



STEVAL-PDETECT1

Setup

Version 1.0 – Mar '24

Agenda

- 1 Hardware and Software overview
- 2 Setup & Demo Examples
- 3 Documents & Related Resources

1 - Hardware and Software overview

STWIN.box development kit - STEVAL-STWINBX1

Hardware Overview

STWIN.box - SensorTile Wireless Industrial Node

The STWIN.box (STEVAL-STWINBX1) is a development kit and reference design that simplifies prototyping and testing of advanced industrial sensing applications in IoT contexts such as condition monitoring and predictive maintenance. It is an evolution of the original STWIN kit (STEVAL-STWINKT1B) and features a higher mechanical accuracy in the measurement of vibrations, an improved robustness, an updated BoM to reflect the latest and best-in-class MCU and industrial sensors, and an easy-to-use interface for external add-ons.

The STWIN.box kit consists of an STWIN.box core system, a 480mAh LiPo battery, an adapter for the ST-LINK debugger (STEVAL-MKIGIBV4), a plastic case, an adapter board for DIL 24 sensors and a flexible cable.

Key Features

- Multi-sensing wireless platform for vibration monitoring and ultrasound detection
- Built around STWIN.box core system board with processing, sensing, connectivity, and expansion capabilities
- Ultra-low power Arm® Cortex®-M33 with FPU and TrustZone at 160 MHz, 2048 kBytes Flash memory (STM32U585AI)
- MicroSD card slot for standalone data logging applications
- On-board Bluetooth® low energy v5.0 wireless technology (BlueNRG-M2), Wi-Fi (EMW3080) and NFC (ST25DV04K)
- Wide range of industrial IoT sensors: Ultra-wide bandwidth (up to 6 kHz), low-noise, 3-axis digital vibration sensor (IIS3DWB), 3D accelerometer + 3D gyro iNEMO inertial measurement unit (ISM330DHCX) with Machine Learning Core, High-performance ultra-low-power 3-axis accelerometer for industrial applications (IIS2DLPC), Ultra-low power 3-axis magnetometer (IIS2MDC), Dual full-scale, 1.26 bar and 4 bar, absolute digital output barometer in full-mold package (ILPS22QS), Low-voltage, ultra low-power, 0.5°C accuracy I²C/SMBus 3.0 temperature sensor (STTS22H), Industrial grade digital MEMS microphone (IMP34DT05), Analog MEMS microphone with frequency response up to 80 kHz (IMP23ABSU)
- Expandable via a 34-pin FPC connector

STEVAL-STWINBX1



Latest info available at
<https://www.st.com/en/evaluation-tools/steval-stwinbx1.html>

STWIN.box development kit - STEVAL-STWINBX1

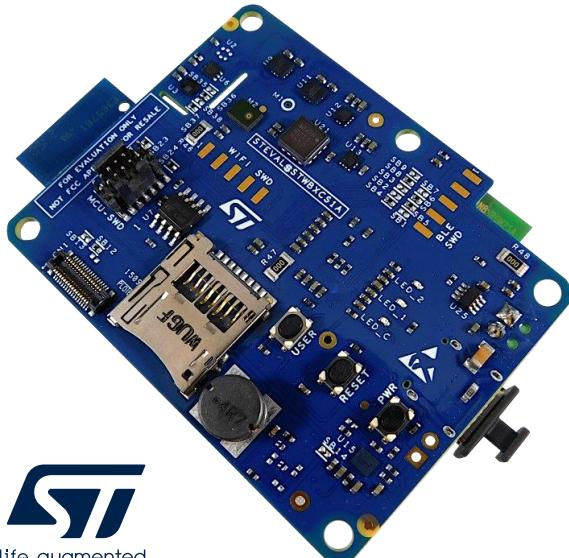
Hardware Overview

STWIN.box - SensorTile Wireless Industrial Node

The STEVAL-STWINBX1 development kit includes:

- The STEVAL-STWBXCS1 STWIN.box core system (main board);
- A plastic case with M3 bolts;
- A 480 mAh 3.7 V LiPo battery;
- The STEVAL-MKIGIBV4 ST-LINK adapter with programming cable;
- The STEVAL-C34DIL24 adapter board for DIL24 sensors with the STEVAL-FLTCB01 flexible cable.

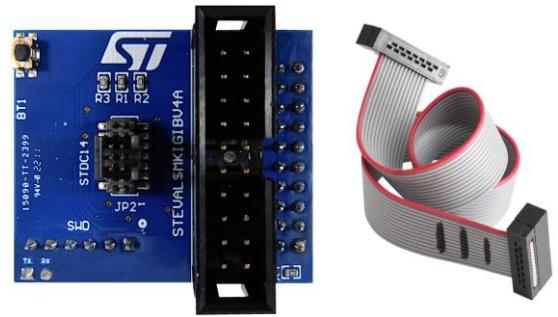
STEVAL-STWBXCS1
STWIN.box Core System



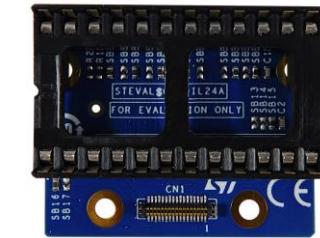
Plastic Case



Battery
LiPo-752535 - 480mAh



STEVAL-MKIGIBV4 + Cable
STLINK Adapter (V2, V2.1)



STEVAL-C34DIL24



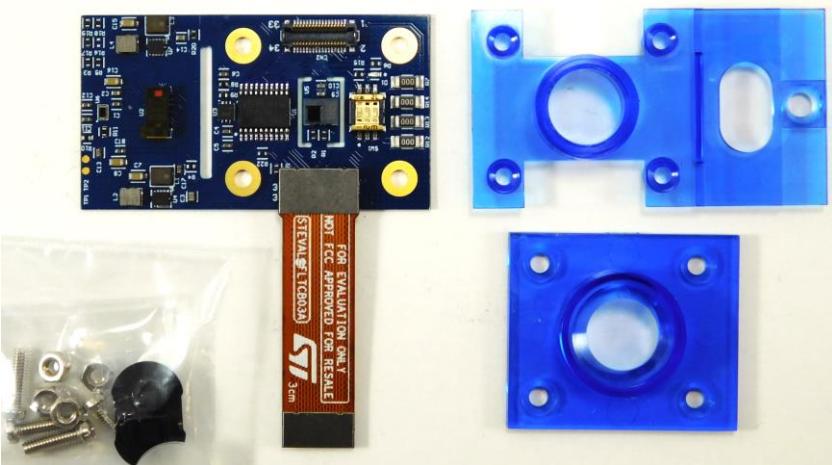
STEVAL-FLTCB01
34 pin Flex cable

STWIN.box Modular platform expansion - STEVAL-PDETECT1

Hardware Overview

Presence Detection add-on for STWIN.box

- Add-on evaluation kit connected to STWIN.box targeting human presence applications. The kit includes:
 - STEVAL-PDETCS1 board
 - TMOS long range Fresnel lens: TMOS63-10
 - Plastic case for TMOS63-10 Fresnel lenses
 - Time-of-Flight cover glass: IR136C0-IC09-A066
 - STEVAL-FLTCB03 flex cable

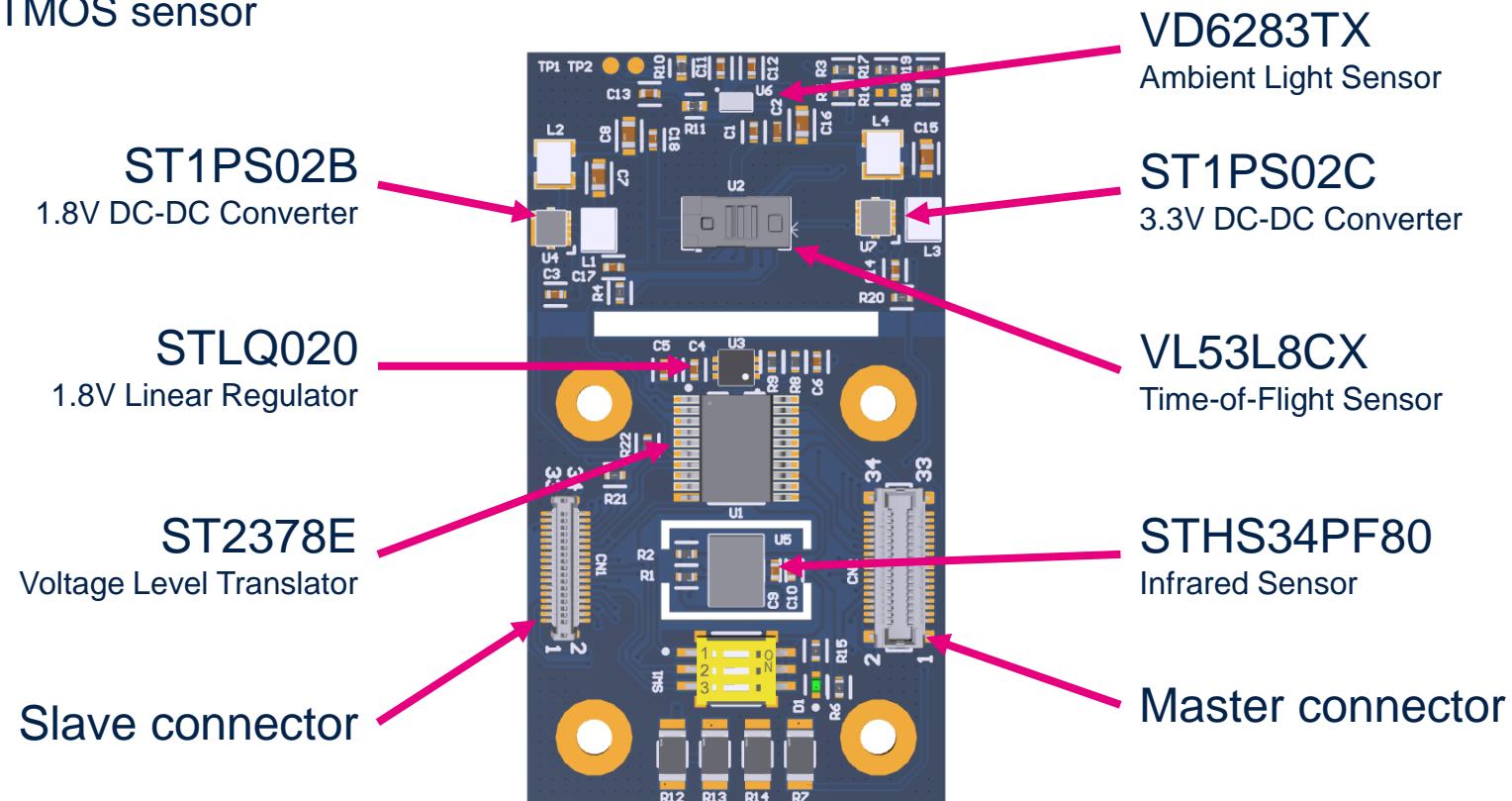


STWIN.box Modular platform expansion - STEVAL-PDETECT1

Hardware Overview

Presence Detection add-on for STWIN.box

- Key Sensor Products:
 - VD6283TX45/1 Ambient Light Sensor
 - VL53L8CXV0GC Proximity Sensor
 - STHS34PF80 Far Infrared TMOS sensor



FP-SNS-DATALOG2

Software Overview

Software Description

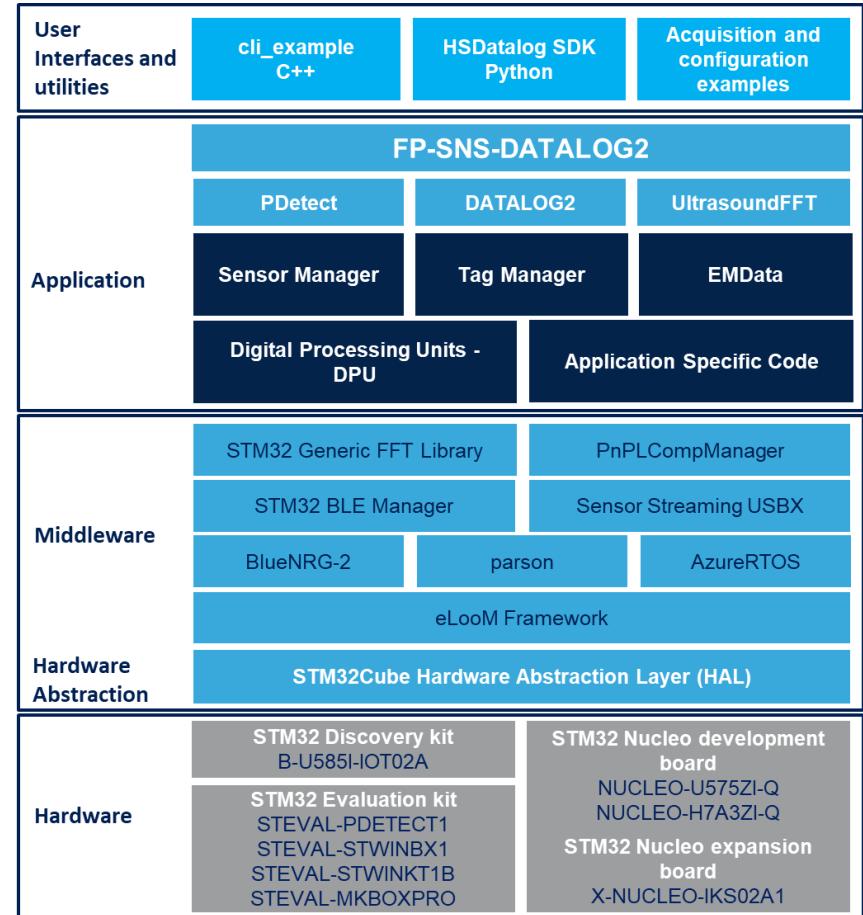
The FP-SNS-DATALOG2 function pack for B-U585I-IOT02A, STEVAL-STWINBX1, STEVAL-STWINKT1B, STEVAL-MKBOXPRO and X-NUCLEO-IKS02A1 with NUCLEO-U575ZI-Q or with NUCLEO-H7A3ZI-Q represents an evolution of FP-SNS-DATALOG1 and provides a comprehensive solution for saving data from any combination of sensors and microphones configured up to the maximum sampling rate. It natively supports STEVAL-STWINBX1 with its external addons (STEVAL-PDETECT1, STEVAL-C34KAT1, STEVAL-C34KAT2, STEVAL-C34DIL24).

The package also includes an example of ultrasound condition monitoring (UltrasoundFFT) for STEVAL-STWINBX1 that calculates the FFT of the IMP23ABSU analog microphone signal and streams the result to a PC GUI via USB and a dedicated example for human presence and motion detection (PDetect) for STEVAL-STWINBX1 with STEVAL-PDETECT1

Key features

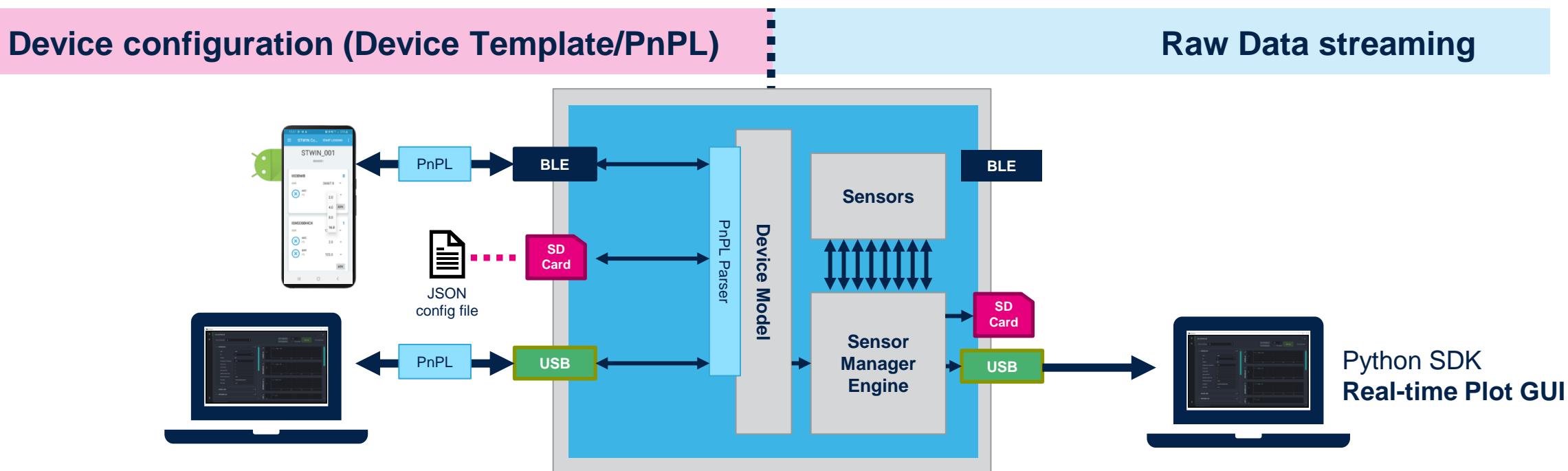
- High-rate (up to 6 Mbit/s) data capture software suite:
 - Dedicated Python SDK, ready-to-use for integration into any data science design flow
 - Compatible with ST BLE Sensor app for system setup and real-time control
 - Allow to setup MLC (Machine Learning Core) or ISPU (Intelligent Sensor Processing Unit)
 - Synchronized timestamping and labelling mechanisms common to all sensors
- Firmware modular examples based on eLooM (Embedded Light Object Oriented framework for STM32) to enable code reusability at application level
- Based on AzureRTOS

Overall Software Architecture



DATALOG2 demonstration

Optimized STM32 FW Supports streaming of all Sensors at Full data rate

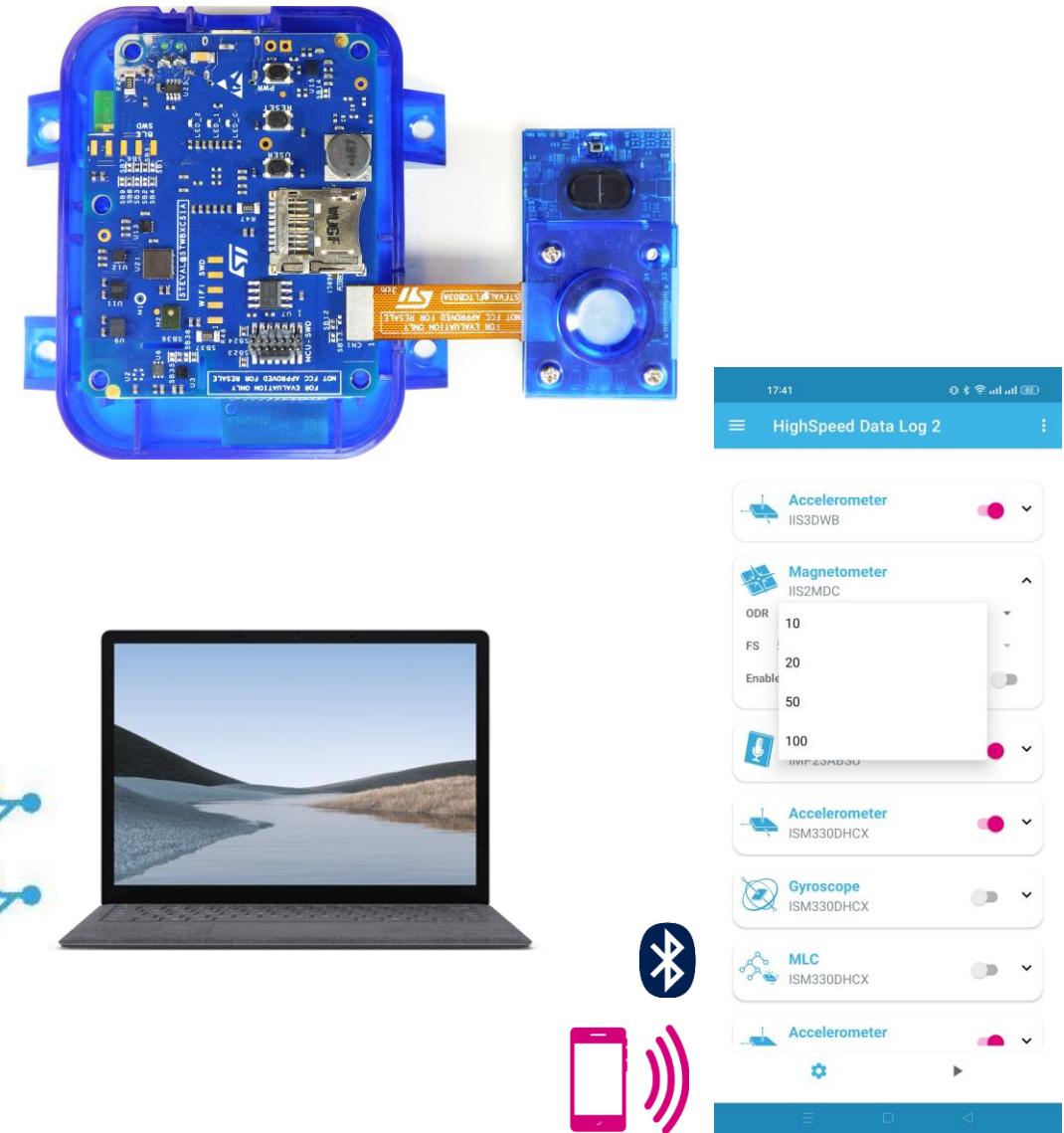


- Based on **AzureRTOS**
 - ThreadX, FileX, USBX
- USBX WCID Streaming class and PC DLL
- SD Card
- Full control of acquisition via BLE app
- Control via **PnP-Like** commands

2 - Setup & Demo Examples

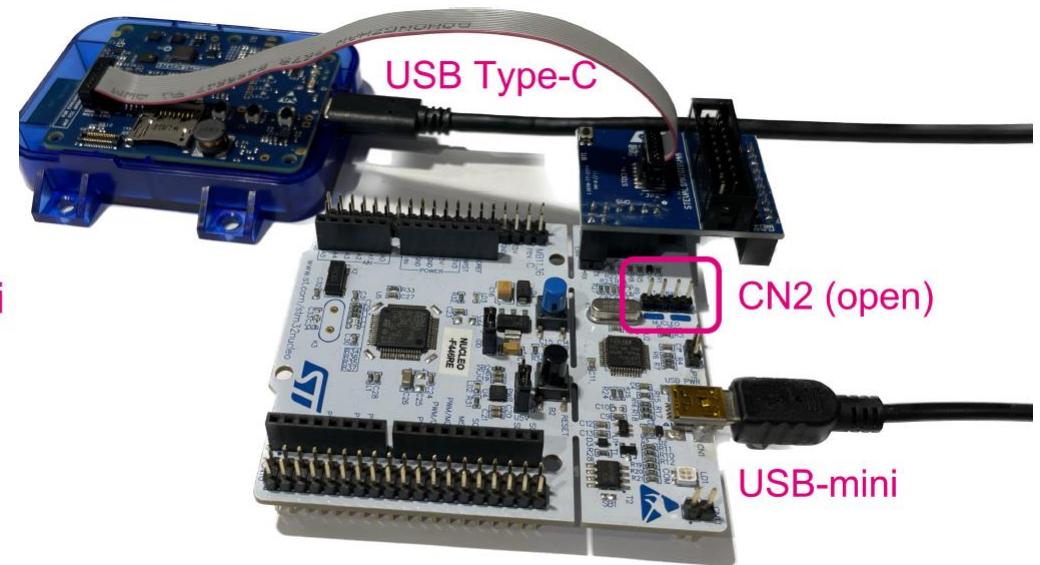
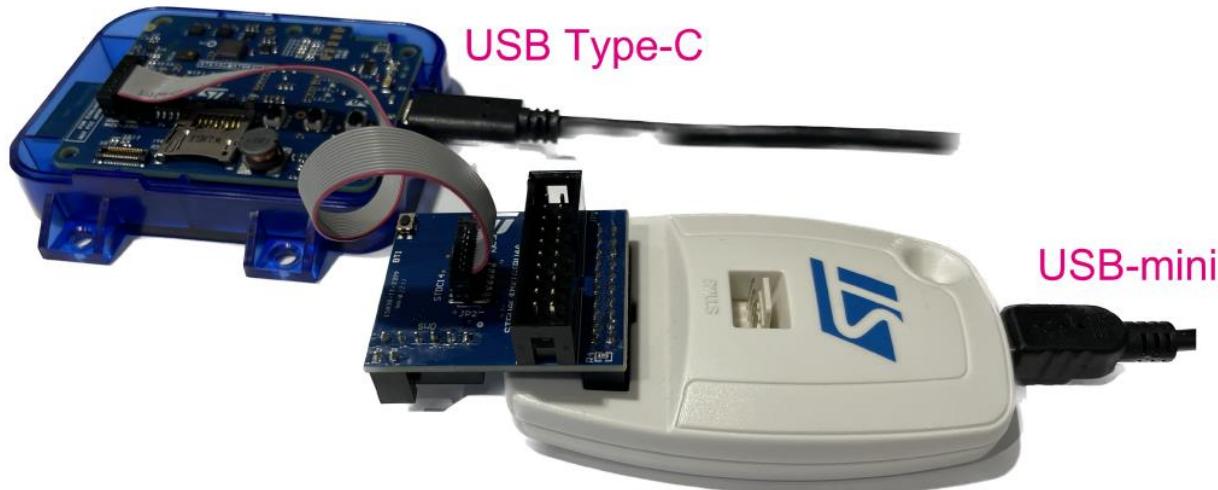
HW prerequisites

- 1 STEVAL-STWINBX1
- 1 STEVAL-PDETECT1
- Laptop/PC
- 2 type-C USB cables + 1 STLINK-V3MINIE
 - In alternative it is also possible to directly program the STWIN.box with a single USB-C cable (refer to [firmware update – USB](#))
- 1 micro-SD card
- 1 smartphone with ST BLE Sensor App



Programming options

- It is possible to directly program the STWIN.box with a single USB-C cable (refer to [firmware update – USB](#))
- If you are interested in FW debugging, STWIN.box programming connector is natively compatible with STLINK-V3 debuggers family (STLINK-V3SET or STLINK-V3MINI). STLINKV3 programmers are NOT included in the kit.
- Alternatively, in order to offer more alternatives, an adapter to ST-Link V2-1 (STM32-Nucleo) or standard JTAG connector is included in the kit.



Software and other prerequisites

- **STM32CubeProgrammer Software**
 - Download and install [STM32CubeProgrammer](#)
- **PDetect**
 - Download the [FP-SNS-DATALOG2](#) package from www.st.com, copy the .zip file contents into a folder on your PC. The package contains binaries and source code with project files ([Keil](#), [IAR](#), [STM32CubeIDE](#))
- **ST BLE Sensor App**
 - Download and install ST BLE Sensor App (for both Android and iOS – v5.2 and above)
- **Python**
 - To save, plot and elaborate data, Python utility scripts are available

PDetect is **not** the default firmware.

To update the firmware, please follow the instructions for [Fast FOTA](#) valid for
STWIN.box

Samples demonstrations



1 www.st.com



2 Select:

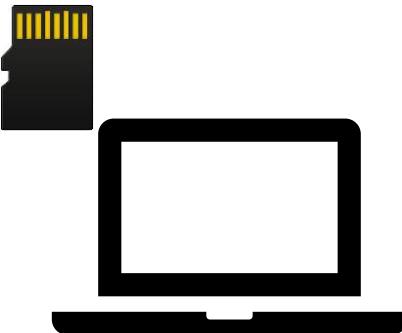
FP-SNS-DATALOG2



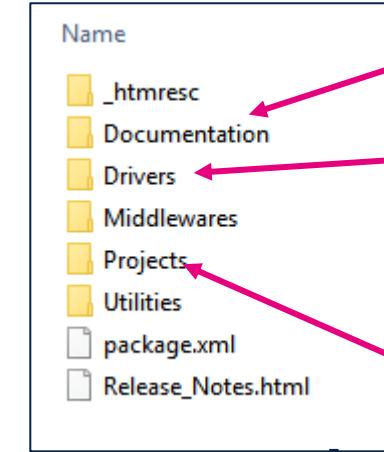
3 Download & unpack



6 Visualize log of sensors data and control the device



Package structure



BSP, HAL
drivers

Sample
applications



5 Use the pre-compiled binaries or re-compile the code customizing your device configuration



Firmware Update – ST BLE Sensor app

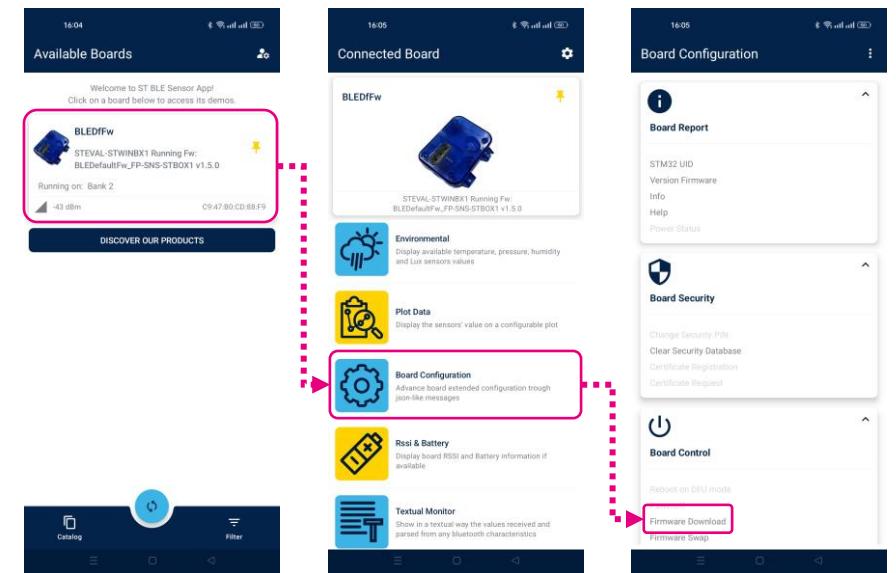
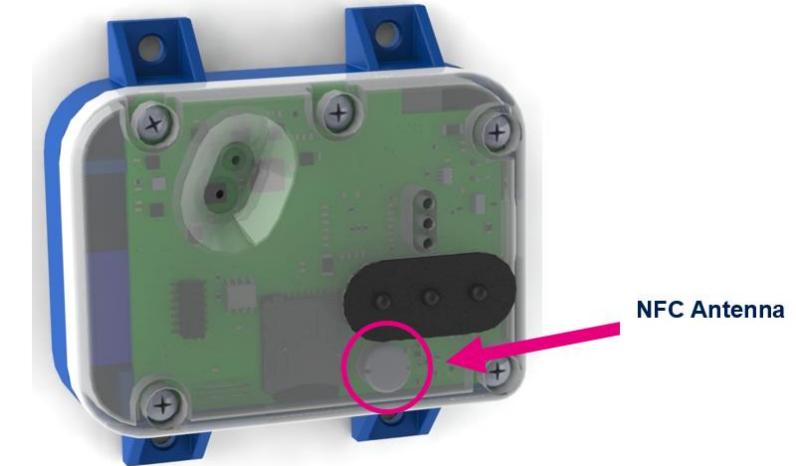
PDetect **is not** the default firmware, so it needs to be downloaded on the board by the user.

The default firmware for STWIN.box enables the Bluetooth pairing via **NFC** and Fast Firmware On-The-Air upgrade through **ST BLESensor app**.

By just turning on Bluetooth and NFC on the smartphone and placing your smartphone on top of the NFC Antenna of the STWIN.box, the smartphone will read the Bluetooth pairing information and it will automatically load the ST BLESensor.

In alternative, just open manually the application. During BLE pairing, if requested, insert the following PIN: **123456**.

At this point, you can choose to upgrade the firmware on the board directly by using the mobile app, by selecting one of the available firmware.



Firmware Update – ST-LINK

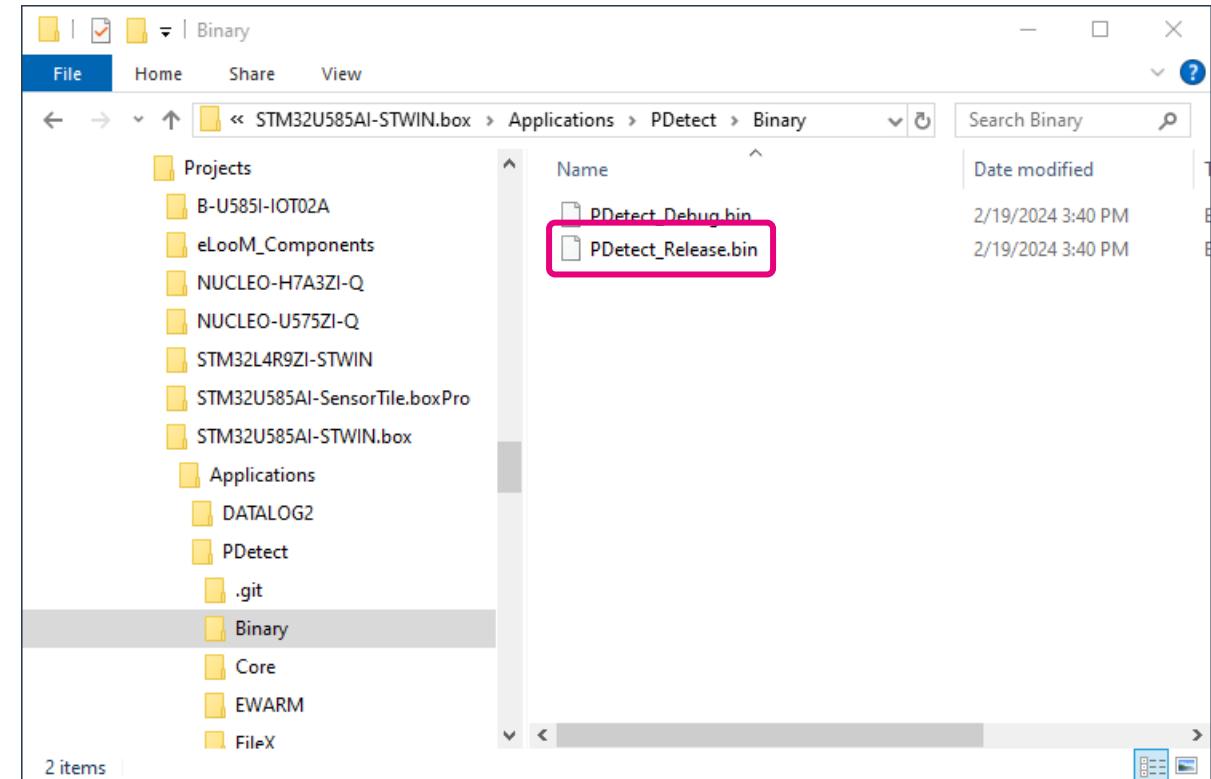
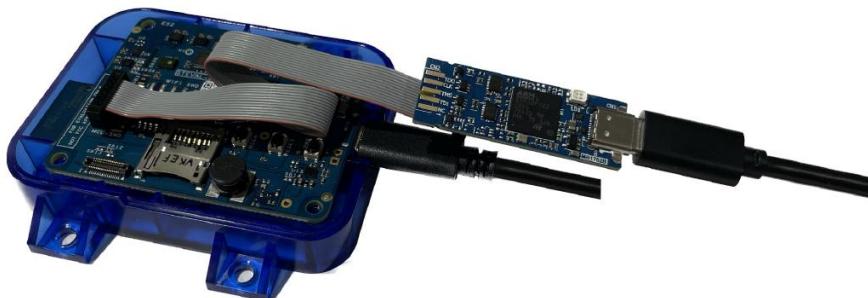
An alternative way is to use the **pre-compiled binary** provided in the package (i.e.: *Projects\STM32U585AI-STWIN.box\Applications\PDetect\Binary*)

To update the firmware:

- Connect the board to the preferred programmer (here we are using STLINK-V3MINIE).
- Connect both the boards to a PC through the proper USB cables.
- Open STM32CubeProgrammer, select the proper binary file and download the firmware.

For further details, see [UM2965](#) for

STWIN.box

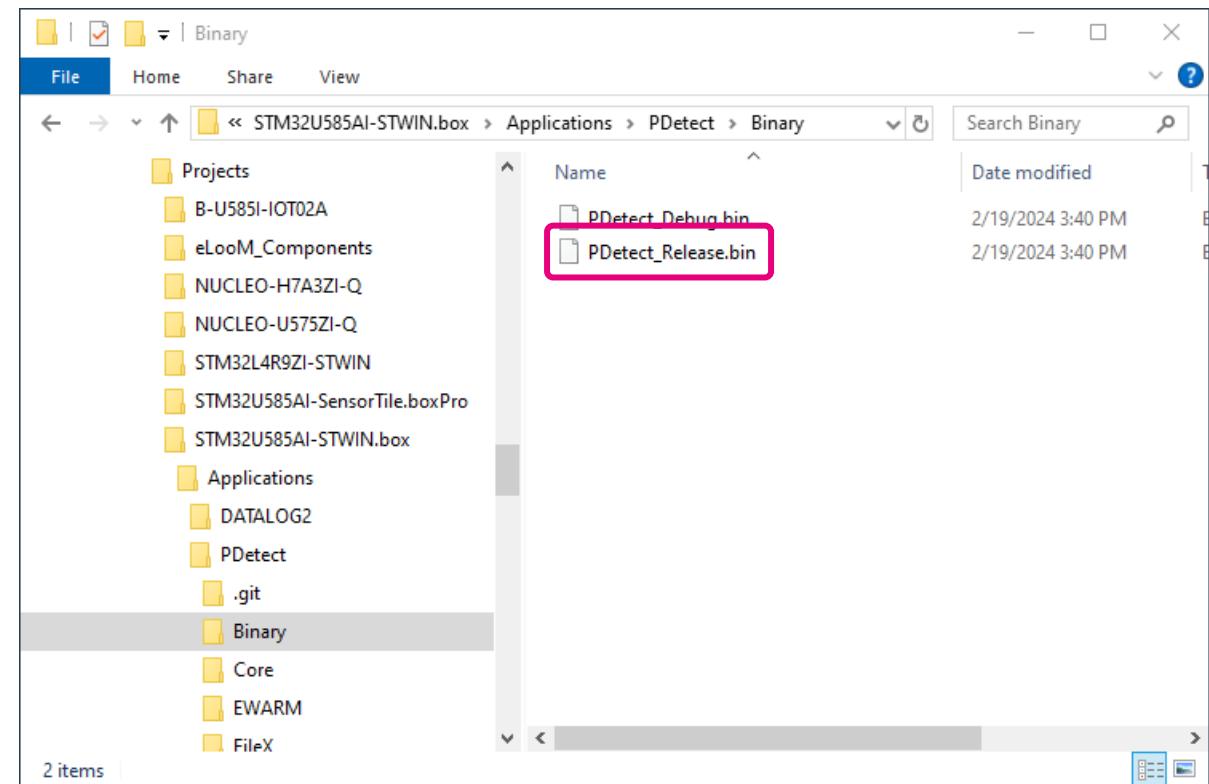


Firmware Update – USB

STEVAL-STWINBX1 can also be reprogrammed via USB using the STM32CubeProgrammer "USB mode".

To enter "Firmware upgrade" mode you must follow the procedure below:

- Unplug the core system board.
- Press the USR button in STWIN.box
- While keeping the button pressed, connect the USB cable to the PC.
- Now the board is in DFU mode. Open [STM32CubeProgrammer](#), select the proper binary file and download the firmware.



For further details, see [UM2965](#) for STWIN.box

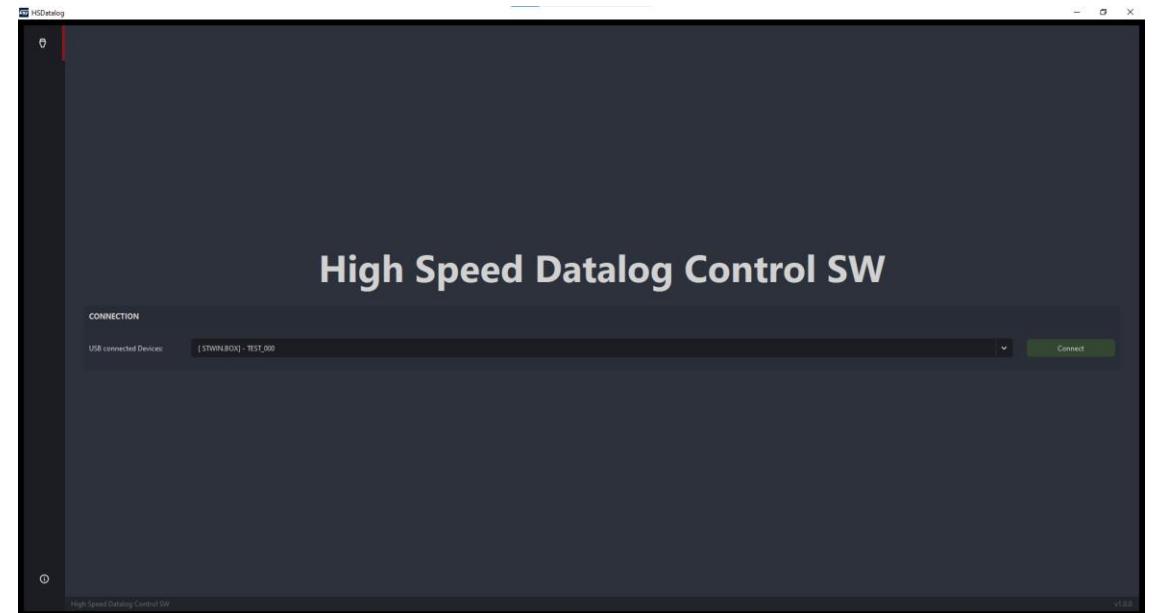
2.1 – PDetect Demonstration

2.1.1 - USB sensor data streaming

Real Time Plot

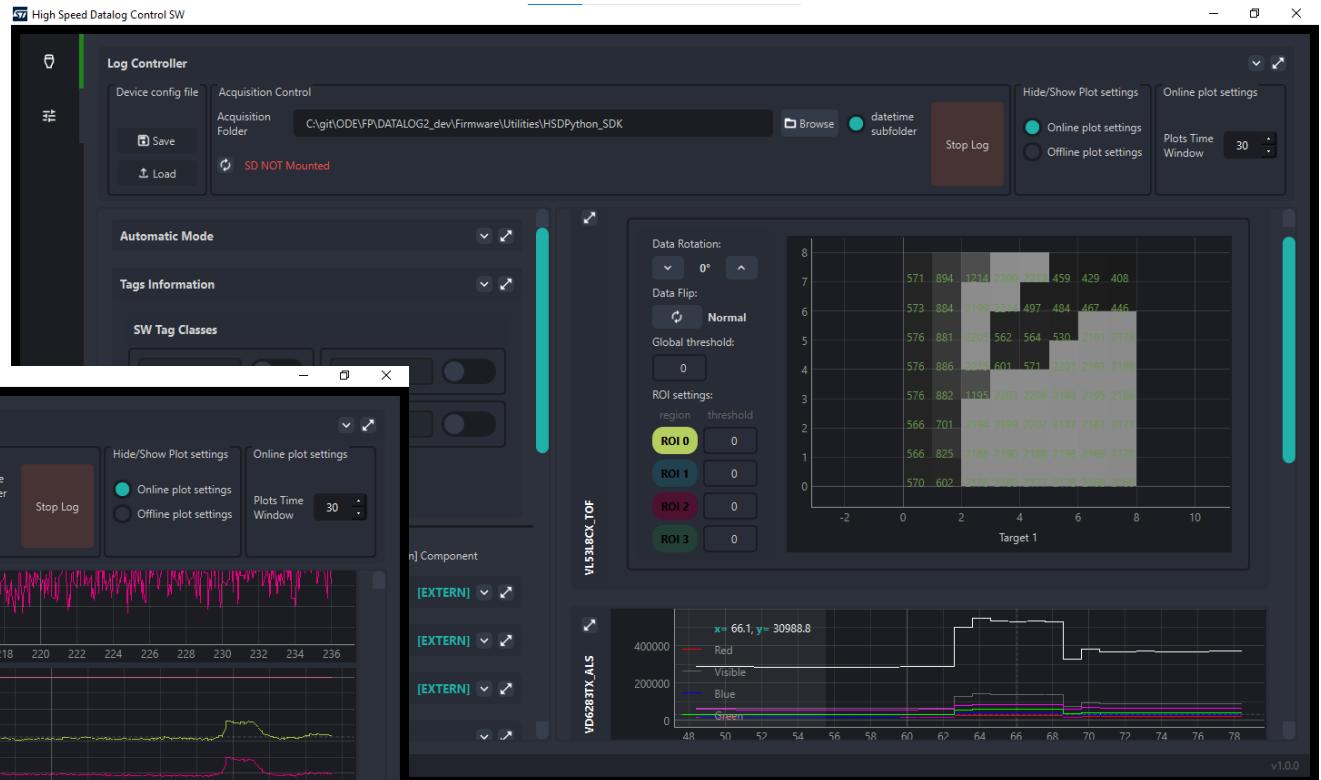
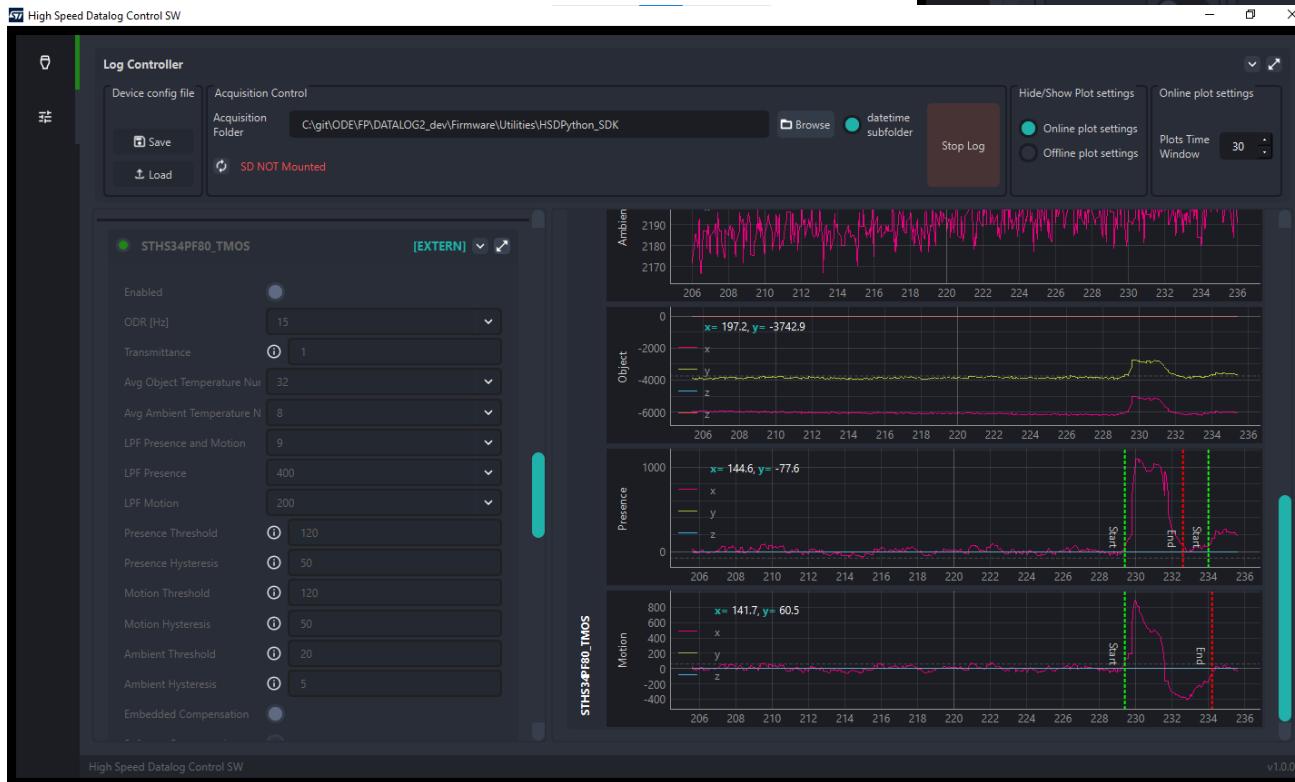
Execute *hsdatalog_GUI.py*

- FP-SNS-DATALOG2 provides a Python example for PDetect real time control and plot
- *hsdatalog_GUI.py* works within the *HSDPython_SDK*, developed in Python 3.10 on Windows and Linux environments.
 - *HSDPython_SDK* requires different Python modules, distributed together with FP-SNS-DATALOG2. By installing them through *the provided installers*, all the required dependencies are automatically solved – see [HSDPython_SDK](#) to fully setup your Python environment
- Once the board is connected via USB and the Python environment has been properly updated, you can launch the real time plot by just executing *hsdatalog_GUI.py* available in *Utilities/HSDPython_SDK/examples*
 - Depending on your local setup, to execute the script you can open a command shell there and run *python hsdatalog_GUI.py*
- Click on *Connect* button to allow the connection between the board and the PC



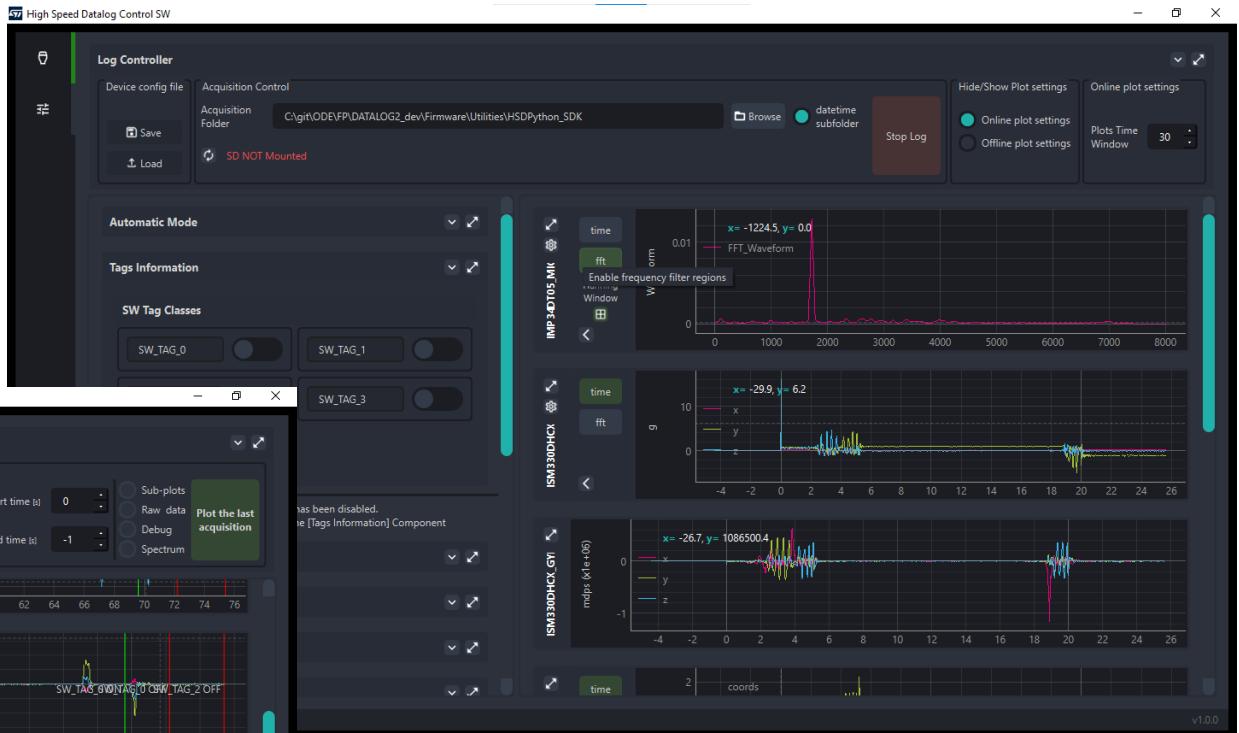
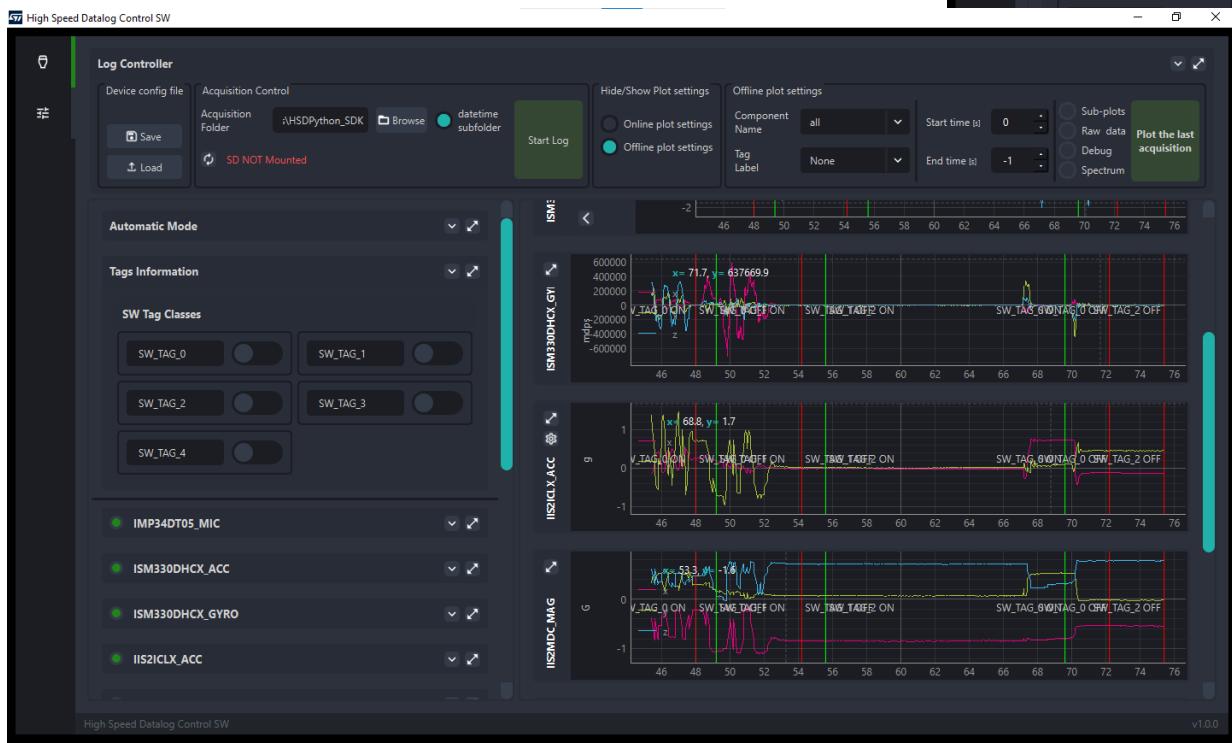
hsdatalog_GUI.py

- Once the connection is established, you can:
 - Enable/disable the needed sensors
 - Setup data rate, full scale, timestamps
 - Retrieve sensor status
 - Save and load a configuration via a JSON file
 - Start/stop logging data on the PC



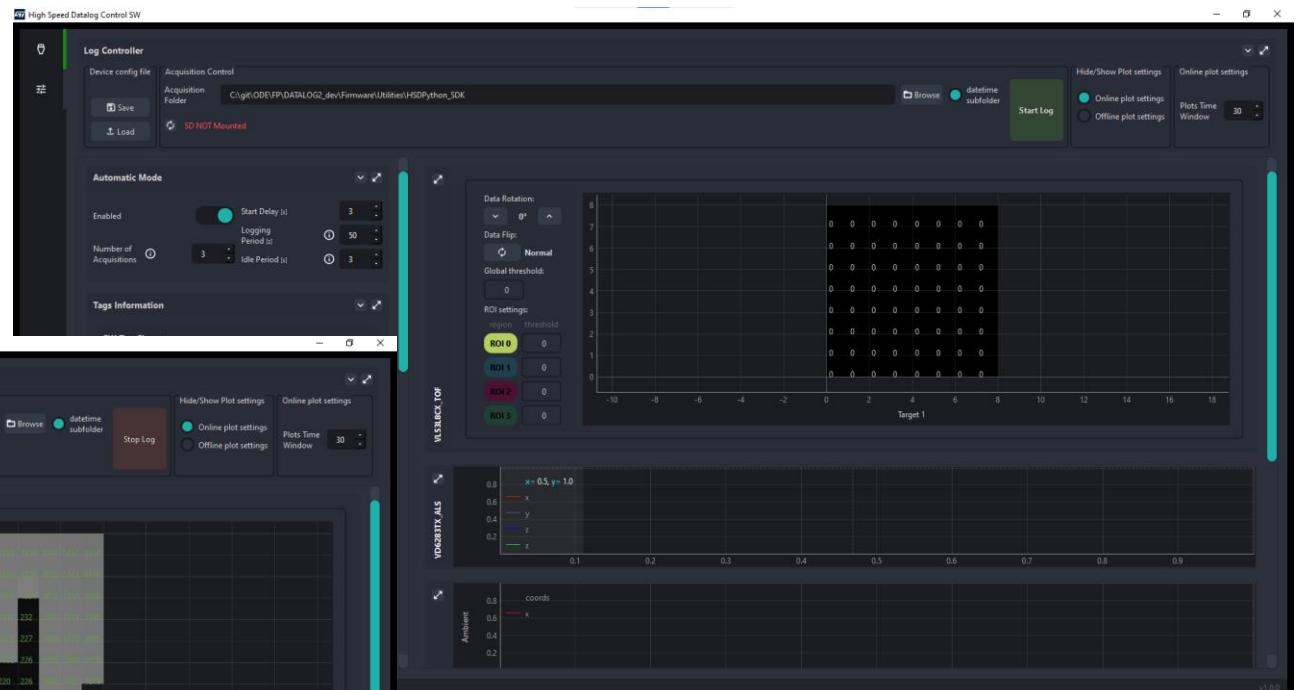
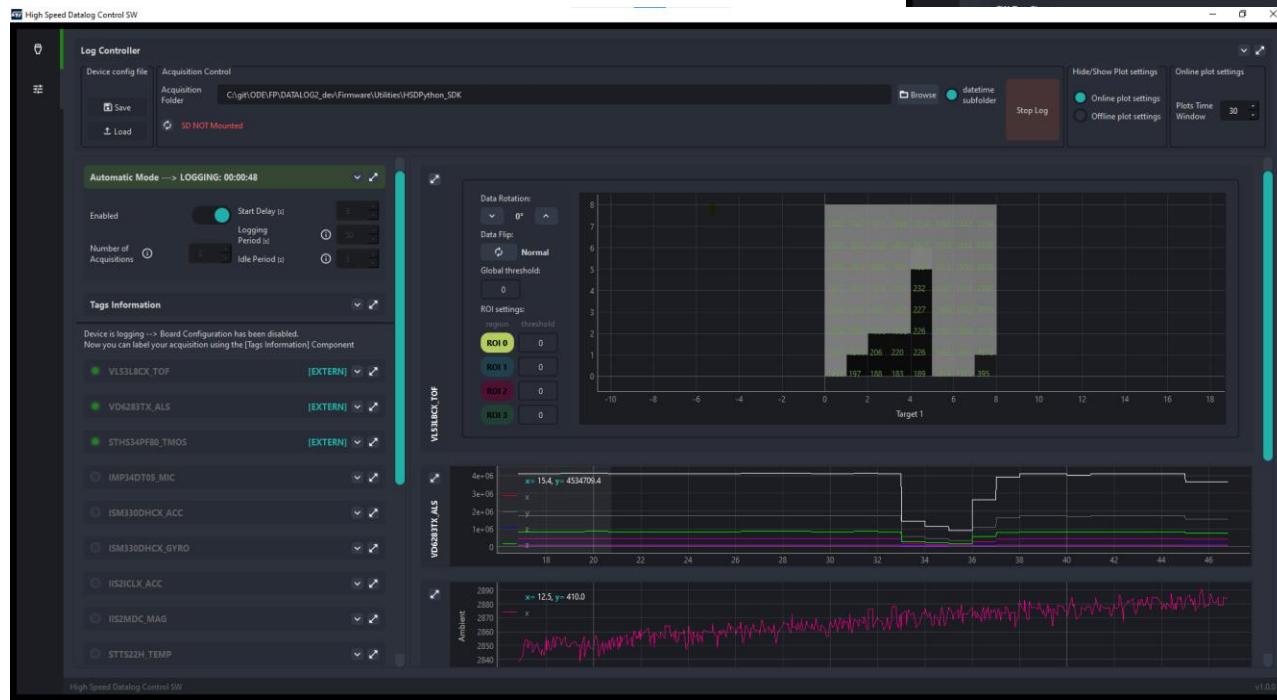
hsdatalog_GUI.py

- Real Time Plot GUI allows also to:
 - setup tag classes and handle data tagging and labelling of an ongoing acquisition
 - Enable FFT calculation for accelerometers and microphones
 - set up the acquisition name and description



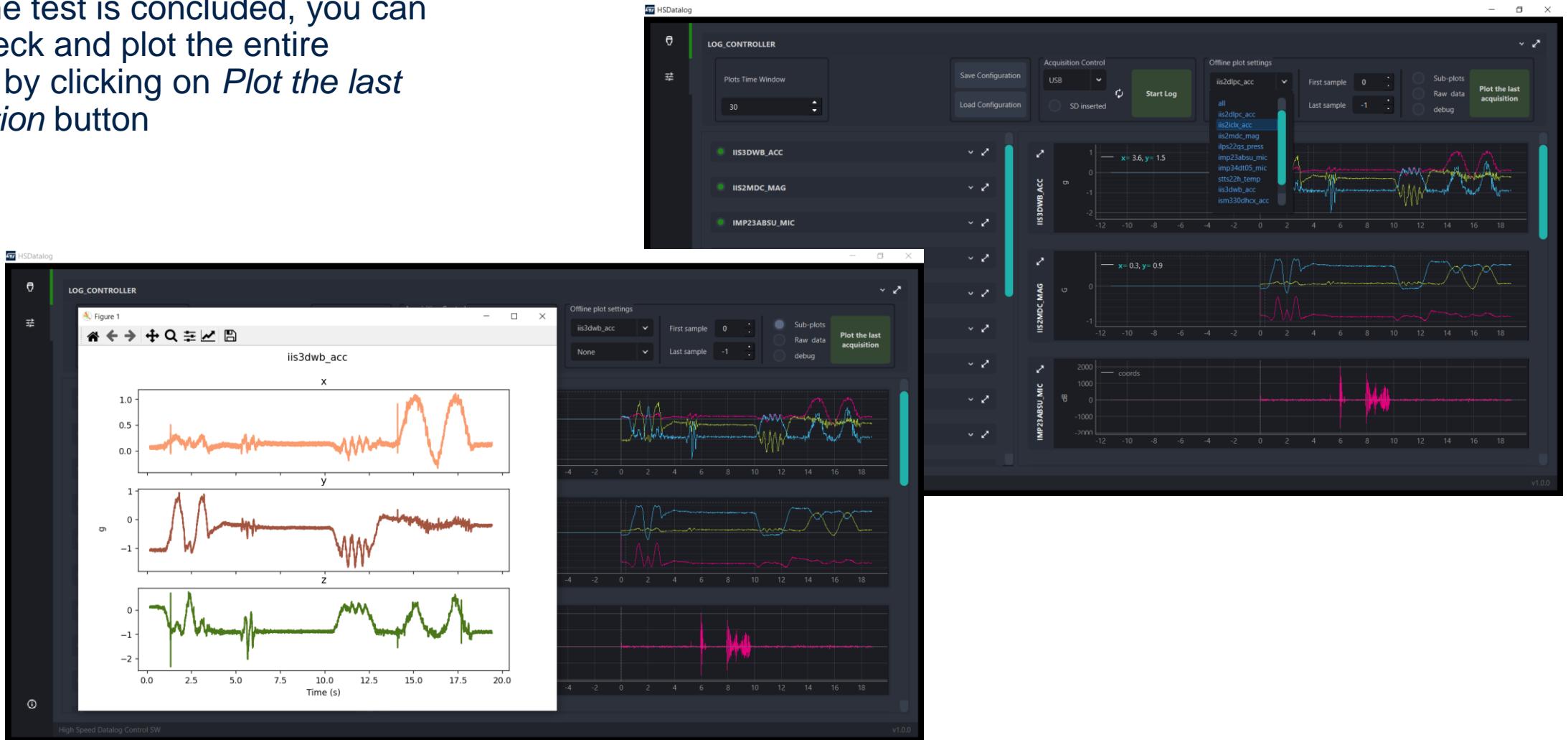
hsdatalog_GUI.py

- Furthermore, you can set up Automode to start the datalog operations or to pause all the executions for specific periods
 - Setup the desired values in Automatic Mode widget
 - See [Automode section](#) for further details



hsdatalog_GUI.py

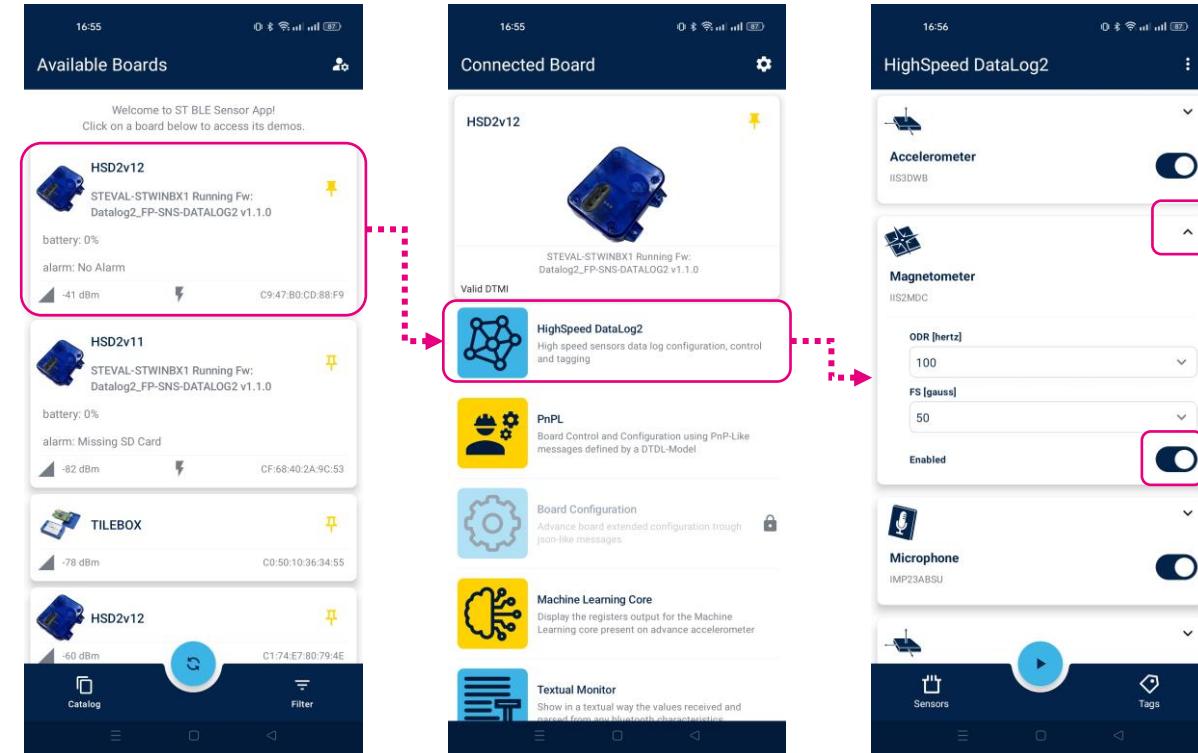
- Once the test is concluded, you can also check and plot the entire dataset by clicking on *Plot the last acquisition* button



2.1.2 – Data logging on SD card, configuration with BLESensor App

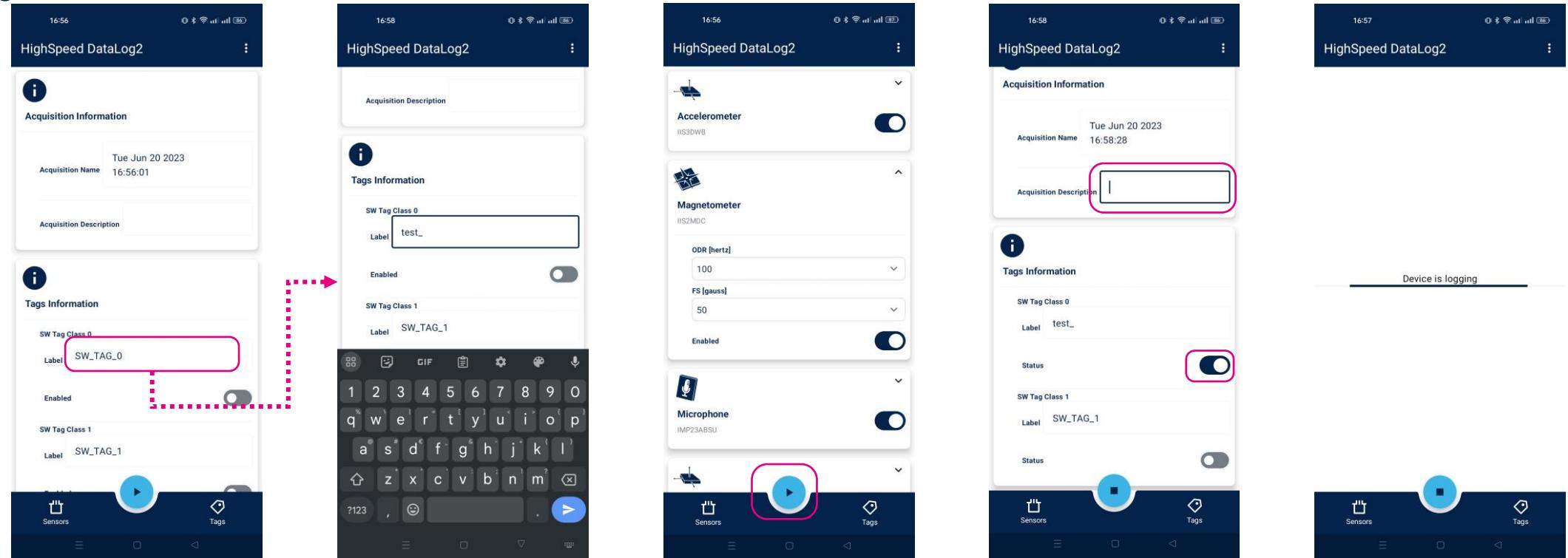
ST BLESensor App: DATALOG2 tab

- PDetect application can be controlled via Bluetooth using the **ST BLE Sensor** app (for both Android and iOS – v5.2 and above) which lets you manage the board and sensor configurations, start/stop data acquisition on SD card and control data labelling.
- Once connected, you can configure the device by:
 - enabling/disabling a specific sensor
 - changing sensor parameters



Acquisition settings and control

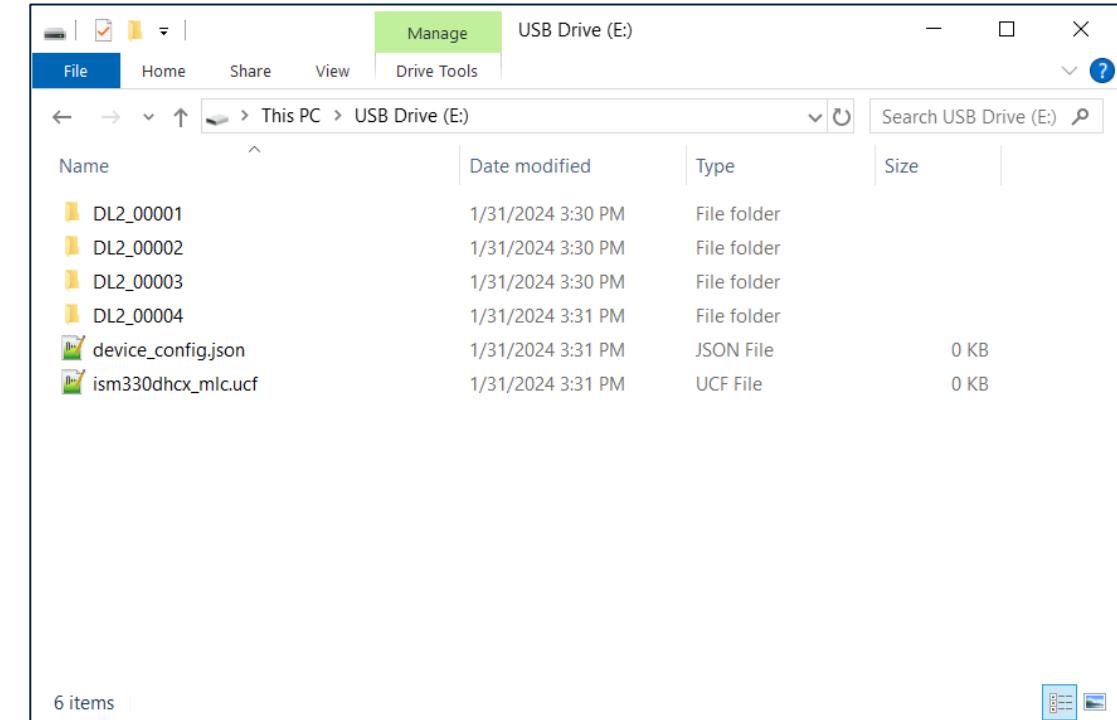
- By clicking to the tags button you can switch to the acquisition settings and control tab to:
 - start and stop an acquisition (to an SD card)
 - choose which tag classes will be used for the next acquisition
 - handle data tagging and labelling of an ongoing acquisition
 - set up the acquisition name and description
- A YYYYMMDD_HH_MM_SS (i.e., 20200128_16_33_00) folder containing the raw data and the JSON configuration file will be created into the SD card



2.1.3 – Data logging on SD card, standalone mode

Start an acquisition in standalone mode

- PDetect can work also standalone, saving selected sensor data at the highest possible rate into the SD card
- PDetect can read a custom sensors configuration from the SD card root folder. To do so, you can simply save a JSON configuration file in the root folder of the SD card



- Once the firmware is flashed on the board:
 - Insert the SD card
 - If the board is battery-powered and switched off, press PWR button to switch on the board. Press the RESET button
 - If the SD card is not inserted properly, the orange LED will be switched off. Otherwise, the orange LED will be switched on
 - If a JSON configuration file is present in the root folder of the SD card, the custom sensor configuration is loaded from the file itself
 - Press the USR button to start saving data. During datalog stage you will see the green LED blinking at 4Hz
 - To stop the data acquisition, press again the USR button. During idle stage you will see the green LED blinking at 1 Hz

- PDetect also features the automode, which can be initiated automatically at the device power-up or reset
- This mode can be used to start the datalog operations or to pause all the executions for a specific period by putting the sensor node in the "idle" phase.
- Automode allows automatically saving data on the SD card, generating different acquisitions folders. It can be useful to automate long acquisition setups, avoid datasets that are too large and reduce SD card errors by avoiding data loss through autosaving.

```
"automode": {  
    "enabled": true,  
    "nof_acquisitions": 7,  
    "start_delay_s": 5,  
    "logging_period_s": 30,  
    "idle_period_s": 5,  
    "c_type": 2  
}
```

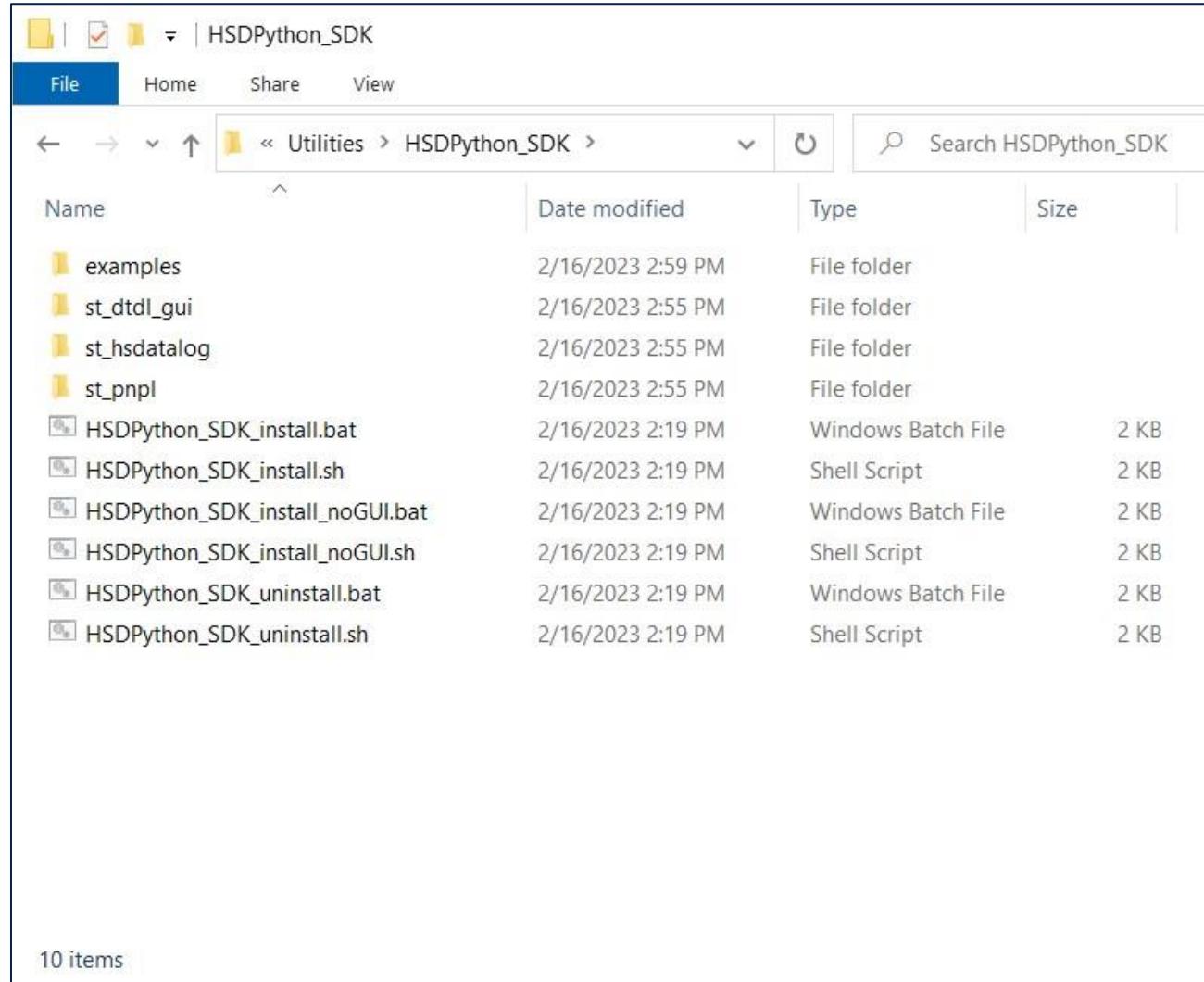
- As for the standalone mode, to enable the automode you must setup the automode component properly in the `device_config.json`
 - **enabled**: if true, the automode starts after the reset and node initialization. If false, automode is not executed.
 - **nof_acquisitions**: gives the number of times the automode is executed; zero indicates an infinite loop and it is the default value
 - **start_delay_s**: indicates the initial delay in seconds applied after reset and before the first execution phase starts
 - **logging_period_s**: specifies the duration in second of the datalog phase
 - **idle_period_s**: specifies the duration in seconds of the idle phase
 - **c_type**: describes the component type (0 “sensor”, 1 “algorithm”, 2 “other”)
- Then place it in the root folder of the SD card
- If the board is battery-powered and switched off, press PWR button to switch on the board. Press the RESET button

```
"automode": {  
    "enabled": true,  
    "nof_acquisitions": 7,  
    "start_delay_s": 5,  
    "logging_period_s": 30,  
    "idle_period_s": 5,  
    "c_type": 2  
}
```

2.2 – HSDPython_SDK

HSDPython_SDK

- FP-SNS-DATALOG2 comes with a dedicated Python SDK, ready-to-use for integration into any data science design flow.
- HSDPython_SDK is compatible also with FP-SNS-DATALOG1 and can handle data acquired from both packages.
- HSDPython_SDK has been developed in Python 3.10
- The SDK contains many Python scripts, examples and Jupiter notebooks that can be used to log and elaborate data
- The scripts take advantage for the API provided by the st_dtdl_gui, st_hsdatalog and st_pnpl Python modules.

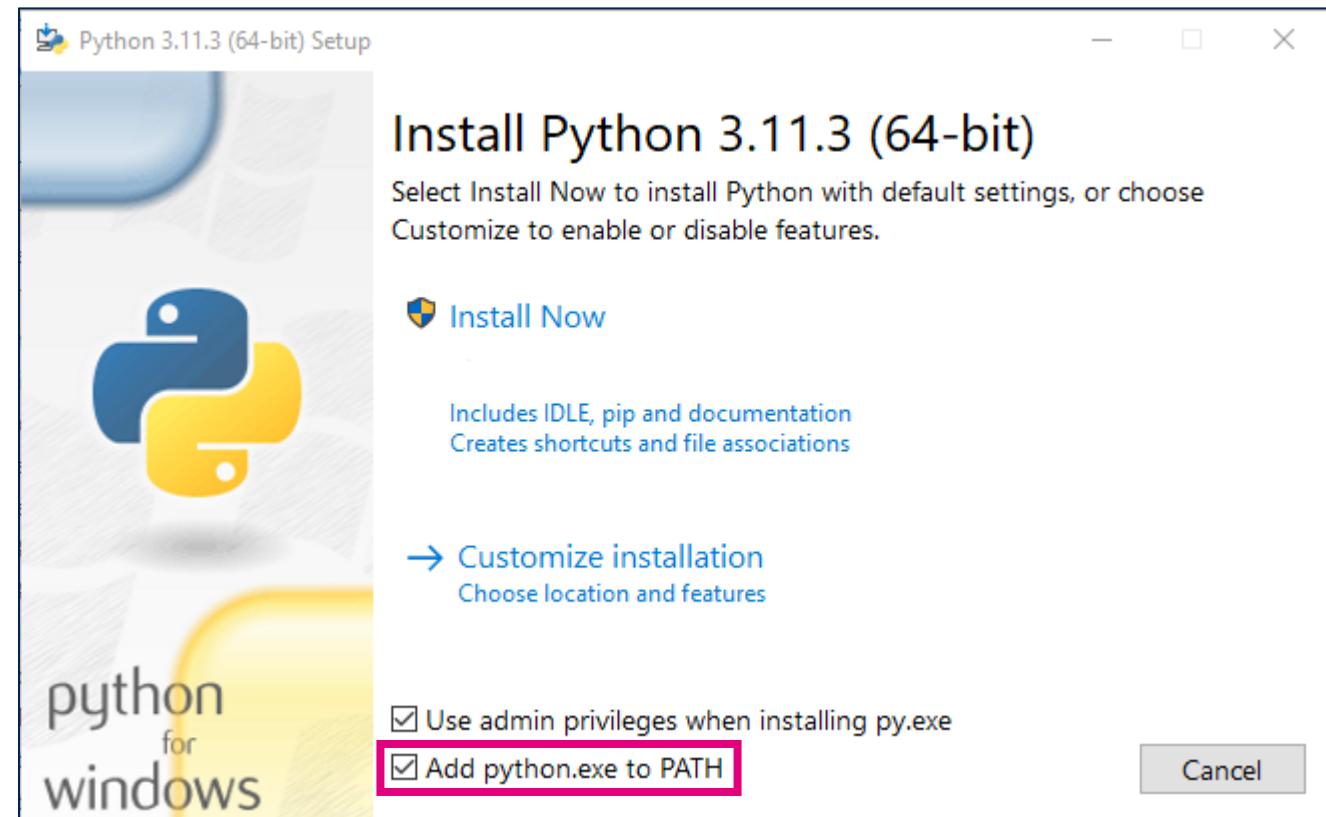


Name	Date modified	Type	Size
examples	2/16/2023 2:59 PM	File folder	
st_dtdl_gui	2/16/2023 2:55 PM	File folder	
st_hsdatalog	2/16/2023 2:55 PM	File folder	
st_pnpl	2/16/2023 2:55 PM	File folder	
HSDPython_SDK_install.bat	2/16/2023 2:19 PM	Windows Batch File	2 KB
HSDPython_SDK_install.sh	2/16/2023 2:19 PM	Shell Script	2 KB
HSDPython_SDK_install_noGUI.bat	2/16/2023 2:19 PM	Windows Batch File	2 KB
HSDPython_SDK_install_noGUI.sh	2/16/2023 2:19 PM	Shell Script	2 KB
HSDPython_SDK_uninstall.bat	2/16/2023 2:19 PM	Windows Batch File	2 KB
HSDPython_SDK_uninstall.sh	2/16/2023 2:19 PM	Shell Script	2 KB

10 items

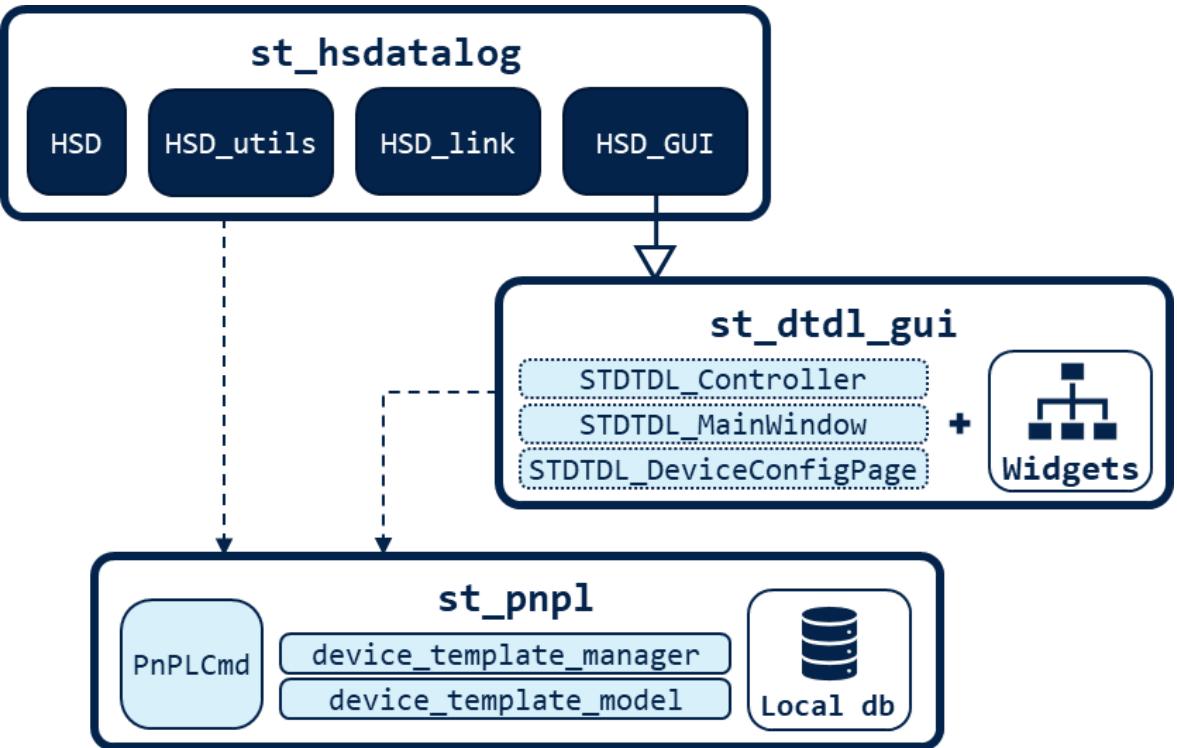
HSDPython_SDK

- Before using HSDPython_SDK, Python3 (>=3.10) must be properly installed on your machine.
- The following steps are valid for a Windows machine. Similar approach can be followed on other OS as well.
 - Download the installer from python.org and launch it
 - Select **Add python.exe to PATH** flag and click **Install Now**. Administrator privileges are needed.
 - Once the setup is complete, you can use Python on your machine



HSDPython_SDK installation

- The three modules are distributed as Python wheels
- Launch *HSDPython_SDK_install.bat* (Windows) or *HSDPython_SDK_install.sh* (Linux)
- The SDK modules and their dependencies will be installed in your Python environment
- For Linux users, further steps are needed.
 - A step-by-step procedure is described in detail in the *readme_linux* file.

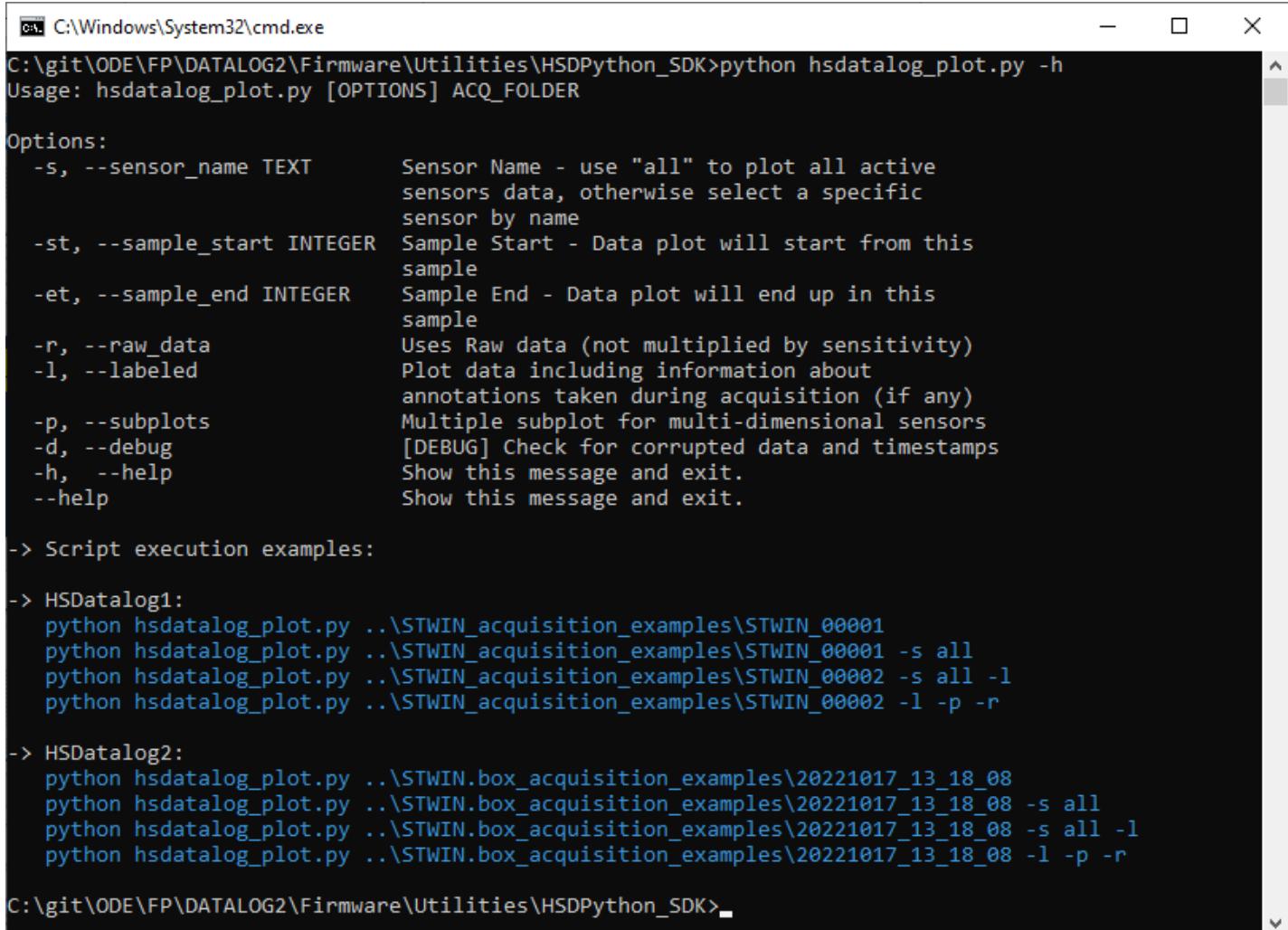


HSDPython_SDK scripts

- The SDK can be used to develop a custom project either by importing the provided modules in a new application or by modifying one of the available scripts
- Here the list of scripts available in the examples folder:
 - *hsdatalog_check_dummy_data.py* can be used to debug the complete application and verify that data are stored or streamed correctly. You must recompile the firmware enabling *HSD_USE_DUMMY_DATA* define (set `#define HSD_USE_DUMMY_DATA 1` into *SensorManager.conf.h*).
 - *hsdatalog_cli.py* is the Python-based CLI
 - *hsdatalog_data_export.py* can convert data into CSV or TSV files.
 - *hsdatalog_data_export_by_tags.py* can be used for tagged acquisition to convert data into different files, one for each tag used.
 - *hsdatalog_dataframes.py* can save data as pandas dataframe for further processing needs.
 - *hsdatalog_plot.py* can plot the desired data.
 - *hsdatalog_GUI.py* provides an example for real time control and plot as described in Section 2.1.1.
 - *hsdatalog_to_nanoedge.py* can prepare data to be imported into NanoEdge AI Studio solution.
 - *hsdatalog_to_unico.py* can prepare data to be imported into Unico-GUI.
 - *hsdatalog_to_wav.py* can convert audio data into a wave file.
 - *ultrasound_fft_app.py* can be used to handle and control the Ultrasound_FFT FW application for STEVAL-STWINBX1.

HSDPython_SDK scripts

- You can execute the scripts in your preferred Python environment
 - i.e.: use the command `python hsdatalog_plot.py`
- Discover the complete list of parameters for each script by executing with the `-h` option
 - i.e.: `python hsdatalog_plot.py -h`



```
C:\Windows\System32\cmd.exe
C:\git\ODE\FP\DATALOG2\Firmware\Utilities\HSDPython_SDK>python hsdatalog_plot.py -h
Usage: hsdatalog_plot.py [OPTIONS] ACQ_FOLDER

Options:
  -s, --sensor_name TEXT      Sensor Name - use "all" to plot all active
                               sensors data, otherwise select a specific
                               sensor by name
  -st, --sample_start INTEGER  Sample Start - Data plot will start from this
                               sample
  -et, --sample_end INTEGER   Sample End - Data plot will end up in this
                               sample
  -r, --raw_data              Uses Raw data (not multiplied by sensitivity)
  -l, --labeled               Plot data including information about
                               annotations taken during acquisition (if any)
  -p, --subplots              Multiple subplot for multi-dimensional sensors
  -d, --debug                 [DEBUG] Check for corrupted data and timestamps
  -h, --help                  Show this message and exit.
  --help                      Show this message and exit.

-> Script execution examples:

-> HSDataLog1:
  python hsdatalog_plot.py ..\STWIN_acquisition_examples\STWIN_00001
  python hsdatalog_plot.py ..\STWIN_acquisition_examples\STWIN_00001 -s all
  python hsdatalog_plot.py ..\STWIN_acquisition_examples\STWIN_00002 -s all -l
  python hsdatalog_plot.py ..\STWIN_acquisition_examples\STWIN_00002 -l -p -r

-> HSDataLog2:
  python hsdatalog_plot.py ..\STWIN.box_acquisition_examples\20221017_13_18_08
  python hsdatalog_plot.py ..\STWIN.box_acquisition_examples\20221017_13_18_08 -s all
  python hsdatalog_plot.py ..\STWIN.box_acquisition_examples\20221017_13_18_08 -s all -l
  python hsdatalog_plot.py ..\STWIN.box_acquisition_examples\20221017_13_18_08 -l -p -r

C:\git\ODE\FP\DATALOG2\Firmware\Utilities\HSDPython_SDK>
```

3 - Documents & Related Resources

Documents & Related Resources

FP-SNS-DATALOG2:

- **DB4865:** STM32Cube function pack for high speed datalogging and ultrasound processing– [databrief](#)
- **UM3106:** Getting started with the STM32Cube function pack for high speed datalogging and ultrasound processing – [user manual](#)

STEVAL-STWINBX1:

- [Gerber files, BOM, Schematic](#)
- **DB4598:** STWIN.box - SensorTile Wireless Industrial Node Development Kit– [databrief](#)
- **UM2965:** Getting started with the STEVAL-STWINBX1 SensorTile wireless industrial node development kit– [user manual](#)

STEVAL-PDETECT1:

- [Gerber files, BOM, Schematic](#)
- **DB5165:** Presence detection add-on for STWIN.box – [databrief](#)
- **UM3320:** Getting started with the STEVAL-PDETECT1 Presence Detection evaluation board – [user manual](#)

Thank you

© STMicroelectronics - All rights reserved.

The STMicroelectronics corporate logo is a registered trademark of the STMicroelectronics group of companies. All other names are the property of their respective owners.

