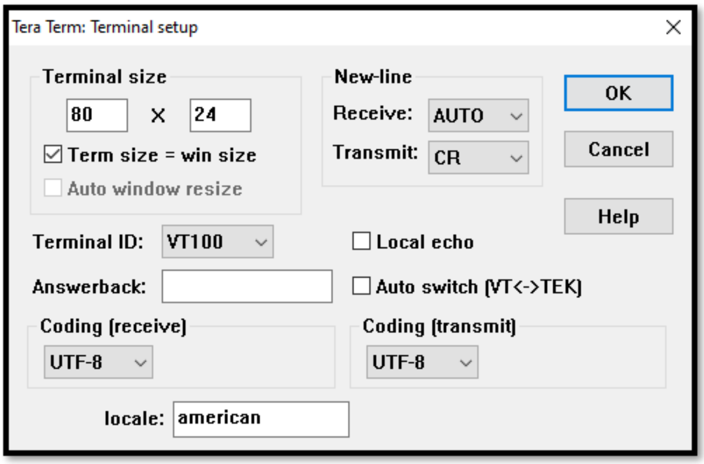
3. Open a Tera Term and configure the settings.

**Figure 11. Tera Term serial port setup**



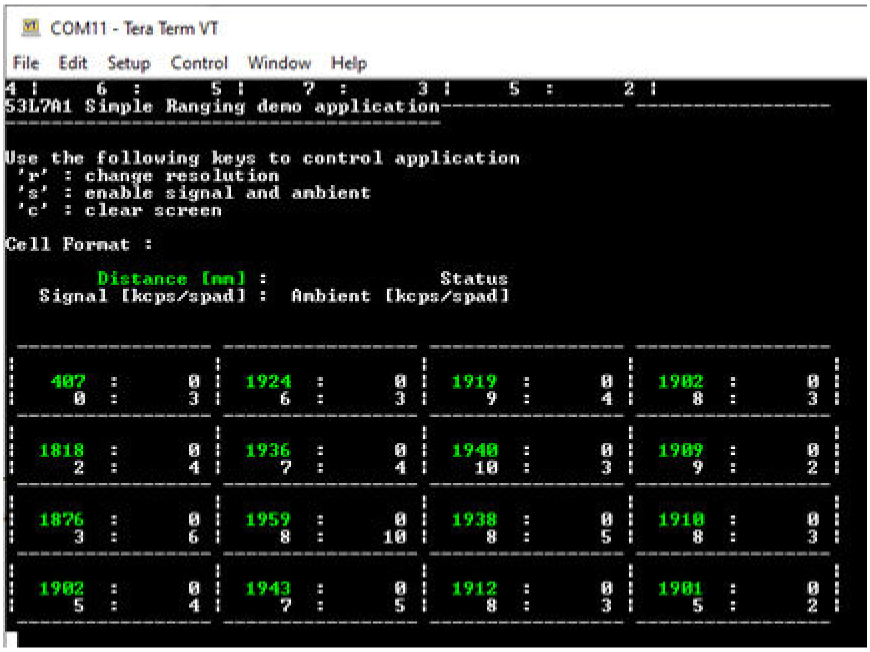

Note that speed is defined in the stm32f4xx_nucleo.c file as "Init.BaudRate = 460800". Speed is also defined in the readme.html file in the examples directory (Figure 8. SimpleRanging folder architecture).

**Figure 12. Tera Term terminal setup**



4. Wave your hand in front of the sensor to display the ranging data on the serial terminal, as shown below.

**Figure 13.** **VL53L7CX ranging results**




5. Use the same method for the different X-CUBE-TOF1 package examples present in the “Examples” folder.

### 4.1.2 Configuring the STM32CubeMX project

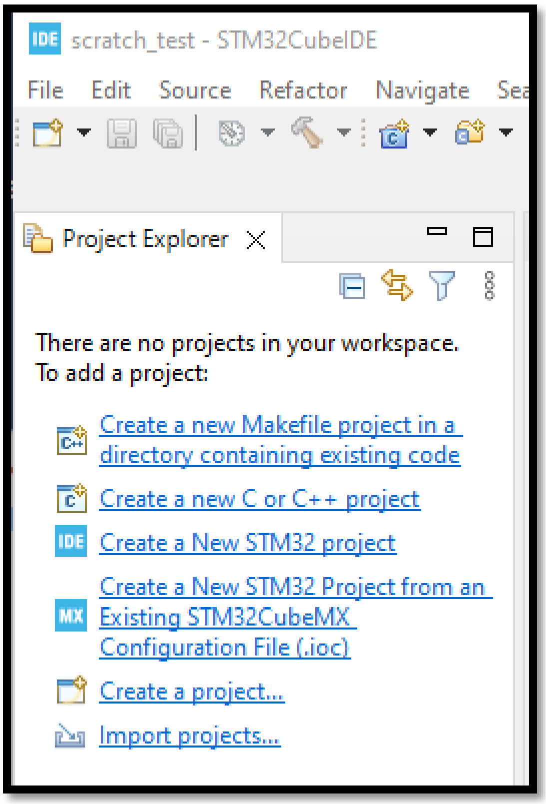
The following two examples are presented:

- SimpleRanging
- ThresholdDetection

#### *4.1.2.1 How to generate the SimpleRanging example with the STM32CubeMX*

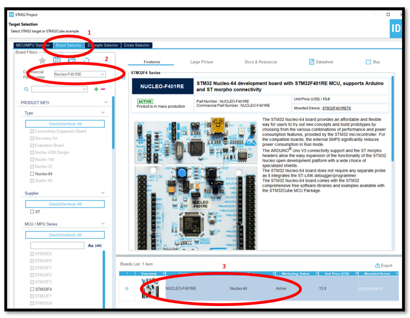
1. Create a new STM32 project.

**Figure 14. Create a new STM32 project**



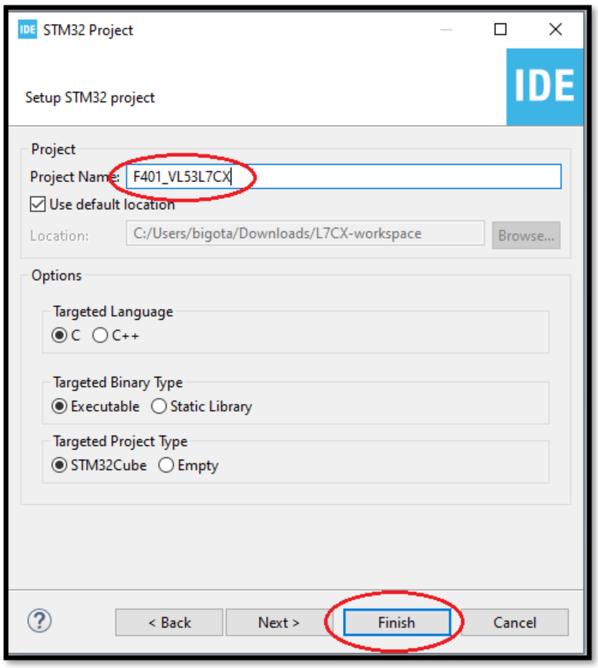

2. Open STM32CubeIDE and follow the instructions to create a workspace area.
3. Select the NUCLEO-F401RE board using the “Board selector” field.

**Figure 15. Choose the NUCLEO-F401RE board**



4. Give the project a name and a location.

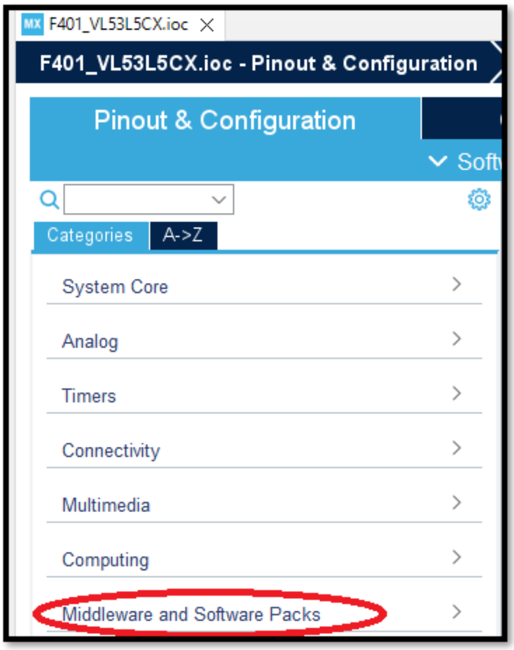
Figure 16. **Give the project name**




Click "Finish" and choose the "default configuration" when it pops up.
The purpose is now to choose the right software pack.

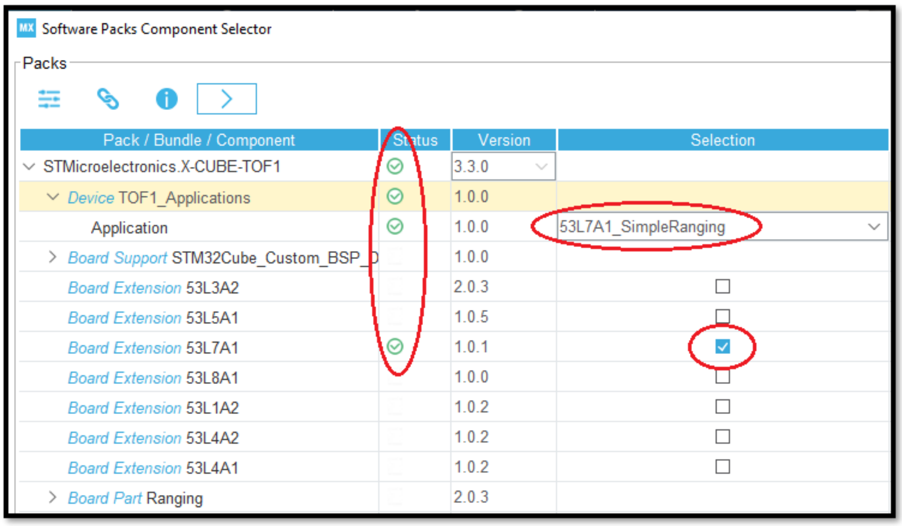
5. Click on "Middleware and Software Packs".

Figure 17. **Middleware and software packs selection**



6. Click on “X-CUBE-TOF1”. Choose the “53L7A1 board”, then the “53L7A1_SimpleRanging” application, then verify the green ticks.

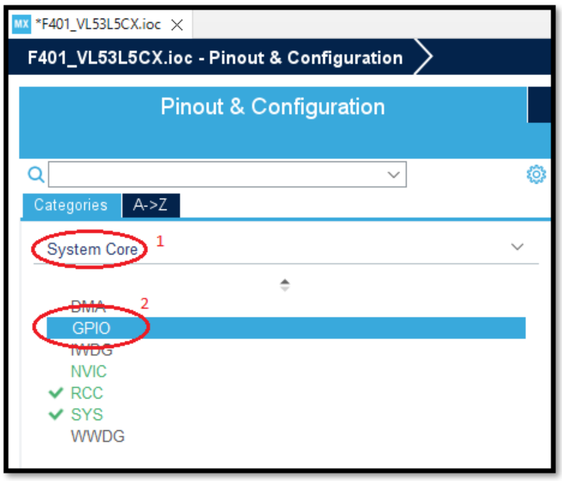
**Figure 18. Board and application choices**




The next step is to configure the hardware.

7. Configure the GPIOs for the application. Click on “System Core” then “GPIO”.

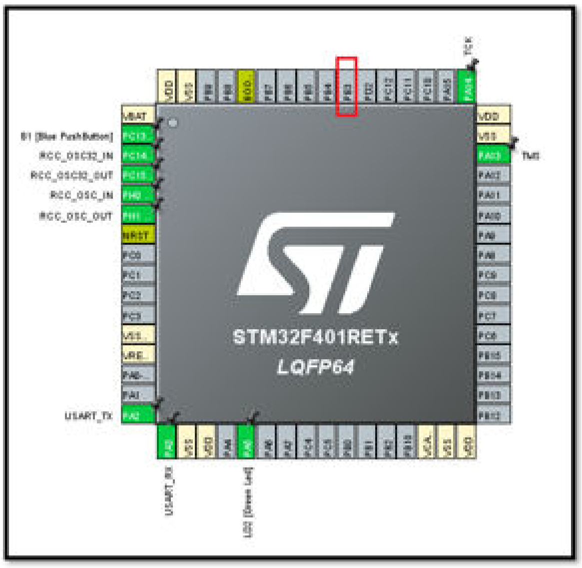
**Figure 19. GPIO selection**




For this particular application, configure the I2C_RST, LPn, PWR_EN, and INT pins.

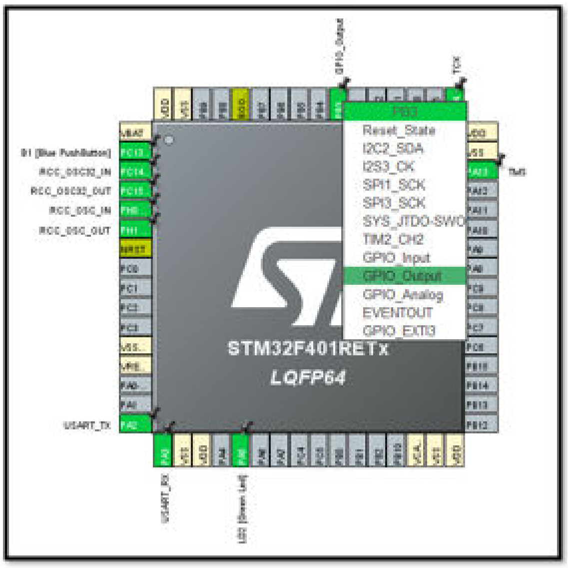
8. Select the “PB3” pin by clicking on it.

Figure 20. **PB3 pin selection**



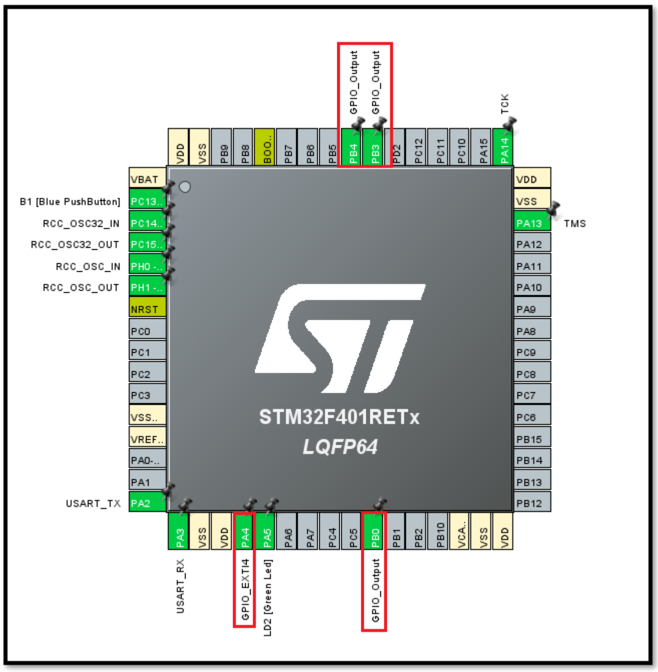

9. Make the PB3 a “GPIO_Output".

Figure 21. **GPIO_output choice**



10. Do the same for PB0, PB4, and PA4.

**Figure 22. PA4, PB0, and PB4 GPIOs selection**




PB0 and PB4 have to be defined as “GPIO_output”. PA4 has to be defined as “GPIO_EXT4”. Consequently, add the GPIOs to the GPIO table as shown Figure 23. GPIO table.

**Figure 23. GPIO table**

GPIO | RCC | SYS | USART | NVIC

Search Signals

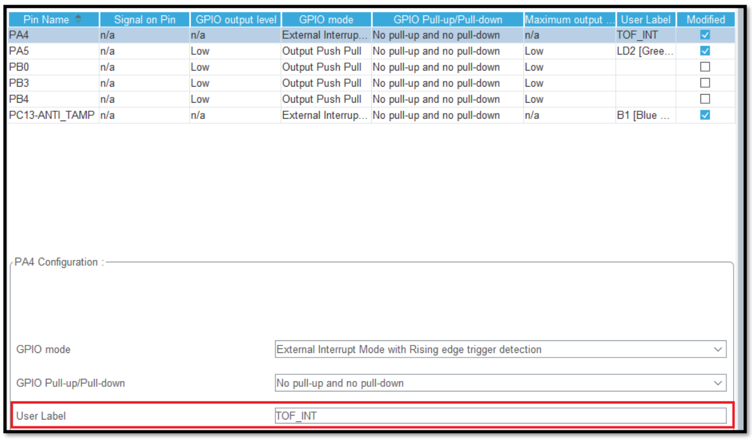
Search (Ctrl+F)

☐ Show only Modified Pins

| Pin Name | Signal on Pin | GPIO output level | GPIO mode | GPIO Pull-up/Pull-down | Maximum output ... | User Label | Modified |
|---|---|---|---|---|---|---|---|
| PA4 | n/a | n/a | External Interrup... | No pull-up and no pull-down | n/a | | ☐ |
| PA5 | n/a | Low | Output Push Pull | No pull-up and no pull-down | Low | LD2 [Gree... | ☑ |
| PB0 | n/a | Low | Output Push Pull | No pull-up and no pull-down | Low | | ☐ |
| PB3 | n/a | Low | Output Push Pull | No pull-up and no pull-down | Low | | ☐ |
| PB4 | n/a | Low | Output Push Pull | No pull-up and no pull-down | Low | | ☐ |
| PC13-ANTI_TAMP | n/a | n/a | External Interrup... | No pull-up and no pull-down | n/a | B1 [Blue ... | ☑ |

11. Next, configure the GPIO output level, GPIO mode, GPIO pull-up/pull-down, and the user label. Click on PA4 and give it a “User Label".

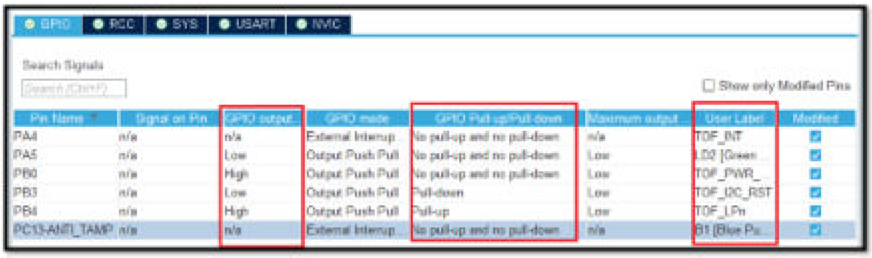
**Figure 24. PA4 configuration**




Do the same for the other GPIOs as per Figure 25. GPIO pin names and configuration. For each GPIO, configure the:

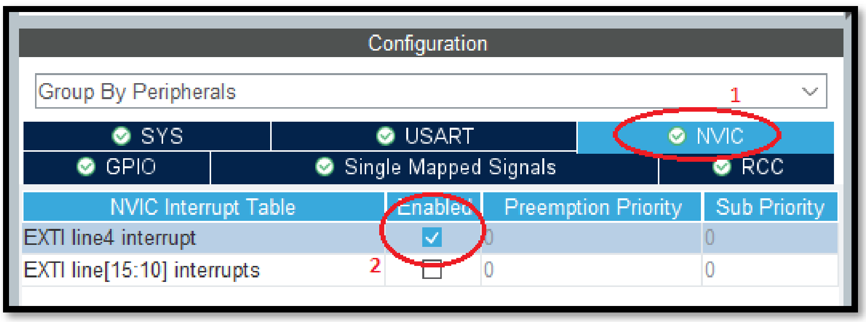
- GPIO output level
- GPIO pull-up/pull-down
- user label

**Figure 25. GPIO pin names and configuration**



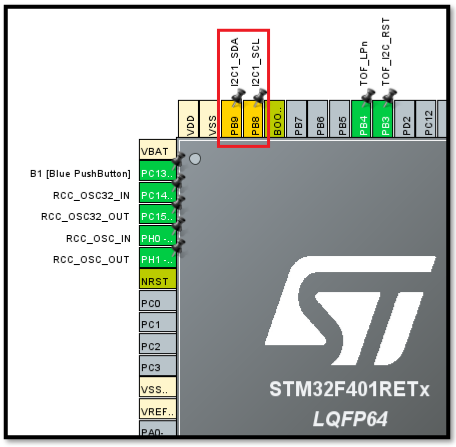

12. Activate the NVIC interrupt vector as shown below by clicking on “NVIC” and checking “Enabled".

**Figure 26. NVIC interrupt activation**



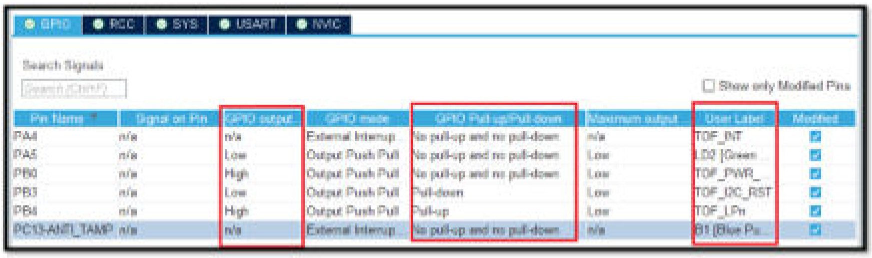
13. Next, configure the I²C. Click on PB9 and change it to “I2C1_SDA”. Click on PB8 and change it to “I2C1_SCL".

**Figure 27.** **PB9 and PB8 selection**



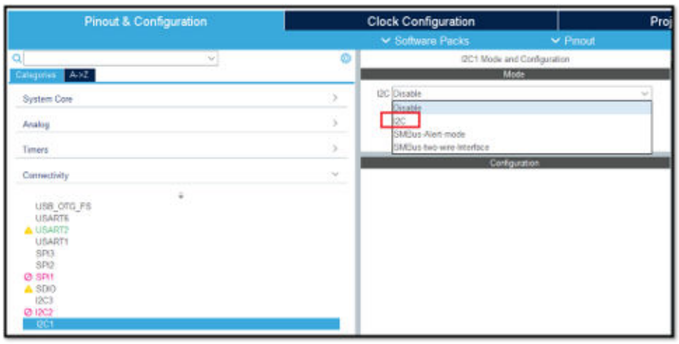

14. Click on "Connectivity" and then on "I2C1".

**Figure 28.** **Connectivity and I2C1 selection**



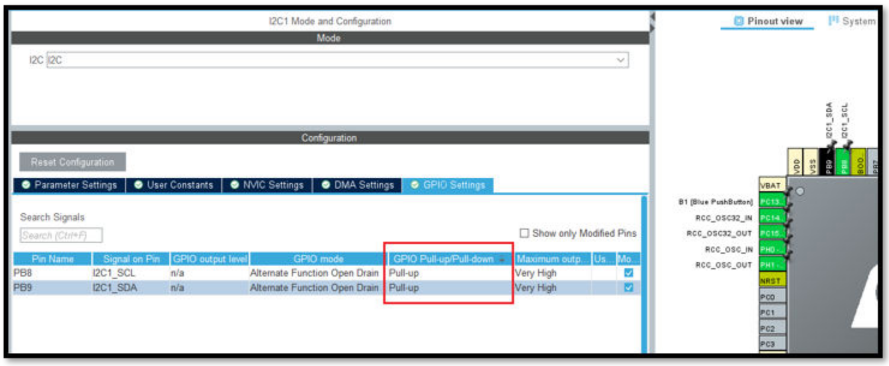

15. Enable I2C1 by clicking on “I²C".

**Figure 29.** **I2C1 enabling**



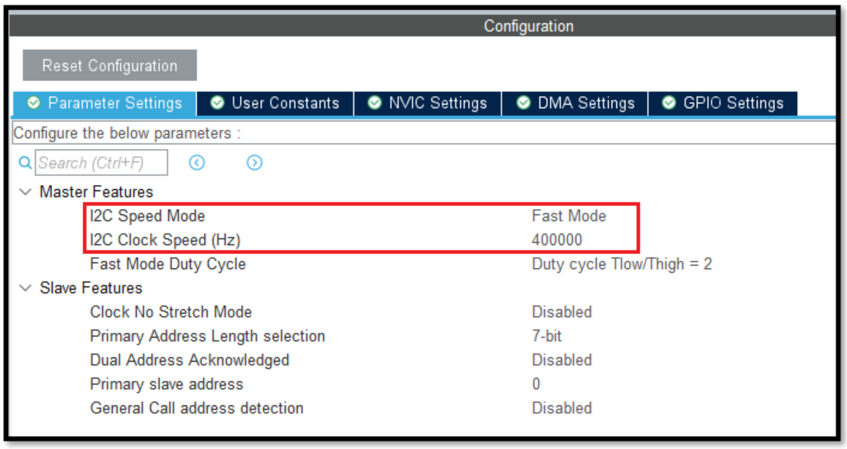
16. Set SCL and SDA in “Pull-up".

Figure 30. **Pull-up SDA and SCL**



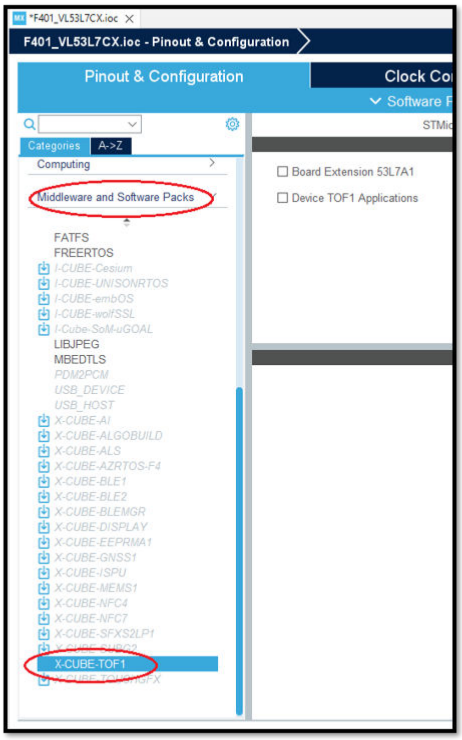

17. Configure the I²C speed in “Fast Mode".

Figure 31. **I²C speed mode configuration**



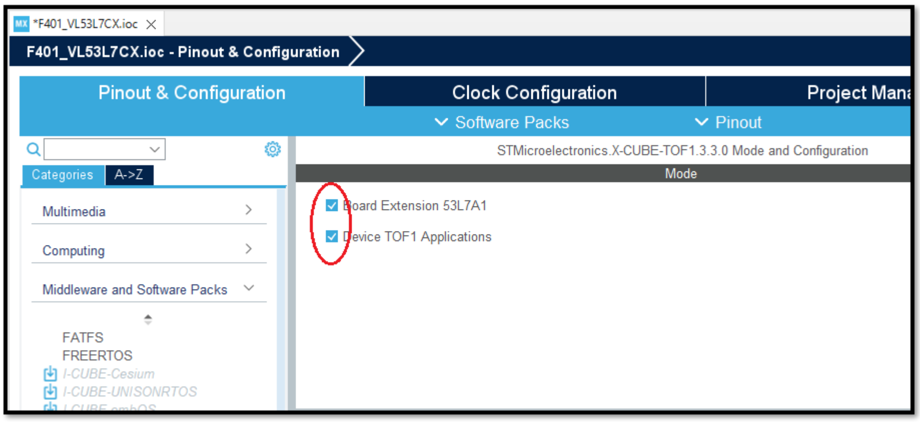
18. Next, tell the application where the dedicated pins are. Click on “Middleware and Software Packs”, then on “X-CUBE-TOF1”, and choose “Platform Settings".

**Figure 32. Platform settings**



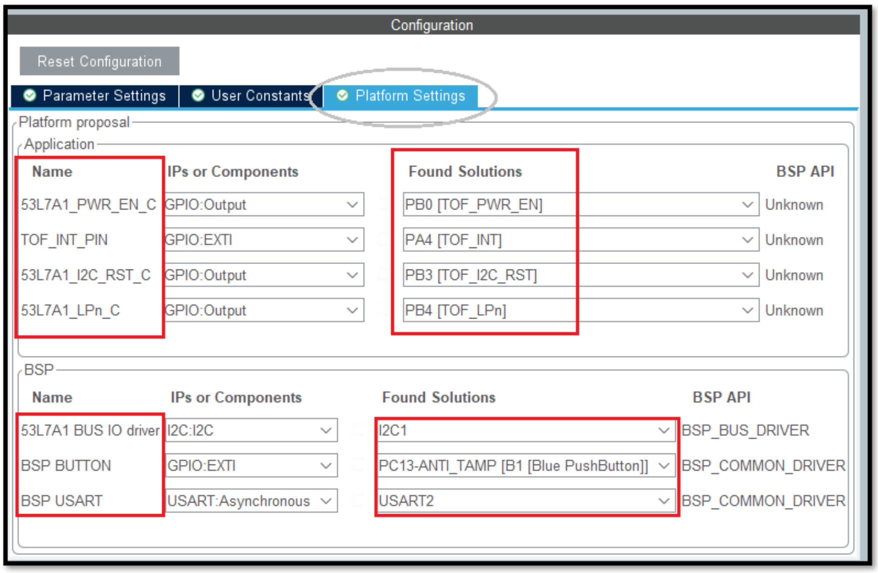

19. Choose “Board Extension” and “Device TOF1 Applications".

**Figure 33. Board and application choice**



20. Click on “Platform Settings” and assign pins as per Figure 34. Hardware and software pin assignment.

**Figure 34. Hardware and software pin assignment**

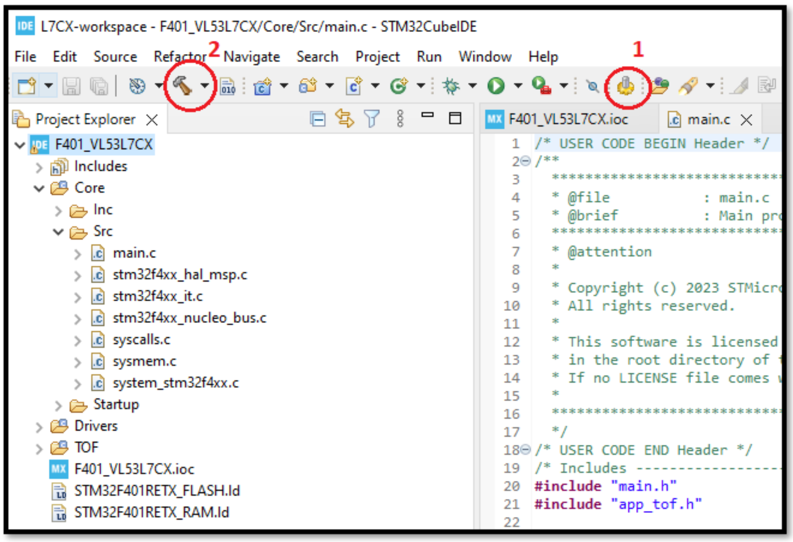



21. Verify the green tick.

**Figure 35. Check of the platform setting**

22. Next, generate and build the application. Click on the generate icon, then the hammer icon.

**Figure 36. Generate and build the code**



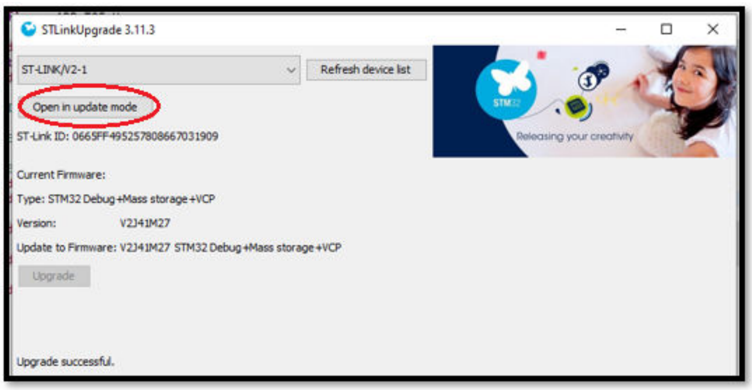

23. Next, run the debugger. Click on the bug.

**Figure 37. Launch the debugger**

You may have to upgrade your STLINK. If so:

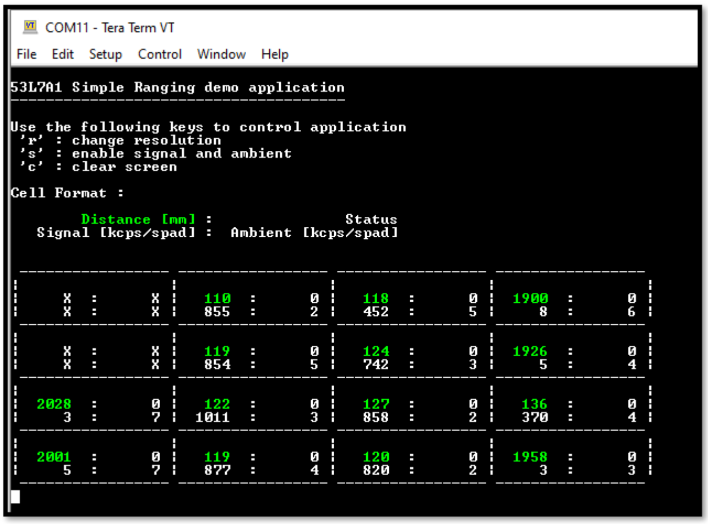
- Open in upgrade mode
- Click on "Upgrade"
- Close the upgrade window
- Restart the debugger

**Figure 38. Upgrade STLINK**



24. Start a serial port terminal emulation by adapting the baud rate as per Figure 11. Tera Term serial port setup. The result should look like Figure 39. Results.

**Figure 39. Results**




The above steps for the VL53L7CX can be used for other sensors. The pin connection is described in the X_CUBE_TOF1.chm file in the "Documentation" folder.

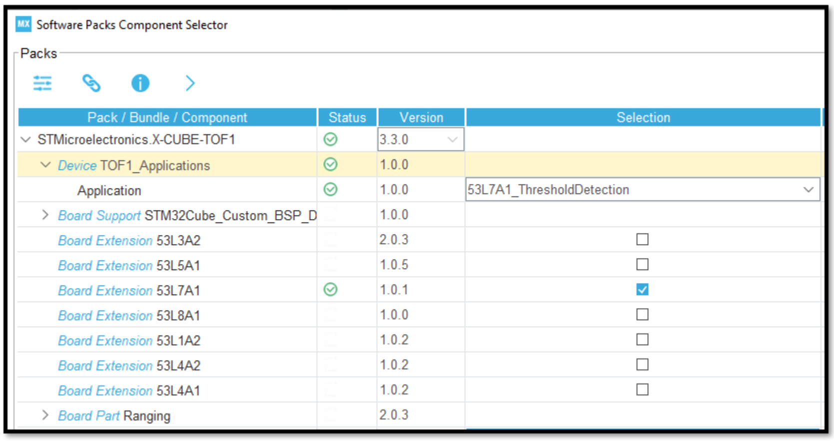
The above steps for the VL53L7CX can be used for other sensors. The pin connection is described in the X_CUBE_TOF1.chm file in the "Documentation" folder.

#### 4.1.2.2 *How to generate the 53L7A1_ThresholdDetection example with the STM32CubeMX*

The steps for generating the 53L7A1_ThresholdDetection example with the STM32CubeMX are almost identical to those in Section 4.1.2.1 How to generate the SimpleRanging example with the STM32CubeMX. Steps 6, 20, and 24 are different. They are described below respectively.

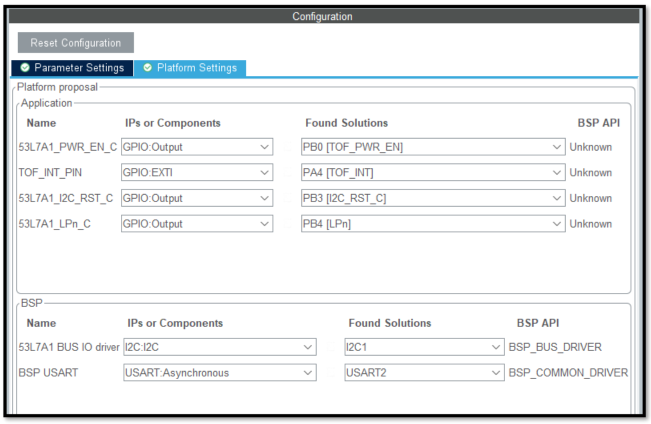
- Step 6: Choose the "ThresholdDetection Application".

**Figure 40. ThresholdDetection application**



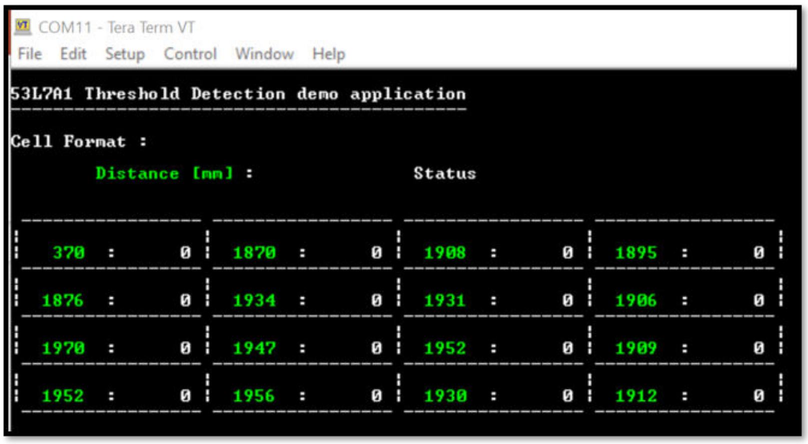

- Step 20: Assign pins.

**Figure 41. Pin assignment for the ThresholdDetection example**



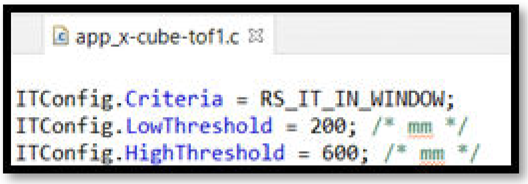
- Step 24: Start a serial port terminal emulation with a baud rate as per Figure 11. Tera Term serial port setup. The result should look like Figure 42. Result of the ThresholdDetection example.

**Figure 42. Result of the ThresholdDetection example**




An interrupt occurs if the target distance (d) = [200 mm and 600 mm] as shown in Figure 43. Interrupt.

**Figure 43. Interrupt**

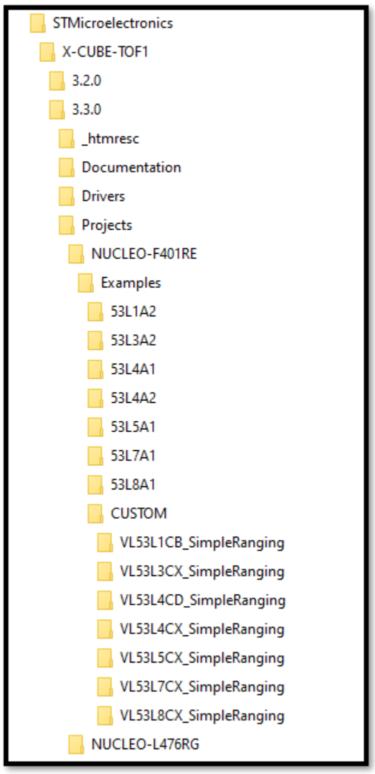



Other examples can be found for the different sensors. Do not hesitate to check in the "Examples” folder.

## 4.2 SimpleRanging examples using an STM32 Nucleo board and a SATEL breakout board

For each Time-of-Flight sensor present in the X-CUBE-TOF1 pack, a simple ranging example is delivered in the "CUSTOM" folder. This example is for use with a NUCLEO-F401RE or a NUCLEO-L476RG as per Figure 44. Customer folder of the SimpleRanging example using a SATEL breakout board.

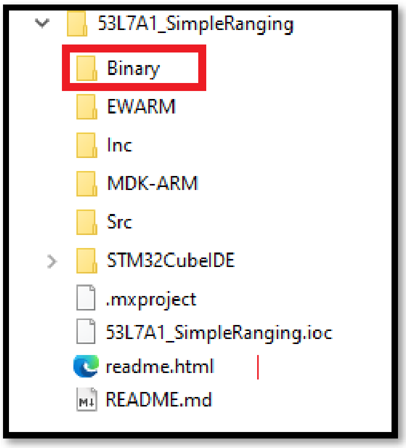
**Figure 44.** **Customer folder of the SimpleRanging example using a SATEL breakout board**




### 4.2.1 Loading a prebuilt binary file

Use the same steps given in Section 4.1.1 Loading a prebuilt binary file by loading the binary file in the CUSTOM/ <Sensor_Name>_SimpleRanging/Binary folder (see Figure 45. Binary folder for the SimpleRanging example using a SATEL breakout board).

**Figure 45.** **Binary folder for the SimpleRanging example using a SATEL breakout board**

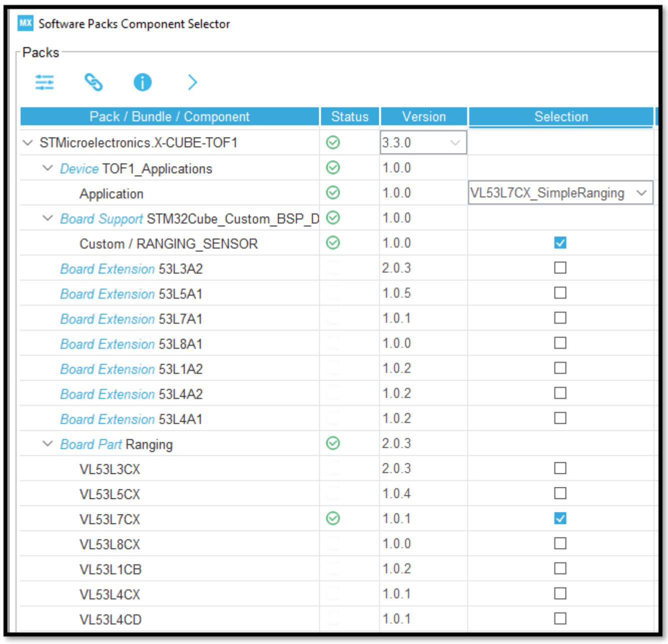


### 4.2.2 Configuring the STM32CubeMX for the SimpleRanging example

The steps for configuring the STM32CubeMX for the SimpleRanging example using an STM32 Nucleo board and a SATEL board are almost identical to those in Section 4.1.2.1 How to generate the SimpleRanging example with the STM32CubeMX. Steps 6 and 19 are different. They are described below respectively.

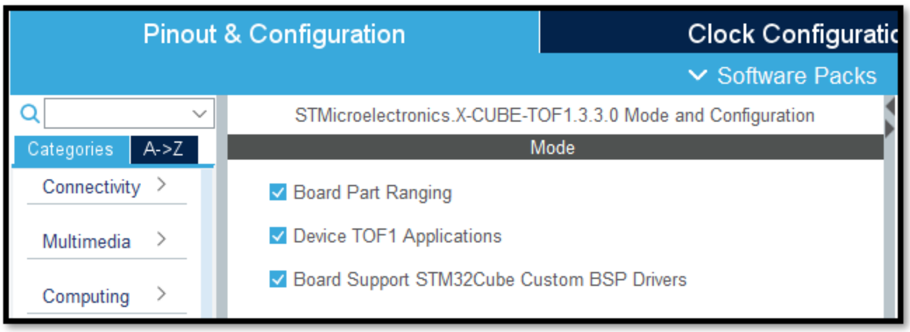
- Step 6: Choose the right application. It is called <Sensor_Name>_SimpleRanging. For the current example with a VL53L7CX sensor, choose "VL53L7CX_SimpleRanging".

**Figure 46. SimpleRanging application example for the SATEL breakout board**




- Step 19: Choose “Board Extension” and “Device TOF1 Applications”.

**Figure 47. Choice of boards and application**



## Revision history

Table 1. Document revision history

| Date | Version | Changes |
|---|---|---|
| 27-Apr-2023 | 1 | Initial release |
| 21-Jun-2023 | 2 | Section 3.1 Hardware description: added a preamble.<br>Updated Figure 4. Example of an X-NUCLEO-53L7A1 expansion board with a cover glass. |

# Contents